Differential Privacy for Metric Spaces: Information-Theoretic Models for Privacy and Utility with New Applications to Metric Domains

Natasha Fernandes

▶ **To cite this version:**

Natasha Fernandes. Differential Privacy for Metric Spaces: Information-Theoretic Models for Privacy and Utility with New Applications to Metric Domains. Information Theory [cs.IT]. École Polytechnique Paris; Macquarie University, 2021. English. NNT : . tel-03344320

HAL Id: tel-03344320
https://theses.hal.science/tel-03344320

Submitted on 14 Sep 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2021IPPAX030

Thèse de doctorat

Confidentialité différentielle pour les espaces métriques: modèles théoriques de l'information pour la confidentialité et l'utilité avec de nouvelles applications aux domaines métriques

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à l'École polytechnique et Macquarie University

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (EDIPP)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Sydney, le 28 Mai, 2021, par

NATASHA FERNANDES

Composition du Jury :

Steve Kremer Directeur de recherche, INRIA Nancy	Président
Marco Gaboardi Professeur associé, Boston University	Rapporteur
Joshua Guttman Professeur, Worcester Polytechnique Institute	Rapporteur
Michele Boreale Professeur, Università di Firenze	Rapporteur
Giovanni Cherubin Chargé de recherche, Alan Turing Institute, UK	Examinateur
David Baelde Chargé de recherche, ENS Saclay	Examinateur
Catuscia Palamidessi Directrice de recherche, INRIA & Institut Polytechnique de Paris	Directeur de thèse
Annabelle McIver Professeur, Macquarie University	Co-directeur de thèse

Differential Privacy for Metric Spaces: Information-Theoretic Models for Privacy and Utility with New Applications to Metric Domains

By

NATASHA FERNANDES

B.Sc (Pure Mathematics & Computer Science), University of Sydney

M.Res (Computer Science), Macquarie University

A thesis submitted in fulfilment
of the requirements for the degree of
Doctor of Philosophy

July 2021

Statement of Originality

This thesis is being submitted to Macquarie University and Institut Polytechnique de Paris (École Polytechnique) in accordance with the Cotutelle agreement dated 6 September, 2018.

To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

(Signed) _____ Dated: _____

Abstract

The problem of data privacy – protecting sensitive or personal data from discovery – has been a long-standing research issue. In this regard, differential privacy, introduced in 2006, is considered to be the gold standard. Differential privacy was designed to protect the privacy of individuals in statistical datasets such as census datasets. Its widespread popularity has led to interest in applying differential privacy to new domains for which it was not originally designed, such as text documents. This raises questions regarding the interpretability of differential privacy's guarantees, which are usually expressed in the language of statistical disclosure control. In addition, it escalates the need for answers to core issues currently debated in the differential privacy community: how does the application of differential privacy protect against inference attacks? How can the use of noise-adding mechanisms guarantee the release of useful information? And how can this privacy-utility balance be achieved?

The goal of this thesis is to address these foundational questions. Firstly, we approach the problem of interpretability by exploring a generalisation of differential privacy for metric domains known as metric differential privacy or **d**-privacy. Metric differential privacy abstracts away from the particulars of statistical databases and permits reasoning about privacy on more general domains endowed with a metric. This allows differential privacy's guarantees to be understood in more general terms which can be applied to arbitrary domains of interest, including text documents.

Secondly, we propose to study the key questions surrounding privacy and utility in differential privacy using the Quantitative Information Flow (QIF) framework, an information-theoretic framework previously used to analyse threats to secure systems. In this thesis, we repurpose QIF to analyse the privacy and utility guarantees provided by differentially private systems modelled as probabilistic channels. Using information flow analysis we examine the privacy characteristics of **d**-private mechanisms, finding new ways to compare them with respect to the protection they afford against arbitrary adversarial threats; we examine the utility characteristics of **d**-private mechanisms, discovering a new characterisation for optimal mechanisms and a proof of the universal optimality of the Laplace mechanism; and we re-examine the well-known privacy-utility trade-off for **d**-private mechanisms, finding new models for describing the relationship between privacy and utility via correlations.

The second part of this thesis is dedicated to the demonstration of the practical applicability of **d**-privacy to novel and complex domains. We present three new sample applications of **d**-privacy: to text document privacy, statistical utility and private nearest neighbour search. In each of these applications, we show how the use of **d**-privacy, and an understanding of the

metrics on the domain, permit reasoning about privacy and utility. This opens up new methods of exploring privacy in these domains, as well as providing guidelines for further applications of differential privacy to new domains.

Résumé

Le problème de la confidentialité des données – la protection des données sensibles ou personnelles – est un problème de recherche de longue date. La confidentialité différentielle, introduite en 2006, est considéré comme *la* référence en la matière. Elle a été conçue pour protéger la confidentialité des données privées dans des jeux de données statistiques tels que les ceux de recensement. Sa grande popularité a conduit à un interêt à l'appliquer dans de nouveaux domaines pour lesquels elle n'était pas originellement conçue, tels que des documents de texte. Cela soulève des questions sur l'interprétabilité des garanties apportées par la confidentialité différentielle, qui sont en général exprimées dans le language de contrôle statistique de la divulgation. De plus, cela accentue le besoin de répondre à des problèmes centraux au débat actuel au sein de la communauté de la confidentialité différentielle: comment l'application de la confidentialité différentielle protège-t-elle contre les attaques d'inférence? Comment l'utilisation de mécanismes d'ajout de bruit peut-elle garantir la publication d'information utile? Et comment l'équilibre "privacy-utility" peut-il être obtenu?

Le but de cette thèse est de répondre à ces questions de fond sur la confidentialité différentielle. Tout d'abord, nous abordons le problème de l'interprétabilité en explorant une généralisation de la confidentialité différentielle pour des espaces métriques, connue sous le nom de confidentialité différentielle métrique, ou "**d**-privacy". La confidentialité différentielle métrique fait abstraction des détails des bases de données statistiques et permet de raisonner sur la confidentialité de domaines plus généralisés, dotés d'une distance. Cela permet une compréhension plus générale des garanties de la confidentialité différentielle, qui peut être appliquée à des domaines d'interêt arbitraire, y compris les documents de texte.

Deuxièmement, nous proposons l'étude des questions clés autour de la confidentialité et la "utility" pour la confidentialité différentielle, en utilisant le système de Flot d'Information Quantitative (Quantitative Information Flow, QIF), un système de théorie de l'information actuellement utilisé pour l'analyse de menaces sur des systèmes de sécurité. Dans cette thèse, nous réutilisons QIF pour analyser les garanties de confidentialité et de "utility" fournies par des systems de confidentialité différentielle modélisés sous forme de canaux probabilistiques. En utilisant l'analyse de flot d'information, nous examinons les caractéristiques de confidentialité des mécanismes **d**-privacy, trouvant de nouveaux moyens de les comparer sur le plan de la protection qu'ils offrent contre des menaces arbitraires; nous examinons les caractéristiques de "utility" des mécanismes **d**-privacy, découvrant une nouvelle caractérisation pour les mécanismes optimaux et une preuve de l'optimalité universelle du mécanisme de Laplace; et nous examinons de nouveau le fameux compromis "privacy-utility" pour les mécanismes **d**-privacy, trouvant de

nouveaux modèles pour décrire la relation entre confidentialité et “utility” via des correlations.

La deuxième partie de cette thèse est consacrée à la démonstration de l’applicabilité pratique de la **d**-privacy dans des domaines nouveaux et complexes. Nous présentons trois nouveaux domaines d’application de la **d**-privacy: la confidentialité des document de texte, l’utilité statistique et la recherche confidentielle de plus proche voisin. Dans chacune de ces applications, nous montrons comment l’utilisation de la **d**-privacy, et une compréhension de la métrique sur le domaine, permet de raisonner sur la confidentialité et l’utilité. Cela ouvre à de nouvelles méthodes pour explorer la confidentialité dans ces domaines, et pour guider l’application de la confidentialité différentielle à de nouveaux domaines.

Acknowledgements

I would like to express deepest gratitude to the many people who have made my PhD journey so incredibly rewarding.

To my husband, Anthony: for your support and encouragement over the past 3 years, following me to the ends of the earth (literally!) and carrying so many burdens with unconditional, unquestioning love. To my children Ben, Jack and Tom: thank you for your patience, resilience and understanding; and for leaving your friends back home to endure a year in French school.

To my supervisors Annabelle McIver and Catuscia Palamidessi: you have both inspired me with your amazing qualities - fierce intelligence, deep wisdom, and extraordinary care of and encouragement for your students. I feel incredibly blessed to have you both accompanying and guiding me on this journey. In particular: to Catuscia, for welcoming me and my family to France with your joyful presence, your enthusiasm and your great cooking. And especially for coping with my incomprehensible Australian accent. And to Annabelle, for your time and your guidance, and for your calm, gentle and supportive presence, especially in times of stress.

A special thank you to my uncle, Leslie Fernandes: without your support, my cotutelle year in France would not have been possible; my family and I are so very grateful for your generosity.

To my collaborators and co-conspirators - Mário Alvim, Kostas Chatzikokolakis, Ming Ding, Mark Dras, Yusuke Kawamoto, Carroll Morgan, Takao Murakami and Gabriel Nunes: you have all contributed in different ways to my growth as a researcher and as a person. Thank you.

Finally, to all of the researchers and students with whom I have shared good beer, good wine, good food and great discussions: thank you for the good times, good conversation, good humour and great memories.

Contents

Contents viii

List of Figures xiv

List of Tables xviii

1 Overview 1

1.1 Background to differential privacy . . . 1

1.1.1 Differential privacy and its applications . . . 3

1.1.2 Metric differential privacy . . . 4

1.2 Key themes in the practice of differential privacy . . . 5

1.2.1 Privacy breaches as inference attacks . . . 5

1.2.2 How to choose epsilon . . . 5

1.2.3 Managing the privacy-utility balance . . . 6

1.2.4 Reasoning about utility . . . 6

1.3 Goals of this thesis . . . 7

1.3.1 Contributions . . . 7

1.3.2 Synopsis . . . 8

1.3.3 Publications . . . 11

I Foundations 12

2 Quantitative Information Flow 13

2.1 Modelling Systems as Channels . . . 14

2.2 Modelling Adversaries . . . 15

2.2.1 Vulnerability and Gain Functions . . . 15

2.2.2 Posterior Vulnerability . . . 20

2.2.3 Leakage . . . 21

2.3 Comparing Channels: Refinement . . . 23

2.4 Hyper-Distributions . . . 24

2.4.1 The Geometry of Hyper-Distributions . . . 26

2.4.2 Refinement of Hypers . . . 27

2.5 Chapter Notes . . . 30

3 Metric Differential Privacy **31**
3.1 Definition and Properties . . . 31
3.1.1 Technical Preliminaries . . . 32
3.1.2 Definition of **d**-Privacy . . . 32
3.1.3 Properties of Metric Differential Privacy . . . 34
3.2 Geo-Indistinguishability - A Canonical Example . . . 37
3.2.1 Utility Goals . . . 39
3.3 Reasoning with Metrics . . . 39
3.3.1 Oblivious Mechanisms . . . 40
3.3.2 Local Differential Privacy . . . 42
3.3.3 Reasoning about Privacy and Utility . . . 43
3.4 Constructing **d**-Private Mechanisms . . . 44
3.4.1 Other Metrics . . . 46
3.5 Discussion and Concluding Remarks . . . 46
3.6 Chapter Notes . . . 46

II Analysis **48**

4 Comparing Privacy Mechanisms **49**
4.1 Motivating Examples . . . 50
4.1.1 Technical Preliminaries . . . 52
4.2 Average-Case Refinement . . . 53
4.3 Max-Case Refinement . . . 53
4.4 Privacy-Based Refinement . . . 57
4.4.1 Comparing mechanisms by their "smallest ε" (for fixed **d**) . . . 57
4.4.2 Privacy-based leakage and refinement orders . . . 58
4.4.3 Application to oblivious mechanisms . . . 60
4.4.4 Privacy as max-case capacity . . . 62
4.4.5 Revisiting the Data Processing Inequality for **d**-Privacy . . . 64
4.5 Verifying the Refinement Orders . . . 65
4.5.1 Average-case refinement . . . 65
4.5.2 Max-case refinement . . . 66
4.5.3 Privacy-based refinement . . . 66
4.6 Lattice properties . . . 67
4.6.1 Average-case refinement . . . 67
4.6.2 Max-case refinement . . . 67
4.6.3 Privacy-based refinement . . . 68
4.6.4 Constructing **d**-private mechanisms for any metric **d** . . . 69
4.7 Comparing **d**-Privacy Mechanisms . . . 69
4.7.1 Technical Preliminaries . . . 70
4.7.2 Refinement order within families of mechanisms . . . 72

4.7.3 Refinement order between families of mechanisms . . . 74
4.7.4 Asymptotic behaviour . . . 76
4.8 Conclusion . . . 79
4.9 Chapter Notes . . . 79

5 Inference Attacks **80**
5.1 Motivation . . . 81
5.2 Informal example: does it matter how to randomise? . . . 82
5.3 Modelling Inferences . . . 84
5.3.1 Quantitative Information Flow . . . 84
5.3.2 Reasoning about Inferences . . . 86
5.4 The Privacy-Utility Trade Off . . . 89
5.5 Application to Differential Privacy . . . 90
5.5.1 Utility-Focussed Privacy . . . 91
5.5.2 Secrecy-Focussed Privacy . . . 92
5.6 Experimental Results . . . 93
5.7 Concluding Remarks . . . 96
5.8 Chapter Notes . . . 97

6 Optimality I: Discrete Mechanisms **98**
6.1 Introduction . . . 99
6.1.1 Motivation and Contribution . . . 99
6.2 Modelling Utility . . . 101
6.2.1 Universal optimality . . . 103
6.2.2 The privacy context . . . 104
6.2.3 Relationship to “counting” and “sum” queries . . . 105
6.2.4 Our results on optimality . . . 106
6.3 The Geometry of $\mathbf{d}$-Privacy . . . 107
6.3.1 The relationship between channels and hypers . . . 107
6.3.2 The space of $\mathbf{d}$-private hypers . . . 108
6.3.3 Kernel and Vertex Mechanisms . . . 111
6.4 Characterising Optimal Mechanisms . . . 113
6.4.1 Existence of universally optimal mechanisms . . . 113
6.4.2 Characterising universal $\mathcal{L}$-optimality . . . 116
6.5 Reasoning about Optimality . . . 117
6.5.1 Loss Function Algebra . . . 117
6.5.2 Classes of Loss Functions . . . 118
6.5.3 Properties of Loss Function Classes . . . 120
6.5.4 Identifying Optimal Mechanisms . . . 121
6.6 Application to Oblivious Mechanisms . . . 125
6.7 Optimality for $\mathbf{d}_2$-Private Mechanisms on $\{0 \ldots N\}$ . . . 125
6.7.1 Kernel mechanisms . . . 125

6.7.2 The geometric mechanism . . . 126
6.7.3 Other optimal mechanisms . . . 128
6.8 Optimality for $\mathbf{d}_{\mathrm{D}}$-Private Mechanisms . . . 128
6.8.1 Kernel mechanisms . . . 128
6.8.2 The randomised response mechanism . . . 133
6.8.3 Impossibility of universal optimality . . . 134
6.8.4 Reframing the impossibility result for monotonic loss functions . . . 135
6.9 Optimality for Other Metric Spaces . . . 136
6.10 Conclusion . . . 138
6.11 Chapter Notes . . . 138

7 Optimality II: Continuous Mechanisms 140
7.1 Introduction . . . 140
7.2 Preliminaries: loss functions; hypers; refinement . . . 141
7.2.1 The relevance of hyper-distributions . . . 141
7.2.2 Refinement of hypers and mechanisms . . . 142
7.3 Discrete and continuous optimality . . . 143
7.3.1 The optimality result for the geometric mechanism . . . 143
7.3.2 The geometric mechanism is never $\varepsilon\cdot\mathbf{d}$-private on dense continuous inputs, eg. when $\mathbf{d}$ on $\mathcal{X}$ is Euclidean . . . 144
7.3.3 Our result — continuous optimality . . . 144
7.3.4 An outline of the proof . . . 145
7.4 Measures on continuous $\mathcal{X}$ and $\mathcal{Y}$ . . . 147
7.4.1 Measures via probability density functions . . . 147
7.4.2 Continuous mechanisms over continuous priors . . . 147
7.4.3 The truncated Laplace mechanism . . . 147
7.5 Approximating Continuity for $\mathcal{X}$ . . . 149
7.5.1 Connecting continuous and discrete . . . 149
7.5.2 Approximations of continuous priors . . . 149
7.5.3 N-step mechanisms and loss functions . . . 149
7.5.4 Approximating continuous $\varepsilon\cdot\mathbf{d}$-private mechanisms . . . 150
7.6 The Laplace and Geometric mechanisms . . . 151
7.6.1 The stability of optimality for geometric mechanisms . . . 152
7.6.2 Bounding the loss for continuous mechanisms . . . 154
7.6.3 Refinement between N-Geometric and T,N-Laplace mechanisms . . . 154
7.6.4 The Laplace approximates the Geometric . . . 156
7.6.5 Approximating monotonic functions . . . 158
7.7 Universal optimality for the Laplace Mechanism . . . 159
7.8 Conclusion and Future Work . . . 159
7.9 Chapter Notes . . . 160

III Applications **162**

8 Statistical Utility for Local Differential Privacy **163**
8.1 Scenario: Estimating a Distribution . . . 164
8.2 Comparing Mechanisms for Privacy . . . 164
8.3 Comparing Mechanisms for Utility . . . 166
8.3.1 Reconstructing the Original Distribution . . . 166
8.3.2 Maximum Likelihood Estimation with IBU . . . 167
8.3.3 Measuring Utility . . . 168
8.4 Experimental Results . . . 169
8.5 Conclusion . . . 172
8.6 Chapter Notes . . . 172

9 Text Document Privacy **173**
9.1 Introduction . . . 173
9.2 Documents, Topic Classification and Earth Moving . . . 175
9.2.1 Word embeddings . . . 175
9.3 Privacy, Utility and the Earth Mover's Metric . . . 178
9.3.1 Utility and Earth Moving . . . 179
9.3.2 Implementing Earth Mover's Privacy . . . 179
9.3.3 Application to Text Documents . . . 181
9.3.4 Properties of Earth Mover's Privacy . . . 181
9.4 Earth Mover's Privacy for bags of vectors in $\mathbb{R}^n$ . . . 182
9.4.1 Earth Mover's Privacy in $\mathbb{BR}^n$ . . . 186
9.4.2 Utility Bounds . . . 186
9.5 Text Document Privacy . . . 187
9.5.1 Privacy as Indistinguishability . . . 188
9.5.2 Privacy as Protection from Inferences . . . 189
9.6 Experimental Results . . . 192
9.7 Conclusions . . . 195
9.8 Chapter Notes . . . 195

10 Locality Sensitive Hashing with Differential Privacy **198**
10.1 Introduction . . . 199
10.1.1 Friend-Matching: Problem Setup . . . 199
10.1.2 Our contributions . . . 200
10.2 Locality Sensitive Hashing (LSH) . . . 200
10.2.1 Random-Projection-Based Hashing . . . 201
10.2.2 Privacy with LSH . . . 202
10.2.3 Is LSH private? . . . 203
10.3 LSH-based Privacy Mechanisms . . . 205
10.3.1 Privacy Measures . . . 205

10.3.2 Construction of LSHRR . . . 207
10.3.3 Construction of LapLSH . . . 208
10.4 Analyses of the Privacy Mechanisms . . . 208
10.4.1 LSHRR: Privacy Guarantees for Hash Values . . . 209
10.4.2 LSHRR: Privacy Guarantees for Inputs . . . 209
10.4.3 LapLSH: Privacy Guarantees . . . 211
10.5 Experimental Evaluation . . . 212
10.5.1 Experimental Setup . . . 212
10.5.2 Results . . . 214
10.5.3 Discussion . . . 218
10.6 Conclusion . . . 219
10.7 Chapter Notes . . . 219

11 Conclusions and Future Work **221**

References **225**

Appendices **242**

A Proofs Omitted from Chapter 6 **243**
A.1 Results supporting §6.3.3 . . . 243
A.2 Results supporting §6.4 . . . 245
A.3 Proofs omitted from §6.5.1 . . . 246
A.4 Results supporting §6.7 . . . 247
A.5 Proofs omitted from §6.8 . . . 248

B Proofs Omitted from Chapter 7 **250**
B.1 Measures and continuous hypers . . . 250
B.2 Results supporting §7.6 . . . 251
B.3 Supporting material for §7.6.4 . . . 254
B.4 Supporting material for §7.6.5 . . . 258
B.5 Step functions are not legal for continuous mechanisms . . . 258
B.6 Example to show that for step loss functions, the Laplace is not optimal . . . 258

C Proofs Omitted from Chapter 10 **260**
C.1 Material supporting §10.4 . . . 260

List of Figures

2.1 Geometric representations of a hyper 26

3.1 Geo-location privacy under the Euclidean metric. Points which are nearby are more indistinguishable whereas points which are further away are more distinguishable to an adversary. 39

3.2 Oblivious $\mathbf{d}_\mathcal{X}$-private mechanisms K are composed of a deterministic function f and a $\mathbf{d}_\mathcal{Y}$-private mechanism H. We reason about privacy using the composition $K = H \circ f$ but we reason about utility using H alone. 41

3.3 Local $\mathbf{d}_\mathcal{X}$-private mechanisms H are applied directly to individuals' data points before being sent to the untrusted data curator. In this case we reason about privacy and utility in the same way, either using the mechanism H directly (if the data curator releases each noisy data point as is) or using the composition of the mechanisms H followed by a post-processing step f if the curator releases the result of a query f on the noisy data. 43

3.4 The α-truncated geometric mechanism (left) and the α-randomised response mechanism (right) for $\alpha = 1/2$ and $\mathcal{X} = \{0, 1, 2\}$. The truncated geometric mechanism is a square matrix satisfying $TG_{x,y} = \alpha TG_{x',y}$ on $x = x' + 1$ or $x' = x + 1$. The randomised response mechanism is a symmetric (square) matrix with maximum values on the diagonal and all other values equal. 45

4.1 Comparison between the truncated geometric (red) and the randomised response (blue) mechanisms. The graphs show the distribution over outputs produced by each mechanism when the input is $x = 50$ for outputs in the range $\{0, 1, \ldots, 100\}$. 51

4.2 Failure of refinement for the hypers $[\upsilon \triangleright A]$ (blue) and $[\upsilon \triangleright B]$ (orange) from Example 4.5. The orange and blue points are posteriors lying on the line $x_1 + x_2 = 1$. The orange points are not both contained within the convex hull of the blue points and vice versa, and therefore $A \not\sqsubseteq^{\max} B$ and $B \not\sqsubseteq^{\max} A$. 61

4.3 Comparison of leakage and refinement orders. All implications are strict. 62

4.4 Plot of the d-vulnerability function $V_d(\pi)$ for Discrete metric d on distributions over 3 secrets showing top view (left) and side view (right). The function is *quasi-convex* and thus can be used to describe a max-case vulnerability. 63

4.5 Refinement of mechanisms under $\sqsubseteq$ for 5×5 channels. The x-axis (ε_1) represents the mechanism on the LHS of the relation. The y-axis (ε_2) represents the refining mechanism. We fix ε_1 and ask for what value of ε_2 do we get a refinement? The top left graph represents refinement of the truncated geometric mechanism (that is, $TG \sqsubseteq$), the top right graph is refinement of randomised response ($R \sqsubseteq$), and the bottom graph is refinement of the exponential mechanism ($E \sqsubseteq$). 78

5.1 Randomised response with N participants 83
5.2 Oblivious response mechanism to announce the total for N participants 84

6.1 The universally optimal mechanism for 2 inputs for $\varepsilon = \ln 2$ and $\mathbf{d}(x_1, x_2) = 1$. The blue lines show the constraints defining the convex region (grey line segment). The region has exactly one mechanism defined by the 2 vertices (orange points). 115
6.2 The space of $\mathbf{d}_2$-private hypers on 3 inputs $\{0, 1, 2\}$. The geometric mechanism for $\varepsilon = \ln 2$ consists of the 3 orange vertices (B, C, D); these are linearly independent and thus form a kernel mechanism. The only other kernel mechanism consists of the posteriors A and C, since their convex hull contains the uniform distribution (υ). 127
6.3 The space of $\mathbf{d}_{\mathrm{D}}$-private hypers over 3 inputs. The randomised response mechanism for $\varepsilon = \ln 2$ corresponds to the 3 orange vertices (A, B, C); these are linearly independent and thus form a kernel hyper. There are 4 other kernel hypers in this space: the hyper formed from posteriors (A', B', C') and the 3 hypers (A, A'), (B, B') and (C, C'). 129
6.4 The Hamming space of bitstrings of length 3. The distance between vertices is given by the Hamming metric $\mathbf{d}_{\mathrm{H}}$ between bitstrings. The orange lines depict a path on which the $\mathbf{d}_{\mathrm{H}}$ metric is linear. Notice there are many other paths for which this holds, including all paths of length 3 from (000) to (111). 136
6.5 A kernel hyper (Δ_K) and corresponding kernel mechanism (K) on bitstrings of length 3 wrt the Hamming distance $\mathbf{d}_{\mathrm{H}}$ with $\varepsilon = \ln 2$. 137

7.1 Illustration of why lifting the geometric mechanism by changing its input domain does not result in a valid $\varepsilon{\cdot}\mathbf{d}$-private mechanism. 145
7.2 Illustration of geometric mechanisms from the example in §7.6.1 in the space of hypers. The supports of hyper $[\upsilon \triangleright G_2^{\varepsilon}]$ are depicted in orange and those of $[\upsilon \triangleright M]$ are the blue and orange points (combined). Observe that M is not a kernel mechanism (recall Def. 6.3.2 from Chapter 6) thus it cannot be (strictly) universally optimal. 153
7.3 N-Geometric and T,N-Laplace mechanisms. 155

8.1 The distributions generated by the randomised response (blue) and truncated geometric (red) mechanisms applied to the input $x = 50$. The value of the privacy parameter ε is $\ln 2$ and $\ln 2/10$, respectively. This means that within a radius of 10 of the true value (ie. from values 40 to 60, indicated in green) the privacy guarantee for both mechanisms is $\varepsilon = \ln 2$, but outside this radius, the truncated geometric mechanism's privacy guarantee degrades with distance while the randomised response mechanism maintains the same guarantee $\varepsilon = \ln 2$. 166
8.2 IBU reconstruction of true (orange) distributions from noisy (green) distributions based on 100k samples drawn from binomial (top) and "4-point" (bottom) distributions. The blue graphs indicate the reconstructed (estimated) distribution computed using IBU. By observation, it appears that the reconstructed distribution (blue) is closer to the true distribution (orange) for the geometric mechanism when the true distribution is binomial, but that the randomised response mechanism performs better when the true distribution is a point distribution. However, using the Kantorovich metric as a measure of utility, we find that the geometric mechanism performs better on both input distributions (see Figure 8.3). 170
8.3 Kantorovich distances between true and estimated distributions at IBU convergence for the geometric and randomised response (RR) mechanisms. Distances were computed over 20 experiments for each of the 4 sample sizes indicated. The graph shows that the geometric mechanism performs much better than the randomised response mechanism on all datasets, as measured by the average Kantorovich distance between the true distribution and the estimated distribution computed using IBU. 171
8.4 Log likelihood function against number of iterations for the geometric and randomised reponse mechanisms. This graph shows how fast each output distribution converges to the MLE for one particular (representative) run of the IBU. We observe that the geometric mechanism converges quickly whereas convergence for the randomised response is almost flat. 171

9.1 Earth Mover's metric between sample documents. 178
9.2 Laplace density function Lap^2_ε in $\mathbb{R}^2$ 183

10.1 The application of random-projection based hashing with noise to user vectors in a movie ratings dataset to produce noisy bitstrings. The LSH function constructs hyperplanes in the space which divide it into equivalence classes. There are 2 places in which we could add noise – either before or after LSH. 203
10.2 Distributions of angular distances $\mathbf{d}_\theta$ to nearest neighbour for $k = 1$ for each user, plotted for vectors with dimensions 100, 500 and 1000. The distance 0.5 represents orthogonal vectors; ie., having no items in common. 214

10.3 Utility loss vs total privacy budget ξ for LSHRR vs LSH for 1000-dimensional vectors. Note that $\mathbf{d}_\theta$ is indicated in brackets in the legend. LSHRR performance does not vary significantly with bitstring length, and is closer to LSH performance for smaller bit lengths. 216
10.4 Utility loss vs total privacy budget ξ for LSHRR vs LapLSH for $k = 10$ on 100- and 1000-dimensional vectors. Note that $\mathbf{d}_\theta$ is indicated in brackets in the legend. LSHRR is much better than LapLSH for smaller $\mathbf{d}_\theta$ and larger n. 217
10.5 Utility loss vs total privacy budget ξ for LSHRR vs LapLSH for $k = 10$ on 100- and 1000-dimensional vectors. Note that $\mathbf{d}_\theta$ is indicated in brackets in the legend. LSHRR is much better than LapLSH for smaller $\mathbf{d}_\theta$ and larger n. 218

List of Tables

4.1 Programs that take as input a secret H and leak information about H via the output L. 51
4.2 The refinements respected by families of mechanisms for decreasing ε. 76
4.3 Comparing different families of mechanisms with respect to the different refinements under the same ε. 77
4.4 Comparing different families of mechanisms with differing ε. 77

5.1 Results of privacy loss and utility on Scenarios A (with highly correlated counting query and secret) and B (with practically independent counting query and secret) for the COMPAS dataset A value close to 1 for privacy means the data release does not pose a risk to an individual; a value close to 1 for utility means that the data release is accurate. 96

6.1 Kernel mechanisms in the space of $\mathbf{d}_2$-private mechanisms. 125
6.2 Enumerating the kernel mechanisms in the space of $\mathbf{d}_{\mathrm{D}}$-private mechanisms. . . . 129

9.1 Number of correct predictions of author/topic in the 20-author set (left) and 50-author set (right), using 1-NN for same-representation author identification (SRauth), 5-NN for same-representation topic classification (SRtopic), the Koppel algorithm for different-representation author identification (DRauth) and fastText for different-representation topic classification (DRtopic). 194

10.1 Values for average utility loss for LSH (compared with True Nearest neighbours). 215
10.2 Values of α given an angular distance $\mathbf{d}_\theta$ between inputs and a length κ of bitstring. These are used to compute ξ_α in the LSHRR guarantee of Prop. 10.8. . . . 216

1 Overview

Data privacy is about the right use of our personal information – how it is stored, how it is handled – and how it can be leaked. It is the latter that distinguishes privacy from security; while passwords can be changed or credit cards re-issued, privacy, once lost, cannot be so easily recovered. Thus, the need for sound techniques for preserving privacy is even more critical.

In this thesis we address some foundational issues in data privacy. Our focus is on a particular flavour of data privacy – differential privacy – which is used to enact privacy in digital systems such as census databases. Differential privacy has many desirable properties which have made it a popular choice for data privacy. However, its definition is tied to statistical databases, as are many of the narratives and subtle understandings which its early proponents developed to advocate for its use. Metric differential privacy[1] – a generalisation of differential privacy – abstracts away from the particulars of statistical databases and permits reasoning about privacy on domains endowed with a metric. Our goal in this thesis is to explore and extend metric differential privacy by providing tools for reasoning about privacy and utility in complex metric domains, and by demonstrating its application in new domains of interest.

1.1 Background to differential privacy

The right to privacy is recognised in many countries around the world, and the basic building blocks for protecting these rights are enacted in law. Before the digital era, the large-scale collection and dissemination of personal data was confined to census bureaus and the release of population statistics. The idea of these statistics being used to enact privacy breaches was

[1] Metric differential privacy is also known as **d**-privacy or generalised differential privacy.

not unthought-of;[2] indeed they were characterised and documented by the statistician Tore Dalenius [1]. However, these sorts of breaches were thought to be infeasible due to the amount of information an attacker would need in order to correlate any released statistics with particular individuals. Up until the early 2000's, anonymisation of data was considered by many to be sufficient for the protection of individuals' privacy, and this understanding is reflected in the privacy laws that still stand today in many countries including Australia and the United States.[3]

Early in the 2000's some now infamous privacy breaches brought to light the weaknesses of anonymisation techniques in the face of large-scale access to publicly available datasets. In 2002, Latanya Sweeney identified the medical data of the Massachusetts governor in de-identified hospital records by linking them with information freely available in a voter database [2]. In 2006, Netflix released an anonymised dataset of movie ratings in a competition designed to improve their recommendation engine. This dataset was famously attacked by researchers Shmatikov and Narayanan [3] who exploited publicly available information to deanonymise individuals in the dataset – one of whom sued Netflix and won.[4] More recently, the US Census Bureau re-examined census data from 1940 and was able to reconstruct the census details of individuals using now-available information and data processing techniques [4]. These attacks were made possible by the scale of publicly available data on the internet, and this opened up the need for new privacy protection techniques which did not rely on weak assumptions about an attacker's capabilities or background knowledge.

In 2006, differential privacy emerged to solve the problem of privacy for individuals in structured datasets (such as census datasets) for whom data releases could pose privacy risks. Dwork et al. [5] recognised the inevitability of privacy leaks in any useful data release, and instead proposed a privacy definition based on 'plausible deniability' for individuals, parametrised by an 'epsilon' value which determines the extent to which plausible deniability is offered. Differential privacy is usually formalised using equations of the form

$$Pr(K(x) \in Y) \;\leq\; e^{\varepsilon} \times Pr(K(x') \in Y) \qquad \text{for all } Y \subseteq \mathcal{Y} \text{ and } x \sim x' \,, \tag{1.1}$$

where K is a mechanism which delivers a noisy output, x and x' are input databases and $\sim$ is an 'adjacency' relation between datasets that distinguishes datasets differing in a single individual. The privacy provided by (1.1) is that any output in Y which is released could just as plausibly have been produced by dataset x (containing some individual) as from dataset x' (without said individual). Thus, it is argued, an individual's presence in a dataset is protected. This sort of guarantee provides some uncertainty to an attacker – not enough to prevent all inferences, but enough to encourage an individual to relinquish their data with the promise that any attacks against their sensitive data could be claimed to be the result of statistical noise.

Warner's 'randomised response protocol' [6] is perhaps the earliest example of a privacy-preserving algorithm in this spirit. In this protocol, survey participants first flip a coin – 'heads'

[2] The US Census Bureau has had formal privacy requirements for statistical data releases in place since the 1920's: https://www.census.gov/history/www/reference/privacy_confidentiality/privacy_and_confidentiality_2.html.

[3] See Australian Privacy Act: https://www.oaic.gov.au/privacy/guidance-and-advice/de-identification-and-the-privacy-act/ or the US HIPAA: https://www.hhs.gov/hipaa/for-professionals/privacy/laws-regulations/index.html.

[4] https://www.wired.com/2009/12/netflix-privacy-lawsuit/

they answer truthfully (yes or no), 'tails' they flip again and answer yes (heads) or no (tails). The protocol guarantees that each individual's response is protected, and yet, with enough participants, the survey-taker can (quite reliably) calculate the approximate proportion of true 'yes' and 'no' responses. Warner's algorithm predates differential privacy by some decades, but captures the insight behind the differential privacy promise. Perhaps not coincidentally, the randomised response algorithm was one of the first shown to satisfy differential privacy [7].

1.1.1 Differential privacy and its applications

Differential privacy is now arguably the gold standard for data privacy, primarily due to its mathematical properties (it behaves well under composition), its robustness to background knowledge (its guarantee holds for all priors) and its simplicity of implementation (statistical noise can be added independently of the value of a data element) [7]. These properties set it apart from anonymisation-based definitions, which are susceptible to adversarial attacks when combined with other anonymised datasets [8]. The Netflix data breach describes one such attack; in this case, researchers made use of publicly available datasets which were *correlated* with the anonymised Netflix data. These correlations allowed researchers – with high confidence – to identify individuals and their sensitive data in the Netflix dataset, even though the dataset itself contained no obviously identifying information.[5] Differential privacy claims immunity to such attacks [7].

An important but understated property of differential privacy is also its ability to promise the release of *useful* information while maintaining some notion of privacy, a property that its early proponents explored [7]. This property sets differential privacy apart from privacy definitions designed to limit leakage. Dwork et al. [7] argued that leakage is an essential property of any *useful* data release, and therefore defining privacy as "disclosure limitation" – as desired by Dalenius[6] – unnecessarily limits *all* leakages, even ones which may not constitute a privacy breach at all.[7] As an example, differential privacy would say that a dataset which teaches that smoking causes cancer is not breaching the privacy of individuals in it, if it could later be inferred that one of them has cancer, since this information would have been learned regardless of whether the individual had participated in the study or not.

Instead, differential privacy proposes a notion of "indistinguishability", in which similar inputs produce similar noisy outputs, creating some uncertainty in an attacker's ability to guess the secret with confidence, while still permitting the release of statistically useful information. In this way, differential privacy finds an in-between, one which has been accepted by the community as the 'best available' compromise.

Because of its dominance and widespread acceptance in the literature, researchers in other

[5] The Netflix dataset release consisted simply of a list of movie ratings for each individual. That it was de-identified proved how informative – and uniquely identifying – certain individual preferences are.

[6] Dalenius has been often misquoted as desiring absolute privacy, ie. with no leakage. While this is too strong, and a disservice to Dalenius, he did desire privacy by constraining leakage [1] – and in this way, differential privacy marks a departure from this type of thinking.

[7] Recall that "privacy" as a concept existed well before differential privacy, and therefore it is reasonable to consider common notions of "privacy breach" and what may or may not constitute one.

areas of privacy have been tempted to adopt differential privacy in their own domains. Differential privacy has been applied to domains as diverse as images [9, 10], text documents [11, 12] and shipping trajectories [13]. However, this diversity comes at a cost – differential privacy was not designed with these sorts of domains in mind, and its definition is couched in terminology specific to statistical datasets. Re-interpreting the 'plausible deniability' style guarantee for datasets in which the secrets are not individuals in rows but some more complex data is not trivial: in an image, how does one protect the individual whose image is displayed? In a document, how would one protect the writing style of the author to prevent re-identification? In these examples, what exactly is plausibly denied by the application of differential privacy? How is privacy achieved and how might it be breached? And how might one ensure that some useful information is retained in the data release?

1.1.2 Metric differential privacy

In 2012, researchers working in geo-location privacy introduced the notion of 'metric differential privacy' in order to enable differential privacy-style reasoning in their domain [14, 15]. Metric differential privacy provides a generalisation (and re-interpretation) of differential privacy for arbitrary domains endowed with a metric. The researchers' insight was to notice that some of the main properties of differential privacy rely on the metric properties of the Hamming distance between datasets; and that this metric can be safely changed without breaking the strong properties that differential privacy offers (robustness to priors, compositionality etc). Plausible deniability can be re-interpreted as 'privacy within a radius', permitting differential privacy to be applied to new (metric) domains. Their example application to geo-location privacy – the problem of hiding users' precise locations while preserving their approximate locations – provided a blueprint for how to re-interpret "adjacency of secrets" and the meaning of epsilon for a novel domain, as well as a new method for reasoning about utility when noise is added directly to the secret values themselves, rather than to the output of a "query" as is done in the statistical database setting.

Metrics are used in a surprising array of privacy domains, from spatio-temporal trajectories to social network graphs, recommender systems, text documents and images. However, despite its potential, metric differential privacy has found few applications beyond the geo-location privacy scenario for which it was designed. Privacy practitioners in new domains often resort to the reasoning designed for individuals in statistical databases, even when this does not correspond directly the privacy issue at hand, nor relate to the utility desired from the data release.[8] This suggests that privacy practitioners are still ill-equipped to reason about privacy and utility in domains which do not match the "blueprints" provided by either standard differential privacy (for statistical datasets) or geo-location privacy.

[8] See for example [13] and [11].

1.2 Key themes in the practice of differential privacy

Since its introduction in 2006, researchers have been stretching differential privacy in various ways to respond to the needs of different domains. While theoretical research into privacy has thrown up an avalanche of new definitions [16], privacy practitioners are left with many questions: which definition should I use for my dataset? How do these definitions compare? How do I interpret what privacy means? And – surprisingly, still – how do I choose epsilon?

1.2.1 Privacy breaches as inference attacks

The separation between the theory and practice of differential privacy is perhaps most notable in the way that experimental researchers investigate privacy breaches: they are usually described as *inference attacks* [17–21]. Inference attacks describe real adversarial threats to a system, providing an *operational scenario* against which privacy mechanisms can be examined and (hopefully) defend. For example, privacy researchers have found that the list of apps on a users smartphone can reveal sensitive attributes such as religion or relationship status [22] and can even be used to identify particular individuals [23]. The need for privacy-preserving techniques in these areas is precisely to protect against these sorts of inferences. Plausible deniability does not have this goal: inferences describe the *potential* for re-identification; plausible deniability simply ensures that there is no *certainty* of it. Differential privacy is careful not to guarantee protection against inference attacks – and indeed, is known to be susceptible to them [24] – despite this being a major concern for practitioners. Even if plausible deniability is the best we can hope for, reasoning about vulnerability to inferences is essential for communicating what privacy means and what sort of privacy is guaranteed for individuals whose data is entrusted to a curator. While there has been some work on incorporating inference attack models into the differential privacy guarantee[9], the lack of significant research into the relationship between inferences and differential privacy means that there is still debate about what sort of protection differential privacy affords in practical adversarial scenarios [26].

1.2.2 How to choose epsilon

In the theoretical literature on differential privacy, the epsilon parameter in (1.1) is a *privacy* control; it describes both the "indistinguishability" level between secrets, and also the "privacy budget" available for subsequent data releases.[10] However, in practical scenarios epsilon is usually seen as a parameter for tweaking *utility* without associating any (specific) meaning to the resulting privacy guarantee [27]. Since epsilon is not meaningful to an end user, privacy practitioners can more easily brush aside concerns about its particular value. Indeed, Apple were lambasted for their cavalier use of differential privacy to provide some notional privacy to end

[9] We make particular note of the work on Pufferfish by Kifer et al. [25], which provides a framework for reasoning about differential privacy in the presence of inferences.

[10] The epsilon value of an *entire system* determines its overall privacy, and this privacy degrades with each release of data, as described by the differential privacy composition theorems. Hence the term "privacy budget" to describe it.

users, without revealing that their chosen epsilon in fact provided next-to-no privacy.[11] Even in situations for which differential privacy was explicitly designed, the question of *how to choose epsilon?* remains problematic [27–29]. Notably, this was an issue in the deployment of differential privacy for the 2020 US Census; researchers remarked that the "literature of differential privacy is very sparse" regarding policies for choosing a value for epsilon [29]. In more complex domains, the relationship between epsilon and the privacy afforded is potentially very tenuous: in a text privacy scenario, how would the value of epsilon correspond to the privacy-protection afforded to the author of a document? And how does changing epsilon correspond to changes in the useful information released? A clearer understanding of the relationship between epsilon and the privacy-utility balance is necessary to ensure its proper use in differential privacy in new and complex domains.[12]

1.2.3 Managing the privacy-utility balance

Some of the early adopters of differential privacy were researchers in machine learning, hoping to ensure privacy-protection for individuals whose data was used to train machine learning models. One of the research questions which arose out of this context was *where to add the noise?*. Researchers found that they could squeeze more utility from a dataset by the judicious application of noise to different parts of the machine learning workflow [30, 31]. Despite these early insights, the methods for determining *where* to add noise in a workflow remain ad hoc and largely experimental.

In more complex domains, the question of *how to add the noise?* also becomes relevant. For example, in the text privacy domain, the addition of noise to *words* in a document might seem like a reasonable approach – but *how* should the noise be added to guarantee some privacy and utility? In geo-location privacy and standard differential privacy, there is a clear relationship between privacy and utility which determines the nature of the noise-adding mechanism that should be applied. Whereas in text privacy, the relationship between privacy and utility is unclear, which complicates the decision of *how* to add noise.

These questions underlie the need for a more principled approach to understanding the nature of the privacy-utility balance and how to compare systems wrt their privacy and utility properties.

1.2.4 Reasoning about utility

A related issue is the lack of tools for reasoning about the *utility* of a differentially private data release, thus requiring practitioners to resort to manipulating epsilon to an experimentally determined value. This sort of problem is not restricted to privacy practitioners working in new domains. In 2020, the US Census Bureau chose to apply differential privacy to their statistical data release mechanisms – precisely the domain for which differential privacy was designed.

[11] https://www.wired.com/story/apple-differential-privacy-shortcomings/

[12] This has been neatly documented by Dwork et al. [28] in a survey on the use of epsilon across a range of businesses.

Even in this scenario, the practitioners reported fundamental difficulties, particularly around the design of mechanisms for utility:

> "Differential privacy lacks a well-developed theory for measuring the relative impact of added noise on the utility of different data products, tuning equity trade-offs, and presenting the impact of such decisions." [29]

If in fact utility is the sole purpose of a data release, the absence of a sound methodology for reasoning about utility in differentially private data releases remains a critical weakness.[13] And differential privacy's claim of independence from the particulars of a dataset is a moot point if utility can only be evaluated through dataset-specific experimentation.

1.3 Goals of this thesis

The aforementioned concerns constitute foundational issues which act as obstacles to the more widespread practical application of differential privacy. This motivates a more general study of privacy and utility for new contexts, as well as a sound methodology for applying differential privacy in complex scenarios.

The goal of this thesis is to address these fundamental questions: How do I design mechanisms which provide some level of privacy and some level of accuracy for a domain in which the privacy or the utility requirements are not well understood? How do I decide how and where to add noise in order to achieve my privacy and utility goals? How do I interpret what privacy means in my domain? And how safe is my system against adversarial threats?

While the above concerns are not new – indeed many works in the literature have made contributions to our understanding of these areas – our goal is to approach this study under a unifying framework for reasoning about privacy and utility and to situate our work in the metric differential privacy context. In this way, we aim to equip privacy practitioners with the tools to reason about privacy and utility, and thereby to assist in the application of differential privacy to novel domains.

1.3.1 Contributions

In this thesis we examine some of the foundational questions around privacy and utility posed in the previous section with the overall goal of advancing the understanding and adoption of differential privacy in new domains. In particular, this work makes the following contributions:

1. We use the framework of Quantitative Information Flow (QIF) to examine questions around privacy and utility in metric differential privacy.

2. We explore the relationship between the epsilon parameter of metric differential privacy mechanisms and the safety of systems against adversarial threats. We show how the *metric* on the domain of inputs relates to *how* privacy is applied, which in turn affects the types of attacks that the system can defend against.

[13] We argue: If a private data release has no utility, it serves no purpose other than to leak privacy. Utility, then is the primary goal of a private data release, and not privacy.

3. We model inference attacks against differential privacy and show that differential privacy's protection against such attacks is dependent on the particulars of the dataset. We show that *where* the noise is added in a privacy workflow determines how the trade-off between privacy and utility can be managed, and what the setting of epsilon means in these scenarios.

4. We introduce a framework for reasoning about optimality based on a Bayesian model of utility. We show how the metric determines the classes of consumers for which optimal mechanisms exist. We use our model to re-evaluate impossibility results on optimality, discovering a new characterisation of optimal mechanisms and demonstrating that optimal mechanisms do exist contra existing results in the literature. Finally, we prove a new result on the universal optimality of the Laplace mechanism for continuous domains.

5. We present 3 new applications of metric differential privacy to problems of interest. We show why an understanding of the metrics involved is important for determining the type of privacy to apply. We also show how to reason about privacy and utility in each situation using the understandings developed in this thesis.

1.3.2 Synopsis

This thesis is divided into 3 parts:

Part I: Foundations is intended as an introduction to the main technical material which forms the basis of the thesis.

- **Chapter 2: Quantitative Information Flow** introduces the QIF framework, which is an algebraic framework for modelling adversarial threats to secure systems by examining their information flow properties. While its use in security has been well-established, the QIF model has not been extensively used in differential privacy.[14] This chapter aims to provide a tutorial-style introduction to the basic QIF notation and results that are used extensively throughout the thesis. Of particular interest is the section on the geometry of hyper-distributions, which is a key motif used in this thesis, especially in the exploration of optimality in Chapter 6 and Chapter 7. This chapter is mainly based on material from the 'QIF book' by Alvim et al. [34].

- **Chapter 3: Metric Differential Privacy** introduces the existing work on **d**-privacy, pioneered by Chatzikokolakis et al. [15] and on which this thesis builds. In this chapter we introduce the canonical example of *geo-location* privacy for which **d**-privacy is most well-known, the metric-based reasoning used for privacy and utility in the standard and local models of differential privacy, and some commonly used **d**-privacy mechanisms which will be encountered throughout. Our contribution in this chapter is to present the technical details in the language and notation of QIF.

Part II: Analysis contains the theoretical chapters representing the technical contributions of

[14] Aside from the works by Alvim et al. to establish strong leakage bounds using a channel model for differential privacy [32, 33].

this thesis. These chapters rely heavily on QIF techniques to analyse privacy and utility properties of metric differential privacy.

- **Chapter 4: Comparing Privacy Mechanisms** explores the privacy-based order induced by the epsilon parameter of differential privacy. We show that this order is strictly weaker than the secure-refinement order of QIF which models safety against adversarial threats. We introduce 2 new orders modelling threat scenarios: a max-case refinement order for reasoning about worst-case adversarial threats – in the spirit of differential privacy – and a metric-based order which quantifies over all metrics, allowing one to safely reason about differential privacy in certain contexts when substituting one mechanism for another. We show that both of these orders are strictly stronger than the epsilon-order, demonstrating that the epsilon parameter of differential privacy provides surprisingly weak guarantees of safety against a variety of attacks. Finally, we show how the privacy ordering wrt epsilon depends on the *metric*, and that comparing mechanisms wrt their safety against all adversarial threats is only safe, in general, within the same metric family of differential privacy mechanisms. *(This chapter is based on the published papers "Comparing Systems: Max-case Refinement Orders and Application to Differential Privacy" [35] and "Refinement Orders for Quantitative Information Flow and Differential Privacy" [36].)*

- **Chapter 5: Inference Attacks** examines the privacy-utility trade-off from the point of view of inference attacks. We use QIF to model inferences as the leakage of secrets via unexpected correlations, thereby providing a simple proof of the "no free lunch" theorem of Kifer et al. [37] which says that there are no mechanisms which are optimal for both privacy and utility. We provide an abstract model of noise-adding mechanisms based on *where* the noise is added – whether to the "secret" or to the "useful" parts of the data – and show that this determines what sort of control epsilon provides, either privacy control (in the former case) or accuracy control in the latter. We also show that the amount of privacy or utility that can be obtained by manipulating epsilon is determined by how this information is *correlated* in the particular dataset, and we demonstrate this in experiments on a medium-sized database. *(This chapter is based on the published paper "On Privacy and Accuracy in Data Releases" [38].)*

- **Chapter 6: Optimality I: Discrete Mechanisms** introduces a framework for reasoning about optimal mechanisms for Bayesian consumers. Our work builds on the existing results of Ghosh et al. [39] for universal optimality of the Geometric mechanism; in this chapter we extend the study of optimality to metric domains and arbitrary classes of consumers using a novel geometric perspective based on 'hyper-distributions'. We formulate a new characterisation of optimality and use this to generalise and extend previous results in this area; importantly we discover new optimality results in the domain of "sum" queries, for which universal optimality had previously been deemed impossible [40]. *(This chapter is unpublished but is under preparation for submission.)*

- **Chapter 7: Optimality II: Continuous Mechanisms** uses the framework developed in Chapter 6 to prove a particular optimality result of interest: that the Laplace mechanism is *universally optimal* for Bayesian consumers on a continuous domain in the same way that the

Geometric mechanism is *universally optimal* in the discrete case. Our result is novel and is the first positive result on optimality for the Laplace mechanism, which has previously shown only to be non-optimal in various discrete settings. Our approach is made possible by the use of the algebraic framework of QIF, for which results on refinement naturally extend to the continuous domain, allowing the extension of our optimality results from Chapter 6 to the continuous setting. *(This chapter is based on the paper "The Laplace Mechanism has optimal utility for differential privacy over continuous queries" [41].)*

Part III: Applications presents new practical applications of metric differential privacy to problems of interest.

- **Chapter 8: Statistical Utility for Local Differential Privacy** shows how to apply metric-based reasoning to the problem of protecting the privacy of individuals while preserving some statistical utility, namely the ability to reconstruct the original (true) distribution of answers given a noisy distribution. We demonstrate experimentally, using geo-location points as a motivating example, that a mechanism designed for the Euclidean distance outperforms the (commonly deployed) mechanism for local differential privacy when the utility of the output is affected by the underlying ground distance between the estimated distribution and the true distribution. *(This chapter is based on the published paper "Utility-Preserving Privacy Mechanisms for Counting Queries" [42].)*

- **Chapter 9: Text Document Privacy** demonstrates an application of metric differential privacy to a problem in text document privacy, namely hiding the writing style of an author. We demonstrate how to reason about privacy and utility in this space, interpreting what the privacy guarantee means for authors when differential privacy is applied to documents. We produce a novel **d**-privacy mechanism based on the Earth Mover's distance and show, using experiments, that our mechanism provides reasonably good utility when outputting documents with the purpose of preserving their topicality. Our mechanism is the first of its kind in the domain of text document privacy, thus representing a genuinely novel application of differential privacy. *(This chapter is based on the published papers "Generalised Differential Privacy for Text Document Processing" [12] and "Processing Text for Privacy: An Information Flow Perspective" [8].)*

- **Chapter 10: Locality Sensitive Hashing with Differential Privacy** explores a problem encountered in recommender systems, namely how individuals can privately relinquish their purchase histories (or other sensitive item lists) to an untrusted data provider, who then requires some utility from the noisy release. We observe that a commonly used nearest neighbour search technique – Locality Sensitive Hashing – can be seen as a probabilistic mapping between metric spaces, and therefore permits metric-based reasoning about privacy and utility when used in conjunction with **d**-private mechanisms. We use this observation to construct an (approximate) d_θ-private mechanism designed for the angular distance d_θ which we show experimentally has reasonably good utility compared with its non-private counterpart. *(This*

chapter is under submission and is available as the arXiv preprint "Locality Sensitive Hashing with Extended Differential Privacy" [43].)

1.3.3 Publications

The following is a list of published (peer-reviewed) papers that I have co-authored during the course of my doctoral studies, all of which have been used as the basis for this thesis.

1. N. Fernandes, A. McIver, and C. Morgan, "The laplace mechanism has optimal utility for differential privacy over continuous queries," in *ACM/IEEE Symposium on Logic in Computer Science (LICS 2021) (to appear)*, IEEE, 2021.

2. K. Chatzikokolakis, N. Fernandes, and C. Palamidessi, "Refinement orders for quantitative information flow and differential privacy," *Journal of Cybersecurity and Privacy*, vol. 1, no. 1, pp. 40–77, 2021.

3. M. S. Alvim, N. Fernandes, A. McIver, and G. H. Nunes, "On privacy and accuracy in data releases (invited paper)," in *31st International Conference on Concurrency Theory, CONCUR 2020, September 1-4, 2020, Vienna, Austria (Virtual Conference)* (I. Konnov and L. Kovács, eds.), vol. 171 of *LIPIcs*, pp. 1:1–1:18, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

4. K. Chatzikokolakis, N. Fernandes, and C. Palamidessi, "Comparing systems: Max-case refinement orders and application to differential privacy," in *32nd IEEE Computer Security Foundations Symposium, CSF 2019, Hoboken, NJ, USA, June 25-28, 2019*, pp. 442–457, IEEE, 2019.

5. N. Fernandes, L. Kacem, and C. Palamidessi, "Utility-preserving privacy mechanisms for counting queries," in *Models, Languages, and Tools for Concurrent and Distributed Programming - Essays Dedicated to Rocco De Nicola on the Occasion of His 65th Birthday* (M. Boreale, F. Corradini, M. Loreti, and R. Pugliese, eds.), vol. 11665 of *Lecture Notes in Computer Science*, pp. 487–495, Springer, 2019.

6. N. Fernandes, M. Dras, and A. McIver, "Generalised differential privacy for text document processing," in *Principles of Security and Trust - 8th International Conference, POST 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings*, pp. 123–148, 2019.

7. N. Fernandes, M. Dras, and A. McIver, "Processing text for privacy: An information flow perspective," in *Formal Methods - 22nd International Symposium, FM 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 15-17, 2018, Proceedings* (K. Havelund, J. Peleska, B. Roscoe, and E. P. de Vink, eds.), vol. 10951 of *Lecture Notes in Computer Science*, pp. 3–21, Springer, 2018.

The following paper is currently under submission and represents a further contribution of this thesis.

1. N. Fernandes, Y. Kawamoto, and T. Murakami, "Locality sensitive hashing with extended differential privacy," *CoRR*, vol. abs/2010.09393, 2020.

Part I

Foundations

2

Quantitative Information Flow

The theory of Quantitative Information Flow (QIF) has its foundations in the study of secure systems which process sensitive information. Such systems can include, for example, computer programs and communication protocols. A fundamental goal in the design of these systems is ensuring that sensitive or secret information is not leaked; such information leaks could be exploited by adversaries who might be able to gain some advantage, such as stealing login credentials, obtaining sensitive data about an individual, or intercepting secret communications. We might conclude that our security goal should be *non-interference* – that is, no information leakage whatsoever.

Unfortunately, non-interference is not always possible, because some systems leak information by design. Consider, for example, an electronic voting system. In an election, the process for determining the winner is by counting votes and *releasing* the vote count. However, such a release could be problematic if the vote of each individual is to remain a secret. For example, in a small town, the vote of each individual might be easy to guess if the vote count for the town is released; moreso if every individual votes for the same candidate, in which case every person's vote is revealed.

Instead of opting for non-interference (which would make the afore-mentioned voting system untenable), we may instead decide that information leaks are to be tolerated, provided that these leaks are 'small'. By quantifying the leakage of the system we can determine how *vulnerable* the system is to attack - measured by the size of the leak and how much an attacker can use the information leaked to their advantage. Moreover, we can use leakage measures to *compare* systems with respect to their vulnerability to attack. This motivates the study of how to quantify information leakage in secure systems.

QIF provides an algebraic framework for quantifying information leakage by modelling how

information "flows" through a system, and how these flows may "leak" sensitive information to an observer. The framework allows modelling of adversarial scenarios describing how an attacker can utilise the leaked information to gain some advantage. Using the QIF model we can quantify the "amount" of leakage, which allows us to determine how much use an attacker can get out of such leakage. In this way, theoretical leakage analyses can be tied to practical adversarial risks, thereby providing a comprehensive view of the security of the system. QIF techniques have previously been used in the analysis of security protocols [44–46], voting systems [47], attacks on cryptographic systems [48–50], security properties of computer programs [51, 52] and differential privacy [53].

In this thesis we will apply QIF to the study of privacy and utility properties of differentially private systems. This chapter provides an introduction to QIF focussing on the tools we will need to analyse these properties. We begin with an introduction to the channel model from information theory (§2.1) which forms the basic model for secure systems in QIF. We then introduce adversarial modelling (§2.2) using the idea of the 'vulnerability' of a secret to an attack, followed by the notion of 'refinement' (§2.3) by which we can compare channels wrt their security properties. Finally, we explore hyper-distributions (§2.4) with a particular focus on their geometric properties which will be of significant interest in later chapters.

2.1 Modelling Systems as Channels

All of the systems that we will encounter are *probabilistic* and thus can be described using the channel model from information theory, which is the fundamental model we will adopt. A channel is a probabilistic mapping C from (discrete) inputs $\mathcal{X}$ to (discrete) outputs $\mathcal{Y}$ which we write as the type $\mathcal{X} \to \mathbb{D}\mathcal{Y}$. We denote by $C(x)(y)$ the probability that output $y \in \mathcal{Y}$ is produced from input $x \in \mathcal{X}$. A channel $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ can *equivalently* be described as a row-stochastic matrix[1] which we call the 'channel matrix', which has type $\mathcal{X}\times\mathcal{Y} \to [0,1]$, where each row corresponds to $x \in \mathcal{X}$, each column to $y \in \mathcal{Y}$ and each entry $C_{x,y}$ has the value $C(x)(y)$. We will abuse notation and write $C_{x,y}$ for a channel described as type $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, with the understanding that these types are isomorphic.

We model a system as a probabilistic channel $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ defined over a space of *secret values* $\mathcal{X}$, and producing *observations* $\mathcal{Y}$. We think of $\mathcal{X}$ as the set of values that the secret can take and $\mathcal{Y}$ as the set of observations that an adversary can make. We will be interested in modelling *Bayesian* adversaries,[2] which are adversaries who maximise the information they can learn from a system by updating their knowledge about the secret values using Bayes' rule. We model the background knowledge of the adversary as a prior, or probability distribution, over secret values, denoted π of type $\mathbb{D}\mathcal{X}$. We denote by π_x the probability that π assigns to the secret value x.

As mentioned above, we model the adversary's interaction with the channel C using Bayes' rule. The adversary combines their prior knowledge π with the probabilities in the channel

[1] Row-stochastic means that each row sums to 1.

[2] We use Bayesian adversaries in our models because this represents the best possible rational attacker, under the assumption that adversaries are also information-theoretic. ie. They are not computationally bound.

C to obtain a joint distribution J over $\mathcal{X} \times \mathcal{Y}$.[3] Taking marginals along each Y component of J produces a set of *posterior* distributions over $\mathcal{X}$, each of which represents the adversary's updated knowledge (ie. a posteriori knowledge) after observing an output y. The marginal probability on Y is a distribution of type $\mathbb{D}\mathcal{Y}$, and each associated posterior is a conditional distribution on $\mathcal{X}$, namely the probability of observing $y \in \mathcal{Y}$ for each $x \in \mathcal{X}$. We write a_y for the marginal probability observing y and δ^y for its associated posterior. Overall this structure of posteriors with its associated (marginal) probability can be represented as a *distribution of distributions* which we call a *hyper-distribution* or *hyper*, and has type $\mathbb{D}(\mathbb{D}\mathcal{X})$ or $\mathbb{D}^2\mathcal{X}$. We will often write a hyper as $[\pi \triangleright C]$, indicating the channel C and prior π from which it was obtained, and we use the term *pushing a prior through a channel* to indicate the process for obtaining a hyper.

An example of how this is done is presented in Example 2.1. Note that this terminology will be explained in more detail in the upcoming sections.

> **REMARK 2.1.1.** Our channel model (using matrices) assumes that the input and output spaces are discrete; this assumption will be adequate for this thesis apart from Chapter 7 where we will encounter continuous channels. We will then make a clearer distinction between channels as probabilistic functions and channels as matrices.

2.2 Modelling Adversaries

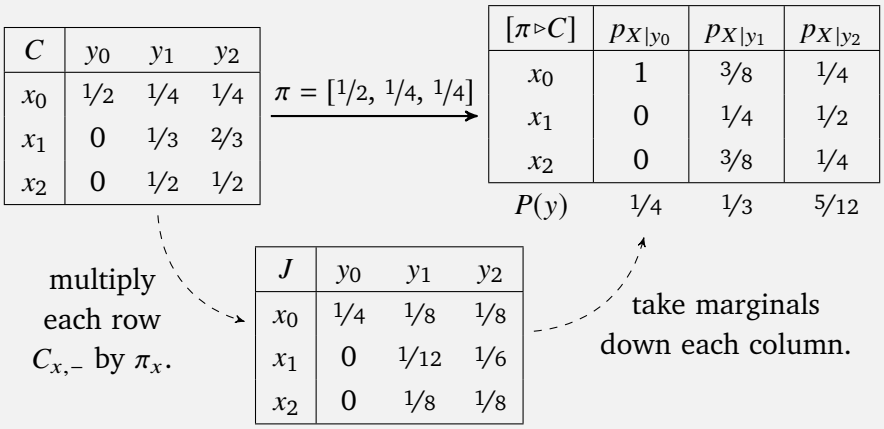
The strength of a secure system is determined by its ability to defend against adversarial attacks. An effective model of security should then be able to account for particular adversarial threats and describe how effective the system is in protecting against those threats. In choosing how to model threats to a system, we may be tempted to adopt one of the traditional information-theoretic models of leakage based on notions of entropy, such as Shannon entropy or mutual information [54], as these are well-studied and well-established in the areas of communication and transmission. However, in his foundational work on QIF, Smith [55] gives examples where Shannon entropy underestimates the vulnerability of a system against certain attacks, because it does not precisely model the *operational scenario* of interest. This insight has led to a general theory of g-vulnerability, introduced by Alvim et al. [56], which models particular adversarial scenarios through the use of gain functions which describe an adversary's gain from information flows in the system. This model has developed into an elegant and robust theory of leakage which forms the basis of what we now call Quantitative Information Flow. In this section we introduce the basic ideas around g-vulnerability and leakage by which we model adversaries.

2.2.1 Vulnerability and Gain Functions

We model *threats* to a system using an operational scenario that describes how an attacker might exploit their knowledge to gain information about a secret. An attacker is assumed to have a set

[3] We always assume that the adversary has access to the channel, avoiding *security through obscurity*.

Example 2.1. Given a channel C and a prior π, an adversary can update their knowledge about the secret using Bayes' rule. Recall that Bayes' rule says: $P(y|x)*P(x) = P(x|y)*P(y)$. Each row $C_{x,-}$ is a distribution $P(Y|X = x)$, which the adversary multiplies by $\pi_x = P(x)$ to produce the joint matrix J whose element $J_{x,y}$ is the joint probability $P(Y = y, X = x)$. Normalising along each column of J produces, for each y, a posterior distribution $P(X|Y = y)$ along with its outer distribution $P(Y = y)$.



The set of posteriors along with the outer (marginal) distribution on Y is called a hyper-distribution or *hyper*, denoted $[\pi \triangleright C]$. We say that the action of *pushing* π *through* C produces the hyper $[\pi \triangleright C]$.

In this example, the hyper $[\pi \triangleright C]$ has 3 posteriors, each corresponding to the adversary's updated knowledge after observing the output from the channel. For example, if the adversary observes y_0 then he knows precisely what the secret is (it is x_0), corresponding to the posterior $(1, 0, 0)$. On the other hand, if he observes y_2 then he deduces that the most likely secret is x_1. The outer probabilities $P(y)$ attached to these posteriors are the probabilities that he will be able to make those deductions at all.

of *actions* that they can take, with each action providing some *gain* to the adversary depending on the true value of the secret. An action may correspond to a *guess* of the secret, but this correspondence is not assumed – an attacker's best option may be to *not* make a guess, which can also be included as a valid action, or to guess, for example, some value close to the secret, or a property of the secret. Our model is thus sufficiently general to cover a wide range of scenarios.

We use a *gain function* to model the set of actions available to an adversary. A gain function over a (possibly infinite) set of *actions* $\mathcal{W}$ and secrets $\mathcal{X}$ is a function $g : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$. A value of 0 indicates 'no gain' and higher values represent 'more gain' for the adversary. Different adversaries can be modelled by proper choices of $\mathcal{W}$ and g. (See Example 2.2 and Example 2.3).

We model the adversary's prior knowledge (or uncertainty) about the secret value as a distribution $\pi : \mathbb{D}\mathcal{X}$. Therefore we can describe the *vulnerability* of the secret wrt a particular adversary

Example 2.2 (Property Gain Function)**.** We can model an adversary who wants to guess a *property* of the secret by choosing some non-empty subset of values $W \subset \mathcal{X}$ with the desired property and defining the set of actions $\mathcal{W} = \{W, \mathcal{X}\backslash W\}$. Then, we define the gain function

$$g_W(w,x) \quad := \quad \begin{cases} 1, & \text{if } x \in W, \\ 0, & \text{otherwise.} \end{cases}$$

This gain function assigns the value 1 whenever the secret x has the desired property, and 0 otherwise. This could be useful, for example, if the secrets represented salaries and the adversary is interested in guessing a particular *salary range*. For example, if the secret takes on values $\{x_1 = \$10\text{k}, x_2 = \$20\text{k}, x_3 = \$60\text{k}, x_4 = \$100\text{k}\}$ then an adversary who wants to guess whether the salary is greater than \$50k would use the following gain function:

g_W	w_1	w_2
x_1	0	1
x_2	0	1
x_3	1	0
x_4	1	0

Property gain functions can be naturally extended to gain functions which describe equivalence classes of $\mathcal{X}$, by defining disjoint sets $W_1, W_2, \ldots W_n \subset \mathcal{X}$ and actions $\mathcal{W} = \{W_1, W_2, \ldots, W_n, \mathcal{X}\backslash(\cup_i W_i)\}$.

using a function V_g on the adversary's prior π which we call a 'g-vulnerability'.

We denote by $\mathbb{G}\mathcal{X}$ the set of all gain functions.

Definition 2.2.1. Given a gain function $g\colon \mathbb{G}\mathcal{X}$, the g-vulnerability of a prior $\pi\colon \mathbb{D}\mathcal{X}$ is

$$V_g(\pi) \quad := \quad \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi_x g(w,x) \ .$$

When $\mathcal{W}$ is infinite, max should be replaced by sup.

This is the *maximum* expected gain that the adversary enjoys using their prior knowledge π of the secrets.

The canonical example for vulnerability is the *Bayes vulnerability* function, denoted V_1, which models an adversary who wishes to guess the secret in one try.

Definition 2.2.2 (Bayes vulnerability)**.** The *Bayes vulnerability* of $\pi\colon \mathbb{D}\mathcal{X}$ is given by

$$V_1(\pi) \quad := \quad \max_{x \in \mathcal{X}} \pi_x \ .$$

Bayes vulnerability says that this adversary's best option is to choose the value $x \in \mathcal{X}$ that has the highest probability, thus it measures the vulnerability of the secret as the most likely value under the adversary's prior.

Although Def. 2.2.2 does not explicitly include a gain function, we can write down the gain function corresponding to V_1; it is called the *identity* gain function.

Definition 2.2.3 (Identity gain function)**.** The *identity* gain function $g_{\text{id}}: \mathcal{X} \times \mathcal{X} \to \{0,1\}$ is given by:

$$g_{\text{id}}(w,x) \quad := \quad \begin{cases} 1, & \text{if } w = x, \\ 0, & \text{if } w \neq x \end{cases} .$$

Note that for the identity gain function the set of actions corresponds to the set of secret values $\mathcal{X}$, and so each action can be thought of as 'making a guess' for the value of the secret. Further illustrations of gain functions are presented in Example 2.2 and Example 2.3.

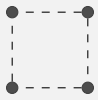
Finally, we may wonder how many adversaries the g-vulnerability framework can model. In fact, it has been shown [57] that any continuous, convex function of $\mathbb{D}\mathcal{X}$ can be written as a V_g, and this is a fundamental characterisation of g-vulnerabilities. Popular *concave* functions such as Shannon entropy can equally be modelled in this framework using the dual notion of loss functions which we describe in §2.2.1.

Example 2.3 (Metric Gain Function)**.** We can model an adversary who wants to guess a value *close* to the secret using a normalised metric $\mathbf{d} : \mathcal{W} \times \mathcal{X} \to [0,1]$:

$$g_{\mathbf{d}}(w,x) \quad := \quad 1 - \mathbf{d}(w,x) \ .$$

This gain function assigns higher values to secrets that are closer together, and might be useful when the secrets represent locations and the adversary wants to guess a nearby location. For example, say that the true locations are the grid points $\mathcal{X} = \{(0,0),(0,1),(1,0),(1,1)\}$ shown below.

$p_1 = (0,0)$ $\qquad$ $p_4 = (0,1)$



$p_2 = (1,0)$ $\qquad$ $p_3 = (1,1)$

We model an adversary using the $g_{\mathbf{d}}$ gain function with metric $\mathbf{d}(p,p') = \frac{1}{\sqrt{2}}\mathbf{d}_2(p,p')$, and whose prior over the secrets is uniform (υ). If the only actions available to the adversary are guesses of the secret (ie. guess w_i corresponds to point p_i) then we can write down the adversary's gain function as follows:

$g_{\mathbf{d}}$	w_1	w_2	w_3	w_4
p_1	1	$1 - \frac{1}{\sqrt{2}}$	0	$1 - \frac{1}{\sqrt{2}}$
p_2	$1 - \frac{1}{\sqrt{2}}$	1	$1 - \frac{1}{\sqrt{2}}$	0
p_3	0	$1 - \frac{1}{\sqrt{2}}$	1	$1 - \frac{1}{\sqrt{2}}$
p_4	$1 - \frac{1}{\sqrt{2}}$	0	$1 - \frac{1}{\sqrt{2}}$	1

We can then compute the vulnerability of the secret values to being guessed by this adversary:

$$V_{g_{\mathbf{d}}}(\upsilon) \;=\; \max_{w} \sum_{p} \upsilon_p g_{\mathbf{d}}(w,p) \;=\; \frac{1}{4}(1+2-\frac{2}{\sqrt{2}}) \;=\; \frac{1}{4}(3-\sqrt{2}) \tag{2.1}$$

which holds for any action $w \in \mathcal{W}$.

However, if the actions available to the adversary include guesses for *any* point within the square defined by the vertices p_1, p_2, p_3, p_4, then the set of guesses $\mathcal{W}$ is infinite, and the adversary's gain function includes, for example, the following column:

$g_{\mathbf{d}}$	w_i
p_1	$1/\sqrt{2}$
p_2	$1/\sqrt{2}$
p_3	$1/\sqrt{2}$
p_4	$1/\sqrt{2}$

which corresponds to a guess of the point $(\frac{1}{2}, \frac{1}{2})$. In fact, the adversary can then always do better now by guessing the point $(\frac{1}{2}, \frac{1}{2})$, since

$$V_{g_{\mathbf{d}}}(\upsilon) \;=\; \max_{w} \sum_{p} \upsilon_p g_{\mathbf{d}}(w,p) \;=\; \frac{1}{4}(4 * \frac{1}{2}) \;=\; \frac{1}{4}(2) \tag{2.2}$$

which is larger than the vulnerability computed in (2.1). Thus this adversary gains more by guessing a point *not* in the set of secrets than he does by guessing one of the secret values.

Loss Functions

Instead of modelling an adversary's gain upon taking some action, we could dually model their loss using a loss function $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$. We would then talk about the adversary's *uncertainty* wrt the secret as measured by their expected loss in the following way:

> **Definition 2.2.4** (ℓ-Uncertainty)**.** Given $\pi : \mathbb{D}\mathcal{X}$ and a loss function $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, the ℓ-uncertainty of π is defined as:
>
> $$U_\ell(\pi) \;:=\; \min_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi_x \ell(w,x)\ .$$
>
> When $\mathcal{W}$ is infinite, min should be replaced by inf.

ℓ-uncertainty is the dual of g-vulnerability, and indeed we can convert between the two by setting $g = B - \ell$ for some upper bound B, yielding $V_g = B - U_\ell$.

Uncertainties are typically used to express the information-theoretic notion of *entropies*; common examples encountered in the literature include Shannon entropy [54], Guessing entropy [58] and Bayes risk [59]. All of these entropies can be modelled in the QIF framework

using loss functions and ℓ-uncertainty. In fact, Bayes risk is exactly the dual of Bayes vulnerability in Def. 2.2.2; that is, Bayes risk U_1 is $1 - V_1$. Dual to g-vulnerabilities, ℓ-uncertainties can model any function that is continuous and *concave*.

> **Remark 2.2.1.** All of the results given in this chapter will use the gain function model for adversaries rather than loss functions, as is standard in QIF. However, in Chapter 6, we will model optimality for **d**-privacy using loss functions. All results presented in this chapter for V_g functions have dual versions for U_ℓ which will be presented as required in Chapter 6.[4]

2.2.2 Posterior Vulnerability

We saw in Example 2.1 how an adversary can update their knowledge about the secret values $\mathcal{X}$ using Bayes' rule to obtain a set of posterior distributions and an outer distribution, the combination of which we called a 'hyper-distribution', or simply 'hyper', denoted as $[\pi \triangleright C]$ for prior $\pi : \mathbb{D}\mathcal{X}$ and channel C.

Once we have computed the hyper-distribution $[\pi \triangleright C]$, we need to decide how to measure the adversary's gain in knowledge from observing an output from C. The usual way to do this is to compute the adversary's *expected* gain over all possible observations and we call this the *posterior g-vulnerability*.

> **Definition 2.2.5.** Given a gain function $g : \mathbb{G}\mathcal{X}$, a prior $\pi : \mathbb{D}\mathcal{X}$ and a channel C, the posterior g-vulnerability $V_g[\pi \triangleright C]$ is defined as
>
> $$V_g[\pi \triangleright C] \quad := \quad \sum_y a_y V_g(\delta^y)$$
>
> where we write a_y for the marginal probability on observation y and δ^y for its associated posterior.

Notice that we overload the function V_g here: it can be applied to priors (as a g-vulnerability – Def. 2.2.1) or hypers (as posterior g-vulnerability – Def. 2.2.5).

It is sometimes more convenient to write Def. 2.2.5 in terms of the channel C and prior π, which we can do as follows:

> **Theorem 2.1** (Thm 5.7 in [34]). Given a gain function $g : \mathbb{G}\mathcal{X}$, a prior $\pi : \mathbb{D}\mathcal{X}$ and a channel C, the posterior g-vulnerability of $\mathcal{X}$ can be written as
>
> $$V_g[\pi \triangleright C] \quad = \quad \sum_y \max_w \sum_x \pi_x C_{x,y}\ g(w,x)\ .$$

Note that the adversary computes their expected gain over all posteriors, and not their *realised* gain after a particular observation, since it may be that the adversary's gain *decreases* after

[4] The only missing result for loss functions in QIF is a dual for the so-called 'miracle theorem', which will not concern us here since we do not make recourse to it in our results.

making an observation (this could be true for example if the adversary's prior beliefs are incorrect). The expected gain therefore gives a better indication of the vulnerability of the secret, given the adversary's prior.

> **Remark 2.2.2.** We call the posterior vulnerability computed in Def. 2.2.5 the *average-case* vulnerability (corresponding to average-case adversaries,) since the vulnerability is averaged over all posteriors. However, in some situations the average-case vulnerability may underestimate the threat to a secret. In these cases, a *max-case* notion would be more suitable, and we can define a corresponding *max-case* notion of vulnerability (for max-case adversaries). This will be explored in detail in Chapter 4.

2.2.3 Leakage

Our models above provide a way to compute the vulnerability of a channel before and after an observation wrt an adversary represented by a prior over secrets and a gain function that models their attack strategy. We can now combine the prior and posterior g-vulnerabilities into a measure of *leakage* for the channel. The leakage of the channel wrt an adversary should correspond (in some way) to the information gain of the adversary. The two ways in which this is modelled in QIF is via *additive* and *multiplicative* leakage [60].

> **Definition 2.2.6** (g-Leakage)**.** Given prior $\pi : \mathbb{D}\mathcal{X}$, gain function $g : \mathbb{G}\mathcal{X}$ and channel $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, the *additive* g*-leakage* of C is given by
>
> $$\mathcal{L}^{+}_{g}(\pi, C) \quad := \quad V_g[\pi \triangleright C] \; - \; V_g(\pi)$$
>
> and the *multiplicative* g*-leakage* of C is given by
>
> $$\mathcal{L}^{\times}_{g}(\pi, C) \quad := \quad \frac{V_g[\pi \triangleright C]}{V_g(\pi)}$$
>
> with the special cases $\mathcal{L}^{\times}_{g}(\pi, C) = 0$ when $V_g[\pi \triangleright C] = V_g(\pi) = 0$ and $\mathcal{L}^{\times}_{g}(\pi, C) = \infty$ when $V_g(\pi) =$ otherwise.

The minimum additive g-leakage is 0 and the minimum multiplicative g-leakage is 1 – both occurring when prior and posterior g-vulnerabilities are the same (ie. for a channel which leaks no information).

Notice that when *comparing* channels wrt a particular adversary, it is sufficient to compare their posterior g-vulnerabilities, since prior g-vulnerability is independent of the channel.

A note on leakage measures

From the information-theoretic perspective, channels are characterised by their *leakage* properties. The leakage of a channel represents the amount of information that can be learned from a channel by an observer, who we usually regard as an *adversary*. However, as we have seen,

leakage depends on the characteristics of the adversary, such as their prior knowledge about the inputs to the channel, as well as the gain function which models the information that they wish to learn. We adopt a *Bayesian* model of inference – that is, we assume that the adversary utilises all of their prior knowledge plus their knowledge of the channel to maximise their gain.

We may wonder if there are other adversarial models which might describe the leakage of the channel differently. Consider, for example, an adversary who does not have a prior over secrets and instead models their gain in knowledge just by observing an output from a channel. In this case, the following channel may appear to be very informative:

C	y_0	y_1
x_0	$9/10$	$1/10$
x_1	$1/10$	$9/10$

If the adversary's gain is modelled by the Bayes vulnerability gain function (Def. 2.2.2), then they might compute their expected gain from using the channel C as 0.9, since each observation (y_0, y_1) gives a gain of 0.9 by choosing the corresponding secret values (x_0, x_1). (Note that this is also the expected posterior vulnerability of a Bayesian adversary with a uniform prior over secrets). However, if the Bayesian adversary has a non-uniform prior over secret values such as $\pi = (0.99, 0.01)$ then, surprisingly, the channel C leaks no information whatsoever to the adversary. This might occur, for example, when the channel C models an imperfect test for a disease in a population in which the disease is highly unlikely to occur.[5] This shows that by using different leakage measures we can draw quite different conclusions if all of the data available (such as prior probabilities) is not taken into account. (Note that different Bayesian adversaries can also draw different conclusions depending on their priors, and we will explore these ideas further in §2.3).

In this thesis we adopt the position that the leakage measures we use will be *information flow* measures. This means that they should satisfy some basic axioms which we consider to be 'sanity checks', in order that our reasoning about the leakage of the channel is sound. For example, we would expect that any channel which leaks its secrets exactly, such as:

$\mathbf{0}$	y_0	y_1	y_2
x_0	1	0	0
x_1	0	1	0
x_2	0	0	1

would have the maximum leakage under any reasonable measure, and any channel which leaks nothing about its secrets, such as the following:

$\mathbf{1}_a$	y_0	y_1	y_2
x_0	$1/3$	$1/3$	$1/3$
x_1	$1/3$	$1/3$	$1/3$
x_2	$1/3$	$1/3$	$1/3$

$\mathbf{1}_b$	y_0	y_1	y_2
x_0	1	0	0
x_1	1	0	0
x_2	1	0	0

$\mathbf{1}_c$	y_0	y_1	y_2	y_3	y_4
x_0	$1/5$	$1/5$	$1/5$	$1/5$	$1/5$
x_1	$1/5$	$1/5$	$1/5$	$1/5$	$1/5$
x_2	$1/5$	$1/5$	$1/5$	$1/5$	$1/5$

[5] This is an example of the base-rate fallacy.

would have the minimum possible leakage under a valid information measure. In addition, we expect that information leakage cannot *decrease* after an observation from a channel; that is, an adversary's posterior knowledge cannot be less than his prior knowledge. This is the property of *monotonicity*. Finally, we require that an adversary cannot learn any more information from a post-processing step than they could have learned from the channel alone – in other words, information leakage cannot increase without access to the channel. This is also known as the *data processing inequality*.

These axioms have been well-studied in the QIF literature and it has been shown that the family of g-vulnerabilities satisfies these basic leakage axioms using either expected (average-case) gain or maximum (worst-case) gain measures for posterior g-vulnerability, and wrt both additive and multiplicative notions of leakage [57, 60].

2.3 Comparing Channels: Refinement

A fundamental question arises in the study of leakage: can we guarantee that system B is safer than system A? Such a scenario may occur when, for example, we are given a system A and we wish to swap it for some new system B without compromising the security promised by A. Using the ideas of vulnerability and gain functions established earlier, we might wonder if we should consider some particular adversary or class of adversaries, modelled as gain functions with priors, in order to decide whether to accept B as a safe replacement for A. Such a comparison would not be particularly *robust* – our security guarantees would only hold so long as our adversarial assumptions were met.

A robust guarantee would quantify over *all* priors and *all* gain functions, guaranteeing that the replacement of B for A is safe against all possible adversarial threats. Remarkably, QIF provides the tools to be able to reason in this way. The primary mechanism we use is called *refinement*: this defines an ordering between channels which allows us to compare them wrt their safety against all adversarial scenarios. Refinement has both a structural characterisation (so that it can be easily verified) and an operational or 'testing' characterisation (which produces a counter-example or *witness* in the case that refinement fails).

We present first the operational characterisation, called the *average-case leakage order*, which says that channel B is at least as safe as channel A if B leaks no more than A, for *all* priors π and *all* gain functions $g : \mathbb{G}\mathcal{X}$. Since comparing leakage is equivalent to comparing posterior vulnerability, we only require posterior vulnerabilities in our definition.

Definition 2.3.1. The average-case leakage order is defined as

$$A \sqsubseteq^{\text{avg}}_{\mathbb{G}} B \qquad \text{iff} \qquad V_g[\pi \triangleright A] \geq V_g[\pi \triangleright B]$$

for all $g : \mathbb{G}\mathcal{X}$, $\pi : \mathbb{D}\mathcal{X}$.

The average-case leakage order provides clear guarantees in terms of an *operational* scenario – if $A \sqsubseteq^{\text{avg}}_{\mathbb{G}} B$ holds, it means that *any* adversary, who can be modelled using a gain function, will

learn at least as much from observing the output of channel B as they will from channel A. Despite this being such a strong guarantee, it is not immediately useful – it is unclear how we would *verify* such a guarantee which requires quantification over all gain functions.

This can be achieved using its *structural* characterisation, defined in terms of the channel matrix, which we call the *average-case refinement order*.

> **Definition 2.3.2.** The average-case refinement order is defined as: $A \sqsubseteq B$ iff $AR = B$ for some channel R.

This says that B can be obtained from A by a post-processing step modelled as a channel. [6] Note that the average-case refinement order is only a preorder, but can be turned into a partial order by determining equivalence classes of channels which have the same leakage properties.

The following fundamental result was proven by McIver et al. [56, 61].

> **Lemma 2.2.** The orders $\sqsubseteq$ and $\sqsubseteq^{\text{avg}}_{\mathbb{G}}$ coincide. That is, $A \sqsubseteq B$ iff $A \sqsubseteq^{\text{avg}}_{\mathbb{G}} B$.

In other words, whenever we can find a *witness* R to verify the average-case refinement order, we can interpret this as saying that B is as safe as A against *all* adversaries, using an average-case notion of leakage. Consequently, we read $A \sqsubseteq B$ as "A is refined by B", or "B is as safe as A". The refinement relation also expresses the *data-processing inequality*, which says that post-processing can never cause leakage to increase.

The equivalence between Def. 2.3.2 and Def. 2.3.1 assures us that the refinement order is *sound*: that is, we know that whenever we find a witness R so that $A \sqsubseteq B$, we can be sure that there is no adversary who will (strictly) prefer B to A. But we also would like to know whether the refinement is *complete*: that is, if we fail to find a witness R, so that we conclude $A \not\sqsubseteq B$, can we also be sure that there exists an adversary who prefers B to A?

Fortunately, we have both *soundness* and *completeness*; not only do we know that such an adversary exists, but we can construct the "counter-example" gain function modelling this adversary. [7] We will see how this is done in Chapter 4 (§4.5) when we explore refinement properties for differential privacy.

2.4 Hyper-Distributions

In this section we will highlight important characteristics of hypers which make them useful in the study of leakage. We begin with a formal definition.

> **Definition 2.4.1 (Hyper-Distribution).** Given a prior $\pi : \mathbb{D}\mathcal{X}$ and a channel $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ we define the hyper-distribution produced by 'pushing π through C' as

$$[\pi \triangleright C] \quad := \quad \sum_{y \in \mathcal{Y}} a_y [\delta^y]$$

[6] Note that post-multiplication by a channel is equivalent to post-processing.
[7] Whether this adversary is of practical interest or not is a different issue.

where δ^y is the distribution $p(X|Y=y)$ called a 'posterior' given by $\delta^y_x = \frac{\pi_x C_{x,y}}{\sum_{z\in\mathcal{X}} \pi_z C_{z,y}}$; and a_y is the marginal on observation y given by $a_y = \sum_{x\in\mathcal{X}} \pi_x C x, y$.

A hyper has type $\mathbb{D}(\mathbb{D}\mathcal{X})$ which we write as $\mathbb{D}^2\mathcal{X}$. The distribution over y given by the marginals a_y is called the 'outer' distribution and each posterior distribution δ^y is also called an 'inner'. We sometimes also call each a_y associated to δ^y the 'outer probability of the inner δ^y'. Note that the probability a_y is undefined when $C_{-,y}$ is a column of 0's – instead we 'throw out' this column of the channel C as it has no effect on the C's leakage.

We will usually write hypers as tables or matrices, in which the posteriors are 'labelled' by their corresponding outer probability, rather than from the observation y that produced them. This is because the posterior vulnerability of a channel wrt a prior (Def. 2.2.5) does not depend on the labels y. We can think of this from the point of view of an adversary: the particular observation y only determines the distribution $p(x|y)$ which the adversary computes, and it is this distribution (posterior) that the adversary uses to make a guess.

We say that two hypers are *equal* when they consist of the same set of posteriors (in the support of the outers), and each hyper assigns equal outer probabilities to equal inners.

An important result [61] says that hyper-distributions characterise the leakage properties of a channel (for any full support prior). What this means is that, given a set of posteriors δ^i with corresponding outers a_i, we can uniquely recover the full support prior π and channel C which produced that hyper (up to equivalence on channels).[8] The prior π is recovered by averaging each of the posteriors wrt its outer probability, ie. $\pi = \sum_i a_i \delta^i$. The channel is recovered by multiplying each posterior by its outer, and then normalising by taking marginals on each $x \in \mathcal{X}$.[9] Since the channel is recovered independently of the choice of π (provided that it is full support), we can always choose π to be the uniform prior. This means that we can construct channels with desirable leakage properties by constructing the appropriate hyper-distributions, from which the desired channel can be recovered.

Another way that this result can be stated is that we can compare channels by comparing the corresponding hypers produced by the action of the uniform prior on each channel.

Lemma 2.3. [61] Let $A:\mathcal{X} \to \mathbb{D}\mathcal{Y}$, $B:\mathcal{X} \to \mathbb{D}\mathcal{Y}$ be channels and let υ be the uniform prior on $\mathcal{X}$. Then for all $g:\mathbb{G}\mathcal{X}$ and $\pi:\mathbb{D}\mathcal{X}$ we have

$$V_g[\pi \triangleright A] = V_g[\pi \triangleright B] \quad \text{iff} \quad [\upsilon \triangleright A] = [\upsilon \triangleright B]\,.$$

Lem. 2.3 is very powerful; it says that we only need to consider one particular realisation of a hyper in order to understand the leakage properties of a channel wrt *any* prior.

Even more importantly, the above result suggests that refinement of channels can be defined correspondingly over hypers such that the following holds:

Lemma 2.4. Let $A:\mathcal{X} \to \mathbb{D}\mathcal{Y}$, $B:\mathcal{X} \to \mathbb{D}\mathcal{Y}$ be channels and let υ be the uniform

[8] Equivalence on channels is defined by their leakage properties - namely posterior g-vulnerability.

[9] Note that we can write this channel down as a matrix in many possible ways (by permutation of columns) but these all have the same leakage, so we can choose any of these matrix representations for the produced channel.

prior on $\mathcal{X}$. Then

$$A \sqsubseteq B \qquad \text{iff} \qquad [\upsilon \triangleright A] \sqsubseteq [\upsilon \triangleright B] \,.$$

We have not yet said what $[\upsilon \triangleright A] \sqsubseteq [\upsilon \triangleright B]$ means; to understand how this is defined we turn to a *geometric* representation of hyper-distributions.

2.4.1 The Geometry of Hyper-Distributions

Recall from earlier that a *hyper* $\Delta : \mathbb{D}^2\mathcal{X}$ is a set of posteriors δ^y, each labelled with its probability a_y of being produced (called an *outer*). We now think of each posterior as a 1-summing vector, lying in the probability simplex of dimension $|X| - 1$[10], and therefore $\Delta = [\pi \triangleright C]$ as a set of vectors, each having weight a_y, with the property that these vectors must average (using the weights a_y) to the prior π. That is, the 'vectors' δ^y must contain the prior π (as a vector) in their convex hull, and π is the "centre of mass" of the weighted vectors. The vectors δ^i can also be thought of as barycentric coordinates situated in the probability simplex, hence we use the term *barycentric* to describe this representation. We illustrate these concepts in Figure 2.1.

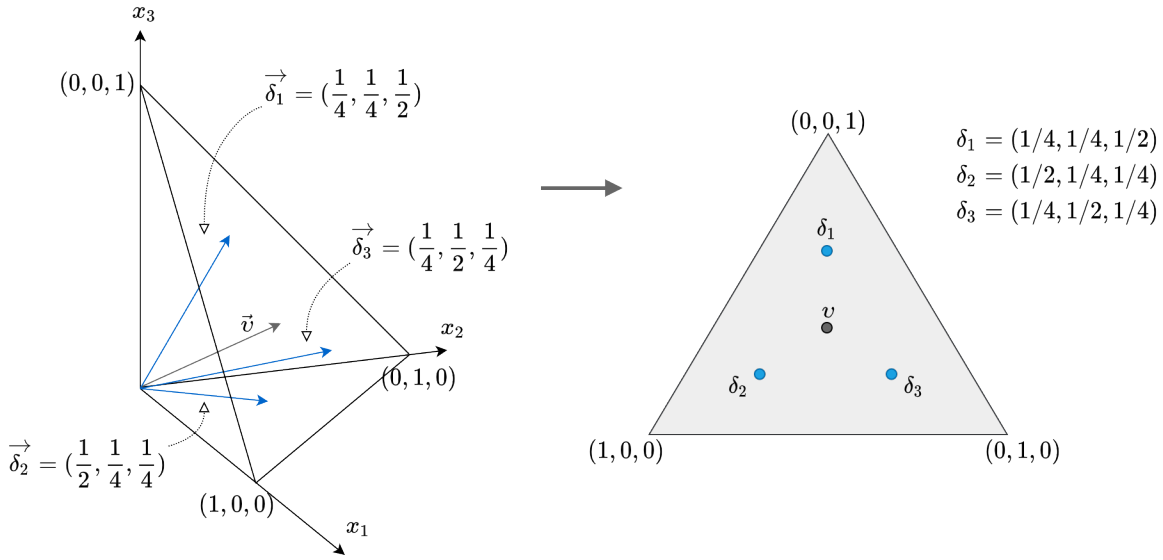


Figure 2.1: Geometric representations of the hyper $\Delta = \frac{1}{3}[\delta_1] + \frac{1}{3}[\delta_2] + \frac{1}{3}[\delta_3]$ defined on 3 secrets $\mathcal{X} = \{x_1, x_2, x_3\}$. The figure on the left shows the posterior distributions δ_i depicted as vectors which all sit on the probability simplex $x_1 + x_2 + x_3 = 1$. The uniform distribution υ is depicted by the grey vector $\vec{\upsilon}$ and can be computed as the weighted (convex) sum of the $\vec{\delta_i}$. ie. $\vec{\upsilon} = \frac{1}{3}\vec{\delta_1} + \frac{1}{3}\vec{\delta_2} + \frac{1}{3}\vec{\delta_3}$.
The figure on the right shows the vectors $\vec{\delta_i}$ as barycentric coordinates located on the triangular probability simplex. The point $\upsilon = (1/3, 1/3, 1/3)$ is contained in the convex hull of the points δ_i and is their "centre of mass" wrt the outer weights a_i.

When describing hypers using the barycentric representation, we use the terminology "points"

[10] To see why it is not the simplex of dimension $|X|$, notice that the 3-dimensional 1-summing vectors all lie on the simplex defined by $x + y + z = 1$ which is 2-dimensional (a triangular face).

to refer to posteriors, and "weights" to refer to the outer probabilities, although we retain the notation δ^i and a_i to describe them. Since these points can likewise be thought of as vectors (see left hand figure of Figure 2.1), we will also use vector space algebra in our reasoning; in particular we will use the term "averaging" to describe the weighted vector sum of posteriors (eg. to produce the prior π as in $\sum_i a_i \delta^i$). We will typically visually represent hypers as points on the projected probability simplex in 3 dimensions (see right hand figure of Figure 2.1), although the ideas are understood to extend to any dimensions. However, we will usually *not* try to depict the weights a_i visually, instead we use diagrams to visualise posteriors and we will supply the corresponding weights in descriptive text.

> **Remark 2.4.1.** Since the leakage properties of hypers are characterised by the action of the uniform distribution (Lem. 2.3), we will assume that the uniform distribution is the centre of mass of the hypers that we construct, unless otherwise stated.

The barycentric representation of hypers loses no information (in terms of leakage) and thus we can use some properties of vector spaces to reason about leakage of hypers. For example, observe that in the space of n secrets (ie. on vector space of dimension n), at most n vectors form a linearly independent set; thus any linearly independent set of (at most n) vectors which contains υ in its convex hull *uniquely* represents a hyper (since there is a unique set of weights that can be used to construct any vector from a linearly independent set of vectors).

2.4.2 Refinement of Hypers

We see how the structural definition for channel refinement (Def. 2.3.2) can be represented as a structural refinement of hypers. We first explore how a channel acts on a prior to produce a hyper-distribution.

For the hyper $[\pi \triangleright C] = \sum_i a_i [\delta^i]$, we can think of the channel C as "splitting" the prior π into posteriors δ^i such that they average ("merge") back to π. The splitting operation produces the hyper $[\pi \triangleright C]$ which is "less safe" than π, ie. $V_g[\pi \triangleright C] \geq V_g(\pi)$. We also observe that $[\pi \triangleright C]$ is less safe than the *hyper* $[\pi]$ (the point hyper on π) – namely $V_g[\pi \triangleright C] \geq V_g[\pi \triangleright \mathbf{1}]$ (where $\mathbf{1}$ is the column channel consisting of all 1's).[11] It turns out that the "splitting" operation applied to the hyper $[\pi]$ is created by the structural anti-refinement between $\mathbf{1}$ and C.

