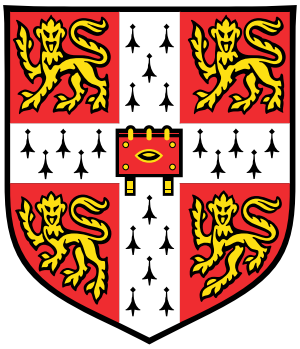
Auto-Encoding with Stochastic Expectation Propagation in Latent Variable Models



Vera Gangeskar Johne

Department of Engineering

University of Cambridge

This dissertation is submitted for the degree of

Master of Philosophy

Fitzwilliam College

August 2016

Declaration

I, Vera Gangeskar Johne of Fitzwilliam College, being a candidate for the M.Phil in Machine Learning, Speech and Language Technology, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose. Total word count: 4702.

Vera Gangeskar Johne
August 2016

Acknowledgements

Professor Richard Turner is known to my fellow students and I for his clear and insightful explanations of challenging concepts. Coming from a computer science background into the degree, I decided to try to do my thesis under Richard to get a more mathematically complete understanding of statistical machine learning. I am so glad I did, thank you Richard!

Along with Rich, I was lucky to be supervised by Yingzhen Li. Yingzhen has that ability of never making the listener feel silly for asking a question. I have learnt a lot from you Yingzhen, thank you!

As I reflect on the past year, my 22 fellow course-mates and I have bonded through this unique experience as the first MLSALT students. Thank you for friendship and a stimulating environment: Both important ingredients for productivity.

I would also like to thank two professors from my undergraduate years whose mentorship has been invaluable on my academic journey thus far. First, Bachelor thesis supervisor Steve Hails whom encouraged me to apply, and who introduced me to modeling in the unknown. Secondly, Christophe Dessimoz who co-supervised me during an internship at the European Bioinformatics Institute where I got my first taste of ML.

Hans, my brother, whom introduced me to computer programming back in the day, and who challenges me to think individually.

Most of all I want to say thank you to my parents who have always encouraged free thinking, and exploration of knowledge.

Abstract

Scalable Bayesian methods are becoming increasingly in demand. It is not uncommon to have datasets at peta- and exabyte scale. Moreover, datasets tend to be high dimensional and scattered with uncertainty. The Bayesian framework provides an elegant and principled system for reasoning under incomplete knowledge. We are interested in performing scalable inference and learning in directed probabilistic models where the latent space is both high dimensional and continuous, yielding an intractable posterior distribution. Seeing the success of variational methods in such settings through stochastic optimization and recognition models, a questions arises naturally: Can similar techniques work in the Expectation Propagation framework?

This thesis proposes a new inference technique which utilizes Stochastic Expectation Propagation for model learning, and a recognition model (inference network) for posterior inference over local latent variables, resulting in constant space complexity. We derive a new local objective function that leverages from stochastic optimization. Auto-encoding-with-Stochastic-Expectation-Propagation (AESEP) demonstrates signs of successful learning for factor analysis.

Contents

1 | Introduction

At the heart of many machine learning problems lies the task of pattern recognition: The task of unravelling hidden patterns in high-dimensional observations. One way to model this scenario is via writing a generative procedure. This involves assuming a latent space, and a set of generative rules relating the latent space to the observed space. For instance, a document (observed space) might be generated by first selecting a set of topics, then words from that topic (latent space) and combining them into sentences. In the literature such models are referred to as latent variable models (LVMs).

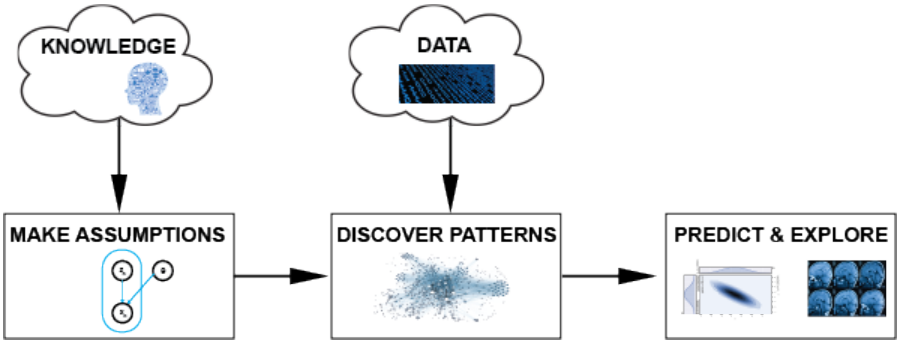


Figure 1.1: Probabilistic Pipeline.[1]

Often the goal in this scenario is inference, and the probabilistic framework offers a principled way for performing this task. In this pipeline (Figure 1.1), we use our prior knowledge to script a model, which is trained for discovering patterns with an available dataset. The trained model then finds its utility in making predictions about the unknown. The dynamics of this pipeline are governed by Bayesian probability, which has proven itself as the optimal language to reason under incomplete and uncertain information, MacKay (2003).

[1]Diagram inspired from Professor David Blei's talk *Scaling And Generalizing Approximate Bayesian Inference*, given 13/07/2016 in the department of Engineering, Cambridge University.

In practice, inference is often faced with the bottleneck of intractability. First is the analytical intractability, and second is the computational intractability. In the case of LVMs, these can stem from high-dimensional latent space, non-linear likelihoods, or large datasets.

In such circumstances, we must turn to approximate inference schemes. In fact, approximate inference techniques are often used in practice where exact inference is possible, to harvest their computational benefits. Over the past decade, the size of a typical dataset has dramatically escalated, and it is not uncommon to have streaming data (e.g. user clicks). This requires both quality analysis and computational efficiency. What is more, the goal is often decision making based on little information (e.g. Netflix movie suggestions to a new user). This has brought a wave of attention to scalable Bayesian learning, for example Stochastic Variational Inference, Hoffman et al. (2013), and Auto-encoding Variational Bayes (AEVB), Kingma and Welling (2013).

Approximate inference can be achieved by directly sampling from the intractable posterior (e.g. Markov Chain Monte Carlo), or through distributional approximations (e.g. Variational Inference and Expectation Propagation). Moreover, recent efforts have attempted to bridge the gap between the two schools of thought, Salimans et al. (2015). Variational Inference (VI) has generally accumulated more attention compared to Expectation Propagation (EP), Minka (2001c). Perhaps because VI casts Bayesian inference as an optimization problem yielding a well-defined lower bound to the model evidence. EP, on the other hand, iteratively approximates moments of factors that together form a global approximation through simpler local computations. It has been shown that variational methods are biased, and that EP outperforms VI in several applications, such as when likelihood functions are non-smooth (See Turner and Sahani (2008) and Turner and Sahani (2011)). This makes us eager to learn more about EP, and interesting research is already doing so (e.g. Li et al. (2016)).

There is however a hurdle with EP: each local approximation must be maintained in memory, leaving an $O(N)$ memory footprint. As a result, Stochastic Expectation Propagation (SEP), Li et al. (2015), was proposed. SEP maintains a global approximation to the posterior through parameter tying of the EP factors, while updating local approximations in the EP spirit. This makes SEP a good choice for Bayesian learning in the large data setting with its $O(1)$ space complexity.

Many notorious machine learning problems involve local latent variables (e.g. dimensionality reduction and topic modeling). AEVB successfully performed approximate bayesian variational inference using an auto-encoding map in latent variable models. However, in their

experiments they were not Bayesian about the global model parameters. This thesis combines SEP and AEVB into a new approximate inference scheme.

