Efficient Bayesian Active Learning and Matrix Modelling

Neil MT Houlsby

St. John's College

University of Cambridge

A thesis submitted for the degree of

Doctor of Philosophy

August 2014

I, NEIL MATTHEW TINMOUTH HOULSBY, confirm that *this dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text.* Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

I also confirm that this thesis is below 70,000 words and contains less than 150 figures, in fulfillment of the requirements set by the degree committee for the Department of Engineering at the University of Cambridge.

Acknowledgements

I would sincerely like to thank my supervisors, Zoubin Ghahramani and Máté Lengyel for their expert advice and kindliness with which they have guided my work. Their contributions to my research, education, and professional development has been invaluable.

I am grateful to the Google European Doctoral Fellowship for funding my PhD, and to St. John's College, Cambridge, for funding my 10th term. I enjoyed two internships at Google, and I would like to thank my mentor Massimiliano Ciaramita. It was a inspiration to work with Massimiliano, and this experience has directly influenced my career path. I would also like thank the rest of the Google Zurich summarization research team for their help and making my internships so enjoyable.

Thank you to the members of the Cambridge Computational and Biological Learning (CBL) group for numerous discussions and providing such a stimulating research environment. I would particularly like to thank Ferenc Huszár and José Miguel Hernández-Lobato with whom I have conducted much of my research. They have been fantastic colleagues to work with and I have learnt a great deal from them both.

Finally, I would like to thank Guy Houlsby, Jenny Houlsby, José Miguel Hernández-Lobato, Sina Tootoonian, Sarah Williams, Marc Deisenroth and Denise Chappell for boldly reading early versions of this thesis and providing valuable feedback.

Publications and Collaborations

This thesis includes published and unpublished work from a number of collaborations. The following paragraphs relate the chapters to publications and list the contributions of the individuals involved.

Chapter 2 contains work by NMTH, co-supervised by Máté Lengyel and Zoubin Ghahramani.

Chapter 3 contains work in collaboration with Ferenc Huszár, Michael Osbourne, ML and ZG. Part of this chapter is documented in a technical report [Houlsby *et al.*, 2011]. NMTH was the primary developer of the BALD framework. FH derived the approximation to BALD for GPC. MO invented the MGP and advised the work on GPR. NMTH derived the active GPR algorithm, wrote the code, and ran and analyzed the experiments.

Chapter 4 contains work published in two papers. The first was work with FH [Huszár & Houlsby, 2012]. Both authors contributed equally to all aspects of research, an alternative exposition is given in Huszár [2013]. The laboratory experiments were conducted by Konstantin Kravtsov, Stansilav Straupe, Igor Radchenko and Sergey Kulik at the Prokhorov General Physics Institute RAS, Moscow, and are reported in Kravtsov *et al.* [2013]. NMTH was the sole UK collaborator for this paper, and contributed the algorithm, derivations and simulations.

Chapter 5 contains work with José Miguel Hernández-Lobato and ZG, published in Hernández-Lobato *et al.* [2014b]. NMTH and JMHL contributed to all aspects of research.

Chapter 6 contains a collaboration with FH, JMH and ZG, published in Houlsby *et al.* [2012]. FH derived the preference kernel. JMH contributed the inference routine and the majority of the code. NMTH contributed to the modelling and code, ran experiments and performed analysis.

Chapter 7 is work with JMH and ZG published in Houlsby *et al.* [2014]. JMH derived the inference algorithm and ran the model comparison experiments.

NMTH derived and implemented the active learning algorithm, and ran the cold-start experiments.

Chapter 8 contains unpublished work by NMTH, supervised by ML with advice from David Stillwell and Michal Kosinski at the Cambridge Psychometrics Centre.

Finally, I would like to acknowledge the collaborators of published work conducted during my PhD that does not feature in this thesis. Collaborative work with Tomoharu Iwata and ZG is published in Iwata *et al.* [2013], this paper is closely related to the active learning work presented in this thesis, but was omitted due to space limitations. Joint work with JMH and ZG is published in Hernández-Lobato *et al.* [2014a]. Work with FH, Mohammad Ghassemi, Gergo Orbán, Daniel Wolpert and ML is in Houlsby *et al.* [2013]. Work with Guy Houlsby is published in Houlsby & Houlsby [2013], work conducted at Google with Massimiliano Ciaramita is in Houlsby & Ciaramita [2014] and work with David Blei, conducted at Princeton University, is in Houlsby & Blei [2014].

Abstract

With the advent of the Internet and growth of storage capabilities, large collections of unlabelled data are now available. However, collecting supervised labels can be costly. Active learning addresses this by selecting, sequentially, only the most useful data in light of the information collected so far. The online nature of such algorithms often necessitates efficient computations. Thus, we present a framework for information theoretic Bayesian active learning, named Bayesian Active Learning by Disagreement, that permits efficient and accurate computations of data utility. Using this framework we develop new techniques for active Gaussian process modelling and adaptive quantum tomography. The latter has been shown, in both simulation and laboratory experiments, to yield faster learning rates than any non-adaptive design.

Numerous datasets can be represented as matrices. Bayesian models of matrices are becoming increasingly popular because they can handle noisy or missing elements, and are extensible to different data-types. However, efficient inference is crucial to allow these flexible probabilistic models to scale to large real-world datasets. Binary matrices are a ubiquitous data-type, so we present a stochastic inference algorithm for fast learning in this domain. Preference judgements are a common, implicit source of binary data. We present a hybrid matrix factorization/Gaussian process model for collaborative learning from multiple users' preferences. This model exploits both the structure of the matrix and can incorporate additional covariate information to make accurate predictions.

We then combine matrix modelling with active learning and propose a new algorithm for cold-start learning with ordinal data, such as ratings. This algorithm couples Bayesian Active Learning by Disagreement with a heteroscedastic model to handle varying levels of noise. This ordinal matrix model is also used to analyze psychometric questionnaires; we analyze classical assumptions made in psychometrics and show that active learning methods can reduce questionnaire lengths substantially.

Contents

Contents **vi**

1 Introduction **1**

1.1 Introduction to Bayesian Machine Learning . . . 2

1.2 Active Learning . . . 4

1.3 Matrix Factorization . . . 5

2 Bayesian Active Learning **7**

2.1 Introduction to Active Learning . . . 7

2.1.1 When Active Learning may be Applied . . . 7

2.1.2 Query Scenarios . . . 9

2.2 Information Theoretic Active Learning . . . 10

2.2.1 Motivation . . . 10

2.2.2 Information Theory . . . 11

2.2.3 The Information Gain Utility Function . . . 13

2.3 Bayesian Active Learning by Disagreement . . . 14

2.3.1 Symmetry in the Objective . . . 14

2.3.2 Computational Advantages . . . 15

2.3.3 Simulation: Estimating the Utility from Samples . . . 17

2.3.4 Extension: Nuisance Parameters and Focused Active Learning . 19

2.3.5 Inductive and Transductive Learning . . . 19

2.3.6 Myopic Assumption and Submodularity . . . 20

2.4 Literature Review . . . 23

2.4.1 Classical Optimal Experimental Design . . . 23

2.4.2 Information Theoretic Methods . . . 26

2.4.3 Data Subsampling . . . 28

2.4.4 Decision Theoretic Methods . . . 29

2.4.5 Non-Probabilistic Methods . . . 30

2.4.6 Summary 31

3 Active Gaussian Processes **33**

3.1 Primer on Gaussian Processes 34

3.2 Active GP Classification 35

3.2.1 Experiments 38

3.2.2 Summary 46

3.3 Active GP Regression with Unknown Hyperparameters 47

3.3.1 Marginal Gaussian Process 47

3.3.2 Focused Active Learning with the MGP 48

3.3.3 Experiments 52

3.3.4 Summary 56

3.4 Conclusions and Extensions 56

4 Adaptive Quantum State Tomography **58**

4.1 Primer on Quantum Statistics 59

4.1.1 States and Density Matrices 59

4.1.2 Measurements with Probabilistic Outcomes 59

4.1.3 Infidelity 61

4.1.4 Entanglement and The Curse of Dimensionality 61

4.2 Current Experimental Designs 61

4.3 Bayesian Quantum Tomography 62

4.3.1 Sequential Importance Sampling 63

4.4 Adaptive Quantum Tomography 64

4.5 Simulations 66

4.5.1 Single Qubit Tomography 66

4.5.2 Separable and MUB Tomography of Two Qubits 69

4.6 Laboratory Experiment 70

4.6.1 Setup and Apparatus 71

4.6.2 Modelling Experimental Imperfections 72

4.6.3 Results 74

4.7 Conclusions 75

5 Stochastic Inference for Large Binary Matrices **76**

5.1 Introduction to Probabilistic Matrix Modelling 76

5.2 Limitations of Batch Inference 77

5.3 A Probabilistic Model for Binary Matrices 79

5.4 Stochastic Variational Inference for Binary Matrices 81
5.4.1 Primer on Variational Bayes 81
5.4.2 VB for Binary Matrices 82
5.4.3 SVI for Binary Matrices 83
5.4.4 Natural Gradients 84
5.4.5 Sampling Distributions for Sparse Imbalanced Matrices 86
5.4.6 Learning the Minibatch Online 87
5.4.7 The Full SIBM Algorithm 90
5.4.8 Related Work 90
5.5 Experiments 93
5.5.1 Sampling Strategies and Automatic Minibatch 94
5.5.2 Comparison to Batch and Alternative Methods 94
5.6 Conclusions and Extensions 99

6 Collaborative Preference Learning 102
6.1 The Preference Kernel 103
6.1.1 Properties of the Preference Kernel 105
6.2 Multi-User Preference Learning 106
6.2.1 Probabilistic Description of the Model 107
6.2.2 The Predictive Distribution 108
6.3 Hybrid EP-VB Inference 108
6.3.1 Primer on Expectation Propagation 109
6.3.2 Inference for Collaborative Preference Learning 110
6.3.3 Algorithmic Details 112
6.3.4 Sparse GPs for Linear Computational Time 113
6.4 Active Preference Elicitation 113
6.5 Related Multi-User Models 114
6.6 Experiments and Discussion 116
6.6.1 Comparison to Other Multi-User Models 117
6.6.2 Active Learning on Large Datasets 119
6.6.3 Tuning the Kernel Lengthscale 120
6.7 Conclusions and Extensions 121

7 Heteroscedastic Matrix Factorization for Cold-Start Learning 124
7.1 A Robust Model for Ordinal Matrix Data 125
7.1.1 Model Description 126

7.1.2 Inference 128
7.1.3 Predictive Distribution 130
7.2 Cold-Start Active Learning 131
7.2.1 Implementation of BALD 132
7.3 Related Work 133
7.3.1 Probabilistic Models for Rating Matrices 133
7.3.2 Cold-Start Learning 134
7.4 Experiments 134
7.4.1 Comparison to Other Models for Rating Data 135
7.4.2 Cold-Start Active Learning 138
7.5 Conclusions and Extensions 144

8 Bayesian Exploratory Psychometrics **147**
8.1 Background on Psychometrics 148
8.1.1 The Big Five Personality Traits 148
8.1.2 IPIP Questionnaires 148
8.2 Models and Metrics 149
8.2.1 Item Response Models 149
8.2.2 Unidimensional Models 150
8.2.3 Factor Analysis 150
8.2.4 Multidimensional Item Response Theory 151
8.2.5 Evaluation of Model Fit 152
8.2.6 A Posterior Predictive Check 154
8.3 Experiments and Analysis 156
8.3.1 Exploratory Analysis 156
8.3.2 Direct Model Comparisons 164
8.3.3 Computer Adaptive Testing 168
8.4 Conclusions and Extensions 170

9 Conclusions **173**
9.1 Summary 173
9.2 Future Work 174
9.2.1 Heteroscedastic Unsupervised Learning 174
9.2.2 Optimizing Utility over a Horizon 175
9.2.3 Meta-Learning 176

References **177**

Chapter 1

Introduction

Machine learning concerns the design of algorithms whose performance improves with accumulated data. To achieve this, a machine learning algorithm must discover patterns in the data that it can exploit. The process of pattern discovery first requires one to posit a *model*, whose structure is governed by *parameters*. Learning, or training, entails adjusting these parameters in the light of some observed data. After the model is trained, it may be used for its desired purpose; such as providing insight into the structure of the data, or making predictions about unseen datapoints. Using models to describe observations and make predictions is central to most scientific methods. The high level goal of machine learning is to automate this process and allow data, rather than human judgement, to drive learning as much as possible.

After specifying the task to be solved, machine learning systems can usually be decomposed into a three step pipeline:

i Collect the training data.

ii Propose a model, and learn its parameters using the data.

iii Use the model for its desired purpose.

These processes are most often carried out sequentially and independently. However, feedback between these steps can improve the quality of the overall system. For example, insights gained from step (iii) can be used to refine the model or learning algorithm in step (ii) [Box, 1976; Gelman *et al.*, 1996].

Similarly, step (ii) may also feedback into step (i), that is, the model may be used to influence the data collection. In *passive learning* the model has no influence over the data collection. The converse, where such feedback is present, is called *active learning*. Active learning is the focus of the first half of this thesis.

The above three steps may be divided into a number of sub-tasks. For example, step (i) can include data collection, cleansing and feature extraction. Step (ii) is normally decomposed into designing the model (this process can be data-driven, and is then called model selection), and then learning the parameters (inference). The processes of modelling and inference have different requirements. The model must be designed to capture all the relevant patterns in the data. Inference is an algorithmic problem; it entails learning the parameters accurately in a reasonable amount of time. Normally, these processed are co-designed; model expressiveness is balanced with the cost of inference. In practice, the best solutions tend to be data and task dependent. The second half of this thesis focuses upon both modelling and inference with a very common data-type: matrix data. The latter chapters also draw on the active learning methods developed in the first half.

To design robust and extensible learning algorithms, it is important to ground machine learning in rigorous mathematical theory and to understand the assumptions made during modelling and inference. This thesis is built upon the Bayesian learning framework. This methodology follows the rules of probability theory [Gelman *et al.*, 2003] and allows assumptions to be clearly encoded. However, to design useful methods, it is also important not to lose sight of step (iii) above, the intended applications for the learning algorithm [Wagstaff, 2012]. Therefore, in this thesis the proposed techniques are tied closely to practical tasks, and solutions to a number of applied problems using Bayesian active learning and matrix modelling are presented.

1.1 Introduction to Bayesian Machine Learning

A primary difficulty that must be addressed when learning from data is uncertainty. Uncertainty can arise from two sources. First, observed datasets are finite. With limited observations it is not possible determine a model's parameters exactly. This source of ambiguity will be referred to as *parameter uncertainty*. Second, uncertainty arises from unpredictable noise in the data. Such noise is often random and independent across datapoints, and therefore has no useful structure for learning and making predictions. This noise may arise from observing the data, or more generally, from randomness in the data that is not explained by the model. This uncertainty will be referred to as *observation noise* or *inherent uncertainty*.

Probability theory provides a principled framework to manipulate uncertain quantities, based upon a unique set of axioms consistent with common sense [Cox, 1946]. In probabilistic or Bayesian machine learning, all quantities, including the parameters of the model, are treated as random variables whose uncertainty is represented by their probability distribution. Bayes rule for inference arises directly from the laws of probability [Jaynes, 2003].

The core principles of Bayesian machine learning follow. Denote the model $\mathcal{M}$, and its parameters θ, before making any observations, the assumed probability distribution over the parameters is called the *prior*, $p(\theta|\mathcal{M})$. After observing data $\mathcal{D}$ the *posterior* distribution over the parameters $p(\theta|\mathcal{D}, \mathcal{M})$ is computed using Bayes' rule,

$$p(\theta|\mathcal{D}, \mathcal{M}) = \frac{p(\mathcal{D}|\theta, \mathcal{M})p(\theta|\mathcal{M})}{p(\mathcal{D}|\mathcal{M})} \,. \tag{1.1}$$

The quantity $p(\mathcal{D}|\theta, \mathcal{M})$ is known as the *likelihood* of the parameters. The likelihood indicates how well the parameters θ explain the observed data. $p(\mathcal{D}|\mathcal{M})$ is known as the *marginal likelihood* or *model evidence*, this quantity measures how appropriate the model is for the data.

To make honest probabilistic predictions on new data $\mathcal{D}^\star$, the rules of probability theory dictate that one should integrate over all sources of uncertainty. A central assumption is that the model captures all the structure in the data, so after conditioning on the parameters, the datapoints are independent. With this assumption, the predictive distribution is given by

$$p(\mathcal{D}^\star|\mathcal{D}, \mathcal{M}) = \int p(\mathcal{D}^\star|\theta, \mathcal{M})p(\theta|\mathcal{D}, \mathcal{M})d\theta \,. \tag{1.2}$$

Equation (1.2) includes both aforementioned sources of uncertainty. The parameter uncertainty is modelled by the posterior distribution $p(\theta|\mathcal{D}, \mathcal{M})$, and the observation noise is captured by the likelihood function $p(\mathcal{D}^\star|\theta, \mathcal{M})$.

The advantages of a Bayesian approach to machine learning include:

- The ability to make quantitative statements about all aspects of uncertainty through the rules of probability theory.
- The ability to deal formally with missing and noisy data. This is a corollary of the above.
- A transparent framework for encoding assumptions. Assumptions about the data generating process and assumptions about the model are separated into the likelihood function and prior distribution, respectively.
- The ability to treat any quantity of the system as a random variable. For example,

one can regard the model $\mathcal{M}$ itself as an uncertain quantity, and reason over this variable also.

- The ability to extend models formally. For example, by adding more complexity, or adapting them to new data types. This is a corollary of the previous two bullet points.

These principles provide the core framework for the work presented in this thesis on active learning and matrix modelling.

1.2 Active Learning

Collecting data is often expensive. The possible costs may include time, human effort, money, battery power etc. In these cases it is advantageous to be selective about which data to collect. As an analogy, an astronomer will choose to observe regions of the sky that they expect to be interesting since they are unlikely to discover something new by directing their telescope randomly. Similarly, active learning algorithms choose which data to collect, but they do so automatically. Whilst large collections of unlabelled data are often readily available (such as from scraping the web), active learning is particularly relevant in the context of *supervised* learning, where *annotated* or *labelled* training data is required. Labelling a datapoint may be expensive, such as in the following domains.

- Engineering systems: training data for a complex system may be expensive. For example, a model to transcribe audio requires sequences annotated at the phoneme level by an expert. A one minute sequence can take several hours to label [Zhu, 2005].

- Scientific experimentation: experiments require scientists' time, or expensive measurements. For example, optimal designs are used to minimize the length of tests on human subjects in cognitive science [Myung & Pitt, 2009].

- Measuring an environment: measurements can be financially costly, such as in hydrocarbon exploration, or require sensors with a limited capacity, for example, due to finite battery life [Osborne *et al.*, 2010].

- Interactive agents: recommender systems can provide a better service by learning about the user, however, they do not wish to over-burden the user with excessive requests for information [Boutilier *et al.*, 2002].

Active learning concerns making modelling *data efficient*, acquiring the most useful data from a limited collection budget. In addition, active learning algorithms often run online and and so must also be *computationally efficient*. This is particularly relevant if the associated labelling cost is time, such as in a scientific experiment, or an interactive system.

Chapter 2 provides an introduction to active learning. We then present a Bayesian framework for active learning, called Bayesian Active Learning by Disagreement. In many domains this framework provides computationally efficient and accurate algorithms for *information theoretic* Bayesian active learning [MacKay, 1992b]. Using this approach, Chapter 3 presents new active learning algorithms for Gaussian processes, a popular supervised machine learning model [Rasmussen & Williams, 2005]. Next, in Chapter 4 this framework is used to tackle an applied problem, quantum tomography, for which we present a new adaptive design. As well as simulations, laboratory experiments show that our adaptive algorithm yields substantially more data-efficient experiments than current designs. During this PhD, we addressed another application, active data visualization. However, for space reasons this work is omitted, but is presented in Iwata *et al.* [2013]. The active learning techniques are developed further with the matrix models in Chapters 6, 7 and 8.

1.3 Matrix Factorization

Numerous sources of data can be represented a matrices. These include any data that can be expressed in tabular form, where elements are associated with a particular row or column, but the rows and columns can be permuted arbitrarily. There are three common matrix data-types:

- Real valued matrices, such as the levels of gene expressions. Here, rows correspond to samples and the columns to genes [Devarajan, 2008].

- Ordinal matrices, such as movie ratings [Bennett & Lanning, 2007] or responses to questionnaires [Goldberg, 1999]. Each row corresponds to a user and each column to an item or question.

- Binary matrices, such as connectivity matrices of unweighted graphs, market basket data [Brin *et al.*, 1997], or pairwise preference data [Fürnkranz & Hüllermeier, 2003].

Many popular approaches to modelling matrix data assume a low rank structure. A classical algorithm for fitting a low rank matrix to a dataset is Singular Value Decomposition (SVD). However, real-world matrices are usually sparse, most of the elements are missing. Furthermore, the observed entries are often corrupted by noise. In these cases, classical algorithms such as SVD can produce poor factorizations. Probabilistic methods can provide improve factorizations. This is because they are well equipped to handle uncertainty, and so they often exhibit strong performance in tasks involving matrix data, such as recommendation [Salakhutdinov & Mnih, 2008]. Bayesian models for matrix factorization are therefore becoming increasingly popular.

The second half of this thesis presents advances in probabilistic modelling and inference with matrix data. In Chapter 5 binary matrices are considered. These are typically very large, necessitating computational efficiency, for which we develop a new inference algorithm. Preference data is an abundant source of binary data as it can be collected implicitly from user behaviour. Chapter 6 presents a new model for learning from multiple users' preferences simultaneously, incorporating side-information where available. Chapter 7 addresses the "cold-start" problem in recommender systems. This chapter brings together matrix modelling and active learning techniques to learn efficiently (in a 'data' and 'computational' sense) about new users or items, for whom one has no previous information. Finally, Chapter 8 presents a study on psychometric questionnaires. With the probabilistic model for ordinal-valued ratings developed in Chapter 7, some of the fundamental assumptions in psychometric analysis are re-visited, and active learning is used to reduce questionnaire lengths.

Chapter 2

Bayesian Active Learning

This chapter introduces the Bayesian framework used to design the active learning algorithms in this thesis. This framework takes an information-theoretic approach to active learning. First, in Section 2.1 we introduce active learning and then describe the Bayesian information theoretic approach in Section 2.2. We discuss possible computational difficulties, and present our framework which can circumnavigate these, called Bayesian Active Learning by Disagreement (BALD), in Section 2.3. We discuss properties of the framework, such as computational complexity and submodularity. The chapter finishes with a review of related literature in active learning and experimental design that has developed in statistics, experimental science, social science, computer science and machine learning.

2.1 Introduction to Active Learning

We first introduce active learning, discuss when it is applicable, and present the different settings for active querying.

2.1.1 When Active Learning may be Applied

Data, particularly labelled data, can be expensive to obtain. Therefore, it is desirable to collect only the most useful points. To address this, active learning algorithms choose their training data. These are 'active' in the sense that they can adjust which points they choose in light of data collected so far.

A concern may arise: when choosing the data to learn from, can one bias the inferences made? Fortunately, this is not the case, provided that we condition our inferences on the actively selected variables. More concretely, consider the setting where we choose

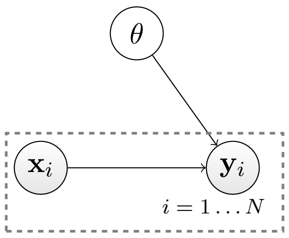

Figure 2.1: Graphical model for a discriminative learner. White nodes indicate latent (unobserved) variables and shaded nodes denote observed variables. Note that the distribution of the input $\mathbf{x}$ is assumed to be independent of the parameters θ.

a query $\mathbf{x} \in \mathcal{X}$, and observe a variable $\mathbf{y} \in \mathcal{Y}$ in response. These quantities shall be referred to as the *input* and *output* variables respectively.[1] Suppose that we also have a model with parameters $\theta \in \Theta$ that describes the dependence of the output on the input, $p(\mathbf{y}|\mathbf{x}, \theta)$. This setting is called *discriminative learning*, see Figure 2.1. Provided that we condition on $\mathbf{x}$, we may perform Bayesian computations without introducing inferential bias into posterior distribution $p(\theta|\mathbf{y}, \mathbf{x})$ or the predictive distribution $p(\mathbf{y}|\mathbf{x})$[2]. More specifically, when computing these quantities, we must compute the likelihood function $p(\mathbf{y}|\mathbf{x}, \theta)$. When computing the likelihood we implicitly integrate over all of the unobserved data. Ignoring this integral does not effect the likelihood provided that we condition on all observed data, regardless of how it was collected, and do not filter it based upon the output variable $\mathbf{y}$.

The alternative to discriminative learning is *generative learning* in which the entire data distribution $p(\mathbf{x}, \mathbf{y})$ is modelled [Ng & Jordan, 2002]. Learning a full generative model actively is not possible because one does not observe an unbiased sample from $p(\mathbf{x})$. However, this terminology may be confusing. For example, probabilistic models of matrix data are normally termed 'generative' as the data is a matrix $\mathbf{Y}$, and matrix models typically generate the entire dataset, they learn $p(\mathbf{Y}|\theta)$. However, active learning with matrices can be reconciled with the framework in Figure 2.1. Here, we select a row and column index and observe that entries' value. The location of each matrix entry corresponds to the input variable $\mathbf{x}$, and we do not model the entries' locations directly, but we condition upon them. However, this variable is usually implicit in the equations.

[1] $\mathcal{X}, \mathcal{Y}$ can be general input and output spaces. Unless stated otherwise, $\mathcal{X}$ will be a real-valued vector space $\mathcal{X} = \mathbb{R}^D$. $\mathcal{Y}$ will usually be uni-dimensional, but may be binary, real-valued, ordinal etc.

[2] In this context we use the term 'bias' loosely to mean that the distributions differ to those computed by integrating all of the unobserved data, not in the formal sense of a biased statistical estimator.

A second distinction that is often made is between *supervised* and *unsupervised* learning. In supervised learning there are clearly defined input and output variables, and the goal is to predict the output corresponding to new, unseen inputs. Supervised learning fits well into the discriminative setting in Figure 2.1. Supervised active learning is the focus of this thesis, although the methods could be extended to unsupervised learning by choosing a partition of the variables and learning the conditional distributions actively.

2.1.2 Query Scenarios

Active learning can be performed in three scenarios: *continuous sampling*, *pool based* and *stream based* learning. In continuous sampling any point in input space may be selected. This is appropriate when the input space can be queried to arbitrary precision. For example, in cognitive science where the queries may be computer generated stimuli with real-valued parameters, or when learning the kinematics of a robot arm by choosing joint angles and measuring hand coordinates [Cohn *et al.*, 1996]. However, in other regimes continuous querying is inappropriate. In a handwriting recognition system, Baum & Lang [1992] found that many generated query images contained no recognizable symbols and hence could not be labelled. In Chapter 4 we perform continuous sampling, adjusting real-valued parameters of a physics experiment to select continuous measurements.

In pool based active learning one has access to a set (pool) of unlabelled data from which to select points for annotation. For example, in a study on the effects of smoking on cancer rates, the pool contains people whose age, smoking habits etc. are known, and the scientist wants to select only the most informative few for expensive clinical trials. This is probably the most common scenario in machine learning, occurring in many applications including text classification [Tong & Koller, 2002], image classification [Tong & Chang, 2001], speech recognition [Tur *et al.*, 2005], cancer diagnosis [Liu, 2004] and recommendation systems [Jin & Si, 2004].

Stream based active learning is closely related to pool based learning, except that the pool is presented sequentially and the algorithm must decide online whether or not to collect the point's label. A famous example is 'triggering' on the CERN particle accelerator; there is insufficient storage capacity to store the vast stream of events, so only potentially interesting measurements are recorded [Aad *et al.*, 2012].

Active learning algorithms must assign a score, or utility, to each location in input space that may be queried. Continuous sampling is the most flexible, but requires optimization of the utility function over input space. This may involve a hard high dimensional, multi-modal optimization problem. If the utility function is not differentiable, continuous optimization may be infeasible. In pool based active learning the utility is normally evaluated for every point in the pool, but if the pool is large, or the utility is expensive to evaluate, pruning may be required [Roy & McCallum, 2001]. We now present general strategies for active learning, these are usually applicable in any of the three scenarios, but the computational issues above should be considered in practice.

2.2 Information Theoretic Active Learning

Probabilistic active learning broadly divides into two categories, *decision* and *information* theoretic. We take the information-theoretic approach, which we first motivate.

2.2.1 Motivation

In Bayesian methods, it is common to separate learning from decision making. This permits general models and learning algorithms to be used in a variety of tasks. Learning, or inference[1], involves applying Bayes' rule (1.1) to compute the posterior distribution of the parameters of the model $p(\theta|\mathcal{D})$. Decision making involves choosing an action a, which incurs a particular loss that depends on the true parameters of the system $\hat{\theta}$, denoted $\mathcal{L}(\hat{\theta}, a)$. However, in practice the true parameters are unknown. Therefore, the optimal course of action, on average, is to minimize the expected loss given our beliefs about the parameters,

$$\mathcal{R}_{\mathcal{D}}(a) = \mathbb{E}_{p(\theta|\mathcal{D})}\mathcal{L}(\theta, a)\,. \tag{2.1}$$

$\mathcal{R}_{\mathcal{D}}(a)$ is called the Bayes posterior risk. Inference and decision making can be separated because the posterior is not a function of the loss. This can be beneficial because the same posterior can be used to solve different tasks.[2]

Active learning algorithms need to quantify how 'useful' datapoints are. To do this, they are equipped with a utility function $\mathcal{U}(\mathbf{x}) : \mathcal{X} \to \mathbb{R}$. The optimal utility

[1] Sometimes these terms are distinguished. Inference is often used to refer to computing the posterior distribution over variables local to each datapoint. Learning then refers to optimizing or computing the posterior over global model parameters. Unless context dictates otherwise, these terms will be used interchangeably.

[2] Recent work couples inference with the loss function in order to incur lower loss when the posterior can only be approximated [Abbasnejad *et al.*, 2013; Lacoste-Julien *et al.*, 2011].

function, from a loss minimization standpoint, minimizes the expected posterior risk after observing the output $\mathbf{y}$ corresponding to input $\mathbf{x}$,

$$\mathcal{U}(\mathbf{x}) = \mathbb{E}_{p(\mathbf{y}|\mathbf{x},\mathcal{D})} \operatorname*{argmin}_{a} \mathcal{R}_{\{\mathcal{D},\mathbf{x},\mathbf{y}\}}(a) \,. \tag{2.2}$$

Choosing samples to maximize Equation (2.2) is known as *decision theoretic* active learning [Kapoor *et al.*, 2007; Roy & McCallum, 2001].

Decision theoretic active learning, although theoretically optimal in terms of expected loss minimization, can be disadvantageous for two reasons. First, the utility function is tied to a particular task and loss function. These may not be known ahead of time, or we may desire a learning algorithm that can perform well with a variety of loss functions. Second, computing the expected change in risk can be expensive. This is because the utility function in Equation (2.2) includes both the learning and decision making processes. Thus, to compute Equation (2.2) one must calculate the expected change in beliefs and then compute the new optimal decision after including any new data $\{\mathbf{x}, \mathbf{y}\}$.

An alternative is to focus upon learning alone, and choose data that is most useful for inferring the parameters θ. This approach decouples learning from future decision making which is consistent with many Bayesian methods. To measure the utility of a datapoint, we must quantify its 'informativeness' about the parameters. Information theory is a natural tool for doing this, so this approach is called *information theoretic* active learning.

2.2.2 Information Theory

Before presenting information-theoretic active learning, a brief overview of elementary information theory is provided. For a complete introduction see Cover *et al.* [1994]. Information theory was founded by Claude Shannon who studied data transmission over noisy communication channels [Shannon, 1948]. Shannon derived a theoretical upper bound for the capacity of a channel, which is the maximum rate that a set of symbols $X = \{x_1 \ldots x_N\}$ which have distribution $P(X)$ can be transmitted with zero reconstruction error. To do this he defined the *information content* in a symbol x, and the *entropy*, which is the average information content in the ensemble.

$$\text{Information content: } \mathrm{J}(x) = -\log P(x) \,, \tag{2.3}$$

$$\text{Entropy: } \mathrm{H}[P(X)] = -\sum_{x} P(x) \log P(x) \,. \tag{2.4}$$

The usual shorthand $P(x) \equiv P(X = x)$ will be used to denote the probability that X takes a particular value x when context makes the usage clear. When unambiguous, $\mathrm{H}[X]$ will be used to denote the entropy of the distribution $P(X)$. Entropy is a measure of uncertainty in a distribution that satisfies two intuitive desiderata. First, $\mathrm{H}[X] \geq 0$, and takes value zero only when all of the mass in $P(X)$ is concentrated on a single symbol. In this case the outcome of sampling from $P(X)$ is certain. Furthermore, $\mathrm{H}[X]$ is maximized when $P(X)$ is a uniform distribution. Intuitively, this corresponds to maximal uncertainty. The second desiderata is that entropy is additive for independent random variables. If X and Y are independent, then $\mathrm{H}[P(X, Y)] = \mathrm{H}[P(X)]+\mathrm{H}[P(Y)]$. Intuitively, if we have n units of uncertainty in X and m units of uncertainty in Y, then we have $n+m$ units of uncertainty in their joint distribution. The base of the logarithm in Equations (2.3) and (2.4) changes the quantities by a multiplicative constant, with base two the units of information are called bits. When measured in bits, the entropy may be interpreted as "The average number of binary questions that must be asked to determine the value of a sample from $P(X)$."

Shannon's entropy can be extended to distributions over continuous variables by replacing the sum in Equation (2.4) with an integral. This quantity is known as the differential entropy.[1]

$$\text{Differential entropy: } \mathrm{H}[p(X)] = -\int p(x) \log p(x) dx \,,$$

where X now denotes a continuous random variable and $p(X)$ is a continuous probability density function. Again, this quantity is maximized when $p(X)$ is a uniform distribution over the domain of X, and it is minimized when $p(X)$ is a Dirac delta function $\delta(x - x')$. However, in this case the differential entropy equals $-\infty$, because x can be specified to infinite precision and hence can carry an infinite amount of information.

Two further information theoretic quantities that occur frequently are the *mutual information*, and *Kullback-Leibler* (KL) divergence. The mutual information between

[1] It may concern some that, unlike entropy in Equation (2.4), the differential entropy is not a dimensionless quantity. However, it be thought of as the KL divergence, Equation (2.7), to an improper uniform distribution $u(x)$, $\mathrm{H}[p(x)] = -\int p(x) \log \frac{p(x)}{u(x)} dx$, which is dimensionless.

two random variables X and Y is

$$\text{mutual information: } \mathrm{I}[X,Y] = \mathrm{H}[p(X)] - \mathbb{E}_{p(Y)}\mathrm{H}[p(X|Y)] \tag{2.5}$$
$$= \mathrm{H}[p(Y)] - \mathbb{E}_{p(X)}\mathrm{H}[p(Y|X)]$$
$$= \mathrm{H}[p(X)] + \mathrm{H}[p(Y)] - \mathrm{H}[p(X,Y)]\,, \tag{2.6}$$

where $\mathbb{E}_{p(Y)}\mathrm{H}[p(X|Y)]$ is the *conditional entropy*, and is often denoted $\mathrm{H}[X|Y]$. Mutual information is a non-negative quantity that is symmetric in its arguments. It measures how much information X carries about Y, and vice versa. Shannon showed that the maximum possible capacity of a channel is given by the mutual information between the sent and received signals. The mutual information equals zero if and only if X and Y are statistically independent, that is $p(X,Y) = p(X)p(Y)$.

The KL divergence is a measure of dissimilarity between two probability distributions $p(X)$ and $q(X)$,

$$\text{KL divergence: } \mathrm{KL}[p(X)||q(X)] = \int p(x)\log\frac{p(x)}{q(x)}dx\,. \tag{2.7}$$

This divergence is non-negative, but asymmetric. It is equal to zero if and only if $p(X)$ is identical to $q(X)$. Intuitively, the KL divergence is the number of additional bits needed to transmit symbols with distribution $p(X)$, if our model of the distribution is $q(X)$.

Entropy is a well established characterization of uncertainty in a probability distribution and provides a natural metric for information theoretic active learning.

2.2.3 The Information Gain Utility Function

In information theoretic Bayesian active learning one is agnostic to future decision tasks and loss functions. The goal is to learn as quickly as possible about the model parameters θ. With Shannon's entropy (2.4) to quantify the uncertainty in a probability distribution, the natural objective is to minimize posterior entropy after collecting data. However, if we collect many datapoints in sequence, optimizing this quantity is NP-hard in the horizon length [Ko *et al.*, 1995; Krause & Guestrin, 2005]. Therefore, as is common in sequential decision making, we take a myopic (greedy) approach, selecting the next point as if it were the last. The implications of the myopic approximation are discussed in Section 2.3.6. The myopic expected information gain is given by

$$\mathcal{U}(\mathbf{x}) = \mathrm{H}[p(\theta|\mathcal{D})] - \mathbb{E}_{p(\mathbf{y}|\mathbf{x},\mathcal{D})}\mathrm{H}[p(\theta|\mathcal{D},\mathbf{x},\mathbf{y})]\,.^{1} \tag{2.8}$$

The expectation over $\mathbf{y}$ is taken because the output is unknown before the query has been made. Equation (2.8) was first proposed for the design of Bayesian experiments in Lindley [1956]. However, just as Bayes' rule itself is usually intractable, Equation (2.8) is difficult to compute with most interesting models. Therefore, much recent work has addressed information theoretic active learning; mathematical approximations and algorithmic techniques have been developed to apply Equation (2.8) to complex models.

As an aside: another intuitive information-theoretic objective is to maximize the KL-divergence between the current and next posterior, $\mathrm{KL}[p(\theta|\mathcal{D},\mathbf{x},\mathbf{y})||p(\theta|\mathcal{D})]$. However, after taking expectations with respect to $\mathbf{y}$ this utility function is equivalent to entropy decrease in Equation (2.8) [MacKay, 1992b].

2.3 Bayesian Active Learning by Disagreement

We now present an entirely equivalent formulation of Equation (2.8). This reformulation can provide substantial practical advantages that shall reappear throughout this thesis. We then discuss various properties of this approach to active learning including computational issues, extensibility, inductivity and submodularity. Inductivity and submodularity are general properties of information-theoretic active learning. The computational properties and extensions are specific to our method used to compute the utility, Bayesian Active Learning by Disagreement.

2.3.1 Symmetry in the Objective

An important insight arises if we note that Equation (2.8) is equivalent to the mutual information (2.5) between parameters and the unobserved output, conditioned upon the input and the observed data $\mathrm{I}[\theta;\mathbf{y}|\mathcal{D},\mathbf{x}]$. Because mutual information is symmetric in its arguments, Equation (2.8) can be re-written as follows,

$$\begin{aligned}\mathcal{U}(\mathbf{x}) &= \mathrm{H}[p(\theta|\mathcal{D})] - \mathbb{E}_{p(\mathbf{y}|\mathbf{x},\mathcal{D})}\mathrm{H}[p(\theta|\mathcal{D},\mathbf{x},\mathbf{y})] \\ &= \mathrm{I}[\theta,\mathbf{y}|\mathcal{D},\mathbf{x}] \qquad (2.9) \\ &= \mathrm{H}[p(\mathbf{y}|\mathbf{x},\mathcal{D})] - \mathbb{E}_{p(\theta|\mathcal{D})}\mathrm{H}[p(\mathbf{y}|\mathbf{x},\theta)]\,. \qquad (2.10)\end{aligned}$$

Equation (2.10) is equivalent to the expected information gain in Equation (2.8), but it provides a different intuition about the utility function. The first term in Equation (2.10) seeks the input $\mathbf{x}$ for which the model has high uncertainty about output

[1]Note that the first term is independent of $\mathbf{x}$, and is therefore constant, this term is included for clarity in subsequent sections.

$\mathbf{y}$. That is, the output has a high *marginal entropy*, $\mathrm{H}[p(\mathbf{y}|\mathbf{x}, \mathcal{D})]$. However, given any particular parameter value θ drawn from the posterior, Equation (2.10) seeks a datapoint with low expected *conditional uncertainty*, $\mathbb{E}_{p(\theta|\mathcal{D})}\mathrm{H}[p(\mathbf{y}|\mathbf{x}, \theta)]$. This relates to the two sources of uncertainty described in Section 1.1: parameter uncertainty and observation noise. Equation (2.10) will reward datapoints whose output has high entropy due to parameter uncertainty, which is captured by the marginal predictive distribution $p(\mathbf{y}|\mathbf{x}, \mathcal{D})$, but penalizes uncertainty due to inherent noise, which is modelled by the likelihood $p(\mathbf{y}|\mathbf{x}, \theta)$. Equivalently, Equation (2.10) can be interpreted as seeking the $\mathbf{x}$ for which the parameters under the posterior make confident predictions, but these predictions are highly diverse. That is, the parameters *disagree* about the output $\mathbf{y}$, hence we name this formulation Bayesian Active Learning by Disagreement (BALD).

The equivalence follows from straightforward application of the properties of Shannon's entropy and has been noted in the literature [Lindley, 1956; Shewry & Wynn, 1987]. A number of active learning algorithms are derived starting from the mutual information in Equation (2.9) between observed variables and 'variables of interest' [Caselton & Zidek, 1984; Ertin *et al.*, 2003; Krause *et al.*, 2008], and thus may compute this quantity in either direction. However, the use of BALD, Equation (2.10), as an general-purpose alternative to posterior entropy minimization is not widely discussed. We argue that this reformulation is a valuable tool for information theoretic active learning, and it has a number of practical advantages. Noting the direct equivalence to information gain allows us to relate and formalize a number of popular active learning methods (Section 2.4). By working in this framework we derive new algorithms for specific domains in the subsequent chapters. However, we highlight BALD in its general form as a starting point for active learning algorithms in many other possible domains.

2.3.2 Computational Advantages

The BALD reformulation in Equation (2.10) can provide a number of computational advantages over direct entropy minimization in Equation (2.8). We discuss the immediate advantages in this section. In the next section we investigate empirically a further computational advantage from improved sample complexity if Monte Carlo is used to estimate the utility function.

A principal difference between Equations (2.8) and (2.10), is that Equation (2.10) computes entropies in output space $\mathcal{Y}$ rather than parameter space Θ. Often parameter space is high dimensional and so posterior entropies are usually intractable. One possibility is to use a histogram estimator [Paninski, 2003], but the number of bins scales exponentially with the dimensionality of parameter space. Another possibility is to sample from the posterior and estimate Equation (2.8) using these samples. However, estimating entropies directly from samples in an unbiased manner is notoriously hard [Panzeri *et al.*, 2007]. Therefore, the intractable posterior entropy is often approximated directly from approximations to the posterior [Herbrich *et al.*, 2002; Krishnapuram *et al.*, 2004; Lewi *et al.*, 2007; MacKay, 1992b].

In Bayesian nonparametric models, such as Gaussian processes (GPs), parameter space is infinite dimensional and so entropies in parameter space are ill-defined. However, output space is often simple. For example, in regression and classification, the output is usually a uni-dimensional real valued or binary variable, respectively. The entropy of these scalars is usually straightforward to compute, either analytically or with a one-dimensional grid. Therefore, BALD can often compute expected changes to posterior entropy exactly, even with infinite dimensional parameter spaces. Furthermore, because Equation (2.10) does not compute the entropy of the posterior distribution directly, estimating information gain in this way is not tied to a particular method for approximating the posterior. Section 2.3.3 presents an empirical study on a toy problem to demonstrate that $\mathcal{U}(\mathbf{x})$ can be estimated more accurately from posterior samples with BALD than with direct estimation using Equation (2.8).

A second advantage that BALD has over the original formulation is that the posterior does not need to be updated until *after* a query has been made. Equation (2.8) requires calculating the updated posteriors after including every possible new datapoint $p(\theta|\mathcal{D}, \mathbf{x}, \mathbf{y})$. However, only the current posterior $p(\theta|\mathcal{D})$ appears in Equation (2.10). Suppose we have a pool of N possible inputs, and can receive one of $|\mathcal{Y}|$ possible outputs – for continuous $\mathbf{y}$ output space could be partitioned into a grid. To compute Equation (2.8) we must calculate $N|\mathcal{Y}|$ new posterior distributions. This will usually require expensive approximate inference. BALD requires updating the posterior only once per sample, after labelling the datapoint. This is the same as number of updates required by *passive* online learning. If c^{inf} is the cost of inference, and h^{θ} is the cost to compute the entropy of the posterior, then the complexity to compute Equation (2.8) on N candidates is $\mathcal{O}(N|\mathcal{Y}|c^{\text{inf}} + N|\mathcal{Y}|h^{\theta})$. With BALD, the cost is $\mathcal{O}(c^{\text{inf}} + Nc^{\text{pred}} + Nh^{\mathbf{y}})$, where c^{pred} is the cost to compute the predictive distribution, and $h^{\mathbf{y}}$ is the cost to compute the entropies in output space. For many models, $c^{\text{pred}} \ll c^{\text{inf}}$ and $h^{\mathbf{y}} \ll h^{\theta}$.

Like any algorithm, BALD does not offer a free lunch. If output space is complicated, such as in structured models [Tsochantaridis *et al.*, 2004], then direct entropy minimization may be easier. Even if this is not the case, the terms in Equation (2.10) may still be non-trivial to compute. In practice, the first term is usually straightforward because the marginal predictive distribution $p(\mathbf{y}|\mathbf{x}, \mathcal{D})$ is central to most applications of Bayesian machine learning. Therefore techniques for computing this quantity have been developed for many models. However, the second term, the posterior mean conditional entropy $\mathbb{E}_{p(\theta|\mathcal{D})}\mathrm{H}[p(\mathbf{y}|\mathbf{x}, \theta)]$ is more unusual. As we shall see in the following chapters, computing this term is normally the main challenge when implementing BALD.

2.3.3 Simulation: Estimating the Utility from Samples

A toy linear-Gaussian model is used to investigate empirically the accuracy of estimating $\mathcal{U}(\mathbf{x})$ from posterior samples, either directly (2.8), or with BALD (2.10). The likelihood function is $p(y|\mathbf{x}, \theta) = \mathcal{N}(y; \theta^\top \mathbf{x}, \sigma^2)$, where the parameters θ and input $\mathbf{x}$ are 10-dimensional real valued vectors and the output y is a scalar: $\mathcal{X} = \mathbb{R}^{10}$,$\mathcal{Y} = \mathbb{R}$, $\Theta = \mathbb{R}^{10}$. $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma)$ denotes a (in general, multivariate) Gaussian distribution over $\mathbf{x}$ with mean $\boldsymbol{\mu}$ and covariance matrix Σ. The noise level σ^2 is fixed and known. With a Gaussian prior on θ, the posterior is tractable and Gaussian, so its entropy can be computed analytically to assess the quality of the sampling approximations. Furthermore, in this model the posterior entropy is independent of the future observations, so $\mathcal{U}(\mathbf{x})$ can be computed without simulating any y values.

Even a 10-dimensional parameter space is impractical to grid up, so we approximate Equation (2.8) using Monte Carlo samples from the posterior,[1]

$$\mathrm{H}[p(\theta|\mathcal{D})] \approx -\frac{1}{N}\sum_{i=1}^{N} \log p(\theta_i|\mathcal{D}), \quad \theta_i \sim p(\theta|\mathcal{D}), \tag{2.11}$$

where N is the number of samples. Note that we do not have to integrate over y as the output value does not influence the posterior entropy. We may also use these samples to estimate $\mathcal{U}(\mathbf{x})$ using BALD,

$$\mathcal{U}(\mathbf{x}) \approx \mathrm{H}\left[\frac{1}{N}\sum_{i=1}^{N} p(y|\mathbf{x}, \theta_i)\right] - \frac{1}{N}\sum_{i=1}^{N} \mathrm{H}[p(y|\mathbf{x}, \theta_i)]. \tag{2.12}$$

With this simple linear-Gaussian model the entropy of the second term in Equation (2.12) is constant with respect to $\mathbf{x}$. The first term is the entropy of a mixture of N one-dimensional Gaussians, which can be computed using a one-dimensional grid.

We define the approximation error as the normalized difference in utility between

[1] In practice posterior samples will have often been computed already from the approximate inference algorithm. This is the case in Chapter 4.

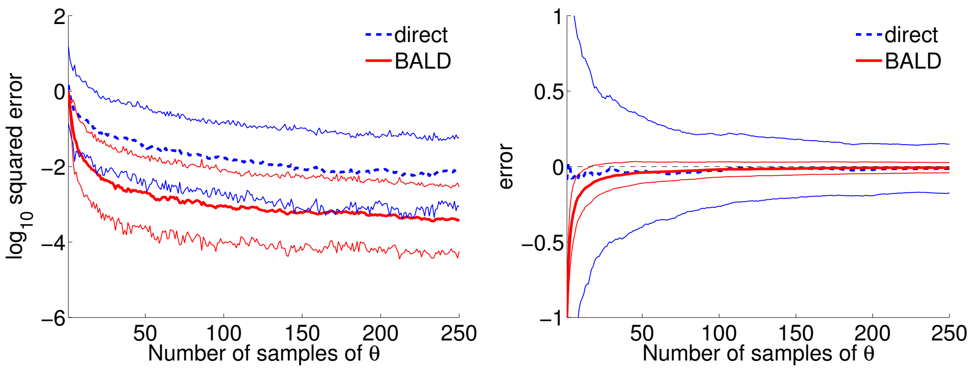

Figure 2.2: Error in the estimation of $\mathcal{U}(\mathbf{x})$ using samples directly (2.11) or with BALD (2.12). Thicker lines give the mean and thinner dash-dot lines indicate ± 1 s.d. over 200 repeats. The x-axis is the number of samples. *Left:* log squared error. *Right:* error, $\mathcal{U}_{\text{approx}}(\mathbf{x}) - \mathcal{U}_{\text{true}}(\mathbf{x})$.

the truth, computed analytically, and the Monte Carlo approximations,

$$\text{error} = \frac{\mathcal{U}_{\text{approx}}(\mathbf{x}) - \mathcal{U}_{\text{true}}(\mathbf{x})}{\mathcal{U}_{\text{true}}(\mathbf{x})}\,. \tag{2.13}$$

The experiment was repeated 200 times, resampling $\mathbf{x}$ each time. Figure 2.2 shows the distribution of errors over these runs as a function of the number of samples used to estimate $\mathcal{U}(\mathbf{x})$ by either method. The left-hand plot in Figure 2.2 shows that BALD provides a more accurate estimate of the information gain as the squared estimation error is over an order of magnitude smaller than with direct estimation of posterior entropy.

However, the right-hand plot in Figure 2.2 demonstrates that with very few samples (< 50) BALD tends to underestimate the entropy change. This is probably due to the mixture of Gaussians used to estimate the marginal predictive entropy (the term of Equation (2.12)). The true predictive distribution has infinitely many components. Each component has the same variance, and with few samples the finite mixture approximation is likely to yield an underestimate. In the extreme case of one sample, BALD will always underestimate this marginal entropy and will return a utility of zero. However, as indicated by the error bands, BALD yields an estimate with smaller variance and the bias is much smaller than the standard deviation of the direct estimate.

2.3.4 Extension: Nuisance Parameters and Focused Active Learning

In many settings some parameters are more important to learn than others. Here, in addition to the parameters of primary interest θ, the model has some other 'nuisance parameters' of lesser importance ϕ. For example, in Bayesian models, ϕ may correspond to hyper-parameters of the prior. Rather than collecting data that provides maximal information about the joint distribution over all of the parameters $p(\theta, \phi|\mathcal{D})$, we would like to focus learning efforts on the parameters of interest alone. In particular, we want to minimize the entropy of the marginal distribution $p(\theta|\mathcal{D})$. Equation (2.10) can be directly extended to this scenario,

$$\mathrm{I}[\theta; \mathbf{y}|\mathbf{x}, \mathcal{D}] = \mathrm{H}\left[\int p(\theta, \phi|\mathcal{D})d\phi\right] - \mathbb{E}_{p(\mathbf{y}|\mathbf{x},\mathcal{D})}\mathrm{H}\left[\int p(\theta, \phi|\mathbf{x}, \mathbf{y}, \mathcal{D})d\phi\right] \tag{2.14}$$

$$= \mathrm{H}[p(\mathbf{y}|\mathbf{x}, \mathcal{D})] - \mathbb{E}_{p(\theta|\mathcal{D})}\mathrm{H}[\mathbb{E}_{p(\phi|\theta,\mathcal{D})}p(\mathbf{y}|\mathbf{x}, \theta, \phi)]\,, \tag{2.15}$$

where Equation (2.14) is the information gain in the marginal written explicitly and Equation (2.15) is the computationally efficient rearrangement. The first term in this 'focused' BALD utility function is the entropy of the marginal predictive distribution, as in the original BALD formula (2.10). The second term differs, the integral over the nuisance parameters has moved inside the entropy. Intuitively, this only penalizes datapoints about whose output we are still uncertain if we know θ, but not ϕ. We only seek disagreement between the parameters of interest θ, and hence focus our efforts on refining the distribution over these parameters only.

In the next two sections we discuss two properties of posterior information-gain for active learning: inductivity and submodularity.

2.3.5 Inductive and Transductive Learning

In learning theory, *inductive* and *transductive* learning are distinguished [Vapnik, 2000]. Posterior information gain (2.8), and hence BALD, is an inductive approach. Inductive methods address the generic task of generalization from samples, whereas transductive methods minimize the expected loss on a particular test set or distribution. A transductive algorithm needs access to test-time information during training and exploits this information. Therefore, inductive algorithms seek good performance in a variety of possible test scenarios, but are not optimal for any single setting.

This distinction between inductive and transductive algorithms applies in active learning [Tong, 2001; Yu *et al.*, 2006]. It relates to the distinction between information and decision theoretic methods. Decision theoretic algorithms are inherently transductive because they focus on loss minimization on a particular test distribution. However, information-theoretic transductive algorithms have also been proposed. These minimize the average *predictive* variance or entropy over regions of interest in input space [Cohn *et al.*, 1996; Krause *et al.*, 2008; MacKay, 1992b]. Note that this is not the same as BALD, which uses predictive entropies to minimize the *parameter* entropy.

These approaches have relative advantages and disadvantages. If the test inputs are known, transductive methods are likely to yield a smaller test loss because they focus on learning in that region. Inductive methods may choose to learn about parameters that have a small effect on the predictions in the interest region. However, transductive methods first require the interest region to be known, then a distribution over it to be defined. This is usually done using a set of reference points. These points may be placed in a grid [Krause *et al.*, 2008], but this is impractical in higher dimensions. Alternatively, samples from the pool may be used. However, in this case an inductive algorithm will naturally focus on the region of interest as well because the samples used to train the model are also from the pool.

An advantage of inductive methods is that information about future inputs is not required. Another advantage is by maximizing information gain in the parameters one can choose which parameters to learn actively using Equation (2.15). With transductive methods, the parameters are learnt selectively, but the algorithm designer cannot select them directly. Directly choosing parameters to focus learning on can be advantageous if: i) The values of particular parameters themselves are of primary interest. ii) Selective learning allows the algorithm to be improved in other ways. For example, in Chapter 3 we find that choosing the order in which parameters are actively learnt appropriately is crucial for robust active GP regression. In Chapter 7 we learn about a new user in a recommendation system who is represented by a few parameters in a much larger model.

2.3.6 Myopic Assumption and Submodularity

The utility functions presented so far have been *myopic* or *greedy*. That is, they seek the optimal $\mathbf{x}$ as if it were the last datapoint to be selected. However, often one has a budget to collect a set of L samples, X, and receive a set of labels Y. In this case we want to maximize information gain over the entire set,

$$\mathcal{U}(X) = \mathrm{H}[\theta|\mathcal{D}] - \mathrm{H}[\theta|Y, X, \mathcal{D}] \,. \tag{2.16}$$

Unfortunately, optimizing Equation (2.16) is, in general, NP-hard in the horizon length L [Ko *et al.*, 1995; Krause & Guestrin, 2005]. Therefore, a common approach is to greedily maximize the utility with respect to the next sample alone [Dasgupta, 2005; Heckerman *et al.*, 1994]. Fortunately, under certain conditions, the greedy strategy is near-optimal. We now discuss when these conditions apply to information theoretic active learning.

The L-step utility (2.16) is a set function. A set function $F(Y) : 2^{\mathcal{Y}} \mapsto \mathbb{R}$ is a function whose input is a set Y and (usually) outputs a real value. Submodularity is an extension of convexity to set functions. Intuitively, submodular set functions exhibit 'diminishing returns'. More precisely, adding an element y to a set Y^{small} produces a greater increase to F than adding y to Y^{large}, where $Y^{\text{large}} \supset Y^{\text{small}}$. A set function is submodular if for all Y^{large} and Y^{small}, and every $y \notin Y^{\text{large}}$

$$F(Y^{\text{small}} \cup y) - F(Y^{\text{small}}) \geq F(Y^{\text{large}} \cup y) - F(Y^{\text{large}}) \,. \tag{2.17}$$

A set function is *non-decreasing* if $F(Y^{\text{large}}) \geq F(Y^{\text{small}})$. The key result of Nemhauser *et al.* [1978] is that if a set function is submodular, non-decreasing, and $F(\emptyset) = 0$, then greedy, sequential maximization is guaranteed to produce a solution whose function value is within $(1 - 1/e) \approx 63\%$ of the global optimum.

Equation (2.16) is a set function from labels Y to information gain. Due to the 'information never hurts' bound [Cover *et al.*, 1994], F is non-decreasing, and clearly $F(\emptyset) = 0$. Unfortunately, as noted in Krause *et al.* [2008] information gain is not in general submodular. For example, consider a collection of three variables (y_1, y_2, z). We wish to gain information about z by observing y_1 or y_2; z may be thought of as the 'parameter of interest' $z \equiv \theta$. The input variable $\mathbf{x}$ is simply the index 1 or 2. Suppose (y_1, y_2, z) are jointly Gaussian distributed as

$$p(y_1, y_2, z) = \mathcal{N}\left([y_1, y_2, z]^\top; \mathbf{0}, \Sigma\right) \,, \quad \text{where } \Sigma = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 2 \end{bmatrix} \,. \tag{2.18}$$

The information gain in observing set Y is $\mathrm{IG}(Y) = \mathrm{H}[z] - \mathrm{H}[z|Y]$. y_2 and z are independent, therefore $\mathrm{IG}(y_2) - \mathrm{IG}(\emptyset) = 0$. Now suppose that we have observed y_1, the joint distribution of the remaining variables given y_1 is

$$p\left(\begin{bmatrix} y_2 \\ z \end{bmatrix} \middle| y_1\right) = \mathcal{N}\left(\frac{1}{2}\begin{bmatrix} y_1 \\ y_1 \end{bmatrix}, \begin{bmatrix} 1.5 & -0.5 \\ -0.5 & 1.5 \end{bmatrix}\right) \,.$$

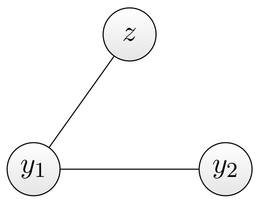

Figure 2.3: Graphical model for the joint Gaussian in Equation (2.18).

Now y_2 and z are conditionally dependent, y_2 provides information about z, $\mathrm{IG}(y_1 \cup y_2) - \mathrm{IG}(y_2) = \mathrm{H}[z|y_1] - \mathrm{H}[z|y_1, y_2] > 0$. Therefore, the increase in information gain when adding y_2 to y_1 is greater than when adding y_2 to $\emptyset$. Since $\emptyset \subset \mathbf{y}_1$, $\mathrm{IG}(Y)$ does not exhibit diminishing returns and is not submodular.

Fortunately, under certain conditional independence conditions, information gain will be submodular. The following theorem is given in Krause & Guestrin [2005].

Theorem 1. *Let S, U be disjoint subsets of (a finite set of random variables) V such that the variables in S are independent given U. Let 'information gain' be $IG(A) = H[U] - H[U \setminus A|A]$, where $A \subseteq W$, for any $W \subseteq S \cup U$. Then IG is submodular and non-decreasing on W, and $IG(\emptyset) = 0$.*

Theorem 1 seems a little daunting, but it has a simple intuition. We wish to learn about some variables U by observing some variables A chosen from a set W. W includes both the variables we are learning about U, or some extra variables S. Provided that these extra variables S are independent given the variables that we are learning U, then the information gain is a submodular set function. There is one further detail; if we are selecting A from U, then we only care about the information gain in the remaining variables in U, $\mathrm{IG}(A) = \mathrm{I}[A, U \setminus A]$.

Information theoretic active learning in the discriminative setting in Figure 2.1 is a special case of Theorem 1. Identifying U with the parameters θ, we never observe U directly, so $W = S$. Furthermore, since $S \cap U = \emptyset$ we have $U \setminus A = U$. S is the set of labels Y in Equation (2.16). The inputs X do not feature in the Theorem, they just index the set of labels. For $\mathrm{IG}(\theta)$ to be submodular and non-decreasing the variables in Y must be independent given θ. This is the case in Figure 2.1 as θ forms a Markov blanket between the variables $\mathbf{y}_i$.

So what is different in the joint Gaussian model in Equation (2.18)? The graphical model for Equation (2.18) is in Figure 2.3. Here, z does not form a Markov blanket between y_1 and y_2, so this figure cannot be mapped to the graphical model in Figure 2.1 and does not satisfy the conditional independence condition required by Theorem 1.

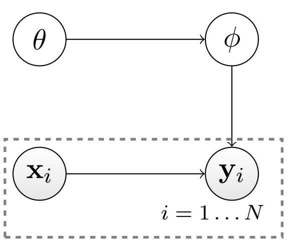

Figure 2.4: Graphical model for the discriminative learner with parameters θ and hyperparameters ϕ.

In summary, if the model can be mapped to the discriminative framework in Figure 2.1, then the information theoretic utility function in Equation (2.8) is submodular and so the myopic assumption only results in a small constant loss of utility.

However, for 'focused' information theoretic active learning this is not always the case. Consider a hierarchical Bayesian model in Figure 2.4. Suppose that the parameters of interest θ are the hyper-parameters and the 'lower level' parameters are considered to be nuisance parameters ϕ. For example, in Section 3.3 we want to learn the hyper-parameters of a GP kernel but do not care about the latent function. In this case BALD in Equation (2.15) integrates out the nuisance parameters. After integrating out ϕ, the outputs $\mathbf{y}_i$ are dependent, even when conditioned on θ. Therefore, with focused active learning, BALD may not be submodular. In practice we found that the myopic strategy performed well empirically, but investigating look-ahead techniques could produce further improvements here.

2.4 Literature Review

Active learning has been widely studied in machine learning, statistics and experimental and social sciences, where it is also referred to as *query learning* or *optimal experimental design* (OED). We outline some of the key related algorithms. It is not possible to cover the entire field, for comprehensive coverages see the textbooks Fedorov [1972] and Settles [2012].

2.4.1 Classical Optimal Experimental Design

Classical OED, developed in statistics, focuses primarily on linear regression,

$$y = \theta^\top \mathbf{x} + \epsilon \,. \tag{2.19}$$

The noise is usually Gaussian distributed with zero mean, $\epsilon \sim \mathcal{N}(0, \sigma^2)$. In this case the maximum likelihood estimator (MLE) of θ is equivalent to the least squares estimator. The MLE is $\hat{\theta} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}$, where $\mathbf{X}$ is the $N \times D$ 'design matrix' consisting of N input vectors $\mathbf{x}$, and $\mathbf{Y}$ is the vector containing N real valued outputs y. The variance of the MLE is given by $\sigma^2(\mathbf{X}^\top\mathbf{X})^{-1}$, the inverse of the variance is the Fisher Information (FI) matrix for this model. The FI provides an alternative to Shannon's entropy for measuring the information in a sample $\mathbf{x}$ about the unknown parameters θ. With linear-Gaussian models classical OEDs maximize the FI, or equivalently minimize the variance of the MLE, with respect to the next datapoint to be added to the design matrix. However, because the FI is a matrix quantity, not a scalar like the posterior entropy, a number of different utility functions have been proposed. These have alphabetized names, such as A-, D-, or E- optimality. We discuss a few of the most famous amongst these, others can be found in Atkinson [1988].