We now explain how "splitting" corresponds to structural anti-refinement on channels. Consider two channels A, B satisfying $A \sqsubseteq B$ with witness R. The hypers $[\upsilon \triangleright A]$ and $[\upsilon \triangleright B]$ are related as follows: for every posterior γ^i of $[\upsilon \triangleright B]$, we can "split" γ^i into a convex sum of the posteriors δ^j of $[\upsilon \triangleright A]$ which conversely "merge" back to γ^i. Moreover, the set of δ^j resulting from splitting all of the γ^i is the full set of posteriors of $[\upsilon \triangleright A]$, and the collection of probabilities generated by the splitting corresponds exactly to the outers of $[\upsilon \triangleright A]$. It has been shown [62] that the "splitting" operation is anti-refining, and corresponds to the action of the witness R (which provides the "merge" from the less safe to the more safe channel). We call the "merge" operation a "refining

[11] Equivalently, $\mathbf{1}$ is the channel which leaks nothing.

Earth Move" because it corresponds to the action of refinement.[12]

> **Definition 2.4.2** (Refining Earth Move)**.** Let $\Delta_1, \Delta_2 : \mathbb{D}^2\mathcal{X}$ be hypers. We say that there is a *refining Earth Move* from hyper $\Delta_1 = \sum_i a_i[\delta^i]$ to hyper $\Delta_2 = \sum_j b_i[\gamma^j]$ if there exist coefficients $\lambda_{ij} \geq 0$ for each i, j such that the following two conditions hold:
>
> 1. $\sum_j \lambda_{ij} = 1$.
> 2. $b_j \gamma^j = \sum_i \lambda_{ij} a_i \delta^i$ for each j.

The first condition says that we split each $a_i\delta^i$ up, sending the weight and vector to the various γ^j. The second condition says that the weighted proportions match the γ^j's. From these conditions we also deduce that (1) the weighted sums $\sum_j b_j\gamma^j$ and $\sum_i a_i\delta^i$ are equal, and (2) each γ^j is formed by the convex combination of the δ^i, and thus the convex hull of posteriors γ_j is contained in the convex hull of the posteriors δ^i.

We are now ready to define refinement of hypers.

> **Definition 2.4.3** (Structural Refinement of Hypers [34])**.** Let $\Delta_1, \Delta_2 : \mathbb{D}^2\mathcal{X}$ be hypers. We say that Δ_2 is a structural refinement of Δ_1, written $\Delta_1 \sqsubseteq_\circ \Delta_2$ iff there is a refining Earth Move from Δ_1 to Δ_2.

An important result [62] says that structural refinement of hypers is equivalent to structural refinement on channels. That is, we can now restate Lem. 2.4 as follows:

> **Lemma 2.5.** Let $A : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, $B : \mathcal{X} \to \mathbb{D}\mathcal{Z}$ be channels and let υ be the uniform prior on $\mathcal{X}$. Then
>
> $$A \sqsubseteq B \qquad \text{iff} \qquad [\upsilon \triangleright A] \sqsubseteq_\circ [\upsilon \triangleright B] \,.$$

We will write $\Delta_1 \sqsubseteq \Delta_2$ for hypers Δ_1, Δ_2 instead of using $\sqsubseteq_\circ$. An example of refining hypers is shown in Example 2.4.

When the posteriors of a hyper are linearly independent (as vectors) it turns out that only the convex hull property is necessary to guarantee refinement.

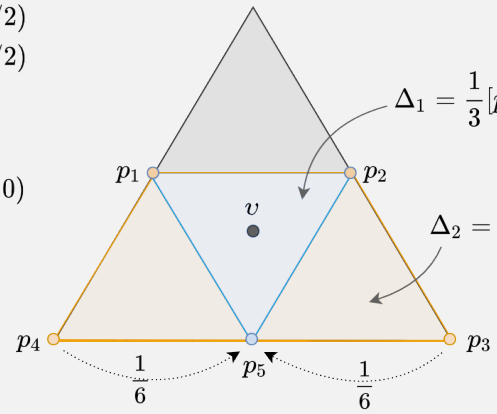
> **Lemma 2.6.** (Lemma 12.2 from [34]) Let $\Delta_1, \Delta_2 : \mathbb{D}^2\mathcal{X}$ be hypers. If the posteriors of Δ_1 are linearly independent, then $\Delta_1 \sqsubseteq \Delta_2$ whenever the posteriors of Δ_2 lie inside the convex hull of posteriors of Δ_1.

One reason that hyper-distributions are so useful for reasoning about leakage properties is that on hypers, refinement is a true partial order. This is because all equivalent channels (wrt refinement) produce the *same* hyper-distribution, and every equivalence class of channels produces a different hyper.

[12] The splitting and merging operations can be expressed much more succinctly using category theory – specifically the operations of the Giry monad.

Example 2.4 (Refining and non-Refining Hypers)**.** Consider the following hypers in the space of 3 secrets. The posteriors of Δ_1 are the orange points p_1 and p_2 and the blue point p_5. The posteriors of Δ_2 are the 4 orange points p_1, p_2, p_3, p_4. Their associated weights are given in the diagram below.

$p_1 = (1/2, 0, 1/2)$
$p_2 = (0, 1/2, 1/2)$
$p_3 = (0, 1, 0)$
$p_4 = (1, 0, 0)$
$p_5 = (1/2, 1/2, 0)$



We can check that each hyper averages to the uniform distribution, denoted by υ, contained in the convex hulls of their posteriors.

Now, the posteriors of Δ_1 are contained in the convex hull of posteriors of Δ_2. And, there is a refining Earth Move from Δ_2 to Δ_1: namely, the probability masses on p_1 and p_2 stay in place (the 'trivial' Earth Move), but the probability masses of $1/6$ on points p_3 and p_4 move to the point p_5 (shown in the diagram), for a total Earth Move of $1/3$ which corresponds to the weight of p_5 in Δ_1. Therefore we conclude $\Delta_2 \sqsubseteq \Delta_1$.

We can construct the channels C_1 and C_2 corresponding to Δ_1 and Δ_2 respectively as shown below.

$$C_1 = \begin{pmatrix} 1/2 & 0 & 1/2 \\ 0 & 1/2 & 1/2 \\ 1/2 & 1/2 & 0 \end{pmatrix} \qquad C_2 = \begin{pmatrix} 1/2 & 0 & 0 & 1/2 \\ 0 & 1/2 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 0 \end{pmatrix}$$

And indeed we find also that $C_2 \sqsubseteq C_1$ (observe that C_1 is obtained from C_2 by a post-processing step of combining the last 2 columns).

We can construct another hyper from the points p_1, p_2, p_3, p_4, namely

$$\Delta_3 = \frac{1}{6}[p_1] + \frac{1}{2}[p_2] + \frac{1}{12}[p_3] + \frac{1}{4}[p_4] .$$

Again, we can check that Δ_3 averages to the uniform distribution. However, in this case there is no refining Earth Move between Δ_1 and Δ_3, nor between Δ_2 and Δ_3. This can be seen by considering the point p_1 which has probability mass $1/3$ assigned by hypers Δ_1 and Δ_2 but only $1/6$ assigned by Δ_3. Since p_1 is not in the convex hull of any points of Δ_3, there is no Earth Move that can assign the extra $1/6$ probability mass required for $\Delta_3 \sqsubseteq \Delta_1$. The same reasoning applies for Δ_2 – in this case we cannot have $\Delta_2 \sqsubseteq \Delta_3$ nor $\Delta_3 \sqsubseteq \Delta_2$.

2.5 Chapter Notes

All of the results of this chapter can be found in the 'QIF book' by Alvim et al. [34] although they have been referenced in their original papers as much as possible. The QIF notions introduced in this chapter have been used extensively in other works as highlighted in the introduction. However, the geometric (barycentric) representation of hypers, explained in [34], has not, to our knowledge, been applied in any other works. In this thesis the barycentric representation will play a pivotal role in Chapter 6 and Chapter 7 in which we explore optimality for discrete and continuous differential privacy mechanisms.

QIF techniques have previously been applied to the study of differential privacy, most notably by Alvim et al. [32, 33] in which the authors studied the leakage properties of differential privacy systems using leakage models inspired by QIF. In [53] the idea of modelling differentially private systems as QIF channels was introduced; the authors observed that the differential privacy property can be expressed as a relationship between elements in the columns of a channel. We will introduce this model in Chapter 3 when we explore metric differential privacy, and indeed the channel model will be our basic model for differentially private mechanisms in this thesis.

3

Metric Differential Privacy

In this thesis, we adopt a natural generalisation of differential privacy to metric spaces, known variously as *metric differential privacy*, *generalised differential privacy*, or simply **d**-privacy.[1] Metric differential privacy was introduced in 2012 by Chatzikokolakis et al. [15] and can model a wide variety of scenarios in which the notion of indistinguishability can be naturally expressed using a metric between secrets. In addition, metric differential privacy can be applied in different privacy *workflows* such as oblivious mechanisms and local differential privacy. Finally, mechanisms employed in standard differential privacy and local differential privacy scenarios can be repurposed for metric differential privacy by use of an appropriate metric on secrets.

In this chapter we review metric differential privacy, based primarily on the seminal work by Chatzikokolakis et al [15]. In §3.1 we give a formal definition and describe some important properties that metric differential privacy shares with differential privacy. In §3.2 we describe geo-indistinguishability, the canonical example for metric differential privacy, for which metric differential privacy is most well-known. In §3.3 we explain how to reason about privacy and utility with respect to oblivious mechanisms and local differential privacy models, to which we give the term 'workflows'. Finally, in §3.4 we give some examples of **d**-privacy mechanisms that are commonly found in the literature and which will be used later in this thesis.

3.1 Definition and Properties

Differential privacy relies on the observation that some pairs of secrets need to be indistinguishable from the point of view of the adversary in order to provide some meaningful notion of privacy. For instance, databases which differ in a single individual should not be able to be

[1] We will use the term 'metric differential privacy' more generally, and **d**-privacy where the metric **d** is important.

distinguished based on the output of a privacy-preserving mechanism, otherwise the privacy of that individual could be violated by an adversary who knows all other values in the database. At the same time, other pairs of secrets *should* be distinguishable so that a data release can provide some utility. For instance, the ability to distinguish databases which differ in many individuals allows us to answer a statistical query about those individuals.

This idea can be formalised by a *distinguishability metric*[2] $\mathbf{d}$, which models how distinguishable we allow these secrets to be. A value of 0 means that we require x and x' to be completely indistinguishable to the adversary, while $+\infty$ means that they can be distinguished completely.

3.1.1 Technical Preliminaries

We recall that a *metric* $\mathbf{d} : \mathcal{X}\times\mathcal{X} \to \mathbb{R}_{\geq 0}$ satisfies $\mathbf{d}(x,y) = 0$ iff $x = y$, $d(x,y) = d(y,x)$ and $d(x,y) + d(y,z) \geq d(x,z)$.[3] A *metric space* $(\mathcal{X}, d)$ is a set $\mathcal{X}$ coupled with a metric $\mathbf{d}$ on $\mathcal{X}$. We denote by $\mathbb{D}\mathcal{X}$ the set of probability distributions on $\mathcal{X}$ and by $\mathbb{M}\mathcal{X}$ the set of metrics on $\mathcal{X}$. We denote by $\mathcal{F}_{\mathcal{Y}}$ a sigma algebra on the set $\mathcal{Y}$. A *mechanism* is a probabilistic mapping from inputs to (distributions on) outputs. For a mechanism $C\colon \mathcal{X} \to \mathbb{D}\mathcal{Y}$ we write $C(x)(Y)$ for the probability that the set Y is assigned by the distribution $C(x)$. When the input and output spaces are discrete, we can model a mechanism as a *probabilistic channel*. In this case, we can describe a mechanism as a channel *matrix* C whose rows $C_{x,-}$ are distributions and whose elements $C_{x,y}$ correspond to the probability of observing output y given input x. As already signalled in Chapter 2, we will abuse notation somewhat and use the same symbol to refer to both mechanisms and channel matrices. ie. We may define $C\colon \mathcal{X} \to \mathbb{D}\mathcal{Y}$ and then refer to the element $C_{x,y}$ when $\mathcal{X}$ and $\mathcal{Y}$ are discrete, with the understanding that these types are isomorphic.

3.1.2 Definition of d-Privacy

We next recall the usual formulation of differential privacy:[4]

> **Definition 3.1.1** (ε-Differential Privacy)**.** Given a space of databases $\mathcal{X}$ and given $\varepsilon \geq 0$, we say that a mechanism $K : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ satisfies ε-differential privacy if for any two adjacent databases $x, x' \in \mathcal{X}$ and any output $Y \subseteq \mathcal{Y}$, the following inequation holds:
>
> $$K(x)(Y) \leq e^{\varepsilon}\ K(x')(Y)$$
>
> where 'adjacent' means differing in one individual.

The intuition behind this definition is that datasets which are 'similar' (according to the adjacency relation) should produce similar outputs from the mechanism K. Thus, the argument is that an adversary observing some value y cannot deduce whether the input was x or x' for any $x \sim x'$. Observe that this definition is couched in terminology from statistical databases for

[2] To be precise, an *extended pseudo* metric, that is one in which distinct secrets can have distance 0, and distance $+\infty$ is allowed.

[3] We will usually require $\mathbf{d}$ to be a metric even though we can in principle reason using extended pseudo metrics.

[4] Differential privacy has an extended form (so-called (ε,δ)-differential privacy) which we will not examine in this thesis.

which differential privacy was originally designed. Notably, the idea of adjacency is tied to the presence of an individual in the dataset.

The idea of metric differential privacy is to abstract away from these specifics and, in doing so, provide new intuitions required to understand differential privacy in more general settings. We now recall the definition of $\mathbf{d}$-privacy given by Chatzikokolakis et al. [15]:

> **Definition 3.1.2 ($\mathbf{d}$-Privacy).** Given a metric space $(\mathcal{X}, \mathbf{d})$, a set $\mathcal{Y}$ and $\varepsilon \geq 0$, a mechanism $K : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ satisfies $\varepsilon{\cdot}\mathbf{d}$-privacy iff $\forall x, x' \in \mathcal{X}$,
>
> $$K(x)(Y) \;\leq\; e^{\varepsilon \cdot \mathbf{d}(x,x')} K(x')(Y) \qquad \text{for all } Y \in \mathcal{F}_{\mathcal{Y}} \;.$$

Using a generic metric $\mathbf{d}$ in this definition allows us to express different scenarios, depending on the domain $\mathcal{X}$ to which the mechanism is applied. For instance, in the standard model of differential privacy, the mechanism is applied to a database x and produces some observation y (eg. a number, the result of a statistical query). Standard ε-differential privacy can then be captured as $\varepsilon{\cdot}\mathbf{d}_{\mathrm{H}}$-privacy where $\mathbf{d}_{\mathrm{H}}$ is the Hamming distance on datasets.[5] The metric $\mathbf{d}$ permits extending the notion of 'adjacency' (Def. 3.1.1) from a discrete notion to a continuous notion (eg. when the set $\mathcal{X}$ is uncountable).

Intuitively we think of $\mathbf{d}$ as the 'type' and ε as the 'amount' of privacy. In other words, the metric $\mathbf{d}$ determines how privacy is to be interpreted, and the parameter ε quantifies 'how much' privacy is afforded. Smaller values of ε correspond with 'more' privacy; the intuition for this is that the smaller ε is, the 'closer' the distributions $K(x)$ and $K(x')$ become (for fixed $\mathbf{d}$), and thus x and x' are harder to distinguish on any output.

Observe however that the ε parameter in Def. 3.1.2 is not required to be *minimal* – that is, if M satisfies $2{\cdot}\mathbf{d}$-privacy (for $\varepsilon = 2$), we could also say that it satisfies $4{\cdot}\mathbf{d}$-privacy, or $100{\cdot}\mathbf{d}$-privacy. This would not be desirable but it may sometimes be *unavoidable* (as occurs, for example, in the construction of so-called 'exponential mechanisms' that we will encounter in Chapter 4).

Common metrics of interest which we will encounter in $\mathbf{d}$-privacy are the Euclidean metric ($\mathbf{d}_2$), the Discrete metric ($\mathbf{d}_{\mathrm{D}}$) and the Hamming metric ($\mathbf{d}_{\mathrm{H}}$).

> **Remark 3.1.1.** Note that we often omit the ε and refer simply to $\mathbf{d}$-privacy – in this case we think of ε as being 'absorbed' into the metric $\mathbf{d}$. (Since $\varepsilon{\cdot}\mathbf{d}$ is itself a metric, both definitions are equivalent.) We will explicitly use ε when it is important, for example when comparing different privacy levels wrt the same metric. In commentary we may also use the term $\mathbf{d}$-privacy in place of the more arduous 'metric differential privacy'.

When $\mathcal{X}$ and $\mathcal{Y}$ are countable sets, the mechanism K can be modelled as a *probabilistic channel* [53] which we model as a matrix (see §3.1.1). Seen this way, Def. 3.1.2 imposes a structural constraint on each column of the channel matrix, namely that the ratio of any two elements $K_{x,y}$, $K_{x',y}$ is bounded by $e^{\varepsilon \mathbf{d}(x,x')}$. We can therefore equivalently define $\mathbf{d}$-privacy as

[5] The *Hamming* metric $\mathbf{d}_{\mathrm{H}}$ on datasets is defined as the number of entries in which rows x and x' differ.

follows:

Definition 3.1.3. Given $\varepsilon \geq 0$, countable sets $\mathcal{X}$, $\mathcal{Y}$ and a metric $\mathbf{d}\colon \mathbb{M}\mathcal{X}$, we say that a channel $K : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ satisfies $\varepsilon\cdot\mathbf{d}$-privacy iff

$$K_{x,y} \leq e^{\varepsilon\cdot\mathbf{d}(x,x')} K_{x',y} \qquad \text{for all } x, x' \in \mathcal{X}, y \in \mathcal{Y} \ .$$

Example 3.1 illustrates some $\mathbf{d}$-private channels.

Example 3.1. Some $\mathbf{d}$-private channels defined on the input space $X = \{0, 1, 2\}$.

G	y_0	y_1	y_2
0	2/3	1/6	1/6
1	1/3	1/3	1/3
2	1/6	1/6	2/3

R	y_0	y_1	y_2
0	3/5	1/5	1/5
1	1/5	3/5	1/5
2	1/5	1/5	3/5

C	y_0	y_1
0	3/5	2/5
1	1/2	1/2
2	1/4	3/4

D	y_0	y_1	y_2
0	1/4	1/4	1/2
1	1/4	1/4	1/2
2	1/2	1/2	0

G is $(\ln 2\cdot\mathbf{d}_2)$-private and the privacy constraints hold 'tightly' on all elements – ie. Either $G_{x,y} = e^{\ln 2\cdot\mathbf{d}_2(x,x')}\, G_{x',y}$ or $G_{x',y} = e^{\ln 2\cdot\mathbf{d}_2(x,x')}\, G_{x,y}$ for all $x, x' \in X$.
R is $(\ln 3\cdot\mathbf{d}_2)$-private and also $(\ln 3\cdot\mathbf{d}_{\mathrm{D}})$-private where recall that $\mathbf{d}_{\mathrm{D}}$ is the discrete metric satisfying $d(x, y) = 1$ whenever $x \neq y$.
C is $(\ln 2\cdot\mathbf{d}_2)$-private because the maximum ratio of elements in any column occurs between C_{1,y_0} and C_{2,y_0}.
D is $(\infty\cdot\mathbf{d})$-private on every $\mathbf{d}$ since it contains a 0 element. We would usually say that D is *not* a differentially private channel.

Observe that a channel can be $\varepsilon\cdot\mathbf{d}$-private for various metrics $\mathbf{d}$ (and various ε). The metric chosen determines how privacy is to be interpreted for that channel, as we will see later in this chapter.

3.1.3 Properties of Metric Differential Privacy

Differential privacy has some important properties which make it both robust and appealing: it is both *compositional* and *post-processing invariant*, which allows us to reason about the privacy of systems which are put together with differentially private components; and its privacy guarantee is (somewhat) independent of the prior knowledge of an attacker, which makes it robust against all sorts of adversaries, regardless of any background information they may possess [5, 7].

Metric differential differential privacy also enjoys these properties, which we now examine in more detail.

Compositionality

Compositionality tells us that mechanisms combine in predictable ways, and thus we can safely reason about privacy against an adversary who may combine his knowledge from different sources. We can also safely reason about the privacy of a whole system by examining the privacy

of its individual components. Compositionality is therefore an important property – it reassures us that there can be no unexpected breakdown of privacy, regardless of how particular mechanisms are implemented or accessed, or how they are put together in complex workflows.

The basic composition results for differential privacy refer to what is known as *sequential composition*.[6] Sequential composition models the way in which observations from different mechanisms operating on the same data can be combined.

Definition 3.1.4 (Sequential Composition)**.** Let $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}_1$, $D : \mathcal{X} \to \mathbb{D}\mathcal{Y}_2$ be channels. We define the *sequential composition* $C;D : \mathcal{X} \to \mathbb{D}(\mathcal{Y}_1 \times \mathcal{Y}_2)$ as

$$(C;D)_{x,(y_1,y_2)} \quad := \quad C_{x,y_1} \times D_{x,y_2} \ .$$

The following well-known result shows how $\mathbf{d}$-privacy behaves wrt sequential composition.

Lemma 3.1. Let $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}_1$, $D : \mathcal{X} \to \mathbb{D}\mathcal{Y}_2$ be mechanisms and which are $\varepsilon_1 \cdot \mathbf{d}$-private and $\varepsilon_2 \cdot \mathbf{d}$-private respectively. Then their sequential composition $(C;D)$ is $(\varepsilon_1 + \varepsilon_2) \cdot \mathbf{d}$-private.

This means that each access to a mechanism increases an attacker's knowledge by ε which we can interpret as 'eroding' the privacy of the overall system. This motivates the term 'privacy budget' which is often used to describe the parameter ε.[7]

Note that we can also compose mechanisms defined over different metrics.

Lemma 3.2. If C, D from Def. 3.1.4 are $\varepsilon_1 \cdot \mathbf{d}_1$-private and $\varepsilon_2 \cdot \mathbf{d}_2$-private respectively, then their sequential composition $(C;D)$ is $(\varepsilon_1 \cdot \mathbf{d}_1 + \varepsilon_2 \cdot \mathbf{d}_2)$-private.

Since $(\varepsilon_1 \cdot \mathbf{d}_1 + \varepsilon_2 \cdot \mathbf{d}_2)$ also describes a metric, we can conclude that the sequential composition of $\mathbf{d}$-private mechanisms is always $\mathbf{d}$-private for some metric $\mathbf{d}$.

Remark 3.1.2. Another type of composition, *adaptive composition*, models an adversary who is allowed to choose inputs to the mechanism at each stage after observing the previous outputs. This type of composition has been extensively studied [7, 63, 64] in the context of (ε, δ)-differential privacy, where composition bounds are non-trivial to calculate. However, in standard differential privacy (and in metric differential privacy), the guarantee (wrt sequential composition) is not affected by whether the adversary chooses adaptively or not.

[6] In QIF this is called parallel composition, which unfortunately has a different meaning in differential privacy. Here we will remain with the differential privacy language.

[7] In practice, it is left to the designer of a system to decide how much access can be permitted to a system before the 'privacy budget' is completed eroded, and the overall ε guarantee becomes worthless.

Post-Processing Invariance

An important property of secure systems, as we saw in Chapter 2, is post-processing invariance, which says that the information leakage of a system cannot *increase* as the result of a post-processing step. In privacy, post-processing invariance tells us that the privacy risk to a system cannot increase as the result of post-processing the output from a privacy mechanism. Post-processing invariance is an example of the data-processing inequality of information theory, and it is crucial in allowing us to reason about the privacy or security of systems which are put together sequentially.

For metric differential privacy, the data processing inequality says the following:

> **Lemma 3.3** (DPI for **d**-Privacy)**.** Let $M : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be an $\varepsilon \cdot \mathbf{d}$-private channel and let $R : \mathcal{Y} \to \mathbb{D}\mathcal{Z}$ be a post-processing step modelled as a channel. Then the composed channel MR is also $\varepsilon \cdot \mathbf{d}$-private, where we write MR for the matrix multiplication of M by R.
>
> *Proof.* Follows from post-processing invariance of differential privacy [7] for any **d**. □

We noted in Chapter 2 that composition of channels corresponds to a post-processing step, also neatly corresponding with matrix multiplication of channel matrices. Lem. 3.3 then tells us that post-processing cannot 'erode' privacy, or in other words, the value of ε of the composed channel cannot increase.

Robustness to Prior Knowledge

The final property we will examine is usually referred to either as 'robustness to the prior knowledge of an attacker', or 'resistance to arbitrary side information' [65]. In the context of differential privacy, it is usually informally conveyed as one of the following two properties:

1. Regardless of external knowledge, an attacker draws the same conclusions whether or not an individual was present in the dataset [66, 67]; OR
2. An attacker who knows everyone except one person u in the dataset gains no information from the reported answer, regardless of side knowledge about u's data. [15]

These ideas have been extended to metric differential privacy using Bayesian formulations [14, 15], which we rewrite now using the language of QIF. In the following, given some hyper-distribution Δ, we write δ^y for the posterior of Δ associated with the observation y, and δ^y_x for the value that the distribution δ^y assigns to secret x.[8] Given a distribution $\pi : \mathbb{D}\mathcal{X}$ we write $\lceil \pi \rceil$ for the support of the distribution (ie. the set of x st. π_x is non-zero). We remark that our use of the channel model here assumes that the input and output domains are discrete, however the below properties have been shown to hold for continuous domains.

The first property can be reinterpreted using the following lemma.

[8] In standard probability notation these correspond to $P(X|Y = y)$ and $P(X = x|Y = y)$ respectively.

LEMMA 3.4 (Geo-Indistinguishability-I from [14]). A channel C satisfies $\mathbf{d}$-privacy iff for all priors $\pi : \mathbb{D}\mathcal{X}$ and all observations $y \in \mathcal{Y}$:

$$\frac{\delta^y_x}{\delta^y_{x'}} \;\leq\; e^{\varepsilon \cdot \mathbf{d}(x,x')} \, \frac{\pi_x}{\pi_{x'}}$$

for any $x, x' \in \lceil \pi \rceil$, where δ^y is the posterior for observation y in the hyper $[\pi \triangleright C]$.

Remembering that δ^y_x represents the adversary's posterior beliefs about the secret x given the observation y, what Lem. 3.4 tells us is that the channel C changes the distinguishability of the secrets x, x' by at most a factor of $\varepsilon \cdot \mathbf{d}(x, x')$, regardless of the prior uncertainty of an attacker. In other words, the adversary cannot distinguish the secrets much more after the observation y than he could beforehand (using his prior knowledge).

We now reinterpret the second property using the following:

LEMMA 3.5 (Geo-Indistinguishability-II from [14]). Let $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be a channel, let $x \in \mathcal{X}$, and let $\pi : \mathbb{D}\mathcal{X}$ be a prior such that $x \in \lceil \pi \rceil$ and for all $x' \in \lceil \pi \rceil$ it holds that $\mathbf{d}(x, x') \leq r$ for some constant $r > 0$. Then C satisfies $\mathbf{d}$-privacy iff for all observations $y \in \mathcal{Y}$:

$$\frac{\delta^y_x}{\pi_x} \;\leq\; e^{\varepsilon \cdot r}$$

where, again we write δ^y for the posterior corresponding to observation y in the hyper $[\pi \triangleright C]$.

Lem. 3.5 says that an adversary who knows only that the secret lies within a ball of radius r from x does not learn much more about whether the secret's value is x from observing the output y.

Notice that we can recover the (informal) properties of differential privacy by substituting $\mathbf{d} = \mathbf{d}_{\mathrm{H}}$ where $\mathbf{d}_{\mathrm{H}}$ is the Hamming distance on datasets. Lem. 3.4 then says that the posterior beliefs of the adversary are changed by a factor of at most e^{ε} whether an individual is in the dataset or not (corresponding to $\mathbf{d}_{\mathrm{H}}(x, x') = 1$). Lem. 3.5 says that the posterior beliefs of an attacker who knows everyone except 1 person in the dataset change by at most e^{ε} (again corresponding to $\mathbf{d}_{\mathrm{H}}(x, x') = 1$).

The above formalisations also allow us to reinterpret these properties in new contexts and with respect to arbitrary metrics for indistinguishability. We now turn to the canonical example from $\mathbf{d}$-privacy to see how these interpretations can be applied in a practical scenario.

3.2 Geo-Indistinguishability - A Canonical Example

The motivating example for which metric differential privacy was introduced is *geo-location privacy*, also known as *geo-indistinguishability* [14]. It is described by the following scenario:

Example 3.2.1 (Geo-location Privacy). *A population of users wishes to send their geo-location co-ordinates to a data provider who responds with recommendations for nearby restaurants for*

each user. The users do not wish to reveal their precise location to the data provider but still want to receive relevant recommendations. How can we apply differential privacy so that individuals locations are kept secret while still providing useful information?

In geo-location privacy, the privacy goal for the user is to ensure that her co-ordinates cannot be identified – thus, her secret is her *current location*. The utility goal for the user is to receive useful recommendations based on her location, therefore the data provider must be able to deduce her *approximate location*. Hence, the goal of the differential privacy mechanism is to release the user's location data so that it cannot be learned precisely but can be learned approximately.

We can apply metric differential privacy by choosing an appropriate metric which permits the user to achieve both her privacy and utility goals. A natural metric for privacy in this instance is the Euclidean distance on the 2D plane of geo-location co-ordinates. This means that the privacy mechanism $M : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ will satisfy the inequation:

$$M(x)(Y) \;\leq\; e^{\mathbf{d}_2(x,x')} M(x')(Y)$$

for all $x, x' \in \mathcal{X}$ and all $Y \subseteq \mathcal{Y}$.

This says that points which are close in Euclidean distance will be more indistinguishable to an adversary, whereas points which are far apart will be more easily distinguished. More specifically, from Lem. 3.4 we can say that if $\mathbf{d}_2(x, x')$ is small, then an adversary whose prior knowledge suggests x and x' are equally possible, does not increase his ability to distinguish x and x' after observing Y. In practical terms, this means that if the user's goal is to hide her location from other nearby locations, then this mechanism will meet her privacy needs.

Lem. 3.5 can be interpreted as saying that if the adversary has knowledge of the user's whereabouts within some radius r, then after observing Y, the adversary's posterior knowledge is only increased by at most a factor of e^r.[9] The bigger r is, the more that the adversary learns – in other words, the more uncertainty the adversary has in the first place, the more the adversary gains in knowledge. However, if the adversary already knows a lot of information about the user's position (and therefore r is small), then he does not gain much more information from the output of the mechanism M.

Notice that the mechanism M will not protect the user's privacy if her goal is to make her location indistinguishable from *far away* locations. For example, an adversary who knows that the user is either in Paris, Los Angeles, or Melbourne but does not know which city, will learn her location with high likelihood even though her precise whereabouts in the city will be concealed. To achieve this alternative privacy goal, the user should select a metric such as the discrete metric $\mathbf{d}_{\mathrm{D}}$, under which all distinct points have distance 1 (and are therefore 'adjacent'). This means that the privacy mechanism would cause *all* points to become 'relatively' indistinguishable to an adversary (relative, that is, to the adversary's prior on secrets). Such a mechanism has a clear impact on the *utility* goal for the user – who, remember, still wishes to receive recommendations

[9] Or $e^{\varepsilon \cdot r}$ if we choose some ε. We ignore the ε values here for now, focussing instead on the *type* of privacy afforded by the particular metric.

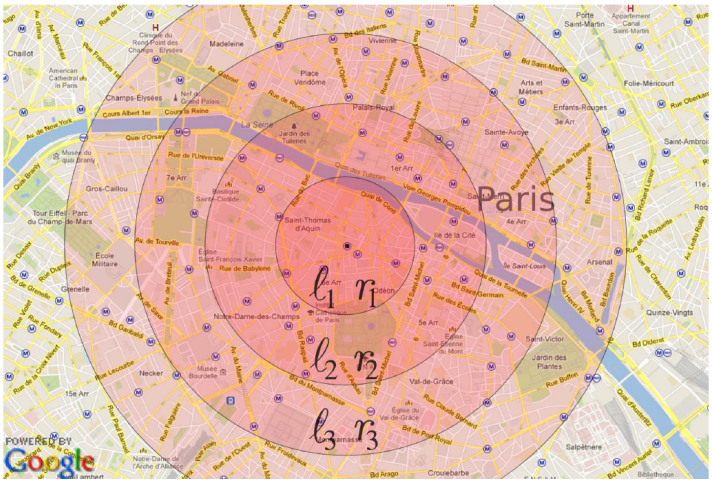


Figure 3.1: Geo-location privacy under the Euclidean metric. Points which are nearby are more indistinguishable whereas points which are further away are more distinguishable to an adversary.

based on her location – which we discuss further below.

3.2.1 Utility Goals

The utility goal in geo-location privacy is for the user's *approximate* location to be determined, so that the user can obtain (with high likelihood) useful recommendations based on her location. Typically the user's utility goals can be met if she specifies a radius of points within which she would like to receive recommendations based on her 'radius of indistinguishability'. She can then filter the most relevant recommendations sent to her based on her true location. Using a Euclidean distance metric for the privacy mechanism ensures that the user can set a radius with respect to this same metric, which guarantees that all useful recommendations are (with high likelihood) within a satisfactory radius of her true location.

However, as discussed above, if the user chooses, say, the discrete metric for her privacy mechanism (because she wants the locations Paris and Melbourne to be indistinguishable), then she is just as likely to receive (useless) distant recommendations as she is to receive nearby recommendations. For example, she is as likely to receive recommendations close to the Champs-Élysées as she is to receive recommendations for downtown Melbourne, since all points at any (Euclidean) radius around her become indistinguishable (under the discrete metric). Thus this mechanism would have very poor utility for this user. We can see how, in this scenario, the metric used for privacy has a direct impact on the utility goal; and in fact the metric for privacy should also reflect the utility goal if it is to be useful to the end user.

3.3 Reasoning with Metrics

In this section we show how to reason about privacy and utility for different scenarios commonly encountered in the differential privacy literature, which we think of as *workflows*. Workflows

describe the ways in which differential privacy mechanisms are put together, and importantly, where the 'noise-adding' component(s) of the workflows are situated. Typically each workflow corresponds to a particular 'model' for privacy, depending on whether the data collector is *trusted* or *untrusted*.

1. In the **standard** or **central model** for differential privacy, a *trusted* data curator collects data on individuals and then releases the result of a query on the data in a privacy-preserving manner. In this scenario it is typical to employ an *oblivious mechanism*, which is a workflow in which the noise-adding mechanism is applied to the result of a query to produce a noisy output.

2. In the **local model** for differential privacy, individuals apply privacy to their data locally before uploading it to an *untrusted* data curator, who then chooses the manner of releasing information about the data. This corresponds to a *local mechanism*, in which each user 'locally' applies a noise-adding mechanism (ie. to their own data) before sending it to the data curator for post-processing.

In this chapter we investigate how to reason about privacy and utility for different workflows. We focus the two most common workflows mentioned above: *oblivious mechanisms* and *local differential privacy*. [10] In the following sections, we describe oblivious mechanisms and local differential privacy in terms of metric differential privacy, which (as has been noted previously [15]) can be applied in both contexts.

3.3.1 Oblivious Mechanisms

Oblivious mechanisms first arose in the original context of differential privacy as a means of preserving the privacy of individuals in statistical datasets when the result of a query f on the dataset might reveal sensitive information about them. The idea is that a query f such as 'How many individuals have cancer?' can be performed on the dataset, and the result of the query, $f(X)$, is passed to a mechanism H to generate a noisy observation z. The mechanism H is constructed in such a way that the composition $H \circ f$ protects the privacy of individuals in the dataset (in the sense of Def. 3.1.1) whilst maintaining the usefulness (to a data analyst) of the output z. The mechanism $H \circ f$ is described as 'oblivious' because the output from mechanism H depends only on the output of $f(X)$ and does not depend on X.

Although oblivious mechanisms are usually described in the aforementioned context of statistical datasets, the ideas are general and can be applied in any context in which we have the sequential composition of a function f with a mechanism H. We formalise the above ideas as follows.

An *oblivious mechanism* describes a workflow which can be decomposed into a (deterministic) query and a privacy mechanism in the following way:

[10] A third model, called 'hybrid', describes arbitrary combinations of the above two, and is usually used to improve the utility of the overall release mechanism.

Definition 3.3.1 (Oblivious Mechanism)**.** Given a metric $\mathbf{d}_{\mathcal{X}}$ on $\mathcal{X}$, an oblivious mechanism $K : \mathcal{X} \to \mathbb{D}\mathcal{Z}$ is a $\mathbf{d}_{\mathcal{X}}$-private mechanism which can be decomposed into a query $f : \mathcal{X} \to \mathcal{Y}$ and a probabilistic mechanism $H : \mathcal{Y} \to \mathbb{D}\mathcal{Z}$ st. $K = H \circ f$. That is, the mechanism H depends only on the inputs $\mathcal{Y}$ and not on the original dataset $\mathcal{X}$.

The $\mathbf{d}$-privacy of the overall system is determined by the composition $H \circ f$, whereas the utility of the system depends on H alone, since utility is a measure of how much the output z reveals and the true query answer $f(x)$. In summary, we reason about *privacy* on the composed mechanism K but we reason about *utility* on the mechanism H. This relationship is depicted in Figure 3.2.

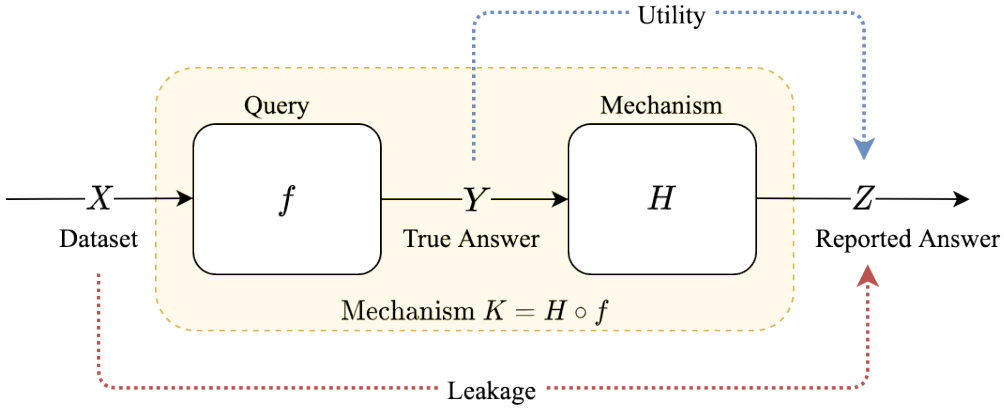


Figure 3.2: Oblivious $\mathbf{d}_{\mathcal{X}}$-private mechanisms K are composed of a deterministic function f and a $\mathbf{d}_{\mathcal{Y}}$-private mechanism H. We reason about privacy using the composition $K = H \circ f$ but we reason about utility using H alone.

In standard differential privacy, we reason about privacy wrt the Hamming metric $\mathbf{d}_{\mathrm{H}}$ since it corresponds to the privacy requirement for indistinguishability of individuals within datasets. Therefore the metric $\mathbf{d}_{\mathrm{H}}$ is imposed on the input space $\mathcal{X}$ for the mechanism K. The goal is then to construct a mechanism H so that the composition $H \circ f$ is $\mathbf{d}_{\mathrm{H}}$-private. This is typically done by first computing the 'sensitivity' of the function f (described below), and then deducing an appropriate mechanism H. Some well-known mechanisms for particular functions f include the *geometric* mechanism which can be applied to so-called 'counting' queries, and the *randomised response* mechanism which can be applied to 'sum' queries. These mechanisms will be described in §3.4.

These ideas can be extended to metric differential privacy. We begin by defining the sensitivity of the function f as follows:

Definition 3.3.2. Let $\mathbf{d}_{\mathcal{X}}, \mathbf{d}_{\mathcal{Y}}$ be metrics on $\mathcal{X}, \mathcal{Y}$ respectively and let $f : \mathcal{X} \to \mathcal{Y}$. We say that f is $\Delta^{f}_{\mathbf{d}_{\mathcal{X}},\mathbf{d}_{\mathcal{Y}}}$-sensitive wrt $\mathbf{d}_{\mathcal{X}}$, $\mathbf{d}_{\mathcal{Y}}$ if

$$\mathbf{d}_{\mathcal{Y}}(f(x), f(x')) \quad \leq \quad \Delta^{f}_{\mathbf{d}_{\mathcal{X}},\mathbf{d}_{\mathcal{Y}}} \mathbf{d}_{\mathcal{X}}(x, x')$$

for all $x, x' \in \mathcal{X}$. The smallest such $\Delta^{f}_{\mathbf{d}_{\mathcal{X}},\mathbf{d}_{\mathcal{Y}}}$ (if it exists) is called the sensitivity of

f wrt $\mathbf{d}_{\mathcal{X}}, \mathbf{d}_{\mathcal{Y}}$.

In other words, f is $\Delta^{f}_{\mathbf{d}_{\mathcal{X}},\mathbf{d}_{\mathcal{Y}}}$-sensitive iff f is $\Delta^{f}_{\mathbf{d}_{\mathcal{X}},\mathbf{d}_{\mathcal{Y}}}$-Lipschitz wrt $\mathbf{d}_{\mathcal{X}}, \mathbf{d}_{\mathcal{Y}}$. The sensitivity of f can be used to reason about privacy for oblivious mechanisms as follows:

Lemma 3.6 (Fact 5 in [15]). Let $\mathbf{d}_{\mathcal{X}}, \mathbf{d}_{\mathcal{Y}}$ be metrics on $\mathcal{X}, \mathcal{Y}$ respectively and let $f : \mathcal{X} \to \mathcal{Y}$ be $\Delta^{f}_{\mathbf{d}_{\mathcal{X}},\mathbf{d}_{\mathcal{Y}}}$-sensitive. If $H : \mathcal{Y} \to \mathbb{D}\mathcal{Z}$ satisfies $\mathbf{d}_{\mathcal{Y}}$-privacy then $H \circ f$ satisfies $\Delta^{f}_{\mathbf{d}_{\mathcal{X}},\mathbf{d}_{\mathcal{Y}}} \cdot \mathbf{d}_{\mathcal{X}}$-privacy.

Note that the converse to Lem. 3.6 does not hold in general. This means that, although we can construct mechanisms H and reason about the privacy induced by f on the composition K, we *cannot* (always) take a $\mathbf{d}_{\mathcal{X}}$-private mechanism K and a $\Delta^{f}_{\mathbf{d}_{\mathcal{X}},\mathbf{d}_{\mathcal{Y}}}$-sensitive function f and deduce that H *must* satisfy $\mathbf{d}_{\mathcal{Y}}$-privacy (although it may be *sufficient* for H to be $\mathbf{d}_{\mathcal{Y}}$-private).

However, in [15] it was shown that the converse does hold under stronger conditions on f, which we rewrite here equivalently in terms of the *metric induced by the graph of f*.

Definition 3.3.3. Let $\mathcal{X}, \mathcal{Y}$ be discrete and let $\mathbf{d}_{\mathcal{X}}$ be a metric on $\mathcal{X}$. Given a surjective function $f : \mathcal{X} \to \mathcal{Y}$, define the *graph of f* as follows: treat each $y \in \mathcal{Y}$ as a node and for each $x, x' \in \mathcal{X}$ draw an edge between $f(x)$ and $f(x')$ with weight $\mathbf{d}_{\mathcal{X}}(x, x')$. Define $\mathbf{d}_{\mathcal{Y}}(y, y')$ as the minimum path weight from y to y'. Then $\mathbf{d}_{\mathcal{Y}}$ is called the metric induced by the graph of f.

The following now holds:

Lemma 3.7. Let $f : \mathcal{X} \to \mathcal{Y}$ and $\mathbf{d}_{\mathcal{X}}, \mathbf{d}_{\mathcal{Y}}$ metrics on $\mathcal{X}, \mathcal{Y}$ such that $\mathbf{d}_{\mathcal{Y}}$ is the metric induced by the graph of f. Then f is 1-sensitive wrt $\mathbf{d}_{\mathcal{X}}, \mathbf{d}_{\mathcal{Y}}$ and $H : \mathcal{Y} \to \mathbb{D}\mathcal{Z}$ satisfies $\mathbf{d}_{\mathcal{Y}}$-privacy if and only if $H \circ f$ satisfies $\mathbf{d}_{\mathcal{X}}$-privacy.

Lem. 3.7 provides important conditions under which it is safe to reason about the utility of $\mathbf{d}$-private mechanisms H in order to guarantee $\mathbf{d}_{\mathcal{X}}$-privacy on $H \circ f$.

3.3.2 Local Differential Privacy

Local differential privacy was first used by Kasiviswanathan et al. [68] to describe the behaviour of the randomised response algorithm, which adds noise to each individual's datum rather than adding noise to the output of a query. It is now also commonly associated with metric differential privacy, and in particular its application to geo-indistinguishability described earlier (§3.2).

A *local* differential privacy mechanism describes a workflow in which the privacy mechanism is directly applied to data before it is combined and released. We will describe this as follows:

Definition 3.3.4 (Local Mechanism). Given a metric $\mathbf{d}_{\mathcal{X}}$ on $\mathcal{X}$, a local mechanism $H : \mathcal{X} \to \mathbb{D}\mathcal{Z}$ is a $\mathbf{d}_{\mathcal{X}}$-private mechanism which releases a noisy output $z \in \mathcal{X}$.

We do not include any post-processing steps as part of this workflow, as these steps are not

typically included in this model (cfr. the case of geo-location privacy). Post-processing steps, if they exist, affect the way in which we reason about *utility*. In the case of geo-location privacy, since no post-processing steps are applied, privacy and utility are reasoned about wrt the mechanism H alone. However, in situations where data is being collected by an untrusted curator, as depicted in Figure 3.3, privacy and utility are reasoned about using the entire workflow.

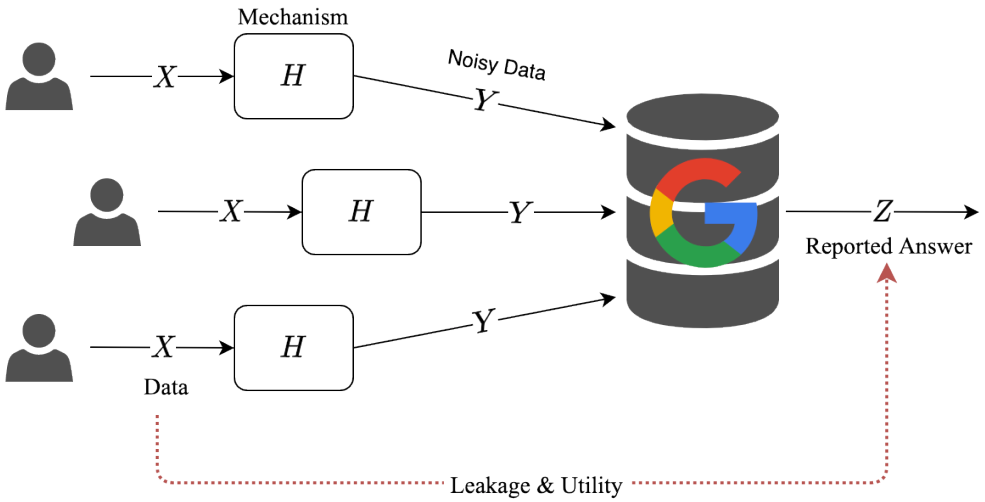


Figure 3.3: Local $\mathbf{d}_\mathcal{X}$-private mechanisms H are applied directly to individuals' data points before being sent to the untrusted data curator. In this case we reason about privacy and utility in the same way, either using the mechanism H directly (if the data curator releases each noisy data point as is) or using the composition of the mechanisms H followed by a post-processing step f if the curator releases the result of a query f on the noisy data.

Since differential privacy is post-processing invariant, it is usual to compute the privacy of the overall workflow (including any post-processing steps) just using the mechanism H. ie. if H is $\varepsilon{\cdot}\mathbf{d}$-private then local differential privacy workflows provide $\varepsilon{\cdot}\mathbf{d}$-privacy regardless of post-processing.

However, it is much more difficult to reason about *utility* of the overall system since it depends on the utility measure of interest and whether post-processing is applied to the system. It is widely accepted that utility for local differential privacy mechanisms is worse than it is for oblivious mechanisms, however there is no established theory for formally reasoning about the utility of local mechanisms or privacy workflows in general.

3.3.3 Reasoning about Privacy and Utility

When reasoning about privacy using standard differential privacy, it is typical to compare mechanisms by their ε values, so that we would say mechanism A is 'better' (more private) than mechanism B if A's ε value is smaller than B's. In metric differential privacy, this comparison can be complicated by the use of different metrics for A and B.[11] It is easy to show that

[11] Note that this argument could also be applied to standard differential privacy, since comparing mechanisms by their ε values alone does not take into account differences in the adjacency relation on the input space $\mathcal{X}$.

comparable metrics gives rise to comparable privacy guarantees:

> **LEMMA 3.8** (Proposition 1 in [15]). Let $\mathbf{d}_x$, $\mathbf{d}_y$ be metrics on $\mathcal{X}$. If $\mathbf{d}_x \leq \mathbf{d}_y$ (pointwise) then $\mathbf{d}_x$-privacy implies $\mathbf{d}_y$-privacy.

However, even when the metrics on A and B are comparable and their ε values are the same, it may be that A and B provide different levels of protection against different types of adversaries, since the ε value does not provide a guarantee against all possible adversaries. We explore this idea further in Chapter 4.

When reasoning about *utility* for differential privacy, there is currently no standard for assessing the utility of a mechanism or comparing the utility of different mechanisms. While it is generally assumed that decreasing the value of ε on a particular mechanism (ie. increasing privacy) results in an associated decrease in utility, this has not been shown formally.[12] Utility for differential privacy systems is typically evaluated empirically based on a utility measure of interest. In this thesis we will look at how to model utility for privacy workflows, how we can compare systems with respect to utility and how we can design mechanisms for optimal utility.

3.4 Constructing d-Private Mechanisms

In general, mechanisms designed for use in differential privacy can be applied in metric differential privacy for an appropriate metric on the domain. An early principle in the design of differentially private *oblivious* mechanisms was that the addition of noise should be independent of the value of secret input x: in other words, privacy is realised by a noise-adding mechanism which maps inputs x to outputs $x + K(\Delta, \varepsilon)$ where K is a probability density function which depends on the sensitivity of f (ie. Δ) and the privacy parameter ε. This general principle is also applicable in the case of metric differential privacy where, in the case of oblivious mechanisms, the noise added depends on the sensitivity of f wrt the metric on $\mathcal{Y}$; and in the case of local mechanisms, the noise added depends on the radius of indistinguishability.

Certain mechanisms are usually preferred for their simplicity of implementation, and these have become 'canonical' examples of privacy mechanisms. We introduce these here now, as they will also act as canonical examples throughout this thesis.

We write the following mechanisms as *channels* since their domains are discrete.

The geometric mechanism was developed for so-called 'counting queries' and is typically applied to integer domains, although it can be easily applied to any set $\{qk : k \in \mathbb{Z}\}$ for fixed $q \in \mathbb{R}$.

> **DEFINITION 3.4.1** (Geometric Mechanism). The α-geometric mechanism $G: \mathcal{X} \to \mathbb{DZ}$ has the following channel matrix:

$$G_{x,y} = \frac{1-\alpha}{1+\alpha} \cdot \alpha^{\mathbf{d}_2(x,y)}$$

[12] To see why this is not obviously true, consider a mechanism which outputs random noise regardless of its input. Since this has no utility whatsoever, manipulating epsilon has no effect – and certainly cannot decrease – its utility.

where $\alpha \in (0,1]$. This mechanism satisfies $\varepsilon{\cdot}\mathbf{d}_2$-privacy where $\mathbf{d}_2$ is the Euclidean metric and $\varepsilon = -\ln \alpha$.

It is often more desirable to apply a 'truncated' version of the geometric mechanism, where the truncation is applied by adding columns of the channel until the input and output domains match.

Definition 3.4.2 (Truncated Geometric Mechanism)**.** The truncated α-geometric mechanism $TG\colon \mathcal{X} \to \mathbb{D}\mathcal{X}$ where $\mathcal{X} = \{0, 1, \ldots, n\}$ has the following channel matrix:

$$TG_{x,y} = \frac{1-\alpha}{1+\alpha} \cdot \alpha^{\mathbf{d}_2(x,y)} \qquad \text{for } y \in \{1, \ldots, n-1\}$$
$$TG_{x,y} = \frac{1}{1+\alpha} \cdot \alpha^{\mathbf{d}_2(x,y)} \qquad \text{for } y \in \{0, n\}$$

where $\alpha \in (0,1]$. This mechanism satisfies $\varepsilon{\cdot}\mathbf{d}_2$-privacy where $\mathbf{d}_2$ is the Euclidean metric and $\varepsilon = -\ln \alpha$.

The randomised response mechanism implements Warner's protocol, described briefly in §1, and was also the mechanism first associated with local differential privacy.

Definition 3.4.3 (Randomised Response Mechanism)**.** The α-randomised response mechanism $R\colon \mathcal{X} \to \mathbb{D}\mathcal{X}$ has the following channel matrix:

$$R_{x,x} = 1/k$$
$$R_{x,y} = \alpha/k \qquad \text{for } x \neq y$$

where k is a normalisation term and $\alpha \in (0,1]$. This mechanism satisfies $\varepsilon{\cdot}\mathbf{d}_{\mathrm{D}}$-privacy where $\mathbf{d}_{\mathrm{D}}$ is the discrete metric and $\varepsilon = -\ln \alpha$.

Examples of the truncated geometric mechanism and the randomised response mechanism are given in Figure 3.4.

TG	0	1	2
0	2/3	1/6	1/6
1	1/3	1/3	1/3
2	1/6	1/6	2/3

R	0	1	2
0	1/2	1/4	1/4
1	1/4	1/2	1/4
2	1/4	1/4	1/2

Figure 3.4: The α-truncated geometric mechanism (left) and the α-randomised response mechanism (right) for $\alpha = 1/2$ and $\mathcal{X} = \{0, 1, 2\}$. The truncated geometric mechanism is a square matrix satisfying $TG_{x,y} = \alpha TG_{x',y}$ on $x = x' + 1$ or $x' = x + 1$. The randomised response mechanism is a symmetric (square) matrix with maximum values on the diagonal and all other values equal.

3.4.1 Other Metrics

To date, there is no general method for the construction of $\mathbf{d}$-private mechanisms for arbitrary metrics of interest. For a few specific metrics, $\mathbf{d}$-private mechanisms are known. In Chapter 7 we will encounter the Laplace mechanism which is a continuous mechanism most similar to the geometric mechanism.[13] The exponential mechanism is a well-known mechanism designed for arbitrary measures on $\mathcal{Y}$ (rather than metrics on $\mathcal{X}$). We will encounter this mechanism in Chapter 4. The planar geometric and planar laplace mechanisms are $\mathbf{d}_2$-private extensions for points in the 2D plane and were developed for geo-indistinguishability [14]. For vectors on $\mathbb{R}^n$, it is well-known that any $\mathbf{d}_2$-private mechanism M on $\mathbb{R}$ can be extended to a $\mathbf{d}_{\mathrm{M}}$-private mechanism on $\mathbb{R}^n$ (where $\mathbf{d}_{\mathrm{M}}$ is the Manhattan metric [14]) by applying M to each element of the vector $v \in \mathbb{R}^n$ [7].

In this thesis we will see other constructions for $\mathbf{d}$-private mechanisms. In Chapter 4 we will show how it is possible to construct a $\mathbf{d}$-private mechanism for *any* metric of interest. In Chapter 9 we will show the construction of $\mathbf{d}_2$-private mechanisms on $\mathbb{R}^n$, extending the results for the 2D plane.[15] We will also show the construction of an $E_{\mathbf{d}_\mathcal{X}}$-private mechanism where $E_{\mathbf{d}_\mathcal{X}}$ is the Earth Mover's distance between distributions with respect to an underlying metric $\mathbf{d}_\mathcal{X}$.[16] In Chapter 10 we will show the construction of an (approximate) $\mathbf{d}_\theta$-private mechanism (where $\mathbf{d}_\theta$ is the angular distance metric) via a probabilistic mapping to the Hamming metric $\mathbf{d}_{\mathrm{H}}$. We also note that this technique can be used for other metrics including the Jaccard distance.

3.5 Discussion and Concluding Remarks

In this chapter we showed how differential privacy can be applied in general settings where indistinguishability is described using a metric. We showed how to reinterpret the privacy afforded in terms of indistinguishability within a radius, and the attacker's knowledge before and after an observation. We explained how the choice of metric affects both privacy and utility in different ways, depending on whether privacy is applied in the oblivious setting or in the local differential privacy model. We also observed that while reasoning about differential privacy guarantees has been well-established, reasoning about utility is much more difficult, and is frequently done empirically and 'after-the-fact'.

3.6 Chapter Notes

Much of the prior work on metric differential privacy presented in this chapter comes from the work of Chatzikokolakis et al. [15] who introduced metric differential privacy in 2012. Although

[13] In fact it is usually described as the continuous version of the geometric mechanism, although this property is non-trivial to prove. We will present this proof in Chapter 7.

[14] The Manhattan metric, or taxicab metric, measures the distance $\mathbf{d}_{\mathrm{M}}(v, v')$ between vectors $v, v' \in \mathbb{R}^n$ as $\mathbf{d}_{\mathrm{M}}(v, v') = \sum_i |v_i - v'_i|$.

[15] We subsequently discovered that our result is an instance of the K-norm mechanism [69], although our work was done independently.

[16] Also known as the Kantorovich distance.

this work contained several example use cases for metric differential privacy, it was the work by Andrés et al. [14] which detailed the use of **d**-privacy for geo-location privacy.

The results in §3.1.3 provide a so-called *semantic interpretation* of metric differential privacy, that regards the privacy guarantee as expressing a bound on the increase of knowledge (from prior to posterior) due to the answer reported by the mechanism. In particular, for **d**-privacy the semantic interpretation is expressed by Lem. 3.4, and was first pointed out (for the location privacy instance) in [14]. The seminal paper on **d**-privacy, [15], also proposed a semantic interpretation, with a rather different flavour, although formally equivalent. In the context of standard differential privacy, Lem. 3.4 wrt to databases and Hamming distance corresponds to the odds ratio on which is based the semantics interpretation provided in [25]. Before that, another version of semantic interpretation was presented in [5] and proved equivalent to a form of differential privacy called ε-indistinguishability. Essentially, in this version an adversary that queries the database, and knows all the database except one record, cannot infer too much about this record from the answer to the query reported by the mechanism. Later on, an analogous version of semantic interpretation was reformulated in [70] and proved equivalent to differential privacy. A different interpretation of differential privacy, called *semantic privacy*, was proposed by [67]. This interpretation is based on a comparison between two posteriors (rather between the posterior and the prior), and the authors show that, within certain limits, it is equivalent to differential privacy.

Part II

Analysis

4

Comparing Privacy Mechanisms

Quantitative information flow (QIF) and differential privacy are both concerned with the protection of sensitive information, but they are rather different approaches. In a QIF model, we reason about the *expected* probability of a successful attack, whereas in differential privacy, the privacy parameter ε is a *max-case* measure, in the sense that privacy is compromised by the existence of a possible attack, regardless of its probability.

Comparing systems is a fundamental task in these areas: one wishes to guarantee that replacing a system A by a system B is a safe operation – ie. that the privacy guarantees for B are no-worse than those of A. This is a basic requirement in *verification*, where computer programs are produced by combining pieces of code and the safety properties of the overall system can be reasoned about using algebraic properties of individual code fragments. One typically uses the *order* induced by the safety property to reason abstractly about program specifications, which are then produced via implementations that are *at least as safe* as the specification requires. In QIF, this is done using the refinement (pre)order: we can say that program B is at least as safe as program A iff $A \sqsubseteq B$. A system that specifies A as a requirement can then be safely implemented by program B. The notion of refinement thus provides a rigorous way of comparing mechanisms for the purposes of reasoning about their safety properties.

In differential privacy, mechanisms are typically compared based on their ε privacy parameter: one would say that B is at least as safe as A iff A satisfies ε-differential privacy while B satisfies ε'-differential privacy for $\varepsilon' \leq \varepsilon$. It is natural to ask: how safe is the order induced by ε? That is, is replacing A with B a safe operation wrt *all* adversarial threats to privacy?

In this chapter we explore the order induced by the ε order on channels in terms of the QIF notion of refinement. In §4.1 we give some examples to motivate the need for a robust method of comparing privacy mechanisms. In §4.2 we briefly review the average-case refinement order

introduced in Chapter 2, followed by a study of the max-case refinement order in §4.3 and then the privacy-based (ε) order in §4.4. In §4.5 we show how the orders can be efficiently verified, and when the refinement fails, how to construct a counter-example. In §4.6 we study the lattice properties of the orders. Finally, in §4.7 we apply the three orders ($\sqsubseteq^{\text{avg}}_{\mathbb{G}}$, $\sqsubseteq^{\text{max}}_{\mathbb{Q}}$, and $\sqsubseteq^{\text{prv}}_{\mathbb{M}}$) to the comparison of some well-known families of **d**-private mechanisms. Regarding the question of whether the order based on ε provides strong privacy guarantees, we find that the standard refinement of QIF holds within families of mechanisms (eg. geometric, randomised response), but rarely across different families. What this tells us is that while ε provides a specific privacy guarantee wrt "differential privacy"-style adversaries, replacing mechanisms based on their ε value is *not* safe wrt adversaries modelled using average-case notions of leakage.

4.1 Motivating Examples

In order to guide the design and the implementation of mechanisms for information protection, it is important to have a rigorous and effective way to establish whether one mechanism is better or worse than another one. This is not always an obvious task. To illustrate the point, consider the following examples.

Example 4.1.1. *Let P_1, P_2, P_3 and P_4 be the programs illustrated in Table 4.1, where H is a "high" (ie. secret) input and L is a "low" (ie. public) output. We assume that H is a uniformly distributed 32-bit integer with range $0 \leq H < 232$. All of these programs leak information about H via L in different ways: P_1 reveals H whenever it is a multiple of 8*[1] *and reveals nothing otherwise. P_2 does the same thing whenever H is a multiple of 4. P_3 reveals the last 8 bits of H.*[2] *and, analogously, P_4 reveals the last 4 bits of H.*

Now, it is clear that P_2 leaks more than P_1, and that P_4 leaks more than P_3, but how could we compare P_1 and P_3, for instance? It is debatable which one is worse because their behaviour is very different: P_1 reveals nothing in most cases, but when it does reveal something, it reveals everything. P_3, on the other hand, always reveals part of the secret. Clearly, we cannot decide which situation is worse unless we have some more information about the goals and the capabilities of the attacker. For instance, if the adversary has only one attempt at his disposal (and no extra information), then the program P_3 would be safer, because even after observing the output of L there are still 24 bits of H that are unknown. On the other hand, if the adversary can launch repeated attacks, then eventually P_3 will leak the secret completely and P_1 would be preferred. Thus, in this situation, the decision about which program is 'best' depends on the type of adversarial threat to be guarded against.

Example 4.1.2. *Consider a truncated geometric mechanism (cfr. Def. 3.4.2) with $\alpha = 25/27$, and a randomised response mechanism (cfr. Def. 3.4.3) with $\alpha = 11/22$. We can represent these mechanisms by the graphs of their (noise-adding) probability distributions on a single input*[3]*, as illustrated in*

[1] Recall that $H \mod 8$ represents the integer division of H by 8.

[2] Note that H & 02418 represents the bitwise conjunction between H and a string of 24 "0" bits followed by 8 "1" bits.

[3] Recall that in both of these mechanisms noise is generated independently of the input x.

P_1	P_2	P_3	P_4
`if H mod 8 = 0 then L := H else L := 1`	`if H mod 4 = 0 then L := H else L := 1`	`L := H & 0^24 1^8`	`L := H & 0^28 1^4`

Table 4.1: Programs that take as input a secret H and leak information about H via the output L.

Figure 4.1. Clearly, it does not make sense to compare them on the basis of their respective privacy parameters ε, because these represent different privacy properties. It is not obvious how to compare them in general: The geometric mechanism tends to make the true value indistinguishable from its immediate neighbours, but more distinguishable for values further away, whereas the randomised response mechanism introduces the same level of indistinguishability between the true value and every other value in the domain. Thus, which mechanism is more private depends on the kind of attack we want to mitigate: if the attacker is trying to guess an approximation of the value, then the randomised response is better. If the attacker is only interested in identifying the true value among the immediate neighbours, then the geometric is better.

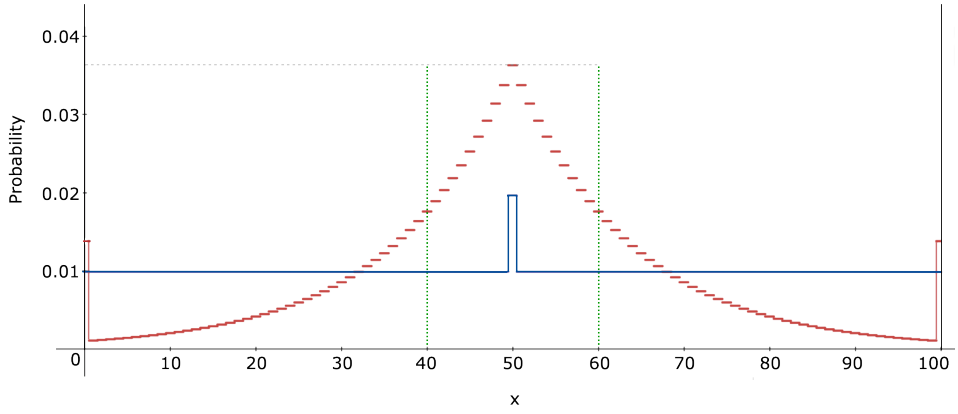


Figure 4.1: Comparison between the truncated geometric (red) and the randomised response (blue) mechanisms. The graphs show the distribution over outputs produced by each mechanism when the input is $x = 50$ for outputs in the range $\{0, 1, \ldots, 100\}$.

In this respect, the QIF approach has led to an elegant theory of refinement order[4], introduced in Chapter 2, which provides strong guarantees: $A \sqsubseteq_{\mathbb{G}}^{\text{avg}} B$ means that *B is safer than A in all circumstances*, in the sense that the *expected gain* of an attack on B is less than on A, for whatever kind of gain the attacker may be seeking. This means that we can always substitute the component A by B without compromising the security of the system. It is important to remark that this order is based on an *average* notion of adversarial gain (*vulnerability*), defined by mediating over all possible observations and their probabilities. We call this perspective *average-case*.

[4] In this paper we call $\sqsubseteq_{\mathbb{G}}^{\text{avg}}$ and the other refinement relations "orders", although, strictly speaking they are preorders.

At the other end of the spectrum, differential privacy and **d**-privacy are *max-case* measures, in that they represent the *worst-case* gain by an adversary, regardless of the probability of the particular observation that realises this gain. This can be seen from Lem. 3.4 in Chapter 3, which expresses a bound on how much the adversary can learn from each individual outcome of the mechanism.

In the literature of differential privacy and **d**-privacy, mechanisms are usually compared on the basis of their ε-value[5], which controls a bound on the log-likelihood ratio of an observation y given two "secrets" x_1 and x_2: smaller ε means more privacy. In differential privacy the bound is ε itself, while in **d**-privacy it is $\varepsilon \times \mathbf{d}(s_1, s_2)$. On the other hand, in QIF channels are compared with respect to their safety against all adversarial attacks, as described by the strong leakage ordering $\sqsubseteq_{\mathbb{G}}^{\text{avg}}$. We remark that the relation induced by ε in **d**-privacy is fragile, in the sense that the definition of **d**-privacy assumes an underlying metric structure **d** on the data, and whether a mechanism B is "better" than A depends in general on the metric considered.

Average-case and max-case are different principles, suitable for different scenarios. The max-case, as adopted by differential privacy, takes the perspective of the individual whose privacy is to be protected: the privacy guarantee is designed to limit the cost of *any* attack on that individual. However the perspective of the data curator is different: their goal is to understand the threat to the system from an adversarial attack – for example, by a machine learner – which is typically modelled (in QIF) using a g-vulnerability with adversarial success measured in the *average* case. It is important for both the data curator and the individual to be able to reason about adversarial threats, and this motivates our study of systems wrt both **d**-privacy guarantees and average-case leakage.

In this chapter, we combine the max-case perspective with the robustness of the QIF approach, and we introduce two refinement orders:

- $\sqsubseteq_{\mathbb{Q}}^{\text{max}}$, based on the max-case leakage introduced in [57]. This order takes into account all possible privacy breaches caused by any observable, but it quantifies over all possible quasi-convex vulnerability functions.
- $\sqsubseteq_{\mathbb{M}}^{\text{prv}}$, based on **d**-privacy, but quantified over all metrics **d**.

We will study their characterisations as structural relations between stochastic matrices (representing the mechanisms to be compared). We will also study the relation between the orders and their algebraic properties. Finally, we will analyse various mechanisms for **d**-privacy to see in which cases the order induced by ε is consistent with the three orders above.

4.1.1 Technical Preliminaries

We recall the following ideas from QIF introduced in Chapter 2.

We model secrets $\mathcal{X}$, about which the adversary has some probabilistic knowledge $\pi : \mathbb{D}\mathcal{X}$ ($\mathbb{D}\mathcal{X}$ denoting the set of probability distributions over $\mathcal{X}$). An adversary is modelled via a gain function $g(w, x)$ representing the adversary's gain when choosing action w when the real secret

[5] In standard differential privacy, ε is a parameter that usually appears explicitly in the definition of the mechanism. In **d**-privacy, it is typically an implicit scaling factor.

is x. The g-vulnerability of the system is then defined as the expected gain of an optimal guess: $V_g(\pi) = \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi_x g(w, x)$. Different adversaries can be modelled by proper choices of $\mathcal{W}$ and g. We denote by $\mathbb{G}\mathcal{X}$ the set of all gain functions.

A system is modelled as a *channel*: a probabilistic mapping from the (finite) set of secrets $\mathcal{X}$ to a finite set of observations $\mathcal{Y}$, described by a stochastic matrix C, where $C_{x,y}$ is the probability that secret x produces the observation y. When the adversary observes y, he can transform his initial knowledge π into a *posterior* knowledge $\delta^y : \mathbb{D}\mathcal{X}$. The *result of running a channel C* on the initial knowledge π is the hyper distribution $[\pi \triangleright C]$: a probability distribution on posteriors δ^y, each having probability a_y.

The (average-case) *posterior vulnerability* of the system is defined as the application of a vulnerability function V to each posterior δ^y, then averaging by its probability a_y of being produced:

$$V[\pi \triangleright C] \quad := \quad \sum_y a_y V(\delta^y) \,. \tag{4.1}$$

Since any continuous and convex function V can be written as V_g for a properly chosen g, when studying average-case leakage we can safely restrict to using g-vulnerabilities.

Leakage can finally be defined by comparing the prior and posterior vulnerabilities, eg. as $\mathcal{L}^+_g(\pi, C) = V_g[\pi \triangleright C] - V_g(\pi)$.[6]

4.2 Average-Case Refinement

In Chapter 2 we encountered the average-case notions of refinement ($\sqsubseteq$) and leakage ordering ($\sqsubseteq^{\text{avg}}_{\mathbb{G}}$). Recall that $\sqsubseteq$ is a *structural* notion that says that 2 channels A, B are related by refinement ($A \sqsubseteq B$) iff B can be written as $A \cdot R$ for some channel R. In contrast, $\sqsubseteq^{\text{avg}}_{\mathbb{G}}$ is an *operational* notion – it says that $A \sqsubseteq^{\text{avg}}_{\mathbb{G}} B$ iff B leaks no more than A against all adversaries that can be modelled using a gain function. $\sqsubseteq^{\text{avg}}_{\mathbb{G}}$ is also called a *testing refinement* because it allows us to produce a counter-example; if $A \not\sqsubseteq^{\text{avg}}_{\mathbb{G}} B$ then it means we can find a gain function g for which B leaks *more* than A. The co-incidence of $\sqsubseteq$ and $\sqsubseteq^{\text{avg}}_{\mathbb{G}}$ provides a powerful ordering on channels which is easy to verify (via $\sqsubseteq$) and *meaningful* in terms of the adversarial threat that it models (via $\sqsubseteq^{\text{avg}}_{\mathbb{G}}$).

4.3 Max-Case Refinement

Although $\sqsubseteq, \sqsubseteq^{\text{avg}}_{\mathbb{G}}$ provide a strong and precise way of comparing systems, one could argue that the average-case order might underestimate the threat to a system. For example, imagine that there exists a certain observation y such that the corresponding posterior δ^y is highly vulnerable (eg. the adversary can completely infer the real secret), but y happens with very small probability. In this case the average-case posterior vulnerability $V_g[\pi \triangleright C]$ can be relatively small even though $V_g(\delta^y)$ is large for that particular y (see Example 4.1).

A *risk-averse* scenario – in which any threat to the secret, no matter how unlikely, cannot be ignored – is better modelled using a *worst-case* measure of leakage rather than an average-case

[6] Comparing vulnerabilities "multiplicatively" is also possible, but is orthogonal to our goals.

Example 4.1. A situation where average-case refinement may underestimate the threat to a secret. Given the channel C, which is parametrised by some arbitrarily small value σ, we can compute the expected leakage of this channel wrt Bayes vulnerability.

C	y_1	y_2	y_3
x_1	σ	$1/2$	$1/2-\sigma$
x_2	0	$1/2$	$1/2$
x_3	0	$1/2$	$1/2$

$\xrightarrow{\upsilon = (1/3,\ 1/3,\ 1/3)}$

$[\upsilon \triangleright C]$	$\sigma/3$	$1/2$	$1/2-\sigma/3 = p(y_3)$
x_1	1	$1/3$	$1/6p(y_3) - \sigma/3p(y_3)$
x_2	0	$1/3$	$1/6p(y_3)$
x_3	0	$1/3$	$1/6p(y_3)$

We first compute the hyper $[\upsilon \triangleright C]$ as shown on the right hand side. The posterior Bayes vulnerability is then computed as

$$V_1[\upsilon \triangleright C] \;=\; \sigma/3 \cdot V_1(\delta^{y_1}) + 1/2 \cdot V_1(\delta^{y_2}) + (1/2 - \sigma/3) \cdot V_1(\delta^{y_3}) \;=\; \sigma/3 + 1/6 + 1/6 \;=\; \sigma/3 + 1/3$$

giving multiplicative leakage $\mathcal{L}_1^{\times}(\upsilon, C) = \frac{V_1[\upsilon \triangleright C]}{V_1(\upsilon)} = \sigma + 1$ which is close to 1 (ie. the minimum possible leakage) when σ is very small. This is despite the fact that $V_1(\delta^{y_1}) = 1$ since the observation y_1 leaks the value of the secret exactly. In this case, the average-case leakage underestimates the risk to the secret x_1 because the likelihood of observing y_1 (ie. $\sigma/3$) is so small.

measure, to properly estimate the risk to the secret regardless of its probability of occurring. We can naturally quantify such a leakage using a max-case variant of posterior vulnerability, where all observations are treated equally regardless of their probability of being produced. It has been shown [57] that in order for max-case vulnerabilities to satisfy fundamental properties such as the data-processing inequality, the *prior* vulnerability has to be a *quasi-convex* function of π instead of a convex function (as modelled by V_g). We will therefore write Q to denote the (quasi-convex) vulnerability function of type $\mathbb{Q}\mathcal{X}$ (the set of all continuous quasi-convex functions $\mathbb{D}\mathcal{X} \to \mathbb{R}_{\geq 0}$.) Recalling the definition of quasi-convex, this means that, given a set $\{\pi^1, \pi^2, \ldots, \pi^n\}$ of distributions and a corresponding set $\{a_1, a_2, \ldots, a_n\}$ of convex coefficients, we require $Q(\sum_i a_i \pi^i) \leq \max_i Q(\pi^i)$. We now define posterior max-case vulnerability as follows.

Definition 4.3.1 (Max-Case Vulnerability)**.** Given a vulnerability $Q\colon \mathbb{Q}\mathcal{X}$, a prior $\pi\colon \mathbb{D}\mathcal{X}$ and a channel C, the *posterior max-case vulnerability* is defined as

$$V_Q^{\max}[\pi \triangleright C] \;:=\; \max_y Q(\delta^y)$$

An example of a quasi-convex vulnerability that is not convex is the function $Q(\pi) = 1 - (\min_x \pi_x)^2$.

Inspired by $\sqsubseteq_{\mathbb{G}}^{\text{avg}}$, we can now define a corresponding max-case leakage order.

Definition 4.3.2. The max-case leakage order is defined as

$$A \sqsubseteq^{\max}_{\mathbb{Q}} B \qquad \text{iff} \qquad V^{\max}_{Q}[\pi \triangleright A] \;\geq\; V^{\max}_{Q}[\pi \triangleright B]$$

for all $Q : \mathbb{Q}\mathcal{X}$, $\pi : \mathbb{D}\mathcal{X}$.

As with its average-case variant, $\sqsubseteq^{\max}_{\mathbb{Q}}$ provides clear privacy guarantees by explicitly requiring that B leaks no more than A for all adversaries that can be modelled using a vulnerability Q. But this explicit quantification makes the order hard to reason about and verify. We would thus like to characterise $\sqsubseteq^{\max}_{\mathbb{Q}}$ by a refinement order that depends only on the structure of the two channels – ie. a *structural* characterisation.

Given a channel $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, we denote by $\tilde{C}$ the channel obtained by normalising[7] C's columns and then transposing:

$$\tilde{C}_{y,x} \;:=\; \frac{C_{x,y}}{\sum_x C_{x,y}} \,.$$

Note that the row y of $\tilde{C}$ can be seen as the posterior distribution δ^y obtained by C under the uniform prior. Note also that $\tilde{C}$ is non-negative and its rows sum up to 1, so it is a valid channel from $\mathcal{Y}$ to $\mathcal{X}$. The average-case refinement order required that B can be obtained by post-processing A. We define the *max-case refinement* order by requiring that $\tilde{B}$ can be obtained by *pre-processing* $\tilde{A}$.

Definition 4.3.3. The max-case refinement order is defined as

$$A \sqsubseteq^{\max} B \qquad \textit{iff} \qquad R\tilde{A} = \tilde{B}$$

for some channel R.

Note that, unlike its average-case variant which relates to the data-processing inequality, $\sqsubseteq^{\max}$ does not have an intuitive interpretation. However, we can also provide a "semantic" characterisation of $\sqsubseteq^{\max}$ by expressing it, not in terms of the channel matrices A and B, but in terms of the *posterior distributions* that they produce. Given the hyper $[\pi \triangleright C]$, its *support* $\lceil [\pi \triangleright C] \rceil$ is the set of all posteriors produced with non-zero probability. We also denote by $\mathbf{ch}\, S$ the *convex hull* of S. We then have the following:

Theorem 4.1. Let A, B be channels and $\pi : \mathbb{D}\mathcal{X}$. If $A \sqsubseteq^{\max} B$ then the posteriors of B (under π) are convex-combinations of those of A, that is

$$\lceil [\pi \triangleright B] \rceil \quad \subseteq \quad \mathbf{ch}\, \lceil [\pi \triangleright A] \rceil \,. \tag{4.2}$$

Moreover, if (4.2) holds and π is full support then $A \sqsubseteq^{\max} B$.

Note that if (4.2) holds for any full-support prior, then it must hold for all priors.

We can now relate $\sqsubseteq^{\max}$ and $\sqsubseteq$ in the space of hyper-distributions using the barycentric

[7] If a column consists of only zeroes it is simply removed.

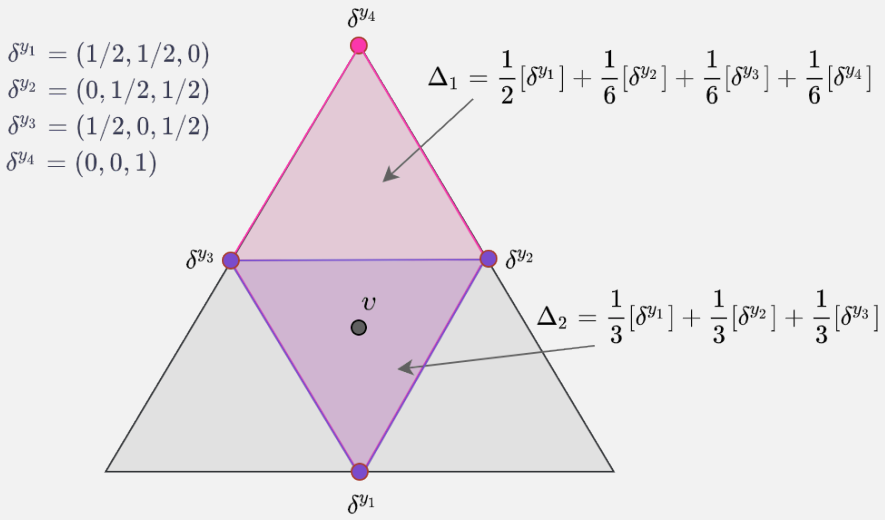
view of refinement (see Example 4.2). Both $\sqsubseteq^{\max}$ and $\sqsubseteq$ require that refining hypers have their posteriors within the convex hull of the refined hyper, however $\sqsubseteq^{\max}$, which ignores its outer distribution, does not have a corresponding Earth Moving constraint between the posteriors (cfr. Def. 2.4.3).

EXAMPLE 4.2 (Geometric relationship between $\sqsubseteq^{\max}$ and $\sqsubseteq$)**.** Consider the following channels.

A	y_1	y_2	y_3	y_4
x_1	$3/4$	0	$1/4$	0
x_2	$3/4$	$1/4$	0	0
x_3	0	$1/4$	$1/4$	$1/2$

B	y_1	y_2	y_3
x_1	$1/2$	0	$1/2$
x_2	$1/2$	$1/2$	0
x_3	0	$1/2$	$1/2$

Pushing the uniform distribution, υ, through A and B produces the hypers $\Delta_1 = \frac{1}{2}\cdot[\delta^{y_1}] + \frac{1}{6}\cdot[\delta^{y_2}] + \frac{1}{6}\cdot[\delta^{y_3}] + \frac{1}{6}[\delta^{y_4}]$ and $\Delta_2 = \frac{1}{3}\cdot[\delta^{y_1}] + \frac{1}{3}\cdot[\delta^{y_2}] + \frac{1}{3}\cdot[\delta^{y_3}]$ respectively. These hypers and their convex hulls are illustrated in the figure below.



Notice that $\lceil\Delta_2\rceil \subset \mathbf{ch}\lceil\Delta_1\rceil$ and so we deduce that $A \sqsubseteq^{\max} B$ (by Thm. 4.1). However, there is no "refining Earth Move" (see Def. 2.4.3) that can take Δ_1 to Δ_2 – specifically, we require an Earth Move to increase the weight of posterior δ^{y_2} from $\frac{1}{6}$ to $\frac{1}{3}$ but it is not inside the convex hull of any vertices which would enable this Earth Move. Thus we conclude $A \not\sqsubseteq B$ and similarly $B \not\sqsubseteq A$ (by Lem. 2.5 from §2.4.2).

We are now ready to give the main result of this section.

THEOREM 4.2. The orders $\sqsubseteq^{\max}$ and $\sqsubseteq^{\max}_{\mathbb{Q}}$ coincide.

Similarly to the average case, $A \sqsubseteq^{\max} B$ gives us a strong leakage guarantee: we can safely replace A by B, knowing that for any adversary, the max-case leakage of B can be no-larger than that of A. Moreover, in case $A \not\sqsubseteq^{\max} B$, we can always find an adversary, modelled by

a vulnerability function Q, who prefers (wrt the max-case) interacting with A over B. This is discussed in §4.5.

Finally, we resolve the question of how $\sqsubseteq^{\max}$ and $\sqsubseteq$ are related.

THEOREM 4.3. $\sqsubseteq$ is strictly stronger than $\sqsubseteq^{\max}$.

This result might appear counter-intuitive at first; we might expect $A \sqsubseteq^{\max} B$ to imply $A \sqsubseteq B$. To understand why it does not, note that the former only requires that, for each output y_B of B, there exists some output of y_A that is at least as vulnerable, regardless of how likely y_A and y_B are to happen. The other direction might also appear tricky: if B leaks no more than A in the average-case, it must also leak no more than B in the max-case. The quantification over all gain functions in the average-case is powerful enough to "detect" differences in max-case leakage. Example 4.2 provides further intuition using the geometry of refinement.

The above result also means that $\sqsubseteq$ could be useful even if we are interested in the max-case, since it gives us $\sqsubseteq^{\max}$ "for free". In other words, we can safely reason using $\sqsubseteq$ with the knowledge that any substitution A for B which is safe wrt average-case adversaries is also safe wrt max-case adversaries.

4.4 Privacy-Based Refinement

So far we have compared systems based on their (average-case or max-case) leakage. In this section we turn our attention to $\mathbf{d}$-privacy and discuss new ways of comparing mechanisms based on ε as a "leakage" notion.

4.4.1 Comparing mechanisms by their "smallest ε" (for fixed d)

In Chapter 3 we defined $\varepsilon{\cdot}\mathbf{d}$-privacy as a constraint between values x, x' of the input space. However, we did not require that the constraint be met on any pair x, x' – that is, we did not specify that ε should be the "smallest possible" value for which a channel satisfies $\varepsilon{\cdot}\mathbf{d}$-privacy. Remembering that smaller values of ε constitute more privacy, it is advantageous to report the minimum ε for a channel, since this provides both stronger guarantees and more leeway in terms of so-called "privacy budget". Additionally, in order to consider ε as a leakage measure, we should associate to each channel, for a given $\mathbf{d}$, the smallest ε by which we can scale $\mathbf{d}$ without violating privacy.

DEFINITION 4.4.1. The privacy-based leakage (wrt $\mathbf{d}$) of a channel C is defined as

$$\mathrm{Priv}_{\mathbf{d}}(C) \quad := \quad \inf\{\varepsilon \geq 0 \mid C \text{ satisfies } \varepsilon \cdot \mathbf{d}\text{-privacy}\} \ .$$

Note that $\mathrm{Priv}_{\mathbf{d}}(C) = +\infty$ iff there is no such ε; also we say that $\mathrm{Priv}_{\mathbf{d}}(C) \leq 1$ iff C satisfies $\mathbf{d}$-privacy (ie. for $\varepsilon = 1$).

We can now compare two mechanisms A and B based on their "smallest ε".

Definition 4.4.2. The privacy-based leakage order is defined as

$$A \sqsubseteq^{\text{prv}}_{\mathbf{d}} B \qquad \text{iff} \qquad \text{Priv}_{\mathbf{d}}(A) \geq \text{Priv}_{\mathbf{d}}(B)\,.$$

For instance, $A \sqsubseteq^{\text{prv}}_{\mathbf{d}_H} B$ means that B satisfies standard differential privacy for ε at least as small as the one of A.

The privacy-based leakage ordering $\sqsubseteq^{\text{prv}}_{\mathbf{d}}$ is a weak ordering, in the sense that it can hold in either direction depending on the metric of interest. Example 4.3 shows when this can occur.

Example 4.3 (Privacy-based leakage order depends on the metric)**.** We revisit 2 channels presented in Chapter 3 which are defined over the input space $X = \{0, 1, 2\}$.

G	y_0	y_1	y_2
0	2/3	1/6	1/6
1	1/3	1/3	1/3
2	1/6	1/6	2/3

R	y_0	y_1	y_2
0	3/5	1/5	1/5
1	1/5	3/5	1/5
2	1/5	1/5	3/5

We can calculate the privacy-based leakage for these channels wrt 2 different metrics on X: the Euclidean metric ($\mathbf{d}_2$) and the Discrete metric ($\mathbf{d}_D$).
For $\mathbf{d}_2$ we compute $\text{Priv}_{\mathbf{d}_2}(G) = \ln 2$ and $\text{Priv}_{\mathbf{d}_2}(R) = \ln 3$ and therefore $R \sqsubseteq^{\text{prv}}_{\mathbf{d}_2} G$.
However, for $\mathbf{d}_D$ we have $\text{Priv}_{\mathbf{d}_D}(G) = \ln 4$ and $\text{Priv}_{\mathbf{d}_D}(R) = \ln 3$ and therefore $G \sqsubseteq^{\text{prv}}_{\mathbf{d}_D} R$.
This shows how the leakage ordering wrt ε can also depend on the metric on X.

4.4.2 Privacy-based leakage and refinement orders

When discussing the average- and max-case leakage orders $\sqsubseteq^{\text{avg}}_{\mathbb{G}}, \sqsubseteq^{\text{max}}_{\mathbb{Q}}$, we obtained strong leakage guarantees by quantifying over *all vulnerability functions*. It is thus natural to investigate a similar quantification in the context of $\mathbf{d}$-privacy. Namely, we define a stronger privacy-based "leakage" order, by comparing mechanisms not on a single metric $\mathbf{d}$, but on *all metrics* simultaneously.

Definition 4.4.3. The privacy-based leakage order is defined as

$$A \sqsubseteq^{\text{prv}}_{\mathbb{M}} B \qquad \text{iff} \qquad A \sqsubseteq^{\text{prv}}_{\mathbf{d}} B \qquad \text{for all } \mathbf{d} : \mathbb{M}X\,.$$

Similarly to the other leakage orders, the drawback of $\sqsubseteq^{\text{prv}}_{\mathbb{M}}$ is that it quantifies over an uncountable family of metrics. As a consequence, our first goal is to characterise it as a property of the channel matrix alone, which would make it much easier to reason about or verify.

To do so, we start by recalling an alternative way of thinking about $\mathbf{d}$-privacy. Consider the *multiplicative total variation* distance between probability distributions $\mu, \mu' \in \mathbb{D}\mathcal{Y}$, defined as:

$$\text{tv}_{\otimes}(\mu, \mu') \;:=\; \max_{y \in \mathcal{Y}} \left|\ln \frac{\mu_y}{\mu'_y}\right|\,.$$

We also recall that a function $f : \mathcal{X} \to \mathcal{Y}$ on metric spaces $(\mathcal{X}, d_x), (\mathcal{Y}, d_y)$ is said to be *1-Lipschitz* if $d_y(f(x), f(x')) \leq d_x(x, x')$ for all $x, x' \in \mathcal{X}$.

Remembering that the matrix C can also be viewed as a function $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ (mapping every x to the distribution $C_{x,-}$), we have that

$$C \text{ satisfies } \mathbf{d}\text{-privacy} \quad \text{iff} \quad \mathrm{tv}_{\otimes}(C_{x,-}, C_{x',-}) \leq \mathbf{d}(x, x') \,.$$

In other words iff C is 1-Lipschitz wrt $\mathrm{tv}_{\otimes}, \mathbf{d}$.

Next, we introduce the concept of the *distinguishability metric* of a channel.

Definition 4.4.4. Given a channel C, we define the *distinguishability metric* $\mathbf{d}_C : \mathbb{M}\mathcal{X}$ induced by C as

$$\mathbf{d}_C(x, x') \quad := \quad \mathrm{tv}_{\otimes}(C_{x,-}, C_{x',-}) \,.$$

Intuitively, $\mathbf{d}_C(x, x')$ expresses exactly how much the channel distinguishes (wrt $\mathrm{tv}_{\otimes}$) the values x, x'. It is easy to see that $\mathbf{d}_C$ is a metric (since $\mathrm{tv}_{\otimes}$ is) and it is the *smallest metric* for which C is private; in other words, for any $\mathbf{d}$:

$$C \text{ satisfies } \mathbf{d}\text{-privacy} \qquad \text{iff} \qquad \mathbf{d} \geq \mathbf{d}_C$$

where the comparison between $\mathbf{d}$ and $\mathbf{d}_C$ is taken pointwise.

We can now give a refinement order on mechanisms by comparing their induced metrics.

Definition 4.4.5. The privacy-based refinement order is defined as

$$A \sqsubseteq^{\mathrm{prv}} B \qquad \text{iff} \qquad \mathbf{d}_A \geq \mathbf{d}_B$$

or equivalently, $A \sqsubseteq^{\mathrm{prv}} B$ iff B satisfies $\mathbf{d}_A$-privacy.

The structural refinement $\sqsubseteq^{\mathrm{prv}}$ is simple to verify as demonstrated in Example 4.4.

Example 4.4. Returning to Example 4.3, we can compute the distinguishability metrics for the channels G and R:

$$\begin{array}{lll} \mathbf{d}_G(0,1) = \ln 2 & \qquad & \mathbf{d}_R(0,1) = \ln 3 \\ \mathbf{d}_G(1,2) = \ln 2 & & \mathbf{d}_R(1,2) = \ln 3 \\ \mathbf{d}_G(0,2) = \ln 4 & & \mathbf{d}_R(0,2) = \ln 3 \end{array}$$

Since $\mathbf{d}_G(0,1) < \mathbf{d}_R(0,1)$ and $\mathbf{d}_G(0,2) > \mathbf{d}_R(0,2)$ we have $G \not\sqsubseteq^{\mathrm{prv}} R$ and $R \not\sqsubseteq^{\mathrm{prv}} G$.

This achieves our goal of characterising $\sqsubseteq^{\mathrm{prv}}_{\mathbb{M}}$.

Proposition 4.4. The orders $\sqsubseteq^{\mathrm{prv}}_{\mathbb{M}}$ and $\sqsubseteq^{\mathrm{prv}}$ coincide.

Intuitively, Prop. 4.4 tells us that channel A is at least as safe as channel B on *all* metrics whenever A is as safe as B wrt their characteristic *distinguishability* metrics. This means we

could substitute channel A for channel B without having to worry about what metric is being used to interpret the privacy guarantee.

Finally we address the question of how these orders relate to each other.

> **THEOREM 4.5.** $\sqsubseteq^{\max}$ is strictly stronger than $\sqsubseteq^{\text{prv}}$, which is strictly stronger than $\sqsubseteq^{\text{prv}}_{\mathbf{d}}$.

One way to understand the fact that $\sqsubseteq^{\max}$ is stronger than $\sqsubseteq^{\text{prv}}$ is by viewing $\text{Priv}_{\mathbf{d}}$ as a max-case information leakage for each $\mathbf{d}$. This means that when $\sqsubseteq^{\max}$ holds, then $\text{Priv}_{\mathbf{d}}$ also holds for all $\mathbf{d}$, which gives $\sqsubseteq^{\text{prv}}$. This is discussed in detail in §4.4.4. For the "strictly" part, a counter-example is given in Example 4.5.

> **EXAMPLE 4.5.** To show that $\sqsubseteq^{\max}$ is strictly stronger than $\sqsubseteq^{\text{prv}}$ (Thm. 4.5) consider the following counter-example:
>
> | A | y_1 | y_2 |
> |---|---|---|
> | x_1 | $4/5$ | $1/5$ |
> | x_2 | $2/5$ | $3/5$ |
>
> | B | y_1 | y_2 |
> |---|---|---|
> | x_1 | $2/5$ | $3/5$ |
> | x_2 | $4/5$ | $1/5$ |
>
> .
>
> The only difference between A and B is that the two rows have been swapped. Hence, $\mathbf{d}_A = \mathbf{d}_B$ which implies $A \sqsubseteq^{\text{prv}} B \sqsubseteq^{\text{prv}} A$. However, consider the hypers of A and B wrt the uniform distribution:
>
> | $[\upsilon \triangleright A]$ | $3/5$ | $2/5$ |
> |---|---|---|
> | x_1 | $2/3$ | $1/4$ |
> | x_2 | $1/3$ | $3/4$ |
>
> | $[\upsilon \triangleright B]$ | $3/5$ | $2/5$ |
> |---|---|---|
> | x_1 | $1/3$ | $3/4$ |
> | x_2 | $2/3$ | $1/4$ |
>
> .
>
> Since $(3/4, 1/4)$ cannot be written as a convex combination of $(2/3, 1/3)$ and $(1/4, 3/4)$, and similarly $(1/4, 3/4)$ cannot be written as a convex combination of $(1/3, 2/3)$ and $(3/4, 1/4)$, from Thm. 4.1 we conclude that $A \not\sqsubseteq^{\max} B \not\sqsubseteq^{\max} A$. This can be observed from their barycentric representation illustrated in Figure 4.2.

The result of Thm. 4.5 means that even if we "only" care about $\mathbf{d}$-privacy, it is safe to reason wrt with $\sqsubseteq$ or $\sqsubseteq^{\max}$ as these both imply $\sqsubseteq^{\text{prv}}_{\mathbf{d}}$. In other words, if channel A is safer than channel B wrt $\sqsubseteq^{\max}$ (say) – ie. $B \sqsubseteq^{\max} A$ – then we know that A must also be safer wrt $\mathbf{d}$-privacy, ie. that $B \sqsubseteq^{\text{prv}}_{\mathbf{d}} A$, or equivalently, that A has a "smaller ε" than B.

The relationship between the orders is summarised in Figure 4.3.

4.4.3 Application to oblivious mechanisms

We conclude the discussion on privacy-based refinement by showing the usefulness of our strong $\sqsubseteq^{\text{prv}}$ order in the case of oblivious mechanisms. Recall from Chapter 3 (§3.3.1) that oblivious mechanisms K are $\mathbf{d}$-private mechanisms which can be decomposed into a deterministic query f and a probabilistic channel M such that $K = M \circ f$.

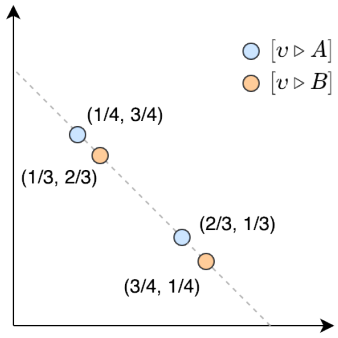


Figure 4.2: Failure of refinement for the hypers $[v \triangleright A]$ (blue) and $[v \triangleright B]$ (orange) from Example 4.5. The orange and blue points are posteriors lying on the line $x_1 + x_2 = 1$. The orange points are not both contained within the convex hull of the blue points and vice versa, and therefore $A \not\sqsubseteq^{\max} B$ and $B \not\sqsubseteq^{\max} A$.