We propose doing Bayesian learning of the model parameters using SEP, and using an auto-encoding map (recognition model) to approximate the posterior over the latent variables.

The trajectory of this thesis is as follows. First the general problem scenario is mapped out by presenting a generic graphical model and a concrete set of goals. Chapter 2 highlights important theory that has been successful in the VI framework, which we braid into the proposed scheme. Chapter 3 summarizes traditional EP, and introduces SEP. Finally chapter 5 presents Auto-encoding-with-SEP (AESEP) and derives its objective function. In chapter 6 experiments and evaluation is done. We end with suggestions for the direction of future work.

1.1 Problem Scenario

We consider a generic class of models[2] with the dataset $\boldsymbol{x_{1:N}}$, where the N samples are independently and identically distributed (i.i.d.), and $\boldsymbol{x_n}$ is either discrete or continuous. The observed variables are generated by some process with latent variable $\boldsymbol{z_n}$ per datapoint, and a set of global parameters, $\boldsymbol{\theta}$. The local latent variables typically represents some hidden structure in our data, while the global parameters denote the model parameters. This structure is depicted in the graphical model in Figure 1.2.

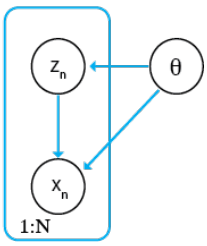


Figure 1.2: Graphical model for class of models under consideration. The model has observations $\boldsymbol{x_{1:N}}$, local latent variables $\boldsymbol{z_{1:N}}$, and global latent variables $\boldsymbol{\theta}$. The arrows represent the generative model.

The generative procedure can be described in two steps: $\boldsymbol{z_n}$ is generated from a prior dis-

[2]Many well-known machine learning problems have graphical models of this form, e.g. Bayesian mixture models, matrix factorization (Factor analysis, PCA), Dirichlet process mixtures, latent dirichlet allocation, multilevel regression etc.

tribution $p(\boldsymbol{z_n}|\boldsymbol{\theta})$, then $\boldsymbol{x_n}$ is generated from a conditional distribution $p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{\theta})$. The probabilistic model can be written as:

$$p(\boldsymbol{\theta}, \boldsymbol{z}, \boldsymbol{x}) = p(\boldsymbol{\theta}) \prod_{n=1}^{N} p(\boldsymbol{z_n}, \boldsymbol{x_n}|\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{n=1}^{N} p(\boldsymbol{z_n}|\boldsymbol{\theta}) p(\boldsymbol{x_n}|\boldsymbol{\theta}, \boldsymbol{z_n}) \tag{1.1}$$

Both the global and local latent variables are unknown. Bayes' rule is applied to do posterior inference over the latent space:

$$p(\boldsymbol{z_{1:N}}, \boldsymbol{\theta}|\boldsymbol{x_{1:N}}) = \frac{p(\boldsymbol{\theta}) \prod_{n=1}^{N} p(\boldsymbol{z_n}|\boldsymbol{\theta}) p(\boldsymbol{x_n}|\boldsymbol{\theta}, \boldsymbol{z_n})}{p(\boldsymbol{x_{1:N}})}$$

The computation of the marginal likelihood, $p(\boldsymbol{x_{1:N}})$, involves integrating out the latent space, and is commonly a source of intractability as discussed in the introduction. To perform this task, we propose an approximate inference scheme with the following goals:

1. Efficient Approximate Bayesian learning of the model parameters $\boldsymbol{\theta}$.
2. Efficient approximate posterior inference on the local latent variables $\boldsymbol{z}$ given an observed value $\boldsymbol{x}$.

In AESEP, SEP is used to achieve goal 1., and recognition models for goal 2.

2 | Variational Inference

In this chapter exciting research that has lead to improved approximate posterior inference in the VI framework is introduced. In particular we discuss the reparameterization trick, Papaspiliopoulos et al. (2007), for improved gradient estimates, and recognition models for efficient approximate posterior inference, Kingma and Welling (2013) . Both techniques are employed in the proposed system.

VI phrases Bayesian inference as an optimization problem. An approximate posterior distribution, $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, is fit to the exact posterior, $p_\theta(\boldsymbol{z}|\boldsymbol{x})$, by optimizing the approximation's parameters, $\boldsymbol{\phi}$, such that the lower bound, $\mathcal{L}$, of the marginal likelihood is maximized. The marginal log-likelihood is composed of the individual datapoint likelihood terms, $\log p(X) = \sum_{n=1}^{N} \log p(\boldsymbol{x_n})$. Each individual term can be expressed in the form:

$$\log p(\boldsymbol{x_n}) = KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x})) + \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x_n})$$

Where the KL term is the Kullback-Leibler (KL) divergence between two distributions. The variational lower bound is derived in appendix A, and the KL-divergence is also defined. The result is the following form of the variational lower bound (also known as the ELBO):

$$\log p(\boldsymbol{x_n}) \geq \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x_n}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\Big[\log p_\theta(\boldsymbol{x}, \boldsymbol{z}) - q_\phi(\boldsymbol{z}|\boldsymbol{x})\Big]$$

This optimization requires gradients with respect to both the generative parameters, $\boldsymbol{\theta}$, and the variational parameters, $\boldsymbol{\phi}$. The gradients with respect to the variational parameters becomes challenging where naive Monte Carlo (MC) estimates of the gradients yield high variance, Paisley et al. (2012). Similar issues are faced in AESEP when taking gradients with respect to the recognition model and SEP parameters.

2.1 Use of Reparameterization Trick

The reparameterization trick is a strategy for generating samples from a distribution, e.g. $z \sim q(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\phi})$. The random variable $\boldsymbol{z}$ can be reparameterized using a differentiable transform: $\tilde{\boldsymbol{z}} = g(\boldsymbol{\epsilon}, \boldsymbol{x}|\boldsymbol{\phi})$. The noise variable, $\boldsymbol{\epsilon}$, is independent from $\boldsymbol{\phi}$, and is sampled from some prior distribution: $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$. The idea is that the mapping, $g(.)$, is differentiable with respect to both $\boldsymbol{\phi}$ and $\boldsymbol{\epsilon}$. For example, suppose $q(\boldsymbol{z}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and let $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, then the transformation becomes:

$$\boldsymbol{z} = g(\boldsymbol{\epsilon}, \boldsymbol{x}|\boldsymbol{\phi}) = \boldsymbol{\mu} + \boldsymbol{\Sigma}^{\boldsymbol{1/2}}\boldsymbol{\epsilon} \tag{2.1}$$

The differentiable transform depends on the nature of the approximate distribution q. Intuitively, this is simply a shift in the source of stochasticity to $\boldsymbol{\epsilon}$; merely substitution in terms of calculus. However, some advantageous consequences appear when computing samples of the gradients. Suppose we want to sample from q with respect to a function of $\boldsymbol{z}$, $f(\boldsymbol{z})$, then by the chain rule this becomes a derivative with respect to $\boldsymbol{\epsilon}$:

$$\nabla_{\phi}\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x_n},\boldsymbol{\phi})}[f(\boldsymbol{z})] = \nabla_{\epsilon}\mathbb{E}_{p(\boldsymbol{\epsilon})}[f(g(\boldsymbol{\epsilon}, \boldsymbol{x_n}|\boldsymbol{\phi}))]$$

The idea of variance reduction is to modify a function of a random variable, $\boldsymbol{z_n}$, such that the variance decreases, but the expectations stays the same, Paisley et al. (2012). The reparameterization trick reduces the variance of the samples, especially if the differential transform has an isotropic covariance. The mechanics of the calculus above is a result of the deterministic mapping which explicitly results in the substitution: $q(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\phi})d\boldsymbol{z} = p(\boldsymbol{\epsilon})d\boldsymbol{\epsilon}$. From this it follows that:

$$\int q(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\phi})f(\boldsymbol{z})d\boldsymbol{z} = \int p(\boldsymbol{\epsilon})f(\boldsymbol{z})d\boldsymbol{\epsilon} = \int p(\boldsymbol{\epsilon})f(g(\boldsymbol{\epsilon}, \boldsymbol{x}|\boldsymbol{\phi}))d\boldsymbol{\epsilon}$$

The reparameterization trick has been applied in the variational setting in a vast amount of published literature. AEVB shows explicitly how it is used to form a more accurate approximation of the ELBO, such that the MC estimate is differentiable with respect to the variational parameters. AESEP utilizes this trick in each local minimization for both the local and global latent variables.