D-optimal design maximizes the determinant of $\mathbf{X}^\top\mathbf{X}$. This is equivalent to the Bayesian information gain criterion used by BALD (2.8) if a flat (non-informative) prior on θ is used. For example, if we have a zero mean Gaussian prior with covariance Σ, then the posterior $p(\theta|\mathbf{X}, \mathbf{Y})$ is Gaussian with covariance $\sigma^2(\mathbf{X}^\top\mathbf{X}+\Sigma^{-1})^{-1}$. The entropy of a Gaussian is proportional to the log determinant of its covariance matrix, and so with a non-informative prior, such as $\Sigma = \lim_{\epsilon\to 0}\frac{1}{\epsilon}I$, the determinant of the design matrix is a monotonic function of the posterior entropy. Bayesian D-optimality extends D-optimality to include a prior. In this case $|(\mathbf{X}^\top\mathbf{X} + \Sigma^{-1})^{-1}|$ is minimized, which is equivalent to posterior entropy minimization with the linear-Gaussian model (2.19). Bayesian equivalents of most classical OEDs are formed by replacing the covariance of the MLE $(\mathbf{X}^\top\mathbf{X})^{-1}$ with the posterior covariance matrix $(\mathbf{X}^\top\mathbf{X}+\Sigma^{-1})^{-1}$ [Chaloner & Verdinelli, 1995].

D-optimal design can be motivated using other utility functions on θ. Posterior entropy minimization is equivalent to minimizing expected log loss on the parameters. If the true parameter is $\hat{\theta}$ we can define a 'score' for the posterior distribution as $S(\hat{\theta}, p(\theta|\mathcal{D})) = \log p(\hat{\theta}|\mathcal{D})$. This score rewards posteriors with high density at the true parameter value. The expected score given the current posterior beliefs is

$$E_{p(\theta|\mathcal{D})}[S(\theta, p(\theta|\mathcal{D}))] = -\mathrm{H}[p(\theta|\mathcal{D})]\,.$$

Maximizing the expected score with respect to the data $\mathcal{D}$ is therefore equivalent to posterior entropy minimization. $S(\theta, p(\theta|\mathcal{D}))$ is an example of a proper scoring rule. These are scores that reward honest estimates of uncertainty. Proper scoring rules are beyond the scope of this thesis, see Dawid [2007]; Huszár [2013] for details. Other scoring rules may be used, and with linear-Gaussian models, D-optimality is also equivalent to maximizing the Brier score [Brier, 1950]. However, unlike Shannon's information, which is induced by the log score, other scoring rules do not result in symmetric utility functions. Hence, computationally efficient rearrangements like BALD in Section 2.3.1 are not generally possible.

A-optimality minimizes the trace of $(\mathbf{X}^\top\mathbf{X})^{-1}$. The Bayesian equivalent, minimizing the trace of the posterior covariance, is motivated if a point estimate of the parameters $\hat{\theta}$ is sought. This is because A-optimality is equivalent to minimizing the expected squared Euclidean distance of an estimator $\hat{\theta}$ to the true parameters, $\mathcal{U}(\mathbf{x}) = -\mathbb{E}_{p(\mathbf{y},\theta|\mathbf{x},\mathcal{D})}||\theta - \hat{\theta}||_2^2$.[1] The trace operation minimizes the average variance of the estimator over the dimensions of θ. If this average is replaced with a minimax criterion, then the objective is called E-optimality. E-optimality minimizes the maximum posterior variance for any linear projection of θ, $\mathrm{argmax}_{\mathbf{c}:||\mathbf{c}||=\mathbf{1}}\, \mathbf{c}^\top(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{c}$.

E-optimality is mathematically similar to G-optimality which minimizes $\mathrm{argmax}_{\mathbf{x}^\star}\, \mathbf{x}^{\star\top}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{x}^\star$, where $\mathbf{x}^\star$ is any point in input space. The Bayesian equivalent has a decision-theoretic flavour: to minimize the greatest predictive variance at any point in input space. This arises because the variance of the Bayesian predictive distribution (1.2) for the linear-Gaussian model is $\mathrm{Var}\left[p(\mathbf{y}^\star|\mathbf{x}^\star,\mathcal{D})\right] = \sigma^2\mathbf{x}^{\star\top}(\mathbf{X}^\top\mathbf{X} + \Sigma^{-1})^{-1}\mathbf{x}^\star$.

These designs have been extended to nonlinear regression models, $y = f(\theta^\top\mathbf{x}) + \eta$, where f is a nonlinear function. In the linear-Gaussian model (2.19) most designs, such as D-optimality, are independent of the output $\mathbf{y}$. This is also true for other models such as the generalized linear model with Poisson likelihood and exponential link function [Lewi *et al.*, 2009]. With linear regression the design is also independent of the parameters. In these cases, the optimal sequence of measurements can be computed *a priori*. However, this is not the case with nonlinearities. Similar to active learning, classical designs often take an iterative approach. They estimate the parameter $\hat{\theta}$ and then select the next measurement using this estimate [Chaloner & Verdinelli, 1995]. In Bayesian OED, the posterior usually becomes intractable, so a number of approximations have been proposed. The most popular methods approximate the posterior with a Gaussian, whose mean is equal to the MLE or MAP estimate $\hat{\theta}$. The variance of the approximation is computed using the inverse of the FI matrix. In general, the FI matrix is the Hessian of the negative log likelihood surface, and is a function

[1] Note that an expectation of the unseen $\mathbf{y}$ is needed here because, although the posterior variance is not a function of $\mathbf{y}$, the estimator $\hat{\theta}$ may be.

	Cost Function	Bayesian Interpretation/Equivalence
D	$\lvert(\mathbf{X}^\top\mathbf{X})^{-1}\rvert$	max Shannon information gain in θ
A	$\mathrm{tr}[(\mathbf{X}^\top\mathbf{X})^{-1}]$	min variance of a point estimate for θ
E	$\mathrm{argmax}_{\mathbf{c}:\lvert\lvert\mathbf{c}\rvert\rvert=\mathbf{1}}\, \mathbf{c}^\top(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{c}$	min max post. variance for any projection of θ
G	$\mathrm{argmax}_{\mathbf{x}^\star}\, {\mathbf{x}^\star}^\top(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{x}^\star$	min max predictive variance at locations $\mathbf{x}^\star$

Table 2.1: A few classic optimal experimental designs. Cost is 'negative utility'. Bayesian equivalents are formed by replacing $\mathbf{X}^\top\mathbf{X}$ by $\mathbf{X}^\top\mathbf{X} + \Sigma^{-1}$, where Σ is the prior covariance of θ. More general equivalents for nonlinear regression models are given by replacing $\mathbf{X}^\top\mathbf{X}$ with the general form of the Fisher Information Matrix, $F(\hat{\theta})$.

of θ and $\mathbf{y}$, unlike for the linear model where it is simply equal to $\mathbf{X}^\top\mathbf{X}$. This is the Laplace approximation, widely used in machine learning and statistics [Kass & Raftery, 1995]. The classical design criteria can then be estimated for nonlinear designs using the approximate distribution over θ.

Table 2.1 summarizes the classical designs presented in this Section.

2.4.2 Information Theoretic Methods

Maximizing Shannon's information for Bayesian experimental design was originally proposed in Lindley [1956]. This paper studies the fundamental properties of this objective. These properties are studied further in Fedorov [1972]; MacKay [1992b]. In the latter, the posterior information gain for an arbitrary parametric model with a Gaussian likelihood is approximated using the Laplace approximation. As described in Section 2.3.2, performing active learning by computing parameter entropies can be expensive, even with a Laplace approximation. More recently, a fast algorithm that minimizes posterior entropies directly is proposed in Lewi *et al.* [2007].

Algorithms that work with predictive distributions tend to be more easily applicable, but do not necessarily optimize an objective, such as information gain. The most well known of these algorithms is Maximum Entropy (or uncertainty) Sampling (MES) [Shewry & Wynn, 1987]. MES selects datapoints with the largest predictive entropy. This corresponds to the first term in BALD, Equation (2.10). MES was originally proposed for regression models with constant observation noise, such as the linear-Gaussian model. Here, the second term in Equation (2.10) is constant with respect to $\mathbf{x}$, so it can be ignored. However, MES is not equivalent to BALD with more complex models, such as classification or heteroscedastic models. This is because it fails to differentiate between parameter uncertainty and observation noise. As a result

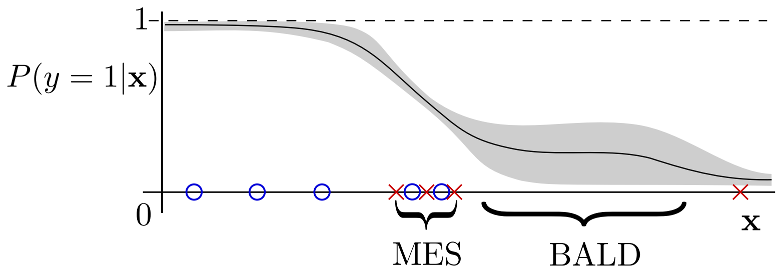

Figure 2.5: Toy example of classification with a 1D input. Circles and crosses denote labelled datapoints. The plot shows the mean and variance of the predictive distribution. Maximum Entropy Sampling (MES) selects data from regions of high marginal uncertainty. BALD seeks more informative data in the region of uncertainty due to high posterior variance.

MES over-samples in regions of noisy data. For example, points near a classification decision boundary may be very noisy, and hence provide no information about the model. BALD, on the other hand, directly maximizes the information gain so does not exhibit such pathologies. BALD will learn that a region has high inherent uncertainty thus will explore elsewhere. Figure 2.5 illustrates this point on a 1D classification example.

The mutual information between measured variables and 'variables of interest' has been proposed in various applications. These include tracking via Bayesian filtering [Ertin *et al.*, 2003]. Here, the hidden state is the variable of interest, and with a linear Gaussian model, the utility function reduces to MES. Fuhrmann [2003] applies mutual information in noisy communication channels with binary variables, in this case entropy computations are simple. With BALD, the interest variables are model parameters, the above approaches do not learn these.

Another application of mutual information is for monitoring environmental variables. Here, the mutual information between measured locations and interest locations is maximized [Caselton & Zidek, 1984]. This approach is transductive (see Section 2.3.5), it learns the function optimally just at the interest locations. This technique is applied with GP regression in Krause *et al.* [2008], where a grid of interest locations is specified. However, gridding the region of interest is impractical for higher dimensional problems.

The above mutual information based methods do not consider hyperparameter learning which we will address for GP regression in Chapter 3. Hyperparameter uncertainty has been considered in a similar setting to GPs, Empirical Kriging. Zimmerman

[2006] minimize the predictive variance at interest locations for active sensor placement with unknown kernel hyperparameters. A linearized approximation of the effect of hyperparameter uncertainty (measured using Fisher Information) on the predictive variance is added to the utility function. Focused BALD (2.15) minimizes hyperparameter uncertainty more directly.

2.4.3 Data Subsampling

Active learning is primarily used for collecting data when labelling is expensive. However, if the dataset is very large, or there is a continuous stream of data, and one does not have the resources to process all of the points then active learning can be used to subsample the dataset to a feasible size. This simple approach to 'compressing' a large dataset is referred to as the *subset of data* approximation [Quiñonero-Candela & Rasmussen, 2005]. Subset of data methods differ to active learning because the label $\mathbf{y}$ can be observed prior to sampling, therefore, although active learning techniques can address the subset of data problem, the reverse is not possible.

A popular dataset subsampling algorithm for GPs is the Informative Vector Machine (IVM) [Herbrich *et al.*, 2002].[1] Like BALD, the IVM maximizes information gain in the parameters. However, $\mathbf{y}$ is observed so the IVM does not work explicitly with predictive distributions, but works directly in parameter space. However, GPs have an infinite dimensional parameter, the latent function. The IVM computes parameter entropies in the marginal subspace that corresponds to the observed datapoints. Now, the entropy decrease after inclusion of a new point is calculated efficiently using the GP covariance matrix. Figure 2.6 shows the difference between this approach and BALD. As a result, the IVM has a transductive bias, the locations of the observed data defines the parameters whose entropy are minimized. BALD is inductive because it works implicitly with the full infinite-dimensional latent function.

Although the IVM and BALD are motivated by the same objective, they have different properties when approximate inference is carried out, such as in Gaussian process classification (GPC). At time t both methods have an approximate posterior $q_t(\theta|\mathcal{D})$. This can be updated with the likelihood of a new data point $p(y_{t+1}|f, \mathbf{x}_{t+1})$, yielding $\hat{p}_{t+1}(\theta|\mathcal{D}, \mathbf{x}_{t+1}, y_{t+1}) = \frac{1}{Z}q_t(\theta|\mathcal{D})p(y_{t+1}|f, \mathbf{x}_{t+1})$. If the posterior at $t+1$ is approximated directly one gets $q_{t+1}(\theta|\mathcal{D}, \mathbf{x}_{t+1}, y_{t+1})$. BALD calculates the entropy difference between q_t and $\hat{p}_{t+1}$, without having to compute q_{t+1} for each candidate $\mathbf{x}_{t+1}$ and label y_{t+1}. In contrast, the IVM calculates the entropy change between q_t

[1] GPs are introduced in Section 3.1.

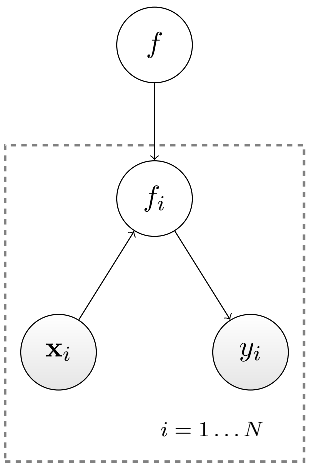

Figure 2.6: Graphical model for Gaussian processes. The infinite dimensional latent parameter f gives rise to scalar latent variables f_i at each input location, $\mathbf{x}_i$. The directed links from the top layer to the 2nd layer are deterministic marginalizations, not probabilistic dependencies. Given the latent function f_i the likelihood function generates the labels y_i. The IVM calculates entropies in the second layer, whereas BALD calculates entropy changes to the entire f at the top.

and q_{t+1}, and so must compute the new posterior for all possible queries. Methods that compute posterior entropies directly require $\mathcal{O}(N|\mathcal{Y}|c^{\text{inf}} + N|\mathcal{Y}|h^{\theta}))$ computations per sample since the posterior must be recomputed for all candidates.[1] Therefore, because c^{inf} enters the complexity per candidate, the IVM for GPC must perform fast approximate inference using assumed density filtering (ADF). BALD requires only one posterior update, so costs $\mathcal{O}(c^{\text{inf}} + Nc^{\text{pred}} + Nh^{\mathbf{y}})$. Therefore, more accurate iterative procedures, such as expectation propagation (EP) [Minka, 2001b], can be used. ADF is a single-pass version of EP, so this results in q_{t+1} being a direct approximation to $\hat{p}_{t+1}$ which BALD implicitly works with.

2.4.4 Decision Theoretic Methods

Decision theoretic methods directly minimize the expected loss, measured using the Bayes posterior risk (2.1). In many tasks, the loss is hard to quantify, but in classification, the misclassification rate provides a sensible loss function [Kapoor *et al.*, 2007; Roy & McCallum, 2001; Zhu *et al.*, 2003]. These methods assume knowledge of the location of the test data and then minimize the expected 0/1 classification loss. However, as

[1] For homoscedastic regression with fixed hyperparameters the posterior variance is independent of $\mathbf{y}$, so only $\mathcal{O}(Nc^{\text{inf}} + Nh^{\theta})$ updates are needed. With further approximations to the likelihood the IVM can be sped up further for regression [Seeger *et al.*, 2003].

Bayesian	**Non-Probabilistic**
parameter setting	hypothesis/hyperplane
posterior distribution	version space (VS)
posterior entropy	log volume of VS
Gaussian approx to posterior e.g. EP, Laplace	hypersphere approx to VS e.g. Tong & Koller [2002]
samples from posterior	committee members
MES	margin sampling
Equation (2.8)	directly minimizing VS volume
BALD	QBC with an infinite committee and probabilistic voting

Table 2.2: Analogies between Bayesian active learning and non-probabilistic active learning methods for SVMs.

noted in Roy & McCallum [2001], a limitation of decision theoretic methods is the computational cost. In general, with T test points they require $\mathcal{O}(N|\mathcal{Y}|c^{\text{inf}} + TN|\mathcal{Y}|c^{\text{pred}})$ computations to calculate the posterior risk on the test set for all possible new datapoints. Incremental re-training and re-classification can speed up computations with certain models [Roy & McCallum, 2001]. Further computational savings require approximations such as pruning the query search space or sub-sampling the test set.

2.4.5 Non-Probabilistic Methods

Non-probabilistic methods have analogies with Bayesian active learning. The most well known non-probabilistic active learning methods are for Support Vector Machines (SVMs) [Seung *et al.*, 1992; Tong & Koller, 2002]. SVMs are a sparse non-probabilistic model mostly used for binary classification. SVMs classify the data using a hyperplane with maximal separation from each class.[1] The set of possible hyperplanes that correctly classifies the training data is called version space (VS). An introduction to SVMs can be found in Burges [1998].

A popular approach to SVM active learning minimizes the number of possible hypotheses (hyperplanes), which is done by minimizing the volume of VS. VS can be interpreted as a deterministic equivalent to the posterior distribution, and its volume is analogous to the posterior entropy. If a uniform (improper) prior and a deterministic likelihood are used then the log volume of VS is equivalent to the entropy of the posterior. Each observed datapoint defines a plane in VS and hypotheses consistent

[1] SVMs are not limited to linear decision planes, they can use kernels to produce complex decision boundaries [Schölkopf & Smola, 2002].

with the data lie on one side of this plane. The goal of active learning with SVMs is to sample a datapoint that cuts VS in half. Thus, after receiving the label, at worst almost half of the hypotheses are eliminated.

However, just as Bayesian posterior distributions can be intractable, VS can become complex after observing many datapoints, and the relevant volumes hard to compute. Therefore, Tong & Koller [2002] propose approximating a complex VS with simpler shapes, such as hyperspheres. Their simplest approximation fits the largest possible hypersphere into VS and selects the point whose dual hyperplane falls closest to the centre of this sphere. This is equivalent to margin sampling, which chooses the point closest to the decision boundary [Campbell *et al.*, 2000]. Similar to MES, margin sampling can over-sample noisy data beside the boundary. This approximation of VS using a simple shape is similar to Bayesian inference methods that approximate the posterior with a simple distribution, such as a Gaussian.

Query by Committee (QBC) sidesteps complex version spaces and, like BALD, works directly with predictions [Seung *et al.*, 1992]. QBC samples parameters from VS 'committee members', each of whom makes a prediction, 'votes', on the label of $\mathbf{x}$. The $\mathbf{x}$ with the most balanced vote is selected, this is termed the 'principle of maximum disagreement'. If Equation (2.10) is approximated by sampling θ, then BALD resembles QBC, but with a probabilistic measure of disagreement. By discarding confidence estimates QBC can exhibit the same pathologies as MES, namely over-sampling noisy data. QBC was proposed for classification, and to extend this algorithm directly to other scenarios, such as regression, the disagreement measure needs to be re-designed [Burbidge *et al.*, 2007]. McCallum & Nigam [1998]; Melville *et al.* [2005] replace the deterministic voting in QBC with the Jensen-Shannon (JS) divergence between a finite number of predictive distributions. These works do not uncover the link to information gain in the parameters; BALD also minimizes the JS-divergence of predictive distributions, but from infinitely many committee members drawn from the posterior. Table 2.2 summarizes the analogies between Bayesian and non-probabilistic methods for active learning.

2.4.6 Summary

We have presented a framework for Bayesian information theoretic active learning called BALD. This framework directly exploits the rearrangement of parameter entropies to predictive entropies. There are many approaches to active learning, but BALD can be advantageous for a number of reasons:

- It is inductive, so does not make any assumptions about a future decision task, loss or test set.
- With many models, the utility function is smooth and so BALD may be applied to both continuous sampling and pool-based active learning.
- The utility function is often, but not always, submodular and hence greedy maximization is near-optimal.

The above advantages are specific to the classical information gain objective function in Equation (2.8). The potential advantages of the rearrangement that BALD exploits (2.10) and the extension in Equation (2.15) are:

- The required computations are not inherently tied to a particular model or inference method.
- If output space is 'simpler' than parameter space, as is often the case, then the required entropies are more straightforward to compute.
- The number of (approximate) posterior updates is reduced from one per *possible* datapoint to one per *observed* datapoint.
- One can focus upon learning particular variables in the model.

However, computation of the utility in Equation (2.10) may still be non-trivial. Whether BALD is computationally useful depends on the particular task and model. Whether BALD is practically useful is a matter of empirical performance. In the following chapters we apply this framework in machine learning and scientific domains to yield efficient algorithms with strong practical performances.

Chapter 3

Active Gaussian Processes

Gaussian processes (GPs) are a powerful, Bayesian non-parametric model for classification and regression. They have been extended to a number of other domains such as optimization [Osborne *et al.*, 2009], quadrature [Ghahramani & Rasmussen, 2002], dimensionality reduction [Lawrence, 2004] and preference learning [Chu & Ghahramani, 2005b]. Using information-theoretic active learning with GPs appears to be challenging because their parameter space is infinite dimensional. However, with BALD (Section 2.3) we can calculate posterior information gains accurately without having to compute entropies of infinite dimensional objects.

In this chapter we first provide a brief introduction to GPs, for full details see Rasmussen & Williams [2005]. In Section 3.2 we demonstrate how BALD may be applied to Gaussian process classification (GPC). In vanilla Gaussian process regression (GPR) the observation noise is constant over the input domain, however, this is not true in GPC which makes active learning more difficult. Other active learning algorithms that work with predictions, such as maximum entropy sampling (MES), confound posterior uncertainty with inherent noise. BALD provides a principled and intuitive balance between these sources of uncertainty in GPC. Furthermore, unlike in GPR, inference in GPC is intractable and so GPC requires more expensive inference routines. Therefore, the reduction in the number of posterior updates for N candidates and l possible labels from $\mathcal{O}(Nl)$ to $\mathcal{O}(1)$ when using BALD (see Section 2.3.2) is important with GPC.

Like most models, Gaussian processes have additional parameters, known as hyperparameters. Obtaining good performance with GPs requires appropriate hyperparameter management. Typically, these are optimized using type-II maximum likelihood Rasmussen & Williams [2005], but this method can perform poorly and integration over the hyperparameters yields better predictions [Garnett *et al.*, 2010]. This problem is particularly important in active learning which usually works in the low-data regime (labels are expensive). Hence, ignoring uncertainty estimates may cause extreme overfitting of the GP. Particularly, in GPR maximizing information gain has been found to yield poor performance after the first couple of samples when the hyperparameters are fixed [Seeger *et al.*, 2003; Seo *et al.*, 2000]. In Section 3.3 we address this problem by combining BALD for 'focused' learning of particular variables of interest with a new algorithm for approximate hyperparameter marginalization, the Marginal GP [Garnett *et al.*, 2013]. Using these techniques we provide a complete pipeline for active GPR with unknown hyperparameters.

3.1 Primer on Gaussian Processes

Informally, Gaussian processes provide a distribution over a broad class of functions. The probabilistic model underlying GPR and GPC is

$$\text{prior:} \quad p(f) = \mathrm{GP}(\mu(\cdot), k(\cdot,\cdot))\,, \tag{3.1}$$

$$\text{regression likelihood:} \quad p(y|\mathbf{x}, f) = \mathcal{N}(y; f(\mathbf{x}), \sigma^2)\,, \tag{3.2}$$

$$\text{classification likelihood:} \quad p(y|\mathbf{x}, f) = \mathrm{Bernoulli}(\Phi(f(\mathbf{x})))\,. \tag{3.3}$$

The latent parameter for this model, f, is a function $\mathcal{X} \to \mathbb{R}$. A Gaussian process prior on this function is fully specified by a mean function $\mu(\mathbf{x}) : \mathcal{X} \mapsto \mathbb{R}$ and covariance function or kernel $k(\mathbf{x}, \mathbf{x}') : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$. Under the GP prior, the marginal of f evaluated at any finite set of points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ follows a multivariate Gaussian distribution with mean $\mathbf{m}$, whose components are $m_i = \mu(\mathbf{x}_i)$, and covariance matrix Σ, where $\Sigma_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

For regression, the output variable y is modelled directly using f plus additive Gaussian noise. For classification we consider the probit likelihood. Here, given the value of f, y takes a Bernoulli distribution with parameter $\Phi(f(\mathbf{x}))$, and $\Phi(\cdot)$ is the standard Gaussian c.d.f. (probit function). As an alternative one can use a logistic likelihood, but in practice there is little difference in performance [Rasmussen & Williams, 2005].

In GPR, the Gaussian process prior (3.1) is *conjugate* to the Gaussian likelihood (3.2), so inference is tractable. Exponential family likelihoods, such as the Gaussian, have conjugate priors. These priors yield tractable posterior distributions that are in the same family as the prior, see Bishop [2006], Chapter 2, for details. However, the GP prior is not conjugate to the classification likelihoods. Therefore exact inference is intractable; given some observations $\mathcal{D}$, the posterior over f is non-Gaussian. There are a number of approximate inference methods, the most common of which – expectation propagation (EP) [Minka, 2001a], the Laplace approximation [Kass & Raftery, 1995], assumed density filtering [Ito & Xiong, 2000] and sparse methods [Naish-Guzman & Holden, 2007] – all approximate the posterior by a Gaussian. Throughout we will assume that such a Gaussian approximation is provided, though the active learning algorithm does not care which. We will denote the use of such approximate inference by $\overset{1}{\approx}$.

After performing inference, given a Gaussian (exact or approximate) posterior, the predictive distribution at a new point $\mathbf{x}^\star$ is computed by integrating over the latent function,

$$p(y|\mathbf{x}^\star, \mathcal{D}) = \int p(y|f_{\mathbf{x}^\star})p(f_{\mathbf{x}^\star}|\mathcal{D})df_{\mathbf{x}^\star} \,, \tag{3.4}$$

where $f_{\mathbf{x}} \triangleq f(\mathbf{x})$. With both the regression and classification likelihoods given in Equations (3.2) and (3.3) respectively, this computation can be performed analytically. We now show how BALD may be used for active GPC.

3.2 Active GP Classification

In classification, the level of observation noise is input-dependent. For example, data near a decision boundary may be highly noisy. Therefore, standard MES can perform poorly, which we confirm empirically in Section 3.2.1. Furthermore, the posterior $p(f|\mathcal{D})$ is intractable in GPC and approximate inference can be expensive. BALD allows us to compute the information gain given by Equation (2.8) directly, but with the same low computational cost as MES.

The BALD utility in the context of GPs is

$$\mathcal{U}(\mathbf{x}) = \mathrm{H}[p(y|\mathbf{x}, \mathcal{D})] - \mathbb{E}_{p(f|\mathcal{D})}\mathrm{H}[p(y|\mathbf{x}, f)] \,, \tag{3.5}$$

where for classification $p(y|f_{\mathbf{x}})$ is the probit likelihood function, Equation (3.3). The entropy of the binary output variable y given a fixed function f is given by the binary entropy function $\mathrm{h}(p)$,

$$\begin{aligned} \mathrm{H}[p(y|\mathbf{x}, f)] &= \mathrm{h}\left(\Phi(f_{\mathbf{x}})\right) , \\ \text{where } \mathrm{h}(p) &= -p \log p - (1-p)\log(1-p) \,. \end{aligned}$$

Using a Gaussian approximation to the posterior, for each input $\mathbf{x}$, $f_{\mathbf{x}}$ follows a Gaussian distribution with mean $\mu_{\mathbf{x}|\mathcal{D}}$ and variance $\sigma^2_{\mathbf{x}|\mathcal{D}}$. To calculate $\mathcal{U}(\mathbf{x})$ we have to compute two entropies. With a probit likelihood the first term in Equation (3.5), $\mathrm{H}[p(y|\mathbf{x},\mathcal{D})]$, can be handled analytically,

$$\begin{aligned}\mathrm{H}[p(y|\mathbf{x},\mathcal{D})] &\overset{1}{\approx} \mathrm{h}\left(\int \Phi(f_{\mathbf{x}})\mathcal{N}(f_{\mathbf{x}};\mu_{\mathbf{x}|\mathcal{D}},\sigma^2_{\mathbf{x}|\mathcal{D}})df_{\mathbf{x}}\right)\\ &= \mathrm{h}\left(\Phi\left(\frac{\mu_{\mathbf{x}|\mathcal{D}}}{\sqrt{\sigma^2_{\mathbf{x}|\mathcal{D}}+1}}\right)\right). \end{aligned} \tag{3.6}$$

The second term, $\mathbb{E}_{p(f|\mathcal{D})}\mathrm{H}[p(y|\mathbf{x},f)]$, requires a second, minor approximation,

$$\begin{aligned}\mathbb{E}_{p(f|\mathcal{D})}\mathrm{H}[p(y|\mathbf{x},f)] &\overset{1}{\approx} \int \mathrm{h}(\Phi(f_{\mathbf{x}}))\mathcal{N}(f_{\mathbf{x}};\mu_{\mathbf{x}|\mathcal{D}},\sigma^2_{\mathbf{x}|\mathcal{D}})df_{\mathbf{x}} \qquad (3.7)\\ &\overset{2}{\approx} \int \exp\left(-\frac{f_{\mathbf{x}}^2}{\pi\ln 2}\right)\mathcal{N}(f_{\mathbf{x}};\mu_{\mathbf{x}|\mathcal{D}},\sigma^2_{\mathbf{x}|\mathcal{D}})df_{\mathbf{x}}\\ &= \frac{C}{\sqrt{\sigma^2_{\mathbf{x}|\mathcal{D}}+C^2}}\exp\left(-\frac{\mu^2_{\mathbf{x}|\mathcal{D}}}{2\left(\sigma^2_{\mathbf{x}|\mathcal{D}}+C^2\right)}\right),\end{aligned}$$

where $C=\sqrt{\frac{\pi\ln 2}{2}}$. The first approximation, $\overset{1}{\approx}$, denotes any standard Gaussian approximation to the posterior. The integral on the left hand side of Equation (3.7) is intractable. We tackle this using a Taylor expansion on $\ln\mathrm{h}(\Phi(f_{\mathbf{x}}))$,

$$\begin{aligned} g(x) &= g(0)+\frac{g'(0)x}{1!}+\frac{g''(0)x^2}{2!}+\dots\\ g(x) &= \ln\mathrm{h}(\Phi(x))\\ g'(x) &= -\frac{1}{\ln 2}\frac{\Phi'(x)}{\mathrm{h}(\Phi(x))}\left[\ln\Phi(x)-\ln(1-\Phi(x))\right]\\ g''(x) &= \frac{1}{\ln 2}\frac{\Phi'(x)^2}{\mathrm{h}(\Phi(x))^2}\left[\ln\Phi(x)-\ln(1-\Phi(x))\right]\\ &\quad-\frac{1}{\ln 2}\frac{\Phi''(x)}{\mathrm{h}(\Phi(x))}\left[\ln\Phi(x)-\ln(1-\Phi(x))\right]\\ &\quad-\frac{1}{\ln 2}\frac{\Phi'(x)^2}{\mathrm{h}(\Phi(x))}\left[\frac{1}{\Phi(x)}+\frac{1}{1-\Phi(x)}\right],\\ \therefore \ln\mathrm{h}(\Phi(x)) &= 1-\frac{1}{\pi\ln 2}x^2+\mathcal{O}(x^4).\end{aligned}$$

Thus, $\ln\mathrm{h}(\Phi(f_{\mathbf{x}}))$ can be approximated up to $\mathcal{O}(f_{\mathbf{x}}^4)$ by an unnormalized Gaussian,

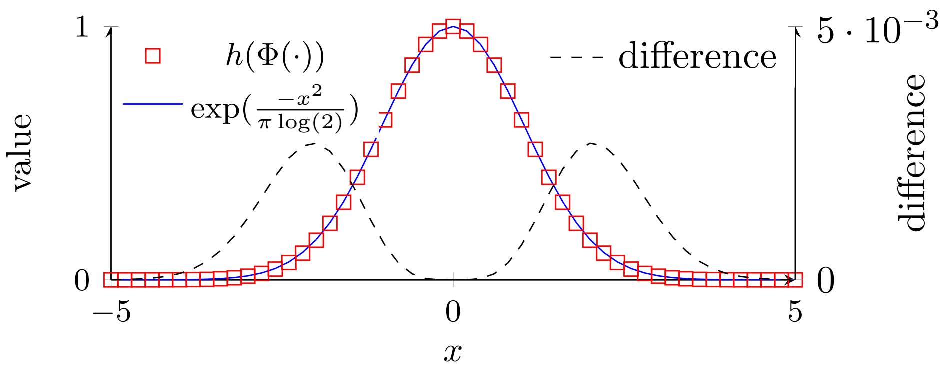

Figure 3.1: Analytic approximation ($\overset{1}{\approx}$) to the binary entropy of the error function (□) by a squared exponential (——). The absolute error, whose scale is given by the right hand y-axis (- - -) remains smaller than $3 \cdot 10^{-3}$.

$\exp(-f_{\mathbf{x}}^2/\pi \ln 2)$. We denote this approximation with $\overset{2}{\approx}$. We now apply the standard convolution formula for Gaussians to get a closed form expression for both terms in Equation (3.5).

Figure 3.1 depicts the striking accuracy of this simple approximation. The maximum error occurs when $\mathcal{N}(f_{\mathbf{x}}; \mu_{\mathbf{x}|\mathcal{D}}, \sigma^2_{\mathbf{x}|\mathcal{D}}) = \delta(f_{\mathbf{x}} - 2.05)$. In this worst case $\overset{2}{\approx}$ yields only a 0.27% error to the integral in Equation (3.7). In Section 3.2.1 we confirm empirically that this approximation is negligible relative to standard approximate inference $\overset{1}{\approx}$.

To summarize, the BALD algorithm for GPC consists of two steps. First it applies any standard approximate inference algorithm for GPC (such as EP) to obtain the posterior predictive mean $\mu_{\mathbf{x}|\mathcal{D}}$ and $\sigma_{\mathbf{x}|\mathcal{D}}$ for each point of interest $\mathbf{x}$. Then it selects the query $\mathbf{x}$ that maximizes the utility function

$$\mathcal{U}(\mathbf{x}) = \mathrm{h}\left(\Phi\left(\frac{\mu_{\mathbf{x}|\mathcal{D}}}{\sqrt{\sigma^2_{\mathbf{x}|\mathcal{D}} + 1}}\right)\right) - \frac{C \exp\left(-\frac{\mu^2_{\mathbf{x}|\mathcal{D}}}{2\left(\sigma^2_{\mathbf{x}|\mathcal{D}} + C^2\right)}\right)}{\sqrt{\sigma^2_{\mathbf{x}|\mathcal{D}} + C^2}}. \tag{3.8}$$

For most kernels, the objective (3.8) is a smooth and differentiable function of $\mathbf{x}$. Therefore gradient-based optimization procedures can be used to find the maximally informative query. The resulting optimization surface may be high dimensional and multimodal, and having access to gradients may be crucial to make optimization feasible. Many non-probabilistic approaches, such as those described in Section 2.4.5, have discontinuous utility surfaces and will be much harder to optimize. In our experiments we only have access to fixed set of inputs $\mathbf{x}$. Therefore, we perform pool-based learning, evaluating Equation (3.8) on the entire pool.

Equation (3.8) gives us insight into the data that is sought. If the latent function has zero mean, $\mu_{\mathbf{x}|\mathcal{D}} = 0$, then we are uncertain about the label, and the first term gives us a maximal $+1$ bit of information. However, if the posterior variance $\sigma^2_{\mathbf{x}|\mathcal{D}}$ is also small, then the GP knows that the label for $\mathbf{x}$ is noisy. Hence the second term will reduce $\mathcal{U}(\mathbf{x})$. Figure 3.2 demonstrates the behaviour of Equation (3.8) on toy 2D GPC problem. In this example, on either side of the points in class 'one' ($+$) are those from a bimodally distributed class 'two' ($\times$). The training data is plotted on each panel in Figure 3.2. Panel (a) shows the true posterior probability contours of the two classes. There is large separation between class one and the mode of class two in the top right corner, but not in the bottom left, where there is a region around $(-0.5, -0.5)$ of high observation noise. Contours in panel (b) are the posterior mean prediction, the green regions indicate areas of high predictive uncertainty. Panels (c) and (d) give contours of utility functions for BALD (3.8) and MES, the first term in Equation (3.8). MES rewards sampling in any region of uncertainty. BALD behaves differently, rewarding regions that have both a posterior predictive mean close to 0.5 and a high posterior uncertainty.

3.2.1 Experiments

We compare to a number of popular active sampling algorithms for classification, described in Section 2.4, including decision theoretic algorithms that are privy to information about the test data. We use a number of challenging hand-constructed and real-world datasets.

Datasets

We used four artificial, and nine real-world datasets. *GP-D5* is generated from the assumed GPC model with a 5D input. *Checkerboard*, *Noisy Block* and *Distracting Block* are three challenging hand-crafted 2D datasets, inspired by Zhu *et al.* [2003]. These datasets are depicted in Figure 3.3, top. *Checkerboard* consists of 16 clusters of datapoints arranged in a checkerboard pattern. *Noisy Block* contains a block of noisy data on the decision boundary in which the class labels are random and *Distracting Block* contains a block of uninformative data far from the decision boundary. On *Checkerboard* a strong active learning algorithm should sample one point from each island of data. On *Distracting* and *Noisy Block* a good active learning algorithm should avoid the non-informative blocks of data.

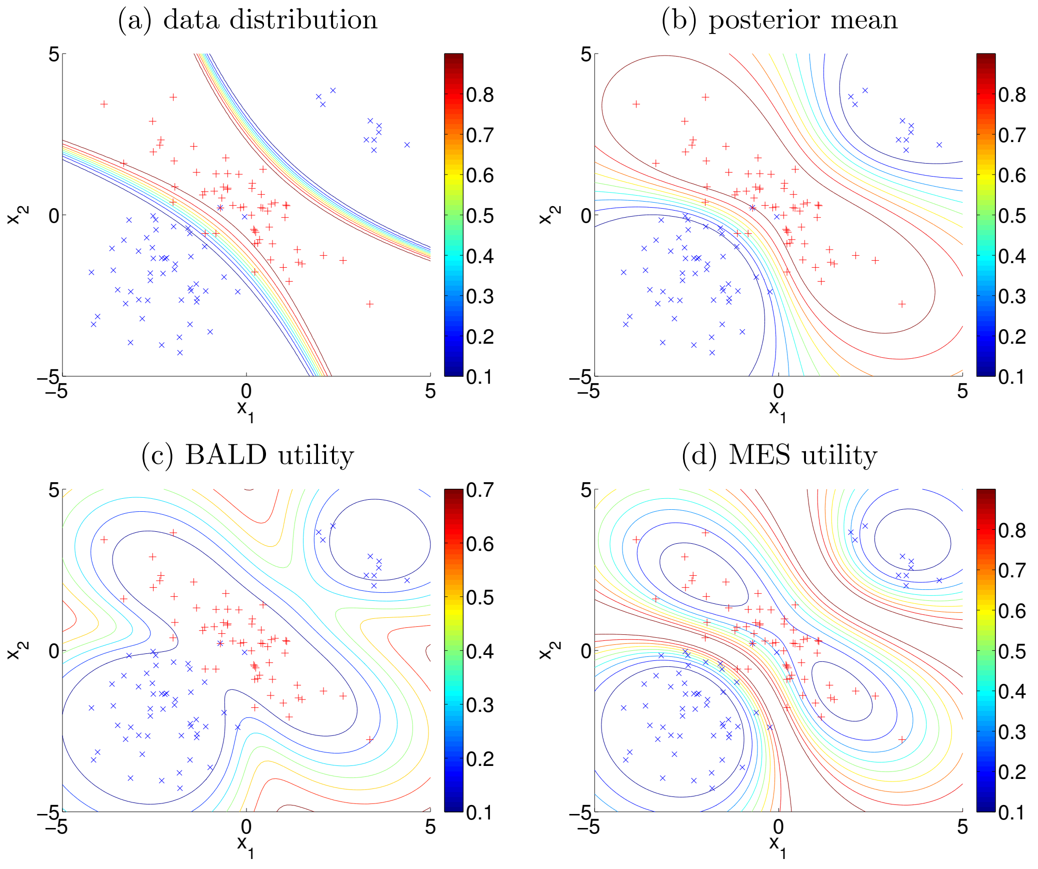

Figure 3.2: Binary classification problem. Points depict the training data. (a) True class probability contours. (b) Posterior mean prediction. Green regions indicate high predictive uncertainty. (c), (d), BALD and MES utility functions, respectively.

Dataset	D	|pool|	|test|	Note
GP-D5	5	250	250	in-model
Checkerboard	2	400	400	hand crafted
Distracting Block	2	194	250	hand crafted
Noisy Block	2	194	250	hand crafted
Austra	14	345	345	UCI
Branin	2	1000	1000	Branin fn. & Probit lik.
Cancer	9	342	341	UCI
Crabs	5	100	100	UCI
Letter D vs. P	16	804	804	UCI
Letter E vs. F	16	772	771	UCI
Vehicle	18	423	423	UCI
WDBC	30	285	284	UCI
Wine	13	100	78	UCI

Table 3.1: Statistics of the binary classification datasets.

	HMC	EP ($\overset{1}{\approx}$)	Laplace ($\overset{1}{\approx}$)
MC	0	7.51 ± 2.51	41.57 ± 4.02
$\overset{2}{\approx}$	0.16 ± 0.05	7.43 ± 2.40	40.45 ± 3.67

Table 3.2: Percentage approximation error (± 1 s.d.) for approximate inference algorithms (*columns*) and for evaluating Equation (3.7) (*rows*). Monte Carlo (MC) sampling from the posterior was used to evaluate the ground truth binary entropy in Equation (3.7), and Markov chain Monte Carlo, implemented using Hamiltonian Monte Carlo (HMC), to evaluate the posterior.

Branin is generated using a 2D Branin function passed through a probit likelihood. The real-world sets *Austra*, *Cancer*, *Crabs*, *Letter E vs. F*, *Letter D vs. P*, *Vehicle*, *WDBC* and *Wine* are classification datasets from the UCI repository.[1] Table 3.1 provides the statistics for all of the datasets and the size of the partitions used in the following experiments. We first quantify the loss from our approximation to the binary entropy function $\overset{2}{\approx}$ presented in Section 3.2.

Quantifying Approximation Losses

To compute BALD for GPC (3.8), two approximations were used: any standard approximate inference scheme that results in a Gaussian approximation to the posterior $\overset{1}{\approx}$, and an approximation to the binary entropy of the Gaussian CDF by a squared exponential $\overset{2}{\approx}$. We evaluate the loss incurred due to these approximations by replacing

[1] http://archive.ics.uci.edu/ml/datasets.html

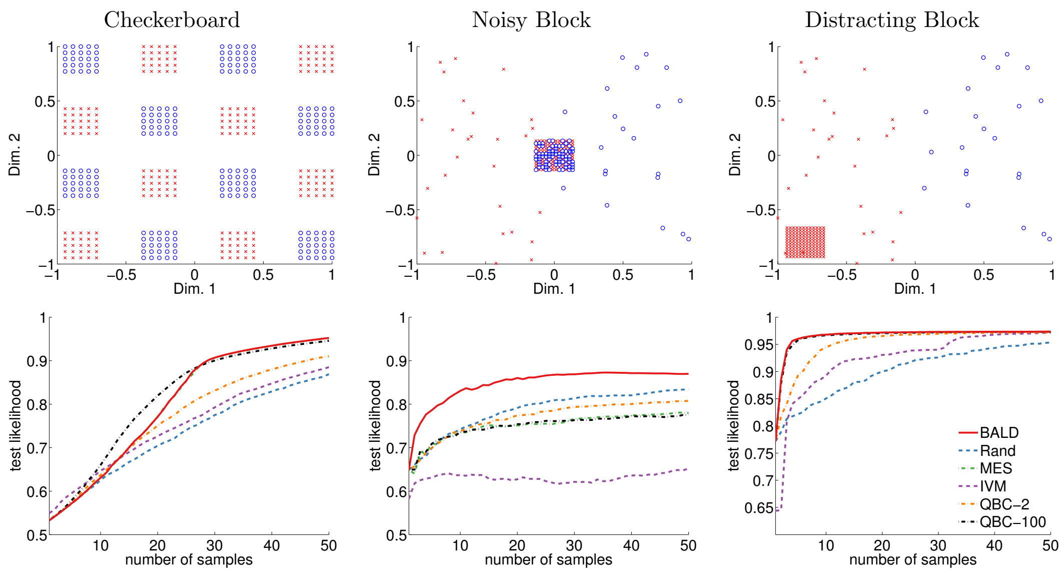

Figure 3.3: Active GPC experiments on three hand-crafted 2D datasets. *Top:* Depictions of the datasets, markers denote datapoints from the two classes. *Bottom:* Learning curves for each information theoretic active learning algorithm. The x-axis gives the number of active datapoints selected. The y-axis gives the performance measured using likelihood on the test set.

them with extensive Monte Carlo as the 'gold standard'. Ultimately we are interested in evaluating the ability of the algorithm to find the most informative sample, therefore a relevant measure of loss is the expected information loss, defined as

$$\frac{\max_{\mathbf{x}} I(\mathbf{x}) - I(\operatorname{argmax}_{\mathbf{x}} \hat{I}(\mathbf{x}))}{\max_{\mathbf{x}} I(\mathbf{x})} \cdot 100\% \,, \tag{3.9}$$

where I is the objective computed using the gold standard (Monte Carlo) and $\hat{I}$ is the approximate objective. Table 3.2 contains the information losses on the *Cancer* dataset. As noted in Section 3.2, the introduced approximation to the binary entropy $\overset{2}{\approx}$ can only yield a maximum error $< 0.3\%$ to the integral in Equation (3.7). Table 3.2 confirms that this approximation yields negligible information loss compared to standard methods of approximate inference, the Laplace approximation and EP. Laplace results in larger loss than EP, which is consistent with the comparison presented in Kuss & Rasmussen [2005]. Therefore, we use EP in all of the following GPC experiments.

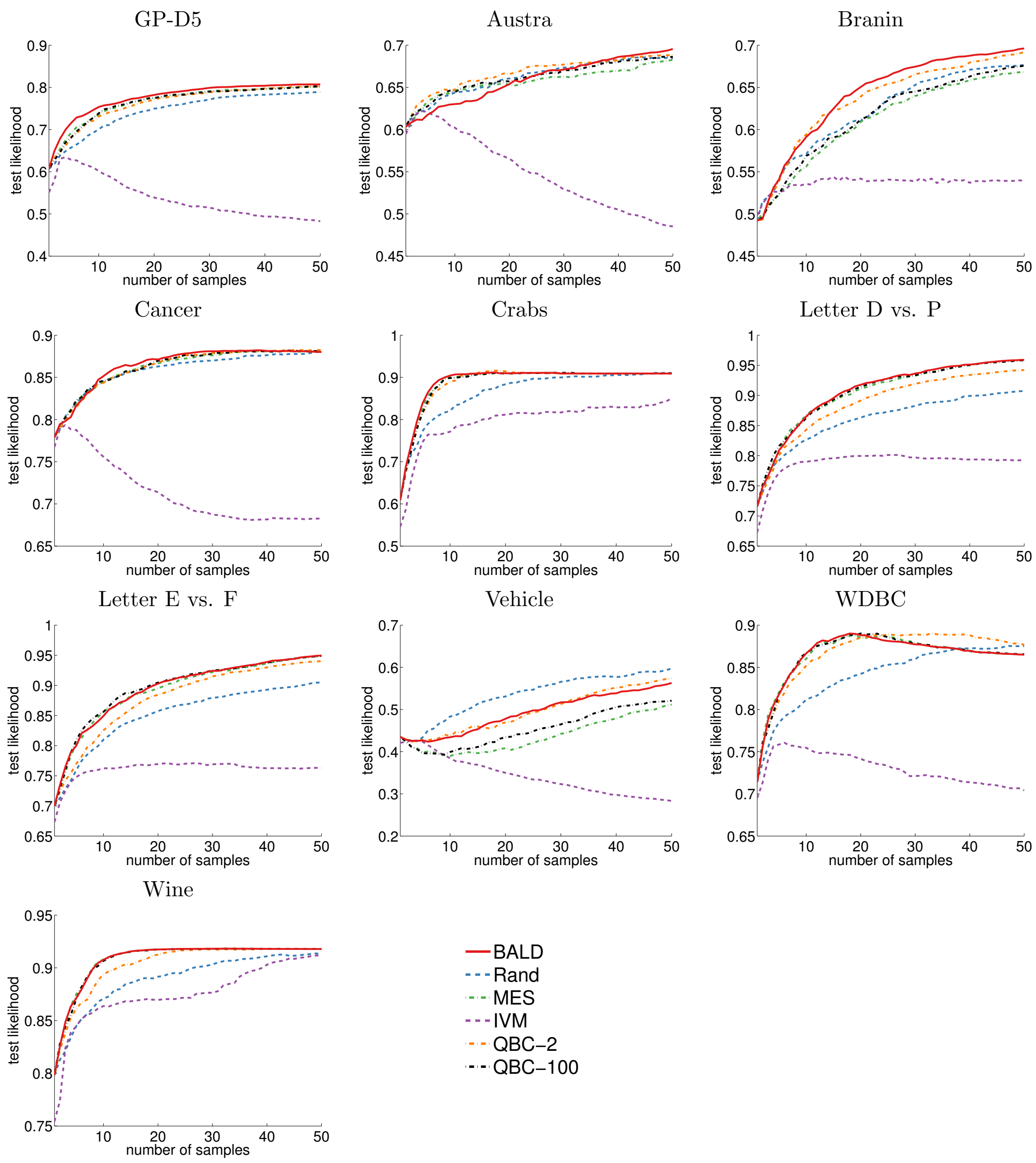

Figure 3.4: Performance versus number of active samples learning curves on the in-model and real-world datasets for all of the information-theoretic algorithms. Predictive performance is measured using exponentiated log likelihood on the test set.

Experimental Procedure

We followed a standard pool-based active learning scenario where the datapoints are selected from a fixed set (pool) of data. To do this, the datasets were first partitioned equally into *pool* and *test* sets, when fewer than 200 points were available, *pool* was

Dataset	BALD	Rand	MES	IVM	QBC$_2$	QBC$_{100}$	Emp	Kap07	Zhu03
GP-D5	**79.3**	77.2	78.8	55.5	78.6	78.7	69.3	78.6	79.0
Checkerboard	88.7	82.5	88.7	84.3	85.7	**89.2**	60.4	88.2	87.9
Distracting Bl.	**96.9**	93.4	**96.9**	94.7	96.2	96.9	82.0	96.5	96.5
Noisy Block	**86.1**	81.7	75.9	63.7	79.3	75.8	85.6	85.9	78.7
Austra	**68.2**	67.6	67.5	49.5	**68.2**	**68.1**	63.9	68.3	68.1
Branin	**67.4**	65.5	64.9	53.7	66.9	65.4	60.1	67.1	67.1
Cancer	**87.6**	**87.4**	**87.5**	71.8	**87.5**	**87.5**	81.0	88.2	88.1
Crabs	**86.5**	84.9	86.4	81.2	86.3	86.3	77.7	86.5	86.5
Letter D vs. P	**93.4**	89.1	**93.4**	81.3	92.1	**93.4**	76.2	92.7	92.7
Letter E vs. F	**92.4**	88.3	92.2	76.3	91.4	**92.4**	74.4	92.0	91.8
Vehicle	55.4	**58.1**	51.1	29.9	55.7	51.8	56.2	56.8	55.4
WDBC	86.4	**86.1**	86.4	69.7	**86.8**	86.4	75.2	88.0	88.1
Wine	**86.1**	85.1	**86.1**	84.5	85.9	**86.1**	80.3	86.2	86.2

Table 3.3: AUC for exponentiated log likelihood classification learning curves up to collecting 100 active samples. Bold indicates the best performing method (and those statistically indistinguishable) that does not observe the test data, underlined indicates the best performing overall.

given 100 so that the active learning algorithms had sufficient data to choose from. The algorithms were initialized by sampling five points at random from the pool and adding them to a *training* set. Each algorithm was trained upon this set and, using its active learning criterion, selected further points from the *pool* to be added to *training*. After each sample the GP was re-compute, and its predictive performance, measured using exponentiated log likelihood, was evaluated on *test*. The exponentiated log likelihood is given by $\exp\{\sum_{i\in\text{test}} \log P(y_i|\mathbf{x}_i, \mathcal{D})\}$. The likelihood is summed outside of the logarithm because this results in a proper scoring rule [Dawid, 2007]. A proper scoring rule rewards accurate estimation of the uncertainty in y_i, the true $P(y_i|\mathbf{x}_i)$ attains the highest score. Averaging the predictive distribution directly is not a proper scoring rule because predicting $\operatorname{argmax}_y P(y|\mathbf{x}_i, \mathcal{D})$ with probability one yields the best expected score.

In this experiment we were interested in evaluating the ability of the algorithms to learn the latent function f, and so the hyperparameters were fixed *a priori* to the type-II maximum likelihood estimate on the pool. We used an RBF kernel with ARD and additive noise. Therefore the hyperparameters corresponded to the length scales along each dimension, the signal variance and the noise variance. The entire procedure, including the random partitioning of the data was repeated 50 times.

Other Algorithms

We compare BALD to random sampling and information-theoretic alternatives for active GPC described in Section 2.4. These methods include Maximum Entropy Sampling (MES), Query by Committee (QBC) with 2 and 100 committee members and the Informative Vector Machine (IVM). This last algorithm does not strictly perform active learning as it has access to the labels before sampling. We also evaluate two decision theoretic algorithms, Zhu *et al.* [2003] and Kapoor *et al.* [2007], denoted Zhu03 and Kap07 respectively. These decision theoretic algorithms should outperform BALD because they observe the test data and minimize classification error directly. However, as described in Section 2.4, they have much higher computational cost. Finally, we minimize the empirical training error directly (Emp); this is not a standard algorithm, but is used for the analysis of Kap07. Due to the high computation cost of decision theoretic algorithms, when actively selecting new data, the pool and test sets were subsampled to 150 points with these algorithms. This approximation yielded no noticeable detriment to performance.

Results

Figure 3.3, bottom, shows the learning curves on the hand-crafted datasets, and Figure 3.4 presents the rest of the datasets. Table 3.3 contains the area under the (exponentiated log likelihood) curve (AUC) for each method on each dataset up to the active collection of 100 samples. AUC evaluates the quality of the algorithms for choosing samples useful for classification performance right from the start.

BALD is the best performing information theoretic algorithm, and is even competitive with the decision theoretic methods (Zhu03, Kap07). BALD is robust to noisy data, e.g. see *Noisy Block* in Figure 3.3. On this dataset, MES focuses on the noisy region, and hence does not learn the classification boundary well. On the noiseless artificial datasets, *Checkerboard* and *Distracting Block*, MES behaves the same as BALD. Therefore, Table 3.3 indicates that some of the real-world datasets, such as *Cancer*, *Letter D vs. P*, *WDBC* and *Wine*, have low noise since MES performs as well as BALD.

QBC can work well, and with more query members it becomes a closer approximation to BALD. However, QBC's deterministic vote criterion can lead to poor performance with noisy data, like MES, e.g. *Noisy Block*, *Cancer*. The IVM can exhibit pathological performance. For example on *Noisy Block* the IVM focuses heavily on the block. This is likely to be because the algorithm has a transductive bias encouraging

Dataset	**BALD**	**MES**	**IVM**	**QBC$_2$**	**QBC$_{100}$**	**Emp**	**Kap07**	**Zhu03**
GP-D5	5.5	7.2	0.6	6.1	13.9	4268.7	3864.2	3369.1
Checkerboard	5.7	6.6	0.6	5.7	19.2	3578.1	4326.2	4193.7
Distracting Bl.	5.6	5.9	0.9	5.8	6.5	5404.2	6071.2	5848.6
Noisy Block	5.5	8.8	0.9	5.7	9.5	3247.8	3261.6	2749.8
Austra	4.3	3.1	0.7	3.4	6.6	3145.1	3466.4	3249.1
Branin	9.5	11.1	0.9	11.1	63.6	3045.3	2949.6	2718.7
Cancer	4.5	5.0	0.6	4.8	14.1	3747.2	3865.3	3564.1
Crabs	5.7	5.9	0.6	5.1	6.7	2005.0	1671.4	1759.3
Letter D vs. P	9.6	10.9	1.0	11.1	58.3	3675.4	4350.5	4217.1
Letter E vs. F	8.7	7.9	0.8	9.2	32.2	3609.8	4473.6	4354.7
Vehicle	9.6	9.1	0.8	10.3	23.4	3658.2	3949.8	3582.7
WDBC	7.0	6.2	0.8	7.2	16.7	4384.0	5444.7	5211.3
Wine	4.8	5.4	0.9	5.8	6.1	1794.8	2044.7	1973.1

Table 3.4: Cumulative clock times (seconds) to compute the active learning criteria for each method up to collecting 100 samples.

it to shrink the posterior near the data it has already collected, and hence not explore.

Overall, the decision theoretic methods perform comparably to BALD. Observing the test data helps in come cases, for example, on *Checkerboard* the decision theoretic methods choose one point from the centre of each island of data during the first 16 samples. BALD also chooses a point from each island, but does not necessarily pick central points because it is unaware that it will have to classify the rest of the data in the surrounding island. However, these decision theoretic methods are very expensive, the clock times are presented in the following section. Emp performs poorly because it only reinforces the classification of the observed data, so fails to explore.

Random sampling usually performs worse than the active methods, except on *Vehicle* where it performs best. On *Vehicle* the GP made poor predictions as little data is available and the length scales were short in most dimensions. With such imprecise predictions all active schemes performed poorly. To get obtain good performance on such high dimensional data, more structured kernels [Duvenaud *et al.*, 2012] or GP embeddings [Garnett *et al.*, 2013] may be required to capture the structure of the latent function from just a few datapoints.

Computational Complexity and Run Times

We recorded the cumulative clock times required to compute the active learning utility functions up to collecting 100 samples. All algorithms were implemented in MATLAB.

To perform EP, we used the GPML toolbox.[1] For the IVM, we used code made publicly available by the authors.[2]

Table 3.4 contains the recorded times. The information theoretic active learning algorithms have similar computational time, which was mostly spent running EP once per iteration to compute the predictive distributions on the pool. After collecting M training points, and for N candidates in the pool, the complexity of BALD, MES and QBC to select the next active sample is $\mathcal{O}(M^3 + NM^2)$. M^3 computations are required to re-train the posterior, and NM^2 to make predictions for each candidate. Should time be critical, this could be reduced with sparse approximations, such as the Fully Independent Training Conditional [Snelson & Ghahramani, 2006] or incremental, rank-one posterior updates.

It is not possible to compute exact entropies of f directly using Equation (2.8), but, as in the IVM, one could approximate this quantity with the entropy of the posterior marginal corresponding to the locations of the data. The cost of this approximation would be $\mathcal{O}(NM^3)$. However, the IVM is fast since it uses ADF for approximate inference and ADF is a single pass approximation to EP. Note that, for a fair comparison of performance, we used EP when making predictions with the data chosen by the IVM. The time spent in evaluation is not included in Table 3.4.

The decision theoretic methods are much slower. In practice, the time to compute Zhu03 and Kap07 up to 100 samples, even after subsampling the pool and test sets to 150 points, was around 1 hour, whereas BALD took < 10s. The cost to select a sample with these methods is $\mathcal{O}(NM^3 + NTM^2)$, where T is the number of test points under consideration. These decision theoretic algorithms could be sped up by making further approximations when re-computing the posterior, such as rank-one updates or ADF. For a fair performance comparison we used the same EP inference scheme for all of the active learning methods. However, even with additional approximations, decision theoretic methods are still likely to be much slower than BALD.

3.2.2 Summary

We have demonstrated that BALD is an effective framework to do active classification with Gaussian processes. Its two main advantages are (i) that it only requires computing entropies of the Bernoulli output variable, like MES, and (ii) it only requires re-computing the posterior once, *after* the acquisition of each active sample. Equation (3.8) allows one to compute information gains to the infinite dimensional latent

[1] http://www.gaussianprocess.org/gpml/code/matlab/doc/
[2] http://staffwww.dcs.shef.ac.uk/people/N.Lawrence/ivm/

function accurately. Besides standard approximate inference, the only approximation is to the binary entropy, which is both theoretically and empirically negligible. Methods that work directly with entropies over the latent function must approximate the utility function more severely. Experiments indicate that BALD's performance compares favourably to many other active learning methods for classification, even decision theoretic algorithms that have access to the test data and have a much greater computational cost.

On *Austra* BALD exhibits weaker performance in the first couple of samples, but then rapidly overtakes the other algorithms. This is likely to be because the hyperparameters were fixed, and their uncertainty was not accounted for. Dealing with hyperparameter uncertainty appropriately is a major research focus for GPs, as the marginal likelihood surface can be highly complex. However, appropriate hyperparameter management is particularly important in active learning, where one usually works in the low-data regime, and hence uncertainty is high. In the next section we address this problem in the context of Gaussian process regression.

3.3 Active GP Regression with Unknown Hyperparameters

Obtaining good performance with GPs requires appropriate hyperparameter management. To address active GPR with unknown hyperparameters, we extend a new technique for hyperparameter marginalization and inference, the marginal Gaussian Process (MGP) [Garnett *et al.*, 2013], to our framework. We use the 'focused' version of BALD in Equation (2.15) to learn actively either about the hyperparameters or the latent function itself. This results in a robust pipeline for active GPR. First, we present the MGP, and then derive the appropriate utility functions to compute BALD in this domain.

3.3.1 Marginal Gaussian Process

The Marginal Gaussian Process (MGP) is a general means of managing uncertainty in GP hyperparameters appropriately, originally proposed for learning the parameters of a linear embedding of a high dimensional GP [Garnett *et al.*, 2013]. It performs approximate marginalization of hyperparameters, improving upon ubiquitous type-II maximum likelihood estimation. An alternative to hyperparameter marginalization would be to sample their values. However, such an approach would require re-inverting the kernel matrix for every sample. The MGP has the advantage that the kernel matrix only needs to be inverted once, as with fixed hyperparameters.

The MGP makes two approximations in order to make the required integrals tractable. First, assume a Gaussian posterior over hyperparameters $p(\xi|\mathcal{D}) = \mathcal{N}(\xi; \hat{\xi}, \Sigma)$. If the true posterior is non-Gaussian, as is usual, a Laplace approximation is used. For this, the mean is set to the MAP estimate, $\hat{\xi}$, and the Hessian of the log posterior is used to fix the covariance as $\Sigma^{-1} = -\nabla\nabla^\top \log p(\xi|\mathcal{D})|_{\hat{\xi}}$.

The second approximation is to linearize the mean of the conditional predictive distribution $p(f_\mathbf{x}|\mathcal{D}, \xi)$ as a function of ξ, and approximate the covariance as constant around the MAP hyperparameters, $\hat{\xi}$,

$$p(f_\mathbf{x}|\mathcal{D}, \xi) = \mathcal{N}(f_\mathbf{x}; \mu(\xi), \sigma^2(\xi)) \approx \mathcal{N}(f_\mathbf{x}; a^\top \xi + b, c)\,. \tag{3.10}$$

The parameters of the approximation a, b, c are fixed by matching derivatives to the true predictive distribution $p(f_\mathbf{x}|\mathcal{D}, \xi)$. The results is a Gaussian marginal posterior, given by

$$p(f_\mathbf{x}|\mathcal{D}) = \mathcal{N}(f_\mathbf{x}; \mu_{\mathbf{x}|\mathcal{D}}, \sigma^2_{\mathbf{x}|\mathcal{D}}) \tag{3.11}$$

$$\text{where } \mu_{\mathbf{x}|\mathcal{D}} = \hat{\mu}_{\mathbf{x}|\mathcal{D}}\,,$$

$$\sigma^2_{\mathbf{x}|\mathcal{D}} = \frac{4}{3}\hat{\sigma}^2_{\mathbf{x}|\mathcal{D}} + \frac{\partial \hat{\mu}_{\mathbf{x}|\mathcal{D}}}{\partial \xi}^\top \Sigma \frac{\partial \hat{\mu}_{\mathbf{x}|\mathcal{D}}}{\partial \xi} + \frac{1}{3\hat{\sigma}^2_{\mathbf{x}|\mathcal{D}}} \frac{\partial \hat{\sigma}^2_{\mathbf{x}|\mathcal{D}}}{\partial \xi}^\top \Sigma \frac{\partial \hat{\sigma}^2_{\mathbf{x}|\mathcal{D}}}{\partial \xi}\,,$$

and $\hat{\mu}_{\mathbf{x}|\mathcal{D}}$, $\hat{\sigma}^2_{\mathbf{x}|\mathcal{D}}$ denote the posterior predictive mean and variance under $\hat{\xi}$, respectively. All gradients in (3.11) are evaluated at $\hat{\xi}$, notation omitted for clarity. The marginalization does not change the MAP predictive mean, but it augments the variance appropriately to account for the hyperparameter uncertainty. Incorporating this uncertainty is crucial for robust active learning.

