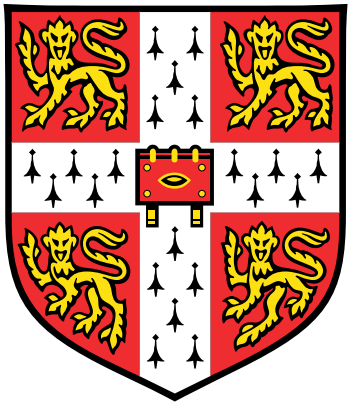
Sequential Matrix Completion



Author:
Anne MARSDEN
Churchill College

Supervisor:
Dr. Sergio BACALLADO
Advisor:
Dr. Anita FAUL

This dissertation is submitted for the degree of Masters of Philosophy
Centre for Scientific Computing
Department of Physics
University of Cambridge
August 2016

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 15,000 words including appendices, bibliography, footnotes, tables and equations.

Anne Christina Marsden
August 2016

Acknowledgements

I would first like to acknowledge my incredible advisor, Sergio Bacallado. I came into the MPhil with little statistics background and Dr. Bacallado devoted so much time to helping me bridge that gap quickly. Thank you for always being there for any of my questions, listening to all my hare-brained ideas, and giving me such a positive experience of pure research. I can't imagine this year without you as my advisor. Thank you again and again.

I next must acknowledge the Centre for Scientific Computing and all the people and parts of it. First, a big thank you to Dr. Philip Blakely, who helped me get Theano installed and running on the centre's GPUs. Thank you to Dr. Anita Faul, who developed a great machine learning community within the CSC and who was always so supportive- even when I needed a ride to the train station! And thank you to Dr. Nikos Nikiforakis for the support and advice. Also thank you to my fellow MPhil students- my coffee crew- for always being there to help answer questions I was too afraid to ask Philip, bounce ideas off of, and take long coffee breaks.

Next I would like to thank the Winston Churchill Foundation for the constant support and for providing whatever I may need to do my research. To be a part of the Churchill Scholar community was something I will never forget. Thank you to all the fellow Churchill Scholars for constantly being an inspiration and a wonderful community of people with the most interesting dinner conversations. And thank you to Churchill College and all the members of the MCR for making this year so happy and fun.

Finally I must thank the people waiting for me back home. To Gareth- thank you for making this year happy despite being so far away. To Kate and Milo thank you for being wonderful and supportive siblings and always knowing how to make me laugh. To my grandparents, aunts, uncles, and cousins- thank you for being the most wonderful family, thank you for every letter and message and thank you for the "top three of the day". And finally and most importantly, thank you to my parents- words cannot express how lucky I am to be your child and to know that, no matter what, you will be there for me.

Abstract

In this work we study sequential low rank matrix completion, or recommender systems. This topic requires analysing first) methods to accurately estimate a missing entry from a low rank matrix with only partial observations, second) methods to quantify the uncertainty in that estimate, and third) methods that exploit this information in an optimal way. Here, we construct a new recommender system. We address the task of matrix completion and uncertainty quantification using a Bayesian approach. We use a Gamma process factor model to estimate the complete matrix and the posterior distribution to quantify the uncertainty. We compare the performance of estimating the posterior using stochastic variational inference (SVI) to stochastic gradient Langevin dynamics (SGLD). To choose the sequence of entries to observe in each iteration we use Thompson sampling. As a baseline comparison we include a singular value thresholding method. Through simulation with both synthetic data sets and the MovieLens data set, we find that the Bayesian method outperforms the singular value thresholding method when the data is higher rank. The performance is particularly good when the posterior distribution is estimated using SVI.

Contents

Chapter 1

Introduction

1.1 Recommender systems

Recommender systems, or collaborative filtering, became an important area of research within the last twenty years as a multitude of settings began to arise in which users are faced with an excessive amount of information and would benefit from recommendations for products or services that suit them. A recommender system uses a database about user preferences to predict new products or services to recommend. Commonly this database is represented as a matrix where, without loss of generality, each row represents a user, each column represents a product, and the row-column pair represents the user's rating of the product. Formally, let $M \in \mathbb{R}^{DxN}$ be the full true matrix with only a few, possibly noisy, observed entries. Let $Z_t \in \left\{ e_i e_j^T ; i, j \in [n] \right\}$ and let $\mathscr{Z} = \{Z_t\}$ be our small set of observations so that for each $Z_t \in \mathscr{Z}$ we have corresponding observation Y_t as,

$$Y_t = \text{tr}(Z_t^T M) + \varepsilon_t, \tag{1.1}$$

where ε_t is the noise in the observation.

Research on recommender systems has focused on the matrix completion problem. Candès and Recht [8] showed that, if we assume the rewards matrix is of low-rank — or that user preferences are explained by a few latent features — it is possible to complete a matrix from a small set of possibly noisy observations. This research will be reviewed in Section 1.3. Our aim here is to study the matrix completion problem sequentially. Briefly, we would like to find a sequential rule or policy for product recommendation which will yield the highest possible ratings after a finite time horizon. For simplicity we assume that if our recommender system makes a recommendation to a user, the user will respond with their rating of that item.

More formally, suppose at each step we can choose an action Z_t that corresponds to observing some, possibly corrupt, entry of the matrix. Where, here, the source of corruption would be that the user does not rate the product consistently. We define the reward, or rating, for taking action Z_t at step t as,

$$r_{t,Z_t} = (\mathrm{tr}(Z_t^T X) + \varepsilon_t)\beta^{T_{t,Z_t}}, \tag{1.2}$$

where $(\varepsilon_t)_{t\geq 1}$ is a sequence of independent noise variables with mean 0 and variance σ^2. The variable T_{t,Z_t} counts the number of times that action Z_t has been chosen before time t, and the parameter $\beta \in [0,1]$ determines a geometric discounting due to observing the same entry of the matrix repeatedly. The role of β is for theoretical purposes since it allows resampling in order to construct confidence intervals for potential future methods, as well as for practical purposes since it ensures that the same product will not be recommended to the same user too many times. Note, for nonnegative entries, $\beta = 0$ is the case when we make a certain recommendation at most once and $\beta = 1$ is the case when the recommender simply aims to discover the maximum value of the matrix.

At each step t, the experimenter observes the reward r_{t,A_t} for the action chosen, A_t. We define the pseudo-regret for finite-horizon M as,

$$R_M(A) = \sup_{Z_1,\ldots,Z_M} \mathbb{E}\left(\sum_{t=1}^{M} r_{t,Z_t}\right) - \mathbb{E}\left(\sum_{t=1}^{M} r_{t,A_t}\right). \tag{1.3}$$

The goal of our recommender system is to construct an estimator policy with as little pseudo-regret as possible.

1.2 Exploration-exploitation tradeoffs and bandit algorithms

Our estimator policy relies on an accurate reconstruction of the complete matrix, M, knowing only observations corresponding to $\mathscr{Z}$. Note that this is only feasible when M is low rank. The methods to construct a good estimator typically need the observed entries to be distributed around the matrix. For instance, imagine an estimator policy that only interacts with one user. Then it would have no hope of completing the matrix to know about the other user preferences. In this sense the estimator policy must explore the user-item pairs. On the other hand, the policy must begin to exploit user-item pairs that are expected to be optimal to have as little regret as possible.

This type of problem relates to well-studied strategies on the multi-armed bandit problem. The narrative for this problem is as follows. A player is at a casino with multiple slot

machines, called a "multi-armed bandit". At each step he chooses an arm to pull and then observes the reward of his choice. The assumption is that the rewards of each machine are drawn from some underlying distribution. To perform well the player must have a good "exploration-exploitation" trade-off. That is, he must pull each arm enough to get an idea of their underlying distributions, but he must also begin to exploit the arms that seem to have high rewards. Historically, bandit problems have been analyzed using either a frequentist or Bayesian perspective. The frequentist analysis assumes the parameter of the model describing the distribution of rewards from each arm is a fixed but unknown value. The Bayesian analysis assumes that the parameter comes from some prior distribution. Suppose there are K arms with underlying parameters $\theta_1, ..., \theta_K \in \Theta$ that respectively describe the distribution of rewards for each arm. At time t we choose some action $A_{t,\theta}$, which depends on the history of rewards before it and corresponds to some arm j. The action then gets a reward X_t which comes from the i.i.d sequence of rewards from arm j, $(Y_{j,t})_{t\geq 1}$, which has distribution ν_{θ_j} and expectation $\mu_j(\theta_j)$. The measure of performance for a frequentist analysis is the cumulated regret, which at time $t = n$ is,

$$R_n(\theta) = \mathbb{E}_\theta\Big[\sum_{t=1}^{n} \mu^* - \mu_{A_{t,\theta}}\Big], \tag{1.4}$$

where $\mu^* = \max\big\{\mu_j, j \in [K]\big\}$.

With regard to minimising $R_n(\theta)$, the most effective algorithm has proved to be the Upper Confidence Bound (UCB) strategy [5]. Let X_i denote the distribution of rewards from arm i. Let $\bar{\mu}_{i,t}$ be the empirical average of s draws from arm i, so $\bar{\mu}_{i,s} = \frac{1}{s}\sum_{m=1}^{s} X_{i,m}$. A UCB algorithm uses the history of rewards to construct a confidence interval on the mean of each arm so that the true mean lies in the interval with probability $1-\alpha$, where α can be tuned. Let $C_{i,t}$ be the upper band of a confidence interval centred at $\overline{\mu}_{i,s_{i,t}}$. The policy chooses arm I_t in the tth step by computing

$$\text{argmax}_i[C_{i,t}], \tag{1.5}$$

and selecting an arm from this set. Analyzing this, one can see that arms that haven't been explored will be likely to be selected due to their large confidence interval, and as t grows, arms that have proven to be profitable will be exploited. Bubeck and Bianchi provide a bound on the regret for the UCB strategy in the case where for each arm i there is a convex function ϕ on the reals such that $\forall\lambda \in \mathbb{R}$,

$$\mathbb{E}[e^{\lambda(X_i - \mathbb{E}[X_i])}] \leq e^{\phi(\lambda)}\ , \mathbb{E}[e^{\lambda(E[X_i] - X_i)}] \leq e^{\phi(\lambda)}, \tag{1.6}$$

with this assumption they are able to construct a general upper confidence bound on the mean of each arm. They show that, in the case of Bernoulli rewards, this is essentially the best regret possible [5].

Alternatively, from the Bayesian perspective let $\pi_1, ..., \pi_K$ parameterize the prior distributions of $\theta_1, ..., \theta_K$. A Bayesian measure of performance is the regret averaged over the parameters $\theta_1, ..., \theta_K$ with respect to $\pi_1, ..., \pi_K$. Formally, the Bayesian regret up to time $t = n$ is,

$$R_n = \mathbb{E}_\pi \Big[R_n(\theta) \Big], \tag{1.7}$$

where $R_n(\theta)$ is as described in Eq. 1.4. The policy which minimises Bayes regret is known as Bayes-optimal and is typically the solution to an intractable dynamic program. An important exception is the case of a geometrically discounted multi-armed bandit, in which the Gittins index strategy is provably Bayes-optimal [29].

An effective and simple strategy from both Bayesian and frequentist perspectives is Thompson Sampling. Let $\pi_{i,t}$ denote the posterior distribution for the mean of arm i, μ_i, at the t^{th}-round. We draw from the posterior $\theta_{i,t} \sim \pi_{i,t}$ and then choose action $Z_t \in \text{argmax}\mu_{i,t}(\theta_{i,t})$. Kaufmann et al. show that Thompson sampling attains essentially the same regret as the UCB-algorithm [15].

They devise an algorithm that is inspired by both Thompson sampling and the UCB method, the "Bayes-UCB" approach. Through simulation they show that this approach proves optimal when evaluated by the frequentist regret. We briefly describe the approach. Let $Q(t, \rho)$ be the "quantile function" with respect to distribution ρ so that $\mathbb{P}_\rho(X \leq Q(t, \rho)) = t$. While this is not the same as the confidence intervals constructed in the UCB strategy, it has a similar flavor. Let n be the horizon, Π^0 be the initial prior, and c be an adjustable constant. In this notation we use θ_j to denote the parameter describing the distribution of the rewards of arm j, but we also use it to denote the probability distribution function as well.

Algorithm 1 Bayes-UCB [15]

Initialize n, Π^0, and c.
for t=1, ..., n **do**
  for j=1, ..., K **do**
    compute

$$q_j(t) = Q\Big(1 = \frac{1}{t(\log n)^c}, \theta_j^{t-1}\Big) \tag{1.8}$$

  end for
  take action $Z_t = \arg\max_{j=1,\ldots,K} q_j(t)$
  using reward $X_t = Y_{Z_t,t}$ update the posterior distribution Π^t,

$$\pi_j^t(\theta_j) \propto \theta_j(X_t)\pi_j^{t-1}(\theta_j). \tag{1.9}$$

end for

Lai and Robbins have provided a lower bound on the number of suboptimal draws for any "good policy" (i.e. the regret is $o(n)$). Kauffman et al. show that for the case of Bernoulli rewards, this algorithm achieves this lower bound. They apply the algorithm to many different types of distributions and show promising simulation results.

Another important approach to bandit problems is the ε-greedy strategy. Let $\varepsilon_t \in (0,1)$ and at each step choose action Z_t with one of the methods described above with probability $1-\varepsilon_t$ and choose an arm uniformly at random with probability ε_t. Auer et al. [5] show that if ε_t evolves in a specific manner then the regret grows logarithmically.

Finally, Russo and Van Roy [25] provide Bayesian regret bounds for the setting in which only Thompson sampling is used to determine the policy. If the set of possible actions is finite and the rewards lie in a bounded set, then we have that

$$R_n \leq 2\min\{|A|, T\} + 4\sqrt{KT(2+6\log(T))}, \tag{1.10}$$

where A denotes the set of possible actions and T is the horizon. In the case of matrix completion where $\beta > 0$, A is the set of the matrix entries and so for an $N \times D$ matrix the Bayeson regret is bounded by $2T + 4\sqrt{NDT(2+6\log(T))}$.It is worth noting that this bound is probably quite weak since the true dimensionality of a low rank matrix is smaller than $N \times D$.

Whether to choose Thompson sampling or a UCB-type algorithm depends on the setting. From a UCB-type algorithm it is generally easier to obtain frequentist guarantees. However,

tight confidence intervals can be more difficult to derive and compute than estimating the posterior distribution. In this case, Thompson sampling is more convenient and efficient.

1.3 Matrix Completion

For exploitation a good estimation method is critical, while for exploration a good quantification of uncertainty is required. Indeed, see Appendix B, which proves that the regret of any policy is bounded above by the frobenius-normed error of its estimator. Here we review methods for constructing an estimator of the partially observed matrix.

1.3.1 Convex Optimisation Methods

Recovering a low rank matrix knowing only a small portion of (possibly corrupt) matrix entries arises in a multitude of different problems including our case of recommender systems as well as dimensionality reduction, embedding problems, and multi-class learning [23]. Recall the notation, let $M \in \mathbb{R}^{DxN}$ be the true matrix, let $Z_t \in \left\{ e_i e_j^T ; i, j \in [n] \right\}$ and let $\mathscr{Z} = \{Z_t\}$ be our set of observations so that for each $Z_t \in \mathscr{Z}$ we have,

$$Y_t = \text{tr}(Z_t^T M) + \varepsilon_t, \tag{1.11}$$

where ε_t is the noise in the observation. First we give a brief summary of the current methods when there is no noise. Define $P_{\mathscr{Z}} : \mathbb{R}^{DxN} \to \mathbb{R}^{DxN}$ to be the operator such that

$$\left(P_{\mathscr{Z}}(X) \right)_{ij} = \begin{cases} X_{ij}, \text{ if } Z_{ij} \in \mathscr{Z} \\ 0, \text{ else.} \end{cases} \tag{1.12}$$

A powerful way of addressing the task of low rank matrix completion without noise is to solve

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \text{rk}(X) \\ \text{subject to} \quad & P_{\mathscr{Z}}(X) = P_{\mathscr{Z}}(M). \end{aligned} \tag{1.13}$$

However the rank minimization problem is NP-hard and there exist no efficient algorithms to solve it [7]. We instead solve the tightest convex relaxation of the problem,

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & ||X||_* \\ \text{subject to} \quad & P_{\mathscr{Z}}(X) = P_{\mathscr{Z}}(M), \end{aligned} \tag{1.14}$$

where $||X||_*$ denotes the nuclear norm, so if $X = \sum_{i=1}^{rk(X)} \sigma_i u_i v_i^*$ is the singular value decomposition of X, then $||X||_* = \sum_{i=1}^{rk(X)} \sigma_i$ is the sum of its singular values. [7] shows that, if the entries $\mathscr{L}$ are chosen uniformly at random, and with a few assumptions on the matrix that are satisfied in most cases, the unique solution to the convex nuclear norm minimization problem above recovers, with high probability, all the entries of M with no error. In [10] it is proven that this method is nearly as good as any other possible method for exact recovery.