2.2 Use of Recogntion Models

One of the stated goals is efficient approximate inference of the local latent variables. The implicit dependency between an observation $\boldsymbol{x_n}$ and corresponding $\boldsymbol{z_n}$ (See Figure 1.2) invites the use of a recognition model, $q(\boldsymbol{z_{1:N}}|\boldsymbol{x_{1:N}}, \boldsymbol{\phi})$, to approximate the exact posterior $p(\boldsymbol{z_{1:N}}|\boldsymbol{x_{1:N}}, \boldsymbol{\theta})$.

This framework has the advantage of bringing flexibility to the approximation by putting few constraints on the recognition model parameters $\boldsymbol{\phi}$. In contrast, the common mean field approximation[1], which assumes a factorized posterior, not only constraints the posterior but also requires parameters for each factor to be stored in memory. Therefore, the recognition model can also be thought of as parameter tying.

To achieve a deterministic mapping from an observation $\boldsymbol{x_n}$ to $\boldsymbol{z_n}$, AEVB makes use of the reparameterization trick (Figure 2.1b.). For example, if the recognition model is assumed Gaussian $q(\boldsymbol{z_n}|\boldsymbol{x_n}, \boldsymbol{\phi}) = \mathcal{N}(\boldsymbol{\mu_n}, \boldsymbol{\Sigma_n})$, then the output of the deterministic map given an observation, $\boldsymbol{x_n}$, is $\boldsymbol{\mu_n}$ and $\boldsymbol{\Sigma_n}$. Given the Gaussian assumed structure, the reparameterization in Equation 2.1 can be utilized to deterministically map $\boldsymbol{x_n}$ to $\boldsymbol{z_n}$.

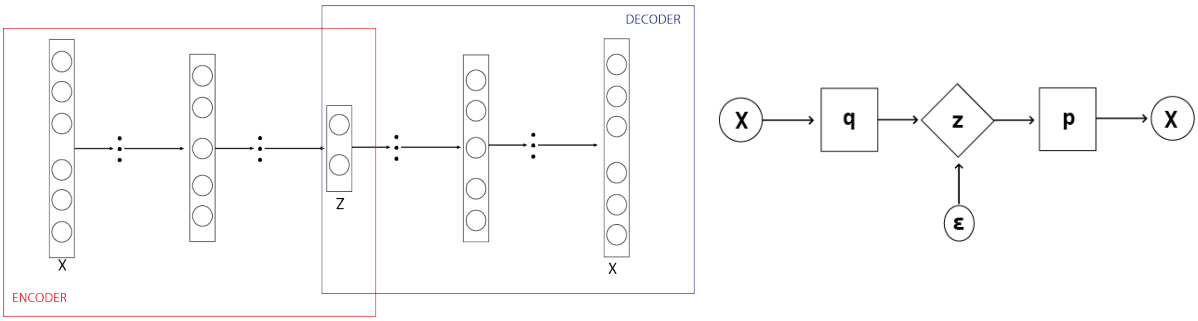


Figure 2.1: a. Autoencoder showing the mapping and terminology used. b. Reparameterization trick's role in the encoding/decoding process, where the kite-shaped square around $\boldsymbol{z}$ implies the random variable is deterministic. Figure b. is inspired by D. Kingma's talk on AEVB, Kingma (2014).

AEVB gives a clear picture of the system through the use of autoencoders (Figure 2.1a.). The deterministic encoding map (e.g. observation to mean and variance) is a neural network, and likewise is the decoding map. Therefore the only parameters necessary are the weights of the network. In the general framework, any regression map could be utilized. To do approximate

[1] $q(\boldsymbol{z_{1:N}}) = \prod_{n=1}^{N} \boldsymbol{q_n}(\boldsymbol{z_n})$

VI, AEVB utilizes this to evaluate the variational lower bound. Specifically, once an observation $\boldsymbol{x_n}$ has been mapped to a latent variable $\boldsymbol{z_n}$, this is used to evaluate the likelihood of that particular observation which is necessary for the evaluation of the ELBO.

3 | Expectation Propagation

This chapter starts by outlining traditional EP, and explains how EP is able to achieve quality approximations of the posterior. Following last chapter's brief introduction to VI, the contrast between the two becomes clear. Once the EP framework is established, SEP is introduced which is how learning is done in the proposed system.

3.1 EP

Expectation propagation approximates the unnormalized posterior, or the joint distribution, and returns a tractable normalized posterior. For convenience, let $*$ denote unnormalized densities. Suppose we have a dataset, X, and $\boldsymbol{\theta}$ is a vector containing all latent variables, both global and/or local, then the intractable exact posterior, $p(\boldsymbol{\theta}|X)$, can be approximated with a tractable approximate distribution $q(\boldsymbol{\theta})$:

$$p(\boldsymbol{\theta}, X) = p(\boldsymbol{\theta}) \prod_{n=1}^{N} p(\boldsymbol{x_n}|\boldsymbol{\theta}) \approx q^*(\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{n=1}^{N} f_n(\boldsymbol{\theta})$$

Each likelihood term is approximated by a corresponding factor, $f_n(\boldsymbol{\theta})$. Often these factors are constrained to be members of a chosen exponential family. The result is a posterior in the same family[1].

Like variational inference, EP too, is based on minimizing the KL- divergence. However, EP obtains its personality by taking the KL between the exact posterior and the approximate posterior: the reverse form compared with the KL used in VI.

$$KL\Big(p(\boldsymbol{\theta}|X)\Big|\Big|q(\boldsymbol{\theta})\Big) = \int p(\boldsymbol{\theta}|X) log \frac{p(\boldsymbol{\theta}|X)}{q(\boldsymbol{\theta})} d\boldsymbol{\theta} = \mathbb{E}_{p(\boldsymbol{\theta}|X)}\left[log \frac{p(\boldsymbol{\theta}|X)}{q(\boldsymbol{\theta})}\right]$$

[1] The product of distributions of the same exponential family is also a member of that family.