We investigate empirically the information lost due to making the MGP approximations, in Section 3.3.3, and find that it is usually small.

3.3.2 Focused Active Learning with the MGP

With the MGP, all predictive distributions are Gaussian, so predictive entropies may be computed analytically. During training we wish to learn optimally about the hyperparameters, but we do not necessarily wish to use up samples to learn the latent function everywhere. Therefore, we use the focused-BALD utility, for reference, the general form of this utility function is given again,

$$\mathcal{U}(\mathbf{x}) = \mathrm{H}[p(y|\mathbf{x},\mathcal{D})] - \mathbb{E}_{p(\theta|\mathcal{D})}\mathrm{H}[\mathbb{E}_{p(\phi|\theta,\mathcal{D})}p(y|\mathbf{x},\theta,\phi)]\,. \tag{3.12}$$

In this setting the hyperparameters are included in the variables of interest, and the latent function is the 'nuisance parameter', that is $\theta = \xi$ and $\phi = f$. The first term in Equation (3.12) is the entropy of the MGP predictive distribution, $\mathrm{H}[p(y|\mathbf{x},\mathcal{D})] \propto 0.5\ln(2\pi e\sigma^2_{\mathbf{x}|\mathcal{D}})$. The second term is computed by assuming that the predictive variance is constant with respect to the hyperparmeters under the posterior. This is the same approximation that is used by the MGP in Equation (3.10). The resulting term is

$$\mathbb{E}_{p(\xi|\mathcal{D})}\mathrm{H}[\mathbb{E}_{p(f_{\mathbf{x}}|\xi,\mathcal{D})}p(y|f_{\mathbf{x}})] \approx \mathrm{H}[\mathbb{E}_{p(f_{\mathbf{x}}|\hat{\xi},\mathcal{D})}p(y|f_{\mathbf{x}})] = 0.5\ln(2\pi e\hat{\sigma}^2_{\mathbf{x}|\mathcal{D}})\,.$$

Now the utility function for active learning of the hyperparameters for GPR is

$$\mathcal{U}_{\xi}(\mathbf{x}) \propto \ln(\sigma^2_{\mathbf{x}|\mathcal{D}}) - \ln(\hat{\sigma}^2_{\mathbf{x}|\mathcal{D}})\,. \tag{3.13}$$

Intuitively, Equation (3.13) prefers locations with large marginal uncertainty under the MGP, but low uncertainty *given* the MAP hyperparameters. That is, we seek data where we have additional predictive entropy over the MAP prediction due to the hyperparameter uncertainty.

The complexity to compute the MGP and Equation (3.13) is $\mathcal{O}(|\xi|^3 + M^3 + NM^2)$. The additional $|\xi|^3$ over the complexity for classification, presented in Section 3.2.1, is to invert the hyperparameter posterior covariance in Equation (3.11). Again, working directly with posterior entropies would be much more expensive, this would cost $\mathcal{O}(N|\xi|^3 + NM^3)$. Note that with fixed hyperparameters, the design is independent of y and can be pre-computed [MacKay, 1992b], this is no longer the case as $p(\xi|\mathcal{D})$ depends on the collected outputs.

If we want to learn the latent function then we should try to decrease the posterior entropy over all model parameters. Now f is also included in the 'parameters of interest' θ and so the second term in Equation (3.12) becomes $\mathbb{E}_{p(f_{\mathbf{x}},\xi|\mathcal{D})}\mathrm{H}[p(y|f_{\mathbf{x}},\xi)]$. We assume homoscedastic observation noise, so this term is constant with respect to $\mathbf{x}$. Therefore in GPR, active learning of the latent function is equivalent to MES,

$$\mathcal{U}_{f,\xi}(\mathbf{x}) = \sigma^2_{\mathbf{x}|\mathcal{D}}\,. \tag{3.14}$$

We may also consider learning only f and treating the hyperparameters as nuisance

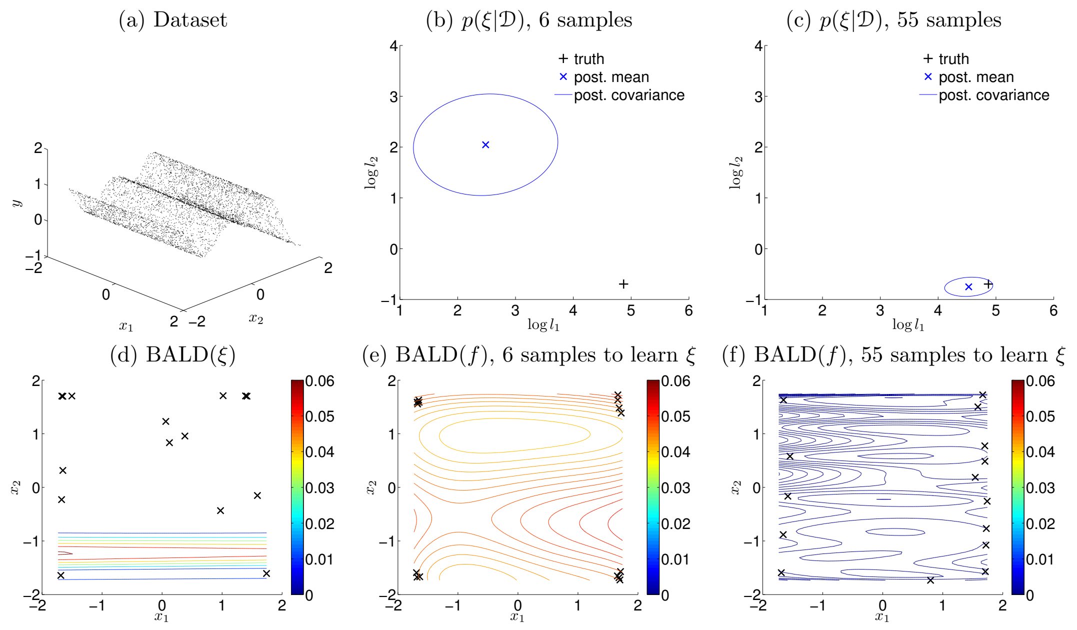

Figure 3.5: Demonstration of BALD for GPR with unknown hyperparameters. (a) 2D in-model dataset. (b), (c) Posterior over hyperparameters after observing 6 and 55 samples respectively. (d) First 15 samples chosen by BALD(ξ), Equation (3.13). Contours depict the posterior predictive variance. (e), (f) First 15 samples chosen by BALD(f), Equation (3.14), after 6 and 55 samples were used to learn ξ, respectively.

parameters, including them in ϕ. Now the second term in (3.12) is given by

$$\mathbb{E}_{p(f|\mathcal{D})}\mathrm{H}[\mathbb{E}_{p(\xi|f,\mathcal{D})}p(y|f_{\mathbf{x}},\xi)]\,.$$

The expectation over $p(f|\mathcal{D})$ may be computed using the MGP approximation to the marginal predictive distribution in Equation (3.11). In general, we require Bayes' rule to compute $p(\xi|f,\mathcal{D})$. However, if we are again consistent with the approximations made by the MGP, the predictive variance and hence entropy is independent of ξ and so this term is constant. Under these assumptions the algorithm also reduces to MES.

Demonstration

We demonstrate the behaviour of Equations (3.13) and (3.14) on a low-noise artificial dataset generated from a GP with a two-dimensional input. Figure 3.5(a) depicts the dataset. The generating GP has a long length scale along dimension x_1 and a short length scale along dimension x_2. The corresponding true log length scales (l_1,l_2) are denoted by the black markers (+) in panels (b), (c). These panels also show the Gaussian MGP approximation to the posterior over the length scales, after drawing samples using BALD to learn the hyperparameters, BALD(ξ), Equation (3.13). The algorithm was initialized with 5 random samples then actively selected 1 and 50 samples in (b) and (c) respectively. With only 6 samples in total, the posterior mean for ξ falls near to the prior mean at $(2, 2)$. As expected with more data the variance shrinks and the mean moves towards the true hyperparameter values. Panels (d)-(f) show the locations of selected datapoints, $\mathbf{x}$, and contours denote the posterior predictive variance $\sigma^2_{\mathbf{x}|\mathcal{D}}$. Panel (d) depicts the first 15 active samples selected by BALD(ξ) to learn the hyperparameters, and the predictive variance after training the GP on these samples. Panels (e) and (f) show the first 15 active samples chosen using BALD to learn the latent function, BALD(f), Equation (3.14). In (e) and (f), the hyperparameters are fixed to their MAP values given in the panels (b) and (c), that is, learnt using 1 and 50 (plus 5 initial) active samples, respectively.

Panel (d) shows the behaviour of BALD when learning the hyperparameters alone. Some datapoints have been placed very close together, and some far apart, in order to learn the length scales. However, this strategy does not try to learn the latent function, and completely ignores a large region of input space, where the predictive variance remains high. Panel (e) shows that trying to learn f with poor estimates of the hyperparameters can be pathological. Remember, the hyperparameters used in (e) are the MAP values given in (b). The algorithm focuses on the extreme corners of input space, and the posterior variance remains high everywhere. However, in panel (f), when the hyperparameters have been learnt from 55 samples, BALD(f) behaves much more sensibly and the variance is low everywhere. The algorithm knows that the length scale along dimension x_1 is long, so simply needs to observe the y values at either edge of the region along this dimension in order to interpolate to the rest of input space. However, the length scale along x_2 is short, so BALD(f) explores that dimension appropriately. In conclusion, BALD(ξ) does not waste data seeking to learn the latent function everywhere, so that it focuses on collecting maximal information about the hyperparameters. As a result, to learn the latent function one needs to use BALD(f). However, this algorithm will probably do a poor job if the hyperparameters have not been reasonably estimated already. With these initial findings we design a two-phased experiment to test the algorithms on real-world data.

Dataset	D	|pool-model|	|pool-eval|	|test|	Note
GP-D1	1	2500	1500	1000	in-model
GP-D2	2	2500	1500	1000	in-model
GP-D5	5	2500	1500	1000	in-model
Auto	7	199	99	99	UCI
Branin	2	2500	1500	1000	Branin + noise
Concrete	8	515	257	258	UCI
Energy1	8	384	192	192	UCI
Energy2	8	384	192	192	UCI
Machine	6	105	52	52	UCI
Servo	4	100	33	34	UCI
Yacht	6	154	77	77	UCI

Table 3.5: Statistics of the regression datasets.

	MC (term 2)	MGP (term 2)
MC (term 1)	0.00± 0.00	2.44± 2.74
MGP (term 1)	7.73± 14.12	1.48± 2.35

Table 3.6: Percentage information loss (3.9) when replacing the MGP with MC sampling in either term in Equation (3.12). Values are averages across all datasets.

3.3.3 Experiments

We evaluate the ability of BALD to actively learn both the hyperparameters and the latent function on a number of real-world regression datasets. For this we use a two-phase 'model learning' and 'evaluation' experiment. We also empirically evaluate the information loss due to the MGP approximation.

Datasets

We used three in-model datasets, *GP-D1*, *GP-D2* and *GP-D5*, the Branin function with Gaussian additive noise, *Branin*, and seven UCI regression datasets: *Auto*, *Concrete*, *Energy 1*, *Energy 2*, *Machine*, *Serve* and *Yacht*. Table 3.5 gives the statistics of the datasets and the size of the experimental splits.

MGP Approximation Loss

We evaluate the information loss (3.9) from the MGP approximation when learning the hyperparameters. Either term in BALD (3.12) may be computed by replacing the MGP approximation to the integral over $p(\xi|\mathcal{D})$ with MC sampling. When using MC for both terms in Equation (3.12), information loss is measured relative to an independent set of samples to verify convergence. Table 3.6 contains the results. Interestingly, approximating both terms using the MGP is the best alternative to sampling both terms, yielding a loss of only about 1.5%. This is because BALD contains the difference between two terms, and to retain the correct *ranking* of candidates, it is important be consistent in approximating each. The linearization and independence assumptions in the MGP result in a small *relative* underestimate to both terms in Equation (3.12) for datapoints with high predictive entropy. Note that the MGP is typically conservative in its *absolute* estimates of predictive uncertainty, mostly due to the 4/3 pre-multiplying factor to the variance in Equation (3.11). By being consistent, the ranking of the pool correlates highly with the MC estimate.

Experimental Procedure

We evaluate BALD for learning both the hyperparameters and the latent function. Therefore, we simulate a two-phase learning scenario. In the first, 'model-learning', we seek to actively learn the hyperparameters. In the second, 'evaluation', we actively learn the latent function, using the learnt hyperparameters. The datasets were split randomly into three sets: *pool-model*, *pool-eval*, and *test*, the sizes of each split on each dataset are given in Table 3.5.

In Phase one, a fixed budget of $M = 50$ datapoints was actively sampled from *pool-model* and the model was retrained after each sample. We used Equation (3.13) to learn optimally about the hyperparameters, BALD(ξ). ξ includes the length scales, signal variance and observation noise of our RBF-ARD kernel. In Phase two, the hyperparameters are fixed to their MAP values at the final iteration in Phase one. We used Equation (3.14), BALD(f), to learn the latent function by actively sampling from *pool-eval*. After each sample, the GP is retrained and evaluated on *test*. In both phases, we compare to random sampling as a control. We also compare to MES for Phase one (in Phase two, BALD(f) is equivalent to MES). The entire procedure, including random partitioning of the data, was repeated 100 times.

Results

Figure 3.6 depicts the learning curves for each algorithm. These curves show the test log likelihood as a function of the number of samples in Phase two. Table 3.7 summarizes these learning curves by the performance after selecting 50 samples in both Phase one and two. BALD is effective for both the learning of the hyperparameters and the latent function. With a limited budget, efficient learning of the hyperparameters may

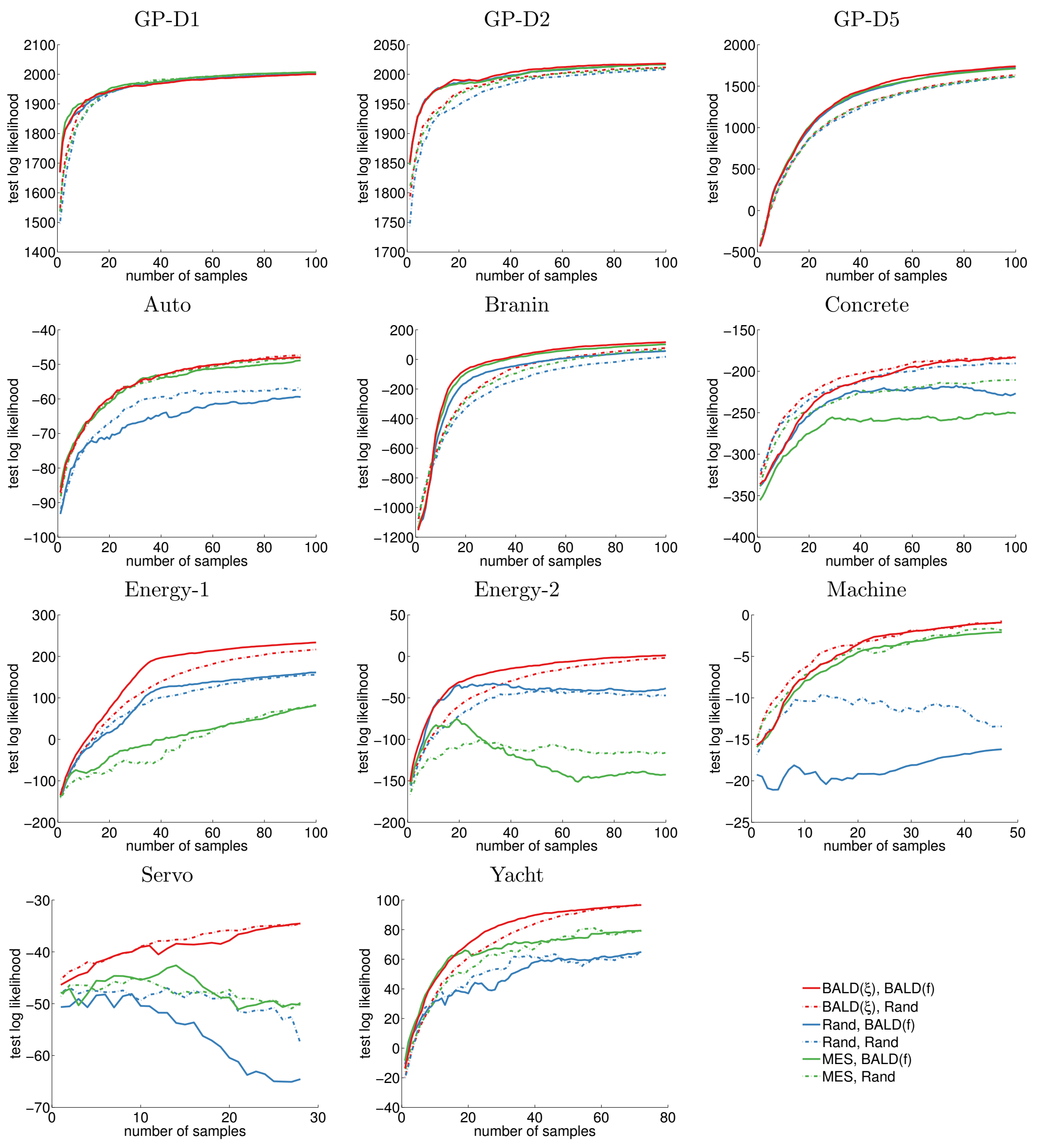

Figure 3.6: Active GPR learning curves as a function of the number of samples taken in Phase two. 50 samples were used to learn the hyperparameters in Phase one.

Dataset	**BALD(ξ), BALD(f)**	**BALD(ξ), Rand**	**Rand, BALD(f)**	**Rand, Rand**	**MES, BALD(f)**	**MES, Rand**
GP-D1	**1.98e+03**	**1.98e+03**	**1.98e+03**	**1.99e+03**	**1.99e+03**	**1.99e+03**
GP-D2	**2.01e+03**	2e+03	**2e+03**	1.99e+03	**2.01e+03**	2e+03
GP-D5	**1.55e+03**	1.37e+03	1.5e+03	1.35e+03	1.51e+03	1.37e+03
Auto	**-51.4**	**-51.5**	-63.9	-58.2	**-52.8**	**-51.5**
Branin	**52.6**	-15.4	-13.4	-91.9	**40.7**	-44.6
Concrete	**-205**	**-198**	-222	**-202**	-259	**-223**
Energy-1	**206**	164	131	112	11	-1.2
Energy-2	**-11**	-20.9	**-41.8**	-41.9	-131	-113
Machine	**-0.93**	**-0.75**	-16.2	-13.4	-2.09	-1.84
Servo	**-34.5**	**-34.5**	-64.6	-57.6	-50.2	-49.8
Yacht	**92.6**	90.4	59.9	58	73.6	75.1

Table 3.7: GP regression experiments. Log likelihood after obtaining 50 samples to learn the hyperparameters in Phase one and 50 samples to learn the latent function in Phase two. Bold denotes the best performing algorithm, and those statistically indistinguishable according to a paired t-test. Each column is labelled with the algorithm used in Phase 1, Phase 2.

be crucial, on some datasets, e.g. *Machine* and *Servo*, the performance after learning the hyperparameters from 50 random samples is poor. Using MES in Phase one is much less robust than BALD, and on some data (*Energy-1,2* and *Concrete*) performs very poorly, worse than random sampling. MES effectively tries to learn the latent function too early, as in Equation (3.14). By doing so it fails to gain information about the hyperparameters, which results in poor predictions and hence, poor samples. As in our classification experiments, BALD is also effective for learning the latent function – particularly when the hyperparameters have been well estimated in Phase one. Using BALD for both phases performs best (or joint best) on all datasets.

We also evaluated the methods using root mean squared error (RMSE), Table 3.8 contains the results. As with log likelihood, using BALD in both phases performs best overall. However, when evaluating with RMSE, using BALD for learning the latent function is more important. Rand+BALD(f) and MES+BALD(f) have better relative performance; for example, Rand+BALD(f) matches BALD(ξ)+BALD(f) more often when using RMSE than with log likelihood. RMSE only scores the predictive mean, and ignores predictive uncertainty. Incorrect values of the signal noise hyperparameter can yield a large detriment to the predictive log likelihood, but may have little effect the posterior mean, and hence RMSE many not be affected greatly. This indicates that using active learning is important for learning the noise as well as the length scales.

Dataset	**BALD(ξ), BALD(f)**	**BALD(ξ), Rand**	**Rand, BALD(f)**	**Rand, Rand**	**MES, BALD(f)**	**MES, Rand**
GP-D1	**0.033**	**0.033**	**0.033**	**0.033**	**0.033**	**0.033**
GP-D2	**0.032**	0.033	0.032	0.033	0.032	0.033
GP-D5	**0.051**	0.069	0.054	0.071	**0.052**	0.067
Auto	**0.396**	**0.400**	**0.400**	0.408	**0.395**	**0.398**
Branin	**0.226**	0.258	0.243	0.280	0.229	0.267
Concrete	**0.512**	**0.508**	**0.507**	**0.508**	0.525	**0.512**
Energy-1	**0.092**	0.159	0.104	0.176	0.145	0.187
Energy-2	0.242	0.259	**0.218**	0.255	0.246	0.264
Machine	**0.564**	**0.565**	**0.581**	**0.579**	**0.575**	**0.572**
Servo	**0.545**	**0.544**	**0.555**	**0.536**	**0.530**	**0.540**
Yacht	**0.076**	0.081	**0.075**	0.080	**0.077**	0.083

Table 3.8: As Table 3.7, but using RMSE to evaluate the test error.

3.3.4 Summary

Appropriate management of hyperparameter uncertainty is important to obtain good performance in Gaussian process regression. This is particularly important when working with small quantities of data. The MGP provides a cheaper alternative to sampling for learning and marginalizing over uncertain hyperparameters. With BALD can gain maximal information about the hyperparameters alone. Once sufficient information has been collected, one can actively learn the latent function also. Our experiments indicate that in both phases of learning, active sampling can substantially improve predictive performance.

3.4 Conclusions and Extensions

In conclusion, Gaussian process classification and regression models are models for which BALD is particularly useful. This is because the parameter space is infinite dimensional, and so directly computing posterior entropies necessarily requires substantial approximations which are normally not inductive. In GPR and GPC output space is only one-dimensional, so computing predictive entropies is relatively straightforward. Furthermore, with non-conjugate likelihood functions, such as the probit, or when doing hyperparameter learning, approximate inference can be expensive, so using BALD to reduce the number of updates to one-per-sample is critical.

In classification, Equation (3.8) intuitively balances uncertainty due to lack of knowledge in the posterior and inherent noise. This provides an information theoretically motivated generalization of the popular technique MES, which is appropriate for GPR with fixed hyperparameters, to classification. The ability to focus learning upon particular parameters of interest enables us to perform robust active learning in the face of uncertain hyperparameters in GPR.

BALD is submodular, hence near-optimal, if the observations are conditionally independent given the parameters of interest (Section 2.3.6). With GPs this is true if the latent function is the parameter of interest. However, when learning the hyperparameters using the MGP, Equation (3.13), this is no longer the case. Nevertheless, we find empirically that BALD is effective for learning the hyperparameters alone even with the myopic assumption. However, due to the objective not being submodular, further performance gains might be achieved by considering a longer horizon.

A direct extension would be to apply the MGP to non-conjugate likelihoods, such as those used in GPC. The challenge here is to devise a method to propagate the uncertainty $p(\xi|\mathcal{D})$ through any approximate inference technique. Active GPs have recently been proposed to tackle global optimization [Hennig & Schuler, 2012], and quadrature [Osborne *et al.*, 2012]. To apply BALD, computing the second term in Equation (2.10) is the central challenge as one must condition the predictions on a particular optimum or non-negative function with a particular integral, respectively. However, as for the models presented here, using BALD for GP optimization and quadrature is potentially rewarding as it can avoid the expensive inferences required in previous approaches.

Chapter 4

Adaptive Quantum State Tomography

Quantum information theory has become a popular field in physics and statistics. This is largely due to the allure of quantum computing, but also the potential for secure quantum communication and highly accurate quantum metrology. Quantum computers can, in theory, solve certain problems exponentially faster than classical computers, such as the factorization of large numbers which is central to cryptography [Shor, 1994]. The engineering required to realize a quantum computer is extremely challenging, but this is an exciting time for this field as early prototypes are able to implement rudimentary algorithms [Ladd *et al.*, 2010].

One problem that arises when working with quantum systems is how to characterize the quantum states being produced by the system. Due to the fundamental laws of quantum mechanics, one cannot determine a quantum state exactly from a single measurement. Therefore, in *quantum tomography* a series of measurements are made on copies of the state, from which the state is estimated [Paris & Řeháček, 2004]. The problem is that many measurements may be required to attain appropriate *fidelity*[1] in the estimation. We employ the active learning methods developed in this thesis to vastly reduce the number of measurements required. Further, in collaboration with experimentalists, our active learning algorithm was implemented in a laboratory experiment using polarized qubits. In these experiments, the large simulated gains are realized in practice.

The chapter is structured as follows, Section 4.1 provides a brief primer on quantum

[1] This is a technical term that will be formalized in Section 4.1.3. As the name implies, it measures the accuracy of an estimate of the state.

statistics. Next we describe how Bayesian inference can be used in this domain. A new algorithm for adaptive experimental design in quantum tomography based upon the BALD framework is proposed in Section 4.3. We then validate our approach with simulated experiments. In Section 4.6.2 we show how to deal with real-world experimental noise, and present the laboratory experiments in Section 4.6.3.

4.1 Primer on Quantum Statistics

The necessities required to follow the work in this chapter are presented here. For a comprehensive introduction to quantum statistics see Petz [2008].

4.1.1 States and Density Matrices

The fundamental unit in quantum mechanics is a qubit, which has two states denoted $|0\rangle$, $|1\rangle$. The angled brackets are the "bra-ket" notation, for our purposes, they simply indicate that the vector inside is complex-valued. A classical bit can exist in only one of the two states, 0 or 1. However, a qubit, denoted $|\psi\rangle$, can exist in any (complex) linear *superposition* of the states. A qubit stochastically realizes one of its states when measured, at which point the qubit is destroyed and no further measurements can be performed on it. Mathematically, a qubit $|\psi\rangle$ is a complex vector of length one. In a single qubit system $|\psi\rangle$ is two-dimensional and hence has two degrees of freedom. For example, if the state being represented is a photon of light, $|\psi\rangle$ represents its horizontal and circular polarization angles.

A state represented by $|\psi\rangle$ is a *pure state*. A system can also emit a statistical combination of states. these are known as *mixed states*, and are represented by a more general quantity, a *density matrix* ρ. In a single qubit system, ρ is a 2×2 complex-valued matrix which must be Hermitian and have unit trace to be a valid state. A pure state is a special case of a mixed state, where $\rho = |\psi\rangle\langle\psi|$. Here, the bra-ket notation denotes the generalization of the the outer product to complex vectors, that is, $|\psi\rangle$ multiplied by its conjugate transpose.

4.1.2 Measurements with Probabilistic Outcomes

When a binary quantum state in measured, one observes one of two outcomes with probability depending on the state and measurement made. For example, if we pass light through a polarizing filter, the two outcomes are i) the photon passing through the filter, or ii) it being reflected. The probability of each outcome depends on the polarization state of the light, fully specified by the density matrix ρ, and the polarization angle of the filter.

More precisely, a single outcome of a measurement γ is characterized by another complex valued Hermitian matrix M_γ. The set of matrices for all outcomes of the measurement is called a positive operator-valued measure (POVM), denoted $\mathbb{M} = \{M_\gamma\}_{\gamma=1}^{\Gamma}$, where $\Gamma = 2$ for single qubit systems. A POVM must satisfy $\sum_{\gamma=1}^{\Gamma} M_\gamma = I$. This ensures that the probability of all outcomes sum to one. When the measured system is in state ρ, the probability of each outcome γ is given by Born's rule,

$$P\left(\gamma|\rho, \mathbb{M}\right) = \mathrm{tr}\left[M_\gamma \rho\right] \, . \tag{4.1}$$

As in many learning scenarios, including those in this thesis, uncertainty in quantum tomography arises from two sources. First, there is uncertainty in the state from observing only a finite number of measurement outcomes γ. Second, due to the laws of quantum mechanics, the measurements are stochastic, and some measurement/state combinations will yield more deterministic outcomes than others. With some states the outcome will be highly unpredictable, regardless of the measurement. For example, for a pure single qubit state (ρ is rank 1), there exists a POVM, such that outcome 'one' is always observed. This is achieved by setting M_2 to live in the null space of ρ, and M_1 to its orthogonal complement. Alternatively, if $\rho = I$, then with any measurements that have equal trace, $\mathrm{tr}[M_1] = \mathrm{tr}[M_2]$, the outcome will be uniformly distributed.

A single POVM is insufficient to infer a qubit state. For a single qubit, ρ is characterized by three real parameters and a POVM has two outcomes. Even after observing infinitely many measurements outcomes, one degree of freedom in ρ is unspecified. Therefore, a full tomographic protocol consists of a series of measurements with different POVMs, each in configuration α, denoted $\{\mathbb{M}_\alpha\}$. The probability of observing a dataset consisting of N measurements $\mathcal{D} = \{\gamma_n, \alpha_n\}_{n=1}^{N}$ is a straightforward extension of Born's rule (4.1), accounting for the possible permutations,

$$P(\mathcal{D}|\rho) = \prod_{n=1}^{N} P\left(\gamma_n|\rho, \alpha_n\right) = \prod_{\alpha} n_\alpha! \prod_{\gamma=1}^{\Gamma} \frac{1}{n_{\alpha\gamma}!} \mathrm{tr}[M_{\alpha\gamma}\rho]^{n_{\alpha\gamma}} \, , \tag{4.2}$$

where n_α is the number of measurements made in configuration α, and $n_{\alpha\gamma}$ is the number of times outcome γ was observed in configuration α.

4.1.3 Infidelity

The main goal of quantum state tomography is to provide an estimate $\hat{\rho}$ for ρ based on the data $\mathcal{D}$ [Paris & Řeháček, 2004]. Methods for computing this estimate are described in Section 4.3. The estimate should be close to the real state in some reasonable sense, therefore various notions of statistical distance between quantum states have been proposed [Bengtsson & Zyczkowski, 2006; Braunstein & Caves, 1994]. One of the most widely used measures of statistical distance is *fidelity*, defined as $F(\rho, \hat{\rho}) = \mathrm{tr}\left[\sqrt{\sqrt{\rho}\hat{\rho}\sqrt{\rho}}\right]^2$. When $\hat{\rho} = \rho$ the fidelity takes its maximum value of one, and when the true and estimated states are orthogonal it equals zero. It is common to work with the *infidelity*, defined as $1 - F(\rho, \hat{\rho})$ The goal of a tomographic protocol is to minimize the infidelity after a fixed number of measurements.

4.1.4 Entanglement and The Curse of Dimensionality

A single pure qubit state is represented by the two dimensional state vector $|\psi\rangle$. In a system with m *separable* qubits, the state vector has dimension $D = 2^m$, and takes the form $|\psi\rangle = |\psi_1\rangle \otimes \ldots \otimes |\psi_m\rangle$. Due to the lack of interaction between the states, the resulting $D \times D$ density matrix for mixed separable states has $3D$ degrees of freedom. Quantum states can also be *entangled*, entangled mixed states can take any valid $D \times D$ density matrix, so have $D^2 - 1$ free variables. With either separable or entangled states, the number of parameters that must be estimated scales exponentially in the number of qubits, hence exponentially many resources are required for reconstruction. In statistics, this difficulty is known as the 'curse of dimensionality' [Bellman, 1961].

This means that, even in systems of modest size, tomography requires many measurements. For example, in a four-qubit system over a week of net experimental time is reported in Amselem & Bourennane [2009]. Therefore, it is crucial to minimize the number of measurements by gathering only the most useful data; this is an active learning task. Note, however, that active learning cannot overcome the fundamental limitation of the curse of dimensionality. However, as we will show, it can provide large practical gains which can substantially reduce experimental time in modest sized systems.

4.2 Current Experimental Designs

Most existing optimal experimental designs in quantum tomography are static, a fixed set of measurements is determined prior to the experiment. It is known that in this setting that mutually unbiased bases (MUBs) yield optimal information gain [Adamson & Steinberg, 2010; Patra, 2007; Wootters & Fields, 1989]. MUBs are a set of POVMs $\{\mathbb{M}_i\}$, such that, $\mathrm{tr}[M_{i\gamma}M_{j\gamma'}] = D^{-1/2}\ \forall i \neq j, \gamma, \gamma'$, informally these can be thought of as an 'orthogonal' measurement set. Research since has focused mainly on proving or disproving the existence of, and implementing MUBs in various dimensions [Adamson & Steinberg, 2010; Raynal *et al.*, 2011; Yan *et al.*, 2010]. Another approach to designing static OEDs is to use the Cramér-Rao bound with maximum likelihood estimation [Kosut *et al.*, 2004; Nunn *et al.*, 2010].

However, with fixed designs the infidelity scales as $1 - F \sim N^{-1/2}$ on average for almost-pure states, which are the interesting regime for most applications. Although smart choices of measurement, such as MUBs, can alter the pre-factor [Bogdanov *et al.*, 2011; de Burgh *et al.*, 2008; Řeháček *et al.*, 2004], the scaling law for large N is unaffected. With adaptive designs, one can hope to beat this limit. In physics, this active learning approach has been referred to as self-learning measurements [Fischer *et al.*, 2000; Hannemann *et al.*, 2002]. However, due to the expensive computations involved, these methods are restricted to two dimensional pure quantum states, or very few measurements. With fast Bayesian inference methods and the BALD framework we can improve learning rates in practical scenarios.

4.3 Bayesian Quantum Tomography

The task in quantum tomography is to infer the unknown state ρ given a number of measurement outcomes. The simplest approach is to invert Born's rule using the matrix pseudo-inverse. However, with finite data, this can result in unphysical density matrices. From Born's rule, we know the data generating process, and so we have access to the appropriate likelihood function (4.2). Therefore, one can perform maximum likelihood estimation of ρ. A well-known drawback of maximum likelihood is that it often yields rank-deficient estimates, and thus assigns zero predictive probability to certain observations [Blume-Kohout, 2010]. This is analogous to statistical 'over-fitting' [Hawkins, 2004].

More recently, Bayesian methods have been proposed because they maintain the uncertainty in ρ [Blume-Kohout, 2010, and refs. therein]. The posterior is computed in the usual manner,

$$p(\rho|\mathcal{D}) \propto P(\mathcal{D}|\rho)p(\rho)\,, \tag{4.3}$$

where $P(\mathcal{D}|\rho)$ is the likelihood given by Born's rule, and $p(\rho)$ is the prior on physical density matrices. Ultimately, when predicting the states being produced by a quantum system we desire a point estimate. For this we may report the posterior mean, called the Bayesian mean estimate (BME) $\hat{\rho} = \mathbb{E}_{p(\rho|\mathcal{D})}[\rho]$. The BME uniquely optimizes any *operational divergence*. These are the quantum equivalent of proper scoring rules [Dawid, 2007] and reward honest estimates of the state [Blume-Kohout & Hayden, 2006].

To perform Bayesian inference, we need to specify a prior over density matrices $p(\rho)$. Typically a non-informative (uniform) prior is selected, unless there is reason to favour particular states *a priori*. Designing appropriate priors is an open area of research [Blume-Kohout, 2010]. We adopt a representation that treats $D \times D$ dimensional state ρ as part of a larger, $D \times K$ dimensional multipartite system.[1] In particular, we put a uniform prior over a $D \times K$ dimensional pure state $|\psi_{D\times K}\rangle$. This representation is easier to work with than with the density matrix ρ because $|\psi\rangle$ has fewer constraints. For example, ρ must be complex conjugate, but the columns of $|\psi\rangle$ can be adjusted independently. The original state is formed by taking the complex outer product $\rho_{D\times D} = |\psi_{D\times K}\rangle\langle\psi_{K\times D}|$ (known as "tracing out" the K ancillary dimensions). For $K \geq D$, this will result in any rank-D state ρ. Larger values of K put more mass on pure states, and smaller values will yield rank-deficient states. Therefore we choose $K = D$ to be maximally uninformative.

As in most Bayesian models, the posterior (4.3) is intractable. Therefore, we use Markov chain Monte Carlo (MCMC) sampling for approximate inference.

4.3.1 Sequential Importance Sampling

To deal with the intractable posterior in (4.3) MCMC approaches have been proposed [Blume-Kohout, 2010, and refs. therein]. However, each time inference is performed, these methods require evaluation of the likelihood given all the data (4.2). This computation has $\mathcal{O}(N)$ cost in the number of different measurement configurations used. This is undesirable for adaptive tomography because: i) If we are fully adaptive, and adjust the measurements after each observation, then N is the total number of observations. ii) Inference must be performed after each measurement; if inference takes longer than the time to produce and measure another state then it may be better to collect more data using a fixed design. Note that (ii) is not always a concern in active learning; for example, when surveying for minerals the measurement cost is financial and one may have a large offline computational budget. To avoid a large inference cost per sample

[1] A multipartite system is one in which there is no *entanglement* between states, see Section 4.1.4.

we use sequential importance sampling (SIS) [Doucet *et al.*, 2001], which has $\mathcal{O}(1)$ cost per iteration.

In SIS one keeps track of S samples, often called particles, ρ_s, and corresponding weights w_s, $(\sum_s w_s = 1)$ which are updated every time a new measurement is made. After n measurements, having observed data $\mathcal{D}_n$, the particles ρ_s and weights $w_s^{(n)}$ constitute an approximation to the posterior,

$$p(\rho|\mathcal{D}_n) \approx \sum_{s=1}^{S} w_s^{(n)} \delta(\rho - \rho_s)\,. \tag{4.4}$$

Using this approximation and Bayes' rule, after observing a new outcome γ_{n+1} in configuration α_{n+1}, the updated posterior is given by

$$\begin{aligned} p(\rho|\alpha_{n+1}, \gamma_{n+1}, \mathcal{D}_n) &= \frac{P(\gamma_{n+1}|\rho, \alpha_{n+1}) p(\rho|\mathcal{D}_n)}{\int P(\gamma_{n+1}|\rho, \alpha_{n+1}) p(\rho|\mathcal{D}_n) d\rho} \\ &\approx \sum_{s=1}^{S} \underbrace{\frac{P(\gamma_{n+1}|\rho_s, \alpha_{n+1}) w_s^{(n)}}{\sum_{r=1}^{S} P(\gamma_{n+1}|\rho_r, \alpha_{n+1}) w_r^{(n)}}}_{w_s^{(n+1)}} \delta(\rho - \rho_s)\,. \end{aligned} \tag{4.5}$$

The new weights $w_s^{(n+1)}$ are the re-normalized product of the current weights $w_s^{(n)}$ and likelihood of the new datapoint $P(\gamma_{n+1}|\rho_s, \alpha_{n+1})$. This update is fast as it only requires computing one term of the full likelihood, thus its complexity is independent of how many configurations have been used before. However, as time progresses, several weights decay towards zero, so the quality of the approximation falls. This issue can be detected and handled by monitoring the effective sample size, defined as $\mathrm{ESS}^{(t)} = \left(\sum_{s=1}^{S} {w_s^{(t)}}^2\right)^{-1}$. When the weights are uniform the ESS takes its maximal value, $\mathrm{ESS}^{(t)} = S$. When the ESS falls too low the particles are resampled using all of the data and given uniform weights again.

4.4 Adaptive Quantum Tomography

We apply Bayesian experimental design to adaptive measurement selection. In quantum tomography the aim is to pick an experimental configuration α, such that after observing the outcome γ, we reduce the entropy of the posterior over the state as much as possible. Therefore, the utility of configuration α is

$$\mathcal{U}(\alpha) = \mathrm{H}\left[p(\rho|\mathcal{D})\right] - \mathbb{E}_{p(\gamma|\alpha,\mathcal{D})}\left[\mathrm{H}\left[p(\rho|\gamma,\alpha,\mathcal{D})\right]\right] . \tag{4.6}$$

If the posterior is not updated, $\mathcal{D} = \emptyset$, Equation (4.6) will select MUBs for its first measurements [Patra, 2007]. We demonstrate this effect in Section 4.5.1. However, the usual difficulties arise. It is hard to estimate the entropy of the, potentially high dimensional, distribution over density matrices from which we only have weighted samples. Furthermore, it is impractical to update all of the weights for each possible configuration α and outcome γ. Therefore, we use BALD (2.10) to rewrite the utility as

$$\mathcal{U}(\alpha) = \mathrm{H}\left[P(\gamma|\alpha,\mathcal{D})\right] - \mathbb{E}_{p(\rho|\mathcal{D})}\left[\mathrm{H}\left[P(\gamma|\alpha,\rho)\right]\right] . \tag{4.7}$$

The predictive distributions are $D-$dimensional multinomials, and so both terms in Equation (4.7) can easily be computed using the sampling approximation to the posterior in Equation (4.4). As demonstrated on a toy model in Section 2.3.3, computing the information gain indirectly from samples of ρ using Equation (4.7) is likely to be much more accurate than estimating $\mathrm{H}[p(\rho|\mathcal{D})]$ directly from the samples.

The inclusion of the second term in (4.7) is particularly important in quantum tomography, because certain measurements will always yield a maximum entropy (uniform) outcome for a particular state. In a one qubit system, a measurement with $\mathrm{tr}[M_\gamma\rho] = 0.5, \ \forall\gamma \in \{1, 2\}$ will always exist. Even after infinitely many measurements using this POVM, ρ can only be constrained to a plane of solutions, but uncertainty sampling would continue to select this measurement.

Information gain has a decision theoretic interpretation because the task involves making predictions about the quantity being actively learnt, the state ρ. Minimizing the entropy of the posterior (4.6) is equivalent to minimization of the Bayes risk if the log loss is used to evaluate probabilistic estimate of the state. In a decision theoretic framework, other loss functions could be used, such as the fidelity [Fischer *et al.*, 2000]. Although this loss is theoretically attractive for quantum tomography, the cost to optimize this loss is very high. Fischer *et al.* [2000] simulate only 60 measurements. In Hannemann *et al.* [2002] experimental designs for all 2^N possible sequences of outcomes are pre-computed, and so the authors are limited to very short experiments ($<$ 20 measurements). With BALD and SIS we can perform more efficient adaptive quantum tomography, and hence run longer experiments. In the next section we demonstrate in simulation that we attain much better estimates of the state than non-adaptive schemes.

4.5 Simulations

We simulate single and two qubit systems. First we demonstrate how BALD behaves in this setting.

4.5.1 Single Qubit Tomography

We first study tomography of single qubits ($D = 2$). Recall that mixed state qubits have three real degrees of freedom, therefore ρ may be represented as a point in a unit ball called the Bloch sphere. For illustration purposes, in our first example we omit the third component, and only infer the two remaining parameters, which lie in a unit (Bloch) disk. For example, this corresponds to determining linear polarization of a photon, assuming that the circular polarization is zero. We allow for arbitrary projective[1] measurements with binary ($\Gamma = 2$) outcomes. These are represented by pairs of antipodal points on the perimeter of the Bloch disk. Geometrically, Born's rule states that the probability of each outcome is proportional to the length of the projection of the state vector onto the corresponding measurement vector. Here $\alpha \in [0, \pi)$ indexes the orientation of the measurement. Figure 4.1 shows the progression of measurement bases chosen by BALD. The first two measurements are mutually unbiased, however, the third measurement is equally biased with respect to both previous bases, demonstrating that using a fixed MUB set is suboptimal in the adaptive setting. Throughout the rest of the experiment the algorithm explores a wide range of measurements.

In the next experiment we compare to random measurements and MUBs, this time inferring all three coordinates in the full Bloch sphere. We average over many runs each with randomly drawn pure states. The performance of the random and adaptive designs is independent of the true state, but this is not the case for MUBs. Figure 4.1(d) shows that the majority of measurements 'point in the direction of' the true state $\bar{\rho}$. Drawing intuition from this, fixed MUBs perform better with certain states. In fact, the best case for MUBs is when $\bar{\rho}$ aligns with one of the measurements, then the rate of convergence of fidelity can be improved from $N^{-1/2}$ to N^{-1} [Mahler *et al.*, 2013]. We denote this scenario MUB-best; note of course that this is not a practical alternative as one does not know $\bar{\rho}$ *a priori* and hence cannot align the MUBs correctly. We denote the opposite case MUB-worst, this occurs when the true state is equally biased with respect to the MUBs.

We average over many runs with randomly drawn pure states, and evaluate the

[1] A projective measurement is analogous to a pure state. Most practical measurement devices give projective measurements.

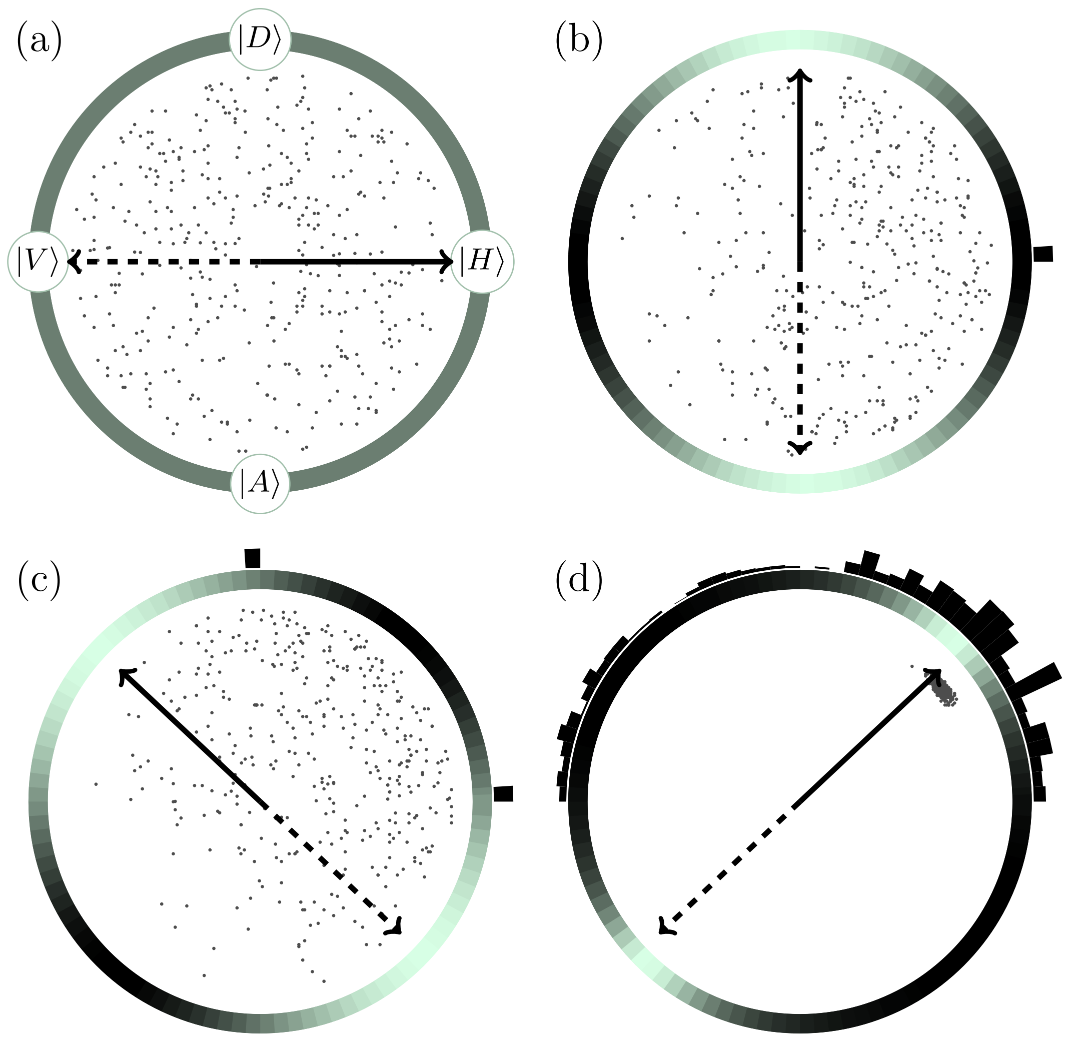

Figure 4.1: Adaptive tomography using BALD. Scatter plots show 400 samples from current posterior. Shaded circles around the ‘Bloch disk’ show the relative value of the objective in Equation (4.7) for different measurement directions (lighter is higher). Pairs of arrows show the most informative next measurement (POVM). Circular histograms show the number of times measurement directions have been used. **(a)** Initially, no observations are made, samples shown are from the uniform prior. All measurements are equally informative, we chose to start with $\{|H\rangle, |V\rangle\}$. **(b)** After one measurement, the posterior is updated, now the best measurement is mutually unbiased w.r.t. the first one. It is now $\{|D\rangle, |A\rangle\}$. **(c)** After two observations, the best measurement is equally biased to the first two bases. **(d)** Posterior after 1000 observations concentrates around true state. The method tries a range of measurements, with a tendency to point towards the solution.

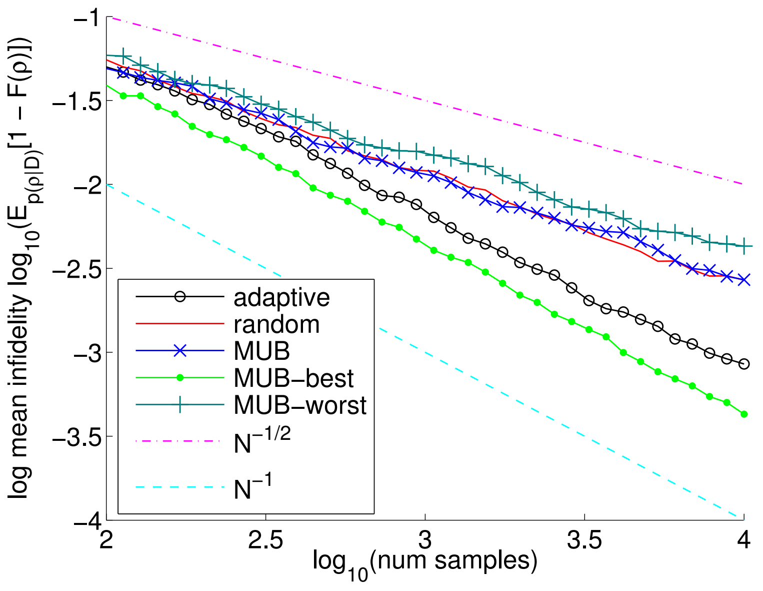

Figure 4.2: Simulated tomography using three measurement selection methods: randomly sampled (red continuous line), MUBs (blue $\times$) and fully adaptive Bayesian tomography, using BALD (black $\circ$). For these methods, the true state is random and pure, the results presented here are the average of 20 independent runs. Functions $1 - F = N^{-1/2}$ (magenta, dash-dotted) and $1 - F = N^{-1}$ (cyan, dashed) are shown for comparison. To account for state-dependence of MUB tomography, we also present its performance for the 'worst' and 'best' true states (dark green $+$, light green $\bullet$, respectively).

performance after each observation using the posterior mean infidelity measured against the true state, $\mathbb{E}_{p(\rho|\mathcal{D})}[1-F(\rho,\bar{\rho})]$. The Bayesian mean is a fairer score than the infidelity of a point estimate, such as the BME $1 - F(\mathbb{E}_{p(\rho|\mathcal{D}_n)}[\rho], \bar{\rho})$. This is because the BME can be correct even if we have no knowledge about the system, for example, the BME of a uniform distribution over states will achieve zero infidelity for a completely mixed state. The posterior mean infidelity rewards distributions that are both centred in the correct location and have low variance.

Figure 4.2 shows the mean infidelity versus the number of measurements made using each algorithm. We fit a power law, $1 - F \propto N^a$ to the data. Random tomography yields $a = -0.66 \pm 0.03$, which is in reasonable correspondence with the expected asymptotic scaling $N^{-1/2}$ for fixed designs [Adamson & Steinberg, 2010]. However, adaptive tomography performs close to the N^{-1} level with average $a = -0.90 \pm 0.03$. In its most favourable scenario, MUBs also perform close to the N^{-1} rate and have a small multiplicative constant improvement over the adaptive scheme. In practice,

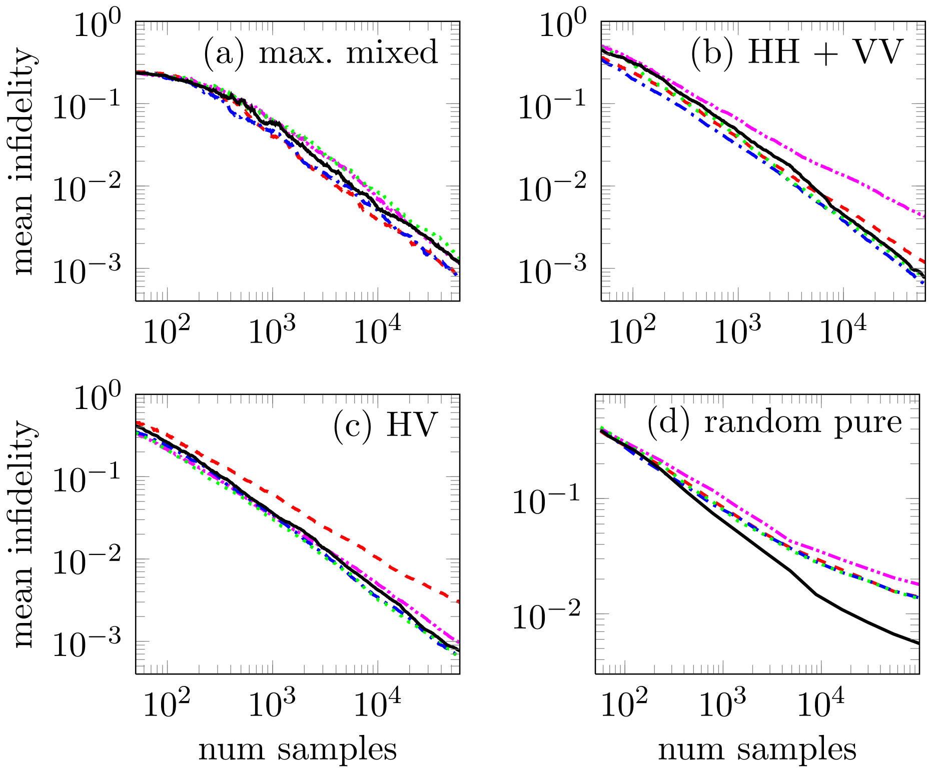

Figure 4.3: Two qubit tomography with uniform selection from MUBs (- - -), SSQT bases (-··-··) and Bayesian adaptive sampling from the same set of MUBs (-·-·-), SSQT bases (······) or a more flexible set of 81 separable bases (——). Panels (a)-(c) are the same as those in Adamson & Steinberg [2010], (d) shows average results over 20 randomly generated entangled pure states.

the optimal MUBs are unknown *a priori*. In the case of arbitrarily chosen MUBs we observe that on average the rate is nearer $N^{-1/2}$: $a = -0.64 \pm 0.05$.

4.5.2 Separable and MUB Tomography of Two Qubits

In multipartite systems, such as m-qubit registers, there are two fundamentally different classes of measurements one can apply: separable or entangling. Separable tomographic experiments are straightforward and cheap to implement, while entangling measurements are statistically more powerful. Notably, entanglement is required to implement MUBs [Adamson & Steinberg, 2010]. To investigate this trade-off in the light of adaptive tomography, we reproduce and extend the experiments in Adamson & Steinberg [2010]. We compare five algorithms. Two non-adaptive methods – uniformly selecting from a set of MUBs, and a set of Standard Separable Quantum Tomographic (SSQT) bases. Three methods that use BALD – adaptively selecting from (i) MUBs, (ii) SSQT bases, and (iii) a larger, over-complete, set of 81 separable bases.

We simulate the tomography of four different states, results are presented in Figure 4.3. Figure 4.3 (a) shows results with a maximally mixed state, $\bar{\rho} = I$. This corresponds to a single qubit state that lies in the centre of the Bloch sphere. In this case all measurements yield uniformly distributed outcomes. As expected, with a maximally mixed state the choice of measurement strategy has little effect. Figure 4.3 (b) shows an entangled state $(|HH\rangle+|VV\rangle)/\sqrt{2}$. MUBs outperform SSQT when uniformly sampled, but by allowing for adaptivity we can close the performance gap. Figure 4.3 (c) shows a separable state $|HV\rangle$. In this case SSQT outperforms MUBs because separable measurements align better with the state than the more flexible entangled MUBs. Again, when choosing measurements adaptively both measurement bases perform equally well. Figure 4.3 (d) shows a random pure state. Here, BALD with the flexible set of separable measurements yields a large improvement in performance. In this case, adaptive tomography with an over-complete set of *separable* bases only needed 10^4 measurements to achieve $\approx 98.7\%$ mean fidelity, which required 10^5 measurements using MUBs.

In summary, all substantial differences between MUBs and standard separable tomography (SSQT) vanish when we allow for adaptivity (Figure 4.3 a–c). Furthermore, for random pure states, we are able to realize a ten-fold improvement over MUBs when using adaptive separable measurements (Figure 4.3 d). The results indicate that allowing for adaptivity with an imperfect, but flexible set of measurements offers greater advantages than using a fixed set of MUBs.

We have demonstrated in simulation the potential for BALD to greatly reduce the number of measurements relative to the optimal, non-adaptive, MUB designs. In two-qubit systems, a ten-fold reduction can be achieved. In a one-qubit systems, we are able to beat the $N^{-1/2}$ scaling of infidelity, observing rates closer to N^{-1}. However, can such gains be made in a real-world, noisy environment? In the next section we present laboratory experiments on single state polarization qubits.

4.6 Laboratory Experiment

We investigate the benefits of adaptive Bayesian quantum tomography in laboratory experiments on a single photon light source. In a real world setup there are additional sources of noise. After presenting the experimental setup we describe how to model this noise and show our findings.

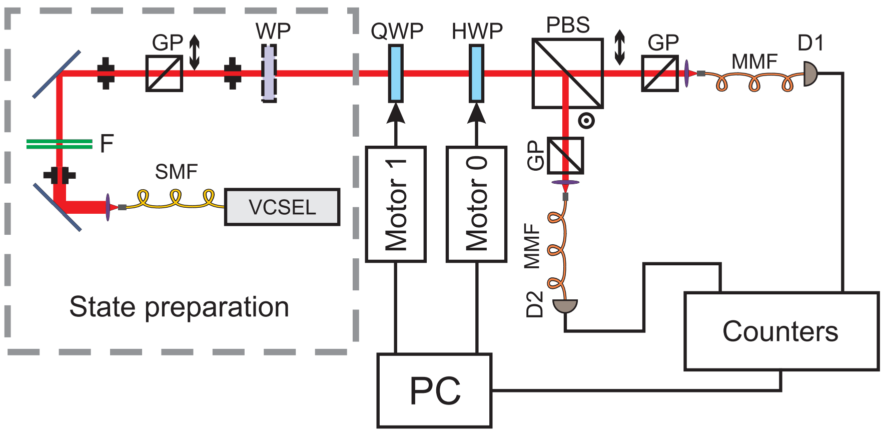

Figure 4.4: Experimental setup. An attenuated laser is used as a source, the polarization state is prepared by a custom waveplate, and analyzed by a sequence of a quarter- and half-wave plates, followed by a polarizing beam-splitter and two single-photon counters. Waveplates are rotated by electronically controlled step-motor drivers to allow for adaptivity.

4.6.1 Setup and Apparatus

We perform polarization tomography on single photons of light emitted from an attenuated laser. A measurement is made by passing the light through a filter called the *polarization beam splitter* (PBS). Depending on the state, the photon will pass though the PBS or be reflected. Detectors, called *single-photon counting modules* (SPCMs), count the photons that follow each path. Recall that in single qubit tomography, a projective measurement is characterized by two degrees of freedom, the polar and azimuth angles in the Bloch sphere. The different measurements are achieved by rotating the photon twice, using a quarter-wave plate (QWP) and a half-wave plate (HWP). Their orientation is set using motors, and during adaptive tomography, the wave plates are rotated to achieve the optimal measurements.

Figure 4.4 depicts the setup. In detail, we use a CW 850 nm vertical-cavity surface-emitting laser (VCSEL) diode laser coupled to a single-mode fibre as the light source. The radiation is attenuated to the single-photon level by a set of neutral density filters (F) and additionally spatially filtered with small iris apertures. The input polarization state is defined by a Glan-Taylor prism GP with high extinction ratio (more than 6000:1), the prism transmits horizontally polarized light, which may be transformed to an arbitrary state with a proper choice of a quartz wave plate (WP).

The measurement scheme consists of an effective zero-order QWP and a HWP. The plates are rotated by step-motor-driven stages, with minimal angular step of 0.1°. The zero position is controlled by a Hall sensor providing uncertainty of $0.2°$. We clean up the polarization states in the output channels of the PBS cube with two additional Glan-Taylor prisms to ensure high extinction ratio. Effectively this is equivalent to introducing some losses in the *ideal* PBS cube without altering the output polarization states. In each channel photons are coupled to multi-mode fibres (MMF) and detected by single photon counting modules D1 and D2 (Perkin-Elmer). Electronic pulses from SPCM's are sent to a counter built in-house which may operate in two ways - count for a fixed period of time or until a specified number of counts is reached.

4.6.2 Modelling Experimental Imperfections

In practice quantum tomography is subject to experimental noise. This noise is not modelled in the likelihood function given by Born's rule (4.2). In our experiment we identified two major additional sources of noise: dark counts with detector-specific rates and attenuation in both channels due to detector inefficiency and losses at the optical elements.

Dark and Background Counts

Dark and background counts are false positive observations that are detected even when there is no photon present. A popular approach to account for dark counts is to model the observed state as a linear mixture of the true state and the maximally mixed state [Lvovsky *et al.*, 2001]. With this approach one can describe certain simple sources of noise, such as dark counts being generated at each detector with equal rates. We take a more flexible approach and model the noise directly in the likelihood function.

We assume that photons produced at the laser source and dark counts are all generated independently. In particular, we assume that the production of photons by the laser source, and generation of dark counts by the detectors can be modelled using independent homogeneous Poisson processes with rate parameters λ_s for the source and λ_d^γ for each detector γ. We assume that the rates of the Poisson processes remain constant over time. This homogeneity assumption is likely to be violated due to parameter drift in the apparatus, but by re-calibrating the system periodically we ensure that the drift is small. Audenaert & Scheel [2009] consider more general noise scenarios, but the resulting computations are more complex and may require numerical methods. The rates $\Lambda = \{\lambda_s, \{\lambda_d^\gamma\}_{\gamma=1}^{\Gamma}\}$ are estimated from prior experimentation.

The new likelihood function follows directly from these assumptions and Born's rule. The total rate of photons (including dark counts) entering the system follows a

Poisson process with rate $\lambda_s + \sum_{\gamma=1}^{\Gamma} \lambda_d^\gamma$. Therefore the probabilities that a detection is from the source or a dark count are given by

$$P(\text{source}|\Lambda) = \frac{\lambda_s}{\lambda_s + \sum_{\gamma=1}^{\Gamma} \lambda_d^\gamma}, \qquad P(\text{dark}|\Lambda) = \frac{\sum_{\gamma=1}^{\Gamma} \lambda_d^\gamma}{\lambda_s + \sum_{\gamma=1}^{\Gamma} \lambda_d^\gamma}.$$

The likelihood follows from Born's rule (4.1),

$$\begin{aligned} P(\gamma|\rho,\alpha,\Lambda) &= P(\gamma|\text{source},\rho,\alpha)P(\text{source}|\Lambda) + P(\gamma|\text{dark})P(\text{dark}|\Lambda) \\ &= \text{tr}[M_{\alpha\gamma}\rho]\frac{\lambda_s}{\lambda_s + \sum_{\gamma=1}^{\Gamma} \lambda_d^\gamma} + \frac{\lambda_d^\gamma}{\sum_{\gamma=1}^{\Gamma} \lambda_d^\gamma}\frac{\sum_{\gamma=1}^{\Gamma} \lambda_d^\gamma}{\lambda_s + \sum_{\gamma=1}^{\Gamma} \lambda_d^\gamma} \\ &= \frac{\text{tr}[M_{\alpha\gamma}\rho]\lambda_s + \lambda_d^\gamma}{\lambda_s + \sum_\gamma \lambda_d^\gamma}. \end{aligned} \tag{4.8}$$

When there are no dark counts, $\lambda_d^\gamma = 0,\ \forall\gamma$, then Equation (4.8) reduces to Born's rule (4.1).