THEOREM 4.6. Let $f : \mathcal{X} \to \mathcal{Y}$ be any query and A, B be two mechanisms on $\mathcal{Y}$. If $A \sqsubseteq^{\text{prv}} B$ then $A \circ f \sqsubseteq^{\text{prv}} B \circ f$.

Proof. We reason as follows:

$$
\begin{array}{lll}
 & A \sqsubseteq^{\text{prv}} B & \\
\Rightarrow & \mathbf{d}_A \geq \mathbf{d}_B & \text{“Def. 4.4.5”} \\
\Rightarrow & \text{tv}_{\otimes}(A_{y,-}, A_{y',-}) \geq \text{tv}_{\otimes}(B_{y,-}, B_{y',-}) & \text{“Def. 4.4.4, for any } y, y' \in \mathcal{Y}\text{”} \\
\Rightarrow & \text{tv}_{\otimes}(A_{f(x),-}, A_{f(x'),-}) \geq \text{tv}_{\otimes}(B_{f(x),-}, B_{f(x'),-}) & \text{“Substituting } y = f(x),\ y' = f(x')\text{”} \\
\Rightarrow & \text{tv}_{\otimes}((A \circ f)_{x,-}, (A \circ f)_{x',-}) \geq \text{tv}_{\otimes}((B \circ f)_{x,-}, (B \circ f)_{x',-}) & \text{“Simplify”} \\
\Rightarrow & A \circ f \sqsubseteq^{\text{prv}} B \circ f & \text{“Def. 4.4.5 and Def. 4.4.4”}
\end{array}
$$

□

This means that, replacing A by B is the context of an oblivious mechanism is always safe, regardless of the query (and its sensitivity) and regardless of the metric by which the privacy of the composed mechanism is evaluated.

Assume, for instance, that we care about standard differential privacy, and we have properly constructed A such that $A \circ f$ satisfies ε-differential privacy for some ε. If we know that $A \sqsubseteq^{\text{prv}} B$ (several such cases are discussed in §4.7) we can replace A by B without even knowing what f does. The mechanism $B \circ f$ is guaranteed to also satisfy ε-differential privacy.

Note also that the above theorem *fails for the weaker order* $\sqsubseteq^{\text{prv}}_{\mathbf{d}}$. Establishing $A \sqsubseteq^{\text{prv}}_{\mathbf{d}_\mathcal{Y}} B$ for some metric $\mathbf{d}_\mathcal{Y} : \mathbb{M}\mathcal{Y}$ gives no guarantees that $A \circ f \sqsubseteq^{\text{prv}}_{\mathbf{d}_\mathcal{X}} B \circ f$ for some other metric of interest $\mathbf{d}_\mathcal{X} : \mathbb{M}\mathcal{X}$. This is because $A \sqsubseteq^{\text{prv}}_{\mathbf{d}_\mathcal{Y}} B$ does not imply $A \sqsubseteq^{\text{prv}}_{\mathbf{d}_\mathcal{X}} B$ (cf. Example 4.3). It is possible that replacing A by B in this case is not safe.

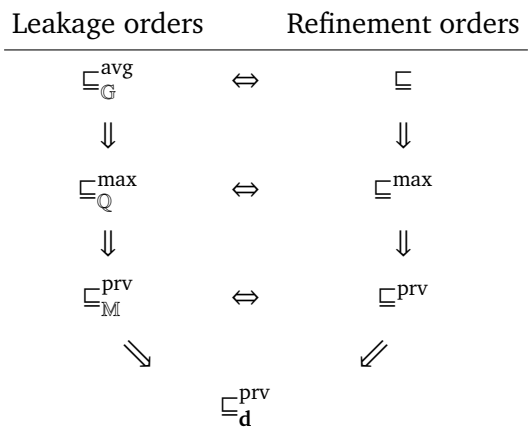


Figure 4.3: Comparison of leakage and refinement orders. All implications are strict.

4.4.4 Privacy as max-case capacity

Recalling Thm. 4.5, we noted that one way to check whether $\sqsubseteq^{\text{max}}$ is stronger than $\sqsubseteq^{\text{prv}}$ is to examine whether $\mathbf{d}$-privacy can be expressed as a (max-case) information leakage. We address this now.

We begin by defining a suitable vulnerability function:

Definition 4.4.6. The $\mathbf{d}$-vulnerability function $V_{\mathbf{d}}$ is defined as

$$V_{\mathbf{d}}(\pi) \quad := \quad \inf\{\varepsilon \geq 0 \mid \forall x, x' \in \mathcal{X}, \pi_x \leq e^{\varepsilon \cdot \mathbf{d}(x,x')}\pi_{x'}\} \ .$$

That is the smallest value ε such that π (seen as a function $\mathcal{X} \to [0,1]$) is $\varepsilon \cdot \mathbf{d}$-private. Namely, it represents the minimum distinction factor by which π increases the separation of two elements with respect to the underlying metric $\mathbf{d}$. Note the difference between $V_{\mathbf{d}}(\pi)$ (a vulnerability function on *distributions*) and $\text{Priv}_{\mathbf{d}}(C)$ (a "leakage" measure on *channels*). We show now that $V_{\mathbf{d}}$ is *quasi-convex* and thus can be used as a max-case vulnerability.

Lemma 4.7. The $\mathbf{d}$-vulnerability $V_{\mathbf{d}}$ (from Def. 4.4.6) is quasi-convex in π.

Proof. Observe that Def. 4.4.6 is equivalent to

$$V_{\mathbf{d}}(\pi) \quad = \quad \sup_{x,x'} \mathbf{d}^{-1}(x,x') \ln \frac{\pi_x}{\pi_{x'}} \quad , \tag{4.3}$$

although the above reformulation is to be used carefully with 0 and ∞ values of ε. Consider the finite set $\{\pi_1, \pi_2, \ldots \pi_n\}$ of full-support distributions on $\mathcal{X}$ and coefficients $\{\alpha_1, \alpha_2, \ldots \alpha_n\}$ satisfying $\alpha_i \in (0,1)$ and $\sum_i \alpha_i = 1$. Writing now $\pi_i(x)$ for the probability assigned by π_i to x, for fixed x, x' let $\max_i \frac{\pi_i(x)}{\pi_i(x')} = \varepsilon_k$ occurring on π_k. ie. $\frac{\pi_i(x)}{\pi_i(x')} \leq \frac{\pi_k(x)}{\pi_k(x')}$, or

$$\frac{\pi_i(x)}{\pi_k(x)} \leq \frac{\pi_i(x')}{\pi_k(x')} \tag{4.4}$$

for all i. Thus we have:

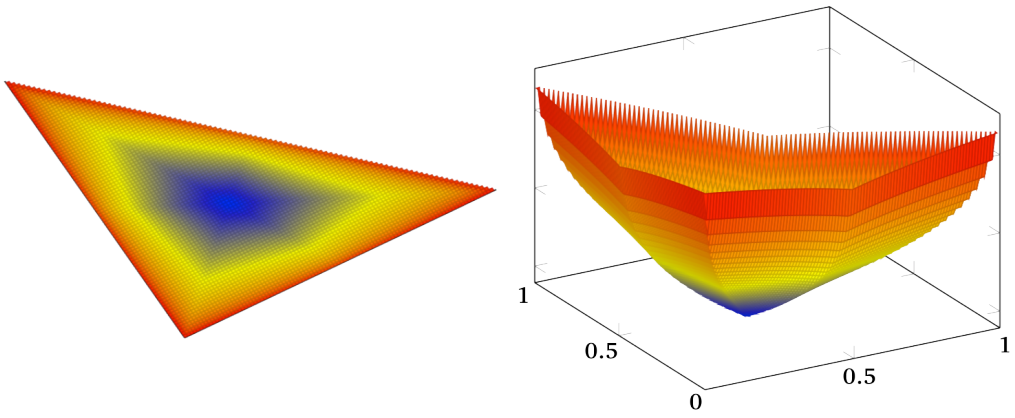


Figure 4.4: Plot of the d-vulnerability function $V_d(\pi)$ for Discrete metric d on distributions over 3 secrets showing top view (left) and side view (right). The function is *quasi-convex* and thus can be used to describe a max-case vulnerability.

$$
\begin{array}{lll}
 & {}^{\sum_i \alpha_i \pi_i(x)}/_{\sum_j \alpha_j \pi_j(x')} & \\
= & {}^{\alpha_k \pi_k(x)}/_{\alpha_k \pi_k(x')} \left(\frac{1+\sum_{i\neq k} {}^{\alpha_i \pi_i(x)}/_{\pi_k(x)}}{1+\sum_{j\neq k} {}^{\alpha_j \pi_j(x')}/_{\pi_k(x')}} \right) & \text{“Factorising”} \\
\leq & {}^{\alpha_k \pi_k(x)}/_{\alpha_k \pi_k(x')} \left(\frac{1+\sum_{i\neq k} {}^{\alpha_i \pi_i(x')}/_{\pi_k(x')}}{1+\sum_{j\neq k} {}^{\alpha_j \pi_j(x')}/_{\pi_k(x')}} \right) & \text{“Substituting (4.4)”} \\
= & {}^{\pi_k(x)}/_{\pi_k(x')} & \text{“Simplify”} \\
= & \max_i \frac{\pi_i(x)}{\pi_i(x')} & \text{“Assumption”}
\end{array}
$$

And now we find:

$$
\begin{array}{lll}
 & V_{\mathbf{d}}(\sum_i \alpha_i \pi_i) & \\
= & \sup_{x,x'} \mathbf{d}^{-1}(x,x') \ln \frac{\sum_i \alpha_i \pi_i(x)}{\sum_j \alpha_j \pi_j(x')} & \text{“Eqn (4.3)”} \\
\leq & \sup_{x,x'} \mathbf{d}^{-1}(x,x') \ln(\max_i \frac{\pi_i(x)}{\pi_i(x')}) & \text{“Above result, monotonicity of ln”} \\
= & \sup_{x,x'} \mathbf{d}^{-1}(x,x') \max_i \ln \frac{\pi_i(x)}{\pi_i(x')} & \text{“Monotonicity”} \\
= & \max_i \sup_{x,x'} \mathbf{d}^{-1}(x,x') \ln \frac{\pi_i(x)}{\pi_i(x')} & \text{“Independence of } \pi_i\text{”} \\
= & \max_i V_{\mathbf{d}}(\pi_i) & \text{“Eqn (4.3)”}
\end{array}
$$

And thus we have that $V_{\mathbf{d}}$ is a quasi-convex function of π. □

An example of the quasi-convex $V_{\mathbf{d}}$ on 3 secrets is plotted in Figure 4.4.

A fundamental notion in QIF is that of *capacity*: the maximum leakage over all priors. It turns out that for $V_{\mathbf{d}}$, the prior that realises the maximum capacity is the uniform one. In the following, $\mathcal{L}_{\mathbf{d}}^{+,\max}$ denotes the additive max-case $\mathbf{d}$ leakage, defined:

$$\mathcal{L}_{\mathbf{d}}^{+,\max}(\pi, C) \quad := \quad V_{\mathbf{d}}^{\max}[\pi \triangleright C] - V_{\mathbf{d}}(\pi). \tag{4.5}$$

and $\mathcal{ML}_{\mathbf{d}}^{+,\max}$ denotes the additive max-case $\mathbf{d}$-capacity, namely:

$$\mathcal{ML}_{\mathbf{d}}^{+,\max}(C) \quad := \quad \max_{\pi} \mathcal{L}_{\mathbf{d}}^{+,\max}(\pi, C). \tag{4.6}$$

Intuitively, $\mathcal{L}_{\mathbf{d}}^{+,\max}$ tells us how much the channel C contributes to the distinguishability of the secrets (in the posteriors) established in the prior π. The bigger $\mathcal{L}_{\mathbf{d}}^{+,\max}$ is, the more influence C has on distinguishability.

THEOREM 4.8. $\mathcal{ML}_{\mathbf{d}}^{+,\max}$ is always achieved on a uniform prior υ. Namely

$$\max_{\pi} \mathcal{L}_{\mathbf{d}}^{+,\max}(\pi, C) \;=\; \mathcal{L}_{\mathbf{d}}^{+,\max}(\upsilon, C) \;=\; V_{\mathbf{d}}^{\max}[\upsilon \triangleright C] \quad .$$

Since the uniform prior makes all secrets indistinguishable, the max $\mathbf{d}$-capacity $\mathcal{ML}_{\mathbf{d}}^{+,\max}$ is achieved when the channel C is *completely* responsible for distinguishability of secrets in the posteriors. Intuitively, this means that when $\mathcal{ML}_{\mathbf{d}}^{+,\max}$ is small, C does not make secrets very distinguishable, which we interpret as C providing *more privacy*.

This finally brings us to our goal of expressing $\mathrm{Priv}_{\mathbf{d}}$ in terms of information leakage (for a proper vulnerability function).

THEOREM 4.9. (DP as max-case capacity) C satisfies $\varepsilon \cdot \mathbf{d}$-privacy iff $\mathcal{ML}_{\mathbf{d}}^{+,\max}(C) \leq \varepsilon$. In other words: $\mathcal{ML}_{\mathbf{d}}^{+,\max}(C) = \mathrm{Priv}_{\mathbf{d}}(C)$.

Thus ε expresses a particular max-case capacity wrt a quasi-convex vulnerability function, showing that $\sqsubseteq^{\mathrm{prv}}$ is indeed weaker than $\sqsubseteq^{\max}$.

4.4.5 Revisiting the Data Processing Inequality for d-Privacy

We can now provide a semantic interpretation of the data processing inequality for $\mathbf{d}$-privacy. We first restate the definition from Chapter 3.

LEMMA 3.3 (DPI for $\mathbf{d}$-Privacy)**.** Let $M : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be an $\varepsilon{\cdot}\mathbf{d}$-private channel and let $R : \mathcal{Y} \to \mathbb{D}\mathcal{Z}$ be a post-processing step modelled as a channel. Then the composed channel MR is also $\varepsilon{\cdot}\mathbf{d}$-private, where we write MR for the matrix multiplication of M by R.

In QIF terms the data processing inequality on channels is stated in terms of the leakage properties of the channels [57]:

LEMMA 4.10 (Data Processing Inequality)**.** Let C, R be channels on $\mathcal{X}$ and let $\pi : \mathbb{D}\mathcal{X}$. Then

$$V_g[\pi \triangleright C] \;\geq\; V_g[\pi \triangleright CR]$$

for all gain functions $g : \mathbb{G}\mathcal{X}$.

Lem. 4.10 tells us that post-processing by any channel R cannot increase an attacker's expected gain wrt *any* gain function g. However, the result in Lem. 3.3 does not express the guarantee in terms of the adversary's knowledge before and after post-processing. We can do this now by rewriting the ε guarantee of the mechanism in terms of the vulnerability function $V_{\mathbf{d}}$ from Def. 4.4.6. The function $V_{\mathbf{d}}$ is *quasi-convex*, and hence we can apply it to max-case

adversaries, which motivates the following result:

> **LEMMA 4.11.** Let $M : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be an $\varepsilon\cdot\mathbf{d}$-private mechanism and let $R : \mathcal{Y} \to \mathbb{D}\mathcal{Z}$ be a post-processing step modelled as a channel. Let $V_{\mathbf{d}}^{\max}$ be the max-case vulnerability function defined for the vulnerability $V_{\mathbf{d}} : \mathbb{Q}\mathcal{X}$. Then for any $\pi : \mathbb{D}\mathcal{X}$ it holds that:
>
> $$V_{\mathbf{d}}^{\max}[\pi \triangleright M] \;\geq\; V_{\mathbf{d}}^{\max}[\pi \triangleright MR] \,.$$
>
> *Proof.* Follows from the result in [57] which says that the data-processing inequality holds for any max-case adversary modelled using a quasi-convex vulnerability. □

Lem. 4.11 is an alternative statement of Lem. 3.3 using vulnerabilities. It tells us that the data processing inequality for $\mathbf{d}$-private mechanisms can be interpreted as: post-processing by any channel cannot increase the max-case leakage to an adversary modelled using the $V_{\mathbf{d}}$ vulnerability function. This is a weaker guarantee than the DPI for channels which applies to all average-case adversaries (and therefore all max-case adversaries) modelled using *any* gain function (or any quasi-convex vulnerability in the max-case).

4.5 Verifying the Refinement Orders

The usefulness of refinement orders stems from both the ability to easily *verify* whether refinement holds and, in the case that refinement does not hold, to find a *counter-example* (eg. a gain function or a vulnerability function) as a witness to the failure of refinement. In this section, we turn our attention to these problems, given two representations of channels A and B (ie. as channel matrices). Since all of our orders have structural refinements, the problem of verification can be easily solved, and we will show that it can be performed in time polynomial in the size of the matrices. Moreover, when one of the order fails, we discuss how to obtain a counter-example (eg. a gain function g or a vulnerability function V), demonstrating this fact.

4.5.1 Average-case refinement

Verifying $A \sqsubseteq B$ can be done in polynomial time (in the size of A, B) by solving the system of equations $AR = B$, with variables R, under the linear constraints that R is a channel matrix (non-negative and rows sum up to 1). However, if the system has no solution (ie. $A \not\sqsubseteq B$), this method does not provide us with a counter-example gain function g.

We now show an alternative efficient method based on the constructive proof of the 'Coriaceous conjecture' from [61]. We first define $A^{\uparrow} := \{AR \mid R \text{ is a channel}\}$, the set of all channels obtainable by post-processing A. The idea is to compute the projection of B on $A^{\uparrow}$. Clearly, the projection is B itself iff $A \sqsubseteq B$; otherwise, the projection can be directly used to construct a counter-example g.

THEOREM 4.12. ('Coriaceous conjecture' from [61]) Let B^* be the projection of B on $A^{\uparrow}$.

1. If $B = B^*$ then $A \sqsubseteq B$.
2. Otherwise, let $G = B - B^*$. The gain function $g(w,x) = G_{x,w}$ provides a counter-example to $A \sqsubseteq B$, that is $V_g[\upsilon \triangleright A] < V_g[\upsilon \triangleright B]$, for uniform υ.

Rewriting A, B as vectors (by 'gluing' each channel's columns together), we can compute the projection B^* as the vector with minimum distance to the closed, convex set $A^{\uparrow}$. Using $\|x - y\|_2^2 = x^T x - 2x^T y + y^T y$, we see that the projection of y to a convex set can be solved as $\min_x x^T x - 2x^T y$. Therefore the projection B^* can be computed using a quadratic program, setting $y = B$ and constraining x to the representation AR, as a member of $A^{\uparrow}$. This can be solved in polynomial time.

Note that the proof of McIver et al. [61] uses the separating-hyperplane theorem to show the existence of a counter example g in case $A \not\sqsubseteq B$. The above technique essentially computes such a separating hyperplane.

4.5.2 Max-case refinement

Similarly to $\sqsubseteq$, we can verify $A \sqsubseteq^{\max} B$ directly using its definition, by solving the system $R\tilde{A} = \tilde{B}$ under the constraint that R is a channel. As explained for average-case refinement, this can be done in polynomial time.

In contrast to $\sqsubseteq$, when $A \not\sqsubseteq^{\max} B$, the proof of Thm. 4.2 directly gives us a counter-example. Recalling that $A \sqsubseteq^{\max} B$ holds whenever the posteriors of $[\pi \triangleright B]$ are contained in the convex hull of posteriors of $[\pi \triangleright A]$ (for any full support $\pi : \mathbb{D}\mathcal{X}$), we can choose as our vulnerability function Q the following:

$$Q(\sigma) \quad := \quad \min_{\sigma' : S} \|\sigma - \sigma'\|_2 \ .$$

where $S = \mathbf{ch}\lceil[\pi \triangleright A]\rceil$ and π is any full-support prior. ie. We compute the distance of each posterior of $[\pi \triangleright B]$ from the convex hull of posteriors of $[\pi \triangleright A]$. (Note that Q is convex, hence also quasi-convex). For this vulnerability function it therefore holds that $V_Q^{\max}[\pi \triangleright A] < V_Q^{\max}[\pi \triangleright B]$ when $A \not\sqsubseteq^{\max} B$. Again, as for average-case refinement, this can be computed using a quadratic program and solved in polynomial time.

4.5.3 Privacy-based refinement

The $\sqsubseteq^{\mathrm{prv}}$ order can be verified directly from its definition, by checking that $\mathbf{d}_A \geq \mathbf{d}_B$. This can be done in time $O(|\mathcal{X}|^2|\mathcal{Y}|)$, by computing $\mathrm{tv}_{\otimes}(C_{x,-}, C_{x',-})$ for each pair of secrets. If $A \not\sqsubseteq^{\mathrm{prv}} B$, then $\mathbf{d} = \mathbf{d}_B$ provides an immediate counter-example metric, since B satisfies $\mathbf{d}_B$-privacy, but A does not.

4.6 Lattice properties

The orders $\sqsubseteq, \sqsubseteq^{\max}$ and $\sqsubseteq^{\text{prv}}$ are all preorders, which means that they are reflexive and transitive but not anti-symmetric.[8] Anti-symmetry fails because there can exist channels which have "syntactic" differences but the same semantics. eg. In the case of $\sqsubseteq$ we know that 2 different channel matrices A, B (ie. having different columns) can have the same leakage properties (ie. $V_g[\pi \triangleright A] = V_g[\pi \triangleright B]$ for all π, g). In this case we have $A \sqsubseteq B \sqsubseteq A$; however, we can view A and B as the "same channel" by writing their channel matrices in some canonical form, thus turning $\sqsubseteq$ from a preorder into a partial order.[9] We can do the same with $\sqsubseteq^{\max}$ and $\sqsubseteq^{\text{prv}}$, thus turning all of the preorders into partial orders.

Seeing now each of the above orders as a partial order, the natural question is whether it forms a lattice. That is, whether every pair of channels A, B has a unique supremum (least upper bound) $A \vee B$ and a unique infimum (greatest lower bound) $A \wedge B$. If it exists, the supremum $A \vee B$ has an interesting property: it is the "least safe" channel that is safer than both A and B. ie. Satisfying $A \sqsubseteq A \vee B$ and $B \sqsubseteq A \vee B$ (for any of $\sqsubseteq, \sqsubseteq^{\max}, \sqsubseteq^{\text{prv}}$). This is because any channel C such that $A \sqsubseteq C$ and $B \sqsubseteq C$ would necessarily satisfy $A \vee B \sqsubseteq C$. If we wanted a channel that is safer than both A and B, $A \vee B$ would be a natural choice.

In this section we briefly discuss this problem and show that – in contrast to $\sqsubseteq$ – both $\sqsubseteq^{\max}$ and $\sqsubseteq^{\text{prv}}$ do have suprema and infima, and thus form a lattice.

4.6.1 Average-case refinement

In the case of $\sqsubseteq$, "equivalent" channels are those producing exactly the same *hypers*. However, it is known [61] that two channels A, B do not necessarily have a *least upper bound* wrt $\sqsubseteq$, hence $\sqsubseteq$ does not form a lattice. We can also express the supremum as a property of the hypers: the supremum of 2 different channels can be expressed as a refining hyper which anti-refines every other refining hyper (of the channels expressed as hypers). An example illustrating when this fails is given in Example 4.6.

4.6.2 Max-case refinement

In the case of $\sqsubseteq^{\max}$, "equivalent" channels are those producing the same *posteriors* (or, more generally, the same convex hull of posteriors). But, in contrast to $\sqsubseteq$, if we identify such channels – that is, if we represent a channel only by the convex hull of its posteriors – then $\sqsubseteq^{\max}$ becomes a lattice.

First, note that given a finite set of posteriors $P = \{\delta^y | y\}$, such that $\pi \in \mathbf{ch}\{\delta^y\}_y$, i.e. such that $\pi = \sum_y a_y \delta^y$, it is easy to construct a channel C producing each posterior δ^y with output probability a_y. It suffices to take $C_{x,y} := \delta^y_x a_y / \pi_x$.

So $A \vee^{\max} B$ can be simply constructed by taking the intersection of the convex hulls of the posteriors of A, B. This intersection is a convex polytope itself, so it has (finitely many) extreme points, so we can construct $A \vee^{\max} B$ as the channel having exactly those as posteriors. $A \wedge^{\max} B$,

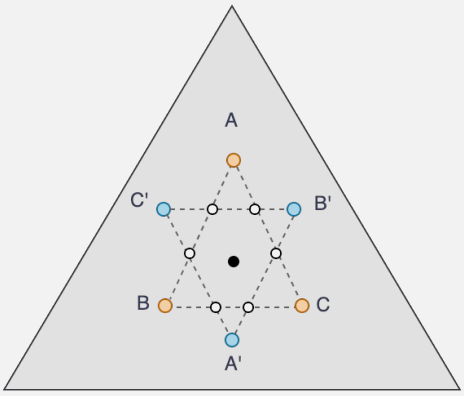
[8] Anti-symmetry means that $A \sqsubseteq B$ implies that $B \not\sqsubseteq A$.

[9] The canonical form is known as an 'abstract channel' in the QIF literature.

on the other hand, can be constructed as the channel having as posteriors the union of those of A and B. An example is illustrated in Figure 4.6.

Note that computing the intersection of polytopes is NP-hard in general [71], so $A \vee^{\max} B$ might be hard to construct. However, efficient special cases do exist [72]; we leave the study of the hardness of $\vee^{\max}$ as future work.

Example 4.6 (Lattice properties for $\sqsubseteq$ and $\sqsubseteq^{\max}$ expressed in the space of hypers.)**.** The following shows the space of hypers on 3 secrets. The orange points A, B, C are posteriors which can be combined (uniquely) to form a valid hyper, say $[\upsilon \triangleright X]$, where υ is the uniform distribution (represented by the black point). Likewise the blue points A', B', C' are posteriors of some hyper $[\upsilon \triangleright Y]$. The convex hulls $\mathbf{ch}\,\lceil[\upsilon \triangleright X]\rceil$ and $\mathbf{ch}\,\lceil[\upsilon \triangleright Y]\rceil$ intersect in the convex region whose vertices are denoted by the 6 white points, which themselves can be combined (non-uniquely) to form hypers (with different weights).



Consider any two, say $[\upsilon \triangleright Z_1]$ and $[\upsilon \triangleright Z_2]$. Since $\sqsubseteq^{\max}$ 'ignores' the weights of the hyper, we have $Z_1 \sqsubseteq^{\max} Z_2 \sqsubseteq^{\max} Z_1$ and so $Z = X \vee^{\max} Y$ is unique (wrt max-case leakage properties) for any choice of Z using the white posteriors. The same argument holds for $X \wedge^{\max} Y$ using hypers formed from the combined blue and orange posteriors.
However, in the case of $\sqsubseteq$, we have $Z_1 \not\sqsubseteq Z_2$ and $Z_2 \not\sqsubseteq Z_1$ since there can be no 'refining Earth Move' between the two hypers. Thus we have no unique infimum since $X \sqsubseteq Z_1$, $Y \sqsubseteq Z_1$ and $X \sqsubseteq Z_2$, $Y \sqsubseteq Z_2$; and the same holds for the supremum.

4.6.3 Privacy-based refinement

In the case of $\sqsubseteq^{\text{prv}}$, "equivalent" channels are those producing the same induced metric, i.e. $\mathbf{d}_A = \mathbf{d}_B$. Representing channels only by their induced metric, we can use the fact than $\mathbb{M}\mathcal{X}$ does form a lattice under $\geq$. We first show that any metric can be turned into a corresponding channel.

Theorem 4.13. For any metric $\mathbf{d}\colon \mathbb{M}\mathcal{X}$, we can construct a channel $C^{\mathbf{d}}$ such that $\mathbf{d}_{C^{\mathbf{d}}} = \mathbf{d}$.

Then $A \vee^{\text{prv}} B$ will be simply the channel whose metric is $\mathbf{d}_A \vee \mathbf{d}_B$, where $\vee$ is the supremum in the lattice of metrics, and similarly for $\wedge^{\text{prv}}$.

Note that the infimum of two metrics $\mathbf{d}_1, \mathbf{d}_2$ is simply the max of the two (which is always a metric). The supremum, however, is more tricky, since the min of two metrics is not always a metric: the triangle inequality might be violated. So we first need to take the min of $\mathbf{d}_1, \mathbf{d}_2$, then compute its "triangle closure", by finding the shortest path between all pairs of elements, for instance using the well-known Floyd-Warshall algorithm.

4.6.4 Constructing d-private mechanisms for any metric d

The proof of Thm. 4.13 is constructive[10], and we now give the construction, which is itself of interest, as it means that we can always construct a **d**-private mechanism for any metric **d**.

> **Lemma 4.14.** Given any (finite) metric space $(\mathcal{X}, \mathbf{d})$, define $\mathcal{Y} = \mathcal{X} \cup \{\bot\}$. Construct the channel $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ as follows:
>
> $$\begin{aligned} C_{x,y} &= |\mathcal{X}|^{-1} e^{-\mathbf{d}(x,y)-1}, && x, y \in \mathcal{X}, \\ C_{x,\bot} &= 1 - |\mathcal{X}|^{-1} \textstyle\sum_{y \in \mathcal{X}} e^{-\mathbf{d}(x,x')-1}, && x \in \mathcal{X}. \end{aligned}$$
>
> Then C is **d**-private.
>
> *Proof.* Follows from proof of Thm. 4.13. □

4.7 Comparing d-Privacy Mechanisms

We noted at the beginning of this chapter that in differential privacy it is common to compare the privacy guarantees provided by different mechanisms by 'comparing the epsilons'. We now use the results from the previous sections to investigate to what extent this is a 'safe' comparison. That is, we would like to know whether reducing the ε value (corresponding to an increase in privacy) also corresponds to increased safety against max-case or average-case adversaries, and whether the same ε value in different mechanisms corresponds to the same level of 'safety' against other types of adversaries. This is useful if, for example, it is important to be able to provide privacy guarantees with respect to other types of adversaries modelled using max-case or average-case attacks.

We observe that, since the ε-based order given by $\sqsubseteq^{\text{prv}}_{\mathbf{d}}$ is (strictly) the weakest of the orders considered here, it *cannot* be the case that reducing ε (eg. by swapping a mechanism with a higher ε for one with a lower ε) is safe against all adversaries (either average- or max-case) – since this is just a refinement. But it may be true that within particular *families* of mechanisms some (or all) of the refinement orders hold.

We investigate 3 families of mechanisms commonly used in differential privacy and **d**-privacy: the *geometric*, *exponential* and *randomised response* mechanisms.

[10] The proof is available in the appendix of [36].

4.7.1 Technical Preliminaries

We define each family of mechanisms in terms of their channel construction. We assume that mechanisms operate on a set of inputs (denoted by $\mathcal{X}$) and produce a set of outputs (denoted $\mathcal{Y}$). In this sense our mechanisms can be seen either as oblivious mechanisms or as local mechanisms. (We use the term 'mechanism' in either sense). We denote by M^ε a mechanism parametrised by ε, where ε is defined to be the same as $\text{Priv}_{\mathbf{d}}(M)$. [11] In order to compare mechanisms, we restrict our input and output domains of interest to sequences of non-negative integers. [12] (We assume $\mathcal{X}, \mathcal{Y}$ are finite unless otherwise specified.) Also, as we are operating in the framework of **d**-privacy, it is necessary to provide an appropriate metric defined over $\mathcal{X}$; here it makes sense to use the Euclidean distance metric $\mathbf{d}_2$.

We recall the following definitions of **d**-privacy mechanisms given in Chapter 3:

> **Definition 3.4.1** (Geometric Mechanism)**.** The α-geometric mechanism $G: \mathcal{X} \to \mathbb{D}\mathbb{Z}$ has the following channel matrix:
>
> $$G_{x,y} = \frac{1-\alpha}{1+\alpha} \cdot \alpha^{\mathbf{d}_2(x,y)}$$
>
> where $\alpha \in (0,1]$. This mechanism satisfies $\varepsilon \cdot \mathbf{d}_2$-privacy where $\mathbf{d}_2$ is the Euclidean metric and $\varepsilon = -\ln \alpha$.

> **Definition 3.4.2** (Truncated Geometric Mechanism)**.** The truncated α-geometric mechanism $TG: \mathcal{X} \to \mathbb{D}\mathcal{X}$ where $\mathcal{X} = \{0, 1, \ldots, n\}$ has the following channel matrix:
>
> $$TG_{x,y} = \frac{1-\alpha}{1+\alpha} \cdot \alpha^{\mathbf{d}_2(x,y)} \qquad \text{for } y \in \{1, \ldots, n-1\}$$
> $$TG_{x,y} = \frac{1}{1+\alpha} \cdot \alpha^{\mathbf{d}_2(x,y)} \qquad \text{for } y \in \{0, n\}$$
>
> where $\alpha \in (0,1]$. This mechanism satisfies $\varepsilon \cdot \mathbf{d}_2$-privacy where $\mathbf{d}_2$ is the Euclidean metric and $\varepsilon = -\ln \alpha$.

We also include in our investigation an 'over-truncated' geometric mechanism whose input space is not entirely included in the output space.

> **Definition 4.7.1.** An over-truncated α-geometric mechanism $OTG: \mathcal{X} \to \mathbb{D}\mathcal{Y}$ where $\mathcal{X} = \{0, 1, \ldots, n\}$ and $\mathcal{Y} \subset \mathbb{Z}$ st. $\mathcal{X} \nsubseteq \mathcal{Y}$ is constructed as follows:
>
> 1. Start with the α-geometric mechanism G.
> 2. Sum up the columns of G at each end until the output domain $\mathcal{Y}$ is reached.
>
> Such a mechanism satisfies $\varepsilon \cdot \mathbf{d}_2$-privacy where $\mathbf{d}_2$ is the Euclidean metric and $\varepsilon = -\ln \alpha$.

[11] We note that the exponential mechanism *under-reports* its ε, thus for the purposes of comparison we make sure that we use the best possible ε for each mechanism.

[12] Our results hold for sequences of quantized integers $q[0..]$ but we use integer sequences to simplify presentation.

For example, the set of inputs to an over-truncated geometric mechanism could be integers in the range $[0 \dots 100]$ but the output space may have a range of $[0 \dots 50]$ or perhaps $[-50 \dots 50]$. In either of these cases, the mechanism has to 'over-truncate' the inputs to accommodate the output space.

We remark that we do not consider the over-truncated mechanism a particularly useful mechanism in practice. However, we provide results on this mechanism for completeness since its construction is possible, if unusual.

We next introduce the exponential mechanism which can be used for any metric but which we consider here wrt the Euclidean metric.

> **Definition 4.7.2.** An ε-exponential mechanism $E : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ is constructed as follows:
>
> $$E_{x,y} = \lambda_x \cdot e^{-\frac{\varepsilon}{2}\mathbf{d}_2(x,y)} \qquad \text{for all } x \in \mathcal{X}, y \in \mathcal{Y}$$
>
> where λ_x are normalising constants ensuring $\sum_y E^{\alpha}_{x,y} = 1$. Such a mechanism satisfies $\alpha \cdot \mathbf{d}_2$-privacy where $\alpha \geq \frac{\varepsilon}{2}$ (which can be calculated exactly from the channel construction).

The exponential mechanism was designed for arbitrary domains, thus the parameter ε does not correspond to the true (best-case) ε privacy guarantee that it provides. We will denote by E^{ε} the exponential mechanism with 'true' privacy parameter ε rather than the reported one, as our intention is to capture the privacy guarantee provided by the channel in order to make reasonable comparisons.

Finally we recall the randomised response mechanism introduced in Chapter 3:

> **Definition 3.4.3** (Randomised Response Mechanism)**.** The α-randomised response mechanism $R : \mathcal{X} \to \mathbb{D}\mathcal{X}$ has the following channel matrix:
>
> $$R_{x,x} = {}^{1}/_{k}$$
> $$R_{x,y} = {}^{\alpha}/_{k} \qquad \text{for } x \neq y$$
>
> where k is a normalisation term and $\alpha \in (0, 1]$. This mechanism satisfies $\varepsilon \cdot \mathbf{d}_{\mathrm{D}}$-privacy where $\mathbf{d}_{\mathrm{D}}$ is the discrete metric and $\varepsilon = -\ln \alpha$.

We note that the randomised response mechanism also satisfies $\varepsilon \cdot \mathbf{d}_2$-privacy.

Intuitively, the randomised response mechanism returns the true answer with high probability and all other responses with equal probability. In the case where the input x lies outside $\mathcal{Y}$ (that is, in 'over-truncated' mechanisms), all of the outputs (corresponding to the outlying inputs) have equal probability. Some examples of mechanisms are illustrated in Example 4.7.

We now have 3 families of mechanisms which we can characterise by channels, and which satisfy $\varepsilon \cdot \mathbf{d}_2$-privacy. For the remainder of this section we will refer only to the ε parameter and

Example 4.7 (Instances of **d**-private mechanisms)**.** The following are examples of each of the mechanisms described above, represented as channel matrices. For this example we set $\varepsilon = \ln 2$ for the geometric and randomised response mechanisms, while for the exponential mechanism we use $\varepsilon = \ln 4$. We assume integer input and output domains. ie. $\mathcal{X} = \{x_1, x_2, x_3\}$ corresponding to $\{1, 2, 3\}$ and similarly for the output.

TG	x_1	x_2	x_3
x_1	$2/3$	$1/6$	$1/6$
x_2	$1/3$	$1/3$	$1/3$
x_3	$1/6$	$1/6$	$2/3$

OTG	x_1	x_2
x_1	$2/3$	$1/3$
x_2	$1/3$	$2/3$
x_3	$1/6$	$5/6$

E	x_1	x_2	x_3
x_1	$4/7$	$2/7$	$1/7$
x_2	$1/4$	$1/2$	$1/4$
x_3	$1/7$	$2/7$	$4/7$

R	x_1	x_2	x_3
x_1	$1/2$	$1/4$	$1/4$
x_2	$1/4$	$1/2$	$1/4$
x_3	$1/4$	$1/4$	$1/2$

Note that the exponential mechanism here actually satisfies $\ln(\frac{16}{7})\cdot\mathbf{d}_2$-privacy even though it is specified by $\varepsilon = \ln 4$.

take $\mathbf{d}_2$ as given, as our goal is to understand the effect of changing ε (for a fixed metric) on the various leakage measures.

> **Remark 4.7.1.** Note that for the remainder of our discussions on **d**-private mechanisms we will typically parametrise mechanisms by ε rather than α for the purposes of comparison. We will denote this by, for example, G^ε for the geometric mechanism parametrised by ε. The use of α in some definitions is more for convenience of notation.

4.7.2 Refinement order within families of mechanisms

We first ask which refinement orders hold within a family of mechanisms. That is, when does reducing ε for a particular mechanism produce a refinement? Since we have the convenient order $\sqsubseteq \subset \sqsubseteq^{\max} \subset \sqsubseteq^{\text{prv}}$ it is useful to first check if $\sqsubseteq$ holds as we get the other refinements 'for free'.

We first make the observation that the (infinite) geometric mechanism and the truncated geometric mechanism are related by refinement:

> **Lemma 4.15.** Let G^ε and TG^ε be geometric and truncated geometric channels respectively, defined over the same domain of secrets $\mathcal{X}$. Then $G^\varepsilon \sqsubseteq TG^\varepsilon$ and $TG^\varepsilon \sqsubseteq G^\varepsilon$. That is, the geometric and truncated geometric mechanisms are *equivalent* under refinement for the same ε.

This means that it is *always safe* to truncate the geometric mechanism (ie. against any

adversaries) since the operation of truncation[13] does not affect the leakage properties of the channel wrt average-case adversaries (and therefore also wrt max-case and differential privacy adversaries).

For the (infinite) geometric mechanism we have the following result.

THEOREM 4.16. Let $G^{\varepsilon}, G^{\varepsilon'}$ be geometric mechanisms. Then $G^{\varepsilon} \sqsubseteq G^{\varepsilon'}$ iff $\varepsilon \geq \varepsilon'$. That is decreasing ε produces a refinement of the mechanism.

This means that reducing ε in an infinite geometric mechanism is safe against *any* adversary that can be modelled using, for example, max-case or average-case vulnerabilities.

For the truncated geometric mechanism, the same result follows from Lem. 4.15 and Thm. 4.16.

COROLLARY 4.17. Let $TG^{\varepsilon}, TG^{\varepsilon'}$ be truncated geometric mechanisms. Then $TG^{\varepsilon} \sqsubseteq TG^{\varepsilon'}$ iff $\varepsilon \geq \varepsilon'$. That is, decreasing ε produces a refinement of the mechanism.

However, the over-truncated geometric mechanism does not behave as nicely.

LEMMA 4.18. Let $OTG^{\varepsilon}, OTG^{\varepsilon'}$ be over-truncated geometric mechanisms. Then $OTG^{\varepsilon} \not\sqsubseteq OTG^{\varepsilon'}$ for any $\varepsilon \neq \varepsilon'$. That is, decreasing ε does **not** produce a refinement.

Proof. Consider the following counter-example:

A^{ε}	x_1	x_2
x_1	$4/5$	$1/5$
x_2	$1/5$	$4/5$
x_3	$1/20$	$19/20$

$B^{\varepsilon'}$	x_1	x_2
x_1	$2/3$	$1/3$
x_2	$1/3$	$2/3$
x_3	$1/6$	$5/6$

Channels A^{ε} and $B^{\varepsilon'}$ are over-truncated geometric mechanisms parametrised by $\varepsilon = 2\ln 2$, $\varepsilon' = \ln 2$ respectively. We expect $B^{\varepsilon'}$ to be safer than A^{ε}, that is, $V_G[\pi \triangleright B^{\varepsilon'}] < V_G[\pi \triangleright A^{\varepsilon}]$ for any $\pi \in \mathbb{D}\mathcal{X}$. However, under the uniform prior υ, the gain function

G	w_1	w_2
x_1	$1/5$	0
x_2	0	1
x_3	$4/5$	0

yields $V_G[\upsilon \triangleright A^{\varepsilon}] = 0.33$ and $V_G[\upsilon \triangleright B^{\varepsilon'}] = 0.36$, thus $B^{\varepsilon'}$ leaks more than A^{ε} for this adversary. (In fact, for this gain function we have $V_G[\upsilon \triangleright A^{\varepsilon}] = V_G(\upsilon)$ and so the adversary learns nothing from observing the output of A^{ε}). □

Intuitively, this means that we can *always* find some (average-case) adversary who prefers the over-truncated geometric mechanism with the smaller ε.

[13] By truncation we mean the standard operation of truncating the geometric mechanism until the input and output domains match. This is *not* true for over-truncated mechanisms.

We remark that the gain function we found can be easily calculated by treating the columns of channel A as vectors, and finding a vector orthogonal to both of these. This follows from the results in Section 4.5.1. Since the columns of A cannot span the space $\mathbb{R}^3$ it is always possible to find such a vector, and when this vector is not orthogonal to the 'column space' of B it can be used to construct a gain function preferring B to A.

Even though the $\sqsubseteq$ refinement does not hold, we can check whether the other refinements are satisfied.

> **LEMMA 4.19.** Let OTG^{ε} be an over-truncated geometric mechanism. Then reducing ε does **not** produce a $\sqsubseteq^{\max}$ refinement, however it **does** produce a $\sqsubseteq^{\text{prv}}$ refinement.

This means that although a smaller ε does not provide safety against all max-case adversaries, it *does* produce a safer mechanism wrt **d**-privacy for *any* choice of metric we like.

Intuitively, the $\sqsubseteq^{\text{prv}}$ order relates mechanisms based on how they distinguish *inputs*. Specifically, if $A \sqsubseteq^{\text{prv}} B$ then for any pair of inputs x, x', the corresponding output distributions are 'further apart' in channel A than in channel B, and thus the inputs are more distinguishable using channel A. When $\sqsubseteq^{\text{prv}}$ fails to hold, it means that there are some inputs in A which are more distinguishable than in B, and vice versa. This means an adversary who is interested in distinguishing some particular pair of inputs would prefer one mechanism to the other.

We now consider the exponential mechanism. In this case, we do not have a theoretical result, but experimentally it appears that the exponential mechanism respects refinement, so we present the following conjecture.

> **CONJECTURE 4.20.** Let E^{ε} be an exponential mechanism. Then decreasing ε in E produces a refinement. That is, $E^{\varepsilon} \sqsubseteq E^{\varepsilon'}$ iff $\varepsilon \geq \varepsilon'$.

Finally, we consider the randomised response mechanism.

> **THEOREM 4.21.** Let R^{ε} be a randomised response mechanism. Then decreasing ε in R produces a refinement. That is, $R^{\varepsilon} \sqsubseteq R^{\varepsilon'}$ iff $\varepsilon \geq \varepsilon'$.

In conclusion, we can say that, in general, the usual **d**-privacy families of mechanisms are 'well-behaved' wrt all of the refinement orders. This means that it is safe (wrt *any* adversary we model here) to replace a mechanism from a particular family with another mechanism from the *same* family with a lower ε.

4.7.3 Refinement order between families of mechanisms

Now, we explore whether it is possible to compare mechanisms from different families. We first ask: can we compare mechanisms which have the same ε? We assume that the input and output domains are the same, and the intention is to decide whether to replace one mechanism with another.

THEOREM 4.22. Let R^ε be a randomised response mechanism, E^ε an exponential mechanism and TG^ε a truncated geometric mechanism. Then $TG^\varepsilon \sqsubseteq^{\text{prv}} R^\varepsilon$ and $TG^\varepsilon \sqsubseteq^{\text{prv}} E^\varepsilon$. However $\sqsubseteq^{\text{prv}}$ does not hold between E^ε and R^ε.

Proof. We present a counter-example to show $E^\varepsilon \not\sqsubseteq^{\text{prv}} R^\varepsilon$ and $R^\varepsilon \not\sqsubseteq^{\text{prv}} E^\varepsilon$. The remainder of the proof can be found in [36].

Consider the following channels:

A	x_1	x_2	x_3	x_4
x_1	$8/15$	$4/15$	$2/15$	$1/15$
x_2	$2/9$	$4/9$	$2/9$	$1/9$
x_3	$1/9$	$2/9$	$4/9$	$2/9$
x_4	$1/15$	$2/15$	$4/15$	$8/15$

B	x_1	x_2	x_3	x_4
x_1	$4/9$	$5/27$	$5/27$	$5/27$
x_2	$5/27$	$4/9$	$5/27$	$5/27$
x_3	$5/27$	$5/27$	$4/9$	$5/27$
x_4	$5/27$	$5/27$	$5/27$	$4/9$

Channel A represents an exponential mechanism and channel B a randomised response mechanism. Both have (true) ε of $\log(12/5)$. [14] However $\mathbf{d}_A(x_1, x_3) > \mathbf{d}_B(x_1, x_3)$ and $\mathbf{d}_A(x_2, x_3) < \mathbf{d}_B(x_2, x_3)$. Thus A does not satisfy $\mathbf{d}_B$-privacy, nor does B satisfy $\mathbf{d}_A$-privacy. □

Intuitively, the randomised response mechanism maintains the same (ε) distinguishability level between inputs, whereas the exponential mechanism causes some inputs to be *less* distinguishable than others. This means that, for the same (true) ε, an adversary who is interested in certain inputs could learn more from the randomised response than the exponential. In the above counter-example, inputs x_2, x_3 in the exponential mechanism of channel A are *less* distinguishable than the corresponding inputs in the randomised response mechanism B.

As an example, imagine that the mechanisms are to be used in geo-location privacy and the inputs represent adjacent locations (such as addresses along a street). Then an adversary (your boss) may be interested in how far you are from work, and therefore wants to be able to distinguish between points distant from x_1 (your office) and points within the vicinity of your office, without requiring your precise location. Your boss chooses channel A as the most informative. However, another adversary (your suspicious partner) is more concerned about where exactly you are, and is particularly interested in distinguishing between your expected position (x_2, the boulangerie) versus your suspected position (x_3, the brothel). Your partner chooses channel B as the most informative.

Regarding the other refinements, we find (experimentally) that *none* of them hold (in general) between families of mechanisms. [15]

We next check what happens when we compare mechanisms with *different* epsilons. We note the following.

THEOREM 4.23. For any (truncated geometric, randomised response, exponential) mechanisms $M_1^\varepsilon, M_2^{\varepsilon'}$, if $M_2^{\varepsilon'} \sqsubseteq M_1^\varepsilon$ for any of our refinements ($\sqsubseteq, \sqsubseteq^{\max}, \sqsubseteq^{\text{prv}}$) then $M_2^{\varepsilon'} \sqsubseteq M_1^{\varepsilon_2}$ for $\varepsilon_2 < \varepsilon$.

[14] Channel A was generated using $\varepsilon = \log(4)$. However, as noted earlier, this corresponds to a lower *true* ε.

[15] Recall that we only need to produce a single counter-example to show that a refinement doesn't hold, and this can be done using the methods presented in Section 4.5.

	Are these valid for decreasing ε?		
Mechanism	$\sqsubseteq$	$\sqsubseteq^{\max}$	$\sqsubseteq^{\text{prv}}$
Geometric	Y	Y	Y
Truncated Geometric	Y	Y	Y
Over-Truncated Geometric	N	N	Y
Exponential	Y	Y	Y
Randomised Response	Y	Y	Y

Table 4.2: The refinements respected by families of mechanisms for decreasing ε.

Proof. This follows directly from transitivity of the refinement relations, and our results on refinement with families of mechanisms. (We recall, however, that our result for the exponential mechanism is only a conjecture.) □

This tells us that once we have a refinement between mechanisms, it continues to hold for reduced ε in the refining mechanism.

COROLLARY 4.24. Let TG, R, E be truncated geometric, randomised response and exponential mechanisms respectively. Then for all $\varepsilon' \leq \varepsilon$ we have that $TG^{\varepsilon} \sqsubseteq^{\text{prv}} R^{\varepsilon'}$ and $TG^{\varepsilon} \sqsubseteq^{\text{prv}} E^{\varepsilon'}$.

So it is safe to 'compare epsilons' wrt $\sqsubseteq^{\text{prv}}$ if we want to replace a geometric mechanism with either a randomised response or exponential mechanism. (As with the previous theorem, note that the results for the exponential mechanism are stated as conjecture only, and this conjecture is assumed in the statement of this corollary.) What this means is that if, for example, we have a geometric mechanism TG that operates on databases with distance measured using the Hamming metric $\mathbf{d}_{\text{H}}$ and satisfying $\varepsilon \cdot \mathbf{d}_{\text{H}}$-privacy, then any randomised response mechanism R parametrised by $\varepsilon' \leq \varepsilon$ will also satisfy $\varepsilon \cdot \mathbf{d}_{\text{H}}$-privacy. Moreover, if we decide we'd rather use the Manhattan metric $\mathbf{d}_{\text{M}}$ to measure distance between the databases, then we only need to check that TG also satisfies $\varepsilon \cdot \mathbf{d}_{\text{M}}$-privacy, as this implies that R will too.

The following tables 4.2, 4.3, and 4.4 summarise the refinement relations with respect to the various families of mechanisms.

4.7.4 Asymptotic behaviour

We now consider the behaviour of the relations when ε approximates 0, which represents the absence of leakage. We start with the following result:

THEOREM 4.25. Every (truncated geometric, randomised response, exponential) mechanism is 'the safest possible mechanism' when parametrised by $\varepsilon = 0$. That is $L^{\varepsilon} \sqsubseteq M^{0}$ for all mechanisms L, M (possibly from different families) and $\varepsilon > 0$.

While this result may be unsurprising, it means that we know that refinement must *eventually*

Refinements across families with same ε		
$TG \not\sqsubseteq R$	$TG \not\sqsubseteq^{\max} R$	$TG \sqsubseteq^{\text{prv}} R$
$R \not\sqsubseteq TG$	$R \not\sqsubseteq^{\max} TG$	$R \not\sqsubseteq^{\text{prv}} TG$
$TG \not\sqsubseteq E$	$TG \not\sqsubseteq^{\max} E$	$TG \sqsubseteq^{\text{prv}} E$
$E \not\sqsubseteq TG$	$E \not\sqsubseteq^{\max} TG$	$E \not\sqsubseteq^{\text{prv}} TG$
$G \not\sqsubseteq R$	$G \not\sqsubseteq^{\max} R$	$G \sqsubseteq^{\text{prv}} R$
$R \not\sqsubseteq G$	$R \not\sqsubseteq^{\max} G$	$R \not\sqsubseteq^{\text{prv}} G$
$G \not\sqsubseteq E$	$G \not\sqsubseteq^{\max} E$	$\mathrm{G} \sqsubseteq^{\text{prv}} E$
$E \not\sqsubseteq G$	$E \not\sqsubseteq^{\max} G$	$E \not\sqsubseteq^{\text{prv}} G$
$R \not\sqsubseteq E$	$R \not\sqsubseteq^{\max} E$	$R \not\sqsubseteq^{\text{prv}} E$
$E \not\sqsubseteq R$	$E \not\sqsubseteq^{\max} R$	$E \not\sqsubseteq^{\text{prv}} R$

Table 4.3: Comparing different families of mechanisms with respect to the different refinements under the same ε.

Comparison of refinements with $\varepsilon_1 > \varepsilon_2$		
$TG^{\varepsilon_1} \not\sqsubseteq R^{\varepsilon_2}$	$TG^{\varepsilon_1} \not\sqsubseteq^{\max} R^{\varepsilon_2}$	$TG^{\varepsilon_1} \sqsubseteq^{\text{prv}} R^{\varepsilon_2}$
$R^{\varepsilon_1} \not\sqsubseteq TG^{\varepsilon_2}$	$R^{\varepsilon_1} \not\sqsubseteq^{\max} TG^{\varepsilon_2}$	$R^{\varepsilon_1} \not\sqsubseteq^{\text{prv}} TG^{\varepsilon_2}$
$TG^{\varepsilon_1} \not\sqsubseteq E$	$TG^{\varepsilon_1} \not\sqsubseteq^{\max} E^{\varepsilon_2}$	$TG^{\varepsilon_1} \sqsubseteq^{\text{prv}} E^{\varepsilon_2}$
$E^{\varepsilon_1} \not\sqsubseteq TG$	$E^{\varepsilon_1} \not\sqsubseteq^{\max} TG^{\varepsilon_2}$	$E^{\varepsilon_1} \not\sqsubseteq^{\text{prv}} TG^{\varepsilon_2}$
$G^{\varepsilon_1} \not\sqsubseteq R^{\varepsilon_2}$	$G^{\varepsilon_1} \not\sqsubseteq^{\max} R^{\varepsilon_2}$	$G^{\varepsilon_1} \sqsubseteq^{\text{prv}} R^{\varepsilon_2}$
$R^{\varepsilon_1} \not\sqsubseteq G^{\varepsilon_2}$	$R^{\varepsilon_1} \not\sqsubseteq^{\max} G^{\varepsilon_2}$	$R^{\varepsilon_1} \not\sqsubseteq^{\text{prv}} G^{\varepsilon_2}$
$G^{\varepsilon_1} \not\sqsubseteq E^{\varepsilon_2}$	$G^{\varepsilon_1} \not\sqsubseteq^{\max} E^{\varepsilon_2}$	$G^{\varepsilon_1} \sqsubseteq^{\text{prv}} E^{\varepsilon_2}$
$E^{\varepsilon_1} \not\sqsubseteq G^{\varepsilon_2}$	$E^{\varepsilon_1} \not\sqsubseteq^{\max} G^{\varepsilon_2}$	$E^{\varepsilon_1} \not\sqsubseteq^{\text{prv}} G^{\varepsilon_2}$
$R^{\varepsilon_1} \not\sqsubseteq E^{\varepsilon_2}$	$R^{\varepsilon_1} \not\sqsubseteq^{\max} E^{\varepsilon_2}$	$R^{\varepsilon_1} \not\sqsubseteq^{\text{prv}} E^{\varepsilon_2}$
$E^{\varepsilon_1} \not\sqsubseteq R^{\varepsilon_2}$	$E^{\varepsilon_1} \not\sqsubseteq^{\max} R^{\varepsilon_2}$	$E^{\varepsilon_1} \not\sqsubseteq^{\text{prv}} R^{\varepsilon_2}$

Table 4.4: Comparing different families of mechanisms with differing ε.

occur when we reduce ε. It is interesting then to ask just *when* this refinement occurs. We examine this question experimentally by considering different mechanisms and investigating for which values of ε average-case refinement holds. For simplicity of presentation, we show results for 5×5 matrices, noting that we observed similar results for experiments across different matrix dimensions. [16] The results are plotted in Figure 4.5.

The plots show the relationship between ε_1 (x-axis) and ε_2 (y-axis) where ε_1 parametrises the mechanism *being refined* and ε_2 parametrises the refining mechanism. For example, the blue line on the top graph represents $TG^{\varepsilon_1} \sqsubseteq E^{\varepsilon_2}$. We fix ε_1 and ask for what value of ε_2 do we get

[16] By similar results, we mean wrt the coarse-grained comparison of plots that we do here.

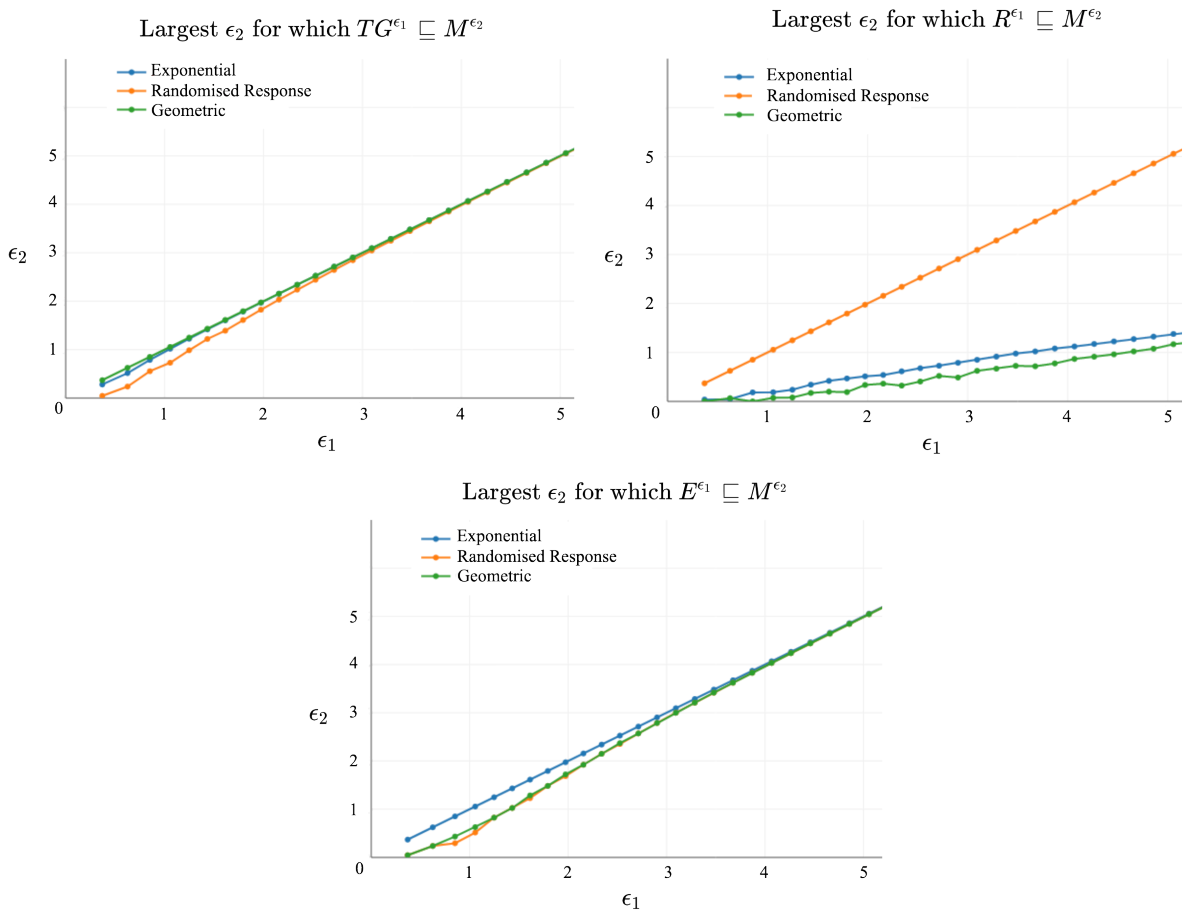


Figure 4.5: Refinement of mechanisms under $\sqsubseteq$ for 5×5 channels. The x-axis (ε_1) represents the mechanism on the LHS of the relation. The y-axis (ε_2) represents the refining mechanism. We fix ε_1 and ask for what value of ε_2 do we get a refinement? The top left graph represents refinement of the truncated geometric mechanism (that is, $TG \sqsubseteq$), the top right graph is refinement of randomised response ($R \sqsubseteq$), and the bottom graph is refinement of the exponential mechanism ($E \sqsubseteq$).

a $\sqsubseteq$ refinement? Notice that the line $\varepsilon_1 = \varepsilon_2$ corresponds to the same mechanism in both axes (since every mechanism refines itself).

We can see that refining the randomised response mechanism requires much smaller values of epsilon in the other mechanisms. For example, from the middle graph we can see that $R^4 \sqsubseteq TG^1$ (approximately) whereas from the top graph we have $TG^4 \sqsubseteq R^4$. This means that the randomised response mechanism is very 'safe' against average-case adversaries compared with the other mechanisms, as it is much more 'difficult' to refine than the other mechanisms.

We also notice that for 'large' values of ε_1, the exponential and geometric mechanisms refine each other for approximately the same ε_2 values. This suggests that for these values, the epsilons are comparable (that is, the mechanisms are equally 'safe' for similar values of ε). However, smaller values of ε_1 require a (relatively) large reduction in ε_2 to obtain a refinement.

4.8 Conclusion

We have investigated various refinement orders for mechanisms for information protection, combining the max-case perspective typical of differential privacy and its variants with the robustness of the QIF approach. We have provided structural characterisations of these preorders and methods to verify them efficiently. Then we have considered various differential privacy mechanisms, and investigated the relation between the ε-based measurement of privacy and our orders. We have shown that, while within the same family of mechanisms a smaller ε implies the refinement order, this is almost never the case for mechanisms belonging to different families.

Although we found that the comparison of mechanisms by "comparing the epsilons" fails to preserve average-case refinement, an interesting question is whether or not the failure of refinement corresponds to a realistic adversarial threat. We leave this exploration to future work.

4.9 Chapter Notes

This chapter is based on the published papers "Comparing Systems: Max-case Refinement Orders and Application to Differential Privacy" [35] and "Refinement Orders for Quantitative Information Flow and Differential Privacy" [36]. Proofs not included in this chapter can be found in the appendix of [36].

In the field of differential privacy there have been various works aimed at trying to understand the operational meaning of the privacy parameter ε and at providing guidelines for the choice of its values. We mention for example [27] and [73], which consider the value of ε from an economical point of view in terms of cost. We are not aware however of studies aimed at establishing orders between the level of privacy of different mechanisms, except the one based on the comparison of the ε's.

We are also not aware of many other studies on refinement relations for QIF. Yasuoka and Terauchi [74] and Malacaria [75] have explored strong orders on *deterministic* mechanisms, focusing on the fact that such mechanisms induce *partitions* on the space of secrets. They showed that the orders produced by min-entropy leakage [55] and Shannon leakage [76, 77] are the same and, moreover, they coincide with the *partition refinement* order in the *Lattice of Information* [78]. This order was extended to the probabilistic case in [56], resulting in the average-case leakage order (introduced in Chapter 2). The max-case leakage, on which the relation $\sqsubseteq_{\mathbb{Q}}^{\max}$ is based, was introduced in [57], but $\sqsubseteq_{\mathbb{Q}}^{\max}$ and its properties were not investigated.

5

Inference Attacks

The fundamental goal of a privacy-protection system is to ensure that any data release does not violate the privacy of individuals whose sensitive information is used in such a release. Differential privacy provides such guarantees couched in terms of *indistinguishability* – an individual can plausibly deny that their personal information contributed to some announced value v because this value could have been produced just as plausibly from some other data v' not containing that individual's information. This idea extends to arbitrary metric spaces; metric differential privacy provides guarantees that an adversary's posterior knowledge about the value of a secret does not change 'much' if the adversary has some uncertainty about the precise value of the secret within some 'radius' of the true value.

In contrast, in practical applications (for example, machine learning settings), the privacy-preserving properties of a system are usually evaluated empirically in terms of their vulnerability to *inference attacks* (see, for example, [18] and [19]). Moreover, practitioners are required to tune their systems to optimise for accuracy (of some utility measure). This is typically done by varying the ε parameter until the desired utility has been achieved, without a good understanding of the ramifications for privacy [79].

Differential privacy does not rule out inferences – indeed, its definition was predicated upon the observation that inferences cannot be eliminated since they are critical for the release of *useful* information [80]. However, from the perspective of a privacy practitioner, an understanding of differential privacy's guarantees regarding inference attacks, as well as the role of ε in managing the privacy-utility balance, is essential for the proper design and evaluation of a privacy-preserving system.

In this chapter we study the privacy-utility trade-off from the point of view of inference attacks. We use QIF to model inferences as correlations in order to analyse the extent to which

inferences succeed once data has been released, and to estimate the accuracy of "useful" queries. From this viewpoint we are able to gauge direct risks to individuals when compared to the benefits of those wishing to use the data, as well as some understanding about the setting of the parameter ε in terms of the trade-off between privacy risks and accuracy of data releases.

5.1 Motivation

Consider the following stories about actual and potential privacy breaches.

(i) *In 2009 a lawsuit (Doe v. Netflix)* was filed against Netflix alleging that it had violated fair-trade laws and a federal privacy law. The complainants argued that Netflix's "anonymisation" still allowed individuals to be identified (and indeed plenty were) by combining movie preferences with other generally available data. And that Netflix should have known about these risks.

[Source: https://www.wired.com/2009/12/netflix-privacy-lawsuit/]

(ii) *Also in 2009, researchers in Natural Language Processing* showed that using powerful machine learning algorithms, authors of anonymously-penned documents could be identified with a high degree of accuracy.

[Source: On the Feasibility of Internet-Scale Author Identification [81]]

(iii) Finally *Privacy researchers at the University of Melbourne* noted that in any "anonymised datasets" records that can be characterised uniquely put the individuals who contributed those records at risk. "While uniqueness does not imply re-identification, particular data that is known to be held by certain parties, does imply the opportunity for re-identification". [Source: https://pursuit.unimelb.edu.au/articles/the-simple-process-of-re-identifying-patients-in-public-health-records]

These three real examples share a common theme: the breaches, or the potential for breaches, are all related to "unintended inferences," meaning that the data published was deemed to be innocuous but turned out to have the potential for inferring information that individuals could claim as infringements to their privacy.

Modern approaches to privacy such as differential privacy recognise that inferences are a problem and, whilst they likely cannot be eliminated, can be mitigated by providing "plausible deniability" for individuals. However, despite the abundance of variations on the theme of differential privacy designed for different scenarios[16], there remains little clarification around the basic questions: how does some particular version of differential privacy (and its choice of ε) affect *both* the potential for unintended inferences to be made of the individuals contributing their data *and* the accuracy of the "useful" data that is released?

Our goal in this chapter is to explore privacy and utility for differential privacy systems from the perspective of a data curator interested in protecting against inference attacks. Significant in this analysis is a contribution to the "no free lunch" style theorems of Kifer et al. [37]. We

demonstrate that it is impossible to design differentially-private mechanisms that offer minimal risk benefits to all individuals in all situations at the same time as preserving a fixed accuracy threshold. And dually it is impossible to design differentially-private mechanisms that offer optimal accuracy in all scenarios whilst guaranteeing a fixed threshold for plausible deniability for all individuals.

The implication of these impossibility results is that the manner in which noisy outputs are created is crucial, and by making reasonable assumptions about adversaries, as argued by Kifer at al. [37], a better understanding of the relationship between inferences about privacy and accuracy can be obtained. Our final contribution is to compare experimentally the different risks to privacy versus accuracy in a large to medium-sized dataset when differential privacy is used as the noise-adding mechanism.

5.2 Informal example: does it matter how to randomise?

In this section we illustrate, using a simple scenario, the challenges faced by designers of privacy mechanisms in deciding what and how to randomise.

Suppose there are N participants invited to contribute to a survey. The survey question is of a personal nature and some respondents might be embarrassed if their true response came to be known. However the organisers of the survey only want to know the total number of (say) "yes" replies. Figure 5.1 and Figure 5.2 are two possible designs for collecting the responses and announcing a count of the total "yes" responses. Figure 5.1 is the well-known "randomised response" protocol which was designed to give respondents plausible deniability so that even the data collectors do not know exactly whether the response they receive from any individual is the "true" response [6]. Figure 5.2 follows the blueprint of traditional "oblivious" privacy mechanisms. An accurate tally of the true answers of the respondents is computed, and then some randomness is added. In this particular example we use the geometric distribution.[1]

In both cases there is a corresponding differential privacy guarantee. In Figure 5.1 the guarantee grants plausible deniability wrt. a $e^{\log 3}$ threshold for each respondent individually. For Figure 5.2 the differential-privacy guarantee is not handed down (directly at least) to an individual, but rather gives a guarantee on the plausible deniability for the final output. In this case more work is required to determine the privacy risk to an individual, but it is relatively easy to provide a guarantee of accuracy: that's because the randomisation is added to the *useful* output, namely the true answer to the query and there are strong results which show that good accuracy can be guaranteed using the geometric distribution for this type of data release [39]. In contrast, for Figure 5.1, as mentioned above, the randomness is added directly to the data that is considered to be private (by the owner of that data), and so in this scenario the survey organisers would want to know the effect on the accuracy of the final count, which also requires more work to compute.

Much of the theoretical analysis of privacy mechanisms is carried out within the context of

[1] That is, the geometric mechanism (Def. 3.4.1), which adds noise drawn from a geometric distribution to the secret.

```
// Assume resp is an array of length N set to
// participants' responses to a survey question.
i := 0;
count:= 0;
while (i<N) {
     coin:= 0 [1/2] 1;        // Random response - coin flip
     count:= (count + coin
                  [1/2]       // Randomly include or not
              count + resp[i]);
     i++;
}
print count;   // Announce the approximate count
```

Assume that all variables cannot be observed by an adversary except for the final "print" of the count. On each iteration, the participant i is randomly selected for inclusion in the count or not. In the case that the participant's true response resp[i] is not included, a random response *coin* for that participant is delivered instead.
This mechanism $\mathcal{R}$ is able to guarantee the following differential-privacy constraint for each participant i providing plausible deniability for their true response:

$$\mathcal{R}(\text{resp[i]=0})(Z) \leq \mathcal{R}(\text{resp[i]=1})(Z)e^{\log 3} \ .$$

Figure 5.1: Randomised response with `N` participants

an adversarial model. In the case of differential privacy that model is deliberately worst-case: namely it is assumed that the adversary knows *all the data* except that of a given individual. Within that setting though, it seems not straightforward to examine vulnerabilities in regard to "unintended inferences" or the potential for such inferences as described by (iii) above. Moreover, Chatzikokolakis et al. [35] have shown that such adversarial models offer surprisingly weak guarantees of privacy operating against other reasonable adversarial settings.

In the case of trying to decide whether to use Figure 5.1 or Figure 5.2 if we use the weak differential privacy adversarial setting, a designer would be unable to determine how randomness might defend against an actual privacy breach ie. an unintended inference. When focussing on randomness as a defence, relevant issues are not only that respondents have plausible deniability, but also what level of ε to use that balances the risk of an unintended inference with a reasonably accurate tally of "yes" respondents, and what might be the adversary's prior knowledge.

In the remaining sections we analyse the potential for inferences using a model of Quantitative Information Flow (QIF), and adversarial settings which include assumptions about an adversary's prior knowledge.

```
// Assume resp is an array of length N set to
// participants' responses to a survey question.
// epsilon is a parameter for randomising the final tally.
i := 0;
tally := 0;
while (i<N) {
     tally := tally + resp[i];
     i++;
}
count := Geom(tally,  epsilon);
print count;   // Announce the approximate count
```

Assume that all variables cannot be observed by an adversary except for the final "print" of the count. On each iteration, the participant i's response resp[i] is included in the count. After the full count has been tallied, the result is randomised according to the (truncated) Geometric mechanism. This mechanism $\mathcal{G}$ is able to guarantee the following differential-privacy constraint for the value of the tally, for $\varepsilon = \log 3$:

$$\mathcal{G}(\mathsf{tally} = k)(Z) \leq \mathcal{G}(\mathsf{tally} = k{+}1)(Z)e^{\log 3} \ .$$

Figure 5.2: Oblivious response mechanism to announce the total for `N` participants

5.3 Modelling Inferences

An inference is commonly regarded as the ability to determine a property about an individual or a group of individuals using any available information. Thus the inferred property might not be obvious, which is why almost all data releases are vulnerable, to some extent, from inference attacks. In this section we show how to use a QIF model to study such vulnerabilities.

5.3.1 Quantitative Information Flow

We recall from Chapter 2 that we model a secret as a distribution of type $\mathbb{D}\mathcal{X}$ where $\mathcal{X}$ is a finite set of possible secret values.[2] Given $\pi\colon \mathbb{D}\mathcal{X}$ we write π_x for the probability that π assigns to $x\colon \mathcal{X}$. We model privacy mechanisms as probabilistic channels $C\colon \mathcal{X} \to \mathbb{D}\mathcal{Y}$ which are row-stochastic matrices mapping secret values $x\colon \mathcal{X}$ to observations $y\colon \mathcal{Y}$.

A *gain function* $g\colon \mathcal{W}\times\mathcal{X} \to \mathbb{R}$ models how an adversary uses their knowledge to gain some advantage wrt an attack scenario by choosing an action $w\colon \mathcal{W}$ when the true value of the secret is $x\colon \mathcal{X}$. Given a gain function we define the *vulnerability* of a secret in $\mathbb{D}\mathcal{X}$ relative to the scenario it describes: it is the maximum average gain to an adversary. A simple example of a gain function is bv[3], where $\mathcal{W} := \mathcal{X}$, and

$$\mathsf{bv}(x, x') \quad := \quad \begin{cases} 1 & \text{if } x = x', \\ 0 & \text{otherwise.} \end{cases} \tag{5.1}$$

For this scenario, the adversary's goal is to determine the exact value of the secret, so he receives

[2] In Chapter 2 we called this the adversary's uncertainty about the secret value, but these ideas are equivalent.
[3] This gain function coincides with the identity gain function which we encountered in Chapter 2 (Def. 2.2.3).

a gain of 1 if he correctly guesses the value of a secret, and zero otherwise. Assuming that he knows the range of possible secrets $\mathcal{X}$, he therefore has $\mathcal{W} := \mathcal{X}$ for his set of possible guesses.

We recall the definition of prior g-vulnerability from §2.2.1:

> **Definition 2.2.1.** Given a gain function $g : \mathbb{G}\mathcal{X}$, the g-vulnerability of a prior $\pi : \mathbb{D}\mathcal{X}$ is
>
> $$V_g(\pi) \quad := \quad \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi_x g(w, x) \; .$$
>
> When $\mathcal{W}$ is infinite, max should be replaced by sup.

Given a prior $\pi : \mathbb{D}\mathcal{X}$ and a mechanism $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, the adversary's posterior knowledge about the secrets is modelled as the average g-vulnerability of the induced posterior distributions, which we recall is defined as follows:

> **Definition 2.2.5.** Given a gain function $g : \mathbb{G}\mathcal{X}$, a prior $\pi : \mathbb{D}\mathcal{X}$ and a channel C, the posterior g-vulnerability $V_g[\pi \triangleright C]$ is defined as
>
> $$V_g[\pi \triangleright C] \quad := \quad \sum_y a_y V_g(\delta^y)$$
>
> where we write a_y for the marginal probability on observation y and δ^y for its associated posterior.

The prior and posterior vulnerabilities for the gain function bv are called prior and posterior *Bayes* vulnerabilities, denoted V_1, and are useful for modelling the vulnerability of a channel against an adversary who wishes to guess the secret in one try.

We also recall the notion of *refinement* (§2.3) which allows us to compare mechanisms wrt their leakage properties — if one mechanism C is more vulnerable than another D under all priors and gain functions, then we say that D is *more secure* than C, written $C \sqsubseteq D$. The refinement order is very robust, having both an operational interpretation (in terms of average-case leakage – cfr. Def. 2.3.1) and a structural characterisation (Def. 2.3.2) which says that $C \sqsubseteq D$ iff D can be written as $C \cdot R$ for some channel R. In this way, refinement also characterises post-processing – the witness R describes a 'remap' of outputs from C to inputs of D. Not surprisingly it can be shown that $C \sqsubseteq C \cdot D$, but perhaps surprisingly if $C \sqsubseteq D$ then there is some post-processing mechanism E such that $C \cdot E = D$.

Dalenius Vulnerability

We now introduce the notion of *Dalenius vulnerability*, which allows us to model the leakage of secrets through unexpected correlations.[4] We model a correlation between sets $\mathcal{X}$ and $\mathcal{Z}$ as a joint distribution $\Pi : \mathbb{D}(\mathcal{X} \times \mathcal{Z})$. Given a channel $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, we can define a channel $C^\star : \mathcal{Z} \times \mathcal{X} \to \mathbb{D}\mathcal{Y}$ as $C^\star_{z,x,y} = C_{x,y}$. Note that $C^\star$ simply repeats rows of C; moreover, $C^\star$ now allows us to investigate how much we can infer about $\mathcal{Z}$ from information flows about $\mathcal{X}$ through

[4] cfr. [34], Ch 10.

C.

Definition 5.3.1 (Dalenius g-vulnerability)**.** Given a channel $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ and a correlation $\Pi : \mathbb{D}(\mathcal{X}\times\mathcal{Z})$ expressed as a matrix, factorise Π into a marginal distribution $\rho : \mathbb{D}\mathcal{Z}$ and a channel $B : \mathcal{Z} \to \mathbb{D}\mathcal{X}$ such that $\Pi = \rho \triangleright B$. Then for any gain function $g : \mathbb{G}\mathcal{X}$, the *Dalenius g-vulnerability* of Π and C is defined as

$$V_g^D(\Pi, C) \quad := \quad V_g[\rho \triangleright BC] \,.$$

When g is the function bv, we call this *Dalenius Bayes vulnerability*, written V_1^D.

The Dalenius g-vulnerability expresses the vulnerability of the secrets $\mathcal{Z}$ caused by the leakage of the channel C and the correlation Π. Note that we can equivalently write:

$$V_g^D(\Pi, C) \quad := \quad V_g[\Pi \triangleright C^\star] \,, \tag{5.2}$$

by 'flattening' the correlation Π into a prior over pairs $\mathcal{Z} \times \mathcal{X}$.

It turns out that the strong refinement order $\sqsubseteq$ respects Dalenius g-vulnerability: that is, $C \sqsubseteq D$ iff $V_g^D(\Pi, C) \geq V_g^D(\Pi, D)$ for any correlation $\Pi : \mathbb{D}(\mathcal{X}\times\mathcal{Z})$ between $\mathcal{X}$ and some $\mathcal{Z}$, and for any $g : \mathbb{G}\mathcal{X}$. In the case of $C \not\sqsubseteq D$, then we can find a witness wrt Dalenius Bayes vulnerability. ie. There is a correlation Π such that

$$V_1^D(\Pi, C) < V_1^D(\Pi, D) \,. \tag{5.3}$$

In other words, the failure of refinement between channels can be expressed using the Bayes vulnerability of a correlated secret $\mathcal{Z}$.

5.3.2 Reasoning about Inferences

As we have noted above, we can use a gain function to model an adversary trying to guess some value within a scenario defined by prior knowledge $\pi \in \mathbb{D}\mathcal{X}$.

We can generalise the idea of Bayes vulnerability to an adversary trying to guess a *property* of the secret (or set of values).

Definition 5.3.2. Given a state space $\mathcal{X}$ let $\mathcal{P} := \{p_1, p_2 \ldots p_n\}$ be a partition of $\mathcal{X}$. Define the gain function $\overline{\mathcal{P}} : \mathcal{P}\times\mathcal{X} \to \{0, 1\}$:

$$\overline{\mathcal{P}}(p, x) \quad := \quad \begin{cases} 1 & \text{if } x \in p, \\ 0 & \text{otherwise.} \end{cases}$$

Then $\overline{\mathcal{P}}$ is called a *property gain function* for the partition $\mathcal{P}$.

We can now model an adversary's ability to use the information in a data release from a mechanism M to infer a property defined by a partition $\mathcal{P}$: it is $V_{\overline{\mathcal{P}}}[\pi \triangleright M]$ (where as usual, $\pi : \mathbb{D}\mathcal{X}$ models the adversary's prior knowledge about the secrets).

It turns out that we can use this idea to analyse the effectiveness of a mechanism in terms of both privacy *and* accuracy. To see how this works, recall the mechanisms in Figure 5.1 and Figure 5.2. The *privacy* aspect is to prevent observers from the inferring true responses of individuals, whereas the *accuracy* requirement is to deliver a result from which observers can infer with a high level of confidence the true count. [5]

To analyse how well a participant's true response can be inferred from the data release we first set the state space $\mathcal{X}$ to be the set of total possible responses, and choose a prior distribution $\pi \in \mathbb{D}\mathcal{X}$. Later, in our experiments (§5.6), we describe how to choose a prior to capture reasonable prior knowledge of the adversary. Writing $\langle r_1, r_2\rangle$ for a scenario where participant 1's true response is $r_1 \in \{0,1\}$ and participant 2's true response is $r_2 \in \{0,1\}$, we can see that the state space of possible responses is $\mathcal{X} := \{\langle 0,0\rangle, \langle 0,1\rangle, \langle 1,0\rangle, \langle 1,1\rangle\}$.

Let $\mathcal{S}$ be the partition $\{\{\langle 0,0\rangle, \langle 0,1\rangle\}, \{\langle 1,0\rangle, \langle 1,1\rangle\}\}$ – notice that it contains two sets: the first grouping the scenarios where participant 1's true response is 0, and the second where participant 1's true response is 1. Computing $V_{\overline{\mathcal{S}}}[\pi \triangleright M]$ gives the probability that the adversary can infer participant 1's true response, whatever it turns out to be. For M to defend strongly against unintended inferences about an individual participant we would like this to be as close to $V_{\overline{\mathcal{S}}}[\pi]$ as possible, because then the information delivered by the data release cannot be used by any adversary in his attack. In fact in our experiments (§5.6), we define privacy loss to be the ratio $V_{\overline{\mathcal{S}}}[\pi \triangleright M]/V_{\overline{\mathcal{S}}}[\pi]$.

On the other hand in both mechanisms the output count is randomised, even though delivering an accurate count is the purpose of the data release. We can therefore gauge the accuracy of the mechanism by calculating the probability with which the observer can infer the true count for the same prior π. In detail, let $\mathcal{U}$ be the partition $\{\{\langle 0,0\rangle\}, \{\langle 0,1\rangle, \langle 1,0\rangle\}, \{\langle 1,1\rangle\}\}$. Here there are three subsets – the first is the (only) case in which the true count is 0, the second contains two instances where the true count is 1 and the third is the unique case where the count is 2. Computing $V_{\overline{\mathcal{U}}}[\pi \triangleright M]$ therefore gives the probability that the observer (or adversary) can infer the true count. Clearly we would like this to be close to 1 for good accuracy.

We can see these ideas working out for our two examples above by constructing the information flow channels for each of them.

Consider first the channel associated with the randomised response mechanism described in Figure 5.1.

$$R \;:=\; \begin{array}{c} \\ \langle 0,0\rangle \\ \langle 1,0\rangle \\ \langle 0,1\rangle \\ \langle 1,1\rangle \end{array} \begin{array}{c} \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{pmatrix} 0.56 & 0.38 & 0.06 \\ 0.19 & 0.62 & 0.19 \\ 0.19 & 0.62 & 0.19 \\ 0.06 & 0.38 & 0.56 \end{pmatrix} \end{array} \tag{5.4}$$

Each row of the matrix corresponds to the probabilities of observing the output count given by 0, 1 or 2 when executed within a scenario defined by the participants' true responses. For example if both participants' responses are 0 (first row corresponding to $\langle 0,0\rangle$) the observer would see the output at 0 with probability 0.56 because 0 is output when both respondents' true

[5] Thus in this scenario the adversary and observer are the same.

values are counted and, when they are not, the random value that *is* counted is still zero. For each participant, his true value is counted with probability $1/2$, but also with probability $1/2$ he randomly picks to respond with zero anyway. Thus each participant contributes 0 to the final survey count with probability 0.75, and since participant responses are independent of each other the count reported will be 0 with probability $0.75{\times}0.75 \sim 0.56$. The other probabilities are computed similarly.

Assuming then a uniform prior distribution υ over possible scenarios, we see that the probability of inferring participant 1's true response using the information in the data release is

$$V_{\overline{\mathcal{S}}}[\upsilon \triangleright R] = 0.63 \,,$$

which is only a little more than the prior $V_{\overline{\mathcal{S}}}[\upsilon] = 0.5$. On the other hand the accuracy of inferring the sum of the responses is *worse*:

$$V_{\overline{\mathcal{U}}}[\upsilon \triangleright R] = 0.59 \,,$$

signifying that the observer cannot have a great deal of confidence in inferring the true tally.

We can perform the same analysis on Figure 5.2 for comparison. Here, the channel matrix is:

$$G \;:=\; \begin{array}{c} \\ \langle 0,0\rangle \\ \langle 1,0\rangle \\ \langle 0,1\rangle \\ \langle 1,1\rangle \end{array} \begin{array}{c} \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \left(\begin{array}{ccc} 0.75 & 0.1667 & 0.0833 \\ 0.25 & 0.5 & 0.25 \\ 0.25 & 0.5 & 0.25 \\ 0.0833 & 0.1667 & 0.75 \end{array}\right) \end{array} \tag{5.5}$$

Notice that the (truncated) geometric mechanism is used to compute the probabilities. In the first row, corresponding to scenario $\langle 0,0\rangle$, the true tally is 0 which is then reported accurately with probability 0.75. As above, assuming a uniform prior υ, we can compute the probabilities of inferring participant 1's true response and the true tally as follows:

$$V_{\overline{\mathcal{S}}}[\upsilon \triangleright G] = 0.67 \quad \textit{and} \quad V_{\overline{\mathcal{U}}}[\upsilon \triangleright G] = 0.625 \,.$$

Notice that whilst the accuracy has improved from 0.59 to 0.625, this has come at a cost of increasing the probability of inferring participant 1's response from 0.63 to 0.67. The reason that the increase in accuracy incurs a decrease in privacy is because the two properties are related: if the ability to infer the participants' responses increases then the ability to infer an accurate tally which depends on those responses must also increase. The extent to which we can have both good utility and good privacy depends crucially on how the $\mathcal{S}$ and $\mathcal{U}$ are correlated in the prior.

In fact, we can define a distribution $\Pi \in \mathbb{D}(\mathcal{S}{\times}\mathcal{U})$ which expresses the *correlation* between the privacy property defined by partition $\mathcal{S}$ and the useful property defined by partition $\mathcal{U}$:

$$\Pi_{su} \;:=\; \pi(s \cap u) \,.$$

From here we can observe that $\mathcal{S}$ and $\mathcal{U}$ can be highly correlated for some of their partition subsets suggesting that a more accurate inference of one must lead to a more accurate inference

of the other. For example if a tally of 0 can be accurately inferred then the probability of subsequently inferring that participant 1's true response is also 0 is $\Pi_{s_0,u_0}/\pi(u_0) = 1$, meaning that a 0 tally implies absolutely that participant 1's true value must be 0. On the other hand, the probability of inferring that his value is 0 if the tally is accurately reported as 1 is $\Pi_{s_0,u_1}/\pi(u_1) = 1/2$. Thus studying the impact of the mechanism in terms of the abstraction of the correlation will enable us to understand the trade off between privacy and accuracy, and the limitations on delivering highly accurate data releases in some scenarios.

5.4 The Privacy-Utility Trade Off

In this section we study the trade off between accuracy and privacy in terms of inferences. Rather than working with the raw data directly, we use an abstraction based on a correlation between $\mathcal{S}$ and $\mathcal{U}$ where, as described above, $\mathcal{S}$ is defined by a partition on raw data that specifies some privacy criterion, and likewise, $\mathcal{U}$ is defined by a partition that specifies the useful data to be released as accurately as possible, and we describe the correlation between $\mathcal{S}$ and $\mathcal{U}$ by a distribution $\mathbb{D}(\mathcal{S}\times\mathcal{U})$.

The next definition gives the effect of a mechanism in the ability for an adversary to infer the component $\mathcal{S}$.

> **Definition 5.4.1.** Let $\Pi\colon\mathbb{D}(\mathcal{S}\times\mathcal{U})$ represent a correlation in $\mathcal{S}\times\mathcal{U}$, and let $M\colon(\mathcal{S}\times\mathcal{U})\to\mathbb{D}\mathcal{Y}$ be a stochastic channel representing a data release. We say that M is susceptible to an inference leak for $\mathcal{S}$ if $V_{\overline{\mathcal{S}}}[\Pi] < V_{\overline{\mathcal{S}}}[\Pi \triangleright M]$. We say that M is completely privacy preserving wrt. $\mathcal{S}$ if and only if $V_{\overline{\mathcal{S}}}[\Pi] = V_{\overline{\mathcal{S}}}[\Pi \triangleright M]$. We measure the privacy loss by the ratio $V_{\overline{\mathcal{S}}}[\Pi \triangleright M]/V_{\overline{\mathcal{S}}}[\Pi]$.

Next, as described above, we can also define the accuracy of inferring the utility.

> **Definition 5.4.2.** Let $\Pi\colon\mathbb{D}(\mathcal{S}\times\mathcal{U})$ represent a correlation in $\mathcal{S}\times\mathcal{U}$, and let $M\colon(\mathcal{S}\times\mathcal{U})\to\mathbb{D}\mathcal{Y}$ be a stochastic channel representing a data release. M's accuracy for $\mathcal{U}$ is the probability that $\mathcal{U}$ can be inferred, ie. $V_{\overline{\mathcal{U}}}[\Pi \triangleright M]$. We say that M is completely accurate for $\mathcal{U}$ if and only if $V_{\overline{\mathcal{U}}}[\Pi \triangleright M] = 1$.

Now we have these definitions, we can provide a simple proof of the well-known "no free lunch" theorem of Kifer [37], which states that it is not possible to have a single mechanism that is arbitrarily accurate and arbitrarily private for all possible correlated secrets.

> **Theorem 5.1.** There exists no mechanism M which guarantees *both* arbitrary levels of accuracy *and* privacy for all datasets $\Pi\colon\mathbb{D}(\mathcal{S}\times\mathcal{U})$.
>
> *Proof.* Let M be a mechanism acting on a (correlated) data set. Pick $\mathcal{S} = \mathcal{U}$, and partition $\mathcal{P}$ on both $\mathcal{S},\mathcal{U}$ so that the private and the useful property are entirely correlated. For any $\varepsilon{>}0$ we might hope that $V_{\overline{\mathcal{P}}}[\Pi \triangleright M] \geq 1{-}\varepsilon$ for good accuracy, whilst at the same time $V_{\overline{\mathcal{P}}}[\Pi \triangleright M] \leq V_{\overline{\mathcal{P}}}[\Pi]{+}\varepsilon$ for good privacy. However if both

constraints hold simultaneously then we deduce that $2\varepsilon \geq 1 - V_{\overline{\mathcal{P}}}[\Pi]$ for all Π, something that cannot be guaranteed. □

Thm. 5.1 applies to all privacy mechanisms, including differential privacy of course. In the next section we investigate two common workflows for differential privacy — one which uses the randomisation in a way to obtain good accuracy, and the other to obtain good privacy. We investigate the trade-off between accuracy and privacy in both cases.

5.5 Application to Differential Privacy

In Chapter 3 we introduced the idea of *privacy workflows* (§3.3) which describe the ways in which $\mathbf{d}$-private mechanisms are constructed. Of particular interest were *oblivious mechanisms*, which model the scenario where a function f is applied to a dataset and the noise-adding mechanism is applied to the result of f; and *local mechanisms*, in which the noise is applied directly to the dataset, and any subsequent functions can be applied as post-processing steps. The overall workflow describes a mechanism which is designed to release useful information while protecting the privacy of individuals against particular threats.

We can alternatively think of modelling mechanisms in terms of the correlation $\Pi : \mathbb{D}(\mathcal{S} \times \mathcal{U})$ between the useful information $\mathcal{U}$ to be released and the private information $\mathcal{S}$ to be kept hidden. In this context we can see that there are two ways in which we can add the noise. The first is to apply it directly to the $\mathcal{S}$ component, in the style of local mechanisms, and somewhat oblivious to the $\mathcal{U}$ component. The second is to the $\mathcal{U}$ component – the accurate result of the query – as is done with oblivious mechanisms (ie. oblivious to the $\mathcal{S}$ component). As we saw in Figure 5.1 which is an example of the first kind, and Figure 5.2, an example of the second kind, both have an impact on accuracy and privacy. We define these two approaches to differential privacy more generally and examine their respective trade-offs.

Definition 5.5.1. A mechanism $C : (\mathcal{S} \times \mathcal{U}) \to \mathbb{D}\mathcal{Y}$ is called $\mathcal{U}$-insensitive if there is some $L : \mathcal{S} \to \mathbb{D}\mathcal{Y}$ such that $C_{suy} = L_{sy}$ (ie. $C = L^\star$). (Recall §5.3.1). Dually, C is called $\mathcal{S}$-insensitive if there is some $R : \mathcal{U} \to \mathbb{D}\mathcal{Y}$ such that $C_{suy} = R_{uy}$ (ie. $C = R^\star$).

Such insensitive mechanisms as described in Def. 5.5.1 add noise either to the $\mathcal{S}/\mathcal{U}$-component respectively, so that the information flow on the other ie. $\mathcal{U}/\mathcal{S}$-component is derived from the original correlation between $\mathcal{U}$ and $\mathcal{S}$.

In general, the use of randomisation can often be tailored to achieve particular probabilistic effects. The use of the Laplace distribution – for example, in oblivious mechanisms – means that good utility can be maintained on $\mathcal{U}$, and the impact on privacy therefore can be investigated through the original correlation. We explore the trade-off between accuracy and privacy in differentially-private mechanisms next.

5.5.1 Utility-Focussed Privacy

An oblivious mechanism for differential privacy adds noise to the result of a query, thereby creating opportunities to tune the randomness for accuracy of that query. The differentially-private guarantee for oblivious mechanisms entails a property of indistinguishability only on similar query *results*. The indistinguishability relating to the *privacy* of any sensitive data is dependent on how it correlates with those query results.

With our model for correlated data, we can decompose an oblivious mechanism into one which acts directly on $\mathcal{U}$ independently of $\mathcal{S}$, effectively treating $\mathcal{U}$ as the secret, and then reinstalling the correlation with $\mathcal{S}$. Recall Def. 3.3.1 from Chapter 3 which defined oblivious mechanisms in terms of factorisation into a query f and a $\mathbf{d}$-private mechanism H as a post-processing step. In the following, we generalise the idea of the 'oblivious component' H to a *utility-focussed* mechanism, for which arbitrary correlations with secrets $\mathcal{S}$ may exist.

> **Definition 5.5.2.** $M : (\mathcal{S}\times\mathcal{U}) \to \mathbb{D}\mathcal{Y}$ is called a *utility-focussed* ε-private mechanism if it is ε-private on $\mathcal{U}$, and $\mathcal{S}$-insensitive.

Whilst the randomisation can be tuned to optimise the accuracy of the query, this mode of adding randomness does not tell us about ability of an adversary, who has knowledge of the correlation, to infer the secret component. Thm. 5.1 tells us we need to consider that correlation to determine the risk. [6]

> **Remark 5.5.1.** We omit the use of metrics in Def. 5.5.2 and in later definitions for simplicity of presentation, but note that the same ideas apply in metric differential privacy, requiring simply a metric on $\mathcal{U}$.

A weaker property though is to consider whether there is a distinguished mechanism that protects better than all other (oblivious) mechanisms wrt. inference attacks. We write $\overline{\mathcal{S}}$ (resp. $\overline{\mathcal{U}}$) for the gain function derived from partitioning $\mathcal{S}$ (resp. $\mathcal{U}$) into its singleton sets.

> **Definition 5.5.3.** An utility-focussed ε-private mechanism $M : \mathcal{S}\times\mathcal{U} \to \mathbb{D}\mathcal{Y}$ is optimal wrt. inference attacks on $\mathcal{S}$, if $V_{\overline{\mathcal{S}}}[\Pi \triangleright M] \leq V_{\overline{\mathcal{S}}}[\Pi \triangleright M']$ for all $\Pi \in \mathbb{D}(\mathcal{S}\times\mathcal{U})$ and all (utility-focussed) ε-private mechanisms M'.

Unfortunately there is no mechanism that is optimal wrt. inference attacks except for the trivial mechanisms that release no information at all.

> **Theorem 5.2.** There is no non-trivial utility-focussed mechanism amongst all ε-private mechanisms on $\mathcal{S}\times\mathcal{U}$ that is optimal wrt. inference attacks on $\mathcal{S}$.
>
> *Proof.* (Sketch) Let M be such a non-trivial ε-private mechanism. We show that

[6] This problem of some individuals being "outliers" and thus potentially identified even in the release of aggregate data is well known and a worst-case mitigation of this situation is the use of sensitivity analysis in for example Laplacian mechanisms [5]. The result is to make any individual's contribution to a noisy announcements of an aggregate statistic minuscule.

it cannot be optimal wrt. inference attacks on $\mathcal{S}$. By Def. 5.5.2 this means that M can be decomposed with ε-private component $H{:}\,\mathcal{U} \to \mathbb{D}\mathcal{Y}$ such that $H^\star = M$. Moreover if M is not the trivial mechanism, then H cannot be trivial and so there exists some mechanism G such that $H \sqsubset G$ (strict refinement), implying that $G \not\sqsubseteq H$. In this case, we have immediately (from (5.3)) that there is some $\Pi \in \mathbb{D}(\mathcal{S}\times\mathcal{U})$ such that $V_{\overline{\mathcal{S}}}[\Pi \triangleright H^\star] > V_{\overline{\mathcal{S}}}[\Pi \triangleright G^\star]$, as required. □

Thm. 5.2 tells us that if privacy is of utmost concern, the use of a utility-focussed mechanism is really about preserving accuracy of the result of the query, and gives no optimal guarantees regarding whether the actual sensitive data is vulnerable to an inference attack. We see for example with Fig 5.2 that whilst this does deliver the most accurate result it is not the "most private" as measured by vulnerability to inferences, amongst non-trivial mechanisms.