The above minimization is intractable since taking an expectation with respect to the intractable exact posterior is analytically impossible. EP gets around this by iteratively optimizing each factor in the context of all the other factors, using what is referred to as the cavity distribution: The product of all factors (including the prior), except for the current factor, n, being updated. The cavity distribution for factor n is:

$$q\_{n}(\boldsymbol{\theta}) = \frac{q(\boldsymbol{\theta})}{f_n(\boldsymbol{\theta})}$$

The local minimization for a factor n, becomes the following. $p(\boldsymbol{x_n}|\boldsymbol{\theta})q\_{n}(\boldsymbol{\theta})$ is referred to as the tilted distribution.

$$KL\Big(\frac{p(\boldsymbol{x_n}|\boldsymbol{\theta})q\_{n}(\boldsymbol{\theta})}{Z_n})\Big|\Big|f_n(\boldsymbol{\theta})q\_{n}(\boldsymbol{\theta})\Big) \tag{3.1}$$

The tilted distribution's presence in the local minimizations is fundamental to EP's accurate global approximations. To offer clairvoyance, one can instead imagine an iterative procedure where the KL-divergence is computed directly between $p(\boldsymbol{x_n}|\boldsymbol{\theta})$ and $f_n(\boldsymbol{x})$. The product of n such entirely independent minimizations would give a globally unreliable approximation. It might seem that one local minimization does not corporate any information about the other $N-1$ minimizations, but this is not the case as the cavity distribution captures the current estimate from the $N-1$ observations. In other words, the advantage of the cavity distribution arrises due to the fact that the cavity defines the areas where the posterior probability is high. Figure 3.1 visualizes the role played by the tilted and cavity distribution in each likelihood approximation.

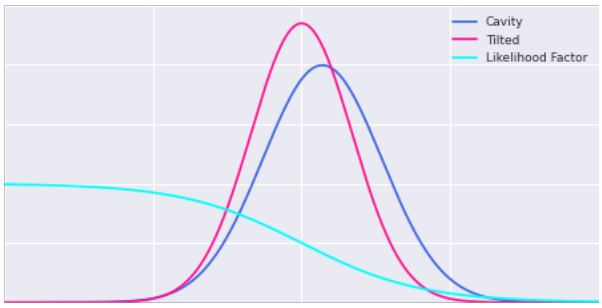


Figure 3.1: Example of EP update visualising the stability EP gets from the cavity distribution in the KL minimization. In this case the likelihood factors are assumed Gaussian. Observe that even though the likelihood terms do not resemble a Gaussian, the cavity distribution ensures that the approximation is more accurate where the posterior has a high value. Figure idea taken from Gelman et al. (2014).

Furthermore, when the approximating factors are members of the exponential family the minimization (Equation 3.1) reduces to moment matching of the moments of the tilted distribution. The proof of this result is provided in appendix B. The result in the EP minimization is the following projection of moments:

$$f_n(\boldsymbol{\theta})q\_{}_n(\boldsymbol{\theta}) \leftarrow \texttt{proj}[p(\boldsymbol{x_n}|\boldsymbol{\theta})q\_{}_n(\boldsymbol{\theta})]$$

Finally, the updated factor is found by dividing out the cavity distribution. In the rare case where the exact likelihood term and the approximating factor are in the same family, moment matching is trivial (For an example see chapter 5.1). Usually the likelihood takes a complicated form, and moments must be sampled from the exact likelihood term $p(\boldsymbol{x_n}|\boldsymbol{\theta})$. The algorithmic steps of EP can be summarized:

1. **Initialization.** All factors are initialized. Usually restricted to be members of an exponential family, e.g. $f_n(\boldsymbol{\theta}) \longleftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$. Also initialize the unnormalized posterior: $q^*(\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{n=1}^{N} f_n(\boldsymbol{\theta})$
2. **Iteration.** Until convergence:
   (a) **Cavity.** Compute the cavity distribution:
   $q\_{}_n(\boldsymbol{\theta}) = \frac{q(\boldsymbol{\theta})}{f_n(\boldsymbol{\theta})}$
   (b) **Tilted.** Compute the tilted distribution:
   $p(\boldsymbol{x_n}|\boldsymbol{\theta})q\_{}_n(\boldsymbol{\theta})$
   (c) **Moment matching**.
   $q^{new}(\boldsymbol{\theta}) = f_n(\boldsymbol{\theta})q\_{}_n(\boldsymbol{\theta}) \leftarrow \texttt{proj}[p(\boldsymbol{x_n}|\boldsymbol{\theta})q\_{}_n(\boldsymbol{\theta})]$
   (d) **Normalisation**. Compute the marginal likelihood of the tilted:
   $Z_n = \int p(\boldsymbol{x_n}|\boldsymbol{\theta})q\_{}_n(\boldsymbol{\theta})d\boldsymbol{\theta}$
   (e) **Update**.Update the new factor and store
   $f_n(\boldsymbol{\theta}) = Z_n \frac{q^{new}(\boldsymbol{\theta})}{q_n(\boldsymbol{\theta})}$

The local nature of EP makes it easy to parallelize and distribute. Moreover, the local computations are cheap compared with a more complex global minimization. Unfortunately, there are two main drawbacks to EP. First is the $O(N)$ memory footprint as a result of the cavity's requirement to store the N approximating factors in memory. Second, traditional EP has no guarantee of convergence, Minka (2001a).

3.2 SEP

The beauty of EP's local computations has a critical limitation: The number of approximating factors linearly increase as the number of observations increase[2]. This becomes an issue when doing approximate bayesian parameter learning over large datasets. As a reference point, VI does not face this type of memory scaling because it maintains a global approximation. SEP also maintains a global approximation, while updating the global approximation in a local manner, resulting in $O(1)$ space complexity.

The eventual goal of EP is to summarize the posterior through the likelihood factors. In its optimization it is assumed that the contribution of each likelihood term would be averaged out at convergence. SEP directly assumes this through parameter tying:

$$f(\boldsymbol{\theta})^N \triangleq \prod_{n=1}^{N} f_n(\boldsymbol{\theta}) \approx \prod_{n=1}^{N} p(\boldsymbol{x_n}|\boldsymbol{\theta})$$

Explicitly, only $f(\boldsymbol{\theta})$ is maintained, and the unnormalized posterior becomes:

$$q(\boldsymbol{\theta}) \propto f(\boldsymbol{\theta})^N p(\boldsymbol{\theta})$$

SEP uses the same iterative local KL-minimization as EP to learn the current datapoint's approximating factor $f_n(\boldsymbol{\theta})$. However, the minimization is in the context of the averaged cavity, $q_{-1}(\boldsymbol{\theta}) \propto \frac{q(\boldsymbol{\theta})}{f(\boldsymbol{\theta})}$: summarizing the average contribution of each datapoint to the posterior.

$$KL\Big(\frac{p(\boldsymbol{x_n}|\boldsymbol{\theta})q_{-1}(\boldsymbol{\theta})}{Z_n})\Big|\Big|p(\boldsymbol{\theta})f(\boldsymbol{\theta})^N)\Big)$$

To achieve a local approximation similarly perform moment matching and divide out the cavity:

$$f_n(\boldsymbol{\theta}) \longleftarrow \texttt{proj}[q_{-1}(\boldsymbol{\theta})p(\boldsymbol{x_n}|\boldsymbol{\theta})]/q_{-1}(\boldsymbol{\theta})$$

Finally, the global factor is updated using damping:

$$f(\boldsymbol{\theta}) = f(\boldsymbol{\theta})^{1-\alpha} f_n(\boldsymbol{\theta})^{\alpha}$$

Setting the damping parameter $\alpha = \frac{1}{N}$ is the most obvious choice, but the SEP authors also suggests other choices, such as an adaptive damping parameter, and decreasing α according to the Robbins-Monroe condition.

[2]Prior to SEP, solving this required a visit to the hardware store or the cloud.

The algorithmic steps are very similar to the detailed outline given in the EP section, with the SEP updates as explained. The algorithm is summarized in Figure 3.2, and the contrast to EP can be seen. In particular take note of step 2, the computation of the cavity, clearly showing how SEP uses the averaged contribution of each datapoint. Theoretically, this is a compromise to the quality of the approximation, however the gain in terms of resources by changing to SEP from EP might often be more dramatic. The authors achieve impressive results on both synthetic and real datasets.