Following the explanation in [7] we give an idea as to why the true matrix M is, with high probability, the unique minimizer to the convex nuclear norm minimization problem. The idea is to show that any perturbation around M, say $M+H$, satisfying $P_{\mathscr{L}}(M+H) = P_{\mathscr{L}}(M)$ will only have increased nuclear norm.

First we define a few bits of notation,

- $\{u_1, ..., u_{rk(M)}, v_1, ..., v_{rk(M)}\}$ are such that $M = \sum_{i=1}^{rk(M)} \sigma_i u_i v_i^*$
- $||\cdot||$ denotes the spectral norm (i.e. the largest singular value),
- P_U denotes the matrix with rows $u_1, ..., u_{rk(M)}$,
- P_V denotes the matrix with columns $v_1, ..., v_{rk(M)}$,
- T denotes the linear space spanned by $\{u_i x^*, y v_i^*\}_{i=1}^{rk(M)}$, where $x, y \in R^N$,
- $T^\perp$ denotes the space perpendicular to T,
- W denotes any matrix such that $||W|| \leq 1$, $P_U W = 0$, and $W P_V = 0$, (i.e. $W \in T^\perp$),
- $\partial ||M||_*$ denotes the subdifferential of $||\cdot||_*$ at M ,
- E denotes any matrix such that $P_T(E) = \sum_{i=1}^{rk(X)} u_i v_i^*$.

In the proofs that follow we make use of the fact that, for any subspace S, the projection operator P_S satisfies

$$\langle P_S A, B \rangle = \langle A, P_S B \rangle, \tag{1.15}$$

where $\langle A, B \rangle = \mathrm{tr}(A^* B)$.

We begin with the following theorem.

Theorem 1. *M is a solution to the convex program (1.14) if and only if there exists some $\Lambda = E + W$, with E and W as defined above, such that $P_{\mathscr{L}}(\Lambda) = \Lambda$.*

Proof 1. *The dual function of (1.14) is*

$$L(X,\lambda) = ||X||_* + \langle \Lambda, P_{\mathscr{Z}}(X-M)\rangle, \tag{1.16}$$

where $\langle \Lambda, P_{\mathscr{Z}}(X-Y)\rangle = tr(\Lambda^*(X-M))$. *Since* $||\cdot||_*$ *is a convex function, by the Karush-Kuhn-Tucker conditions, M is a solution to the convex program (1.14) if and only if* $\exists\Lambda$ *such that* $\forall X \in \mathbb{R}^{DxN}$, $L(X,\Lambda) \geq L(M,\Lambda)$, *or equivalently,*

$$||X||_* + \langle \Lambda, P_{\mathscr{Z}}(X-M)\rangle \geq ||M||_* + \langle \Lambda, P_{\mathscr{Z}}(M-M)\rangle, \tag{1.17}$$

so,

$$||X||_* \geq ||M||_* + \langle P_{\mathscr{Z}}\Lambda, M-X\rangle, \tag{1.18}$$

which is true if and only if $P_{\mathscr{Z}}\Lambda \in \partial||M||_*$. *Thus, M solves (1.14) if and only if* $\exists\Lambda$ *such that* $P_{\mathscr{Z}}\Lambda \in \partial||M||_*$. *Now, letting* $X = 0$ *we have that any* $P_{\mathscr{Z}}\Lambda \in \partial||M||_*$ *must satisfy*

$$\langle P_{\mathscr{Z}}\Lambda, M\rangle \geq ||M||_* \text{ or equivalently, } tr((P_{\mathscr{Z}}\Lambda)^*M) \geq ||M||_*, \tag{1.19}$$

where we bound the spectral norm: $||P_{\mathscr{Z}}\Lambda|| \leq 1$. *Since the spectral norm and nuclear norm are dual to one another it is true that*

$$\sup_{||W||\leq 1} tr(W^*M) = ||M||_*. \tag{1.20}$$

Therefore if $P_{\mathscr{Z}}\Lambda \in \partial||M||_*$ *from Eq 1.19 and Eq 1.20 it must be that*

$$tr((P_{\mathscr{Z}}\Lambda)^*M) = ||M||_*. \tag{1.21}$$

Notice that $P_T(M) = M$ *and thus for any X,* $tr(X^*M) = \langle X, M\rangle = \langle X, P_T(M)\rangle = \langle P_T(X), M\rangle = tr\left(P_T(X)^*M\right)$. *For some X let*

$$P_T(X) = \sum_{k_1} \lambda_{k_1} u_{k_1} x_{k_1}^* + \sum_{k_2} \lambda_{k_2} y_{k_2} v_{k_2}^*, \tag{1.22}$$

be the singular value decomposition of $P_T(X)$. Then notice that if $||X|| \leq 1$ and $X \neq E$ then

$$\begin{aligned} tr(X^*M) &= tr\left(P_T(X)^* M\right) \\ &= tr\left(\left(\sum_{k_1} \lambda_{k_1} u_{k_1} x_{k_1}^* + \sum_{k_2} \lambda_{k_2} y_{k_2} v_{k_2}^*\right)^* \left(\sum_{i=1}^{rk(M)} \sigma_i u_i v_i^*\right)\right) \\ &< \sum_{k_1} \lambda_{k_1} \sigma_{k_1} + \sum_{k_2} \lambda_{k_2} \sigma_{k_2} \\ &< \sum_{i=1}^{rk(M)} \sigma_i = ||M||_*, \end{aligned} \tag{1.23}$$

where the first inequality follows because $u_i x_i^* < u_i u_i^*$ if $x \neq u_i$ and the second inequality follows because $||X|| \leq 1$. Therefore, $P_T(P_{\mathscr{Z}}\Lambda) = \sum_{i=1}^{rk(M)} \sigma_i$. And thus we must have,

$$P_{\mathscr{Z}}\Lambda = E + W, \tag{1.24}$$

since in this case we have

$$\begin{aligned} tr((P_{\mathscr{Z}}\Lambda)^* M) &= tr\left((E+W)^* \left(\sum_{i=1}^{rk(M)} \sigma_i u_i v_i^*\right)\right) \\ &= \sum_{i=1}^{rk(M)} \sigma_i + tr\left(W^* \sum_{i=1}^{rk(M)} \sigma_i u_i v_i^*\right) \\ &= \sum_{i=1}^{rk(M)} \sigma_i = ||M||_*. \end{aligned} \tag{1.25}$$

Thus if we find some Λ supported on the image of $P_{\mathscr{Z}}$, then $\Lambda = P_{\mathscr{Z}}\Lambda = E + W$ and M solves Eq 1.14.

Now we want to show that if M solves Eq 1.14, it is unique. This follows from the following two lemmas.

Lemma 2. *Suppose there exists some $\Lambda = E + W$, $P_{\mathscr{Z}}\Lambda = \Lambda$. Then for any legal perturbation H (legal in that $P_{\mathscr{Z}}(H) = 0$) we have,*

$$||M + H||_* \geq ||M||_* + (1 - ||P_{T^\perp}(\Lambda)||)||P_{T^\perp}(H)||_* \tag{1.26}$$

Proof 2. *Since the nuclear norm and spectral norm are dual to one another there exists some Z, $||Z|| \leq 1$ such that*

$$\langle Z, P_{T^\perp}(H) \rangle = ||P_{T^\perp}(H)||_*. \tag{1.27}$$

Let $W_0 = P_{T^\perp}(Z)$ *so that*

$$\langle W_0, H\rangle = \langle P_{T^\perp}(Z), H\rangle = \langle Z, P_{T^\perp}(H)\rangle = ||P_{T^\perp}(H)||_* \tag{1.28}$$

Let $Z' = E + W_0$ *and notice that since* $||W_0|| \leq 1$ *and* $W_0 \in P_{T^\perp}$*, then* $Z' \in \partial||M||_*$*. Then by definition of a subgradient we have,*

$$||M+H||_* \geq ||M||_* + \langle Z', H\rangle. \tag{1.29}$$

Since $\Lambda = E + W$ *and* $Z' = E + W_0$ *we have* $Z' = \Lambda + W_0 - W$*. Plugging in to the above equation gives us,*

$$||M+H||_* \geq ||M||_* + \langle \Lambda, H\rangle + \langle W_0 - W, H\rangle. \tag{1.30}$$

Using fact (1.15) and that

$$\Lambda = P_{\mathscr{Z}}(\Lambda) \text{ and } W - W_0 = P_{T^\perp}(W_0 - W), \tag{1.31}$$

we can rewrite the equation above as

$$||M+H||_* \geq ||M||_* + \langle \Lambda, P_{\mathscr{Z}}(H)\rangle + \langle W_0 - W, P_{T^\perp}(H)\rangle. \tag{1.32}$$

Finally we note that

- $P_{\mathscr{Z}}(H) = 0$
- $\langle W_0, P_{T^\perp}(H)\rangle = \langle P_{T^\perp}(W_0), H\rangle = \langle W_0, H\rangle = ||P_{T^\perp}(H)||_*$
- $\langle W, P_{T^\perp}(H)\rangle \leq ||W||\ ||P_{T^\perp}(H)||_*$

Using these we get that

$$||M+H||_* \geq ||M||_* + (1 - ||W||)||P_{T^\perp}(H)||_* \tag{1.33}$$

Lemma 3. *Suppose there exists some* $\Lambda = E + W$*,* $P_{\mathscr{Z}}\Lambda = \Lambda$*, with strict inequality* $||P_{T^\perp}(W)|| < 1$*. And suppose that* $P_{\mathscr{Z}}$ *restricted to* T *is injective, then* M *is the unique solution to the convex program (1.14).*

Proof 3. *From the above proof we have that*

$$||M+H||_* \geq ||M||_* + (1 - ||W||)||P_{T^\perp}(H)||_*. \tag{1.34}$$

So we only need to show that (1) $1 - ||W|| \neq 0$ *and (2)* $||P_{T^\perp}(H)||_* \neq 0$*. Strict inequality of the spectral norm of* W *ensures (1). As for (2) notice that if* $P_{T^\perp}(H) = 0$*, then* $H \in T$*. Then by injectivity of* $P_{\mathscr{Z}}$ *and the fact that* $P_{\mathscr{Z}}(H) = 0$*, we have* $H = 0$*.*

All that is left to prove the solution to the convex program recovers exactly the true matrix is to prove the existence of such a Λ and injectivity of $P_{\mathscr{Z}}$ restricted to T. To see the details of this proof consult [9].

This proof assumed no noise in the observations. In most applications however, the observed entries of Y have some corruption. Recall the original model

$$Y_t = \text{tr}(Z_t^T M) + \varepsilon_t, \tag{1.35}$$

where ε_t is the noise in the observation. Let ε be the matrix with entries ε_t. Assume that $||P_{\mathscr{Z}}(\varepsilon)||_F < \delta$. In this case we solve

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & ||X||_* \\ \text{subject to} \quad & ||P_{\mathscr{Z}}(X - Y)||_F \leq \delta \end{aligned} \tag{1.36}$$

In [7], Candès and Plan show that, under certain conditions that assure the existence of a Λ and the injectivity of $P_{\mathscr{Z}}$ restricted to T as described above, the solution to (1.36), call it $\hat{M}$, satisfies

$$||M - \hat{M}||_F \leq 4\sqrt{\frac{C_p \min(N,P)}{p}}\delta + 2\delta, \tag{1.37}$$

where p is the fraction of observed entries and $C_p = 2 + p$.

The convex optimisation problem can be solved through an iterative Singular Value Thresholding (SVT) algorithm. In what follows, we apply a sequential policy for matrix completion which employs the SVT estimate of the matrix. The regret of this policy will depend on the rate at which the error in the matrix estimate decays (see Appendix B). In [8] it is shown that for an $N \times D$ matrix, if the number of observed entries m satisfies

$$m \geq C\sqrt{N} r \log D, \tag{1.38}$$

for some constant C, then matrix completion will be exact with very high probability. Thus, for the analysis of our estimator policies we only need to consider the horizon before which it is possible to complete the entire matrix with high accuracy.

1.3.2 Matrix Completion with Bayesian Methods

We discuss three different models to perform low rank matrix completion in a Bayesian way. The Beta-Binomial process, the infinite factor model, and the Gamma process. Regardless of the model, completing a low rank matrix in a Bayesian way requires calculating or estimating the posterior distribution of the model. Several different methods are possible and most of the literature reviewed below employs Gibbs sampling for posterior inference. The next main section of this chapter discusses stochastic variational inference and stochastic gradient Langevin dynamics as methods to sample from the posterior distribution for models with big data.

Beta-Binomial

Let $M \in \mathbb{R}^{DxN}$ be the data matrix whose column vectors are the observations $m_n \in \mathbb{R}^D$. In the Beta-Binomial process we assume that each column of Y can be written as

$$m_n = \sum_{k=1}^{K} \lambda_k z_{nk} \mathbf{u}_k \mathbf{v}_k^T + \varepsilon_n, \tag{1.39}$$

where $\lambda_k \in \mathbb{R}$, $z_{nk} \in \{0,1\}$, $\mathbf{u}_k \sim \mathrm{N}\left(0, \frac{1}{D}\mathrm{I}_D\right)$, $\mathbf{v}_k \sim \mathrm{N}\left(0, \frac{1}{N}\mathrm{I}_N\right)$, and $\varepsilon_n \in \mathbb{R}^D$. Let Z denote the $N \times K$ matrix with entries z_{nk} and columns denoted as Z_n. Note that with this form the columns of M are independent given $\{u_k\}$, Z, and λ. Let $\Lambda_n \in \mathbb{R}^{K \times K}$ be a diagonal matrix such that $(\Lambda_n)_{kk} = \lambda_k^2 z_{nk}$. We can integrate out the $\mathbf{v}_k$s to get that

$$m_n | \lambda, Z, U \sim N\left(0, \frac{1}{N} U \Lambda_n U^T + \sigma^2 \mathrm{I}_D\right), \tag{1.40}$$

where $U \in \mathbb{R}^{D \times K}$ is the matrix with columns u_k. If $z_{nk} = 1$ then observation m_n observes u_k and if $z_{nk} = 0$ then it does not. Note that the number of k for which $\exists\ n$ such that $z_{nk} = 1$ gives the rank of the matrix. The Indian Buffet Process (IBP), which is a distribution over matrices $Z \in \{0,1\}^{NxK}$ where K is unfixed, is used to define a model that favors a lower rank matrix. In practice, since K corresponds to the rank of the matrix, it exceeds D with very low probability, however in high dimensional problems D is very large and so treating K as possibly infinite makes sense. The analogy to understand the IBP is that a series of N customers in an Indian Buffet restaurant choose dishes to sample. Customer n samples a dish with probability proportional to its popularity (i.e. the number of customers that have already sampled the dish), and then explores Poisson(α/n) new dishes. In this case the customers are the observations m_n and the dishes are the u_ks. If customer m_n samples dish u_k then $z_{nk} = 1$ and otherwise $z_{nk} = 0$ [13]. One can show that for any permutation σ

we have that $P(Z_1,...Z_n) = P(Z_{\sigma(1)},...,Z_{\sigma(n)})$, i.e. exchanging the order of the customers gives the same probability. Given some underlying distribution B that renders the sequence $\{Z_i\}$ conditionally independent, by the de Finetti Theorem it holds that for any infinitely exchangeable distribution

$$P(Z_1,...,Z_n) = \int \prod_{i=1}^{N} P(Z_i|B)dP(B). \tag{1.41}$$

Thibaux & Jordan ([28]) find that the underlying distribution that renders the sequence of customers conditionally independent is the Beta-Bernoulli process. We draw $\pi_k \sim \text{Beta}(a/K,1)$ and then $z_{nk}|\pi_k \sim \text{Bernoulli}(\pi_k)$. The Indian Buffet Process is the distribution over Z as $K \to \infty$. For a more complete review on the Indian Buffet Process and the Beta Bernoulli Process the reader is referred to [13], [27]. In practice we draw $\pi_k \sim \text{Beta}(a/K, b(K-1)/K)$ which allows the number of new "dishes" explored by customer n to have some dependence on the diversity of dishes already explored rather than being a draw from $\text{Poisson}(\alpha/n)$. One can see that through the choice of a and b one can tune the rank of the data. We define the full model below as described in [32],

$$\begin{aligned}
m_n &\sim N\left(0, \frac{1}{N}U\Lambda_n U^T + \sigma^2 \mathrm{I}_D\right) \\
u_k &\sim N\left(0, \frac{1}{D}\mathrm{I}_D\right) \\
\lambda_k &\sim N(0, \beta^{-1}) \\
z_{nk} &\sim \text{Bernoulli}(\pi_k) \\
\pi_k &\sim \text{Beta}(a/K, b(K-1)/K) \\
\beta &\sim G(1,1).
\end{aligned} \tag{1.42}$$

For a more complete review of results with this approach see [32]. Large-scale problems are considered using Gibbs sampling yielding encouraging results compared to existing methods.

