Probabilistic and Bayesian Principal Component Analysis

Feiyu Chen, Jianzhun Du, Manish Reddy Vuyyuru

December 24, 2018

Abstract

When handling a high-dimensional dataset, practitioners may wish to use Principal Component Analysis to retain the most important dimensions and remove the uninformative features. Conventional PCA has limitations in some cases, such as on dataset with missing data and when users are hard to determine the number of principal components. In this report, we examined two generalized versions of conventional PCA from a statistical perspective: Probabilistic PCA (PPCA) and Bayesian PCA (BPCA). We compared their behaviors on synthetic data and real-world data with different distributions, and also explored the possible application for estimating missing data.

1 Introduction

Principal Component Analysis (PCA) is a widely-used technique for dimension reduction which finds its applications in data compression, visualization, image processing etc. PCA can be derived by finding a set of orthonormal basis that maximizes the variance in the projected data. It can also be proved that this set of basis minimizes the reconstruction error of the original data.

Let $\{\mathbf{t}_n\}, n \in \{1...N\}$ be the observations and $\boldsymbol{\mu}$ be the sample mean. The sample covariance matrix is then

$$\mathbf{S} = \mathbb{E}[(\mathbf{t} - \boldsymbol{\mu})(\mathbf{t} - \boldsymbol{\mu})^T]. \tag{1}$$

It can be shown that the aforementioned basis is given by the q dominant eigenvectors of $\mathbf{S}$: $\{\mathbf{w_j}\}, j \in \{1...q\}$. Let $\mathbf{W} = (\mathbf{w_1}, ...\mathbf{w_q})$. For observation $\mathbf{t}_n$, the projected data can be written as $\mathbf{x_n} = \mathbf{W}^T(\mathbf{t}_n - \boldsymbol{\mu})$.

One of the limitations of conventional PCA is that it does not define a probability density model. This can prevent PCA to be used in statistical testing, Bayesian inference, mixture model etc. In this report, we examined two generalized versions of conventional PCA from a statistical perspective: Probabilistic PCA (PPCA) and Bayesian PCA (BPCA)In the following sections, we will first elaborate on the derivation and properties of probabilistic PCA and Bayesian PCA. We then compared their behaviors with conventional PCA on datasets with different distributions. At last, we utilized PPCA and BPCA to impute some synthetic data and historical sea surface data.

1.1 Probabilistic PCA

A probabilistic model for PCA [Tipping and Bishop, 1999] defines a probability distribution of the samples, which allows us to carry out statistical testing based on PCA, also opens possible extensions like Bayesian inference and mixture of PCA. Probabilistic PCA is a latent variable model whose solution of maximum likelihood estimation extracts the principal components of the observations.

The Probabilistic PCA first defines a linear mapping from the q-dimensional latent variable $\mathbf{x}$ to the d-dimensional observation $\mathbf{t}$:

$$\mathbf{t} = \mathbf{W}\mathbf{x} + \boldsymbol{\mu} + \boldsymbol{\epsilon} \tag{2}$$

where $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_q$ and $\mathbf{I}_q)$ is a q-dimensional identity matrix, $\mathbf{W}$ is a $d \times q$ matrix, $\boldsymbol{\mu}$ is a d-dimensional vector and $\boldsymbol{\epsilon}$ is a zero mean Gaussian noise: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}_d)$. This implies a multivariate Gaussian distribution of $\mathbf{t}$ conditioned on $\mathbf{x}$:

$$p(\mathbf{t}|\mathbf{x}) = \mathcal{N}(\mathbf{W}\mathbf{x} + \boldsymbol{\mu}, \sigma^2\mathbf{I}_d) = (2\pi\sigma^2)^{-d/2} \exp\left\{-\frac{1}{2\sigma^2}||\mathbf{t} - \mathbf{W}\mathbf{x} - \boldsymbol{\mu}||^2\right\} \tag{3}$$

The marginal distribution of the observed variable $\mathbf{t}$ is then given by the convolution of two Gaussians and is itself Gaussian

$$p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{x})p(\mathbf{x}) = \int (2\pi)^{-d/2}|\mathbf{C}|^{-1/2}\exp\left\{-\frac{1}{2}(\mathbf{t} - \boldsymbol{\mu})\mathbf{C}^{-1}(\mathbf{t} - \boldsymbol{\mu})^T\right\} = \mathcal{N}(\boldsymbol{\mu}, \mathbf{C}) \tag{4}$$

where $\mathbf{C} = \sigma^2\mathbf{I} + \mathbf{W}\mathbf{W}^T$. The log-likelihood of the observations is then

$$\mathcal{L} = \sum_{n=1}^{N}\{p(\mathbf{t}_n)\} = -\frac{Nd}{2}\ln 2\pi - \frac{N}{2}\ln\mathbf{C} - \frac{N}{2}\operatorname{Tr}\left[\mathbf{C}^{-1}\mathbf{S}\right] \tag{5}$$

where $\mathbf{S}$ is observations covariance matrix given by (1). It can be shown that the non-stationary of $\mathcal{L}$ occurs for

$$\mathbf{W} = \mathbf{U}_q(\boldsymbol{\Lambda}_q - \sigma^2\mathbf{I})^{1/2}\mathbf{R} \tag{6}$$

where each column of $\mathbf{U_q}$ is a eigenvector of $\mathbf{S}$, and each diagonal element of (the diagonal matrix) $\boldsymbol{\Lambda}_q$ is the corresponding eigenvalue. $\mathbf{R}$ is an arbitrary rotational matrix. The global maximum of $\mathcal{L}$ is reached only when $\mathbf{U}_q$ consists of the principle eigenvectors of $\mathbf{S}$. Let $\mathbf{W}_{ML}$ be the maximum likelihood solution of W. The maximum likelihood of σ when $\mathbf{W} = \mathbf{W}_{ML}$ is

$$\sigma^2_{ML} = \frac{1}{d-q}\sum_{j=q+1}^{d}\lambda_j \tag{7}$$

where λ_j is the j^{th} dominant eigenvalue of $\mathbf{S}$. It has a natural interpretation as the average variance lost per discarded dimension. Also, conventional PCA can be easily recovered in the limit $\sigma^2 \to 0$.

We notice that $\mathbf{W}_{ML}$ is not necessarily an orthonormal basis because

$$(\mathbf{W}_{ML})^T\mathbf{W}_{ML} = \mathbf{R}^T(\boldsymbol{\Lambda}_q - \sigma^2\mathbf{I})\mathbf{R} \tag{8}$$

is not necessarily diagonal. However, the orthonormal basis $\mathbf{U}_q$ can be recovered, since $\mathbf{R}$ and $\mathbf{\Lambda}_q - \sigma^2\mathbf{I}$ can be calculated by an eigendecompocision of $(\mathbf{W}_{ML})^T\mathbf{W}_{ML}$ (see the equations above).

We can easily see that the probabilistic formulation of PCA does not offer a better result than conventional PCA, nor is it computationally easier to estimate (as we'll see in later sections). The advantage of a probabilistic formulation is that we can obtain a generative latent variable model. Specifically, with this model, we can produce a host of "reduced" and "original" samples, by sampling from the probability distribution of $p(\mathbf{x})$ and $p(\mathbf{t})$.

1.2 Bayesian PCA

One of the limitations of the conventional PCA and Probabilistic PCA is that users have to determine the number of principal components they tend to retain. The general approach is to use variance as the measure of how important a particular dimension is, which is also one of the main assumptions of PCA. However, in some cases, such as in very high dimension datasets or in a mixture modeling context (we would like the components to have potentially different dimensionalities), it is not an easy task to generalize the number of components to capture an appropriate amount of variance. However, an exhaustive search (cross-validation) over the choice of the dimensionality for each of the components is often computationally intractable.

Thus, a Bayesian treatment of PCA was developed by [Bishop, 1999], leading to an *automatic selection* of the appropriate model dimensionality. Instead of discrete model search, Bayesian PCA builds hierarchical model [Allenby and Rossi, 2006] based on Probabilistic PCA, and introduces continuous hyper-parameters to determine an effective number of principal components automatically.

Armed with the probabilistic reformulation of PCA, a Bayesian treatment of PCA is obtained by first introducing a prior distribution $P(\boldsymbol{\mu}, \mathbf{W}, \sigma^2)$ over the parameters of the model. The corresponding posterior distribution $P(\boldsymbol{\mu}, \mathbf{W}, \sigma^2|D)$ is then obtained by multiplying the prior by the likelihood function, whose logarithm is given by 5, and normalizing. Finally, the predictive posterior is obtained by marginalizing over the parameters, so that

$$p(\mathbf{t}|D) = \int\int\int p(\mathbf{t}|\boldsymbol{\mu}, \mathbf{W}, \sigma^2)p(\boldsymbol{\mu}, \mathbf{W}, \sigma^2|D)d\boldsymbol{\mu}d\mathbf{W}d\sigma^2. \tag{9}$$