5.5.2 Secrecy-Focussed Privacy

Dual to utility-focussed mechanisms, we can model privacy mechanisms which add randomness directly to the secrets $\mathcal{S}$ as follows:

Definition 5.5.4. $M{:}\,(\mathcal{S}\times\mathcal{U}) \to \mathbb{D}\mathcal{Y}$ is called a *secrecy-focussed* ε-private mechanism if it is ε-private on $\mathcal{S}$, and is $\mathcal{U}$-insensitive.

Local differential privacy mechanisms are somewhat in this style because they first produce a noisy version of the data through a noise-adding mechanism applied to $\mathcal{S}$, rather than to the result of a query. Subsequent queries can then be applied to the noisy version of the data, in the style of randomised response (Figure 5.1). However, as with utility-focussed mechanisms, we can reason more generally about workflows in which noise is added to the $\mathcal{S}$ part of the data, and utility must be inferred afterwards through the correlation $\mathbb{D}(\mathcal{S}\times\mathcal{U})$.

We now define optimality wrt. utility, and show that secrecy-focussed mechanisms cannot be universally optimal wrt. accuracy of utility.

Definition 5.5.5. An ε-differentially private mechanism M for datasets $\mathbb{D}(\mathcal{S}\times\mathcal{U})$ is optimal wrt. accuracy on $\mathcal{U}$, if $V_{\overline{\mathcal{U}}}[\Pi \triangleright M] \geq V_{\overline{\mathcal{U}}}[\Pi \triangleright M']$ for all $\Pi \in \mathbb{D}(\mathcal{S}\times\mathcal{U})$ and all ε-differentially private mechanisms $M' \in \mathbb{D}(\mathcal{S}\times\mathcal{U})$.

Theorem 5.3. Let $|\mathcal{S}| \geq 3$. There is no secrecy-focussed ε-differentially private mechanism on $\mathbb{D}(\mathcal{S}\times\mathcal{U})$ amongst all ε-differentially private mechanisms which is optimal wrt. accuracy on $\mathcal{U}$.

Proof. (Sketch.) Let M be such a secrecy-focussed mechanism, and let L be such that $L^\star = M$ as per Def. 5.5.2. If M is optimal then we require $V_1[\Pi \triangleright M] \geq V_1[\Pi \triangleright M']$ for all Π and M'. From (5.3) this means that for all ε-differentially private L' over $\mathcal{S}$ we require $L \sqsubseteq L'$. But we know that refinement does not hold in general between families of ε-private mechanisms (see, eg., Chapter 4 or

[35]).[7] □

Thm. 5.3 is interesting because it says that mechanisms that add randomness to enable a direct guarantee to the privacy component, cannot be optimised universally for all utility measures. For example, mechanisms that are designed to be locally-differentially private use post-processing to compute the utility on the noisy data. Post-processing, however is just refinement. Indeed it can be shown, for example, that Figure 5.1 is a refinement of a *secrecy-focussed* ε-private mechanism but more work is needed to explain the observation that locally private mechanisms often exhibit poor accuracy on medium-sized datasets. We end this section by showing that any refinements of secrecy-focussed mechanisms cannot provide universal accuracy.

COROLLARY 5.4. If $M \sqsubseteq M'$ and M is secrecy-focussed as in Thm. 5.3, then M' is not optimal wrt. accuracy on $\mathcal{U}$.

Proof. Follows since $V_{\overline{\mathcal{U}}}[\Pi \triangleright M] \geq V_{\overline{\mathcal{U}}}[\Pi \triangleright M']$. □

5.6 Experimental Results

In this section we present some experiments illustrating the main points presented in previous sections. We consider scenarios in which both a data analyst and an adversary can observe the output of a mechanism that reports a (possibly randomised) count of the number of rows in a dataset that satisfy some property. However, whereas the data analyst wants to infer the real value of the counting query performed on the dataset, the adversary wants only to infer the value of a sensitive attribute in a row just added to the dataset. More precisely, we consider experimental scenarios under the following conditions.

1. There is a dataset D of interest, consisting in a multiset of *rows*, each of which is a tuple defined on a set $\mathcal{A}$ of *attributes*. Each attribute $a \in \mathcal{A}$ has domain $domain(a)$, and we denote by $rows(\mathcal{A})$ the set of all possible rows that can be formed from attribute set $\mathcal{A}$, and by $x[a]$ the value assumed by attribute $a \in \mathcal{A}$ on a row $x \in rows(\mathcal{A})$.

2. A new row $x^\star \in rows(\mathcal{A})$ will be added to D (due to, eg., data from a new individual), yielding an extended dataset denoted (with a slight abuse of notation) by $D \cup x^\star$.

3. A *data analyst* wants to learn the result of a counting query $\texttt{count}_q$ that returns the number of rows in the extended dataset $D \cup x^\star$ satisfying a given condition q on a *useful attribute* $a_u \in \mathcal{A}$ (eg., how many rows have attribute *gender* set to *female*). The value $\texttt{count}_q(D \cup x^\star)$ is the *real count* for the counting query performed on the extended dataset.

4. An *adversary* has full knowledge of all rows in the dataset D, but is unsure about the contents of the newly added row $x^\star$. His goal is to learn the value $x^\star[a_s]$ of a *sensitive attribute* $a_s \in \mathcal{A}$ for this new row.

[7] When $|\mathcal{S}| = 2$ we will see (in Chapter 6) that there is a universally optimal mechanism for which refinement holds.

5. Both the data analyst and the adversary learn the count on the extended dataset $D \cup x^\star$ via a *query mechanism* $M_{\texttt{count}_q}$ that returns a (possibly randomised) version of the real count $\texttt{count}_q(D \cup x^\star)$. We call the output $M_{\texttt{count}_q}(D \cup x^\star)$ the *reported count*, by the mechanism, for the counting query performed on the extended dataset.

6. Both the data analyst and the adversary know the value of the real count $\texttt{count}_q(D)$ on the original dataset D, and both have as prior knowledge a distribution $\pi^\star{:}\mathbb{D}(\mathit{rows}(\mathcal{A}))$ on all values that the new added row $x^\star$ can assume (eg., the adversary and the data analyst may be the same entity). From that, they can derive, in the usual way, distributions on the value $x^\star[a_s]$ of the sensitive value of the newly added individual and on the real count $\texttt{count}_q(D \cup x^\star)$ for the query.

To properly formalise the privacy loss and utility of such scenarios, we introduce the following notation. Given a scenario Γ as described above, let Pr^Γ denote the corresponding joint probability distribution –depending on the coin tosses of the distribution $\pi^\star$ on the values for the new row $x^\star$ and on the query mechanism $M_{\texttt{count}_q}$ employed–, s.t. $Pr^\Gamma(x^\star{=}x, x^\star[a_s]{=}s, \texttt{count}_q(D \cup x^\star){=}u, M_{\texttt{count}_q}(D \cup x^\star){=}u')$ is the probability that in scenario Γ: (i) the new added row $x^\star$ assumes value $x \in \mathit{rows}(\mathcal{A})$; (ii) the sensitive value $x^\star[a_s]$ of the new added row $x^\star$ assumes value $s \in \mathit{domain}(a_s)$; (iii) the real count of query $\texttt{count}_q$ performed on the extended dataset $D \cup x^\star$ assumes value $u \in \mathbb{N}$; and (iv) the reported count of query $\texttt{count}_q$ produced by the mechanism $M_{\texttt{count}_q}$, w.r.t. the extended dataset $D \cup x^\star$, assumes value $u' \in \mathbb{N}$.

We then define the *privacy loss* of a scenario as the multiplicative Bayes leakage of the new row's sensitive value $x^\star[a_s]$ given the reported count $M_{\texttt{count}_q}(D \cup x^\star)$ on the extended dataset $D \cup x^\star$. (Compare Def. 5.4.1.) Intuitively, privacy loss reflects by how much knowledge of the reported count increases the adversary's chance of correctly guessing the secret value in one try. Formally:

$$\mathit{privacy\text{-}loss}(\Gamma) \stackrel{\text{def}}{=} \frac{\sum_{u' \in \mathbb{N}} \max_{s \in \mathit{domain}(a_s)} Pr^\Gamma(x^\star[a_s]{=}s, M_{\texttt{count}_q}(D \cup x^\star){=}u')}{\max_{s \in \mathit{domain}(a_s)} Pr^\Gamma(x^\star[a_s]{=}s)} \,. \tag{5.6}$$

On the other hand, we define the *utility* of a scenario as the posterior Bayes vulnerability of the real count $\texttt{count}_q(D \cup x^\star)$ of query $\texttt{count}_q$ performed on the extended dataset $D \cup x^\star$ given the reported count $M_{\texttt{count}_q}(D \cup x^\star)$ on the extended dataset $D \cup x^\star$. (Compare Def. 5.4.2.) Formally:

$$\mathit{utility}(\Gamma) \stackrel{\text{def}}{=} \sum_{u' \in \mathbb{N}} \max_{u \in \mathbb{N}} Pr^\Gamma(\texttt{count}_q(D \cup x^\star){=}u, M_{\texttt{count}_q}(D \cup x^\star){=}u') \,. \tag{5.7}$$

Equations (5.6) and (5.7) depend on the joint probability distribution Pr^Γ induced by the scenario, which itself depends on the coin tosses of the query mechanism $M_{\texttt{count}_q}$ employed. Hence, to fully flesh out these definitions we need to determine how the mechanism $M_{\texttt{count}_q}$ works. We consider two differentially-private mechanisms adding noise in different ways.

- An *oblivious mechanism* $M^{obv}_{\texttt{count}_q}$ that first applies counting query $\texttt{count}_q$ to the input dataset to produce a real count u, and then applies a (differentially-private) *oblivious randomisation function* $R^{obv}{:}\mathbb{N}{\to}\mathbb{D}(\mathbb{N})$ to u in order to produce a reported count u' as the output

of the mechanism. This is similar to a utility-focussed mechanism.

- A *local mechanism* $M^{loc}_{\texttt{count}_q}$ that first applies a (differentially-private) *local randomisation function* R^{loc}:$domain(a_u)$→$\mathbb{D}(domain(a_u))$ independently to each row's useful value to produce a randomised dataset D', and only then applies the counting query $\texttt{count}_q$ to D' in order to produce a reported count u'=$\texttt{count}_q(D')$ as the output of the mechanism. This is in the spirit of a privacy-focussed mechanism.

In our experiments we used the dataset released by ProPublica [8] corresponding to two years worth of data from the COMPAS tool (Correctional Offender Management Profiling for Alternative Sanctions), which is one of the most popular algorithmic tools used in the United States criminal justice system for pretrial and sentencing evaluation of the risk of bad behaviour for criminal defendants. We focused on the tool's assessment scores for "Risk of Failure to Appear", and eliminated from the dataset all but the most recent record for any given person, as well as all records with invalid entries for attributes `marital_status` and `score_text`. This treatment resulted in a database containing 11,710 unique records.

We then considered two scenarios. In both, the adversary's goal is to learn the newly added row's value for sensitive attribute `custody_status`, which can be 0-"pretrial defendant", 1-"residential program", 2-"probation", 3-"parole", 4-"jail inmate", or 5-"prison inmate". However, in scenario A the data analyst's query of interest is

$$\text{``}\texttt{select count * from D where custody_status=0}\text{''},$$

whose result is highly correlated with the secret information, whereas in scenario B the query of interest is

$$\text{``}\texttt{select count * from D where marital_status=0}\text{''}$$

(the range for `marital_status` is 0-"single", 1-"significant other", 2-"married", 3-"separated", 4-"divorced", or 5-"widowed"), whose result is highly independent from the secret information. In both scenarios data analyst and adversary assume the new row $x^\star$ added to dataset D follows a distribution $\pi^\star$:$\mathbb{D}(rows(\mathcal{A}))$ s.t. the probability of each x∈$rows(\mathcal{A})$ is the value's frequency in the dataset D. Moreover, for fairness in comparison, we instantiate both the oblivious randomisation function R^{obv} and the local randomisation function R^{loc} as the truncated geometric mechanism (recall Def. 3.4.2 from Chapter 3) with the same value of ε.

Table 5.1 presents the results of our experiments for various values of ε. In terms of inferences, we want the privacy loss to be low to offer good protection for individuals. This means, in the scale of Bayes vulnerability leakage, that the value should be close to 1 because then we can argue that in the studied scenario the information flow does not increase the adversary's prior knowledge. [9] For accuracy however, to offer good utility, we want it to be as high as possible. In the scale of Bayes vulnerability a value close to 1 represents high certainty that the true value of the query can be accurately inferred.

[8] https://github.com/propublica/compas-analysis

[9] In a leakage measure we do not tabulate the actual probability of inference, but rather than increase in inference compared to the prior.

	Scenario A				Scenario B			
	Oblivious mechanism		Local mechanism		Oblivious mechanism		Local mechanism	
ε	Priv. loss	Util.	Priv. loss	Util.	Priv. loss	Util.	Priv. loss	Util.
0 (theoretical minimum)	1.0000	0.7984	1.0000	0.7984	1.0000	0.7704	1.0000	0.7704
$\ln$ **3**	1.0000	0.7984	1.0000	0.7984	1.0000	0.7704	1.0000	0.7704
$\ln$ **5**	1.0452	0.8333	1.0000	0.7984	1.0000	0.8333	1.0000	0.7704
$\ln$ **10**	1.1402	0.9091	1.0000	0.7984	1.0000	0.9091	1.0000	0.7704
$\ln$ **100**	1.2418	0.9901	1.0000	0.7984	1.0000	0.9901	1.0000	0.7704
$\ln$ **200**	1.2480	0.9950	1.0001	0.7984	1.0000	0.9950	1.0000	0.7704
$\ln$ **500**	1.2517	0.9980	1.0035	0.8011	1.0000	0.9980	1.0000	0.7704
$\ln$ $\mathbf{10^3}$	1.2529	0.9990	1.0234	0.8169	1.0000	0.9990	1.0000	0.7704
$\ln$ $\mathbf{10^5}$	1.2542	1.0000	1.2481	0.9952	1.0000	1.0000	1.0000	0.9330
∞ (theoretical maximum)	1.2607	1.0000	1.2607	1.0000	1.2607	1.0000	1.2607	1.0000

Table 5.1: Results of privacy loss and utility on Scenarios A (with highly correlated counting query and secret) and B (with practically independent counting query and secret) for the COMPAS dataset A value close to 1 for privacy means the data release does not pose a risk to an individual; a value close to 1 for utility means that the data release is accurate.

As we can notice, in both scenarios described above and for each value of ε the local mechanism is consistently more private, but less useful, than its oblivious counterpart. The experiments also illustrate the well known fact from the literature that local mechanisms tend not to provide good utility in small datasets like the one we consider here: utility remains at its theoretical minimum for relatively high values of ε ($\approx \ln 200$ in Scenario A and $\approx \ln 10^3$ in Scenario B). Finally, note that in Scenario A, where real count and secret are highly correlated, it is hard to achieve a satisfactory trade-off between utility and privacy, as these values are always in opposition to each other. On the other hand, since in Scenario B the real count is practically independent of the secret, it is possible to maintain privacy at a minimum level even for very high values of utility.

5.7 Concluding Remarks

In this chapter we have studied the relationship between accuracy and privacy in data releases from the perspective of inferences. We have shown, using a channel model for quantitative information flow, how to capture reasonable assumptions about prior knowledge to compare the accuracy of a query result versus the ability for the adversary to infer additional information about individuals in the database. Finally, we have demonstrated how correlations in databases of moderate size pose challenges for protecting the privacy of individuals.

In future work we hope to use this kind of analysis to expose potential vulnerabilities in databases by considering the threats posed by inferences in proposed data releases.

5.8 Chapter Notes

This chapter is based on the published paper "On Privacy and Accuracy in Data Releases" [38].

Our work builds upon the "no free lunch" theorem of Kifer and Machanavajjhala [37], who provided the first analysis of the limitations of differential privacy in the presence of correlations between secrets. Their work also uses inference attack models to demonstrate the effect of correlations on possible inferences resulting in unexpected privacy breaches. Our work complements theirs by utilising the QIF framework to model the effect of inference attacks through the lens of information flow.

The same authors proposed Pufferfish, a framework providing a more nuanced approach to privacy that depends on the idiosyncracies of particular datasets [25]. In particular, Pufferfish allows definitions of privacy which extend differential privacy through the specification of particular inference attacks against which the data analyst would like to defend. Inspired by Pufferfish, He et al. [82] introduced the Blowfish framework to allow a more tailored approach to privacy policies which depend on known (public) correlations in the dataset.

In spirit, our observation that arbitrary correlations can exist in datasets – thus thwarting efforts to optimise for both privacy and utility – is related to the idea of *group privacy* [7]. In particular it has been observed that when correlations exist between groups of individuals in a dataset (for example), then differential privacy guarantees are limited according to the size of the group. Other authors have noted related issues regarding the privacy protection afforded by differential privacy in the face of correlations. Zhu et al. [83] proposed strengthening privacy mechanisms for correlated datasets based on a modified measure of the sensitivity of the dataset. Liu et al. [84] demonstrate an inference attack against a differentially private dataset by exploiting known correlations in the data. Works in this area focus on known correlations – particularly correlations within the dataset – and do not address the more general problem of unknown correlations which may be known only to an adversary.

Other works on inference attacks on private data consider alternate privacy metrics to measure privacy loss. Salamatian et al. [17] use traditional information theoretic measures such as entropy and mutual information to produce an alternate privacy framework to differential privacy which is focussed on protecting against inference attacks. Most recently, Jayaraman et al. [19] propose new privacy metrics to evaluate the risk of inference attacks.

Finally, there is considerable interest in understanding inference attacks on machine learning models which employ differential privacy to protect their training data. Rahman et al. [18] empirically evaluate the success of membership inference attacks against machine learning models trained with different privacy parameters. Most recently, Jayaraman et al. [19] empirically evaluate the risk of inference attacks on differentially private machine learned models using their own privacy metrics. Works in this area typically use ad hoc methods to evaluate privacy leakage, whereas our QIF framework permits rigorous analysis based on an operational inference attack model.

6

Optimality I: Discrete Mechanisms

Managing the privacy-utility trade-off is a core concern in the design of privacy-preserving mechanisms. One way to explore this trade-off is to look for so-called "optimal mechanisms". An optimal mechanism is one that releases the most useful information to a consumer while preserving some (fixed) level of privacy. The study of optimality has as its primary goal the discovery of mechanisms that provide the "best possible utility" for many consumers simultaneously. While this problem has been previously studied in the differential privacy literature, in this chapter we will situate optimality in the framework of QIF and study its properties using the notions of refinement and leakage stemming from an understanding of information flow. Our goal is to compare mechanisms wrt their usefulness for different *consumers* while keeping the overall $\mathbf{d}$-privacy guarantee fixed. That is, we would like to know, within the $\varepsilon{\cdot}\mathbf{d}$-privacy based order, which mechanism is the "best possible" mechanism for some class of consumers.

Using our framework, we generalise existing results on optimality in the literature with a new characterisation of optimal mechanisms for discrete spaces based on the notion of refinement. We utilise the barycentric representation of hyper-distributions introduced in Chapter 2 to bring key insights into the relationship between geometry and optimality for $\mathbf{d}$-privacy. We show how the metric affects the geometry and therefore the characterisation of optimality, giving examples for counting queries and sum queries. We re-explore the existing optimality results for these queries and find new results based on our characterisation. Finally, we show how our results generalise to arbitrary metric spaces.

6.1 Introduction

The study of optimal mechanisms for differential privacy has been strongly influenced by two landmark papers which have guided research directions in this area:

1. Ghosh et al. [39] produced a remarkable result which shows that the geometric mechanism is *universally optimal* in the context of oblivious mechanisms for "counting queries", a class of queries defined over statistical datasets. Their optimality result was proven for a particular class of functions called *monotonic loss functions* (to be defined later). Ghosh et al.'s paper was noteworthy not only for its result, but also because it was the first to model optimality through the lens of Bayesian consumers who use their prior knowledge to remap observations to guesses in order to maximise the utility they gain from a data release.

2. Brenner and Nissim [40] showed that for any other query type (eg. sum, average etc.) – again in the traditional oblivious setting of statistical datasets – that there are *no* universally optimal mechanisms. Their proof adopted the same model for utility as Ghosh et al.[1] but they generalised their result to metrics, using graph-theoretic techniques to analyse the effect of the metric on optimality, finding that only *linearly ordered* metrics (eg. the Euclidean distance on integers) could produce optimality results. However, their impossibility result was tied to the Bayes' risk loss function, and their graph-based approach does not naturally generalise to other loss functions.

Despite this foundational work, there have been no generalisations of these results to other loss functions or other scenarios (eg. outside counting and sum queries).[2] Our goal in this chapter is to examine these results using the framework of QIF in order to reframe optimality in terms of **d**-privacy and extend the results to other classes of loss functions.

6.1.1 Motivation and Contribution

Our interest in these results is that the Bayesian framework used for reasoning about utility corresponds to the QIF model for adversaries using g-vulnerability (or more precisely, its dual notion of ℓ-uncertainty). This makes QIF a natural choice for reasoning about optimality in this setting, whereby we think of adversaries now as *consumers* and the goal of optimality is to discover mechanisms which leak *as much as possible*.

However both of the above results are tied to *monotonic* loss functions, which are functions that capture the idea that guesses 'closer' to the secret should incur less loss than guesses 'further away'. In the differential privacy literature these sorts of loss functions are frequently used to model utility. For example, the functions

$$\ell_1(w, x) \quad := \quad |w - x|$$

[1] The result of [40] was also shown to hold for risk-averse consumers who we will not consider in this chapter.

[2] We note here the work of Chatzikokolakis et al. [15] who reframed the results to optimality for metric differential privacy, although without extending them.

and

$$\ell_2(w,x) \quad := \quad (w-x)^2$$

are commonly used to describe consumers whose expected loss depends on the absolute difference (ℓ_1) or squared difference (ℓ_2) of guesses from inputs. The Bayes risk loss function (described previously, see eg. §2.2.1) is another popular monotonic loss function which describes a consumer whose goal is to guess the input in one try. However, in QIF we are interested in modelling a variety of adversaries (and consumers), some of whose gain (loss) functions may be described in less typical, and non-monotonic ways. For example, consider the following scenarios:

(1) (The Weary Traveler)[3] A weary traveler wishes to accurately guess the time that the last RER-B train will depart Lozère for Paris. Clearly, guessing 2 minutes too late is very different from guessing 2 minutes too early. We would then model his loss using an *asymmetric* function which penalises guesses too late more than guesses too early.

(2) (The Real-Estate Maverick) An agent wishes to guess the parity (odd or even) of house numbers released privately from a dataset. [4] The agent chooses a *property* loss function[5] with 2 guesses (odd/even) which assign values of 0 when guessing correctly and 1 when guessing incorrectly.

Both of the above examples describe *non-monotonic* loss functions which are not covered by the results of [39] or [40], but which may represent scenarios of interest in other domains.

Finally, neither paper considers the possibility of extending the results to continuous input domains, nor are their proof techniques amenable to such extension. Our use of QIF for reasoning provides an algebraic framework which can naturally be extended to continuous spaces, which we will do in Chapter 7.

Contributions

Our goal here is to apply these results to the more general setting of metric differential privacy and situate them in the framework of QIF so that they can be generalised and extended to different scenarios of interest which may arise in the context of metric differential privacy. Our study makes the following contributions:

1. We introduce an algebraic framework for reasoning about optimality based on the channel model of QIF which allows optimal mechanisms to be described by a metric $\mathbf{d}$ and a choice of ε and supports reasoning about arbitrary loss functions.

2. We provide an algebraic characterisation of optimal mechanisms which deals with arbitrary classes of loss functions.

[3] This example is taken from [34].

[4] In some places, the side of the street on which a house is positioned – as determined by the parity of its street number – can make a significant difference in its sale value.

[5] No pun intended.

3. Based on this algebraic characterisation, we demonstrate an optimal, non-trivial loss function and mechanism for sum queries, contra the impossibility result of Brenner and Nissim [40]. However, we strengthen their result, showing that the impossibility holds for a smaller class than the "monotonic" functions but larger than Bayes' risk.

4. Our characterisation allows us to explore the richness of the space of optimal mechanisms, showing that the number of possible mechanisms grows exponentially as the size of the input space, even for the counting query class.

5. We use the novel barycentric representation supported by the QIF framework which allows visualisation of our results and gives access to general results from QIF regarding the geometry of refinement.

6. Finally, we are able to use the geometric perspective to generalise the results of Ghosh et al. to continuous mechanisms where we find that the Laplace mechanism is universally optimal; this is the work of Chapter 7.

6.2 Modelling Utility

Our study of optimality begins with a definition of *utility*, which specifies how we measure "goodness" for mechanisms (modelled here as channels), and leads us to optimality which seeks the "best possible" mechanism wrt the chosen goodness measure.

We model mechanisms as probabilistic channels $C : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ which take inputs of type $\mathcal{X}$ and produce outputs (distributions) of type $\mathbb{D}\mathcal{Y}$. We will not yet introduce any $\mathbf{d}$-privacy constraints on the channel – these will be introduced later – instead focussing for now on the model for utility. Note that we refer to *inputs* rather than *secrets*, since in our utility context these are no longer secrets to be kept hidden but useful information to release.

Dual to the QIF notion of gain functions (recall §2.2.1) we model consumers using *loss functions* $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, which describe the consumer's loss upon guessing w (after observing y) when the real value of the input is x.

The connection between the observation y and the loss-function parameter w is that the consumer does not necessarily have to "take what she sees" — there might be good reasons for her making a different choice. For example, in a word-guessing game where the last, obfuscated letter `?` in a word `SA?` is shown on the board, the consumer might have to guess what it really is. Even if it looks like a blurry `Y` (value 4 in Scrabble), she might instead guess `X` (value 8) because that would earn more points on average if from prior knowledge she knows that `X` is strictly *more* than half as likely as `Y` is – ie. it's worth her taking the risk. Thus rather than mandating that the consumer must accept what she thinks the letter is most likely to be, she uses the obfuscated query y to deduce information about the *whole* posterior distribution of the *actual* query... and might suggest that she guess some $w \neq y$, because the expected loss of doing that is less than (the expected utility is greater than) it would be if she simply accepted the y she saw. That rational strategy is called "remapping" [39]. Thus she sees y, but y tells her that

w is what she should choose as her least-loss inducing guess for x. That is, the *simplest* strategy is "take what you see"; but it might not be the best one.

Moreover, the consumer's strategy does not require that the number of guesses matches the number of inputs – for example, if the consumer's goal is to guess a property of the input rather than the input itself. This is illustrated in Example 6.1.

Example 6.1 (Example loss function with fewer actions than inputs)**.**

M	y_1	y_2	y_3
x_T	$3/4$	$1/8$	$1/8$
x_A	$2/3$	0	$1/3$
x_S	$1/4$	$1/4$	$1/2$

ℓ	w_1	w_2
x_T	0	1
x_A	1	0
x_S	1	0

A mechanism M defined over 3 inputs representing heights of individuals: 'Tall' (x_T), 'Short' (x_S) and 'Average' (x_A). A consumer chooses the loss function ℓ which distinguishes between 'Tall' (x_T) and 'not Tall' (x_A, x_S) individuals. This loss function has only 2 actions even though the mechanism M produces 3 outputs. In general there need not be a correspondence between outputs of a mechanism and actions of a loss function.

Our formulation for the expected loss to the consumer of her remapping strategy is then expressible using posterior uncertainty:

Definition 6.2.1. Given prior $\pi \in \mathbb{D}\mathcal{X}$, loss function $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ and channel $M : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, the posterior ℓ-uncertainty of M wrt π is defined as:

$$U_\ell[\pi \triangleright M] \quad := \quad \sum_{y \in \mathcal{Y}} \min_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi_x M_{x,y} \ell(w, x) \; .$$

In the differential privacy literature, a different formulation for expected loss – but in the same spirit – has been adopted [39, 40], incorporating the remapping strategy of the consumer more explicitly:

Definition 6.2.2 (Ghosh et al. [39])**.** The expected utility loss for a user with prior π and loss function ℓ using mechanism M is given by

$$U_\ell(\pi, M) \quad := \quad \min_r \sum_{x \in \mathcal{X}} \pi_x \sum_{y \in \mathcal{Y}} M_{x,y} \; \ell(r(y), x)$$

That is, the user chooses a remapping $r(y)$ for each observation y that minimises her expected loss.

We show now that these definitions coincide.

Lemma 6.1. The expected utility loss in Def. 6.2.2 is equivalent to the posterior ℓ-uncertainty defined in Def. 6.2.1.

Proof. We reason as follows:

$$
\begin{aligned}
& U_\ell(\pi, M) \\
= \quad & \min_r \sum_{x\in\mathcal{X}} \pi_x \sum_{y\in\mathcal{Y}} M_{x,y}\ \ell(r(y), x) && \text{“Def. 6.2.2”} \\
= \quad & \min_r \sum_{y\in\mathcal{Y}} \sum_{x\in\mathcal{X}} \pi_x M_{x,y}\ \ell(r(y), x) && \text{“Reorganising”} \\
= \quad & \sum_{y\in\mathcal{Y}} \min_r \sum_{x\in\mathcal{X}} \pi_x M_{x,y}\ \ell(r(y), x) && \text{“Min } r \text{ is over each } y\text{”} \\
= \quad & \sum_{y\in\mathcal{Y}} \min_w \sum_{x\in\mathcal{X}} \pi_x M_{x,y}\ \ell(w, x) && \text{“Letting } w = r(y)\text{”} \\
= \quad & U_\ell[\pi \triangleright M] && \text{“Def. 6.2.1”}
\end{aligned}
$$

□

Note that in the second-last step we employ a mapping from remaps $r(y)$ to guesses w. We can think of this as simply a relabelling of observations y to guesses w, and we can always ‘duplicate’ guesses w so that there is a one-one correspondence between guesses and inputs.

Remark 6.2.1. We note that in the literature there are also examples of non-Bayesian methods for computing loss, such as the Maximum Likelihood (ML) estimate ($\operatorname{argmin}_x M_{x,y}$) or the Maximum A Posteriori (MAP) estimate ($\operatorname{argmin}_x \pi_x M_{x,y}$). The Bayesian approach subsumes these methods; ML and MAP can be expressed as the optimal solution for a particular Bayesian consumer – one whose prior is uniform on the inputs – and so the Bayesian formulation we use produces the optimal mechanism for ML and MAP estimates as well. However, the converse is not true, as the earlier example in Chapter 2 (§2.2.3) demonstrates.

6.2.1 Universal optimality

Given a model for utility, we can now describe what it means to be the “best possible” mechanism within some class of mechanisms $\mathcal{M}$:

Definition 6.2.3 (Optimality)**.** We say that a mechanism $M\colon \mathcal{M}$ is *optimal* for a consumer modelled using a prior π and loss function ℓ for the class of mechanisms $\mathcal{M}$ if, for all mechanisms $M' \in \mathcal{M}$,

$$U_\ell[\pi \triangleright M] \ \leq\ U_\ell[\pi \triangleright M'] \,.$$

The above describes an optimal mechanism for a single consumer; we can naturally extend this idea to *universal optimality*, which we define for all consumers simultaneously.

Definition 6.2.4 (Universal Optimality)**.** Call a mechanism $M : \mathcal{M}$ *universally optimal* for the class $\mathcal{M}$ if

$$U_\ell[\pi \triangleright M] \;\leq\; U_\ell[\pi \triangleright M'] \tag{6.1}$$

for all priors π, all loss functions ℓ and all mechanisms $M' \in \mathcal{M}$.

We have seen Def. 6.2.4 before: it is a leakage order (recall Def. 2.3.1), equivalent to the secure refinement (pre)order on channels (Lem. 2.2).[6] Thus we have:

Lemma 6.2. Mechanism $M : \mathcal{M}$ is *universally optimal* iff $M \sqsubseteq M'$ for all $M' \in \mathcal{M}$.

Def. 6.2.4 tells us that a mechanism M is universally optimal if it is the minimal element in the standard refinement order restricted to $\mathcal{M}$. Noting now that $\sqsubseteq$ is not a lattice in general [61], we might be concerned that Def. 6.2.4 is too strong – that there may be no minimal element – and so we explore a weaker definition of optimality restricted to particular classes of loss functions. Letting $\mathbb{L}\mathcal{X}$ be the set of all loss functions, we define 'universal $\mathcal{L}$-optimality' for the subclass of loss functions $\mathcal{L} \subset \mathbb{L}\mathcal{X}$ as follows:

Definition 6.2.5 (Universal $\mathcal{L}$-Optimality)**.** Let $\mathcal{L} \subset \mathbb{L}\mathcal{X}$ be a set of loss functions. We say that $M \in \mathcal{M}$ is universally $\mathcal{L}$-optimal iff

$$U_\ell[\pi \triangleright M] \;\leq\; U_\ell[\pi \triangleright M']$$

for all $M' \in \mathcal{M}$, all priors $\pi \in \mathbb{D}\mathcal{X}$ and all loss functions $\ell \in \mathcal{L}$.

Note that this definition is universal in the sense that it quantifies over all priors. Note also that this weaker definition does not have a structural characterisation in terms of refinement. However, we can still safely reason using refinement, since if M is universally $\mathcal{L}$-optimal for some class $\mathcal{L}$ then it cannot be the case that there is some M' such that $M' \sqsubset M$ (strict refinement), otherwise M' would also have better utility for $\mathcal{L}$. Refinement finds its use in determining *candidates* for optimality, which we can then explore for optimality within the loss classes of interest.

We have not yet spoken about **d**-private mechanisms; we next place these definitions in the context of differential privacy and explain how they relate to the results of [39] and [40].

6.2.2 The privacy context

In order to compare different mechanisms for utility, they must be able to be applied to the same privacy problem. This privacy scenario thereby defines the class $\mathcal{M}$ of mechanisms within which we seek an optimal one.

The optimality results of [39, 40] are situated in the context of *oblivious mechanisms*, and we

[6] Actually its dual; we remark that all results for gain functions and vulnerabilities that we call on here hold dually loss functions and uncertainties [34].

likewise choose this context, noting that reasoning about utility for local models is not straightforward, and we leave the exploration of optimality in this space to future work.[7] Recall (§3.3.1) that an oblivious mechanism $K: \mathcal{X} \to \mathbb{D}\mathcal{Z}$ is one which decomposes into a query $f: \mathcal{X} \to \mathcal{Y}$ and a noise-adding component $H: \mathcal{Y} \to \mathbb{D}\mathcal{Z}$, where the utility of the mechanism K is determined solely by the utility of H. The universally optimal mechanism, if it is to be found, is one such H – and the class of $\mathcal{M}$ of mechanisms are the ones which induce, via f, the same $\varepsilon\cdot\mathbf{d}$-privacy guarantees on K.

We know exactly this class: it is the set of $\mathbf{d}_f$-private mechanisms, where $\mathbf{d}_f$ is the metric induced by f and the metric on K (Lem. 3.7 from Chapter 3). Moreover, we know how these are related by refinement: if $M \sqsubseteq M'$ and M' is $\mathbf{d}$-private then M is also $\mathbf{d}$-private (Thm. 4.3, Thm. 4.5 from Chapter 4). Our search for optimal mechanisms, then, is not in vain: all of the $\mathbf{d}$-private mechanisms remain candidates for optimality, and the universally optimal mechanism (if it exists) is the minimal one (in the refinement anti-chain) for an appropriately chosen $\mathbf{d}$.

6.2.3 Relationship to "counting" and "sum" queries

The results on optimal mechanisms which motivate our study are based on the traditional model of differential privacy and oblivious mechanisms: there, the space of inputs is *databases* of individuals, and privacy is measured using the Hamming metric between databases – thus a Hamming distance of 1 describes databases which differ in one row: a single individual. The space of outputs is determined by a *query* f which takes the input dataset and outputs some real-value; in the case of "counting queries", the output is always a natural number, whereas in the case of "sum queries" it could be any real value.[8] The noise-adding mechanism, operating on the query output, is optimal if it allows a consumer to learn the *most* information about the query result – whatever information that consumer is seeking (modelled by her loss function) – out of all the mechanisms which guarantee some fixed level of (differential) privacy on the original databases.

The optimality result of Ghosh et al. [39] applies specifically to counting queries; we can generalise this result using Lem. 3.7 and noting the following [15]:

> **Lemma 6.3.** Let $(\mathcal{X}, \mathbf{d}_{\mathrm{H}})$ be a metric space where $\mathbf{d}_{\mathrm{H}}$ denotes the Hamming distance. A *counting query* is a 1-Lipschitz map $f : (\mathcal{X}, \mathbf{d}_{\mathrm{H}}) \to (\{0\,..\,n\}, \mathbf{d}_2)$, where $n = |\mathcal{X}|$ and $\mathbf{d}_2$ is the Euclidean distance. Moreover, $\mathbf{d}_2$ is the *induced metric* on $\{0\,..\,n\}$ wrt f, $\mathbf{d}_{\mathrm{H}}$.

We can now restate the optimality result(s) of Ghosh et al. in our more general metric-based setting:

> **Theorem 6.4** (Theorem 3.2, Corollary 3.3 in [39]). Fix $N \geq 1$ and $e^{-\varepsilon} \in [0, 1]$,

[7] This is due to the fact that utility for local models is determined by the composition of individual privacy mechanisms, and analysing this composition is beyond the scope of this work.

[8] Other queries of interest, such as "average" fall under our study of "sum" queries, so that in fact the sum and counting queries cover all of the cases of interest for statistical datasets.

> and let $\mathcal{M}$ be the class of $\varepsilon{\cdot}\mathbf{d}_2$-private mechanisms on $\{0\ldots N\}$. Then the geometric mechanism $G^\varepsilon \in \mathcal{M}$ is universally $\mathcal{L}$-optimal in $\mathcal{M}$, where $\mathcal{L}$ is the class of *monotonic* loss functions.

Our discussion of monotonic loss functions is yet to come; we note that the result above applies equally to quantised reals $\{0, q, 2q, \ldots\}$ by simply scaling the domain [15].

Thm. 6.4 allows us to generalise the optimality result for any oblivious context in which $\mathbf{d}_2$ is the induced metric on a (scaled) domain of $\{0\ldots N\}$. We will study this space of mechanisms in §6.7.

Likewise, we can generalise the impossibility result of Brenner and Nissim [40] to metrics. Their theorem also applies in the context of monotonic loss functions but this time for sum queries, which generalise to the discrete metric $\mathbf{d}_\mathrm{D}$:[9]

> **Lemma 6.5.** Let $(\mathcal{X}, \mathbf{d}_\mathrm{H})$ be a metric space. A *sum query* is a 1-Lipschitz map $f : (\mathcal{X}, \mathbf{d}_\mathrm{H}) \to (\{0\ldots N\}, \mathbf{d}_\mathrm{D})$. Moreover, $\mathbf{d}_\mathrm{D}$ is the *induced metric* on $\{0\ldots N\}$ wrt f, $\mathbf{d}_\mathrm{H}$.

We now restate their result, an impossibility, which (essentially) says that there are no other optimal mechanisms apart from the ones for counting queries:

> **Theorem 6.6** (Theorem 3.4 in [40])**.** Fix $N{\geq}1$ and $e^{-\varepsilon} \in [0,1]$, and let $\mathcal{M}$ be the class of $\varepsilon{\cdot}\mathbf{d}_\mathrm{D}$-private mechanisms on $\{0\ldots N\}$. Then there is *no* universally $\mathcal{L}$-optimal in $\mathcal{M}$, where $\mathcal{L}$ is the class of *monotonic* loss functions.

We will study the space of $\mathbf{d}_\mathrm{D}$-private mechanisms on $\{0\ldots N\}$ in §6.8, where we show that Thm. 6.6 is too strong, and we find that there are in fact optimal mechanisms in this space.

6.2.4 Our results on optimality

Now with the above results in mind, we will appeal to the optimality result of Ghosh et al. to more broadly characterise optimal mechanisms and their corresponding loss functions for $\mathbf{d}_2$-private mechanisms. We re-investigate the space of $\mathbf{d}_\mathrm{D}$-private mechanisms and find characterisations for optimality in this space – as hinted above – along with examples of optimal mechanisms and their corresponding loss functions. Finally we investigate the Hamming cube – a space of length-3 bitstrings wrt the Hamming distance – as a simple demonstration of how to put into practice the optimality results uncovered in this exploration.

More generally, our goal is to study optimality in the broader context of metric spaces wrt arbitrary loss functions of interest, and to provide a general method for characterising which mechanisms are optimal and for which loss functions. Our approach utilises the geometric understanding of refinement outlined in Chapter 2 (§2.4.2) for expressing refinement as a (geometric) relationship between hyper-distributions. We will see that the classes of optimal mechanisms can be characterised in terms of the structure of their corresponding hypers. This will

[9] Recall that the discrete metric assigns distance 1 to all pairs of distinct elements.

allow us to provide a characterisation of optimality in terms of the geometry of the loss functions and hypers in the metric space of interest.

6.3 The Geometry of d-Privacy

In this section we explore the geometry of **d**-private hypers generally – that is, without choosing a particular metric **d**. We will first observe that the **d**-privacy constraints on channels C carry through to the posteriors of the corresponding hyper $[\upsilon \triangleright C]$, and then that these posteriors – treated as vectors – lie in a convex region, and that this region completely characterises the set of **d**-private channels. Finally, we will identify the *minimal* elements under refinement in this region; these are the **d**-private hypers which will form the basis of our characterisation of optimal mechanisms to come.

6.3.1 The relationship between channels and hypers

We first recall that the refinement (pre)order on channels can be expressed as a refinement on hypers:

> **Lemma 2.5.** Let $A : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, $B : \mathcal{X} \to \mathbb{D}\mathcal{Z}$ be channels and let υ be the uniform prior on $\mathcal{X}$. Then
>
> $$A \sqsubseteq B \qquad \text{iff} \qquad [\upsilon \triangleright A] \sqsubseteq_\circ [\upsilon \triangleright B] \,.$$

On hypers, refinement becomes a true partial order, since the action of forming hypers collapses equivalent channels onto the same hyper [61]. The equivalence is realised by combining columns of a channel which are scalar multiples of each other. Take the following example.

$$A = \begin{pmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{pmatrix} \qquad B = \begin{pmatrix} 2/3 & 1/6 & 1/12 & 1/12 \\ 1/3 & 1/3 & 1/6 & 1/6 \end{pmatrix}$$

The channels A and B are *equivalent*: they have the same leakage properties since the last 3 columns of B can be merged resulting in the channel A. And indeed the action of the uniform prior υ on both channels yields the same hyper:[10]

$$[\upsilon \triangleright A] = \begin{bmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \\ 1/2 & 1/2 \end{bmatrix} = [\upsilon \triangleright B]$$

(Note that in this chapter we will sometimes write hypers as bracketed arrays with each outer written underneath its corresponding posterior. We omit the inputs for brevity when they are understood from context.)

It will therefore be sufficient for us to focus our reasoning entirely on valid hypers[11] knowing that a) every **d**-private hyper corresponds to an equivalence class of **d**-private channels and b) if

[10] This was shown to hold for any full-support prior [61]. Here we will always choose the uniform distribution.

[11] A hyper is 'valid' if the convex hull of its supporting posteriors contains the uniform distribution.

the hyper is optimal, then so is every channel in the equivalence class generating it.[12]

Next we note we can take convex combinations of hypers, eg., $\sum_i \lambda_i \Delta_i$ by distributing the convex coefficients λ_i to the outer probabilities of the corresponding Δ_i and then combining common posteriors so that, for example, the inner δ^y is assigned the outer $\sum_k \lambda_k a_{\Delta_k}$ where a_{Δ_k} is the outer assigned to δ^y in Δ_k.

We can also take convex combinations of *channels* in correspondence with hypers as follows: for convex coefficients λ_i we define

$$C \;:=\; \sum_i \lambda_i C_i \qquad \text{if} \quad [\pi \triangleright C] \;=\; \sum_i \lambda_i [\pi \triangleright C_i]\,.$$

That is, the convex sum of channels can be computed by converting channels to hypers, performing a convex combination, and recovering the resulting channel. In this chapter we will take $\pi = \upsilon$.

Terminology

We use the notation $\lceil \Delta \rceil$ to refer to the set of posteriors of a hyper Δ which occur with non-zero probability and we call these 'inners'. We denote by $\mathbf{ch}\lceil \Delta \rceil$ the convex hull of the inners of the hyper Δ. We will call two channels equivalent, written $A \equiv B$, whenever $[\upsilon \triangleright A] = [\upsilon \triangleright B]$. Finally, we will say that channel C 'corresponds to hyper Δ' if $[\upsilon \triangleright C] = \Delta$. When starting with Δ, we find a 'corresponding channel' in the usual way[13] – in this case we do not worry about which particular (equivalent) channel we end up with (it will, of course, be **d**-private).

6.3.2 The space of d-private hypers

Rewriting now the **d**-privacy channel constraints (see Def. 3.1.3) as the linear inequalities

$$\left.\begin{array}{l} C_{x,y} - e^{\mathbf{d}(x,x')} \cdot C_{x',y} \geq 0 \\ e^{\mathbf{d}(x,x')} \cdot C_{x,y} - C_{x',y} \geq 0 \end{array}\right\} \quad \forall\, x, x', y\,, \tag{6.2}$$

we observe that these inequalities also hold for the posteriors of the hyper produced by the action of the uniform distribution.

> **Lemma 6.7.** Let C be a **d**-private channel matrix, υ be the uniform distribution on $\mathcal{X}$ and let δ^y be the posterior of the hyper $[\upsilon \triangleright C]$ corresponding to observation y. Then for every $x, x' \in \mathcal{X}$ it holds that
>
> $$\frac{C_{x,y}}{C_{x',y}} \;=\; \frac{\delta^y_x}{\delta^y_{x'}}$$
>
> where δ^y_x is the x-indexed element of the column vector δ^y.

[12] In the QIF literature, the equivalence class of channels is defined by its canonical 'abstract channel' representation [61].

[13] ie. by multiplying each inner by its corresponding outer and normalising each row of the resulting matrix to produce a channel.

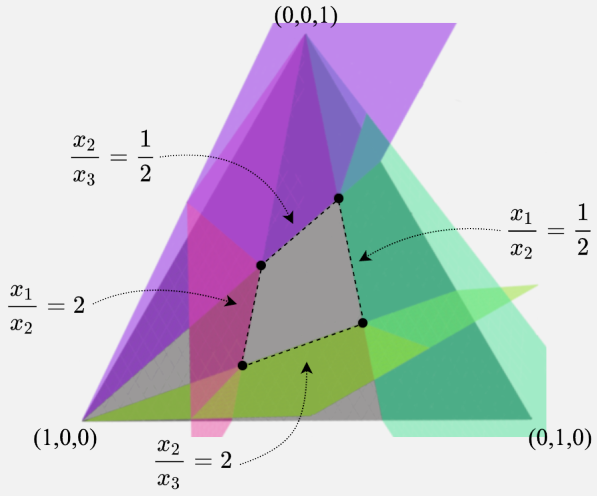
Proof. Rewrite δ_x^y as $\upsilon_x \cdot C_{x,y} / \sum_{z \in \mathcal{X}} \upsilon_x \cdot C_{z,y}$ and the result follows. □

That is, we can rewrite (6.2) as linear constraints on posteriors. Remembering that the posteriors are vectors, (6.2) then defines a set of hyperplanes whose intersection is a convex region (or polytope) containing all of the posteriors of the **d**-private hypers. Moreover, every **d**-private mechanism can be constructed from posteriors lying inside this convex region. We call this convex region "the space of **d**-private hypers" (see Example 6.2).

Example 6.2 (The Space of **d**-Private Hypers)**.** Constructing the space of hypers over 3 inputs $\{x_1, x_2, x_3\}$ for $\varepsilon \cdot \mathbf{d}$-private mechanisms with $\varepsilon = \ln 2$ and Euclidean metric **d**, assuming that we have $d(x_1, x_2) = 1$, $d(x_2, x_3) = 1$, $d(x_1, x_3) = 2$.
We treat the values x_1, x_2, x_3 as co-ordinates in a 3-dimensional space. The $\ln 2 \cdot \mathbf{d}$-privacy constraints become linear inequalities:

$$1/2 \leq x_1/x_2 \leq 2$$
$$1/2 \leq x_2/x_3 \leq 2$$

Note that the third constraint between x_1 and x_3 is trivially satisfied in this example.
The posteriors over 3 inputs all lie on the simplex defined by $x_1 + x_2 + x_3 = 1$, and the above constraints become hyperplanes which intersect the simplex in a convex region with 4 vertices shown below.



The coloured hyperplanes are the linear constraints which intersect the simplex forming the convex region shown by the dotted lines. Every point inside this convex region represents a posterior whose constraints are $\ln 2 \cdot \mathbf{d}$-private.

We also recall some important facts about hypers:

Lemma 6.8. The uniform distribution υ lies in the space of **d**-private hypers.

Proof. υ satisfies **d**-privacy for any metric **d** and therefore is a point in the convex region defined by the **d**-private linear constraints from Eqn (6.2). □

LEMMA 6.9. (Cor. 4.8 from [34]) Let C be a channel and let $\Delta = [\upsilon \triangleright C]$ be the hyper formed from pushing the uniform distribution through C. Then, writing $\Delta = \sum_i a_i[\delta^i]$, it holds that $\sum_i a_i \delta^i = \upsilon$.

Notice in Lem. 6.9 that the first sum $\sum_i a_i[\delta^i]$ expresses a convex sum of posteriors as point hypers, whereas the second sum $\sum_i a_i \delta^i$ expresses the convex combination of posteriors as vectors.

We bring the above lemmas together to show that the convex space characterises the class of **d**-private hypers – and, by extension, the **d**-private channels. This tells is that it is safe to reason about **d**-private channels in this space – all of their hypers can be found there, and conversely every hyper we find corresponds to a class of **d**-private channels.

LEMMA 6.10. Let C be a **d**-private channel with corresponding hyper $\Delta = [\upsilon \triangleright C]$. Then the posteriors of Δ lie in the convex space of **d**-private hypers. Conversely, let $S = \{\delta^1, \delta^2, \ldots, \delta^n\}$ be a set of posteriors in the convex space of **d**-private hypers st. υ is contained in their convex hull. Then there exists a **d**-private channel C' with corresponding hyper $\Delta' = [\upsilon \triangleright C']$ whose support is S.

Proof. The forward direction follows directly from Lem. 6.7 which tells us that the posteriors all satisfy the privacy constraints from Eqn (6.2). For the reverse direction, since υ is in the convex hull of S it can be written as a convex combination of the δ^i. ie. There exists convex coefficients λ_i st. $\upsilon = \sum_i \lambda_i \delta^i$ and $\sum_i \lambda_i = 1$. We can now construct the channel C' by multiplying each δ^i by its corresponding λ_i and treating them as columns of the 'joint distribution' matrix. Note that this multiplication preserves the **d**-privacy constraints on each column. Finally, we normalise along each row to produce the channel C'. However, we know that each row sums to $1/n$ by the choice of λ's. Therefore normalisation preserves the **d**-privacy constraints, and we have that C' is **d**-private. □

To summarise: Lem. 6.10 tells us that we can construct a hyper that corresponds to a **d**-private channel[14] by

1. choosing a set of **d**-private posteriors whose convex hull contains υ, and
2. choosing an outer distribution a_y for each posterior δ^y such that $\sum_y a_y \delta^y = \upsilon$.

We can therefore proceed by constructing hypers with the above properties without having to worry about the construction of the original channel.

[14] We omit referring to 'the equivalence class' for brevity, noting that the corresponding channel characterises all of the properties we need.

6.3.3 Kernel and Vertex Mechanisms

Now that we have carefully mapped out the space of **d**-private channels – remember, these are our candidates for universal optimality – we can turn our attention to the task of identifying the minimal elements in the refinement order. That is, the relation $\Delta_1 \sqsubseteq \Delta_2$ (for hypers Δ_1, Δ_2) says that Δ_1 has better utility than Δ_2; we seek the Δ_1 that anti-refines all other Δ_2 hypers corresponding to **d**-private channels.

Two properties of refinement [34] will be key to finding these minimal elements:

(1) The posteriors of a refined hyper (eg. Δ_1 above) contain, in their convex hull, the posteriors of the refining hyper (Δ_2).

(2) Each posterior of a refining hyper (Δ_2) is constructed via an Earth Move from posteriors of the refined hyper (Δ_1).

The universally optimal hyper Δ_1 that we seek must therefore consist of posteriors which contain all other posteriors in their convex hull. Since the space we are examining is convex, and constructed from a finite set of linear constraints, these posteriors must be the vertices of this convex polytope – the points of intersection of the hyperplanes forming the **d**-privacy constraints.

We now tease out the details carefully, showing that indeed these vertices correspond to the minimal elements that we seek – the optimal hyper(s) – upon which we will base our investigation of optimality in later sections.[15]

Definitions

We begin by distinguishing two types of mechanisms (and their corresponding hypers) formed from vertices in the space of **d**-private hypers: vertex mechanisms and kernel mechanisms.

> **Definition 6.3.1.** (Vertex Mechanism/Hyper) Let K be a **d**-private mechanism with corresponding hyper Δ. We say that K is a *vertex mechanism* (and Δ a *vertex hyper*) if the inners in $\lceil\Delta\rceil$ are vertices in the space of **d**-private hypers.

Vertex mechanisms always exist: every **d**-private space must have at least one, since the space contains the uniform distribution (Lem. 6.8) and therefore the set of vertices (of the defining polytope) forms a valid hyper (Lem. 6.10).

We next distinguish an additional property – linear independence of posteriors – which allows us to drop the Earth Move requirement on refinement (Lem. 2.6, [34]).

> **Definition 6.3.2.** (Kernel Mechanism/Hyper) Let K be a **d**-private mechanism with corresponding hyper Δ. We say that K is a *kernel mechanism* and Δ a *kernel hyper* if K is a vertex mechanism, and if the inners in $\lceil\Delta\rceil$ are linearly independent (as vectors).

[15] We will not, in this section, tease out the uniqueness of the minimal element – this is coming in the next section.

We may wonder if kernel mechanisms always exist – ie. whether it is always possible to find a linearly independent set of vertices that contains the uniform distribution. It turns out that they do (see Appendix §A.1 for details), and this fact is used to prove Prop. 2 below: that, in fact, every vertex mechanism decomposes into a convex sum of kernel mechanisms.

Properties

Kernel mechanisms have a number of important properties that flow from their corresponding kernel hypers. (See Appendix §A for proofs). The first property tells us that kernel mechanisms are *irreducible*.

> **PROPERTY 1.** If K is a kernel mechanism then there is no kernel mechanism K' st. $\lceil \Delta_{K'} \rceil \subset \lceil \Delta_K \rceil$ where $\Delta_K, \Delta_{K'}$ are hypers corresponding to mechanisms K, K' respectively.

The next property says that kernel mechanisms generate all of the vertex mechanisms.

> **PROPERTY 2.** Any vertex mechanism can be written (non-uniquely) as a convex sum of kernel mechanisms. Conversely, any convex sum of kernel mechanisms is a vertex mechanism.

Our final property says that kernel mechanisms are not related by refinement.

> **PROPERTY 3.** If K, K^* are kernel mechanisms then $K \not\sqsubseteq K^*$ and $K^* \not\sqsubseteq K$.

Observe that although Property 2 says that kernel mechanisms generate the space of vertex mechanisms, the representation of a vertex hyper as a convex sum of kernel hypers is not necessarily unique (see Example 6.3).

These properties lead us to an important characterisation of **d**-private mechanisms:

> **LEMMA 6.11.** (Characterisation of **d**-Private Mechanisms) Every **d**-private mechanism is a refinement of a convex sum of kernel mechanisms.
>
> *Proof.* Given a **d**-private mechanism M which is not a vertex mechanism, and its corresponding hyper Δ_M, we know that each posterior δ^i in $\lceil \Delta_M \rceil$ sits inside the convex hull of the vertices in the space of **d**-private hypers (by Lem. 6.10), and we can thus perform a "reverse Earth Move" (Def. 2.4.2), moving mass from each posterior δ^i to some set of vertices whose convex hull encloses δ^i, preserving the overall centre of mass υ of Δ. Thus we get a valid anti-refining vertex mechanism, which from Property 2 is a convex sum of kernel mechanisms. □

The usefulness of this characterisation may not be immediately apparent, until we look (later) at leakage properties of channels (eg. for understanding $\mathcal{L}$-optimality). We will observe that the leakage properties of channels are characterised by the leakage properties of their

constituent parts – the kernel mechanisms – which can (conversely) be used as building blocks to form classes of mechanisms with particular leakage characteristics.

Lem. 6.11 also gives an alternative perspective on the relationship between $\sqsubseteq$ and the $\mathbf{d}$-privacy order induced by ε – recall that in Chapter 4 we showed that the former implies the latter.

Finally, we come to the main result of this section: identifying the minimal elements in the refinement order; and we now show that these are – as anticipated – the vertex mechanisms.

> **COROLLARY 6.12.** The vertex mechanisms are the minimum elements under refinement.
>
> *Proof.* Lem. 6.11 tells us that every $\mathbf{d}$-private mechanism is a refinement of a vertex mechanism. Now, if V is a vertex mechanism, then its anti-refinement (if it exists) can only be another vertex mechanism (by the convex hull property of refinement, Def. 2.4.3). However there is no Earth Move possible between vertices of a convex space, thus no anti-refinement of a vertex mechanism. Therefore, every $\mathbf{d}$-private mechanism has a vertex mechanism as the bottom element of its anti-refinement chain. □

6.4 Characterising Optimal Mechanisms

We have seen (§6.2) that the various notions of optimality, universal optimality and universal $\mathcal{L}$-optimality are different leakage notions, with the strongest – universal optimality – corresponding exactly to channel refinement $\sqsubseteq$; and the universally optimal mechanism (if it exists) is the minimal element in this order. We have also seen (§6.3) how to characterise the minimal elements in this order restricted to the class of $\mathbf{d}$-private mechanisms: they are the *vertex mechanisms*. What remains to be done – the work of this section – is to determine:

a) do universally optimal mechanisms exist?

b) can we characterise the universally $\mathcal{L}$-optimal mechanisms?

A reminder: that our class of mechanisms for optimality $\mathcal{M}$ is the set of all $\mathbf{d}$-private mechanisms for some designated $\mathbf{d}$. We do not yet need to say what the metric $\mathbf{d}$ is – our results for this section remain independent of the choice of metric.

6.4.1 Existence of universally optimal mechanisms

We bring together the results from the previous sections to characterise the spaces containing universally optimal mechanisms.

> **LEMMA 6.13** (Existence of universally optimal mechanisms)**.** A space of $\mathbf{d}$-private mechanisms admits universal optimality if it contains a unique kernel mechanism

Example 6.3. A counter-example to uniqueness of representation of vertex mechanisms as convex sums of kernel mechanisms. Consider the following hyper.

$$\Delta = \begin{bmatrix} 1/2 & 1/4 & 1/4 & 1/5 & 2/5 & 2/5 \\ 1/4 & 1/2 & 1/4 & 2/5 & 1/5 & 2/5 \\ 1/4 & 1/4 & 1/2 & 2/5 & 2/5 & 1/5 \\ \frac{4}{27} & \frac{4}{27} & \frac{4}{27} & \frac{5}{27} & \frac{5}{27} & \frac{5}{27} \end{bmatrix}$$

We can check that this is a valid hyper – its inners are valid distributions, its outer is a distribution and the inners average (via the outer) to the uniform prior. It is also a vertex hyper in the space of $\ln 2 \cdot \mathbf{d}_{\mathrm{D}}$-private hypers where $\mathbf{d}_{\mathrm{D}}$ is the Discrete metric. This space consists of the following kernel hypers:

$$H_1 = \begin{bmatrix} 1/4 & 2/5 \\ 1/4 & 2/5 \\ 1/2 & 1/5 \\ 4/9 & 5/9 \end{bmatrix} \qquad H_2 = \begin{bmatrix} 1/4 & 2/5 \\ 1/2 & 1/5 \\ 1/4 & 2/5 \\ 4/9 & 5/9 \end{bmatrix} \qquad H_3 = \begin{bmatrix} 1/2 & 1/5 \\ 1/4 & 2/5 \\ 1/4 & 2/5 \\ 4/9 & 5/9 \end{bmatrix}$$

$$R_1 = \begin{bmatrix} 1/2 & 1/4 & 1/4 \\ 1/4 & 1/2 & 1/4 \\ 1/4 & 1/4 & 1/2 \\ 1/3 & 1/3 & 1/3 \end{bmatrix} \qquad R_2 = \begin{bmatrix} 1/5 & 2/5 & 2/5 \\ 2/5 & 1/5 & 2/5 \\ 2/5 & 2/5 & 1/5 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

We can express the hyper Δ in the following ways:

$$\Delta = 1/3 H_1 \oplus 1/3 H_2 \oplus 1/3 H_3$$
$$\Delta = 4/9 R_1 \oplus 5/9 R_2$$

ie. As 2 different convex sums of kernel hypers. And therefore the same holds for the corresponding vertex and kernel mechanisms.

– and this mechanism, if it exists, is *universally optimal* (ie. for all priors and loss functions).

Proof. Let K, K' be two kernel mechanisms in the space. Then K, K' are not related by refinement (Prop. 3) and therefore cannot be universally optimal (Lem. 6.2). If K is the unique kernel mechanism in the space, then every $\mathbf{d}$-private mechanism is a refinement of it (Lem. 6.11) and it has no anti-refinement (Cor. 6.12) so it must be the minimal element under refinement, hence universally optimal (Lem. 6.2). □

Lem. 6.13 is very strong, however it turns out that we can find such a space - the trivial space of mechanisms on 2 inputs – and this admits a universally optimal mechanism as we show below.

THEOREM 6.14. For $n = 2$, the following $\mathbf{d}$-private mechanism (written as a channel matrix) is *universally optimal* (ie. for *all* loss functions and priors):

$$T = \begin{bmatrix} k \cdot e^{\mathbf{d}} & k \\ k & k \cdot e^{\mathbf{d}} \end{bmatrix}$$

where $k = 1/(1+e^{\mathbf{d}})$ is a scaling factor to ensure rows sum to 1.

Proof. The space of $\mathbf{d}$-private hypers on 2 inputs $\{x_1, x_2\}$ is an interval on the line $x_1 + x_2 = 1$ with only 2 vertices corresponding to $x_1 = e^{\mathbf{d}} x_2$ and $x_2 = e^{\mathbf{d}} x_1$. Since these are linearly independent, they must be the posteriors of a kernel hyper Δ_K which is also the only vertex hyper in the space. The result follows from Lem. 6.13. □

The optimal mechanism on 2 inputs for $\varepsilon = \ln 2$ is depicted in Figure 6.1.[16]

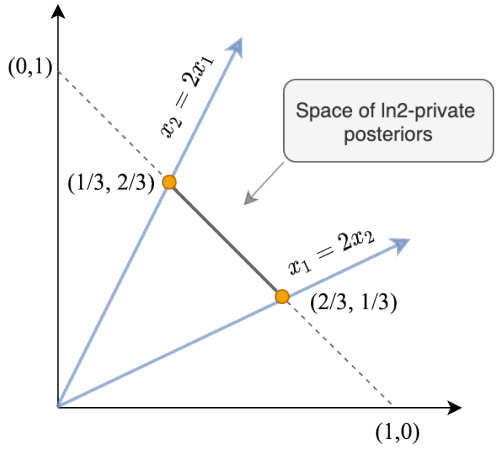


Figure 6.1: The universally optimal mechanism for 2 inputs for $\varepsilon = \ln 2$ and $\mathbf{d}(x_1, x_2) = 1$. The blue lines show the constraints defining the convex region (grey line segment). The region has exactly one mechanism defined by the 2 vertices (orange points).

However, for $n > 2$ inputs we find that there can be no universally optimal mechanisms.

THEOREM 6.15 (Impossibility of Universally Optimal Mechanisms)**.** For $n > 2$ there are no universally optimal $\mathbf{d}$-private mechanisms over n inputs.

Proof. (Sketch) The space of hypers on $n > 2$ inputs is formed from the intersection of at least $2(n-1)$ hyperplanes, thus has more than n vertices, which cannot

[16] We re-introduce the ε parameter explicitly here, remembering that we have been incorporating it into the metric $\mathbf{d}$ for notational convenience.

be linearly independent (as vectors). Therefore we can construct more than one valid hyper from these vertices – using different convex combinations of posteriors – implying there are multiple vertex mechanisms, thus multiple kernel mechanisms (Prop. 2) and therefore no universally optimal mechanisms (Lem. 6.13). (See Appendix §A.2 for details.) □

This result is not the same as the one of [40]; ours is weaker in that the impossibility is over *all* consumers, but does not (yet) say anything about the impossibility of universal optimality for classes of loss functions. This will be the focus of the next section.

6.4.2 Characterising universal $\mathcal{L}$-optimality

Our goal now is to describe universal optimality under restricted classes of loss functions in their generality, rather than restricting our definition to a particular class; however in later sections when we seek particular optimal mechanisms we will have to make recourse to the particulars of the loss function classes.

We again make use of refinement: although we do not have a structural characterisation for refinement under restricted loss functions, we observe that any $\mathcal{L}$-optimal mechanism M cannot have a strict anti-refinement M', because then M' would also be better than M for the class $\mathcal{L}$. So we can – once more – use kernel mechanisms, this time to understand $\mathcal{L}$-optimality.

THEOREM 6.16 (Characterisation of Universally $\mathcal{L}$-Optimal Mechanisms)**.** Every universally $\mathcal{L}$-optimal mechanism is a refinement of a convex combination of universally $\mathcal{L}$-optimal kernel mechanisms.

Proof. (Sketch) If M is universally $\mathcal{L}$-optimal then there is some vertex mechanism V such that $V \sqsubseteq M$ (Cor. 6.12) and so V must also be universally $\mathcal{L}$-optimal. The result follows from showing that V can be decomposed into kernel mechanisms, each of which is universally $\mathcal{L}$-optimal. (See Appendix §A.2 for details). □

This tells us that the $\mathcal{L}$-optimal *kernel* mechanisms generate the class of universally $\mathcal{L}$-optimal mechanisms. Note that Thm. 6.16 does not mean that every refinement of $\mathcal{L}$-optimal kernel mechanisms is $\mathcal{L}$-optimal – this clearly cannot be true – but it means that we can focus our attention on just the kernel mechanisms, knowing that these will be *as good* as any other $\mathcal{L}$-optimal mechanism in the space. Moreover, these kernel mechanisms will be at least as good on *every other* loss function since they are minimal in the refinement chain.

We know that universally $\mathcal{L}$-optimal mechanisms exist: the optimality result of Ghosh et al. gives us an example for the monotonic class. Our goal in the next section is to look for other examples by examining classes of loss functions and their properties.

6.5 Reasoning about Optimality

In this section we introduce our algebra for reasoning about optimality. We begin with an analysis of loss functions, concluding with an analysis of mechanisms which admit optimality.

6.5.1 Loss Function Algebra

We begin our exploration with an examination of properties of loss functions which will allow the construction of larger classes of functions for which optimality holds. Proofs for this section can be found in Appendix §A.3.

For $a, b \geq 0$ and loss functions ℓ, ℓ' we define:

$$(\ell * a)(w, x) := \ell(w, x) \times a \tag{6.3}$$

$$(\ell + a)(w, x) := \ell(w, x) + a \tag{6.4}$$

$$(\ell - a)(w, x) := \ell(w, x) - a \tag{6.5}$$

$$(\ell * a + \ell' * b)((w_1, w_2), x) := \ell(w_1, x) \times a + \ell'(w_2, x) \times b \tag{6.6}$$

Note that we can extend Defs (6.4), (6.5) to arbitrary functions $q : \mathcal{X} \to \mathbb{R}$:

$$(\ell + q)(w, x) := \ell(w, x) + q(x) \tag{6.7}$$

The following results are easy to show; firstly for prior ℓ-uncertainties we have, for any prior $\pi : \mathbb{D}\mathcal{X}$ and $a, b \geq 0$:

$$U_{\ell * a}(\pi) = U_\ell(\pi) \times a \tag{6.8}$$

$$U_{\ell + a}(\pi) = U_\ell(\pi) + a \tag{6.9}$$

$$U_{\ell - a}(\pi) = U_\ell(\pi) - a \tag{6.10}$$

$$U_{\ell * a + \ell' * b}(\pi) = U_\ell(\pi) \times a + U_{\ell'}(\pi) \times b \tag{6.11}$$

And for functions $q : \mathcal{X} \to \mathbb{R}$:

$$U_{\ell + q}(\pi) = U_\ell(\pi) + \sum_x \pi_x q(x) \tag{6.12}$$

These properties flow through to posterior ℓ-uncertainties; for any prior $\pi : \mathbb{D}\mathcal{X}$, $a, b \geq 0$ and channel C we have:

$$U_{\ell * a}[\pi \triangleright C] = U_\ell[\pi \triangleright C] \times a \tag{6.13}$$

$$U_{\ell + a}[\pi \triangleright C] = U_\ell[\pi \triangleright C] + a \tag{6.14}$$

$$U_{\ell - a}[\pi \triangleright C] = U_\ell[\pi \triangleright C] - a \tag{6.15}$$

$$U_{\ell * a + \ell' * b}[\pi \triangleright C] = U_\ell[\pi \triangleright C] \times a + U_{\ell'}[\pi \triangleright C] \times b \tag{6.16}$$

And again for functions $q: \mathcal{X} \to \mathbb{R}$:

$$U_{\ell+q}[\pi \triangleright C] \;=\; U_\ell[\pi \triangleright C] + \sum_x \pi_x q(x) \tag{6.17}$$

We can now deduce some useful refinement properties. In the following we write $M \sqsubseteq_\ell M^*$ to mean $U_\ell[\pi \triangleright M] \leq U_\ell[\pi \triangleright M^*]$ for all $\pi \in \mathbb{D}\mathcal{X}$.

Property 4. If $M \sqsubseteq_\ell M^*$ then $M \sqsubseteq_{(\ell * a)} M^*$ for $a \geq 0$.

Property 5. If $M \sqsubseteq_\ell M^*$ then $M \sqsubseteq_{(\ell + a)} M^*$ for $a \geq 0$.

Property 6. If $M \sqsubseteq_\ell M^*$ then $M \sqsubseteq_{(\ell - a)} M^*$ for $a \geq 0$.

Property 7. If $M \sqsubseteq_\ell M^*$ and $M \sqsubseteq_{\ell'} M^*$ then $M \sqsubseteq_{(\ell * a + \ell' * b)} M^*$ for $a, b \geq 0$.

Property 8. If $M \sqsubseteq_\ell M^*$ then $M \sqsubseteq_{(\ell + q)} M^*$ for $q: \mathcal{X} \to \mathbb{R}$.

The above properties allow us to reason about optimality on larger classes of loss functions by leveraging optimality results for known classes.

Finally, we present the following important result which explains why it is always safe to reason about loss functions wrt the action of the uniform distribution.[17]

We first define for prior $\pi: \mathbb{D}\mathcal{X}$ and loss function ℓ:

$$(\pi \star \ell)(w, x) \quad := \quad \pi_x \ell(w, x) \tag{6.18}$$

We now have the following:

Lemma 6.17. For any loss function ℓ, channel M and prior π,

$$U_\ell[\pi \triangleright M] \;=\; n \times U_{\pi \star \ell}[\upsilon \triangleright M]$$

where υ is the uniform prior and $n = |\mathcal{X}|$.

6.5.2 Classes of Loss Functions

In this section we explore some distinguished classes of loss functions – including the monotonic class – for which we find optimality results.[18]

We start by noting that there exists a trivial loss function for which all mechanisms are universally optimal.

Definition 6.5.1 (Trivial Loss Function)**.** A loss function $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ is

[17] This result is known in its dual form via gain functions [34].

[18] We note that there are other loss function classes of interest than the ones described here – the property loss functions for example – but we have included only the classes for which we have optimality results.

called *trivial* if there exists a $w^* \in \mathcal{W}$ such that $\ell(w^*, x) \leq \ell(w, x)$ for all $x \in \mathcal{X}$ and $w \in \mathcal{W} \setminus \{w^*\}$.

Trivial loss functions describe consumers whose guesses are independent of the mechanism, and thus each consumer's loss depends only on their prior. For example, the loss function:

ℓ	w_1	w_2
x_0	0	1
x_1	1	2
x_2	1	3

is trivial since the action w_1 always produces a smaller loss than the action w_2 on any distribution on $\mathcal{X}$. In essence, the consumer will always choose action w_1 as loss-minimising, regardless of what output they observe from the channel. We will call the set of all such loss functions the "trivial class" of loss functions, which we designate by $\mathcal{L}_0$.

We will also make use of two operations on loss functions. We denote by $\ell{\downarrow}X$ the restriction of the loss function $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ to the subset $X \subset \mathcal{X}$, defined

$$\ell{\downarrow}X(w, x) \quad := \quad \ell(w, x) \qquad \text{for all } x \in X \ .$$

We also denote by $\ell{\uparrow}\mathcal{X}$ the 'lifting' of the loss function $\ell : \mathcal{W} \times X \to \mathbb{R}_{\geq 0}$ to the superset $\mathcal{X} \supset X$, defined

$$\ell{\uparrow}\mathcal{X}(w, x) \quad := \quad \begin{cases} \ell(w, x) & \text{if } x \in X \\ 0 & \text{if } x \in \mathcal{X} \setminus X \end{cases} \tag{6.19}$$

We now describe monotonic loss functions, which are a class of loss functions for which the optimality results of Ghosh et al. [39] and the impossibility results of Brenner and Nissim [40] apply.

Definition 6.5.2 (Monotonic Loss Function)**.** The loss function $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}$ is said to be *monotonic in* $\mathbf{d}$ if there is some mapping $\alpha : \mathcal{W} \to \mathcal{X}$, and function $f : \mathbb{R} \times \mathcal{X} \to \mathbb{R}$ that is monotone (non-decreasing) in its first argument, such that

$$\ell(w, x) \quad = \quad f(\mathbf{d}(\alpha(w), x), x) \ .$$

Monotonic loss functions penalise guesses that are further away from the true value, but give no additional penalty for guessing low rather than high. Notice that Bayes' risk is a monotonic function on the Euclidean metric (for example), as is the average distance function $\ell(w, x) = |w - x|$ and the average squared distance $\ell(w, x) = (w - x)^2$. Notice also that the property of monotonicity depends on the metric $\mathbf{d}$. This is illustrated in Example 6.4.

Remark 6.5.1. Monotonicity properties of loss functions depend on the metric

even though the expected loss for a consumer does not. These properties are considered useful and important in the literature – indeed, for $\mathbf{d}_2$ they correspond to well-known loss functions – however, they are not useful in our Bayesian setting, since the actions w of a consumer are not meaningful in and of themselves. Our reference to these functions solely serves the purpose of comparing our results with those of the literature. Meanwhile, we will continue to suggest and use "non-monotonic" loss functions, for which we have already argued the case.

Our last distinguished class of loss functions we call *extended* loss functions. These are loss functions which ignore inputs outside some subset $X \subset \mathcal{X}$; this could model a consumer who is only interested in a subset of the released information.

Definition 6.5.3 (Extended Loss Function)**.** Let $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ be a loss function and let $X \subset \mathcal{X}$. We call ℓ an *extended loss function* on X if $(\ell{\downarrow}X){\uparrow}\mathcal{X} = \ell$.

Example 6.4 (Examples of loss functions)**.**

ℓ_m	w_1	w_2	w_3
x_1	0	1/2	3/5
x_2	3/4	1/4	3/4
x_3	1	2/3	1/3

ℓ_e	w_1	w_2
x_1	1	0
x_2	0	1
x_3	1	0
x_4	0	0

ℓ_t	w_1	w_2
x_1	1/2	1/2
x_2	3/4	3/4
x_3	1	1

ℓ_s	w_1	w_2
x_1	1/2	1
x_2	1/2	1
x_3	1/2	1
x_4	0	0

The loss function ℓ_m is monotonic on $\mathbf{d}_2$ where $\mathbf{d}_2(x_i, x_j) = |i - j|$ using the mapping $\alpha(w_i) = x_i$. However, it is *not* monotonic on $\mathbf{d}_\mathrm{D}$, for example.
The loss function ℓ_e is an extended loss function on $\{x_1, x_2, x_3\}$ which distinguishes inputs x_2 and x_1, x_3 and ignores input x_4.
The loss function ℓ_t is a trivial loss function which is also monotonic on $\mathbf{d}_2$ and $\mathbf{d}_\mathrm{D}$ using the mapping $\alpha(w_1) = \alpha(w_2) = x_1$.
Finally, the loss function ℓ_s is trivial and extended but not monotonic.

6.5.3 Properties of Loss Function Classes

We will now examine the properties of each of the aforementioned loss function classes. These properties will be used to prove the optimality results to come.

Monotonic loss functions enjoy many interesting properties which may explain why they exhibit such nice behaviour wrt optimality. The following properties apply for monotonicity wrt any metric $\mathbf{d}$.

Property 9. If ℓ is a monotonic loss function then $\ell * a$, $\ell + a$ and $\ell - a$ are monotonic loss functions for any function $a(x)$.

Property 10. If ℓ is a monotonic loss function then $\pi \star \ell$ is a monotonic loss function for any $\pi \in \mathbb{D}\mathcal{X}$.

Property 11. If ℓ is a monotonic loss function then $\ell{\downarrow}X$ is monotonic for any $X \subset \mathcal{X}$.

Property 12. If ℓ is a monotonic loss function then $\ell{\uparrow}X$ is monotonic for any $X \supset \mathcal{X}$.

We now have the following property of extended loss functions. For this, we denote by $M{\downarrow}X$ the restriction of the channel matrix M to the set of inputs X defined by simply removing all rows of M outside of X.

Lemma 6.18. Let $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ be an extended loss function on $X \subset \mathcal{X}$, and M be a mechanism on $\mathcal{X}$. Then for any $\pi : \mathbb{D}\mathcal{X}$

$$U_\ell[\pi \triangleright M] = U_{\ell{\downarrow}X}[\pi \triangleright M{\downarrow}X]$$

Proof. We reason as follows:

$$\begin{aligned}
& U_\ell[\pi \triangleright M] \\
= \;& U_{(\ell{\downarrow}X){\uparrow}\mathcal{X}}[\pi \triangleright M] && \text{“Def. 6.5.3”} \\
= \;& \sum_{y \in \mathcal{Y}} \min_{w \in \mathcal{W}} \sum_{x \in X} M_{x,y} \pi_x \ell(w,x) && \text{“Def. 6.2.1, } \ell \text{ is 0 outside } X\text{”} \\
= \;& \sum_{y \in \mathcal{Y}} \min_{w \in \mathcal{W}} \sum_{x \in X} (M{\downarrow}X)_{x,y} \pi_x \ell{\downarrow}X(w,x) && \text{“Inner sum restricted to } X\text{”} \\
= \;& U_{\ell{\downarrow}X}[\pi \triangleright M{\downarrow}X] && \text{“Simplify, Def. 6.2.1”}
\end{aligned}$$

□

This tells us that extended loss functions essentially ignore the inputs that they are ‘zeroed out’ on, and optimality therefore only depends on the non-zeroed parts of the loss function.

6.5.4 Identifying Optimal Mechanisms

We now ask: for which kernel mechanisms can we find a class $\mathcal{L}$ of non-trivial loss functions such that optimality holds? In this section we will show how to identify optimal mechanisms based on their structure on subsets of inputs. We will lean on the optimality results that we have already: the universally optimal mechanism T on 2 inputs (from Thm. 6.14), and the universally $\mathcal{L}$-optimal mechanism (the geometric, G) for the class of monotonic loss functions (from [39]). Our strategy is to identify mechanisms with the same structure as T or G on some subset of inputs and generate loss functions defined over that subset, which are consequently universally optimal for that mechanism.

We first remark that there is a class of loss functions $\mathcal{L}$ for which *every* mechanism is universally $\mathcal{L}$-optimal.

> **Lemma 6.19.** Let $\mathcal{L}_0^n$ be the class of trivial loss functions on n inputs. Then any mechanism M on n inputs is universally $\mathcal{L}_0^n$-optimal.

Our search for optimal mechanisms should clearly be restricted to *non-trivial* classes of loss functions.

We now show how the structure of mechanisms on pairs of inputs can be used to construct loss functions for which a mechanism is ℓ-optimal. We recall Thm. 6.14 which tells us that there is a universally optimal mechanism on 2 inputs which we denote by T, where universal optimality holds for all loss functions. Given any mechanism M, we can consider mechanisms of the form $M{\downarrow}\{x_1, x_2\}$ for any pair of inputs x_1, x_2. If it turns out that one of these mechanisms is *equivalent* to T (recall channel equivalence from §6.3.1) then loss functions which are non-zero only on $\{x_1, x_2\}$ behave *as though* there are only those 2 inputs, and the channel behaves like the universally optimal channel T on those inputs.

As an example consider the following mechanism.

M	y_0	y_1	y_2
x_0	$2/3$	$1/6$	$1/6$
x_1	$1/3$	$1/3$	$1/3$
x_2	$1/2$	$1/4$	$1/4$

Observe that $M{\downarrow}\{x_0, x_1\}$ yields a mechanism which is equivalent to the universally optimal mechanism T (by merging the columns y_1, y_2 of $M{\downarrow}\{x_0, x_1\}$). Therefore, any loss function defined over $\{x_0, x_1\}$ and ignoring x_2, such as

ℓ	w_0	w_1
x_0	3	1
x_1	0	2
x_2	0	0

will realise universal ℓ-optimality for M. And this holds regardless of the structure of M on x_2. However, the mechanism $M{\downarrow}\{x_1, x_2\}$ is not equivalent to T – the ratio of elements in each column differs – so loss functions defined over $\{x_1, x_2\}$ and ignoring x_0 will not (necessarily) yield universal optimality for M.

> **Remark 6.5.2.** It is possible in some instances that we can get a universal optimality result even when the restriction to 2 inputs is not T – it depends on the structure of other kernel mechanisms in the space. We will see an example of this in §6.7 when we study optimality results for the geometric mechanism.

We formalise the above observations with the following results.

Lemma 6.20. Let M be a $\mathbf{d}$-private mechanism on $\mathcal{X}$. If there exists a pair of inputs x_1, x_2 such that $M{\downarrow}\{x_1, x_2\} \equiv T$ then M is universally ℓ-optimal for any extended loss function ℓ on $\{x_1, x_2\}$.

Proof. We reason as follows:

$$
\begin{array}{lll}
& U_\ell[\pi \triangleright M] & \\
= & U_{\ell{\downarrow}\{x_1,x_2\}}[\pi \triangleright M{\downarrow}\{x_1,x_2\}] & \text{“Lem. 6.18”} \\
= & U_{\ell{\downarrow}\{x_1,x_2\}}[\pi \triangleright T] & \text{“Equivalence assumption on } T\text{”} \\
\leq & U_{\ell{\downarrow}\{x_1,x_2\}}[\pi \triangleright C{\downarrow}\{x_1,x_2\}] & \text{“Universal optimality of } T \text{ for any } \mathbf{d}\text{-private } C\text{”} \\
= & U_\ell[\pi \triangleright C] & \text{“Lem. 6.18”}
\end{array}
$$

□

Note that the converse of Lem. 6.20 is not true – if M is universally ℓ-optimal then it does not follow that its restriction to 2 inputs is equivalent to T.

The following result extends the above Lem. 6.20 to arbitrary subsets of inputs, provided we know an optimality result on the subset.

Lemma 6.21. Let M be a $\mathbf{d}$-private mechanism on $\mathcal{X}$ which is universally ℓ-optimal mechanism for some loss function ℓ. Let C be a $\mathbf{d}$-private mechanism on $Z \supset \mathcal{X}$ such that $C{\downarrow}\mathcal{X} \equiv M$. Then C is universally ℓ'-optimal for any extended loss function ℓ' on $\mathcal{X}$.

Proof. Let D be any $\mathbf{d}$-private mechanism on Z. We reason as follows:

$$
\begin{array}{lll}
& U_{\ell'}[\pi \triangleright D] & \\
= & U_\ell[\pi \triangleright D{\downarrow}X] & \text{“Lem. 6.18”} \\
\geq & U_\ell[\pi \triangleright M] & \text{“Optimality of } M\text{”} \\
= & U_\ell[\pi \triangleright C{\downarrow}X] & \text{“Assumption”} \\
= & U_{\ell'}[\pi \triangleright C] & \text{“Lem. 6.18”}
\end{array}
$$

Thus C is universally ℓ'-optimal. □

Next, notice that instead of ‘zero-ing out’ the loss function we can ‘zero out’ the prior, since universal optimality (over all priors) implies universal optimality on non-full-support priors.

Lemma 6.22. Let $M : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be a $\mathbf{d}$-private mechanism and let $\pi : \mathbb{D}\mathcal{X}$ be a prior with $\lceil \pi \rceil = X$ for $X \subset \mathcal{X}$. Then

$$U_\ell[\pi \triangleright M] = U_{\ell{\downarrow}X}[\pi \triangleright M{\downarrow}X] \,.$$

Proof. We reason as follows:

$$
\begin{array}{lll}
& U_\ell[\pi \triangleright M] & \\
= & \sum_{y \in \mathcal{Y}} \min_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} M_{x,y} \pi_x \ell(w, x) & \text{“Def. 6.2.1”}
\end{array}
$$

$$
\begin{aligned}
&= \sum_{y\in\mathcal{Y}} \min_{w\in\mathcal{W}} \sum_{x\in X} M_{x,y}\pi_x \ell(w,x) && \text{“}\pi \text{ is 0 outside } X\text{”}\\
&= \sum_{y\in\mathcal{Y}} \min_{w\in\mathcal{W}} \sum_{x\in X} (M{\downarrow}X)_{x,y}\pi_x \ell{\downarrow}X(w,x) && \text{“Restriction to } X\text{”}\\
&= U_{\ell\downarrow X}[\pi \triangleright M{\downarrow}X] && \text{“Simplify, Def. 6.2.1”}
\end{aligned}
$$

□

Corollary 6.23. If M is universally ℓ-optimal then $M{\downarrow}X$ is universally $\ell{\downarrow}X$-optimal for all $X \subset \mathcal{X}$.

We can also compare mechanisms over subsets of inputs if their restricted mechanisms are in refinement.

Lemma 6.24. Let M, M' be $\mathbf{d}$-private mechanisms on $\mathcal{X}$ and let $X \subset \mathcal{X}$. Then

$$M{\downarrow}X \sqsubseteq M'{\downarrow}X \implies U_\ell[\pi \triangleright M] \leq U_\ell[\pi \triangleright M']$$

for all priors $\pi : \mathbb{D}\mathcal{X}$ and all extended loss functions ℓ on X.

Proof. We reason as follows:

$$
\begin{aligned}
& U_\ell[\pi \triangleright M] && \\
&= U_{\ell\downarrow X}[\pi \triangleright M{\downarrow}X] && \text{“Lem. 6.18”}\\
&\leq U_{\ell\downarrow X}[\pi \triangleright M'{\downarrow}X] && \text{“Lem. 6.2, Def. 6.2.4”}\\
&= U_\ell[\pi \triangleright M'] && \text{“Lem. 6.18”}
\end{aligned}
$$

□

Finally, we consider the specific case of a totally ordered input space, such as the integers wrt $\mathbf{d}_2$ on which the geometric mechanism is defined. In this case we observe that we can always find non-trivial loss functions which yield optimality on *every* mechanism in the space.

Lemma 6.25. Let $(\mathcal{X}, \mathbf{d})$ be a metric space which is linearly ordered. Then every $\mathbf{d}$-private kernel mechanism on $\mathcal{X}$ is universally ℓ-optimal for extended loss functions ℓ on adjacent inputs in $\mathcal{X}$.

Proof. Consider any 2 adjacent inputs $x_1, x_2 \in \mathcal{X}$, where adjacency is defined wrt the linear order. Let M be any kernel mechanism. Since $\mathcal{X}$ is linearly ordered then all triangle inequalities are *equalities*. This means that any differential privacy constraint which holds with equality over a pair of inputs x_i, x_j, must also hold with equality over every input between x_i and x_j. For each posterior in $\lceil [\upsilon \triangleright M] \rceil$ we must have that the differential privacy constraint holds with equality between x_1 and x_2 (since M is a kernel mechanism and $\mathcal{X}$ is). ie. Either $M_{x_1,y} = e^{\mathbf{d}(x_1,x_2)} M_{x_2,y}$ or $M_{x_2,y} = e^{\mathbf{d}(x_1,x_2)} M_{x_1,y}$ for each $y \in \mathcal{Y}$. Second, we can find 2 posteriors with these constraints holding in opposite directions. (eg. by

taking one posterior and 'reversing' all the constraints on adjacent values.) Then by restricting M to $\{x_1, x_2\}$ we have that $M{\downarrow}\{x_1, x_2\} \equiv T$ and the result follows from Lem. 6.20. □

6.6 Application to Oblivious Mechanisms

We now have the theoretical tools that we need to address optimality in differential privacy. Recall that we motivated this exploration of optimality by two results in the literature: one on counting queries and the other on sum queries, which we abstracted to $\mathbf{d}_2$-private mechanisms and $\mathbf{d}_{\mathrm{D}}$-private mechanisms respectively. In the next sections we re-examine these spaces using the perspective of QIF and our characterisations in terms of refinement.

6.7 Optimality for $\mathbf{d}_2$-Private Mechanisms on $\{0 \ldots N\}$

In this section we consider optimality for $\mathbf{d}_2$-private mechanisms on the space of inputs $\{0 \ldots N\}$.[19] These mechanisms include the noise-adding mechanisms designed for "counting queries" for which the optimality results of Ghosh et al. apply.

6.7.1 Kernel mechanisms

We begin by noting that we can enumerate all of the $\mathbf{d}_2$-private kernel mechanisms: these are the ones for which the linear constraints in Eqn (6.2) hold with equality. On n inputs there are $2 \times (n-1)$ linear constraints resulting in 2^{n-1} vertices in the space, and the kernel mechanisms are those with at most n (linearly independent) vertices whose convex hull contains the uniform distribution. Table 6.1 enumerates the kernel mechanisms for n up to 5.

Table 6.1: Kernel mechanisms in the space of $\mathbf{d}_2$-private mechanisms.

Dimensions	Vertices	Kernel Mechanisms
2	2	1
3	4	2
4	8	11
5	16	187

We can see that the space is rich with kernel mechanisms, although none of note have been identified in the literature apart from the geometric mechanism, which – we repeat – is known to be universally $\mathcal{L}$-optimal for the class of *monotonic* loss functions [39]. (We will show in the next section that the geometric mechanism is a kernel mechanism).

However, immediately from Lem. 6.25 we have that *every* kernel mechanism in this space is universally $\mathcal{L}$-optimal for a class of loss functions $\mathcal{L}$. This class is generated from the loss

[19] As mentioned earlier, these results generalise to quantised reals $\{0, q, 2q, \ldots\}$. We use integer domains for simplicity of presentation.

functions defined over 2 inputs (which, from Prop. 12, are monotonic). This result means that every kernel mechanism in this space is *equally* good for consumers modelled using any of these functions. Note that using our loss function algebra we can construct a very large class of loss functions from the set defined over adjacent inputs.

6.7.2 The geometric mechanism

The geometric mechanism is the most well-known mechanism in this space (and arguably the *only* known mechanism in this space of note). It has two instantiations – the (infinite) geometric with outputs over the whole of $\mathbb{Z}$ (Def. 3.4.1) and the truncated geometric, in which the output domain matches the input domain (Def. 3.4.2). We have previously noted (Chapter 4, Lem. 4.15) that these have the same leakage properties; in other words, they are *equivalent* as channels and thus produce the same hyper-distribution. It turns out that this hyper is a kernel hyper.

Lemma 6.26. The (infinite/truncated) geometric mechanism is a $\mathbf{d}_2$-private kernel mechanism.

Proof. Denote by G, G_t the geometric and truncated geometric mechanisms respectively. The $\mathbf{d}$-privacy constraints ((6.2)) on both G and G_t hold with equality (by construction) and therefore on the posteriors of $[\upsilon \triangleright G]$ and $[\upsilon \triangleright G_t]$ (by Lem. 6.7) – which (by Lem. 4.15) are the same hyper, call it Δ_G. We have now that Δ_G is a vertex hyper (Def. 6.3.1). Since G_t is invertible [36] its columns are linearly independent and therefore so are the posteriors of $[\upsilon \triangleright G_t] = \Delta_G$ (by linearity). And so Δ_G is a kernel hyper (Def. 6.3.2) corresponding to both G and G_t. □

Figure 6.2 shows the geometric mechanism in the space of 3 inputs for $\varepsilon = \ln 2$.

In addition to the known results on monotonic loss functions, we now present a set of loss functions – which includes non-monotonic functions – for which optimality of the geometric mechanism holds.

Lemma 6.27. Let $\mathcal{X} = \{x_1, x_2, \ldots x_n\}$ with $|x_{i+1} - x_i| = 1$. For any $a, b, c, d \in \mathbb{R}$, let $\mathcal{L}$ be the set of loss functions defined as:

$$\ell(w_1, x_1) = a \;;\; \ell(w_1, x_2) = 1 \;;\; \ell(w_1, x_3) = b$$
$$\ell(w_2, x_1) = c \;;\; \ell(w_2, x_2) = 1 \;;\; \ell(w_2, x_3) = d$$
$$\ell(w_i, x_j) = 0 \;\; \textit{otherwise}$$

Then the geometric mechanism on $\mathcal{X}$ is universally $\mathcal{L}$-optimal.

Proof. (Sketch) We sketch a proof for the case $\varepsilon = \ln 2$. The full proof can be found in Appendix §A.4. First, observe that on 3 inputs, there are 2 kernel hypers

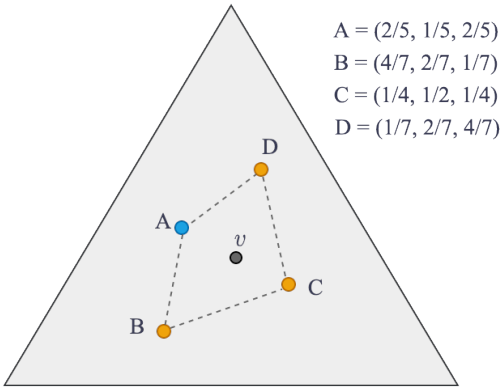


Figure 6.2: The space of $\mathbf{d}_2$-private hypers on 3 inputs $\{0,1,2\}$. The geometric mechanism for $\varepsilon = \ln 2$ consists of the 3 orange vertices (B, C, D); these are linearly independent and thus form a kernel mechanism. The only other kernel mechanism consists of the posteriors A and C, since their convex hull contains the uniform distribution (υ).

in the space of $\mathbf{d}_2$-private hypers (see Figure 6.2). Their corresponding mechanisms are

G_3	y_1	y_2	y_3
x_1	$2/3$	$1/6$	$1/6$
x_2	$1/3$	$1/3$	$1/3$
x_3	$1/6$	$1/6$	$2/3$

H_3	y_1	y_2
x_1	$2/3$	$1/3$
x_2	$1/3$	$2/3$
x_3	$2/3$	$1/3$

where we use the subscript 3 to denote the number of inputs they are defined over. Notice now that $H{\downarrow}\{x_1,x_3\}$ is the trivial mechanism[20] and which is anti-refined by everything, ie. $G_3{\downarrow}\{x_1,x_3\} \sqsubseteq H_3{\downarrow}\{x_1,x_3\}$. Which by Lem. 6.24 gives $U_{\ell_2}[\pi \triangleright G_3] \leq U_{\ell_2}[\pi \triangleright H_3]$ for all priors π and all extended loss functions ℓ_2 on $\{x_1,x_3\}$. Since G_3 and H_3 are the only kernel mechanisms in the space, it follows that $U_{\ell_2}[\pi \triangleright G_3] \leq U_{\ell_2}[\pi \triangleright C]$ for all channels C in the space. And since ℓ is an extended loss function on $\{x_1,x_3\}$ it holds for ℓ.

Secondly we notice that the geometric mechanism has a special property, namely that $G \downarrow \{x_1,\ldots,x_k\}$ is equivalent (in terms of leakage properties) to the geometric mechanism defined on $\{x_1,\ldots,x_k\}$. In particular, we have $G_N \downarrow \{x_1,x_2,x_3\} \equiv G_3$.

Putting these together, we deduce:

$$
\begin{array}{lll}
 & U_\ell[\pi \triangleright G_N] & \\
= & U_{\ell\downarrow\{x_1,x_2,x_3\}}[\pi \triangleright G_3] & \text{“Lem. 6.24, using property of geometric”} \\
\leq & U_{\ell\downarrow\{x_1,x_2,x_3\}}[\pi \triangleright C] & \text{“For all channels } C\text{, as shown above”} \\
= & U_\ell[\pi \triangleright C_N] & \text{“Lem. 6.21, lifting } C \text{ to } C_N \text{ on } N \text{ inputs”}
\end{array}
$$

□

[20] The trivial mechanism, also called **1** is the channel which leaks nothing.

6.7.3 Other optimal mechanisms

Although the geometric mechanism has shown to be optimal for a very large class of loss functions, we know that there must exist loss functions for which the other kernel mechanisms in the space are *strictly* better: otherwise the geometric mechanism would be universally optimal in the space, contradicting Thm. 6.15. We can use this idea to find a consumer who prefers another mechanism to the geometric.