Channel Inefficiency

As well as dark counts, the detectors can produce false negatives. Photons may also be reflected at the optical elements, such as the wave-plates and PBS. Furthermore, the GT prisms may have different attenuation factors. To model these channel-specific losses, each detector is assigned an efficiency $\eta_\gamma \in [0,1]$. the probability of a photon being 'lost' in the channel ending in detector γ is given by $1-\eta_\gamma$. Therefore, the probability of observing a measurement at detector γ is proportional to $\text{tr}[M_{\alpha\gamma}\rho]\eta_\gamma$. The likelihood is straightforward,

$$P(\gamma|\rho,\alpha,\eta_1,\ldots,\eta_\gamma) = \frac{\text{tr}[M_{\alpha\gamma}\rho]\eta_\gamma}{\sum_\gamma \text{tr}[M_{\alpha\gamma}\rho]\eta_\gamma}. \tag{4.9}$$

In Equations (4.8) and (4.9), both the numerator and denominator contain only linear terms in the additional parameters $(\Lambda, \{\eta_\gamma\})$. Therefore, one only requires estimates of the ratio of the dark count rates to the source rate $\lambda_d^\gamma/\lambda_s$, and, for single-qubit tomography, the ratio of the efficiencies of the two channels η_1/η_2. For this reason we do not need to measure the absolute attenuations in the GT prisms.

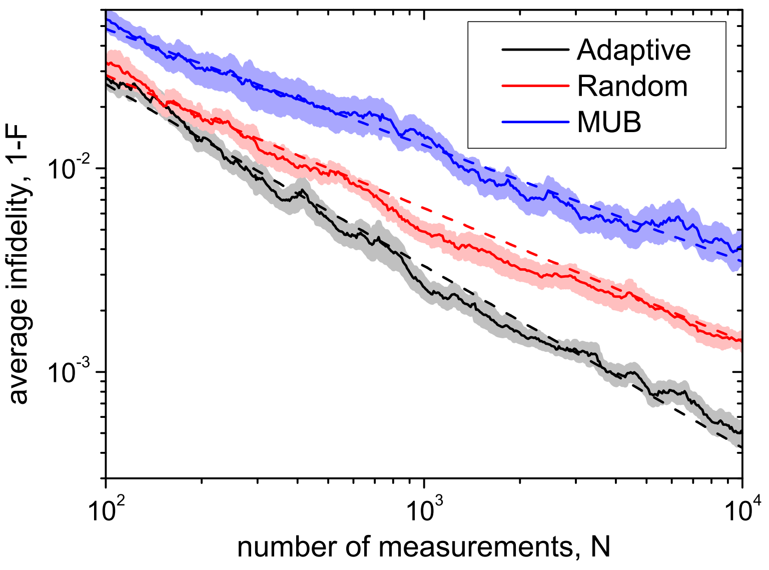

Figure 4.5: Experimental results: mean infidelity $1 - \mathbb{E}_{p(\rho|\mathcal{D})}F(\rho, \bar{\rho})$ with true state $\bar{\rho}$ for random measurements – red (middle) line, adaptive measurements – black (lower) line, and measurements in MUBs – blue (upper) line. We average over 10 experimental runs, shaded areas show the standard deviation. Dashed straight lines indicate the power law fits.

Block Sampling

The time taken to rotate the WPs into position is longer than rate of generation of the states or the time required to run SIS or BALD. Therefore, we adjust the apparatus after blocks of measurements that increase in size with amount of data collected as $\lceil N/100 \rceil$. In simulation we found no statistical difference between this strategy and adjusting after every measurement.

4.6.3 Results

In a real world application of tomography the true state is unknown, so we estimate the prepared state by averaging over many runs of adaptive protocol. Figure 4.5 gives the mean infidelity to the (estimated) true state. Power law fits give $a = -0.64 \pm 0.02$ and $a = -0.60 \pm 0.05$ for random and MUB protocols, respectively, while adaptive strategy yields $a = -0.92 \pm 0.03$.

Within the errors bands, the scaling laws obtained in the experiments agree with the simulations in Section 4.5.1. This demonstrates that we were able to realize in practice the advantages of using BALD for adaptive Bayesian tomography. Our model does not take into account systematic errors, such as imprecise waveplate rotations. However, for the infidelities values that we reached, $10^{-4} - 10^{-3}$, we did not observe deviations from the expected behaviour and could not identify the influence of systematic errors.

4.7 Conclusions

The ability to characterize states in a quantum system in reasonable time is important for the practical application of quantum technology. We have presented the use of Bayesian methods and the BALD framework for adaptive quantum tomography. This adaptive approach outperforms MUBs, widely accepted as the optimal fixed measurements. In both simulation and laboratory experiments we can achieve much faster convergence rates in single qubit tomography with nearly pure states than those achieved by MUBs or a random design. We approach the theoretical limit for any tomographic protocol, N^{-1}, and so any further improvements can only effect the multiplicative constant.

Moreover, the adaptive framework applies regardless of dimensionality, and can be applied to spaces where MUBs do not even exist [Patra, 2007; Raynal *et al.*, 2011]. In simulation we achieved up to a 10-fold reduction in the number of measurements required in two-qubit systems using adaptive separable measurements. This motivates a shift in experimental focus from implementing complex entangling measurements to quickly reconfigurable simpler measurements.

Although we have demonstrated a substantial leap forward in terms of empirical performance, it is important to keep in mind that adaptive tomography does not resolve the curse of dimensionality; the size of the parameter space still scales exponentially with the number of qubits. To achieve feasible tomography in higher dimensional spaces, it is necessary to restrict the search space. Bayesian methods could be extremely useful here, as the prior can be used to impose the desired assumptions about the state.

Chapter 5

Stochastic Inference for Large Binary Matrices

Chapters 5, 6 and 7 focus upon modelling of matrices, a common data-type in machine learning, engineering and science. Chapters 6, 7 and 8 also draw on techniques developed in the previous chapters to do active learning with matrix data in various scenarios.

5.1 Introduction to Probabilistic Matrix Modelling

Many datasets take the form of a matrix or table. Examples include: user-by-product rating or purchase matrices; sample-by-gene expression bioinformatic matrices; person-by-person social network graphs; and user by response questionnaires. Recently, matrix data received attention in machine learning due to the widely contested $1M Netflix Challenge [Bennett & Lanning, 2007]. In this challenge participants were given a large user-by-movie ratings matrix with very many missing entries and were required to predict unobserved ratings. This is one of the most common tasks with matrix data; to predict or rank missing elements in highly sparse matrices. Sometimes the rows and columns have covariates, and predictions can be improved using supervised learning with these features (this is investigated in Chapter 6). When there are no features, patterns in elements of the matrix must be exploited directly. The task of discovering the structure of a matrix to make predictions is known as *collaborative filtering*.

A number of approaches have been developed for matrix modelling, such as clustering models [Ungar & Foster, 1998], mixture models [Hofmann, 2004], neighbourhood methods [Sarwar *et al.*, 2001] and matrix factorizations (MF) [Koren *et al.*, 2009; Srebro *et al.*, 2005]. MF techniques are probably the most successful due to their simplicity and often superior predictive performance. They were central to many of the best single models in the Netflix challenge [Bell *et al.*, 2010]. Recently, probabilistic methods for matrix factorization have been developed. These have become popular because i) they are robust to overfitting [Salakhutdinov & Mnih, 2008], ii) they can produce estimates of uncertainty in their predictions, and iii) they can be adapted to different data-types, such as continuous matrix entries [Salakhutdinov & Mnih, 2008], binary data [Rendle *et al.*, 2009] and ordinal valued data [Stern *et al.*, 2009].

However, a number of challenges arise in probabilistic matrix modelling. The main challenge is to scale inference algorithms to handle large modern datasets. Further open issues include extending models to include covariates or to use other forms of feedback, such as binary preferences, being robust when there is very little data available, and collecting entries in an active manner. In the following chapters we tackle these challenges. First we address the task of scaling inference with large binary matrices for which we develop stochastic inference techniques.

5.2 Limitations of Batch Inference

Probabilistic models for matrix factorization assume that a partially observed data matrix $\mathbf{X}$ is well approximated by a low rank matrix $\mathbf{U}\mathbf{V}^{\mathrm{T}}$. Normally $\mathbf{X}$ is very sparse, with most elements being unobserved. The objective is then to find the two matrices $\mathbf{U}$ and $\mathbf{V}$ given $\mathbf{X}$. Probabilistic methods treat each element in $\mathbf{U}$ and $\mathbf{V}$ as model parameters to be inferred. Fast approximate inference is usually implemented using variational Bayes [Lim & Teh, 2007; Nakajima *et al.*, 2010; Raiko *et al.*, 2007]. The resulting techniques are computationally efficient because their cost depends only on the number of entries observed in $\mathbf{X}$, which is usually low, and not on the size of $\mathbf{X}$, which can be large.

Many real-world datasets are binary, that is, the entries of $\mathbf{X}$ take values in $\{0, 1\}$. Some common examples of sources of binary data include include market basket data [Mild & Reutterer, 2003], click-stream data [Joachims, 2002], network data [Airoldi *et al.*, 2008] or file dependencies in complex software systems [Hu *et al.*, 2010]. However, for binary matrices, $\mathbf{X}$ is usually fully observed, entries take either zero or one and there is no 'unobserved' value. For example, in a news portal, we know which articles a user has visited, and which they have not.[1] With fully observed matrices the aforementioned probabilistic approaches to solving the MF problem are infeasible in practice because they require looking at the entire matrix before making any adjustments to the parameters.

More specifically, current popular inference methods are based on batch variational algorithms that require processing all the entries in $\mathbf{X}$ before producing even a single update to the variational parameters. An alternative is to use a likelihood function for continuous data instead of one for binary data [Nakajima *et al.*, 2010]. In this case, an analytic solution exists which scales with the number of ones in $\mathbf{X}$. However, this solution is restricted to zero-mean spherical priors on $\mathbf{U}$ and $\mathbf{V}$, and homoscedastic Gaussian likelihood functions for $\mathbf{X}$. In our experiments, we find that these restrictions lead to poor predictions when $\mathbf{X}$ is binary.

We address scalable learning with probabilistic MF models that are flexible enough to produce state-of-the-art predictions on large binary matrices. To meet this challenge we propose an algorithm based upon stochastic inference. Stochastic methods have the advantage that, with large datasets, they can make reasonably accurate predictions before batch algorithms generate a single parameter update. The algorithm is based on a recent technique called stochastic variational inference (SVI) [Hoffman *et al.*, 2013]. Existing implementations of SVI do not extend to MF models directly, which present specific challenges that are not encountered in models currently addressed by this inference algorithm, such as topic models. This is because in MF we subsample individual matrix entries instead of complete data instances, such as an entire document in a topic model. In standard SVI all the variational parameters are updated each time a data instance is subsampled. With matrices, we have different parameters for each row and column in $\mathbf{X}$ and each time we subsample a matrix entry, we update only the variational parameters associated with the row and column of that entry. This makes the data sub-sampling strategy more important because it determines which parameters are updated and how often. For this reason, we develop a data subsampling strategy with different sampling probabilities across the rows and columns of $\mathbf{X}$. This method significantly outperforms standard uniform subsampling.

A second challenge for SVI presented by MF is that parameter estimates in MF models often exhibit heavy-tailed empirical distributions [Lakshminarayanan *et al.*, 2011]. These heavy tails can significantly reduce the convergence speed of stochastic

[1] In some domains it may be ambiguous whether a 'zero' corresponds to a negative observation or lack of observation. In these ambiguous cases it is advantageous to treat the zeros as observed, since if they were unobserved the maximum likelihood solution would predict ones everywhere. We return to this point in Section 5.6.

algorithms. A solution is to use minibatches to reduce the effect of outliers in the noisy estimates of the gradients. However, the best minibatch size S can be dataset-dependent. To avoid having to hand-tune S to each dataset, which is common practice [Orr & Müller, 1998], we propose a method that adaptively selects the value of S online.

With this approach we scale probabilistic MF methods to large binary matrices whilst maintaining strong empirical performance. Experimentally, our algorithm demonstrates faster convergence than batch alternatives [Raiko *et al.*, 2007] and yields more accurate solutions than existing scalable variational methods [Nakajima *et al.*, 2010; Paquet & Koenigstein, 2013; Seeger & Bouchard, 2012]. The focus of this chapter is on improving the state-of-the-art in probabilistic MF methods, but we also compare to one of the best alternative non-probabilistic techniques for MF [Rendle *et al.*, 2009]. Encouragingly, our method performs favourably. We can improve upon the state-of-the-art because:

1. We handle fully observed matrices and learn by subsampling individual matrix entries.

2. We use a likelihood function for binary data and not for continuous data.

3. Flexible priors and additional bias parameters may be incorporated easily with our method.

4. We use improved subsampling strategies and automatically select the appropriate minibatch size for the data.

The chapter is organized as follows. In the next section we introduce a model for binary matrices and present our core stochastic variational inference algorithm. We then describe the extensions including our sampling strategy in Section 5.4.5 and our automatic minibatch size selection strategy in Section 5.4.6. Related literature is discussed are in Section 5.4.8 and experiments with a number of real world binary matrices in Section 5.5. Section 5.6 finishes with a summary, discussion and extensions.

5.3 A Probabilistic Model for Binary Matrices

We describe a probabilistic model for the generation of an $L \times M$ sparse binary matrix $\mathbf{X}$. The assumption made by MF methods is that the rows and columns are the result of a linear combination of a small number of unobserved latent factors. Following this, we assume that there are two low rank matrices or latent factors $\mathbf{U} \in \mathbb{R}^{L \times D}$ and

$\mathbf{V} \in \mathbb{R}^{M\times D}$, where $D \ll \min(L, M)$, such that $\mathbf{X}$ is obtained as a function of $\mathbf{U}$, $\mathbf{V}$ and some additive noise. In particular, we assume that

$$\mathbf{X} = \Theta[\mathbf{U}\mathbf{V}^{\mathrm{T}} + z + \mathbf{E}]\,, \tag{5.1}$$

where $\Theta[\cdot]$ applies the Heaviside step function to the entries of a matrix, $z \in \mathbb{R}$ is a global bias parameter and $\mathbf{E}$ is an $L \times M$ additive noise matrix whose entries e_{ij} are i.i.d. with cumulative distribution function given by the logistic sigmoid $\sigma(x) = 1/[1 + \exp(-x)]$. This results is the likelihood

$$\begin{aligned} p(\mathbf{X}|\mathbf{U},\mathbf{V},z) &= \prod_{i=1}^{L}\prod_{j=1}^{M} p(x_{i,j}|\mathbf{u}_i,\mathbf{v}_j,z) \\ &= \prod_{i=1}^{L}\prod_{j=1}^{M} \left[\sigma(\mathbf{u}_i\mathbf{v}_j^{\top} + z)^{x_{i,j}} \cdot \sigma(-\mathbf{u}_i\mathbf{v}_j^{\top} - z)^{1-x_{i,j}}\right]\,, \end{aligned} \tag{5.2}$$

where $\mathbf{u}_i$ and $\mathbf{v}_j$ are the i-th and j-th rows of $\mathbf{U}$ and $\mathbf{V}$, respectively. We specify fully factorized Gaussian priors for $\mathbf{U}$, $\mathbf{V}$ and z,

$$\begin{aligned} p(\mathbf{U}) &= \prod_{i=1}^{L}\prod_{d=1}^{D} \mathcal{N}(u_{i,d}; \bar{u}^0_{i,d}, \tilde{u}^0_{i,d})\,, \\ p(\mathbf{V}) &= \prod_{j=1}^{M}\prod_{d=1}^{D} \mathcal{N}(v_{j,d}; \bar{v}^0_{j,d}, \tilde{v}^0_{j,d})\,, \\ p(z) &= \mathcal{N}(z; \bar{z}^0, \tilde{z}^0)\,. \end{aligned}$$

In all of our experiments we used priors with zero-mean and unit variance.

We also incorporate a local bias to each row and column. To do this, column D in $\mathbf{V}$ may contain the biases for the columns. In this case, $\bar{u}^0_{i,D} = 1$ and $\tilde{u}^0_{i,D} = \epsilon$, where ϵ is a small positive constant. Similarly, column $D-1$ in $\mathbf{U}$ may contain the biases for the rows and $\bar{v}^0_{j,D-1} = 1$ and $\tilde{v}^0_{j,D-1} = \epsilon$. The posterior distribution for $\mathbf{U}$, $\mathbf{V}$ and z is computed as

$$p(\mathbf{U},\mathbf{V},z|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{U},\mathbf{V},z)p(\mathbf{U})p(\mathbf{V})p(z)}{p(\mathbf{X})}\,. \tag{5.3}$$

As given in the generative process in (5.1), we assume that the observed matrix $\mathbf{X}$ is corrupted by noise $\mathbf{E}$, and we would like to reason about the noise-free latent matrix. To do this we make predictions about the possible value $x^{\star}_{i,j}$ that an entry $x_{i,j}$ in $\mathbf{X}$

could have taken during the generation of $\mathbf{X}$ from $\mathbf{U}$, $\mathbf{V}$, b and $\mathbf{E}$. For this, we use

$$p(x^{\star}_{i,j}|\mathbf{X}) = \int \left[\sigma(\mathbf{u}_i\mathbf{v}_j^\top + z)^{x^{\star}_{i,j}} \cdot \sigma(-\mathbf{u}_i\mathbf{v}_j^\top - z)^{1-x^{\star}_{i,j}}\right] p(\mathbf{U},\mathbf{V},z|\mathbf{X})\, d\mathbf{U}d\mathbf{V}dz\,. \tag{5.4}$$

The computation of Equations (5.3) and (5.4) is infeasible in practice and we have to use approximations. In the following section we present variational Bayes to compute approximations to (5.3) and (5.4).

5.4 Stochastic Variational Inference for Binary Matrices

5.4.1 Primer on Variational Bayes

Variational Bayes (VB) is a general purpose inference algorithm that approximates an exact posterior over some parameter θ, $p(\theta)$ with a simpler, tractable distribution $q(\theta)$ [Jordan *et al.*, 1998]. The parameters that govern q are known as variational parameters. These are optimized by maximizing the following lower bound on the marginal likelihood given some data $\mathbf{X}$,

$$\begin{aligned}
\log p(\mathbf{X}) &= \log \int p(\mathbf{X},\theta)d\theta \\
&= \log \int q(\theta)\frac{p(\mathbf{X},\theta)}{q(\theta)}d\theta \\
&\geq \int q(\theta)\log\left[\frac{p(\mathbf{X},\theta)}{q(\theta)}\right]d\theta \\
&= \mathbb{E}_{q(\theta)}[\log p(\mathbf{X},\theta)] - \mathbb{E}_{q(\theta)}[\log q(\theta)] && (5.5)\\
&= -\mathrm{KL}[q(\theta)||p(\theta|\mathbf{X})] + \log p(\mathbf{X})\,, && (5.6)
\end{aligned}$$

where the step from lines 2 to 3 follows from Jensen's inequality. Equation (5.5) is known as the evidence lower bound, or ELBO. This lower bound holds for all q, and the objective of VB is to maximize this lower bound with respect to q. The ELBO can be re-written as Equation (5.6). $\log p(\mathbf{X})$ is independent of q, therefore maximizing the ELBO is equivalent to minimizing the KL between the approximation $q(\theta)$ and the true posterior $p(\theta|\mathbf{X})$. Equation (5.6) shows that if the true posterior is contained in the same the family of distributions as q, then the lower bound will be maximized by recovering the true posterior, that is setting $q(\theta) = p(\theta|\mathbf{X})$. In summary, variational Bayes turns inference, an integration problem, into an optimization task for which many techniques, such as stochastic methods, have been developed.

5.4.2 VB for Binary Matrices

VB for binary matrices proceeds by approximating the posterior in (5.3) with the simpler distribution $q(\mathbf{U}, \mathbf{V}, z)$. We choose $q(\mathbf{U}, \mathbf{V}, z)$ to be a fully factorized Gaussian,

$$q(\mathbf{U}, \mathbf{V}, z) = \left[\prod_{i=1}^{L}\prod_{d=1}^{D} \mathcal{N}(u_{i,d}; \bar{u}_{i,d}, \tilde{u}_{i,d})\right]\left[\prod_{j=1}^{M}\prod_{d=1}^{D} \mathcal{N}(v_{j,d}; \bar{v}_{j,d}, \tilde{v}_{j,d})\right]\mathcal{N}(z; \bar{z}, \tilde{z})\,, \tag{5.7}$$

where $\Phi = \{\{\{\bar{u}_{i,d}, \tilde{u}_{i,d},\}_{i=1}^{L}, \{\bar{v}_{j,d}, \tilde{v}_{j,d}\}_{j=1}^{M}\}_{d=1}^{D}, \bar{z}, \tilde{z}\}$ are the variational parameters that are adjusted so that $q(\mathbf{U}, \mathbf{V}, z)$ is as similar as possible to $p(\mathbf{U}, \mathbf{V}, z|\mathbf{X})$ by minimizing the KL divergence between Equations (5.7) and (5.3), or maximizing the ELBO

$$\mathcal{L}(\Phi) = \mathbb{E}_q\left[\log p(\mathbf{U}, \mathbf{V}, z, \mathbf{X})\right] - \mathbb{E}_q\left[\log q(\mathbf{U}, \mathbf{V}, z)\right]\,. \tag{5.8}$$

Once $q(\mathbf{U}, \mathbf{V}, z)$ has been adjusted, we approximate (5.4) by first approximating the posterior distribution of $\mathbf{u}_i\mathbf{v}_j^{\mathrm{T}} + z$ by a Gaussian with mean $\mu_{i,j} = \sum_d \bar{u}_{i,d}\bar{v}_{j,d} + \bar{z}$, and variance $s_{i,j}^2 = \sum_d \bar{u}_{i,d}^2\tilde{v}_{j,d} + \tilde{u}_{i,d}\bar{v}_{j,d}^2 + \tilde{u}_{i,d}^2\tilde{v}_{j,d} + \tilde{z}$. After this, we approximate the logistic function with a rescaled probit function that has the same gradient at the origin as the logistic function $\sigma(\cdot)$ [MacKay, 1992a]. We finally obtain

$$\begin{aligned} p(x_{i,j}^{\star}|\mathbf{X}) &\approx \int \sigma[(2x_{i,j}^{\star} - 1)a]\mathcal{N}(a; \mu_{i,j}, s_{i,j}^2)\, da \\ &\approx \sigma[\varphi(s_{i,j}^2)\mu_{i,j}(2x_{i,j}^{\star} - 1)]\,, \end{aligned} \tag{5.9}$$

where $\varphi(x) = (1 + \pi x/8)^{-1/2}$.

However, in (5.8), $\mathbb{E}_q\left[\log p(\mathbf{U}, \mathbf{V}, z, \mathbf{X})\right]$ cannot be evaluated analytically. To address this, we use the Gaussian lower bound to the logistic function described in Jaakkola & Jordan [1997]. We choose this approximation because it yields Gaussian complete conditional distributions. A complete conditional is the conditional distribution of a variable given all of the other variables and observations. Models with conjugate complete conditionals admit tractable update equations with variational Bayes [Ghahramani & Beal, 2000] and allow us to use *natural gradients*, which improve convergence [Hoffman *et al.*, 2013]. We lower bound $\sigma(a)^{x_{i,j}} \cdot \sigma(-a)^{1-x_{i,j}}$ in (5.2) with

$$\tau(a, \xi) = e^{a x_{i,j}}\sigma(\xi)e^{-\frac{a+\xi}{2} + \lambda(\xi)(a^2 - \xi^2)}\,, \tag{5.10}$$

where $\lambda(\xi) = (0.5 - \sigma(\xi))/(2\xi)$ and ξ is adjusted to maximize the lower bound, making it tight at $a = \pm\xi$. When we replace each $p(x_{i,j}|\mathbf{u}_i, \mathbf{v}_j, z)$ in (5.2) with an instantiation of (5.10) that includes its own parameter $\xi_{i,j}$, we obtain a new lower bound of the form

$$\mathcal{L}'(\Phi, \Xi) = \sum_{i=1}^{L} \sum_{j=1}^{M} \alpha_{i,j} + \sum_{i=1}^{L} \sum_{d=1}^{D} \beta_{i,d} + \sum_{j=1}^{M} \sum_{d=1}^{D} \gamma_{j,d} + \kappa \,, \tag{5.11}$$

where $\Xi = \{\{\{\xi_{i,j}\}_{i=1}^{L}\}_{j=1}^{M}\}$ is the collection of all the additional variational parameters associated with the lower bound to the logistic function for each entry in $\mathbf{X}$, and

$$\begin{aligned}
\alpha_{i,j} =& \log \sigma(\xi_{i,j}) - \frac{\mu_{i,j}(1-2x_{i,j}) + \xi_{i,j}}{2} + \lambda(\xi_{i,j})(\mu_{i,j}^2 + s_{i,j}^2 - \xi_{i,j}^2)\,,\\
\beta_{i,d} =& \rho(\tilde{u}_{i,d}, \tilde{u}_{i,d}^0, \bar{u}_{i,d}, \bar{u}_{i,d}^0),\\
\gamma_{j,d} =& \rho(\tilde{v}_{j,d}, \tilde{v}_{j,d}^0, \bar{v}_{j,d}, \bar{v}_{j,d}^0),\\
\kappa =& \rho(\tilde{z}, \tilde{z}, \bar{z}^0, \bar{z}^0)\,,\\
\text{with } \rho(a,b,c,d) =& -\frac{1}{2} - \frac{1}{2}\log\frac{a}{b} + \frac{(c-d)^2 + a}{2b}\,.
\end{aligned}$$

One solution would be to tune q using block coordinate descent, that is, by alternative maximization of $\mathcal{L}'$ with respect to Φ and Ξ. Given Φ, Ξ is optimized by setting

$$\xi_{i,j} = \sqrt{\mu_{i,j}^2 + s_{i,j}^2}\,. \tag{5.12}$$

Given Ξ, Φ can be optimized by doing an iteration of gradient descent. Raiko *et al.* [2007] describe a state-of-the-art batch method for optimization of the ELBO in MF models with Gaussian likelihood. Although effective with small datasets, the resulting batch algorithm is infeasible when $\mathbf{X}$ is very large and fully observed since each iteration requires the examination of all the entries in $\mathbf{X}$ before updating any variational parameters. For massive matrices, we propose to use stochastic optimization methods [Robbins & Monro, 1951]. These techniques produce parameter updates after examining only a reduced fraction of the data. The following section describes our stochastic method for optimizing $\mathcal{L}'$ based on the technique stochastic variational inference [Hoffman *et al.*, 2013].

5.4.3 SVI for Binary Matrices

Stochastic optimization methods follow noisy estimates of the gradient of the target function to be optimized. This function is often constructed by summing over a large number of terms. Noise in the gradient arises because the target function is approximated by a noisy estimate which is cheaper to compute. This estimate is obtained by summing over a reduced set of terms that are randomly subsampled. To optimize the correct objective function, the subsampled terms must be rescaled so that the expectation of the gradient of the noisy estimate is the same as the gradient of the original target function.

The difficulty with computing the ELBO and its gradients in Equation (5.11) is the sum over the $L \times M$ terms $\alpha_{i,j}$ which correspond to the likelihood for each entry in the matrix. To avoid computing this expensive sum at each iteration we apply stochastic optimization to $\mathcal{L}'(\Phi) \triangleq \max_{\Xi} \mathcal{L}'(\Phi, \Xi)$. For this, we iterate over the following process. First, we randomly select row and column indices $i \in \{1, \ldots, L\}$ and $j \in \{1, \ldots, M\}$ with probability $p(i,j)$. Second, we optimize $\xi_{i,j}$ by setting $\xi_{i,j} = \sqrt{\mu_{i,j}^2 + s_{i,j}^2}$ as in (5.12). Third, we compute a noisy estimate of $\mathcal{L}'(\Phi)$,

$$\mathcal{L}'_{\text{noisy}}(\Phi_{i,j}) = [c_{i,j}^{\alpha}]^{-1}\alpha_{i,j} + \sum_{d=1}^{D} \beta_{i,d} + \sum_{d=1}^{D} \gamma_{j,d} + \kappa\,, \tag{5.13}$$

where $c_{i,j}^{\alpha}$ is a rescaling constant. Finally, we update $\Phi_{i,j} = \{\{\bar{u}_{i,d}, \tilde{u}_{i,d}, \bar{v}_{j,d}, \tilde{v}_{j,d}\}_{d=1}^{D}, \{\bar{z}, \tilde{z}\}\}$ by making a small step in the direction of the gradient of (5.13). Intuitively, (5.13) is an appropriately rescaled version of (5.11) that includes only those terms which have the same indexes i and j as the subsampled matrix entry $x_{i,j}$. Importantly, the constant $c_{i,j}^{\alpha}$ is chosen to guarantee that the expectation under the data-sampling strategy $p(i,j)$ of the gradient of (5.13) with respect to the elements of $\Phi_{i,j}$ is the same as the gradient of $\mathcal{L}'(\Phi)$ with respect to those elements. That is, when we update $\bar{u}_{i,d}$ or $\tilde{u}_{i,d}$ we set $c_{i,j}^{\alpha} = p(j|i)$. For $\bar{v}_{j,d}$ or $\tilde{v}_{j,d}$ we set $c_{i,j}^{\alpha} = p(i|j)$ and finally, for $\bar{z}$ or $\tilde{z}$ we set $c_{i,j}^{\alpha} = p(i,j)$.

5.4.4 Natural Gradients

Instead of standard gradients, one can achieve much faster convergence using natural gradients [Amari, 1998]. For this, we work with the natural parameters of (5.7). For a Gaussian distribution the natural parameters are the precision and precision times the mean,

$$\dot{u}_{i,d} = \bar{u}_{i,d}/\tilde{u}_{i,d}\,, \qquad \ddot{u}_{i,d} = 1/\tilde{u}_{i,d}\,,$$

and $\dot{v}_{j,d}$, $\ddot{v}_{j,d}$, $\dot{z}$ and $\ddot{z}$ are defined equivalently. Denote the two-dimensional vector of natural parameters for the Gaussians associated with each element in $\mathbf{U}$ as $\mathring{\mathbf{u}}_{i,d} = (\dot{u}_{i,d}, \ddot{u}_{i,d})$ and let $\nabla\mathcal{L}'(\mathring{\mathbf{u}}_{i,d})$ denote the natural gradient of (5.13) with respect to $\mathring{\mathbf{u}}_{i,d}$.

When the model has exponential family complete conditionals, as provided by the Gaussian lower bound to the logistic function in (5.10), then $\nabla\mathcal{L}'(\mathring{\mathbf{u}}_{i,d}) = \mathring{\mathbf{u}}^\star_{i,d} - \mathring{\mathbf{u}}_{i,d}$, where $\mathring{\mathbf{u}}^\star_{i,d} = (\dot{u}^\star_{i,d}, \ddot{u}^\star_{i,d})$ is the value of $\mathring{\mathbf{u}}_{i,d}$ that maximizes (5.13) when all the other natural parameters are kept fixed to their current values. Note that $\mathring{\mathbf{u}}^\star_{i,d}$ is a *noisy estimate* of the maximizer of the exact ELBO (5.11) with respect to $\mathring{\mathbf{u}}_{i,d}$. Thus, the stochastic update rules for $\mathring{\mathbf{u}}_{i,d}$, $\mathring{\mathbf{v}}_{j,d}$ and $\mathring{\mathbf{z}}$ are

$$\mathring{\mathbf{u}}^{\text{new}}_{i,d} = \mathring{\mathbf{u}}^{\text{old}}_{i,d} + \rho^u_i \nabla\mathcal{L}'(\mathring{\mathbf{u}}_{i,d}) = (1-\rho^u_i)\mathring{\mathbf{u}}^{\text{old}}_{i,d} + \rho^u_i \mathring{\mathbf{u}}^\star_{i,d}, \tag{5.14}$$

$$\mathring{\mathbf{v}}^{\text{new}}_{j,d} = \mathring{\mathbf{v}}^{\text{old}}_{j,d} + \rho^v_j \nabla\mathcal{L}'(\mathring{\mathbf{v}}_{j,d}) = (1-\rho^v_j)\mathring{\mathbf{v}}^{\text{old}}_{j,d} + \rho^v_j \mathring{\mathbf{v}}^\star_{j,d}, \tag{5.15}$$

$$\mathring{\mathbf{z}}^{\text{new}} = \mathring{\mathbf{z}}^{\text{old}} + \rho^z \nabla\mathcal{L}'(\mathring{\mathbf{z}}) = (1-\rho^z)\mathring{\mathbf{z}}^{\text{old}} + \rho^z \mathring{\mathbf{z}}^\star, \tag{5.16}$$

where ρ^u_i, ρ^v_j and ρ^z are the stepsizes taken in the direction of the natural gradient. The values of $\dot{u}^\star_{i,d}$, $\ddot{u}^\star_{i,d}$, $\dot{v}^\star_{i,d}$, $\ddot{v}^\star_{i,d}$, $\dot{z}^\star$, $\ddot{z}^\star$ that maximize Equation (5.13) when we have subsampled the entry $x_{i,j}$ from $\mathbf{X}$ are

$$\begin{aligned}
\dot{u}^\star_{i,d} &= \bar{u}^0_{i,d}/\tilde{u}^0_{i,d} + \bar{v}_{j,d}\left[0.5(2x_{i,j}-1) + 2\lambda(\xi_{i,j})(\mu_{i,j} - \bar{u}_{i,d}\bar{v}_{j,d})\right]/p(i|j)\,,\\
\ddot{u}^\star_{i,d} &= 1/\tilde{u}^0_{i,d} - 2\lambda(\xi_{i,j})(\bar{v}^2_{j,d} + \tilde{v}_{j,d})/p(i|j)\,,\\
\dot{v}^\star_{j,d} &= \bar{v}^0_{j,d}/\tilde{v}^0_{j,d} + \bar{u}_{i,d}\left[0.5(2x_{i,j}-1) + 2\lambda(\xi_{i,j})(\mu_{i,j} - \bar{u}_{i,d}\bar{v}_{j,d})\right]/p(j|i)\,,\\
\ddot{v}^\star_{j,d} &= 1/\tilde{v}^0_{j,d} - 2\lambda(\xi_{i,j})(\bar{u}^2_{i,d} + \tilde{u}_{i,d})/p(j|i)\,,\\
\dot{z}^\star &= \left[0.5(2x_{i,j}-1) + 2\lambda(\xi_{i,j})(\mu_{i,j} - \bar{z})\right]/p(i,j) + \bar{z}^0/\tilde{z}^0\,,\\
\ddot{z}^\star &= 1/\tilde{z}^0 - 2\lambda(\xi_{i,j})/p(i,j)\,.
\end{aligned}$$

Note that performing full coordinate updates, that is, computing in alternation $\mathring{\mathbf{u}}^{\text{new}}_{i,d} = \mathring{\mathbf{u}}^\star_{i,d}$, $\mathring{\mathbf{v}}^{\text{new}}_{j,d} = \mathring{\mathbf{v}}^\star_{j,d}$ and $\mathring{\mathbf{z}}^{\text{new}} = \mathring{\mathbf{z}}^\star$, is equivalent to taking steps of size one in the direction of the natural gradient. Even if one observed all of the data and computed $\mathring{\mathbf{u}}^\star_{i,d}$, $\mathring{\mathbf{v}}^\star_{j,d}$, and $\mathring{\mathbf{z}}^\star$ exactly, this step-size can yield slow convergence, and with noisy estimates of these quantities using a fixed unit stepsize can cause the algorithm to be unstable.

The resulting Stochastic Inference method for Binary Matrices (SIBM) works by iterating over the following two steps. First, randomly subsample an entry $x_{i,j}$ from $\mathbf{X}$ with probability $p(i,j)$. Second, perform a small update to the variational parameters that approximate the posterior distribution of the i-th row of $\mathbf{U}$, the j-th row of $\mathbf{V}$ and the global bias z. In practice, each time we sample the indices i and j, we first update $\mathring{\mathbf{z}}$, then all the $\mathring{\mathbf{v}}_{j,d}$ and finally all the $\mathring{\mathbf{u}}_{i,d}$. For faster convergence, each of these operations is performed using the updated parameter values produced by the previous

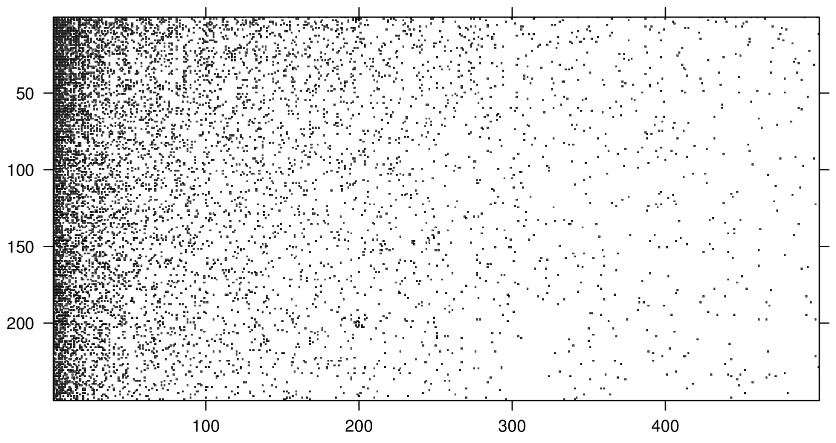

Figure 5.1: Binary matrix obtained by selecting randomly 250 rows with at least 10 ones and the 500 columns with the most ones from the BMS-POS dataset. This matrix is very sparse and has different frequencies of ones across rows and columns.

operations. Furthermore, we recompute the optimal value for $\xi_{i,j}$ whenever any of the natural parameters change.

5.4.5 Sampling Distributions for Sparse Imbalanced Matrices

We consider different choices of $p(i, j)$, the probability distribution used to subsample the entries of $\mathbf{X}$. The usual objective in binary matrix factorization is to predict the location of those entries in $\mathbf{X}$ that would have taken value one but actually took value zero due to the additive noise matrix $\mathbf{E}$; for example, to recommend new products to a user or discover new links in a network. However, real-world binary matrices are usually highly sparse, as illustrated in Figure 5.1. This means that when the sampling strategy is uniform (denoted S-Uniform), that is $p(i, j) = 1/(LM)$, most of the sampled entries $x_{i,j}$ will take value zero. As a result, SIBM may take many iterations to obtain good predictive performance.

We propose strategies that subsample the more useful entries of $\mathbf{X}$ so that the model converges rapidly. This resembles active learning, except that unlike in active learning, we must eliminate the sampling bias introduced by our specific choice of $p(i, j)$. That is, we must select $c^{\alpha}_{i,j}$ so that the expected gradient of (5.13) is the same as the gradient of (5.11). Therefore, we propose two simple strategies for which we can compute the appropriate rescaling constants.

To ensure that we see enough ones, a better strategy is to sample zeros and ones with equal probability, regardless of the empirical frequencies found in $\mathbf{X}$ (S-Balanced),

$$p(i,j) = \frac{1}{2\sum_{l=1}^{L}\sum_{m=1}^{M}\mathbb{I}[x_{i,j} = x_{l,m}]}\,,$$

where $\mathbb{I}[\cdot]$ is the indicator function. Now, each time that an entry is sampled we obtain a zero or a one with equal probability. However, another characteristic of real-world binary matrices is that the frequency of ones and zeros can change considerably across rows or columns. For example, the matrix in Figure 5.1 contains a few columns with a large number of ones and many columns with very few ones. A similar pattern is observed in the rows, although in this case the effect is smaller. In practice, it will take SIBM a long time to accurately model those ones located in rows/columns with many zeros. Any entry sampled in those rows/columns will usually take value zero and sampling a zero there is unlikely to be useful since SIBM can learn quickly that these rows/columns are very sparse. Therefore, we propose a new sampling strategy (S-Biased) to account for this by biasing S-Balanced so that the probability of sampling a one at location (i,j) is proportional to i) the number of zeros found in the i-th row and ii) the number of zeros found in the j-th column. The equivalent bias is introduced for the zeros. The resulting sampling distribution is

$$p(i,j) = \frac{r_i^{(1-x_{i,j})}c_j^{(1-x_{i,j})}}{2\sum_{l=1}^{L}\sum_{m=1}^{M}\mathbb{I}[x_{i,j} = x_{l,m}]r_l^{(1-x_{l,m})}c_m^{(1-x_{l,m})}}\,,$$

where $r_i^{(0)}$ and $r_i^{(1)}$ are the number of zeros and ones in the i-th row of $\mathbf{X}$ and likewise $c_j^{(0)}$ and $c_j^{(1)}$ count the number of zeros and ones in the j-th column. These counts are lower thresholded at 1 so that $p(i,j) \neq 0$.

5.4.6 Learning the Minibatch Online

Stochastic methods often use minibatches to reduce variance in the noisy estimates of the natural gradient to help the algorithm converge faster. Instead of updating the variational parameters after subsampling a single matrix entry, the updates are averaged over a minibatch of data. When using a minibatch of size S, we randomly subsample S entries from $\mathbf{X}$. For each subsampled entry $x_{i,j}$, we compute and store the parameter values $\mathring{\mathbf{u}}_{i,d}^{\star}$ and $\mathring{\mathbf{v}}_{j,d}^{\star}$ that would have been produced during the normal execution of SIBM without minibatches. After subsampling S entries, we update each $\mathring{\mathbf{u}}_{i,d}$ if at least one of the last S subsampled entries belongs to the i-th row of $\mathbf{X}$. The minibatch update rule follows from Equation (5.14),

$$\mathring{\mathbf{u}}_{i,d}^{\text{new}} = (1-\rho_i^u)\mathring{\mathbf{u}}_{i,d}^{\text{old}} + \rho_i^u \mathring{\mathbf{u}}_{i,d}^{\star,\text{avg}}\,, \tag{5.17}$$
$$\text{where} \quad \mathring{\mathbf{u}}_{i,d}^{\star,\text{avg}} = \frac{1}{n(i)}\sum_{s=1}^{n(i)} \mathring{\mathbf{u}}_{i,d}^{\star,s}\,,$$

and $n(i)$ is the number of entries from the i-th row found in the last minibatch of S subsampled entries with $\mathring{\mathbf{u}}_{i,d}^{\star,s}$ being the value of $\mathring{\mathbf{u}}_{i,d}^{\star}$ produced when the s-th of those entries is subsampled. The minibatch update rules for $\mathring{\mathbf{v}}_{j,d}$ and z are similar.

An important question in stochastic methods is how to choose the minibatch size S. The choice of S is particularly relevant when working with matrix factorization models, because parameter distributions are often heavy tailed [Lakshminarayanan *et al.*, 2011]. In our stochastic method, this results in heavy tailed noisy estimates of the natural gradients. The choice of S governs a trade-off between the reduction of these heavy tails and slow convergence due to excessively large minibatches, in the limit of $S = LM$ we reduce to batch optimization.

Typically S is hand-tuned to each dataset or optimized with expensive cross-validation search. To avoid these procedures, we propose an adaptive algorithm that selects S appropriately to the statistics of the data during learning. In particular, we choose S so that we bound the magnitude of the error in the noisy gradient. Let $\mathring{\mathbf{u}}_{i,d}^{\star,\star}$ be the value of $\mathring{\mathbf{u}}_{i,d}$ that maximizes the *exact* ELBO (5.11), that is, the optimum given all of the data with the other parameters fixed. We obtain a probabilistic bound on the relative error of $\mathring{\mathbf{u}}_{i,d}^{\star,\text{avg}}$ in (5.17) with respect to the global maximizer of the ELBO, $\mathring{\mathbf{u}}_{i,d}^{\star,\star}$, using Markov's inequality. Markov's inequality is an upper bound on the probability that a non-negative random variable exceeds a particular value. This is a general bound that makes no assumptions about the distribution of the variable. This gives us the following bound on the error,

$$\begin{aligned}
\delta = p\left[\frac{\|\mathring{\mathbf{u}}_{i,d}^{\star,\text{avg}} - \mathring{\mathbf{u}}_{i,d}^{\star,\star}\|_2^2}{\|\mathring{\mathbf{u}}_{i,d}^{\star,\star}\|_2^2} \geq \theta\right] &\leq \frac{\mathbb{E}[\|\mathring{\mathbf{u}}_{i,d}^{\star,\text{avg}} - \mathring{\mathbf{u}}_{i,d}^{\star,\star}\|_2^2]}{\theta\|\mathring{\mathbf{u}}_{i,d}^{\star,\star}\|_2^2} \\
&= \frac{\|\text{Var}[\mathring{\mathbf{u}}_{i,d}^{\star}]\|_1}{\theta\|\mathbb{E}[\mathring{\mathbf{u}}_{i,d}^{\star}]\|_2^2}\mathbb{E}\left[\frac{1}{n(i)}\right] \\
&\approx \frac{\|\text{Var}[\mathring{\mathbf{u}}_{i,d}^{\star}]\|_1}{Sp(i)\theta\|\mathbb{E}[\mathring{\mathbf{u}}_{i,d}^{\star}]\|_2^2}\,,
\end{aligned} \tag{5.18}$$

where $\text{Var}[\mathring{\mathbf{u}}_{i,d}^{\star}]$ is a vector with the variances of the entries in $\mathring{\mathbf{u}}_{i,d}^{\star}$ and $p(i)$ is the probability of sampling an element from the i-th row of $\mathbf{X}$, $p(i) = \sum_{j=1}^{M} p(i,j)$. In Equation (5.18) we approximate $\mathbb{E}\left[1/n(i)\right]$ by $1/[p(i)S]$. Also note that $\mathring{\mathbf{u}}_{i,d}^{\star,\star} = \mathbb{E}[\mathring{\mathbf{u}}_{i,d}^{\star,\text{avg}}]$. We now solve for S to obtain a minibatch size that approximately limits the probability that the relative error of $\mathring{\mathbf{u}}_{i,d}^{\star,\text{avg}}$ is larger than θ,

$$S_{i,d}^{u} = \frac{\|\text{Var}[\mathring{\mathbf{u}}_{i,d}^{\star}]\|_1}{\theta\delta p(i)\|\mathbb{E}[\mathring{\mathbf{u}}_{i,d}^{\star}]\|_2^2}\,. \tag{5.19}$$

Intuitively, the resulting minibatch size increases with the inverse of the signal to noise ratio (SNR) in the estimate $\mathring{\mathbf{u}}_{i,d}^{\star}$ of the global maximizer of the exact ELBO in (5.11), $\mathring{\mathbf{u}}_{i,d}^{\star,\star}$. If the SNR decreases, this rule chooses larger minibatches to mitigate the greater relative errors. The rule in (5.19) provides a different minibatch size $S_{i,d}^{u}$ for each $\mathring{\mathbf{u}}_{i,d}$, and similarly for each $\mathring{\mathbf{v}}_{j,d}$. Therefore, to select the overall size S we average of the minibatch sizes chosen for each parameter,

$$S = \frac{\sum_{i=1}^{L}\sum_{d=1}^{D} S_{i,d}^{u} + \sum_{j=1}^{M}\sum_{d=1}^{D} S_{j,d}^{v}}{DL + DM}\,. \tag{5.20}$$

The proposed approach requires choosing a single *dataset-independent* parameter, the product of θ and δ, as opposed to hand-tuning S to each dataset. By making $\theta\delta$ small we limit the expected deviation of $\mathring{\mathbf{u}}_{i,d}^{\star,\text{avg}}$ from $\mathring{\mathbf{u}}_{i,d}^{\star,\star}$. Empirically we find $\theta\delta = 2$ leads to good performance.

Equation (5.19) requires $\mathbb{E}[\mathring{\mathbf{u}}_{i,d}^{\star}]$ and $\text{Var}[\mathring{\mathbf{u}}_{i,d}^{\star}]$ which are unknown *a priori*. Therefore, we estimate these quantities online using exponentially weighted moving averages. Let $\bar{\mathbf{u}}_{i,d}$ and $\bar{\bar{\mathbf{u}}}_{i,d}$ denote respectively estimates of the mean and mean squared value of $\mathring{\mathbf{u}}_{i,d}^{\star}$. Each time we draw a sample from the i-th row of $\mathbf{X}$, we update these averages as

$$\begin{aligned}\bar{\mathbf{u}}_{i,d} &= (1-\hat{\rho}_i^{u})\bar{\mathbf{u}}_{i,d} + \hat{\rho}_i^{u}\mathring{\mathbf{u}}_{i,d}^{\star}\,,\\ \bar{\bar{\mathbf{u}}}_{i,d} &= (1-\hat{\rho}_i^{u})\bar{\bar{\mathbf{u}}}_{i,d} + \hat{\rho}_i^{u}[\mathring{\mathbf{u}}_{i,d}^{\star}\circ\mathring{\mathbf{u}}_{i,d}^{\star}]\end{aligned}$$

where "$\circ$" denotes the Hadamard element-wise product operation. The interpolation weight $\hat{\rho}_i^{u}$ is selected as $\hat{\rho}_i^{u} = (1+\hat{t}_u^{i})^{-\lambda}$, where $\hat{t}_u^{i}$ is the number of times that we have sampled an entry in the i-th row of $\mathbf{X}$ and we set $\lambda = 0.7$. The quantities $\mathbb{E}[\mathring{\mathbf{u}}_{i,d}^{\star}]$ and $\text{Var}[\mathring{\mathbf{u}}_{i,d}^{\star}]$ are then estimated using $\mathbb{E}[\mathring{\mathbf{u}}_{i,d}^{\star}] \approx \bar{\mathbf{u}}_{i,d}$ and $\text{Var}[\mathring{\mathbf{u}}_{i,d}^{\star}] \approx \bar{\bar{\mathbf{u}}}_{i,d} - \bar{\mathbf{u}}_{i,d}\circ\bar{\mathbf{u}}_{i,d}$. The minibatch sizes $S_{j,d}^{v}$ for the natural parameters $\mathring{\mathbf{v}}_{j,d}$ are obtained in a similar manner. As learning progresses and the parameters are updated these statistics will change, therefore the algorithm adapts the minibatch size online.

Finally S in (5.20) is computed efficiently by exploiting the fact that $S_{i,d}^{u}$ and $S_{j,d}^{v}$

only change if the minibatch includes a sample in the i-th row or j-th column. To collect the initial statistics, we use $S = 5L$ for the first minibatch, subsequent values of S chosen by the algorithm are insensitive to this choice, as evidenced by our experiments in Section 5.5.

5.4.7 The Full SIBM Algorithm

The final detail required for SIBM is the choice of the stepsizes ρ_i^u, ρ_j^v and ρ_z. These should be reduced each time $\mathring{\mathbf{u}}_{i,d}$, $\mathring{\mathbf{v}}_{j,d}$ and $\mathring{\mathbf{z}}$ are updated in order to satisfy the requirements for correct convergence of the stochastic gradient descent routine described in Robbins & Monro [1951]. We use a simple Robbins-Monro schedule. For this, let t_i^u, t_j^v, t_j^v be the number of times that each vector of natural parameters $\mathring{\mathbf{u}}_{i,d}$, $\mathring{\mathbf{v}}_{j,d}$ and $\mathring{\mathbf{z}}$ have been updated respectively. After each stochastic update, the stepsizes are computed as $\rho_i^u = (1 + t_i^u)^{-\lambda}$, $\rho_j^v = (1 + t_j^v)^{-\lambda}$ and $\rho^z = (1 + t^z)^{-\lambda}$, where $\lambda \in (0.5, 1]$. In our experiments we found $\lambda = 0.7$ produced good overall results. The full SIBM routine is summarized in Algorithm 1.

5.4.8 Related Work

Specific Challenges for SVI in Matrix Factorization

SVI has been applied to other probabilistic models such as Latent Dirichlet Allocation [Hoffman *et al.*, 2010], the Hierarchical Dirichlet Process, [Hoffman *et al.*, 2013], and Bayesian Nonparametric models [Bryant & Sudderth, 2012; Wang *et al.*, 2011]. In these cases there is a clear distinction between *local* and *global* parameters or variables. The distinction is governed by the conditional dependencies in the model. A local variable is associated with each observation, and the conditional distribution of each observation and its local variable is independent of all other local variables and observations given the global variables [Hoffman *et al.*, 2013].

Therefore, local parameters are updated only when a particular data point is subsampled and in the aforementioned models the global variational parameters are updated when any datapoint is subsampled. In MF, the definition of a datapoint is more ambiguous: does a datapoint correspond to a row, column, entry or entire matrix? We subsample individual matrix entries. In this case the row and column parameters $\mathbf{U}$ and $\mathbf{V}$ are *partially global* since they do not satisfy the conditional independence assumptions to be local, and are only updated when elements in the corresponding row or column are subsampled. With MF, the partially global nature of the row and column parameters makes the data sub-sampling strategy more important because it

Algorithm 1 Stochastic Inference for Binary Matrices

```
1: Input: matrix $\mathbf{X}$, initial parameters $\Phi$, # samples $T$
2: for $t = 1$ to $T$ do
3:   select minibatch size $S$ using (5.20)
4:   for $s = 1$ to $S$ do
5:     save $\mathring{\mathbf{u}}_{i,1}, \ldots, \mathring{\mathbf{u}}_{i,D}$, $\mathring{\mathbf{v}}_{j,1}, \ldots, \mathring{\mathbf{v}}_{j,D}$ and $\mathring{\mathbf{z}}$
6:     sample row and column indices $(i, j) \sim p(i, j)$
7:     compute stepsize $\rho^z$ using Robbins-Monro
8:     update $\xi_{i,j}$ using (5.12)
9:     compute $\mathring{\mathbf{z}}^\star$ and update $\mathring{\mathbf{z}}$ using (5.16)
10:    for $d = 1$ to $D$ do
11:      update $\xi_{i,j}$ using (5.12)
12:      compute $\mathring{\mathbf{v}}^\star_{j,d}$ and update $\mathring{\mathbf{v}}_{j,d}$ using (5.15)
13:      update $\mathring{\mathbf{v}}^{\star,\mathrm{avg}}_{j,d}$
14:    end for
15:    for $d = 1$ to $D$ do
16:      update $\xi_{i,j}$ using (5.12)
17:      compute $\mathring{\mathbf{u}}^\star_{i,d}$ and update $\mathring{\mathbf{u}}_{i,d}$ using (5.14)
18:      update $\mathring{\mathbf{u}}^{\star,\mathrm{avg}}_{i,d}$
19:    end for
20:    restore $\mathring{\mathbf{u}}_{i,1}, \ldots, \mathring{\mathbf{u}}_{i,D}$, $\mathring{\mathbf{v}}_{j,1}, \ldots, \mathring{\mathbf{v}}_{j,D}$ and $\mathring{\mathbf{z}}$
21:  end for
22:  for any row $i$ sampled in the last minibatch do
23:    compute stepsize $\rho^u_i$ using Robbins-Monro
24:    update $\mathring{\mathbf{u}}_{i,1}, \ldots, \mathring{\mathbf{u}}_{i,D}$ using (5.17)
25:  end for
26:  for any column $j$ sampled in the last minibatch do
27:    compute stepsize $\rho^v_j$ using Robbins-Monro
28:    update $\mathring{\mathbf{v}}_{j,1}, \ldots, \mathring{\mathbf{v}}_{j,D}$
29:  end for
30:  compute stepsize $\rho^z$ using Robbins-Monro
31:  update $\mathring{\mathbf{z}}$
32: end for
33: Output: $\{\mathring{\mathbf{u}}_{i,1}, \ldots, \mathring{\mathbf{u}}_{i,D}\}_{i=1}^{L}$, $\{\mathring{\mathbf{v}}_{j,1}, \ldots, \mathring{\mathbf{v}}_{j,D}\}_{j=1}^{M}$ and $\mathring{\mathbf{z}}$
```

determines which parameters are updated and how often. A more closely related application of SVI is to the Mixed-Membership Stochastic Blockmodel for L-node binary networks [Gopalan *et al.*, 2012], but in this case only one $L \times D$ matrix of parameters, the community memberships, is partially global.

A second difficulty for SVI posed by MF models arises from the direct coupling of the parameters updates. For example, the update for the row variational parameters in (5.14) is a direct function of the column parameters $\mathring{\mathbf{v}}_{j,d}$. As noted in Section 5.4.6, the parameters in MF models are often heavy tailed. The combination of update coupling and heavy tailed parameters results in heavy tailed noisy gradients. This makes the minibatch size selection particularly important with MF models. Algorithms that adaptively change the stepsize online have been proposed [Ranganath *et al.*, 2013; Schaul *et al.*, 2012]. However, these algorithms assume a Gaussian distribution of noisy estimates of the natural gradients. Since the noisy estimates have heavy tails these methods can result in unstable behaviour in MF models, and we found that the sequences of the stepsizes ended up diverging.

Algorithms for Probabilistic Binary MF

An alternative stochastic algorithm just subsamples the zeros is proposed in Paquet & Koenigstein [2013]. However, unlike SIBM, this method does not correct for the bias introduced by the subsampling process and hence yields poorer solutions, as we observe in our experiments.

With sparse matrices batch variational inference schemes can be efficient since the time required to update the parameters scales linearly only in the number of observations [Lim & Teh, 2007]. However, with fully observed matrices this is usually impractical since each update costs $\mathcal{O}(LM)$. We note that with sparse binary matrices the required computations with a Gaussian likelihood in Raiko *et al.* [2007] can be rearranged so that the cost per iteration is linear only in the number of ones. This can be achieved essentially by decomposing the likelihood into a sum of a term corresponding to a full matrix of zeros and correction factors for the observed ones. Now any $\mathcal{O}(LM)$ terms may be pre-computed, however, this is not possible with the logistic likelihood which is more appropriate for binary data.

With a Gaussian likelihood, one can avoid optimization altogether, and use the analytic solution for the global maximum of the ELBO derived in Nakajima *et al.* [2010]. However, the solution only applies to highly restricted models. The limitations include i) the likelihood must be Gaussian with equal variance across matrix entries,[1] ii) **U** and **V** must have zero-mean isotropic priors, and iii) no bias parameters can be included. These constraints yield a large negative effect to predictive performance, as we show in our experiments. An iterative scheme has been proposed to extend this approach to binary likelihoods at the cost of making very crude approximations to the logistic likelihood function [Seeger & Bouchard, 2012]. In practice, with binary matrices this method tends to produce only small gains in performance with respect to the solution in Nakajima *et al.* [2010].

A large number of non-probabilistic algorithms have been proposed for MF. With binary matrices, one of the best performing is Bayesian Personalized Ranking (BPR) which directly optimizes a ranking loss function. BPR has shown state-of-the-art results on item recommendation against a wide range of systems [Rendle *et al.*, 2009]. It was also a key component in many of the best solutions in Track 2 of the KDD-Cup'11 music recommendation competition [Dror *et al.*, 2012]. We show comparisons to all of the above methods, including BPR, in our experiments.

5.5 Experiments

SIBM is evaluated in experiments with synthetic and real-world binary matrices. We consider six datasets that include i) a synthetic dataset generated by sampling **X** from the generative model assumed by SIBM. We fix $D = 5$ and generate **U** and **V** by sampling all the $u_{i,d}$ and $v_{j,d}$ independently from $\mathcal{N}(0, 100)$. The global bias is fixed to $z = -500$, yielding binary matrices with about 98% sparsity. We consider two real-world datasets from the FIMI repository: ii) purchase data from a retail store (retail) [Brijs *et al.*, 1999] and iii) click data from an online news portal (Kosarak). We include two datasets from the 2000 KDD Cup [Kohavi *et al.*, 2000; Zheng *et al.*, 2001], iv) point-of-sale data from a retailer (POS, originally BMS-POS) and v) click data from an e-commerce website (WebView, originally BMS-WebView-2). Finally, we include vi) the Netflix data, treating 4-5 star ratings as ones. We pre-process the original datasets to be able to compare to the computationally expensive batch approach. We keep the 1000 columns with the highest number of ones and discard rows with fewer than 10 ones. We consider *small* and *large* versions of each dataset. We subsample 2000 rows for the small and 40,000 rows for the large datasets, except in retail and WebView, where

[1] The restriction to equal likelihood variances across the matrix means that the analytic solution cannot be used directly with the Gaussian approximation to the sigmoid function in (5.10) to handle the logistic likelihood.

we use approximately the maximum number of rows for the large datasets, 10,000 and 5000, respectively.

Each matrix is randomly split into a training matrix and a set of test entries with value one. The training matrix is generated by randomly removing a single one from each row in the original matrix and adding it to the test set. Predictive performance is evaluated using recall at N, which is equivalent to precision when a single one is held out. Recall is popular metric for recommendation tasks [Gunawardana & Shani, 2009] because it measures directly the ability to find the items a user may like. We iterate over the rows, using (5.4) to compute the probability of each zero entry actually taking value one. We select the top N zero entries with highest probability in that row. Recall is computed as the average number of times that the test entry appears in this list. We use $N = 10$ and repeat the experiment 25 times on each small dataset and 10 times on each large one.

5.5.1 Sampling Strategies and Automatic Minibatch

Figure 5.2, left, shows results for SIBM when using the sampling strategies S-Uniform, S-Balanced and S-Biased on the small Netflix dataset. To eliminate the dependence of these strategies on the minibatch size, we select the value of S for each strategy using cross-validation. The results for all other datasets were similar. On all of the datasets S-Biased performs best, followed by S-Balanced. As expected, with sparse binary matrices, uniform sampling (S-Uniform) yields slow convergence.

Figure 5.2, right, shows the evolution of the minibatch size S on each small dataset. Similar results are obtained for the large datasets. The plot shows that the chosen value of S is highly dataset-dependent. Interestingly, for some datasets S grows as learning progresses, but for others it shrinks. We fix the minimum value for S to $\max(L, M) = 2000$. This value is selected in the retail dataset.

5.5.2 Comparison to Batch and Alternative Methods

Other Algorithms

We compare the full SIBM algorithm that selects the minibatch size S automatically (SIBM-auto) with an alternative in which S is selected via cross-validation to maximize recall on a validation set (SIBM-recall).

We also compare with a version of SIBM-recall that finds the *maximum a posteriori* (MAP) solution using stochastic gradient ascent with the same data subsampling strategy (MAP-recall). MAP-recall employs the same rescaling constants $c^{\alpha}_{i,j}$ as SIBM. Two

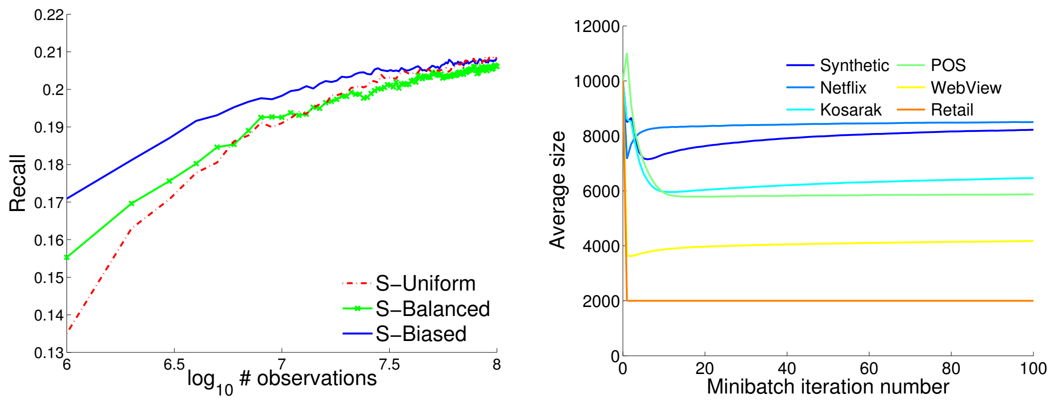

Figure 5.2: *Left*: average recall obtained for different sampling strategies in the small Netflix dataset. *Right*: evolution of the average minibatch size S selected in each small dataset.

modifications to SIBM-recall are required to obtain MAP-recall. First, we no longer use the variational parameters related to the variance of the posterior approximation since we only seek a point estimate of the model parameters. Second, MAP-recall uses standard gradients and not natural gradients. Natural gradients are not available when doing MAP inference because the Kullback-Leibler divergence between probability distributions is no longer being minimized. In MAP-recall we select the stepsizes taken in the direction of the noisy gradient using a Robbins-Monro schedule similar to the one used in SIBM-recall. In particular, let t_i^u, t_j^v, t_j^v be the number of times that the parameters $\mathbf{u}_{i,d}$, $\mathbf{v}_{j,d}$ and z have been updated, respectively. The stepsizes are given by $\rho_i^u = (t_0^u + t_i^u)^{-\lambda}$, $\rho_j^v = (t_0^v + t_j^v)^{-\lambda}$ and $\rho^z = (t_0^z + t^z)^{-\lambda}$. We fixed $\lambda = 1$, which worked better than $\lambda = 0.7$ used by SIBM. We also hand-tuned t_0^z, t_0^y and t_0^v to yield the best possible overall performance. We found that MAP-recall was more sensitive to the values of t_0^z, t_0^y and t_0^v than SIBM, for which $t_0^z = t_0^y = t_0^v = 1$ works well. The increased sensitivity to these learning parameters is probably due to the inability to use natural gradients with MAP inference.