In order to avoid discrete model search, Bayesian PCA introduces a hierarchical prior $p(\mathbf{W}|\boldsymbol{\alpha})$ over matrix W, governed by a q-dimensional vector of hyper-parameters $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_q\}$. Motivated by the framework of automatic relevance determination (ARD) [MacKay, 2003], the dimensionality of the latent space is set to its maximum possible value $q = d - 1$ (σ^2 in 7 controls the last dimension), and each hyper-parameter controls one of the columns of matrix $\mathbf{W}(d \times q)$ through a conditional Gaussian distribution of the form

$$p(\mathbf{W}|\boldsymbol{\alpha}) = \prod_{i=1}^{q}(\frac{\alpha_i}{2\pi})^{d/2}\exp\left\{-\frac{1}{2}\alpha_i||\mathbf{w_i}||^2\right\}. \tag{10}$$

where $\{\mathbf{w_i}\}$ are the columns of $\mathbf{W}$. Further, we can reformulate this prior distribution as Matrix Gaussian distribution [Gupta and Nagar, 2018] $\mathcal{MVN}_{d,q}(\mathbf{M}, \mathbf{U}, \mathbf{V})$, where $\mathbf{M}$ is $d \times q$

mean matrix, $\mathbf{U}$ is $d \times d$ column covariance matrix, $\mathbf{V}$ is $q \times q$ row covariance matrix. In Bayesian PCA case, we have the form

$$p(\mathbf{W}|\mathbf{M}, \mathbf{U}, \mathbf{V}) = \frac{\exp(-\frac{1}{2}\,\mathrm{Tr}\,[\mathbf{V}^{-1}(\mathbf{X}-\mathbf{M})^T\mathbf{U}^{-1}(\mathbf{X}-\mathbf{M})])}{(2\pi)^{dq/2}|\mathbf{V}|^{d/2}|\mathbf{U}|^{q/2}} \tag{11}$$

where Tr denotes trace, $\mathbf{M} = \mathbf{O}_{d\times q}$ is zero matrix, $\mathbf{U} = \mathrm{diag}(1/\boldsymbol{\alpha})_{d\times d}$ is a diagonal matrix whose diagonal elements are given by $1/\boldsymbol{\alpha}$, $\mathbf{U} = \mathbf{I}_{q\times q}$ is the identity matrix. $\boldsymbol{\alpha}$ controls the inverse variance of $\mathbf{W}$ columns and each $1/\alpha_i$ determines the variance of the corresponding $\mathbf{w_i}$. Therefore, after obtaining the posterior distribution of $\boldsymbol{\alpha}$, if there are some $1/\alpha_i$ with small values, we can "switch off" the corresponding $\mathbf{w_i}$, which should not be the principal components we intends to retain. In intuition, the distribution of $\boldsymbol{\alpha}$ indicates the prior beliefs we have for $\mathbf{W}$. However, while training data coming in, the knowledge from data will soon dominate the distribution of $\boldsymbol{\alpha}$ and gives the appropriate dimensionalities. In practice, Bayesian PCA is so powerful that differences between "large" α_i and normal "normal" α_i are very obvious, which looks like $\{\alpha_i\}$ are divided into different groups automatically. Hence, it is easy to establish the appropriate criterion for filtering out "large" α_i and retain the principal components. The hierarchical model structure is shown in Figure 1.

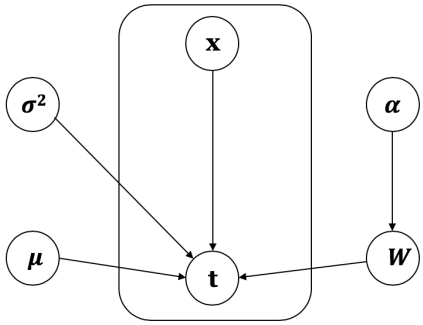

Figure 1: Representation of Bayesian PCA model. The hierarchical prior over $\mathbf{W}$ governed by hyper-parameter $\boldsymbol{\alpha}$. The box denotes a 'plate' comprising a data set of N independent observations of the visible vector $\mathbf{t}$ together with the corresponding hidden variables $\mathbf{x}$.

Unfortunately, marginalizing over the parameters to get (9) is analytically intractable. There are several effective approaches to get posterior distributions over parameters and predictive distribution. [MacKay, 2003] proposed *type-II maximum likelihood* using a local Gaussian approximation to a mode of the posterior distribution. Also, *Markov Chain Monte Carlo (MCMC)* methods, such as Metropolis-Hasting algorithms [Metropolis *et al.*, 1953] and Hamiltonian Monte Carlo [Neal and others, 2011] can be applied to get accurate posteriors. Moreover, *Variational Inference* methods [Blei *et al.*, 2017] are well-developed in this day and age for Bayesian inference. In this project, we implemented Variational Bayesian PCA, whose details are shown in the later section.

To sum up, on the one hand, Bayesian PCA can help select the appropriate number of components automatically. On the other hand, armed with the formulation of Probabilistic

PCA, we can obtain joint/conditional/posterior distribution of all variables via Bayesian PCA, which not only provides an approach for new data generation or statistical test but also makes the model more robust while capturing the uncertainty in the dataset.

2 Implementations

2.1 Implementation of Probabilistic PCA

2.1.1 E-M Algorithm

We can use an Expectation-Maximization algorithm (E-M algorithm) [Dempster *et al.*, 1977] to estimate the parameters of the Probabilistic PCA. The EM algorithm tries to maximize the complete data likelihood $\mathcal{L}_C$ iteratively. The algorithm can be break into two part. In the estimation step (E-step), we estimate $\mathcal{L}_C$. In the maximization step (M-step) we update $\mathbf{W}$ and σ^2. for the E-step, the complete data likelihood is

$$\mathcal{L}_C = \sum_{n=1}^{N} \ln p(\mathbf{t}_n, \mathbf{x}_n) \tag{12}$$

where $p(\mathbf{t}_n, \mathbf{x}_n)$ can be obtained by 3 and 27:

$$p(\mathbf{t}_n, \mathbf{x}_n) = (2\pi\sigma^2)^{-d/2} \exp\left\{-\frac{1}{2\sigma^2}||\mathbf{t}_n - \mathbf{W}\mathbf{x}_n - \boldsymbol{\mu}||^2\right\} (2\pi)^{-d/2} \exp\left\{-\frac{1}{2}\mathbf{x}_n^T\mathbf{x_n}\right\} \tag{13}$$

In order to optimize $\mathcal{L}_C$ with respect to $\mathbf{W}$ and σ, we need to know $\mathbf{x}$, which is a latent variable we don't have access to. So at each iteration of the EM algorithm, $\mathbf{x}$ is estimated to be its expectation, this expectation is then used to estimate $\mathcal{L}_C$:

$$\begin{aligned}\langle\mathcal{L}_C\rangle = \text{const.} - \frac{d}{2}\ln\sigma^2 - \sum_{n=1}^{N}\{-\frac{1}{2}\operatorname{Tr}\left[\langle\mathbf{x}_n\mathbf{x}_n^T\rangle\right] - \frac{1}{2\sigma^2}\operatorname{Tr}\left[(\mathbf{t}_n - \boldsymbol{\mu})(\mathbf{t}_n - \boldsymbol{\mu})^T\right. \\ \left. -2(\mathbf{t}_n - \boldsymbol{\mu})\langle\mathbf{x}_n\rangle^T\mathbf{W}^T + \mathbf{W}\langle\mathbf{x}_n\mathbf{x}_n^T\rangle\mathbf{W}^T\right]\}\end{aligned} \tag{14}$$

where

$$\langle\mathbf{x}_n\rangle = (\sigma^2\mathbf{I} + \mathbf{W}\mathbf{W}^T)^{-1}\mathbf{W}^T(\mathbf{t}_n - \boldsymbol{\mu}) \tag{15}$$

$$\langle\mathbf{x}_n\mathbf{x}_n^T\rangle = \sigma^2(\sigma^2\mathbf{I} + \mathbf{W}\mathbf{W}^T)^{-1} + \langle\mathbf{x}_n\rangle\langle\mathbf{x}_n^T\rangle \tag{16}$$

In the M-step, we maximize $\mathcal{L}_C$ by differentiate (12) with respect to $\mathbf{W}$ and σ^2 and set the derivatives to zero, which leads to new estimation of $\mathbf{W}$ and σ^2:

$$\widetilde{\mathbf{W}} = \mathbf{S}\mathbf{W}(\sigma^2\mathbf{I} + \mathbf{M}^{-1}\mathbf{W}^T\mathbf{S}\mathbf{W})^{-1} \tag{17}$$

$$\tilde{\sigma}^2 = \frac{1}{d}\operatorname{Tr}\left[\mathbf{S} - \mathbf{S}\mathbf{W}\mathbf{M}^{-1}\widetilde{\mathbf{W}}\right] \tag{18}$$

where $\mathbf{S}$ is given by (1) and $\mathbf{M} = \sigma^2\mathbf{I} + \mathbf{W}\mathbf{W}^T$. Note that we don't actually perform any computation in the E-step, the role of (12) is to provide the math foundation for the M-step. The maximum likelihood solution for $\mathbf{W}$ and σ is reached when the iteration converges.

2.1.2 Eigenvalue Decomposition

Alternatively, the EM-algorithm can be avoided by estimating $\mathbf{U}_q$ and $\mathbf{\Lambda}_q$ by eigenvalue decomposition of the covariance matrix $\mathbf{S}$ as we do in conventional PCA. We can then estimate σ by (7).

After the $\mathbf{W}$ and σ are fitted. We can perform transformation (project observations to latent space) and inverse transformation (project latent variables back to observations). We implement two kinds of transformations, a probabilistic one that returns a sample of $\mathbf{x}_n$'s or $\mathbf{t}_n$'s drawn from $p(\mathbf{x}|\mathbf{t})$ or $p(\mathbf{t}|\mathbf{x})$, and a deterministic one that returns the expectation of $\mathbf{x}_n$'s or $\mathbf{t}_n$'s.

2.2 Implementation of Bayesian PCA

2.2.1 Variational Inference

We decided to utilize Variational Inference to approximate the posterior distribution $p(\boldsymbol{\theta}|D)$ and obtain predictive posterior distribution $p(\mathbf{t}|\boldsymbol{\theta}, D)$, where $\boldsymbol{\theta} = \{\theta_i\}$ denotes the set of all parameters and latent variables in Bayesian models. The main idea behind Variational Inference is to approximate the posterior distribution with respect to latent variables by optimization, rather than sampling with MCMC methods (sampling is too computationally expensive to deploy models in practice).