Observe from Figure 6.2 that on 3 inputs there are only 2 kernel mechanisms in the space: the geometric mechanism and another mechanism, call it H, formed from the vertices A and C see-sawing the uniform distribution. Since $G \not\sqsubseteq H$ (by Prop. 3) there must exist a loss function and a prior for which the expected loss under H is strictly lower than the loss under G (property of refinement). We can find one such loss function (using Thm. 4.12 from Chapter 4): we get, on the space of $\ln 2{\cdot}\mathbf{d}_2$ mechanisms on 3 inputs, for uniform prior υ, the loss function

$$\ell = \begin{array}{c} \begin{matrix} w_1 & w_2 \end{matrix} \\ \begin{bmatrix} 7/10 & 3/10 \\ 0 & 1 \\ 7/10 & 3/10 \end{bmatrix} \end{array}$$

produces lower expected loss on H than on G. Thinking of the inputs as heights of individuals: 'Tall', 'Average' and 'Short', ℓ models a consumer who is interested in distinguishing between individuals who are 'Average' and 'not Average'. Since H is the only other kernel mechanism in this space, we have that H is *optimal* for this consumer. However, we do not have a universal ℓ-optimality result (ie. over all priors) for H; we leave open this possibility for future work.

The above results suggest that the dominance of the geometric mechanism in the literature is justified. We suggest that further examination of the symmetry properties of this mechanism using our algebraic framework could reveal other interesting optimality results. We leave the exploration of these ideas to future work.

6.8 Optimality for $\mathbf{d}_{\mathrm{D}}$-Private Mechanisms

We now investigate optimality for the space of $\mathbf{d}_{\mathrm{D}}$-private mechanisms on n inputs. This space includes the mechanisms for "sum queries" which we discussed in §6.2.3, although in that discussion we restricted the input space to integers $\{0 \ldots N\}$. Here we generalise to any input space on n elements – outfitted with the discrete metric.

6.8.1 Kernel mechanisms

As for the space on $\mathbf{d}_2$-private mechanisms, we can again enumerate the mechanisms in this space using (6.2) and a careful observation of exactly which constraints can hold with equality at once. We note that for an input space $\mathcal{X}$ of size n, the vertices of the convex region are points of intersection of $n-1$ hyperplanes, corresponding to $n-1$ distinct privacy constraints. Since there are only 2 possible distances under the discrete metric, each vertex can only contain 2

possible values, and so the number of possible vertices is given by $\sum_{i=1}^{n-1} \binom{n}{i} = 2^n - 2$.[21] From this vertex set we enumerate all of the kernel hypers (and therefore kernel *mechanisms*) by solving the appropriate linear equations.[22] We present some of these results in Table 6.2.

Table 6.2: Enumerating the kernel mechanisms in the space of $\mathbf{d}_{\mathrm{D}}$-private mechanisms.

Dimensions	Vertices	Kernel Mechanisms
2	2	1
3	6	5
4	14	41
5	30	1291

The kernel mechanisms in the space of $\mathbf{d}_{\mathrm{D}}$-private hypers on 3 inputs are illustrated in Figure 6.3.

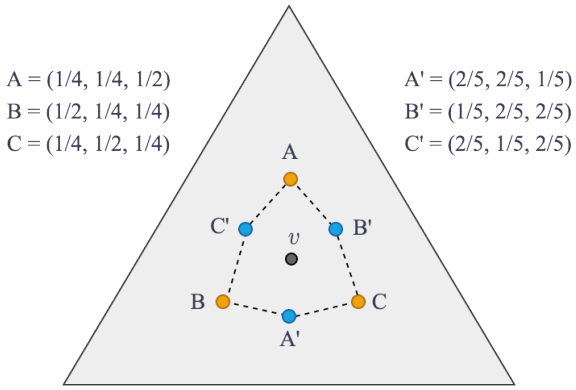


Figure 6.3: The space of $\mathbf{d}_{\mathrm{D}}$-private hypers over 3 inputs. The randomised response mechanism for $\varepsilon = \ln 2$ corresponds to the 3 orange vertices (A, B, C); these are linearly independent and thus form a kernel hyper. There are 4 other kernel hypers in this space: the hyper formed from posteriors (A', B', C') and the 3 hypers (A, A'), (B, B') and (C, C').

We again notice that this space is rich with kernel mechanisms, although the only mechanism of note (from the literature) is the randomised response mechanism. We show in the next section that this is indeed a kernel mechanism.

We first recall the impossibility result of Brenner and Nissim [40] which was proven specifically for the Bayes' risk loss function; we restate it as: for the Bayes' risk loss function **br** there are no universally **br**-optimal mechanisms in the class of mechanisms for sum queries. Importantly, there are many other loss functions in this space – some of which are monotonic (on $\mathbf{d}_{\mathrm{D}}$). And in the following, we prove that universally optimal mechanisms for some of these loss functions; further, we present an example of a universally optimal mechanism for a monotonic loss function in Example 6.5.

[21] Thinking of each vertex as having either the 'high' or the 'low' value, this is the number of combinations for each choice of 'high' values.

[22] Namely, we need the vertices to be linearly independent and be able to average to the uniform distribution.

THEOREM 6.28. In the space of $\mathbf{d}_{\mathrm{D}}$-private hypers there exist non-trivial monotonic (on $\mathbf{d}_{\mathrm{D}}$) loss functions $\mathcal{L}$ and mechanisms which are universally $\mathcal{L}$-optimal.

Proof. Our proof is constructive - we show how to find a $\mathbf{d}_{\mathrm{D}}$-private mechanism with a structure over 2 inputs which enables universal optimality on the space of 2 inputs. Consider the following $\mathbf{d}_{\mathrm{D}}$-private hyper in the space of n inputs:

Δ	$k/k+j$	$j/k+j$
x_1	$1/k$	α/j
x_2	α/k	$1/j$
x_3	α/k	$1/j$
x_4	α/k	$1/j$
$\dots$	$\dots$	$\dots$
x_n	α/k	$1/j$

where $\alpha = e^{-\varepsilon}$ and k, j are normalising constants ensuring columns sum to 1.
It is easy to check that the posteriors average to the uniform distribution using the outer probabilities, hence the hyper corresponds to a proper mechanism K. We can likewise observe that K is $\mathbf{d}_{\mathrm{D}}$-private, and it is also a kernel mechanism since each posterior has $n-1$ tight constraints.
Now, if we construct $K{\downarrow}\{x_1, x_2\}$ we get:

$K{\downarrow}\{x_1,x_2\}$	y_1	y_2
x_1	$1/(k+j)$	$\alpha/(k+j)$
x_2	$\alpha/(k+j)$	$1/(k+j)$

which corresponds to the universally optimal mechanism on 2 inputs (Thm. 6.14). Therefore by Lem. 6.20 every loss function on $\{x_1, x_2\}$ can be extended to a (non-trivial) loss function ℓ on $\{x_1, x_2, \dots, x_n\}$ for which K is universally ℓ-optimal. For example, the following loss function will suffice:

ℓ	w_1	w_2
x_1	0	1
x_2	1	0
x_3	0	0
x_4	0	0
$\dots$	$\dots$	$\dots$
x_n	0	0

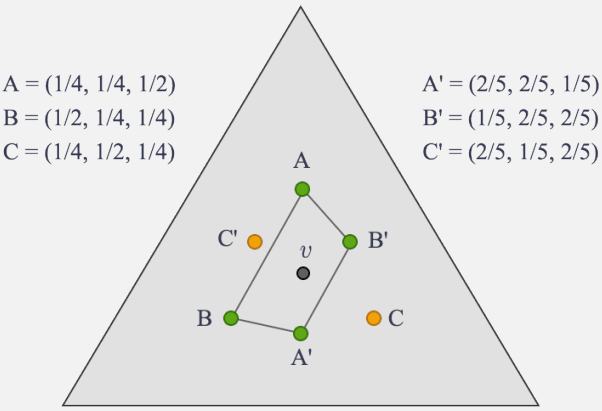
Since $\ell{\downarrow}\{x_1, x_2\}$ is monotonic on $\mathbf{d}_{\mathrm{D}}$, we have that ℓ is monotonic on $\mathbf{d}_{\mathrm{D}}$ (Prop. 12) and the mechanism K (with corresponding hyper Δ) is universally ℓ-optimal.

□

Note that the construction in the proof of Thm. 6.28 is general, and it is easy to find kernel

mechanisms in this space with the appropriate structure on pairs of inputs $\{x_i, x_j\}$. Their convex combinations (forming vertex mechanisms) are therefore also universally optimal. Likewise, there are many loss functions which can be constructed (using the algebraic properties of loss functions developed earlier) for which there are universally ℓ-optimal mechanisms in this space. We present a sample construction in Example 6.5.

Example 6.5 (Constructing a universally ℓ-optimal $\mathbf{d}_{\mathrm{D}}$-private mechanism)**.** We show the construction of a non-trivial *monotonic* loss function on $\mathbf{d}_{\mathrm{D}}$ for a vertex mechanism in the space of $\mathbf{d}_{\mathrm{D}}$-private hypers. The following diagram depicts the space on 3 inputs $\mathcal{X} = \{x_1, x_2, x_3\}$ for $\varepsilon = \ln 2$.



The green points A, A', B, B' comprise a vertex hyper Δ_V, itself composed of the kernel hypers $\Delta_{K_A} = AA'$ and $\Delta_{K_B} = BB'$ in some convex combination (not yet specified). ie. $\Delta_V = \lambda\Delta_{K_A} + (1-\lambda)\Delta_{K_B}$ where:

$$\Delta_{K_A} = \begin{bmatrix} 1/4 & 2/5 \\ 1/4 & 2/5 \\ 1/2 & 1/5 \\ 4/9 & 5/9 \end{bmatrix} \qquad \Delta_{K_B} = \begin{bmatrix} 1/2 & 1/5 \\ 1/4 & 2/5 \\ 1/4 & 2/5 \\ 4/9 & 5/9 \end{bmatrix}$$

It is easy to check that $K_A{\downarrow}\{x_1, x_3\} \equiv T$ and $K_B{\downarrow}\{x_1, x_3\} \equiv T$, where as usual we write T for the universally optimal mechanism on 2 inputs and K_A, K_B for the kernel mechanisms corresponding to hypers $\Delta_{K_A}, \Delta_{K_B}$ respectively. Now, using loss function algebra, we show that K_A, K_B are universally ℓ-optimal for the following loss function:

ℓ	w_1	w_2
x_1	3	0
x_2	1	1
x_3	0	3

We first construct $\ell_1 : \{x_1, x_3\} \times \mathcal{W} \to \mathbb{R}_{\geq 0}$ as $\ell_1(w_1, x_1) = \ell_1(w_2, x_3) = 1$ and $\ell_1(w_1, x_2) = \ell_1(w_2, x_1) = 0$. T is universally ℓ_1-optimal (by Thm. 6.14), and therefore so are $K_A{\downarrow}\{x_1, x_3\}$

and $K_B{\downarrow}\{x_1, x_3\}$ (by channel equivalence). Therefore, writing: $\ell_2 = \ell_1{\uparrow}\mathcal{X}$, $\ell_3 = (\ell_2 * 3)$, $\ell_4 = (\ell_3 + q(x))$ where $q(x_1) = q(x_3) = 0, q(x_2) = 1$, we reason:

$K_A{\downarrow}\{x_1, x_3\}, K_B{\downarrow}\{x_1, x_3\}$ are universally ℓ_1-optimal
$\Rightarrow$ K_A, K_B are universally ℓ_2-optimal "Lem. 6.20"
$\Rightarrow$ K_A, K_B are universally ℓ_3-optimal "Prop. 4"
$\Rightarrow$ K_A, K_B are universally ℓ_4-optimal "Prop. 8"
$\Rightarrow$ $\lambda K_A + (1-\lambda)K_B$ is universally ℓ_4-optimal "Thm. 6.16 (for $\lambda \in [0,1]$)"

Finally, we note that $\ell_4 = \ell$ and that ℓ is monotonic on $\mathbf{d}_{\mathrm{D}}$ using the mapping $\alpha(w_1) = x_3, \alpha(w_2) = x_1$. Therefore, for any choice of λ we find that the vertex mechanism V corresponding to hyper Δ_V is *universally ℓ-optimal*. Note that it is also ℓ_2, ℓ_3-optimal – and there are many possible constructions for which optimality holds.

The following corollary explains why we can always find universally optimal mechanisms in this space.

Corollary 6.29. Given any pair of inputs $x_1, x_2 \in \mathcal{X}$, there are always $\mathbf{d}_{\mathrm{D}}$-private mechanisms $M : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ for which $M{\downarrow}\{x_1, x_2\} \equiv T$.

Proof. Follows from the construction given in the proof of Thm. 6.28. □

As an example of how to apply Cor. 6.29, consider a dataset of 3 inputs corresponding to heights of individuals, labelled 'Tall', 'Average' and 'Short', and a consumer who is only interested in distinguishing between the individuals who are 'Tall' and 'Average'. We might model this consumer's loss using the following function:

ℓ	w_1	w_2
Tall	0	1
Average	1	0
Short	0	0

Notice that this loss function ignores the input 'Short', since all guesses give the same loss. We might now consider the following mechanisms for this consumer: the first is the randomised response mechanism R, and the second is another kernel mechanism which we label K. Both of these have the same $\ln 2{\cdot}\mathbf{d}_{\mathrm{D}}$-privacy guarantee.

R	y_0	y_1	y_2
Tall	1/2	1/4	1/4
Average	1/4	1/2	1/4
Short	1/4	1/4	1/2

K	y_0	y_1
Tall	2/3	1/3
Average	1/3	2/3
Short	2/3	1/3

Observe that ℓ is an *extended* loss function on the inputs {Tall, Average} and Lem. 6.20 tells us that any mechanism equivalent to the trivial mechanism on those inputs will be optimal

for this consumer. K is one such mechanism, however R is not – so K should be optimal for any consumer modelled using ℓ (ie. for any prior). We can check: choosing π as the uniform distribution υ we compute

$$U_\ell[\upsilon \triangleright R] \;=\; 3/4 \;>\; 2/3 \;=\; U_\ell[\upsilon \triangleright K]$$

and so indeed K is better than R on the uniform prior, and this will be true for all priors.

6.8.2 The randomised response mechanism

The randomised response mechanism is the most well-known mechanism in this space. It derives its name from Warner's randomised response protocol, famously instituted for privacy-protection in surveys, and its popularity stems from its simplicity of implementation and its natural application to local differential privacy.

We defined the randomised response mechanism in Chapter 3 (recall Def. 3.4.3) where we noted that it is a $\mathbf{d}_{\mathrm{D}}$-private mechanism. It turns out to also be a kernel mechanism.

> **Lemma 6.30.** The randomised response mechanism is a $\mathbf{d}_{\mathrm{D}}$-private kernel mechanism.
>
> *Proof.* See Appendix §A.5.

The randomised response mechanism has the following interesting property which means that the optimality result of Thm. 6.28 does not apply to it.

> **Lemma 6.31.** Let $R : \mathcal{X} \to \mathbb{D}\mathcal{X}$ be the randomised response mechanism on $n > 2$ inputs. Let $x_1, x_2 \in \mathcal{X}$. Then $R{\downarrow}\{x_1, x_2\} \not\equiv T$ for any choice of x_1, x_2.
>
> *Proof.* Observe that the channel $R{\downarrow}\{x_1, x_2\}$ contains at least one column where every element has the form α/k (see Def. 3.4.3). Therefore it is not equivalent to T, the universally optimal mechanism on 2 inputs for any pair $\{x_1, x_2\}$. □

This means that the randomised response mechanism cannot be universally optimal for a large class of loss functions.

> **Corollary 6.32.** Let $R : \mathcal{X} \to \mathbb{D}\mathcal{X}$ be the randomised response mechanism on $n > 2$ inputs and let ℓ be a loss function with the property that $\ell{\downarrow}\{x_i, x_j\}$ is non-trivial for some $x_i, x_j \in \mathcal{X}$. Then R cannot be universally ℓ-optimal.
>
> *Proof.* Follows from Lem. 6.31 and Cor. 6.29. □

The above result is important as it rules out many classes of loss functions of interest – for example, strictly monotonic loss functions, the Bayes' risk loss function, and any extended loss function on 2 inputs.

6.8.3 Impossibility of universal optimality

Finally, we come to the main result of this section which is our reframing of the impossibility result of [40].

We begin by noting another property of loss functions: we call a loss function ℓ *pairwise non-trivial* if for every pair of elements $x_1, x_2 \in \mathcal{X}$ it holds that $\ell{\downarrow}\{x_1, x_2\}$ is non-trivial.

Examples of pairwise non-trivial loss functions include Bayes' risk, mean average error and mean-squared error. However, extended loss functions are not pairwise non-trivial, and neither was the loss function given in Example 6.5. It turns out that this property of loss functions describes sufficient conditions for impossibility of universal optimality.

THEOREM 6.33. There are no universally ℓ-optimal $\mathbf{d}_{\mathrm{D}}$-private mechanisms over $n > 2$ inputs for any pairwise non-trivial loss function ℓ.

Proof. By Cor. 6.29, it is sufficient to show that for every kernel mechanism K there exists some pair of inputs x_1, x_2 such that $K{\downarrow}\{x_1, x_2\} \not\equiv T$. Let K be any $\mathbf{d}_{\mathrm{D}}$-private kernel mechanism on $n > 2$ inputs with corresponding kernel hyper Δ_K. Then $\lceil \Delta_K \rceil$ has at least 2 posteriors, call them δ^1 and δ^2. Moreover, each δ^i is a vertex and so the $\mathbf{d}_{\mathrm{D}}$-privacy constraints hold tightly on $n-1$ input pairs. Pick any 3 inputs x_1, x_2, x_3 such that $\delta^1_{x_1} = \alpha\delta^1_{x_2}$ and $\delta^2_{x_2} = \alpha\delta^2_{x_1}$ (ie. the constraints hold in opposite directions). This means that $K{\downarrow}\{x_1, x_2\} \equiv T$. We now show that this cannot be true for both $\{x_1, x_3\}$ and $\{x_2, x_3\}$. The 4 possible constructions for δ^1 and δ^2 restricted to $\{x_1, x_2, x_3\}$ are:

Δ_1	δ^1	δ^2
x_1	α/k	$1/j$
x_2	$1/k$	α/j
x_3	$1/k$	α/j

Δ_2	δ^1	δ^2
x_1	α/k	$1/j$
x_2	$1/k$	α/j
x_3	$1/k$	$1/j$

Δ_3	δ^1	δ^2
x_1	α/k	$1/j$
x_2	$1/k$	α/j
x_3	α/k	α/j

Δ_4	δ^1	δ^2
x_1	α/k	$1/j$
x_2	$1/k$	α/j
x_3	α/k	$1/j$

It is clear that for each possible construction of K there exists a pair x_i, x_j st. $K{\downarrow}\{x_i, x_j\} \not\equiv T$. Therefore there is no K which can be universally ℓ-optimal for any pairwise non-trivial loss function ℓ and thus no universally ℓ-optimal $\mathbf{d}_{\mathrm{D}}$-mechanisms exist over the space of $n > 2$ inputs. □

Since we noted that Bayes' risk is pairwise non-trivial, the following is immediate.

COROLLARY 6.34. There are no universally **br**-optimal $\mathbf{d}_{\mathrm{D}}$-private mechanisms on $n > 2$ inputs for the Bayes' risk loss function **br**.

Cor. 6.34 is a strengthening of the result of [40], which proved impossibility for values of ε above some threshold. Our result holds for all values of ε.

6.8.4 Reframing the impossibility result for monotonic loss functions

We have not made reference to monotonicity of the Bayes' risk loss function in Thm. 6.33. In order to frame our results in the setting of [40], which specifies monotonicity of loss functions on $\mathbf{d}_{\mathrm{D}}$, we first define a subclass of monotonic loss functions which we call *strictly* monotonic.

> **Definition 6.8.1.** A loss function $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ is called *strictly monotonic in* $\mathbf{d}$ if ℓ is monotonic in $\mathbf{d}$, α is bijective and f is strictly monotonic (ie. strictly increasing).

Notice that the average distance function is strictly monotonic in $\mathbf{d}_2$ although it is non-monotonic in $\mathbf{d}_{\mathrm{D}}$. Note also that Bayes' risk is not strictly monotonic in $\mathbf{d}_2$, however it *is* strictly monotonic in $\mathbf{d}_{\mathrm{D}}$. Strict monotonicity was used by Ghosh et al. in their proof of optimality, however they showed their result extends to the more general class of monotonic functions.

We now have the following property of strictly monotonic loss functions.

> **Lemma 6.35.** Given a loss function $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, if there exists a metric $\mathbf{d}$ such that ℓ is strictly monotonic in $\mathbf{d}$, then $\ell{\downarrow}\{x, x'\}$ must be non-trivial for all $x, x' \in \mathcal{X}$ with $x \neq x'$.
>
> *Proof.* Assume that there exists $x, x' \in \mathcal{X}$ with $x \neq x'$ st. $\ell{\downarrow}\{x, x'\}$ is trivial. Then by Def. 6.5.1 there exists a w^* st. $\ell(w^*, x) \leq \ell(w, x)$ and $\ell(w^*, x') \leq \ell(w, x')$ for all $w \neq w^*$. Since ℓ is strictly monotonic, by Def. 6.8.1 we must have $\mathbf{d}(\alpha(w^*), x) \leq \mathbf{d}(\alpha(w), x)$ and $\mathbf{d}(\alpha(w^*), x') \leq \mathbf{d}(\alpha(w), x')$ for all $w \neq w^*$. But we also know from Def. 6.8.1 that there exists $w_a, w_b \in \mathcal{W}$ with $w_a \neq w_b$ st. $\alpha(w_a) = x$ and $\alpha(w_b) = x'$. Therefore we have
>
> $$\begin{aligned} \mathbf{d}(\alpha(w^*), x) &\leq \mathbf{d}(\alpha(w_a), x) = 0 \\ \mathbf{d}(\alpha(w^*), x') &\leq \mathbf{d}(\alpha(w_b), x') = 0 \end{aligned}$$
>
> Since α is bijective, this implies that $w_a = w_b$ and hence $x = x'$, contradicting our assumption that $x \neq x'$. Therefore no such x, x' exists. □

And now the following is immediate.

> **Corollary 6.36.** Let ℓ be a strictly monotonic loss function on $\mathbf{d}$ for some metric $\mathbf{d}$. Then there are no universally ℓ-optimal $\mathbf{d}_{\mathrm{D}}$-private mechanisms over $n > 2$ inputs.

This final result explains how monotonicity relates to the impossibility result, although it is not clear how this is meaningful, since curiously the impossibility holds regardless of the metric on the loss function. The result given in Thm. 6.33 suggests that the reason for non-optimality for Bayes' risk rests in its behaviour over pairs of inputs and that mechanisms designed for the discrete metric are not particularly good at differentiating over all pairs – by the nature of the

discrete metric.

However, consumers whose goal it is to distinguish between a particular pair of inputs can expect better utility in this space, from Cor. 6.29, which again describes the nature of the discrete metric – it can distinguish between a pair of inputs but not all pairs at the same time. This highlights the importance of designing mechanisms with the utility of consumers in mind.

6.9 Optimality for Other Metric Spaces

We now show how the results from the previous sections can be applied to other metric spaces to discover optimal mechanisms and their corresponding loss functions. We choose as a simple example the metric space of bitstrings of length 3 under the Hamming distance[23], although our results can be applied more generally to bitstrings of length n. This metric space (on n-length bitstrings) is used to describe a variety of privacy scenarios including user personalisation settings [85] and image search [86]. The randomised response mechanism is most commonly deployed in these scenarios since it is the only well-known mechanism for binary data; the geometric mechanism is typically reserved for discrete numeric data and the laplace for real-valued data.

Figure 6.4 depicts this space, also known as the Hamming cube. We ask: are there any optimal mechanisms in this space? And for which loss functions are these mechanisms optimal?

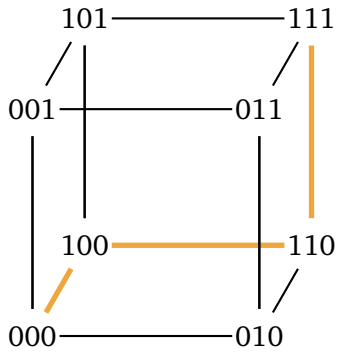


Figure 6.4: The Hamming space of bitstrings of length 3. The distance between vertices is given by the Hamming metric $\mathbf{d}_{\mathrm{H}}$ between bitstrings. The orange lines depict a path on which the $\mathbf{d}_{\mathrm{H}}$ metric is linear. Notice there are many other paths for which this holds, including all paths of length 3 from (000) to (111).

From Thm. 6.16 the optimal mechanisms are the kernel mechanisms of the space of $\mathbf{d}_{\mathrm{H}}$-private hypers. These are the mechanisms whose vertices must have the $\mathbf{d}$-privacy constraints holding with equality on every adjacent bitstring.

Notice (Figure 6.4) that there exist paths in the Hamming cube on which $\mathbf{d}_{\mathrm{H}}$ induces a linear order, for example the path **(000) – (100) – (110) – (111)**. Letting $X = \{000, 100, 110, 111\}$, from Lem. 6.25 we can find loss functions ℓ on $K\downarrow X$ (for kernel mechanisms K) for which K is universally $\ell\uparrow X$-optimal. Further, using Lem. 6.21, if we can find a kernel mechanism K st.

[23] Or, alternatively the Manhattan metric, which coincides with the Hamming distance in this space.

$K\downarrow X \equiv G$ where G is the geometric mechanism, then we have universal ℓ-optimality for all monotonic loss functions on $K\downarrow X$ which we can extend to loss functions on $\mathcal{X}$.

Inspired by the latter, using $\varepsilon = \ln 2$ we find a kernel hyper with the required properties and construct its corresponding kernel mechanism (see Figure 6.5).

$$\Delta_K = \begin{bmatrix} 8/27 & 2/21 & 1/21 & 1/27 \\ 4/27 & 4/21 & 2/21 & 2/27 \\ 2/27 & 2/21 & 4/21 & 4/27 \\ 1/27 & 1/21 & 2/21 & 8/27 \\ 4/27 & 4/21 & 2/21 & 2/27 \\ 2/27 & 2/21 & 4/21 & 4/27 \\ 2/27 & 2/21 & 4/21 & 4/27 \\ 4/27 & 4/21 & 2/21 & 2/27 \\ 9/32 & 7/32 & 7/32 & 9/32 \end{bmatrix}$$

K	y_1	y_2	y_3	y_4
000	2/3	1/6	1/12	1/12
100	1/3	1/3	1/6	1/6
110	1/6	1/6	1/3	1/3
111	1/12	1/12	1/6	2/3
010	1/3	1/3	1/6	1/6
011	1/6	1/6	1/3	1/3
101	1/6	1/6	1/3	1/3
001	1/3	1/3	1/6	1/6

Figure 6.5: A kernel hyper (Δ_K) and corresponding kernel mechanism (K) on bitstrings of length 3 wrt the Hamming distance $\mathbf{d}_{\mathrm{H}}$ with $\varepsilon = \ln 2$.

Notice that $K\downarrow X$ is exactly the geometric mechanism on 4 inputs with $\varepsilon = \ln 2$. The same holds for $K\downarrow\{000, 010, 011, 111\}$ and $K\downarrow\{000, 001, 101, 111\}$, corresponding to 2 alternate paths from 000 to 111. Therefore K is universally ℓ-optimal for any (extended) monotonic loss functions defined on these subsets of $\mathcal{X}$. Using our loss function algebra these can be combined to form more complex loss functions for which K is universally optimal.

Interestingly, if the randomised response mechanism had been used (eg. by applying random response to each bit of the bitstring), the resulting mechanism would not have the above structure (ie. we cannot find a "geometric mechanism" by restricting the resulting mechanism to any subset of inputs). Nor could we find such a structure by a post-processing step, since this is a refinement and therefore can only reduce utility.

However, notice also that if we consider the subset $X = \{100, 010, 001\}$ we find that $K\downarrow X \equiv \mathbf{1}$. ie., the mechanism which leaks nothing. So there are no non-trivial loss functions for which $K\downarrow X$ is optimal.

Finally, on any adjacent inputs x, x' we have $K\downarrow\{x, x'\} \equiv T$ and so K is also universally ℓ-optimal for any extended loss function on adjacent inputs, and for their combination using our loss function algebra (Eqn (6.6)).

Therefore, we conclude that there are optimal mechanisms in this space (and K is one such example) which are optimal for extended monotonic loss functions on paths of length 4 (for which $\mathbf{d}_{\mathrm{H}}$ induces a linear order), as well as extended loss functions on adjacent inputs.

6.10 Conclusion

We have presented an algebraic framework for reasoning about optimality based on the robust model of QIF developed by Alvim et al. [34]. We have used our framework to characterise the class of universally optimal mechanisms, for all consumers and for classes of consumers defined by restricted classes of loss functions, using various notions of refinement.

We have also extended and generalised the foundational results of Ghosh et al. [39] and Brenner et al. [40] to other classes of loss functions and to mechanisms defined for $\mathbf{d}$-privacy. Importantly, we have established that there are other universally optimal mechanisms in both the domain of counting queries (extended to the Euclidean metric) – aside from the well-known geometric mechanism – and sum queries (extended to the Discrete metric) – overturning the impossibility result in this space for monotonic functions, and thus opening up the possibility of new optimality results in this space.

Finally, our framework naturally extends to the domain of continuous mechanisms, allowing us to explore optimality in non-discrete settings, which will be the subject of Chapter 7.

We leave as future work the exploration of other loss function of classes of interest and the exploration of other optimal mechanisms in the above spaces of interest as well as new metric spaces relevant to other domains.

Our observation that the space of mechanisms is very rich also brings up the problem of *efficiently* finding optimal mechanisms. We have not addressed this problem here and we also leave this open for future work.

6.11 Chapter Notes

This chapter is unpublished material.

The framework developed in this chapter is the first, to our knowledge, to build on the works of [39, 40] using the Bayesian model of consumers. Gupte et al. [87] proved a universal optimality result for a type of risk-averse consumer using a *minimax* formulation. We note that we did not address their optimality results because their formulation does not correspond to an information-theoretic measure in our setting; an alternative would be to consider our *max-case* leakage notion, explored in Chapter 4. This has a simpler geometric representation than standard (average-case) refinement and would thus lend itself to reasoning using this framework. We leave this open as future work.

Other optimality results have been found in non-Bayesian settings. Kairouz et al. [88] proved optimality for "staircase mechanisms" using f-divergence as a utility measure, while Koufogiannis et al. [89] showed that the Laplace mechanism is optimal for the mean-squared error.

We note that optimality for metric differential privacy was explored by Chatzikokolakis et al. [15]; in particular they showed that optimality for "sum" queries (in the traditional database setting) could be achieved by changing the metric on databases (ie. so that the induced metric for sum queries is the Euclidean distance) so that Ghosh et al.'s optimality result can then be invoked.

Finally, we do not know of many works exploring refinement restricted to particular loss or gain functions, apart from the exploration of the Bayes' risk refinement order by McIver et al. [90].

7

Optimality II: Continuous Mechanisms

In Chapter 6 we explored optimality for discrete mechanisms defined for arbitrary metrics in the context of oblivious workflows, with a focus on the Euclidean and Discrete metrics which are of particular interest in this context. In this chapter we continue our exploration of optimality, this time for a particular class of oblivious mechanisms designed for the Euclidean distance metric over a single *continuous* dimension (eg. the real numbers). While the geometric mechanism is well-known to be optimal for many classes of queries in the corresponding discrete space, it turns out that this mechanism is not optimal over continuous inputs. Our contribution is to show that optimality is regained by using the *Laplace* mechanism for the obfuscation.

As with Chapter 6, the technical apparatus involved employs the barycentric representation of hyper-distributions – extended here to continuous domains – as well as the earlier result of Ghosh et al. [39] regarding universal optimality of the geometric mechanism. In addition, we make use of the dual interpretations of distance between distributions provided by the Kantorovich-Rubinstein Theorem, both to complete our proof and to formulate a bound on the utility loss when employing the Laplace mechanism over a discrete domain for which the geometric mechanism is better suited.

7.1 Introduction

Although popular in the literature and in privacy applications, the Laplace mechanism – to be introduced – has yet to be ascribed any optimality properties; in fact, results on optimality for the Laplace mechanism have uncovered only negative results [91, 92], showing that variations on the Geometric mechanism are preferable to the Laplace mechanism in various scenarios. However, none of these studies is conducted over continuous input spaces. Our contribution in

this chapter is to extend the study of optimality begun in Chapter 6 to continuous mechanisms, focussing on the optimality results of Ghosh et al. [39] previously described. Here we find that the Laplace mechanism is optimal over continuous inputs in the same way that the geometric mechanism is optimal for discrete inputs. We make use of the strategies developed in the previous chapter – that is, reasoning about **d**-private mechanisms in the space of hypers, utilising the notion of a refining Earth Move in this space and employing kernel mechanisms to demonstrate some of the interesting features of the geometric mechanism which make it optimal under precisely the right conditions. We show not only that the Laplace mechanism is optimal in the continuous domain, but that we can compute a bound on the loss of optimality when the Laplace mechanism is used for a discrete space instead of a properly designed Geometric mechanism.

We begin with some technical preliminaries – reminders from Chapter 6 – about discrete mechanisms and how we model optimality using QIF ideas of refinement, loss functions and – most importantly – hyper-distributions.

7.2 Preliminaries: loss functions; hypers; refinement

We recall the usual QIF model that describes (discrete) probabilistic channels as functions $C: \mathcal{X} \to \mathbb{D}\mathcal{Y}$ and isomorphically as channel matrices. Our obfuscating mechanisms are channels in the discrete case — the result of the query is the channel's input x, and the (perturbed) value the observer sees is the channel's output y. In this chapter we will write the type of C as $\mathcal{X} \mathrel{-\!\!\triangleright} \mathcal{Y}$, when the output could be either discrete distributions $\mathbb{D}\mathcal{Y}$ or continuous measures $\mathbb{M}\mathcal{Y}$.

The loss functions $\ell(w,x)$ will quantify the loss to the observer of seeing (only) y, and then choosing w, when what she really wants to know is x. We recall that she does this via a *remapping*, which determines her best strategy for choosing w. In general, we write $U_\ell[\pi \triangleright M]$ for the *expected* loss to a rational observer, given the prior π, channel M and the loss function ℓ she has chosen: it is

$$\sum_y p(y) \;\; \min_w \sum_x \ell(w,x)\, p(x|y) \quad , \tag{7.1}$$

that is: the expected value, over all possible observations y and their marginal probabilities, of the *least* loss she could rationally achieve over *all* her possible choices w given the knowledge that y will have provided about the posterior distribution $p(X|y)$ of the actual raw input x. Note that M and π determine the $p(y)$ and $p(x|y)$ that appear in (7.1). We remark that this formulation for measuring expected loss corresponds precisely to the formulation used by Ghosh et al. in the optimality theorem (recall Chapter 6, Lem. 6.1).

7.2.1 The relevance of hyper-distributions

We recall that a *hyper-distribution* is a distribution of distributions on $\mathcal{X}$ that is of type $\mathbb{D}\mathbb{D}\mathcal{X}$; abbreviated as "hyper" and "$\mathbb{D}^2\mathcal{X}$" respectively. Given a joint distribution $J: \mathbb{D}(\mathcal{X}\times\mathcal{Y})$, we write $[J]$ for the hyper-distribution whose support is posterior distributions [1] $p(X|y)$ on X and which

[1] Recall that in the hyper-distribution literature these are called "inners".

assigns the corresponding marginal distribution $p(y)$ to each. (Zero-valued marginals are left out.) We now re-express (7.1) in those terms.

If we write $\ell(w,-)$ for the function on $\mathcal{X}$ that ℓ determines once w is fixed, and write $\mathcal{E}_{\text{DIST}}\,\text{RV}$ for expected value of random-variable RV with distribution DIST, then $\min_w \mathcal{E}_{p(X|y)}\,\ell(w,-)$ is the inner part of (7.1). Then fix some ℓ and define for general distribution $\delta\colon \mathbb{D}X$ that

$$Y_\ell(\delta) \quad := \quad \min_w \mathcal{E}_\delta\, \ell(w,-) \quad , \tag{7.2}$$

(using Y for "entrop$Y$") so that Y_ℓ is itself a real-valued function on distributions δ (as eg. Shannon entropy is). With that preparation, the expression (7.1) becomes the expected value of Y_ℓ over the hyper produced by abstracting from $J = \pi \triangleright M$ as above. That is (7.1) gives equivalently

$$U_\ell[\pi \triangleright M] \quad = \quad \mathcal{E}_{[\pi \triangleright M]}\, Y_\ell \quad , \tag{7.3}$$

in which the M and π now explicitly appear and where –we recall– the brackets $[-]$ convert the joint distribution $\pi \triangleright M$ to a hyper. (If Y_ℓ were in fact Shannon entropy, then (7.3) would be the *conditional* Shannon entropy. But Y_ℓ's are much more general than Shannon entropy alone [56, 60].)

We remark that the notation $U_\ell[\pi \triangleright M]$ makes explicit the hyper produced by the action of π on M; and this is well-defined in the continuous case (ie. when π is defined over a continuous set $\mathcal{X}$, which is also the input domain for M). Further details are supplied in Appendix §B.1.

7.2.2 Refinement of hypers and mechanisms

We recall from Lem. 2.5 that the QIF notion of refinement on channels carries through to hypers; moreover, on hypers, refinement is a true *partial* order which admits several equivalent interpretations. Below, we write Δ etc. for general hypers in $\mathbb{D}^2\mathcal{X}$.

We have that $\Delta \sqsubseteq \Delta'$, that hyper Δ is refined by hyper Δ', under any of these equivalent conditions:

(a) when $\mathcal{E}_\Delta\, Y_\ell \le \mathcal{E}_{\Delta'}\, Y_\ell$ for *all* loss functions ℓ (ie. whether legal or not).

(b) when considered as distributions on posteriors $\mathbb{D}\mathcal{X}$ it is possible to convert Δ into Δ' via a Wasserstein-style "earth move" of probability from one posterior to another (recall Chapter 2, Def. 2.4.3) .

(c) when generated from joint-distribution matrices D in $\mathbb{D}(\mathcal{X}\times\mathcal{Y})$ generating Δ, and D' in $\mathbb{D}(\mathcal{X}\times\mathcal{Y}')$ generating Δ', there is a "post-processing matrix" R of type $\mathcal{Y} \to \triangleright \mathcal{Y}'$ such that as matrices we have $D \cdot R = D'$ via matrix multiplication.

And we say that one mechanism M is refined by another M' just when $[\pi \triangleright M] \sqsubseteq [\pi \triangleright M']$ for all priors π. When this occurs we also write $M \sqsubseteq M'$. From Formulation (a) we will use the fact that the $(\sqsubseteq)$-infimum of the ${}^T L^\varepsilon_N$'s (indexed over a sequence of T's) is just L^ε itself [62].

Formulation (b) is particularly useful. If we find a specific earth move from Δ to Δ' that defines a refinement we can then use the equivalent (a) to deduce that $\mathcal{E}_\Delta\, Y_\ell \le \mathcal{E}_{\Delta'}\, Y_\ell$. However

if we can also compute the cost [2] of the particular earth move we can conclude in addition that the difference $|\mathcal{E}_\Delta Y_\ell - \mathcal{E}_{\Delta'} Y_\ell|$ must be bounded above by an amount we can compute. This follows from the well-known Kantorovich-Rubinstein duality [94] which says that $|\mathcal{E}_\Delta Y_\ell - \mathcal{E}_{\Delta'} Y_\ell|$ is no more than minimal cost incurred by any earth move transforming Δ to Δ' scaled by the "Lipschitz constant" [3] of Y_ℓ. We use these ideas in Lem. 7.7 and Thm. 7.9.

7.3 Discrete and continuous optimality

In this section we recall the optimality result for the geometric mechanism and show why it does not apply in the continuous space. We then outline our approach to optimality for continuous domains.

We will find it useful to think of the geometric mechanism as a (probabilistic) function from inputs to outputs derived from the Geometric distribution. The Geometric distribution centred on 0 with parameter α assigns (discrete) probability

$$G_\alpha(n) \quad := \quad {}^{1-\alpha}/_{1+\alpha} \cdot \alpha^{|n|} \tag{7.4}$$

to any integer n (positive or negative) [39]. It implements an $\varepsilon\cdot\mathbf{d}$-private Geometric mechanism by obfuscating the query according to (7.4) above: thus setting $\alpha := e^{-\varepsilon}$ we define

$$G^\varepsilon(n)(n') \quad := \quad G_\alpha(n'-n) = {}^{1-\alpha}/_{1+\alpha} \cdot \alpha^{|n'-n|} \tag{7.5}$$

to be the probability that integer n is input and n' is output. (Note that this now corresponds with the channel-based definition given in Def. 3.4.1).

7.3.1 The optimality result for the geometric mechanism

Our appeal to Ghosh et al. [39] uses their Theorem 3.3 and Corollary 3.3, transliterated here for our notation.

> **Theorem 7.1** (Theorem 3.2 in [39]). Fix arbitrary values for $N \geq 1$ and $e^{-\varepsilon} \in [0,1]$, and a count query. A mechanism M is universally utility maximising if and only if there is a remap Y of X such that $Y \circ M$ is the truncated $\varepsilon\cdot\mathbf{d}$-private-geometric mechanism.

> **Theorem 7.2** (Corollary 3.3 in [39]). For every $N \geq 1$ and $e^{-\varepsilon} \in [0,1]$, and every count query, the corresponding $\varepsilon\cdot\mathbf{d}$-private-geometric and truncated $\varepsilon\cdot\mathbf{d}$-private-geometric mechanisms are universally loss minimising (utility maximising).

The "loss" in these results is computed in terms of loss functions $\ell : N \times N \to \mathbb{R}$, and assumed

[2] The cost is determined by the amount of "earth" to be moved, and the distance it must be moved. See for example [93].

[3] The Lipschitz constant of a function is the amount by which the difference in outputs can vary when compared to the difference in inputs.

to be monotone in its first argument. We have seen already (Chapter 6, Lem. 6.1) that the QIF understanding of expected loss takes care of the remapping step in Thm. 7.1 above.

The fact that these loss functions also assume that there are only N possible choices in the first argument w means that the result holds no matter the number of outputs in the mechanism M. This can be seen easily in the hyper representation because the posteriors that correspond to the same choice w naturally collapse (under addition and scaling). The result of this is that the same loss for M is incurred by that for a mechanism M', but with no more than N outputs. Ghosh et al. deal with these details in a similar way.

7.3.2 The geometric mechanism is never $\varepsilon\cdot\mathbf{d}$-private on dense continuous inputs, eg. when d on $\mathcal{X}$ is Euclidean

We wonder now if the geometric mechanism, defined on *discrete* inputs, can be naturally lifted to a $\varepsilon\cdot\mathbf{d}$-private mechanism on *continuous* inputs. We show here that the obvious lifting – namely, applying (7.5) to continuous inputs – results in a mechanism which is not $\varepsilon\cdot\mathbf{d}$-private for *any* ε.

Notice that when G's input $\mathcal{X}$ is continuous, G's output remains discrete, taking some number of steps, each of fixed length say $\lambda{>}0$, in either direction. That is, any G input x is perturbed to $x{+}i\lambda$ for some integer i.

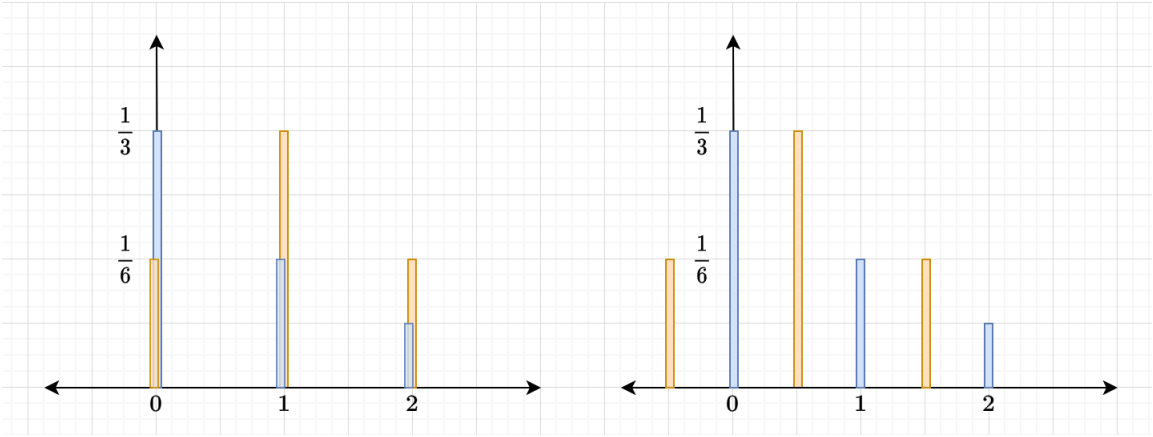
Now because $\mathcal{X}$ is continuous and dense, we can vary the input x itself, by a tiny amount, to some x' so that $\mathbf{d}(x,x') < \lambda$ no matter how small λ might be, producing perturbations $x'{+}i\lambda$ each of which is distant that same (constant) $\mathbf{d}(x,x')$ from the original $x{+}i\lambda$ and, precisely because $\mathbf{d}(x,x') < \lambda$, those new perturbations cannot overlap the ones based on the original x.

Thus the two distributions produced by G acting on x and on x' have supports that do not intersect at all. And therefore the d_D distance between the two distributions is infinite, meaning that G cannot be be $\varepsilon\cdot\mathbf{d}$-private for any (finite) ε. That is, for a database producing truly real query results $\mathcal{X}$, a standard (discrete) G cannot establish $\varepsilon\cdot\mathbf{d}$-privacy for any ε, however large ε might be. This is illustrated in Figure 7.1.

There are two possible solutions. The first solution, both obvious and practical, is to "discretise" the input and to scale appropriately: a person's height of 1.75m would become 175cm instead. A second solution however is motivated by taking a more theoretical approach. Rather than discretise the type of the query results, we *leave* it continuous — and seek our optimal mechanism among those that – unlike the geometric – do not take only discrete steps. It will turn out to be the Laplace mechanism.

7.3.3 Our result — continuous optimality

In the discrete case typically the set $\mathcal{X}$ of inputs is $\{0\ldots N\}$ for some $N{\geq}0$, and the prior knowledge π is a (discrete) distribution on that. For our continuous setting we will use $\mathcal{X}{=}[0,1]$ for inputs, the unit interval $\mathcal{U}$, and the discrete distribution π will become a proper measure on $[0,1]$ expressed as a probability density function. The $\varepsilon\cdot\mathbf{d}$-private mechanisms, now K^ε for "kontinuous", will take an input x from a continuous set $\mathcal{X}$ rather than a discrete one. And the metric $\mathbf{d}$ will be Euclidean.



The x-axis represents outputs distributed according to a coloured distribution; the y-axis is the probability of observing an output.

The left hand figure shows (part of) an infinite geometric mechanism with step-size $\lambda = 1$ defined for secret space $\mathcal{X} = \mathbb{Z}$, with (partial) output distributions for $x = 0$ (blue) and $x = 1$ (orange). These distributions overlap on the output space (ie. the outputs are integers $0, 1, 2$ etc) and so we can compute a finite ε for this mechanism (it is $\ln 2$).

The right hand figure shows the geometric mechanism for the same step-size $\lambda = 1$ applied to the secrets $x = 0$ (blue) and $x = 1/2$ (orange). Since the step-size determines how these distributions are juxtaposed, we find that the output spaces do not intersect and the corresponding mechanism has infinite ε. To fix this, we would define a geometric mechanism using step-size $1/2$, but then this would fail for inputs $x = 0, 1/4$, and so on. Since $\lambda > 0$, we find that the geometric will fail for $x = 0, \lambda/2$ and thus cannot be made $\varepsilon\cdot\mathbf{d}$-private for any finite ε on [0,1].

Figure 7.1: Illustration of why lifting the geometric mechanism by changing its input domain does not result in a valid $\varepsilon\cdot\mathbf{d}$-private mechanism.

Our (continuous) optimality result formalised at Thm. 7.3 is that the $\varepsilon\cdot\mathbf{d}$-private Laplace mechanism L^ε minimises loss over all continuous priors π on $\mathcal{X}{=}\mathcal{U}$ and all legal loss functions ℓ out of the class of $\varepsilon\cdot\mathbf{d}$-private mechanisms with respect to the Euclidean metric on the continuous input $\mathcal{X}{=}[0,1]$. Since the theorem requires that *all* mechanisms are $\varepsilon\cdot\mathbf{d}$-private, the argument in §7.3.2 above shows therefore that geometric mechanisms are no longer suitable (for optimality) because on continuous $\mathcal{X}$ they are no longer $\varepsilon\cdot\mathbf{d}$-private.[4]

7.3.4 An outline of the proof

Our proof technique relies on the work developed in the previous chapter on the space of hyper-distributions and the geometric interpretation of refinement of hypers. In summary, we notice that

(a) the Kantorovich-Rubinstein duality lifted to *hyper-distributions* describes a minimum Earth Move which upper bounds the expected loss over all 1-Lipschitz functions on hypers (think: posterior uncertainties),

[4] We note that our lifting of the geometric mechanism seems to be the most obvious and natural; and we are not aware of any standard lifting of the geometric to continuous domains, or of any that would be $\varepsilon\cdot\mathbf{d}$-private.

(b) the geometric and Laplace mechanisms are related by refinement which describes an Earth Move between hypers (not necessarily minimal), and

(c) the refining Earth Move distance between the geometric and Laplace hypers approaches 0 as the discretised space "approaches continuity" (in the limit), which means the *minimal* Earth Move must do the same,

(d) and therefore the difference in expected loss (measured over all consumers) also approaches 0, proving that the utility of the Laplace mechanism is the same as that of the geometric mechanism in the limit.

In more detail: We access the existing discrete result from within the continuous $\mathcal{U}$ by "pixelating" it, that is defining $\mathcal{U}_N = \{0, 1/N, 2/N, \ldots, N-1/N, 1\}$ for integer $N>0$, and mapping $\{0 \ldots N\}$ isomorphically onto that discrete subset. We then establish near optimality for a similarly pixelated Laplace mechanism, showing that "near" becomes "equal to" when N tends to infinity. Specifically,

(a) (We show in §7.5.2 that) Any (discrete) prior on $\mathcal{U}_N$ corresponds to some prior on the original $\mathcal{U}$, but can also be obtained by pixelating some *continuous* prior π on all of $\mathcal{U}$, concentrating its (now discrete) probabilities onto elements of $\mathcal{U}_N$ only: eg. the probability $\pi[n/N, n+1/N)$ of the entire $1/N$-sized interval is moved onto the point n/N. We write it π_N.

(b) (§7.5.3) Any function f acting on all of $\mathcal{U}$ can be made into an N-step function by first restricting its inputs to $\mathcal{U}_N$ and then filling in the "missing" values $f(x)$ for x in $(n/N, n+1/N)$ by copying the value for $f(n/N)$. If f is an $\varepsilon{\cdot}\mathbf{d}$-private mechanism K^ε, we write its N-stepped version as K^ε_N, and note that K^ε_N remains $\varepsilon{\cdot}\mathbf{d}$-private when restricted to the points in $\mathcal{U}_N$ only. If f is a loss function ℓ on ($\mathcal{W}$ and) $\mathcal{X}$ we write ℓ_N for its stepped version.

(c) (§7.6.2; Lem. 7.6) Now for any N, mechanism K^ε_N, prior π_N, and legal N-step loss function ℓ_N we can appeal to the discrete optimality result: for the pixelated prior π_N and the N-step and legal ℓ_N the loss due to G^ε_N is $\leq$ the loss due to K^ε_N.

(d) (§7.6.4; Thm. 7.9) The replacement of G^ε_N by L^ε_N (both N-step functions on $[0,1]$) is via pixelating the *output* (continuous) distribution of L^ε_N to a multiple T of N: we write that ${}^T L^\varepsilon_N$. The Kantorovich-Rubinstein Theorem, provided additionally that ℓ_N is p-Lipschitz for some $p>0$ independent of N, shows that the (additive) difference between the G^ε_N-loss and the ${}^T L^\varepsilon_N$-loss, for any π_N and ℓ_N and T a multiple of N, tends to zero as N increases.

(e) (§7.5.4) We can remove the subscript π_N on the prior, and on the mechanisms K^ε_N and ${}^T L^\varepsilon_N$, relying now on the $\varepsilon{\cdot}\mathbf{d}$-privacy of the two mechanisms to make the (multiplicative) ratio between the losses they cause tend to 1.

(f) (§7.7) The final step, removing the subscript $_N$ from ℓ_N, is that the loss-calculating procedure is continuous and that ℓ_N tends to ℓ as N tends to infinity.

7.4 Measures on continuous $\mathcal{X}$ and $\mathcal{Y}$

7.4.1 Measures via probability density functions

Continuous analogues of priors π, mechanisms M and loss functions ℓ will be our principal concern here: ultimately we we will use $\mathbb{M}[0,1]$ for our measurable spaces $\mathcal{X}$ and $\mathcal{Y}$, and will suppose for simplicity that $\mathcal{X}$=$\mathcal{Y}$=$[0,1]$. (More generality is achieved by simple scaling).

Measures $\mathbb{M}[0,1]$ (that is $\mathbb{M}\mathcal{X}$ and $\mathbb{M}\mathcal{Y}$) will be given as probability density functions, where a PDF say $\mu\colon[0,1]\to\mathbb{R}_{\geq 0}$ determines the probability $\int_a^b \mu$ assigned to the sample $[a,b]\subseteq[0,1]$ using the standard Borel measure on $[0,1]$, and more generally the expected value of some random variable V on $[a,b)$ given by PDF μ is $\int_a^{b-} \mu(x)V(x)\mathrm{d}x$.

Even though μ is of type PDF, we abuse notation to write for example $\mu[a,b)$ for the probability $\int_a^{b-}\mu$ that μ assigns to that interval, and μ_a for the probability μ assigns to the point a alone, i.e. some r just when when the actual PDF-value of $\mu(a)$ is the Dirac delta-function scaled by r, written δ_r.

7.4.2 Continuous mechanisms over continuous priors

Our mechanisms M, up to now discrete, will now become "kontinuous", renamed K as a mnemonic. Thus a continuous mechanism $K\colon\mathcal{X}\to\mathbb{M}\mathcal{Y}$ given input x produces measure $K(x)$ on the observations $\mathcal{Y}$=$[0,1]$. And given a a whole continuous prior $\pi\colon\mathbb{M}[0,1]$, that same K therefore determines a joint measure over $\mathcal{X}\times\mathcal{Y}$.[5] By analogy with (7.2,7.3) we have

> **Definition 7.4.1 (Continuous version of (7.1)).** The expected loss $U_\ell[\pi\triangleright K]$ due to continuous prior π, continuous mechanism K and loss function ℓ is given by [6]
>
> $$\int_0^1 (\inf_w (\int_0^1 \ell(w,x)\pi(x)K(x)(y)\ \mathrm{d}x))\mathrm{d}y\,. \tag{7.6}$$

The continuous version of uncertainty (7.2) is now

$$Y_\ell(\delta) \quad := \quad \inf_{w\colon\mathcal{W}} \int_0^1 \ell(w,x)\delta(x)\mathrm{d}x$$

and the continuous version of expected loss (7.3) is now

$$U_\ell[\pi\triangleright K] \quad = \quad \int_{y\colon\mathcal{Y}} Y_\ell\ \mathrm{d}K(\pi) \quad .$$

7.4.3 The truncated Laplace mechanism

The Laplace mechanism is based on the Laplace distribution:[7] it works out to be

[5] See (B.1) in Appendix §B.1.

[6] This is well defined whenever the $\mathcal{W}$-indexed family of functions of y given by $\int_0^1 \ell(w,x)\pi(x)K(x)(y)\ \mathrm{d}x$ contains a countable subset $\mathcal{W}'$ such that the inf over $\mathcal{W}$ is equal to the inf over $\mathcal{W}'$ [95]. This is clear if $\mathcal{W}$ is finite, and whenever $\mathcal{W}'$ can be taken to be the rationals.

[7] In the same way that the geometric mechanism is based on the geometric distribution.

DEFINITION 7.4.2 (Laplace distribution). The *ε-Laplace mechanism* with with input x in $\mathcal{X}$=[0,1] and probability density in $\mathbb{R}_{\geq 0}$ for output y in $\mathcal{Y}$ is usually written as a PDF in y (for given x) as [96]

$$L^{\varepsilon}(x,y) \quad := \quad \varepsilon/2 \cdot e^{-\varepsilon|y-x|} \quad .$$

The $\varepsilon/2$ is a normalising factor. It is known [15] that the mechanism L^{ε} is $\varepsilon\cdot\mathbf{d}$-private over [0,1] (where the underlying metric on $\mathcal{X}$ is Euclidean). Just as for the Geometric mechanism we truncate L^{ε}'s outputs so that they also lie inside $\mathcal{U}$. We do so in the same manner, by remapping all outputs greater than 1 to 1, and all outputs less than 0 to 0.

DEFINITION 7.4.3 (Truncated Laplace mechanism). The truncated Laplace mechanism $Ł^{\varepsilon}$ for inputs restricted to [0,1], and output restricted to [0,1], is defined in the following way (as a PDF):

$$Ł^{\varepsilon}(x)(y) \quad := \quad \begin{array}{ll} \delta_a & \textit{if } y{=}0 \\ L^{\varepsilon}(x,y) & \textit{if } 0{<}y{<}1 \\ \delta_b & \textit{if } y{=}1 \end{array} \quad ,$$

where the constants a, b are $\int_{-\infty}^{0} L^{\varepsilon}(x,y)\mathrm{d}y = e^{\varepsilon x}/2$ and $\int_{1}^{\infty} L^{\varepsilon}(x,y)\mathrm{d}y = e^{\varepsilon(1-x)}/2$ respectively, and δ_r is the Dirac delta-function with weight r.

We can now state our **principal contribution**. It is to show that the *truncated* Laplace $Ł^{\varepsilon}$ is universally optimal, in this continuous setting, in the same way that G^{ε} was optimal in the discrete setting:

THEOREM 7.3 (Truncated Laplace is optimal). Let K^{ε} be any continuous $\varepsilon\cdot\mathbf{d}$-private mechanism with input and output both [0,1], and let π be any continuous (prior) probability distribution over [0,1] and ℓ any Lipschitz continuous [8], legal loss function on $\mathcal{X}$=$\mathcal{U}$. Then $U_{\ell}[\pi \triangleright Ł^{\varepsilon}] \leq U_{\ell}[\pi \triangleright K^{\varepsilon}]$.

As we foreshadowed in the proof outline in §7.3.4, Thm. 7.3 relies ultimately on the earlier-proven optimality G^{ε} in the discrete case: we must show how we can approximate continuous $\varepsilon\cdot\mathbf{d}$-private mechanisms in discrete form, each one satisfying the conditions under which the earlier result applies, and in §7.5 we fill in the details. Along the way we show how the Laplace mechanism provides a smooth approximation to the Geometric with discrete inputs.

[8] Lipschitz continuous is less general than continuous. It means that the difference in outputs is within a constant κ>0 scaling factor of the difference between the inputs.

7.5 Approximating Continuity for $\mathcal{X}$

7.5.1 Connecting continuous and discrete

Our principal tool for connecting the discrete and continuous settings is the evenly-spaced discrete subset $\mathcal{U}_N = \{0, 1/N, 2/N \ldots, N-1/N, 1\}$ of the unit interval $\mathcal{U}$=[0,1] for ever-increasing N>0. The separation $1/N$ is the *interval width*.

7.5.2 Approximations of continuous priors

The N-approximation of prior $\pi: \mathbb{M}\mathcal{U}$ of type $\mathbb{D}\mathcal{U}_N$, ie. yielding actual probabilities (not densities), is defined

$$\pi_N(n/N) \quad := \quad \begin{array}{ll} \pi[n/N, n+1/N) & \text{if } n < N-1 \\ \pi[n/N, 1] & \text{if } n = N-1 \\ 0 & \text{otherwise} \end{array} .$$

The discrete π_N gathers each of the continuous π-interval's measure onto its left point, with as a special case [1,1] from π included onto the point $N-1/N$ of π_N.

As an example take N to be 2, and π to be the uniform (continuous) distribution over $\mathcal{U}$, which can be represented by the constant 1 PDF. Since the interval width is $1/2$, we see that π_N assigns probability $1/2$ to both 0 and $1/2$, and probability zero to all other points in $\mathcal{U}$.

7.5.3 N-step mechanisms and loss functions

In the other direction, we can lift discrete M and loss-function ℓ on $\mathcal{U}_N$ into the continuous $\mathcal{X}$=$\mathcal{U}$ by replicating their values for the x's *not* in $\mathcal{U}_N$ in a way that constructs N-step functions: we have

Definition 7.5.1. For x in $\mathcal{U}$=[0,1] define $\lfloor x \rfloor_N := \lfloor Nx \rfloor / N$.

Definition 7.5.2. Given mechanism $M: \mathcal{U}_N \to\!\!\!\!\triangleright \mathcal{Y}$, define $M_N: [0,1] \to \mathbb{R}_{\geq 0}$ so that

$$M_N(x) \quad := \quad \begin{array}{ll} M(\lfloor x \rfloor_N) & \text{if } 0 \leq x < 1 \\ M(N-1/N) & \text{if } x = 1 \end{array} .$$

Note that we have not yet committed here to whether M produces discrete or continuous distributions on its *output* $\mathcal{Y}$. We are concentrating only on its input (from $\mathcal{X}$).

Similarly, given loss function $\ell: \mathcal{W} \times \mathcal{U}_N, \to \mathbb{R}_{\geq 0}$, define $\ell_N: \mathcal{W} \times [0,1] \to \mathbb{R}_{\geq 0}$ so that

$$\ell_N(w,x) \quad := \quad \begin{array}{ll} \ell(w, \lfloor x \rfloor_N) & \text{if } 0 \leq x < 1 \\ \ell(w, N-1/N) & \text{if } x = 1 \end{array} .$$

Say that mechanisms and loss functions over [0,1] are *N-step functions* just when they are constructed as above.

The important property enabled by the above definitions is the correspondence between loss functions' values in their pixelated and original versions, which will allow us to apply the earlier discrete-optimality result, based on Lem. 7.5 to come. That is, we have

Lemma 7.4. For any continuous prior π in $\mathbb{M}\mathcal{U}$, mechanism M in $\mathcal{U}\to\!\triangleright\mathcal{Y}$ and loss function ℓ in $\mathcal{W}\times\mathcal{U}\to\mathbb{R}_{\geq 0}$ we have

$$\overbrace{U_{\ell_N}[\pi\triangleright M_N]}^{\text{continuous } \mathcal{X}} \quad = \quad \overbrace{U_{\ell}[\pi_N\triangleright M]}^{\text{discrete } \mathcal{X}} \quad .$$

That is, the loss realised via a pixelated π_N, and (already discrete) M and ℓ, all operating on $\mathcal{U}_N$, is the same as the loss realised via the original continuous π and the lifted (and thus N-step) mechanism M_N and ℓ_N, now operating over all of $\mathcal{X}{=}\mathcal{U}$.

Proof. We interpret the losses using Def. 7.4.1, focussing on the integrand of the inner integral. Note that we can split it up into a finite sum of integrals of the form $\int_{n/N}^{n+1/N-} \pi(x)V(x)\,\mathrm{d}x$. When we do that for the left-hand formula $U_{\ell_N}[\pi\triangleright M_N]$ we see that throughout the interval $[{}^{n}/_{N}, {}^{n+1}/_{N})$ the contribution of the mechanism and the loss is constant, ie. $M_N(x)(y)\cdot\ell_N(w,x) = M({}^{n}/_{N})(y)\cdot\ell(w,{}^{n}/_{N})$. This means the integral becomes

$$M({}^{n}/_{N})(y)\cdot\ell(w,{}^{n}/_{N})\cdot\int_{n/N}^{n+1/N-}\pi(x)\,\mathrm{d}x$$

which is equal to $M({}^{n}/_{N})(y)\cdot\ell(w,{}^{n}/_{N})\cdot\pi_N(n/N)$. A similar argument applies to the last interval (which includes 1), compensated for by the definitions of ℓ_N and M_N to take their corresponding values from ${}^{1-N}/_{N}$.

Looking now at the right-hand formula, $U_{\ell}[\pi_N\triangleright M]$ we see that it is now exactly the finite sum of the integrals just described. □

7.5.4 Approximating continuous $\varepsilon\cdot\mathbf{d}$-private mechanisms

The techniques above give good discrete approximations for continuous-input $\varepsilon\cdot\mathbf{d}$-private mechanisms M acting on continuous priors simply by considering M_N's for increasing N's, using §7.5.3. As a convenient abuse of notation, when we *start* with a continuous-input mechanism M on $[0,1]$ we write M_N to mean the N-step mechanism that is made by first restricting M to the subset $\mathcal{U}_N$ of $[0,1]$ and then lifting that restriction "back again" as in Def. 7.5.2, effectively converting it into an N-step function. When we do this we find that the posterior loss wrt. N-step loss functions can be bounded above and below by using pixelated priors and N-stepped mechanisms.

Lemma 7.5. Let K be a continuous-input $\varepsilon\cdot\mathbf{d}$-private mechanism, and π in $\mathbb{M}[0,1]$

a continuous prior and ℓ a (non-negative) N-step function. Then the following inequalities hold:

$$\overbrace{e^{-\frac{\varepsilon}{N}} \cdot U_\ell[\pi_N \triangleright K_N]}^{discrete\,\mathcal{X}} \;\leq\; \overbrace{U_\ell[\pi \triangleright K]}^{continuous\,\mathcal{X}} \;\leq\; \overbrace{e^{\frac{\varepsilon}{N}} \cdot U_\ell[\pi_N \triangleright K_N]}^{discrete\,\mathcal{X}} .$$

(Notice that the middle formula $U_\ell[\pi \triangleright K]$, the mechanism K is not N-stepped, but in the formulae on either side they are as in Lem. 7.4.)

Proof. The proof is as for Lem. 7.4, but noting also that K's being $\varepsilon{\cdot}\mathbf{d}$-private implies that for all N we have $K(\lfloor x \rfloor_N)(y) \times e^{-\frac{\varepsilon}{N}} \leq K_N(x)(y) \leq K(\lfloor x \rfloor_N)(y) \times e^{\frac{\varepsilon}{N}}$. □

With Lem. 7.4 and Lem. 7.5 we can study optimality of $Ł^\varepsilon$ on finite discrete inputs $\mathcal{U}_N$. We will see that, although Geometric mechanisms are still optimal for the (effectively) discrete inputs $\mathcal{U}_N$, the Laplace mechanism provides increasingly good *approximate optimality* for $\mathcal{U}_N$ as N increases, and is in fact (truly) optimal in the limit.

7.6 The Laplace and Geometric mechanisms

In this section we make precise the restriction of the Geometric mechanism G^ε to inputs and outputs both in $\mathcal{U}_N$ (a subset of $[0,1]$): for both x, y in $\mathcal{U}_N$ we define

$$\overbrace{G^\varepsilon_N(x)(y)}^{\text{on } \mathcal{U}_N} \;:=\; \overbrace{G^{\frac{\varepsilon}{N}}(Nx)(Ny)}^{\text{on } \{0 \ldots N\}} \quad . \tag{7.7}$$

As an illustration, we take $\varepsilon{=}2\ln 4$ and input $\mathcal{X}{=}\mathcal{U}_2$, in which the 2 comes from $\mathcal{U}_2$ and the $\ln 4$ comes from the $\alpha{=}1/4$ of the Geometric distribution used to make the mechanism G^ε. Using the *three* points $0, 1/2$ and 1 of the input, we compute the truncated geometric mechanism G^ε_2 as the channel below, where the rows' labels are the inputs $\mathcal{U}_2$, and the columns are similarly labelled by the outputs (also $\mathcal{U}_2$ in this case). This means that if the input was 0, then the output (after truncation) will be 0 with probability $4/5$, and $1/2$ with probability $3/20$ etc:

$$G^\varepsilon_2 = \begin{array}{c|ccc} & 0 & 1/2 & 1 \\ \hline 0 & 4/5 & 3/20 & 1/20 \\ 1/2 & 1/5 & 3/5 & 1/5 \\ 1 & 1/20 & 3/20 & 4/5 \end{array} \quad .$$

Notice now that the ratio of adjacent probabilities that are in the same column satisfy the $\varepsilon{\cdot}\mathbf{d}$-privacy constraint, so for example $4/5 \div 1/5 = 3/5 \div 3/20 = 4 \leq e^{(2\ln 4)/2}$. Notice also that the distance between adjacent inputs in $\mathcal{U}_2$ under the Euclidean distance is $1/2$, not 1 as it would be in the conventional $\mathcal{X}{=}(0,1,2)$.

Suppose now that we consider $\mathcal{U}_4$ instead, consisting of the *five* points $0, 1/4, 1/2, 3/4$ and 1, and we adjust the α in the underlying Geometric distribution G_α. The parameter ε, now $4\ln 2$,

is the *same* as before – and the resulting matrix is

$G_4^\varepsilon =$

	0	1/4	1/2	3/4	1
0	2/3	1/6	1/12	1/24	1/24
1/4	1/3	1/3	1/6	1/12	1/12
1/2	1/6	1/6	1/3	1/6	1/6
3/4	1/12	1/12	1/6	1/3	1/3
1	1/24	1/24	1/12	1/6	2/3

As before though, the ratio of adjacent probabilities that are in the same column satisfy the $\varepsilon\cdot\mathbf{d}$-privacy constraint over all of $\mathcal{U}_4$: now we have $2/3 \div 1/3 = 1/3 \div 1/2 = 2 \leq e^{(4\ln 2)/4}$.

This demonstrates that the $\varepsilon\cdot\mathbf{d}$-privacy constraints over discrete inputs $\mathcal{U}_N$ must take into account the underlying metric on the input space. More generally, whenever we double N in $\mathcal{U}_N$, the α-parameter must become $\sqrt{\alpha}$.

7.6.1 The stability of optimality for geometric mechanisms

Using the above example we can explore the stability of the optimality result for the geometric mechanism when applied to other geometric mechanisms designed for different domains.

Consider the geometric mechanisms G_2^ε (designed for $\mathcal{U}_2$) and G_4^ε (designed for $\mathcal{U}_4$). Note that we can restrict G_4^ε to priors π_2 which only assign non-zero weight to the points in $\mathcal{U}_2$. The matrix made from G_4^ε but restricted to $\mathcal{U}_2$ consists of the first, middle and last rows of G_4^ε ie. :

$M =$

	0	1/4	1/2	3/4	1
0	2/3	1/6	1/12	1/24	1/24
1/2	1/6	1/6	1/3	1/6	1/6
1	1/24	1/24	1/12	1/6	2/3

We can see that the $\varepsilon\cdot\mathbf{d}$-privacy constraints consistent with $\mathcal{U}_2$ are still satisfied, ie. the relations between successive column entries in the restricted matrix lie between $1/4$ and 4 — because they represent *two* adjacencies from the original matrix, where the ratios were between $1/2$ and 2.

Now suppose that we choose a ℓ_2-legal loss function known as "Bayes' Risk", defined

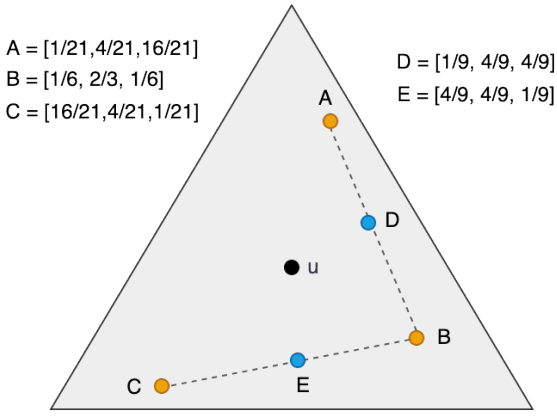
$$\mathsf{br}_2(w,x) \quad := \quad \begin{cases} 1, & \text{if } \lfloor x \rfloor_2 \neq w, \\ 0, & \text{otherwise.} \end{cases}$$

where $\mathcal{W} = \mathcal{U}_2$.

Assuming a uniform prior υ_2 over $\mathcal{U}_2$, we can compute the losses directly:

$$U_{\mathsf{br}_2}[\upsilon_2 \triangleright G_2^\varepsilon] \;=\; 4/15 \;<\; 1/3 \;=\; U_{\mathsf{br}_2}[\upsilon_2 \triangleright M]$$

showing that M is no longer universally optimal. This can alternatively be observed by comparing the barycentric representations for G_2^ε and M, depicted in Figure 7.2.



Construction of hypers for the mechanisms G_2^ε and M.

Figure 7.2: Illustration of geometric mechanisms from the example in §7.6.1 in the space of hypers. The supports of hyper $[\upsilon \triangleright G_2^\varepsilon]$ are depicted in orange and those of $[\upsilon \triangleright M]$ are the blue and orange points (combined). Observe that M is not a kernel mechanism (recall Def. 6.3.2 from Chapter 6) thus it cannot be (strictly) universally optimal.

Alternatively, we may want to *lift* a geometric mechanism designed for, say, $\mathcal{U}_2$ to an $\varepsilon{\cdot}\mathbf{d}$-private mechanism on $\mathcal{U}_4$. Notice that we cannot do this by simply taking G_2^ε and adding in the missing rows, since the $\varepsilon{\cdot}\mathbf{d}$-privacy constraints are no longer satisfied for the filled in rows. In fact we will need to *square root* the ε parameter for G_2^ε (ie. use $\varepsilon = \ln 2$, which produces the following channel

$$M_2 = \begin{array}{|c|ccc|}\hline & 0 & 1/2 & 1 \\ \hline 0 & 2/3 & 1/6 & 1/6 \\ 1/4 & 2/3 & 1/6 & 1/6 \\ 1/2 & 1/3 & 1/3 & 1/3 \\ 3/4 & 1/3 & 1/3 & 1/3 \\ 1 & 1/6 & 1/6 & 2/3 \\ \hline \end{array} .$$

Notice that now the ratio between adjacent elements is at most 2 so the privacy guarantee for M_2, like that of G_4^ε above, is $\varepsilon = 4 \ln 2$.

We can now compare the Bayes' Risk loss bf_4 (for $\mathcal{W} = \mathcal{U}_4$) under the uniform prior υ_4 for M_2 against G_4^ε, and we find,

$$U_{\mathsf{br}_4}[\upsilon_4 \triangleright G_4^\varepsilon] \;=\; 8/15 \;<\; 2/3 \;=\; U_{\mathsf{br}_4}[\upsilon_4 \triangleright M_2]$$

again showing that the lifted geometric mechanism M_2 is not optimal on $\mathcal{U}_4$.

These examples show in some sense the fragility of the optimality guarantee for the geometric mechanism, and for privacy mechanisms in general – optimality is very sensitive to the domain for which the mechanism is designed, and 'retro-fitting' mechanisms for new domains

has consequences for optimality.

7.6.2 Bounding the loss for continuous mechanisms

At this point, we have enough to be able to appeal to the discrete optimality result, to bound below the losses for continuous mechanisms, provided that the loss ℓ_N is N-legal, ie. that its legality obtains at least for the distinct points in $\mathcal{U}_N$.

> **Lemma 7.6.** For any continuous prior π in $\mathbb{M}\mathcal{U}$, $\varepsilon{\cdot}\mathbf{d}$-private-mechanism $M{:}\,\mathcal{U}\to\triangleright\mathcal{Y}$ and loss function $\ell{:}\,\mathcal{W}{\times}\mathcal{U}\to\mathbb{R}_{\geq 0}$ such that ℓ_N is N-legal, we have:
>
> $$U_{\ell_N}[\pi_N \triangleright G^\varepsilon_N] \quad\leq\quad U_{\ell_N}[\pi \triangleright M_N]$$
>
> *Proof.* Follows from Lem. 7.4 and noting that M restricted to $\mathcal{U}_N$ satisfies the conditions for universal discrete optimality [39]. □

Our next task is to study the relationship between the Geometric- and Laplace mechanisms. We show first that G^ε_N is refined (§7.2.2) by the truncated Laplace mechanism also restricted to to $\mathcal{U}_N$. Since $Ł^\varepsilon$ is already defined over the whole of $\mathcal{U}$ we continue to write its restriction to $\mathcal{U}_N$ as $Ł^\varepsilon$. This will immediately show that losses under the Geometric are no more than those under the Laplace (§7.2.2(1)), consistent with observations that, on discrete inputs, Laplace obfuscation does not necessarily minimise the loss [92]. Since the output $\mathcal{Y}$ of $Ł^\varepsilon$ is continuous, we proceed by first approximating it using post-processing to make Laplace-based mechanisms ${}^T Ł^\varepsilon$, defined below, which have discrete output, and which can form an anti-refinement chain converging to $Ł^\varepsilon$. We are then able to show separately the refinements between G^ε_N and ${}^T Ł^\varepsilon$, using methods designed for finite mechanisms.

The T,N-Laplace mechanisms approximate $Ł^\varepsilon$ by T-pixelation of their outputs. Here x is (still) in $\mathcal{U}_N$ but y is in $\mathcal{U}_T$.

$$\begin{array}{rcll} {}^T Ł^\varepsilon(x)(y) & := & Ł^\varepsilon(x)[y, y{+}1/T) & \textit{if } y{<}1{-}1/T \\ & & Ł^\varepsilon[1{-}1/T, 1] & \textit{otherwise.} \end{array} \tag{7.8}$$

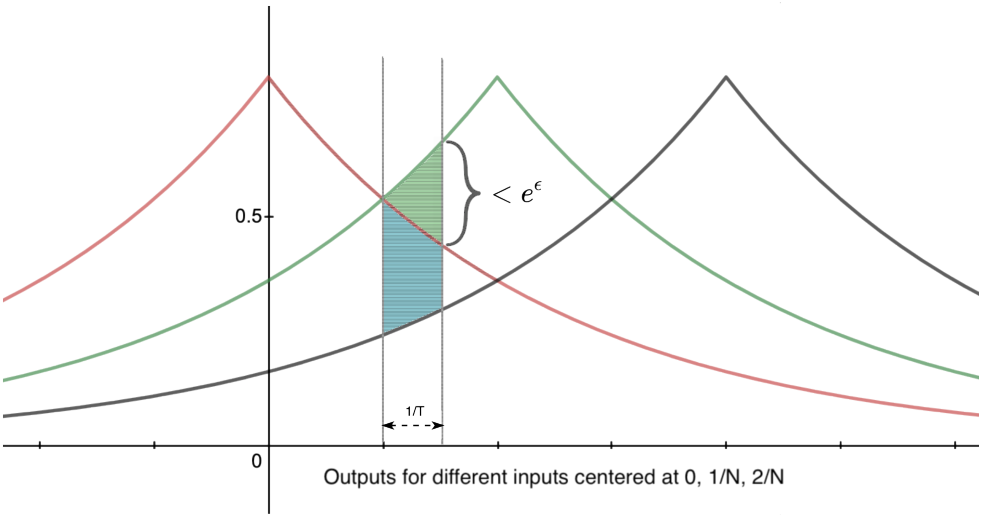
That is, we pixelate the $\mathcal{Y}$ using T for the Laplace (independently of the N we use for $\mathcal{X}$.) This is illustrated in Figure 7.3a.

Observe that as this is a post-processing (§7.2.2(3)) of the output of $Ł^\varepsilon$, the refinement $Ł^\varepsilon \sqsubseteq {}^T Ł^\varepsilon$ follows.

7.6.3 Refinement between N-Geometric and T,N-Laplace mechanisms

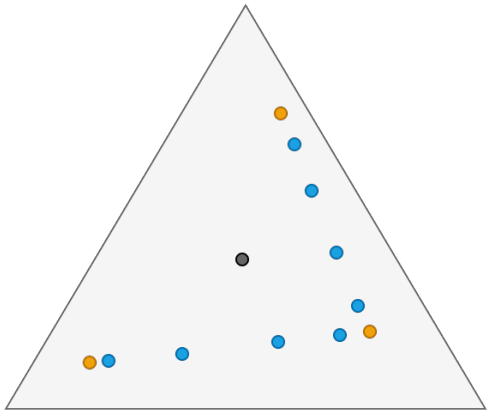
We now demonstrate the crucial fact that G^ε_N is refined by ${}^T Ł^\varepsilon$. We use version (b) of refinement, described in §7.2.2, and establish a Wasserstein-style earth-move between hypers $[\upsilon \triangleright G^\varepsilon_N]$ and $[\upsilon \triangleright {}^T Ł^\varepsilon]$ (ie. for uniform prior υ).

> **Lemma 7.7.** For all integer $T{>}0$ we have that $G^\varepsilon_N \sqsubseteq {}^T Ł^\varepsilon$.



The width of the central "vertical slice" is $1/T$.

(a) Illustrates batching the output for ${}^{T}L$ (similar for ${}^{T}Ł^{\varepsilon}$). The outputs (shown here as PDF plots) are batched into output segments of length $1/T$ in this example, for T=8. The segment from $[x, x{+}1/T)$ is indicated by the two vertical lines. The probability assigned to this segment is the area under the relevant curves. For the red curve it is the sum of the white and blue regions; for the green curve it is the sum of the white, blue and green regions and for the black curve it is only the white region.



(b) The supports of hypers $[\upsilon \triangleright G_2^{\varepsilon}]$ (orange) and $[\upsilon \triangleright {}^{8}Ł^{\varepsilon}]$ (blue) for inputs $\{0, 1/2, 1\}$ in the space of hypers. The blue points are within the convex hull of the orange points.

Figure 7.3: N-Geometric and T,N-Laplace mechanisms.

Proof. Take Δ, Δ' in $\mathbb{D}^2\mathcal{U}_N$ as hypers both with finite supports. We can depict such hypers in $\mathbb{R}^{N+1}$-space by locating their supports, each of which is a point in $\mathbb{R}^{N+1}$, where the axes of the diagram correspond to each point in $\mathcal{U}_N$. For example if we take Δ to be the hyperdistribution $[\upsilon \triangleright G_2^{\varepsilon}]$, it has three posterior distributions, which are 1-summing triples in $\mathbb{R}^3$. They are depicted by the orange points in Figure 7.3b. Similarly the supports of the hyperdistribution Δ' taken to be $[\upsilon \triangleright {}^{T}Ł^{\varepsilon}]$ are represented by the blue dots. Notice that the blue dots are contained in the convex hull of the orange dots, and this observation allows us

to prove that the mechanisms G_2^ε and ${}^8Ł^\varepsilon$ are in a refinement relation.

We recall Lem. 2.6 from Chapter 2 which we reformulate as follows:

> Let $C, C' : \mathcal{U}_N \to \mathcal{U}_T$ be channels and let υ be the uniform prior. If the supports of $[\upsilon \triangleright C]$ are linearly independent when considered as vectors in $\mathbb{R}^N$, and their convex hull encloses the supports of $[\upsilon \triangleright C']$, then $C \sqsubseteq C'$. [9]

To apply this result, we let C be G_N^ε recall that indeed the supports of $[\upsilon \triangleright G_N^\varepsilon]$ are linearly independent (see for example [35]). Moreover in general, the supports of $[\upsilon \triangleright {}^T Ł^\varepsilon]$ are also contained in the convex hull. We provide details of this latter fact in Appendix §B.2. □

Finally we can show full refinement between the Laplace and the Geometric mechanism, which follows from continuity of refinement [62].

THEOREM 7.8. $G_N^\varepsilon \sqsubseteq Ł^\varepsilon$.

Proof. We first form an anti-refinement chain $\ldots \sqsubseteq {}^{T_1}Ł^\varepsilon \sqsubseteq {}^{T_0}Ł^\varepsilon$ so that (a) $Ł^\varepsilon \sqsubseteq {}^{T_i}Ł^\varepsilon$ for all i, *and* (b) the chain converges to $Ł^\varepsilon$.

We reason as follows:

$$
\begin{array}{lll}
& G_N^\varepsilon \sqsubseteq Ł^\varepsilon & \\
\text{iff} & G_N^\varepsilon \sqsubseteq {}^{T_i}Ł^\varepsilon \quad \text{for all } i{\geq}0 & \text{“}\sqsubseteq \text{ is continuous; (a), (b) above”}
\end{array}
$$

which follows from Lem. 7.7. We provide details of (a), (b) just above in Appendix §B.2. □

We have shown that the Laplace mechanism is a refinement of the Geometric mechanism. This means that it genuinely leaks less information than does the Geometric mechanism and therefore affords greater privacy protections. On the other hand this means that we have lost utility with respect to the aggregated information. In the next section we turn to the comparison of the Laplace and Geometric mechanisms with respect to that loss.

7.6.4 The Laplace approximates the Geometric

The geometrical interpretation of the Laplace and Geometric mechanisms set out above indicates how the Laplace approximates the Geometric as $\mathcal{U}_N$'s interval-width approaches 0. In particular the refinement relationship established in Thm. 7.8 describes how the posteriors of $[\upsilon \triangleright {}^T Ł^\varepsilon]$ all lie in between pairs of posteriors of $[\upsilon \triangleright G_N^\varepsilon]$. This relationship between posteriors translates to a bound between the corresponding expected losses $U_\ell[\upsilon \triangleright Ł^\varepsilon]$ and $U_\ell[\upsilon \triangleright G_N^\varepsilon]$ via the Kantorovich-Rubinstein theorem applied to the hypers $[\upsilon \triangleright {}^T Ł^\varepsilon]$ and $[\upsilon \triangleright G_N^\varepsilon]$. We sketch the argument in the next theorem, and provide full details in Appendix §B.3.

[9] The lemma applies to channels because of the direct correspondence between channels and the supports of hyperdistrbutions formed from uniform priors.

THEOREM 7.9. Let ℓ be a κ-Lipschitz loss function, and υ the uniform distribution over $\mathcal{U}_N$. Then

$$U_\ell[\upsilon \triangleright \text{Ł}^\varepsilon] - U_\ell[\upsilon \triangleright G_N^\varepsilon] \quad \leq \quad c\kappa/N\,, \tag{7.9}$$

where $c = 3/(1-e^{-\varepsilon})^2$ is constant for fixed ε.

Proof. We appeal to the Kantorovich-Rubinstein theorem which states that the "Kantorovich distance" between probability distributions Δ, Δ' bounds above the difference between expected values $|\mathcal{E}_\Delta f - \mathcal{E}_{\Delta'} f|$ whenever f satisfies the κ-Lipschitz condition. In our case the relevant distributions are the *hyper*-distributions $[\upsilon \triangleright {}^T\text{Ł}^\varepsilon]$ and $[\upsilon \triangleright G_N^\varepsilon]$, and the relevant Lipschitz functions are Y_ℓ for loss functions ℓ. [10]

We write $\mathbb{W}(\Delta, \Delta')$ for the Wasserstein distance between hyperdistributions Δ, Δ' which is determined by the minimal *earth-moving* cost to transform Δ to Δ'. For any such earth move each posterior δ of Δ is reassigned to a selection of posteriors of Δ' in proportion to the probability mass that Δ assigns to δ. The cost of the move is the expected value of the distance between posterior reassignment (weighted by the proportion of the reassignment). Thus the cost of any specific earth move provides an upper bound to $\mathbb{W}(\Delta, \Delta')$. [11] Importantly for us, the relation of refinement $\sqsubseteq$ determines a specific earth move (recall Def. 2.4.2) whose cost we can calculate.

Referring to Lem. 7.7 and Figure 7.3b, we see that the refinement between the approximation to the Laplace $[\upsilon \triangleright {}^T\text{Ł}^\varepsilon]$ and $[\upsilon \triangleright G_N^\varepsilon]$, reassigns the Geometric's posteriors (the orange dots) to the Laplace's posteriors (the blue dots). Crucially though the Geometric's posteriors form a linear order according to their distance from one another, and the refinement described in Lem. 7.7 shows how each Laplace posterior lies in between adjacent pairs of Geometric posteriors (according to the linear ordering), provided that N divides T. Therefore any redistribution of a Geometric posterior is bounded above by the distance to one or other of its adjacent posteriors. We show in Appendix §B.3 that distances between adjacent pairs is bounded above by c/N, and therefore $\mathbb{W}([\upsilon \triangleright {}^T\text{Ł}^\varepsilon], [\upsilon \triangleright G_N^\varepsilon]) \leq c/N$.

Next we observe that if $\ell(w, x)$ is a κ-Lipschitz function on $[0, 1]$ (as a function of x), then Y_ℓ is a κ-Lipschitz function, and so by the Kantorovich-Rubinstein theorem we must have (recalling from (7.3)) that $U_\ell[\pi \triangleright M] = \mathcal{E}_{[\pi \triangleright M]} Y_\ell$:

$$U_\ell[\upsilon \triangleright {}^T\text{Ł}^\varepsilon] - U_\ell[\upsilon \triangleright G_N^\varepsilon] \quad \leq \quad c\kappa/N\,. \tag{7.10}$$

By Thm. 7.8 and postprocessing we see that $G_N^\varepsilon \sqsubseteq \text{Ł}^\varepsilon \sqsubseteq {}^T\text{Ł}^\varepsilon$. Recall from (a) that

[10] Some $f : \mathbb{D}\mathcal{X} \to \mathbb{R}$ is κ-Lipschitz if $|f(\delta) - f(\delta')| \leq \kappa \mathbb{W}(\delta, \delta')$, for $\kappa > 0$, and $\mathbb{W}(\delta, \delta')$ is the Kantorovich distance between δ, δ'.

[11] All the costs are determined by the underlying metric used to define the probability distributions. For us this is determined by the Euclidean distance on the interval $[0, 1]$.

refinement means that the corresponding losses are also ordered, ie.

$$U_\ell[\upsilon \triangleright G_N^\varepsilon] \;\leq\; U_\ell[\upsilon \triangleright Ł^\varepsilon] \;\leq\; U_\ell[\upsilon \triangleright^T Ł^\varepsilon]$$

and so the difference $U_\ell[\upsilon \triangleright Ł^\varepsilon] - U_\ell[\upsilon \triangleright G_N^\varepsilon]$ must be no more than the difference $U_\ell[\upsilon \triangleright^T Ł^\varepsilon] - U_\ell[\upsilon \triangleright G_N^\varepsilon]$, thus (7.9) follows from (7.10). Full details are set out in Appendix §B.3. □

More generally, (7.9) holds whatever the prior.

THEOREM 7.10. Let ℓ be a κ-Lipschitz loss function, and π any prior over $\mathcal{U}_N$. Then

$$U_\ell[\pi \triangleright Ł^\varepsilon] - U_\ell[\pi \triangleright G_N^\varepsilon] \quad\leq\quad c\kappa/N \,. \tag{7.11}$$

Proof. This follows as for Thm. 7.9, by direct calculation, noting that for discrete distributions we have $U_{\ell^*}[\upsilon \triangleright M] = U_\ell[\pi_N \triangleright M]$, where $\ell^*(w,x) := \ell(w,x) \times \pi_N(x) \times N$. Details are given in Appendix §B.3. □

7.6.5 Approximating monotonic functions

The final piece needed to complete our generalisation of the Ghosh et al.'s optimality theorem is monotonicity. We describe here how to approximate continuous monotonic functions, and expose the limitations for the Laplace mechanism. We begin by recalling the notion of monotone loss functions, defined in Chapter 6 on discrete domains but used here in the continuous setting:

DEFINITION 7.6.1. The loss function $\ell : \mathcal{W} \times \mathcal{X} \to \mathbb{R}$ is said to be *monotone* if: there is some mapping $\theta : \mathcal{W} \to [0,1]$, such that

$$\ell(w,x) \quad := \quad m(|\theta(w) - x|, x) \,,$$

where $m : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is monotone in its first argument.

Notice how θ takes care of any remapping that might need to be applied for computing expected losses. Interestingly step functions are not in general monotone on the whole of the continuous input $[0,1]$, but fortunately they are for the restrictions to $\mathcal{U}_N$ that we need.

LEMMA 7.11. Let ℓ be monotone. Then the approximation ℓ_T restricted to $\mathcal{U}_N$ is monotone whenever T is a multiple of N.

Proof. If $x \in \mathcal{U}_N$ then $\lfloor x \rfloor_T = x$ since N divides T. □

Examples of continuous monotone loss functions include ℓen and ℓen^2, where $x, w \in [0,1]$, and $\ell en(w,x) := |x - w|$. Note that ℓen is 1-Lipschitz and ℓen^2 is 2-Lipschitz.

We note finally that as the pixelation N of ℓ increases the approximations ℓ_N converge to ℓ.

7.7 Universal optimality for the Laplace Mechanism

We finally have all the pieces in place to prove Thm. 7.3, our generalisation of discrete optimality.

> **THEOREM 7.3 (Truncated Laplace is optimal).** Let K^ε be any continuous $\varepsilon\cdot\mathbf{d}$-private mechanism with input and output both $[0,1]$, and let π be any continuous (prior) probability distribution over $[0,1]$ and ℓ any Lipschitz continuous [12], legal loss function on $\mathcal{X}{=}\mathcal{U}$. Then $U_\ell[\pi \triangleright Ł^\varepsilon] \leq U_\ell[\pi \triangleright K^\varepsilon]$.
>
> *Proof.* We use the above results to approximate the expected posterior loss by step functions; these approximations are equivalent to posterior losses over discrete mechanisms satisfying $\varepsilon\cdot\mathbf{d}$-privacy, enabling appeal to Ghosh et al.'s universal optimality result on discrete mechanisms. We reason as follows:
>
> $$\begin{array}{lll} & U_{\ell_N}[\pi \triangleright K^\varepsilon] \times e^{\varepsilon/N} & \\ \geq & U_{\ell_N}[\pi_N \triangleright K^\varepsilon_N] & \text{“Lem. 7.5”} \\ \geq & U_{\ell_N}[\pi_N \triangleright G^\varepsilon_N] & \text{“Lem. 7.6: } \ell_N \text{ is legal by Lem. 7.11”} \\ \geq & U_{\ell_N}[\pi_N \triangleright Ł^\varepsilon] - \ c\kappa/N & \text{“Thm. 7.10; } \ell_N \text{ is } \kappa\text{-Lipschitz”} \\ \geq & U_{\ell_N}[\pi \triangleright Ł^\varepsilon] \times e^{-\varepsilon/N} \ - c\kappa/N \ . & \text{“Lem. 7.5”} \end{array}$$
>
> The result now follows as above by taking N to ∞, and noting that $e^{\varepsilon/N}$, $e^{-\varepsilon/N}$, $c\kappa/N$ and ℓ_N converge to $1, 1, 0, \ell$ respectively, and that taking expected values over fixed distributions is continuous. □

Note that Thm. 7.3 only holds for mechanisms that are $\varepsilon\cdot\mathbf{d}$-private. An arbitrary embedding K_N is not necessarily $\varepsilon\cdot\mathbf{d}$-private, and in particular Thm. 7.3 does not apply to G^ε_N. Also the continuous property on ℓ is required, because ℓ_N must be monotone for all N. Thus arbitrary step functions do not satisfy this property, and so the Laplace mechanism is not in general universally optimal wrt. arbitrary step functions. Two popular loss functions however are continuous, and thus we have the following corollary.

> **COROLLARY 7.12.** The Laplace mechanism is universally optimal for ℓ_{en} and $\ell_{en}{}^2$.

7.8 Conclusion and Future Work

We have studied the relationship between differential privacy (good) and loss of utility (bad) when the input $\mathcal{X}$ can be over an interval of the reals, instead of having $\mathcal{X}$ described as in the optimality result of Ghosh et al. [39], ie. as a discrete space. Here we used as input space $[0,1]$, but note that the result extends straightforwardly to any finite interval $[a,b]$ of $\mathbb{R}$. Our result also imposes the condition that the losses must be κ-Lipschitz for some finite κ. We do not know whether this condition can be removed in general.

[12] Lipschitz continuous is less general than continuous. It means that the difference in outputs is within a constant $\kappa{>}0$ scaling factor of the difference between the inputs.

We observe that for N-step loss functions, the Laplace mechanism is not optimal, and in fact a bespoke Geometric mechanism *will* be optimal for such loss functions. However our Thm. 7.10 however provides a way to estimate the error relative to the optimal loss, when using the Laplace mechanism.

Finally we note that the space of $\varepsilon{\cdot}\mathbf{d}$-private mechanisms is very rich, even for discrete inputs $\mathcal{U}_N$, suggesting that the optimality result given here will be useful whenever the input domain can be linearly ordered.

7.9 Chapter Notes

This chapter is based on the paper "The Laplace Mechanism has optimal utility for differential privacy over continuous queries" [41]. Omitted proofs and supporting materials can be found in Appendix §B.

The study of (universally) optimal mechanisms is one way to understand the cost of obfuscation, needed to implement privacy, but with a concomitant loss of utility of queries to databases. Pai and Roth [97] provide a detailed survey of the principles underlying the design of mechanisms including the need to trade utility with privacy, and Dinur et al. [98] explore the relationship between how much noise needs to be added to database queries relative to the usefulness of the data released. Our use of loss functions to measure utility follows both that of Ghosh et al. [39] and Alvim et al. [56], and concerns optimality for entire mechanisms that satisfy a particular level of $\varepsilon{\cdot}\mathbf{d}$-privacy. The mean error ℓen and the mean squared error ℓen^2 are used to evaluate loss, as described by Ghosh et al. [39].

Our result on the optimality for the Laplace mechanism is the first positive result in this area for this mechanism. This is surprising, given that differential privacy was originally designed with the Laplace mechanism in mind.[13] The Geometric mechanism, on the other hand, appears to be special for discrete inputs, as Ghosh et al. [39] showed when utility is measured using their "legal" loss functions.