Infinite Factor Model

In the Infinite Factor Model we assume that each column of M can be written as

$$m_n = \sum_{k=1}^{K} \mathbf{u}_k \mathbf{v}_k^T + \varepsilon_n, \tag{1.43}$$

where the prior construction is as follows.

$$
\begin{aligned}
\mathbf{u}_{ij} &\sim N\left(0, \left(\gamma_{ij}\prod_{\ell=1}^{j}\delta_\ell\right)^{-1}\right), \quad \mathbf{v}_{ij} \sim N(0,1) \\
\varepsilon_n &\sim N(0,s) \\
\gamma_{ij} &\sim \text{Gamma}(1/2,1/2) \\
\delta_1 &\sim \text{Gamma}(a_1,1), \quad \delta_\ell \sim \text{Gamma}(a_2,1) \quad \ell \geq 2 \\
a_1 &\sim \text{Gamma}(1,b), \quad a_2 \sim \text{Gamma}(1,b) \\
s &\sim \text{Exp}(1).
\end{aligned}
\tag{1.44}
$$

This model begins with a conceivably infinite number of factors and a finite number of factors manifests in learning a $\delta_\ell > 1$ and a bound on each γ_{ij}. In this case as $k \to \infty$ we have $\left(\gamma_{ij}\prod_{\ell=1}^{j}\delta_\ell\right)^{-1} \to 0$, which in turn means that $u_k \to \mathbf{0}$. Bhattacharya and Dunson [3] use this model with a simple Gibbs sampler to learn the posterior distribution. They show it performs well compared to other competitive methods in covariance matrix estimation, regression coefficient estimation, and variable simulation.

Gamma Process

Finally, in the Gamma process setting we assume that our data can be modeled so that $m_n|x_n \sim N(\mathbf{W}x_n, \sigma^2\mathbf{I})$ where $x_n \sim N(0,\mathbf{I}_K)$. We can integrate out x_n to get that $m_n \sim N(0, \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I})$. In this model we place a gamma prior on the entries of W and use a hierarchical gamma process to complete the prior construction for W,

$$
\begin{aligned}
W_{dk}|r_k,\gamma &\sim \text{G}(\gamma r_k,\gamma) \\
r_k|\gamma_0,c_0 &\sim \text{G}(\gamma_0/K,c_0) \\
\gamma &\sim \text{G}(1,1) \\
\gamma_0 &\sim \text{G}(1,1) \\
c_0 &\sim \text{G}(1,1).
\end{aligned}
\tag{1.45}
$$

Where $G(x|a,b)$ denotes the gamma distribution

$$
G(x|a,b) = \frac{b^a}{\Gamma(a)}x^{a-1}e^{-xb}. \tag{1.46}
$$

As was the case for the two models previously discussed, this model in practice assumes a large fixed value of K. Unlike the other two models, it is difficult to explicitly calculate the number of latent factors. The Gamma process performs a soft thresholding on the true number

of latent factors, K^*, in that along each row of W there will be roughly K^* positive entries and $K - K^*$ entries closer to zero. We now discuss how to learn the posterior distribution in these Bayesian models in a more efficient way.

For all of these Bayesian methods, it is key to learn the posterior distribution in a computationally efficient way. MCMC methods can be unacceptably slow and while the authors for the infinite factor model and the Beta-Binomial process are able to use efficient Gibbs-sampling, they are not able to apply the models to big data. This motivates the next section, which introduces a methodology for fast and efficient posterior learning.

1.4 Variational Inference

1.4.1 Introduction

A key task in many machine learning applications is to calculate posterior moments in Bayesian analyses of probabilistic models. Historically, Markov Chain Monte Carlo (MCMC) has been used to do this [4]. In this section we present an alternative method, stochastic variational inference (SVI). Stochastic variational inference approximates the posterior distribution and as algorithm evolves, minimizes the "distance" between the true posterior and the approximated variational distribution. SVI has been used in many different applications including our application of recommender systems, large scale document classification [14], computer vision [4], and a multitude of data sets across many different fields. It has been found to be faster and more easily scalable to big data sets than MCMC methods. In what follows, we develop the general theory of stochastic variational inference and describe methods to make it more scalable to big data.

Let $\pi(x,D)$, $\pi(x)$, and $\pi(x|D)$ denote the joint, prior, and posterior distributions respectively, where D is the observed data. Suppose $\pi(x|D)$ is intractable, then let q_θ be the "variational distribution" which is an estimate of $\pi(x|D)$. The goal of Variational Inference is to optimize the parameters θ of q_θ so that it is as close as possible to $\pi(x|D)$. "Closeness" is measured by the Kullback-Leibler divergence:

$$\mathbb{KL}(\pi(\cdot|D)||q_\theta) = \int_\chi \pi(\cdot|D)\log\frac{\pi(x|D)}{q_\theta(x)}dx = \mathbb{E}_{\pi(x|D)}[\log\pi(x|D)] - \mathbb{E}_p[\log q_\theta(x)]. \quad (1.47)$$

In general, $\mathbb{KL}(\pi(\cdot|D)||q_\theta)$ cannot be calculated since $\pi(x|D)$ is intractable. Instead of minimizing the KL divergence one can obtain the same result by maximizing the evidence

lower bound (ELBO):

$$\begin{aligned} L(\theta) &= \int_{\chi} q_\theta(x) \log \frac{\pi(x,D)}{q_\theta(x)} dx \\ &= \mathbb{E}_{q_\theta}[\log \pi(x,D)] - \mathbb{E}_{q_\theta}[\log q_\theta(x)] \\ &= \mathbb{E}_{q_\theta}[\log \pi(x,D)] - \mathbb{E}_{q_\theta}[\log q_\theta(x)]. \end{aligned}$$

Note that $\pi(x,D) = \pi(x|D)\pi(D)$ and is much easier to evaluate than $\pi(x|D)$. Another way of writing $L(\theta)$ shows clearly that it is a lower bound on the log likelihood of the data:

$$\begin{aligned} L(\theta) &= \int_{\chi} q_\theta(x) \log \frac{\pi(x|D)}{q_\theta(x)} dx + \log \pi(D) \int_{\chi} q_\theta(x) dx \\ &= -\mathbb{KL}(q_\theta || \pi(\cdot|D)) + \log \pi(D) \leq \log \pi(D), \end{aligned}$$

where the last inequality holds because the KL-divergence is nonnegative (A.1).

In the Mean Field Method we assume that our approximation to the posterior can be fully factorized:

$$q_\theta(x) = \prod_{i=1}^{D} q_{\theta_i}(x_i). \tag{1.48}$$

With this fully factored form we can analytically derive an expression for each $q_{\theta i}(x_i)$ that produces a local maximum for $L(\theta)$ (A),

$$q_{\theta_i}(x_i) = \frac{1}{Z_i} \exp\left(\mathbb{E}_{q_{-i}}\left[\log \pi(x,D)\right]\right), \tag{1.49}$$

where q_{-i} is the posterior distribution of all parameters excluding the i^{th} and Z_i is a normalization constant. When Equation 1.49 can be solved for analytically, the variables are updated via coordinate ascent. Coordinate ascent variational inference (CAVI) works similarly to Gibbs-Sampling where the parameters are iteratively set to their local maximum, dependent on the setting of the other parameters [12]. If we are working with conjugate exponential families then the above expression is often in closed form and can be calculated exactly [14]. However, conjugate exponential families are sometimes limiting and a more complex model is needed to describe the data. In these cases, the update equation is not necessarily available analytically. Stochastic Variational Inference (SVI) uses the idea that the posterior distribution can instead be updated using gradient ascent, where the gradient is a noisy, unbiased, estimate of the true gradient. If the noisy estimate of the gradient can be obtained more easily then this allows variational inference to be applied to many types of models and larger data sets.

We update $q_{\theta_i}(x_i)$ to maximize $L(\theta)$ by taking a noisy, but unbiased estimate of the gradient

$$\nabla_\theta L(\theta) = \nabla_\theta \mathbb{E}_{q_\theta}[\log\pi(x,D) - \log q_\theta(x)]. \tag{1.50}$$

Note that we update $q_{\theta_i}(x_i)$ by updating only the parameter θ_i that describes it within some probability distribution form, we do not allow $q_{\theta_i}(x_i)$ to change to a different form of probability distribution. We thus update

$$\theta^{(t)} = \theta^{(t-1)} + \rho_t \nabla_\theta L(\theta^{(t-1)}), \tag{1.51}$$

where ρ_t is the step-size. Robbins & Munro [24] showed that if the estimate of the gradient is unbiased, so $\mathbb{E}(\nabla_\theta L(\theta)) = \nabla_\theta L(\theta)$ and

$$\sum \rho_t = \infty \text{ and } \sum \rho_t^2 < \infty \tag{1.52}$$

then convergence to the optimal θ is assured. Although convergence is assured, it can be unacceptably slow. We discuss our method of increasing the speed of convergence in the next chapter.

1.4.2 Big Data and Variance Reduction Methods

A difficulty of stochastic variational inference is scalability. Many models arise in the context of big data for which there is a small set of global parameters and a large set of local parameters [14]. Examples of this type of model are the hierarchical Dirichlet process topic model, latent Dirichlet allocation, and the model we use later in this work. In these types of situations, the ELBO can be separated into a global term and a sum of local terms. Let $p(x)$ denote the joint distribution $\pi(x,D)$ for simplicity. Let ϕ_i denote the i^{th} local variable. Then the ELBO is,

$$L(\theta) = \mathbb{E}_{q_{\theta_\beta}}[\log\pi(\beta) - \log q_{\theta_\beta}(\beta)] + \sum_{i=1}^{N} \mathbb{E}_{q_{\theta_{\phi_i}}}[\log\pi(\phi_i|\beta) - \log q_{\theta_{\phi_i}}(\phi_i|\beta)]. \tag{1.53}$$

Variational inference using coordinate ascent can be slow because the global variational parameter update depends on all the local variational parameters. Thus the local variational parameters must all be optimized before the global parameters can be reevaluated. Hoffman et al. [14] alter this so that the gradients of the ELBO are computed much more cheaply but are still unbiased estimates. A uniformly picked local variational parameter is locally optimized given fixed values of the global variational parameters. Then the global variational parameters are updated via stochastic gradient ascent using the optimized local variational

parameter as though it represented the entire set of local variational parameters. Choose $t \in \{1,...,N\}$ uniformly and optimize ϕ_t with β fixed. Then approximate the ELBO as

$$L_t(\theta) = \mathbb{E}_{q_{\theta_\beta}}[\log\pi(\beta) - \log q_{\theta_\beta}(\beta)] + N\mathbb{E}_{q_{\theta_{\phi_t}}}[\log\pi(\phi_t|\beta) - \log q_{\theta_{\phi_t}}(\phi_t|\beta)]. \tag{1.54}$$

And since $\mathbb{E}_t[L(\theta)] = L(\theta)$ this is an unbiased estimate that is much cheaper to compute. This is used to update the global parameters. Then the local variational parameters are updated via coordinate ascent (which could be approximated stochastically if the update equation cannot be solved for analytically).

Algorithm 2 Scalable Stochastic Variational Inference [14]

Initialize $\theta^{(0)}$ and set a step size schedule ρ_t.
repeat
  Sample $\beta \sim q_{\theta_\beta}$.
  Choose index $t \sim \text{Unif}(1,...,N)$.
  Optimize the local parameters ϕ_t with respect to $\beta^{(t)}$ to construct $L_t(\theta)$.
  Update $\theta_\beta^{(t+1)} = \rho_t \theta_\beta^{(t)} + (1-\rho)\nabla_{\theta_\beta} L_t(\theta^{(t)})$.
  Update local parameters ϕ_i using coordinate ascent.
until Convergence

Now we consider maximizing the ELBO via stochastic gradient ascent. Let $f(x) = \log p(x,D) - \log q_\theta(x)$, we have the following identity:

$$\nabla_\theta L(\theta) = \nabla_\theta \mathbb{E}_{q_\theta}[f(x)] = \mathbb{E}_{q_\theta}[f(x)\nabla_\theta \log q_\theta(x)]. \tag{1.55}$$

We can approximate this stochastically as

$$\nabla_\theta \mathbb{E}_q[f(x)] \approx \frac{1}{S}\sum_{s=1}^{S} f(x^{(s)})\nabla_\theta \log q_\theta(x^{(s)}), \tag{1.56}$$

where $x^{(s)} \sim q_\theta$. Let $g(x)$ be a function whose expectation with respect to q_θ can be calculated analytically and that approximates $f(x)$ well in the highly probable regions of q_θ. An important component of the stochastic gradient method is to reduce the variance of the approximation to the gradient. Several different methods have been introduced to do this [22], [19]. One popular method is the control variate method, which uses a tractable function g that is highly correlated with the intractable update equation and then corrects the gradient for bias [22]. Here, by "tractable" we mean that g has an analytical form. We construct our

control variate $\hat{f}$ so that it has the same expectation as $f(x)$ and variance as small as possible

$$\hat{f}(x) = f(x) - a\left(g(x) - \mathbb{E}_q[g(x)]\right). \tag{1.57}$$

Choosing $a = \frac{\text{Cov}(f,g)}{Var(g)}$ ensures that $\hat{f}$ has minimal variance, however the variance and covariance are unknown and so are approximated with the sample variance and covariance that are generated by the stochastic variational algorithm. It can be shown that the reduction in variance is directly related to the correlation between f and g so that the variance is reduced more when f and g are more highly correlated. The stochastic approximation with reduced variance then becomes

$$\begin{aligned}\nabla_\theta \mathbb{E}_q[\hat{f}(x)] &\approx \nabla_\theta \mathbb{E}_q[f(x) - a\left(g(x) + \mathbb{E}_q[g(x)]\right)] \\ &= a\nabla_\theta \mathbb{E}_q[g(x)] + \frac{1}{S}\sum_{s=1}^{S}\left(f(x^{(s)}) - ag(x^{(s)})\right)\nabla_\theta \log q(x^{(s)}|\theta).\end{aligned} \tag{1.58}$$

This describes how to construct a control variate for $f(x)$, however it would be better to construct a control variate for $f(x)\nabla_\theta \log q(x|\theta)$ which follows the same method with a modification described in [22]. A simple form of the algorithm is given below.