First, variational methods introduce a family of approximate densities $\mathcal{Q}(\boldsymbol{\theta})$, which is a set of probability densities over the latent variables. Then, the goal of variational methods (19) is to find the member of that family that minimizes the Kullback-Leibler (KL) divergence (20), which is always greater or equal to zero and measures how one probability distribution is different from a second, reference probability distribution, towards the exact posterior $p(\boldsymbol{\theta}|D)$.

$$q^*(\boldsymbol{\theta}) = \underset{q(\boldsymbol{\theta})\in\mathcal{Q}}{\operatorname{argmax}}\, \mathcal{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|D)). \tag{19}$$

$$\mathcal{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|D)) = \int q(\boldsymbol{\theta}) \ln \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|D)} d\boldsymbol{\theta}. \tag{20}$$

With *Bayes Theorem* and $D = \{X, Y\}$, (20) can be rewritten to

$$\begin{aligned}
\mathcal{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|D)) &= \int q(\boldsymbol{\theta})[\ln q(\boldsymbol{\theta}) - \ln \frac{p(\boldsymbol{\theta}|D)p(D)}{p(D)}]d\boldsymbol{\theta} \\
&= \int q(\boldsymbol{\theta})[\ln q(\boldsymbol{\theta}) - \ln p(D, \boldsymbol{\theta})]d\boldsymbol{\theta} + \int q(\boldsymbol{\theta}) \ln p(D) d\boldsymbol{\theta} \\
&= \int q(\boldsymbol{\theta})[\underbrace{\ln q(\boldsymbol{\theta})}_{\text{entropy}} - \underbrace{\ln p(Y|\boldsymbol{\theta}, X)}_{\text{log-likelihood}} - \underbrace{\ln p(\boldsymbol{\theta})}_{\text{log-prior}}]d\boldsymbol{\theta} + \underbrace{\ln p(D) \int q(\boldsymbol{\theta}) d\boldsymbol{\theta}}_{\text{const.}}.
\end{aligned} \tag{21}$$

Let

$$\mathcal{L}(\mathcal{Q}) = \int q(\boldsymbol{\theta})[-\ln q(\boldsymbol{\theta}) + \ln p(Y|\boldsymbol{\theta}, X) + \ln p(\boldsymbol{\theta})]d\boldsymbol{\theta}, \tag{22}$$

(21) is simplified to

$$\ln p(D) \int q(\boldsymbol{\theta}) d\boldsymbol{\theta} = \mathcal{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|D)) + \mathcal{L}(\mathcal{Q}). \tag{23}$$

We can find that $p(D)$ is constant and $\mathcal{KL}(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|D)) \geq 0$, minimizing $\mathcal{L}(\mathcal{Q})$ (we call it "ELBO") is equivalent to maximizing KL divergence (20). Since the log-likelihood term and log-prior term are both easy to calculate, now the main issues are (I) how to select an appropriate variational family $\mathcal{Q}(\boldsymbol{\theta})$; (II) how to find optimal parameters minimizing $\mathcal{L}(\mathcal{Q})$.

One of the most common approaches for resolving issue (I) is *mean-field* variational family, where the latent variables are mutually independent and each governed by a distinct factor in the variational density. A generic member of the mean-field variational family is

$$q(\boldsymbol{\theta}) = \prod_i q_i(\theta_i). \tag{24}$$

We usually choose Gaussian for the distribution of $q_i(\theta_i)$, but there are also lots of alternatives, such as Matrix Gaussian distribution we mentioned before. Consequently, $q(\theta)$ can be easily calculated, and fixed-point iteration (if $\mathcal{L}(\mathcal{Q})$ can be solved analytically) or gradient-based optimization methods, can be adopted to minimize $\mathcal{L}(\mathcal{Q})$.

2.2.2 Variational Bayesian PCA

Equipped with Variational Inference described in last section, we choose mean-field variational family - factorized (24), then $\mathcal{L}(\mathcal{Q})$ can be minimized over $q_i(\theta_i)$ with the form (see derivation in Appendix A)

$$q_i(\theta_i) = \frac{\exp\left\{\mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D, \boldsymbol{\theta})]\right\}}{\int \exp\left\{\mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D, \boldsymbol{\theta})]\right\} d\theta_i} \tag{25}$$

where $\mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}$ denotes the expectation with respect to the distribution

$$q(\boldsymbol{\theta})/q(\theta_i) = \prod_{k \neq i} q_k(\theta_k). \tag{26}$$

We specify priors over $\mathbf{x}$, $\boldsymbol{\alpha}$, $\boldsymbol{\mu}$, $\tau = \sigma^{-2}$ as follow

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \mathbf{I}_q), \tag{27}$$

$$p(\boldsymbol{\alpha}) = \prod_{i=1}^{q} \Gamma(\alpha_i | a_\alpha, b_\alpha), \tag{28}$$

$$p(\boldsymbol{\mu}) = \mathcal{N}(\mathbf{0}, \beta^{-1}\mathbf{I}_d) \tag{29}$$

$$p(\tau) = \Gamma(a_\tau, b_\tau). \tag{30}$$

where $\Gamma(x|a, b)$ is a Gamma distribution and $\beta, a_\alpha, b_\alpha, a_\tau, b_\tau$ are all hyper-parameters. In practice, we set all of them to 0.001 to get broad priors. The variational distribution of $q(\boldsymbol{\theta})$ has the form

$$q(\mathbf{X}, \mathbf{W}, \boldsymbol{\alpha}, \boldsymbol{\mu}, \tau) = q(\mathbf{X})q(\mathbf{W})q(\boldsymbol{\alpha})q(\boldsymbol{\mu})q(\tau). \tag{31}$$

where $X = \{\mathbf{x}_n\}$. The likelihood in (22) is

$$p(D|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\mathbf{t}_n|\mathbf{x}_n, \mathbf{W}, \boldsymbol{\mu}, \tau). \tag{32}$$

The prior in (22) is

$$p(\boldsymbol{\theta}) = p(X)p(\mathbf{W}|\boldsymbol{\alpha})p(\boldsymbol{\alpha})p(\boldsymbol{\mu})p(\tau). \tag{33}$$

Combine (10) and (25)-(33), we are able to get the following results for the distribution of $q(\theta_i)$ (see derivation in Appendix B)

$$q(\mathbf{X}) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{x}_n|\mathbf{m}_{\mathbf{x}}^{(n)}, \boldsymbol{\Sigma}_{\mathbf{x}}) \tag{34}$$

$$q(\boldsymbol{\mu}) = \mathcal{N}(\mathbf{m}_{\boldsymbol{\mu}}, \boldsymbol{\Sigma}_{\boldsymbol{\mu}}) \tag{35}$$

$$q(\mathbf{W}) = \prod_{k=1}^{d} \mathcal{N}(\tilde{\mathbf{w}}_k|\mathbf{m}_{\mathbf{w}}^{(k)}, \boldsymbol{\Sigma}_{\mathbf{w}}) \tag{36}$$

$$q(\boldsymbol{\alpha}) = \prod_{i=1}^{q} \Gamma(\boldsymbol{\alpha}|\tilde{a}_\alpha, \tilde{b}_{\alpha_i}) \tag{37}$$

$$q(\tau) = \Gamma(\tau|\tilde{a}_\tau, \tilde{b}_\tau) \tag{38}$$

where $\tilde{\mathbf{w}}_k$ is a column vector corresponding to the kth row of $\mathbf{W}$, and

$$\mathbf{m}_{\mathbf{x}}^{(n)} = \langle\tau\rangle\boldsymbol{\Sigma}_{\mathbf{x}}\langle\mathbf{W}^T\rangle(\mathbf{t}_n - \langle\boldsymbol{\mu}\rangle) \tag{39}$$

$$\boldsymbol{\Sigma}_{\mathbf{x}} = (\mathbf{I} + \langle\tau\rangle\langle\mathbf{W}^T\mathbf{W}\rangle)^{-1} \tag{40}$$

$$\mathbf{m}_{\boldsymbol{\mu}} = \langle\tau\rangle\boldsymbol{\Sigma}_{\boldsymbol{\mu}} \sum_{n=1}^{N} (\mathbf{t}_n - \langle\mathbf{W}\rangle\langle\mathbf{x}_n\rangle) \tag{41}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{\mu}} = (\beta + N\langle\tau\rangle)^{-1}\mathbf{I} \tag{42}$$

$$\mathbf{m}_{\mathbf{w}}^{(k)} = \langle\tau\rangle\boldsymbol{\Sigma}_{\mathbf{w}} \sum_{n=1}^{N} \langle\mathbf{x}_n\rangle(t_{nk} - \langle\mu_k\rangle) \tag{43}$$

$$\boldsymbol{\Sigma}_{\mathbf{w}} = (\mathrm{diag}\langle\alpha\rangle + \langle\tau\rangle \sum_{n=1}^{N} \langle\mathbf{x}_n\mathbf{x}_n^T\rangle)^{-1} \tag{44}$$

$$\tilde{a}_\alpha = a_\alpha + \frac{d}{2} \tag{45}$$

$$\tilde{b}_\alpha = b_\alpha + \frac{\langle\tilde{\mathbf{w}}_k^T\tilde{\mathbf{w}}_k\rangle}{2} \tag{46}$$

$$\tilde{a}_\tau = a_\tau + \frac{Nd}{2} \tag{47}$$

$$\tilde{b}_\tau = b_\tau + \frac{1}{2}\sum_{n=1}^{N}\{\mathbf{t}_n^T\mathbf{t}_n + \langle\boldsymbol{\mu}^T\boldsymbol{\mu}\rangle + \mathrm{Tr}\left(\langle\mathbf{W}^T\mathbf{W}\rangle\langle\mathbf{x}_n^T\mathbf{x}_n\rangle\right) \\ +2\langle\boldsymbol{\mu}^T\rangle\langle\mathbf{W}\rangle\langle\mathbf{x}_n\rangle - 2\mathbf{t}_n^T\langle\mathbf{W}\rangle\langle\mathbf{x}_n\rangle - 2\mathbf{t}_n^T\langle\boldsymbol{\mu}\rangle\} \tag{48}$$

and $\mathrm{diag}\langle\alpha\rangle$ is a diagonal matrix whose diagonal elements are given by $\boldsymbol{\alpha}$. Now, with (39)-(48), we can utilize fixed-point iteration to find optimal parameters $\mathbf{x}$, $\mathbf{W}$, $\boldsymbol{\alpha}$, $\boldsymbol{\mu}$, σ^2. Specifically, starting at an initial guess, we can cycle through the groups of variables in turn, estimating one variable with others fixed iteratively until convergence. This is partly because the log-likelihood term in (21) has closed form due to the linear relationship (2) between $\mathbf{t}$ and $\mathbf{x}$, (25) can be solved analytically. Thus, the target parameters can be computed easily and accurately. However, fixed-point iteration might converge very slowly or be trapped into local minimum, which are two critical defects. For example, when the initial guess is not appropriate or the data is highly skewed or contain a great amount of noise, we might not be able to get correct results. Moreover, as we can see the example in AM205 Lecture 17, fixed-point iteration is linearly converged and sometimes diverges.