On the small datasets we compare with the batch algorithm (batch) that maximizes the exact ELBO (5.11) [Raiko *et al.*, 2007]. This method is too expensive with the large datasets. Therefore, with these datasets we run batch by subsampling zeros, keeping only 20 times as many zeros as ones.

We compare our method to the analytic solution for a Gaussian likelihood [Nakajima *et al.*, 2010] (Nak10) and the extension of this method to binary matrices [Seeger &

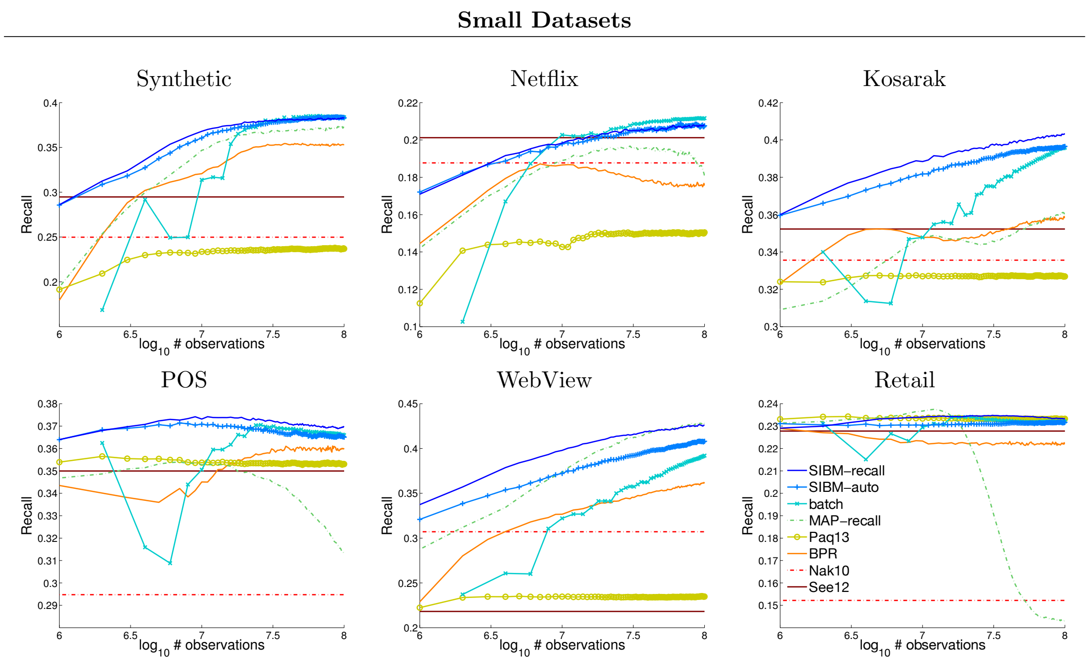

Figure 5.3: Average recall for each method on each small dataset versus number of samples drawn from $\mathbf{X}$.

Bouchard, 2012] (See12). We also evaluate the scheme described in Paquet & Koenigstein [2013] (Paq13). Finally, we compare to one of the best performing non-variational Bayesian algorithms, BPR [Rendle *et al.*, 2009].

Results

Figures 5.3 and 5.4 show the average recall obtained by each method versus the number of entries subsampled from $\mathbf{X}$ on the small and large datasets respectively. Other than the analytic solutions (Nak10 and See12), all algorithms have linear cost in the number of observations. It is hard to quantify the number of entries observed by Nak10 and See12, which are based on iterative calls to an SVD subroutine. Therefore, we assume that they run instantaneously and their performance is presented as a constant line.[1] We present also the negative ELBO (cost) versus number of samples in Figure 5.5

Tables 5.1 and 5.2 contain the average recall and cost after observing 10^7 samples in the small datasets and WebView and Retail large datasets, and 10^8 on the others. We take a slice rather than presenting performance after convergence of the algorithms because we are interested in scalable methods that produce good solutions with a

[1]This is a generous assumption for See12, see wall-clock times in Figure 5.6.

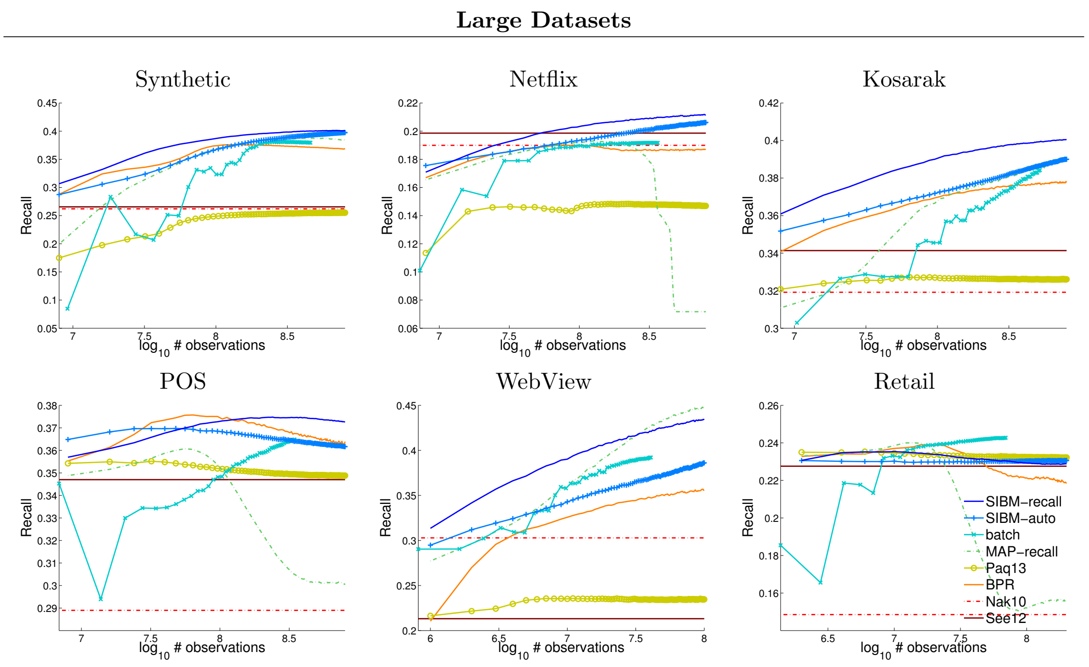

Figure 5.4: Average recall for each method on each large dataset versus number of samples drawn from **X**.

limited computational budget. Running the algorithms to convergence on massive matrices can take an infeasible amount of time. With the large datasets, computing the ELBO is too expensive so we do not report cost values. We only report cost for the stochastic methods and See12 since BPR, MAP, Paq13 and Nak10 do not yield comparable lower bound values. This is because BPR and MAP are not performing VB, the model in Paq13 subsamples the zeros and Nak10 uses a Gaussian likelihood so the ELBO is incomparable. Bold typeface indicates the best results (and those statistically indistinguishable using a paired t-test at the 5% level), underlining denotes the second best. Tables 5.1 and 5.2 show that in terms of recall, the best method is SIBM-recall, with SIBM-auto coming close. Regarding the ELBO, SIBM-auto yields the best results.

Figures 5.3 and 5.4 show that SIBM converges faster than batch and sometimes to better solutions, such as in the WebView dataset. SIBM-auto produces the greatest improvements during the first iterations of learning. These first iterations are most relevant for large scale learning. With large datasets, only a few passes over the available data are possible. It is then when stochastic methods are most useful. In terms of the ELBO, the batch algorithm will converge to an optimum of the lower bound. However, early in learning the stochastic algorithm achieves much better values. In most cases

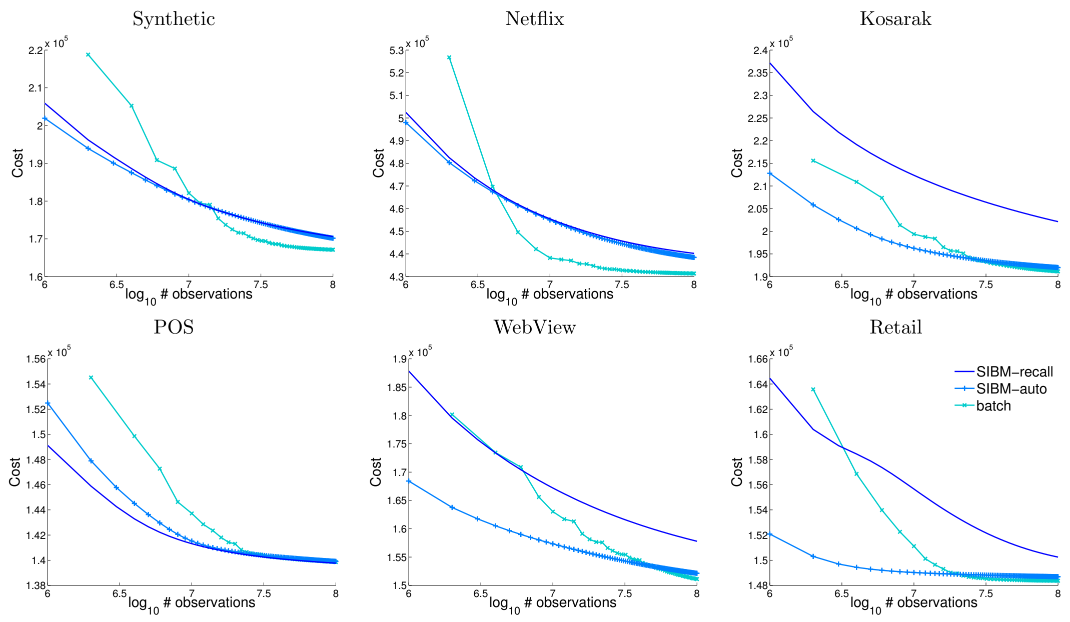

Figure 5.5: Average cost (negative ELBO) versus number of samples for each method on each small dataset.

SIBM achieves a reasonably good solution before batch has completed a single iteration. On recall, the results of SIBM-auto are very close to those of the gold-standard SIBM-recall and MAP-recall performs worse in general than the variational methods SIBM-auto and SIBM-recall. MAP-recall seems to overfit since its performance sometimes deteriorates during the later iterations.

The analytic algorithms (Nak10, See12) obtain poor results due to the simplistic modelling assumptions that they make. Paq13 performs poorly because this method subsamples the zeros and does not correctly account for the bias introduced by the subsampling process. As a result, it converges to suboptimal solutions. BPR converges to worse solutions than SIBM and batch.

Wall Clock Times

We recorded the wall-clock times for each algorithm. Figure 5.6 gives the times for each small dataset. As with 'number of samples' in Figures 5.3 and 5.4. the results on the large datasets are very similar to those on the small datasets. Clock time is more implementation-dependent than number of samples observed, to be as fair as possible, all algorithms were implemented in C. Nevertheless, results for recall vs. time and recall vs. number of observed entries are similar. The main difference is that now

	recall								cost$\times 10^{-5}$			
Dataset	**SIBM recall**	**SIBM auto**	**batch**	**MAP recall**	**Paq13**	**BPR**	**Nak10**	**See12**	**SIBM recall**	**SIBM auto**	**batch**	**See12**
Synthetic	**0.368**	0.360	0.314	0.347	0.234	0.321	0.250	0.295	**1.804**	**1.803**	1.821	4.313
Netflix	0.198	0.198	**0.203**	0.189	0.143	0.187	0.188	**0.201**	4.555	4.550	**4.383**	6.807
Kosarak	**0.388**	0.382	0.348	0.348	0.327	0.348	0.336	0.352	2.124	**1.963**	1.994	3.607
POS	**0.373**	**0.371**	0.351	0.353	0.354	0.345	0.295	0.350	**1.413**	**1.415**	1.437	2.674
WebView	**0.398**	0.372	0.322	0.374	0.235	0.327	0.307	0.218	1.672	**1.573**	1.630	2.886
Retail	0.234	0.230	0.229	**0.237**	0.233	0.223	0.152	0.228	1.557	**1.490**	1.511	2.430

Table 5.1: Small datasets, recall and cost after observing 10^7 samples. Bold typeface indicates the best result (and those statistically indistinguishable) on each dataset, underlining indicates the second best.

Dataset	**SIBM recall**	**SIBM auto**	**batch**	**MAP recall**	**Paq13**	**BPR**	**Nak10**	**See12**
Synthetic	**0.387**	0.367	0.324	0.368	0.249	0.374	0.262	0.266
Netflix	**0.203**	0.193	0.190	0.192	0.146	0.190	0.190	0.199
Kosarak	**0.391**	0.372	0.346	0.368	0.327	0.370	0.319	0.341
POS	0.373	0.368	0.348	0.352	0.352	**0.374**	0.289	0.347
WebView	**0.390**	0.343	0.359	0.360	0.235	0.326	0.303	0.213
Retail	**0.235**	0.230	0.233	**0.239**	0.235	**0.237**	0.149	0.228

Table 5.2: Large datasets, recall after observing 10^7 samples from WebView, Retail and 10^8 from others.

SIBM-recall, MAP-recall and BPR are penalized due to the additional time that they require to run cross-validation searches for selecting the minibatch size (SIBM-recall and MAP-recall) and regularization parameters (BPR).

5.6 Conclusions and Extensions

In this chapter we have addressed one particular difficulty encountered in matrix factorization: scaling probabilistic inference with fully observed binary matrices. For this we have presented a complete algorithm that can handle heavy tailed parameter values and sparse, imbalanced matrices that are common in practice. The approach extends stochastic variational inference to matrix factorization models, a class of models not addressed before by SVI. The proposed method has the following advantages with respect to existing probabilistic solutions for binary matrix factorization: i) we can handle fully observed matrices, ii) learning occurs by subsampling the matrix entries, iii) we use a likelihood function for binary data instead of for continuous data, iv) flexible priors and additional bias parameters can be easily incorporated into the model. As a result,

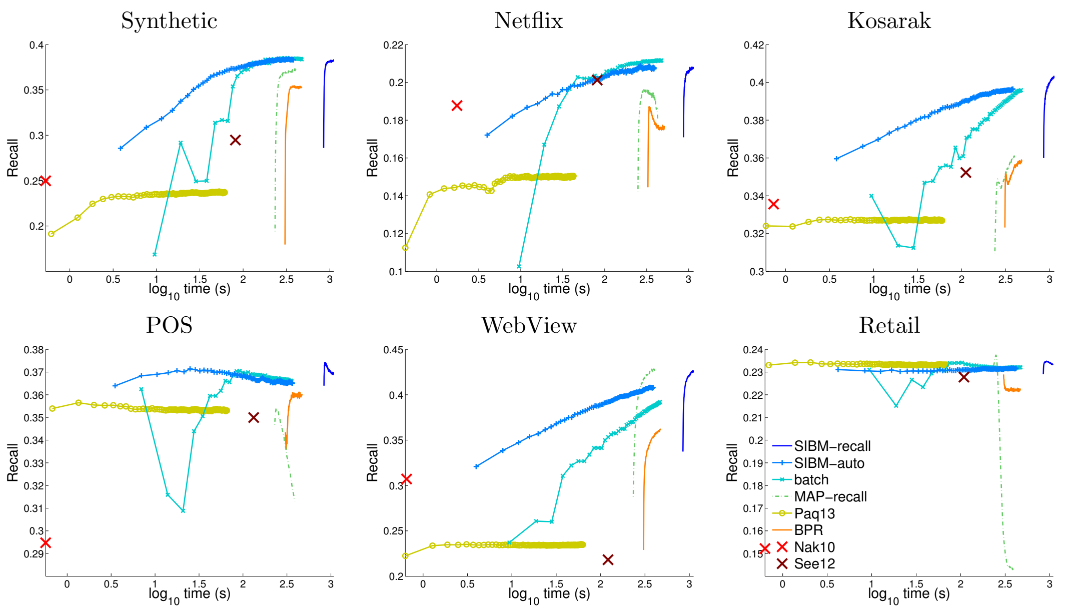

Figure 5.6: Average recall for each method on each dataset versus wall clock times.

our algorithm achieves faster convergence than an alternative batch approach and has better predictive performance than other state-of-the-art scalable solutions or analytic methods based on the SVD decomposition. Good performance in this domain requires appropriate data subsampling mechanisms and the use of minibatches. To account for this, we have provided new data subsampling strategies and a technique to adjust the minibatch size automatically and adaptively to the data. Our technique for learning the minibatch size could be applied more generally to other SVI algorithms.

One extension would be to learn the stepsize schedule also. However, as noted in Section 5.4.8, such an algorithm has to be robust to the heavy tailed noisy updates observed in MF models. For a second extension, to achieve scalability to truly massive matrices, would be to combine parallel architectures with our online algorithm. Certain 'inner loop' operations could be trivially parallelized, such as the **for** loops in lines 10 and 15 of Algorithm 1. However, parallelization of larger operations is more likely to help due to the time taken to transfer data to, and results from, worker machines. Larger parallelizations, say of the outer loop over minibatches (line 2 in Algorithm 1), would require more care since parameter values would become out-dated in parallel machines. However, although theoretically hard to justify, asynchronous stochastic optimization architectures have enjoyed recent success with neural networks [Dean *et al.*, 2012].

In some binary matrix datasets it may be ambiguous whether an entry with value zero corresponds to an observed negative or a lack of observation. For example, in market basket data a zero could correspond to a user deciding not to buy an item, or never having observed the item. Modelling all of the zeros as observed is advantageous over treating them as unobserved, since in the latter case the maximum likelihood solution would predict ones everywhere. If we modelled this ambiguity we would need a latent variable for each zero in $\mathbf{X}$. Also, for matrices which are not fully observed, such as rating matrices, the missing entries may not be selected at random, which is often the case with rating data [Marlin & Zemel, 2009]. Proper probabilistic treatment requires consideration of all the matrix entries, both observed and unobserved, during inference. When modelling ambiguous zeros or missing data, the computational cost is at least as great as with a fully observed matrix. Therefore, in these cases our stochastic routine could also be used to achieve computationally efficient inference.

Finally, the scaling constants $c^{\alpha}_{i,j}$ in Equation (5.13) require knowing the matrix dimensions L and M. Currently, our algorithm cannot handle formally new rows or columns being added to the matrix. The inability to handle streamed data, whose size is not known *a priori*, is a difficulty encountered by all SVI algorithms since the 'goal posts' of the optimization move when the dataset changes size. Current research addresses this problem in similar inference frameworks [Broderick *et al.*, 2013]. This is particularly relevant for matrix factorization, since in many applications such as online retail, new users and items are continuously introduced. We address the problem of making good recommendations with new users and items in Chapter 7. Extending stochastic inference routines to this setting could yield a practical scalable probabilistic recommendation framework.

Chapter 6

Collaborative Preference Learning

Preference data is a common source of binary data. If we have preference judgements from multiple users, this data can be represented by a binary matrix. There are two main differences between preference matrices and the binary matrices in Chapter 5. First, entries can be unobserved because now there is a distinction between the two possible preferences and 'no observation'. Second, preference data has an additional anti-symmetry structure that should be leveraged by the model. In this chapter we develop probabilistic matrix factorization techniques, introduced in Chapter 5, and Gaussian process (GP), introduced in Chapter 3, to model preference data from multiple users. Furthermore, BALD is exploited for efficient active preference elicitation with this model.

Preference learning concerns making inferences from data consisting of pairs of items and corresponding binary labels indicating user preferences. This data arises in many contexts, including medical assistive technologies [Birlutiu *et al.*, 2010], graphical design [Brochu *et al.*, 2007] and recommendation systems [De Gemmis *et al.*, 2009]. This datatype is abundant because it may often be collected implicitly, such as from clickthrough logs [Joachims, 2002], hence preference learning is a rapidly growing sub-field of machine learning [Fürnkranz & Hüllermeier, 2010].

A popular approach to modelling preference data assumes the existence of a utility function $f(\mathbf{x}) : \mathcal{X} \mapsto \mathbb{R}$ that gives the 'value' of an item with feature vector $\mathbf{x}$; $f(\mathbf{x}_i) > f(\mathbf{x}_j)$ indicates that item i is preferred to item j. Bayesian methods can be used to learn f, for example, Chu & Ghahramani [2005b] model f with a GP prior. However, when data from many users is available, this method does not leverage similarities between the users' preferences because the GPs are fitted independently. For example, news preferences may be summarized by interests in latent themes such as sports, politics or technology. By identifying these common themes, at the individual level we only need to infer a user's relative interest in each of them.

Current probabilistic multi-user models are limited to one of two possible scenarios: i) user features are available and they are useful for prediction, or ii) no features are available. In particular, Bonilla *et al.* [2010] require that features are available for each user and assume that users with similar features have similar preferences. Birlutiu *et al.* [2010] perform single-user learning, ignoring user features, but tie information across users with a hierarchical prior. Additionally, these methods involve solving at least U GP problems, where U is the number of users. This cost is prohibitive even for modest U. Our model can address both i) and ii) by combining collaborative information with user features, if available. Furthermore, we perform scalable inference to handle problems with large U.

To do this, our model has two components: first, supervised GP utility function learning [Chu & Ghahramani, 2005b] is included to learn users' preferences. Second, unsupervised matrix factorization methods from collaborative filtering are included to learn similarities in users' behaviours without requiring access to user-specific features. However, if user features are available they may be useful, so the model can incorporate them also. Our method is based on a connection between preference learning and GP binary classification. We show that both problems are equivalent when a covariance function called the *preference kernel* is used. This kernel simplifies the inference process, allowing us to implement relatively complex models such as the proposed multi-user approach. Finally, in real scenarios, querying users may be costly and intrusive, so it is desirable to learn their preferences from as little data as possible. For this we exploit BALD to perform active preference elicitation.

The chapter is organized as follows. We derive the preference kernel in Section 6.1. In Section 6.2 we present the model. In Section 6.3 we introduce our inference algorithm which uses a hybrid of EP and VB. We show how BALD can be applied in this scenario in Section 6.4 and discuss related probabilistic models in Section 6.5. Section 6.6 contains our experiments. Conclusions and extensions follow.

6.1 The Preference Kernel

The problem of pairwise preference learning can be recast as a special case of binary classification. Consider two items i and j with corresponding feature vectors $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$.

In the pairwise preference learning problem we are given pairs of feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ and corresponding class labels $y \in \{-1, 1\}$ such that $y = 1$ if the user prefers item i to item j and $y = -1$ otherwise. The task is to predict the class label for a new pair of feature vectors not seen before. This problem can be addressed by introducing a latent preference function $f : \mathcal{X} \mapsto \mathbb{R}$ such that $f(\mathbf{x}_i) > f(\mathbf{x}_j)$ whenever the user prefers item i to item j and $f(\mathbf{x}_i) < f(\mathbf{x}_j)$ otherwise. If we assume that the evaluations of f are corrupted by additive Gaussian noise with zero mean and variance σ^2, we obtain the following likelihood for f given $\mathbf{x}_i$, $\mathbf{x}_j$ and y,

$$p(y|\mathbf{x}_i, \mathbf{x}_j, f) = \Phi\left(\frac{f(\mathbf{x}_i) - f(\mathbf{x}_j)}{\sqrt{2}\sigma^2} y\right) , \tag{6.1}$$

where $\Phi(\cdot)$ is the probit function (standard Gaussian c.d.f.). As in GP classification, we may assume without loss of generality that $\sqrt{2}\sigma^2 = 1$. A Bayesian model for preference learning is specified by combining the likelihood function in (6.1) with a GP prior on f: $f \sim GP(\mu, k)$. The posterior for f may then be used to predict the user's preferences on new pairs of items.

Note, however, that the likelihood in (6.1) only depends on the difference between $f(\mathbf{x}_i)$ and $f(\mathbf{x}_j)$. If we define $g : \mathcal{X}^2 \mapsto \mathbb{R}$ as a new latent function $g(\mathbf{x}_i, \mathbf{x}_j) = f(\mathbf{x}_i) - f(\mathbf{x}_j)$, we may recast the inference problem in terms of g and forget about f. When the evaluation of g is contaminated with standard Gaussian noise, the likelihood for g given $\mathbf{x}_i$, $\mathbf{x}_j$ and y is

$$p(y|\mathbf{x}_i, \mathbf{x}_j, g) = \Phi[g(\mathbf{x}_i, \mathbf{x}_j)y] . \tag{6.2}$$

Since g is obtained from f via a linear operation, the GP prior over f induces a GP prior over g. The mean μ_{pref} and covariance function k_{pref} of the GP on g can be computed from the mean and covariance of the GP on f as

$$\begin{aligned}
\mu_{pref}(\mathbf{x}_i, \mathbf{x}_j) &= \mathbb{E}\left[g(\mathbf{x}_i, \mathbf{x}_j)\right] \\
&= \mathbb{E}\left[f(\mathbf{x}_i) - f(\mathbf{x}_j)\right] \\
&= \mu(\mathbf{x}_i) - \mu(\mathbf{x}_j) ,
\end{aligned}$$

and

$$\begin{aligned}
k_{pref}((\mathbf{x}_i,\mathbf{x}_j),(\mathbf{x}_k,\mathbf{x}_l)) &= \mathrm{Cov}[g(\mathbf{x}_i,\mathbf{x}_j),g(\mathbf{x}_k,\mathbf{x}_l)] \\
&= \mathrm{Cov}\left[\left(f(\mathbf{x}_i)-f(\mathbf{x}_j)\right),\left(f(\mathbf{x}_k)-f(\mathbf{x}_l)\right)\right] \\
&= \mathbb{E}\left[\left(f(\mathbf{x}_i)-f(\mathbf{x}_j)\right)\cdot\left(f(\mathbf{x}_k)-f(\mathbf{x}_l)\right)\right] \\
&\quad -\left(\mu(\mathbf{x}_i)-\mu(\mathbf{x}_j)\right)\left(\mu(\mathbf{x}_k)-\mu(\mathbf{x}_l)\right) \\
&= k(\mathbf{x}_i,\mathbf{x}_k)+k(\mathbf{x}_j,\mathbf{x}_l)-k(\mathbf{x}_i,\mathbf{x}_l)-k(\mathbf{x}_j,\mathbf{x}_k)\,.
\end{aligned}$$

We call k_{pref} the *preference kernel*. Similar kernels have been derived for large margin classifiers [Fürnkranz & Hüllermeier, 2010], however, to our knowledge, this preference kernel has not been used previously in GP-based models.

6.1.1 Properties of the Preference Kernel

Kernel functions must be positive semi-definite. Since the preference kernel, k_{pref}, is constructed from the covariance of a stochastic process it is guaranteed to have this property. The preference kernel also respects the anti-symmetry property of preference learning. The prior correlation between $g(\mathbf{x}_i,\mathbf{x}_j)$ and $g(\mathbf{x}_j,\mathbf{x}_i)$ is

$$\begin{aligned}
&\mathrm{Corr}(g(\mathbf{x}_i,\mathbf{x}_j),g(\mathbf{x}_j,\mathbf{x}_i)) = \frac{k_{pref}((\mathbf{x}_i,\mathbf{x}_j),(\mathbf{x}_j,\mathbf{x}_i))}{\sqrt{k_{pref}((\mathbf{x}_i,\mathbf{x}_j),(\mathbf{x}_i,\mathbf{x}_j))}\sqrt{k_{pref}((\mathbf{x}_j,\mathbf{x}_i),(\mathbf{x}_j,\mathbf{x}_i))}} \\
&\quad= \frac{k(\mathbf{x}_i,\mathbf{x}_j)+k(\mathbf{x}_j,\mathbf{x}_i)-k(\mathbf{x}_i,\mathbf{x}_i)-k(\mathbf{x}_j,\mathbf{x}_j)}{\sqrt{k(\mathbf{x}_i,\mathbf{x}_i)+k(\mathbf{x}_j,\mathbf{x}_j)-k(\mathbf{x}_i,\mathbf{x}_j)-k(\mathbf{x}_j,\mathbf{x}_i)}\sqrt{k(\mathbf{x}_j,\mathbf{x}_j)+k(\mathbf{x}_i,\mathbf{x}_i)-k(\mathbf{x}_j,\mathbf{x}_i)-k(\mathbf{x}_i,\mathbf{x}_j)}} \\
&\quad= -1\,,
\end{aligned}$$

where we have assumed that $\mu_{\text{pref}} = 0$ to simplify the derivations. This shows that the value of $g(\mathbf{x}_i,\mathbf{x}_j)$ is perfectly anti-correlated with the value of $g(\mathbf{x}_j,\mathbf{x}_i)$ under the prior. With a zero mean function $g(\mathbf{x}_i,\mathbf{x}_j) = -g(\mathbf{x}_j,\mathbf{x}_i)$, $\forall \mathbf{x}_i,\mathbf{x}_j$, which is the desired anti-symmetry property for modelling binary preferences.

Note also that the preference kernel respects transitivity between pairwise item preferences. Since $g(\mathbf{x}_i,\mathbf{x}_j) = f(\mathbf{x}_i)-f(\mathbf{x}_j)$, we have that if $g(\mathbf{x}_i,\mathbf{x}_j) > 0$ then $f(\mathbf{x}_i) > f(\mathbf{x}_j)$ and if $g(\mathbf{x}_j,\mathbf{x}_k) > 0$ then $f(\mathbf{x}_j) > f(\mathbf{x}_k)$, so $f(\mathbf{x}_i) > f(\mathbf{x}_k)$. Therefore, if $g(\mathbf{x}_i,\mathbf{x}_j) > 0$ and $g(\mathbf{x}_j,\mathbf{x}_k) > 0$ then $g(\mathbf{x}_i,\mathbf{x}_k) > 0$.

The original preference likelihood function in (6.1) is more complicated than likelihood functions used in standard regression or classification. Thus, previous GP-based preference models have used relatively simple approximate inference algorithms, such as the Laplace approximation [Bonilla *et al.*, 2010; Chu & Ghahramani, 2005b]. The preference kernel moves the additional structure in preference learning from the likelihood function into the prior. The combination of the new likelihood in Equation (6.2) with a GP prior based on the preference kernel allows us to transform pairwise preference learning into binary classification. This means that state-of-the-art algorithms for GP binary classification, such as expectation propagation, can be applied directly to preference learning. Thus, the preference kernel allows us to implement complex methods such as the following multi-user approach.

6.2 Multi-User Preference Learning

Consider I items with feature vectors $\mathbf{x} \in \mathcal{X}$. The single-user approach to preference learning assumes an independent latent function for each of U users, $g_u(\mathbf{x}, \mathbf{x}') : \mathcal{X}^2 \mapsto \mathbb{R}$. We approach the multi-user problem by assuming a common structure in these user latent functions. In particular, we assume a set of D *shared* latent functions, $h_d(\mathbf{x}, \mathbf{x}') : \mathcal{X}^2 \mapsto \mathbb{R}$, where $D \ll U$. The user latent functions are generated using a linear combination of these shared functions,

$$g_u(\mathbf{x}_i, \mathbf{x}_j) = \sum_{d=1}^{D} w_{u,d} h_d(\mathbf{x}_i, \mathbf{x}_j) \,, \tag{6.3}$$

where $w_{u,d} \in \mathbb{R}$ is the weight given to function h_d for user u. We place a GP prior over the shared latent functions h_d using the preference kernel described in the previous section. This allows different users' preferences to share some common structure represented by the shared latent functions. This assumption results in a matrix factorization dimensionality reduction as is common in collaborative filtering.

We extend this model to the case where, for each user u, there is a feature vector $\mathbf{u}_u$ containing relevant information about the user. We denote the set of all the users' feature vectors as $\mathbf{U} = \{\mathbf{u}_1, \dots, \mathbf{u}_U\}$. The user features are incorporated by placing a separate GP prior over each user's weights. That is, we replace the scalars $w_{u,d}$ in Equation (6.3) with functions $w'_d(\mathbf{u}_u)$. These weight functions describe the contribution of shared latent function h_d to the user latent function g_u as a function of the user feature vector $\mathbf{u}_u$.

In the multi-user setting we have a set of P *pairs* of items evaluated by the users, where $P \leq I(I-1)/2$ (the maximum number of item pairs). Denote a preference judgement as $y_{i,u}$, for $i \in \{1, \dots, P\}$, $u \in \{1, \dots, U\}$, where $y_{i,u} = 1$, indicates that user u prefers the first item in pair i to the second and $y_{i,u} = -1$ otherwise. Denote the set of user/item-pair indices for which we have observed preference judgements as $\mathcal{D}$. The complete data consists of the set of feature vectors for the users $\mathbf{U}$ (if available), features for the items $\mathbf{X}$, and the preferences $\{y_{u,i}\}_{(i,u)\in\mathcal{D}}$.

6.2.1 Probabilistic Description of the Model

To predict preferences on unseen item pairs we cast the model into a probabilistic framework. Let $\mathbf{G}$ be a real valued $U \times P$ 'user-function' matrix, where each row corresponds to a particular user's latent function. That is, the entry in the u-th column and i-th row is $g_{u,i} = g_u(\mathbf{x}_{\alpha(i)}, \mathbf{x}_{\beta(i)})$ where $\alpha(i)$ and $\beta(i)$ denote respectively the first and second item in the i-th pair. Let $\mathbf{H}$ be a $D \times P$ 'shared-function' matrix, where each row represents the shared latent functions, that is, the entry in the d-th row and i-th column is $h_{d,i} = h_d(\mathbf{x}_{\alpha(i)}, \mathbf{x}_{\beta(i)})$. Finally, we introduce the $U \times D$ weight matrix $\mathbf{W}$, where each row contains a user's weights. The entry in the u-th row and d-th column is $w_{d,u} = w_d(\mathbf{u}_u)$. Equation (6.3) can now be written as a matrix factorization $\mathbf{G} = \mathbf{WH}$.

Let $\mathbf{Y}$ be the $U \times P$ binary target matrix given by $\mathbf{Y} = \text{sign}[\mathbf{G} + \mathbf{E}]$, where $\mathbf{E}$ is a $U \times P$ noise matrix with entries sampled i.i.d. from a standard Gaussian. The function "sign[$\cdot$]" retains only the sign of the elements in a matrix. Let $\mathbf{Y}^{\mathcal{D}}$ and $\mathbf{G}^{\mathcal{D}}$ represent the elements of $\mathbf{Y}$ and $\mathbf{G}$ for which we have observed preferences. Then, the likelihood for $\mathbf{G}^{\mathcal{D}}$ given the observations $\mathbf{Y}^{\mathcal{D}}$, and conditional distribution for $\mathbf{G}^{\mathcal{D}}$ given $\mathbf{H}$ and $\mathbf{W}$ are

$$p(\mathbf{Y}^{\mathcal{D}}|\mathbf{G}^{\mathcal{D}}) = \prod_{(u,i)\in\mathcal{D}} \Phi(t_{u,i} g_{u,i})\,, \quad p(\mathbf{G}^{\mathcal{D}}|\mathbf{W},\mathbf{H}) = \prod_{(u,i)\in\mathcal{D}} \delta(g_{u,i} - \mathbf{w}_u \mathbf{h}_{\cdot,i})$$

respectively, where $\mathbf{w}_u$ is the u-th row in $\mathbf{W}$, $\mathbf{h}_{\cdot,i}$ is the i-th column in $\mathbf{H}$ and δ is the Dirac delta function.

We now select the priors for $\mathbf{W}$ and $\mathbf{H}$. We put GP priors on each function $w_1, \ldots, w_D$ with zero mean and some covariance function. Let $\mathbf{K}_{\text{users}}$ be the $U \times U$ covariance matrix for the entries in each column of $\mathbf{W}$. Then

$$p(\mathbf{W}|\mathbf{U}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{w}_{\cdot,d}; \mathbf{0}, \mathbf{K}_{\text{users}})\,, \tag{6.4}$$

where $\mathbf{w}_{\cdot,d}$ is the d-th column in $\mathbf{W}$. If user features are unavailable, we use independent standard Gaussian priors on each element in $\mathbf{W}$, so $\mathbf{K}_{\text{users}}$ becomes the identity matrix. Lastly, we put a GP prior on each shared latent function $h_1, \ldots, h_D$ with zero mean and covariance function given by a preference kernel. Let $\mathbf{K}_{\text{items}}$ be the $P \times P$ preference covariance matrix for the observed item pairs. The prior for $\mathbf{H}$ is

$$p(\mathbf{H}|\mathbf{X}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{h}_d; \mathbf{0}, \mathbf{K}_{\text{items}}), \tag{6.5}$$

where $\mathbf{h}_j$ is the j-th row in $\mathbf{H}$. The resulting posterior for the latent variables $\mathbf{W}$, $\mathbf{H}$ and $\mathbf{G}^{\mathcal{D}}$ is

$$p(\mathbf{W}, \mathbf{H}, \mathbf{G}^{\mathcal{D}}|\mathbf{Y}^{\mathcal{D}}, \mathbf{X}, \mathbf{U}) = \frac{p(\mathbf{Y}^{\mathcal{D}}|\mathbf{G}^{\mathcal{D}})p(\mathbf{G}^{\mathcal{D}}|\mathbf{W}, \mathbf{H})p(\mathbf{W}|\mathbf{U})p(\mathbf{H}|\mathbf{X})}{p(\mathbf{Y}^{\mathcal{D}}|\mathbf{X}, \mathbf{U})}, \tag{6.6}$$

where $p(\mathbf{Y}^{\mathcal{D}}|\mathbf{X}, \mathbf{U})$ is the (intractable) marginal likelihood, or model evidence.

6.2.2 The Predictive Distribution

Given a new item pair with index $P+1$, we compute the predictive distribution for the preference of the u-th user on this pair by integrating over the posterior on parameters $\mathbf{H}, \mathbf{W}$ and $\mathbf{G}^{\mathcal{D}}$ as

$$\begin{aligned} p(y_{u,P+1}|\mathbf{Y}^{\mathcal{D}}, \mathbf{X}) = \int & p(y_{u,P+1}|g_{u,P+1})p(g_{u,P+1}|\mathbf{w}_u, \mathbf{h}_{\cdot,P+1}) \\ & p(\mathbf{h}_{\cdot,P+1}|\mathbf{H}, \mathbf{X})p(\mathbf{H}, \mathbf{W}, \mathbf{G}^{\mathcal{D}}|\mathbf{Y}^{\mathcal{D}}, \mathbf{X}, \mathbf{U})\, d\mathbf{H}\, d\mathbf{W}\, d\mathbf{G}^{\mathcal{D}}, \end{aligned} \tag{6.7}$$

where

$$\begin{aligned} p(y_{u,P+1}|g_{u,P+1}) &= \Phi(y_{u,P+1}g_{u,P+1}), \\ p(g_{u,P+1}|\mathbf{w}_u, \mathbf{h}_{\cdot,P+1}) &= \delta(g_{u,P+1} - \mathbf{w}_u\mathbf{h}_{\cdot,P+1}), \\ p(\mathbf{h}_{\cdot,P+1}|\mathbf{H}, \mathbf{X}) &= \prod_{d=1}^{D} \mathcal{N}(h_{d,P+1}; \mathbf{k}_\star^\top \mathbf{K}_{\text{items}}^{-1}\mathbf{h}_d, k_\star - \mathbf{k}_\star^\top \mathbf{K}_{\text{items}}^{-1}\mathbf{k}_\star). \end{aligned}$$

$k_\star$ is the prior variance of $h_d\left(\mathbf{x}_{\alpha(P+1)}, \mathbf{x}_{\beta(P+1)}\right)$ and $\mathbf{k}_\star$ is a P-dimensional vector that contains the prior covariances between $h_d\left(\mathbf{x}_{\alpha(P+1)}, \mathbf{x}_{\beta(P+1)}\right)$ and $h_d\left(\mathbf{x}_{\alpha(1)}, \mathbf{x}_{\beta(1)}\right), \ldots,$ $h_d\left(\mathbf{x}_{\alpha(P)}, \mathbf{x}_{\beta(P)}\right)$. The posterior (6.6) and predictive distribution (6.7) are intractable so approximations must be used. For this, we use a combination of EP and VB.

6.3 Hybrid EP-VB Inference

Approximate inference in our model is implemented using a combination of expectation propagation (EP) [Minka, 2001b] and variational Bayes (VB) [Attias, 1999; Ghahramani

& Beal, 2000]. We choose EP as the core inference routine since empirical studies show that EP obtains state-of-the-art performance in the related problem of GP binary classification [Nickisch & Rasmussen, 2008]. We first give a brief primer on EP.

6.3.1 Primer on Expectation Propagation

Expectation propagation is a deterministic algorithm for approximate Bayesian inference, originally developed in Minka [2001a]. Similar to VB, introduced in Section 5.4.1, the algorithm approximates an intractable distribution over variables θ, $p(\theta)$, with a simpler, factorized distribution $q(\theta)$. The EP algorithm matches the approximation q to the posterior p by attempting to minimizing the KL divergence between the two, $\mathrm{KL}[p(\theta)||q(\theta)]$, with respect to the parameters of q. Note that the direction of the KL is the reverse of that used in VB, Equation (5.6). For most models, with many parameters, minimizing $\mathrm{KL}[p(\theta)||q(\theta)]$ over the entire distribution q is intractable. Therefore, EP uses an iterative procedure.

For most models, the posterior distribution can be decomposed into a product of factors: $p(\theta) = \prod_a f_a(\theta)$. In EP, the posterior approximation q is decomposed into approximate factors $\hat{f}_a(\theta)$ that approximate the true factors $f_a(\theta)$. The approximate posterior is the re-normalized product of approximate factors $q(\theta) \propto \prod_a \hat{f}_a(\theta)$. EP iteratively refines each approximate factor $\hat{f}_a(\theta)$ by minimizing the following KL divergence,

$$\mathrm{KL}\left[f_a(\theta)q^{\setminus a}(\theta)||\hat{f}_a(\theta)q^{\setminus a}(\theta)\right] = \int f_a q^{\setminus a} \log \frac{f_a q^{\setminus a}}{\hat{f}_a q^{\setminus a}} + f_a q^{\setminus a} - \hat{f}_a q^{\setminus a} d\theta\,, \tag{6.8}$$

where $q^{\setminus a}(\theta)$ is the current approximation with the a-th term removed, $q^{\setminus a}(\theta) = q(\theta)/\hat{f}_a(\theta) \propto \prod_{b\neq a} \hat{f}_b(\theta)$. The form of the KL in Equation (6.8) accounts for the distributions being unnormalized. For exponential family distributions, optimizing Equation (6.8) corresponds to matching the expected sufficient statistics of distributions on either side of the KL. With a Gaussian approximate posterior this computation is equivalent to moment matching.

EP iterates over the approximate factors, minimizing (6.8) until convergence. This procedure does not guarantee to minimize the global KL divergence between $p(\theta)$ and $q(\theta)$, or even converge. However, in practice it has demonstrated strong empirical performance with many models and has become a popular inference algorithm. For thorough overview see Minka [2001a].

6.3.2 Inference for Collaborative Preference Learning

We describe our EP routine for our multi-user preference learning model. We approximate the posterior in (6.6) with fully factorized Gaussian distributions over all of the elements in $\mathbf{W}$, $\mathbf{H}$ and $\mathbf{G}^{\mathcal{D}}$,

$$q(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}) = \left[\prod_{u=1}^{U}\prod_{d=1}^{D}\mathcal{N}(w_{ud}; m^w_{u,d}, v^w_{u,d})\right]\left[\prod_{d=1}^{D}\prod_{i=1}^{P}\mathcal{N}(h_{d,i}; m^h_{d,i}, v^h_{d,i})\right] \left[\prod_{(u,i)\in\mathcal{D}}\mathcal{N}(g_{u,i}; m^g_{u,i}, v^g_{u,i})\right], \tag{6.9}$$

where $m^w_{u,d}$, $v^w_{u,d}$, $m^h_{d,i}$, $v^h_{d,i}$, $m^g_{u,i}$, and $v^g_{u,i}$ are free parameters to be determined by EP. The superscripts w, h and g indicate the random variables described by these parameters.

The joint distribution $p(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}, \mathbf{Y}^{\mathcal{D}}|\mathbf{X}, \mathbf{U})$ consists of four factors $f_1, \ldots, f_4$,

$$p(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}, \mathbf{Y}^{\mathcal{D}}|\mathbf{X}, \mathbf{U}) = \prod_{a=1}^{4} f_a(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}),$$

with correspondences $f_1(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}) = p(\mathbf{Y}^{\mathcal{D}}|\mathbf{G}^{\mathcal{D}})$, $f_2(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}) = p(\mathbf{G}^{\mathcal{D}}|\mathbf{W}, \mathbf{H})$, $f_3(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}) = p(\mathbf{W}|\mathbf{U})$ and $f_4(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}) = p(\mathbf{H}|\mathbf{X})$. EP approximates these exact factors by approximate factors $\hat{f}_1(\mathbf{W}, \mathbf{H}, \mathbf{G}^{\mathcal{D}}), \ldots, \hat{f}_4(\mathbf{W}, \mathbf{H}, \mathbf{G}^{\mathcal{D}})$ that have the same functional form as q,

$$\hat{f}_a(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}) = \left[\prod_{u=1}^{U}\prod_{d=1}^{D}\mathcal{N}(w_{ud}|\hat{m}^{a,w}_{u,d}, \hat{v}^{a,w}_{u,d})\right]\left[\prod_{d=1}^{D}\prod_{i=1}^{P}\mathcal{N}(h_{d,i}|\hat{m}^{a,h}_{d,i}, \hat{v}^{a,h}_{d,i})\right] \left[\prod_{(u,i)\in\mathcal{D}}\mathcal{N}(g_{u,i}|\hat{m}^{a,g}_{u,i}, \hat{v}^{a,g}_{u,i})\right]\hat{s}_a, \tag{6.10}$$

where $\hat{m}^{a,w}_{u,d}$, $\hat{v}^{a,w}_{u,d}$, $\hat{m}^{a,h}_{d,i}$, $\hat{v}^{a,h}_{d,i}$, $\hat{m}^{a,g}_{u,i}$, $\hat{v}^{a,g}_{u,i}$ and $\hat{s}_a$ are free parameters of the approximate factors. As described in Section 6.3.1, q is obtained from the normalized product $\prod_a \hat{f}_a$. The first step is to initialize $\hat{f}_1, \ldots, \hat{f}_4$ and q to be uniform. Then EP iteratively refines each $\hat{f}_a$ by minimizing the KL divergence between $f_a q^{\backslash a}$ and $\hat{f}_a q^{\backslash a}$, $\mathrm{KL}[f_a q^{\backslash a} || \hat{f}_a q^{\backslash a}]$ with respect to the parameters of $\hat{f}_a$.

However, such KL minimization does not perform well for refining $\hat{f}_2$. This term corresponds to the matrix factorization $\mathbf{G} = \mathbf{WH}$. Using EP for this factor is likely to perform poorly. This is because the matrix factorization has many invariances; the solution is invariant to rotations, reflections or re-scalings of $\mathbf{W}$ and $\mathbf{H}$. This means that the posterior is multimodal. EP will average across the modes of the posterior (6.9), which will result in a poor overall solution [Bishop, 2006; Stern *et al.*, 2009]. Therefore we use VB to refine $\hat{f}_2$. As discussed in Chapter 5, VB is a popular inference routine for probabilistic matrix factorization because it will model just one of the modes of the posterior, which is sufficient for making good predictions. Therefore, instead of minimizing $\mathrm{KL}[q^{\setminus 2}f_2 \| q^{\setminus 2}\hat{f}_2]$ as is required by EP, the direction KL divergence is reversed, that is we minimize $\mathrm{KL}[q^{\setminus 2}\hat{f}_2 \| q^{\setminus 2}f_2]$.

EP iteratively refines all the approximate factors until convergence. After running inference we approximate the predictive distribution (6.7) by replacing the exact posterior with q. The approximate predictive distribution is

$$p(y_{u,P+1}|\mathbf{Y}^{\mathcal{D}},\mathbf{X},\mathbf{U}) \approx \Phi\left(\frac{y_{u,P+1}m^g_{u,P+1}}{\sqrt{v^g_{u,P+1}+1}}\right),$$

where

$$m^g_{u,P+1} = \sum_{d=1}^{D} m^w_{u,d} m^h_{d,P+1}\,,$$
$$v^g_{u,P+1} = \sum_{d=1}^{D} [m^w_{u,d}]^2 v^h_{d,P+1} + \sum_{d=1}^{D} v^w_{u,d}[m^h_{d,P+1}]^2 + \sum_{d=1}^{D} v^w_{u,d} v^h_{d,P+1}\,,$$

and $m^h_{d,P+1}$ and $v^h_{d,P+1}$ are given by

$$m^h_{d,P+1} = \mathbf{k}_\star^\top\left[\mathbf{K}_{\text{items}} + \mathrm{diag}[\hat{\mathbf{v}}^{h,2}_d]\right]^{-1}\hat{\mathbf{m}}^{h,2}_d\,,$$
$$v^h_{d,P+1} = k_\star - \mathbf{k}_\star^\top\left[\mathbf{K}_{\text{items}} + \mathrm{diag}[\hat{\mathbf{v}}^{h,2}_d]\right]^{-1}\mathbf{k}_\star\,,$$

where $\mathrm{diag}[\cdot]$ converts a vector into a diagonal matrix and $\hat{\mathbf{m}}^{h,2}_d$, $\hat{\mathbf{v}}^{h,2}_d$ are the vectors $\hat{\mathbf{m}}^{h,2}_d = (\hat{m}^{h,2}_{1,d},\ldots,\hat{m}^{h,2}_{P,d})^\top$ and $\hat{\mathbf{v}}^{h,2}_d = (\hat{v}^{h,2}_{1,d},\ldots,\hat{v}^{h,2}_{P,d})^\top$. Note that EP approximates the posterior with fully factorized Gaussians for each factor. However, to interpolate to the new item pairs we use the full GP prior over the user latent functions h_d. Therefore, when computing the EP approximation to the predictive distributions, we replace the approximate factor $\hat{f}_3$ corresponding to an uncorrelated prior over the item pairs with the full GP prior covariance matrix.

EP is also used to approximate the normalization constant in Equation (6.6) (the model evidence) with the integral of the product of all the approximate factors $\hat{f}_1, \ldots, \hat{f}_4$. Computing the model evidence with EP requires moment matching the 0th order moments of the distributions in Equation (6.8). With the VB routine for second factor $\hat{f}_2$ we use the variational lower bound to the evidence (the ELBO).

6.3.3 Algorithmic Details

Damping

Unlike VB, EP does not optimize a bound on the likelihood, and is not guaranteed to converge so may oscillate. This undesirable behaviour can be prevented by *damping* the EP updates [Minka & Lafferty, 2002]. Let $\hat{f}_a^{\text{new}}$ denote the value of the approximate factor that minimizes the KL in (6.8). Damping consists of using

$$\hat{f}_a^{\text{damp}} = \left[\hat{f}_a^{\text{new}}\right]^{\epsilon} \left[\hat{f}_a\right]^{(1-\epsilon)} , \tag{6.11}$$

instead of $\hat{f}_a^{\text{new}}$ to update the approximate factor, where $\hat{f}_a$ is the factor before the update. The parameter $\epsilon \in [0, 1]$ controls the degree of damping. $\epsilon = 1$ yields no damping and with $\epsilon = 0$ the factor $\hat{f}_a$ remains unchanged. To improve the converge of EP, we use a damping schedule recommended in Hernández-Lobato [2010] that uses an initial $\epsilon = 1$ and then progressively reduces ϵ by a constant multiplicative factor.

Refinement of $\hat{f}_2$

The specific computations required to refine the probit likelihood function $\hat{f}_1$ follow from those for GP classification [Rasmussen & Williams, 2005]. Refining $\hat{f}_3$ and $\hat{f}_4$ requires standard moment matching of a multivariate Gaussian to independent Gaussians.

For the second factor $\hat{f}_2$, we use VB in a similar manner to Stern *et al.* [2009]. To do this, we first marginalize $q^{\backslash 2} f_2$ with respect to $\mathbf{G}^{\mathcal{D}}$. This yields an auxiliary unnormalized distribution $s(\mathbf{W}, \mathbf{H})$ which can be computed analytically using

$$s(\mathbf{W}, \mathbf{H}) = \int \prod_{(u,i) \in \mathcal{D}} \delta[g_{u,i} - \mathbf{w}_u \mathbf{h}_{\cdot, z_{u,i}}] q^{\backslash 2}(\mathbf{G}^{\mathcal{D}}, \mathbf{W}, \mathbf{H}) \, d\mathbf{G}^{\mathcal{D}} . \tag{6.12}$$

Let $q_{\mathbf{W},\mathbf{H}}$ be the posterior approximation (6.9) after marginalizing out $\mathbf{G}^{\mathcal{D}}$. The parameters of $q_{\mathbf{W},\mathbf{H}}$, are then optimized by minimizing $\text{KL}[q_{\mathbf{W},\mathbf{H}} \| s]$. This corresponds to performing a VB matrix factorization with a Gaussian likelihood, for which we use the gradient descent algorithm described in Raiko *et al.* [2007]. After this, $\hat{f}_2$ is updated using the ratio of Gaussians, $\hat{f}_2 = q_{\mathbf{W},\mathbf{H}}/q^{\backslash 2}$.

6.3.4 Sparse GPs for Linear Computational Time

The cost to perform inference with GPs is cubic in the number of observations. In our case, refining the third factor $\hat{f}_3$ costs $\mathcal{O}(DU^3)$, where U is the number of users, and D is the number of shared latent functions. The cost to refine the fourth factor $\hat{f}_4$ is $\mathcal{O}(DP^3)$, where P is the number of observed item pairs. These cubic costs can be prohibitive. However, GP inference may be reduced to a cost linear in the number of observations with sparse approximations. We use the Fully Independent Training Conditional (FITC) approximation [Snelson & Ghahramani, 2006]. In essence, FITC channels covariance information from the full dataset through a small number of *pseudo-inputs* that can be located arbitrarily.

We reduce the costs of refining $\hat{f}_3$ and $\hat{f}_4$ by approximating $\mathbf{K}_{\text{users}}$ and $\mathbf{K}_{\text{items}}$ in Equations (6.4) and (6.5) using FITC. Under this approximation, an $N \times N$ covariance matrix $\mathbf{K}$ resulting from the evaluation of a covariance function at N locations is approximated by a low rank matrix $\mathbf{K}' = \mathbf{Q}+\text{diag}(\mathbf{K}-\mathbf{Q})$, where $\mathbf{Q} = \mathbf{K}_{NN_0}\mathbf{K}_{N_0N_0}^{-1}\mathbf{K}_{NN_0}^{\top}$. The $N_0 \times N_0$ matrix $\mathbf{K}_{N_0N_0}$ contains evaluations of the covariance function at only $N_0 < N$ pseudo-inputs and the $N \times N_0$ matrix $\mathbf{K}_{NN_0}$ contains the covariances between the original data and the pseudo-inputs.

This approximation allows us to refine $\hat{f}_3$ and $\hat{f}_4$ in $\mathcal{O}(DU_0^2U)$ and $\mathcal{O}(DP_0^2P)$ operations, where U_0 and P_0 are the number of pseudo-inputs for the users and the item pairs respectively. We choose U_0 and P_0 to balance cost and accuracy. The calculations required to implement EP and to approximate the predictive distribution and model evidence with FITC follow from those in Lázaro Gredilla [2010]; Naish-Guzman & Holden [2007]. Both of the other factors, $\hat{f}_1$ and $\hat{f}_2$, have linear cost in the total number of datapoints, $\mathcal{O}(|\mathcal{D}|)$. Without FITC, this cost is dominated by the cubic cost of the GPs, and the total cost of our inference routine is $\mathcal{O}(|\mathcal{D}| + DU^3 + DP^3)$. With FITC, the total cost is linear in the number or users, item pairs and datapoints, $\mathcal{O}(|\mathcal{D}| + DU_0^2U + DP_0^2P)$.

6.4 Active Preference Elicitation

In many applications one can present users with new item pairs to elicit new preference judgements. However, often users will only provide limited feedback, therefore it is desirable to collect data that will yield maximal information about their preferences. For this we use the active learning techniques developed in Chapters 2 and 3.

With the multi-user preference model the parameter that we are interested in learning is the user latent function g_u. We may apply BALD, introduced in Chapter 2, directly to this task. In particular, following Equation (2.10), we suggest items pairs $(\mathbf{x}, \mathbf{x}')$ to a user to maximize the information gain about g_u,

$$\mathcal{U}(\mathbf{x}, \mathbf{x}') = \mathrm{H}[p(g_u|\mathbf{Y}^{\mathcal{D}})] - \mathbb{E}_{p(y|\mathbf{x},\mathbf{x}',\mathbf{Y}^{\mathcal{D}})}\mathrm{H}[p(g_u|y, \mathbf{x}, \mathbf{x}', \mathbf{Y}^{\mathcal{D}})]\,, \tag{6.13}$$

$$= \mathrm{H}[p(y|\mathbf{x}, \mathbf{x}', \mathbf{Y}^{\mathcal{D}})] - \mathbb{E}_{p(g_u|\mathbf{Y}^{\mathcal{D}})}\mathrm{H}\left[p(y|\mathbf{x}, \mathbf{x}', g_u)\right]\,. \tag{6.14}$$

As in GP regression and classification, rearranging from the direct computation of posterior entropies in Equation (6.13) to the formulation used by BALD (6.14) is necessary to compute information gain about the entire latent function accurately. Since inference with our multi-user model requires running the EP-VB routine, it is particularly important to reduce the number of posterior updates from $2P^{\text{new}}$ to 1 per sample, where P^{new} is the number of possible new item pairs, as permitted by the BALD rearrangement.

Since we have used the preference kernel to reduce the likelihood function for preference learning to the probit likelihood used in classification models, we may use the techniques proposed in Section 3.2 for active GPC. Therefore, we apply Equation (3.8) directly to compute Equation (6.14) with our preference model.

6.5 Related Multi-User Models

We describe the differences between our approach and two alternative multi-user GP preference learning models described in Birlutiu *et al.* [2010]; Bonilla *et al.* [2010]. We compare to them empirically in Section 6.6.

Model of Birlutiu *et al.* As in the single-task preference learning model, this model fits a different GP to data from each user. However, the different classifiers are now connected by a hierarchical GP prior over the latent user-preference functions g_u. This model does not include user-features. The parameters of the hierarchical prior (the mean and covariance) are optimized using the expectation-maximization (EM) algorithm [Dempster *et al.*, 1977]. In the E-step the GP posterior mean and covariance over the item pairs is computed for each user. Then, in the M-step the mean and covariance of the hierarchical prior are adjusted to maximize the log likelihood of all of the data.

This model is flexible because it learns the full covariance matrix for each of the

P item pairs for each user. However, this causes the EM routine to be computationally expensive since each iteration requires the inversion of U covariance matrices of dimension $P \times P$, this results in $\mathcal{O}(UP^3)$ computations. The cost of our equivalent model, that does not incorporate user features, is $\mathcal{O}(DP^3)$.[1] Our model is significantly cheaper because $D \ll U$. In our implementation, to reduce the computational burden of the model in Birlutiu *et al.* [2010], we limit the EM algorithm to 20 iterations. In our experiments, increasing this number did not lead to improvements in the predictive performance of this method.

Birlutiu *et al.* [2010] do not use the preference kernel. Without this, their cost is reduced to $\mathcal{O}(UI^3)$, where I is the total number of items. However, they still need to solve U GP problems. Furthermore, GP inference with their likelihood is harder, so they resort to a sampling-based approximation. For a fair comparison of the underlying models, we implemented their method using the preference kernel and EP.

Model of Bonilla *et al.* This model assumes that user features are available and users with similar characteristics have similar preferences. This model uses a single large latent function g which depends on both the item features $\mathbf{x}$ and user features $\mathbf{u}$. With the preference kernel, the likelihood function for their model is

$$p(y|\mathbf{x}_i, \mathbf{x}_j, \mathbf{u}_u, g) = \Phi(g(\mathbf{x}_i, \mathbf{x}_j, \mathbf{u}_u)y)\,. \tag{6.15}$$

A GP prior is used for g, and its covariance function is constructed using a product kernel,

$$k_{\text{Bonilla}}((\mathbf{u}_u, \mathbf{x}_i, \mathbf{x}_j), (\mathbf{u}_s, \mathbf{x}_k, \mathbf{x}_l)) = k_{\text{users}}(\mathbf{u}_u, \mathbf{u}_s) k_{\text{pref}}((\mathbf{x}_i, \mathbf{x}_j), (\mathbf{x}_k, \mathbf{x}_l))\,, \tag{6.16}$$

where k_{pref} is the preference kernel and k_{users} is a covariance function for user features. Therefore k_{users} encourages the model to assign similar preferences to users with similar feature vectors. The preference kernel allows us to do efficient approximate inference in this model using a standard EP implementation for GPC. However, the computational cost of this method is high, it has cubic cost in the *total* number of observations, $\mathcal{O}(|\mathcal{D}|^3)$. Our model, with user features, has a much lower cost of $\mathcal{O}(|\mathcal{D}| + DU^3 + DP^3)$.

Again, Bonilla *et al.* [2010] do not use the preference kernel, but instead use the standard GP preference likelihood in Equation (6.1). In this case the cost is lower, but still large at $O(|\mathcal{I}|^3)$, where $|\mathcal{I}|$ is the sum of number of unique items evaluated by each user. Inference with this likelihood is more complex so the authors use the

[1] Without the FITC approximation, which is not used in Birlutiu *et al.* [2010].

Laplace approximation. EP normally outperforms Laplace in binary GPC [Nickisch & Rasmussen, 2008], so to compare the underlying models we implemented their model with the preference kernel and EP. In practice, with cubic cost in the total number of observations, the method in Bonilla *et al.* [2010] is infeasible with more than a few hundred users (they report experiments with only 50 training users). Additionally, this model cannot be used if user features are absent, and if users with similar features do not have similar preferences one obtains poor predictive performance as we observe in our experiments.

6.6 Experiments and Discussion

We evaluated the performance of our collaborative preference model with BALD in experiments on five datasets. There are not many public multi-user preference learning datasets available, so some of the datasets were converted from multi-user regression. The datasets were:

i) Synthetic This data was generated from the assumed multi-user preference learning model with $D = 5$.

ii) Jura This dataset contains concentration measurements for 7 heavy metals in soils of the Swiss Jura region at 359 locations [Atteia *et al.*, 1994]. We converted this into preferences by first standardizing the measurements of each metal. Then the standardized measurements were used as utility values to generate preferences between pairs of heavy metals at each location. The locations correspond to 'users' and metals to 'items'. The item features were generated using the standardized measurements at 20 randomly held-out locations. We used the x and y coordinates for the measurements, and rock and land types for the item features.

iii) MovieLens This dataset contains 1 million ratings from 6,000 users on 4,000 movies, available at `http://grouplens.org/datasets/movielens/`. 10 movies were sampled from the 50 movies with most ratings. We selected users with at least 7 ratings on these movies. The missing ratings were filled in using a nearest neighbour method. The ratings were used as utility values to generate preferences between movies. The user features included gender, age and occupation. The item features are genres such as action, comedy or adventure.

iv) Sushi This dataset contains complete rankings of 10 sushi types by 5,000 users [Kamishima *et al.*, 2005], where each sushi includes features such as style, group, heaviness, consumption frequency etc. The user features include gender, age and geographical information.

v) Election This dataset contains the votes for 8 political parties (items) in 650 constituencies (users) in the 2010 general elections in the UK, available from `http://www.electoralcommission.org.uk/`. We kept constituencies with votes for more than 6 parties. Missing votes were estimated using a nearest neighbour method. We generated 'item' feature vectors as the votes from 20 randomly held-out constituencies. The 'user' features were the map coordinates of each constituency's centroid.