Algorithm 1 EP

1: choose a factor f_n to refine:
2: compute cavity distribution
$q_{-n}(\boldsymbol{\theta}) \propto q(\boldsymbol{\theta})/f_n(\boldsymbol{\theta})$
3: compute tilted distribution
$\tilde{p}_n(\boldsymbol{\theta}) \propto p(\boldsymbol{x}_n|\boldsymbol{\theta})q_{-n}(\boldsymbol{\theta})$
4: moment matching:
$f_n(\boldsymbol{\theta}) \leftarrow \texttt{proj}[\tilde{p}_n(\boldsymbol{\theta})]/q_{-n}(\boldsymbol{\theta})$
5: inclusion:
$q(\boldsymbol{\theta}) \leftarrow q_{-n}(\boldsymbol{\theta})f_n(\boldsymbol{\theta})$

Algorithm 2 SEP

1: choose a datapoint $\boldsymbol{x}_n \sim \mathcal{D}$:
2: compute cavity distribution
$q_{-1}(\boldsymbol{\theta}) \propto q(\boldsymbol{\theta})/f(\boldsymbol{\theta})$
3: compute tilted distribution
$\tilde{p}_n(\boldsymbol{\theta}) \propto p(\boldsymbol{x}_n|\boldsymbol{\theta})q_{-1}(\boldsymbol{\theta})$
4: moment matching:
$f_n(\boldsymbol{\theta}) \leftarrow \texttt{proj}[\tilde{p}_n(\boldsymbol{\theta})]/q_{-1}(\boldsymbol{\theta})$
5: inclusion:
$q(\boldsymbol{\theta}) \leftarrow q_{-1}(\boldsymbol{\theta})f_n(\boldsymbol{\theta})$
6: *implicit update*:
$f(\boldsymbol{\theta}) \leftarrow f(\boldsymbol{\theta})^{1-\frac{1}{N}} f_n(\boldsymbol{\theta})^{\frac{1}{N}}$

Figure 3.2: Comparing EP and SEP algorithms. The procedure is done iteratively until convergence. In step one, EP chooses a factor to refine, while SEP chooses a datapoint (as local factors are not kept in memory) to compute the local minimization for updating the global approximation.

4 | Auto-encoding with SEP

This chapter introduces the proposed approximate inference scheme: Auto-encoding with SEP. First, we motivate the EP-like approximations. We incorporate parameter tying strategies to define a new set of approximations. Then we derive a local objective function by minimizing a KL-divergence in an EP-like fashion. Following this we propose the use of a sampling based stochastic optimization of the objective function.

4.1 Approximating Factors and Parameter Tying

The Problem Scenario introduced the generic probabilistic model (Equation 1.1) being considered. Mathematically, the joint likelihood $p(\boldsymbol{z_n}, \boldsymbol{x_n}|\boldsymbol{\theta})$ is a function $t_n(\boldsymbol{x_n}, \boldsymbol{z_n}, \boldsymbol{\theta})$ with parameters $\boldsymbol{x_n}, \boldsymbol{z_n}$ and $\boldsymbol{\theta}$. In alignment with the EP philosophy, we approximate $t_n(\boldsymbol{x_n}, \boldsymbol{z_n}, \boldsymbol{\theta})$ with $\widetilde{t_n}(\boldsymbol{x_n}, \boldsymbol{z_n}, \boldsymbol{\theta})$.

$$p(\boldsymbol{z_n}|\boldsymbol{\theta})p(\boldsymbol{x_n}|\boldsymbol{\theta}, \boldsymbol{z_n}) = t_n(\boldsymbol{x_n}, \boldsymbol{z_n}, \boldsymbol{\theta}) \approx \widetilde{t_n}(\boldsymbol{x_n}, \boldsymbol{z_n}, \boldsymbol{\theta}) = f_n(\boldsymbol{z_n})\widetilde{t_n}(\boldsymbol{\theta})$$

This can further be approximated using SEP parameter tying on the $\widetilde{t_n}(\boldsymbol{\theta})$ factors, such that $\widetilde{t}(\boldsymbol{\theta})^N = \prod_{n=1}^{N} \widetilde{t_n}(\boldsymbol{\theta})$, resulting in the following expression:

$$\widetilde{t_n}(\boldsymbol{x_n}, \boldsymbol{z_n}, \boldsymbol{\theta}) = f_n(\boldsymbol{z_n})\widetilde{t}(\boldsymbol{\theta})$$

At this stage each $f_n(\boldsymbol{z_n})$ could be approximated using EP, yielding an SEP-EP model. We tie parameters at one more level. Intuitively there is a local dependency between the observed $\boldsymbol{x_n}$ and the corresponding latent variable $\boldsymbol{z_n}$. Unlike the traditional EP approach, where each f_n factor would have separate parameters, $\boldsymbol{\lambda_n}$, we can exploit this intuitive relationship via an encoder map (like in chapter 2.2). Such that $f_n(\boldsymbol{z_n}) = f(g_\phi(\boldsymbol{x_n}))$. For convenience we write $f(\boldsymbol{z_n})$, to represent the deterministic mapping from $\boldsymbol{x_n}$ to $\boldsymbol{z_n}$ via a recogntion model utilizing the reparameterization trick. This makes the final approximation with both sources

of parameter tying:

$$p(\boldsymbol{z_n}|\boldsymbol{\theta})p(\boldsymbol{x_n}|\boldsymbol{\theta},\boldsymbol{z_n}) = t_n(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta}) \approx \widetilde{t_n}(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta}) = f(\boldsymbol{z_n})\widetilde{t}(\boldsymbol{\theta})$$

The complete approximation can be expressed as:

$$p(\boldsymbol{z_{1:N}},\boldsymbol{x_{1:N}},\boldsymbol{\theta}) = p(\boldsymbol{\theta})\prod_{n=1}^{N} p(\boldsymbol{z_n}|\boldsymbol{\theta})p(\boldsymbol{x_n}|\boldsymbol{\theta},\boldsymbol{z_n}) \approx p(\boldsymbol{\theta})\widetilde{t}(\boldsymbol{\theta})^N \prod_{n=1}^{N} f(\boldsymbol{z_n})$$

4.2 Local KL-divergence

The EP minimization step takes the KL-divergence between the tilted distribution, and the unnnormalised approximate posterior. These densities are expressed respectively:

$$q(\boldsymbol{\theta},\boldsymbol{z_{1:N}}) \propto p(\boldsymbol{\theta})\prod_n \widetilde{t_n}(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta}) \tag{4.1}$$

$$q^{tilt}(\boldsymbol{\theta},\boldsymbol{z_{1:N}}) = \frac{1}{Z}\frac{q(\boldsymbol{\theta},\boldsymbol{z_{1:N}})}{\widetilde{t_n}(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})}t_n(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta}) \tag{4.2}$$

Z is the normalization constant, which ensures that $q^{tilt}(\boldsymbol{\theta},\boldsymbol{z_{1:N}})$ integrates to unity. By taking the KL-divergence between Equation 4.1 and 4.2, a local objective, E, is derived.