Algorithm 3 Stochastic Variational Inference using Control Variates [22]

Initialize $\theta^{(0)}$ and step size schedule ρ_t.
repeat
  Let $g(x)$ be a tractable function that is highly correlated with $f(x)$.
  Sample a small collection $x^{(s)} \sim q(x|\theta)$ to form a.
  Using the samples to approximate $\text{Cov}(f,g)$ and $\text{Var}(g)$ set $a = \frac{\text{Cov}(f,g)}{\text{Var}(g)}$.
  Sample $x^{(s)} \sim q(x|\theta)$ for $s = 1,...,S$, where S is an appropriate sample size.
  Construct stochastic gradient: $\delta = \frac{1}{S}\sum_{s=1}^{S}\left(f(x^{(s)}) - ag(x^{(s)})\right)\nabla_\theta \log q(x^{(s)}|\theta)$.
  Update $\theta^{(t+1)} = (1-\rho_t)\theta^{(t)} + \rho_t\delta$.
until Convergence of parameter θ

The next approach we discuss to control the variance of the approximated gradient is what we will use later on. In this approach, the variational distribution is expressed as a transformation of a standard random variate [19]. Let $\mathbf{z} \sim \pi(\mathbf{z})$ be a random variable such that $\mathbf{x} = \psi(\mathbf{z},\theta)$ has the same distribution as $\mathbf{x} \sim q_\theta$. (For instance let $z \sim U[0,1]$ and define $\psi(\mathbf{z},\theta)$ to be the inverse CDF: $\psi(\mathbf{z},\theta) = F_\theta^{-1}(z)$). We can rewrite the form of our update

equation in terms of this random variate $\mathbf{z}$,

$$\nabla_\theta \mathbb{E}_{q_\theta}[f(x)] = \mathbb{E}_{\pi(\mathbf{z})}[\nabla_\theta \psi(\mathbf{z},\theta)\nabla_x f(\psi(\mathbf{z},\theta)]. \tag{1.59}$$

We approximate this by Monte Carlo by first drawing $z^{(s)} \sim \pi(z)$ and then setting $x^{(s)} = \psi(z^{(s)},\theta)$ and approximating the gradient as

$$\nabla_\theta \mathbb{E}_{q_\theta}[f(x)] \approx \frac{1}{S}\sum_{s=1}^{S} \nabla_\theta \psi(\mathbf{z}^{(s)},\theta)\nabla_x f(x^{(s)}). \tag{1.60}$$

This transformation generally reduces the variance so well that only one sample is needed [19]. Some idea as to why this reduces the variance is that on the left side of Equation 1.59 we must compute the gradient of $f(x)$ with respect to the parameter θ that describes its variational distribution, but on the right side we compute the gradient of $f(x)$ directly with respect to x.

1.5 Langevin Method

An alternate method to estimate an intractable posterior distribution is the stochastic gradient Langevin dynamics method. Recall that $\pi(x,D)$, $\pi(x)$, and $\pi(x|D)$ denote the joint, prior, and posterior distributions respectively. Similar to the Metropolis-Hastings algorithm we can construct a Markov chain to converge to the target distribution. The idea is that if we know a density function up to a constant of proportionality we can define a "Langevin diffusion" with the target function as the stationary distribution. We say X is a Langevin diffusion for distribution f with "scaling" σ if it satisfies the following stochastic differential equation,

$$dX(t) = b(X(t))dt + \sigma(X(t))dB(t),\ X(0) \in \mathbb{R}^n, \tag{1.61}$$

where $B(\cdot)$ is an n-dimensional Brownian motion, $a(x) = \sigma(x)\sigma'(x)$ is an $n \times n$ symmetric positive-definite matrix with locally uniformly Holder continuous entries, and $b(x)$ satisfies for each coordinate $b_i(x)$,

$$b_i(x) = \frac{1}{2}\sum_{j=1}^{n} a_{ij}(x)\partial \log\frac{f}{\partial x_j} + \delta^{1/2}(x)\sum_{j=1}^{n}\frac{\partial}{\partial x_j}\left(a_{ij}(x)\delta^{-1/2}(x)\right), \tag{1.62}$$

where $\delta(x) = \det a(x)$. Kent [16] shows that if X satisfies the above equation, and a few non-explosion criteria, then X has f as its unique stationary distribution.

We know the posterior distribution up to a constant since $\pi(x|D) \propto \pi(D|x)\pi(x)$, so we use this method. Now let $\sigma = \mathbf{I}$, $f = \pi(x, D)$ and let $\eta_t \sim N(0, \varepsilon_t)$ and consider the discretized version of Equation 1.61,

$$\nabla x_t = \frac{\varepsilon_t}{2} \nabla \log(\pi(x|D)) + \eta_t. \tag{1.63}$$

This discrete version must be corrected by a Metropolis-Hastings algorithm. However, if we let $\varepsilon \to 0$ as $t \to \infty$ then, in the limit, this discrete version approaches Langevin dynamics and thus $\pi(x(|D))$ is its stationary distribution. Thus for large enough t, updating x_t becomes the same as drawing it from its posterior distribution.

Welling and Teh [30] combine the idea of Langevin diffusion with stochastic gradients to form the "stochastic gradient Langevin dynamics" method. At each time t we update the parameters of the posterior distribution, x, by taking the step

$$\Delta x_t = \frac{\varepsilon_t}{2} \left(\nabla \log \pi(x_t) + \sum \nabla \log \pi(D_i|x_t) \right) + \eta_t, \tag{1.64}$$

where ε_t decays in such a way that,

$$\sum_{t=0}^{\infty} \varepsilon_t = \infty \text{ and } \sum_{t=0}^{\infty} \varepsilon_t^2 < \infty. \tag{1.65}$$

For smaller values of t we have the stochastic gradient method which gets us closer to the maximum a posteriori estimate. As shown by Welling and Teh, as $t \to \infty$, Eq 4.1 approaches Langevin dynamics and converges to the posterior distribution.

Chapter 2

Approximate Thompson sampling for sequential matrix completion

The aim of this chapter is to construct a recommender system that uses the gamma process factor model introduced in the previous chapter to estimate a partially observed matrix and Thompson sampling to determine the next sequence of entries to observe.

In this application of the gamma process factor model, slightly differently from the treatment in the first chapter, we treat the columns of the true matrix as parameters and we have the observation at time t as,

$$Y_t = \text{tr}\left(Z_t^T M\right) + \varepsilon_t, \tag{2.1}$$

where, again, $\text{tr}\left(Z_t^T M\right)$ corresponds to some m_{ij} entry of M. We assume the N columns of the true matrix, denoted by $m_1, ..., m_N \in \mathbb{R}^P$ are drawn from a multivariate normal

$$m_n | x_n \sim N(\mathbf{W}x_n, \sigma^2\mathbf{I}), \tag{2.2}$$

where $x_n \sim N(0, \mathbf{I})$. We can integrate out x_n to get that $m_n \sim N(0, \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I})$. In this model we place a gamma prior on the entries of W and use a hierarchical gamma process to complete the prior construction for W,

$$\begin{aligned} W_{dk} | r_k, \gamma &\sim \text{G}(\gamma r_k, \gamma) \\ r_k | \gamma_0, c_0 &\sim \text{G}(\gamma_0/K, c_0) \\ \gamma &\sim \text{G}(1,1) \\ \gamma_0 &\sim \text{G}(1,1) \\ c_0 &\sim \text{G}(1,1). \end{aligned} \tag{2.3}$$

In the case that $\beta = 0$, which is the case we test empirically in the next chapter, each entry is only observed once. Because of this we simply treat each m_{ij} as the noisy observation rather than a parameter to update. To choose an entry sequentially in Thompson sampling, we must sample the posterior distribution of the unobserved entries. Let m_n be our vector of observations such that $m_n = \begin{bmatrix} m_{nU} \\ m_{nO} \end{bmatrix}$ where m_{nU} is unobserved, and m_{nO} is observed. The posterior distribution can be written as,

$$p(m_{nU}|m_{1O},...,m_{NO},W,\sigma) = p(m_{nU}|m_{nO},W,\sigma)p(W,\sigma|m_{1O},...,m_{NO}). \tag{2.4}$$

The first factor on the right hand side is a normal distribution, which we sample efficiently using a method described below. The more difficult task is sampling the posterior of W and σ in the second factor. The sections below describe several approximate schemes to perform this task. For now assume we are able to sample from this distribution.

To sample from $p(m_{nU}|m_{nO},W,\sigma)$, let $\Sigma_n = WW^T + \sigma^2 I$, but with permuted rows and columns so that if $\Sigma_{nUU}, \Sigma_{nOO}$ correspond to the covariance between m_{nU} and m_{nO} respectively then,

$$\Sigma_n = \begin{bmatrix} \Sigma_{nUU} & \Sigma_{nUO} \\ \Sigma_{nOU} & \Sigma_{nOO} \end{bmatrix}. \tag{2.5}$$

Then

$$m_{nU,W,\sigma}|m_{nO} \sim N(\bar{\mu}_n, \bar{\Sigma}_n), \tag{2.6}$$

where

$$\bar{\mu}_n = \Sigma_{nUO}\Sigma_{nOO}^{-1} m_{nO}, \tag{2.7}$$

and

$$\bar{\Sigma}_n = \Sigma_{nUU} + \Sigma_{nUO}\Sigma_{nOO}^{-1}\Sigma_{nOU} \tag{2.8}$$

For our policy to choose entries we employ Thompson sampling. For computational efficiency, rather than observing only a single entry at a time and then re-evaluating the posterior we observe N_{Obs} at each step.

Before going into detail in both posterior estimation methods we mention that both stochastic variational inference and stochastic gradient Langevin dynamics estimate the posterior distribution using the log joint distribution given only the observed entries along the way. To construct this we must modify the covariance matrix for each m_n as described

previously and the log joint distribution is,

$$\begin{aligned}\log p(m_{1O},...,m_{NO},W,r,\gamma,\gamma_0,c_0) = &\sum_n \left(-\frac{1}{2}\log|\bar{\Sigma}_{nOO}| - \frac{1}{2} m_{nO}^T (\bar{\Sigma}_{nOO})^{-1} m_{nO}) \right) \\ &+ \sum_d \sum_k \left(\gamma r_k \log(\gamma) - \log\Gamma(\gamma r_k) + (\gamma r_k - 1)\log W_{dk} - W_{dk}\gamma \right) \\ &+ \sum_k \left(\frac{\gamma_0}{K}\log(c_0) - log(\Gamma(\frac{\gamma_0}{K}) + (\frac{\gamma_0}{K} - 1)\log r_k - c_0 r_k \right) \\ &- \gamma - \gamma_0 - c_0.\end{aligned} \tag{2.9}$$

On an implementation note, to sample a conditional multivariate normal efficiently we use the trick described in [11]. To sample from $N(\bar{\mu}_n, \bar{\Sigma}_n)$ we sample

$$Z = \begin{pmatrix} Z_{nU} \\ Z_{nO} \end{pmatrix} \sim N(0, WW^T + \sigma^2 I), \tag{2.10}$$

and use this to set,

$$m_{nU} = Z_{nU} + \Sigma_{12}\Sigma_{22}^{-1}(m_{nO} - Z_{nO}), \tag{2.11}$$

refer to [11] for a full explanation of this method.

For the more general case of $\beta \in (0,1]$ we must reformulate the method. In this paper, the algorithms we use do not construct confidence intervals, however it may be useful to do so in later work. In order to construct a confidence interval, we must be able to sample the same entry more than once. Thus, for theoretical purposes, we must have $\beta > 0$. The logarithm of the joint distribution in this case is,

$$\begin{aligned}\log p\left(Y_1,...,Y_t,M,W,\sigma,\varepsilon_t\right) \propto &-\frac{1}{2}\sum_{k=1}^{t}\log(\varepsilon) - \frac{1}{2\varepsilon^2}\sum_{i,j}\sum_{k=(i,j)}\left(Y_k - m_{ij}\right)^2 \\ &-\frac{N}{2}\log\left(|WW^T + \sigma^2 I|\right) + \sum_{n=1}^{N} m_n^T \left(|WW^T + \sigma^2 I|\right)^{-1} m_n \\ &+ \sum_d \sum_k \left(\gamma r_k \log(\gamma) - \log\Gamma(\gamma r_k) + (\gamma r_k - 1)\log W_{dk} - W_{dk}\gamma \right) \\ &+ \sum_k \left(\frac{\gamma_0}{K}\log(c_0) - log(\Gamma(\frac{\gamma_0}{K}) + (\frac{\gamma_0}{K} - 1)\log r_k - c_0 r_k \right) \\ &- \gamma - \gamma_0 - c_0.\end{aligned} \tag{2.12}$$

In this case we estimate the posterior distribution for the parameters of M as well and then for each (i, j) draw corresponding

$$Y_k \sim N\left(\mathrm{tr}\left(Z_k^T Y\right), \varepsilon_k\right) p\left(Y|W,\sigma\right) p\left(W,\sigma|X_1,...,X_t\right). \tag{2.13}$$

2.0.1 Stochastic Variational Inference Method

In order to calculate the posterior distribution we must approximate it. We use two different methods and compare their performance- stochastic variational inference (SVI) and stochastic gradient Langevin dynamics (SGLD). First we discuss explicitly how we use SVI.

For the variational posterior, we can use the form of the update equation 1.49 to see that the variational posterior should be a product of gamma distributions,

$$q(W,r,\gamma,\gamma_0,c_0) = \prod_{d=1}^{D}\prod_{k=1}^{K} q(W_{dk}|a_{dk},b_{dk}) \prod_{k=1}^{K} q(r_k|a_{r_k},b_{r_k}) q(\gamma|a_\gamma,b_\gamma) q(\gamma_0|a_{\gamma_0},b_{\gamma_0}) q(c_0|a_{c_0},b_{c_0}). \tag{2.14}$$

Normally we would iteratively maximize the ELBO with respect to each variational posterior by setting each $q_j(x_j)$ to the locally optimal update derived previously. However this can only be solved analytically for c_0, thus we use the stochastic gradient method for each parameter. We will derive the update equation for c_0 to demonstrate how the mean field approximation simplifies the update,

$$\begin{aligned}\log q(c_0|a_{c_0},b_{c_0}) = \int_{W_{dk}} \int_{r_k} \int_{\gamma} \int_{\gamma_0} \Bigg(\sum_{i=1}^{N} \log N(\mathbf{m_i}|0, \mathbf{W}\mathbf{W}^T + \sigma^2 I) + \sum_{d=1}^{D}\sum_{k=1}^{K} \log G(W_{dk}|\gamma r_k, \gamma) + \\ \sum_{k=1}^{K} \log G(r_k|\gamma_0/K, c_0) + \log G(\gamma|1,1) + \log G(\gamma_0|1,1) + \\ \log G(c_0|1,1)\Bigg) \prod_{d=1}^{D}\prod_{k=1}^{K} q(W_{dk}|a_{dk},b_{dk}) \prod_{k=1}^{K} q(r_k|a_{r_k},b_{r_k}) q(\gamma|a_\gamma,b_\gamma) q(\gamma_0|a_{\gamma_0},b_{\gamma_0}).\end{aligned} \tag{2.15}$$

Because the mean field assumption makes the posterior a product of independent distributions this simplifies to

$$\left[\sum_{k=1}^{K} \int_{r_k} \int_{\gamma_0} \log G(r_k|\gamma_0/K, c_0) q(r_k|a_{r_k},b_{r_k}) q(\gamma_0|a_{\gamma_0},b_{\gamma_0})\right] + \log G(c_0|1,1) + \mathrm{const.} \tag{2.16}$$

Working this out one finds that indeed the variational posterior is a gamma distribution with parameter updates

$$a_{c_0}^{(t+1)} = \frac{\Gamma(a_{\gamma_0}^{(t)} + 1)}{\Gamma(a_{\gamma_0}^{(t)}) b_{\gamma_0}^{(t)}} + 1, \tag{2.17}$$

and

$$b_{c_0}^{(t)} = \sum_{k=1}^{K} \frac{\Gamma(a_{r_k}^{(t)} + 1)}{\Gamma(a_{r_k}^{(t)}) b_{r_k}^{(t)}}. \tag{2.18}$$

For the other parameters, we cannot solve the update analytically because it amounts to integrating over the gamma function. Thus we use the stochastic gradient method where, for computational ease, we use the very similar log-normal distribution rather than the gamma distribution. To be clear, the variational posterior construction is,

$$\log W_{dk} \sim N(\mu_{W_{dk}}, \sigma^2_{W_{dk}}), \tag{2.19}$$

$$\log r_k \sim N(\mu_{r_k}, \sigma^2_{r_k}), \tag{2.20}$$

$$\log \gamma \sim N(\mu_\gamma, \sigma^2_\gamma), \tag{2.21}$$

$$\log \gamma_0 \sim N(\mu_{\gamma_0}, \sigma^2_{\gamma_0}), \tag{2.22}$$

$$\log c_0 \sim N(\mu_\gamma, \sigma^2_{c_0}). \tag{2.23}$$

The ELBO is $\mathbb{E}_q(\log p(x, D)) + H(q)$, where q represents the product of variational posterior gamma distributions and $H(q)$ is the entropy. The log joint is given in the previous section and the entropy is,

$$H(q) \propto \left(\sum_d \sum_k \log \sigma_{W_{dk}} \exp(\mu_{W_{dk}} + 1/2) \right) + \left(\sum_k \log \sigma_{r_k} \exp(\mu_{r_k} + 1/2) \right) \tag{2.24}$$

$$+ \log \sigma_\gamma \exp(\mu_\gamma + 1/2) + \log \sigma_{\gamma_0} \exp(\mu_{\gamma_0} + 1/2) + \log \sigma_{c_0} \exp(\mu_{c_0} + 1/2). \tag{2.25}$$

Algorithm 4 Stochastic Variational Inference for Gamma Process Factor Analysis

repeat

For each parameter W_{1-DK}, r_{1-K}, γ, γ_0, (c_0), sample a separate $z_d \sim N(0,1)$.