3 Experiments

Experiments were carried out on toy examples to ensure that our implementations of Probabilistic PCA (PPCA) and Bayesian PCA (BPCA) were sound. The examples also serve to illustrate some of the capabilities and downfalls of the different flavors of PCA.

3.1 Toy Examples

We first considered a standard toy dataset with 10 dimensions and 100 data points. Data were drawn from a normal distribution with standard deviations of [5, 4, 3, 2] in 4 directions and 1 in the other directions. Below, the results from fitting a standard implementation of PCA from *scikit-learn* and our PPCA, BPCA implementations are shown. The covariance matrix for the toy dataset is visualized in Figure 6.

3.1.1 Orthogonality

We visualized the elements in the weight matrix $\mathbf{W}$ in each flavor of PCA in Hinton diagrams as shown in Figure 2. The orthogonality, or lack of, can be visually motivated by the form of the Hinton diagrams. In the weight matrix inferred by PCA, the vectors W_i are orthogonal. This is not the case for the weight matrix inferred by PPCA and BPCA.

However, recall that as discussed previously (see (6)), there exists an analytical maximum likelihood solution to PPCA in the singular value decomposition (SVD) form. Therefore, U_q, the matrix of q leading principle directions, can be recovered from the inferred W_{ML} in PPCA. This is demonstrated in Figure 3 below where we successfully determine the leading principle directions via SVD of $\mathbf{W}_{ML}$.

Also, where PPCA comes in handy is that by reformulating PCA in a probabilistic fashion we have a likelihood measure. We can then use this likelihood measure as a baseline to evaluate other probabilistic models.

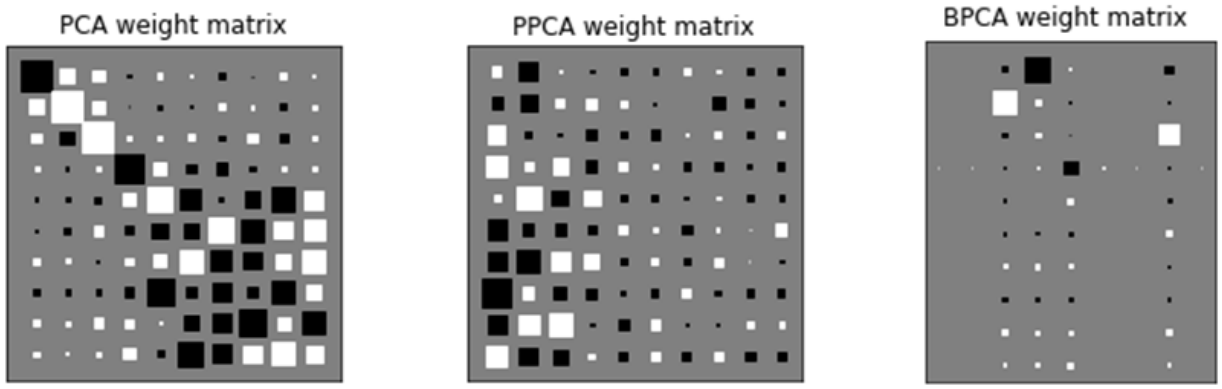

Figure 2: Hinton diagrams of weights for PCA, PPCA, BPCA

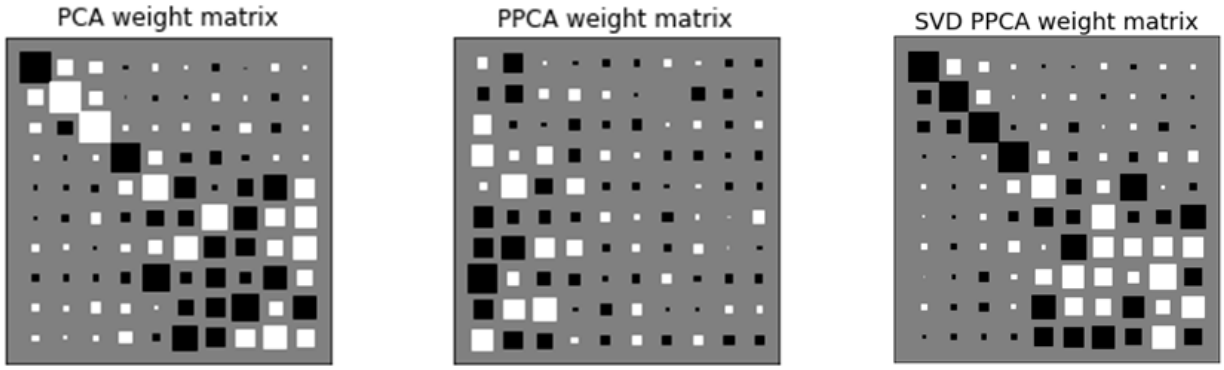

Figure 3: Hinton diagrams of weights for PCA, PPCA, Orthogonal representation of PPCA weights

Notice that in the weight matrix of BPCA, several vectors W_i have been zeroed out. BPCA has successfully determined that there were 4 directions of significant variance in the dataset, as indicated by the 4 vectors w_i in the W matrix that were not zeroed out.

3.1.2 Number of Principal Components

In [Bishop, 1999] the number of principal components was determined as the number of weight vectors $\mathbf{w}_i$ that are non-zero. Alternatively, also notice that the vector of hyperparameters $\boldsymbol{\alpha}$ controls the columns of the matrix W through a conditional distribution (see equation 10). Therefore the number of principal components can also be directly inferred from the $\boldsymbol{\alpha}$ vector. For example, consider the spread of $\boldsymbol{\alpha}$ parameters for a BPCA model fit to the standard MNIST dataset shown in Figure 4. Recall that as previously demonstrated, a component of $\boldsymbol{\alpha}$ with a large value will zero out the corresponding weight vector $\mathbf{w}_i$. Therefore, by setting a minimum threshold value for components of $\boldsymbol{\alpha}$, it is possible to automatically determine the number of significant weight vectors $\mathbf{w}_i$. These two approaches should be equivalent and in our experiments, we have found this to indeed be the case. Our implementation of BPCA uses the hyper-parameters directly to infer the number of principal

components, however, all measurements of the number of components presented here use the weight vectors.

Now that we have an approach to determine the number of principal components given the weight matrix, we can determine if BPCA had indeed picked up the effective dimensionality of the latent space. For this toy example, this can be immediately inferred from the Hinton diagram. But it is useful to walk through a more general approach that is generalizable to more complicated datasets (e.g. see MNIST section below).

As discussed in [Bishop, 1999], we fit multiple PPCA models to the dataset using all possible values that the dimension of the latent space q can take (values between 1-9 in this case) and calculate their log-likelihoods. This exhaustive search helps to determine the optimal value of q, which corresponds to the optimal (highest) value of the log-likelihood. We then use BPCA to automatically determine the effective dimensionality and compare the log-likelihoods as shown in Figure 5. Values in the figure were from averaging 50 repeats of this trial. As expected, searching over the values of q using PPCA models indicates that the latent space has an effective dimensionality of 4 (beyond 4, the log-likelihood stagnates). BPCA automatically discovers this as the appropriate dimensionality.

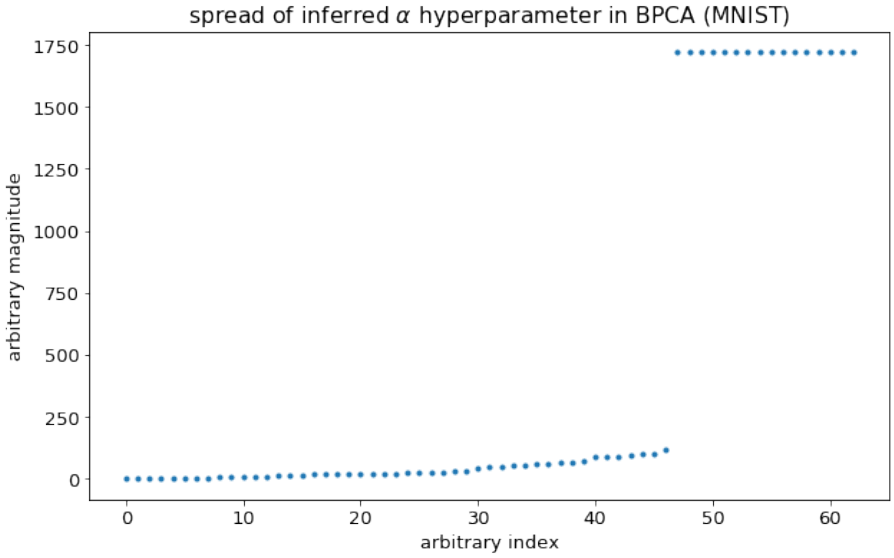

Figure 4: Spread of $\{\alpha_i\}$ hyper-parameter values for BPCA fit to MNIST. Notice the large gap in magnitudes between two "groups" $\{\alpha_i\}$ that corresponds to zero and non-zero weight vectors.

3.1.3 Inferred Representations

We also performed several additional tests on the representations of the datasets inferred by PPCA and BPCA. For example, Figure 6 shows a comparison of the original covariance matrix of the toy dataset, alongside the covariance matrix representation inferred by PPCA and BPCA.

Additionally, in Figure 7, the first 3 dimensions of samples drawn from the inferred underlying distributions using BPCA, PPCA, PCA were compared.