6.6.1 Comparison to Other Multi-User Models

Alternative Models

We compared the two versions of our collaborative preference (CP) model. The first (CPU) takes into account the available user features, as described in Section 6.2. The second (CP) ignores these features by replacing $\mathbf{K}_{\text{users}}$ in Equation (6.4) with the identity matrix. We compare to the two multi-user preference models described in Section 6.5, Birlutiu *et al.* [2010] (BI) and Bonilla *et al.* [2010] (BO). Finally, we consider a single user baseline (SU) that fits an independent GP classifier independently to the data of each user.

Experimental Procedure

Due to the high computational cost of BI and BO, to compare to these methods we must subsample the datasets, keeping only 100 users. The datasets were split randomly into training and test sets of item pairs, where the training sets contained 20 pairs per user in Sushi, MovieLens and Election, 15 pairs in Jura and 30 in Synthetic. The remaining data was used to evaluate predictive performance.

In CPU and CP, we set the number of latent functions to $D = 20$. We set the kernel lengthscales to be equal to the median distance between feature vectors. This lead to good empirical performance for most methods. An exception is BO, where the kernel hyperparameters are tuned to some held-out data using automatic relevance determination. In our model, we can also estimate the kernel lengthscales by maximizing the EP approximation of the model evidence, see experiments below. This approach may be used when it is necessary to tune the lengthscale parameters to the data. In CPU

Dataset	**CPU**	**CP**	**BI**	**BO**	**SU**
Synthetic	0.162	0.180	0.175	**0.157**	0.226
Sushi	0.171	0.163	**0.160**	0.266	0.187
MovieLens	0.182	**0.166**	0.168	0.302	0.217
Election	0.199	0.123	**0.077**	0.401	0.300
Jura	0.159	**0.153**	**0.153**	0.254	0.181

Table 6.1: Average test error with 100 users.

Dataset	**CPU**	**CP**	**BI**	**BO**	**SU**
Synthetic	7.793	9.498	22.524	311.574	0.927
Sushi	5.694	4.307	20.028	215.136	0.817
MovieLens	5.313	4.013	19.366	69.048	0.604
Election	13.134	12.408	20.880	120.011	0.888
Jura	3.762	2.404	15.234	88.502	0.628

Table 6.2: Training times (seconds) with 100 users.

we used $U_0 = 25$ pseudo inputs for approximating $\mathbf{K}_{\text{users}}$. These pseudo inputs were selected randomly from the set of available datapoints. Similarly, in CP and CPU, we used $P_0 = 25$ pseudo inputs for approximating $\mathbf{K}_{\text{items}}$, except in the Jura and Election datasets (which contain fewer items) where we used $P_0 = 15$. The results are not sensitive to the number of pseudo inputs, provided the number is not excessively low. The results were averaged over 25 repeats of the entire routine, including the random data partitioning.

Results

Error was measured as the fraction of held out preferences incorrectly predicted by the models. Average test errors are contained in Table 6.1. Those highlighted in bold are statistically better to those not highlighted (according to a paired t test).

Overall, CP and CPU outperform SU and BO, and break even with BI; this last result is notable as BI learns the full mean and covariance structure for all users, our model uses only a few latent dimensions, which provides the key to scaling to many more users. CP outperforms CPU in all cases except on the Synthetic dataset, where we know that the features correlate with the preferences. Therefore, it appears that in these real-world datasets, users with similar features do not have similar preferences, so correlating the behaviour of users with similar features is detrimental. The unsupervised, collaborative learning of similarities in user preferences is more useful for making predictions than the user features. This also explains the poor overall performance of

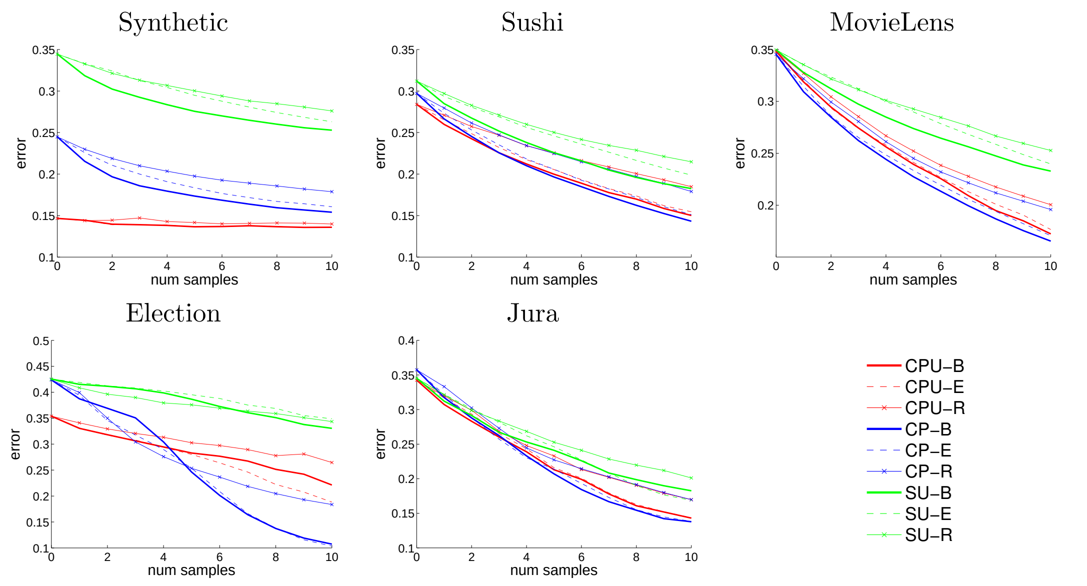

Figure 6.1: Average test error for CPU, CP and SU, using the strategies BALD (-B), entropy (-E) and random (-R) for active learning.

BO.

The Synthetic dataset was generated using a latent dimension of $D = 5$, but CPU and CP still obtain good results using $D = 20$. Hence, CPU and CP appear to be robust to overfitting, over-estimation of D does not harm predictive performance. This automatic pruning of unnecessary degrees of freedom is common in methods based on variational Bayes [MacKay, 2001].

Wall-clock run times are presented in Table 6.2. The entries for BO do not include the time spent tuning the kernel hyper-parameters with this method. CP and CPU are faster than BO and BI. The FITC approximation adds a large multiplicative constant to the cost of CP and CPU, so for larger datasets the gains are much greater.

6.6.2 Active Learning on Large Datasets

We now evaluate BALD for active preference elicitation on all the available users from each dataset, up to a maximum of 1000 users. We compare CPU, CP, and SU using BALD (-B), Maximum Entropy Sampling (-E) and random sampling (-R). For each user the available preferences were split randomly into training, pool and test sets with 5, 35, 5 pairs respectively in Synthetic, Sushi and MovieLens, 3, 22, 3 pairs in Election and 3, 15, 3 pairs in Jura. Each model was fitted using the training sets and

Dataset	CPU-B	CPU-E	CPU-R	CP-B	CP-E	CP-R	SU-B	SU-E	SU-R
Synthetic	**0.135**	**0.135**	0.139	**0.153**	0.160	0.173	**0.249**	0.259	0.268
Sushi	**0.148**	0.153	0.178	**0.144**	0.151	0.176	**0.179**	0.197	0.212
MovieLens	**0.170**	0.176	0.199	**0.163**	0.170	0.195	**0.225**	0.235	0.248
Election	0.202	**0.158**	0.224	**0.097**	**0.093**	0.151	**0.332**	0.346	0.338
Jura	**0.143**	**0.141**	0.168	**0.138**	**0.138**	0.169	0.176	**0.166**	0.197

Table 6.3: Test error for each method and active learning strategy with 1000 users, or the maximum available.

its performance was then evaluated on the corresponding test sets. Next, the most informative datapoint was identified in each user's pool set. These datapoints were moved into the corresponding training sets and the process was repeated for 10 of these active additions. The entire process, including the dataset partitioning was repeated 25 times.

Figure 6.1 shows the test errors versus the number of active samples. Average errors after 10 queries from the pool presented in Table 6.3. For each model (CPU, CP and SU), the results of the best active learning strategy are highlighted in bold. The results of the best overall model/active learning strategy combination are underlined. BALD appears to be effective for preference elicitation. It always outperforms random sampling and significantly outperforms MES in 9 cases, while MES is better in only 2 cases.

6.6.3 Tuning the Kernel Lengthscale

We attained good performance in CP and CPU by setting the GP kernel hyperparameters to the the median distance between feature vectors. The hyperparameter may be tuned to the data by maximizing the marginal likelihood (model evidence). We perform an additional experiment to show that the approximation to the model evidence returned by EP may be used for this task

Figure 6.2 shows a contour plot of the log-evidence returned by EP on the synthetic dataset with 100 users. We used a squared exponential (RBF) kernel with isotropic lengthscales. The evidence is plotted against different values for the lengthscale parameters σ_{users} and σ_{items}. The synthetic data was generated using $\log \sigma_{\text{users}} = 0$ and $\log \sigma_{\text{items}} = 0$. The highest evidence returned by EP corresponds to values of $\log \sigma_{\text{users}}$ and $\log \sigma_{\text{items}}$ close to the true values of zero. Again, the model used $D = 20$ latent functions, while the data are generated using $D = 5$, so our proposed model seems to be robust to over-specification of the number of latent dimensions.

Although EP can return an estimate of the evidence that includes the VB lower

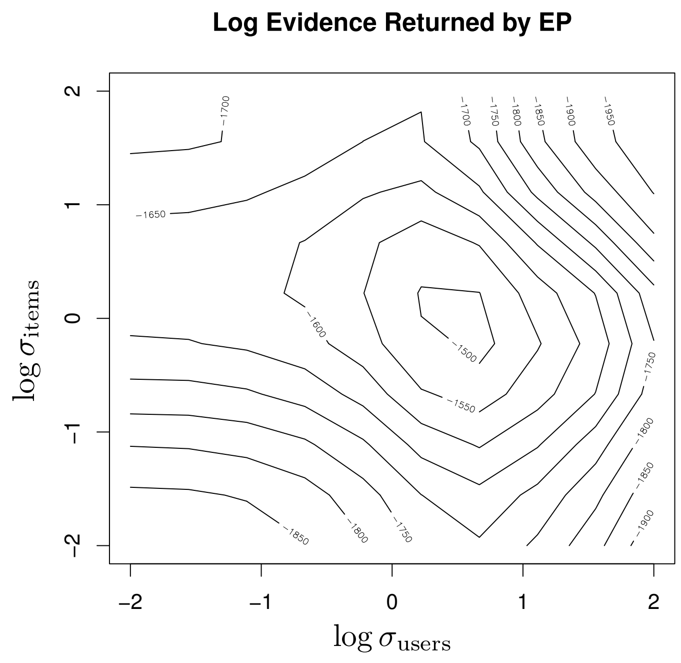

Figure 6.2: Log evidence returned by EP on the synthetic data. x and y coordinates are the log of the lengthscales σ_{users} and σ_{items} respectively. The synthetic data was generated with $\log \sigma_{\text{users}} = 0$ and $\log \sigma_{\text{items}} = 0$.

bound, it is difficult to compute the gradient of the evidence with our hybrid inference algorithm. Thus, tuning the lengthscales using the evidence with more hyperparameters, such as with an ARD kernel may be difficult and is the subject of further investigation. However, our experiments indicate that most of the information about the users arises from the collaborative information in their preferences, and not their features, so in practice further tuning of the hyperparameters may not be necessary with our model.

6.7 Conclusions and Extensions

We have proposed a multi-user model that combines collaborative filtering methods with GP preference learning. To perform efficient inference with this relatively complex model we recast preference learning as a particular case of binary classification with GPs. This is made possible by the preference kernel. The proposed multi-user model performs favourably compared to the single user approach and current multi-user models which are more expensive. Furthermore, our model may incorporate user features if they are available and useful. Current methods either must use them, or cannot include this information. We also show that BALD for GPC can be directly extended to elicit useful preferences from the users.

One abundant source of implicit preference data is web clickthough logs [Joachims, 2002]. Here, the number of users and items may be enormous. Although our method is more scalable than current probabilistic models for this task, more work is required to extend to such web-scale data. With FITC, the linear scaling in the number of users is favourable, however, placing the pseudo-inputs randomly (as we do) may not capture all of the relevant patterns in a GP with many users. Snelson & Ghahramani [2006] locate the pseudo-inputs by optimizing the model evidence. However, computing the gradient of the evidence with respect to these parameters is difficult in our model. An alternative could be to use BALD to select informative locations for the pseudo-inputs. An extension to the IVM has been proposed that uses decision-theoretic loss functions to subsample items for GP-sparsification in preference learning [Abbasnejad *et al.*, 2013]. BALD could select pseudo-inputs from any continuous location.

Although the preference kernel is crucial for implementing accurate EP inference, the resulting cubic (or linear with FITC) scaling in the number of observed item *pairs* is undesirable. The recent work in Abbasnejad *et al.* [2013] has developed EP for the model in Bonilla *et al.* [2010] (BO), discussed in Section 6.5. Since they use the original preference likelihood (6.1), their factors are bivariate, which complicates inference. This may result in difficulties in our model because we include a dimensionality reduction also. However, extending our inference procedure in a similar manner to handle the more complicated original likelihood in (6.1) may be required to scale to large numbers of items.

Our model can incorporate user features if available, or fall back on collaborative information alone if not. However, it does not handle partially missing features vectors formally. Dealing with partially missing inputs in discriminative models is, in general, unsolved. With a prior distribution over the possible values of a missing input, one should integrate the likelihood function over this prior. However, with GPs this is intractable. A heuristic approximation integrates the kernel directly against the input uncertainty [Girard *et al.*, 2003; Turner, 2011]. With a Gaussian prior over the missing input and a squared exponential kernel this results in augmenting the length scales. However, with our preference model, this approach is somewhat unsatisfactory because CPU would not reduce to CP when all the inputs are missing. Designing a method that can smoothly interpolate between CP and CPU with partially missing inputs is an open problem.

In this chapter we considered elicitation of preferences from users for whom we have already observed a number of judgements. However, suppose a new user arrives for whom we have little data. In this case, preference elicitation is hard because the model will be very uncertain about the user's weights $w_{u,d}$ in Equation (6.3). It will only have covariate information to go by, and this may not even be available. Making recommendations with new users or items is referred to as the cold-start problem in collaborative filtering. As we have seen with GPs in Chapter 3, performing robust Bayesian active learning requires appropriate modelling of all sources of uncertainty. In the next chapter we address the cold-start problem with a new robust matrix factorization model for rating data.

Chapter 7

Heteroscedastic Matrix Factorization for Cold-Start Learning

In the previous two chapters we have modelled binary and preference matrix data. We now address the classic data-type in collaborative filtering (CF) systems: rating data. Here, users assign items ordinal-valued ratings or responses, such as $\{\star, \star\star, \ldots\}$, {strongly agree, agree, ...}, etc. Specifically, we address the *cold-start* problem. Cold-start is one of the most challenging problems for recommender systems: what to recommend to new users or items for which one has little or no data. For this we propose a new matrix factorization model for rating data that can be combined with BALD to elicit maximally useful ratings in a manner that is robust in this setting.

Collaborative filtering (CF) based recommender systems exploit shared regularities in behaviour to learn about entities such as users and items. The patterns learnt can then be used to make decisions such as recommending new items to a user. However, CF methods can perform poorly when new users or items are introduced to the system and the amount of data available for such entities is very limited. This scenario is referred to as the *cold-start* problem [Maltz & Ehrlich, 1995; Schein *et al.*, 2002]. One solution to the cold-start problem is to use information from features (e.g. age and gender for users) to make predictions about the new entities [Ahn, 2008; Claypool *et al.*, 1999; Park & Chu, 2009; Park *et al.*, 2006]. However, such features may not be available, e.g. for privacy reasons. A complementary strategy is to collect an initial set of ratings so that the system learns as much as possible about the new entities from a minimal number of user interactions. This is active learning.

We address the cold-start problem using our Bayesian approach to active learning developed in Chapter 2. Bayesian methods exhibit a number of advantages for CF, discussed in Section 5.1. One of the primary advantages is that they provide estimates of uncertainty in predictions and parameter values. This property is important for the success of active learning. In practice, obtaining correct estimates of uncertainty in both the model parameters and the noise levels is essential for identifying the most informative data to collect. This is especially relevant to cold-start learning, as the parameters relating to the new user or item are highly uncertain. To achieve good estimates of uncertainty, we propose a new probabilistic model for rating data that both encodes uncertainty through a posterior distribution over the model parameters and includes a likelihood function for ordinal data with different noise levels (heteroscedasticity) across users and items. We demonstrate superior performance of this model on several rating datasets relative to current state-of-the-art alternatives.

As in the preference learning model presented in Chapter 6, inference with this model requires an iterative EP routine. Thus, we use BALD to yield efficient computation of information gains, avoiding having to perform multiple runs of EP per sample. Furthermore, in cold-start learning it is important to use the data as efficiently as possible and collect information only about the model parameters of primary relevance. For this we extend the 'focused' active learning formulation presented in Section 2.3.4. In cold-start learning it is critical to gain maximal information from the very first sample so as not to deter a new user with multiple requests for information. Empirically, we find that to get the same predictive performance as from a single rating requested by BALD, requires on average 1.59 ratings with random sampling and 1.85 with uncertainty sampling. An increase from one to two initial rating requests may be critical to whether a user stays with the system.

7.1 A Robust Model for Ordinal Matrix Data

In CF with rating data, we are given a dataset $\{r_{u,i} : 1 \leq u \leq U, 1 \leq i \leq I, r_{u,i} \in \{1, \dots R\}, (u,i) \in \mathcal{D}\}$ of discrete ratings by U users on I items, where the possible rating values are ordinal, $1 < \dots < R$, for example, 1 to R 'stars' assigned to a product. $\mathcal{D}$ is the set of pairs of users and items for which a rating is available (observed). We assume that the dataset is a sample from a full $U \times I$ rating matrix $\mathbf{R}$, where the entry $r_{u,i}$ in the u-th row and i-th column of $\mathbf{R}$ contains the u-th user's rating for the i-th item. In practice, the observed dataset contains only a small fraction of the entries in $\mathbf{R}$, we denote the observed ratings $\mathbf{R}^{\mathcal{D}}$.

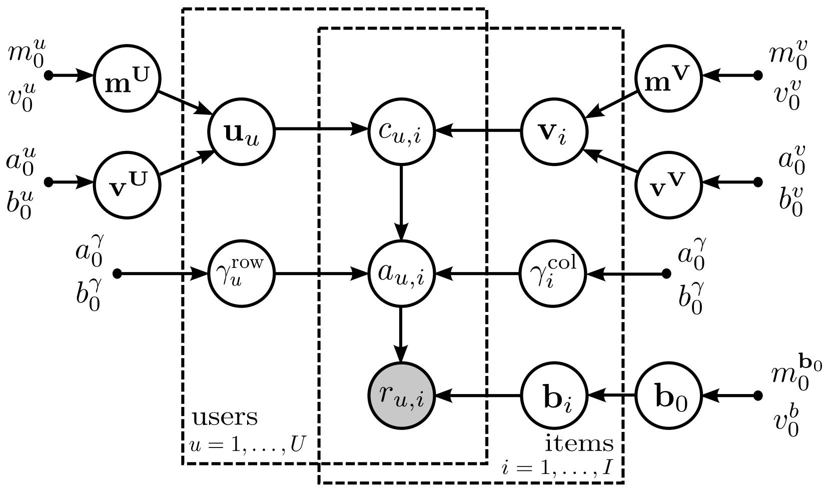

Figure 7.1: Graphical representation of the model for ordinal matrix data as it is described in the main text. The observed variables are the rating values $r_{u,i}$. All the other variables are latent. Dots denote fixed hyperparameters.

We propose a new probabilistic model for $\mathbf{R}$. This model is robust, in that it allows the level of noise in the entries of $\mathbf{R}$ to vary across rows and columns. This is particularly important for active learning, where collecting data from users or items that are too noisy is wasteful. To capture the discrete nature and natural ordering of rating data, our model follows an ordinal regression approach [Chu & Ghahramani, 2005a; Stern *et al.*, 2009]. This is an advantage over the Gaussian likelihood, usually used for this task, that inappropriately assumes continuous entries in $\mathbf{R}$. To obtain better predictions, our ordinal likelihood function has different hyperparameters for each column (item) of $\mathbf{R}$. We learn these hyperparameters using a hierarchical prior. Further, the model is based on a low rank matrix factorization with a hierarchical prior on the latent low rank factors. This second hierarchical prior increases the robustness of the model to overfitting and parameter specification. Figure 7.1 shows the graphical model for this probabilistic method, which we now describe in detail.

7.1.1 Model Description

We model the generation of the ratings $\mathbf{R}$ as a function of two low rank latent matrices $\mathbf{U} \in \mathbb{R}^{U\times D}$ and $\mathbf{V} \in \mathbb{R}^{I\times D}$, where $D \ll \min(U, I)$. The value of $r_{u,i}$ is determined by i) the scalar $\mathbf{u}_u^\mathrm{T}\mathbf{v}_i$, where $\mathbf{u}_u$ is the vector contained in the u-th row of $\mathbf{U}$ and $\mathbf{v}_i$ in the i-th row of $\mathbf{V}$, and ii) a partition of the real line into $R - 1$ contiguous intervals with boundaries $b_{i,0} < \ldots < b_{i,R}$, where $b_{i,0} = -\infty$ and $b_{i,R} = \infty$. The value of $r_{u,i}$

is obtained as a function of the interval in which $\mathbf{u}_u^{\mathrm{T}}\mathbf{v}_i$ lies. Note that the interval boundaries are different for each column of $\mathbf{R}$. A simple model would be $r_{u,i} = k$ if $\mathbf{u}_u^\top \mathbf{v}_i \in (b_{i,k-1}, b_{u,k}]$. However, due to noise, there may be no $b_{i,0}, \ldots, b_{i,R}$, $\mathbf{U}$ and $\mathbf{V}$ that guarantee $\mathbf{u}_u^\top \mathbf{v}_i \in (b_{i,r_{u,i}-1}, b_{i,r_{u,i}}]$ for all of the observed ratings in $\mathbf{R}^{\mathcal{D}}$. We model this by adding zero-mean Gaussian noise $e_{u,i}$ to $\mathbf{u}_u^\top \mathbf{v}_i$ before generating $r_{u,i}$, and introducing the latent variable $a_{u,i} = \mathbf{u}_u^\top \mathbf{v}_i + e_{u,i}$. The probability of $r_{u,i}$ given $a_{u,i}$ and $\mathbf{b}_i = (b_{i,1}, \ldots, b_{i,R-1})$ is

$$\begin{aligned} p(r_{u,i}|a_{u,i}, \mathbf{b}_i) &= \prod_{k=1}^{r_{u,i}-1} \Theta[a_{u,i} - b_{i,d}] \prod_{k=r_{u,i}}^{R-1} \Theta[b_{j,k} - a_{u,i}] \\ &= \prod_{k=1}^{R-1} \Theta\left[\mathrm{sign}[r_{u,i} - k - 0.5](a_{u,i} - b_{i,d})\right] , \end{aligned} \tag{7.1}$$

where Θ denotes the Heaviside step function; $\Theta[x] = 1$ for $x \geq 1$ and zero otherwise. Thus, the likelihood (7.1) takes value 1 when $a_{u,i} \in (b_{r_{u,i}-1}, b_{r_{u,i}}]$ and 0 otherwise. Note the dependence of (7.1) on all the entries in $\mathbf{b}_i$ and not only on $b_{r_{u,i}-1}$ and $b_{r_{u,i}}$. The prior for the vector of boundary variables $\mathbf{b}_i$ for the i-th column of $\mathbf{R}$ is hierarchical Gaussian, $p(\mathbf{b}_i|\mathbf{b}_0) = \prod_{k=1}^{R-1} \mathcal{N}(b_{i,k}; b_{0,k}, v_0)$, where $\mathbf{b}_0$ is a vector of base interval boundaries, with prior $p(\mathbf{b}_0) = \prod_{k=1}^{R-1} \mathcal{N}(b_{0,k}; m_k^{\mathbf{b}_0}, v_0)$. $m_1^{\mathbf{b}_0}, \ldots, m_{R-1}^{\mathbf{b}_0}$ and v_0 are hyperparameters. Note that although the boundaries may cross *a priori*, crossed boundaries have zero likelihood, so the posterior means remain in order.

We include heteroscedasticity in the additive noise $e_{u,i}$ across users and items. For this, $e_{u,i}$ follows *a priori* a zero-mean Gaussian distribution with variance $\gamma_i^{\text{row}} \times \gamma_j^{\text{col}}$, where γ_i^{row} and γ_j^{col} are factors that specify the variance of $e_{u,i}$ for the u-th row and i-th column of $\mathbf{R}$. We define $c_{u,i} = \mathbf{u}_u^{\mathrm{T}}\mathbf{v}_i$ and assume that the conditional distribution of $a_{u,i}$ given $c_{u,i}$, γ_i^{row} and γ_j^{col} is $p(a_{u,i}|c_{u,i}, \gamma_i^{\text{row}}, \gamma_j^{\text{col}}) = \mathcal{N}(a_{u,i}; c_{u,i}, \gamma_i^{\text{row}}\gamma_j^{\text{col}})$. To learn the user and item specific noise levels we put inverse Gamma priors on γ_i^{row} and γ_j^{col}.

For robustness to fixing parameter values, we use a hierarchical Gaussian prior for $\mathbf{U}$ and $\mathbf{V}$, that is, $p(\mathbf{U}|\mathbf{m}^{\mathbf{U}}, \mathbf{v}^{\mathbf{U}}) = \prod_{u=1}^{U} \prod_{d=1}^{D} \mathcal{N}(u_{u,d}; m_d^{\mathbf{U}}, v_d^{\mathbf{U}})$ and $p(\mathbf{V}|\mathbf{m}^{\mathbf{V}}, \mathbf{v}^{\mathbf{V}}) = \prod_{i=1}^{I} \prod_{d=1}^{D} \mathcal{N}(v_{i,d}; m_d^{\mathbf{V}}, v_d^{\mathbf{V}})$, where $\mathbf{m}^{\mathbf{U}}$ and $\mathbf{m}^{\mathbf{V}}$ are mean parameters for the rows of $\mathbf{U}$ and $\mathbf{V}$ respectively. We select factorized standard Gaussian priors for these parameters. Similarly, $\mathbf{v}^{\mathbf{U}}$ and $\mathbf{v}^{\mathbf{V}}$ are variance parameters for the rows of $\mathbf{U}$ and $\mathbf{V}$ and are given factorized inverse Gamma priors.

Lastly, let $\mathbf{C}^{\mathcal{D}}$ be the set of variables $c_{u,i}$ for which $r_{u,i}$ is observed, then $p(\mathbf{C}^{\mathcal{D}}|\mathbf{U}, \mathbf{V}) = \prod_{(u,i)\in\mathcal{D}} \delta(c_{u,i} - \mathbf{u}_u^\top \mathbf{v}_i)$. Similarly we collect the variables $a_{u,i}$ for the observed ratings into $\mathbf{A}^{\mathcal{D}}$, and the threshold boundary variables $\mathbf{b}_i$ into a $d \times (R-1)$ matrix $\mathbf{B}$. Let

$\mathbf{R}^{\mathcal{D}}$ denote the set of entries in $\mathbf{R}$ that are observed. Then the likelihood factorizes as $p(\mathbf{R}^{\mathcal{D}}|\mathbf{A}^{\mathcal{D}},\mathbf{B}) = \prod_{(u,i)\in\mathcal{D}} p(r_{u,i}|a_{u,i},\mathbf{b}_i)$. Given $\mathbf{R}^{\mathcal{D}}$, the posterior distribution over all of the variables $\mathbf{\Xi} = \{\mathbf{A}^{\mathcal{D}},\mathbf{C}^{\mathcal{D}},\mathbf{U},\mathbf{V},\mathbf{B},\boldsymbol{\gamma}^{\text{row}},\boldsymbol{\gamma}^{\text{col}},\mathbf{b}_0,\mathbf{m}^{\mathbf{U}},\mathbf{m}^{\mathbf{V}},\mathbf{v}^{\mathbf{U}},\mathbf{v}^{\mathbf{V}}\}$ is

$$\begin{aligned} p(\mathbf{\Xi}|\mathbf{R}^{\mathcal{D}}) =& p(\mathbf{R}^{\mathcal{D}}|\mathbf{A}^{\mathcal{D}},\mathbf{B})p(\mathbf{A}^{\mathcal{D}}|\mathbf{C}^{\mathcal{D}},\boldsymbol{\gamma}^{\text{row}},\boldsymbol{\gamma}^{\text{col}})p(\mathbf{C}^{\mathcal{D}}|\mathbf{U},\mathbf{V})p(\mathbf{U}|\mathbf{m}^{\mathbf{U}},\mathbf{v}^{\mathbf{U}})p(\mathbf{V}|\mathbf{m}^{\mathbf{V}},\mathbf{v}^{\mathbf{V}}) \\ & p(\mathbf{B}|\mathbf{b}_0)p(\mathbf{b}_0)p(\boldsymbol{\gamma}^{\text{row}})p(\boldsymbol{\gamma}^{\text{col}})p(\mathbf{m}^{\mathbf{U}})p(\mathbf{m}^{\mathbf{V}})p(\mathbf{v}^{\mathbf{U}})p(\mathbf{v}^{\mathbf{V}})[p(\mathbf{R}^{\mathcal{D}})]^{-1}\,, \end{aligned} \tag{7.2}$$

where $p(\mathbf{R}^{\mathcal{D}})$ is the normalization constant. Conditioning on hyperparameters has been omitted for clarity.

Hyperparameter Values

All of the Gaussian hyper-priors were given standard Normal distributions. The prior means $m_1^{\mathbf{b}_0},\ldots,m_{L-1}^{\mathbf{b}_0}$ were set to form an evenly spaced grid on the interval $[-6,6]$, as suggested in Paquet *et al.* [2012]. The prior variance v_0 for each component of $\mathbf{b}_0$ is initialized to $v_0 = 0.1$.

The hyperparameters a_0^{γ} and b_0^{γ} for the priors on γ_u^{row} and γ_i^{col} are set to $a_0^{\gamma} = 5$ and $b_0^{\gamma} = 5\sqrt{10}$. This yields a mean value of 10 for the product of γ_i^{row} and γ_j^{col}, which is the recommended noise level in the (homoscedastic) ordinal MF model in Paquet *et al.* [2012]. The other inverse gamma hyperpriors were given values $a_0^u = a_0^v = 5$ and $b_0^u = b_0^v = 5$. The is equivalent to having seen a random sample of size 10 with unit empirical variance.

7.1.2 Inference

As with most non-trivial models, computing the posterior (7.2) exactly is intractable. Thus we perform approximate inference using a combination of expectation propagation (EP) and variational Bayes (VB). This algorithm follows a similar procedure to the scheme used for the preference learning model in Chapter 6, except that the ordinal model has more factors to refine. We use the following parametric approximation to the exact posterior:

$$
\begin{aligned}
&q(\boldsymbol{\Xi}) = \\
&\left[\prod_{(u,i)\in\mathcal{D}} \mathcal{N}(a_{u,i}; m^a_{u,i}, v^a_{u,i})\right] \left[\prod_{(u,i)\in\mathcal{D}} \mathcal{N}(c_{u,i}; m^c_{u,i}, v^c_{u,i})\right] \left[\prod_{u=1}^{U}\prod_{d=1}^{D} \mathcal{N}(u_{u,d}; m^u_{u,d}, v^u_{u,d}\right] \\
&\left[\prod_{i=1}^{I}\prod_{d=1}^{D} \mathcal{N}(v_{i,d}; m^v_{i,d}, v^v_{i,d}\right] \left[\prod_{i=1}^{I}\prod_{k=1}^{R-1} \mathcal{N}(b_{i,d}; m^b_{i,d}, v^b_{i,d})\right] \left[\prod_{k=1}^{R-1} \mathcal{N}(b_{0,k}; m^{b_0}_k, v^{b_0}_k)\right] \\
&\left[\prod_{k=1}^{D} \mathcal{N}(m^{\mathbf{U}}_k; m^{m^{\mathbf{U}}}_k, v^{m^{\mathbf{U}}}_k)\right] \left[\prod_{d=1}^{D} \mathcal{N}(m^{\mathbf{V}}_d; m^{m^{\mathbf{V}}}_d, v^{m^{\mathbf{V}}}_d)\right] \left[\prod_{d=1}^{D} \mathcal{IG}(v^{\mathbf{U}}_d; a^{v^{\mathbf{U}}}_d, a^{v^{\mathbf{U}}}_d)\right] \\
&\left[\prod_{d=1}^{D} \mathcal{IG}(v^{\mathbf{V}}_d; a^{v^{\mathbf{V}}}_d, b^{v^{\mathbf{V}}}_d)\right] \left[\prod_{u=1}^{U} \mathcal{IG}(\gamma^{\text{row}}_u; a^{\gamma^{\text{row}}}_u, b^{\gamma^{\text{row}}}_u)\right] \left[\prod_{i=1}^{I} \mathcal{IG}(\gamma^{\text{row}}_i; a^{\gamma^{\text{col}}}_i, b^{\gamma^{\text{col}}}_i)\right] . \qquad (7.3)
\end{aligned}
$$

The parameters on the right hand side of (7.3) are refined using EP. There are 13 approximate factors, these correspond to the each of the terms in the numerator of the posterior in Equation (7.2). These approximate factors take the same functional form as the posterior approximation in (7.3). The approximation to the posterior $q(\boldsymbol{\Xi})$ is obtained as the normalized product of the 13 factors. To refine q, each factor is refined in turn. A overview of how to refine approximate factors using EP is given in Section 6.3.1. There are various levels of approximation required to refine the factors of our ordinal MF model, which we now describe. Full details can be found in the supplementary material to Houlsby *et al.* [2014].

Exact Factors

Some of the approximate factors have the same functional form as the factors in the true posterior (7.2). Thus the EP update results in no approximation and these factors are exact. This applies to the Gaussian priors $p(\mathbf{m}^{\mathbf{U}})$, $p(\mathbf{m}^{\mathbf{U}})$, $p(\mathbf{b}_0)$ and the inverse Gamma priors $p(\mathbf{v}^{\mathbf{U}})$, $p(\mathbf{v}^{\mathbf{U}})$, $p(\gamma^{\text{row}})$,$p(\gamma^{\text{col}})$. For these, the parameters in the approximate factors are set to the hyperparameters given above. Further, since these updates do not depend on the parameters of any other approximate factor, they are refined once during initialization and not adjusted any further.

Exact EP

Some of the approximate factors will not have the same functional form as their corresponding true factors, but the EP refinement procedure, which involves moment matching the approximate factor to the true factor (see Section 6.3.1) can be performed exactly. This includes the likelihood $p(\mathbf{R}^{\mathcal{D}}|\mathbf{A}^{\mathcal{D}},\mathbf{B})$ and the hierarchical prior on the layer boundaries $p(\mathbf{B}|\mathbf{b}_0)$. The prior on $\mathbf{B}$ requires the moment matching of Gaussian distributions. The update equations for likelihood follow from those required for classification with a probit likelihood [Rasmussen & Williams, 2005].

Approximate EP

The EP refinement calculations are intractable for some of the factors. These are $p(\mathbf{A}^{\mathcal{D}}|\mathbf{C}^{\mathcal{D}},\boldsymbol{\gamma}^{\text{row}},\boldsymbol{\gamma}^{\text{col}})$, $p(\mathbf{U}|\mathbf{m}^{\mathbf{U}},\mathbf{v}^{\mathbf{U}})$, $p(\mathbf{V}|\mathbf{m}^{\mathbf{V}},\mathbf{v}^{\mathbf{V}})$. This is due to the inverse Gamma priors over the variances in these factors. To update the approximations to the latter two factors, after integrating out these variances we need to solve the integral of a Gaussian times a Student's t-distribution, which has no analytic form. Therefore, we approximate the Student's t-distribution with a Gaussian with the same mean and variance. To do this we follow the technique described in Hernández-Lobato [2007]. In $p(\mathbf{A}^{\mathcal{D}}|\mathbf{C}^{\mathcal{D}},\boldsymbol{\gamma}^{\text{row}},\boldsymbol{\gamma}^{\text{col}})$ the variance depends on the product of two variables that follow inverse Gamma distributions. To make the required marginalizations tractable, we approximate the inverse Gammas with delta functions at their mode.

Variational Bayes

As described Section 6.3.2, EP provides a poor approximation to factors that correspond to matrix factorizations due to invariances in the solutions to MFs [Stern *et al.*, 2009]. Therefore, to refine the approximation to the final factor $p(\mathbf{C}^{\mathcal{D}}|\mathbf{U},\mathbf{V})$ we use VB. For this we follow the same routine outlined in Section 6.3.3.

7.1.3 Predictive Distribution

Given the approximation to the posterior in (7.3), we estimate the predictive probability of a new entry $r^{\star}_{u,i}$ of $\mathbf{R}$ that is not contained in the observed ratings $\mathbf{R}^{\mathcal{D}}$ using

$$\begin{aligned} p(r^{\star}_{u,i}|\mathbf{R}^{\mathcal{D}}) &\approx \int p(r^{\star}_{u,i}|a^{\star}_{u,i},\mathbf{b}_i)p(a^{\star}_{u,i}|c^{\star}_{u,i},\gamma^{\text{row}}_i,\gamma^{\text{col}}_j)p(c^{\star}_{u,i}|\mathbf{u}_u,\mathbf{v}_i)q(\boldsymbol{\Xi})\,d\boldsymbol{\Xi}\,da^{\star}_{u,i}\,dc^{\star}_{u,i} \\ &\approx \Phi\left[\zeta(r^{\star}_{u,i})\right] - \Phi\left[\zeta(r^{\star}_{u,i}-1)\right] \,, \end{aligned} \tag{7.4}$$

where

$$\zeta(r^\star_{u,i}) = \frac{m^b_{i,r^\star_{u,i}} - m^{c,\star}_{u,i}}{\sqrt{v^{c,\star}_{u,i} + v^b_{j,r^\star_{u,i}} + v^\gamma_{u,i}}}, \tag{7.5}$$

$$m^{c,\star}_{u,i} = \sum_{k=1}^{h} m^u_{u,d} m^v_{i,d},$$

$$v^{c,\star}_{u,i} = \sum_{k=1}^{h} [m^u_{u,d}]^2 v^v_{i,d} + v^u_{u,d}[m^v_{j,k}]^2 + v^u_{u,d} v^v_{j,k},$$

$$v^\gamma_{u,i} = \frac{b^{\gamma^{\text{row}}} b^{\gamma^{\text{col}}}}{(a^{\gamma^{\text{row}}}+1)(a^{\gamma^{\text{col}}}+1)}$$

and Φ is the standard Gaussian c.d.f.

As well as replacing the true posterior with the EP approximation q, two approximations were used to compute Equation (7.4). These were required to integrate over the inner product of random vectors in $p(c^\star_{u,i}|\mathbf{u}_u, \mathbf{v}_i)$ and the product of variance parameters in $p(a^\star_{u,i}|c^\star_{u,i}, \gamma^{\text{row}}_u, \gamma^{\text{col}}_i)$. In both cases we moment-match Gaussian distributions to the integral over these factors, and as before we approximate the inverse Gammas over γ^{row}_u and γ^{col}_i with point masses at their modes.

Intuitively, the above predictive distribution (7.4) incorporates *two sources* of uncertainty. The first originates from the unknown value of the variables in $\boldsymbol{\Xi}$. This uncertainty is captured by the width (variance) of the different factors that form q and it is summarized in $\zeta(r^\star_{u,i})$ by the variance terms $v^{c,\star}_{u,i}$ and $v^b_{i,r^\star_{u,i}}$. The second originates from the heteroscedastic additive noise in $a^\star_{u,i}$. This uncertainty is encoded in $\zeta(r^\star_{u,i})$ by the variance term $v^\gamma_{u,i}$. Therefore, (7.4) allows us to take into account the uncertainty in model parameters $\boldsymbol{\Xi}$ and the intrinsic noisiness of the data when making predictions. Equipped with this model we can take a robust Bayesian approach to active learning.

7.2 Cold-Start Active Learning

In the cold-start scenario we have little information about the new user or item. Furthermore, in recommender systems the budget for making queries for ratings may be very small, since a user will expect reasonable recommendations without providing lots of initial information. It is therefore crucial to gain maximal information about only the most relevant parameters, such as the new user's latent feature vector, and not waste data reinforcing all of the other model parameters in $\boldsymbol{\Xi}$. Thus, the best framework for this task is the version of BALD that focuses on particular parameters, introduced in Section 2.3.4.

To recap, denote the set of parameters of interest $\mathbf{\Theta}$, and the set of additional nuisance parameters $\mathbf{\Phi}$. After running inference on the observed data $\mathbf{R}^{\mathcal{D}}$ we have a posterior $p(\mathbf{\Theta}, \mathbf{\Phi}|\mathbf{R}^{\mathcal{D}})$. In the context of CF we seek to elicit the an additional $r^{\star}_{u,i}$ from $\mathbf{R}$ to maximize the information gain about $\mathbf{\Theta}$. The utility function, and computationally efficient BALD rearrangement are then

$$\mathcal{U}(u,i) = \mathrm{H}\left[\int p(\mathbf{\Theta}, \mathbf{\Phi}|\mathbf{R}^{\mathcal{D}})\, d\mathbf{\Phi}\right] - \mathbb{E}_{p(r^{\star}_{u,i}|\mathbf{R}^{\mathcal{D}})}\left\{\mathrm{H}[\int p(\mathbf{\Theta}, \mathbf{\Phi}|r^{\star}_{u,i}, \mathbf{R}^{\mathcal{D}})\, d\mathbf{\Phi}]\right\}\,, \tag{7.6}$$

$$= \mathrm{H}[p(r^{\star}_{u,i}|\mathbf{R}^{\mathcal{D}})] - \mathbb{E}_{p(\mathbf{\Theta}|\mathbf{R}^{\mathcal{D}})}\left\{\mathrm{H}[\mathbb{E}_{p(\mathbf{\Phi}|\mathbf{\Theta},\mathbf{R}^{\mathcal{D}})} p(r^{\star}_{u,i}|\mathbf{\Theta}, \mathbf{\Phi})]\right\}\,.^{1} \tag{7.7}$$

Equation (7.7) indicates that for effective cold-start active learning we must capture both the uncertainty in the model parameters with an accurate inference routine (to compute the first term), and the intrinsic heteroscedastic noisiness in the data (to compute the second term).

7.2.1 Implementation of BALD

Let u be the index of a new user. We would like to make good predictions for this user from minimal user-interactions. For this, we have to gain maximal information about the user's latent vector $\mathbf{u}_u$, the u-th row of matrix $\mathbf{U}$. Thus, $\mathbf{u}_u$ forms the parameters of interest $\mathbf{\Theta}$ and all the other model parameters $\mathbf{\Xi} \setminus \{\mathbf{u}_u\}$ are collected into the set of nuisance parameters $\mathbf{\Phi}$. We approximate the terms in Equation (7.7) by replacing the posterior with posterior approximation (7.3) and the predictive distribution with Equation (7.4). The first term in (7.7) is then straightforward to compute since it is the entropy of the multinomial distribution given in Equation (7.4). The second term requires the computation of

$$\mathbb{E}_{q(\mathbf{u}_u)}\mathrm{H}[\mathbb{E}_{q(\mathbf{\Phi})} p(r^{\star}_{u,i}|\mathbf{u}_u, \mathbf{\Phi})] = \mathbb{E}_{q(\mathbf{u}_u)}\mathrm{H}[p(r^{\star}_{u,i}|\mathbf{u}_u)]\,, \tag{7.8}$$

where

$$p(r^{\star}_{u,i}|\mathbf{u}_u) = \Phi[\zeta(r^{\star}_{u,i})] - \Phi[\zeta(r^{\star}_{u,i} - 1)]$$

[1]Note that in the general discriminative framework given in Figure 2.1 there are no input features $\mathbf{x}$. Here we may consider the row and column indices (u, i) as the input that we choose for 'labelling'.

in this case the components $m^{c,\star}_{u,i}$ and $v^{c,\star}_{u,i}$ of $\zeta(\cdot)$ in (7.5) are given by:

$$m^{c,\star}_{u,i} = \sum_{d=1}^{D} m^{v}_{i,d} u_{u,d}, \quad v^{c,\star}_{u,i} = \sum_{d=1}^{D} v^{v}_{i,d} u^{2}_{u,d}$$

since we have now conditioned on a particular $\mathbf{u}_u = (u_{u,1}, \dots, u_{u,D})$.

Equation (7.8) includes an intractable D-dimensional Gaussian integral over $\mathbf{u}_u$. We approximate this integral by Monte Carlo sampling. In particular, we compute the expectation of $p(r^{\star}_{u,i}|\mathbf{u}_u)$ over a random sample from the Gaussian $q(\mathbf{u}_u)$. Experimentally we found that this estimate converges quickly; fewer than 100 samples were required for accurate computation of (7.7). However, when computational time is critical we can use the unscented approximation which uses only $2D + 1$ samples placed at fixed locations [Julier & Uhlmann, 1997]. This method is fast, but can generate biased estimates to the integral. In practice, we found that the unscented approximation is sufficiently accurate to identify the most informative item in most cases, see Section 7.4.

We may use exactly the same methodology described above to learn actively about new items rather than new users. In this case we wish to learn optimally about a new item with index i, so now $\mathbf{\Theta} = \mathbf{v}_i$, the i-th row of $\mathbf{V}$. The required computations are symmetric to those for the new user scenario.

7.3 Related Work

We review previous probabilistic models for rating matrices and both model-based and model-free strategies for cold-start active learning.

7.3.1 Probabilistic Models for Rating Matrices

Bayesian ordinal matrix factorization is addressed in Paquet *et al.* [2012], but their model does not include heteroscedasticity. This component is important since users and items can exhibit variable noise levels. We show empirically that this particularly important to take account when performing active learning. The model in Paquet *et al.* [2012] does not learn the boundary variables $\mathbf{B}$ either. Both of these components yield improvements in predictive performance, as we demonstrate empirically in Section 7.4. Furthermore, the method in Paquet *et al.* [2012] uses Gibbs sampling for inference while our EP-VB method produces accurate and compact approximations that can be easily stored and manipulated. Heteroscedasticity has been previously considered in a MF model with a Gaussian likelihood [Lakshminarayanan *et al.*, 2011]. However, as observed in Chapter 5, using an inappropriate Gaussian likelihood for discrete data yields poor predictions. Our experiments in Section 7.4 confirm this for ordinal data. Another alternative for discrete ratings has been proposed in Marlin & Zemel [2007]. This work assumes that each row in the rating matrix **R** is sampled i.i.d. from a Bayesian Mixture of Multinomials (BMM) model. This mixture model is not as expressive as MF approaches.

7.3.2 Cold-Start Learning

Cold-start active learning has been investigated in other probabilistic models [Boutilier *et al.*, 2002; Harpale & Yang, 2008; Jin & Si, 2004]. These methods either seek to maximize the expected value of information or compute posterior uncertainties directly. As in Equation (7.6), this requires the (potentially expensive) re-computation of the posterior for all possible ratings that could be collected. To reduce the computational cost, such methods approximate their utility function [Boutilier *et al.*, 2002] or perform approximate incremental updates [Jin & Si, 2004]. Furthermore, these works restrict themselves to relatively simple models where parameter updates can be fast, such as the multiple-cause vector quantization model [Ross & Zemel, 2002], naive Bayes, the aspect model [Hofmann, 2003] and the flexible mixture model [Si & Jin, 2003]. With Equation (7.7) we only update the posterior distribution *after* collecting the new rating.

Model-free strategies have also been proposed for active data collection [Rashid *et al.*, 2002, 2008]. Here, empirical statistics of the data such as item popularity or rating entropy are used to select items. These heuristics are computationally cheap, but they tend to perform poorly relative to model-based approaches. In complementary lines of work to ours, query strategies have been designed for non-probabilistic models, such as Mahalanobis distance-based methods in the co-clustering model, and decision trees in functional MF [Le & Tu, 2010; Zhou *et al.*, 2011].

7.4 Experiments

We evaluated our model and active learning strategy with experiments on seven rating datasets from a diverse set of domains. Unless otherwise stated, the ratings in each dataset were ordinal valued, in the range $1, \ldots, 5$. These were obtained and processed as follows:

i) Book A set of ratings for books, publicly available from http://www.informatik.uni-freiburg.de/~cziegler/BX/. The ratings take values $1, \dots, 10$. Most of them take value higher than 6, so we merged the ratings $1, \dots, 6$ to yield 5 values in total.

ii) Dating A set of ratings from an online dating website, described in Brozovsky & Petricek [2007] and available at http://www.occamslab.com/petricek/data/. These original ratings take values in $\{1 \dots, 10\}$. We mapped these to $\{1, \dots, 5\}$ as: $\{1, 2\} \to 1$, $\{3, 4\} \to 2$, $\{5, 6\} \to 3$, $\{7, 8\} \to 4$, $\{9, 10\} \to 5$.

iii) IPIP This dataset contains responses to a 336 item International Item Pool questionnaire [Goldberg *et al.*, 2006]. These data were collected from Facebook [Kosinski *et al.*, 2013] and are available for research upon request at http://mypersonality.org/wiki/doku.php?id=start. This dataset is dense, all of the ratings are observed. All of the other datasets have many missing entries.

iv) Jester A collection of ratings for jokes, available at http://goldberg.berkeley.edu/jester-data/. The ratings on this dataset are real valued $\in [-10, 10]$. We converted these to ordinal ratings by grouping the values into 5 bins with equal counts.

v) MovieLens100K and MovieLens1M Collections of ratings for movies, commonly used for benchmarking recommendation systems. These are available at http://grouplens.org/datasets/movielens/.

vi) MovieTweets This set was released recently, and consists of ratings for movies collected from Tweets. It is described in [Dooms *et al.*, 2013], and available from https://github.com/sidooms/MovieTweetings. The original ratings take values in $\{0 \dots, 10\}$. We mapped these to values in $\{1, \dots, 5\}$ as: $\{0, 1, 2\} \to 1$, $\{3, 4\} \to 2$, $\{5, 6\} \to 3$, $\{7, 8\} \to 4$, $\{9, 10\} \to 5$.

vii) Webscope A collection of ratings on songs. It has been made available for research upon request from Yahoo! Labs. We used the 'R3' dataset at http://webscope.sandbox.yahoo.com/catalog.php?datatype.

7.4.1 Comparison to Other Models for Rating Data

We first evaluate the predictive accuracy of our matrix factorization model against a number of state-of-the-art alternatives. We then investigate the performance of our

Dataset	HOMF	OMF	HOMF -NoB	OMF -NoB	Paquet	RBMF	BMF	BMM
Book	**-1.415**	-1.436	-1.507	-1.439	-1.427	-1.545	-1.544	-1.622
Dating	**-0.867**	-0.906	-0.890	-1.028	-1.009	-1.045	-1.140	-0.948
IPIP	**-1.096**	-1.140	-1.131	-1.189	-1.188	-1.194	-1.225	-1.270
Jester	**-1.238**	-1.306	**-1.240**	-1.320	-1.320	-1.312	-1.368	-1.290
ML1M	**-1.136**	-1.165	-1.141	-1.177	-1.170	-1.173	-1.210	-1.324
ML100K	**-1.203**	-1.234	-1.208	-1.243	-1.232	-1.238	-1.277	-1.493
MTweet	**-0.956**	-0.991	-0.984	-1.025	-1.012	-1.014	-1.077	-1.115
WebScope	**-1.207**	-1.253	-1.209	-1.257	-1.236	-1.529	-1.532	-1.298

Table 7.1: Average test log likelihood. Bold typeface denotes the best method, and those statistically indistinguishable.

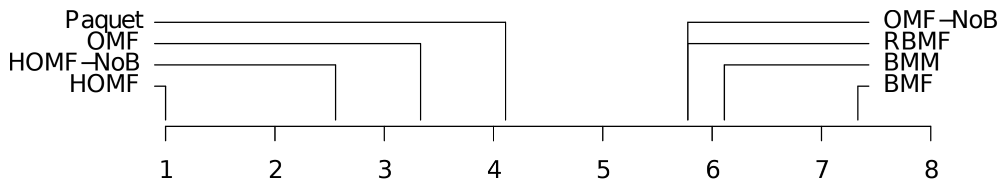

Figure 7.2: Mean rank of each method across all of the datasets.

method for cold-start active learning. Our model for heteroscedastic ordinal matrix factorization (HOMF) is compared to the following methods: i) the homoscedastic model with an ordinal likelihood presented in Paquet *et al.* [2012] (Paquet); ii) a method for robust Bayesian matrix factorization (RBMF) based on a Gaussian likelihood which also has heteroscedastic additive noise [Lakshminarayanan *et al.*, 2011]; iii) the Bayesian mixture of multinomials model (BMM) described in Marlin & Zemel [2007]; and iv) a matrix factorization model like RBMF but with homoscedastic noise (BMF). We evaluate the improvements in predictive performance produced in HOMF by considering heteroscedasticity and learning the boundary variables $\mathbf{B}$. For this, we compare to OMF, a homoscedastic version of HOMF, where the variance parameters γ_u^{row} and γ_i^{col} are equal across rows and columns, respectively. We also compare to HOMF-NoB which uses fixed boundary parameters $\mathbf{b}_j$ rather than learning them for each item. Finally, OMF-NoB is a homoscedastic version of HOMF that does not learn $\mathbf{B}$. For all of the matrix factorization models and we fixed the latent dimension to $D = 10$, and used 10 mixture components in BMM.

Some of the datasets are very sparse, so we selected only users and items that have 10 ratings or more, as proposed in Rendle *et al.* [2009]. Then we randomly split the

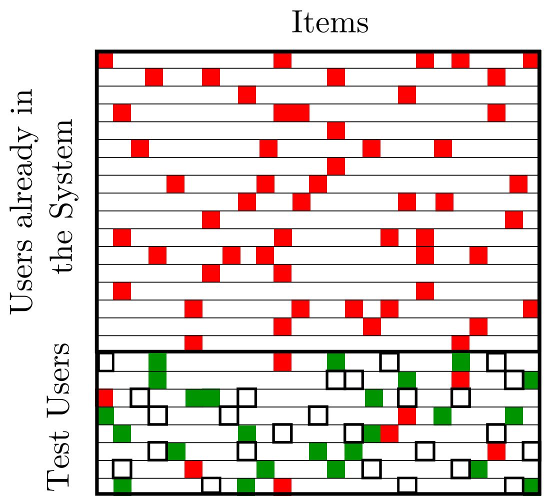

Figure 7.3: Experimental setup for cold-start active learning. The squares depict available ratings. Red squares form the training set. These are all of the ratings for those users *already in the system* and one rating per test user. Green squares form the test set. The remaining hollow squares form the pool set for the test users. Note that most ratings are missing.

available ratings for each dataset into a training and a test set with 80% and 20% of the ratings respectively. Each method was adjusted using the entries in the training set and then we evaluated the predictive log likelihood on the corresponding test set. The entire procedure was repeated 20 times.

Table 7.1 contains the test log likelihood obtained by each method on each of the datasets, and Figure 7.2 summarizes the overall performance of each algorithm. The proposed model, HOMF, outperforms all of the other methods on all datasets. Significance is assessed with a paired t-test at the 5% level. The likelihood function for ordinal data is more appropriate for ratings than the Gaussian likelihood: HOMF and Paquet outperform RBMF and BMF. Furthermore, one attains improvements in predictive performance by modelling heteroscedasticity across rows and across columns since HOMF outperforms OMF and Paquet, and RBMF outperforms BMF. Learning the biases also results in substantial improvements to the performance of our model. Finally, the matrix factorization models (HOMF, Paquet, RBMF and BMF) usually outperform the mixture model BMM.

7.4.2 Cold-Start Active Learning

For each dataset we selected the 2000 users and 1000 items (up to the maximum available) with the most ratings. This provides the active sampling strategies with the largest possible pool of data to choose from. We partitioned the data randomly into three sets: training, test and pool. For this, we randomly sampled 75% of the users and then added all of their ratings to the training set. These represent the ratings for the users that are *already in the system*. Each of the remaining 25% *test users* were initialized with a single item, adding that rating to the training set. For each test user, we randomly selected three ratings and add these to the test set. The remaining ratings were added to the pool set. Figure 7.3 illustrates this setup. We also simulated new items arriving to the system. In this case the setup is identical except that the role of the users and items were interchanged. We denote the new-users and new-items experiments by appending -U and -I to the dataset names respectively.

HOMF was adjusted using the available ratings in the training set. Then, during each active learning iteration, a single rating was selected from the pool set for each test user using an active learning strategy. The selected ratings were then added to the training set and HOMF is incrementally re-adjusted using the new training set. We evaluated the second term in Equation (7.7) using Monte Carlo sampling from q with 100 samples. As alternatives to BALD we consider random sampling (*Rand*) maximum entropy sampling (*Entropy*), and a model-free version of Entropy that selects the item whose empirical rating distribution in the training data has the largest entropy (*Emp-Ent*). On each active learning iteration, we compute the average log likelihood on the test set. The whole procedure was repeated 25 times.

Active Learning Strategies

Figures 7.4 and 7.5 show the learning curves for each strategy on each new-user and new-item experiment respectively. Table 7.2, left hand columns, summarize the results with the average test log likelihood after drawing 10 samples from the pool set of each test user (-U) or item (-I). With HOMF, BALD yields the best (or joint best) predictions in all but one case. Both the model based and empirical methods for entropy sampling often perform poorly in our experiments because they ignore the inherent noisiness in the users or items.

Note that in most datasets most users have only a few ratings available. This means that BALD is restricted to sampling from a limited pool set. In particular,

	Heteroscedastic (HOMF)				**Homoscedastic (OMF)**				**BMM**			
	BALD	**Entro**	**Emp-Ent**	**Rand**	**BALD**	**Entro**	**Emp-Ent**	**Rand**	**BALD**	**Entro**	**Emp-Ent**	**Rand**
Book-U	**-2122**	-2129	-2129	-2126	-2146	-2149	-2150	-2147	-2405	-2418	-2413	-2411
Dating-U	**-1214**	-1239	-1241	-1248	-1217	-1230	-1235	-1244	-1234	-1309	-1305	-1255
IPIP-U	**-1944**	-1977	-1960	-1967	**-1945**	-1978	-1964	-1973	-1964	-1988	-1983	-1987
Jester-U	-2051	-2095	-2070	-2064	-2080	-2119	-2100	-2099	**-2041**	-2075	-2054	**-2045**
MLens100k-U	**-918**	-928	-926	**-920**	-926	-927	-929	-926	-989	-1001	-997	-988
MLens1M-U	**-1831**	-1843	-1844	**-1835**	-1840	-1850	-1854	-1846	-1877	-1899	-1898	-1879
MTweets-U	**-1467**	-1475	-1475	**-1471**	-1503	-1508	-1508	-1503	-1608	-1624	-1622	-1613
Webscope-U	**-1837**	-1869	-1869	-1846	-1882	-1898	-1903	-1880	-1951	-1984	-1970	-1958
Book-I	**-2038**	-2039	**-2037**	**-2038**	-2095	-2094	-2094	-2095	-2186	-2198	-2202	-2195
Dating-I	-1630	-1720	-1655	**-1612**	-1672	-1722	-1684	-1643	**-1603**	-1691	-1631	**-1602**
IPIP-I	**-319**	-325	-339	-329	-325	-325	-339	-330	-335	-347	-346	-339
Jester-I	**-99**	**-99**	**-99**	**-100**	-102	-102	-101	-102	-104	-107	-106	-104
Mlens100k-I	**-1085**	-1103	-1095	-1099	-1110	-1112	-1111	-1113	-1160	-1186	-1171	-1170
Mlens1M-I	**-1831**	-1843	-1844	**-1835**	-1840	-1850	-1854	-1846	-1877	-1899	-1898	-1879
MTweets-I	**-1470**	-1479	-1475	-1476	-1519	-1520	-1520	-1520	-1605	-1617	-1613	-1608
Webscope-I	**-1837**	-1869	-1869	-1846	-1882	-1898	-1903	-1880	-1951	-1984	-1970	-1958
Wins / 16	15	1	2	7	15	7	5	12	16	1	2	12

Table 7.2: Log likelihood after receiving 10 samples. Underlining indicates the top performing active sampling algorithms for each model, and bold denotes the best overall method. The bottom row gives the number of datasets on which each strategy yields the best (or joint best) performance with each model.

Book, MovieTweets and Webscope are the most sparse, with only 2, 3 and 5% of ratings available respectively. Unsurprisingly, BALD exhibits smaller performance gains on these datasets. In practice, in most of these domains, most users would be able to provide ratings for a larger number of items; they may watch a new movie, listen to a song, read a book, etc. Consequently, we would expect to see larger gains in performance on these sparse datasets like in the denser datasets, such as IPIP and Jester. However, this assumption may not always hold, for example, in dating recommendation a user may not take any recommendation to provide a rating. In this case, the probability of receiving a rating should be accounted for; this is a subject of future research.

In cold-start learning it is crucial to elicit useful information from the very first sample, so as not to deter the user with multiple requests for information. The average (over datasets) number of queries required to achieve the same predictive performance as the first active sample chosen by BALD is 1.85 with entropy, 1.83 with Emp-Ent and 1.59 with random sampling. This means that on average around 60% more random samples are required to gain the same performance as the first sample from BALD.

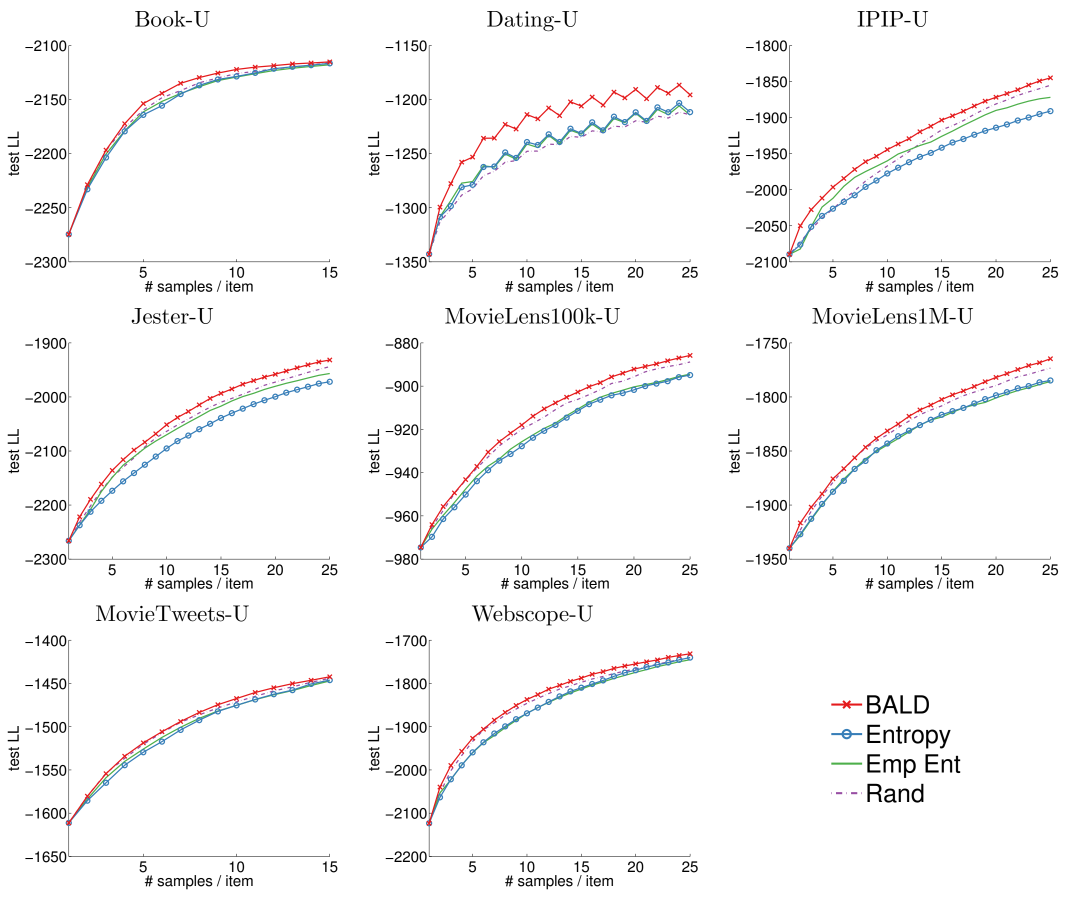

Figure 7.4: Predictive log likelihood with new users after each round of active sampling for each selection algorithm. The x-axis is truncated to 15 active samples when the methods have converged within that number.