Set each parameter to log-normal: $\log W_{dk} = \mu_{W_{dk}} + z_{W_{dk}} \sigma^{\frac{1}{2}}_{W_{dk}}$, $\log r_k = \mu_{r_k} + z_{r_k} \sigma^{\frac{1}{2}}_{r_k}$, etc.

Set $\mathbf{g}_W = \nabla_W f$, $\mathbf{g}_r = \nabla_r f$, $\mathbf{g}_g = \frac{\partial}{\partial g} f$, $\mathbf{g}_{g_0} = \frac{\partial}{\partial g_0} f$, $\left(\mathbf{g}_{c_0} = \frac{\partial}{\partial c_0} f\right)$.

Using chain rule compute $\mathbf{g}^{\mu_W}_{\text{ELBO}}$, $\mathbf{g}^{\sigma_W}_{\text{ELBO}}$, etc.

Compute step size Δ using AdaDelta.

Update $\mu_W \leftarrow \mu_W + \Delta_{\mu_W} \mathbf{g}^{\mu_W}_{\text{ELBO}}$, repeat for μ_r,μ_γ, μ_{γ_0}, μ_{c_0}.

Update $\sigma_W \leftarrow \sigma_W + \Delta_{\sigma_W} \mathbf{g}^{\sigma_W}_{\text{ELBO}}$, repeat for σ_r,σ_γ, σ_{γ_0}, σ_{c_0}

until convergence

We express the variational distribution as a transformation of a standard random variate to control the variance of the gradient as discussed in Section 1.4.2. That is we use that,

$$\nabla_\theta \mathbb{E}_{q_\theta}[f(x)] = \mathbb{E}_{\pi(\mathbf{z})}[\nabla_\theta \psi(\mathbf{z},\theta)\nabla_x f(\psi(\mathbf{z},\theta)]. \tag{2.26}$$

Here we let $\psi(\mathbf{z},\{\mu,\sigma\}) = \exp(\mu + \sigma^{1/2}\mathbf{z})$ and $\mathbf{z} \sim N(0,1)$.

Given the updated variational parameters μ_{dk}, and σ_{dk} we can approximate $p(W,\sigma|m_{1O},...,m_{NO})$ by drawing the model parameters from their variational posterior. For instance, for W_{dk} we draw,

$$z_{dk} \sim N(0,1), \tag{2.27}$$

and then set

$$W_{dk} = \mu_{W_{dk}} + z_{dk}\sigma_{W_{dk}}^{\frac{1}{2}}. \tag{2.28}$$

The choice of step size for the variational parameter updates is critical to the runtime of the algorithm. Here, we use the AdaDelta Method to ensure fast convergence that is mostly unaffected by the initial choice of parameters. The idea behind AdaDelta is to take a step size proportional to the ratio of the previous $w-1$ step sizes to the previous w gradients. This helps to ensure that the parameters do not oscillate about the optimum because the gradients are so large that the optimum is overshot. It also helps ensure that the algorithm does not search too slowly in a region where the gradient is small. The AdaDelta method removes the need to artificially construct a step size schedule and is insensitive to large gradients and noise [31].

2.0.2 Stochastic Gradient Langevin Dynamics Method

Let $x = (W_{11},...,W_{DK},r_1,...,r_K,\sigma,\sigma_0,c_0)$. We update our model parameters as described in Eq. 4.1,

$$\Delta x_t = \frac{\varepsilon_t}{2}\left(\nabla \log \pi(x_t) + \sum \nabla \log \pi(m_{1O},...,m_{NO}|x_t)\right) + \eta_t, \tag{2.29}$$

where $\eta_t \sim N(0,\varepsilon_t)$, ε_t decays as $t^{-1/3}$, and $\log(\pi(x_t))$ and $\log\pi(D_i|x_t)$ are computed using Eq. 2.9. As is discussed in the previous chapter, for t big enough, updating the parameters is the same as drawing from the posterior distribution. Thus by updating σ and W we can sample from

$$p(W,\sigma|m_{1O},...,m_{NO}) \tag{2.30}$$

Since each entry of x must be nonnegative we do the following transformation to avoid a constrained optimisation. Let u denote our unconstrained variable and define the transformation

Algorithm 5 Marsden and Bacallado Algorithm with SVI Posterior Estimation

Initialize N, P, some upper bound on the rank, K, some upper bound on the variance, σ, and the discount factor $\beta \in [0,1]$.
Sample $K(N+D-K)$ entries uniformly with replacement
repeat
    Construct the ELBO using only the observed entries, i.e. from Eq. 2.9 and Eq. 2.24
    Approximate the variance of $WW^T + \sigma^2 I$ from the yet-to-be-updated variational posterior using MCMC to determine number of steps for Variational Inference, N_{steps} and number of observations to be made, N_{Obs}.
    for i=1, ..., N_{steps} **do**
        Update ELBO and variational parameters
    end for
    for n=1, ..., N **do**
        Compute $p(m_{nU}|m_{nO}, W, \sigma)$
    end for
    for i=1, ..., N_{obs} **do**
        For each n, draw $m_n \sim p(m_{nU}|m_{nO}, W, \sigma)p(W, \sigma|m_{1O}, ..., m_{NO})$
        Incorporate discount factor for each entry: $m_{nk} \leftarrow m_{nk}\beta^{N_{m_{nk}}}$, where $N_{m_{nk}}$ is the number of times entry (n,k) has been observed
        Observe the entry with largest corresponding reward
    end for
until Matrix can be solved with convex optimisation method

T as,

$$x = T(u) = \log(1+e^u). \tag{2.31}$$

We update our unconstrained variable u by computing

$$\Delta u_t = \left(\frac{\varepsilon_t}{2}\left(\nabla_x \log\pi(T(u_t)) + \sum \nabla_x \log\pi(D_i|T(u_t))\right)\right)T'(u_t) \tag{2.32}$$

We update the unconstrained variable by taking step $u_{t+1} = u_t + \Delta u_t$ and then set $x_{t+1} = T(u_{t+1}) + |\eta_t|$.

Note that in practice we now add the absolute value of the noise after applying the transformation to our unconstrained variable. Furthermore, from [21], it seems that, in practice, it is better to let t be fixed rather than decay. With a decaying t a Metropolis-Hastings step to correct for discretization becomes unnecessary, however with a fixed t we would technically need to do this and we instead omit it. Finally, we add noise with variance to match the scale of the data so that it doesn't dominate the noise in the model, but also so that it is not so small it is completely negligible.

Algorithm 6 Marsden and Bacallado Algorithm with SGLD Posterior Sampling

Initialize N, P, some upper bound on the rank, K, some upper bound on the variance, σ, and the discount factor $\beta \in [0,1]$.
Sample $K(N+D-K)$ entries uniformly with replacement
repeat
    Compute $\log p(m_{1O},...,m_{NO},W,r,\gamma,\gamma_0,c_0,\sigma)$ from Eq. 2.9
    for i=1, ..., N_{steps} **do**
        For each model parameter:$W,r,\gamma,\gamma_0,c_0,\sigma$ compute the corresponding unconstrained update from Eq. 2.32
        Set $W_{t+1} = T(uW_{t+1}) + |\eta_{W_t}|$, $r_{t+1} = T(ur_{t+1}) + |\eta_{r_t}|$, etc.
    end for
    for i=1, ..., N_{obs} **do**
        For each n, draw $m_n \sim p(m_{nU}|m_{nO},W,\sigma)p(W,\sigma|m_{1O},...,m_{NO})$
        Incorporate discount factor for each entry: $m_{nk} \leftarrow m_{nk}\beta^{N_{m_{nk}}}$, where $N_{m_{nk}}$ is the number of times entry (n,k) has been observed
        Observe the entry with largest corresponding reward
    end for
until Matrix can be solved with convex optimisation method

2.0.3 Singular Value Thresholding Method

Algorithm 7 SVT Comparison Method

Initialize N, P, some upper bound on the rank, K, some upper bound on the variance, σ, and the discount factor $\beta \in [0,1]$.
Sample $K(N+D-K)$ entries uniformly with replacement
repeat
    Compute estimate $\hat{M}$ of the true matrix M using nuclear norm minimisation
    Incorporate discount factor for each entry: $m_{nk} \leftarrow m_{nk}\beta^{N_{m_{nk}}}$, where $N_{m_{nk}}$ is the number of times entry (n,k) has been observed
    Observe the entry with largest corresponding reward
until Matrix can be solved with convex optimisation method

As a baseline comparison we use a singular value thresholding method. At each step we simply use our set of observed entries to construct an estimate of the true matrix using nuclear norm minimisation as explained in the introduction. Given this estimation we greedily pick the N_{Obs} largest predicted entries. The idea is that exploration is achieved in the beginning when the estimate has a large amount of error, and exploitation is achieved as soon as the estimation becomes more accurate.

Chapter 3

Results

For the Thompson sampling simulations we use Theano [2] in conjunction with Lasagne, a small library build to train neural networks in Theano, to compute and update the parameters and hyperparameters of our model. We run our code on an NVIDIA Tesla K20 graphic card. Below is a profile of the runtime using a CPU and using a GPU, showing the benefits of using a GPU. For the singular value thresholding comparison method we use the MATLAB software written by Emmanuel Candés and Stephen Becker [6].

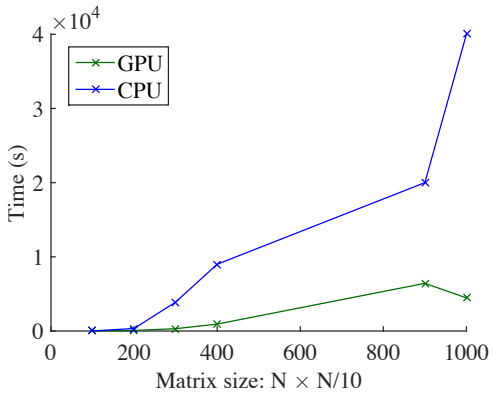


Figure 3.1 Runtime speed-up of the SVI Thompson sampling algorithm using the NVIDIA Tesla K20 graphics card.

3.1 Posterior Estimation Tests

We first test that both SVI and SGLD are able to effectively estimate the posterior distribution when given the full data matrix M without any missing entries. To see the ability of each

algorithm to adapt to different true ranks of the covariance matrix we test a 100×100 matrix with true ranks $1, 30, 100$ and user-defined rank threshold $K = 30$.

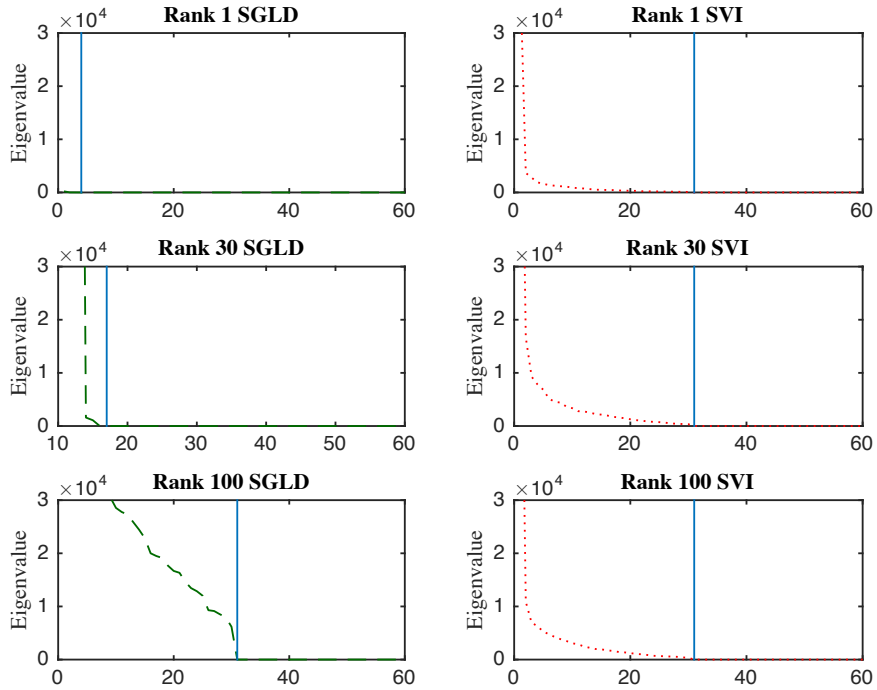


Figure 3.2 Plot of the eigenvalues of the estimated covariance matrix. Left: SGLD Method. Right: SVI Method.

We give the algorithm the full data set and let it update the model parameters as defined in the SGLD pseudocode and SVI pseudocode. We plot the eigenvalues of the estimated covariance matrix for each rank and each method. For the SGLD we see a clear response to the true rank of the data. For the SVI method we also see a response to the true rank of the data since the eigenvalues decay much faster for the rank 1 matrix, however not quite as obvious as that for SGLD. Below we plot the convergence of the log-likelihood when the parameters are updating with SVI with different true ranks. We see that convergence occurs only a few steps into the algorithm, which is generally not the case. This might suggest that the SVI parameters are quickly trapped in a local minimum or possibly that the mean-field approximation makes the centre of the variational posterior very inaccurate.

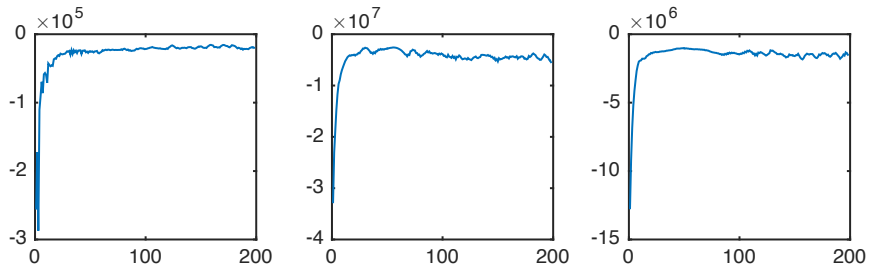


Figure 3.3 Evolution of log-likelihood as parameters are updated with SVI. Left: True Rank 1, Middle: True Rank 30, Right: True Rank 100

3.2 Bandit Results with Synthetic Data

To test the performance of each algorithm we construct a $D \times K$ factor matrix W, for ranks $K = 1, 5, 10, 15, 20$, and $D = 100$. We draw $N = 1000$ columns from a multivariate normal with covariance matrix $WW^T + 0.1I$. Due to slow computation time we are only able to average each algorithm over 5 different runs. We measure the "regret" in that we construct the optimal sequence of actions knowing the full true matrix and subtract the cumulative reward from the cumulative reward achieved by the policy of each algorithm. For these Bayesian methods we set the user-defined threshold rank to be $\bar{K} = 20$. In the analysis we also include the "Oracle" results where the true covariance matrix is known and at each step the entries are chosen by drawing from the conditional multivariate normal given the set of observations. Every test run considers the horizon $H = \bar{K}(N + D - \bar{K})$. This is the number of degrees of freedom for a matrix with rank $\bar{K}$. In general most methods can complete an $N \times D$ matrix of rank $\bar{K}$ within a factor of this horizon [7]. For the SVI and SVT algorithms we either update the posterior distribution or perform singular value thresholding 20 times and observe $N_{Obs} = \lfloor H/20 \rfloor$ entries before updating again. For the SGLD algorithm we iterate 200 times since drawing from the posterior requires the stochastic gradient iteration.