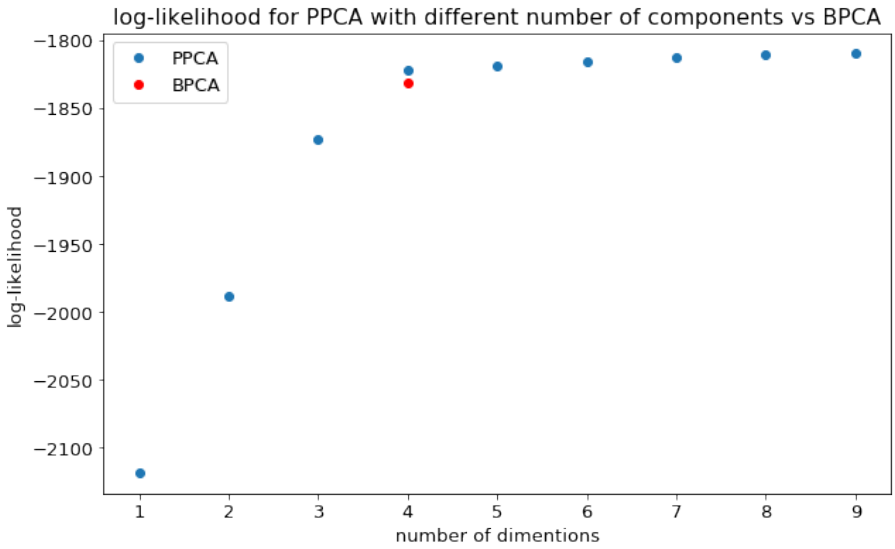

Figure 5: Log-likelihood for PPCA and BPCA models on toy datasets. The best number of dimensions, determined by an exhaustive search, for PPCA is 4. We recover the same best number of dimensions for the latent space automatically for BPCA

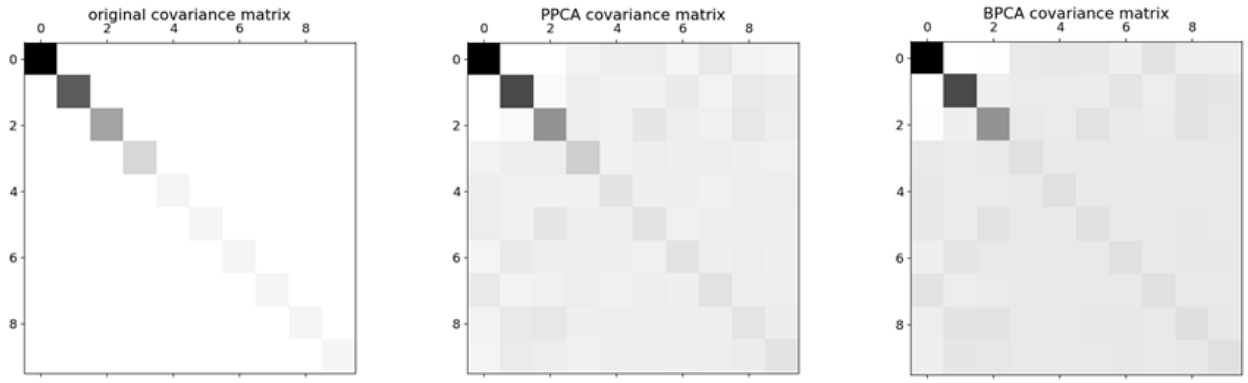

Figure 6: Covariance matrices of original dataset, and inferred by PPCA and BPCA. PPCA and BPCA infer representations close to the actual.

3.1.4 Number of Dimensions and Data Points

We also studied the relationship between the true and captured dimensionality of the latent space and the number of data points.

We used a 25-dimensional dataset draw from a normal distribution with standard deviations of [25, 24, 23, ..., 1] in the 25 directions. We fitted different BPCA models with data points ranging from 20 to 100 and recorded the inferred number of dimensions of the latent space. This trial was averaged over 50 repeats. As the number of data points approaches infinity, we would expect the PPCA and BPCA to converge and infer correctly the true number of dimensions in the latent space.

In BPCA, components of the weight matrix that do not have sufficient evidence are driven to 0. For a small number of data points, we would expect that BPCA would guess a lower

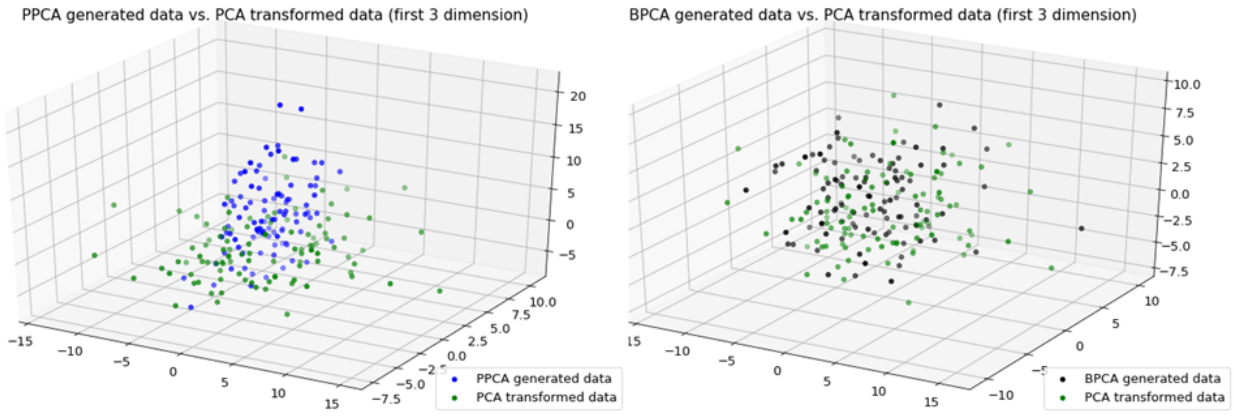

Figure 7: Comparison of samples drawn from the inferred latent space between BPCA, PPCA, PCA. BPCA and PCA infer a more similar underlying representation, compared to PPCA.

dimensional latent space because of insufficient evidence in certain directions leading to the corresponding weight vector W_i bring zeroed out.

This is demonstrated in Figure 8, where for a low number of data points, BPCA underestimates the dimensionality of the data points. But given a sufficiently high number of data points, given the sufficient evidence, BPCA correctly determines the dimensionality of the latent space.

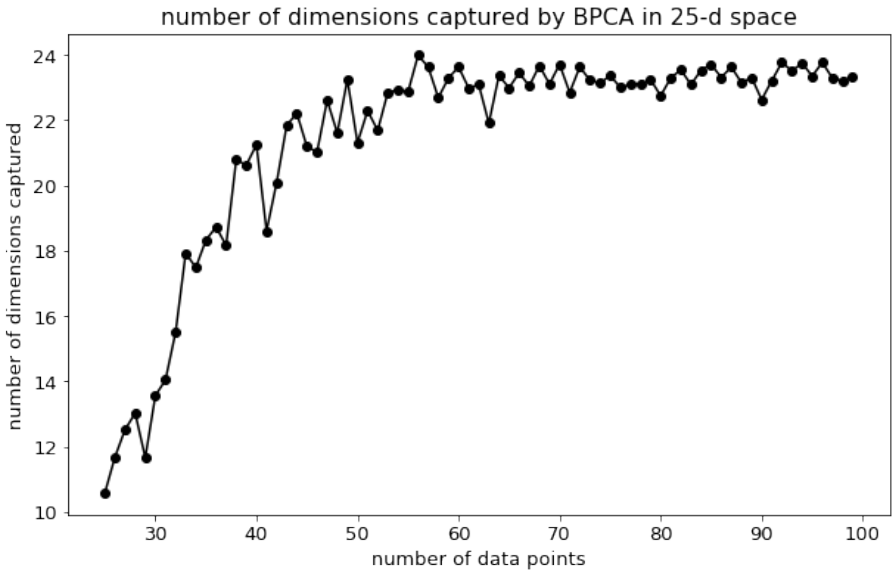

Figure 8: Inferred dimensionality of latent space as a function of the number of datapoints in the 25d dataset for the BPCA model (results averaged from 50 repeats). With only a few data points, BPCA is unable to infer the true dimentionality because of insufficient evidence.

3.2 MNIST

Beyond the toy dataset, we tested our PCA flavors on the handwritten digits database, MNIST. We 'reconstructed' the digits with the different flavors PCA, where we transformed the digits to the latent space and back, as shown in Figure 9. To determine the dimensionality of the latent space for fixed q models such as PCA and PPCA, we first fit BPCA and use the inferred principal dimensionality.

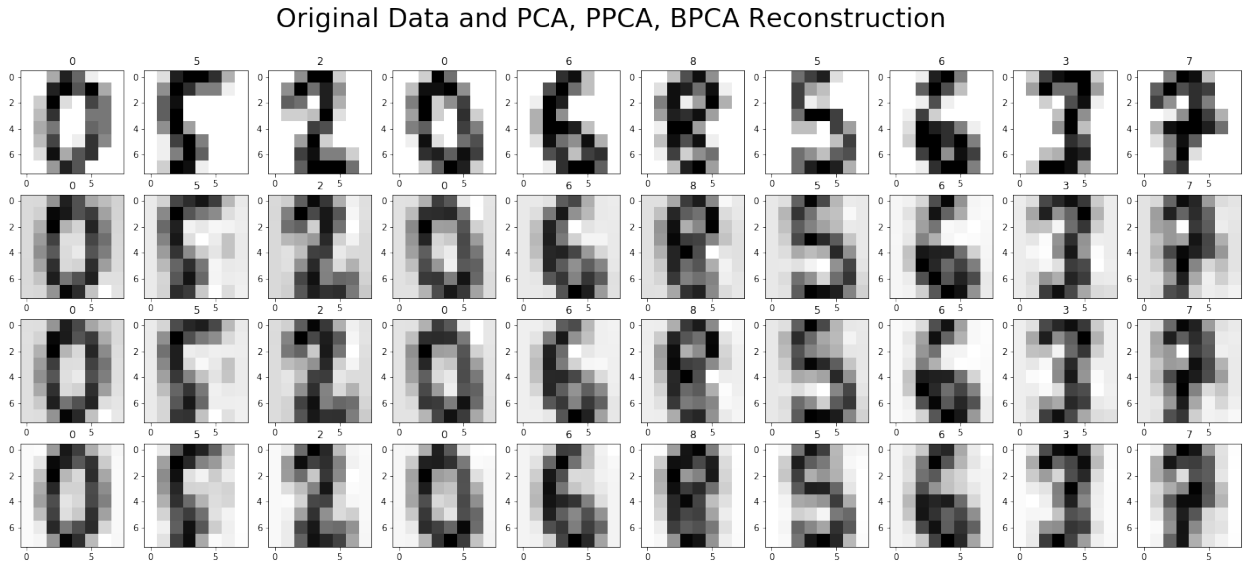

Figure 9: Top row to bottom row: original image, PCA, PPCA, BPCA reconstruction of MNIST. Images were projected into the latent space and back. Scrutinizing the border of images (white part), we can find the reconstructed images from BPCA are more closed to the original ones.