$$\begin{aligned}
E &= KL\Big(q^{tilt}(\boldsymbol{\theta},\boldsymbol{z_{1:N}})\Big|\Big|q(\boldsymbol{\theta},\boldsymbol{z_{1:N}})\Big) \\
&= \int_{\dots}\int \frac{1}{Z}\frac{q(\boldsymbol{\theta},\boldsymbol{z_{1:N}})}{\widetilde{t_n}(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})}t_n(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})\log\frac{\frac{1}{Z}\frac{q(\boldsymbol{\theta},\boldsymbol{z_{1:N}})}{\widetilde{t_n}(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})}t_n(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})}{q(\boldsymbol{\theta},\boldsymbol{z_{1:N}})}d\boldsymbol{\theta}d\boldsymbol{z_{1:N}} \\
&= \int d\boldsymbol{z_{\neq n}}\int\int \frac{1}{Z}\Big[\prod_{m\neq n}\widetilde{t_n}(\boldsymbol{x_m},\boldsymbol{z_m},\boldsymbol{\theta})\Big]p(\boldsymbol{\theta})t_n(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})\log\frac{t_n(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})}{Z\widetilde{t_n}(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})}d\boldsymbol{\theta},\boldsymbol{z_n} \\
&= \int\int \frac{1}{Z}\widetilde{t}(\boldsymbol{\theta})^{N-1}p(\boldsymbol{\theta})p(\boldsymbol{z_n},\boldsymbol{x_n}|\boldsymbol{\theta})\log\frac{p(\boldsymbol{z_n},\boldsymbol{x_n}|\boldsymbol{\theta})}{Zf_n(\boldsymbol{z_n})\widetilde{t}(\boldsymbol{\theta})}d\boldsymbol{\theta}d\boldsymbol{z_n} \\
&= \int\int \frac{1}{Z}\widetilde{t}(\boldsymbol{\theta})^{N-1}p(\boldsymbol{\theta})p(\boldsymbol{z_n},\boldsymbol{x_n}|\boldsymbol{\theta})\frac{\Gamma(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})}{\Gamma(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})}\log\Gamma(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})d\boldsymbol{\theta}d\boldsymbol{z_n} \\
&= \int\int \frac{1}{Z}\widetilde{t}(\boldsymbol{\theta})^{N-1}p(\boldsymbol{\theta})p(\boldsymbol{z_n},\boldsymbol{x_n}|\boldsymbol{\theta})\frac{Zf_n(\boldsymbol{z_n})\widetilde{t}_n(\boldsymbol{\theta})}{p(\boldsymbol{x_n},\boldsymbol{z_n}|\boldsymbol{\theta})}\Gamma(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})\log\Gamma(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})d\boldsymbol{\theta}d\boldsymbol{z_n} \\
&= \int\int \widetilde{t}(\boldsymbol{\theta})^{N}p(\boldsymbol{\theta})f_n(\boldsymbol{z_n})\Gamma(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})\log\Gamma(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})d\boldsymbol{\theta}d\boldsymbol{z_n} \\
&= \int\int q(\boldsymbol{\theta})f_n(\boldsymbol{z_n})\Gamma(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})\log\Gamma(\boldsymbol{x_n},\boldsymbol{z_n},\boldsymbol{\theta})d\boldsymbol{\theta}d\boldsymbol{z_n}
\end{aligned}$$

Γ is the following expression:

$$\Gamma(\boldsymbol{x_n}, \boldsymbol{z_n}, \boldsymbol{\theta}) = \frac{p(\boldsymbol{x_n}, \boldsymbol{z_n}|\boldsymbol{\theta})}{Z f_n(\boldsymbol{z_n}) \widetilde{t_n}(\boldsymbol{\theta})} \tag{4.3}$$

4.3 Stochastic Objective

The objective can be approximated by employing sampling schemes. The unobserved variables are now sampled, $\{\boldsymbol{\theta^{(m)}}, \boldsymbol{z_n^{(m)}}\}_{m=1}^{M} \sim q(\boldsymbol{\theta}) f(\boldsymbol{z_n})$, such that $\boldsymbol{\theta}$ is sampled from the SEP posterior approximation, and $\boldsymbol{z_n}$ from the recognition model approximation. The samples are taken via the reparameterization trick, to achieve benefits (see chapter 2.1) when computing the gradients with respect to both the SEP and recognition model parameters. Approximating E via sampling results in:

$$E \approx \widetilde{E} = \frac{1}{M} \sum_{m=1}^{M} \Gamma(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}}) \log \Gamma(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}}) \tag{4.4}$$

In a similar manner, the normalization constant in Equation 4.3 can be approximated from the unnormalized Γ:

$$Z \approx \frac{1}{M} \sum_{m=1}^{M} \Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}}) \tag{4.5}$$

At this stage, the tradeoff between a biased and an unbiased estimate becomes relevant. Note that an unbiased estimate of Z, does not mean that $\frac{1}{Z}$ is unbiased. Theoretical work on unbiased estimates of $\frac{1}{Z}$ exists. To achieve an unbiased estimate for the final approximation, separate sets of samples must be used in the approximation of Z and $\Gamma*$. Unfortunately, this leads to a resulting approximation with high variance. Intuitively, the two sets invite two sources of randomness, which leads to the observed increased variance. Self normalized important sampling reduces the variance by using one set of samples for both approximations.

Following Equation 4.4, we utilize this fact which results in the following biased estimate:

$$\widetilde{E} = \frac{1}{M}\sum_{m=1}^{M}\frac{1}{Z}\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}}) \log \frac{\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})}{Z}$$

$$= \frac{1}{M}\sum_{m=1}^{M}\frac{\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})}{Z}\Big[\log \Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}}) - \log Z\Big]$$

Plugging in for Z using Equation 4.5:

$$= \frac{1}{M}\sum_{m=1}^{M}\frac{\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})}{\frac{1}{M}\sum_{m=1}^{M}\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})}\Big[\log \Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}}) - \log\Big(\frac{1}{M}\sum_{m=1}^{M}\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})\Big)\Big]$$

$$= \sum_{m=1}^{M}\frac{\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})}{\sum_{m=1}^{M}\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})}\Big[\log \Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}}) - \log\Big(\sum_{m=1}^{M}\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})\Big) - \log\frac{1}{M}\Big]$$

Next the self normalising weights $\widetilde{w}$ are defined:

$$\widetilde{w}^{(m)} = \frac{\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})}{\sum_{m=1}^{M}\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})}$$

$$\log \widetilde{w}^{(m)} = \log \Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}}) - \log \sum_{m=1}^{M}\Gamma^*(\boldsymbol{x_n}, \boldsymbol{z_n^{(m)}}, \boldsymbol{\theta^{(m)}})$$

Combining the expression, we get:

$$\widetilde{E} = \sum_{m=1}^{M}\widetilde{w}^{(m)}\log\widetilde{w}^{(m)} + M\log(M)$$

Since the objective is being minimized the constant can be ignored, giving:

$$\widetilde{E} = \sum_{m=1}^{M}\widetilde{w}^{(m)}\log\widetilde{w}^{(m)}$$

4.4 Algorithm

We denote $\boldsymbol{\psi}$ as the parameters of the global SEP factor (i.e. the moments of the chosen parametric distribution), and $\boldsymbol{\phi}$ the parameters of the recognition model (e.g. the weights of a neural network). α and β are the damping constants for the SEP and recognition model, respectively, and $g(.)$ is a differentiable transform via the reparameterization trick.