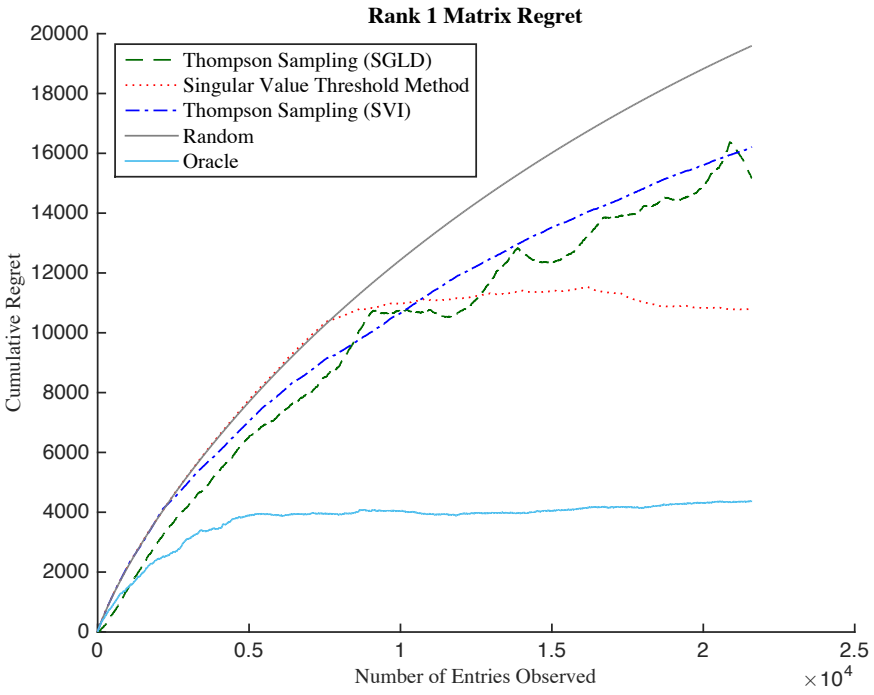


Figure 3.4

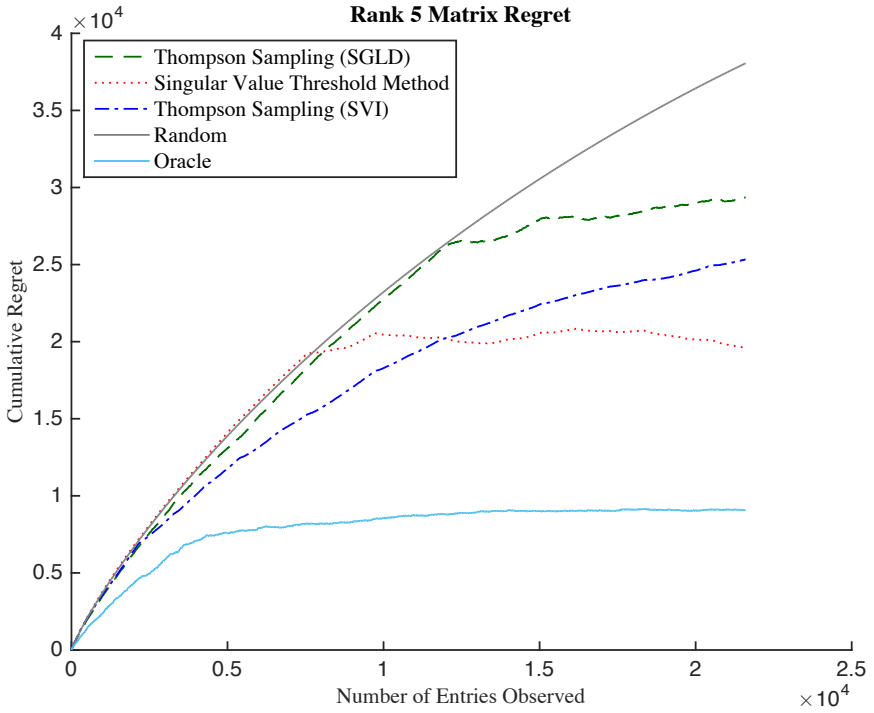


Figure 3.5

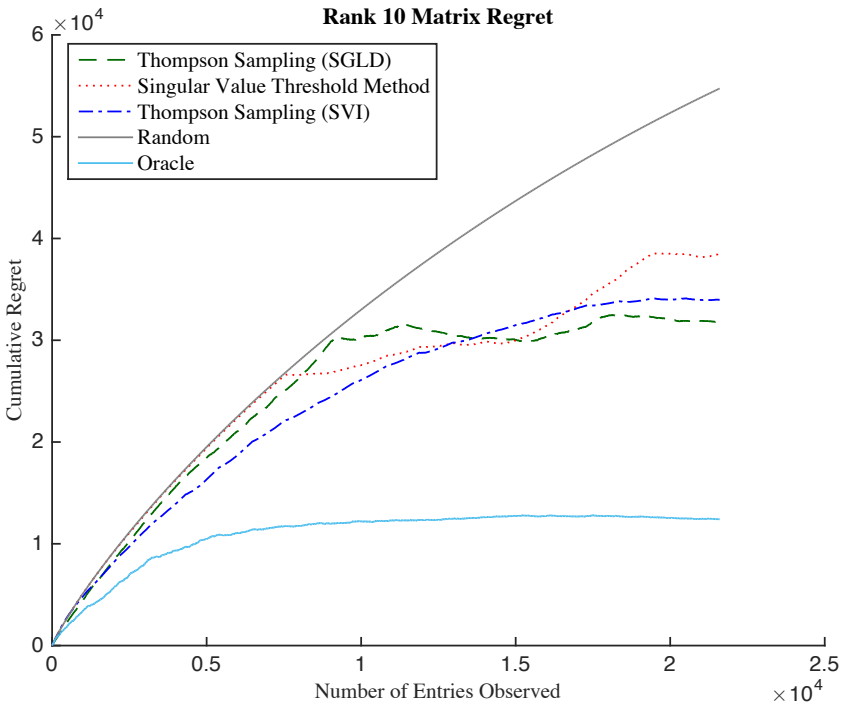


Figure 3.6

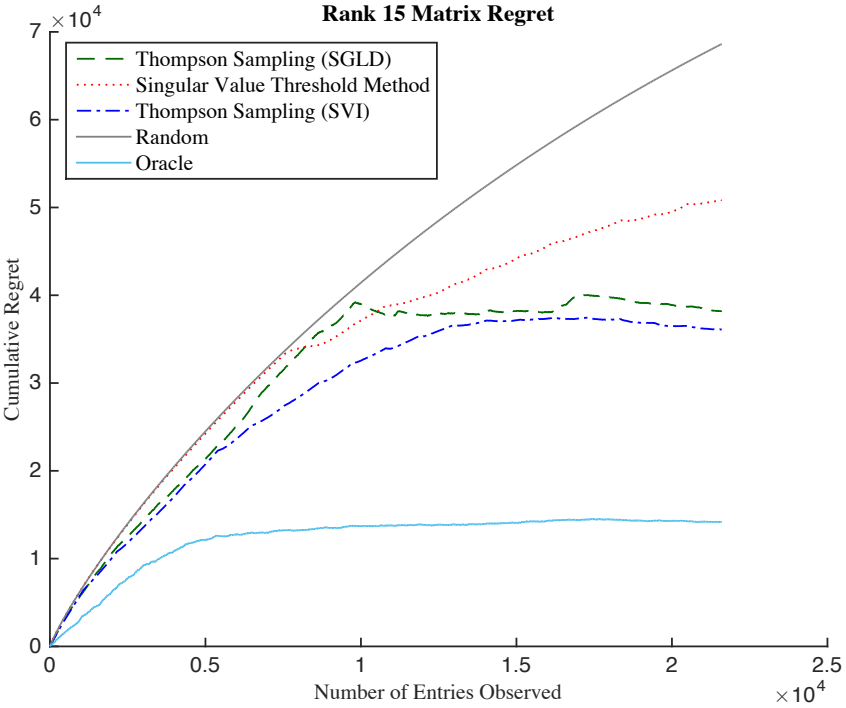


Figure 3.7

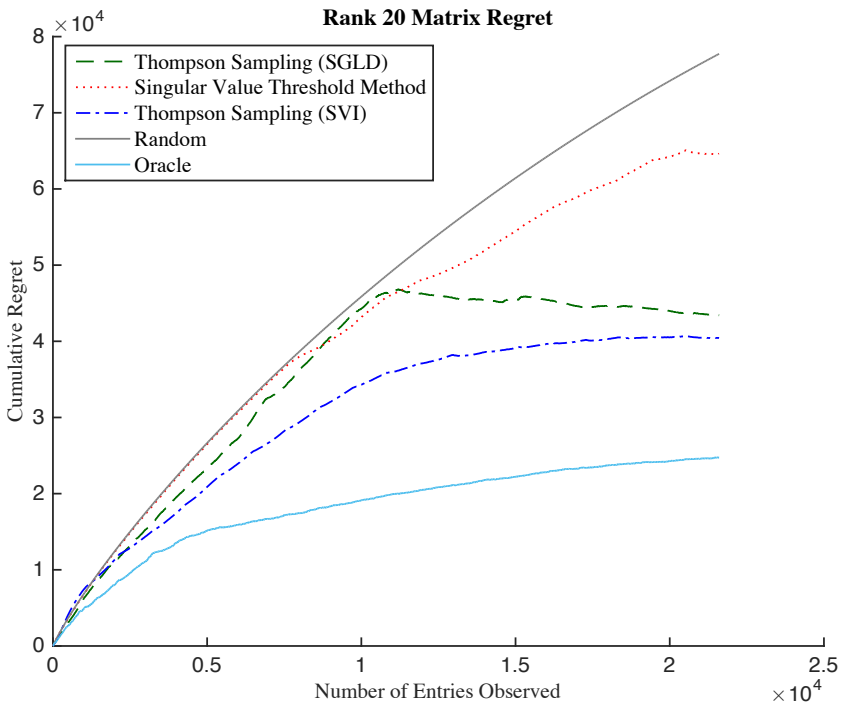


Figure 3.8

When the rank is low we see that the Thompson sampling methods have faster growing regret than the singular value thresholding method. The regret of the singular value thresholding method increases at nearly the same rate as the Thompson sampling methods, but quite soon levels off. This indicates that the method is able to complete the entire matrix with good accuracy. However the next plots show that as the true rank of the data grows, the Thompson sampling methods begin to outperform SVT. The transition from increasing regret to leveled regret is sharper for the SGLD method than the SVI method.

To analyse the exploration of each method we show a sequence of heatplots in which each sequential frame shows the next 500 observations. To see if there is an effect of the true rank of the data on the exploration we show the plots for rank 1, however the behavior is generally the same for ranks 5, 10, 15, and 20.

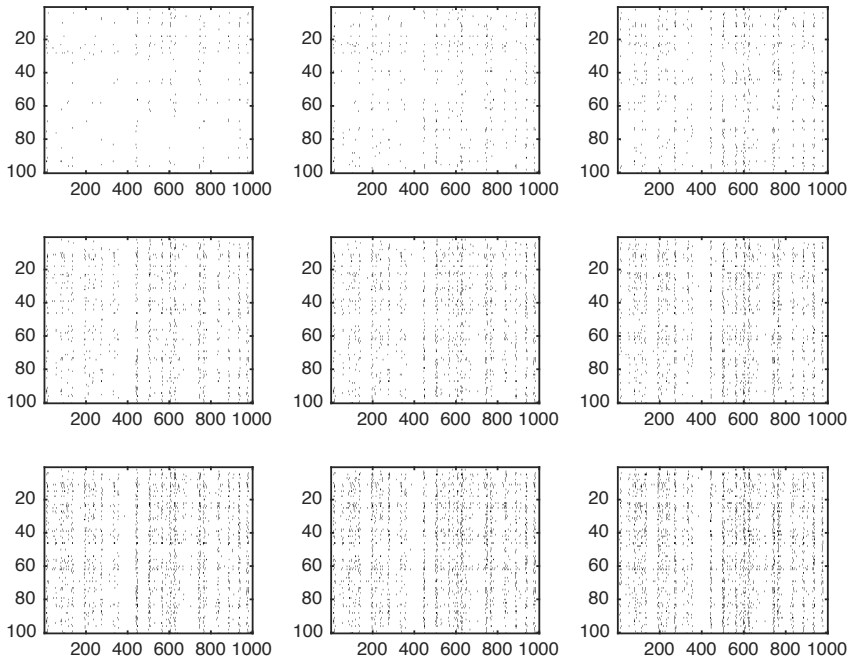


Figure 3.9 Sequence of observations with Oracle policy. True Rank: 1.

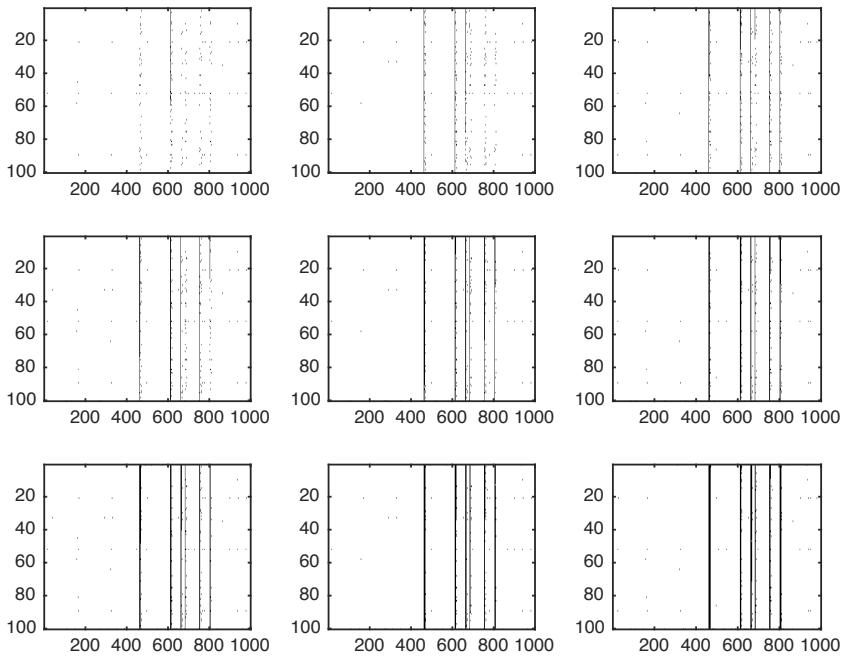


Figure 3.10 Sequence of observations with SGLD Thompson sampling policy. True Rank: 1.