As discussed before, we also verified this inferred dimensionality of the latent space by performing an exhaustive search over possible values of q using PPCA models. In figure 10, we compare the log-likelihood of these models with the log-likelihood of a BPCA fit. The dimensionality of the latent space inferred by BPCA corresponds to almost 100% variance capture by conventional PCA. At this dimensionality (49 dimensions) of the latent space, BPCA achieves a log-likelihood of -168.3, while PPCA and PCA achieved log-likelihoods of -123.8 and -121.5 respectively.

4 Application: Estimating Missing Data

4.1 Imputation using PCA

One of the promising applications of probabilistic PCA is to impute the missing data in datasets [Ishii *et al.*, 2003]. Suppose we have random missing entries in our observations,

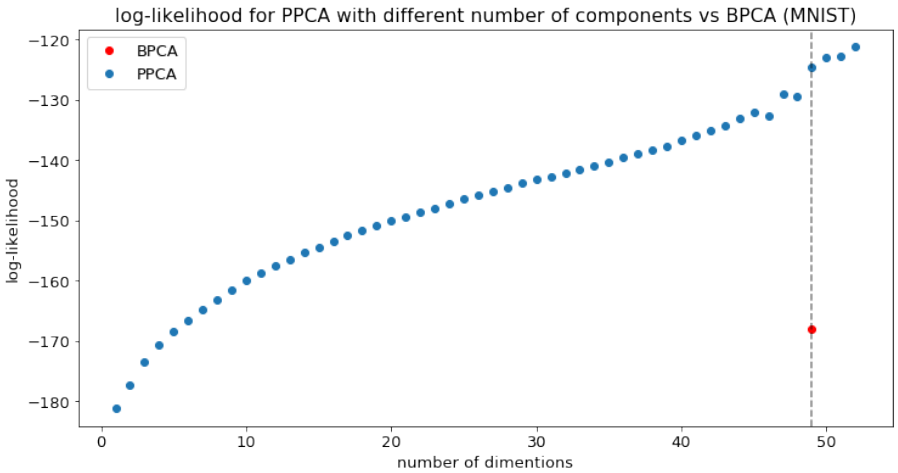

Figure 10: Log-likelihood for PPCA and BPCA models on MNIST. A large number of dimensions are required to capture the variance. The number of dimensions inferred by BPCA corresponds to roughly 100% variance capture by PCA.

like:

$$\begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} & \times \\ t_{21} & \times & t_{23} & \times & \times \\ t_{31} & \times & t_{32} & \times & t_{35} \end{bmatrix}$$

We can impute the dataset by fitting a probabilistic PCA that maximizes the likelihood of having the entries that we did observe. By doing this, we find a representation such that $\mathbf{t} \approx \mathbf{W}\mathbf{x} + \boldsymbol{\mu} + \boldsymbol{\epsilon}$ for the observed entries, and the rest of the matrix can be taken as the reconstruction of missing values. One of the pitfalls of using this technique for imputation is that we may overfit the original data if we wrongly chose the number of latent dimensions. An extreme example to illustrate how this kind of overfitting is shown in Figure 11. This is a scenario where Bayesian PCA may be useful. With Bayesian PCA we don't have to specify the number of dimensions as the algorithm will make an estimation of the dimension by itself.

Motivated by E-M algorithm, we implemented a simple imputation routine. We first fill in all the missing entries with the mean of that dimension. We then repeatedly fit a PCA model (either by conventional PCA, probabilistic PCA or Bayesian PCA) of the data, and fill in the missing entries with the reconstructed value of the PCA until the values in those entries converge.

4.2 Toy Examples

We test our imputation routine with three types of datasets.

- The first toy dataset is 1000 samples generated from 10-dimensional Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{S})$ with 5 principal components, where

  $\mathbf{S} = \text{diag}(5^2, 4^2, 3^2, 2^2, 1^2, 0.5^2, 0.5^2, 0.5^2, 0.5^2, 0.5^2)$.

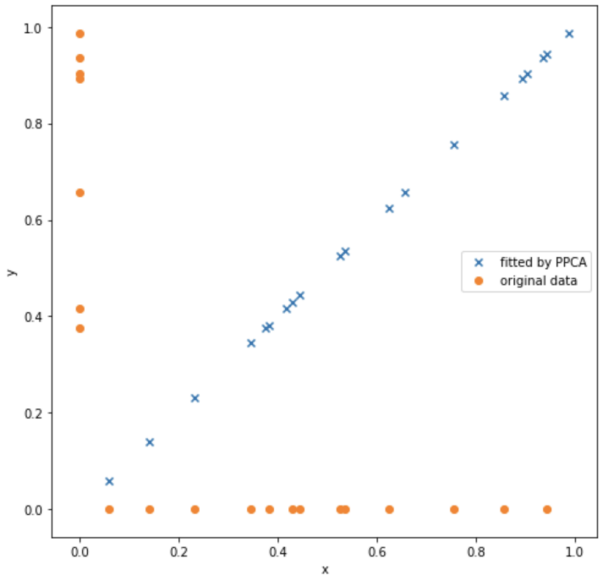

Figure 11: Probabilistic PCA overfitting a 2-dimensional dataset. Suppose we have a 2-dimensional dataset, where each observation has exactly one missing value in one of the dimensions, and the missing values are actually all zero. In this case, a 1-dimensional probabilistic PCA model will reconstruct the data into a straight line.

- Then we feed the first dataset as input to a multi-layer perceptron (1 hidden layer with 50 hidden units and the output layer has 10 units) and take the output of this neural network as the second dataset. ReLU ($\max\{0, x\}$) is used as the activation function. Weights are initialized randomly by sampling from $N(0, \frac{1}{16})$. Thus, in the second dataset, we attain 1000 10-dimensional but *non-linear* data. Since PCA is based on the assumption that the principal components are a linear combination of the original features, we intend to see whether PCA and BPCA work in this case.

- We have noticed that the datasets we experimented so far all have the property that the number of samples are much larger than the number of dimensions. Hence, we design the last toy dataset with 100 samples but 100 dimensions. Samples are generated from 100-dimensional Gaussian Distribution with zero means and 10 principal components with variance from 10 to 1, and the rest of the components all have variance 0.1. We would like to the generalization ability of PPCA and BPCA when the samples are insufficient.

Besides, we consider the rate of missing data in the datasets. In general, high missing rate jeopardizes the performance of imputation. Missing rates are set in three level 10%, 40%, and 70%. We use Mean Square Error to measure the performance of imputation. Mean imputation is used as the baseline. The results of imputation by PCA, PPCA, and BPCA on these three toy datasets are shown in Figure 12.

From the test results on these three synthetic datasets, we first conclude that PCA should not be used to estimate missing data, though in some cases the MSE of PCA decreases first then increases. PPCA diverges when the missing rate is high, but BPCA works in that case.

However, surprisingly, BPCA and PPCA both perform well when the data is non-linearly transformed. We are still trying to find an appropriate interpretation in this scenario. Lastly, when the number of samples is close to the number of dimensions, all three types of PCA fail. It is acceptable due to uninformative samples. BPCA fluctuates around a high MSE value probably because it fails in a local minimum.

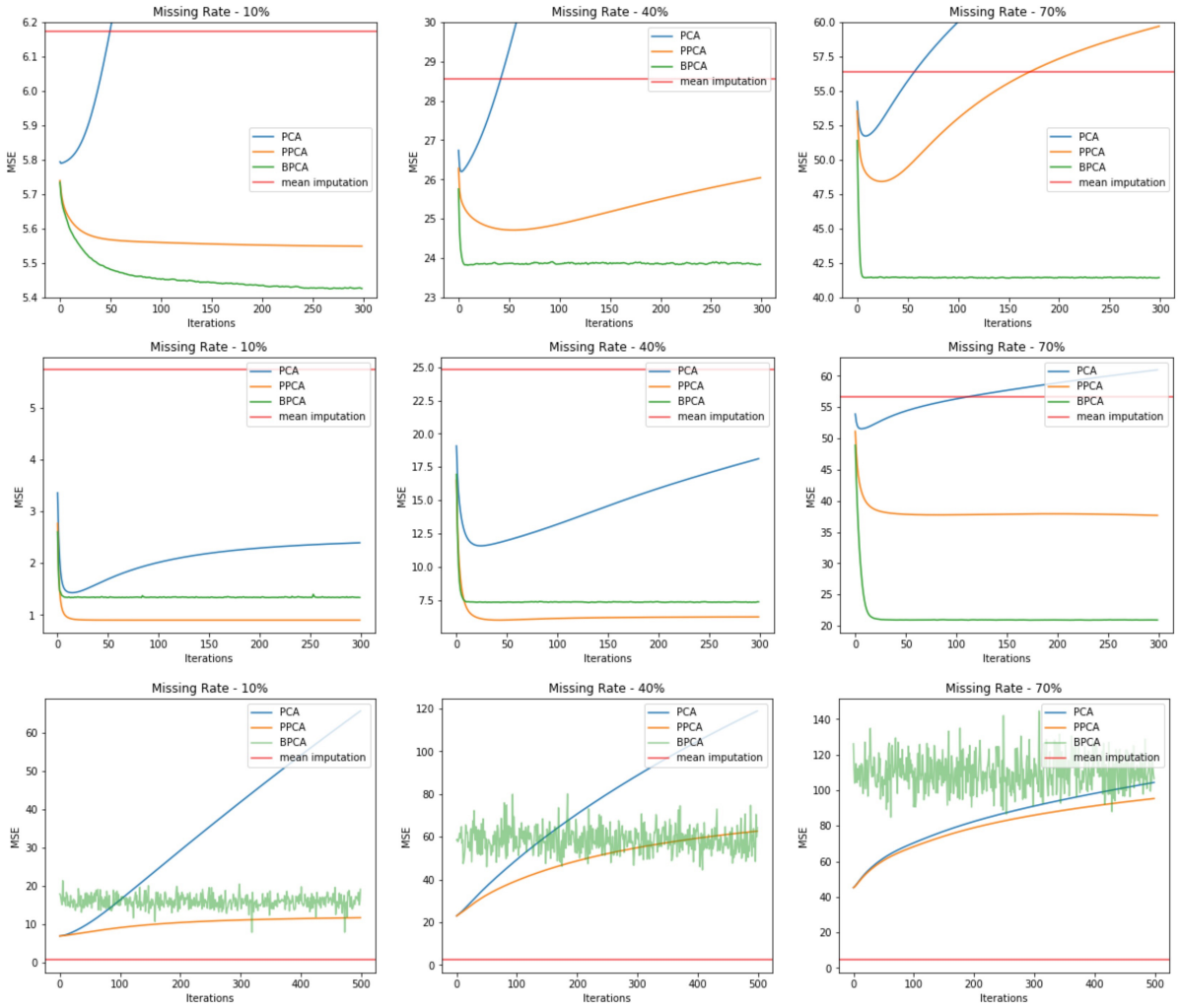

Figure 12: The imputation results of PCA, PPCA, and BPCA on three toy datasets. Obviously, PCA fails (diverges) in all cases. *Top*: first normal toy dataset. We can see in this normal dataset when the missing rate is low (10%), PPCA and BPCA both converge and BPCA betters. However, when the missing rate increases, PPCA fails but BPCA still works. *Middle*: second non-linear dataset. PPCA and BPCA converge in all three levels of missing rates, PPCA performs better when most of the data is available, but BPCA is superior when most of the data is missed. *Bottom*: third insufficient dataset. PPCA and BPCA both fail in this dataset. PPCA diverges and BPCA fluctuates around a high MSE value.