1. **Initialization**. Initialize the global SEP factor and the Recognition model parameters:
   $\boldsymbol{\psi}$, $\boldsymbol{\phi} \longleftarrow$ Initialize
2. **Iteration**. Until convergence:
   - **Datapoint**. Choose a datapoint $\boldsymbol{x_n}$.
   - **Sample**. Random samples for the reparameterization of $\boldsymbol{\theta}$ and $\boldsymbol{z_n}$
     $\boldsymbol{\epsilon_\theta} \sim p(\boldsymbol{\epsilon})$
     $\boldsymbol{\epsilon_z} \sim p(\boldsymbol{\epsilon})$
   - **Gradients**. Compute gradients of local objective $\widetilde{E}$
     $\boldsymbol{g} \longleftarrow \nabla_{\boldsymbol{\psi},\boldsymbol{\phi}} \widetilde{E}(\boldsymbol{x_n}, g_\phi(\boldsymbol{x_n}, \boldsymbol{\epsilon_z}), g_\psi(\boldsymbol{x_n}, \boldsymbol{\epsilon_\theta}))$
   - **Optimize**. Optimize parameters using gradient based method to achieve local optimal set of parameters.
     $\boldsymbol{\psi}_n$, $\boldsymbol{\phi}_n$
   - **SEP Update**.Update global SEP factor
     $\widetilde{t}(\boldsymbol{\theta}) = \widetilde{t}(\boldsymbol{\theta})^{1-\alpha} \widetilde{t_n}(\boldsymbol{\theta})^{\alpha}$
   - **Update Recognition Model**. Update recognition model using damping
     $\boldsymbol{\phi} \longleftarrow \boldsymbol{\phi}^{1-\beta} \boldsymbol{\phi}^{\beta}$

5 | Experiments

We use Factor Analysis (FA) as an instance of the generic latent variable model (Figure 1.2) to test AESEP. FA is a linear model where each observation, $\boldsymbol{x_n} \in \mathbb{R}^p$, is generated by some lower dimensional latent variable, $\boldsymbol{z_n} \in \mathbb{R}^q$, through a lineal transformation, $\boldsymbol{W} \in \mathbb{R}^{p \times q}$, with added isotropic noise, σ_x. The transform can for example be: $\boldsymbol{x_n} = \boldsymbol{W z_n} + \boldsymbol{\sigma_x I}$. This has conditional density[1]:

$$p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{W}, \sigma_x^2) \propto exp\{-0.5(\boldsymbol{x_n} - \boldsymbol{W z_n})^T(\sigma_x^2 \boldsymbol{I})^{-1}(\boldsymbol{x_n} - \boldsymbol{W z_n})\}$$

From the likelihood, observe that the posterior is the following:

$$p(\boldsymbol{z_n}|\boldsymbol{x_n}, \boldsymbol{W}, \sigma_x^2) \sim \mathcal{N}([\boldsymbol{I} + \boldsymbol{W^T}(\boldsymbol{\sigma_x^2 I})^{-1}\boldsymbol{W}]^{-1}\boldsymbol{W^T}(\sigma_x^2 \boldsymbol{I})^{-1}\boldsymbol{x_n}, [\boldsymbol{I} + \boldsymbol{W^T}(\boldsymbol{\sigma_x^2 I})^{-1}\boldsymbol{W}]^{-1}) \tag{5.1}$$

Equation 5.1 is compared to the posterior inference approximation produced by the recognition model to verify the approximation. Because SEP's convergence properties are not well understood, Li et al. (2015), evaluation of model learning by SEP is more challenging. This is in contrast to VI with a well-defined lower bound on marginal likelihood. As an alternative, we use the predictive marginal likelihood as our metric. Another option is to plot the fixed points of the EP energy function (See Minka (2001b)). However, this is also questionable as the direct relationship between EP and SEP is an open question, Li et al. (2015). The exact generative model used to generate synthetic data is:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$p(\mathbf{x}|\mathbf{z}, \theta) = \mathcal{N}(\mathbf{Wz}, \sigma_{\mathbf{x}}^{\mathbf{2}}\mathbf{I})$$

$$w_{i,j} \sim \mathcal{N}(0, \sigma_w^2)$$

$$\sigma_{x,w} \sim \mathcal{N}(0, 1)$$

[1] See Fokoué (2009) for a detailed introduction to Factor Analysis.

Implementation details can be found on https://github.com/verajohne/Autoencoding_SEP

5.1 Vanilla EP to Infer Latent Variables

We show that EP performs exact inference in the case of factor analysis. This also serves as a clear demonstration of moment matching. The likelihood terms of FA are Gaussian, therefore when the approximated factors are assumed Gaussian, moments are easily computed through the exact projection:

$$f_n(\boldsymbol{z_n}) = p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{\theta})$$

This derivation is included in appendix C. Figure 5.1 shows that EP successfully performs exact inference over the latent space in FA.

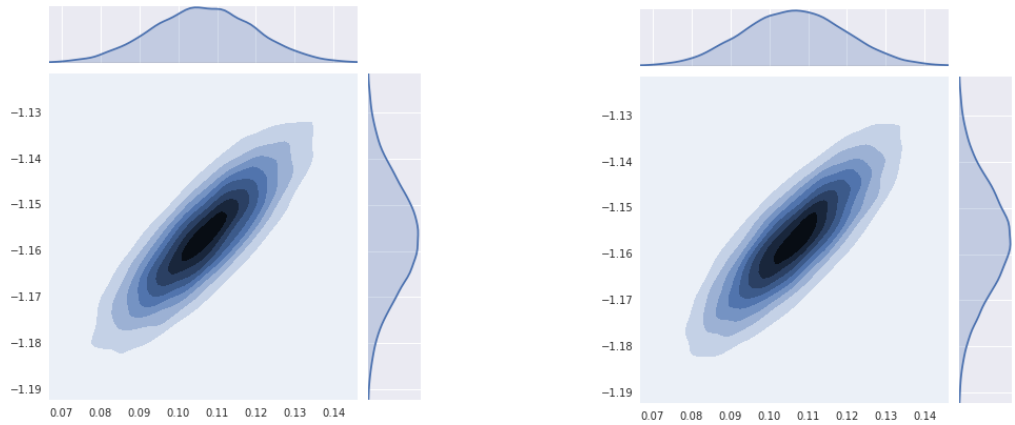


Figure 5.1: Posterior over local latent variable $\boldsymbol{z_n}$ where the latent space is 2 dimensional. samples are generated using a. Equation 5.1 b. Exact EP inference on $\boldsymbol{z_n}$. The small differences in the covariance is due to the stochasticity in the sampler.

5.2 Auto-encoding with SEP

The global SEP factor, $t(\boldsymbol{\theta})$, is assumed Gaussian with mean, $\boldsymbol{u}$, and covariance, $\boldsymbol{V}$, such that $t(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{u}, \boldsymbol{V})$. Since factor analysis is a linear model, the recognition model is utilized to learn the following encoding:

$$f(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{Rx}, \boldsymbol{S})$$

$\boldsymbol{R}$ and $\boldsymbol{S}$ are shared across all N data-points. Thus, the only variables maintained in memory are $\boldsymbol{R}, \boldsymbol{S}, \boldsymbol{u}, \boldsymbol{V}$; independent of dataset size. Indeed, an increase in dimensionality of either latent or observed space will result in larger memory occupancy of each individual variable.

A number of observations are worth noting for future efforts. Initial testing used separate MC samples for estimates of Γ^* and the normalization constant Z. This resulted in numerical issues due to the two sources of variance (see chapter 4.3). As a result, we turned to self normalizing importance sampling. This solved the numerical issues, but noisy gradients were still observed, despite the presence of the reparameterization trick as a control variate. One strategy that reduced observed noise was fixing the random number generator seed in each local minimization. This constrained the variance between each batch of MC samples.