Several negative results for the Laplace mechanism in terms of utility have been demonstrated by others when the inputs to the obfuscation are discrete [91], and where the optimisation is based on minimising the probability of reporting an incorrect value, subject to the $\varepsilon{\cdot}\mathbf{d}$-private-constraint. Similarly Geng et al. [92] show that adding noise according to a kind of "pixelated" distribution appears to produce the best utility for arbitrary discrete datasets. Such examples are consistent with our Thm. 7.8 showing where the Laplace mechanism is a refinement of the Geometric mechanism (loses more utility) when restricted to a discrete input (to the obfuscation). Note also that these definitions of optimality do not use a prior, and therefore represent the special case of utility per exact input, rather than a scenario where the observer's prior knowledge is included.

Alvim et al. [33] also use the framework of Quantitative Information Flow to study the relationship between the privacy and the utility of $\varepsilon{\cdot}\mathbf{d}$-private mechanisms. In their work they model utility in terms of a leakage measure, where leakage is defined as the ratio of input gain

[13] The use of e^{ε} in the definition allows a simple proof that the Laplace mechanism satisfies differential privacy.

to output gain wrt. a mechanism modelled as a channel.

The usefulness of the Laplace mechanism in real privacy applications has been demonstrated by Chatzikokolakis et al. [15] for geo-location privacy, in [99] for privacy-preserving graph analysis, and by Phan et al. [100] in deep learning.

Part III

Applications

8

Statistical Utility for Local Differential Privacy

Warner's randomised response protocol – described briefly in Chapter 1 and again in Chapter 5 (§5.2)– was designed to protect individuals' responses to survey questions while preserving the distribution of answers from the surveyed population. This algorithm has been shown to work remarkably well when the population of survey users is sufficiently large and, since it is also differentially private, has become almost synonymous with local differential privacy for protecting individuals' (binary) responses in any application. Notably, randomised response forms part of Google's RAPPOR mechanism [85] which is used for collecting and analysing user's personalisation settings while promising some privacy to the individuals who participate.

In this chapter we consider a scenario in which there is a metric of interest on the responses, and where the statistical utility of the mechanism depends on the underlying metric. In this case we find that a geometric mechanism, designed with a distance measure on inputs has much better utility than the conventional randomised response mechanism. We show how to recover the most likely distribution of responses given the noisy outputs using a Bayesian approach known as Iterative Bayesian Update (IBU)[1] and then measuring how far the recovered distribution is from the true distribution. The "distance between distributions" determines the overall utility, and when there is a metric of interest on the domain, the well-known Kantorovich distance is an appropriate distance measure between distributions. Experimentally we find that using this measure of utility, the geometric mechanism outperforms the randomised response mechanism on the two different types of distributions examined.

[1] Recall that differential privacy permits recovering estimates of distributions – these are not considered to be privacy breaches since the individual maintains "plausible deniability" as to whether their responses were the result of noise.

8.1 Scenario: Estimating a Distribution

Consider the following scenario:

Example 8.1.1. *A town planner wishes to collect location information on individuals within an urban area in order to decide the best location for wifi hotspots which operate within a fixed radius. The planner decides to use differential privacy to allow users to protect their locations while permitting her to perform a reliable statistical analysis of the distribution of locations. However, the town planner needs to estimate the population distribution in such a way that estimates too far (in physical distance) from the true distribution are penalised, as these would result in incorrect placement of the hotspots. How should the town planner choose an appropriate mechanism?*

The usual mechanism for collecting survey responses, as described in the introduction, is the randomised response mechanism. However, in this example, the utility of the estimated distribution is affected by the ground distance of the estimate from the true distribution. If the estimated peaks are too far (in ground distance) from the true population, then the hotspots will be poorly located and unable to service the population effectively. In this case, we expect to be better serviced by a mechanism designed for the underlying metric which better preserves the ground distance.

In order to test this theory we require a measure of utility appropriate to the town planner's requirements. The well-known Kantorovich distance, also known as the Earth Mover's distance, is such a measure, since it describes the distance between distributions by penalising variations which are further from each other according to some ground measure. In the above scenario, we would lift the *Euclidean* distance to a Kantorovich metric in order to penalise estimated population clusters which are too far (in ground distance) from the true clusters, so as to reduce the cost to the town planner of (inevitable) inaccuracies in the statistical analysis.

In this work we investigate the statistical utility of the randomised response and geometric mechanisms for a simplified (1-dimensional) version of the above problem. We suppose that the data collector wishes to collect some sensitive numeric data from users (eg. *age* or *salary*) in a local differential privacy setting (ie. each user obfuscates their own sensitive value). As with the above example, the data collector's utility is affected by the Euclidean distance between secret values, in the sense that the estimated distribution of outputs should not deviate too far from the original (true) distribution as measured by the Kantorovich distance between the distributions. This will be described in more detail in the coming sections.

8.2 Comparing Mechanisms for Privacy

In terms of privacy, we recall that the randomised response and the geometric mechanisms are designed for different metrics; that is, they protect individuals' privacy in fundamentally different ways. The randomised response mechanism, designed for the Discrete metric, promises that an individual's response is indistinguishable from *all other participants*, since it distributes outputs uniformly (apart from the true response). In contrast, the geometric mechanism, designed for the Euclidean metric, promises indistinguishability within a (Euclidean) radius of

inputs. We therefore need to establish a reasonable way to compare the (truncated) geometric and randomised response mechanisms from the perspective of privacy.

We begin with some technical details. We assume a discrete input space of secrets $\mathcal{X} = \{0, 2, \ldots, 100\}$, and the same output space, ie. $\mathcal{Y} = \mathcal{X}$. We recall the *truncated* geometric and randomised response mechanisms from Chapter 3:

> **Definition 3.4.2** (Truncated Geometric Mechanism)**.** The truncated α-geometric mechanism $TG: \mathcal{X} \to \mathbb{D}\mathcal{X}$ where $\mathcal{X} = \{0, 1, \ldots, n\}$ has the following channel matrix:
>
> $$TG_{x,y} = \frac{1-\alpha}{1+\alpha} \cdot \alpha^{\mathbf{d}_2(x,y)} \qquad \text{for } y \in \{1, \ldots, n-1\}$$
> $$TG_{x,y} = \frac{1}{1+\alpha} \cdot \alpha^{\mathbf{d}_2(x,y)} \qquad \text{for } y \in \{0, n\}$$
>
> where $\alpha \in (0, 1]$. This mechanism satisfies $\varepsilon \cdot \mathbf{d}_2$-privacy where $\mathbf{d}_2$ is the Euclidean metric and $\varepsilon = -\ln \alpha$.

> **Definition 3.4.3** (Randomised Response Mechanism)**.** The α-randomised response mechanism $R: \mathcal{X} \to \mathbb{D}\mathcal{X}$ has the following channel matrix:
>
> $$R_{x,x} = 1/k$$
> $$R_{x,y} = \alpha/k \qquad \text{for } x \neq y$$
>
> where k is a normalisation term and $\alpha \in (0, 1]$. This mechanism satisfies $\varepsilon \cdot \mathbf{d}_{\mathrm{D}}$-privacy where $\mathbf{d}_{\mathrm{D}}$ is the discrete metric and $\varepsilon = -\ln \alpha$.

Recall (Lem. 4.15) that the truncated geometric mechanism can be deployed safely in place of the (infinite) geometric mechanism as they have the same leakage properties.

As mentioned above, in order to make a fair comparison, the privacy parameters of these mechanisms need to be calibrated so that they represent a comparable level of privacy.

Consider the randomised response mechanism with parameter $\varepsilon = \ln 2$ operating over integer-valued input and output domains with range $[0, 100]$. The privacy guarantee provided by this mechanism is given by the upper bound $e^{\varepsilon \ln 2} = 2$, representing the maximum likelihood ratio between any possible reported value and the true value. This upper bound is realised for every pair of different values in the input and output domains. By comparison, the truncated geometric mechanism with the same $\varepsilon = \ln 2$ would provide such an upper bound 2 only for values immediately adjacent to the true one (ie. at distance 1). For values further away, the bound is smaller (ie. $e^{\varepsilon \cdot \mathbf{d}_2}$), making more distant values less likely. If we want to provide the same upper bound 2 on the entire domain for the truncated geometric, then we would have to set ε to a value 100 times smaller, namely $\ln 2/100$, which would result in a very flat curve, making the true value almost indistinguishable from every other value.

However, we argue that it is not necessary to inject so much noise, as this destroys the utility-by-design of the geometric mechanism. As a compromise we will require the upper bound 2 on

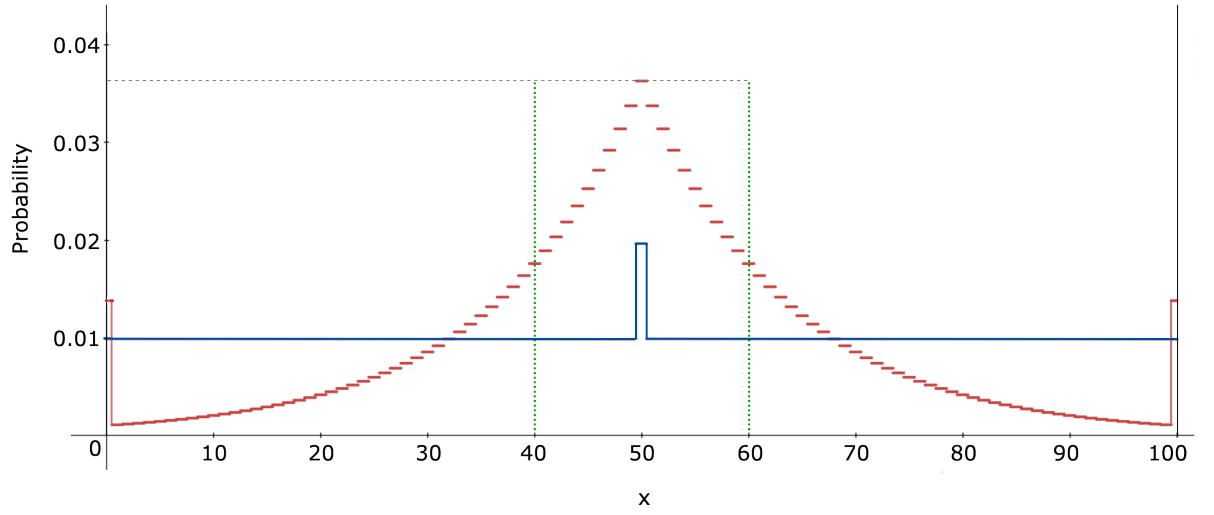


Figure 8.1: The distributions generated by the randomised response (blue) and truncated geometric (red) mechanisms applied to the input $x = 50$. The value of the privacy parameter ε is $\ln 2$ and $\ln 2/10$, respectively. This means that within a radius of 10 of the true value (ie. from values 40 to 60, indicated in green) the privacy guarantee for both mechanisms is $\varepsilon = \ln 2$, but outside this radius, the truncated geometric mechanism's privacy guarantee degrades with distance while the randomised response mechanism maintains the same guarantee $\varepsilon = \ln 2$.

a restricted subset of elements, for instance those in a radius 10 from the true value. This can be achieved by setting ε to $\ln 2/10$. Figure 8.1 illustrates the situation.

8.3 Comparing Mechanisms for Utility

We now turn to the comparison of these mechanisms from the perspective of utility. We begin with the problem of how to estimate the original distribution given a noisy distribution, and for this we use a technique known as Iterative Bayesian Update (IBU) [101], which we show can be applied to our problem. We then introduce Kantorovich metric which we will use to compare the statistical utility of the estimated distribution against the true distribution for the geometric and randomised response mechanisms.

8.3.1 Reconstructing the Original Distribution

Assume that we have a collection of N noisy data elements representing the result of the independent application of a randomised mechanism to the data of a certain population. We assume that each datum is a number in $\{0 \ldots n\}$. Let $\pi \in \mathbb{D}\{0 \ldots n\}$ be the prior distribution on the original data. The set of original data is generated by a sequence of random variables $X_1, X_2, \ldots, X_N$ independently and identically distributed (i.i.d.), according to π. To each of the $X_1, X_2, \ldots, X_N$ we apply the (truncated) geometric mechanism G, thus obtaining a sequence of random variables $Y_1, Y_2, \ldots, Y_N$. Let $q \in \mathbb{D}\{0 \ldots n\}$ be the *empirical* distribution determined by $Y_1, Y_2, \ldots, Y_N$. ie. The probability $q_j = q(j)$ is obtained by counting the frequency of the value j in $Y_1, Y_2, \ldots, Y_N$

so that $q_j = {}^{|\{h|Y_h=i\}|}/_N$.

The task we consider here is how best to reconstruct the original distribution π from q. An obvious method for computing the most likely original distribution given the empirical distribution q is to invert the channel matrix (if this is possible) and multiply it by q, since we have for invertible channel M the simple formula

$$\pi M = q \quad \Rightarrow \quad \pi = qM^{-1} \,.$$

ie. The empirical (observed) distribution q is computed as the *most likely* distribution given the prior π, and by inverting M we can recover the most likely prior given q.

Unfortunately, in some instances the calculation does not produce a sensible result at all, eg., an estimated "distribution" whose values lie outside $[0, 1]$. For example, consider the randomised response mechanism

$$R = \begin{pmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{pmatrix}$$

and imagine we observe the empirical distribution $q = (9/10, 1/10)$. The matrix inversion method yields $\pi = qR^{-1} = (17/10, -7/10)$ which is not a valid distribution.

Our solution is to use Iterative Bayesian Update (IBU) [101] to estimate the original distribution; IBU is guaranteed to produce a sensible estimate – and, in fact, the most likely distribution – under reasonable assumptions (discussed below).

8.3.2 Maximum Likelihood Estimation with IBU

Instead of using the "matrix inversion" method described above, we consider the following iterative procedure, inspired by Bayes' theorem.[2] As before, recall that the task we consider here is how best to reconstruct the original distribution π from q. Our iterative procedure will estimate π using a sequence of estimates $p^{(k)}$ which can be shown to converge to the most likely estimate of π. We begin with an estimate $p \in \mathbb{D}\{0 \ldots n\}$ which is full support, and we denote by p_i the probability $p(i)$ assigned to the input i. G is the (truncated) geometric mechanism on $\{0 \ldots n\}$ and we denote by $G_{i,j}$ the probability $G(i)(j)$, ie. of output j given input i, as assigned by G.

Definition 8.3.1. Define $\{p^{(k)}\}_k$ as follows:

$$\begin{aligned} p^{(0)} &= p \\ p_i^{(k+1)} &= \sum_j q_j \frac{p_i^{(k)} G_{i,j}}{\sum_h p_h^{(k)} G_{h,j}} \end{aligned}$$

The interest of the above definition relies in the following result:

Theorem 8.1. [103] Let $\{p^{(k)}\}_k$ be the sequence of distributions constructed according to Def. 8.3.1. Then:

[2] This algorithm is also known as an Expectation Maximisation (EM) algorithm [102].

1. The sequence converges, ie., $\lim_{k\to\infty} p^{(k)}$ exists.

2. $\lim_{k\to\infty} p^{(k)}$ is the *Maximum Likelihood Estimator* (MLE) of π given q.

We will denote by p^* the limit of the sequence $\{p^{(k)}\}_k$, ie., $p^* := \lim_{k\to\infty} p^{(k)}$. Thm. 8.1 means that the probability of observing an output distribution q produced from a geometric mechanism G is maximised by the prior p^*. Note that this does not mean that p^* is necessarily the true distribution, simply that it is the most likely given G and q.

Furthermore, p^* can be characterised using G. Namely,

PROPOSITION 8.2. [103] If $r = q\,G^{-1}$ is a probability distribution, then $p^* = r$.

We note that the above results have been shown to hold for any mechanism that is invertible (as a matrix). That is, Thm. 8.1 holds when G in Def. 8.3.1 is replaced with any invertible (channel) matrix [104]. This means it can equally be applied to the randomised response mechanism, since we have noted previously that it is invertible.[3] We also note that the convergence property is only guaranteed to hold under the conditions that the initial estimate for p is full support [104]. In this case it is always safe to choose $p = \upsilon$, the uniform prior.

We will use the above procedure to estimate the true distribution given an empirical distribution in our experimental setup.

8.3.3 Measuring Utility

As for statistical utility, intuitively it should account for how well we can approximate statistics on the original data by using only the collected noisy data. This can be formalised in terms of the distance between the original distribution and the most likely one given the noisy data. In the scenario provided earlier, the notion of *distance* played a part in the utility for the provider, whose goal was to place hotspots to maximise coverage for users; in this example, errors *further away* from the true location carry higher penalties than errors close to the true location of users. Therefore an appropriate notion of distance would be the Kantorovich metric, since it takes into account the ground distance between values in the distribution, and is related to a large class of statistical functions [105].

We recall the definition of the Kantorovich metric between distributions which we state as follows:

DEFINITION 8.3.2. Let $(\mathcal{X}, \mathbf{d})$ be a metric space and let $\mu, \mu' : \mathbb{D}\mathcal{X}$. The Kantorovich distance based on $\mathbf{d}$ between distributions μ and μ' is defined as follows:

$$K_d(\mu,\mu') \quad := \quad \max_{g\in\mathcal{G}} \left| \sum_{x\in\mathcal{X}} g(x)\mu(x) - \sum_{x\in\mathcal{X}} g(x)\mu'(x) \right|$$

[3] See proof of Lem. 6.30.

where $\mathcal{G}$ is the set of 1-Lipschitz functions on $\mathcal{X}$. ie.

$$g \in \mathcal{G} \quad \text{iff} \quad \forall x, x' \in \mathcal{X}, \ \mid g(x) - g(x') \mid \ \leq \ \mathbf{d}(x, x') \ .$$

8.4 Experimental Results

We now present the results of experiments designed to assess the statistical utility of the geometric- and randomised response- mechanisms using the IBU method outlined in §8.3. In our experiments, we construct synthetic datasets of users drawn from a distribution, then create a distribution of noisy outputs by applying a noise-adding mechanism to each user's input. We then determine the utility of the noise-adding mechanism by comparing the Kantorovich distances between the true input distribution of user values and the estimated distribution constructed from the IBU process. We describe the details below.

We assume that the space of inputs to each mechanism are integers in the range $[0, 100]$. We constructed two different mechanisms to output noisy values: a truncated geometric mechanism parametrised by $\varepsilon = ln2/10$ and a randomised response mechanism parametrised by $\varepsilon = ln2$.

Experiments were conducted on "populations" of size 1000, 10 000, 50 000 and 100 000. We experimented with 2 different population distributions: a binomial distribution, and a "4-point" distribution (ie. a random distribution over 4 'points' in the output range). For each of the 8 samples (4 populations over 2 distributions) we conducted 10 experiments using the following method:

1. Obfuscate the sample using each of the (geometric and randomised response) mechanisms to produce 2 obfuscated sets.

2. Convert each set into an empirical distribution over outputs using the frequency counts of elements in each set.

3. Run IBU for 5000 iterations over each empirical distribution to compute the maximum likelihood estimate (MLE) for the true distribution.[4]

4. Compare the Kantorovich distance between the MLE and the true distribution as an estimate of the error caused by the obfuscation.

In Figure 8.2 we present some sample runs of IBU for each mechanism and distribution. Interestingly, the reconstructed distribution for randomised response appears to be much closer to the true distribution for the '4-point' sample than for the binomial sample. Conversely, the reconstructed distribution for the geometric mechanism appears much closer to the true distribution for the binomially distributed sample.

However, the computed Kantorovich distances at the 5000 iteration point for each run tell a different story. These results are shown in Figure 8.3. We computed the Kantorovich distance between the estimated distribution and the true distribution, providing an approximation of

[4] 5000 iterations was experimentally determined to be a sufficient number to approach convergence of the IBU procedure.

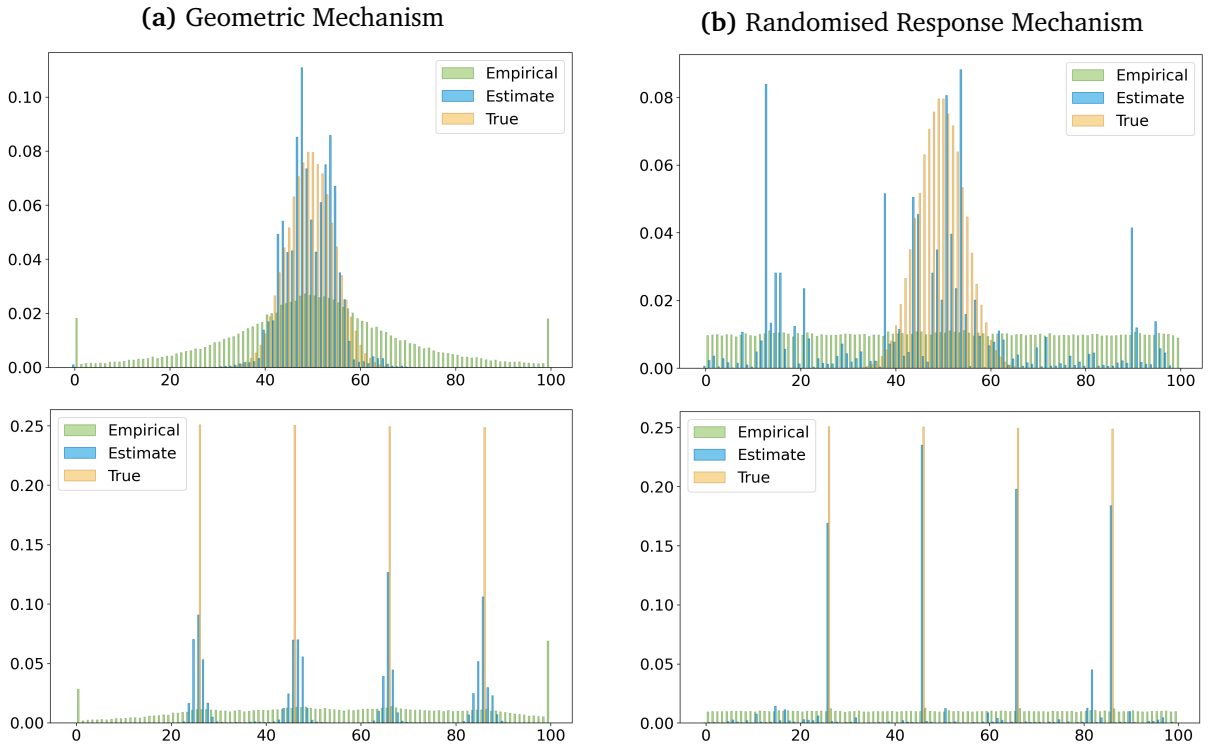


Figure 8.2: IBU reconstruction of true (orange) distributions from noisy (green) distributions based on 100k samples drawn from binomial (top) and "4-point" (bottom) distributions. The blue graphs indicate the reconstructed (estimated) distribution computed using IBU. By observation, it appears that the reconstructed distribution (blue) is closer to the true distribution (orange) for the geometric mechanism when the true distribution is binomial, but that the randomised response mechanism performs better when the true distribution is a point distribution. However, using the Kantorovich metric as a measure of utility, we find that the geometric mechanism performs better on both input distributions (see Figure 8.3).

the distance between the true distribution and the distribution resulting from obfuscation. We can see that the average Kantorovich distances for the geometric mechanism are significantly lower (up to 5 times) than the corresponding distances for the randomised response mechanism. We conjecture that this is because the errors caused by randomised response are randomly distributed over the entire output space, which directly affects the Kantorovich distance since it depends on the ground distance between points. This means that for statistical applications in which the ground distance is important, the geometric mechanism is still better performing than the randomised response mechanism.

Another interesting observation we make is in the convergence rates for the IBU method when applied to the different distributions. This is graphed in Figure 8.4. For each iteration of IBU we computed the 'log likelihood' function

$$L(\Theta) = \sum_{y} q_y \log(\Theta \cdot M_y)$$

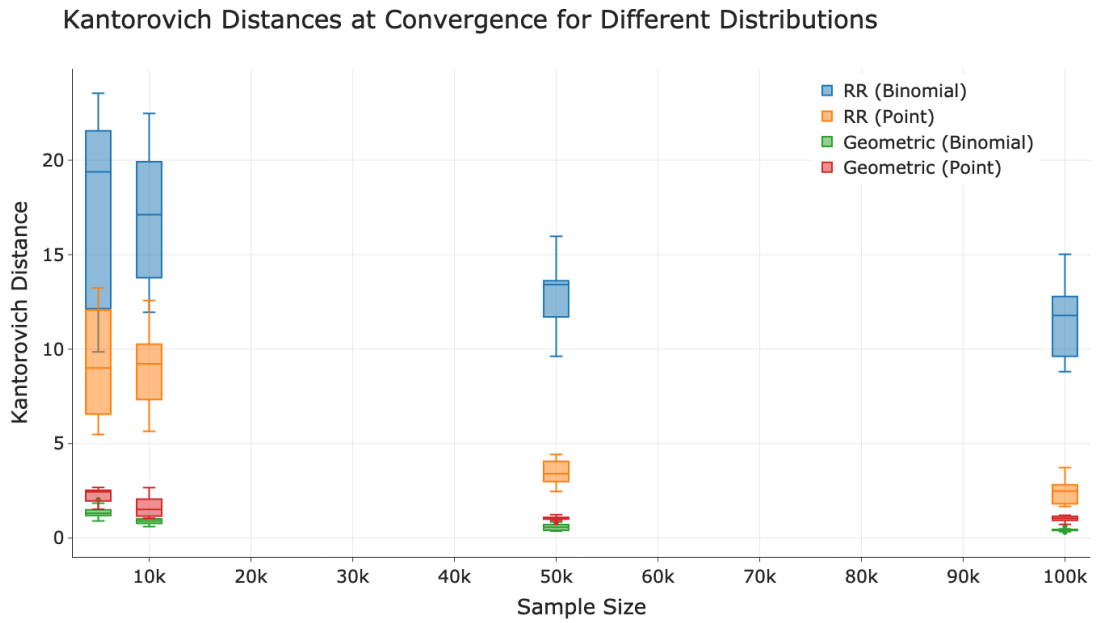


Figure 8.3: Kantorovich distances between true and estimated distributions at IBU convergence for the geometric and randomised response (RR) mechanisms. Distances were computed over 20 experiments for each of the 4 sample sizes indicated. The graph shows that the geometric mechanism performs much better than the randomised response mechanism on all datasets, as measured by the average Kantorovich distance between the true distribution and the estimated distribution computed using IBU.

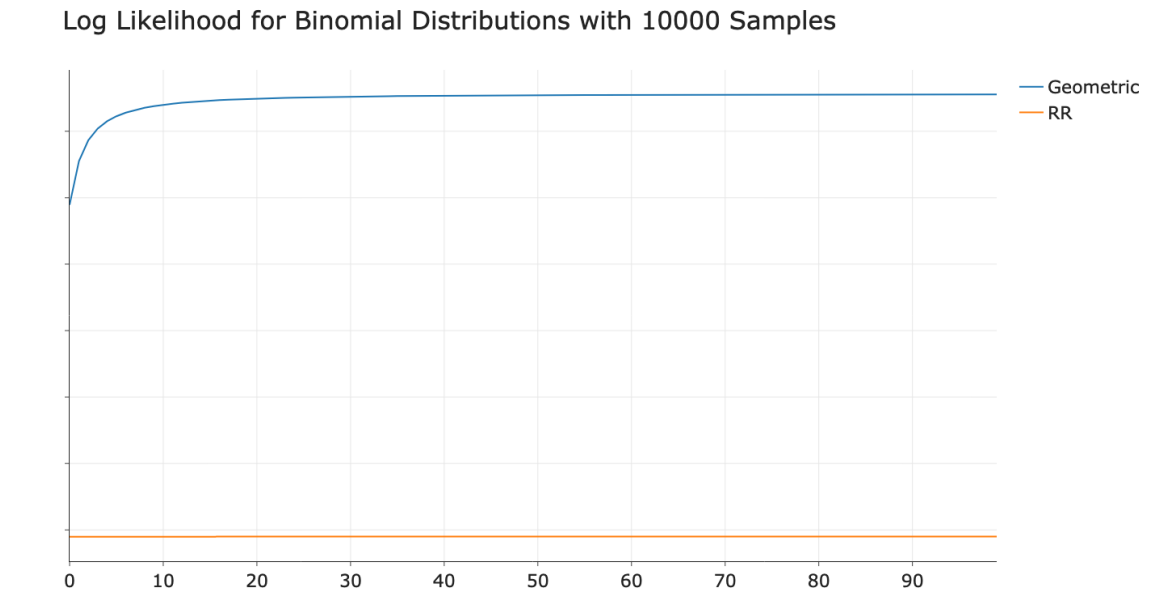


Figure 8.4: Log likelihood function against number of iterations for the geometric and randomised reponse mechanisms. This graph shows how fast each output distribution converges to the MLE for one particular (representative) run of the IBU. We observe that the geometric mechanism converges quickly whereas convergence for the randomised response is almost flat.

where Θ is the current estimated distribution, q_y is the empirical distribution and M is the mechanism represented as a channel matrix. [5] The log likelihood function indicates how close the current estimate is to the true MLE. The results for one particular run are shown in Figure 8.4. We can see that the geometric mechanism converges to a close approximation of the MLE within 10 iterations, whereas the convergence for randomised response is linear and almost flat. This may also explain the better performance of the randomised response output on the '4-point' sample, since there were far fewer 'skyscrapers' in the original distribution to estimate. The shape of the geometric mechanism seemed to favour the more 'natural' shape of the binomial distribution sample.

8.5 Conclusion

In this chapter we have shown how an understanding of the utility requirements of a problem can influence the choice of metric to use for local differential privacy. In particular, we showed that using the geometric mechanism instead of the randomised response mechanism in the application of local differential privacy improves the statistical utility of the outputs – as measured by the ability to recompute the original distribution from the noisy distribution in a way which minimises the Kantorovich distance. In addition, we saw that the use of the geometric mechanism involved a compromise for privacy, in that the differential privacy guarantee is weakened to a guarantee over a Euclidean radius of inputs rather than the entire dataset (as with the Discrete metric). Finally, we found that the Iterative Bayesian Update (IBU) method for estimating the true distribution was also much more efficient when applied to the noisy outputs of the geometric mechanism than for the randomised response mechanism. We leave as future work the further investigation of the properties of the IBU mechanism.

8.6 Chapter Notes

This chapter is based on the published paper "Utility-Preserving Privacy Mechanisms for Counting Queries" [42].

The Iterative Bayesian Update procedure was first introduced in [101] and its properties have been recently re-examined and generalised by ElSalamouny et al. [104].

[5] The notation $\Theta \cdot M_y$ indicates the dot product of Θ with the yth column of M.

9

Text Document Privacy

The problem of how to "obfuscate" texts by removing stylistic clues which can identify authorship is a challenging problem inspired by the success of author *identification* methods in revealing the identity of authors who wish to remain anonymous. In this chapter we explain how this interesting problem can be addressed with **d**-privacy. We will show that privacy for text documents can be modelled with **d**-privacy by utilising the natural metric space that arises from the representation of text documents as "bags-of-words" – these representations are typical in machine learning and contain sufficient information to carry out many kinds of classification tasks including *topic identification* and *authorship attribution* (of the original documents). We will provide a **d**-private mechanism for a novel metric – the Earth Mover's distance – which is the metric for semantic similarity used in natural language processing. We will also demonstrate our mechanism with some experiments on a fan fiction dataset, demonstrating that the mechanism provides sufficient utility for some text processing tasks of interest.

9.1 Introduction

Partial public release of formerly classified data incurs the risk that more information is disclosed than intended. This is particularly true of data in the form of text such as government documents or patient health records. Nevertheless there are sometimes compelling reasons for declassifying data in some kind of "sanitised" form — for example government documents are frequently released as redacted reports when the law demands it, and health records are often shared to facilitate medical research. Sanitisation is most commonly carried out by hand but, aside from the cost incurred in time and money, this approach provides no guarantee that the original privacy or security concerns are met.

To encourage researchers to focus on privacy issues related to text documents, the digital forensics community PAN@Clef ([106], for example) proposed a number of challenges that are typically tackled using *machine learning*. In this chapter we will demonstrate how to use **d**-privacy to address some aspects of the PAN@Clef challenges by showing how to provide strong a priori privacy guarantees in document disclosures.

We focus on the problem of *author obfuscation*, namely to automate the process of changing a given document so that as much as possible of its original substance remains, but that the author of the document can no longer be identified. Author obfuscation is very difficult to achieve because it is not clear exactly what to change that would sufficiently mask the author's identity. In fact author properties can be determined by "writing style" with a high degree of accuracy: this can include author identity [107] or other undisclosed personal attributes such as native language [108, 109], gender or age [110, 111]. These techniques have been deployed in real world scenarios: native language identification was used as part of the effort to identify the anonymous perpetrators of the 2014 Sony hack [112], and it is believed that the US NSA used author attribution techniques to uncover the identity of the real humans behind the fictitious persona of Bitcoin "creator" Satoshi Nakamoto.[1]

This work concentrates on the perspective of the "machine learner" as an adversary that works with the standard "bag-of-words" representation of documents often used in text processing tasks. A *bag-of-words* representation retains only the original document's words and their frequency (thus forgetting the order in which the words occur). Remarkably this representation still contains sufficient information to enable the original authors to be identified (by a stylistic analysis) *as well as* the document's topic to be classified, both with a significant degree of accuracy. [2] Within this context we reframe the PAN@Clef author obfuscation challenge as follows:

> Given an input bag-of-words representation of a text document, provide a mechanism which changes the input without disturbing its topic classification, but that the author can no longer be identified.

We implement a mechanism K which takes b, b' bag-of-words inputs and produces "noisy" bag-of-words outputs determined by $K(b), K(b')$ with the following properties:

> **PRIVACY:** If b, b' are classified to be "similar in topic" then, depending on a privacy parameter ε the *outputs* determined by $K(b)$ and $K(b')$ are also "similar to each other", irrespective of authorship.
>
> **UTILITY:** Possible outputs determined by $K(b)$ are distributed according to a Laplace probability density function scored according to a semantic similarity metric.

In what follows we define *semantic similarity* in terms of the classic *Earth Mover's distance* used in machine learning for topic classification in text document processing. [3] We explain how

[1] https://medium.com/cryptomuse/how-the-nsa-caught-satoshi-nakamoto-868affcef595

[2] This includes, for example, the character n-gram representation used for author identification in [113].

[3] In NLP, this distance measure is known as the Word Mover's distance. We use the classic Earth Mover's here for generality.

to combine this with **d**-privacy to produce a mechanism which obfuscates texts while preserving useful information.

In §9.2 we set out the details of the bag-of-words representation of documents and define the Earth Mover's metric for topic classification. In §9.3 we define a generic mechanism which satisfies "$E_{\mathbf{d}}$-privacy" relative to the Earth Mover's metric $E_{\mathbf{d}}$ and show how to use it for our obfuscation problem. We note that our generic mechanism is of independent interest for other domains where the Earth Mover's metric applies. In §9.4 we describe how to implement the mechanism for data represented as real-valued vectors and prove its privacy/utility properties with respect to the Earth Mover's metric; in §9.5 we show how this applies to bags-of-words. Finally in §9.6 we provide an experimental evaluation of our obfuscation mechanism, and discuss the implications.

Preliminaries

Throughout we assume standard definitions of probability spaces [114]. For a set $\mathcal{A}$ we write $\mathbb{D}\mathcal{A}$ for the set of (possibly continuous) probability distributions over $\mathcal{A}$. For $\eta \in \mathbb{D}\mathcal{A}$, and $A \subseteq \mathcal{A}$ a (measurable) subset we write $\eta(A)$ for the probability that (wrt. η) a randomly selected a is contained in A. In the special case of singleton sets, we write $\eta\{a\}$. For mechanism $K\colon \alpha \to \mathbb{D}\alpha$, we write $K(a)(A)$ for the probability that if the input is a, then the output will be contained in A.

9.2 Documents, Topic Classification and Earth Moving

In this section we summarise the required elements from machine learning and text processing required to address our problem. Our first definition sets out the representation for documents we shall use throughout. It is a typical representation of text documents used in a variety of classification tasks.

> **Definition 9.2.1.** Let $\mathcal{S}$ be the set of all words (drawn from a finite alphabet). A *document* is defined to be a finite bag over $\mathcal{S}$, also called a *bag-of-words*. We denote the set of documents as $\mathbb{B}\mathcal{S}$, i.e. the set of (finite) bags over $\mathcal{S}$.

Once a text is represented as a bag-of-words, depending on the processing task, further representations of the words within the bag are usually required. We shall focus on two important representations: the first is when the task is semantic analysis for eg. topic classification, and the second is when the task is author identification. We describe the representation for topic classification in this section, and leave the representation for author identification for §9.5 and §9.6.

9.2.1 Word embeddings

Machine learners can be trained to classify the topic of a document, such as "health", "sport", "entertainment"; this notion of topic means that the words within documents will have particular semantic relationships to each other. There are many ways to do this classification, and in this

paper we use a technique that has as a key component "word embeddings", which we summarise briefly here.

A *word embedding* is a real-valued vector representation of words where the precise representation has been experimentally determined by a neural network sensitive to the way words are used in sentences [115]. Such embeddings have some interesting properties, but here we only rely on the fact that when the embeddings are compared using a distance determined by a pseudometric[4] on $\mathbb{R}^n$, words with similar meanings are found to be close together as word embeddings, and words which are significantly different in meaning are far apart as word embeddings.

DEFINITION 9.2.2. An n-dimensional word embedding is a mapping $Vec : \mathcal{S} \to \mathbb{R}^n$. Given a pseudometric $dist$ on $\mathbb{R}^n$ we define a distance on words $dist_{Vec} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\geq}$ as follows:

$$dist_{Vec}(w_1, w_2) \quad := \quad dist(Vec(w_1), Vec(w_2)) \ .$$

Observe that the property of a pseudometric on $\mathbb{R}^n$ carries over to $\mathcal{S}$.

LEMMA 9.1. If $dist$ is a pseudometric on $\mathbb{R}^n$ then $dist_{Vec}$ is also a pseudometric on $\mathcal{S}$.

Proof. Immediate from the definition of a pseudometric: i.e. the triangle equality and the symmetry of $dist_{Vec}$ are inherited from $dist$. □

Word embeddings are particularly suited to language analysis tasks, including topic classification, due to their useful semantic properties. Their effectiveness depends on the quality of the embedding *Vec*, which can vary depending on the size and quality of the training data. We provide more details of the particular embeddings in §9.6. Topic classifiers can also differ on the choice of underlying metric *dist*, and we discuss variations in §9.3.4.

In addition, once the word embedding *Vec* has been determined, and the distance *dist* has been selected for comparing "word meanings", there are a variety of semantic similarity measures that can be used to compare documents, for us bags-of-words. In this work we use the "Word Mover's Distance", which was shown to perform well across multiple text classification tasks [116].

The *Word Mover's Distance* is based on the classic *Earth Mover's Distance* [117] used in transportation problems with a given distance measure. We shall use the more general Earth Mover's definition with *dist* [5] as the underlying distance measure between words. We note that our results can be applied to problems outside of the text processing domain.

Let $X, Y \in \mathbb{B}\mathcal{S}$; we denote by X the tuple $\langle x_1^{a_1}, x_2^{a_2}, \ldots, x_k^{a_k} \rangle$, where a_i is the number of times that x_i occurs in X. Similarly we write $Y = \langle y_1^{b_1}, y_2^{b_2}, \ldots, y_l^{b_l} \rangle$; we have $\sum_i a_i = |X|$ and $\sum_j b_j = |Y|$,

[4] Recall that a pseudometric satisfies both the triangle inequality and symmetry; but different words could be mapped to the same vector and so $dist_{Vec}(w_1, w_2) = 0$ no longer implies that $w_1 = w_2$.

[5] In our experiments we take *dist* to be defined by the Euclidean distance.

the sizes of X and Y respectively. We define a *flow matrix* $F \in \mathbb{R}_{\geq 0}^{k \times l}$ where F_{ij} represents the (non-negative) amount of flow from $x_i \in X$ to $y_j \in Y$.

DEFINITION 9.2.3. (Earth Mover's Distance) Let $d_{\mathcal{S}}$ be a (pseudo)metric over $\mathcal{S}$. The Earth Mover's Distance with respect to $d_{\mathcal{S}}$, denoted by $E_{d_{\mathcal{S}}}$, is the solution to the following linear optimisation:

$$E_{d_{\mathcal{S}}}(X,Y) := \min \sum_{x_i \in X} \sum_{y_j \in Y} d_{\mathcal{S}}(x_i, y_j) F_{ij} \,, \quad \text{subject to:} \tag{9.1}$$

$$\sum_{i=1}^{k} F_{ij} = \frac{b_j}{|Y|} \quad \text{and} \quad \sum_{j=1}^{l} F_{ij} = \frac{a_i}{|X|} \,, \quad F_{ij} \geq 0, \quad 1 \leq i \leq k, 1 \leq j \leq l \tag{9.2}$$

where the minimum in (9.1) is over all possible flow matrices F subject to the constraints (9.2). In the special case that $|X| = |Y|$, the solution is known to satisfy the conditions of a (pseudo)metric [117] which we call the Earth Mover's Metric.

In this work we are interested in the special case $|X| = |Y|$, hence we will use the term *Earth Mover's metric* to refer to $E_{d_{\mathcal{S}}}$.

We end this section by describing how texts are prepared for machine learning tasks, and how Def. 9.2.3 is used to distinguish documents. Consider the text snippet "The President greets the press in Chicago". The first thing is to remove all "stopwords" – these are words which do not contribute to semantics, and include things like prepositions, pronouns and articles. The words remaining are those that contain a great deal of semantic and stylistic traits.[6]

In this case we obtain the bag:

$$b_1 := \langle \text{President}^1, \text{greets}^1, \text{press}^1, \text{Chicago}^1 \rangle \,.$$

Consider a second bag: $b_2 := \langle \text{Chief}^1, \text{speaks}^1, \text{media}^1, \text{Illinois}^1 \rangle$, corresponding to a different text. Figure 9.1 illustrates the optimal flow matrix which solves the optimisation problem in Def. 9.2.3 relative to $d_{\mathcal{S}}$. Here each word is mapped completely to another word, so that $F_{i,j} = 1/4$ when $i = j$ and 0 otherwise. We show later that this is always the case between bags of the same size. With these choices we can compute the distance between b_1, b_2:

$$\begin{aligned} E_{d_{\mathcal{S}}}(b_1, b_2) &= \frac{1}{4}(d_{\mathcal{S}}(\text{President}, \text{Chief}) + d_{\mathcal{S}}(\text{greets}, \text{speaks}) + \\ &\qquad d_{\mathcal{S}}(\text{press}, \text{media}) + d_{\mathcal{S}}(\text{Chicago}, \text{Illinois})) \\ &= 2.816 \,. \end{aligned} \tag{9.3}$$

[6] In fact the way that stopwords are used in texts turn out to be characteristic features of authorship. Here we follow standard practice in natural language processing to remove them for efficiency purposes and study the privacy of what remains. All of our results apply equally well had we left stopwords in place.

For comparison, consider the distance between b_1 and b_2 to a third document,

$$b_3 := \langle \text{Chef}^1, \text{breaks}^1, \text{cooking}^1, \text{record}^1 \rangle.$$

Using the same word embedding metric, [7] we find that $E_{d_S}(b_1, b_3) = 4.121$ and $E_{d_S}(b_2, b_3) = 3.941$. Thus b_1, b_2 would be classified as semantically "closer" to each other than to b_3, in line with our own (linguistic) interpretation of the original texts.

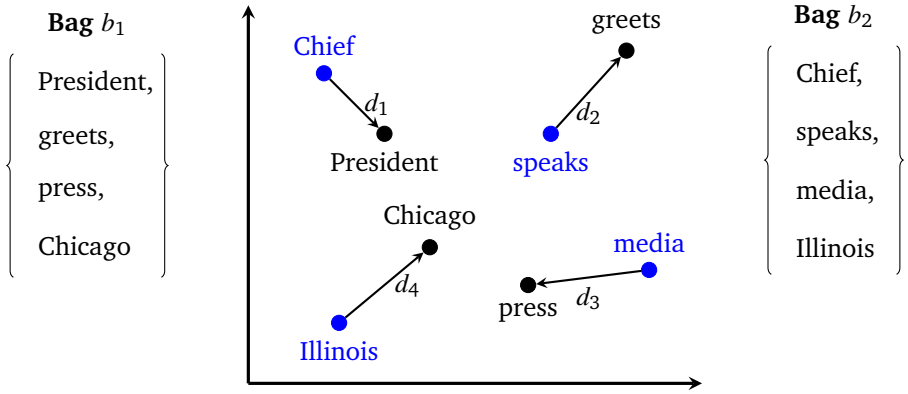


Figure 9.1: Earth Mover's metric between sample documents.

9.3 Privacy, Utility and the Earth Mover's Metric

In order to relate our privacy task with differential privacy, we need to explore what privacy in text processing means. Our aim is to release the topic-related contents of each document (in our case the bag-of-words) – this relates to *utility* because we would like to reveal the semantic content. The privacy relates to investing *individual documents* with indistinguishability, rather than *individual authors* directly.

What this means for privacy is the following. Suppose we are given two documents b_1, b_2 written by two distinct authors A_1, A_2, and suppose further that b_1, b_2 are changed through a privacy mechanism so that it is difficult or impossible to distinguish between them (by any means). Then it is also difficult or impossible to determine whether the authors of the original documents are A_1 or A_2, or some other author entirely. Thus our privacy goal is not associated to a particular metric of interest: we simply aim to ensure that there is some guarantee of indistinguishability of documents with respect to *different* authors. This is our goal for obfuscating authorship whilst preserving semantic content. Using differential privacy we can achieve this goal in practice by judicious choices of epsilon, which can be chosen for a dataset of interest ensuring every author within some pre-determined radius has some minimal level of indistinguishability. Here *radius* concerns the *utility* metric – which we now address.

[7] We use the same word2vec-based metric as per our experiments; this is described in §9.6.

9.3.1 Utility and Earth Moving

Our approach to obfuscating documents replaces words with other words, governed by probability distributions over possible replacements. Thus the type of our mechanism is $\mathbb{B}\mathcal{S} \to \mathbb{D}(\mathbb{B}\mathcal{S})$, where (recall) $\mathbb{D}(\mathbb{B}\mathcal{S})$ is the set of probability distributions over the set of (finite) bags of $\mathcal{S}$. Since we are aiming to find a careful trade-off between utility and privacy, our objective is to ensure that there is a high probability of outputting a *useful* document – that is, one on a similar topic to that of the input document. Our privacy mechanism then should be designed in such a way that utility (document topic) is preserved, even though the particulars (words) are obfuscated so that the original document could have been one of several by different authors. As explained in §9.2, topic similarity of documents is determined by the Earth Mover's distance relative to a given (pseudo)metric on word embeddings, and so our privacy definition must also be relative to the Earth Mover's distance. Recalling the definition of **d**-privacy from Chapter 3 (Def. 3.1.2), we make the following definition, which is an instance of **d**-privacy designed for the Earth Mover's metric $E_{d_\mathcal{X}}$.

> **DEFINITION 9.3.1.** (Earth Mover's Privacy) Let $\mathcal{X}$ be a set, and $d_\mathcal{X}$ be a (pseudo)metric on $\mathcal{X}$ and let $E_{d_\mathcal{X}}$ be the Earth Mover's metric on $\mathbb{B}\mathcal{X}$ relative to $d_\mathcal{X}$. Given $\varepsilon \geq 0$, a mechanism $K : \mathbb{B}\mathcal{X} \to \mathbb{D}(\mathbb{B}\mathcal{X})$ satisfies $\varepsilon E_{d_\mathcal{X}}$-privacy iff for any $b, b' \in \mathbb{B}\mathcal{X}$ and $Z \subseteq \mathbb{B}\mathcal{X}$:
>
> $$K(b)(Z) \quad \leq \quad e^{\varepsilon E_{d_\mathcal{X}}(b,b')} K(b')(Z)\ . \tag{9.4}$$

Def. 9.3.1 tells us that when two documents are measured to be very close, so that $\varepsilon E_{d_\mathcal{X}}(b, b')$ is close to 0, then the multiplier $e^{\varepsilon E_{d_\mathcal{X}}(b,b')}$ is approximately 1 and the outputs $K(b)$ and $K(b')$ are almost identical. On the other hand the more that the input bags can be distinguished by $E_{d_\mathcal{X}}$, the more their outputs are likely to differ. This flexibility is what allows us to strike a balance between utility and privacy; we discuss this issue further in §9.5 below.

9.3.2 Implementing Earth Mover's Privacy

Our next task is to show how to implement a mechanism that can be proved to satisfy Def. 9.3.1. We follow the basic construction of Dwork et al. [5] for lifting a differentially private mechanism $K : \mathcal{X} \to \mathbb{D}\mathcal{X}$ to a differentially private mechanism $\underline{K}^\star : \mathcal{X}^N \to \mathbb{D}\mathcal{X}^N$ on *vectors* in $\mathcal{X}^N$. (Note that, unlike a bag, a vector imposes a fixed order on its components.) Here the idea is to apply K independently to each component of a vector $v \in \mathcal{X}^N$ to produce a random output vector, also in $\mathcal{X}^N$. In particular the probability of outputting some vector v' is the product:

$$\underline{K}^\star(v)\{v'\} \quad = \quad \prod_{1 \leq i \leq N} K(v_i)\{v'_i\}\ . \tag{9.5}$$

Thanks to the compositional properties of differential privacy when the underlying metric on $\mathcal{X}$ satisfies the triangle inequality, it's possible to show that the resulting mechanism $\underline{K}^\star$ satisfies the following privacy inequation [7]:

$$\underline{K}^\star(v)(Z) \quad \leq \quad e^{M_{d_\mathcal{X}}(v,v')}\underline{K}^\star(v')(Z)\,, \tag{9.6}$$

where $M_{d_\mathcal{X}}(v,v') := \sum_{1\leq i\leq N} d_\mathcal{X}(v_i, v'_i)$, the Manhattan metric relative to $d_\mathcal{X}$.

However Def. 9.3.1 does not follow from Eqn. (9.6), since Def. 9.3.1 operates on bags of size N, and the Manhattan distance between any vector representation of bags is *greater* than $N\times E_{d_\mathcal{X}}$. Remarkably however, it turns out that $K^\star$ –the mechanism that applies K independently to each item in a given bag– in fact satisfies the much stronger Def. 9.3.1, as the following theorem shows, provided the input bags have the same size as each other.

THEOREM 9.2. Let $d_\mathcal{X}$ be a pseudo-metric on $\mathcal{X}$ and let $K : \mathcal{X} \to \mathbb{D}\mathcal{X}$ be a mechanism satisfying $\varepsilon d_\mathcal{X}$-privacy, i.e.

$$K(x)(Z) \quad \leq \quad e^{\varepsilon d_\mathcal{X}(x,x')}K(x')(Z)\,, \textit{ for all } x, x' \in \mathcal{X},\ Z \subseteq \mathcal{X}. \tag{9.7}$$

Let $K^\star : \mathbb{B}\mathcal{X} \to \mathbb{D}(\mathbb{B}\mathcal{X})$ be the mechanism obtained by applying K independently to each element of X for any $X \in \mathbb{B}\mathcal{X}$. Denote by $K^\star\!\downarrow N$ the restriction of $K^\star$ to bags of fixed size N. Then $K^\star\!\downarrow N$ satisfies $\varepsilon N E_{d_\mathcal{X}}$-privacy.

Proof. (Sketch)

Let b, b' be input bags, both of size N, and let c a possible output bag (of $K^\star$). Observe that both output bags determined by $K^\star(b_1), K^\star(b_2)$ and c also have size N. We shall show that (9.4) is satisfied for the set containing the singleton element c and multiplier εN, from which it follows that (9.4) is satisfied for all sets Z.

By Birkhoff-von Neumann's theorem ([118]), in the case where all bags have the same size, the minimisation problem in Def. 9.2.3 is optimised for transportation matrix F where all values F_{ij} are either 0 or $1/N$. This implies that the optimal transportation for $E_{d_\mathcal{X}}(b,c)$ is achieved by moving each word in the bag b to a (single) word in bag c. The same is true for $E_{d_\mathcal{X}}(b',c)$ and $E_{d_\mathcal{X}}(b,b')$. Next we use a vector representation of bags as follows. For bag b, we write $\vec{b}$ for a vector in $\mathcal{X}^N$ such that each element in b appears at some $\vec{b}_i$.

Next we fix $\vec{b}$ and $\vec{b'}$ to be vector representations of respectively b, b' in $\mathcal{X}^N$ such that the optimal transportation for $E_{d_\mathcal{X}}(b,b')$ is

$$E_{d_\mathcal{X}}(b,b') \quad = \quad 1/N\times \sum_{1\leq i\leq N} d_\mathcal{X}(\vec{b}_i, \vec{b'}_i) \quad = \quad M_{d_\mathcal{X}}(\vec{b},\vec{b'})/N\,. \tag{9.8}$$

The final fact we need is to note that there is a relationship between $K^\star$ acting on bags of size N and $\underline{K}^\star$ which acts on vectors in $\mathcal{X}^N$ by applying K independently to each component of a vector: it is characterised in the following way. Let b, c be bags and let $\vec{b}, \vec{c}$ be any vector representations. For permutation $\sigma \in \{1\ldots N\} \to \{1\ldots N\}$ write $\vec{c}^\sigma$ to be the vector with components permuted by σ, so that $\vec{c}^\sigma_i = \vec{c}_{\sigma(i)}$. With these definitions, the following equality between

probabilities holds:

$$K^\star(b)\{c\} \quad = \quad \sum_\sigma \underline{K}^\star(\vec{b})\{\vec{c}^\sigma\}\,, \tag{9.9}$$

where the summation is over all permutations that give distinct vector representations of c. We now compute directly:

$$\begin{array}{lll} & K^\star(b)\{c\} & \\ = & \sum_\sigma \underline{K}^\star(\vec{b})\{\vec{c}^\sigma\} & \text{“(9.9) for } b,c\text{”} \\ \leq & \sum_\sigma e^{\varepsilon M_d(\vec{b},\vec{b}')}\underline{K}^\star(\vec{b}')\{\vec{c}^\sigma\} & \text{“(9.6) for } \vec{b},\vec{b}',\vec{c}\text{”} \\ = & e^{\varepsilon N E_d(\vec{b},\vec{b}')}\sum_\sigma \underline{K}^\star(\vec{b}')\{\vec{c}^\sigma\} & \text{“Arithmetic and (9.8)”} \\ = & e^{\varepsilon N E_d(\vec{b},\vec{b}')}K^\star(b')\{c\}\,, & \text{“(9.9) for } b',c\text{”} \end{array}$$

as required.

□

9.3.3 Application to Text Documents

Recall the bag-of-words

$$b_2 \quad := \quad \langle \text{Chief}^1, \text{speaks}^1, \text{media}^1, \text{Illinois}^1 \rangle\,,$$

and assume we are provided with a mechanism K satisfying the standard $\varepsilon d_\mathcal{X}$-privacy property (9.7) for individual words. As in Thm. 9.2 we can create a mechanism K^* by applying K independently to each word in the bag, so that, for example the probability of outputting $b_3 = \langle \text{Chef}^1, \text{breaks}^1, \text{cooking}^1, \text{record}^1 \rangle$ is determined by (9.9):

$$K^\star(b_2)(\{b_3\}) \quad = \quad \sum_\sigma \prod_{1\leq i\leq 4} K(b_{2i})\{\vec{b_3}_i^{\,\sigma}\}\,.$$

By Thm. 9.2, $K^\star$ satisfies $4\varepsilon E_{d_\mathcal{S}}$-privacy. Recalling (9.3) that $E_{d_\mathcal{S}}(b_1, b_2) = 2.816$, we deduce that if $\varepsilon \sim 1/16$ then the output distributions $K^\star(b_1)$ and $K^\star(b_2)$ would differ by the multiplier $e^{2.816\times 4/16} \sim 2.02$; but if $\varepsilon \sim 1/32$ those distributions differ by only 1.42. In the latter case it means that the outputs of $K^\star$ on b_1 and b_2 are almost indistinguishable.

The parameter ε depends on the randomness implemented in the basic mechanism K; we investigate that further in §9.4.

9.3.4 Properties of Earth Mover's Privacy

In machine learning a number of "distance measures" are used in classification or clustering tasks, and in this section we explore some properties of privacy when we vary the underlying metrics of an Earth Mover's metric used to classify complex objects.

Let $v, v' \in \mathbb{R}^n$ be real-valued n-dimensional vectors. We use the following (well-known) metrics. Recall in our applications we have looked at bags-of-words, where the words themselves

are represented as n-dimensional vectors. [8]

1. Euclidean: $\|v-v'\| := \sqrt{\sum_{1\le i\le n}(v_i - v'_i)^2}$

2. Manhattan: $\lfloor v-v' \rfloor := \sum_{1\le i\le n} |v_i - v'_i|$

Note that the Euclidean and Manhattan distances determine pseudometrics on words as defined at Def. 9.2.2 and proved at Lem. 9.1.

LEMMA 9.3. If $d_{\mathcal{X}} \le d_{\mathcal{X}'}$ (point-wise), then $E_{d_{\mathcal{X}}} \le E_{d_{\mathcal{X}'}}$ (point-wise).

Proof. Trivial, by contradiction. If $d_{\mathcal{X}} \le d_{\mathcal{X}'}$ and $F_{ij}, F^{\star}_{ij}$ are the minimal flow matrices for $E_{d_{\mathcal{X}}}, E_{d_{\mathcal{X}'}}$ respectively, then $F^{\star}_{ij}$ is a (strictly smaller) minimal solution for $E_{d_{\mathcal{X}}}$ which contradicts the minimality of F_{ij}. □

COROLLARY 9.4. If $d_{\mathcal{X}} \le d_{\mathcal{X}'}$ (point-wise), then $E_{d_{\mathcal{X}}}$-privacy implies $E_{d_{\mathcal{X}'}}$-privacy.

This shows that, for example, $E_{\|\cdot\|}$-privacy implies $E_{\lfloor\cdot\rfloor}$-privacy, and indeed any distance measure d which exceeds the Euclidean distance then $E_{\|\cdot\|}$-privacy implies E_d-privacy.

We end this section by noting that Def. 9.3.1 satisfies *post-processing*; i.e. that privacy does not decrease under post processing. We write $K;K'$ for the composition of mechanisms $K, K' : \mathbb{B}\mathcal{X} \to \mathbb{D}(\mathbb{B}\mathcal{X})$, defined:

$$(K;K')(b)(Z) := \sum_{b' : \mathbb{B}\mathcal{X}} K(b)(\{b'\}) \times K'(b')(Z) \,. \tag{9.10}$$

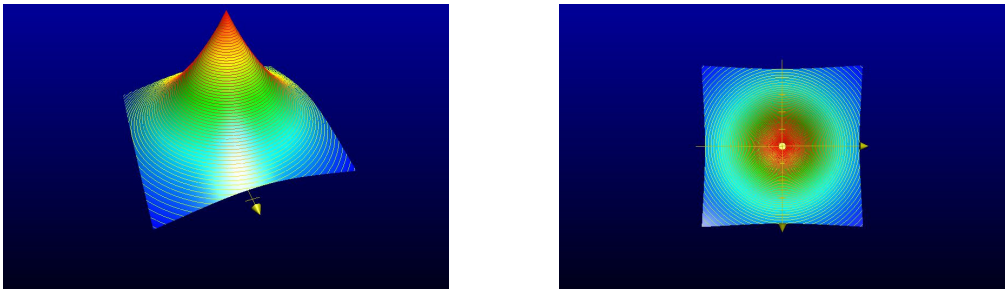
LEMMA 9.5. [Post processing] If $K, K' : \mathbb{B}\mathcal{X} \to \mathbb{D}(\mathbb{B}\mathcal{X})$ and K is $\varepsilon E_{d_{\mathcal{X}}}$-private for (pseudo)metric d on $\mathcal{X}$ then $K;K'$ is $\varepsilon E_{d_{\mathcal{X}}}$-private.

9.4 Earth Mover's Privacy for bags of vectors in $\mathbb{R}^n$

In Thm. 9.2 we have shown how to promote a privacy mechanism on components to $E_{d_{\mathcal{X}}}$-privacy on a bag of those components. In this section we show how to implement a privacy mechanism satisfying (9.7), when the components are represented by high dimensional vectors in $\mathbb{R}^n$ and the underlying metric is taken Euclidean on $\mathbb{R}^n$, which we denote by $\|\cdot\|$.

We begin by summarising the basic probabilistic tools we need. A *probability density function* (PDF) over some domain $\mathcal{D}$ is a function $\phi : \mathcal{D} \to [0,1]$ whose value $\phi(z)$ gives the "relative likelihood" of z. The probability density function is used to compute the probability of an outcome

[8] As we shall see, in the machine learning analysis *documents* are represented as bags of n-dimensional vectors (word embeddings), where each bag contains N such vectors.



3D plot Contour diagram

Figure 9.2: Laplace density function Lap^2_ε in $\mathbb{R}^2$

"$z \in A$", for some region $A \subseteq \mathcal{D}$ as follows:

$$\int_A \phi(x)\, dx\ . \tag{9.11}$$

In differential privacy, a popular density function used for implementing mechanisms is the *Laplacian*, defined next.

> **Definition 9.4.1.** Let $n \geq 0$ be an integer $\varepsilon > 0$ be a real, and $v \in \mathbb{R}^n$. We define the Laplacian probability density function in n-dimensions:
>
> $$Lap^n_\varepsilon(v) \quad := \quad c^\varepsilon_n \times e^{-\varepsilon \|v\|}\ ,$$
>
> where $\|v\| = \sqrt{(v_1^2 + \cdots + v_n^2)}$, and c^ε_n is a real-valued constant satisfying the integral equation $1 = \int \cdots \int_{\mathbb{R}^n} Lap^n_\varepsilon(v) dv_1 \ldots dv_n$.

When $n = 1$, we can compute $c^\varepsilon_1 = \varepsilon/2$, and when $n = 2$, we have that $c^\varepsilon_2 = \varepsilon^2/2\pi$.

In privacy mechanisms, probability density functions are used to produce a "noisy" version of the released data. The benefit of the Laplace distribution is that, besides creating randomness, the likelihood that the released value is different from the true value decreases exponentially. This implies that the utility of the data release is high, whilst at the same time masking its actual value. In Figure 9.2 the probability density function $Lap^2_\varepsilon(v)$ depicts this situation, where we see that the highest relative likelihood of a randomly selected point on the plane being close to the origin, with the chance of choosing more distant points diminishing rapidly. Once we are able to select a vector v' in $\mathbb{R}^n$ according to Lap^n_ε, we can "add noise" to any given vector v as $v+v'$, so that the true value v is highly likely to be perturbed only a small amount.

In order to use the Laplacian in Def. 9.4.1, we need to implement it. Andrés et al. [14] exhibited a mechanism for $Lap^2_\varepsilon(v)$, and here we show how to extend that idea to the general case. (Note: it turns out that this extension was discovered earlier and expressed more generally as the K-norm mechanism [69]; our result was discovered independently and represents a particular instantiation of the K-norm mechanism.) The main idea of the construction for $Lap^2_\varepsilon(v)$ uses the fact that any vector on the plane can be represented by spherical coordinates (r, θ), so that the probability of selecting a vector distance no more than r from the origin can be

achieved by selecting r and θ independently. In order to obtain a distribution which overall is equivalent to $Lap^2_\varepsilon(v)$, Andrés et al. computed that r must be selected according to a well-known distribution called the "Lambert W" function, and θ is selected uniformly over the unit circle. In our generalisation to $Lap^n_\varepsilon(v)$, we observe that the same idea is valid [119]. Observe first that every vector in $\mathbb{R}^n$ can be expressed as a pair (r,p), where r is the distance from the origin, and p is a point in B^n, the unit *hypersphere* in $\mathbb{R}^n$. Now selecting vectors according to $Lap^n_\varepsilon(v)$ can be achieved by independently selecting r and p, but this time r must be selected according to the *Gamma distribution*, and p must be selected uniformly over B^n. We set out the details next.

Definition 9.4.2. The *Gamma distribution* of (integer) shape n and scale $\delta > 0$ is determined by the probability density function:

$$Gam^n_\delta(r) \quad := \quad \frac{r^{n-1}e^{-r/\delta}}{\delta^n(n-1!)} \,. \tag{9.12}$$

Definition 9.4.3. The uniform distribution over the surface of the unit hypersphere B^n is determined by the probability density function:

$$Uniform^n(v) \quad := \quad \frac{\Gamma(\frac{n}{2})}{n\pi^{n/2}} \; \textit{if} \;\; v \in B^n \;\; \textit{else} \;\; 0 \,, \tag{9.13}$$

where $B^n := \{v \in \mathbb{R}^n \mid \|v\| = 1\}$, and $\Gamma(\alpha) := \int_0^\infty x^{\alpha-1}e^{-x}\, dx$ is the "Gamma function".

With Def. 9.4.2 and Def. 9.4.3 we are able to provide an implementation of a mechanism which produces noisy vectors around a given vector in $\mathbb{R}^n$ according to the Laplacian distribution in Def. 9.4.1. The first task is to show that our decomposition of Lap^n_ε is correct.

Lemma 9.6. The n-dimensional Laplacian $Lap^n_\varepsilon(v)$ can be realised by selecting vectors represented as (r,p), where r is selected according to $Gam^n_{1/\varepsilon}(r)$ and p is selected independently according to $Uniform^n(p)$.

Proof. (Sketch) The proof follows by changing variables to spherical coordinates and then showing that $\int_A Lap^n_\varepsilon(v)\, dv$ can be expressed as the product of independent selections of r and p.

We use a spherical-coordinate representation of v as:

$$r := \|v\| \,, \; \textit{and}$$
$$v_1 := r\cos\theta_1 \,, \; v_2 := r\sin\theta_1\cos\theta_2 \,, \ldots v_n := r\sin\theta_1\sin\theta_2 \ldots \,, \sin\theta_{n-2}\sin\theta_{n-1} \,.$$

Next we assume for simplicity that A is a hypersphere of radius R; with that we can reason:

$$\begin{array}{ll} & \int_A Lap^n_\varepsilon(v)\, dv \\ = & \qquad\qquad\qquad\qquad\qquad\qquad \text{"Def. 9.4.1; } A \text{ is a hypersphere"} \\ & \int_{\|v\|\le R} c^\varepsilon_n \times e^{-\varepsilon|v|}\, dv \end{array}$$

$$
\begin{aligned}
&= && \text{“}\|v\| = \sqrt{v_1^2 + \cdots + v_n^2}\text{”} \\
&\quad \int_{\|v\| \le R} c_n^\varepsilon \times e^{-\varepsilon\sqrt{v_1^2+\cdots+v_n^2}}\, dv \\
&= && \text{“Change of variables to spherical coordinates; see below (9.14)”} \\
&\quad \int_{r \le R} \int_{A_\theta} c_n^\varepsilon \times e^{-\varepsilon r} \tfrac{\partial(z_1,z_2,\ldots,z_n)}{\partial(r,\theta_1,\ldots,\theta_{n-1})}\, dr d\theta_1 \ldots d\theta_{n-1} \\
&= && \text{“See below (9.14)”} \\
&\quad \int_{r \le R} \int_{A_\theta} c_n^\varepsilon \times e^{-\varepsilon r} r^{n-1} \sin^{n-2}\theta_1 \sin^{n-3}\theta_2 \ldots \sin^2\theta_{n-3} \sin\theta_{n-2}\, dr d\theta_1 \ldots d\theta_{n-1}\,.
\end{aligned}
$$

Now rearranging we can see that this becomes a product of two integrals. The first $\int_{r \le R} e^{-\varepsilon r} r^{n-1}$ is over the radius, and is proportional to the integral of the Gamma distribution Def. 9.4.2; and the second is an integral over the angular coordinates and is proportional to the surface of the unit hypersphere, and corresponds to the PDF at (9.4.3). Finally, for the “see below's” we are using the “Jacobian”:

$$\frac{\partial(z_1, z_2, \ldots, z_n)}{\partial(r, \theta_1, \ldots, \theta_{n-1})} = r^{n-1} \sin^{n-2}\theta_1 \sin^{n-3}\theta_2 \ldots \tag{9.14}$$

□

We can now assemble the facts to demonstrate the n-Dimensional Laplacian.

THEOREM 9.7 (n-Dimensional Laplacian)**.** Given $\varepsilon > 0$ and $n \in \mathbb{Z}^+$, let $K : \mathbb{R}^n \to \mathbb{DR}^n$ be a mechanism that, given a vector $x \in \mathbb{R}^n$ outputs a noisy value as follows:

$$x \overset{K}{\longmapsto} x + x'$$

where x' is represented as (r, p) with $r \ge 0$, distributed according to $Gam^n_{1/\varepsilon}(r)$ and $p \in B^n$ distributed according to $Uniform^n(p)$. Then K satisfies (9.7) from Thm. 9.2, i.e. K satisfies $\varepsilon\|\cdot\|$-privacy where $\|\cdot\|$ is the Euclidean metric on $\mathbb{R}^n$.

Proof. (Sketch) Let $z, y \in \mathbb{R}^n$. We need to show that for any (measurable) set $A \subseteq \mathbb{R}^n$ that:

$$K(z)(A)/K(y)(A) \quad \le \quad e^{\varepsilon\|z-y\|}\,. \tag{9.15}$$

However (9.15) follows provided that the probability densities of respectively $K(z)$ and $K(y)$ satisfy it. By Lem. 9.6 the probability density of $K(z)$, as a function of x is distributed as $Lap^n_\varepsilon(z-x)$; and similarly for the probability density of $K(y)$. Hence we reason:

$$
\begin{aligned}
& Lap^n_\varepsilon(z-x)/Lap^n_\varepsilon(y-x) \\
= \quad & c_n^\varepsilon \times e^{-\varepsilon\|z-x\|} / c_n^\varepsilon \times e^{-\varepsilon\|y-x\|} && \text{“Def. 9.4.1”} \\
= \quad & e^{-\varepsilon\|z-x\|} \times e^{\varepsilon\|y-x\|} && \text{“Arithmetic”} \\
\le \quad & e^{\varepsilon\|z-y\|}\,, && \text{“Triangle inequality; } s \mapsto e^s \text{ is monotone”}
\end{aligned}
$$

as required. □

Thm. 9.7 reduces the problem of adding Laplace noise to vectors in $\mathbb{R}^n$ to selecting a real value according to the Gamma distribution and an independent uniform selection of a unit vector. Several methods have been proposed for generating random variables according to the Gamma distribution [120] as well as for the uniform selection of vectors on the unit n-sphere [121]. The latter can be achieved by selecting n random variables from the standard normal distribution to produce vector $v \in \mathbb{R}^n$, and then normalising to output $\frac{v}{|v|}$.

9.4.1 Earth Mover's Privacy in $\mathbb{BR}^n$

Using the n-dimensional Laplacian, we can now implement an algorithm for $\varepsilon NE_{\|\cdot\|}$-privacy. Algorithm 1 takes a bag of n-dimensional vectors as input and applies the n-dimensional Laplacian mechanism described in Thm. 9.7 to each vector in the bag, producing a noisy bag of n-dimensional vectors as output. Cor. 9.8 summarises the privacy guarantee.

Algorithm 1 Earth Mover's Privacy Mechanism

Require: vector v, dimension n, epsilon ε
1: **procedure** GENERATENOISYVECTOR(v, n, ε)
2: $r \leftarrow Gamma(n, \frac{1}{\varepsilon})$
3: $u \leftarrow \mathcal{U}(n)$
4: **return** $v + ru$
5: **end procedure**

Require: bag X, dimension n, epsilon ε
1: **procedure** GENERATEPRIVATEBAG(X, n, ε)
2: $Z \leftarrow ()$
3: **for all** $x \in X$ **do**
4: $z \leftarrow$ GENERATENOISYVECTOR(x, n, ε)
5: add z to Z
6: **end for**
7: return Z
8: **end procedure**

COROLLARY 9.8. Algorithm 1 satisfies $\varepsilon NE_{\|\cdot\|}$-privacy, relative to any two bags in $\mathbb{BR}^n$ of size N.

Proof. Follows from Thm. 9.2 and Thm. 9.7. □

9.4.2 Utility Bounds

We prove a lower bound on the utility for this algorithm, which applies for high dimensional data representations. Given an output element x, we define Z to be the set of outputs within distance $\Delta > 0$ from x. Recall that the distance function is a measure of utility, therefore $Z = \{z \mid E_{\|\cdot\|}(x, z) \leq \Delta\}$ represents the set of vectors within utility Δ of x. Then we have the following:

THEOREM 9.9. Given an input bag b consisting of N n-dimensional vectors, the mechanism defined by Algorithm 1 outputs an element from $Z = \{z \mid E_{\|\cdot\|}(b,z) \leq \Delta\}$ with probability at least

$$1 - e^{-\varepsilon N\Delta} e_{n-1}(\varepsilon N\Delta)\,,$$

whenever $\varepsilon N\Delta \leq n/e$. (Recall that $e_k(\alpha) = \sum_{0\leq i\leq k} \frac{\alpha^i}{i!}$, the sum of the first $k+1$ terms in the series for e^α .)

Proof. (Sketch) Let $\vec{b} \in (\mathbb{R}^n)^N$ be a (fixed) vector representation of the bag b. For $v \in (\mathbb{R}^n)^N$, let $v^\circ \in \mathbb{BR}^n$ be the bag comprising the N components if v. Observe that $NE_{\|\cdot\|}(b,v^\circ) \leq M_{\|\cdot\|}(\vec{b},v)$, and so

$$Z_M = \{v \mid M_{\|\cdot\|}(\vec{b},v) \leq N\Delta\} \quad \subseteq \quad \{v \mid E_{\|\cdot\|}(b,v^\circ) \leq \Delta\} = Z_E\,. \tag{9.16}$$

Thus the probability of outputting an element of Z is the same as the probability of outputting Z_E, and by (9.16) that is at least the probability of outputting an element from Z_M by applying a standard n-dimensional Laplace mechanism to each of the components of $\vec{b}$. We can now compute:

$$\begin{array}{ll} & \textit{Probability of outputting an element in } Z_E \\ \geq & \text{“(9.16)”} \\ & \int\cdots\int_{v\in Z_M} \prod_{1\leq i\leq N} Lap^n_\varepsilon(\vec{b}_i - v_i) dv_1 \ldots dv_N \\ = & \text{“Lem. 9.6”} \\ & \int\cdots\int_{v\in Z_M} \prod_{1\leq i\leq N} c^\varepsilon_n e^{-\varepsilon\|\vec{b}_i - v_i\|} dv_1 \ldots dv_N\,. \end{array}$$

The result follows by completing the multiple integrals and applying some approximations, whilst observing that the variables in the integration are n-dimensional vector valued. □

We note that our application word embeddings are typically mapped to vectors in $\mathbb{R}^{300}$, thus we would use $n \sim 300$ in Thm. 9.9.

9.5 Text Document Privacy

In this section we bring everything together, and present a privacy mechanism for text documents; we explore how it contributes to the author obfuscation task described above. Algorithm 2 describes the complete procedure for taking a document as a bag-of-words, and outputting a “noisy” bag-of-words. Depending on the setting of parameter ε, the output bag will be likely to be classified to be on a similar topic as the input.

Algorithm 2 uses a function *Vec* to turn the input document into a bag of word embeddings; next Algorithm 1 produces a noisy bag of word embeddings, and, in a final step the inverse Vec^{-1} is used to reconstruct an actual bag-of-words as output. In our implementation of Algorithm 2,

Algorithm 2 Document privacy mechanism

Require: Bag-of-words b, dimension n, epsilon ε, Word embedding $Vec : \mathcal{S} \to \mathbb{R}^n$

1: **procedure** GENERATENOISYBAGOFWORDS(b, n, ε, Vec)
2: $X \leftarrow Vec^\star(b)$
3: $Z \leftarrow$ GENERATEPRIVATEBAG(X, n, ε)
4: **return** $(Vec^{-1})^\star(Z)$
5: **end procedure**

Note that $Vec^\star : \mathbb{B}\mathcal{S} \to \mathbb{B}\mathbb{R}^n$ applies Vec to each word in a bag b, and $(Vec^{-1})^\star : \mathbb{B}\mathbb{R}^n \to \mathbb{B}\mathcal{S}$ reverses this procedure as a post-processing step; this involves determining the word w that minimises the Euclidean distance $\|z - Vec(w)\|$ for each z in Z.

described below, we compute $Vec^{-1}(x)$ to be the word w that minimises the Euclidean distance $\|z - Vec(w)\|$. The next result summarises the privacy guarantee for Algorithm 2.

> **THEOREM 9.10.** Algorithm 2 satisfies $\varepsilon N E_{d_\mathcal{S}}$-privacy, where $d_\mathcal{S} = dist_{Vec}$. That is to say: given input documents (bags) b, b' both of size N, and c a possible output bag, define the following quantities as follows: $k := E_{\|\cdot\|}(Vec^\star(b), Vec^\star(b'))$, $pr(b, c)$ and $pr(b', c)$ are the respective probabilities that c is output given the input was b or b'. Then:
>
> $$pr(b, c) \quad \leq \quad e^{\varepsilon N k} \times pr(b', c) \ .$$
>
> *Proof.* The result follows by appeal to Thm. 9.7 for privacy on the word embeddings; the step to apply Vec^{-1} to each vector is a post-processing step which by Lem. 9.5 preserves the privacy guarantee. □

9.5.1 Privacy as Indistinguishability

Although Thm. 9.10 utilises ideas from differential privacy, an interesting question to ask is how it contributes to the PAN@Clef author obfuscation task, which recall asked for mechanisms that preserve content but mask features that distinguish authorship. Algorithm 2 does indeed attempt to preserve content (to the extent that the topic can still be determined) but it does not directly "remove stylistic features". So has it, in fact, disguised the author's characteristic style? To answer that question, we review Thm. 9.10 and interpret what it tells us in relation to author obfuscation. The theorem implies that it is indeed possible to make the (probabilistic) output from two distinct documents b, b' almost indistinguishable by choosing ε to be extremely small in comparison with $N \times E_{\|\cdot\|}(Vec^\star(b), Vec^\star(b'))$. However, if $E_{\|\cdot\|}(Vec^\star(b), Vec^\star(b'))$ is very large – meaning that b and b' are on entirely different topics, then ε would need to be so tiny that the noisy output document would be highly unlikely to be on a topic remotely close to either b or b' (recall Lem. 9.9).

This observation is actually highlighting the fact that, in some circumstances, the topic itself is actually a feature that characterises author identity. (First-hand accounts of breaking the world record for highest and longest free fall jump would immediately narrow the field down to the title holder.) This means that *any* obfuscating mechanism would, as for Algorithm 2, only be

able to obfuscate documents so as to disguise the author's identity if there are several authors who write on similar topics. And it is in that spirit, that we have made the first step towards a satisfactory obfuscating mechanism: provided that documents are similar in topic (ie. are close when their embeddings are measured by $E_{\|\cdot\|}$) they can be obfuscated so that it is unlikely that the content is disturbed, but that the contributing authors cannot be determined easily.

We can see the importance of the "indistinguishability" property wrt the PAN obfuscation task. In stylometry analysis the representation of words for eg. author classification is completely different to the word embeddings which have used for topic classification. State-of-the-art author attribution algorithms represent words as "character n-grams" [107] which have been found to capture stylistic clues such as systematic spelling errors. A *character 3-gram* for example represents a given word as the complete list of substrings of length 3. For example character 3-gram representations of "color" and "colour" are:

- "color" $\mapsto$ ⟦ "col", "olo", "lor" ⟧
- "colour" $\mapsto$ ⟦ "col", "olo", "lou", "our" ⟧

For author identification, any output from Algorithm 2 would then need to be further transformed to a bag of character n-grams, as a post processing step; by Lem. 9.5 this additional transformation preserves the privacy properties of Algorithm 2. We explore this experimentally in §9.6.

9.5.2 Privacy as Protection from Inferences

We may wonder how well our indistinguishability property of documents protects against, say, machine learning attacks on authorship. In this case we can use QIF to model the likely success of a machine learner to infer the authorship of a document released by a noise-adding mechanism.

Recalling the usual QIF notions from Chapter 2, we can use the channel model to describe a probabilistic mechanism $M : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ taking secret values in $\mathcal{X}$ to distributions over outputs $\mathcal{Y}$. We recall that adversaries are modelled using a prior $\pi : \mathbb{D}\mathcal{X}$ over secret values along with a gain function $g : \mathcal{W} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ over secrets $\mathcal{X}$ and *guesses* $\mathcal{W}$. We use the identity gain function bv[9] defined

$$bv(w, x) \quad := \quad \begin{cases} 1 & \text{if } w = x, \\ 0 & \text{otherwise,} \end{cases} \tag{9.17}$$

to model an adversary whose goal is to guess the secret exactly in one try. Recall (Chapter 2, Example 2.2) that we can extend this to the case of an attacker who tries to guess a property of the secret, rather than the entire secret. For example, letting $\sim$ be an equivalence class over secrets, the guesses $\mathcal{W}$ now correspond to equivalence classes and we can define

$$bv_{\sim}(w, x) \quad := \quad \begin{cases} 1 & \text{if } x \in w, \\ 0 & \text{otherwise.} \end{cases} \tag{9.18}$$

[9] Recall also Chapter 2, Def. 2.2.3.

We can now use the familiar notions of prior and posterior g-vulnerability (recall §2.2.1 and §2.2.2) to compute how much an adversary learns as the result of the release of an output from a mechanism. In this chapter we will use the *multiplicative g-leakage*, defined by

$$\mathcal{L}_g^{\times}(\pi, M) \quad := \quad \frac{V_g[\pi \triangleright M]}{V_g(\pi)}\,, \tag{9.19}$$

which gives the relative increase in gain.

We can now use this leakage measure to provide an important robust approximation with regards to machine learning adversaries. Given a noise-adding mechanism M modelled as a channel, for any prior π the following leakage bound holds:[10]

$$\begin{array}{ll} & \textit{Probability of correctly guessing the secret after applying } M \\ \leq & V_{bv}[\pi \triangleright M] \\ \leq & (\textit{Sum of the column maxima of } M) \;\times\; V_{bv}(\pi) \end{array}$$

What this means is that even if the attacker uses machine learning to try to deduce properties about the original data, its ability to do so is constrained by this upper bound. Now, if M is an $\varepsilon{\cdot}\mathbf{d}$-private mechanism this gives the following bound on information leakage, even if we do not know M exactly:

> **THEOREM 9.11.** Let M be an $\varepsilon{\cdot}\mathbf{d}$-private mechanism. Then for any gain function g and prior π,
>
> $$\mathcal{L}_g^{\times}(\pi, M) \quad \leq \quad e^{\varepsilon \times d^{\star}}\,,$$
>
> where $d^{\star} := \max_{x,x' \in \mathcal{X}} \mathbf{d}(x, x')$.

Thus if ε is very small, the indistinguishability property of $\mathbf{d}$-privacy tells us that the leakage of information from the channel will likewise be small.

Privacy versus Utility

Information leakage on its own, in the case that it is large, implies that the probability of determining some property of the system will be high; if the upper bound is small, then it implies the mechanism does not leak very much information about anything. When we bring utility into the mix what we want is that the mechanism leaks a lot of information about a property which is not deemed sensitive, but keeps secret some other property that is deemed private. This is the privacy-utility trade-off which we wish to manage effectively.

We can use some notions from QIF to understand how this trade-off occurs with our author and topic classes. Let $\sim_A$ and $\sim_T$ represent two equivalence classes on a set of (secret) data S. We will think of these as the equivalence classes of authors and topics respectively; thus we want to release the equivalence class $\sim_T$ but keep $\sim_A$ private using our mechanism M. We can determine how successful we are by measuring the leakage with respect to the two equivalence classes,

[10] This is a result of the Miracle Theorem of [56].

using the property gain function defined earlier, which gives the scenario for an adversary trying to guess the equivalence class.

DEFINITION 9.5.1. We say that mechanism M is ε-hiding wrt $\sim_A$ if

$$\mathcal{L}_{bv_{\sim_A}}(M) \quad \leq \quad 1+\varepsilon\,,$$

where $bv_{\sim_A}$ is defined at (9.18) and leakage is defined at (9.19).

The maximum chance of an adversary guessing which equivalence class of $\sim_A$ the secret is for an ε-hiding mechanism is bounded about by $(1+\varepsilon)\times V_{bv_{\sim_A}}(\pi)$, giving a robust privacy guarantee on $\sim_A$.

DEFINITION 9.5.2. We say that mechanism M is Δ-revealing wrt $\sim_T$ if

$$1+\Delta \quad \leq \quad \mathcal{L}_{bv_{\sim_T}}(M)\,,$$

where $bv_{\sim_T}$ is defined at (9.18) and leakage is defined at (9.19).

The best chance of an adversary guessing which equivalence class of $\sim_T$ the secret is for a Δ-revealing mechanism is therefore at least as much as $(1+\Delta)\times V_{bv_{\sim_T}}(\pi)$.

We can now compare mechanisms wrt their ε-hiding and Δ-revealing properties.

LEMMA 9.12. Let M_1, M_2 be mechanisms such that $M_1 \sqsubseteq M_2$. Then:

- If M_1 is ε-hiding wrt $\sim_A$ then so is M_2
- If M_2 is Δ-revealing wrt $\sim_T$ then so is M_1

Note that any post-processing steps, such as converting the data into character n-grams, results in more privacy (according to the data processing inequality) but (potentially) less accuracy for utility.

Next we can look at some constraints between privacy and utility.

THEOREM 9.13. If $\sim_A \subseteq \sim_T$ and M is both ε-hiding wrt $\sim_A$ and Δ-revealing wrt $\sim_T$ (both under a uniform prior) then $\Delta \leq \varepsilon$.

In particular, if $\sim_A = \sim_T$ then revealing any of $\sim_T$ will reveal the same about $\sim_A$. In general, if $\sim_A$ is finer than $\sim_T$ (as equivalence relations), then revealing the equivalence class for $\sim_T$ almost exactly already reveals quite a lot about the equivalence classes of $\sim_A$.

Consider however the following example where there are four secret values: $\{a,b,c,d\}$. Suppose we have that the equivalence classes of $\sim_T$ are $\{\{a,b\},\{c,d\}\}$ and for $\sim_A$ they are $\{\{a,c\},\{b,d\}\}$. The mechanism given by

$$M_{x,y} \quad := \quad 1 \text{ if } \left\{ \begin{array}{ll} & x \in \{a,b\} \wedge y = 0 \\ \vee & x \in \{c,d\} \wedge y = 1 \end{array} \right\} \text{ else } 0$$

has maximum leakage 2 and is 1-revealing wrt $\sim_T$ and 0-revealing wrt $\sim_A$; this means that the adversary has maximum chance of 1 of guessing $\sim_T$ but minimal chance of $1/2$ of guessing $\sim_A$.

This suggests that where $\sim_A$ represents equivalence classes over authors and $\sim_T$ represents equivalence classes over topics, if enough different authors write on the same topic, then there is a good chance of being able to protect authorship against inference attacks by machine learners whilst remaining in the same topic.

9.6 Experimental Results

Document Set The PAN@Clef tasks and other similar work have used a variety of types of text for author identification and author obfuscation. Our desiderata are that we have multiple authors writing on one topic (so as to minimise the ability of an author identification system to use topic-related cues) and to have more than one topic (so that we can evaluate utility in terms of accuracy of topic classification). Further, we would like to use data from a domain where there are potentially large quantities of text available, and where it is already annotated with author and topic.

Given these considerations, we chose "fan fiction" as our domain. Wikipedia defines *fan fiction* as follows: "Fan fiction ... is fiction about characters or settings from an original work of fiction, created by fans of that work rather than by its creator." This is also the domain that was used in the PAN@Clef 2018 author attribution challenge,[11] although for this work we scraped our own dataset. We chose one of the largest fan fiction sites and the two largest "fandoms" there;[12] these fandoms are our topics. We scraped the stories from these fandoms, the largest proportion of which are for use in training our topic classification model. We held out two subsets of size 20 and 50, evenly split between fandoms/topics, for the evaluation of our privacy mechanism.[13] We follow the evaluation framework of [107]: for each author we construct an known-author TEXT and an unknown-author SNIPPET that we have to match to an author on the basis of the known-author texts.

Word Embeddings There are sets of word embeddings trained on large datasets that have been made publicly available. Most of these, however, are already normalised, which makes them unsuitable for our method. We therefore use the Google News word2vec embeddings as the only large-scale unnormalised embeddings available.

Inference Mechanisms We have two sorts of machine learning inference mechanisms: our adversary mechanism for author identification, and our utility-related mechanism for topic classification. For each of these, we can define inference mechanisms both within the same representational space or in a different representational space. As we noted above, in practice both

[11] https://pan.webis.de/clef18/pan18-web/author-identification.html

[12] https://www.fanfiction.net/book/, with the two largest fandoms being Harry Potter (797,000 stories) and Twilight (220,000 stories).

[13] Our Algorithm 2 is computationally quite expensive, because each word $w = Vec^{-1}(x)$ requires the calculation of Euclidean distance with respect to the whole vocabulary. We thus use relatively small evaluation sets, as we apply the algorithm to them for multiple values of ε.

author identification adversary and topic classification will use different representations, but examining same-representation inference mechanisms can give an insight into what is happening within that space.

Different-representation author identification For this we use the algorithm by [107]. This algorithm is widely used: it underpins two of the winners of PAN shared tasks [122, 123]; is a common benchmark or starting point for other methods [124–127]; and is a standard inference attacker for the PAN shared task on authorship obfuscation.[14] It works by representing each text as a vector of space-separated character n-gram counts, and comparing repeatedly sampled subvectors of known-author texts and snippets using cosine similarity. We use as a starting point the code from a reproducibility study [128], but have modified it to improve efficiency.

Different-representation topic classification Here we choose fastText [129, 130], a high-performing supervised machine learning classification system. It also works with word embeddings; these differ from word2vec in that they are derived from embeddings over character n-grams, learnt using the same skipgram model as word2vec. This means it is able to compute representations for words that do not appear in the training data, which is helpful when training with relatively small amounts of data; also useful when training with small amounts of data is the ability to start from pretrained embeddings trained on out-of-domain data that are then adapted to the in-domain (here, fan fiction) data. After training, the accuracy on a validation set we construct from the data is 93.7%.

Same-representation author identification In the space of our word2vec embeddings, we can define an inference mechanism that for an unknown-author snippet chooses the closest known-author text by Euclidean distance.

Same-representation topic classification Similarly, we can define an inference mechanism that considers the topic classes of neighbours and predicts a class for the snippet based on that. This is essentially the standard k "Nearest Neighbours" technique (k-NN) [131], a non-parametric method that assigns the majority class of the k nearest neighbours. 1-NN corresponds to classification based on a Voronoi tesselation of the space, has low bias and high variance, and asymptotically has an error rate that is never more than twice the Bayes rate; higher values of k have a smoothing effect. Because of the nature of word embeddings, we would not expect this classification to be as accurate as the fastText classification above: in high-dimensional Euclidean space (as here), almost all points are approximately equidistant. Nevertheless, it can give an idea about how a snippet with varying levels of noise added is being shifted in Euclidean space with respect to other texts in the same topic. Here, we use $k = 5$. Same-representation author identification can then be viewed as 1-NN with author as class.

[14] http://pan.webis.de/clef17/pan17-web/author-obfuscation.html

	20-author set			
ε	SRauth	SRtopic	DRauth	DRtopic
none	12	16	15	18
30	8	18	16	18
25	8	18	14	17
20	5	11	11	16
15	2	11	12	17
10	0	15	11	19

	50-author set			
ε	SRauth	SRtopic	DRauth	DRtopic
none	19	36	27	43
30	19	37	29	43
25	17	34	24	41
20	12	28	19	42
15	9	22	13	42
10	1	24	10	43

Table 9.1: Number of correct predictions of author/topic in the 20-author set (left) and 50-author set (right), using 1-NN for same-representation author identification (SRauth), 5-NN for same-representation topic classification (SRtopic), the Koppel algorithm for different-representation author identification (DRauth) and fastText for different-representation topic classification (DRtopic).

Results: Table 9.1 contains the results for both document sets, for the unmodified snippets ("none") or with the privacy mechanism of Algorithm 2 applied with various levels of ε: we give results for ε between 10 and 30, as at $\varepsilon = 40$ the text does not change, while at $\varepsilon = 1$ the text is unrecognisable. For the 20-author set, a random guess baseline would give 1 correct author prediction, and 10 correct topic predictions; for the 50-author set, these values are 1 and 25 respectively.

Performance on the unmodified snippets using different-representation inference mechanisms is quite good: author identification gets 15/20 correct for the 20-author set and 27/50 for the 50-author set; and topic classification 18/20 and 43/50 (comparable to the validation set accuracy, although slightly lower, which is to be expected given that the texts are much shorter). For various levels of ε, with our different-representation inference mechanisms we see broadly the behaviour we expected: the performance of author identification drops, while topic classification holds roughly constant. Author identification here does not drop to chance levels: we speculate that this is because (in spite of our choice of dataset for this purpose) there are still some topic clues that the algorithm of [107] takes advantage of: one author of Harry Potter fan fiction might prefer to write about a particular character (e.g. Severus Snape), and as these character names are not in our word2vec vocabulary, they are not replaced by the privacy mechanism.

In our same-representation author identification, though, we do find performance starting

relatively high (although not as high as the different-representation algorithm) and then dropping to (worse than) chance, which is the level we would expect for our privacy mechanism. The k-NN topic classification, however, shows some instability, which is probably an artefact of the problems it faces with high-dimensional Euclidean spaces.

9.7 Conclusions

In this chapter we have shown how to protect authorship of documents using metric differential privacy. We showed that the construction of a utility-focussed mechanism can be achieved wrt the Earth Mover's distance, which is the utility measure for document similarity used in the natural language processing literature. Moreover we demonstrated experimentally the trade off between utility and privacy when using a machine learner to classify texts according to their topic versus guessing the authorship of the document.

This represents an important step towards the implementation of privacy mechanisms that could produce readable summaries of documents with a privacy guarantee. One way to achieve this goal would be to reconstruct readable documents from the bag-of-words output that our mechanism currently provides. A range of promising techniques for reconstructing readable texts from bag-of-words have already produced some good experimental results [132–134]. In future work we aim to explore how techniques such as these could be applied as a final post processing step for our mechanism.

9.8 Chapter Notes

This chapter is based on the published papers "Generalised Differential Privacy for Text Document Processing" [12] and "Processing Text for Privacy: An Information Flow Perspective" [8]. Omitted proofs and additional experimental results can be found in the appendix of [12].

Author Obfuscation Since the publication of the above papers, other works along a similar vein have been introduced in the literature extending Earth Mover's privacy to other metrics, most notably [135] which considers a more general hyperbolic representation.

A similar work prior to our publications is by Weggenmann and Kerschbaum [11] who also consider the author obfuscation problem but apply standard differential privacy using a Hamming distance of 1 between all documents. As with our approach, they consider the simplified utility requirement of topic preservation and use word embeddings to represent documents. Our approach differs in our use of the Earth Mover's metric to provide a strong utility measure for document similarity.

An early work in this area by Kacmarcik et al. [136] applies obfuscation by modifying the most important stylometric features of the text to reduce the effectiveness of author attribution. This approach was used in Anonymouth [137], a semi-automated tool that provides feedback to authors on which features to modify to effectively anonymise their texts. A similar approach was also followed by Karadhov et al. [138] as part of the PAN@Clef 2017 task.

Other approaches to author obfuscation, motivated by the PAN@Clef task, have focussed on the stronger utility requirement of semantic sensibility [139–141]. Privacy guarantees are therefore ad hoc and are designed to increase misclassification rates by the author attribution software used to test the mechanism.

Most recently there has been interest in training neural networks models which can protect author identity whilst preserving the semantics of the original document [142, 143]. Other related deep learning methods aim to obscure other author attributes such as gender or age [144, 145]. While these methods produce strong empirical results, they provide no formal privacy guarantees. Importantly, their goal also differs from the goal of our paper: they aim to obscure properties of authors in the *training set* (with the intention of the author-obscured learned representations being made available), while we assume that an adversary may have access to raw training data to construct an inference mechanism with full knowledge of author properties, and in this context aim to hide the properties of some other text external to the training set.

Machine Learning and Differential Privacy Outside of author attribution, there is quite a body of work on introducing differential privacy to machine learning: [7] gives an overview of a classical machine learning setting; more recent deep learning approaches include [146, 147]. However, these are generally applied in other domains such as image processing: text introduces additional complexity because of its discrete nature, in contrast to the continuous nature of neural networks. A recent exception is [148], which constructs a differentially private language model using a recurrent neural network; the goal here, as for instances above, is to hide properties of data items in the training set.

Text Document Privacy This typically refers to the sanitisation or redaction of documents either to protect the identity of individuals or to protect the confidentiality of their sensitive attributes. For example, a medical document may be modified to hide specifics in the medical history of a named patient. Similarly, a classified document may be redacted to protect the identity of an individual referred to in the text.

Most approaches to sanitisation or redaction rely on first identifying sensitive terms in the text, and then modifying (or deleting) only these terms to produce a sanitised document. Abril et al. [149] proposed this two-step approach, focussing on identification of terms using NLP techniques. Cumby and Ghani [150] proposed k-confusability, inspired by k-anonymity [2], to perturb sensitive terms in a document so that its (utility) class is confusable with at least k other classes. Their approach requires a complete dataset of similar documents for computing (mis)classification probabilities. Anandan et al. [151] proposed t-plausibility which generalises sensitive terms such that any document could have been generated from at least t other documents. Sánchez and Batet [152] proposed *C-sanitisation*, a model for both detection and protection of sensitive terms (C) using information theoretic guarantees. In particular, a *C-sanitised* document should contain no collection of terms which can be used to infer any of the sensitive terms.

Finally, there has been some work on noise-addition techniques in this area. In particular, Rodriguez-Garcia et al. [153] propose semantic noise, which perturbs sensitive terms in a document using a distance measure over the directed graph representing a predefined ontology.