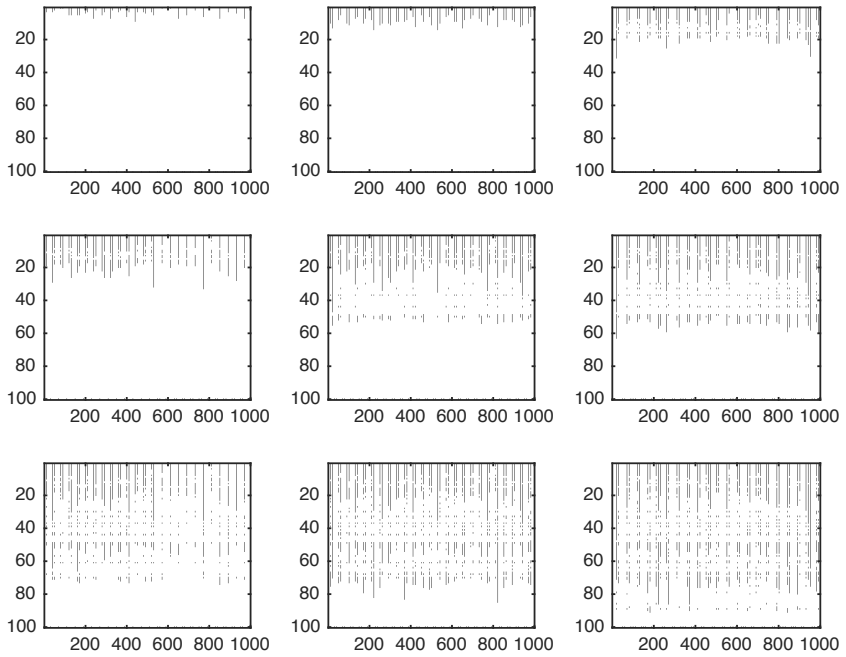


Figure 3.11 Sequence of observations with SVI Thompson sampling policy. True Rank: 1.

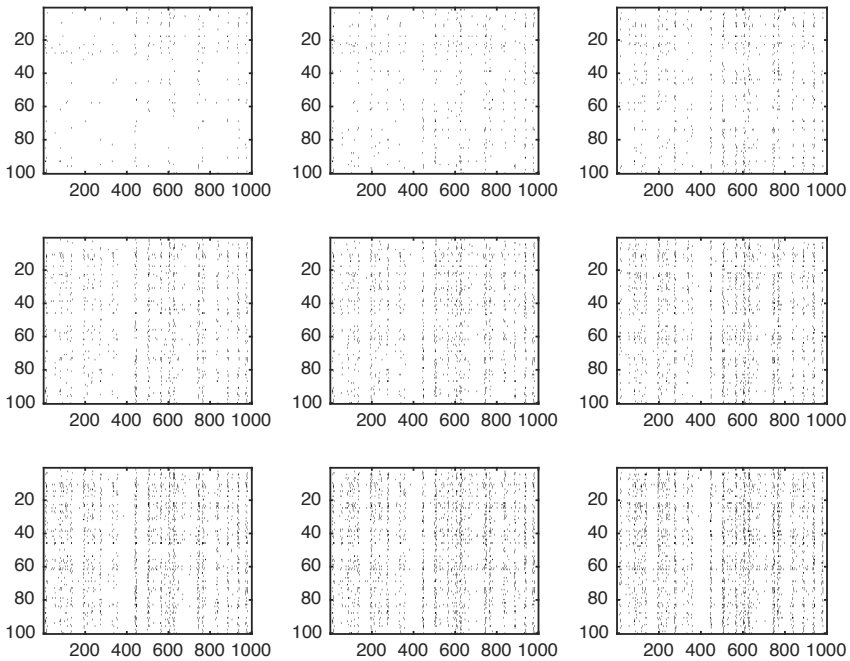


Figure 3.12 Sequence of observations with SVT policy. True Rank: 1.

One obvious observation is the tendency towards exploitation rather than exploration on part of the Thompson sampling algorithms, and in particular the SGLD algorithm. Both

methods seem to observe entries in a highly non-random fashion, even towards the very beginning when the posterior distributions should not be too concentrated. While the regret of the methods changes in response to increasing rank, the exploration pattern of the methods seems to be unaffected.

3.3 Bandit Results with MovieLens Data

We test the algorithms on a 1000×100 matrix from the MovieLens dataset [1]. To complete the missing entries initially we use OptSpace [18], an algorithm known to perform well on the MovieLens data set [17]. Thus we ultimately use a modified version of the MovieLens data so that we can compute the regret of each algorithm. Again, we only consider the horizon $K(N+D-K)$, where $N = 1000$, $D = 100$, and we let $K = 20$.

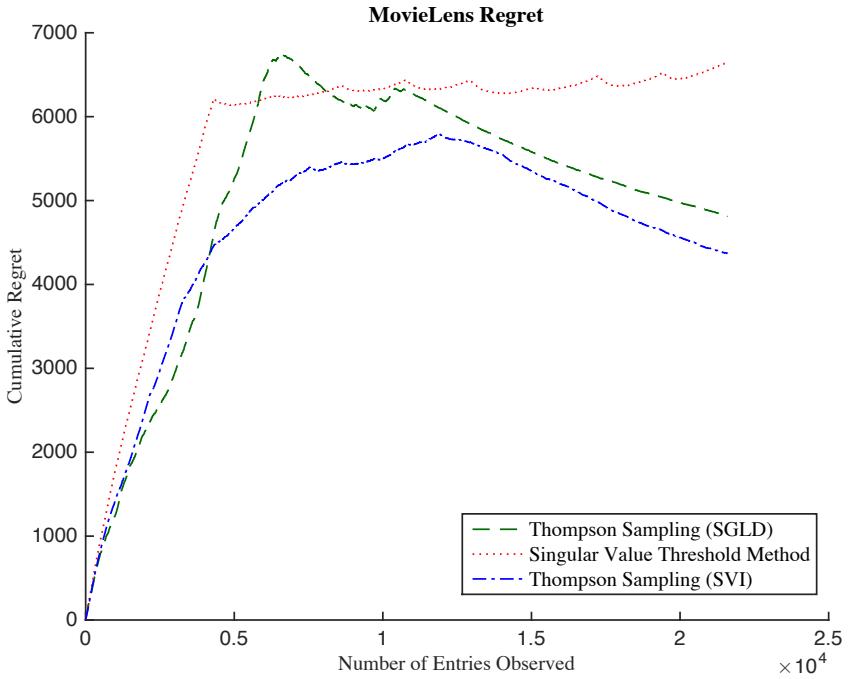


Figure 3.13 Cumulated Regret with the MovieLens data set.

The data shows that the Thompson sampling methods outperform the singular value thresholding method for this data set. The performance is particularly good when the posterior distribution is updated using stochastic variational inference. Also note that within this small horizon the regret of the Thompson sampling methods begins to decrease whereas the singular value thresholding method stays level. This may indicate that even after SVT

has levelled off, the Bayesian methods have a more accurate estimate of the true matrix since they are able to account for the large entries not previously chosen.

Chapter 4

Conclusion

This chapter consists of two parts. The first is a discussion of the results and suggestions for future research. The second is a more rigorous proposal for a new bandit policy.

4.1 Discussion of Results

Both the synthetic data results and the MovieLens results show superior performance of the Thompson sampling algorithms, particularly the SVI algorithm, for larger rank data. The poor performance of the Thompson sampling algorithms in the rank 1 and rank 5 case may be due to the fact that the rank threshold value is too large compared to the true rank. However, from the initial rank thresholding tests it seems that both the SGLD and SVI algorithms are able to adapt to small ranks. Thus we see conflicting behavior. In the matrix completion setting with a full data set these methods have shown to be independent of the user-defined rank threshold value. However, in the sequential matrix completion setting this does not seem to be the case. It is possible that in the sequential matrix completion case it requires more observations to be able to deviate further from the user-defined rank threshold value. In future research, further testing of this behavior is necessary.

The heatmaps show interesting behavior on the part of the Thompson sampling algorithms. We see that the variance of the model parameters is too low to allow for efficient exploration. This is particularly obvious with the SGLD method. While SGLD is effective in reconstructing the true matrix, it is unable to accurately quantify the uncertainty. The reason for this most likely lies in the way the algorithm was modified to make it faster. Recall that in the stochastic gradient Langevin method, at each time t we update the parameters of the

posterior distribution, x, by taking the step

$$\Delta x_t = \frac{\varepsilon_t}{2}\left(\nabla \log \pi(x_t) + \sum \nabla \log \pi(D_i|x_t)\right) + \eta_t, \tag{4.1}$$

where $\eta_t \sim N(0, \varepsilon_t)$ and ε_t decays in such a way that,

$$\sum_{t=0}^{\infty} \varepsilon_t = \infty \text{ and } \sum_{t=0}^{\infty} \varepsilon_t^2 < \infty.$$

For smaller values of t we have the stochastic gradient method which updates the parameters closer to the maximum a posteriori estimate. Welling and Teh show that Eq 4.1 converges to the posterior distribution as $t \to \infty$. However, in the case of sequential matrix completion we have an iterative process where at each step we must re-estimate the posterior distribution given the new observations. In order to make the algorithm not too slow we cannot explore the behavior for very large t. Thus we are able to update the parameters close to the MAP estimate, which explains why the regret of the SGLD method is low. However, we are not yet able to fully converge to the true posterior distribution, which may explain why the algorithm is not an efficient explorer.

The reason that the SVI algorithm does not exhibit much exploration may lie in the technicalities of the parameter updates. When the gradients are very large, in order to stay in the computable regime, the gradients must be scaled. For the SVI algorithm we used the well-tested and easily computed AdaDelta method [31]. The idea behind AdaDelta is to take a step size proportional to the ratio of the previous s step sizes to the previous $s+1$ gradients, for some s. This avoids two common pitfalls. The first is oscillating about the optimum because the gradients are so large that the optimum is overshot. The second is searching slowly in a region where the gradient is small. The AdaDelta method removes the need to artificially construct a step size schedule and is insensitive to large gradients and noise [31]. However, the scaling of the AdaDelta gradients essentially ensures that the gradients, and thus the parameter updates, stay on the same scale and within a reasonable range. Perhaps the variational parameters that describe the variance of the posterior distribution are kept within too small a range from the AdaDelta method. Again, research into this behavior is necessary in the future.

4.2 A new proposal

The primitive singular value thresholding algorithm had impressive performance for the rank 1 and 5 cases. Furthermore it was much faster than the Thompson sampling algorithms

and did not require to be run on a GPU. This makes it an appealing method to use in the future. However, nowhere does the SVT algorithm make use of the fact that the columns are drawn from a multivariate normal with a low rank covariance matrix. This leads to the idea of using an Empirical Bayes approach. That is, perhaps it is more efficient to use SVT to complete the empirical covariance matrix and then using this low rank approximation for the true covariance matrix, draw the unobserved entries given the observed entries from the conditional multivariate. This uses the power of singular value thresholding and incorporates the knowledge of the model. This method can be cast as a modified matrix Lasso estimator. For a full set of observations, the estimator would be

$$\hat{\Sigma}_{LASSO} = \underset{S \in S_+^D}{\operatorname{argmin}} ||\bar{\Sigma} - S||_2^2 + \lambda ||S||_*$$

where S_+^D denotes the set of DxD symmetric positive-semidefinite matrices, $\bar{\Sigma}$ denotes the empirical covariance matrix, and $||\cdot||_*$ again denotes the nuclear norm. In the case of only partial observations we construct one of two possible empirical "covariance matrices". Let $(z)_{ij}$ be a mask matrix where $z_{ij} \sim \text{Bernoulli}(\delta_{ij})$. If δ_{ij} is constant for all entries, i.e. $\forall i, j,\ \delta_{ij} = \delta$ then we can construct the empirical covariance matrix

$$\Sigma'_\delta = \frac{1}{N} \sum_{n=1}^{N} y_n \otimes y_n.$$

However if we choose to sample the covariance matrix in a specific manner it may not be the case that δ_{ij} is constant. In this scenario we cannot simply divide each entry in the empirical covariance matrix by N since it is possible that some entries are observed much more often than others. Here we let

$$f_{ij} = \begin{cases} 1, \text{if } \forall n \in N, y_{ni} y_{nj} = 0 \\ s_{ij}, \text{ else } \left(\text{where } s_{ij} = \left|\left\{n | y_{ni} y_{nj} \neq 0\right\}\right|\right). \end{cases}$$

Then we construct the empirical covariance matrix

$$\Sigma' = \frac{1}{f} .* \sum_{n=1}^{N} y_n \otimes y_n,$$

where $.*$ is meant to denote element-wise multiplication. We assume for now that the entries are indeed chosen randomly within our data matrix Y so that the first empirical covariance matrix estimator is legitimate. Following the work of [20] we construct a new formulation of

the matrix Lasso with only partial observations. It is an easy calculation to see that

$$\mathbb{E}(\Sigma'_\delta) = (\delta - \delta^2)\text{diag}(\Sigma) + \delta^2\Sigma,$$

where Σ simply represents the true covariance matrix. To construct an unbiased estimator of the empirical covariance matrix we take

$$\tilde{\Sigma}_\delta = (\delta^{-1} - \delta^{-2})\text{diag}(\Sigma'_\delta) + \delta^{-2}\Sigma'_\delta.$$

To see that this is an unbiased estimator note that,

$$\Sigma = \delta^{-2}\Sigma'_\delta + (\delta^{-1} - \delta^{-2})\text{diag}(\Sigma'_\delta).$$

Thus the modified matrix Lasso estimator we define to be the following

$$\hat{\Sigma}_{LASSO-PARTIAL} = \underset{S \in S_+^D}{\text{argmin}} ||\tilde{\Sigma}_\delta - S||_2^2 + \lambda ||S||_*. \tag{4.2}$$

Lounici [20] shows that this estimator is minimax optimal up to a logarithmic factor. Interestingly, this estimator has a Bayesian interpretation as a maximum a posteriori (MAP) estimate under the following prior,

$$\begin{aligned} y_n &\sim N(0, \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I}), \\ W_{dk} &\sim N(0, 1/\sqrt{2\lambda}), \end{aligned} \tag{4.3}$$

where we treat λ and σ as fixed. In the case of the full data set, i.e. no partial observations, we have the log joint as

$$\log p(y_1, ..., y_N, W, r, \gamma, \gamma_0, c_0) = \left(-\frac{N}{2}\log|S| - \frac{1}{2}\sum_{n=1}^{N} y_n^T S^{-1} y_n \right) - \frac{\lambda}{2}||W||_2^2,$$

where $S = WW^T + \sigma^2\mathbf{I}$. Aside from calculating the full posterior distribution, if we want to find the maximum a posteriori we can simply maximize the log joint since it is proportional to the posterior distribution. So the maximum a posteriori estimator, $\hat{\Sigma}_{MAP}$, satisfies,

$$\hat{\Sigma}_{MAP} = \underset{S = WW^T + \sigma^2 I}{\text{argmax}} \; -\frac{N}{2}\log|S| - \frac{1}{2}\sum_{n=1}^{N} y_n^T S^{-1} y_n - \frac{\lambda}{2}||W||_2^2. \tag{4.4}$$

.

The relationship between this maximum a posteriori estimator and the matrix Lasso estimator follows from two facts. The first is that that the empirical covariance matrix $\bar{\Sigma}$ satisfies,

$$\bar{\Sigma} = \underset{S \in S_+^D}{\operatorname{argmax}} \left(-\frac{N}{2}\log|S| - \frac{1}{2}\sum_{n=1}^{N} y_n^T S^{-1} y_n \right)$$

.

And so we have,

$$\underset{S \in S_+^D}{\operatorname{argmax}} \left(-\frac{N}{2}\log|S| - \frac{1}{2}\sum_{n=1}^{N} y_n^T S^{-1} y_n \right) = \underset{S \in S_+^D}{\operatorname{argmin}} ||S - \bar{\Sigma}||_2^2.$$

The second fact, as shown in [26], is that if X can be decomposed as UV^T, then

$$||X||_* = \min_{UV^T = X} ||U||_2 ||V||_2,$$

thus if we restrict $S = WW^T$, then $||W||_2^2 = ||S||_*$. We use these two facts to recast our maximum a posteriori estimator as

$$\hat{\Sigma}_{MAP} = \underset{S \in S_+^D}{\operatorname{argmin}} ||S - \bar{\Sigma}||_2^2 - \lambda ||S||_* = \hat{\Sigma}_{LASSO} \tag{4.5}$$

Now in the case of partial data, this equality is no longer necessarily true. Using the notation from the methods section, the logarithm of the joint distribution is

$$\begin{aligned} \log p(y_{1O}, ..., y_{NO}, W, r, \gamma, \gamma_0, c_0) = \sum_n \left(-\frac{1}{2}\log|\bar{\Sigma}_{nOO}| - \frac{1}{2} y_{nO}^T (\bar{\Sigma}_{nOO})^{-1} y_{nO}) \right) \\ + \frac{1}{2}\log(\lambda) - \lambda ||W||_2^2. \end{aligned}$$

Thus the MAP estimate in the setting of partial data satisfies

$$\hat{\Sigma}_{MAP-PARTIAL} = \underset{S = WW^T + \sigma^2 I}{\operatorname{argmax}} -\frac{N}{2}\log|S_{nOO}| - \frac{1}{2}\sum_{n=1}^{N} y_n^T S_{nOO}^{-1} y_n - \frac{\lambda}{2}||S||_*.$$

Whereas, again, the modified matrix LASSO estimator is

$$\hat{\Sigma}_{LASSO-PARTIAL} = \underset{S \in S_+^D}{\operatorname{argmin}} ||\tilde{\Sigma}_\delta - S||_2^2 + \lambda ||S||_*.$$

The relationship between these two estimators has not yet been studied. Future work might consider studying the relationship between these two estimators and a comparison of simulations of their corresponding regret when used in the recommender system problem.