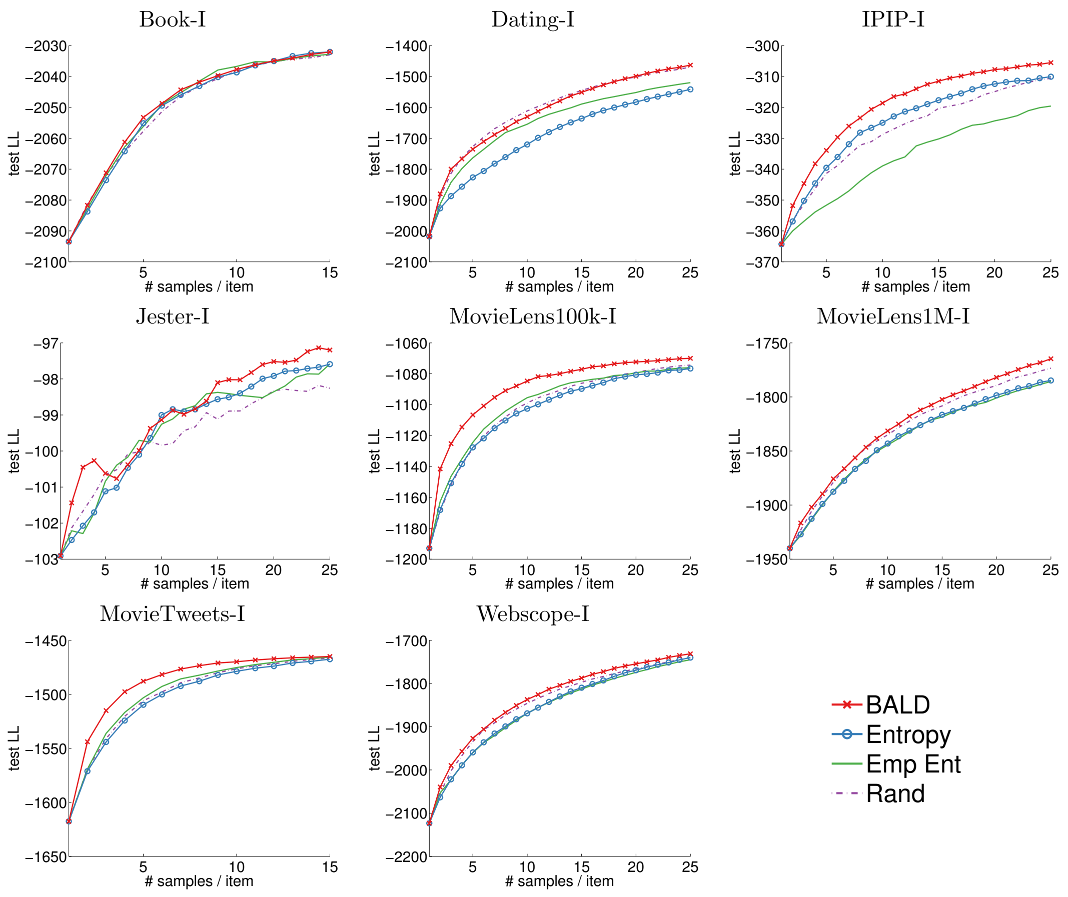

Figure 7.5: Predictive log likelihood with new items after each round of active sampling for each selection algorithm. The x-axis is truncated to 15 active samples when the methods have converged within that number.

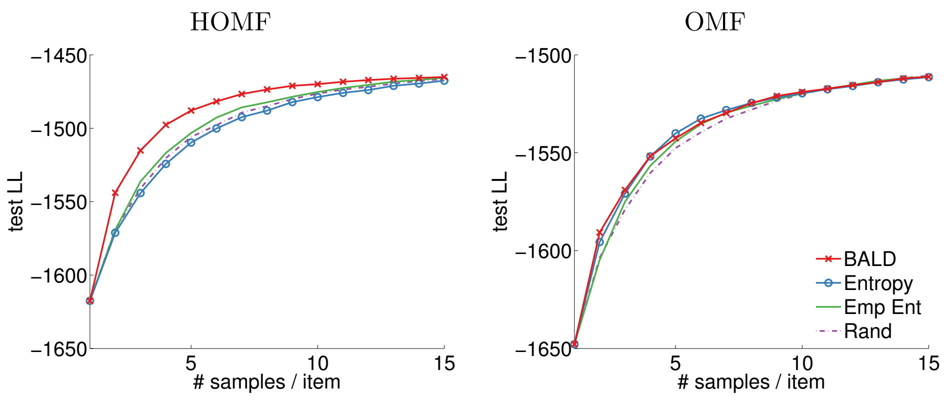

Figure 7.6: Comparison of active learning strategies when using HOMF and OMF with the MovieTweets-I dataset.

Heteroscedasticity vs. Homoscedasticity

We run each active learning strategy with the homoscedastic version of our model, OMF, and the homoscedastic method BMM. Table 7.2 contains the results for all methods. With the homoscedastic models active learning significantly outperforms random sampling on fewer datasets than with the heteroscedastic model. This demonstrates that accurate estimates of the intrinsic noisiness of the data are required to unlock the full potential of Bayesian active learning. Figure 7.6 presents example learning curves on MovieTweets-I, where the difference in relative performance of BALD and Rand when using HOMF and OMF is large. One can see that BALD provides much faster learning than the other strategies with HOMF, but all strategies are indistinguishable with OMF. This indicates that some users in this dataset provide highly noisy ratings. HOMF is able to model this and elicit ratings only from the useful, low noise users.

Root Mean Squared Error

We also used root mean squared error (RMSE) to evaluate performance. RMSE scores only the predictive mean and discards confidence. Furthermore, unlike log likelihood, it is not invariant to the (normally arbitrary) assignment of numeric values $\{1, \ldots, 5\}$ to ordinal valued ratings. Tables 7.3 and 7.4 contain the RMSE values for the model comparison and cold-start experiments respectively.

With RMSE, the best performing model is OMF, very closely followed by HOMF. Learning heteroscedasticity does not appear to effect the predictive mean, just the predictive confidence. We speculate that the small improvement of OMF over HOMF with RMSE is due to the fact that OMF has fewer parameters to learn.

Dataset	**HOMF**	**OMF**	**HOMF -NoB**	**OMF -NoB**	**Paquet**	**RBMF**	**BMF**	**BMM**
Book	1.207	**1.204**	1.246	1.204	1.214	1.281	1.280	1.390
Dating	0.822	**0.821**	0.823	0.836	0.829	0.825	0.838	0.913
IPIP	0.886	**0.885**	0.887	0.887	0.887	0.893	0.895	1.046
Jester	1.019	**1.006**	1.015	1.008	1.009	1.016	1.015	1.078
MLens1M	0.838	0.836	0.839	0.837	**0.836**	0.842	0.847	0.965
MLens100K	**0.895**	**0.894**	**0.895**	0.895	**0.895**	0.898	0.903	1.077
MTweet	0.699	**0.698**	0.701	**0.699**	0.703	0.712	0.722	0.817
WebScope	1.200	1.195	1.201	1.195	**1.185**	1.215	1.218	1.283

Table 7.3: Average test RMSE for the model comparison experiments.

	Heteroscedastic (HOMF)				**Homoscedastic (OMF)**				**BMM**			
Dataset	**BALD**	**Entro**	**Emp-Ent**	**Rand**	**BALD**	**Entro**	**Emp-Ent**	**Rand**	**BALD**	**Entro**	**Emp-Ent**	**Rand**
Book-U	**1.185**	1.188	1.189	**1.187**	**1.186**	1.188	1.189	**1.187**	1.345	1.352	1.353	1.350
Dating-U	**0.768**	0.788	0.790	0.795	**0.769**	0.783	0.788	0.794	0.789	0.841	0.838	0.807
IPIP-U	1.033	1.058	1.047	1.055	**1.030**	1.051	1.042	1.050	1.063	1.080	1.075	1.085
Jester-U	**1.089**	1.119	1.103	1.103	**1.086**	1.110	1.100	1.101	1.121	1.140	1.130	1.129
MLens100k-U	**0.968**	0.983	0.979	**0.974**	0.975	**0.973**	0.977	**0.973**	1.047	1.054	1.053	1.043
MLens1M-U	**0.888**	0.897	0.899	0.894	**0.890**	0.895	0.898	0.894	0.916	0.926	0.926	0.917
MTweets-U	**0.704**	0.708	0.708	0.706	**0.703**	**0.704**	**0.704**	**0.703**	0.777	0.775	0.783	0.774
Webscope-U	**1.192**	1.213	1.217	1.201	**1.199**	1.205	1.216	**1.198**	1.290	1.313	1.311	1.291
Book-I	1.175	1.175	**1.174**	**1.175**	**1.175**	**1.174**	**1.174**	**1.175**	1.250	1.256	1.258	1.254
Dating-I	**0.910**	0.962	0.941	0.914	0.924	0.951	0.937	**0.909**	0.966	1.019	0.989	0.963
IPIP-I	**1.039**	1.066	1.121	1.088	1.058	1.059	1.122	1.089	1.102	1.163	1.155	1.125
Jester-I	**1.086**	**1.100**	**1.095**	**1.108**	**1.105**	**1.101**	**1.096**	1.113	1.155	1.175	1.176	1.162
Mlens100k-I	**0.943**	0.960	0.955	0.957	0.953	0.954	0.954	0.957	1.004	1.030	1.016	1.015
Mlens1M-I	**0.888**	0.897	0.899	0.894	**0.890**	0.895	0.898	0.894	0.916	0.926	0.926	0.917
MTweets-I	**0.721**	0.725	0.724	0.724	0.725	0.725	0.725	0.725	0.768	0.774	0.774	0.769
Webscope-I	**1.192**	1.213	1.217	1.201	**1.199**	1.205	1.216	**1.198**	1.290	1.313	1.311	1.291
Wins /16	15	1	2	6	15	10	5	9	16	2	0	11

Table 7.4: RMSE after receiving 10 samples in the cold-start active learning experiments.

In cold-start active learning, when evaluated with RMSE heteroscedasticity is still important for gaining improved predictive performance with BALD. Table 7.4 indicates this in two ways: first, within the HOMF model, BALD outperforms Rand in more cases than it does with OMF. Second, HOMF+BALD outperforms OMF+BALD overall but HOMF+Rand loses to OMF+Rand. Thus, although OMF achieves a slightly better RMSE with random sampling, adding heteroscedasticity yields a greater performance gain from BALD. In summary, although heteroscedasticity does not assist in the final evaluation of RMSE, it is still necessary to enable BALD to find the informative ratings.

Speeding Up the Computation of BALD

In online settings, the time available for selecting the most informative matrix entries may be limited. In this case, we can reduce the cost of BALD by making approximations when computing the second term in the utility function in Equation (7.7), $\mathbb{E}_{q(\mathbf{u}_u)}\mathrm{H}[p(r^\star_{u,i}|\mathbf{u}_u)]$, as described in Section 7.2.1. We evaluate the accuracy of three approximations: Monte Carlo (MC) sampling, the unscented approximation, and evaluating the integral with a delta function located at the mode of q. We are interested in finding the most informative item, so we evaluate the error in the estimation of Equation (7.7) using fraction information loss, measured as:

$$\frac{\max_i \hat{I}(i) - \hat{I}(\operatorname{argmax}_i I(i))}{\max_i \hat{I}(i)}\,, \tag{7.9}$$

where $I(i)$ is the value of Equation (7.7) evaluated on item i using the approximation under analysis and $\hat{I}(i)$ is the gold standard obtained with MC with a separate set of 1000 samples. The results are averaged over all test users. The average losses across all datasets (± 1 s.d.) from MC with 100 samples, the unscented approximation and the posterior mode approximation were 0.017 ± 0.007, 0.035 ± 0.031 and 0.136 ± 0.073 respectively. Figure 7.7 depicts the loss as a function of the number of evaluations of $\mathrm{H}[p(r^\star_{i,j}|\mathbf{u}_i)]$ on the Book and Movielens100k datasets. Results for the other datasets are similar. When using MC sampling the integral converges rapidly, and the loss falls below 5% in fewer than 50 samples on all datasets. The unscented approximation requires only $2D + 1 = 21$ evaluations, and in most cases yields a better estimate than MC with this number of samples. In practice, we found no statistical difference in performance when running the experiments using the unscented approximation or MC with 100 samples. We therefore recommend the unscented approximation as an efficient solution for systems with computational constraints.

7.5 Conclusions and Extensions

We have used BALD to address the cold-start problem in recommender systems. The key to achieving good performance here is to accurately model all sources of uncertainty: the uncertainty in the parameters of the posterior distribution and the varying levels of intrinsic noise across the matrix. For this we have developed a new matrix factorization model that takes into account the ordinal nature of rating data. The proposed model uses hierarchical priors to provide robustness to fixing hyperparameter values, and we

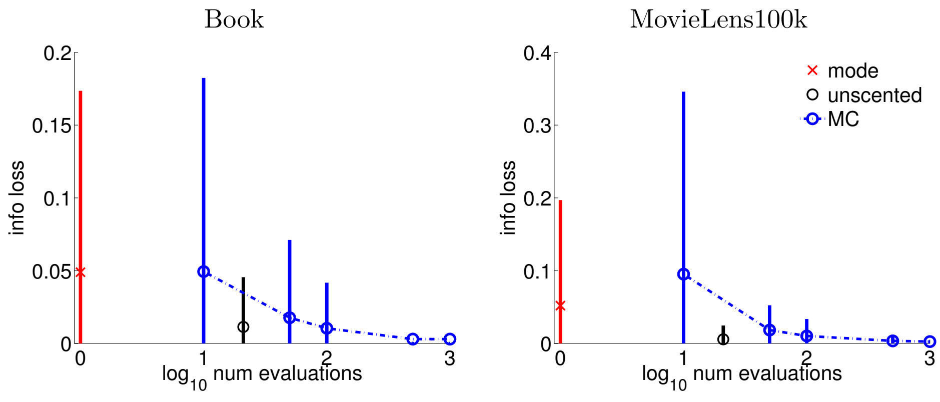

Figure 7.7: Information loss (7.9) from approximations to $\mathbb{E}_{q(\mathbf{u}_i)}\mathrm{H}[p(r^\star_{i,j}|\mathbf{u}_i)]$ versus the number of samples drawn from q. Vertical bars indicate the 10th and 90th percentiles.

use hybrid EP-VB inference to obtain accurate estimates of parameter uncertainty. The model also includes heteroscedastic noise to account for the varying noise levels across users and items. The model alone generates state-of-the-art predictions on rating data, and when combined with BALD acquires ratings useful for prediction from the very first active sample.

In this work we have assumed that users will rate any item. However, this will not always be the case. For example, in a holiday recommender system users will only rate locations they have already visited. Model-free approaches to this problem use hand crafted utility functions to balance informativeness with the chance of receiving a rating [Rashid *et al.*, 2002, 2008]. A simple probabilistic approach is to multiply the expected information gain with the predictive probability of observation [Harpale & Yang, 2008]. Although this method is theoretically optimal, in terms of expected information gain, in a pilot investigation we found that this approach is fragile. Any misscalibration between the model used to predict the probability of observation and the rating model caused the algorithm to perform poorly. Ultimately, it is the ranking of the items that matters for selection. Casting this task as a learning to rank problem with two (or more) models providing rankings based on different criteria could yield a more robust approach.

Another possible extension to HOMF is to data missing not at random. Most probabilistic models assume that the data is missing at random. In this case the observation mechanism can be ignored and one can condition upon the observed data alone without biasing inference. However, there is substantial evidence that in CF data is missing *not* at random [Marlin & Zemel, 2007]. For example, users may only rate movies that they like. In this case one cannot ignore the data observation mechanism without biasing inferences. We have recently published an approach that combines HOMF with the binary model presented in Chapter 5 to model the missing data and hence remove the aforementioned bias [Hernández-Lobato *et al.*, 2014a].

Chapter 8

Bayesian Exploratory Psychometrics

Psychometrics concerns the measurement and assessment of human psychological variables. These variables include personality traits, skills, IQ, opinions, happiness and education. Measurement of these traits involves designing psychological tests, such as questionnaires, and statistical models to interpret the test results. Most classical psychometric studies are limited to few (tens or hundreds of) subjects. Nowadays testing can be performed online, often through social media. These online questionnaires may be taken by thousands or millions of users, and so they yield new computational challenges in psychometrics.

With small traditional datasets, most psychometric techniques did not need to focus greatly upon computational scalability. Furthermore, new models of increasing complexity, such multi-dimensional models for ordinal-valued questionnaires, are under active research [Makransky *et al.*, 2013]. Scalable data analysis is required to take advantage of large subject pools and advanced modelling techniques. We address this problem using the machine learning tools developed in this thesis. In particular, we analyze a large dataset of personality questionnaires collected from Facebook. We use the Heteroscedastic Ordinal Matrix Factorization (HOMF) model, developed in Chapter 7, to leverage information across thousands of subjects.

Using HOMF and the Facebook data, we first visit a popular assumption in psychometrics, that there are five main personality factors (the Big Five). With a purely data-driven analysis, we investigate what conclusions may be drawn from using traditional factor analysis and HOMF. This is an exploratory analysis, in that we let the data drive the conclusions. This differs to a confirmatory analysis, in which the model is used to verify, or otherwise, a pre-determined hypothesis

In the second half of this chapter, we focus on a machine learning analysis; we compare the performance of different methods directly. We test how well HOMF, factor analysis, and state-of-the-art psychometric models can predict questionnaire responses. We also perform active questionnaire design, known as Computer Adaptive Testing (CAT) in psychometrics, and show that combining Bayesian Active Learning by Disagreement (BALD, Chapter 2) with HOMF results in an effective algorithm for multidimensional CAT with ordinal responses.

8.1 Background on Psychometrics

The Big Five personality traits and the IPIP questionnaires used to measure them are well-known principles in psychometrics. Here, a brief overview is provided with references to details.

8.1.1 The Big Five Personality Traits

A central objective in psychological research is to discover a unified structure in human personality. Several decades of research have sought a small number of *traits* or *facets* that summarize personality. Two main questions are: how many of these traits are there? And, what 'dimensions' of personality do these traits represent? Many independent studies have reported five traits. This conclusion has been drawn using a number of different methods, including lexical analysis, peer ratings, and self-report via questionnaires. A review of this work is presented in Digman [1990], in which they describe convergence to the 'Big-Five' factors [Goldberg, 1990; McCrae & Costa, 1987]. These five factors have become widely accepted, and are interpreted as 'Openness', 'Conscientiousness', 'Extraversion', 'Agreeableness' and 'Neuroticism' (OCEAN). These dimensions are sometimes partitioned into finer-grained traits [Costa & McCrae, 1992a], but the top-level clustering normally remains as the Big-Five factors.

8.1.2 IPIP Questionnaires

We examine questionnaire data for evidence of the Big-Five factors. We analyze IPIP-NEO questionnaires.[1] These questions were designed to assess the Big-Five factors directly [Costa & McCrae, 1992b; Goldberg *et al.*, 2006]. Thus, most models for this

[1]IPIP-NEO stands for International Personality Item Pool - Neuroticism, Extraversion & Openness. Agreeable and conscientiousness feature similarly, but are not included in the acronym.

data are confirmatory; the Big-Five are assumed and the models infer individual subject's traits. We perform an exploratory data-driven analysis where we do not assume that the Big-Five traits exist, but we infer the latent dimensions. However, as we show, the confirmatory nature of the IPIP questionnaires can introduce biases.

Our data consists of two IPIP-NEO questionnaires collected from Facebook. These were provided by the Cambridge University Psychometrics Centre as a part of the MyPersonality project [Kosinski *et al.*, 2013].[1] The questions have $R = 5$ responses that follow an ordinal Likert scale: 'strongly disagree', 'disagree', 'neutral', 'agree', 'strongly agree'. The first questionnaire consists of $P = 100$ questions (IPIP100) and has a large number of subjects, $N \approx 3\text{M}$. The second contains $P = 336$ questions (IPIP336) and has $N \approx 7\text{k}$ subjects.

8.2 Models and Metrics

We first overview the types of models encountered in psychometrics and describe in more detail the exploratory models that we compare with. Then, we present the metrics that we use to evaluate the models and analyze the data.

8.2.1 Item Response Models

Item Response Theory (IRT) concerns the design and analysis of tests to measure latent attributes [Baker, 2001]. IRT models seek to infer latent features in matrix data. Let $\mathbf{Y}$ be the $N \times P$ dataset of N users' responses to P questions. IRT typically assume two low rank matrices that underlie $\mathbf{Y}$. The first is an $N \times D$ matrix of latent factors for each user $\mathbf{X}$, where each row is a user's personality vector. The second is the $P \times D$ *factor loading* matrix $\mathbf{A}$, each row determines the influence of each personality dimension on the response to a question.[2] The latent trait and loading matrices are usually real-valued. Many IRT models are based upon a linear low rank factorization plus a non-linear likelihood function used to generate the responses, $\mathbf{Y} \sim p(\mathbf{X}\mathbf{A}^\top)$. IRT models differ along the following dimensions: The structure of latent matrices, the form of the likelihood function $f(\cdot)$, the assumptions about the data-type (real-valued, binary, discrete, or ordinal) and whether the factor loading matrix is known in advance.

[1] `www.mypersonality.org`

[2] We have changed the notation from Chapter 7 to be more consistent with psychometric literature. $\mathbf{X}$ equivalent to $\mathbf{U}$ in Equation (7.2) and $\mathbf{A}$ is equivalent to $\mathbf{V}$.

8.2.2 Unidimensional Models

The majority of IRT models are unidimensional, only a single latent trait influences each question. In this case the rows in the factor loading matrix $\mathbf{A}$ contain a single one, and the other entries are zeros. Knowledge of which trait influences which dimension is assumed, so the factor loading matrix is fixed. This is how the Big-Five factors are usually computed; each IPIP question is assumed to have a fixed influence on one of the factors. Unidimensional models have been developed for binary (dichotomous) or ordinal (polytomous) responses. Key models include the Graded Response Model [Samejima, 1969], (Generalized) Partial Credit Model [Masters, 1982; Muraki, 1992] and the Rating Scale Model [Andrich, 1978]. The models have different likelihood functions, whose parameters may be learnt. Recently in machine learning, expectation propagation has been used to train an expressive Bayesian unidimensional IRT model [Hennig, 2011].

Unidimensional models do not exploit correlations in the latent traits, or that a single item may be influenced by many factors. Thus, in recent psychometric research multidimensional IRT models have been developed [Reckase, 2009]. In confirmatory analyses, the factor loading matrix is assumed, but in exploratory analyses it is learnt. This requires the dimensionality reduction from P questions to D factors to be learnt, and HOMF provides a model to do this.

8.2.3 Factor Analysis

A well-known model for multidimensional IRT is Gaussian Factor Analysis (GFA), this model assumes that the responses are given by the linear dimensionality reduction plus Gaussian noise. The noise level can vary over the questions. This model may be used in a confirmatory or exploratory setting by fixing or learning $\mathbf{A}$, respectively. GFA is not strictly an question response model because the continuous likelihood cannot model binary or ordinal responses properly. Nevertheless, this model is popular due to its simplicity and computationally efficiency. Denote the indices of the observed entries in $\mathbf{Y}$ as $\mathcal{D}$ (although our datasets have no missing entries we may hold some out for testing), the GFA model is

$$p(\mathbf{Y}|\mathbf{X},\mathbf{A},\Sigma) = \prod_{u,i\in\mathcal{D}} \mathcal{N}(y_{u,i};\mathbf{x}_u^\top \mathbf{a}_i, \sigma_i^2)\,, \quad p(\mathbf{X}) = \prod_u \mathcal{N}(\mathbf{x};\mathbf{0}, I_{D\times D})\,, \tag{8.1}$$

where Σ is the set of variance parameters for each item $\{\sigma_1,\ldots,\sigma_P\}$ and $\mathbf{0}$ is a vector of zeros. A factorized Gaussian prior is used for the traits $\mathbf{X}$. Unlike in HOMF where we infer distributions over all of the latent variables, the factor loading matrix $\mathbf{A}$ and variance Σ are optimized by maximum likelihood. The expectations maximization (EM) algorithm is an efficient coordinate descent method for this optimization [Dempster *et al.*, 1977; Roweis & Ghahramani, 1999].

Because GFA does not model ordinal data, the responses are mapped to the real values $\{1, \dots, 5\}$. To make probabilistic predictions with GFA, the continuous predictive distribution is divided into five bins with boundaries $\{-\infty, 1.5, \dots, 4.5, +\infty\}$ and the mass in each bin is assigned to the corresponding rating.

8.2.4 Multidimensional Item Response Theory

Recently, multidimensional Item Response Theory (MIRT) exploratory models for non real-valued data have been developed. Two state-of-the-art methods are described in Chalmers [2012]. The first is a polytomous extension of the 2-Parameter Logistic (2PL) model [Birnbaum, 1968]. This is a 'graded' model that consists of a sequence of 2PL likelihood functions, MIRT-graded. The likelihood is

$$
\begin{aligned}
P(r_{u,i} = k | \mathbf{X}, \Theta) &= P(r_{u,i} \geq k | \mathbf{X}, \Theta) - P(r_{u,i} \geq k+1 | \mathbf{X}, \Theta)\,, \qquad (8.2) \\
\text{where } P(r_{u,i} \geq k | \mathbf{X}, \Theta) &= \sigma(\mathbf{x}_u^\top \mathbf{a}_i + d_{i,k})\,,
\end{aligned}
$$

and $\sigma(\cdot)$ denotes the logistic sigmoid function. The set of model parameters Θ contains $R-1$ offsets, also known as 'difficulties', for each item $\{d_{i,k}\}$, and the factor loadings $\mathbf{A}$. Equation (8.2) is similar to the likelihood used by HOMF (7.1), except with a logistic rather than probit function. The offsets are similar to the item boundary variables. The main difference is the heteroscedasticity in HOMF, which is not present in MIRT-graded.

The second MIRT model is a multidimensional extension of the Generalized Partial Credit model, MIRT-GPCM. The likelihood function is

$$
P(r_{u,i} = k | \mathbf{X}, \Theta) = \frac{\exp\left\{(k-1)\mathbf{x}_u^\top \mathbf{a}_i + d_{i,k}\right\}}{\sum_{l=1}^{R} \exp\left\{(l-1)\mathbf{x}_u^\top \mathbf{a}_i + d_{i,l}\right\}}\,. \qquad (8.3)
$$

This model is also parametrized by a factor loading matrix and item difficulties. Equation (8.3) includes a softmax function, which is used for discrete observations with no natural order, such as in multiclass classification. The factors $(k-1)$ in Equation (8.3) encourage the model to respect the natural ordering of the ordinal data. This is because an increase in $\mathbf{x}_u^\top \mathbf{a}_i$ increases the probability of a higher valued response more than a lower value. The extra factors in Equation (8.3) make the likelihoods for MIRT-GPCM and MIRT-graded closely related. In a non-probabilistic setting, where the softmax is replaced with an argmax and the sigmoids are replaced with step functions, these two likelihood functions are equivalent [Antoniuk *et al.*, 2013].

An important difference between HOMF and the methods in Chalmers [2012] is the inference algorithm. Two algorithms are presented in Chalmers [2012] to infer the traits **X** and optimize the other variables in both MIRT models. The first is exact EM using Gauss-Hermite quadrature and the second is Metropolis-Hastings Robbins-Monro (MHRM) [Cai, 2010]. MHRM is stochastic algorithm, closely related to EM. We found that EM was infeasible with more than a few (5-10) latent dimensions because a large number of numerical quadratures are required. We show in our experiments that EP and HOMF attain much better solutions at higher dimensions than the MHRM algorithm also.

8.2.5 Evaluation of Model Fit

The marginal likelihood, or model evidence, is a popular measure of the quality of Bayesian models. This metric is often used to optimize hyperparameters, such as the Gaussian process kernel hyperparameters in Section 3.2 and preference kernel parameters in Section 6.6. However, GFA and MIRT optimize many parameters, such as the thresholds and loading matrix. When optimizing, maximizing the (marginal) likelihood can cause overfitting because adding more parameters will only increase the likelihood (if optimization is performed correctly). To assess the optimal number of dimensions D we need an evaluation metric that penalizes overfitting.

The Bayesian Information Criterion (BIC) is a classic metric. For a model $\mathcal{M}$ with parameters Θ it is

$$\mathrm{BIC}(\mathcal{M}) = -2\log p(\mathbf{Y}|\Theta) + k\log(N)\,, \tag{8.4}$$

where $k = |\theta|$, the number of optimized parameters of the model. A lower score indicates a better fit. BIC trades-off a large likelihood $p(\mathbf{Y}|\mathcal{M})$ with a small number of parameters. Although setting k is non-trivial due to invariances [Beal & Ghahramani, 2006], we may evaluate GFA using BIC. However, HOMF has no optimized parameters, and the model evidence is hard to approximate in this complex model. Therefore, we need a more model-agnostic metric.

Predictive power is a transparent metric with which to different models. We measure predictive power on both the observed (training) and unobserved (test) entries in **Y** using log likelihood,

$$\mathrm{LL}_{\mathrm{train}}(\mathcal{M}) = \exp\left\{\frac{1}{|\mathcal{D}|}\sum_{(u,i)\in\mathcal{D}} \log p(y_{u,i}|\Theta)\right\}, \tag{8.5}$$

$$\mathrm{LL}_{\mathrm{test}}(\mathcal{M}) = \exp\left\{\frac{1}{|\neg\mathcal{D}|}\sum_{(u,i)\in\neg\mathcal{D}} \log p(y_{u,i}|\Theta)\right\}, \tag{8.6}$$

where $\neg\mathcal{D}$ denotes the indices of the unobserved elements in $\mathbf{Y}$. We average the likelihoods for each datapoint in log space. This ensures that probabilistic models are rewarded most greatly for honest predictions of the uncertainty in $\mathbf{Y}$ [Dawid, 2007] (see Section 3.2.1). A more complex model will have a larger training likelihood, but a lower test likelihood if it is overfitting.

To discover the optimal number of latent dimensions D we need choose the single best model using the above metrics. With BIC, we select the model with the lowest (best) value. However, this may not work with predictive performance. Robust Bayesian models can integrate out unnecessary degrees of freedom and prune excess parameters (particularly with VB, that is part of the HOMF inference routine [MacKay, 2001]). With these models the test log likelihood may not be reduced if D is overestimated. Therefore, we propose a heuristic to select the best model from a set of models with different dimensions $\{\mathcal{M}_D\}$. We select the model with the smallest D whose test likelihood is statistically indistinguishable from the model with highest likelihood in $\{\mathcal{M}_D\}$,

$$D_{\mathrm{opt}} = \underset{D}{\operatorname{argmin}} \left\{\mathrm{LL}_{\mathrm{test}}(\mathcal{M}_D);\, p_{\text{t-test}}[\mathrm{LL}_{\mathrm{test}}(\mathcal{M}_D), \mathrm{LL}_{\mathrm{test}}(\hat{\mathcal{M}})] < 0.05\right\}, \tag{8.7}$$
$$\text{where } \hat{\mathcal{M}} = \underset{\mathcal{M}_D}{\operatorname{argmin}}\, \mathrm{LL}_{\mathrm{test}}(\mathcal{M}_D)\,,$$

and $p_{\text{t-test}}[\mathrm{LL}_{\mathrm{test}}(\mathcal{M}_D), \mathrm{LL}_{\mathrm{test}}(\hat{\mathcal{M}})]$ is the p-value of a t-test over experimental repeats on the likelihoods returned by the models $\mathcal{M}_D$ and $\hat{\mathcal{M}}$. The intuition is that by Occam's Razor, with all else being equal, the model with lower complexity is more likely to explain the data correctly.

The optimal dimensionality chosen by this criterion will depend on the choice of statistical test and confidence level, therefore it is unsuitable for making strong conclusions about the absolute optimal dimensionality. However, it may be useful to assess the existence of a finite optimal dimensionality, or make comparisons across datasets and models.

8.2.6 A Posterior Predictive Check

Posterior predictive checks (PPCs) are another class of methods for analyzing model fit [Box, 1980; Gelman *et al.*, 1996; Rubin, 1984]. Unlike predictive power, PPCs do not require data to be held-out during training. Furthermore, they provide more information about the model misfit than the scalars provided by BIC, Bayesian model evidence or log likelihood.

PPCs can assess particular characteristics of the model that we care about. A general framework to do this involves sampling data from the model and comparing chosen statistics of the sampled data to the training data [Rubin, 1984]. Although these methods are flexible, they may require careful construction and expertise to interpret. Therefore we propose a simple PPC based upon p-values of the predictive distribution.

We introduce the test in a general form. Consider a model with fixed parameters Φ and per-datapoint vectors of latent variables $\mathbf{x}$. The predictive p-value is the cumulative distribution function (c.d.f.) of the model's predictive distribution for a scalar observation y given the parameters and latent variables, $\pi = \int_{-\infty}^{y} p(y'|\mathbf{x},\Phi)dy'$. If the data was generated by the model, then the distribution of predictive p-values over possible outputs and latent variables $(\mathbf{x}, y)$ will be uniform between 0 and 1, $p(\pi|\Phi) = \text{Unif}(0,1)$. This is intuitive, if the model correctly describes the data generating process then the outputs will be distributed according to model's predictive distribution. Marginalizing over $(\mathbf{x}, y)$, the distribution over π is

$$p(\pi|\Phi) = \mathbb{E}_{p(y)}\mathbb{E}_{p(\mathbf{x}|y)}\left[p(\pi|y,\mathbf{x},\Phi)\right] , \tag{8.8}$$
$$\text{where } p(\pi|y,\mathbf{x},\Phi) = \delta\left(\pi - \int_{-\infty}^{y} p(y'|\mathbf{x},\Phi)dy'\right) .$$

Suppose that the parameters are unknown, but we have a posterior distribution over them given some data $\mathcal{D}$ and the model $\mathcal{M}$. The distribution of p-values for the model is the expectation of Equation (8.8) under the posterior

$$\begin{aligned} p(\pi|\mathcal{M}) &= \mathbb{E}_{p(\Phi|\mathcal{D})}\mathbb{E}_{p(y)}\mathbb{E}_{p(\mathbf{x}|y)}\left[p(\pi|\mathbf{x},y,\Phi)\right] \\ &= \mathbb{E}_{p(y)}\mathbb{E}_{p(\Theta|y,\mathcal{D})}\left[p(\pi|y,\Theta)\right] \\ &\approx \frac{1}{|\mathcal{D}|}\sum_{y\in\mathcal{D}} \mathbb{E}_{p(\Theta|\mathcal{D})}\left[p(\pi|y,\Theta)\right] . \end{aligned} \tag{8.9}$$

In the second line, the parameters and latent variables are collected into a single set $\Theta = \{\Phi, \mathbf{x}\}$. In practice the true data distribution $p(y)$ is unknown, so in Equation (8.9) we estimate $p(\pi|\mathcal{M})$ using the training data.

With the polytomous MIRT models and HOMF, the expectation in Equation (8.9) is intractable. Therefore we approximate the distribution of predictive p-values further by sampling parameters Θ,

$$p(\pi|\mathcal{M}) = \frac{1}{|\mathcal{D}|} \sum_{(u,i)\in\mathcal{D}} \mathbb{E}_{p(\Theta|\mathcal{D})} \left[p(\pi|y_{u,i}, \Theta) \right]$$
$$\approx \frac{1}{|\mathcal{D}|S} \sum_{(u,i)\in\mathcal{D}} \sum_{s=1}^{S} p\left(\pi|y_{u,i}, \Theta^{(s)}\right), \quad \Theta^{(s)} \sim q_{\mathcal{D}}(\Theta)\,. \tag{8.10}$$

To obtain an unbiased estimate we should sample from the true posterior $p(\Theta|\mathcal{D})$. However, we perform approximate inference, such as EP, with the ordinal matrix factorization models. Therefore we sample from the posterior approximation $q_{\mathcal{D}}(\Theta)$ to compute Equation (8.9). We could perform approximate inference by sampling from the exact posterior and then re-using those samples for the PPC. We do not do this because we want to evaluate both the models and inference algorithms for fitting the data. Equation (8.10) does not involve a test set, so the models may be trained on all of the elements in $\mathbf{Y}$.

If the model correctly describes the generative process and the inference algorithm returns an accurate posterior approximation, then $p(\pi)$ will be a uniform distribution. If the model is inaccurate then the p-values will be non-uniform. If the model underfits, the p-values will lie around 0.5 and if it overfits, the test data will appear unexpected and the p-values will lie close to zero or one. However, this is not always the case. If the likelihood has the correct functional form and the model learns the noise level, then model mismatch can be accounted for by adjusting the noise. For example, data generated by GFA is marginally Gaussian, that is $p(y|A) = \mathbb{E}_{p(\mathbf{x})} p(y|\mathbf{x}, A)$ is Gaussian. If we trained a GFA model with too few dimensions, the data would appear more noisy, but the marginals would still be Gaussian. We would overestimate Σ in Equation (8.1) and the p-values would still be uniform. Therefore, this test is a necessary but not sufficient condition for model correctness. It controls for type-I error and will not reject a correct model, but it does not control for type-II error, an incorrect model may pass the test.

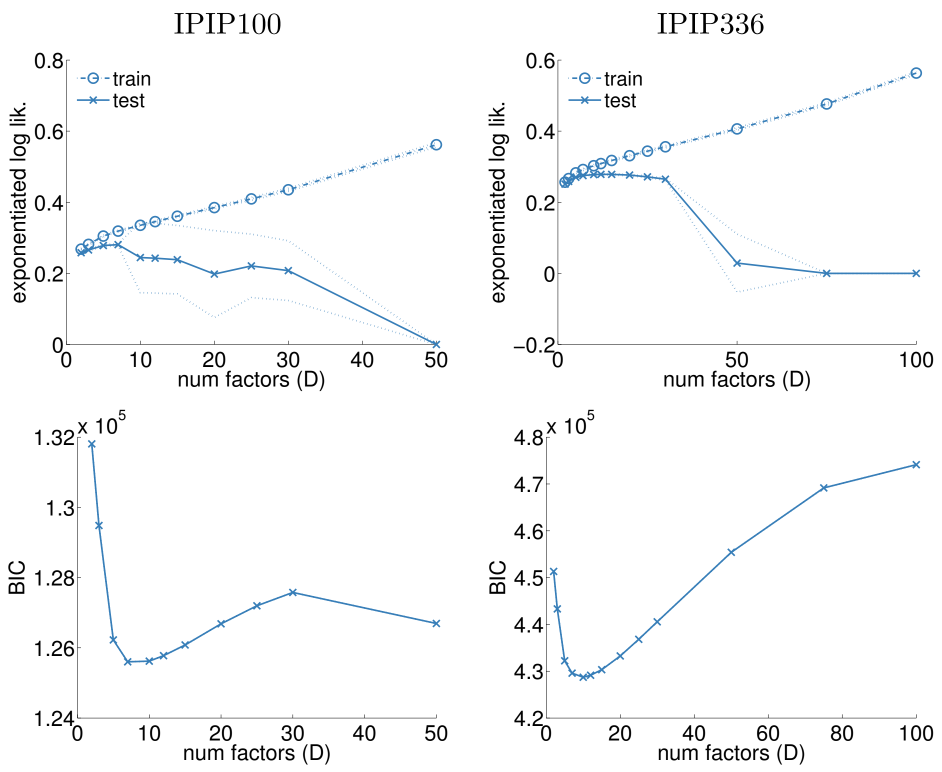

Figure 8.1: Predictive likelihood (top) and BIC (bottom) with the IPIP100 (left) and IPIP336 (right) questionnaires using GFA with different latent dimensionalities. $N = 500$ subjects were used in all cases. Dotted lines indicate ± 1 standard deviation across experimental repeats.

8.3 Experiments and Analysis

We first present exploratory experiments in which we seek evidence for the optimal number of factors. We then investigate the models' abilities to make accurate predictions and test our adaptive questionnaire design.

8.3.1 Exploratory Analysis

Evidence for Five Factors?

We first perform an experiment that indicates that a small number of dimensions, five to ten, is optimal. However, we show that this conclusion follows only when there is too little data and the model is inappropriate.

We randomly subsampled N subjects with completed questionnaires, so the data matrix $\mathbf{Y}$ had no missing entries. Some subjects had taken the 100-question test

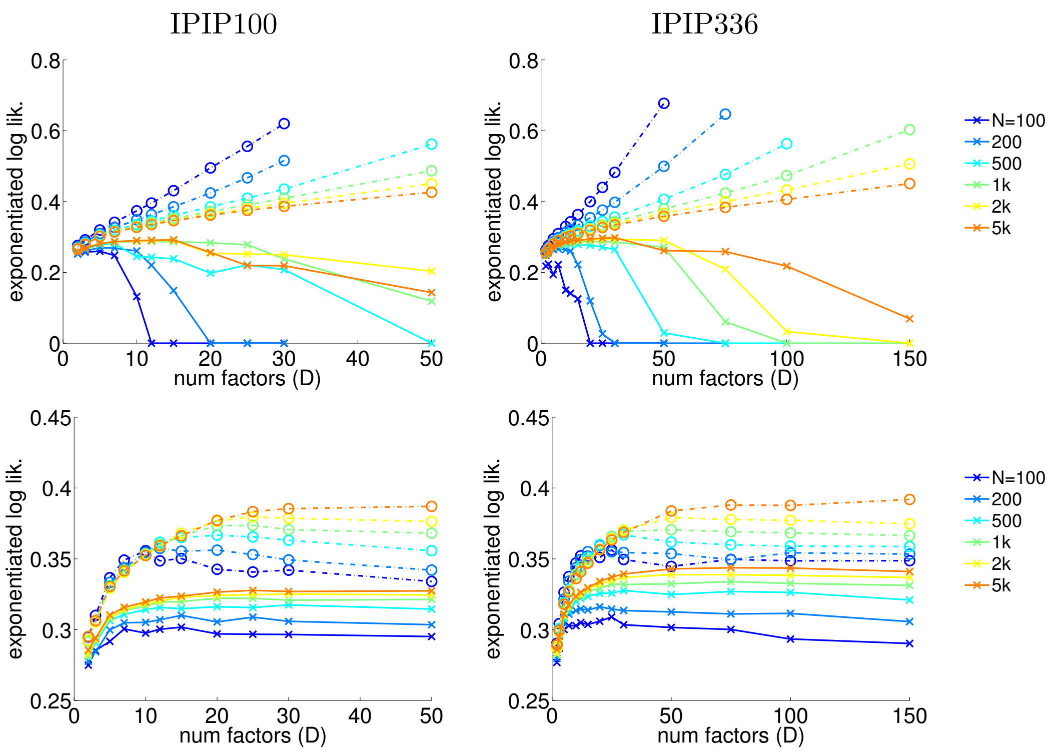

Figure 8.2: Log likelihood as a function of the number of traits on IPIP100 (left) and IPIP336 (rights). Each curve indicates a different number of subjects N used. *Top*: using GFA. *Bottom*: using HOMF. Dashed lines with circular markers give the training LL, solid with crosses show the test LL.

(IPIP100) multiple times, we used their first questionnaire sitting. We randomly split the data 80 : 20 into training and test sets. We trained GFA models over a grid of dimensionalities D and assessed the fits using BIC and LL. In each experimental repeat the subjects and data splits were resampled.

Figure 8.1 shows the results with 500 subjects on the IPIP100 and IPIP336 datasets. The training likelihood increases with D in both datasets. However, the maximum test likelihood occurs between 5 and 10 dimensions. The rapid fall-off in test LL occurs because the log likelihood heavily penalizes overfitting; if the model assigns very low probability to *any* response the overall likelihood will be close to zero. BIC also indicates that the optimal dimension is around $D \in [5, 10]$.

We now use larger datasets, and fit HOMF instead of GFA. Figure 8.2 shows the predictive performance versus dimensionality for different dataset sizes. Figure 8.2 indicates that HOMF is much more robust to overfitting than GFA in two ways. First, with increasing data the test likelihood for GFA improves, but the training likelihood decreases. With HOMF, a greater training likelihood always implies a greater test likelihood and increasing N increases both. Second, GFA fails (attains a test likelihood of zero) when D grows too large. The failure point moves to larger dimensionalities with more data, but even with 5k subjects the test predictive performance peaks at around $D = 20$. HOMF, on the other hand, is robust to over-specification of D. Even with 100 subjects and 50 dimensions, HOMF shows little signs of overfitting. This robustness advocates integrating over uncertain parameters, and not optimizing them.

Regarding the optimal dimensionality, the dimension at which GFA fails increases as the dataset size grows. With HOMF, the dimensional at which test likelihood plateaus is more consistent across different dataset sizes, but it appears to be larger than $D = 5$. We conclude that if one uses too little data and a classical model that optimizes many parameters then one can incorrectly conclude that a small number of dimensions, around 5, is optimal. This is not the case with more data or a better model, which we investigate further in the next section.

True Latent Dimension

We first check that the model selection criteria in Section 8.2.5 can uncover the true dimensionality with in-model data. We created ordinal datasets from the generative process assumed by HOMF. We consider three datasets, two with $P = 100$ items: *synth100-D10* and *synth100-D50* which were generated using 10 and 50 latent traits respectively, and one dataset with $P = 336$ and 10 latent traits, *synth336-D10*. We trained HOMF and GFA using the same experimental procedure as above, using a grid of different dimensions and number of subjects. Using BIC (for GFA only) and Equation (8.7) we choose the best model for each dataset size.

Figure 8.3 shows the results. BIC often overestimates the number of dimensions. BIC can be imprecise due to parameter counting difficulties for estimating k in Equation (8.4). In our experiments we subtracted D^2 degrees of freedom to account for the arbitrary rotation of the factor loadings and traits, and normalizing the mean of the data to zero. Nevertheless, BIC is only accurate in the easiest case, synth336-D10, where the data is largest and dimensionality is smallest.

The test log likelihood requires more data to provide evidence for a more latent traits. When the true dimensionality is 10, the likelihood-based criterion discovers the correct value with fewer than 500 subjects. When the true latent dimension is 50, HOMF requires 5k subjects to infer the dimensionality correctly, but GFA is unable

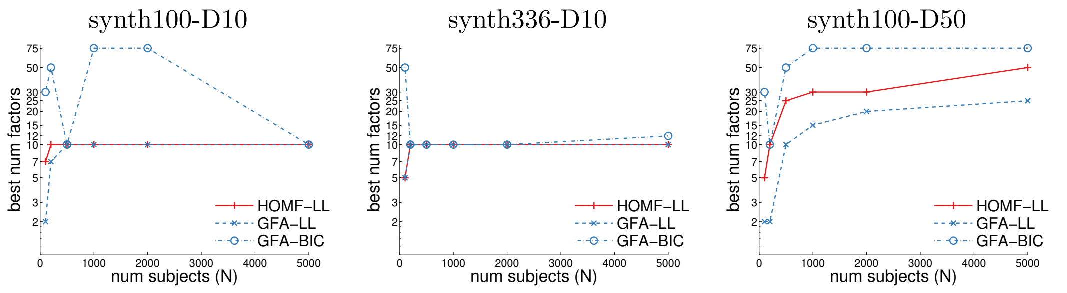

Figure 8.3: Optimal number of dimensions as a function of the number of users in the dataset. Each curve denotes a different model or metric used to select the optimal dimension. y-axis labels indicate the grid of dimensions used. Data is generated from HOMF with $P = 100, D = 10$ (left), $P = 336, D = 10$ (centre), and $P = 100, D = 50$ (right).

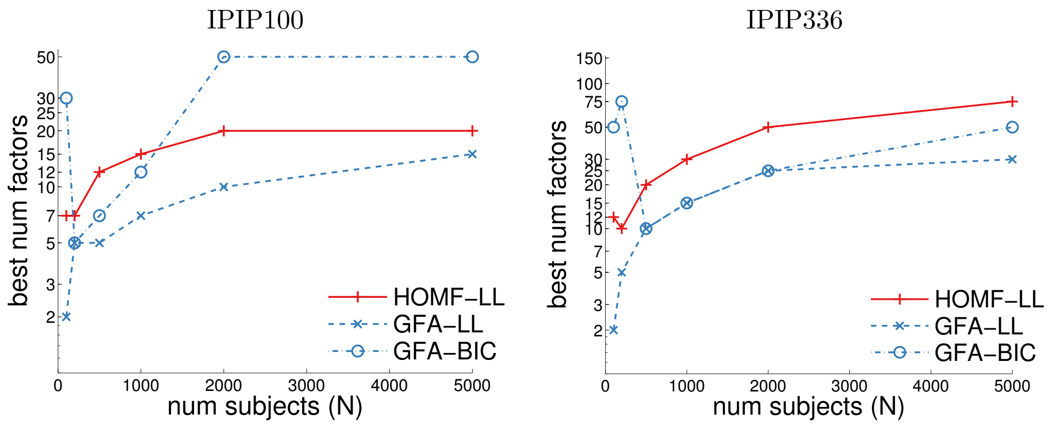

Figure 8.4: Optimal number of dimensions as a function of the number of users in the IPIP100 dataset (left) and IPIP336 (right).

to do so with up to 5k subjects. This is unsurprising because the data was generated using HOMF and not GFA. The fact that HOMF finds the correct number of dimensions with sufficient data indicates that the heuristic selection criterion in Equation (8.7) is sensible.

Figure 8.4 shows the same experiment but on the real datasets. Using the likelihood-based selection criterion, HOMF indicates that around 20 dimensions are optimal on IPIP100, but selects 50-75 dimensions on IPIP336. If the true dimensionality is low, then the synthetic experiments indicate that HOMF uncovers the correct dimensionality regardless of the number of questions. Therefore, this experiment indicates there is no fixed *small* number of latent traits.

These conclusions assume that the models' structures are correct. A central assumption in both GFA and HOMF is that the responses are functions of linear mappings of the latent trait vectors. If the mapping were nonlinear, these models would require more than the true number of dimensions to account for the misspecification. Most MIRT and matrix factorization models, however, assume a linear dimensionality reduction; our experiments indicate that in this case there is no small true dimensionality in our IPIP data.

Posterior Predictive Checks

We visualize the distribution of predictive p-values on the training set $p(\pi|\mathcal{M})$ computed using our PPC in Equation (8.10). We trained the models on the entire data matrix of 5000 subjects. Figure 8.5 shows the PPCs on the synth100-D10 and IPIP100 datasets. To assess uniformity we measured the KL-divergence between the distributions of p-values and a uniform distribution, $\text{KL}[\text{Unif}(0,1)||p(\pi|\mathcal{M})]$. This value was computed over the grid used in the histograms in Figure 8.5.[1]

Synth100-D10 is an ordinal matrix generated using HOMF with $D = 10$. As expected, when modelling this data using HOMF with $D = 10$, the distribution of p-values is almost uniform. There is slight upwards bow which may be due to EP-VB inference overfitting slightly. The pattern is identical with $N = 500$ (not plotted), indicating that this bow is not caused by too little data. With 50-dimensional HOMF the p-values are near-uniform. This is either because HOMF prunes excess dimensions or because the PPC can be insensitive to type-II error. The latter is observed when HOMF uses too few dimensions, $D = 5$, because the p-values are still near-uniform. HOMF appears to model the IPIP data well, the distributions appear uniform, and their KL-divergences are only $2 - 3$ times larger on IPIP than on in-model data.

Interestingly, GFA is more informative about the IPIP data with this test. On the synthetic data, GFA with too few dimensions ($D = 5$) yields a highly non-uniform distribution of p-values, as expected. However, when the dimensionality is correct ($D = 10$) or larger than the true value, the distribution is relatively uniform. The KL reduces by a factor of 14 from $D = 5$ to $D = 50$. Surprisingly, at $D = 50$ the KL is smaller than for HOMF. This indicates that the discretized Gaussian predictive distribution (Section 8.2.1) can approximate the ordinal likelihood well. However, on

[1]We also used a Kolmogorov-Smirnoff test to quantify uniformity. However, this test was too sensitive and gave $p < 10^{-30}$ in all cases because of the large number of samples (5M – 5k subjects, 100 questions, 10 parameter samples for each). Therefore this test was not informative about the relative uniformity of the distributions.

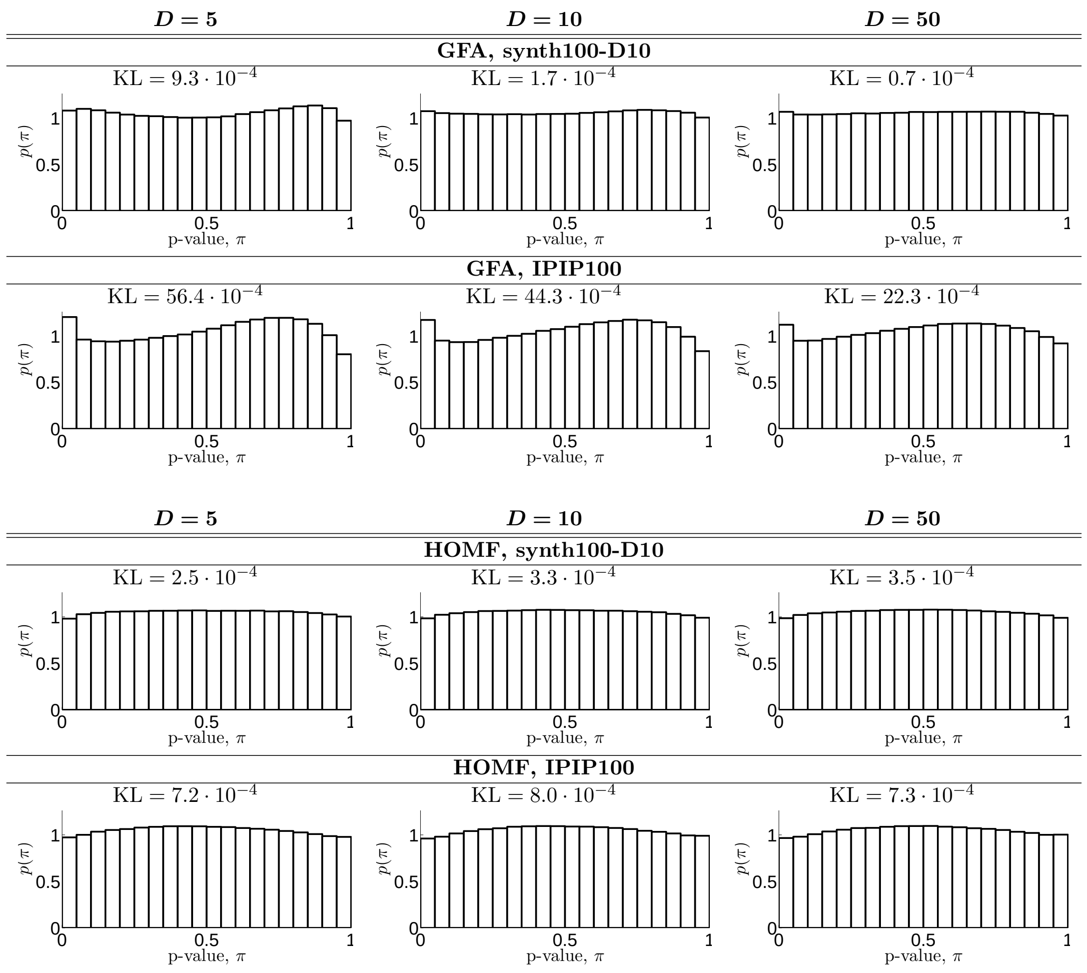

Figure 8.5: Empirical distribution of predictive p-values, Equation (8.10). *Top row:* GFA, synthetic dataset with $P = 100, D = 10$. *Second row:* GFA, IPIP100 dataset. *Third row:* HOMF, synthetic dataset. *Fourth row:* HOMF, IPIP100 dataset. Columns correspond to different latent dimensionalities.

the IPIP100 data, the p-values of GFA are highly non-uniform at all dimensionalities. On the dataset with a small true dimensionality, synth100-D10, GFA can compensate for an inappropriate likelihood when many dimensions are used. Therefore, the non-uniform p-values in Figure 8.5 for GFA $D = 50$ on IPIP100 also provides strong evidence that there is not a small number of linear latent factors in this data.

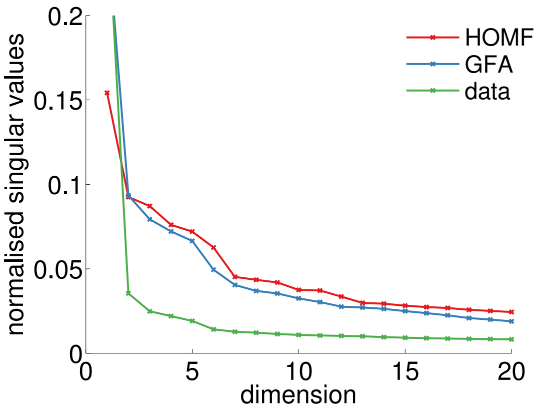

Figure 8.6: Normalized singular values of the data matrix $\mathbf{Y}$, and the factor loading matrices $\mathbf{A}$ returned by HOMF and GFA with 20 dimensions. In all cases 5000 subjects were used.

Dataset Bias

We finish this section with a caution regarding biases introduced by the confirmatory IPIP questionnaires. Singular Value Decomposition (SVD) is a non-probabilistic tool that can be used to estimate the true dimensionality of the data. SVDs decompose a matrix into orthogonal singular vectors with corresponding singular values. Informally, each singular vector's singular value indicates its prevalence in the matrix. If a $P \times D$ matrix only has $d < \min(P, D)$ singular values significantly larger than zero, then only d dimensions contribute substantially to the values in the matrix. In real-world matrices, the other $[\min(P, D) - d]$ dimensions could be artifacts of noise.

We perform an SVD of the IPIP data matrix $\mathbf{Y}$ and factor loading matrices $\mathbf{A}$ learnt by the models. The results are plotted in Figure 8.6 for $N = 5$k and $D = 20$. A kink appears in the curves in Figure 8.6 around $D = 5$ or 6. This could indicate that human personality has $5 - 6$ more dominant dimensions.

However, the IPIP-NEO questions are designed for confirmatory studies. Each question probes for one Big-Five trait. Questions that probe for the same trait are often very similar. For example, two questions for determining agreeableness ask whether the subject "holds a grudge" or "gets back at others", and two questions for conscientiousness ask whether the subject "completes tasks successfully" or "does things according to plan". Subjects' answers to these questions are very likely to correlate regardless of their personality. An SVD will find these five strong dimensions in the IPIP data, regardless of the structure of human personality.

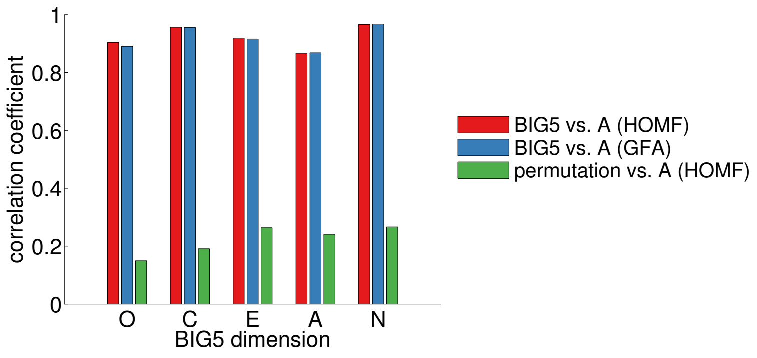

Figure 8.7: Correlation coefficient ρ of the factor loading matrices $\mathbf{A}$ learnt by each model with each dimension of the Big-Five factor loading matrix $\mathbf{A}_{\mathrm{BIG5}}$ and a random baseline. Each group of 3 bars corresponds to one Big-Five traits (columns of $\mathbf{A}_{\mathrm{BIG5}}$).

We can observe the Big-Five question structure directly in the factor loading matrices. The traits that each question in IPIP-NEO are designed to assess imply a particular factor loading matrix $\mathbf{A}_{\mathrm{BIG5}}$ with a single one per row (the responses to negative questions which would imply a -1 are reversed). We compute the correlation of each dimension in the inferred loading matrices with each dimension (O, C, E, A and N) of the Big-Five loading matrix $\mathbf{A}_{\mathrm{BIG5}}$. A large correlation indicates that these dimensions are recovered by the models. We need to account for rotations of the latent matrices in the exploratory models. To do this, we rotate the loading matrices such that the first five columns correlate maximally with the five columns of $\mathbf{A}_{\mathrm{BIG5}}$ using Canonical Correlation Analysis (CCA) [Thompson, 2005]. By chance, even random vectors of finite length will correlate to some degree. Therefore, as a baseline we perform the same test but with the rows of $\mathbf{A}_{\mathrm{BIG5}}$ permuted randomly. The models were trained with $D = 5$ and $N = 5000$ and correlation was evaluated using Pearson's linear correlation coefficient.

Figure 8.7 shows the results. The rotated factor loading matrices have much higher correlation with the Big-Five dimensions than the random matrix. This correlation can be observed directly in the factor loading matrices. Figure 8.8 shows 15 rows of: $\mathbf{A}$ learnt by HOMF, $\mathbf{A}_{\mathrm{BIG5}}$, and $\mathbf{A}$ rotated to so that it correlates maximally with the columns of $\mathbf{A}_{\mathrm{BIG5}}$. A clear resemblance between the rotated matrix and the Big-Five loading matrix can be seen.

One interpretation of these results is that the Big-Five traits are fundamental to human psychology. However, this presence of the Big-Five dimensions in the learnt factors loadings is probably due to the similarity between the questions for each trait.

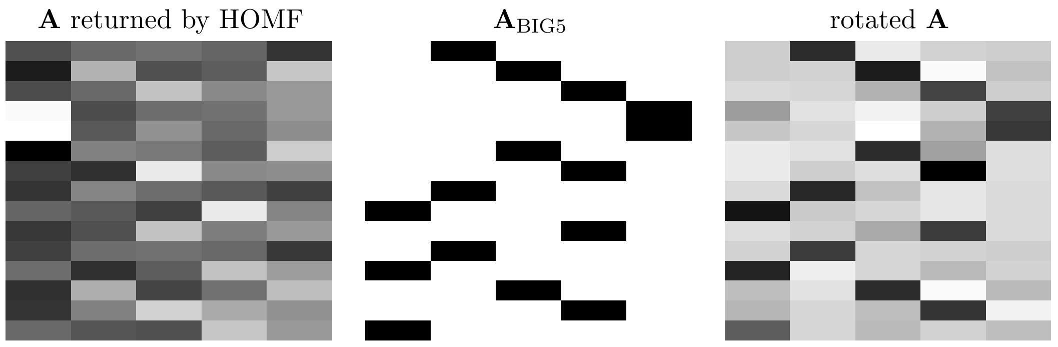

Figure 8.8: *Left:* First 15 rows of the factor loading matrices returned by HOMF (left). *Centre:* Loadings used by the Big-Five. *Right:* HOMF loading matrix rotated to correlate maximally with the Big-Five matrix.

When performing exploratory analysis with IPIP-NEO questionnaires one must be wary of this dataset bias and interpret evidence for the Big-Five dimensions in light of it.

In this section we have provided evidence that human behaviour on IPIP-NEO questionnaires is not governed by linear dimensionality reduction to a small number of latent traits. Nevertheless, although models for questionnaires may not capture the data generating process exactly (which is likely to be very complex), they are still useful for discovering structure in the data, inferring properties of the subjects, and making predictions. In the next section we pit HOMF directly against state-of-the-art MIRT models for making predictions.

8.3.2 Direct Model Comparisons

Weak and Strong Predictive Power

We test the models' abilities to make accurate predictions. We used IPIP100, which many subjects had taken multiple times, referring to each repeat as a 'sitting' of the questionnaire. We only used subjects with two or more than complete sittings. The repeated tests serve two purposes. First, they provide a model-free estimate of how well one can predict responses. Second, they allow us to evaluate both the predictive power on held-out questions from within a sitting, we call this *weak* predictive power, and across sittings, the *strong* predictive power. We compute the predictive powers using both exponentiated log likelihood, Equation (8.6), and 'fraction correct', the proportion of times that most probable (MAP) response predicted by the model is correct. The first sitting for each subject was used for training. 20% of the ratings were held out from the training set to evaluate the weak predictive power and the entire second sitting was used to evaluate the strong predictive power.

Test-retest Reliability

The test-retest reliability is the proportion of times that the subject's response is the same across questionnaire sittings. It is an estimate of the best possible predictive performance. For example, a subject who responds randomly will have a reliability of $1/R = 0.2$, and no model can perform better than that. In particular, if a subject provides response k with probability p_k, their reliability is

$$\text{reliability} = \mathbb{E}\left[\sum_{k=1}^{R} p_k^2\right] , \tag{8.11}$$

where the expectation is over the users and items in the dataset. The best possible fraction correct is achieved by selecting most likely response based on p_k. The score is then,

$$\text{best fraction correct} = \mathbb{E}\left[\max_k(p_k)\right] . \tag{8.12}$$

Equation (8.12) is an upper bound on Equation (8.11). Therefore the reliability is a lower bound on the best fraction correct achievable. An upper bound can also be derived from the reliability [Neri & Levi, 2006]. However, this bound may be loose and it returned values greater than one on our data, so we do not use it.