10

Locality Sensitive Hashing with Differential Privacy

Recommender engines are a key feature of many large-scale internet websites; these systems are designed to provide users with recommendations relating to their individual profile – interests, purchase histories or other such collected information – thus motivating individuals to give up their sensitive data with the promise of some benefit. Of course, the data provider promises likewise not to breach the privacy of the individuals within the dataset. However large-scale breaches sometimes occur – such as Netflix's unwise release of user movie ratings data[1] – or else the data provider may be less-than-scrupulous – as occurred in the Cambridge Analytica scandal in which sensitive, *private* Facebook data was harvested for the purpose of maliciously manipulating individuals[2]. This motivates the study of privacy-preserving recommender systems in which users instead offer up their "noisy" information – with less risk but likewise less reward.

In this chapter we study differential privacy for recommender systems. We explore one particular application – "friend-matching" – in which users are recommended "friends" based on the similarity between their profile vectors. Our goal is to incorporate privacy-protection into the user profile, while preserving some utility of friend recommendations. Since friend-matching requires searching the entire dataset of users for the closest matches, an important aspect of this problem is the inefficiency of nearest neighbour search in very large datasets. One way in which this has been resolved is through the use of dimensionality reduction techniques which map similar users in the high-dimensional space to close points in a lower dimensional space. We will examine the use of one such method – Locality Sensitive Hashing (LSH) – which can be modified to provide privacy protection for the user. We show how to incorporate a metric

[1] https://www.wired.com/2009/12/netflix-privacy-lawsuit/
[2] https://www.nytimes.com/2018/04/04/us/politics/cambridge-analytica-scandal-fallout.html

differential privacy guarantee into LSH and study its properties wrt arbitrary metrics of interest. We then design a mechanism using a specific LSH – designed for the angular distance $\mathbf{d}_\theta$ – and demonstrate our mechanism on two datasets: the MovieLens dataset of movie ratings and the FourSquare dataset of location check-ins.

10.1 Introduction

Collaborative filtering broadly describes the model for user recommendations based on their interactions with items, such as prior purchases or ratings [154]. Recommendations to users are usually made in two forms: they can be recommended *items* to purchase, based on similar users who also purchased that item, or they can be recommended *users* to connect with, based on the similarity of their profiles as determined by their purchases or ratings. The former may be more commonly associated with ecommerce sites; the latter with social networks. This problem has been well-studied in the literature with well-established methods for efficient and accurate collaborative filtering [154]. Privacy concerns have caused researchers to adapt established methods to incorporate some level of privacy protection. In this chapter we concern ourself with the problem of privacy-preserving friend-matching, for which there are have already been several studies [155–158]. Our contribution to this problem is to situate it in the framework of metric differential privacy under a local differential privacy setting and provide a rigorous analysis of the privacy guarantees provided by our mechanism.

While there have been many works in the literature investigating the use of privacy-preserving techniques in this domain, including differential privacy, our interest in this problem stems from the particular properties of LSH. As we will see, an LSH acts as a type of probabilistic mapping between metric spaces: it maps elements from a metric space $(\mathcal{X}, \mathbf{d}_\mathcal{X})$ into the Hamming space $(\{0,1\}^k, \mathbf{d}_\mathrm{H})$ such that the distances $\mathbf{d}_\mathcal{X}(x, x')$ are approximately preserved. Applying differential privacy as a post-processing step on the Hamming space – using, for example, the randomised response mechanism – allows reasoning about privacy and utility wrt the original metric $\mathbf{d}_\mathcal{X}$ via the probabilistic LSH mapping. LSH schemes exist for various metrics, including: angular distance, Earth Mover's distance, Jaccard distance and l_p distances. However, differential privacy mechanisms do *not* exist for, eg., the angular distance or the Jaccard distance. An LSH-plus-random-response mechanism could fill this gap, and this motivates our exploration of LSH and, in particular, its instantiation for the angular distance.

10.1.1 Friend-Matching: Problem Setup

Our approach to this problem is to apply differential privacy in the local setting; that is, we assume that the data curator is *untrusted* and each user does not wish their precise data to be held by the data curator – instead they will pass over a noisy version of their ratings/purchases, which we will model as a real-valued vector. For each user the data curator then applies a nearest neighbour search on the collection of noisy vectors using the Locality Sensitive Hashing (LSH) technique for efficiency. This establishes a list of 'nearby' friend recommendations for that user.

The literature on this subject suggests that the *cosine distance* between user vectors is a good measure of similarity between users [159, 160]; this then is our *utility* measure. The reason that the LSH technique is effective is that it hashes users into buckets in such a way that the angular distance between users is approximately preserved; that is, users whose vectors are close wrt angular distance are highly likely to be hashed into the same bucket. Since angular distance roughly correlates with cosine distance, this means that the nearest neighbour search on the hash buckets typically finds users who are cosine-ly similar, and therefore provides good utility.

10.1.2 Our contributions

In summary, our contributions are as follows:

a. We observe that the LSH technique is a mapping between metrics and so it is natural to frame this problem in the context of metric differential privacy.

b. We show, using a toy example, that LSH is not privacy-preserving on its own, despite its use of underlying randomisation, thus countering some claims in the literature.

c. We propose different methods of adding noise to the system so as to satisfy local differential privacy – either adding the noise before LSH (ie. directly to elements in $\mathcal{X}$) or adding it afterwards (ie. on the bitstrings $\{0,1\}^k$). We provide some informal reasoning about privacy and utility resulting from each method and confirm our conclusions through experiments.

d. We formally analyse the privacy properties of the LSH-plus-random-response mechanism under the umbrella of metric differential privacy.

e. We apply our proposed mechanism to the problem of friend-matching and perform some experiments on 2 datasets of interest - the MovieLens dataset of movie ratings and the FourSquare dataset of location check-ins to demonstrate that our method approximately preserves the output of LSH.

10.2 Locality Sensitive Hashing (LSH)

Locality sensitive hashing (LSH), introduced by Indyk and Motwani [161], is an efficient, probabilistic method for determining (with high likelihood) which items in a set are similar when the number of items is very large.

We start with some technical preliminaries. Given a set $\mathcal{X}$, a similarity function $sim_{\mathcal{X}} : \mathcal{X} \times \mathcal{X} \to [0,1]$ assigns a score to pairs of elements $x, x' \in \mathcal{X}$ according to their similarity, where $sim_{\mathcal{X}}(x,x') = 1$ iff $x = x'$ and $sim_{\mathcal{X}}(x,x') = 0$ if x and x' are maximally 'dissimilar'. Dually, the *dissimilarity* function $\mathbf{d}_{\mathcal{X}}: \mathcal{X} \times \mathcal{X} \to [0,1]$ is defined as $\mathbf{d}_{\mathcal{X}}(x,x') = 1 - sim_{\mathcal{X}}(x,x')$. When $\mathbf{d}_{\mathcal{X}}$ is symmetric and sub-additive, we call it a dissimilarity *metric*. We will later instantiate $sim_{\mathcal{X}}$ and $\mathbf{d}_{\mathcal{X}}$ with specific metrics corresponding to hashing schemes. We denote by d_{euc} the Euclidean distance and by $\mathbf{d}_{\text{H}}$ the Hamming distance.

Finally, we recall the angular distance metric $\mathbf{d}_\theta : \mathbb{R}^n \times \mathbb{R}^n \to [0,1]$ is defined between non-zero vectors $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^n$ as

$$\mathbf{d}_\theta(\boldsymbol{x}, \boldsymbol{x}') \quad := \quad \tfrac{1}{\pi} \cos^{-1}\left(\tfrac{\boldsymbol{x} \cdot \boldsymbol{x}'}{\|\boldsymbol{x}\| \|\boldsymbol{x}'\|}\right)$$

We are now ready to define LSH.

DEFINITION 10.2.1 (Locality Sensitive Hashing [162])**.** A *locality sensitive hashing (LSH) scheme* wrt a dissimilarity metric $\mathbf{d}_\mathcal{X} : \mathcal{X} \times \mathcal{X} \to [0,1]$ is a pair $(\mathcal{H}, D_\mathcal{H})$ where $\mathcal{H}$ is a family of functions $h : \mathcal{X} \to \{0,1\}$ and $D_\mathcal{H}$ is a probability distribution over $\mathcal{H}$ such that given a pair $x, x' \in \mathcal{X}$, it holds that

$$\Pr_{h \sim D_\mathcal{H}}[h(x) \neq h(x')] = \mathbf{d}_\mathcal{X}(x, x'),$$

where the probability is taken over choices of h from the distribution $D_\mathcal{H}$.

Def. 10.2.1 says that when $\mathbf{d}_\mathcal{X}(x, x')$ is small, then x and x' are more likely to be assigned to the same bucket by any selected hash function h, whereas when x and x' are further apart, then there is a higher probability of selecting a hash function h which sends them to different buckets.

The LSH scheme defines a family of 1-bit hash functions. To construct an effective hash function from this family, we simply select a number κ of them to form a κ-bit hash string as follows:

DEFINITION 10.2.2 (κ-bit LSH)**.** Let $(\mathcal{H}, D_\mathcal{H})$ be an LSH scheme on $\mathcal{X}$. Given $\kappa \in \mathbb{N}$ define the function $H : \mathcal{X} \to \{0,1\}^\kappa$ by

$$H(x) \quad := \quad (h_1(x), h_2(x), \ldots, h_\kappa(x))$$

where $h_1, h_2, \ldots, h_\kappa$ are independently drawn from $D_\mathcal{H}$. Then H is called a κ-bit LSH function.

Finally, letting $D_\mathcal{H}^\kappa$ be the distribution of κ-bit LSH functions H, we denote by $H^* : \mathcal{X} \to \mathbb{D}(\{0,1\}^\kappa)$ the randomised algorithm that chooses H according to $D_\mathcal{H}^\kappa$ and outputs the hash value $H(x)$ of a given input x.

10.2.1 Random-Projection-Based Hashing

There are a variety of LSH families corresponding to useful metrics, such as the angular distance [162, 163], Jaccard metric [164], Earth Mover's metric [162], and l_p metric with $p \in (0,2]$ [165]. In this section we present an LSH scheme wrt the angular distance $\mathbf{d}_\theta$ called *random-projection-based hashing*.

A random-projection-based hash is a one-bit hash associated with a randomly chosen normal vector $\boldsymbol{r}$ that defines a hyperplane through the origin. We define this formally as follows.

DEFINITION 10.2.3 (Random-Projection-Based Hash [162])**.** A random-projection-based hash is a function $h_{\boldsymbol{r}} : \mathbb{R}^n \to \{0,1\}$ satisfying:

$$h_{\boldsymbol{r}}(\boldsymbol{x}) = \begin{cases} 0 & (\text{if } \boldsymbol{r} \cdot \boldsymbol{x} < 0) \\ 1 & (\text{otherwise}) \end{cases}$$

for all $\boldsymbol{x} \in \mathbb{R}^n$, where $\cdot$ denotes inner product and $\boldsymbol{r} \in \mathbb{R}^n$ is an n-dimensional vector chosen by selecting each element from the standard normal distribution $N(0,1)$.

It has been shown [162] that the family of random-projection-based hashes $h_{\boldsymbol{r}}$ is an LSH scheme wrt the $\mathbf{d}_\theta$ metric. In other words, for any pair x, x', the probability that a randomly chosen 1-bit hash (Def. 10.2.3) will hash x and x' to the same value (0 or 1) is higher when $\mathbf{d}_\theta(x, x')$ is smaller.

10.2.2 Privacy with LSH

Returning to our problem of interest – friend-matching – we describe informally how the application of a privacy-preserving LSH might proceed. The process is depicted in Figure 10.1. Each user is equipped with an n-length vector $\vec{u}$, with each value $\vec{u}[i]$ an integer-value in $\{-5 \ldots 5\}$ corresponding to, say, a movie rating for movie i.[3] Clearly not every user will have watched every movie – these vectors will be sparse with many missing values which we can fill in with some default value (say, 0). Also, n could be large – somewhere around 10,000 in a typical dataset of interest.

The data curator chooses a value $\kappa \ll n$ and randomly generates κ hyperplanes to use as random projection hashes in a κ-bit LSH (Def. 10.2.2) defined by a function H. These are sent to each user. At this point there are two possible workflows:

1. the user applies a noise-adding mechanism M for integer-valued vectors to their vector, before applying H, and returns $H \circ M(\vec{u})$; or
2. the user applies H and then a noise-adding mechanism B for bitstrings, returning $B \circ H(\vec{u})$.

Friend-matching then proceeds for a user identified by the noisy hash bucket $\vec{b}$ by searching the bucket $\vec{b}$ and its nearest neighbours wrt the Hamming distance on bitstrings, for a list of k users to recommend.

Considering *where to add the noise?*, notice that for the mechanism $H \circ M$ the privacy guarantee is relatively easy to compute – H is a post-processing step and so the mechanism $H \circ M$ satisfies $\mathbf{d}$-privacy where $\mathbf{d}$ is the metric for which M is designed. In the case of $B \circ H$ however, the privacy guarantee is determined by both B and the probabilistic projection H, and so we need to do more work to reason about the overall privacy provided.

Regarding utility, since we want to preserve angular distance, our mechanism M should be designed for this purpose (in the case that we choose the mechanism $H \circ M$). However, on

[3] As per standard practice in this domain, we subtract eg. the median or mean value to ensure that a default value of 0 is representative of an average score.

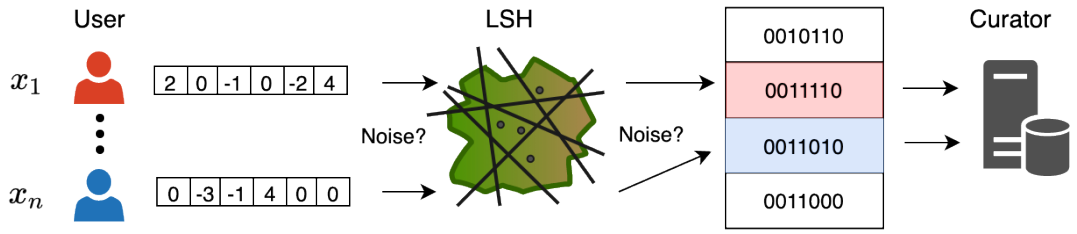


Figure 10.1: The application of random-projection based hashing with noise to user vectors in a movie ratings dataset to produce noisy bitstrings. The LSH function constructs hyperplanes in the space which divide it into equivalence classes. There are 2 places in which we could add noise – either before or after LSH.

integer-valued vectors, the literature on metric differential privacy provides only options for Euclidean distance (eg. the n-dimensional Laplace mechanism) or Manhattan distance. While these may seem like reasonable choices, observe that the n-dimensional Laplace mechanism (introduced in Chapter 9, Thm. 9.7) can be decomposed into a mechanism which selects noise vectors *uniformly at random* wrt angular distance, with length scaled by a Gamma distribution. This suggests that, if preserving angular distance is our goal, this mechanism is not going to be well-suited.

However, in the case that we choose a mechanism $B \circ H$, our utility goals are aligned with the metrics on H and B (since B is designed for the metric induced by H), and so the utility of the overall mechanism will be determined by the usefulness of each of H and B wrt the particular metric of utility we have chosen.[4]

We will experiment with these two ideas, introducing two privacy mechanisms for an angular distance utility goal: a *Laplace-then-LSH* mechanism in the style of $H \circ M$ and a *LSH-then-RR* mechanism in the style of $B \circ H$.

10.2.3 Is LSH private?

Since LSH is itself a 'noisy' mechanism, it might seem reasonable to think that LSH is already privacy-preserving in some manner, as other works have suggested [166–168]. Taking random-projection-based LSH as our example, since the choice of each hyperplane is random, a user may simply apply LSH directly to their own data, and it could be argued that the probabilistic nature of LSH affords some 'plausible deniability' as to whether the user is truly a nearest neighbour to the other users in their hash bucket. However, this 'plausible deniability' guarantee crucially relies on keeping secret the chosen hash functions that form the κ-bit LSH (Def. 10.2.2). When these are known, privacy can break down completely, as illustrated in Example 10.1. Unfortunately, in practice, the choice of hash functions must be revealed in order to get any utility from the LSH mechanism.

[4] Recalling our investigation of optimality in Chapter 6, there is still work to do to pick the *right* mechanism for the utility measure of choice even when the mechanism is designed for the correct metric.

In general, for any number of hash functions and any length input, a random-projection-based hashing mechanism which releases its choice of hash functions also leaks the equivalence classes of the secrets. Such mechanisms belong to the 'k-anonymity'-style of privacy mechanisms which promise privacy by hiding secrets in equivalence classes of size at least k. These have been shown to be unsafe due to their failure to compose well [8, 169]. This failure leads to the potential for linkage or intersection attacks by an adversary equipped with the right sort of auxiliary information. For this reason, we consider compositionality an essential property for a privacy-preserving system. LSH with hash function release does not provide such privacy guarantees.

Example 10.1 (Privacy Breakdown of LSH)**.** We present a simple example to show how privacy with LSH can break down. Consider the set of secret values $\mathcal{X} = \{(0,1),(1,0),(1,1)\}$. Each element of $\mathcal{X}$ could, for example, correspond to whether or not an individual has rated two movies A and B. We model an LSH as a probabilistic channel $h^* : \mathcal{X} \to \mathbb{D}\{0,1\}$ that maps a secret input to a binary observation.

For brevity we deal with a single random-projection-based hashing h presented in §10.2.1. That is, we randomly choose a vector $\boldsymbol{r}$ representing the normal to a hyperplane, and compute the inner product of $\boldsymbol{r}$ with an input vector $\boldsymbol{x} \in \mathcal{X}$. The hash function h outputs 0 if the inner product is negative and 1 otherwise. For example, if $\boldsymbol{r} = (1, -\frac{1}{2})$ is chosen, then the hash function h is defined as:

$$\begin{aligned} h : \mathcal{X} &\to \{0,1\} \\ (0,1) &\mapsto 0 \\ (1,0) &\mapsto 1 \\ (1,1) &\mapsto 1 \end{aligned}$$

In fact, there are exactly 6 possible (deterministic) hash functions for any choice of the normal vector $\boldsymbol{r}$, corresponding to hyperplanes which separate different pairs of points:

$$\begin{array}{ccc} h_1 & h_2 & h_3 \\ (0,1) \mapsto 1 & (0,1) \mapsto 0 & (0,1) \mapsto 1 \\ (1,0) \mapsto 0 & (1,0) \mapsto 1 & (1,0) \mapsto 0 \\ (1,1) \mapsto 0 & (1,1) \mapsto 1 & (1,1) \mapsto 1 \end{array}$$

$$\begin{array}{ccc} h_4 & h_5 & h_6 \\ (0,1) \mapsto 0 & (0,1) \mapsto 1 & (0,1) \mapsto 0 \\ (1,0) \mapsto 1 & (1,0) \mapsto 1 & (1,0) \mapsto 0 \\ (1,1) \mapsto 0 & (1,1) \mapsto 1 & (1,1) \mapsto 0 \end{array}$$

Each of h_1, h_2, h_3, and h_4 occurs with probability $1/8$, while h_5 and h_6 each occur with probability $1/4$. The resulting channel h^*, computed as the probabilistic sum of these deterministic hash functions, turns out to leak no information on the secret input (ie., all outputs have equal probability conditioned on each input).

This indicates that the LSH mechanism h^* above is perfectly private. However, in practice

LSH also requires the release of the choice of the normal vector $\boldsymbol{r}$ (eg. [166])[a]. In other words, the choice of hash function is leaked. Notice that in our example, the functions h_1 to h_4 correspond to deterministic mechanisms which leak exactly 1 bit of the secret, while h_5 and h_6 leak nothing. In other words, with 50% probability, 1 bit of the 2-bit secret is leaked. Not only that, but mechanisms h_1 and h_2 leak the secret $(0,1)$ exactly, and similarly, mechanisms h_3 and h_4 leak $(1,0)$ exactly. Thus, the release of the normal vector $\boldsymbol{r}$ destroys the privacy guarantee.

[a] In fact, since the channel on its own leaks nothing, there *must* be further information released in order to learn anything useful from this channel.

10.3 LSH-based Privacy Mechanisms

In this section we present two privacy mechanisms: the *LSH-then-RR (LSHRR)* mechanism which applies the noise *after* LSH, and the *Laplace-then-LSH (LapLSH)* which applies the noise beforehand.

The LSHRR mechanism is similar to Google's RAPPOR mechanism [85] except that it uses an LSH hash function instead of "Bloom filters".[5] Specifically, LSHRR first computes the hash value of an input vector using LSH, and applies the randomised response (RR) to the hash value. For comparison, we introduce the LapLSH mechanism which applies the Laplace mechanism to the original input before computing its hash value using LSH.

Technical Preliminaries

In the following for $\kappa \in \mathbb{N}$ we let $\mathcal{V} = \{0,1\}^\kappa$ be the set of κ-length bitstrings. We recall the randomised response mechanism from Chapter 3 (Def. 3.4.3) which we denote as R^ε (for fixed ε). We also recall the n-dimensional Laplace mechanism $Lap^n_\varepsilon(v)$ defined in Chapter 9 (Thm. 9.7) which is implemented by drawing a unit vector uniformly at random over the unit sphere, and then scaling according to the Gamma distribution with shape n and scale $1/\varepsilon$. For a probabilistic mechanism $M\colon \mathcal{X} \to \mathbb{D}\mathcal{Y}$ we write $M(x)(Y)$ for the probability that an output $y \in Y$ is produced from input x.

10.3.1 Privacy Measures

The privacy guarantees for our LSH-based mechanisms will use the extended versions of differential privacy and metric differential privacy which we now recall and define.

First, the extended (ε,δ) notion of differential privacy is usually defined as follows [7]:

> **Definition 10.3.1** (Differential privacy)**.** A randomised algorithm $A : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ provides (ε,δ)-*differential privacy* wrt an adjacency relation $\Phi \subseteq \mathcal{X} \times \mathcal{X}$ if for any

[5] Bloom filters are a type of hash function which is not distance-preserving.

$(x, x') \in \Phi$ and any $S \subseteq \mathcal{Y}$,

$$A(x)(S) \leq e^{\varepsilon} A(x')(S) + \delta$$

where the probability is taken over the random choices in A.

We can extend this idea to metric differential privacy. In this paper, we introduce a generalised definition using a function δ over inputs and an arbitrary function ξ over $\mathcal{X}$ rather than a metric $\mathbf{d}$ as follows.

Definition 10.3.2 (Extended differential privacy)**.** Given two functions $\xi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ and $\delta : \mathcal{X} \times \mathcal{X} \to [0, 1]$, a randomised algorithm $A : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ provides (ξ, δ)-*extended differential privacy (XDP)* if for all $x, x' \in \mathcal{X}$ and for any $S \subseteq \mathcal{Y}$,

$$A(x)(S) \leq e^{\xi(x,x')} A(x')(S) + \delta(x, x'),$$

where the probability is taken over the random choices in A.

Although we use an arbitrary function ξ here, we will relate this back to the metric $\mathbf{d}$ in this chapter when analysing our privacy mechanisms. Hereafter we sometimes abuse notation and write δ when $\delta(x, x')$ is a constant function returning the same real number independently of the inputs x and x'.

Next, we review the notion of *privacy loss* and *privacy loss distribution* [170].

Definition 10.3.3 (Privacy loss random variable)**.** Let $A : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be a randomised algorithm. The *privacy loss* on an output $y \in \mathcal{Y}$ wrt inputs $x, x' \in \mathcal{X}$ is defined by:

$$\mathcal{L}_{x,x',y} = \ln\left(\frac{A(x)(y)}{A(x')(y)}\right),$$

where the probability is taken over the random choices in A. When $A(x)(y) \neq 0$ and $A(x')(y) = 0$, then define $\mathcal{L}_{x,x',y} = \infty$. When $A(x)(y) = 0$, then define $\mathcal{L}_{x,x',y} = -\infty$. Then the *privacy loss random variable* $\mathcal{L}_{x,x'}$ of x over x' is the real-valued random variable representing the privacy loss $\mathcal{L}_{x,x',y}$ where y is sampled from $A(x)$.

Given inputs $x, x' \in \mathcal{X}$ and a privacy loss $\ell \in \mathbb{R}$, let $\mathcal{Y}_{x,x',\ell} = \{y \in \mathcal{Y} \mid \mathcal{L}_{x,x',y} = \ell\}$. Then *the privacy loss distribution* $\omega_{x,x'}$ of x over x' is defined as a distribution over $\mathbb{R} \cup \{-\infty, \infty\}$ such that:

$$\omega_{x,x'}(\ell) = \sum_{y \in \mathcal{Y}_{x,x',\ell}} A(x)(y).$$

Next, we recall the notion of *mean-concentrated differential privacy* [170] which is defined using the notion of subgaussian random variables as follows.

Definition 10.3.4 (Subgaussian)**.** For $\tau \in \mathbb{R}_{\geq 0}$, a random variable Z over $\mathbb{R}$ is *τ-subgaussian* if for all $s \in \mathbb{R}$, $\mathbb{E}[\exp(sZ)] \leq \exp(\frac{s^2\tau^2}{2})$. A random variable Z is *subgaussian* if there exists a $\tau \in \mathbb{R}_{>0}$ such that Z is τ-subgaussian.

Definition 10.3.5 (Mean-concentrated *DP*)**.** Let $\mu, \tau \in \mathbb{R}_{\geq 0}$. A randomised algorithm A provides *(μ, τ)-mean-concentrated differential privacy (CDP)* wrt an adjacency relation $\Phi \subseteq \mathcal{X} \times \mathcal{X}$ if for any $(x, x') \in \Phi$, the privacy loss random variable $\mathcal{L}_{x,x'}$ of x over x' satisfies that $\mathbb{E}[\mathcal{L}_{x,x'}] \leq \mu$, and that $\mathcal{L}_{x,x'} - \mathbb{E}[\mathcal{L}_{x,x'}]$ is τ-subgaussian.

Finally, we recall the notion of probabilistic differential privacy [171, 172].

Definition 10.3.6 (Probabilistic *DP*)**.** Let $\varepsilon, \delta \in \mathbb{R}_{\geq 0}$. A randomised algorithm A provides *(ε, δ)-probabilistic differential privacy (PDP)* wrt an adjacency relation $\Phi \subseteq \mathcal{X} \times \mathcal{X}$ if for any $(x, x') \in \Phi$, the privacy loss random variable satisfies $\Pr[\mathcal{L}_{x,x'} > \varepsilon] \leq \delta$.

We recall [170] that mean-concentrated differential privacy is a strictly stronger notion than probabilistic differential privacy, which is again stronger than standard differential privacy.

Our motivation for including the above definitions is that we will extend these to versions corresponding to metric differential privacy, showing firstly that the strict implications still hold in the extended definitions, and secondly that our privacy-preserving LSH mechanism implements a type of mean-concentrated metric differential privacy.

10.3.2 Construction of LSHRR

The *LSH-then-RR privacy mechanism* (*LSHRR*) is a randomised algorithm $Q_{\mathsf{LSHRR}} : \mathcal{X} \to \mathbb{D}\mathcal{V}$ that (i) randomly chooses a κ-bit LSH function H, (ii) computes the κ-bit hash code $H(\boldsymbol{x})$ of a given input $\boldsymbol{x}$, and then (iii) applies randomised response to each bit of the hash code $H(\boldsymbol{x})$.

We first recall the *bitwise randomised response* as the privacy mechanism that applies the randomised response R^ε to each bit of an input (bitstring) independently.

Definition 10.3.7 ((ε, κ)-bitwise randomised response)**.** Let $\varepsilon \in \mathbb{R}_{\geq 0}$ and $\kappa \in \mathbb{N}$. The *(ε, κ)-bitwise randomised response* is defined as the randomised algorithm $Q_{\mathsf{brr}} : \mathcal{V} \to \mathbb{D}\mathcal{V}$ that maps a bitstring $\boldsymbol{v} = (v_1, v_2, \ldots, v_\kappa)$ to another $\boldsymbol{y} = (y_1, y_2, \ldots, y_\kappa)$ with the following probability:

$$Q_{\mathsf{brr}}(\boldsymbol{v})(\boldsymbol{y}) = \prod_{i=1}^{\kappa} R^\varepsilon(v_i)(y_i).$$

Now we define the LSHRR mechanism as follows.

Definition 10.3.8 (LSHRR)**.** Let $Q_{\mathsf{brr}} : \mathcal{V} \to \mathbb{D}\mathcal{V}$ be the (ε, κ)-bitwise randomised response. The *ε-LSH-then-RR privacy mechanism* (*LSHRR*) with a κ-bit

LSH function $H : \mathcal{X} \to \mathcal{V}$ is the randomised algorithm $Q_H : \mathcal{X} \to \mathbb{D}\mathcal{V}$ defined by $Q_H = Q_{\mathsf{brr}} \circ H$. Given a distribution $D^{\kappa}_{\mathcal{H}}$ of the κ-bit LSH functions, the *ε-LSH-then-RR privacy mechanism* wrt $D^{\kappa}_{\mathcal{H}}$ is the randomised algorithm $Q_{\mathsf{LSHRR}} : \mathcal{X} \to \mathbb{D}\mathcal{V}$ defined by $Q_{\mathsf{LSHRR}} = Q_{\mathsf{brr}} \circ H^*$.

Note that there are two kinds of randomness in the LSHRR: (a) the randomness in choosing a (deterministic) LSH function H from $D^{\kappa}_{\mathcal{H}}$ (e.g., the random seed $\boldsymbol{r}$ in the random-projection-based hashing h_{proj}[6]), and (b) the random noise added by the bitwise randomised response Q_{brr}.

10.3.3 Construction of LapLSH

For the purposes of experimental evaluation and discussion, we define the *Laplace-then-LSH privacy mechanism* (LapLSH) as a randomised algorithm $Q_{\mathsf{LapLSH}} : \mathcal{X} \to \mathbb{D}\mathcal{V}$ that (i) randomly chooses a κ-bit LSH function H, (ii) applies the multivariate Laplace mechanism Q_{Lap} to $\boldsymbol{x}$, and then (iii) computes the κ-bit hash code $H(Q_{\mathsf{Lap}}(\boldsymbol{x}))$ of the obfuscated input $Q_{\mathsf{Lap}}(\boldsymbol{x})$.

Formally, we define the mechanism as follows. Recall (§10.2) that $H^* : \mathcal{X} \to \mathbb{D}\mathcal{V}$ is the randomised algorithm that randomly chooses a κ-bit LSH function H according to a distribution $D^{\kappa}_{\mathcal{H}}$ and outputs the hash value $H(\boldsymbol{x})$ of a given input $\boldsymbol{x}$.

Definition 10.3.9 (LapLSH). Let $(\mathcal{X}, \mathbf{d})$ be a metric space and let $Q_{\mathsf{Lap}} : \mathcal{X} \to \mathbb{D}\mathcal{X}$ be an $(\varepsilon, \mathbf{d})$-Laplace mechanism.[7] The *$(\varepsilon, \mathbf{d})$-Laplace-then-LSH privacy mechanism* (*LapLSH*) with a κ-bit LSH function $H : \mathcal{X} \to \mathcal{V}$ is the randomised algorithm $Q_{\mathsf{Lap}H} : \mathcal{X} \to \mathbb{D}\mathcal{V}$ defined by $Q_{\mathsf{Lap}H} = H \circ Q_{\mathsf{Lap}}$. Given a distribution $D^{\kappa}_{\mathcal{H}}$ of the κ-bit LSH functions, the *$(\varepsilon, \mathbf{d})$-Laplace-then-LSH privacy mechanism* wrt $D^{\kappa}_{\mathcal{H}}$ is the randomised algorithm $Q_{\mathsf{LapLSH}} : \mathcal{X} \to \mathbb{D}\mathcal{V}$ defined by $Q_{\mathsf{LapLSH}} = H^* \circ Q_{\mathsf{Lap}}$.

The LapLSH mechanism we use will be a $(\varepsilon, d_{\mathrm{euc}})$-LapLSH mechanism defined over n-dimensional real-valued vectors. This is detailed further in §10.5.

10.4 Analyses of the Privacy Mechanisms

In this section we show two types of privacy provided by the two mechanisms LSHRR (Section 10.3.2) and LapLSH (Section 10.3.3): (i) the *privacy guarantee for hash values* that we learn after both input vectors and hash seeds are selected, and (ii) the *privacy guarantee for inputs* that represents the probability distribution of possible levels of privacy guarantees. Note that proofs for this section can be found in Appendix §C.1.

The privacy guarantee for hash values is defined using the Hamming distance $\mathbf{d}_{\mathrm{H}}$ between hash values, and does not immediately derive the privacy wrt the metric $\mathbf{d}_{\mathcal{X}}$ on the input, since LSH preserves $\mathbf{d}_{\mathcal{X}}$ only probabilistically and approximately. To obtain the privacy guarantee for

[6] More specifically, in the definition of Q_H, a tuple of seeds $r = (r_1, \ldots, r_\kappa)$ for the κ-bit LSH function $H = (h_1, \ldots, h_\kappa)$ is randomly chosen.

[7] Note that we use the general terminology 'Laplace mechanism' broadly as per [15] to describe mechanisms Q satisfying $Q(y)(z) = \lambda(z) e^{-\mathbf{d}(y,z)}$.

the inputs, we introduce a variety of privacy notions with shared randomness, including new notions of extended/concentrated privacy with a metric.

10.4.1 LSHRR: Privacy Guarantees for Hash Values

We first present the privacy guarantee for LSHRR, which relies on the XDP of the bitwise randomised response Q_{brr} wrt the Hamming distance $\mathbf{d}_{\mathrm{H}}$ as follows.

> **Proposition 10.1** (XDP of BRR). Let $\varepsilon \in \mathbb{R}_{\geq 0}$ and $\kappa \in \mathbb{Z}_{>0}$. The (ε, κ)-bitwise randomised response Q_{brr} provides $(\varepsilon \cdot \mathbf{d}_{\mathrm{H}}, 0)$-XDP.

The exact degree of privacy provided by LSHRR depends on the hash values $H(\boldsymbol{x})$, hence on the random choice of hash seeds $\boldsymbol{r}$. This means that an input $\boldsymbol{x}$ may be protected only weakly when an 'unlucky' seed is chosen to produce the hash value $H(\boldsymbol{x})$. For example, if many hyperplanes given by unlucky seeds split a small specific area in the input space $\mathcal{X}$, then the hash value $H(\boldsymbol{x})$ reveals much information on the input $\boldsymbol{x}$ chosen from that small area.

Hence the exact degree of privacy can be evaluated only after obtaining the hash seeds and the input vectors. Specifically, the privacy provided by LSHRR is proportional to the Hamming distance $\mathbf{d}_{\mathrm{H}}$ between hash values as follows.

> **Proposition 10.2** (Privacy of Q_H wrt $d_{\varepsilon H}$). Let $\varepsilon \in \mathbb{R}_{\geq 0}$, $H : \mathcal{X} \to \mathcal{V}$ be a κ-bit LSH function, and $d_{\varepsilon H} : \mathcal{X} \times \mathcal{X} \to \mathbb{Z}_{\geq 0}$ be the pseudometric defined by $d_{\varepsilon H}(\boldsymbol{x}, \boldsymbol{x}') = \varepsilon \cdot \mathbf{d}_{\mathrm{H}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))$ for each $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$. Then the ε-LSHRR mechanism Q_H provides $(d_{\varepsilon H}, 0)$-XDP.

Since the LSH function H preserves the metric $\mathbf{d}_{\mathcal{X}}$ on the input only probabilistically and approximately, Prop. 10.2 does not immediately derive the degree of XDP wrt $\mathbf{d}_{\mathcal{X}}$ for inputs. In §10.4.2 we will show LSHRR's privacy guarantee for the inputs.

Nevertheless, by Prop. 10.2, the LSHRR provides $\kappa\varepsilon$-*DP* in the worst case, ie., when the Hamming distance between vectors is maximum due to an 'unlucky' choice of hash seeds and/or large original distance $\mathbf{d}_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$ between the inputs $\boldsymbol{x}, \boldsymbol{x}'$.

> **Proposition 10.3** (Worst-case privacy of Q_H). Let $\varepsilon \in \mathbb{R}_{\geq 0}$ and $H : \mathcal{X} \to \mathcal{V}$ be a κ-bit LSH function. The ε-LSHRR mechanism Q_H provides $\kappa\varepsilon$-*DP*.

This shows that Q_H achieves weaker privacy for a larger κ.

10.4.2 LSHRR: Privacy Guarantees for Inputs

Now we aim to derive the privacy guarantee wrt the original inputs, ie., taking into account the probabilistic mapping H from inputs wrt $\mathbf{d}_{\mathcal{X}}$ to bitstrings wrt $\mathbf{d}_{\mathrm{H}}$. We will show that the guarantee is a form of concentrated differential privacy (CXDP) over a function of the input metric $\mathbf{d}_{\mathcal{X}}$.

In typical applications of LSH such as approximate nearest neighbour search, multiple users produce hash values by employing the same hash seeds. To take such shared seeds into account we will need to modify some existing privacy notions.

Hereafter we denote by $\mathcal{R}$ a finite set of shared input, and by $A_r : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ a randomised algorithm A from $\mathcal{X}$ to $\mathcal{Y}$ with a shared input $r \in \mathcal{R}$. Given a distribution λ over $\mathcal{R}$, we denote by $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ the randomised algorithm that draws a shared input r from λ and behaves as A_r; ie., for $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, $A_\lambda(x)(y) = \sum_{r\in\mathcal{R}} \lambda_r A_r(x)(y)$. For brevity we sometimes abbreviate A_λ as A.

Now we introduce privacy loss variables with shared randomness.

> **Definition 10.4.1** (Privacy loss random variable with shared randomness)**.** Given a randomised algorithm $A_r : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ with a shared input r, the *privacy loss* on an output $y \in \mathcal{Y}$ wrt inputs $x, x' \in \mathcal{X}$ and $r \in \mathcal{R}$ is defined by:
>
> $$\mathcal{L}_{x,x',y,r} = \ln\left(\frac{A_r(x)(y)}{A_r(x')(y)}\right),$$
>
> where the probability is taken over the random choices in A_r. Given a distribution λ over $\mathcal{R}$, the *privacy loss random variable* $\mathcal{L}_{x,x'}$ of x over x' wrt λ is the real-valued random variable representing the privacy loss $\mathcal{L}_{x,x',y,r}$ where r is sampled from λ and y is sampled from $A_r(x)$. Here we call r a *shared randomness*.

Then the *the privacy loss distribution* $\omega_{x,x'}$ of x over x' is defined analogously to Def. 10.3.3.

Along the lines of concentrated differential privacy (Def. 10.3.5) and probabilistic differential privacy (Def. 10.3.6), we introduce the following definitions which incorporate a metric $\mathbf{d}_\mathcal{X}$ of interest as well as privacy loss with shared randomness. We leave the examination of properties of these definitions to future work, noting here only that they are somewhat analogous to the aforementioned versions for standard differential privacy.

> **Definition 10.4.2** (Mean-concentrated XDP)**.** Let $\mu \in \mathbb{R}_{\geq 0}$, $\tau \in \mathbb{R}_{>0}$, $\lambda \in \mathbb{D}\mathcal{R}$, and $\mathbf{d}_\mathcal{X} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ be a metric. A randomised algorithm $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ provides $(\mu, \tau, \mathbf{d}_\mathcal{X})$-*mean-concentrated extended differential privacy (CXDP)* if for all $x, x' \in \mathcal{X}$, the privacy loss random variable $\mathcal{L}_{x,x'}$ of x over x' wrt. λ it holds that $\mathbb{E}[\mathcal{L}_{x,x'}] \leq \mu \cdot \mathbf{d}_\mathcal{X}(x, x')$, and that $\mathcal{L}_{x,x'} - \mathbb{E}[\mathcal{L}_{x,x'}]$ is τ-subgaussian.

We also introduce the notion of probabilistic XDP (abbreviated as PXDP) by extending the notions of XDP and PDP as follows.

> **Definition 10.4.3** (Probabilistic XDP)**.** Let $\lambda \in \mathbb{D}\mathcal{R}$, $\xi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, and $\delta : \mathcal{X} \times \mathcal{X} \to [0, 1]$. A randomised algorithm $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ provides (ξ, δ)-*probabilistic extended differential privacy (PXDP)* if for all $x, x' \in \mathcal{X}$, the privacy loss random variable $\mathcal{L}_{x,x'}$ of x over x' wrt. λ satisfies $\Pr[\, \mathcal{L}_{x,x'} > \xi(x, x') \,] \leq \delta(x, x')$.

Again, we sometimes abuse notation and simply write δ when $\delta(x, x')$ is constant for all x, x'.

Now we show that CXDP implies PXDP, and that PXDP implies XDP as follows.

Proposition 10.4 (CXDP ⇒ PXDP)**.** Let $\mu \in \mathbb{R}_{\geq 0}$, $\tau \in \mathbb{R}_{>0}$, $\lambda \in \mathbb{D}\mathcal{R}$, $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be a randomised algorithm, and $\mathbf{d}_\mathcal{X}$ be a metric over $\mathcal{X}$. Let $\delta \in (0,1]$ and $\varepsilon = \tau\sqrt{-2\ln\delta}$. We define $\xi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ by $\xi(x,x') = \mu{\cdot}\mathbf{d}_\mathcal{X}(x,x') + \varepsilon$ for all $x, x' \in \mathcal{X}$. If A_λ provides $(\mu,\tau,\mathbf{d}_\mathcal{X})$-CXDP, then it provides (ξ,δ)-PXDP.

Proposition 10.5 (PXDP ⇒ XDP)**.** Let $\lambda \in \mathbb{D}\mathcal{R}$, $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be a randomised algorithm, $\xi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, and $\delta : \mathcal{X} \times \mathcal{X} \to [0,1]$. If A_λ provides (ξ,δ)-PXDP, then it provides (ξ,δ)-XDP.

Now we can examine the privacy properties of LSHRR, and we have that it provides CXDP.

Theorem 10.6 (CXDP of the LSHRR)**.** The ε-LSH-based privacy mechanism Q_{LSHRR} provides $(\varepsilon\kappa, \frac{\varepsilon\kappa}{2}, \mathbf{d}_\mathcal{X})$-CXDP.

This implies that the LSHRR provides PXDP, hence XDP.

Theorem 10.7 (PXDP/XDP of the LSHRR)**.** Let $\delta \in \mathbb{R}_{>0}$ and $\varepsilon' = \varepsilon\sqrt{\frac{-\ln\delta}{2}}$. We define $\xi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ by $\xi(\boldsymbol{x},\boldsymbol{x}') = \varepsilon\kappa{\cdot}\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}') + \varepsilon'\sqrt{\kappa}$. The ε-LSH-based mechanism Q_{LSHRR} provides (ξ,δ)-PXDP, hence (ξ,δ)-XDP.

Finally, we present a better bound for PXDP of the proposed mechanism. For $a, b \in \mathbb{R}_{>0}$, let $D_{\mathrm{KL}}(a\|b) = a\ln\frac{a}{b} + (1-a)\ln\frac{1-a}{1-b}$.

Proposition 10.8 (Tighter bound for PXDP/XDP)**.** For an $\alpha \in \mathbb{R}_{>0}$, we define $\xi_\alpha : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ and $\delta_\alpha : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ by:

$$\xi_\alpha(\boldsymbol{x},\boldsymbol{x}') = \varepsilon\kappa(\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}') + \alpha)$$
$$\delta_\alpha(\boldsymbol{x},\boldsymbol{x}') = \exp\bigl(-\kappa D_{\mathrm{KL}}(\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}') + \alpha\|\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}'))\bigr).$$

The ε-LSH-based mechanism Q_{LSHRR} provides $(\xi_\alpha,\delta_\alpha)$-PXDP, hence $(\xi_\alpha,\delta_\alpha)$-XDP.

While this guarantee is not a direct form of metric differential privacy, we can untangle the above for particular values of the metric $\mathbf{d}_\mathcal{X}$ of interest. In particular, in our experiments in §10.5, we apply Prop. 10.8 and numerically compute α from a constant δ using the convexity of D_{KL}.

10.4.3 LapLSH: Privacy Guarantees

Finally, we also show that LapLSH provides XDP. This is immediate from the fact that XDP is preserved under the post-processing by an LSH function.

PROPOSITION 10.9 (XDP of LapLSH)**.** Let $\varepsilon \in \mathbb{R}_{\geq 0}$. The $(\varepsilon, \mathbf{d}_{\mathcal{X}})$-LapLSH mechanism $Q_{\mathsf{Lap}H}$ with a κ-bit LSH function H provides $(\varepsilon \cdot \mathbf{d}_{\mathcal{X}}, 0)$-XDP. Hence the $(\varepsilon, \mathbf{d}_{\mathcal{X}})$-LapLSH mechanism Q_{LapLSH} wrt a distribution $D^{\kappa}_{\mathcal{H}}$ of the κ-bit LSH functions also provides $(\varepsilon \cdot \mathbf{d}_{\mathcal{X}}, 0)$-XDP.

10.5 Experimental Evaluation

In this section we present our experimental results on 2 medium-sized datasets for the application of LSH to friend-matching using the random-projection-based hashing. Our analysis involves a comparison of LSHRR with LapLSH from various aspects (utility, time complexity, and general applicability). In order to evaluate the usefulness of our mechanisms, we additionally evaluate the utility of vanilla LSH as compared with a true nearest neighbour search to establish utility of the overall mechanisms, not simply the noise-adding components.

10.5.1 Experimental Setup

As already discussed, our problem of interest is *privacy-preserving friend matching* using the local model of differential privacy. Recall that in this scenario, each user u is represented as a (real-valued) vector of attributes, ie., $u \in \mathbb{R}^n$. Each user applies to u a noisy mechanism which incorporates an LSH, and passes the noisy result to a curator, whose goal is to provide recommendations to each user (represented as u) based on their nearest k neighbours wrt an appropriate distance measure $\mathbf{d}_{\mathcal{X}}$ over users.

We compare the two privacy mechanisms introduced in this chapter:

1. *LSHRR Mechanism*: For each user we apply *κ-bit LSH* to the attribute vector according to a precomputed (shared) hash function and subsequently apply *ε-bitwise randomised response* (Def. 10.3.7) to the computed hash. The privacy guarantee is computed according to Prop. 10.8, which gives a (ξ, δ)-XDP -style guarantee that depends on κ.

2. *LapLSH Mechanism*: For each user we first add multivariate Laplace noise parametrised by ε to the attribute vector before applying LSH in order to generate noisy hash values. Our implementation of the multivariate Laplace follows that described in [12]; namely, we generate additive noise by constructing a unit vector uniformly at random over the unit n-sphere, scaled by a random value generated from the gamma distribution with shape n and scale $1/\varepsilon$.

Comparing Privacy and Utility

We compare the utility loss of each mechanism wrt a comparable privacy guarantee, namely the overall ε for the mechanism; for LapLSH this is $\varepsilon d_{\text{euc}}(\boldsymbol{x}, \boldsymbol{x}')$ and for LSHRR it is $\xi_{\alpha}(\boldsymbol{x}, \boldsymbol{x}')$ from Prop. 10.8. However, as the privacy guarantee of LSHRR depends on $\mathbf{d}_{\theta}$ and that of LapLSH depends on d_{euc}, we cannot simply compare the ε guarantees directly; we also need to compare

the privacy guarantee wrt the distances between users. Hence we make use of the relationship between the Euclidean and cosine distances for normalised vectors $\boldsymbol{x}, \boldsymbol{x}'$:

$$d_{\text{euc}}(\boldsymbol{x}, \boldsymbol{x}') \;=\; \sqrt{2 - 2\cos(\pi \cdot \mathbf{d}_\theta(\boldsymbol{x}, \boldsymbol{x}'))}\,.$$

Since ξ_α depends on α and $\mathbf{d}_\theta(\boldsymbol{x}, \boldsymbol{x}')$, we perform comparisons against various reasonable ranges of these variables to properly evaluate the range of utility loss. The LSHRR privacy guarantee also includes a δ component which we fix to a constant value.

For utility, for each user represented as a data point x_0 in the original space, we compare the average distance of the k returned nearest neighbours to x_0 using the noise-adding mechanism, versus the average distance of the k *true* nearest neighbours to x_0, ie., for mechanism A, the *utility loss* wrt user x_0 is:

$$\mathcal{U}_A(x_0) \;:=\; {}^1\!/_k \sum_{x \in N} \mathbf{d}_\theta(x_0, x) \;-\; {}^1\!/_k \sum_{x \in T} \mathbf{d}_\theta(x_0, x). \tag{10.1}$$

where N is the set of nearest neighbours returned by A, and T is the set of true nearest neighbours. We can then compute the utility loss of mechanism A as the average utility loss over all users. Notice that we use angular distance $\mathbf{d}_\theta$ as our distance measure, closely related to *cosine distance* which has been established as a good measure of similarity in previous work [154].

We prefer this measure to recall and precision measures as the LSH algorithm can return many neighbours with the same distance (ie. in the same hash bucket) and our approximate algorithm determines the choice of neighbours at random (a reasonable choice given the usefulness of the output is determined by how similar the neighbours are to the original point x_0).

Datasets

For our experiments we use the following two datasets:

MovieLens. The MovieLens 100k dataset [173] contains 943 users with ratings across 1682 movies, with ratings ranging from 1 to 5. We generated rating vectors of size 100, 500 and 1000 for each user by selecting only the top rated movies from the 4 most rated genres. Unrated movies were assigned a value of 0, and vectors were normalised to length 1. [8]

Foursquare. The Foursquare dataset (Global-scale Check-in Dataset with User Social Networks) [174] contains 90048627 check-ins by 2733324 users on POIs all over the world. We extracted 107091 POIs in New York and 88063 users who have visited at least one POI in New York. We sorted POIs by visit-count and generated user vectors for the 100, 500 and 1000 most-visited POIs, for a reduced dataset consisting of 1000 users. We again normalised vectors to length 1.

Note that the datasets were reduced in size in order to perform an overall evaluation against a true nearest neighbour search, which was too computationally expensive to run on the full-size datasets.

For both datasets, we computed the k nearest neighbours (wrt the angular distance $\mathbf{d}_\theta$) for each user for $k = 1, 5, 10$ using standard nearest neighbour search (ie. pairwise comparisons

[8] This step is required for the comparison with LapLSH.

over all ratings vectors). We refer to these data as the True Nearest neighbours; these are used to evaluate the performance of standard LSH and, by extension, the LapLSH and LSHRR mechanisms. The distributions of True Nearest neighbour distances are shown in Figure 10.2.

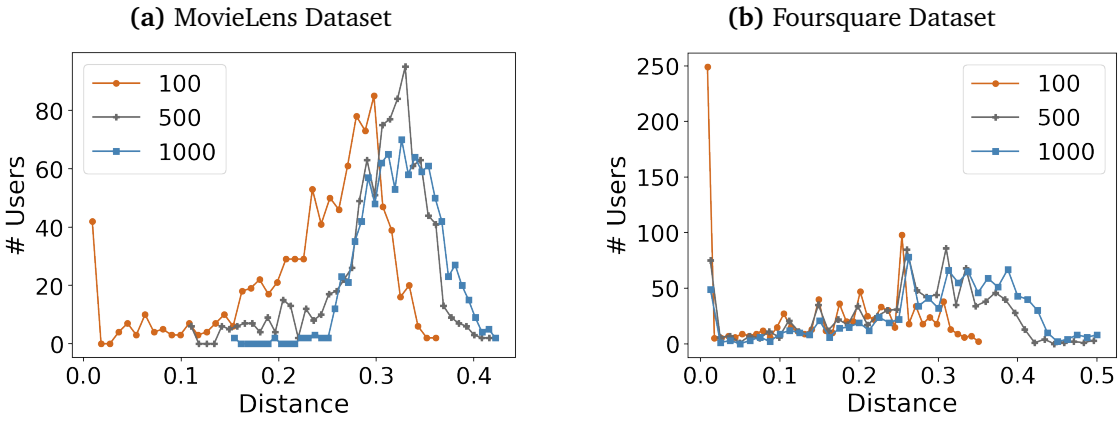


Figure 10.2: Distributions of angular distances $\mathbf{d}_\theta$ to nearest neighbour for $k = 1$ for each user, plotted for vectors with dimensions 100, 500 and 1000. The distance 0.5 represents orthogonal vectors; ie., having no items in common.

The Foursquare dataset has a high proportion of users with $\mathbf{d}_\theta < 0.1$, whereas the MovieLens dataset shows nearest neighbours are skewed towards an average of between 0.25 and 0.35 with few users having nearest neighbours closer than $\mathbf{d}_\theta = 0.1$. These distances are important for computing the privacy guarantee, which depends on the *true* distance between users. We therefore use distances between 0.05 and 0.25 in our experiments, noting that the privacy guarantee serves to protect nearby users. A privacy guarantee for $\mathbf{d}_\theta = 0.5$ would require protecting all users (even users which have no items in common), which would result in no utility for the user.

10.5.2 Results

Performance of vanilla LSH

We performed a baseline comparison of LSH against True Nearest neighbours to establish the utility of vanilla LSH. κ-bit LSH was implemented (for $\kappa = 10, 20, 50$) using the *random-projection-based hashing* described in §10.2.1, since this provides a guarantee wrt the angular distance between vectors. A (uniformly random) n-dimensional normal vector $\boldsymbol{r}$ can be constructed by selecting each component from a standard normal distribution; this is done for each of the κ hashes to generate the LSH hash function H. The same function H is used to encode each user's ratings vector into a κ-bit string.

For each user we then computed their k nearest neighbours for $k = 1, 5, 10$ using the Hamming distance on bitstrings. Where multiple neighbours shared the same distance (ie. they were in the same hash bucket) we chose a nearest neighbour at random.

These experiments were repeated 30 times to account for variations due to the randomness in the computations. The results for utility loss for LSH against True Nearest neighbours are

shown in Table 10.1. For the same vector dimension n, utility consistently improves for increasing bitstring length, and also as the number of nearest neighbours k increases. This is consistent with the observation that the bitstring encodes information about the vector. We notice that for higher vector dimensions, the utility loss is less pronounced as κ varies. This is because LSH is a dimensionality reduction of dimensions $n - \kappa$, which will clearly be less accurate for larger n and fixed κ.

Table 10.1: Values for average utility loss for LSH (compared with True Nearest neighbours).

		Length of bitstring (κ)			
n	k	10	20	30	50
100	1	0.139	0.097	0.074	0.053
100	5	0.119	0.087	0.068	0.048
100	10	0.109	0.082	0.065	0.046
500	1	0.117	0.096	0.081	0.063
500	5	0.101	0.084	0.071	0.056
500	10	0.092	0.076	0.065	0.052
1000	1	0.096	0.083	0.073	0.060
1000	5	0.084	0.073	0.065	0.053
1000	10	0.079	0.068	0.060	0.049

Performance of Privacy Mechanisms

We now compare the performance of LapLSH and LSHRR against LSH. We implemented both mechanisms as described in §10.5.1. The same LSH hash function H was used for comparing vanilla LSH, LapLSH and LSHRR. To compute the overall (ξ, δ)-XDP guarantee as per Prop. 10.8, we fixed $\delta = 0.01$ and varied $\mathbf{d}_\theta$ from 0.05 to 0.25 [9] to obtain corresponding values of α [10]. These are shown in Table 10.2. We then used values of ε ranging from 0.001 to 3 for the randomised response mechanism of LSHRR in order to generate ξ_α values in the range 0.1 to 15.

We plotted the utility loss of LSHRR against vanilla LSH for various values of $\mathbf{d}_\theta$ since the performance of our mechanism is bounded by that of LSH; the plot for vector dimension 1000 is shown in Figure 10.3. Note that we only show comparisons of LSHRR against LSH for simplicity of presentation; we will next compare the LSHRR and LapLSH mechanisms.

Finally we compared the utility loss of LSHRR against LapLSH for various vector dimensions and values of k (see Figure 10.5). We observe that LSHRR outperforms LapLSH across all experiments, although performance is comparable for small ε. LSHRR significantly outperforms LapLSH for smaller bitstring length ($\kappa = 10$), smaller distance ($\mathbf{d}_\theta = 0.05$) and higher vector dimensions ($n = 1000$). We consider this is because LapLSH is affected by the 'curse of dimensionality' – the total amount of noise added to the vector is very large in high-dimensional space

[9] We chose values of $\mathbf{d}_\theta$ corresponding to actual distances observed for true nearest neighbours as described in §10.5.1.

[10] $\mathbf{d}_\theta$ is the distance measure corresponding to $\mathbf{d}_\chi$ in §10.4.2

Table 10.2: Values of α given an angular distance $\mathbf{d}_\theta$ between inputs and a length κ of bitstring. These are used to compute ξ_α in the LSHRR guarantee of Prop. 10.8.

	Length of bitstring (κ)			
$\mathbf{d}_\theta$	10	20	30	50
0.05	0.31111	0.2028	0.15866	0.11713
0.1	0.3766	0.25209	0.19988	0.14979
0.2	0.44181	0.30509	0.24546	0.18683
0.25	0.45747	0.31977	0.25866	0.19796
0.3	0.46544	0.32908	0.26749	0.20573
0.4	0.46266	0.33474	0.27462	0.21313
0.5	0.43792	0.32553	0.26969	0.21123

(a) MovieLens Dataset with 1000 dimensional vectors

(b) Foursquare Dataset with 1000 dimensional vectors

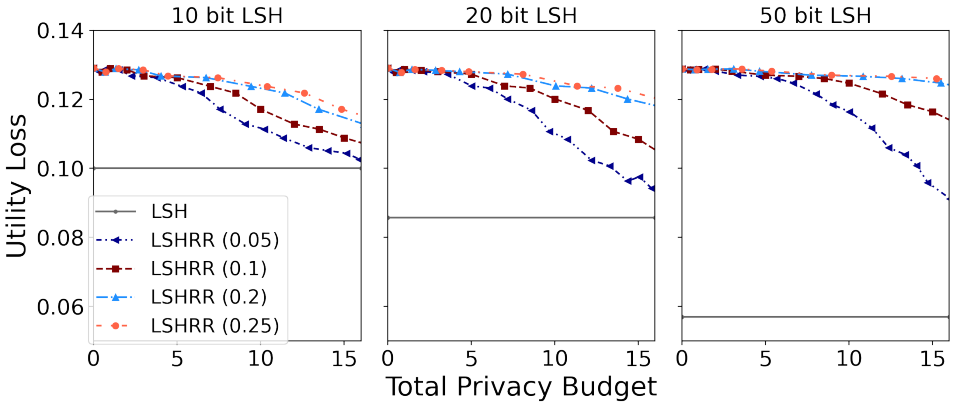


Figure 10.3: Utility loss vs total privacy budget ξ for LSHRR vs LSH for 1000-dimensional vectors. Note that $\mathbf{d}_\theta$ is indicated in brackets in the legend. LSHRR performance does not vary significantly with bitstring length, and is closer to LSH performance for smaller bit lengths.

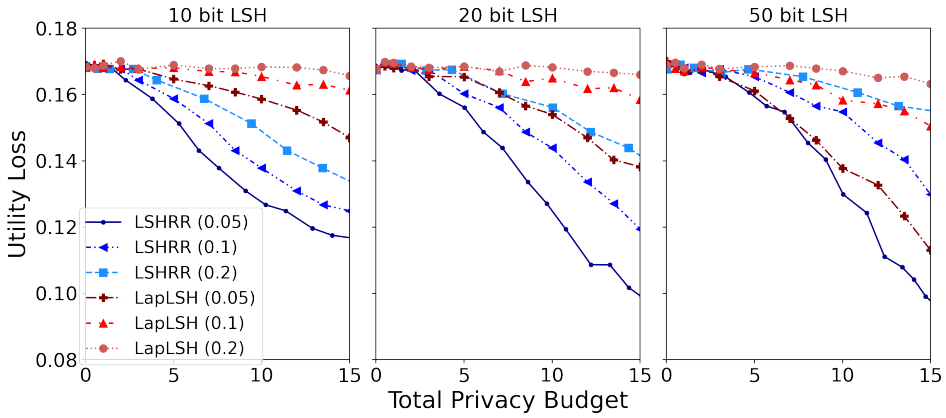
– as well as the issues outlined in §10.2.2. We conjecture that LSHRR will further outperform LapLSH for much higher dimensional vectors wrt close users (small $\mathbf{d}_\theta$).

Figure 10.5 also shows that when the total privacy budget ξ is around 2, LSHRR achieves lower utility loss than a uniformly randomly generated hash (ie., the LSHRR when the total

privacy budget is 0). The LSHRR achieves much lower utility loss when the total privacy budget is around 5. We can interpret the value of total privacy budget in terms of the flip probability in the RR (random response mechanism). For example, when we use the 20-bit hash, the total privacy budget of 5 for $\mathbf{d}_\theta = 0.05$ corresponds to the case in which the RR flips each bit of the hash with probability approx. 0.27. Therefore, we flip around 5-bits on average out of 20-bits in this case.

We also note that the total privacy budget used in our experiments is much smaller than the privacy budget ε in the low privacy regime [175] when the input data are in a high-dimensional space. Specifically, Kairouz *et al.* [175] refer to $\varepsilon = \ln|\mathcal{X}|$ as a privacy budget in the low privacy regime, and claim that this value of ε is still reasonable; subsequent work (eg. [176–179]) also considers this value of ε. Since we deal with high-dimensional data, the privacy budget in the low privacy regime is very large in our experiments. For example, when we use the 1000-dimensional rating vector in the MovieLens dataset, the privacy budget in the low privacy regime is: $\varepsilon = \ln|\mathcal{X}| = \ln 5^{1000} = 1609$. The total privacy budget used in our experiments is much smaller than this value, and falls into the *medium privacy regime* [176, 179].

(a) MovieLens Dataset with 100 dimensional vectors



(b) MovieLens Dataset with 1000 dimensional vectors

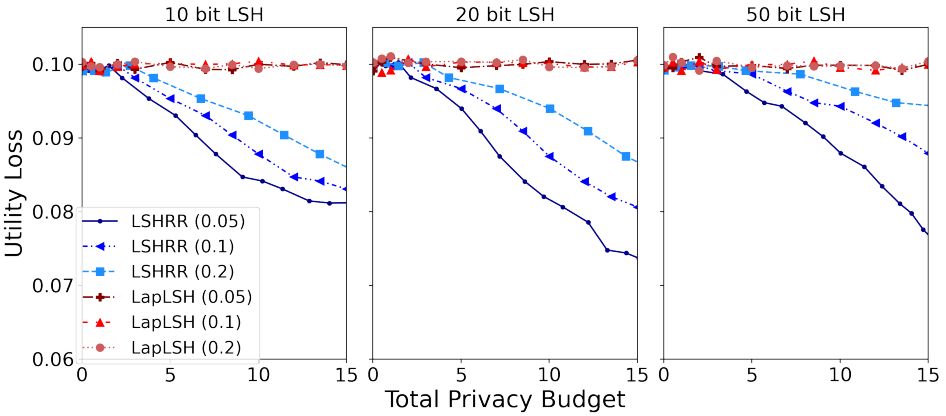


Figure 10.4: Utility loss vs total privacy budget ξ for LSHRR vs LapLSH for $k = 10$ on 100- and 1000-dimensional vectors. Note that $\mathbf{d}_\theta$ is indicated in brackets in the legend. LSHRR is much better than LapLSH for smaller $\mathbf{d}_\theta$ and larger n.

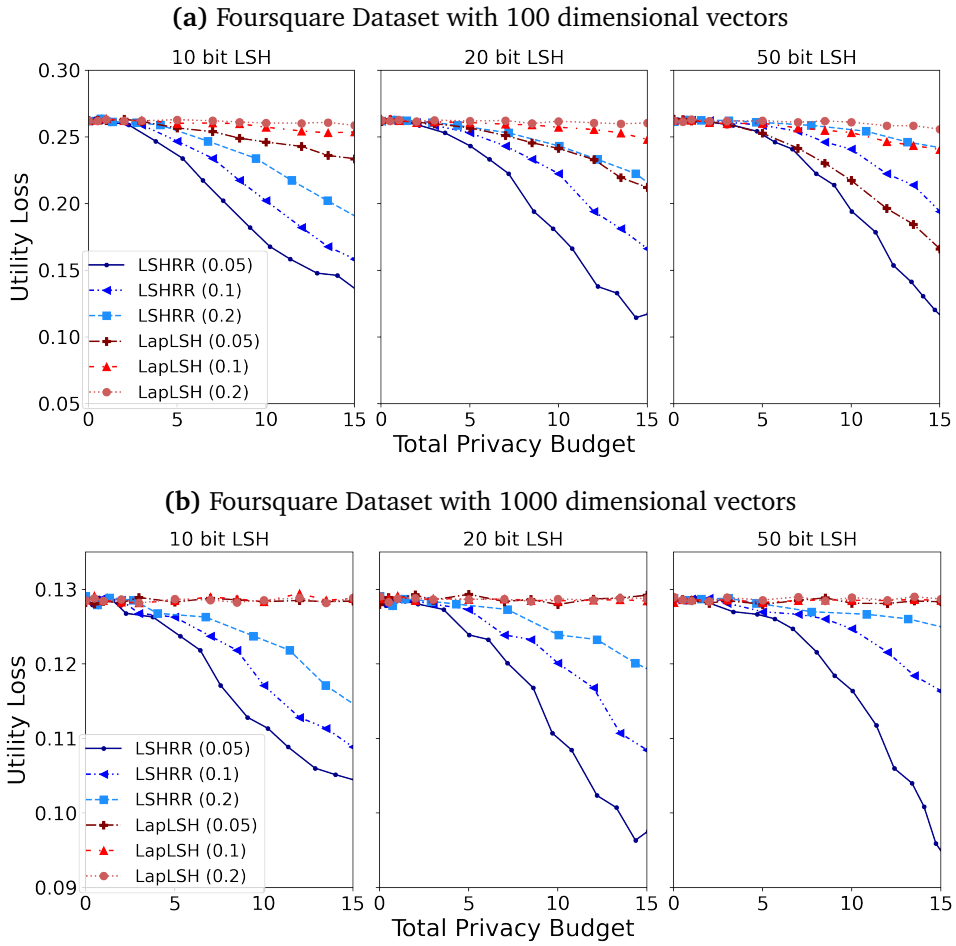


Figure 10.5: Utility loss vs total privacy budget ξ for LSHRR vs LapLSH for $k = 10$ on 100- and 1000-dimensional vectors. Note that $\mathbf{d}_\theta$ is indicated in brackets in the legend. LSHRR is much better than LapLSH for smaller $\mathbf{d}_\theta$ and larger n.

10.5.3 Discussion

For time complexity, LapLSH requires $O(n\kappa)$ operations (construction of n-dimensional noise, then κ-bit hashing), whereas the LSHRR mechanism requires $O(r\kappa)$ operations (κ-bit hashing on r non-zero elements followed by κ-randomised response). Therefore for large n, LSHRR is also significantly more efficient.

We note also that LSHRR is designed for the angular distance, whereas LapLSH is designed for the Euclidean distance, and our comparison of the two mechanisms (wrt the angular distance) was made possible by normalising the original vectors. In general, it is not possible to compare $\mathbf{d}$-private mechanisms designed for different metrics. As discussed in §10.2.2, we would consider the LSHRR mechanism more suitable for this space, and expect it to have better performance than LapLSH – which it does. We remark that the LapLSH mechanism in our experiments performed about as poorly as *random choice* for the 1000-dimensional vectors. Further, the LSHRR mechanism can be used with other metrics such as Jaccard, Earth Mover's and l_p by choosing a suitable hashing function.

In summary, our experiments show that LSHRR has significantly better utility than LapLSH, with particular performance differences observed for higher dimensional vectors, smaller bitstring lengths and smaller distances $\mathbf{d}_\theta$. The performance of LSHRR approaches the performance of LSH for smaller bitstring lengths, but has significantly lower utility than vanilla LSH for higher bitstring lengths. We conjecture that the dimensionality reduction of input vectors before adding perturbation noise would be useful to obtain better privacy-utility trade-offs for nearest neighbour search.

10.6 Conclusion

In this chapter we explored privacy-preserving LSH using a mechanism which adds random response noise to the output of an LSH, thus providing extended differential privacy for a wide range of metrics through an appropriate choice of hashing function. We first showed that LSH itself does not provide strong privacy guarantees and could result in complete privacy collapse in some situations. We then proved that our LSH-based mechanism provides concentrated/probabilistic versions of extended *DP*. Experimentally we demonstrated that our mechanism when applied to angular distance provides much higher utility than a naive, Laplace-based mechanism. These results matched our earlier reasoning which suggested that a Laplace-based mechanism would be ill-suited for the friend-matching domain since it is designed for the Euclidean distance metric and does not preserve angular distance. We leave as future work the theoretical exploration of properties of our concentrated-extended differential privacy definition, as well as further work to improve the performance of vanilla LSH, thus boosting the overall performance of our mechanism.

10.7 Chapter Notes

This chapter is unpublished work, however it is publicly available as an arXiv preprint [43]. Omitted proofs from this chapter can be found in Appendix §C.

Since the publication of our preprint, other works on a similar vein have appeared. Notably, Hu et al. [180] proposed the same mechanism (LSH + random response), also demonstrating that LSH is not privacy-preserving on its own, and showing that the mechanism has good utility wrt LSH on small bitstrings. Our work also examines the utility of LSH itself, showing that LSH on small bitstrings has poor utility and therefore the overall mechanism (LSH + random response) has poor overall utility.

Metric differential privacy and its applications

As noted earlier in this thesis, there have been a limited number of applications of metric differential privacy; in particular: location-based services [14, 181–186], document processing [12], linear and range queries [187], and linear queries in the centralised model [188]. These works focus on a specific metric such as the Euclidean metric, the l_1 metric, and the Earth Mover's metric, and cannot be applied to other metrics. In addition, most of the studies on metric *DP* have

studied low-dimensional data such as two-dimensional [14, 181–186, 188] and six-dimensional [187] data. One exception is the work in [12], which proposed the multivariate Laplace mechanism for 300-dimensional data points.

Privacy-preserving friend matching

A number of studies [155–158, 189–196] have been made on algorithms for privacy-preserving friend matching (or friend recommendation). Many of them [156, 158, 190–192, 194–196] use cryptographic techniques such as homomorphic encryption and secure multiparty computation. However, such techniques require high computational costs or focus on specific algorithms, and are not suitable for a more complicated calculation of distance such as the angular distance between two rating vectors.

The techniques in [155, 157, 189, 193] are based on perturbation. The mechanisms in [157, 189, 193] do not provide *DP* or its variant, whereas that in [155] provides *DP*. The technique in [155], however, is based on social graphs and cannot be applied to our setting, where a user's personal data is represented as a rating vector or visit-count vector. Moreover, *DP*-based friend matching in social graphs can require prohibitive trade-offs between utility and privacy [189, 197].

The same applies to friend matching based on rating vectors or visit-counts vectors. Specifically, it is possible to apply *DP* mechanisms (eg., [198, 199]) to each user's rating or visit-counts vector. However, this requires too much noise when the data point is in a high-dimensional space, since *DP* guarantees that a data point is indistinguishable from any data point in the data domain. Consequently, *DP*-based approaches need a very large privacy budget (eg., $\varepsilon \geq 250$ [198], $\varepsilon \geq 2 \times 10^4$ [199]) to provide high utility for a high-dimensional vector such as a rating or visit-counts vector. In contrast, our use of extended *DP* permits rigorous, meaningful privacy guarantees in high-dimensional spaces with good utility.

Privacy-preserving LSH

To our knowledge, the work in [200] is the only existing work (aside from the above-mentioned [180]) that proposed privacy-preserving LSH with formal guarantees of privacy. Specifically, the authors in [200] considered a problem of similarity search under the Jaccard similarity. For an integer $B \geq 2$, they used the range-B MinHash [201] κ times to transform a vector $x \in \mathcal{X}$ into a hash $h(x) \in [B]^\kappa$. Then they applied the B-ary randomised response [175] for each element of the hash. They proved that their mechanism provides *DP*.

Our work differs from [200] in the following ways. First, their algorithm is specific to the Jaccard similarity, and cannot be applied to other metrics such as the angular distance metric. Second, their algorithm provides *DP*, whereas our algorithm provides extended *DP*. Third, we compared LSHRR with LapLSH in the detailed evaluation, and our work provides a careful analysis of $\mathbf{d}$-privacy for both input data and hashes by taking into account the shared randomness between users (while the authors in [200] only analysed *DP* for hashes).

11

Conclusions and Future Work

In this thesis we have explored foundational questions about privacy and utility for differentially private systems using the framework of Quantitative Information Flow. We used the insights gained through the theoretical analysis to investigate three new applications of metric differential privacy to novel domains. We now bring together the ideas explored in this thesis to show how the core themes and questions posed in the introductory chapter have been addressed, and how these open up new avenues for future work.

Protection against adversarial attacks

We began by investigating the protection afforded by differential privacy systems in the context of arbitrary adversarial threats, modelled in QIF as *average-case* attacks. Although in differential privacy it is normally the case that mechanisms are compared by "comparing the epsilons", we showed in Chapter 4 that mechanisms with a larger epsilon (ie. "less private") can provide better protection against certain adversaries than mechanisms which are "more private". This possibly surprising result is because the epsilon parameter does not in itself characterise a defence against all possible adversarial scenarios. Potential vulnerabilities still remain in spite of the application of differential privacy. An added difficulty – made apparent in the metric differential privacy formulation – is that mechanisms designed for different metrics are not usually comparable, since the *meaning* of epsilon is determined by the metric in question. This causes additional problems when it comes to experimental design, as we noted in Chapter 8 and Chapter 10, where compromises had to be made in order that mechanisms could be fairly compared.

A more robust method for comparison would be to consider how different mechanisms protect against particular adversarial threats – ie. a comparison under various *operational* scenarios.

In Chapter 9 we gave a model for an adversarial threat – the Bayes vulnerability function, modelling a machine learning adversary that guesses the secret in one try – and provided some simple reasoning about how a privacy mechanism could provide protection against this adversary. Further work is needed in this area to model, for example, author-identification attacks – which have been enacted in real-world scenarios – and concretely describe the guarantees provided by differential privacy in these contexts. The work on locality sensitive hashing in Chapter 10 would benefit from a threat-focussed analysis, as the (metric-based) privacy guarantees in this scenario were complex to present and compare with other mechanisms found in the literature. We leave this analysis to future work.

We note also that the idea of modelling Bayesian adversaries alongside differential privacy has been advocated before; indeed this was a core contribution of the Pufferfish framework introduced by Kifer et al. [25]. We agree that there is a need to incorporate Bayesian threat models into reasoning about differential privacy so that privacy systems can be meaningfully compared, giving decision-makers more transparency about the sorts of threats that their systems can defend against. An advantage of the Bayesian/QIF approach is that it is worst-case (in the sense of the *optimal* adversary), and therefore upper bounds the expected success of the best possible machine learning attack, for example.

Finally, we observe that a robust comparison of systems with respect to adversarial threats would not only permit comparisons of differential privacy systems designed for different metrics, but also systems which do not implement differential privacy at all, such as k-anonymous mechanisms or encrypted data releases. QIF provides a general-purpose framework within which these threats can be studied and which could be used to fill this gap. We note that QIF has already been used to examine privacy threats against encrypted databases [202]; we leave to future work the continuation of this line of research on adversarial threats to privacy, and the comparison of different privacy models with regard to particular threats.

Managing the privacy-utility balance and choosing epsilon

The privacy-utility trade-off is well-known to be one of the key concerns in the design of privacy systems, and our work in Chapter 5 shows why this is the case. In that chapter we showed that privacy and utility cannot be optimised simultaneously, and indeed mechanisms which optimise for one – either privacy, or utility – lose control over the other. In general we have argued for a *utility-focussed* approach, which allows for greater control over the useful information released by a mechanism. This is essentially the purpose of the data release, the benefits of which need to be weighed against the potential harm caused by the leakage of information. While Dwork et al.'s original approach was to give plausible deniability to individuals, recent extensions and applications of differential privacy do not do this. In these cases the privacy that can be guaranteed is determined by the *correlation* between the useful information released and the secrets to be kept private, which depends on the particular dataset in question since, as was explained in Chapter 5.

This then provides some guidelines as to how we might answer two important questions in the design of privacy workflows: *where to add the noise?* and *how to add the noise?*. These

questions were encountered and addressed in our experimental works in the following ways:

i. In Chapter 8 we observed that a non-standard mechanism designed for the metric on the domain can provide better utility, although its privacy guarantee is different-by-design. Thus, adding noise according to the *utility metric* on the domain proved to be a better strategy for improving statistical utility.

ii. In Chapter 9 we constructed a *utility-focussed* mechanism which was designed for a utility metric between documents (the Earth Mover's distance) and therefore provided strong guarantees regarding the useful release of topic information. The privacy guarantee simply meant providing indistinguishability of authors within some radius; in this case the manipulation of epsilon would be done to provide a desirable radius of indistinguishability – ie. to provide adequate *privacy* protection. The privacy afforded in this case is determined experimentally, since it is determined by the correlation between the sensitive and the useful information in the dataset.

iii. In Chapter 10 we considered two possible locations for the noise-adding mechanism in the workflow, opting again for a utility-first approach – designing the mechanism for the appropriate metric – which we showed experimentally produced better results for a fixed privacy guarantee. Again the utility obtainable is determined by the correlation in the dataset, as well as the correlation induced by the hashing function.

The work of these chapters still leaves open questions about how to design privacy workflows and where to add noise. For example, our work on locality sensitive hashing could be extended to an exploration of scenarios in which metric-based mechanisms exist on the original domain (eg. Euclidean metrics). In this case, it may be easier to reason about privacy, since it is possible to design mechanisms for the correct metric for each workflow under consideration. However, the decision about where to add noise may be less clear – it may be influenced by details such as the particulars of the dataset, the accuracy of the embedding function and the dimensionality of the space. We leave the exploration of this problem to future work.

Finally, although we argue for a utility-focussed approach, this makes reasoning about privacy harder – as borne out by our experimental work – and thus emphasises the need for robust methods to compare privacy in different systems, as raised in the previous section. We leave open as a future goal the exploration of utility-focussed mechanisms for which privacy can be compared robustly – either relying on comparable metrics, or using robust comparisons wrt operational scenarios defined by adversarial threats.

Reasoning about utility

In Chapter 6 we began an investigation of optimality, showing that Bayesian consumers described in the literature can be modelled using the QIF notion of uncertainty and subsequently that optimality can be understood in terms of refinement. This connection between *utility* and *refinement* provides an important bridge between differential privacy and QIF, and enables the

use of QIF results to reason about utility in arbitrary privacy workflows. In future work we envisage investigating utility for arbitrary data release mechanisms in which the utility measure of interest is not known beforehand; this occurs, for example, in data releases designed for mining by machine learners whose goal is to discover useful information in the dataset. Such scenarios include complex datasets such as geo-spatial trajectories, for which privacy-preserving data releases are desired but the utility to be gained from these datasets is not clearly understood. The use of refinement-based reasoning about utility also promises to assist in reasoning about privacy-utility trade-offs in privacy workflows, as well as in comparing mechanisms for utility. This is particularly important in scenarios where mechanisms need to be designed in the absence of real data, as was the case in the 2020 US Census data release [29].

Our investigation of optimality for the Laplace mechanism in Chapter 7 underscores the need to extend existing results – made under assumptions of the statistical database context – to new domains for which metric differential privacy applies. Future work in this area could involve extending our results on discrete domains to continuous domains with respect to other metrics of interest. We expect the continuation of this work to provide a robust methodology for reasoning about optimality in arbitrary workflows which may arise in future applications of metric differential privacy.

Further applications of metric differential privacy

In the final part of this thesis we considered the application of metric differential privacy to three new domains of interest, guided by the analyses of the previous chapters. In all three applications we emphasised that designing mechanisms for *utility* should be the core concern of the data curator, and that privacy can be reasoned about either directly (as when local mechanisms are used, in the example of Chapter 8) or through the correlations in the workflow (as in author obfuscation of Chapter 9 and locality sensitive hashing of Chapter 10). We consider our explorations to be first steps towards solving these problems using the perspective of metric differential privacy.

Finally, we remark that there are many interesting and diverse applications of differential privacy that are still to be explored, and the investigations in this thesis have opened up new ways of exploring these ideas. In particular, the application of privacy to trajectory data appears to be of significant interest, and we would consider metric differential privacy to be suitable in this regard. The analyses and examples in this thesis could provide some guidelines towards an acceptable approach to this problem: firstly, considering possible metrics for utility, and reasoning about what sort of utility could be achievable for different consumers under general assumptions about classes of utility measures; secondly, measuring privacy by the sorts of attacks to be protected against; and finally, setting epsilon to achieve some level of indistinguishability for which such attacks can be reasonably diffused.

References

[1] T. Dalenius. *Towards a methodology for statistical disclosure control*. Statistik Tidskrift **15**, 429 (1977).

[2] L. Sweeney. *k-anonymity: A model for protecting privacy*. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems **10**(5), 557 (2002).

[3] A. Narayanan and V. Shmatikov. *Robust de-anonymization of large sparse datasets*. In *Proceedings of the Symposium on Security and Privacy (SP)*, pp. 111–125 (IEEE, 2008).

[4] S. Garfinkel, J. M. Abowd, and C. Martindale. *Understanding database reconstruction attacks on public data*. Communications of the ACM **62**(3), 46 (2019).

[5] C. Dwork, F. McSherry, K. Nissim, and A. Smith. *Calibrating noise to sensitivity in private data analysis*. In *Theory of cryptography conference*, pp. 265–284 (Springer, 2006).

[6] S. Warner. *Randomized response: A survey technique for eliminating evasive answer bias*. Journal of the American Statistical Association **60**, 63Ð69 (1965).

[7] C. Dwork, A. Roth, *et al. The algorithmic foundations of differential privacy*. Foundations and Trends® in Theoretical Computer Science **9**(3–4), 211 (2014).

[8] N. Fernandes, M. Dras, and A. McIver. *Processing text for privacy: An information flow perspective*. In K. Havelund, J. Peleska, B. Roscoe, and E. P. de Vink, eds., *Formal Methods - 22nd International Symposium, FM 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 15-17, 2018, Proceedings*, vol. 10951 of *Lecture Notes in Computer Science*, pp. 3–21 (Springer, 2018). URL https://doi.org/10.1007/978-3-319-95582-7_1.

[9] L. Fan. *Image pixelization with differential privacy*. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pp. 148–162 (Springer, 2018).

[10] M. A. P. Chamikara, P. Bertok, I. Khalil, D. Liu, and S. Camtepe. *Privacy preserving face recognition utilizing differential privacy*. Computers & Security **97**, 101951 (2020).

[11] B. Weggenmann and F. Kerschbaum. *Syntf: Synthetic and differentially private term frequency vectors for privacy-preserving text mining*. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 305–314 (2018).

[12] N. Fernandes, M. Dras, and A. McIver. *Generalised differential privacy for text document processing*. In *Principles of Security and Trust - 8th International Conference, POST 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings*, pp. 123–148 (2019). URL https://doi.org/10.1007/978-3-030-17138-4_6.

[13] K. Jiang, D. Shao, S. Bressan, T. Kister, and K.-L. Tan. *Publishing trajectories with differential privacy guarantees*. In *Proceedings of the 25th International Conference on Scientific and Statistical Database Management*, pp. 1–12 (2013).

[14] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. *Geo-indistinguishability: Differential privacy for location-based systems*. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pp. 901–914 (2013).

[15] K. Chatzikokolakis, M. E. Andrés, N. E. Bordenabe, and C. Palamidessi. *Broadening the scope of differential privacy using metrics*. In *International Symposium on Privacy Enhancing Technologies Symposium*, pp. 82–102 (Springer, 2013).

[16] D. Desfontaines and B. Pejó. *Sok: Differential privacies*. Proceedings on Privacy Enhancing Technologies **2020**(2), 288 (2020).

[17] S. Salamatian, A. Zhang, F. du Pin Calmon, S. Bhamidipati, N. Fawaz, B. Kveton, P. Oliveira, and N. Taft. *Managing your private and public data: Bringing down inference attacks against your privacy*. IEEE Journal of Selected Topics in Signal Processing **9**(7), 1240 (2015).

[18] M. A. Rahman, T. Rahman, R. Laganière, N. Mohammed, and Y. Wang. *Membership inference attack against differentially private deep learning model.* Trans. Data Priv. **11**(1), 61 (2018).

[19] B. Jayaraman, L. Wang, D. Evans, and Q. Gu. *Revisiting membership inference under realistic assumptions*. arXiv preprint arXiv:2005.10881 (2020).

[20] N. Z. Gong and B. Liu. *You are who you know and how you behave: Attribute inference attacks via users' social friends and behaviors*. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pp. 979–995 (2016).

[21] V. Bindschaedler, P. Grubbs, D. Cash, T. Ristenpart, and V. Shmatikov. *The tao of inference in privacy-protected databases*. Proceedings of the VLDB Endowment **11**(11), 1715 (2018).

[22] S. Seneviratne, A. Seneviratne, P. Mohapatra, and A. Mahanti. *Predicting user traits from a snapshot of apps installed on a smartphone*. ACM SIGMOBILE Mobile Computing and Communications Review **18**(2), 1 (2014).

[23] J. P. Achara, G. Acs, and C. Castelluccia. *On the unicity of smartphone applications*. In *Proceedings of the 14th ACM Workshop on Privacy in the Electronic Society*, pp. 27–36 (2015).

[24] J. Hamm. *Minimax filter: Learning to preserve privacy from inference attacks*. The Journal of Machine Learning Research **18**(1), 4704 (2017).

[25] D. Kifer and A. Machanavajjhala. *Pufferfish: A framework for mathematical privacy definitions*. ACM Trans. Database Syst. **39**(1) (2014). URL https://doi.org/10.1145/2514689.

[26] T. Humphries, M. Rafuse, L. Tulloch, S. Oya, I. Goldberg, and F. Kerschbaum. *Differentially private learning does not bound membership inference*. arXiv preprint arXiv:2010.12112 (2020).

[27] J. Hsu, M. Gaboardi, A. Haeberlen, S. Khanna, A. Narayan, B. C. Pierce, and A. Roth. *Differential privacy: An economic method for choosing epsilon*. In *IEEE 27th Computer Security Foundations Symposium, CSF 2014, Vienna, Austria, 19-22 July, 2014*, pp. 398–410 (IEEE Computer Society, 2014). URL https://doi.org/10.1109/CSF.2014.35.

[28] C. Dwork, N. Kohli, and D. Mulligan. *Differential privacy in practice: Expose your epsilons!* Journal of Privacy and Confidentiality **9**(2) (2019).

[29] S. L. Garfinkel, J. M. Abowd, and S. Powazek. *Issues encountered deploying differential privacy*. In *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*, pp. 133–137 (2018).

[30] Z. Ji, Z. C. Lipton, and C. Elkan. *Differential privacy and machine learning: a survey and review*. arXiv preprint arXiv:1412.7584 (2014).

[31] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. *Deep learning with differential privacy*. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318 (2016).

[32] M. S. Alvim, M. E. Andrés, K. Chatzikokolakis, and C. Palamidessi. *On the relation between differential privacy and quantitative information flow*. In *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, pp. 60–76 (2011). URL https://doi.org/10.1007/978-3-642-22012-8_4.

[33] M. S. Alvim, M. E. Andrés, K. Chatzikokolakis, P. Degano, and C. Palamidessi. *On the information leakage of differentially-private mechanisms*. Journal of Computer Security **23**(4), 427 (2015). URL https://doi.org/10.3233/JCS-150528.

[34] M. S. Alvim, K. Chatzikokolakis, A. McIver, C. Morgan, C. Palamidessi, and G. Smith. *The science of quantitative information flow* (2019).

[35] K. Chatzikokolakis, N. Fernandes, and C. Palamidessi. *Comparing systems: Max-case refinement orders and application to differential privacy*. In *Proc. CSF* (IEEE Press, 2019).

[36] K. Chatzikokolakis, N. Fernandes, and C. Palamidessi. *Refinement orders for quantitative information flow and differential privacy*. Journal of Cybersecurity and Privacy **1**(1), 40 (2021).

[37] D. Kifer and A. Machanavajjhala. *No free lunch in data privacy*. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, SIGMOD Õ11, p. 193Ð204 (Association for Computing Machinery, New York, NY, USA, 2011). URL https://doi.org/10.1145/1989323.1989345.

[38] M. S. Alvim, N. Fernandes, A. McIver, and G. H. Nunes. *On privacy and accuracy in data releases (invited paper)*. In I. Konnov and L. Kovács, eds., *31st International Conference on Concurrency Theory, CONCUR 2020, September 1-4, 2020, Vienna, Austria (Virtual Conference)*, vol. 171 of *LIPIcs*, pp. 1:1–1:18 (Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020). URL https://doi.org/10.4230/LIPIcs.CONCUR.2020.1.

[39] A. Ghosh, T. Roughgarden, and M. Sundararajan. *Universally utility-maximizing privacy mechanisms*. SIAM Journal on Computing **41**(6), 1673 (2012).

[40] H. Brenner and K. Nissim. *Impossibility of differentially private universally optimal mechanisms*. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pp. 71–80 (IEEE, 2010).

[41] N. Fernandes, A. McIver, and C. Morgan. *The laplace mechanism has optimal utility for differential privacy over continuous queries*. In *ACM/IEEE Symposium on Logic in Computer Science (LICS 2021) (to appear)* (IEEE, 2021).

[42] N. Fernandes, L. Kacem, and C. Palamidessi. *Utility-preserving privacy mechanisms for counting queries*. In M. Boreale, F. Corradini, M. Loreti, and R. Pugliese, eds., *Models, Languages, and Tools for Concurrent and Distributed Programming - Essays Dedicated to Rocco De Nicola on the Occasion of His 65th Birthday*, vol. 11665 of *Lecture Notes in Computer Science*, pp. 487–495 (Springer, 2019). URL https://doi.org/10.1007/978-3-030-21485-2_27.

[43] N. Fernandes, Y. Kawamoto, and T. Murakami. *Locality sensitive hashing with extended differential privacy*. CoRR **abs/2010.09393** (2020). 2010.09393, URL https://arxiv.org/abs/2010.09393.

[44] A. McIver and C. Morgan. *Sums and lovers: Case studies in security, compositionality and refinement*. In A. Cavalcanti and D. Dams, eds., *Proc. FM '09*, vol. 5850 of *LNCS* (Springer, 2009).

[45] A. McIver and C. Morgan. *The thousand-and-one cryptographers*. In C. B. Jones, A. Roscoe, and K. R. Wood, eds., *Reflections on the Work of C.A.R. Hoare* (Springer, 2010).

[46] K. Chatzikokolakis, C. Palamidessi, and P. Panangaden. *Anonymity protocols as noisy channels*. Inf. Comput. **206**(2-4), 378 (2008).

[47] A. McIver, T. Rabehaja, R. Wen, and C. Morgan. *Privacy in elections: How small is "small"?* Journal of information security and applications **36**, 112 (2017).

[48] C. M. Jeremy Gibbons, Annabelle Mciver and T. Schrijvers. *Quantitative information flow with monads in haskell*. In *Foundations of Probabilistic Programming* (CUP, 2019). To appear.

[49] D. M. Smith and G. Smith. *Tight bounds on information leakage from repeated independent runs*. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pp. 318–327 (IEEE, 2017).

[50] B. Köpf and D. Basin. *An information-theoretic model for adaptive side-channel attacks*. In *Proceedings of the 14th ACM conference on Computer and communications security*, CCS '07, pp. 286–296 (ACM, New York, NY, USA, 2007). URL http://doi.acm.org/10.1145/1315245.1315282.

[51] A. McIver, C. C. Morgan, and T. Rabehaja. *Program algebra for quantitative information flow*. Journal of Logical and Algebraic Methods in Programming **106**, 55 (2019).

[52] A. McIver and C. Morgan. *Abstraction, Refinement and Proof for Probabilistic Systems*. Tech Mono Comp Sci (Springer, New York, 2005). URL http://www.cse.unsw.edu.au/~carrollm/arp/.

[53] M. S. Alvim, K. Chatzikokolakis, P. Degano, and C. Palamidessi. *Differential privacy versus quantitative information flow*. CoRR **abs/1012.4250** (2010).

[54] C. Shannon. *A mathematical theory of communication*. Bell System Technical Journal **27**, 379 (1948).

[55] G. Smith. *On the foundations of quantitative information flow*. In L. de Alfaro, ed., *Proc. 12th International Conference on Foundations of Software Science and Computational Structures (FoSSaCS '09)*, vol. 5504 of *Lecture Notes in Computer Science*, pp. 288–302 (2009).

[56] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and G. Smith. *Measuring information leakage using generalized gain functions*. In *2012 IEEE 25th Computer Security Foundations Symposium*, pp. 265–279 (IEEE, 2012).

[57] M. S. Alvim, K. Chatzikokolakis, A. McIver, C. Morgan, C. Palamidessi, and G. Smith. *Axioms for information leakage*. In *2016 IEEE 29th Computer Security Foundations Symposium (CSF)*, pp. 77–92 (IEEE, 2016).

[58] J. Massey. *Guessing and entropy*. In *Proc. IEEE International Symposium on Information Theory*, p. 204 (1994).

[59] K. Chatzikokolakis, C. Palamidessi, and P. Panangaden. *On the Bayes risk in information-hiding protocols*. Journal of Computer Security **16**(5), 531 (2008).

[60] M. S. Alvim, K. Chatzikokolakis, A. McIver, C. Morgan, C. Palamidessi, and G. Smith. *Additive and multiplicative notions of leakage, and their capacities*. In *IEEE 27th Computer Security Foundations Symposium, CSF 2014, Vienna, Austria, 19-22 July, 2014*, pp. 308–322 (IEEE, 2014). URL http://dx.doi.org/10.1109/CSF.2014.29.

[61] A. McIver, C. Morgan, G. Smith, B. Espinoza, and L. Meinicke. *Abstract channels and their robust information-leakage ordering*. In M. Abadi and S. Kremer, eds., *Principles of Security and Trust - Third International Conference, POST 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, vol. 8414 of *Lecture Notes in Computer Science*, pp. 83–102 (Springer, 2014).

[62] A. McIver, L. Meinicke, and C. Morgan. *A Kantorovich-monadic powerdomain for information hiding, with probability and nondeterminism*. In *Proc. LiCS 2012* (2012).

[63] P. Kairouz, S. Oh, and P. Viswanath. *The composition theorem for differential privacy*. In *International conference on machine learning*, pp. 1376–1385 (PMLR, 2015).

[64] J. Dong, D. Durfee, and R. Rogers. *Optimal differential privacy composition for exponential mechanisms*. In *International Conference on Machine Learning*, pp. 2597–2606 (PMLR, 2020).

[65] S. P. Kasiviswanathan and A. Smith. *A note on differential privacy: Defining resistance to arbitrary side information*. CoRR abs/0803.3946 (2008).

[66] C. Dwork. *Differential privacy*. In *Proc. 33rd International Colloquium on Automata, Languages, and Programming (ICALP 2006)*, pp. 1–12 (2006).

[67] S. P. Kasiviswanathan and A. Smith. *On the 'semantics' of differential privacy: A bayesian formulation*. Journal of Privacy and Confidentiality **6**(1) (2014).

[68] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. *What can we learn privately?* SIAM Journal on Computing **40**(3), 793 (2011).

[69] M. Hardt and K. Talwar. *On the geometry of differential privacy*. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pp. 705–714 (2010).

[70] G. Barthe and B. Köpf. *Information-theoretic bounds for differentially private mechanisms*. In *Proceedings of the 24th IEEE Computer Security Foundations Symposium (CSF)*, pp. 191–204 (IEEE Computer Society, 2011).

[71] H. R. Tiwary. *On the hardness of computing intersection, union and minkowski sum of polytopes*. Discrete & Computational Geometry **40**(3), 469 (2008).

[72] K. Fukuda, T. M. Liebling, and C. Lütolf. *Extended convex hull*. Comput. Geom. **20**, 13 (2000).

[73] A. Ghosh and A. Roth. *Selling privacy at auction*. In *Proceedings of the 12th ACM Conference on Electronic Commerce*, EC '11, pp. 199–208 (ACM, New York, NY, USA, 2011). URL http://doi.acm.org/10.1145/1993574.1993605.

[74] H. Yasuoka and T. Terauchi. *Quantitative information flow — verification hardness and possibilities*. In *Proc. 23rd IEEE CSF Symp.*, pp. 15–27 (2010).

[75] P. Malacaria. *Algebraic foundations for quantitative information flow*. Mathematical Structures in Computer Science **25**(2), 404 (2015). URL https://doi.org/10.1017/S0960129513000649.

[76] D. Clark, S. Hunt, and P. Malacaria. *Quantitative information flow, relations and polymorphic types.* J. Logic and Computation **15**(2), 181 (2005).

[77] P. Malacaria. *Assessing security threats of looping constructs*. In M. Hofmann and M. Felleisen, eds., *Proceedings of the 34th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2007)*, pp. 225–235 (ACM, 2007). URL http://doi.acm.org/10.1145/1190216.1190251.

[78] J. Landauer and T. Redmond. *A lattice of information*. In *Proc. 6th IEEE Computer Security Foundations Workshop (CSFW'93)*, pp. 65–70 (1993).

[79] B. Jayaraman and D. Evans. *Evaluating differentially private machine learning in practice*. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, p. 18951912 (USENIX Association, USA, 2019).

[80] C. Dwork and M. Naor. *On the difficulties of disclosure prevention in statistical databases or the case for differential privacy*. Journal of Privacy and Confidentiality **2**(1) (2010).

[81] A. Narayanan, H. Paskov, N. Z. Gong, J. Bethencourt, E. Stefanov, E. C. R. Shin, and D. Song. *On the feasibility of internet-scale author identification*. In *2012 IEEE Symposium on Security and Privacy*, pp. 300–314 (IEEE, 2012).

[82] X. He, A. Machanavajjhala, and B. Ding. *Blowfish privacy: Tuning privacy-utility trade-offs using policies*. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 1447–1458 (2014).