Bibliography

[1] Movielens. http://grouplens.org/datasets/movielens/. Accessed: 2016-08-01.

[2] Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., Bastien, F., Bayer, J., Belikov, A., Belopolsky, A., Bengio, Y., Bergeron, A., Bergstra, J., Bisson, V., Bleecher Snyder, J., Bouchard, N., Boulanger-Lewandowski, N., Bouthillier, X., de Brébisson, A., Breuleux, O., Carrier, P.-L., Cho, K., Chorowski, J., Christiano, P., Cooijmans, T., Côté, M.-A., Côté, M., Courville, A., Dauphin, Y. N., Delalleau, O., Demouth, J., Desjardins, G., Dieleman, S., Dinh, L., Ducoffe, M., Dumoulin, V., Ebrahimi Kahou, S., Erhan, D., Fan, Z., Firat, O., Germain, M., Glorot, X., Goodfellow, I., Graham, M., Gulcehre, C., Hamel, P., Harlouchet, I., Heng, J.-P., Hidasi, B., Honari, S., Jain, A., Jean, S., Jia, K., Korobov, M., Kulkarni, V., Lamb, A., Lamblin, P., Larsen, E., Laurent, C., Lee, S., Lefrancois, S., Lemieux, S., Léonard, N., Lin, Z., Livezey, J. A., Lorenz, C., Lowin, J., Ma, Q., Manzagol, P.-A., Mastropietro, O., McGibbon, R. T., Memisevic, R., van Merriënboer, B., Michalski, V., Mirza, M., Orlandi, A., Pal, C., Pascanu, R., Pezeshki, M., Raffel, C., Renshaw, D., Rocklin, M., Romero, A., Roth, M., Sadowski, P., Salvatier, J., Savard, F., Schlüter, J., Schulman, J., Schwartz, G., Serban, I. V., Serdyuk, D., Shabanian, S., Simon, E., Spieckermann, S., Subramanyam, S. R., Sygnowski, J., Tanguay, J., van Tulder, G., Turian, J., Urban, S., Vincent, P., Visin, F., de Vries, H., Warde-Farley, D., Webb, D. J., Willson, M., Xu, K., Xue, L., Yao, L., Zhang, S., and Zhang, Y. (2016). Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688.

[3] Bhattacharya, A. and Dunson, D. B. (2011). Sparse bayesian infinite factor models. *Biometrika*, 98(2):291–306.

[4] Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2016). Variational inference: A review for statisticians. *arXiv preprint arXiv:1601.00670*.

[5] Bubeck, S. and Cesa-Bianchi, N. (2012). Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *arXiv preprint arXiv:1204.5721*.

[6] Cai, J.-F., Candès, E. J., and Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982.

[7] Candes, E. J. and Plan, Y. (2010). Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936.

[8] Candès, E. J. and Recht, B. (2009a). Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772.

[9] Candès, E. J. and Recht, B. (2009b). Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772.

[10] Candès, E. J. and Tao, T. (2010). The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080.

[11] Doucet, A. (2010). A note on efficient conditional simulation of gaussian distributions. *Departments of Computer Science and Statistics, University of British Columbia*.

[12] Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2014). *Bayesian Data Analysis*, volume 2. Taylor & Francis.

[13] Gershman, S. J. and Blei, D. M. (2012). A tutorial on bayesian nonparametric models. *Journal of Mathematical Psychology*, 56(1):1–12.

[14] Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347.

[15] Kaufmann, E., Cappé, O., and Garivier, A. (2012). On bayesian upper confidence bounds for bandit problems. In *AISTATS*, pages 592–600.

[16] Kent, J. (1978). Time-reversible diffusions. *Advances in Applied Probability*, pages 819–835.

[17] Keshavan, R. H., Montanari, A., and Oh, S. (2009a). Low-rank matrix completion with noisy observations: a quantitative comparison. In *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*, pages 1216–1222. IEEE.

[18] Keshavan, R. H., Oh, S., and Montanari, A. (2009b). Matrix completion from a few entries. In *2009 IEEE International Symposium on Information Theory*, pages 324–328. IEEE.

[19] Knowles, D. A. (2015). Stochastic gradient variational bayes for gamma approximating distributions. *arXiv preprint arXiv:1509.01631*.

[20] Lounici, K. et al. (2014). High-dimensional covariance matrix estimation with missing observations. *Bernoulli*, 20(3):1029–1058.

[21] Mattingly, J. C., Stuart, A. M., and Higham, D. J. (2002). Ergodicity for sdes and approximations: locally lipschitz vector fields and degenerate noise. *Stochastic processes and their applications*, 101(2):185–232.

[22] Paisley, J., Blei, D., and Jordan, M. (2012). Variational bayesian inference with stochastic search. *arXiv preprint arXiv:1206.6430*.

[23] Recht, B. (2011). A simpler approach to matrix completion. *Journal of Machine Learning Research*, 12(Dec):3413–3430.

[24] Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

[25] Russo, D. and Van Roy, B. (2014). Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243.

[26] Srebro, N., Rennie, J., and Jaakkola, T. S. (2004). Maximum-margin matrix factorization. In *Advances in neural information processing systems*, pages 1329–1336.

[27] Teh, Y. W., Görür, D., and Ghahramani, Z. (2007). Stick-breaking construction for the indian buffet process. In *International Conference on Artificial Intelligence and Statistics*, pages 556–563.

[28] Thibaux, R. and Jordan, M. I. (2007). Hierarchical beta processes and the indian buffet process. In *International conference on artificial intelligence and statistics*, pages 564–571.

[29] Weber, R. et al. (1992). On the gittins index for multiarmed bandits. *The Annals of Applied Probability*, 2(4):1024–1033.

[30] Welling, M. and Teh, Y. W. (2011). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688.

[31] Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

[32] Zhou, M., Wang, C., Chen, M., Paisley, J., Dunson, D., and Carin, L. (2010). Nonparametric bayesian matrix completion. *Proc. IEEE SAM*, 2:12.

Appendix A

Proof of the analytical expression for the optimal ELBO update

A.1 The KL Divergence is nonnegative

We use Jensen's inequality and the fact that logarithm is concave:

$$\begin{aligned} -\mathbb{KL}(f||g) &= \mathbb{E}_f[\log\frac{g(x)}{f(x)}] \\ &\leq \log\mathbb{E}_f[\frac{g(x)}{f(x)}] \\ &= \log\int_\chi g(x)dx = 0 \end{aligned}$$

And note that the integral is zero if $\log\frac{p(x)}{q(x)} = 0$ almost everywhere, which is equivalent to $f = g$ almost everywhere.

A.2 The Mean Field Method Update Equation

$$
\begin{aligned}
L(q) &= \int_{\chi} \prod q_i(x_i) \Big[\log p(x,D) - \sum_k \log q_k(x_k) \Big] dx \\
&= \int_{\chi_j} q_j(x_j) \int_{\chi_{-j}} \prod_{i \neq j} q_i(x_i) \big[\log p(x,D) - \sum_k \log q_k(x_k) \big] dx \\
&= \int_{\chi_j} q_j(x_j) \int_{\chi_{-j}} \prod_{i \neq j} q_i(x_i) \log p(x,D) dx \\
&- \int_{\chi_j} q_j(x_j) \int_{\chi_{-j}} \prod_{i \neq j} q_i(x_i) \big[\sum_{k \neq j} \log q_k(x_k) + \log q_j(x_j) \big] dx \\
&= \int_{\chi_j} q_j(x_j) \mathbb{E}_{q_{-j}} \big[\log p(x,D) \big] - \int_{\chi_j} q_j(x_j) \log q_j(x_j) \Big[\int_{\chi_{-j}} \prod_{i \neq j} q_i(x_i) \Big] dx \\
&- \sum_{k \neq j} \int_{\chi_j} q_j(x_j) \int_{\chi_{-j}} \prod_{i \neq j} q_i(x_i) \log q_k(x_k) \\
&= \int_{\chi_j} q_j(x_j) \mathbb{E}_{q_{-j}} \big[\log p(x,D) \big] - \int_{\chi_j} q_j(x_j) \log q_j(x_j) dx - \sum_{k \neq j} \int_{\chi_{-j}} \prod_{i \neq j} q_i(x_i) \log q_k(x_k),
\end{aligned}
$$

where the last equality holds because $\int_{\chi_{-j}} \prod_{i \neq j} q_i(x_i) = 1$ and $\int_{\chi_j} q_j(x_j) = 1$, since each $q_i(x_i)$ is a probability density on its own, and any combination of them is a probability density. Then looking at the ELBO in terms of just $q_j(x_j)$ we have that,

$$
\begin{aligned}
L(q) &= \int_{\chi_j} q_j(x_j) \mathbb{E}_{q_{-j}} \big[\log p(x,D) \big] - \int_{\chi_j} q_j(x_j) \log q_j(x_j) dx - \sum_{k \neq j} \int_{\chi_{-j}} \prod_{i \neq j} q_i(x_i) \log q_k(x_k) \\
&= \int_{\chi_j} q_j(x_j) \mathbb{E}_{q_{-j}} \big[\log p(x,D) \big] - \int_{\chi_j} q_j(x_j) \log q_j(x_j) dx + \text{const} \\
&= \int_{\chi_j} q_j(x_j) \log \frac{\frac{1}{Z_j} \exp\big(\mathbb{E}_{q_{-j}} \big[\log p(x,D) \big] \big)}{q_j(x_j)} dx + \text{const} \\
&= \mathbb{KL} \Big(q_j || \frac{1}{Z_j} \exp\big(\mathbb{E}_{q_{-j}} \big[\log p(x,D) \big] \big) \Big) + \text{const}.
\end{aligned}
$$

Using the fact that the KL-divergence is nonnegative, the best we can do is when $\mathbb{KL} \Big(q_j || \frac{1}{Z_j} \exp\big(\mathbb{E}_{q_{-j}} \big[\log p(x,D) \big] \big) \Big) = 0$ which occurs only when

$$
q_j = \tfrac{1}{Z_j} \exp\big(\mathbb{E}_{q_{-j}} \big[\log p(x,D) \big] \big).
$$

Thus, this is the form for our update equation.

Appendix B

Frobenius Normed-Error of Estimator Bounds Regret

(Need to reread all of this- don't waste time editing it yet). We show an inherent connection between the frobenius-norm error of the matrix estimation and the pseudo-regret. Recall the notation. The reward for taking action Z at step t is

$$r_{t,Z} = (\mathrm{Tr}(Z^T X) + \varepsilon_t)\beta^{T_{t,Z}},$$

where $(\varepsilon_t)_{t\geq 1}$ is a sequence of independent noise variables with mean 0 and variance σ^2. The variable $T_{t,Z}$ counts the number of times that action Z has been chosen before time t, and the parameter $\beta \in [0,1]$ determines a geometric discounting due to observing the same entry of the matrix repeatedly. We introduce the shorthand: $Tr(Z^T X)$ as X_Z. At each step t, the experimenter observes the reward r_{t,A_t} for the action chosen, A_t.

The pseudo-regret for finite-horizon M is,

$$R_M(A) = \sup_{Z_1,\ldots,Z_M} \mathbb{E}\left(\sum_{t=1}^{M} r_{t,Z_t}\right) - \mathbb{E}\left(\sum_{t=1}^{M} r_{t,A_t}\right).$$

We will refer to the sequence of Z_t's above as the "optimal policy" and the sequence of A_t's as the "estimator policy".

In the notation above we use Z to denote an action chosen by the optimal policy and A to denote the action chosen by the estimator policy. These actions refer to some $(i,j)^{\text{th}}$ entry of the matrix and it can certainly be the case that $Z_t = A_{t'}$ for some t,t'. As a warning- we will have the case that $Z_t = Z_{t'}$ since the same action can be chosen more than once. To distinguish between the number of times the optimal policy chooses action $Z_{t'}$ before time t and the number of times it is chosen by the estimator policy we let $T_{t,Z_{t'}}$ be the number of

times action $Z_{t'}$ is chosen before time t by the optimal policy and we let $S_{t,Z_{t'}}$ be the number of times action $Z_{t'}$ is chosen before time t by the estimator policy.

Theorem 4. *Let $R_M(A)$ be as defined above. Then $R_M(A) < 2\sqrt{M}\frac{1-\beta^{M+1}}{1-\beta}\|X-\hat{X}\|_F$*

We only prove this for the case of zero noise, however a bound on the noise extends to proving the full claim.

Proof. Assume that $\sigma^2 = 0$ so that $r_{t,Z_t} = X_{Z_t}\beta^{T_{t,Z_t}}$ and $r_{t,A_t} = X_{A_t}\beta^{S_{t,A_t}}$. Let $\hat{X}_{A_t}$ denote the value of the A_t entry from our estimator. With $\sigma^2 = 0$ we can rewrite $R_M(A)$ as

$$R_M(A) = \sup_{Z_1,\ldots,Z_M} \sum_{t=1}^{M} r_{t,Z_t} - r_{t,A_t},$$

or equivalently,

$$R_M(A) = \sup_{Z_1,\ldots,Z_M} \sum_{t=1}^{M} X_{Z_t}\beta^{T_{t,Z_t}} - X_{A_t}\beta^{S_{t,A_t}}.$$

Reordering terms, we have the identity:

$$(X_{Z_t}\beta^{T_{t,Z_t}} - X_{A_t}\beta^{S_{t,Z_t}}) + (\hat{X}_{A_t}\beta^{S_{t,Z_t}} - \hat{X}_{Z_t}\beta^{T_{t,Z_t}}) = (\hat{X}_{A_t}\beta^{S_{t,Z_t}} - X_{At}\beta^{S_{t,Z_t}}) + (X_{z_t}\beta^{T_{t,Z_t}} - \hat{X}_{Z_t}\beta^{T_{t,Z_t}}).$$

Summing the left hand side over $t = 1,\ldots,M$ gives you the sum of two regrets- the first is the regret of using policy (A_t) when the true matrix is X, the second is the regret of using policy (Z_t) when the true matrix is $\hat{X}$. In particular, both sums on the left hand side are positive. So, each sum is smaller than the sum on the right hand side. Thus we have

$$\begin{aligned} R_M(A) &\leq \sum_{t=1}^{M} (\hat{X}_{A_t}\beta^{S_{t,Z_t}} - X_{At}\beta^{S_{t,Z_t}}) + (X_{z_t}\beta^{T_{t,Z_t}} - \hat{X}_{Z_t}\beta^{T_{t,Z_t}}) \\ &\leq \sum_{t=1}^{M} \beta^{S_{t,A_t}}|\hat{X}_{A_t} - X_{A_t}| + \beta^{T_{t,Z_t}}|X_{Z_t} - \hat{X}_{Z_t}| \\ &< 2\frac{1-\beta^{M+1}}{1-\beta}\|X-\hat{X}\|_1, \end{aligned}$$

and since $\|X-\hat{X}\|_1^2 < M\|X-\hat{X}\|_F^2$ we have

$$R_M(A) < 2\sqrt{M}\frac{1-\beta^{M+1}}{1-\beta}\|X-\hat{X}\|_F$$

□