Subjects may be more consistent within a sitting, so the weak fraction correct cannot be compared to the reliability. Furthermore, it is harder to relate the reliability to the log likelihood, which will always be smaller than or equal to the fraction correct. However, we can provide a chance baseline for both metrics, presented in the next section.

Frequency-weighted Chance

A naïve chance baseline is $1/R = 0.2$. However, a better baseline takes into account the imbalance in the responses in $\mathbf{Y}$. This may still be considered 'chance' because the identities of the subject and question are ignored when making predictions. Under fraction correct, the best constant predictor assigns a point mass to the most frequent response, $p(y = k) = \mathbb{I}[k = \text{argmax}_i\, \hat{p}_i]$, where $\mathbb{I}[\cdot]$ is the indicator function and $\hat{p}_i$ is the empirical proportion of response i. This predictor attains a chance level of $\max_i \hat{p}_i$. Intuitively, if the data is more imbalanced, predictions are easier, and chance increases.

Under log likelihood, the best constant predictor assigns the empirical proportion to

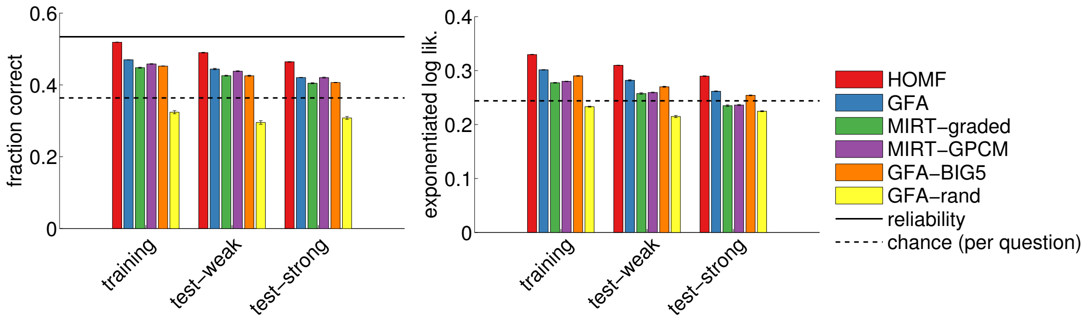

Figure 8.9: Weak (intra-questionnaire) and strong (inter-questionnaire) predictive powers using $D = 5$ dimensions in all models. Error bars indicate ± 1 s.d. across experimental repeats. *Left:* Fraction correct, solid horizontal line is the test-retest reliability, dashed line is the mean question-specific baseline. *Right:* Exponentiated log likelihood.

each response, $p(y = k) = \hat{p}_k$. The chance likelihood is then $\exp(-\mathrm{H}[\{\hat{p}_k\}])$, where $\mathrm{H}[\cdot]$ is the entropy function. These chance levels are improved further by making different predictions for each question, but ignoring the identity of the subjects. The predictions are now based on the empirical statistics in the corresponding columns of $\mathbf{Y}$. There are too few questions to compute a good user-specific baseline. In our experiments we computed these chance levels using the entire first sittings of the questionnaires.

Methods

In Section 7.4.1 we showed that HOMF outperforms a number of models for ordinal matrices developed in machine learning. In this section we focus on models used in psychometrics: GFA, MIRT-graded and MIRT-GPCM, see Section 8.2.1. The MIRT models were implemented using the R package described in Chalmers [2012]. We also investigate how well the data can be predicted using the Big-Five traits. To do this we use a unidimensional confirmatory GFA model with fixed loading matrix corresponding to the Big-Five measurements, as depicted in Figure 8.8, centre (GFA-BIG5). GFA-BIG5 only learns the latent traits $\mathbf{X}$ and the item noise levels Σ, and is constrained to five dimensions. As a baseline for GFA-BIG5, we run the same algorithm but using a random loading matrix with i.i.d. standard normal elements (GFA-rand). We used $N = 5000$ subjects, and repeated the entire procedure, including sampling of the subjects and dataset splits, five times.

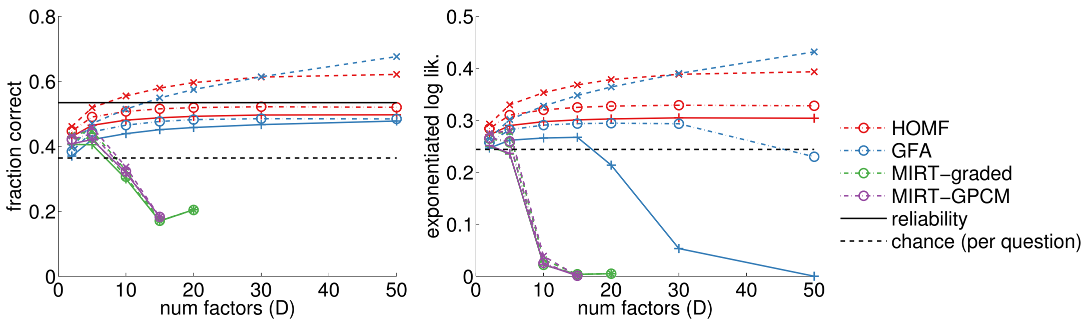

Figure 8.10: Training (dashed line, $\times$ markers), weak-test (dash-dot line, $\circ$ markers) and strong-test (solid line, $+$ markers) predictive powers using different latent dimensions (GFA-Big5 and GFA-rand not plotted because they are constrained to $D = 5$). *Left:* Fraction correct. *Right:* Exponentiated log likelihood.

Results

Figure 8.9 and 8.10 show the results using five latent traits and across dimensionalities, respectively. HOMF performs best by a substantial margin with both metrics. The MIRT models are significantly outperformed by HOMF at $D = 5$. Beyond five dimensions, the MIRT models perform very poorly, and failed to run with $D > 20$. Furthermore, they only beat GFA at very low dimensionalities, $D < 5$. MIRT-graded has a similar likelihood to HOMF (see Section 8.2.1), this indicates that the MHRM inference algorithm used by the MIRT models is ineffective as the dimensionality grows.

GFA-BIG5 substantially outperforms the baseline GFA-rand. As noted in the previous sections, the Big-Five dimensions are highly prevalent in IPIP data, so provide useful basis vectors. However, exploratory GFA improves upon GFA-Big5, which indicates that some questions provide information about multiple traits, which the multi-dimensional model can exploit.

According to log likelihood, HOMF is the only model that makes robust inter-questionnaire predictions at larger dimensionalities. The heteroscedasticity is likely to be contributing to HOMF's robustness. We use the re-tests to assess whether HOMF learns the noise levels correctly. We correlate the reliability of each subject with the MAP noise level for each subject returned by HOMF, γ^{row} in Equation (7.2). We do the same for the items. Note that HOMF only observes a single sitting of the questionnaire and so does not directly observe inconsistent behaviour.

Figure 8.11 shows the correlation coefficients using different dimensionalities. For the subjects, the learnt noise levels correlate negatively ($p < 0.05$) with the reliabilities, indicating that HOMF learns the noise correctly. For the questions, there is negative

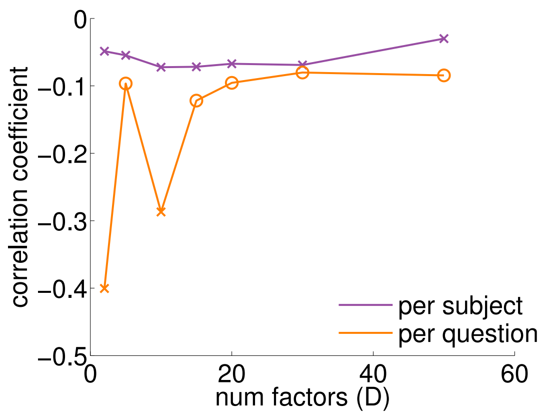

Figure 8.11: Pearson's correlation coefficient ρ between the test-retest reliability for each user and question and their corresponding noise level inferred using HOMF. Points marked with an $\times$ have significant correlation ($p < 0.05$) points with a $\circ$ are not significant.

correlation at low dimensionalities, but not at high dimensionalities. This may be because the intra-questionnaire response entropy for each item (empirical entropy of the columns of $\mathbf{Y}$) correlates negatively with the reliability ($\rho = -0.85, p < 10^{-20}$). At low dimensionalities the model cannot capture the response patterns, so models high entropy items with high noise. Therefore, the noise also correlates with the unreliable items. However, at high dimensionalities the model decouples response entropy from noise, and there are insufficient questions to attain a strong correlation with reliability.

8.3.3 Computer Adaptive Testing

Computer Adaptive Testing (CAT) concerns the design of active questionnaires. Many adaptive designs have been proposed for unidimensional models, see Gershon [2005] for a review. Recently, CAT for multidimensional models (MCAT) has been developed. Most MCAT algorithms apply to dichotomous (binary) response models [Segall, 2010]. Extensions have been proposed for a model similar to MIRT-GPCM [Wang & Chen, 2004]. In this work a D-optimal design is used learn optimally about the traits (see Section 2.4). Makransky *et al.* [2013] provide experimental evidence that with MCAT the Big-Five traits can be recovered from IPIP data with many fewer questions. However, most polytomous MCAT algorithms do not consider exploratory models where the parameters are learnt online. Current methods fix the item-parameters to values learnt *a priori*.

We propose the active learning algorithm in Chapter 7 for MCAT. This algorithm

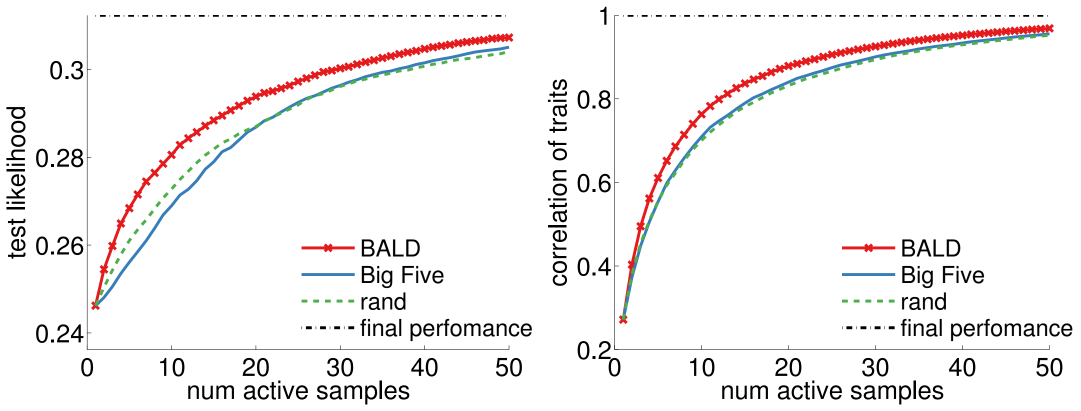

Figure 8.12: Performance of adaptive questionnaire designs versus number of selected questions. *Left:* Test exponentiated log likelihood, dash-dot horizontal line shows the likelihood after observing all of the questions in the pool. *Right:* Mean correlation of the traits with those learnt using all of the 95 questions in the pool.

combines HOMF with BALD to learn optimally about the user's latent traits. The advantages of this approach over MCAT designs include: (i) we can handle (high dimensional) multidimensional traits; (ii) we can work with polytomous or dichotomous responses, (iii) we update both the traits and other model parameters online, whilst focusing active learning on the users' traits. Here we consider exploratory learning, but the factor loading matrix can easily be fixed to apply HOMF to confirmatory analyses.

We validate our MCAT method on IPIP100 in a similar experiment to those in Section 7.4.2. We initially train HOMF ($D = 5$) on 1500 subjects. We then use BALD to select questions for 500 further 'test' subjects. These questions are selected from a pool of 95 questions per subject, the remaining 5 are held-out for evaluation using log likelihood. HOMF is incrementally re-trained after each response is collected. We repeated the experiment ten times.

Psychologists are often most interested in the subjects' latent trait vectors themselves $\mathbf{x}$ and not just the predictive power. Therefore, we also evaluated the methods using the correlation of the test subjects' trait vectors with the 'final' traits. The final traits are those inferred using all 95 questions in the pool. To account for model invariance due to the arbitrary rotations of the latent matrices, we use CCA to rotate the traits (posterior mean) $\mathbf{X}$ to correlate maximally with the final traits. We report the mean correlation returned by CCA over the five dimensions of $\mathbf{x}$.

Current MCAT methods do not handle polytomous responses with exploratory models. Therefore we compare to a simple heuristic (Big-Five) that selects questions that assess each of the Big-Five traits in turn. The intuition is that because the Big-Five dimensions are highly prevalent in the data, see Section 8.3.1, then probing each dimension equally will result in a reasonable questionnaire. We also use random sampling of questions as a baseline.

Figure 8.12 shows the performance versus the number of questions chosen by each method. The heuristic Big-Five is not effective in this exploratory setting and does not improve over random sampling. With BALD, one requires substantially fewer questions to gain the same performance as random sampling. To gain 90% of the predictive log likelihood of the entire pool we required 10 questions one average chosen by BALD, 14 by random sampling and 15 by the Big-Five heuristic. To achieve a 90% correlation with the final trait vectors we required 23 questions chosen by BALD, 31 by random sampling and 29 by the Big-Five heuristic. Thus, we achieve over a 25% reduction in the number of questions when using BALD instead of random sampling.

8.4 Conclusions and Extensions

From our experiments we concluded that the IPIP data does not support a small number of linear latent factors, as assumed by most factor analysis techniques. The two main sources of evidence are: First, our model selection heuristic in Equation (8.7) finds the correct dimension using HOMF on simulated data, but on IPIP the optimal dimensionality grows with the number of users N and questions P (up to $N = 5000$, $P = 336$). Second, with GFA our posterior predictive check behaves very differently on synthetic ordinal data with a low dimensionality ($D = 10$) than on the IPIP data.

We note that spurious evidence for five dimensions can be found. If one uses too little data and optimizes many parameters, as in GFA, then overfitting occurs at higher dimensionalities and lower dimensionalities appear better. Furthermore, the IPIP questionnaire is primarily designed for confirmatory analyses with the Big-five factors. Therefore, SVDs of the data or factor loading matrices indicate that there are around five more dominant factors. However, these factors correspond to the intended unidimensional factor loadings intended by the IPIP questions.

We next showed that HOMF substantially out-predicts state-of-the-art MIRT models. HOMF infers a distribution over all parameters, whereas the MIRT models optimize the item parameters, usually with EM-like routines. Our experiments indicate that our EP-VB algorithm is more robust than the MHRM algorithm, particularly at higher dimensionalities.

A central contribution of HOMF to psychometrics is that it differentiates structure in the matrix (interindividual variations) from noise (intraindividual variations). The validity of models that do not differentiate between interindividual and intraindividual variations (also known as trait versus state) has been questioned [Borsboom *et al.*, 2003]. Other models distinguish between these types of variation [Hamaker *et al.*, 2005, 2007], but these models are trained using extensive longitudinal studies with few subjects, for example $N = 22$ in Hamaker *et al.* [2007]. The heteroscedastic component in HOMF learns the intraindividual variability directly, and the interindividual variability is modelled by the latent factors. Our test-retest analysis (Figure 8.11) indicates that HOMF learns the intrasubject variability correctly without a longitudinal study.

Finally, we have proposed a new MCAT algorithm using BALD with HOMF. This routine yields a substantial reduction in the amount of data required to infer the latent traits. Unlike previous algorithms, we can handle polytomous data and online parameter learning in an exploratory setting.

Extending HOMF to confirmatory settings would be straightforward. We could then infer the Big-Five traits directly as well as the intraindividual variances and item noise. The quality of the inferred Big-Five traits could be assessed by making predictions on external variables (such as gender, age, location, number of friends etc.) [Kosinski *et al.*, 2013].

We only sampled a small fraction of the 3M subjects in IPIP100. HOMF was faster than the MIRT models we compared to, and took around 10 minutes to run on 5k subjects with $D = 10$. HOMF scales as $\mathcal{O}(\mathcal{D})$ which will be prohibitive on very large fully observed matrices. Parallelization would be required to scale to millions of users. However, even with more users, a fixed optimal number of traits may not appear if the true dimensionality reduction is non-linear. Nonlinear dimensionality reduction methods, such as kernel PCA, could be used [Mika *et al.*, 1998]. However, in general, kernel methods require inverting an $N \times N$ matrix, so scaling these methods to large numbers of datapoints is an active area of research.

One approach to overcoming the IPIP dataset bias (Section 8.3.1) would be to use a joint model. This model could have one component to capture the known correlations due to the construction of the questions, and another exploratory component to learn additional structure in the data. This approach has been used for modelling genetic regulatory factors and confounding environmental factors jointly [Fusi *et al.*, 2012].

Finally, recent work has questioned the existence of latent personality traits as a *cause* of human behaviour [Cramer *et al.*, 2012]. The authors posit that behaviour is governed by a network of cognitive and behavioural variables that depend on each other for causal or logical reasons. Personality traits then arise from the structure of this network and not due to some underlying hidden factors. In this manner, the responses to the IPIP-NEO questions can be modelled as a network of variables whose covariation is governed by the unknown network structure. Learning the structure of Bayesian networks [Friedman & Koller, 2003] or directions of causality [Pearl, 2000] are hard problems being tackled by current research. The application of new statistical tools, analogous to the ones developed here, to address these tasks could result in substantial advances in psychometrics.

Chapter 9

Conclusions

In this thesis we have tackled various applied problems and developed general algorithms and models. We first summarize these, then present three areas of future work to build upon this research.

9.1 Summary

Bayesian techniques for active learning and probabilistic modelling of matrices have been developed. With these methods we have addressed a number of engineering and scientific problems. Specifically, we have provided new active learning algorithms in the following domains.

- General regression and classification tasks with Gaussian processes.
- Adaptive designs for quantum tomographic experiments.
- Learning from preferences made by many users.
- Elicitation of ratings in collaborative filtering systems in the cold-start setting.
- Multidimensional computer adaptive testing with psychometric questionnaires.

With advances in probabilistic matrix modelling we have addressed the following tasks.

- Learning efficiently with large binary matrices, such as market basket data, click-through data and networks.
- Modelling multi-user preference data, including side information where available.
- Robust modelling of rating data in collaborative filtering systems.

- Analysis of psychometric questionnaires.

The goal of machine learning research is not just to provide solutions to specific problems such as the above, but to develop tools for practitioners to use to solve new tasks. In this thesis the following general-purpose methods have been developed.

Bayesian Active Learning by Disagreement (Chapter 2) is a framework for information theoretic active learning with probabilistic models. In many cases, including those presented in this thesis, one can derive efficient algorithms with this framework and avoid difficult computations that are often required by Bayesian active learning methods. In practice, when implementing BALD, computing the second term in Equation (2.10) or (2.15) efficiently usually requires the most thought.

Our stochastic variational inference (SVI) algorithm (Chapter 5) has general applicability to data that can be represented as a binary matrix. For example, a binary matrix that represents the *location* of observed elements in any sparse matrix. Modelling this fully observed binary matrix provides the first step towards modelling the data generation mechanism for data missing not at random. The sampling strategies given in Chapter 5 could be generalized to other data-types and the minibatch size selection strategy could be applied to any SVI algorithm.

We have proposed two new general-purpose probabilistic matrix models. These include the collaborative model for preference data (CP/CPU, Chapter 6) and the heteroscedastic model for rating data (HOMF, Chapter 7). These models may be applied directly to any preference or ordinal matrix, respectively. Both models scale to datasets with a few thousand rows or columns. Going significantly beyond this number would require parallelization or extending our online routine to these models.

9.2 Future Work

Direct extensions to the methods presented in this thesis are given at the end of the chapters. Here we outline three broader research topics that could build upon this work.

9.2.1 Heteroscedastic Unsupervised Learning

Many of our methods perform well because they can distinguish the two sources of uncertainty outlined in the introduction: parameter uncertainty and observation noise. BALD outperforms the ubiquitous uncertainty sampling because uncertainty sampling rolls both types of uncertainty into one quantity, but BALD distinguishes the two. HOMF, Chapter 7, is robust because it learns variable levels of observation noise across the rows and columns of a matrix.

Most models assume constant observation noise, but in the statistics literature, and more recently in machine learning, heteroscedastic models have been developed. These heteroscedastic models are mostly supervised. They usually are regression models where the noise level is a function of the input variable $\mathbf{x}$ [Harvey, 1976; Kersting *et al.*, 2007]. In unsupervised learning, where $\mathbf{x}$ is a latent variable to be inferred, the noise levels may also vary with $\mathbf{x}$. For example, in psychometrics a subject's response noise level is likely to depend on their personality (one might expected noise to correlate negatively with the Big-Five trait 'conscientiousness').

Heteroscedastic unsupervised models have not been widely explored. Continuing the psychometric example, a heteroscedastic factor analysis model could use an extended version of the likelihood function in Equation (8.1),

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{A}) = \prod_{u,i\in\mathcal{D}} \mathcal{N}(y_{u,i}; \mathbf{x}_u^\top \mathbf{a}_i, v_i(\mathbf{x}))$$

where $v_i(\cdot)$ is a non-negative item specific function that maps the trait to the noise level. This model implies more structured noise than HOMF that assumes that v_i are different constants for each user, independent of $\mathbf{x}$. Although writing down the model is straightforward, deriving an efficient inference algorithm to learn the latent traits $\mathbf{X}$, the factor loadings $\mathbf{A}$, and the parameters of each function $v_i(\mathbf{x})$ jointly may be challenging.

9.2.2 Optimizing Utility over a Horizon

For all of the active learning problems addressed in this thesis we have proposed greedy algorithms. As discussed in Section 2.3.6, this approximation yields little loss when the utility function is submodular. This is often the case in active learning, but not always, such as when learning GP hyperparameters actively but not the latent function (Section 3.3). Furthermore, in many scenarios there may be an immediate reward associated with the value of each obtained measurement. Often, the ultimate objective is to maximize the accumulated reward over a set of active samples. For example, in cold-start recommendation, one is ultimately rewarded when a user purchases a product. The maximally informative product may not be one that the user likes. Nonetheless, learning a good model of the user is useful for identifying high-reward items. Thus, one must balance immediate reward with attaining information about future rewards; this balance is known as the exploration/exploitation trade-off. When optimizing over a horizon it is essential to avoid the intractable evaluation of exponentially many possible outcomes. Techniques developed in reinforcement learning for maximization of accumulated utility over a horizon, such as Markov decision processes (MDPs), address this issue [Puterman, 2009]. Bandit theory also provides a framework to balance exploration and exploitation [Gittins, 1979]. A line of future work could combine Bayesian active learning for identifying informative data with a task specific reward function within an MDP or bandit framework for maximizing reward over a horizon.

9.2.3 Meta-Learning

Approximate inference algorithms will always fail to capture some characteristics of the true posterior distribution. These methods usually have hyperparameters or subroutines that control their behaviour. For example, the location of the pseudo-inputs in the FITC approximation to the GPs in our preference learning model (Chapter 6), or our sampling strategies for SVI with matrices (Chapter 5). It is often hard to directly optimize the overall performance metric with respect to these hyperparameters. Returning to the previous examples, it would be expensive to optimize the marginal likelihood of our preference model with respect to the location of the pseudo-inputs. It would also be hard to compute *a priori* the sampling strategy that would yield fastest convergence of SVI. However, active learning techniques could be used as a tractable alternative objective. For example, one could select the FITC pseudo-inputs to be maximally informative about the GP, or adjust the sampling strategy online to favour entries that will be informative about the direction of the true natural gradient.

Meta-learning, using a simpler model to optimize a more complex procedure, is a new area of machine learning research. Interesting examples include learning the Hessian for quasi-Newton optimization [Hennig & Kiefel, 2013], learning the number of datapoints to use when evaluating acceptance in a Metropolis-Hastings sampler [Korattikara *et al.*, 2014], and modelling an algorithm's generalization performance for hyperparameter optimization [Snoek *et al.*, 2012]. Our minibatch selection strategy, Section 5.4.6, is a meta-learning algorithm. With complex inference algorithms, such the expectation propagation and stochastic variational routines presented in this thesis, meta-learning of optimal strategies, such as which factors to refine, or the order in which to process the data could produce substantially more efficient learning algorithms.

References

Aad, G., Abbott, B., Abdallah, J., Abdelalim, A., Abdesselam, A., Abdinov, O., Abi, B., Abolins, M., Abramowicz, H. & Abreu, H. (2012). Performance of the atlas trigger system in 2010. *The European Physical Journal C*, **72**, 1–61. 9

Abbasnejad, M.E., Bonilla, E.V. & Sanner, S. (2013). Decision-theoretic sparsification for Gaussian process preference learning. In *Machine Learning and Knowledge Discovery in Databases*, 515–530, Springer. 10, 122

Adamson, R.B.A. & Steinberg, A.M. (2010). Improving quantum state estimation with mutually unbiased bases. *Physical Review Letters*, **105**, 030406. 62, 68, 69

Ahn, H.J. (2008). A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, **178**, 37–51. 124

Airoldi, E.M., Blei, D.M., Fienberg, S.E. & Xing, E.P. (2008). Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, **9**, 3. 77

Amari, S.I. (1998). Natural gradient works efficiently in learning. *Neural Computation*, **10**, 251–276. 84

Amselem, E. & Bourennane, M. (2009). Experimental four-qubit bound entanglement. *Nature Physics*, **5**, 748 – 752. 61

Andrich, D. (1978). A rating formulation for ordered response categories. *Psychometrika*, **43**, 561–573. 150

Antoniuk, K., Franc, V. & Hlaváč, V. (2013). Mord: Multi-class classifier for ordinal regression. In *Machine Learning and Knowledge Discovery in Databases*, 96–111, Springer. 152

Atkinson, A.C. (1988). Recent developments in the methods of optimum and related experimental designs. *International Statistical Review/Revue Internationale de Statistique*, 99–115. 24

Atteia, O., Dubois, J.P. & Webster, R. (1994). Geostatistical analysis of soil contamination in the Swiss Jura. *Environmental Pollution*, **86**, 315 – 327. 116

Attias, H. (1999). Inferring parameters and structure of latent variable models by variational Bayes. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 21–30. 108

Audenaert, K.M. & Scheel, S. (2009). Statistical inference from imperfect photon detection. *New Journal of Physics*, **11**, 113052. 72

Baker, F.B. (2001). *The basics of item response theory*. Education Resources Information Center. 149

Baum, E. & Lang, K. (1992). Query learning can work poorly when a human oracle is used. In *International Joint Conference on Neural Networks*. 9

Beal, M.J. & Ghahramani, Z. (2006). Variational Bayesian learning of directed graphical models with hidden variables. *Bayesian Analysis*, **1**, 793–831. 152

Bell, R.M., Koren, Y. & Volinsky, C. (2010). All together now: A perspective on the netflix prize. *Chance*, **23**, 24–29. 77

Bellman, R.E. (1961). *Adaptive control processes - A guided tour*. Princeton University Press, Princeton, New Jersey, U.S.A. 61

Bengtsson, I. & Zyczkowski, K. (2006). *Geometry of quantum states: an introduction to quantum entanglement*. Cambridge University Press. 61

Bennett, J. & Lanning, S. (2007). The netflix prize. In *Proceedings of KDD cup and workshop*, vol. 2007, 35. 5, 76

Birlutiu, A., Groot, P. & Heskes, T. (2010). Multi-task preference learning with an application to hearing aid personalization. *Neurocomputing*, **73**, 1177 – 1185. 102, 103, 114, 115, 117

Birnbaum, A. (1968). Some latent trait models and their use in inferring an examinees ability. *Statistical Theories of Mental Test Scores*, 395–479. 151

Bishop, C.M. (2006). *Pattern recognition and machine learning*, vol. 1. Springer New York. 34, 111

Blume-Kohout, R. (2010). Optimal, reliable estimation of quantum states. *New Journal of Physics*, **12**, 043034. 62, 63

Blume-Kohout, R. & Hayden, P. (2006). Accurate quantum state estimation via "keeping the experimentalist honest". *arXiv preprint quant-ph/0603116*. 63

Bogdanov, Y.I., Brida, G., Bukeev, I.D., Genovese, M., Kravtsov, K.S., Kulik, S.P., Moreva, E.V., Soloviev, A.A. & Shurupov, A.P. (2011). Statistical estimation of the quality of quantum-tomography protocols. *Physical Review A*, **84**, 042108. 62

Bonilla, E.V., Guo, S. & Sanner, S. (2010). Gaussian process preference elicitation. In *Advances in Neural Information Processing Systems 23*, 262–270. 103, 105, 114, 115, 116, 117, 122

Borsboom, D., Mellenbergh, G.J. & Van Heerden, J. (2003). The theoretical status of latent variables. *Psychological review*, **110**, 203. 171

Boutilier, C., Zemel, R.S. & Marlin, B. (2002). Active collaborative filtering. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, 98–106, Morgan Kaufmann Publishers Inc. 4, 134

Box, G.E. (1976). Science and statistics. *Journal of the American Statistical Association*, **71**, 791–799. 1

Box, G.E. (1980). Sampling and Bayes' inference in scientific modelling and robustness. *Journal of the Royal Statistical Society. Series A (General)*, 383–430. 154

Braunstein, S.L. & Caves, C.M. (1994). Statistical distance and the geometry of quantum states. *Physical Review Letters*, **72**, 3439–3443. 61

Brier, G.W. (1950). Verification of forecasts expressed in terms of probability. *Monthly weather review*, **78**, 1–3. 25

Brijs, T., Swinnen, G., Vanhoof, K. & Wets, G. (1999). Using association rules for product assortment decisions: a case study. In *KDD*, 254–260. 93

Brin, S., Motwani, R., Ullman, J.D. & Tsur, S. (1997). Dynamic itemset counting and implication rules for market basket data. In *ACM SIGMOD Record*, vol. 26, 255–264, ACM. 5

Brochu, E., de Freitas, N. & Ghosh, A. (2007). Active preference learning with discrete choice data. *Advances in Neural Information Processing Systems 20*, **20**, 409–416. 102

Broderick, T., Boyd, N., Wibisono, A., Wilson, A.C. & Jordan, M. (2013). Streaming variational Bayes. In *Advances in Neural Information Processing Systems*, 1727–1735. 101

Brozovsky, L. & Petricek, V. (2007). Recommender system for online dating service. In *Proceedings of Conference Znalosti 2007*, VSB, Ostrava. 135

Bryant, M. & Sudderth, E. (2012). Truly nonparametric online variational inference for hierarchical Dirichlet processes. In *Advances in Neural Information Processing Systems*, 2708–2716. 90

Burbidge, R., Rowland, J.J. & King, R.D. (2007). Active learning for regression based on query by committee. In *Intelligent Data Engineering and Automated Learning-IDEAL 2007*, 209–218, Springer. 31

Burges, C.J. (1998). A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, **2**, 121–167. 30

Cai, L. (2010). High-dimensional exploratory item factor analysis byametropolishastings robbinsmonro algorithm. *Psychometrika*, **75**, 33–57. 152

Campbell, C., Cristianini, N. & Smola, A. (2000). Query learning with large margin classifiers. In *International Conference on Machine Learning*, 111–118. 31

Caselton, W.F. & Zidek, J.V. (1984). Optimal monitoring network designs. *Statistics & Probability Letters*, **2**, 223–227. 15, 27

Chalmers, R.P. (2012). Mirt: a multidimensional item response theory package for the r environment. *Journal of Statistical Software*, **48**, 1–29. 151, 152, 166

Chaloner, K. & Verdinelli, I. (1995). Bayesian experimental design: A review. *Statistical Science*, 273–304. 24, 25

Chu, W. & Ghahramani, Z. (2005a). Gaussian processes for ordinal regression. In *Journal of Machine Learning Research*, 1019–1041. 126

Chu, W. & Ghahramani, Z. (2005b). Preference learning with Gaussian processes. In *22nd International Conference on Machine learning*, 137–144. 33, 102, 103, 105

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. & Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. In *SIGIR workshop on recommender systems*, vol. 60, Citeseer. 124

Cohn, D.A., Ghahramani, Z. & Jordan, M.I. (1996). Active learning with statistical models. *Journal of Artificial Intellegence Research*, **4**, 129–145. 9, 20

Costa, P.T. & McCrae, R.R. (1992a). *Neo PI-R professional manual*. Odessa, FL: Psychological assessment resources. 148

Costa, P.T. & McCrae, R.R. (1992b). Normal personality assessment in clinical practice: the neo personality inventory. *Psychological assessment*, **4**, 5. 148

Cover, T.M., Thomas, J.A. & Kieffer, J. (1994). Elements of information theory. *SIAM Review*, **36**, 509–510. 11, 21

Cox, R.T. (1946). Probability, frequency and reasonable expectation. *American journal of physics*, **14**, 1. 2

Cramer, A.O., Sluis, S., Noordhof, A., Wichers, M., Geschwind, N., Aggen, S.H., Kendler, K.S. & Borsboom, D. (2012). Dimensions of normal personality as networks in search of equilibrium: You can't like parties if you don't like people. *European Journal of Personality*, **26**, 414–431. 171

Dasgupta, S. (2005). Analysis of a greedy active learning strategy. vol. 17, 337–344. 21

Dawid, A. (2007). The geometry of proper scoring rules. *Annals of the Institute of Statistical Mathematics*, **59**, 77–93. 25, 43, 63, 153

de Burgh, M.D., Langford, N.K., Doherty, A.C. & Gilchrist, A. (2008). Choice of measurement sets in qubit tomography. *Physical Review A*, **78**, 052122. 62

De Gemmis, M., Iaquinta, L., Lops, P., Musto, C., Narducci, F. & Semeraro, G. (2009). Preference learning in recommender systems. In *ECML/PKDD-09 Workshop on Preference Learning*. 102

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Le, Q.V., Mao, M.Z., Ranzato, M., Senior, A.W. & Tucker, P.A. (2012). Large scale distributed deep networks. In *Advances in Neural Information Processing Systems*, 1232–1240. 100

Dempster, A.P., Laird, N.M. & Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, **39**, 1–38. 114, 151

Devarajan, K. (2008). Nonnegative matrix factorization: an analytical and interpretive tool in computational biology. *PLoS computational biology*, **4**, e1000029. 5

Digman, J.M. (1990). Personality structure: Emergence of the five-factor model. *Annual review of psychology*, **41**, 417–440. 148

Dooms, S., De Pessemier, T. & Martens, L. (2013). Movietweetings: a movie rating dataset collected from twitter. In *Workshop on Crowdsourcing and Human Computation for Recommender Systems, CrowdRec at RecSys 2013*. 135

Doucet, A., de Freitas, N. & Gordon, N. (2001). *Sequential Monte Carlo in Practice*. Springer-Verlag. 64

Dror, G., Koenigstein, N., Koren, Y. & Weimer, M. (2012). The Yahoo! music dataset and KDD-Cup'11. *Journal of Machine Learning Research-Proceedings Track*, **18**, 8–18. 93

Duvenaud, D., Nickisch, H. & Rasmussen, C.E. (2012). Additive Gaussian processes. In *Advances in Neural Information Processing Systems*, 226–234, Granada, Spain. 45

Ertin, E., Fisher, J.W. & Potter, L.C. (2003). Maximum mutual information principle for dynamic sensor query problems. In *Information Processing in Sensor Networks*, 405–416, Springer. 15, 27

Fedorov, V.V. (1972). *Theory of optimal experiments*. Elsevier. 23, 26

Fischer, D.G., Kienle, S.H. & Freyberger, M. (2000). Quantum-state estimation by self-learning measurements. *Physical Review A*, **61**, 032306. 62, 65

Friedman, N. & Koller, D. (2003). Being Bayesian about network structure. a Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, **50**, 95–125. 172

Fuhrmann, D.R. (2003). *Active Testing Surveillance Systems, or, Playing Twenty Questions with a Radar*. Defense Technical Information Center. 27

Fürnkranz, J. & Hüllermeier, E. (2003). Pairwise preference learning and ranking. In *European Conference on Machine Learning*, 145–156, Springer. 5

Fürnkranz, J. & Hüllermeier, E. (2010). *Preference learning*. Springer-Verlag New York Inc. 102, 105

Fusi, N., Stegle, O. & Lawrence, N.D. (2012). Joint modelling of confounding factors and prominent genetic regulators provides increased accuracy in genetical genomics studies. *PLoS computational biology*, **8**. 171

Garnett, R., Osborne, M.A., Reece, S., Rogers, A. & Roberts, S.J. (2010). Sequential Bayesian prediction in the presence of changepoints and faults. *The Computer Journal*, **53**, 1430–1446. 33

Garnett, R., Osborne, M.A. & Hennig, P. (2013). Active learning of linear embeddings for Gaussian processes. *arXiv preprint arXiv:1310.6740*. 34, 45, 47

Gelman, A., Meng, X.L. & Stern, H. (1996). Posterior predictive assessment of model fitness via realized discrepancies. *Statistica Sinica*, **6**, 733–760. 1, 154

Gelman, A., Carlin, J.B., Stern, H.S. & Rubin, D.B. (2003). *Bayesian Data Analysis*. Chapman and Hall/CRC. 2

Gershon, R.C. (2005). Computer adaptive testing. *Journal of Applied Measurement*. 168

Ghahramani, Z. & Beal, M.J. (2000). Graphical models and variational methods. *Advanced Mean Field Method — Theory and Practice*, 37–50. 82, 108

Ghahramani, Z. & Rasmussen, C.E. (2002). Bayesian Monte Carlo. In *Advances in Neural Information Processing Systems*, 489–496. 33

Girard, A., Rasmussen, C.E., Candela, J.Q. & Murray-Smith, R. (2003). Gaussian process priors with uncertain inputs-application to multiple-step ahead time series forecasting. *Advances in Neural Information Processing Systems*, 545–552. 122

Gittins, J.C. (1979). Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, 148–177. 176

Goldberg, L.R. (1990). An alternative "description of personality": the big-five factor structure. *Journal of Personality and Social Psychology*, **59**, 1216. 148

Goldberg, L.R. (1999). A broad-bandwidth, public domain, personality inventory measuring the lower-level facets of several five-factor models. *Personality psychology in Europe*, **7**, 7–28. 5

Goldberg, L.R., Johnson, J.A., Eber, H.W., Hogan, R., Ashton, M.C., Cloninger, C.R. & Gough, H.G. (2006). The international personality item pool and the future of public-domain personality measures. *Journal of Research in Personality*, **40**, 84–96. 135, 148

Gopalan, P., Mimno, D., Gerrish, S., Freedman, M. & Blei, D. (2012). Scalable inference of overlapping communities. In *Advances in Neural Information Processing Systems*, 2258–2266. 92

Gunawardana, A. & Shani, G. (2009). A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research*, **10**, 2935–2962. 94

Hamaker, E.L., Dolan, C.V. & Molenaar, P.C. (2005). Statistical modeling of the individual: Rationale and application of multivariate stationary time series analysis. *Multivariate behavioral research*, **40**, 207–233. 171

Hamaker, E.L., Nesselroade, J.R. & Molenaar, P. (2007). The integrated trait–state model. *Journal of Research in Personality*, **41**, 295–315. 171

Hannemann, T., Reiss, D., Balzer, C., Neuhauser, W., Toschek, P. & Wunderlich, C. (2002). Self-learning estimation of quantum states. *Physical Review A*, **65**, 050303. 62, 65

Harpale, A.S. & Yang, Y. (2008). Personalized active learning for collaborative filtering. In *Special Interest Group on Information Retrieval*, 91–98, ACM. 134, 145

Harvey, A.C. (1976). Estimating regression models with multiplicative heteroscedasticity. *Econometrica: Journal of the Econometric Society*, 461–465. 175

Hawkins, D.M. (2004). The problem of overfitting. *Journal of Chemical Information and Computer Sciences*, **44**, 1–12. 62

Heckerman, D., Breese, J. & Rommelse, K. (1994). Troubleshooting under uncertainty. *Communications of the ACM*, 121–130. 21

Hennig, P. (2011). *Approximate inference in graphical models*. Ph.D. thesis, University of Cambridge. 150

Hennig, P. & Kiefel, M. (2013). Quasi-newton methods: A new direction. *Journal of Machine Learning Research*, **14**, 843–865. 176

Hennig, P. & Schuler, C.J. (2012). Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, **13**, 1809–1837. 57

Herbrich, R., Lawrence, N.D. & Seeger, M. (2002). Fast sparse Gaussian process methods: The informative vector machine. In *Advances in Neural Information Processing Systems*, 609–616. 16, 28

Hernández-Lobato, D. (2007). Approximating Gaussian integrals by replacing a Student distribution by a Gaussian distribution. 130

Hernández-Lobato, J.M. (2010). *Balancing Flexibility and Robustness in Machine Learning: Semi-parametric Methods and Sparse Linear Models*. Ph.D. thesis, Universidad Autónoma de Madrid. 112

Hernández-Lobato, J.M., Houlsby, N.M.T. & Ghahramani, Z. (2014a). Probabilistic matrix factorization with non-random missing data. In *31st International Conference on Machine Learning*, 379–387. iv, 146

Hernández-Lobato, J.M., Houlsby, N.M.T. & Ghahramani, Z. (2014b). Stochastic inference for scalable probabilistic modeling of binary matrices. In *31st International Conference on Machine Learning*, 1512–1520. iii

Hoffman, M., Blei, D.M. & Bach, F. (2010). Online learning for latent dirichlet allocation. *Advances in Neural Information Processing Systems*, **23**, 856–864. 90

Hoffman, M.D., Blei, D.M., Wang, C. & Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, **14**, 1303–1347. 78, 82, 83, 90

Hofmann, T. (2003). Collaborative filtering via Gaussian probabilistic latent semantic analysis. In *Special Interest Group on Information Retrieval*, 259–266, ACM, New York, NY, USA. 134

Hofmann, T. (2004). Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, **22**, 89–115. 76

Houlsby, N.M.T. & Blei, D.M. (2014). A filtering approach to stochastic variational inference. In *Advances in Neural Information Processing Systems*. iv

Houlsby, N.M.T. & Ciaramita, M. (2014). A scalable Gibbs sampler for probabilistic entity linking. *European Conference on Information Retrieval*. iv

Houlsby, N.M.T. & Houlsby, G.T. (2013). Statistical fitting of undrained strength data. *Geotechnique*, **63**, 1253–1263. iv

Houlsby, N.M.T., Huszár, F., Ghahramani, Z. & Lengyel, M. (2011). Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*. iii

Houlsby, N.M.T., Hernández-Lobato, J.M., Huszar, F. & Ghahramani, Z. (2012). Collaborative Gaussian processes for preference learning. In *Advances in Neural Information Processing Systems*, 2105–2113. iii

Houlsby, N.M.T., Huszár, F., Ghassemi, M.M., Orbán, G., Wolpert, D.M. & Lengyel, M. (2013). Cognitive tomography reveals complex task-independent mental representations. *Current Biology*, **23**, 2169–2175. iv

Houlsby, N.M.T., Hernández-Lobato, J.M. & Ghahramani, Z. (2014). Cold-start active learning with robust ordinal matrix factorization. In *31st International Conference on Machine Learning*, 766–774. iii, 129

Hu, D., Van Der Maaten, L., Cho, Y., Saul, L.K. & Lerner, S. (2010). Latent variable models for predicting file dependencies in large-scale software development. In *Advances in Neural Information Processing Systems*, 865–873. 77

Huszár, F. (2013). *Scoring Rules, Divergences and Information in Bayesian Machine Learning*. Ph.D. thesis, University of Cambridge. iii, 25

Huszár, F. & Houlsby, N.M.T. (2012). Adaptive Bayesian quantum tomography. *Physical Review A*, **85**, 052120. iii

Řeháček, J., Englert, B.G. & Kaszlikowski, D. (2004). Minimal qubit tomography. *Physical Review A*, **70**, 052321. 62

Ito, K. & Xiong, K. (2000). Gaussian filters for nonlinear filtering problems. *Automatic Control, IEEE Transactions on*, **45**, 910–927. 35

Iwata, T., Houlsby, N.M.T. & Ghahramani, Z. (2013). Active learning for interactive visualization. In *16th International Conference on Aritificial Intelligence and Statistics*. iv, 5

Jaakkola, T. & Jordan, M. (1997). A variational approach to Bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*. 82

Jaynes, E.T. (2003). *Probability theory: the logic of science*. Cambridge University Press. 3

Jin, R. & Si, L. (2004). A Bayesian approach toward active learning for collaborative filtering. In *Uncertainty in Artificial Intelligence*, 278–285, AUAI Press. 9, 134

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Knowledge Discovery and Data Mining*, 133–142, ACM. 77, 102, 122

Jordan, M.I., Ghahramani, Z., Jaakkola, T.S. & Saul, L.K. (1998). An introduction to variational methods for graphical models. In *Learning in Graphical Models*, vol. 89, 105–161, Springer Netherlands. 81

Julier, S.J. & Uhlmann, J.K. (1997). New extension of the Kalman filter to nonlinear systems. In *AeroSense'97*, 182–193, International Society for Optics and Photonics. 133

Kamishima, T., Kazawa, H. & Akaho, S. (2005). Supervised ordering - an empirical survey. In *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005)*, 673–676. 117

Kapoor, A., Horvitz, E. & Basu, S. (2007). Selective supervision: Guiding supervised learning with decision-theoretic active learning. In *International Joint Conference on Artificial Intellegence*, vol. 7, 877–882. 11, 29, 44

Kass, R.E. & Raftery, A.E. (1995). Bayes factors. *Journal of the American Statistical Association*, **90**, 773–795. 26, 35

Kersting, K., Plagemann, C., Pfaff, P. & Burgard, W. (2007). Most likely heteroscedastic Gaussian process regression. In *24th International Conference on Machine learning*, 393–400, ACM. 175

Ko, C.W., Lee, J. & Queyranne, M. (1995). An exact algorithm for maximum entropy sampling. *Operations Research*, **43**, 684–691. 13, 21

Kohavi, R., Brodley, C.E., Frasca, B., Mason, L. & Zheng, Z. (2000). KDD-Cup 2000 organizers' report: peeling the onion. *SIGKDD Explorations Newsletter*, **2**, 86–93. 93

Korattikara, A., Chen, Y. & Welling, M. (2014). Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In *Proceedings of the 31st International Conference on Machine learning*, 181–189. 176

Koren, Y., Bell, R. & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, **42**, 30–37. 76

Kosinski, M., Stillwell, D. & Graepel, T. (2013). Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, **110**, 5802–5805. 135, 149, 171

Kosut, R., Walmsley, I.A. & Rabitz, H. (2004). Optimal experiment design for quantum state and process tomography and Hamiltonian parameter estimation. *arXiv preprint quant-ph/0411093*. 62

Krause, A. & Guestrin, C. (2005). Near-optimal value of information in graphical models. In *Uncertainty in Artificial Intelligence*. 13, 21, 22

Krause, A., Singh, A. & Guestrin, C. (2008). Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, **9**, 235–284. 15, 20, 21, 27

Kravtsov, K., Straupe, S., Radchenko, I., Houlsby, N., Huszár, F. & Kulik, S. (2013). Experimental adaptive Bayesian tomography. *Physical Review A*, **87**, 062122. iii

Krishnapuram, B., Williams, D., Xue, Y., Carin, L., Figueiredo, M. & Hartemink, A.J. (2004). On semi-supervised classification. In *Advances in Neural Information Processing Systems*, 721–728. 16

Kuss, M. & Rasmussen, C.E. (2005). Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research*, **6**, 1679–1704. 41

Lacoste-Julien, S., Huszár, F. & Ghahramani, Z. (2011). Approximate inference for the loss-calibrated Bayesian. *Artificial Intelligence and Statistics*, **15**, 416–424. 10

LADD, T.D., JELEZKO, F., LAFLAMME, R., NAKAMURA, Y., MONROE, C. & OBRIEN, J.L. (2010). Quantum computers. *Nature*, **464**, 45–53. 58

LAKSHMINARAYANAN, B., BOUCHARD, G. & ARCHAMBEAU, C. (2011). Robust Bayesian matrix factorisation. In *International Conference on Artificial Intelligence and Statistics*, 425–433. 78, 88, 134, 136

LAWRENCE, N. (2004). Gaussian process latent variable models for visualisation of high dimensional data. vol. 16, 329–336, of. 33

LÁZARO GREDILLA, M. (2010). *Sparse Gaussian processes for large-scale machine learning*. Ph.D. thesis, Universidad Carlos III de Madrid. 113

LE, Q.T. & TU, M.P. (2010). Active learning for co-clustering based collaborative filtering. In *Computing and Communication Technologies, Research, Innovation, and Vision for the Future*, 1–4, IEEE. 134

LEWI, J., BUTERA, R.J. & PANINSKI, L. (2007). Efficient active learning with generalized linear models. In *International Conference on Artificial Intelligence and Statistics*, 267–274. 16, 26

LEWI, J., BUTERA, R. & PANINSKI, L. (2009). Sequential optimal design of neurophysiology experiments. *Neural Computation*, **21**, 619–687. 25

LIM, Y.J. & TEH, Y.W. (2007). Variational Bayesian approach to movie rating prediction. In *Proceedings of KDD Cup and Workshop*, 15–21. 77, 92

LINDLEY, D.V. (1956). On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 986–1005. 14, 15, 26

LIU, Y. (2004). Active learning with support vector machine applied to gene expression data for cancer classification. *Journal of Chemical Information and Computer Sciences*, **44**, 1936–1941. 9

LVOVSKY, A., HANSEN, H., AICHELE, T., BENSON, O., MLYNEK, J. & SCHILLER, S. (2001). Quantum state reconstruction of the single-photon fock state. *Physical review letters*, **87**, 50402. 72

MACKAY, D.J.C. (1992a). The evidence framework applied to classification networks. *Neural Computation*, **4**, 720–736. 82

MacKay, D.J.C. (1992b). Information-based objective functions for active data selection. *Neural Computation*, **4**, 590–604. 5, 14, 16, 20, 26, 49

MacKay, D.J.C. (2001). Local minima, symmetry-breaking, and model pruning in variational free energy minimization, http://www.inference.phy.cam.ac.uk/mackay/minima.pdf. 119, 153

Mahler, D.H., Rozema, L.A., Darabi, A., Ferrie, C., Blume-Kohout, R. & Steinberg, A.M. (2013). Adaptive quantum state tomography improves accuracy quadratically. *Physical Review Letters*, **111**, 183601. 66

Makransky, G., Mortensen, E.L. & Glas, C.A. (2013). Improving personality facet scores with multidimensional computer adaptive testing an illustration with the neo pi-r. *Assessment*, **20**, 3–13. 147, 168

Maltz, D. & Ehrlich, K. (1995). Pointing the way: active collaborative filtering. In *SIGCHI*, 202–209, ACM Press/Addison-Wesley Publishing Co. 124

Marlin, B.M. & Zemel, R.S. (2007). Collaborative filtering and the missing at random assumption. In *Uncertainty in Artificial Intelligence*. 134, 136, 145

Marlin, B.M. & Zemel, R.S. (2009). Collaborative prediction and ranking with non-random missing data. In *Third ACM conference on Recommender systems*, 5–12, ACM. 101

Masters, G.N. (1982). A rasch model for partial credit scoring. *Psychometrika*, **47**, 149–174. 150

McCallum, A. & Nigam, K. (1998). Employing EM and pool-based active learning for text classification. vol. 98, 350–358. 31

McCrae, R.R. & Costa, P.T. (1987). Validation of the five-factor model of personality across instruments and observers. *Journal of Personality and Social Psychology*, **52**, 81. 148

Melville, P., Yang, S.M., Saar-Tsechansky, M. & Mooney, R. (2005). Active learning for probability estimation using jensen-shannon divergence. In *European Conference on Machine Learning*, 268–279, Springer. 31

Mika, S., Schölkopf, B., Smola, A.J., Müller, K.R., Scholz, M. & Rätsch, G. (1998). Kernel PCA and de-noising in feature spaces. In *Advances in Neural Information Processing Systems*, vol. 11, 536–542. 171

MILD, A. & REUTTERER, T. (2003). An improved collaborative filtering approach for predicting cross-category purchases based on binary market basket data. *Journal of Retailing and Consumer Services*, **10**, 123–133. 77

MINKA, T. (2001a). *A family of algorithms for approximate Bayesian inference*. Ph.D. thesis, MIT. 35, 109

MINKA, T. & LAFFERTY, J. (2002). Expectation-propagation for the generative aspect model. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, 352–359. 112

MINKA, T.P. (2001b). Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, 362–369, Morgan Kaufmann Publishers Inc. 29, 108

MURAKI, E. (1992). A generalized partial credit model: Application of an em algorithm. *Applied Psychological Measurement*, **16**, 159–176. 150

MYUNG, J.I. & PITT, M.A. (2009). Optimal experimental design for model discrimination. *Psychological review*, **116**, 499. 4

NAISH-GUZMAN, A. & HOLDEN, S.B. (2007). The generalized FITC approximation. In *Advances in Neural Information Processing Systems*. 35, 113

NAKAJIMA, S., SUGIYAMA, M. & TOMIOKA, R. (2010). Global analytic solution for variational Bayesian matrix factorization. *Advances in Neural Information Processing Systems*, **23**, 1759–1767. 77, 78, 79, 92, 93, 95

NEMHAUSER, G.L., WOLSEY, L.A. & FISHER, M.L. (1978). An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, **14**, 265–294. 21

NERI, P. & LEVI, D.M. (2006). Receptive versus perceptive fields from the reverse-correlation viewpoint. *Vision research*, **46**, 2465–2474. 165

NG, A.Y. & JORDAN, A. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. *Advances in Neural Information Processing Systems*, **14**, 841. 8

NICKISCH, H. & RASMUSSEN, C.E. (2008). Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, **9**, 2035–2078. 109, 116

Nunn, J., Smith, B.J., Puentes, G., Walmsley, I.A. & Lundeen, J.S. (2010). Optimal experiment design for quantum state tomography: Fair, precise, and minimal tomography. *Physical Review A*, **81**, 042109. 62

Orr, G.B. & Müller, K.R., eds. (1998). *Neural Networks: Tricks of the Trade*, Springer-Verlag. 79

Osborne, M., Duvenaud, D., Garnett, R., Rasmussen, C., Roberts, S. & Ghahramani, Z. (2012). Active learning of model evidence using Bayesian quadrature. In *Advances in Neural Information Processing Systems*, 46–54. 57

Osborne, M.A., Garnett, R. & Roberts, S.J. (2009). Gaussian processes for global optimization. In *International Conference on Learning and Intelligent Optimization*, 1–15. 33

Osborne, M.A., Garnett, R. & Roberts, S.J. (2010). Active data selection for sensor networks with faults and changepoints. In *Advanced Information Networking and Applications, 2010 24th IEEE International Conference on*, 533–540, IEEE. 4

Paninski, L. (2003). Estimation of entropy and mutual information. *Neural Computation*, **15**, 1191–1253. 15

Panzeri, S., Senatore, R., Montemurro, M.A. & Petersen, R.S. (2007). Correcting for the sampling bias problem in spike train information measures. *Journal of Neurophysiology*, **98**, 1064–1072. 16

Paquet, U. & Koenigstein, N. (2013). One-class collaborative filtering with random graphs. *World Wide Web*, 999–1008. 79, 92, 96

Paquet, U., Thomson, B. & Winther, O. (2012). A hierarchical model for ordinal matrix factorization. *Statistics and Computing*, **22**, 945–957. 128, 133, 136

Paris, M. & Řeháček, J. (2004). *Quantum state estimation*, vol. 649. Springer. 58, 61

Park, S.T. & Chu, W. (2009). Pairwise preference regression for cold-start recommendation. In *RecSys*, 21–28, ACM. 124

Park, S.T., Pennock, D., Madani, O., Good, N. & DeCoste, D. (2006). Naïve filterbots for robust cold-start recommendations. In *Knowledge Discovery and Data Mining*, 699–705, ACM. 124

Patra, M.K. (2007). Quantum state determination: estimates for information gain and some exact calculations. *Journal of Physics A: Mathematical and Theoretical*, **40**, 10887. 62, 65, 75

Pearl, J. (2000). *Causality: models, reasoning and inference*, vol. 29. Cambridge Univ Press. 172

Petz, D. (2008). *Quantum information theory and quantum statistics*. Springer. 59

Puterman, M.L. (2009). *Markov decision processes: discrete stochastic dynamic programming*, vol. 414. John Wiley & Sons. 176

Quiñonero-Candela, J. & Rasmussen, C.E. (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, **6**, 1939–1959. 28

Raiko, T., Ilin, A. & Juha, K. (2007). Principal component analysis for large scale problems with lots of missing values. In *European Conference on Machine Learning*, vol. 4701, 691–698, Springer Berlin / Heidelberg. 77, 79, 83, 92, 95, 112

Ranganath, R., Wang, C., Blei, D.M. & Xing, E.P. (2013). An adaptive learning rate for stochastic variational inference. In *International Conference on Machine Learning*, 298–306. 92

Rashid, A.M., Albert, I., Cosley, D., Lam, S.K., McNee, S.M., Konstan, J.A. & Riedl, J. (2002). Getting to know you: learning new user preferences in recommender systems. In *International Conference on Intelligent User Interfaces*, 127–134, ACM. 134, 145

Rashid, A.M., Karypis, G. & Riedl, J. (2008). Learning preferences of new users in recommender systems: an information theoretic approach. *ACM SIGKDD Explorations Newsletter*, **10**, 90–100. 134, 145

Rasmussen, C.E. & Williams, C.K.I. (2005). *Gaussian Processes for Machine Learning*. The MIT Press. 5, 33, 34, 112, 130

Raynal, P., Lü, X. & Englert, B.G. (2011). Mutually unbiased bases in six dimensions: The four most distant bases. *Physical Review A*, **83**, 062303. 62, 75

Reckase, M. (2009). *Multidimensional Item Response Theory*. Springer. 150

Rendle, S., Freudenthaler, C., Gantner, Z. & Schmidt-Thieme, L. (2009). Bpr: Bayesian personalized ranking from implicit feedback. In *Uncertainty in Artificial Intelligence*, 452–461, AUAI Press. 77, 79, 93, 96, 136

Robbins, H. & Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, **22**, 400–407. 83, 90

Ross, D.A. & Zemel, R.S. (2002). Multiple cause vector quantization. In *Advances in Neural Information Processing Systems*, 1017–1024. 134

Roweis, S. & Ghahramani, Z. (1999). A unifying review of linear gaussian models. *Neural Computation*, **11**, 305–345. 151

Roy, N. & McCallum, A. (2001). Toward optimal active learning through Monte Carlo estimation of error reduction. *International Conference on Machine Learning*. 10, 11, 29, 30

Rubin, D.B. (1984). Bayesianly justifiable and relevant frequency calculations for the applies statistician. *The Annals of Statistics*, **12**, 1151–1172. 154

Salakhutdinov, R. & Mnih, A. (2008). Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, vol. 20. 6, 77

Samejima, F. (1969). Estimation of latent ability using a response pattern of graded scores. *Psychometrika monograph supplement*. 150

Sarwar, B., Karypis, G., Konstan, J. & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *World Wide Web*, 285–295, ACM. 76

Schaul, T., Zhang, S. & LeCun, Y. (2012). No more pesky learning rates. *arXiv preprint arXiv:1206.1106*. 92

Schein, A.I., Popescul, A., Ungar, L.H. & Pennock, D.M. (2002). Methods and metrics for cold-start recommendations. In *Special Interest Group on Information Retrieval*, 253–260, ACM. 124

Schölkopf, B. & Smola, A.J. (2002). *Learning with kernels*. The MIT Press. 30

Seeger, M. & Bouchard, G. (2012). Fast variational Bayesian inference for non-conjugate matrix factorization models. *Journal of Machine Learning Research - Proceedings Track*, **22**, 1012–1018. 79, 93, 95

Seeger, M., Williams, C.K. & Lawrence, N.D. (2003). Fast forward selection to speed up sparse Gaussian process regression. In *Workshop on AI and Statistics*, vol. 9, 2003. 29, 34

Segall, D.O. (2010). Principles of multidimensional adaptive testing. In *Elements of Adaptive Testing*, 57–75, Springer. 168

Seo, S., Wallat, M., Graepel, T. & Obermayer, K. (2000). Gaussian process regression: Active data selection and test point rejection. In *Mustererkennung 2000*, 27–34, Springer. 34

Settles, B. (2012). Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, **6**, 1–114. 23

Seung, H.S., Opper, M. & Sompolinsky, H. (1992). Query by committee. In *Fifth annual workshop on Computational Learning Theory*, 287–294, ACM. 30, 31

Shannon, C.E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, **27**, 379–423, 623–656. 11

Shewry, M.C. & Wynn, H.P. (1987). Maximum entropy sampling. *Journal of Applied Statistics*, **14**, 165–170. 15, 26

Shor, P.W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. In *Foundations of Computer Science*, 124–134, IEEE. 58

Si, L. & Jin, R. (2003). Flexible mixture model for collaborative filtering. In *International Conference on Machine Learning*, vol. 3, 704–711. 134

Snelson, E. & Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 19*. 46, 113, 122

Snoek, J., Larochelle, H. & Adams, R.P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, 2960–2968. 176

Srebro, N., Rennie, J.D. & Jaakkola, T. (2005). Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems*, 1329–1336. 76

Stern, D.H., Herbrich, R. & Graepel, T. (2009). Matchbox: large scale online Bayesian recommendations. In *World Wide Web*, 111–120, ACM. 77, 111, 112, 126, 130

Thompson, B. (2005). Canonical correlation analysis. *Encyclopedia of Statistics in Behavioral Science*. 163

Tong, S. (2001). *Active learning: theory and applications*. Ph.D. thesis, Stanford University. 19

Tong, S. & Chang, E. (2001). Support vector machine active learning for image retrieval. In *Ninth ACM international conference on Multimedia*, 107–118, ACM. 9

Tong, S. & Koller, D. (2002). Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, **2**, 45–66. 9, 30, 31

Tsochantaridis, I., Hofmann, T., Joachims, T. & Altun, Y. (2004). Support vector machine learning for interdependent and structured output spaces. In *International Conference on Machine Learning*, 104, ACM. 16

Tur, G., Hakkani-Tür, D. & Schapire, R.E. (2005). Combining active and semi-supervised learning for spoken language understanding. *Speech Communication*, **45**, 171–186. 9

Turner, R.D. (2011). *Gaussian Processes for State Space Models and Change Point Detection*. Ph.D. thesis, University of Cambridge, Cambridge, UK. 122

Ungar, L.H. & Foster, D.P. (1998). Clustering methods for collaborative filtering. In *AAAI Workshop on Recommendation Systems*, vol. 1. 76

Vapnik, V. (2000). *The Nature of Statistical Learning Theory*. Springer. 19

Wagstaff, K. (2012). Machine learning that matters. In *International Conference on Machine Learning*. 2

Wang, C., Paisley, J. & Blei, D.M. (2011). Online variational inference for the hierarchical Dirichlet process. In *Artificial Intellegence and Statistics*, 752–760. 90

Wang, W.C. & Chen, P.H. (2004). Implementation and measurement efficiency of multidimensional computerized adaptive testing. *Applied Psychological Measurement*, **28**, 295–316. 168

Wootters, W. & Fields, B. (1989). Optimal state-determination by mutually unbiased measurements. *Annals of Physics*, **191**, 363 – 381. 62

Yan, F., Yang, M. & Cao, Z.L. (2010). Optimal reconstruction of the states in qutrit systems. *Physical Review A*, **82**, 044102. 62

Yu, K., Bi, J. & Tresp, V. (2006). Active learning via transductive experimental design. In *International Conference on Machine Learning*, 1081–1088, ACM. 19

Zheng, Z., Kohavi, R. & Mason, L. (2001). Real world performance of association rule algorithms. In *Knowledge Discovery and Data Mining*, 401–406, ACM. 93

Zhou, K., Yang, S.H. & Zha, H. (2011). Functional matrix factorizations for cold-start recommendation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, 315–324, ACM. 134

Zhu, X. (2005). *Semi-supervised Learning with Graphs*. Ph.D. thesis, Carnegie Mellon University. 4

Zhu, X., Lafferty, J. & Ghahramani, Z. (2003). Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions. In *ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, 58–65. 29, 38, 44

Zimmerman, D.L. (2006). Optimal network design for spatial prediction, covariance parameter estimation, and empirical prediction. *Environmetrics*, **17**, 635–652. 27