4.3 Recovering History Sea Surface Temperature

We also try using this method to recover historical sea temperature (SST) data [Ilin and Kaplan, 2009]. We used 2 datasets to test the imputation quality. One is the GOSTA atlas8 MOHSST5 dataset [1] and the other is the NOAA (OI) SST dataset [2]. The first is the monthly mean SST from Jan 1856 - Dec 1995, and the second is the monthly mean SST from Dec 1981 - Nov 2018. For the years 1981 - 1991, the MOHSST dataset contains quite many missing entries, while the NOAASST dataset is more complete and accurate. We tried to impute the MOHSST dataset and compare the imputed SSTs with the observations in NOAASST to estimate the reconstruction error.

We treated the SST data of each month as one observation, where each observation is 36×72-dimensional (the SST were recorded in $5^\circ \times 5^\circ$ blocks on the longitude-latitude grid). We used our imputation routine to fit the MOHSST dataset with conventional PCA, probabilistic PCA and Bayesian PCA. The reconstruction error is estimated by the mean square error between the imputed SSTs in MOHSST dataset and the SSTs in NOAASST dataset.

Sadly, in this case, we failed to recover the SST. Figure 13 are two examples of the imputed SSTs. The left is the SST of July 1991 imputed by conventional PCA. We see that there are some red blocks indicating high temperature near the south pole, which is unlikely. We think the reason for this may be that information of SST near the south pole is scarce in the MOHSST dataset, and conventional PCA is overfitting with the few data points that we do have. Therefore, we used variational Bayesian PCA to imputed the data again. The reconstructed SST is shown on the right, which is no better than the reconstructed SST on the left. In fact, if we look at the reconstruction error in Figure 14, we can see that neither of our imputations performs better than the baseline model (which is to simply use the 1951-1980 climatological means as the imputed SSTs).

In retrospect, one of the reasons why our imputation model performs poorly is that we did not incorporate the information of time in our model, because by fitting a PCA on all the original data, we are assuming each monthly SST observation is drawn from the distribution, which can only capture the correlations of SSTs of different locations, and the SST's correlation with time is ignored. One way to address this problem is to split the original data into 12 groups (from January to December) and to fit a separate PCA model for each group. In this way, the seasonal difference in the distribution of SST will be accounted for. However, we will still lose the information of longer-term trend of SST. A further improvement will be to develop a time-series model for the principal components of the monthly SST. Another possible reason for the poor performance is that our implementation of probabilistic PCA and Bayesian PCA will only work, as we have shown, when the data is drawn from a multivariate Gaussian distribution. In the case of sea surface temperature, this property may not hold.

[1] https://iridl.ldeo.columbia.edu/SOURCES/.GOSTA/.atlas8/.MOHSST5/

[2] https://www.esrl.noaa.gov/psd/data/gridded/data.noaa.oisst.v2.html

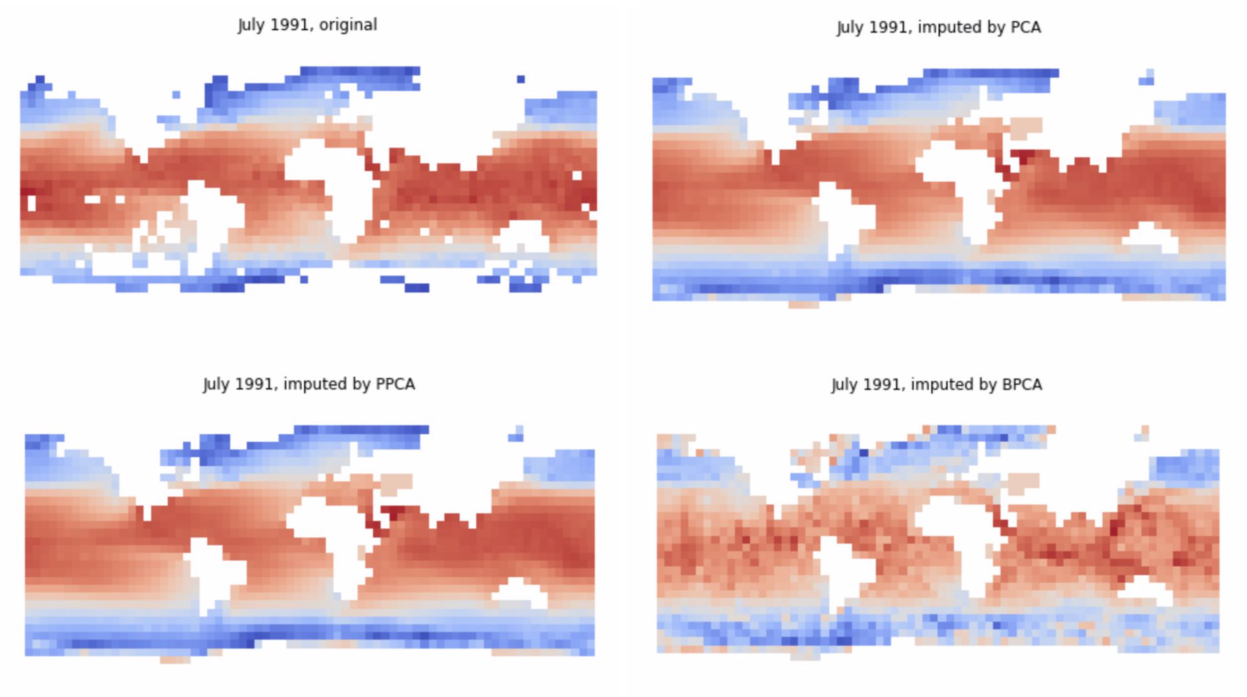

Figure 13: Sea surface temperature data imputed by PCA, probabilistic PCA and Bayesian PCA. The result of PCA and PPCA are the same. Note that there are some obvious errors near the south pole.

5 Discussion

In the imputation process, we notice the reconstruction error goes down each iteration. However, for imputation with Bayesian PCA, we only computed 5 iterations. The reason is that it took a very long time to finish a single iteration. This is in part because fixed-point iteration requires complicated calculations for parameters, such as the inverse of extremely high-dimensional matrices, and also has poor convergence rate. It is not acceptable in practice, compared with conventional PCA and Probabilistic PCA. There are some recent gradient-based optimization methods in Variational Inference, such as Black Box Variational Inference (BBVI) [Ranganath *et al.*, 2014] and Automatic Differentiation Variational Inference (ADVI) [Kucukelbir *et al.*, 2017]. We tried to implement ADVI for Bayesian PCA with *pymc3*, whereas due to limited time, unfortunately, we are not able to recover the original Bayesian PCA with the same function (automatically detect effective dimensionalities).

6 Conclusion

In this project, we implemented Probabilistic PCA and Bayesian PCA, elaborated mathematical derivations in previous papers, interpreted them with our intuitions and experimented them on several synthetic datasets. In most cases, Probabilistic PCA and Bayesian PCA generate similar results with conventional PCA. The difference is that Probabilistic PCA and Bayesian PCA are able to capture the underlying distribution of the data, which

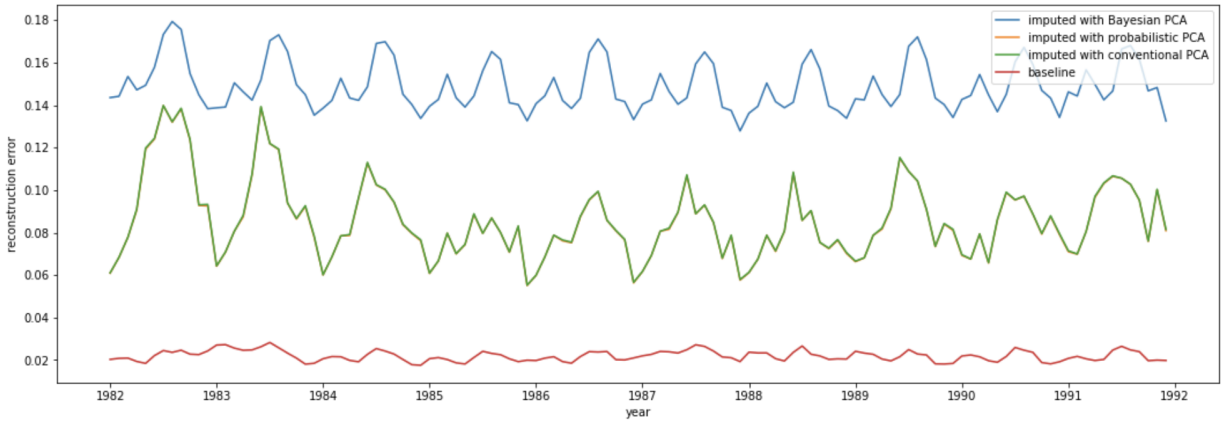

Figure 14: Comparing the reconstruction error of 3 imputation methods (probabilistic PCA and conventional PCA shares the same reconstruction error)

captures the uncertainty of the data, makes our PCA more robust and make it possible for data generation. Furthermore, introducing the hierarchical model and a new latent variable, Bayesian PCA can select the number of principal components automatically, thus lift away the job of discrete model search. However, we notice that our current implementation of Bayesian PCA is not efficient for high-dimensional datasets. Therefore, in future, to truly make use of the great properties of Bayesian PCA, we will try to adopt an efficient and effective gradient-based optimization method to approximate the posterior distribution.