[83] T. Zhu, P. Xiong, G. Li, and W. Zhou. *Correlated differential privacy: Hiding information in non-iid data set*. IEEE Transactions on Information Forensics and Security **10**(2), 229 (2014).

[84] C. Liu, S. Chakraborty, and P. Mittal. *Dependence makes you vulnerable: Differential privacy under dependent tuples.* In *NDSS*, vol. 16, pp. 21–24 (2016).

[85] Úlfar Erlingsson, V. Pihur, and A. Korolova. *RAPPOR: Randomized aggregatable privacy-preserving ordinal response*. In *Proc. CCS*, pp. 1054–1067 (2014).

[86] V. Erin Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou. *Deep hashing for compact binary codes learning*. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2475–2483 (2015).

[87] M. Gupte and M. Sundararajan. *Universally optimal privacy mechanisms for minimax agents*. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 135–146 (2010).

[88] P. Kairouz, S. Oh, and P. Viswanath. *Extremal mechanisms for local differential privacy*. The Journal of Machine Learning Research **17**(1), 492 (2016).

[89] F. Koufogiannis, S. Han, and G. J. Pappas. *Optimality of the laplace mechanism in differential privacy*. arXiv preprint arXiv:1504.00065 (2015).

[90] A. McIver, L. Meinicke, and C. Morgan. *Compositional closure for Bayes Risk in probabilistic noninterference*. In *Proceedings of the 37th international colloquium conference on Automata, languages and programming: Part II*, vol. 6199 of *ICALP'10*, pp. 223–235 (Springer, Berlin, Heidelberg, 2010). URL http://portal.acm.org/citation.cfm?id=1880999.1881023.

[91] J. D.-F. Jordi Soria-Comas. *Optimal data-independent noise for differential privacy*. Information Sciences **250**, 200 (2012).

[92] Q. Geng, P. Kairouz, S. Oh, and P. Viswanath. *The staircase mechanism in differential privacy*. IEEE Journal of Selected Topics in Signal Processing **9**(7), 1176 (2015).

[93] E. L. Lawler. *Combinatorial optimization: networks and matroids* (Courier Corporation, 2001).

[94] S. T. Rachev and L. Rüschendorf. *Mass Transportation Problems: Volume I: Theory*, vol. 1 (Springer Science & Business Media, 1998).

[95] P. Meyer-Nieberg. *Banach Lattices* (Springer-Verlag, 1991).

[96] E. B. Wilson. *First and second laws of error*. Journal of the American Statistical Association **18**(143), 841 (1923).

[97] M. M. Pai and A. Roth. *Privacy and mechanism design*. SIGecom Exch. **12**(1), 8 (2013). URL https://doi.org/10.1145/2509013.2509016.

[98] I. Dinur and K. Nissim. *Revealing information while preserving privacy*. In F. Neven, C. Beeri, and T. Milo, eds., *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 9-12, 2003, San Diego, CA, USA*, pp. 202–210 (ACM, 2003). URL https://doi.org/10.1145/773153.773173.

[99] Y. Wang, X. Wu, and L. Wu. *Differential privacy preserving spectral graph analysis*. In J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu, eds., *Advances in Knowledge Discovery and Data Mining*, pp. 329–340 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2013).

[100] N. Phan, X. Wu, H. Hu, and D. Dou. *Adaptive laplace mechanism: Differential privacy preservation in deep learning*. In *2017 IEEE International Conference on Data Mining (ICDM)*, pp. 385–394 (2017).

[101] R. Agrawal, R. Srikant, and D. Thomas. *Privacy preserving olap*. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pp. 251–262 (2005).

[102] A. P. Dempster, N. M. Laird, and D. B. Rubin. *Maximum likelihood from incomplete data via the em algorithm*. Journal of the Royal Statistical Society: Series B (Methodological) **39**(1), 1 (1977).

[103] L. Kacem and C. Palamidessi. *Geometric noise for locally private counting queries*. In *Proceedings of the 13th Workshop on Programming Languages and Analysis for Security*, pp. 13–16 (2018).

[104] E. ElSalamouny and C. Palamidessi. *Generalized iterative bayesian update and applications to mechanisms for privacy protection*. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 490–507 (IEEE, 2020).

[105] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and A. Pazii. *Local differential privacy on metric spaces: Optimizing the trade-off with utility*. In *31st IEEE Computer Security Foundations Symposium, CSF 2018, Oxford, United Kingdom, July 9-12, 2018*, pp. 262–267 (IEEE Computer Society, 2018). URL https://doi.org/10.1109/CSF.2018.00026.

[106] M. Potthast, F. Rangel, M. Tschuggnall, E. Stamatatos, P. Rosso, and B. Stein. *Overview of PAN'17: Author Identification, Author Profiling, and Author Obfuscation*. In G. Jones, S. Lawless, J. Gonzalo, L. Kelly, L. Goeuriot, T. Mandl, L. Cappellato, and N. Ferro, eds., *Experimental IR Meets Multilinguality, Multimodality, and Interaction. 7th International Conference of the CLEF Initiative (CLEF 17)* (Springer, Berlin Heidelberg New York, 2017).

[107] M. Koppel, J. Schler, and S. Argamon. *Authorship attribution in the wild*. Language Resources and Evaluation **45**(1), 83 (2011).

[108] J. Tetreault, D. Blanchard, and A. Cahill. *A Report on the First Native Language Identification Shared Task*. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 48–57 (Association for Computational Linguistics, Atlanta, Georgia, 2013). URL http://www.aclweb.org/anthology/W13-1706.

[109] S. Malmasi and M. Dras. *Native Language Identification With Classifier Stacking and Ensembles*. Computational Linguistics **44**(3), 403 (2018). https://doi.org/10.1162/coli_a_00323, URL https://doi.org/10.1162/coli_a_00323.

[110] M. Koppel, S. Argamon, and A. R. Shimoni. *Automatically categorizing written texts by author gender*. Literary and Linguistic Computing **17**(4), 401 (2002). /oup/backfile/content_public/journal/dsh/17/4/10.1093/llc/17.4.401/2/170401.pdf, URL http://dx.doi.org/10.1093/llc/17.4.401.

[111] Francisco Manuel, Rangel Pardo, P. Rosso, M. Potthast, and B. Stein. *Overview of the 5th Author Profiling Task at PAN 2017: Gender and Language Variety Identification in Twitter*. In L. Cappellato, N. Ferro, L. Goeuriot, and T. Mandl, eds., *Working Notes Papers of the CLEF 2017 Evaluation Labs*, vol. 1866 of *CEUR Workshop Proceedings* (CLEF and CEUR-WS.org, 2017). URL http://ceur-ws.org/Vol-1866/.

[112] T. Global. *Native Language Identification (NLI) Establishes Nationality of Sony's Hackers as Russian*. Tech. rep., Taia Global, Inc. (2014).

[113] M. Koppel and Y. Winter. *Determining if two documents are written by the same author*. JASIST **65**(1), 178 (2014). URL http://dx.doi.org/10.1002/asi.22954.

[114] G. Grimmett and D. Stirzaker. *Probability and Random Processes* (Oxford Science Publications, 1992), second ed.

[115] T. Mikolov, K. Chen, G. Corrado, and J. Dean. *Efficient estimation of word representations in vector space*. CoRR **abs/1301.3781** (2013).

[116] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger. *From word embeddings to document distances*. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 957–966 (2015).

[117] Y. Rubner, C. Tomasi, and L. J. Guibas. *The earth mover's distance as a metric for image retrieval*. International journal of computer vision **40**(2), 99 (2000).

[118] D. König. *Theorie der endlichen und unendlichen Graphen* (Akademische. Verlags Gesellschaft, Leipzig, 1936).

[119] A. Boisbunon. *The class of multivariate spherically symmetric distributions*. Université de Rouen, Technical Report **5**, 2012 (2012).

[120] D. P. Kroese, T. Taimre, and Z. I. Botev. *Handbook of monte carlo methods*, vol. 706 (John Wiley & Sons, 2013).

[121] G. Marsaglia *et al. Choosing a point from the surface of a sphere*. The Annals of Mathematical Statistics **43**(2), 645 (1972).

[122] S. Seidman. *Authorship Verification Using the Imposters Method*. In *Working Notes for CLEF 2013 Conference* (2013). URL http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-Seidman2013.pdf.

[123] M. Khonji and Y. Iraqi. *A Slightly-modified GI-based Author-verifier with Lots of Features (ASGALF)*. In *Working Notes for CLEF 2014 Conference* (2014). URL http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-KonijEt2014.pdf.

[124] U. Sapkota, S. Bethard, M. Montes, and T. Solorio. *Not All Character N-grams Are Created Equal: A Study in Authorship Attribution*. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language*

Technologies, pp. 93–102 (Association for Computational Linguistics, Denver, Colorado, 2015). URL http://www.aclweb.org/anthology/N15-1010.

[125] S. Ruder, P. Ghaffari, and J. G. Breslin. *Character-level and Multi-channel Convolutional Neural Networks for Large-scale Authorship Attribution*. CoRR **abs/1609.06686** (2016). 1609.06686, URL http://arxiv.org/abs/1609.06686.

[126] O. Halvani, C. Winter, and L. Graner. *Authorship Verification based on Compression-Models*. CoRR **abs/1706.00516** (2017). 1706.00516, URL http://arxiv.org/abs/1706.00516.

[127] N. Potha and E. Stamatatos. *An Improved Impostors Method for Authorship Verification*. In *Proceedings of CLEF 2017* (2017).

[128] M. Potthast, S. Braun, T. Buz, F. Duffhauss, F. Friedrich, J. Gülzow, J. Köhler, W. Lötzsch, F. Müller, M. Müller, R. Paßmann, B. Reinke, L. Rettenmeier, T. Rometsch, T. Sommer, M. Träger, S. Wilhelm, B. Stein, E. Stamatatos, and M. Hagen. *Who Wrote the Web? Revisiting Influential Author Identification Research Applicable to Information Retrieval*. In N. Ferro, F. Crestani, M.-F. Moens, J. Mothe, F. Silvestri, G. Di Nunzio, C. Hauff, and G. Silvello, eds., *Advances in Information Retrieval. 38th European Conference on IR Research (ECIR 16)*, vol. 9626 of *Lecture Notes in Computer Science*, pp. 393–407 (Springer, Berlin Heidelberg New York, 2016).

[129] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. *Enriching word vectors with subword information*. arXiv preprint arXiv:1607.04606 (2016).

[130] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. *Bag of tricks for efficient text classification*. arXiv preprint arXiv:1607.01759 (2016).

[131] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning* (Springer-Verlag, 2009), 2nd ed.

[132] S. Wan, M. Dras, R. Dale, and C. Paris. *Improving Grammaticality in Statistical Sentence Generation: Introducing a Dependency Spanning Tree Algorithm with an Argument Satisfaction Model*. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pp. 852–860 (Association for Computational Linguistics, 2009). URL http://aclweb.org/anthology/E09-1097.

[133] Y. Zhang and S. Clark. *Discriminative Syntax-Based Word Ordering for Text Generation*. Computational Linguistics **41**(3), 503 (2015). https://doi.org/10.1162/COLI_a_00229, URL https://doi.org/10.1162/COLI_a_00229.

[134] E. Hasler, F. Stahlberg, M. Tomalin, A. de Gispert, and B. Byrne. *A Comparison of Neural Models for Word Ordering*. In *Proceedings of the 10th International Conference on Natural Language Generation*, pp. 208–212 (Association for Computational Linguistics, 2017). URL http://aclweb.org/anthology/W17-3531.

[135] O. Feyisetan, T. Diethe, and T. Drake. *Leveraging hierarchical representations for preserving privacy and utility in text*. In *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 210–219 (IEEE, 2019).

[136] G. Kacmarcik and M. Gamon. *Obfuscating document stylometry to preserve author anonymity*. In *ACL*, pp. 444–451 (2006).

[137] A. W. McDonald, S. Afroz, A. Caliskan, A. Stolerman, and R. Greenstadt. *Use fewer instances of the letter "i": Toward writing style anonymization*. In *International Symposium on Privacy Enhancing Technologies Symposium*, pp. 299–318 (Springer, 2012).

[138] G. Karadzhov, T. Mihaylova, Y. Kiprov, G. Georgiev, I. Koychev, and P. Nakov. *The case for being average: A mediocrity approach to style masking and author obfuscation*. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pp. 173–185 (Springer, 2017).

[139] O. Bakhteev and A. Khazov. *Author Masking using Sequence-to-Sequence Models—Notebook for PAN at CLEF 2017*. In L. Cappellato, N. Ferro, L. Goeuriot, and T. Mandl, eds., *CLEF 2017 Evaluation Labs and Workshop – Working Notes Papers, 11-14 September, Dublin, Ireland* (CEUR-WS.org, 2017). URL http://ceur-ws.org/Vol-1866/.

[140] D. Castro, R. Ortega, and R. Muñoz. *Author Masking by Sentence Transformation—Notebook for PAN at CLEF 2017*. In L. Cappellato, N. Ferro, L. Goeuriot, and T. Mandl, eds., *CLEF 2017 Evaluation Labs and Workshop – Working Notes Papers, 11-14 September, Dublin, Ireland* (CEUR-WS.org, 2017). URL http://ceur-ws.org/Vol-1866/.

[141] M. Mansoorizadeh, T. Rahgooy, M. Aminiyan, and M. Eskandari. *Author Obfuscation using WordNet and Language Models—Notebook for PAN at CLEF 2016*. In K. Balog, L. Cappellato, N. Ferro, and C. Macdonald, eds., *CLEF 2016 Evaluation Labs and Workshop – Working Notes Papers, 5-8 September, Évora, Portugal* (CEUR-WS.org, 2016). URL http://ceur-ws.org/Vol-1609/.

[142] R. Shetty, B. Schiele, and M. Fritz. *A4nt: Author attribute anonymity by adversarial training of neural machine translation*. In *27th USENIX Security Symposium*, pp. 1633–1650 (USENIX Association, 2018).

[143] C. Emmery, E. Manjavacas, and G. Chrupała. *Style obfuscation by invariance*. arXiv preprint arXiv:1805.07143 (2018).

[144] Y. Li, T. Baldwin, and T. Cohn. *Towards Robust and Privacy-preserving Text Representations*. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 25–30 (Association for Computational Linguistics, 2018). URL http://aclweb.org/anthology/P18-2005.

[145] M. Coavoux, S. Narayan, and S. B. Cohen. *Privacy-preserving Neural Representations of*

Text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1–10 (Association for Computational Linguistics, Brussels, Belgium, 2018). URL http://www.aclweb.org/anthology/D18-1001.

[146] R. Shokri and V. Shmatikov. *Privacy-preserving deep learning*. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pp. 1310–1321 (ACM, New York, NY, USA, 2015). URL http://doi.acm.org/10.1145/2810103.2813687.

[147] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. *Deep Learning with Differential Privacy*. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pp. 308–318 (ACM, New York, NY, USA, 2016). URL http://doi.acm.org/10.1145/2976749.2978318.

[148] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. *Learning Differentially Private Recurrent Language Models*. In *International Conference on Learning Representations* (2018). URL https://openreview.net/forum?id=BJ0hF1Z0b.

[149] D. Abril, G. Navarro-Arribas, and V. Torra. *On the declassification of confidential documents*. In *International Conference on Modeling Decisions for Artificial Intelligence*, pp. 235–246 (Springer, 2011).

[150] C. Cumby and R. Ghani. *A machine learning based system for semi-automatically redacting documents*. In *Proceedings of the Twenty-Third Conference on Innovative Applications of Artificial Intelligence (IAAI)* (2011).

[151] B. Anandan, C. Clifton, W. Jiang, M. Murugesan, P. Pastrana-Camacho, and L. Si. *t-plausibility: Generalizing words to desensitize text*. Trans. Data Privacy **5**(3), 505 (2012).

[152] D. Sánchez and M. Batet. *C-sanitized: A privacy model for document redaction and sanitization*. Journal of the Association for Information Science and Technology **67**(1), 148 (2016).

[153] M. Rodriguez-Garcia, M. Batet, and D. Sánchez. *Semantic noise: privacy-protection of nominal microdata through uncorrelated noise addition*. In *Tools with Artificial Intelligence (ICTAI), 2015 IEEE 27th International Conference on*, pp. 1106–1113 (IEEE, 2015).

[154] C. C. Aggarwal. *Recommender Systems* (Springer, 2016).

[155] L. Chen and P. Zhu. *Preserving the privacy of social recommendation with a differentially private approach*. In *Proc. SmartCity*, pp. 780–785 (IEEE, 2015).

[156] D. Li, Q. Lv, L. Shang, and N. Gu. *Efficient privacy-preserving content recommendation for online social communities*. Neurocomputing **219**, 440 (2017).

[157] C. Liu and P. Mittal. *Linkmirage: Enabling privacy-preserving analytics on social relationships*. In *Proc. NDSS* (2016).

[158] X. Ma, J. Ma, H. Li, Q. Jiang, and S. Gao. *Armor: A trust-based privacy-preserving framework for decentralized friend recommendation in online social networks*. Future Generation Computer Systems **79**, 82 (2018).

[159] X. Su and T. M. Khoshgoftaar. *A survey of collaborative filtering techniques*. Advances in artificial intelligence **2009** (2009).

[160] F. Ricci, L. Rokach, and B. Shapira. *Introduction to recommender systems handbook*. In *Recommender systems handbook*, pp. 1–35 (Springer, 2011).

[161] P. Indyk and R. Motwani. *Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality*. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pp. 604–613 (ACM, New York, NY, USA, 1998). URL http://doi.acm.org/10.1145/276698.276876.

[162] M. S. Charikar. *Similarity estimation techniques from rounding algorithms*. In *Proc. STOC*, pp. 380–388 (2002).

[163] A. Andoni, P. Indyk, T. Laarhoven, I. Razenshteyn, and L. Schmidt. *Practical and optimal lsh for angular distance*. In *Proc. NIPS*, pp. 1–9 (2015).

[164] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. *Min-wise independent permutations*. Journal of Computer and System Sciences **60**, 630 (2000).

[165] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. *Locality-sensitive hashing scheme based on p-stable distributions*. In *Proc. SCG*, pp. 253–262 (2004).

[166] R. Chow, M. A. Pathak, and C. Wang. *A practical system for privacy-preserving collaborative filtering*. In *2012 IEEE 12th International Conference on Data Mining Workshops*, pp. 547–554 (IEEE, 2012).

[167] A. Aghasaryan, M. Bouzid, D. Kostadinov, M. Kothari, and A. Nandi. *On the use of lsh for privacy preserving personalization*. In *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 362–371 (IEEE, 2013).

[168] L. Qi, X. Zhang, W. Dou, and Q. Ni. *A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data*. IEEE Journal on Selected Areas in Communications **35**(11), 2616 (2017).

[169] S. R. Ganta, S. P. Kasiviswanathan, and A. Smith. *Composition attacks and auxiliary information in data privacy*. In *Proc. KDD*, pp. 265–273 (ACM, 2008).

[170] C. Dwork and G. N. Rothblum. *Concentrated differential privacy*. CoRR **abs/1603.01887** (2016). 1603.01887.

[171] C. Dwork, G. N. Rothblum, and S. P. Vadhan. *Boosting and differential privacy*. In *Proc. FOCS*, pp. 51–60 (2010).

[172] D. M. Sommer, S. Meiser, and E. Mohammadi. *Privacy loss classes: The central limit theorem in differential privacy*. PoPETs **2019**(2), 245 (2019).

[173] MovieLens 100K Dataset. https://grouplens.org/datasets/movielens/100k/ (accessed in 2020).

[174] D. Yang, B. Qu, J. Yang, and P. Cudre-Mauroux. *Revisiting user mobility and social relationships in LBSNs: A hypergraph embedding approach*. In *Proc.WWW*, pp. 2147–2157 (2019).

[175] P. Kairouz, K. Bonawitz, and D. Ramage. *Discrete distribution estimation under local privacy*. In *Proc. ICML*, pp. 2436–2444 (2016).

[176] J. Acharya, Z. Sun, and H. Zhang. *Hadamard response: Estimating distributions privately, efficiently, and with little communication*. In *Proc. AISTATS*, pp. 1120–1129 (2019).

[177] T. Murakami and Y. Kawamoto. *Utility-optimized local differential privacy mechanisms for distribution estimation*. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pp. 1877–1894 (2019).

[178] S. Wang, L. Huang, P. Wang, Y. Nie, H. Xu, W. Yang, X.-Y. Li, and C. Qiao. *Mutual information optimally local private discrete distribution estimation*. CoRR **abs/1607.08025** (2016). URL https://arxiv.org/abs/1607.08025.

[179] M. Ye and A. Barga. *Optimal schemes for discrete distribution estimation under local differential privacy*. In *Proc. ISIT*, pp. 759—763 (2017).

[180] H. Hu, G. Dobbie, Z. Salcic, M. Liu, J. Zhang, L. Lyu, and X. Zhang. *Differentially private locality sensitive hashing based federated recommender system*. Concurrency and Computation: Practice and Experience p. e6233 (2021).

[181] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and A. Pazii. *Metric-based local differential privacy for statistical applications*. CoRR **abs/1805.01456** (2018). 1805.01456.

[182] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. *Optimal geo-indistinguishable mechanisms for location privacy*. In *Proc. CCS*, pp. 251–262 (2014).

[183] K. Chatzikokolakis, E. ElSalamouny, and C. Palamidessi. *Efficient utility improvement for location privacy*. PoPETs **2017**(4), 308 (2017).

[184] S. Oya, C. Troncoso, and F. Pérez-González. *Back to the drawing board: Revisiting the design of optimal location privacy-preserving mechanisms*. In *Proc. CCS*, pp. 1959–1972 (2017).

[185] R. Shokri. *Privacy games: Optimal user-centric data obfuscation*. Proceedings on Privacy Enhancing Technologies (PoPETs) **2015**(2), 299 (2015).

[186] Y. Kawamoto and T. Murakami. *On the anonymization of differentially private location obfuscation*. In *Proc. ISITA*, pp. 159–163 (IEEE, 2018).

[187] Z. Xiang, B. Ding, X. He, and J. Zhou. *Linear and range counting under metric-based local differential privacy*. In *Proc. ISIT*, pp. 908–913 (2020).

[188] P. Kamalaruban, V. Perrier, H. J. Asghar, and M. A. Kaafar. *Not all attributes are created equal: d_x-private mechanisms for linear queries*. Proceedings on Privacy Enhancing Technologies (PoPETs) **2020**(1), 103 (2020).

[189] W. Brendel, F. Han, L. Marujo, L. Jie, and A. Korolova. *Practical privacy-preserving friend recommendations on social networks*. In *Proc. WWW*, pp. 111–112 (2018).

[190] H. Cheng, M. Qian, Q. Li, Y. Zhou, and T. Chen. *An efficient privacy-preserving friend recommendation scheme for social network*. IEEE Access **6**, 56018 (2018).

[191] W. Dong, V. Dave, L. Qiu, and Y. Zhang. *Secure friend discovery in mobile social networks*. In *Proc. InfoCom*, pp. 1647–1655 (IEEE, 2011).

[192] T. Guo, K. Dong, L. Wang, M. Yang, and J. Luo. *Privacy preserving profile matching for social networks*. In *Proc. CBD*, pp. 263–268 (IEEE, 2018).

[193] M. Li, N. Ruan, Q. Qian, H. Zhu, X. Liang, and L. Yu. *Spfm: Scalable and privacy-preserving friend matching in mobile clouds*. IEEE Internet of Things Journal **4**(2), 583 (2017).

[194] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, D. Boneh, *et al. Location privacy via private proximity testing.* In *Proc. NDSS*, vol. 11 (2011).

[195] B. K. Samanthula, L. Cen, W. Jiang, and L. Si. *Privacy-preserving and efficient friend recommendation in online social networks.* Trans. Data Privacy **8**(2), 141 (2015).

[196] H. Zhu, S. Du, M. Li, and Z. Gao. *Fairness-aware and privacy-preserving friend matching protocol in mobile social networks*. IEEE Transactions on Emerging Topics in Computing **1**(1), 192 (2013).

[197] A. Machanavajjhala, A. Korolova, and A. D. Sarma. *Personalized social recommendations - accurate or private?* Proc. VLDB **4**(7), 440 (2020).

[198] Z. Liu, Y.-X. Wang, and A. J. Smola. *Fast differentially private matrix factorization*. In *Proc. RecSys*, pp. 171–178 (2015).

[199] T. Murakami, K. Hamada, Y. Kawamoto, and T. Hatano. *Privacy-preserving multiple tensor factorization for synthesizing large-scale location traces*. CoRR **abs/1911.04226** (2019). URL https://arxiv.org/abs/1911.04226.

[200] M. Aumüller, A. Bourgeat, and J. Schmurr. *Differentially private sketches for jaccard similarity estimation*. CoRR **abs/2008.08134** (2020). URL https://arxiv.org/abs/2008.08134.

[201] P. Li and A. C. Konig. *b-bit minwise hashing*. In *Proc. WWW*, pp. 671–680 (2010).

[202] M. Jurado and G. Smith. *Quantifying information leakage of deterministic encryption*. In *Proceedings of the 2019 ACM SIGSAC Conference on Cloud Computing Security Workshop*, pp. 129–139 (2019).

Appendices

A

Proofs Omitted from Chapter 6

A.1 Results supporting §6.3.3

We begin by proving the properties of kernel mechanisms omitted from the main chapter.

> **Property 1.** If K is a kernel mechanism then there is no kernel mechanism K' st. $\lceil\Delta_{K'}\rceil \subset \lceil\Delta_K\rceil$ where $\Delta_K, \Delta_{K'}$ are hypers corresponding to mechanisms K, K' respectively.

Proof. Assume that there exists some $S \subset \lceil\Delta_K\rceil$ st. $S = \lceil\Delta_{K'}\rceil$ for some kernel mechanism K'. Then the posteriors in S must be linearly independent (Def. 6.3.2) and the uniform distribution can be expressed as a convex combination of inners of $\lceil\Delta_{K'}\rceil$ (Lem. 6.9). ie. We can write $\upsilon = \sum_i \lambda_i \delta^i$ for convex coefficients λ_i and $\delta_i \in \lceil\Delta_{K'}\rceil$. Similarly for Δ_K, we can write $\upsilon = \sum_j \kappa_j \gamma^j$ for convex coefficients κ_j and $\gamma^j \in \lceil\Delta_K\rceil$. But therefore we have:

$$
\begin{aligned}
& \textstyle\sum_i \lambda_i \delta^i - \sum_j \kappa_j \gamma^j = 0 \\
\Rightarrow \quad & \textstyle\sum_i \lambda_i \gamma^i - \sum_j \kappa_j \gamma^j = 0 && \text{“}\lceil\Delta_{K'}\rceil \subset \lceil\Delta_K\rceil\text{”} \\
\Rightarrow \quad & \textstyle\sum_i (\lambda_i - \kappa_i)\gamma^i + \sum_{k=j-i} \kappa_k \gamma^k = 0 && \text{“Rearranging”}
\end{aligned}
$$

Since the γ^j are linearly independent, this requires $\lambda_i = \kappa_i$ and $\kappa_k = 0$. ie. $\lceil\Delta_K\rceil = \lceil\Delta_{K'}\rceil$, which contradicts our assumption. Therefore no such S exists. □

> **Property 2.** Any vertex mechanism can be written (non-uniquely) as a convex sum of kernel mechanisms. Conversely, any convex sum of kernel mechanisms is a vertex mechanism.

Proof. For the converse, note that a convex sum of channels is computed by taking the union of the inners of all of the corresponding hypers and summing the corresponding outers. Since each inner is a vertex, the union of them forms a set of posteriors all of which are vertices. To check that it is a valid hyper we need the posteriors to average to the uniform distribution; since each hyper averages to υ, the convex sum does also. Therefore $\sum_i \lambda_i \Delta_i$ corresponds to a vertex mechanism.

For the forward direction, we describe an algorithm which rearranges a hyper Δ_M (corresponding to a vertex mechanism) into a convex sum $\lambda\Delta_K + (1-\lambda)\Delta_{M'}$ st. Δ_K corresponds to a kernel mechanism and $\Delta_{M'}$ corresponds to a vertex mechanism with $\lceil\Delta_{M'}\rceil \subset \lceil\Delta_M\rceil$. Denote by v_i the inners of $\lceil\Delta\rceil$. If the v_i are linearly independent then Δ already corresponds to a kernel mechanism and we are done. If the v_i are not linearly independent, then we can use Carathéodory's theorem to find a Δ_K and Δ_M. Note that $\upsilon = \sum_i p_i v_i$ (from Lem. 6.9) and so by Carathéodory's theorem, we can find an affinely independent subset $V \subset \lceil\Delta\rceil$ st. $\upsilon = \sum_j \lambda_j v_j$ for $v_j \in V$ and $\lambda_j \geq 0$, $\sum_j \lambda_j = 1$. However, the set V is also *linearly* independent. This follows because if there is some μ_i st. $\sum_i \mu_i v_i = \vec{0}$ then we have:

$$
\begin{array}{lll}
& \sum_i \mu_i v_i = \vec{0} & \\
\Rightarrow & \sum_i \sum_j \mu_i v[j]_i = 0 & \text{“Summing elements of each vector } v_i\text{”} \\
\Rightarrow & \sum_i \mu_i \sum_j v[j]_i = 0 & \text{“Rearranging”} \\
\Rightarrow & \sum_i \mu_i = 0 & \text{“}v_i \text{ are 1-summing vectors”} \\
\Rightarrow & \mu_i = 0 \quad \text{for each } i & \text{“Affine independence”}
\end{array}
$$

Therefore the v_i are linearly independent. This means we can choose $\lceil\Delta_K\rceil = V$, and we can uniquely write $\upsilon = \sum_i a_i v_i$ for $v_i \in V$ and convex coefficients a_i.

Now, scale Δ_K by the largest $\lambda > 0$ so that $\lambda a_i \leq p_i$ (where a_i, p_i are the coefficients of the same $v_i \in V$ in Δ_K, Δ respectively) and there is some j for which $\lambda a_j = p_j$. Define

$$\Delta_{M'} \quad := \quad \frac{\Delta - \lambda\Delta_K}{1-\lambda}\,.$$

Then $\Delta_{M'}$ is a valid hyper (since Δ and Δ_K both average to υ) and $\lceil\Delta_{M'}\rceil \subset \lceil\Delta\rceil$ (since our choice of λ cancels out one of the v_i). Hence we have $\Delta = \lambda\Delta_K + (1-\lambda)\Delta_{M'}$ as required.

We can repeat this algorithm for $\Delta_{M'}$, continuing until we are (inevitably) left with a kernel mechanism. □

PROPERTY 3. If K, K^* are kernel mechanisms then $K \not\sqsubseteq K^*$ and $K^* \not\sqsubseteq K$.

Proof. Let Δ, Δ^* be the corresponding hypers of K, K^* respectively. From Property 1 we cannot have $\lceil\Delta\rceil \subset \lceil\Delta^*\rceil$ or $\lceil\Delta^*\rceil \subset \lceil\Delta\rceil$. Therefore, since all posteriors in the supports of Δ, Δ^* are vertices, by convexity we know that $\mathbf{ch}\lceil\Delta\rceil$ cannot lie inside $\mathbf{ch}\lceil\Delta^*\rceil$ and vice versa. Thus, by Lem. 2.6, it follows that $K \not\sqsubseteq K^*$ and $K^* \not\sqsubseteq K$. □

Kernel mechanisms always exist

The proof that kernel mechanisms always exist (ie. it is always possible to find a linearly independent set of vertices that contains the uniform distribution) follows from the proof of Prop. 2 above. ie. By noting that vertex mechanisms always exist, and that they can always be decomposed into kernel mechanisms.

A.2 Results supporting §6.4

We first prove that for the space of $n > 2$ inputs there is no single minimal element under refinement. For this we first need to prove a technical lemma.

> **LEMMA B1.** Let (X, d) be a metric space and let $|X| > 2$. Then the space of d-hypers contains at least $2(n-1)$ vertices.

Proof. Pick any 3 inputs x_1, x_2, x_3. If these inputs are collinear (ie. $d(x_1, x_2) + d(x_2, x_3) = d(x_1, x_3)$) then the **d**-privacy constraints on (x_1, x_2) and (x_2, x_3) imply the constraints on (x_1, x_3), and so the 3 hyperplanes (corresponding to constraints in each direction) contribute only 4 vertices to the convex region. If the points are not collinear, then each pair (x_i, x_j) contributes 2 constraints, resulting in 6 constraint hyperplanes in total and 6 points in the convex space. The minimum number of vertices in the space therefore occurs when the set X is totally ordered (ie. maximising the number of linear, and thus inferred, constraints), and so we need only consider constraints on 'adjacent' vertices. In total this yields $2(n-1)$ vertices (corresponding to 2 constraints per adjacent pair). □

We now have the details in place to complete the sketch proof of the impossibility of universally optimal mechanisms for $n > 2$.

> **THEOREM 6.15** (Impossibility of Universally Optimal Mechanisms)**.** For $n > 2$ there are no universally optimal **d**-private mechanisms over n inputs.

Proof. By Lem. B1 we have that the space contains at least $2(n-1)$ vertices. Since this is larger than n for $n > 2$, we must have more than one kernel mechanism, and thus more than one vertex mechanism. The result follows. □

Characterisation of universal $\mathcal{L}$-optimality

We first need the following technical result.

> **LEMMA B2.** Given a loss function ℓ and a prior π, the minimum expected loss is realised on a vertex mechanism *and* on every kernel mechanism from which it can be derived.

Proof. Assume that the minimum expected loss is realised on the mechanism M. Then M is a

vertex mechanism (Cor. 6.12) and we can write it as $M = \sum_i \lambda_i K_i$ where K_i are kernel mechanisms (Prop. 2). We now reason

$$
\begin{aligned}
& U_\ell[\pi \triangleright M] \\
= \;& U_\ell[\pi \triangleright (\textstyle\sum_i \lambda_i K_i)] && \text{“Property 2”} \\
= \;& \textstyle\sum_i \lambda_i U_\ell[\pi \triangleright K_i] && \text{“Linearity of } U_\ell\text{”}
\end{aligned}
$$

Since $U_\ell[\pi \triangleright M] \leq U_\ell[\pi \triangleright C]$ for any other mechanism C, we must have $U_\ell[\pi \triangleright M] \leq U_\ell[\pi \triangleright K_i]$ for every K_i since these are valid mechanisms. This can only happen when $U_\ell[\pi \triangleright M] = U_\ell[\pi \triangleright K_i]$ for all K_i. Thus the minimum expected loss is realised on each kernel mechanism. □

We now complete the proof sketch for the characterisation of universally $\mathcal{L}$-optimal mechanisms.

> **THEOREM 6.16** (Characterisation of Universally $\mathcal{L}$-Optimal Mechanisms)**.** Every universally $\mathcal{L}$-optimal mechanism is a refinement of a convex combination of universally $\mathcal{L}$-optimal kernel mechanisms.

Proof. If M is universally $\mathcal{L}$-optimal then there is some vertex mechanism V such that $V \sqsubseteq M$ (Cor. 6.12) and so V must also be universally $\mathcal{L}$-optimal. By Prop. 2 this can be written as a convex sum of kernel mechanisms and by Lem. B2 these are also universally $\mathcal{L}$-optimal. □

A.3 Proofs omitted from §6.5.1

For completeness we provide the following proofs of the refinement properties of this section.

> **PROPERTY 4.** If $M \sqsubseteq_\ell M^*$ then $M \sqsubseteq_{(\ell * a)} M^*$ for $a \geq 0$.
>
> *Proof.* We reason as follows:
>
> $$
> \begin{aligned}
> & M \sqsubseteq_\ell M^* \\
> \Rightarrow \;& U_\ell[\pi \triangleright M] \leq U_\ell[\pi \triangleright M^*] && \text{“Expand”} \\
> \Rightarrow \;& \textstyle\sum_i a_i U_\ell(\delta^i) \leq \sum_j b_j U_\ell(\gamma^j) && \text{“Dual of Def. 2.2.5”} \\
> \Rightarrow \;& \textstyle\sum_i a_i a U_\ell(\delta^i) \leq \sum_j b_j a U_\ell(\gamma^j) && \text{“Arithmetic”} \\
> \Rightarrow \;& \textstyle\sum_i a_i U_{\ell * a}(\delta^i) \leq \sum_j b_j U_{\ell * a}(\gamma^j) && \text{“Eqn (6.8)”} \\
> \Rightarrow \;& M \sqsubseteq_{(\ell * a)} M^* && \text{“Definition”}
> \end{aligned}
> $$
>
> □

> **PROPERTY 5.** If $M \sqsubseteq_\ell M^*$ then $M \sqsubseteq_{(\ell + a)} M^*$ for $a \geq 0$.
>
> *Proof.* We reason as follows:
>
> $$
> \begin{aligned}
> & M \sqsubseteq_\ell M^* \\
> \Rightarrow \;& U_\ell[\pi \triangleright M] \leq U_\ell[\pi \triangleright M^*] && \text{“Expand”} \\
> \Rightarrow \;& \textstyle\sum_i a_i U_\ell(\delta^i) \leq \sum_j b_j U_\ell(\gamma^j) && \text{“Dual of Def. 2.2.5”}
> \end{aligned}
> $$

$$
\begin{array}{lll}
\Rightarrow & a + \sum_i a_i U_\ell(\delta^i) \leq a + \sum_j b_j U_\ell(\gamma^j) & \text{“Arithmetic”} \\
\Rightarrow & \sum_i a_i U_{\ell+a}(\delta^i) \leq \sum_j b_j U_{\ell+a}(\gamma^j) & \text{“Eqn (6.9), and } \sum_i a_i = 1, \sum_j b_j = 1\text{”} \\
\Rightarrow & M \sqsubseteq_{(\ell+a)} M^* & \text{“Definition”}
\end{array}
$$

□

PROPERTY 6. If $M \sqsubseteq_\ell M^*$ then $M \sqsubseteq_{(\ell-a)} M^*$ for $a \geq 0$.

Proof. Follows that of Prop.5. □

PROPERTY 7. If $M \sqsubseteq_\ell M^*$ and $M \sqsubseteq_{\ell'} M^*$ then $M \sqsubseteq_{(\ell * a + \ell' * b)} M^*$ for $a, b \geq 0$.

Proof. We reason as follows:

$$
\begin{array}{lll}
 & M \sqsubseteq_\ell M^* \text{ and } M \sqsubseteq_{\ell'} M^* & \\
\Rightarrow & U_\ell[\pi \triangleright M] + U_{\ell'}[\pi \triangleright M] \leq U_\ell[\pi \triangleright M^*] + U_{\ell'}[\pi \triangleright M^*] & \text{“Expand and add”} \\
\Rightarrow & \sum_y (a \min_w \sum_x \pi_x M_{x,y} \ell(w,x) + b \min_{w'} \sum_x \pi_x M_{x,y} \ell'(w',x)) & \text{“Dual of Thm. 2.1”} \\
 & \quad \leq \sum_y (a \min_w \sum_x \pi_x M^*_{x,y} \ell(w,x) + b \min_{w'} \sum_x \pi_x M^*_{x,y} \ell'(w',x)) & \\
\Rightarrow & \sum_y \min_{w,w'} \sum_x \pi_x M_{x,y} (a\ell(w,x) + b\ell'(w',x)) & \text{“Arithmetic”} \\
 & \quad \leq \sum_y \min_{w,w'} \sum_x \pi_x M^*_{x,y} (a\ell(w,x) + b\ell'(w',x)) & \\
\Rightarrow & M \sqsubseteq_{(a\ell + b\ell')} M^* & \text{“Eqn (6.6), simplify”}
\end{array}
$$

□

LEMMA 6.17. For any loss function ℓ, channel M and prior π,

$$U_\ell[\pi \triangleright M] \;=\; n \times U_{\pi \star \ell}[\upsilon \triangleright M]$$

where υ is the uniform prior and $n = |\mathcal{X}|$.

Proof. We reason as follows

$$
\begin{array}{lll}
 & U_\ell[\pi \triangleright M] & \\
= & \sum_y \min_w \sum_x \pi_x M_{x,y}\ \ell(w,x) & \text{“Thm. 2.1”} \\
= & \sum_y \min_w \sum_x \upsilon_x \frac{\pi_x}{\upsilon_x} M_{x,y}\ \ell(w,x) & \text{“Factor in } \upsilon\text{”} \\
= & \sum_y \min_w \sum_x \upsilon_x M_{x,y} (\frac{\pi_x}{\upsilon_x} * \ell)(w,x) & \text{“Rearranging”} \\
= & U_{\frac{\pi}{\upsilon} \star \ell}[\upsilon \triangleright M] & \text{“Def (6.18)”} \\
= & n \times U_{\pi \star \ell}[\upsilon \triangleright M] & \text{“Since } \upsilon_i = \tfrac{1}{n}\text{”}
\end{array}
$$

□

A.4 Results supporting §6.7

To complete the result of Lem. 6.27, which shows that the geometric mechanism is optimal for some non-monotonic functions, we need first to show that the geometric has the interesting

property noted in the sketch proof, namely that its restriction to any subset of inputs $\{0...K\}$ has the same leakage properties as the geometric mechanism defined over exactly those inputs. (ie. In the language of QIF they represent the same *abstract channel*).

Writing $G_{N,K}$ for the geometric mechanism on inputs $\{0...N\}$ and outputs $\{0...K\}$, and writing simply G_N for the non-truncated (infinite) geometric mechanism on inputs $\{0...N\}$, we reason (for $K < N$):

$$
\begin{aligned}
& U_\ell[\pi \triangleright G_{N,N}{\downarrow}\{0...K\}] \\
= \;& U_\ell[\pi \triangleright G_N{\downarrow}\{0...K\}] && \text{“}G_N \equiv G_{N,N} \text{ by Lem. 6.26”} \\
= \;& U_\ell[\pi \triangleright G_K] && \text{“Def. 3.4.1 is identical on } \{0...K\} \text{ for } G_N \text{ and } G_K\text{”} \\
= \;& U_\ell[\pi \triangleright G_{K,K}] && \text{“Lem. 6.26”}
\end{aligned}
$$

Thus the mechanisms $G_{N,N}{\downarrow}\{0...K\}$ and $G_{K,K}$ have the same leakage properties.

Finally we showed the result for $\varepsilon = \ln 2$, however our proof did not rely on the particular values of ε, only the fact that the constraints hold tightly for G and H. This completes the proof for Lem. 6.27.

A.5 Proofs omitted from §6.8

Lemma 6.30. The randomised response mechanism is a $\mathbf{d}_{\mathrm{D}}$-private kernel mechanism.

Proof. It is easy to see (by construction) that R is $\mathbf{d}_{\mathrm{D}}$-private, since every column has either $R_{x,y} = R_{x',y}$ or $R_{x,y} = \alpha R_{x',y}$ for any $x, x' \in \mathcal{X}$. To show it is a kernel mechanism we need to show that the hyper $\Delta_R = [\upsilon \triangleright R]$ has linearly independent inners, their convex hull contains the uniform distribution, and they are vertices in the space of $\mathbf{d}_{\mathrm{D}}$-private hypers. Observe that R is doubly stochastic and thus the inners of Δ_R are exactly the columns of R. Observe also that each column of R has $n-1$ constraints holding tightly, and thus we have that R is a vertex mechanism. Now writing R as

$$
R = \frac{1}{k}\begin{pmatrix}
\alpha & 1 & 1 & \dots & 1 \\
1 & \alpha & 1 & \dots & 1 \\
1 & 1 & \alpha & \dots & 1 \\
 & & \dots & & \\
1 & 1 & 1 & \dots & \alpha
\end{pmatrix}
$$

we can perform basic row operations, subtracting row $k-1$ from row k for rows n down to 2 to yield:

$$
R' = \frac{1}{k}\begin{pmatrix}
\alpha & 1 & 1 & \dots & 1 \\
1-\alpha & \alpha-1 & 0 & \dots & 0 \\
0 & 1-\alpha & \alpha-1 & \dots & 0 \\
 & & \dots & & \\
0 & 0 & 0 & \dots & \alpha-1
\end{pmatrix}
$$

Noting that $\det R = \det R'$, we compute:

$$\begin{aligned}\det R' &= \frac{1}{k^n}\Big(\alpha(\alpha-1)^{n-1} - 1(1-\alpha)(\alpha-1)^{n-2} + 1(1-\alpha)^2(\alpha-1)^{n-3} + \\ &\qquad \ldots + (-1)^{n-1}(1-\alpha)^{n-1}\Big) \\ &= \frac{1}{k^n}\Big((\alpha(\alpha-1)^{n-1} + (\alpha-1)^{n-1} + (\alpha-1)^{n-1} + \ldots + (\alpha-1)^{n-1}\Big) \\ &= \frac{1}{k^n}(\alpha-1)^{n-1}(\alpha + (n-1))\end{aligned}$$

which is non-zero for $\alpha \in (0,1)$. Thus, except for the case where $\alpha = 1$ we have that the randomised response matrix is invertible, and so its columns are linearly independent. And therefore the inners of $\Delta_R = [\upsilon \triangleright R]$ are likewise linearly independent. Finally, it is easy to check that we can write the uniform distribution as the following convex combination of columns of R:

$$\frac{1}{n}R_{(-,1)} + \frac{1}{n}R_{(-,2)} + \ldots + \frac{1}{n}R_{(-,n)}$$

where $R_{(-,i)}$ denotes the ith column of R. Therefore this also holds for the inners of Δ_R. Thus R is a kernel mechanism as required. □

B

Proofs Omitted from Chapter 7

B.1 Measures and continuous hypers

Measures in general

A *measurable space* is a pair (X, Σ) where X is a set and Σ is a "sigma algebra" on X, that is a set of subsets of X that is closed under complement and countable unions, and contains X itself. ("Borel algebra" is a quasi-synonym for sigma algebras made in a particular way.)

A *measure space* is a measurable space (as above) together with an actual measure μ, thus a triple (X, Σ, μ) where μ is a function from Σ to the reals that is non-negative, assigns measure 0 to the empty set, and is countably additive, ie. the measure of the union of countable many pairwise disjoint sets is the sum of their individual measures.

Product measures

Given two measurable spaces (X, Σ_X) and (Y, Σ_Y) one can make a *product* measurable space (J, Σ) where $J = X \times Y$ and $\Sigma = \Sigma_X \otimes \Sigma_Y$, where the latter is the sigma algebra generated by $\{\sigma_X \times \sigma_Y \mid \sigma_{X,Y} : \Sigma_{X,Y} \text{ resp.}\}$. (Note "generated by"...).

Given two actual measures $\mu_{X,Y}$ on the spaces above, any measure μ satisfying $\mu(\sigma_X \times \sigma_Y) = \mu_X(\sigma_X) \times \mu_Y(\sigma_Y)$ is said to be a product measure $\mu_X \times \mu_Y$ — which is not always unique. It *is* unique however when the constituent spaces are *σ-finite*, in which case

$$\begin{aligned} & (\mu_X \times \mu_Y)(\sigma) \\ = \quad & \int_{y:Y} \mu_X(\sigma{\downarrow_y}) \, \mathrm{d}\mu_Y \\ = \quad & \int_{x:X} \mu_Y(\sigma{\downarrow_x}) \, \mathrm{d}\mu_X \quad , \end{aligned} \tag{B.1}$$

where $\sigma{\downarrow_y}$ is $\{x \mid (x,y){\in}\sigma\}$ etc.

Disintegration

The disintegration theorem shows how to decompose ("disintegrate") a product measure into what we would here call a prior and a channel if coming from the input side. From the output side, it gives the marginal and the posteriors. It requires the spaces concerned to have certain properties (Radon).

Given those properties, disintegration says that for measure μ on $(X,\sigma_X){\times}(Y,\sigma_Y)$, there is a measure μ_X on (X,σ_X) alone, and a function $M{:}\, X \to$ "measures on (Y,σ_Y)", such that (written our way)

$$\int_{x:X} M(x)(\sigma{\downarrow_x})\,\mathrm{d}\mu_X \quad = \quad \mu(\sigma) \quad , \tag{B.2}$$

and of course the same for the "$-_Y$ side". The μ_X is our prior, and the M is our channel (aka. mechanism); the μ is the joint measure (on $J = X{\times}Y$) that they make.

Continuous hypers

To take a continuous prior π and continuous channel M and make a continuous hyper, we use (B.1) to make the joint measure J from π and M, and then use (B.2) to pull out the outer and the inners.

That is, in order to take that joint distribution μ on $J = X{\times}Y$ and extract a hyper from it, one uses disintegration to get a μ_Y and a function $H{:}\, Y \to$ "measures on (X,σ_X)" such that

$$\int_{y:Y} H(y)(\sigma{\downarrow_y})\,\mathrm{d}\mu_Y \quad = \quad \mu(\sigma) \quad ,$$

where now the y is the observation and $H(y)$ is the posterior (inner) that corresponds to it. The μ_Y is the marginal, a measure on all those y's. To finish that off, one "pushes forward" the function H through the marginal μ_Y. Since the codomain of H is measures on X, that gives us our measure of measures — the "continuous hyper" that came –originally– from continuous π and continuous-valued K.

B.2 Results supporting §7.6

Lemma B1. For $T{>}0$, we have that $Ł^{\varepsilon} \sqsubseteq {}^{T}Ł^{\varepsilon}$.

Proof. This follows from the post-processing lemma [34], since ${}^{T}Ł^{\varepsilon}$ takes the output of $Ł^{\varepsilon}$ and bundles it together, reporting only the interval in which the obfuscated output from $Ł^{\varepsilon}$ occurs. □

Next, we show that if T' is a multiple of T then their respective approximants are related by refinement. This is because in that case the corresponding partitioning into outputs are also related by post-processing.

LEMMA B2. Take integers $T, k > 0$. Then ${}^{kT}Ł^{\varepsilon} \sqsubseteq {}^{T}Ł^{\varepsilon}$.

The next lemma shows that the T-approximants converge to the Laplace mechanism.

LEMMA B3. The T-approximants converge to the Laplace mechanism: $\lim_{T\to\infty} {}^{T}Ł^{\varepsilon} = Ł^{\varepsilon}$.

Proof. Observe that two channels are equal if and only if their expected posterior loss is the same for all loss functions [62]. Thus from Def. 7.4.1, specialised to the discrete case for ${}^{T}Ł^{\varepsilon}$, we see that as $T\to\infty$ the integral/summation involving ${}^{T}Ł^{\varepsilon}(x)(y) \times \ell(w,x)$ converges to the integral involving $Ł^{\varepsilon}(x)(y) \times \ell(w,x)$, for any loss function ℓ. □

It now follows that we can find an anti-refinement chain needed for Thm. 7.8: select a strictly increasing sequence for T such as $2, 4, 8, 16 \ldots$. By Lem. B2, the corresponding mechanisms ${}^{2}Ł^{\varepsilon}$, ${}^{4}Ł^{\varepsilon}$, ${}^{8}Ł^{\varepsilon} \ldots$ form a refinement antichain, which by Lem. B3 converges to $Ł^{\varepsilon}$.

We now show that each of the approximants ${}^{T}Ł^{\varepsilon}$ is also a refinement of the geometric mechanism G^{ε}_{N}. For simplicity however we label the $\mathcal{U}_N$ points $1..N$ since this is convenient for indexing, with the original ordering maintained.

The posteriors of ε·d-private mechanisms lie in a convex region

If M is an ε·**d**-private mechanism then the posteriors of $[\upsilon \triangleright M]$ lie in a convex region in $\mathbb{R}^N$ satisyfing the ε·**d**-privacy constraints:

$$0 \le v_{i+1}/\alpha \le v_i \le \alpha v_{i+1} \quad , \quad \textit{for } 1 \le i \le N$$
$$\textstyle\sum_{1\le i\le N} v_i = 1 \ .$$

Note that α is adjusted to reflect the actual distance between the original points in $\mathcal{U}_N$, thus $\ln \alpha = \varepsilon/N$. This was proven in Chapter 6.

The posteriors of $[\upsilon \triangleright {}^{T}Ł^{\varepsilon}]$ satisfy the constraints above

We only need show that the inequalities hold wrt. the underlying PDF, since integration over the same intervals preserves the inequality. Let $q \in [0,1]$, then:

$$\begin{array}{lll} & {L^{\varepsilon}}_{xq}/{L^{\varepsilon}}_{x'q} & \\ = & \varepsilon/2e^{-\varepsilon|q-x|}/\varepsilon/2e^{-\varepsilon|q-x'|} & \text{“Def. 7.4.2”} \\ = & e^{\varepsilon||q-x|-|q-x'||} & \text{“Arithmetic”} \\ \le & e^{\varepsilon|x-x'|} \ . & \text{“Triangle inequality”} \end{array}$$

The posteriors of the Geometric G^{ε}_N are vertices

The posteriors of the Geometric are all vertices of the convex region: in fact they satisfy the above inequalities as *equalities*. They are therefore of the form [35]:

$$(\gamma\alpha^r, \gamma\alpha^{r-1}, \ldots, \gamma\alpha, \gamma, \gamma\alpha, \ldots, \gamma\alpha^{n-r-1}) \,, \tag{B.3}$$

where γ is a constant to ensure that the sum of the components equal 1. We can recognise these vertices by their satisfaction of the following N equality constraints of the form:

$$\begin{aligned} v_i &= \alpha v_{i+1} \quad , \quad \textit{for } 1{\leq}i{\leq}r \\ v_{i+1} &= \alpha v_i \quad , \quad \textit{for } r{+}1{\leq}i{<}N \,, \end{aligned}$$

together with $\sum_{1\leq i\leq N} v_i = 1$.

Note that each vertex satisfies a set of constraints wrt. a unique r. We say the *turning point is* the index where the maximum occurs. In the above case this occurs at index $r{+}1$. We say that two of the Geometric's vertices are *adjacent* if their turning points differ by no more than 1.

Notice that adjacent pairs of vertices define a convex line segment whose points lie in the convex region defined above. The line segment is defined by the $N{-}2$ (independent) equalities of the form:

$$\begin{aligned} v_i &= \alpha v_{i+1} \quad , \qquad \textit{for } 1 \leq i < r{-}1 \\ v_{i+1} &= \alpha v_i \quad , \quad \textit{for } r+1 \leq i < N \,, \end{aligned}$$

namely by the equalities that are satisfied by both vertices, plus the equality $\sum_i v_i = 1$.

Any posterior in $[\upsilon \triangleright {}^{T}\text{Ł}^{\varepsilon}]$ lies in between some line segment defined by adjacent vertices when N divides T

For any posterior of $[\upsilon \triangleright {}^{T}\text{Ł}^{\varepsilon}]$, we show that its components satisfy the constraints for being on the line segment defined by an adjacent pair of vertices as above. Note that the components of the posteriors (considered as a vector in $\mathbb{R}^N$) are a scalar multiple of ${}^{T}\text{Ł}^{\varepsilon}(x)(z)$ for $x \in \mathcal{U}_N$ and $z \in \mathcal{U}_T$. We show that these values satisfy the $N{-}1$ constraints defining a pair of vertices as above.

Given any output interval $[z, z{+}1/T)$ corresponding to the "batched" outputs ${}^{T}\text{Ł}^{\varepsilon}(x)(z)$ we have that for $x \in \mathcal{U}_N$ either $x \leq z$ or $z{+}1/T \leq x$. This follows because $\lfloor x \rfloor_T = x$ since N divides T. Moreover, there is a unique $x^* \in \mathcal{U}_N$ such that $x^* \leq z \leq z{+}1/T \leq x^*{+}1/N$.

Suppose first that $0{<}x^*{<}1{-}1/N$. Now if $x \leq x^*$ and $z \leq y < z{+}1/T$, we have

$$\begin{aligned} & L^{\varepsilon}(x{-}1/N)(y) & \\ = \;& \varepsilon/2 e^{-\varepsilon|y-(x-1/N)|} & \text{“Def. 7.4.2”} \\ = \;& \varepsilon/2 e^{-\varepsilon|y-x|} e^{-\varepsilon/N} & \text{“}y \geq x\text{; arithmetic”} \\ = \;& (e^{-\varepsilon/N}) L^{\varepsilon}(x)(y) & \text{“Def. 7.4.2”} \\ = \;& \alpha L^{\varepsilon}(x)(y) \,. & \text{“}\alpha = e^{-\varepsilon/N}\text{”} \end{aligned}$$

Since there is equality for all outputs $y \geq x^* \geq x$ this means that the integral will be equality:

$$\begin{aligned}
& {}^{T}Ł^{\varepsilon}(x-1/n)(z) \\
= \quad & \int_{z \leq y < z+1/k} L^{\varepsilon}(x-1/n)(y)\,\mathrm{d}y && \text{“Def. } {}^{T}Ł^{\varepsilon}\text{, §7.6”} \\
= \quad & \alpha \int_{z \leq y < z+1/k} L^{\varepsilon}(x)(y)\,\mathrm{d}y && \text{“Above”} \\
= \quad & \alpha({}^{T}Ł^{\varepsilon}(x)(z))\ . && \text{“Def. } {}^{T}Ł^{\varepsilon}\text{, §7.6”}
\end{aligned}$$

This shows that for the secrets in $\mathcal{U}_N$ corresponding to indices $1 \leq i < r$ we have the same set of constraints as for two vertices, one with turning point r is defined by the position of x^*, and the other with turning point $r+1$.

A similar argument shows that when $x \geq x^* + 1/N$

$$\int_{z \leq y < z+1/T} {}^{T}Ł^{\varepsilon}(x+1/N)(y)\,\mathrm{d}y$$

$$= \quad (e^{-\varepsilon/N}) \int_{z \leq y < z+1/T} {}^{T}Ł^{\varepsilon}(x)(y)\,\mathrm{d}y\ ,$$

allowing us to deduce $\alpha({}^{T}Ł^{\varepsilon}(x-1/N)(z)) = {}^{T}Ł^{\varepsilon}(x)(z)$ for the corresponding remaining components.

If $0 = x^*$ or $x = 1 - 1/N$ then we only need to use one half of the argument above to obtain the $N-2$

Thus the posterior satisfies $N-1$ constraints satisfied by two adjacent Geometric posteriors, which means it lies on the line passing through the two vertices. However the Laplace approximation lies in the convex region defining the posteriors of $\varepsilon \cdot \mathbf{d}$-private mechanisms and so it must lie on the convex line segment between the two posteriors.

Summary of the proofs of Lem. 7.7 and Thm. 7.8

Lem. 7.7 now follows, since we can establish an antichain of approximants ${}^{T}Ł^{\varepsilon}$ which all refine $Ł^{\varepsilon}$ and G^{ε}_{N}. The former refinements follow directly from the discussion above, and the latter follow when N divides T, and using the argument set out in Lem. 7.7 — which now applies since the posteriors of $[\upsilon \triangleright {}^{T}Ł^{\varepsilon}]$ have been shown to lie within the convex hull of the posteriors of the geometric. This is enough to establish Lem. 7.7 in the special case that N divides T; this is also enough to establish Thm. 7.8.

But now the general case of Lem. 7.7 is immediate since for any $T > 0$ we have that ${}^{T}Ł^{\varepsilon}$ refines $Ł^{\varepsilon}$.

B.3 Supporting material for §7.6.4

Here we provide the details for Thm. 7.9.

Earth move for adjacent posteriors in the geometric

We show that two adjacent posteriors in the geometric G^{ε}_{N} on $\mathcal{U}_N$ differ according to the Manhattan metric by no more than c/N, where $c = 3/(1-1/e^{\varepsilon})^2$, and whenever $N \geq \varepsilon$.

Let v, v' be adjacent posteriors represented as a one-summing vectors in $\mathbb{R}^N$. We show that the amount of earth to be moved from each component v_i is at most c/N^2, where $c = 3/(1-1/e^{\varepsilon})^2$. Let v, v' be two adjacent vertices, as defined above.

$$v = [a\lambda^{\alpha}, a\lambda^{\alpha-1}, \ldots, a, a\lambda, \ldots, a\lambda^{n-\alpha}]$$

and

$$v' = [b\lambda^{\alpha+1}, b\lambda^{\alpha}, \ldots, b\lambda, b, \ldots, b\lambda^{n-\alpha-1}]$$

Recall that $\lambda = e^{-\varepsilon/N}$. Let $\Lambda = 1 + \lambda + \cdots + \lambda^{\alpha}$, and $\Gamma = \lambda + \cdots + \lambda^{N-\alpha-1}$.

From above we have $a = 1/(\Lambda + \Gamma + \lambda^{N-\alpha})$ and $b = 1/(\Lambda + \Gamma + \lambda^{\alpha+1})$.

We note first that both $\Lambda, \Gamma \leq 1/(1-\lambda)$ (since both are sub-sums of the summation $1 + \lambda + \ldots$) and so:

$$\Lambda + \Gamma \leq 2/(1-\lambda) \tag{B.4}$$

We now observe the following. Since $\lambda \leq 1$ we have:

$$1 + \lambda + \cdots + \lambda^{N-1} \leq \Lambda + \Gamma$$

which, since $\lambda = e^{-\varepsilon/N}$ is equivalent to

$$(1 - 1/e^{\varepsilon})/(1-\lambda) = (1-\lambda^N)/(1-\lambda) \leq \Lambda + \Gamma\,. \tag{B.5}$$

We now reason:

$$\begin{aligned}
& a - \lambda b \\
= \;& 1/(\Lambda+\Gamma+\lambda^{N-\alpha}) - \lambda/(\Lambda+\Gamma+\lambda^{\alpha+1}) \\
= \;& ((\Lambda+\Gamma+\lambda^{\alpha+1}) - \lambda(\Lambda+\Gamma+\lambda^{N-\alpha}))/(\Lambda+\Gamma+\lambda^{N-\alpha})(\Lambda+\Gamma+\lambda^{\alpha+1}) \\
= \;& ((\Lambda+\Gamma)(1-\lambda) + \lambda^{\alpha+1} - \lambda^{N-\alpha+1})/(\Lambda+\Gamma+\lambda^{N-\alpha})(\Lambda+\Gamma+\lambda^{\alpha+1}) \\
\leq \;& 3/(\Lambda+\Gamma+\lambda^{N-\alpha})(\Lambda+\Gamma+\lambda^{\alpha+1}) && \text{“(B.4); } |\lambda^{\alpha+1} - \lambda^{N-\alpha+1}| \leq 1\text{”} \\
\leq \;& 3/((1-1/e^{\varepsilon})/(1-\lambda) + \lambda^{N\alpha})((1-1/e^{\varepsilon})/(1-\lambda) + \lambda^{\alpha+1}) && \text{“(B.5)”} \\
= \;& 3(1-\lambda)^2/((1-1/e^{\varepsilon}) + (1-\lambda)\lambda^{N-\alpha})((1-1/e^{\varepsilon}) + (1-\lambda)\lambda^{\alpha+1}) && \text{“Multiply top and bottom by } (1-\lambda)^2\text{”} \\
\leq \;& 3(1-\lambda)^2/(1-1/e^{\varepsilon})^2 && \text{“Reduce denominator: } (1-\lambda)\lambda^r \geq 0\text{”} \\
\leq \;& 3/N^2(1-1/e^{\varepsilon})^2\,. && \text{“}1-\lambda = 1-e^{-\varepsilon/N} \leq \varepsilon/N \text{ when } N \geq \varepsilon\text{”}
\end{aligned}$$

From this, noting that $0<\lambda\leq 1$ it follows that:

$$|a\lambda^k - \lambda^{k+1}b| \leq 3/(1-1/e^{\varepsilon})^2N^2 \,,$$

whenever $N \geq \varepsilon$.

A similar argument shows also that

$$|a\lambda^{k+1} - \lambda^k b| \leq 3/(1-1/e^{\varepsilon})^2N^2 \,.$$

Kantorovich distance between adjacent vertices representing posteriors in $\mathbb{D}\mathcal{U}_N$

The Kantorovich distance $\mathbb{W}(\cdot,\cdot)$ on distributions in $\mathbb{D}\mathcal{U}_N$ is determined by the underlying Euclidean metric on points in $\mathcal{U}_N$. However since all such distances are no more than 1, this means that $\mathbb{W}(v,v')$ is no more than the Manhattan metric, and this can be easily calculated from above.

In particular the Earth Move to transform v to v', we have $|v_i - v'_i| \leq 3/(1-1/e^{\varepsilon})^2N^2 = c/N^2$. Note that for fixed ε, c is a constant.

The Manhattan metric between v, v' is equal to $\sum_i |v_i - v'_i|/2 \leq c/N$, and therefore $\mathbb{W}(v,v') \leq c/N$.

The difference between loss functions on adjacent vertices

By the Kantorovich-Rubinstein theorem, this means that whenever $\ell(w,\cdot)$ is κ-Lipschitz function of x, we must have:

$$|U_\ell(v) - U_\ell(v')| \leq c\kappa/N \,.$$

To see this, suppose that $U_\ell(v) \geq U_\ell(v')$, let $w^* \in \mathcal{W}$ be the choice that achieves the minimum $U_\ell(v')$. Note that w^* exists since $\mathcal{W}$ is compact.

$$U_\ell(v') = \sum_i \ell(w^*,x_i)\times v_i \leq U_\ell(v) \leq \sum_i \ell(w^*,x_i)\times v'_i \,.$$

Now since $\ell(w^*,x)$ is κ-Lipschitz we see that:

$$\begin{aligned} & |U_\ell(v) - U_\ell(v')| \\ \leq \;& |\textstyle\sum_i \ell(w^*,x_i)\times v_i - \sum_i \ell(w^*,x_i)\times v'_i| \leq c\kappa/N \,. \end{aligned}$$

Kantorovich distance between the hypers in Thm. 7.9

The Kantorovich distance between hyper-distributions is equal to the least expected Earth Move to transform one distribution Δ to another Δ'. The cost of the Earth Move is the average distance each inner of Δ moves in the transformation to Δ'. For us, this is therefore bounded above by the the average move computed using the Manhattan distance.

Since a refinement describes an Earth Move, we see from the proof of Lem. 7.7 that the Earth move described taking $[\upsilon \triangleright G^{\varepsilon}_N]$ to $[\upsilon \triangleright {}^{T}\text{Ł}^{\varepsilon}]$ involves moving a vertex v representing an inner of $[\upsilon \triangleright G^{\varepsilon}_N]$ to an inner v'' of $[\upsilon \triangleright {}^{T}\text{Ł}^{\varepsilon}]$ which lies on a line *in between* adjacent inners v, v' of $[\upsilon \triangleright G^{\varepsilon}_N]$. The cost of such a move is therefore no more than the cost of a move from v to v' which, as shown above, is no more than c/N.

This applies to all moves in the transformation of $[\upsilon \triangleright G^{\varepsilon}_N]$ to $[\upsilon \triangleright {}^{T}\text{Ł}^{\varepsilon}]$, and therefore the cost of the Earth Move defined by the refinement is no more than c/N also. Thus $\mathbb{W}([\upsilon \triangleright G^{\varepsilon}_N], [\upsilon \triangleright {}^{T}\text{Ł}^{\varepsilon}]) \leq c/N$.

Comparison of losses related to Laplace and Geometric mechanisms in Thm. 7.10

The Kantorovich-Rubinstein theorem now applies at the level of hyper-distributions, so we have:

$$U_\ell[\upsilon \triangleright {}^{T}\text{Ł}^{\varepsilon}] - U_\ell[\upsilon \triangleright G^{\varepsilon}_N] \quad \leq \quad c\kappa/N\,,$$

whenever U_ℓ is κ-Lipschitz, which by the above, follows whenever $\ell(w, -)$ is a κ-Lipschitz function of x.

The result holds also for Ł^{ε} since $G^{\varepsilon}_N \sqsubseteq \text{Ł}^{\varepsilon} \sqsubseteq {}^{T}\text{Ł}^{\varepsilon}$.

Comparison of losses for any prior π

We show that generally for π that:

$$U_\ell[\pi \triangleright {}^{T}\text{Ł}^{\varepsilon}] - U_\ell[\pi \triangleright G^{\varepsilon}_N] \quad \leq \quad c\kappa/N\,,$$

This follows as above from the following facts:

1. $U_\ell[\pi \triangleright M] = U_{\ell \star \pi \star N}[\upsilon \triangleright M]$ (Lem. 6.17), where

$$(\ell \star \pi \star N)(w, x) \quad := \quad \ell(w, x) \times \pi_x \times N\,,$$

2. and, for adjacent vertices v, v' of the Geometric:

$$|U_{\ell \star \pi \star N}(v) - U_{\ell \star \pi \star N}(v')| \leq c\kappa/N\,.$$

We can establish (2) by a direct calculation.

Since $\ell(w, x)$ is κ-Lipschitz (for x) we may assume that $0 \leq \ell(w, x) \leq \kappa$ whenever $x \in \mathcal{U}_N \subseteq [0, 1]$. Note that we write v_x for the component of v that corresponds to the probability assigned to x. As above, the result follows if we can show the bound for any w. (This is because for $U_{\ell \star \pi \star N}$ we pick the w which achieves the loss for whichever is the lesser of $U_{\ell \star \pi \star N}(v)$ and $U_{\ell \star \pi \star N}(v')$.) We now compute:

$$\begin{array}{ll} & |\sum_x (N\pi_x \times \ell(w, x) \times v_x) - \sum_x (N\pi_x \times \ell(w, x) \times v'_x)| \\ = & |\sum_x N\pi_x \times \ell(w, x)(v_x - v'_x)| \qquad \text{“Arithmetic”} \end{array}$$

$$
\begin{aligned}
&\leq \sum_x N\pi_x \times \ell(w,x)|v_x - v'_x)| && \text{“Triangle inequality for } |\cdot|\text{”}\\
&\leq \sum_x N\pi_x \times \ell(w,x)c/N^2 && \text{“}v,v'\text{ adjacent, thus } |v_x - v'_x| \leq c/N^2\text{”}\\
&= c/N \sum_x \pi_x \times \ell(w,x) && \text{“Arithmetic”}\\
&\leq c/N \sum_x \pi_x \times \kappa && \text{“}0 \leq \ell(w,x) \leq \kappa\text{”}\\
&= c\kappa/N\ . && \text{“}\textstyle\sum_x \pi_x = 1\text{”}
\end{aligned}
$$

B.4 Supporting material for §7.6.5

Finally, we note that U_ℓ can be approximated by step functions.

LEMMA D4. If ℓ is continuous in w and x, and $\mathcal{W}$ is compact then:

$$U_{\ell_N} \longrightarrow U_\ell\ ,\quad as \quad N \to \infty$$

Proof. Compactness of $\mathcal{W}$ implies that $U_\ell(\pi) = \mathcal{E}_\pi\,(\ell(w^*,-))$ for some w^*. The result now follows since ℓ is continuous and $\lfloor x \rfloor_N$ converges to x. □

B.5 Step functions are not legal for continuous mechanisms

In the proof of Thm. 7.3 we use the fact that approximations to continuous monotone loss functions remain monotone for the approximations we choose. However if ℓ itself is a step function (and so not continuous) then Thm. 7.3 does not apply because the proof relies on the step approximations being monotone. This does not always happen as the next lemma shows.

LEMMA E5. If ℓ is monotone on $[0,1]\times[0,1]$ then ℓ_N is not necessarily monotone on $\mathcal{U}_T$.

Proof. Let $\ell(w,x)$:= $len(w, \lfloor x \rfloor_2)$, and we show that ℓ is not monotone on $\{0, 1/4, 1/2, 3/4\}$. This follows by setting $x = \frac{1}{2}$ and $w_1 = \frac{1}{4}$ and $w_2 = \frac{3}{4}$. We note that $|x - w_1| = |x - w_2|$ but $\ell(w_1, x) = \frac{1}{2} \neq 0 = \ell(w_2, x)$. □

This means that for step functions, Geometrics could still have better utility than the Laplace.

B.6 Example to show that for step loss functions, the Laplace is not optimal

In this example we set $\varepsilon = 4\ln 2$. The first matrix is the geometric mechanism over $\mathcal{U}_2$, specifically with inputs $\{0, 1/2, 1\}$ so that all inputs are exactly multiples of $1/2$ apart. Here the rows represent the probabilities assigned to the output observations also in $\mathcal{U}_2$. Notice that the $\varepsilon{\cdot}\mathbf{d}$-privacy constraints mandate that the ratios between successive column entries lie between $1/4$ and 4. This is therefore a *scaled* geometric mechanism to reflect the underlying differences in $\mathcal{U}_2$.

$$G_2^\varepsilon = \begin{bmatrix} 4/5 & 3/20 & 1/20 \\ 1/5 & 3/5 & 1/5 \\ 1/20 & 3/20 & 4/5 \end{bmatrix}$$

Similarly for G_4^ε is the geometric mechanism over $\mathcal{U}_4 := \{0, 1/4, 1/2, 3/4, 1\}$. Observe that there are of course many more outputs because the possibilities for potential inputs now also include values $1/4, 3/4$. Notice that $\varepsilon \cdot \mathbf{d}$-privacy constraints mandate successive ratios lie between $1/2$ and 2.

$$G_4^\varepsilon = \begin{bmatrix} 2/3 & 1/6 & 1/12 & 1/24 & 1/24 \\ 1/3 & 1/3 & 1/6 & 1/12 & 1/12 \\ 1/6 & 1/6 & 1/3 & 1/6 & 1/6 \\ 1/12 & 1/12 & 1/6 & 1/3 & 1/3 \\ 1/24 & 1/24 & 1/12 & 1/6 & 2/3 \end{bmatrix}$$

Note however that we can restrict G_4^ε to priors π_2 which only assign non-zero weight to the points in $\mathcal{U}_2$. Moreover when we do this, we find that the $\varepsilon \cdot \mathbf{d}$-privacy constraints consistent with $\mathcal{U}_2$ are still satisfied: ie. the relations between successive column entries in the restricted matrix lie between $1/4$ and 4 — because they represent *two* adjacencies from the original matrix, where the ratios were between $1/2$ and 2. The matrix made from G_4^ε but restricted to $\mathcal{U}_2$ consists of the first, middle and last rows of G_4^ε ie. :

$$M = \begin{bmatrix} 2/3 & 1/6 & 1/12 & 1/24 & 1/24 \\ 1/6 & 1/6 & 1/3 & 1/6 & 1/6 \\ 1/24 & 1/24 & 1/12 & 1/6 & 2/3 \end{bmatrix}$$

Now suppose that we choose a ℓ_2-legal loss function known as "Bayes' Risk", that is br_2 defined

$$\mathsf{br}_2(w, x) \quad := \quad 1 \text{ if } \lfloor x \rfloor_2 \neq w \text{ else } 0\,,$$

where $\mathcal{W} = \mathcal{U}_2$.

Assuming a uniform prior υ_2 over $\mathcal{U}_2$, we can compute the losses directly:

$$U_{\mathsf{br}_2}[\upsilon_2 \triangleright G_2^\varepsilon] = 4/15 < 1/3 = U_{\mathsf{br}_2}[\upsilon_2 \triangleright M]\,.$$

However from Thm. 7.8 we have the refinement $G_4^\varepsilon \sqsubseteq \text{Ł}^\varepsilon$, and by observing that for υ_2 we have $U_{\mathsf{br}_2}[\upsilon_2 \triangleright M] = U_{\mathsf{br}_2}[\upsilon_2 \triangleright G_4^\varepsilon]$ we deduce also:

$$U_{\mathsf{br}_2}[\upsilon_2 \triangleright G_2^\varepsilon] < U_{\mathsf{br}_2}[\upsilon_2 \triangleright G_4^\varepsilon] \leq U_{\mathsf{br}_2}[\upsilon_2 \triangleright \text{Ł}^\varepsilon]\,.$$

Thus for discrete datasets, the Laplace mechanism is not necessarily optimal for step functions, as has indeed been noted in other works [91].

C

Proofs Omitted from Chapter 10

C.1 Material supporting §10.4

We first recall the Chernoff bound, which is used in the proof for Prop. 10.4.

LEMMA A1 (Chernoff bound). Let Z be a real-valued random variable. Then for all $t \in \mathbb{R}$,

$$\Pr[Z \geq t] \leq \min_{s \in \mathbb{R}} \frac{\mathbb{E}[\exp(sZ)]}{\exp(st)}.$$

Next we recall Hoeffding's lemma, which is used in the proof for Thm. 10.6.

LEMMA A2 (Hoeffding). Let $a, b \in \mathbb{R}$, and Z be a real-valued random variable such that $\mathbb{E}[Z] = \mu$ and that $a \leq Z \leq b$. Then for all $t \in \mathbb{R}$,

$$\mathbb{E}[\exp(tZ)] \leq \exp\left(t\mu + \tfrac{t^2}{8}(b-a)^2\right).$$

Note that Lem. A2 implies that $\mathbb{E}[\exp(t(Z - \mathbb{E}[Z]))] \leq \exp\left(\frac{t^2}{8}(b-a)^2\right)$.

Then we recall the Chernoff-Hoeffding Theorem, which is used in the proof for Thm. 10.7. Recall that the Kullback-Leibler divergence $D_{\mathrm{KL}}(a \| b)$ between Bernoulli distributed random variables with parameters a and b is defined by:

$$D_{\mathrm{KL}}(a \| b) = a \ln \tfrac{a}{b} + (1-a) \ln \tfrac{1-a}{1-b}.$$

LEMMA A3 (Chernoff-Hoeffding). Let $Z \sim \mathit{Binomial}(k, p)$ be a binomially distributed random variable where k is the total number of experiments and p is the probability that an experiment yields a successful outcome. Then for any

$\alpha \in \mathbb{R}_{>0}$,

$$\Pr[Z \geq k(p+\alpha)] \leq \exp\bigl(-k\, D_{\mathrm{KL}}(p+\alpha \| p)\bigr).$$

By relaxing this, we have a simpler bound:

$$\Pr[Z \geq k(p+\alpha)] \leq \exp\bigl(-2k\alpha^2\bigr).$$

We show the proofs for technical results as follows.

Proposition 10.1 (XDP of BRR). Let $\varepsilon \in \mathbb{R}_{\geq 0}$ and $\kappa \in \mathbb{Z}_{>0}$. The (ε, κ)-bitwise randomised response Q_{brr} provides $(\varepsilon \cdot \mathbf{d}_{\mathrm{H}}, 0)$-XDP.

Proof. This is a straightforward extension of the existing result on vectors wrt Manhattan metric [7]. In particular, recalling Def. 3.4.3, and letting $r = \frac{1}{e^{\varepsilon}+1}$, $\boldsymbol{v} = (v_1, v_2, \ldots, v_\kappa) \in \mathcal{V}$, $\boldsymbol{v}' = (v'_1, v'_2, \ldots, v'_\kappa) \in \mathcal{V}$, and $\boldsymbol{y} = (y_1, y_2, \ldots, y_\kappa) \in \mathcal{V}$, we have (from Def. 10.3.7) that:

$$Q_{\mathsf{brr}}(\boldsymbol{v})(\boldsymbol{y}) = \textstyle\prod_{i=1}^{\kappa} r^{|y_i - v_i|}(1-r)^{1-|y_i - v_i|}$$
$$Q_{\mathsf{brr}}(\boldsymbol{v}')(\boldsymbol{y}) = \textstyle\prod_{i=1}^{\kappa} r^{|y_i - v'_i|}(1-r)^{1-|y_i - v'_i|}.$$

By $Q_{\mathsf{brr}}(\boldsymbol{v}')(\boldsymbol{y}) > 0$ and the triangle inequality, we have:

$$\ln \tfrac{Q_{\mathsf{brr}}(\boldsymbol{v})(\boldsymbol{y})}{Q_{\mathsf{brr}}(\boldsymbol{v}')(\boldsymbol{y})} \leq \ln \prod_{i=1}^{\kappa} \bigl(\tfrac{1-r}{r}\bigr)^{|v_i - v'_i|} = \ln \bigl(\tfrac{1-r}{r}\bigr)^{\mathbf{d}_{\mathrm{H}}(\boldsymbol{v}, \boldsymbol{v}')} = \varepsilon \mathbf{d}_{\mathrm{H}}(\boldsymbol{v}, \boldsymbol{v}').$$

Therefore Q_{brr} provides $(\varepsilon \mathbf{d}_{\mathrm{H}}, 0)$-XDP. □

Proposition 10.2 (Privacy of Q_H wrt $d_{\varepsilon H}$). Let $\varepsilon \in \mathbb{R}_{\geq 0}$, $H : \mathcal{X} \to \mathcal{V}$ be a κ-bit LSH function, and $d_{\varepsilon H} : \mathcal{X} \times \mathcal{X} \to \mathbb{Z}_{\geq 0}$ be the pseudometric defined by $d_{\varepsilon H}(\boldsymbol{x}, \boldsymbol{x}') = \varepsilon \cdot \mathbf{d}_{\mathrm{H}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))$ for each $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$. Then the ε-LSHRR mechanism Q_H provides $(d_{\varepsilon H}, 0)$-XDP.

Proof. Let $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ and $\boldsymbol{y} \in \mathcal{V}$. We reason as follows:

$$\begin{aligned}
& Q_H(\boldsymbol{x})(\boldsymbol{y}) \\
= \;& Q_{\mathsf{brr}}(H(\boldsymbol{x}))(\boldsymbol{y}) && \text{“Def. 10.3.8”} \\
\leq \;& e^{\varepsilon \cdot \mathbf{d}_{\mathrm{H}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))} Q_{\mathsf{brr}}(H(\boldsymbol{x}'))(\boldsymbol{y}) && \text{“Prop. 10.1”} \\
= \;& e^{d_{\varepsilon H}(\boldsymbol{x}, \boldsymbol{x}')} Q_H(\boldsymbol{x}')(\boldsymbol{y}) && \text{“Definition of } d_{\varepsilon H}\text{, Def. 10.3.8”}
\end{aligned}$$

Hence Q_H provides $(d_{\varepsilon H}, 0)$-XDP. □

Proposition 10.3 (Worst-case privacy of Q_H). Let $\varepsilon \in \mathbb{R}_{\geq 0}$ and $H : \mathcal{X} \to \mathcal{V}$ be a κ-bit LSH function. The ε-LSHRR mechanism Q_H provides $\kappa\varepsilon$-DP.

Proof. Since $d_H(\boldsymbol{x}, \boldsymbol{x}') \leq \kappa$ holds for all $\boldsymbol{x}, \boldsymbol{x}'$, this proposition follows from Proposition 10.2. □

Proposition 10.4 (CXDP ⇒ PXDP). Let $\mu \in \mathbb{R}_{\geq 0}$, $\tau \in \mathbb{R}_{>0}$, $\lambda \in \mathbb{D}\mathcal{R}$, $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be a randomised algorithm, and $\mathbf{d}_\mathcal{X}$ be a metric over $\mathcal{X}$. Let $\delta \in (0,1]$ and $\varepsilon = \tau\sqrt{-2\ln\delta}$. We define $\xi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ by $\xi(x,x') = \mu{\cdot}\mathbf{d}_\mathcal{X}(x,x') + \varepsilon$ for all $x, x' \in \mathcal{X}$. If A_λ provides $(\mu,\tau,\mathbf{d}_\mathcal{X})$-CXDP, then it provides (ξ,δ)-PXDP.

Proof. Assume that A_λ provides $(\mu,\tau,\mathbf{d}_\mathcal{X})$-CXDP. Let $x, x' \in \mathcal{X}$. Then we will show $\Pr[\,\mathcal{L}_{x,x'} > \mu{\cdot}\mathbf{d}_\mathcal{X}(x,x') + \varepsilon\,] \leq \delta$ as follows.

Let $Z = \mathcal{L}_{x,x'} - \mathbb{E}[\mathcal{L}_{x,x'}]$. By the definition of CXDP (Def. 10.4.2), we have that:

$$\mathbb{E}[\mathcal{L}_{x,x'}] \leq \mu{\cdot}\mathbf{d}_\mathcal{X}(x,x'), \text{ and} \tag{C.1}$$

$$Z \text{ is } \tau\text{-subgaussian} \tag{C.2}$$

Therefore we reason:

$$\begin{aligned}
& \Pr[Z \geq t] \\
\leq\ & \min_{s\in\mathbb{R}} \tfrac{\mathbb{E}[\exp(sZ)]}{\exp(st)} && \text{“Lem. A1”} \\
\leq\ & \min_{s\in\mathbb{R}} \exp\left(\tfrac{\tau^2 s^2}{2} - st\right) && \text{“Def. 10.3.4 and (C.2)”} \\
=\ & \min_{s\in\mathbb{R}} \exp\left(\tfrac{\tau^2}{2}\left(s - \tfrac{t}{\tau^2}\right)^2 - \tfrac{t^2}{2\tau^2}\right) && \text{“Rearranging”} \\
=\ & \exp\left(-\tfrac{t^2}{2\tau^2}\right) && \text{“when } s = \tfrac{t}{\tau^2}\text{”}
\end{aligned}$$

And so we deduce:

$$\begin{aligned}
& \Pr[\mathcal{L}_{x,x'} > \mu{\cdot}\mathbf{d}_\mathcal{X}(x,x') + \varepsilon] \\
\leq\ & \Pr[\mathcal{L}_{x,x'} > \mathbb{E}[\mathcal{L}_{x,x'}] + \varepsilon] && \text{“(C.1)”} \\
=\ & \Pr[Z > \varepsilon] && \text{“Def. of } Z\text{”} \\
\leq\ & \exp\left(-\tfrac{\varepsilon^2}{2\tau^2}\right) && \text{“Proven above”} \\
=\ & \delta && \text{“Subst. } \varepsilon = \tau\sqrt{-2\ln\delta}\text{, simplify”}
\end{aligned}$$

Therefore the randomised algorithm A_λ provides (ξ,δ)-PXDP. □

Proposition 10.5 (PXDP ⇒ XDP). Let $\lambda \in \mathbb{D}\mathcal{R}$, $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be a randomised algorithm, $\xi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, and $\delta : \mathcal{X} \times \mathcal{X} \to [0,1]$. If A_λ provides (ξ,δ)-PXDP, then it provides (ξ,δ)-XDP.

Proof. Assume that A_λ provides (ξ,δ)-PXDP. Let $x, x' \in \mathcal{X}$. By the definition of (ξ,δ)-PXDP (Def. 10.4.3), we have $\Pr[\,\mathcal{L}_{x,x'} > \xi(x,x')\,] \leq \delta(x,x')$. Let $S \subseteq \mathcal{Y}$. For each $r \in \mathcal{R}$, let $S'_r = \{y \in S \mid \mathcal{L}_{x,x',y,r} > \xi(x,x')\}$. Then

$$\sum_r \lambda_r A_r(x)(S'_r) \leq \delta(x,x') \tag{C.3}$$

and for each $r \in \mathcal{R}$,

$$A_r(x)(S \setminus S'_r) \leq \exp(\xi(x,x')) \cdot A_r(x')(S \setminus S'_r). \tag{C.4}$$

And so we reason:

$$A_\lambda(x)(S)$$

$$
\begin{aligned}
&= \sum_r \lambda_r A_r(x)(S) && \text{“Expanding”}\\
&= \sum_r \lambda_r A_r(x)(S \setminus S'_r) + \sum_r \lambda_r A_r(x)(S'_r) && \text{“Arithmetic”}\\
&\le \left(\sum_r \lambda_r \exp(\xi(x,x')) \cdot A_r(x')(S \setminus S'_r)\right) + \delta(x,x') && \text{“(C.3) and (C.4)”}\\
&\le \exp(\xi(x,x')) \cdot \left(\sum_r \lambda_r A_r(x')(S)\right) + \delta(x,x') && \text{“Arithmetic, } S \setminus S'_r \subseteq S\text{”}\\
&= \exp(\xi(x,x')) \cdot A_\lambda(x')(S) + \delta(x,x') && \text{“Simplify”}
\end{aligned}
$$

Therefore A_λ provides (ξ,δ)-XDP. □

To prove the CXDP of the LSHRR, we show that the Hamming distance between hash values follows a binomial distribution.

> **Lemma A4** (Distribution of the Hamming distance of LSH). Let $\mathcal{H}$ be an LSH scheme wrt a metric $\mathbf{d}_\mathcal{X}$ over $\mathcal{X}$ coupled with a distribution $D_\mathcal{H}$. Let $\boldsymbol{x},\boldsymbol{x}' \in \mathcal{X}$ be any two inputs, and Z be the random variable of the Hamming distance between their κ-bit hash values, ie., $Z = \mathbf{d}_\mathrm{H}(H(\boldsymbol{x}),H(\boldsymbol{x}'))$ where a κ-bit LSH function H is drawn from the distribution $D^\kappa_\mathcal{H}$. Then Z follows the binomial distribution with mean $\kappa\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}')$ and variance $\kappa\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}')(1-\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}'))$.

Proof. By the definition of the Hamming distance $\mathbf{d}_\mathrm{H}$ and the construction of the LSH-based κ-bit function H, we have $\mathbf{d}_\mathrm{H}(H(\boldsymbol{x}),H(\boldsymbol{x}')) = \sum_{i=1}^\kappa |\, h_i(\boldsymbol{x}) - h_i(\boldsymbol{x}')\,|$. Since $\sum_{i=1}^\kappa |\, h_i(\boldsymbol{x}) - h_i(\boldsymbol{x}')\,|$ represents the number of non-collisions between hash values of $\boldsymbol{x}$ and $\boldsymbol{x}'$, it follows the binomial distribution with mean $\kappa\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}')$ and variance $\kappa\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}')(1-\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}'))$. □

> **Theorem 10.6** (CXDP of the LSHRR). The ε-LSH-based privacy mechanism Q_{LSHRR} provides $(\varepsilon\kappa, \frac{\varepsilon\kappa}{2}, \mathbf{d}_\mathcal{X})$-CXDP.

Proof. From Prop. 10.2 we have that $Q_H(\boldsymbol{x})(y) \le e^{\varepsilon\cdot\mathbf{d}_\mathrm{H}(H(\boldsymbol{x}),H(\boldsymbol{x}'))} Q_H(\boldsymbol{x}')(y)$. Let Z be the random variable defined by $Z \stackrel{\text{def}}{=} \mathbf{d}_\mathrm{H}(H(\boldsymbol{x}),H(\boldsymbol{x}'))$ where $H = (h_1,h_2,\dots,h_\kappa)$ is distributed over $\mathcal{H}^\kappa$, namely, the seeds of these LSH functions are chosen randomly. Then Z takes values in $\{0,1,\dots,\kappa\}$. By Lemma A4, Z follows the binomial distribution with mean $\mathbb{E}[Z] = \kappa\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}')$. Then the random variable $\varepsilon Z - \mathbb{E}[\varepsilon Z]$ is centered, ie., $\mathbb{E}[\varepsilon Z - \mathbb{E}[\varepsilon Z]] = 0$, and ranges over $[-\varepsilon\kappa\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}'), \varepsilon\kappa(1-\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}'))]$. Hence it follows from Hoeffding's lemma (Lemma A2) that:

$$\mathbb{E}[\exp(t(\varepsilon Z - \mathbb{E}[\varepsilon Z]))] \le \exp\left(\tfrac{t^2}{8}(\varepsilon\kappa)^2\right) = \exp\left(\tfrac{t^2}{2}\left(\tfrac{\varepsilon\kappa}{2}\right)^2\right).$$

Hence by Def. 10.3.4, $\varepsilon Z - \mathbb{E}[\varepsilon Z]$ is $\frac{\varepsilon\kappa}{2}$-subgaussian. Therefore, the LSH-based mechanism Q_{LSHRR} provides $(\varepsilon\kappa, \frac{\varepsilon\kappa}{2}, \mathbf{d}_\mathcal{X})$-CXDP. □

> **Theorem 10.7** (PXDP/XDP of the LSHRR). Let $\delta \in \mathbb{R}_{>0}$ and $\varepsilon' = \varepsilon\sqrt{\frac{-\ln\delta}{2}}$. We define $\xi : \mathcal{X}\times\mathcal{X} \to \mathbb{R}_{\ge 0}$ by $\xi(\boldsymbol{x},\boldsymbol{x}') = \varepsilon\kappa\cdot\mathbf{d}_\mathcal{X}(\boldsymbol{x},\boldsymbol{x}') + \varepsilon'\sqrt{\kappa}$. The ε-LSH-based mechanism Q_{LSHRR} provides (ξ,δ)-PXDP, hence (ξ,δ)-XDP.

Proof. Let $\alpha = \sqrt{\frac{-\ln\delta}{2\kappa}}$. Let Z be the random variable defined by $Z \stackrel{\text{def}}{=} \mathbf{d}_\mathrm{H}(H(\boldsymbol{x}),H(\boldsymbol{x}'))$ where $H = (h_1,h_2,\dots,h_\kappa)$ is distributed over $\mathcal{H}^\kappa$. By Lemma A4, Z follows the binomial distribution

with mean $\mathbb{E}[Z] = \kappa \mathbf{d}_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$. Hence it follows from Chernoff-Hoeffding theorem (Lemma A3) that:

$$\Pr[Z \geq \kappa(\mathbf{d}_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha)] \leq \exp\left(-2\kappa\alpha^2\right) = \delta.$$

Hence $\Pr[\varepsilon Z \geq \varepsilon\kappa \mathbf{d}_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \varepsilon'\sqrt{\kappa}] \leq \delta$. Therefore Q_{LSHRR} provides (ξ, δ)-PXDP. By Proposition 10.5, Q_{LSHRR} provides (ξ, δ)-XDP. □

Proposition 10.8 (Tighter bound for PXDP/XDP)**.** For an $\alpha \in \mathbb{R}_{>0}$, we define $\xi_\alpha : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ and $\delta_\alpha : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ by:

$$\xi_\alpha(\boldsymbol{x}, \boldsymbol{x}') = \varepsilon\kappa(\mathbf{d}_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha)$$
$$\delta_\alpha(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\kappa D_{\mathrm{KL}}(\mathbf{d}_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha \| \mathbf{d}_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'))\right).$$

The ε-LSH-based mechanism Q_{LSHRR} provides $(\xi_\alpha, \delta_\alpha)$-PXDP, hence $(\xi_\alpha, \delta_\alpha)$-XDP.

Proof. Let Z be the random variable defined by $Z \stackrel{\text{def}}{=} \mathbf{d}_{\mathrm{H}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))$ where $H = (h_1, h_2, \ldots, h_\kappa)$ is distributed over $\mathcal{H}^\kappa$. By Chernoff-Hoeffding theorem (Lemma A3),

$$\Pr[Z \geq \kappa(\mathbf{d}_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha)] \leq \delta_\alpha(\boldsymbol{x}, \boldsymbol{x}').$$

Then $\Pr[\varepsilon Z \geq \xi_\alpha(\boldsymbol{x}, \boldsymbol{x}')] \leq \delta_\alpha(\boldsymbol{x}, \boldsymbol{x}')$. Therefore Q_{LSHRR} provides $(\xi_\alpha, \delta_\alpha)$-PXDP. By Proposition 10.5, Q_{LSHRR} provides $(\xi_\alpha, \delta_\alpha)$-XDP. □

Proposition 10.9 (XDP of LapLSH)**.** Let $\varepsilon \in \mathbb{R}_{\geq 0}$. The $(\varepsilon, \mathbf{d}_{\mathcal{X}})$-LapLSH mechanism $Q_{\mathsf{Lap}H}$ with a κ-bit LSH function H provides $(\varepsilon \cdot \mathbf{d}_{\mathcal{X}}, 0)$-XDP. Hence the $(\varepsilon, \mathbf{d}_{\mathcal{X}})$-LapLSH mechanism Q_{LapLSH} wrt a distribution $D_{\mathcal{H}}^\kappa$ of the κ-bit LSH functions also provides $(\varepsilon \cdot \mathbf{d}_{\mathcal{X}}, 0)$-XDP.

Proof. Since the application of an LSH function is post-processing, the proposition follows from the XDP of the Laplace mechanism. □

DOCTORAL SCHOOL

Titre : Confidentialité différentielle pour les espaces métriques: modèles théoriques de l'information pour la confidentialité et l'utilité avec de nouvelles applications aux domaines métriques

Mots clés : Confidentialité différentielle, Théorie de l'information, "Privacy" et "utility"

Résumé : La confidentialité différentielle, introduite par Dwork et al. en 2006, est devenue la référence en matière de protection de la vie privée dans les ensembles de données statistiques. Malgré sa popularité, son utilisation dans d'autres domaines a été relativement limitée. Dans cette thèse, nous explorons une généralisation de la confidentialité différentielle pour les domaines métriques appelée d-privacy. Notre approche intègre un cadre fondé sur la théorie de l'information pour analyser les flux d'informations, ce qui nous permet de fournir une caractérisation structurelle de la d-privacy et d'analyser ses propriétés de "privacy" et "utility". En utilisant l'analyse des flux d'informations, nous examinons l'ampleur de la fuite d'information induite par le paramètre epsilon de "privacy", nous trouvons une nouvelle caractérisation des mécanismes optimaux, étendant les résultats existants dans le domaine de l'optimalité universelle, et nous réexaminons le compromis "privacy-utility" pour les flux d'information dans un contexte "oblivious" et local. Enfin, nous démontrons l'applicabilité de la d-privacy à des domaines nouveaux et complexes avec des exemples d'applications à la "privacy" des documents texte, l'utilité statistique et la recherche confidentielle des "plus proches voisins".

Title : Differential privacy for metric spaces: information-theoretic models for privacy and utility with new applications to metric domains

Keywords : Differential Privacy, Information Theory, Privacy-Utility Trade-Off

Abstract : Differential privacy, introduced by Dwork et al. in 2006, has become the benchmark for data privacy in statistical datasets. Despite its widespread popularity, its use in other domains has been relatively limited. In this thesis, we explore a generalisation of differential privacy known as d-privacy, which is designed for metric domains. Our approach incorporates an information-theoretic framework for analysing information flows, which allows us to provide a structural characterisation of d-privacy and analyse its privacy and utility properties. Using information flow analysis we examine the amount of information leakage induced by the privacy parameter epsilon, we find a new characterisation for optimal mechanisms, extending existing results in the area of universal optimality, and we re-examine the privacy-utility trade-off for oblivious and local differential privacy workflows. Finally, we demonstrate the applicability of d-privacy to novel and complex domains with example applications in text document privacy, statistical utility and private nearest neighbour search.

Institut Polytechnique de Paris
91120 Palaiseau, France