References

Greg M Allenby and Peter E Rossi. Hierarchical bayes models. *The handbook of marketing research: Uses, misuses, and future advances*, pages 418–440, 2006.

Christopher M Bishop. Bayesian pca. In *Advances in neural information processing systems*, pages 382–388, 1999.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

Arjun K Gupta and Daya K Nagar. *Matrix variate distributions*. Chapman and Hall/CRC, 2018.

Alexander Ilin and Alexey Kaplan. Bayesian pca for reconstruction of historical sea surface temperatures. In *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, pages 1322–1327. IEEE, 2009.

S Ishii, K Matsubara, and M Monden. A bayesian missing value estimation method. *Bioinformatics*, 19:2088–2096, 2003.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *The Journal of Machine Learning Research*, 18(1):430–474, 2017.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11):2, 2011.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822, 2014.

Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

Appendix A Derivation of $q_i(\theta_i)$

With the mean-field family (24), (22) can be rewritten to

$$\begin{aligned}
\mathcal{L}(\mathcal{Q}) &= \int q(\boldsymbol{\theta}) \ln \frac{p(D,\boldsymbol{\theta})}{q(\boldsymbol{\theta})} d\boldsymbol{\theta} = \int q(\boldsymbol{\theta}) \ln p(D,\boldsymbol{\theta}) d\boldsymbol{\theta} - \int q(\boldsymbol{\theta}) \ln q(\boldsymbol{\theta}) d\boldsymbol{\theta} \\
&= \int \prod_i q_i(\theta_i) \ln p(D,\boldsymbol{\theta}) d\boldsymbol{\theta} - \int \prod_k q_k(\theta_k) \sum_k \ln q_k(\theta_k) d\boldsymbol{\theta} \\
&= \int q_i(\theta_i) \underbrace{\left\{ \int \prod_{k\neq i} q_k(\theta_k) \ln p(D,\boldsymbol{\theta}) d\boldsymbol{\theta}_{k\neq i} \right\}}_{\mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})]} d\theta_i \\
&\quad - \int q_i(\theta_i) \left\{ \int \prod_{k\neq i} q_k(\theta_k) \left[\ln q_i(\theta_i) + \sum_{j\neq i} \ln q_j(\theta_j) \right] d\boldsymbol{\theta}_{k\neq i} \right\} d\theta_i \\
&= \int q_i(\theta_i) \mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})] d\theta_i - \int q_i(\theta_i) \ln q_i(\theta_i) d\theta_i \int \prod_{k\neq i} q_k(\theta_k) d\boldsymbol{\theta}_{k\neq i} \\
&\quad - \int q_i(\theta_i) d\theta_i \int \prod_{k\neq i} q_k(\theta_k) \sum_{j\neq i} \ln q_j(\theta_j) d\boldsymbol{\theta}_{k\neq i}.
\end{aligned} \tag{49}$$

With the fact $\int q_i(\theta_i) d\theta_i = 1$ and $\int \prod_{k\neq i} q_k(\theta_k) d\boldsymbol{\theta}_{k\neq i} = \prod_{k\neq i} \int q_k(\theta_k) d\theta_{k\neq i} = 1$ (the integral of pdf should be 1), (49) is equivalent to

$$\mathcal{L}(\mathcal{Q}) = \int q_i(\theta_i) \mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})] d\theta_i - \int q_i(\theta_i) \ln q_i(\theta_i) d\theta_i + \text{const.} \tag{50}$$

Take the gradient of $\mathcal{L}(\mathcal{Q})$ in (50) with respect to θ_i,

$$\frac{\partial \mathcal{L}(\mathcal{Q})}{\partial \theta_i} = q_i(\theta_i) \mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})] - q_i(\theta_i) \ln q_i(\theta_i). \tag{51}$$

In order to maximize $\mathcal{L}(\mathcal{Q})$, (51) ought to be zero, so we get

$$q_i(\theta_i) = \exp\left\{ \mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})] \right\}. \tag{52}$$

In order to keep the invariant of $\int q_i(\theta_i) d\theta_i = 1$, we have to normalize $q_i(\theta_i)$, eventually we obtain (25)

$$q_i(\theta_i) = \frac{\exp\left\{ \mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})] \right\}}{\int \exp\left\{ \mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})] \right\} d\theta_i}. \tag{53}$$

In intuition, (50) can be reformulated to

$$\mathcal{L}(\mathcal{Q}) = \int q_i(\theta_i) \frac{\mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})]}{\ln q_i(\theta_i)} d\theta_i = \mathcal{KL}(q(\theta_i) || \exp\{\mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})]\}). \tag{54}$$

Therefore, maximizing $\mathcal{L}(\mathcal{Q})$ is equivalent to minimizing $\mathcal{KL}(q(\theta_i) || \exp\{\mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D,\boldsymbol{\theta})]\})$. Because the minimum of KL divergence is 0 if and only if two distributions are identical, we can also obtain (52).

Appendix B Details of Fixed-Point Update

Using (53) and taking ln term, we can get the equation

$$\ln q_i(\theta_i) = \mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\ln p(D, \boldsymbol{\theta})] = \mathbb{E}_{q(\boldsymbol{\theta})/q(\theta_i)}[\underbrace{\ln p(Y|\boldsymbol{\theta}, X)}_{\text{log-likelihood}} + \underbrace{\ln p(\boldsymbol{\theta})}_{\text{log-prior}}]. \tag{55}$$

With (55) and all parameters fixed except the desired one, we can update them iteratively. Specifically, for example, we intend to estimate $\mathbf{x}_n$, with (3) and (27)-(31), we have

$$\begin{aligned}
\ln q(\mathbf{x}_n) &= \mathbb{E}_{q(\mathbf{W},\boldsymbol{\alpha},\boldsymbol{\mu},\tau)}[\ln p(\mathbf{t}_n|\mathbf{x}_n, \mathbf{W}, \boldsymbol{\alpha}, \boldsymbol{\mu}, \tau)] \\
&\quad + \mathbb{E}_{q(\mathbf{W},\boldsymbol{\alpha},\boldsymbol{\mu},\tau)}[\ln p(\mathbf{x}_n)p(\mathbf{W}|\boldsymbol{\alpha})p(\boldsymbol{\alpha})p(\boldsymbol{\mu})p(\tau)] + \text{const.} \\
&= \ln \mathcal{N}(\mathbf{W}\mathbf{x}_n + \boldsymbol{\mu} - \mathbf{t}_n, \tau^{-1}\mathbf{I}) + \mathbb{E}_{q(\mathbf{W},\boldsymbol{\alpha},\boldsymbol{\mu},\tau)}[\ln p(\mathbf{x}_n)] + \text{const.} \\
&= -\frac{\tau}{2}(\mathbf{W}\mathbf{x}_n + \boldsymbol{\mu} - \mathbf{t}_n)^T(\mathbf{W}\mathbf{x}_n + \boldsymbol{\mu} - \mathbf{t}_n) - \frac{1}{2}\mathbf{x}_n^T\mathbf{x}_n + \text{const.} \\
&= -\frac{\tau}{2}\mathbf{x}_n^T\mathbf{W}^T\mathbf{W}\mathbf{x}_n - \tau\boldsymbol{\mu}^T\mathbf{W}\mathbf{x}_n + \tau\mathbf{t}_n^T\mathbf{W}\mathbf{x}_n - \frac{1}{2}\mathbf{x}_n^T\mathbf{x}_n + \text{const.} \\
&= -\frac{1}{2}\mathbf{x}_n^T(\tau\mathbf{W}^T\mathbf{W} + \mathbf{I})\mathbf{x}_n + \tau(\mathbf{t}_n^T - \boldsymbol{\mu}^T)\mathbf{W}\mathbf{x}_n + \text{const.}
\end{aligned} \tag{56}$$

Also, we assume that $q(\mathbf{x}_n)$ follows Gaussian distribution

$$\ln q(\mathbf{x}_n) = \ln \mathcal{N}(\mathbf{x}_n|\mathbf{m}_{\mathbf{x}}^{(n)}, \boldsymbol{\Sigma}_{\mathbf{x}}) = \frac{1}{2}\mathbf{x}_n^T\boldsymbol{\Sigma}_{\mathbf{x}}^{-1}\mathbf{x}_n + (\mathbf{m}_{\mathbf{x}}^{(n)})^T\boldsymbol{\Sigma}_{\mathbf{x}}^{-1}\mathbf{x}_n. \tag{57}$$

Note that in Bayesian inference, all parameters are stochastic, so here *"fixed" parameters are set with their expectation values*, just like what we do in (14). Armed with expectations and comparing (56) with (57), we can get (39) and (40).

While taking expectation $\langle\cdot\rangle$ for Gaussian distribution $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$,

$$\langle\mathbf{x}\rangle = \boldsymbol{\mu} \tag{58}$$

$$\langle\mathbf{x}\mathbf{x}^T\rangle = \boldsymbol{\mu}\boldsymbol{\mu}^T + \boldsymbol{\Sigma} \tag{59}$$

$$\langle\mathbf{x}^T\mathbf{x}\rangle = \boldsymbol{\mu}^T\boldsymbol{\mu} + \text{Tr}(\boldsymbol{\Sigma}) \tag{60}$$

and for Gamma distribution $\Gamma(x|a, b)$

$$\langle x\rangle = \frac{a}{b} \tag{61}$$

Likewise, assuming $q(\mathbf{w}_k)$ and $q(\boldsymbol{\mu})$ follow Gaussian distribution, and $q(\alpha_i)$ and τ follow Gamma distribution, we can obtain (41)-(48) imitating (56) and (57).