Moreover, since the objective function is a MC estimate, stochastic gradient based optimizers, such as Adam Kingma and Ba (2014), can be leveraged. The implementation of Adam dramatically improved computation time (See Figure 5.2).

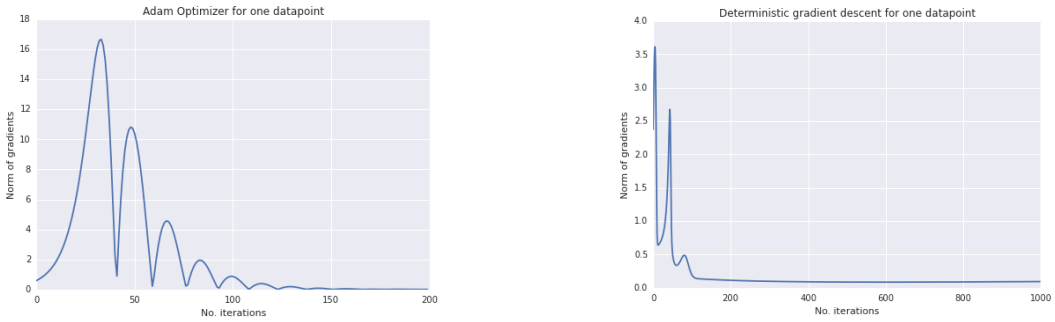


Figure 5.2: The norm of the gradients of $\boldsymbol{u}, \boldsymbol{R}, \boldsymbol{S}$. Taken during the minimization of the first datapoint during the first iteration, using a. Adam b. deterministic gradient descent (GD). Observing the range of the axises shows that (i) Adam stabilized much faster than deterministic GD (x-axis) (ii) Adam adds momentum to the search (y-axis). The parameters of Adam were initialized as advised by the authors

The initial sanity check set $\boldsymbol{R}, \boldsymbol{S}, \boldsymbol{u}, \boldsymbol{V}$ to their optimal values, and it was observed that marginal likelihood stayed maximized, as well as the parameters remaining at their optimal values; indicative of a stable local objective function. Prior to the proposed joint optimization of AESEP, both SEP and the recognition model were tested separately. Figure 5.3 shows the results after fixing the SEP factor to optimal values, and learning the encoding map. Comparing the exact and approximated posterior, we can conclude that the recognition model learns the required encoding.

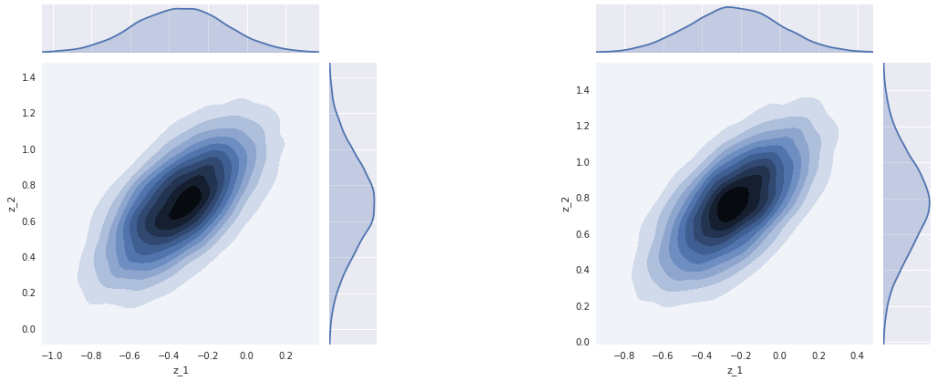


Figure 5.3: Learning $\boldsymbol{R}$, while keeping $\boldsymbol{u}$ and $\boldsymbol{V}$ fixed. The two plots show the posterior over local latent variable $\boldsymbol{z_n}$ where the latent space is 2 dimensional. a. Exact posterior (Equation 5.1) b. Approximated posterior.

For the next experiment the setup was flipped; fixing the recognition model parameters, $\boldsymbol{R}$ and $\boldsymbol{S}$, and training SEP (See Figure 5.4). In theory the predictive marginal likelihood should gradually increase towards the true marginal likelihood, and this is observed when large datasets were used; indicating that SEP requires more observations in order to generalize. When small datasets are used AESEP stabilizes at a local minima. Albeit in a noisy-fashion, predictive marginal likelihood is slowly increasing. However, observe that, unlike in VI, where we expect to see a monotonically increase marginal likelihood, this is not expected in EP-like algorithms.

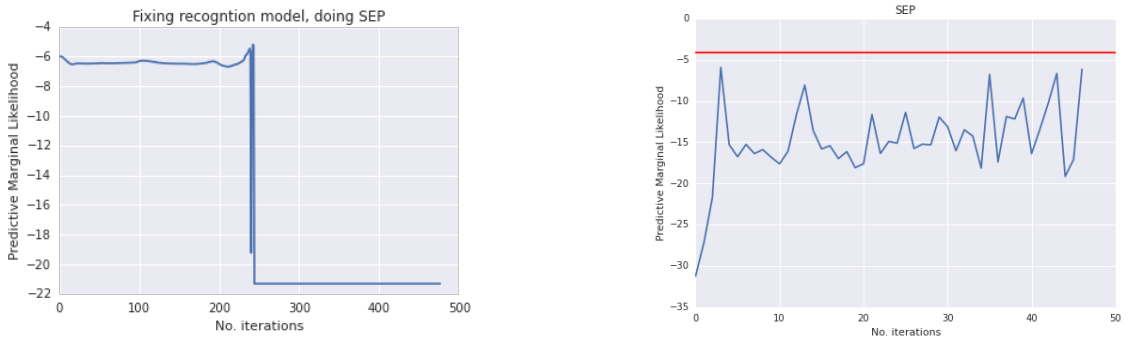


Figure 5.4: Fixing the recognition model, and learning the generative parameters using SEP. a. Small dataset b. Large dataset. The red line indicates true marginal likelihood.

The recognition model successfully trained with SEP fixed, and the previous experiment indicate signs of convergence while fixing the recognition model. However, more experiments are mandated for reliable conclusions. Figure 5.5 shows the predictive marginal likelihood

using full AESEP. Comparing Figure 5.4 and 5.5 similar patterns are observed, except that in the case of full AESEP the stabilization at a local minima happens much sooner when small datasets are used.

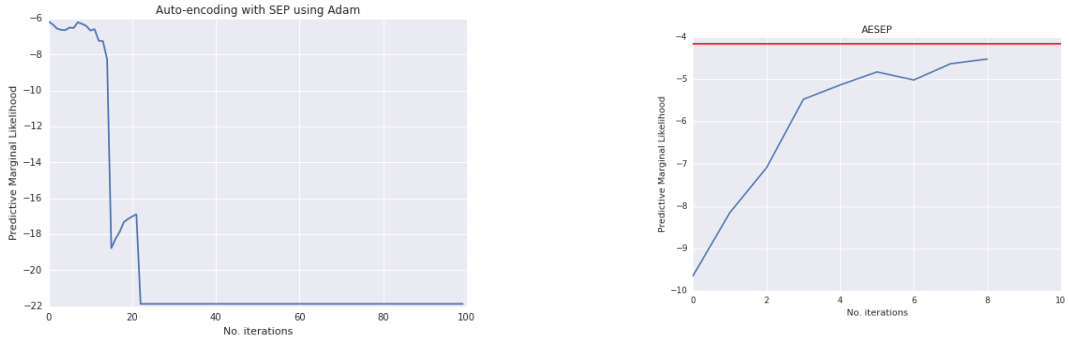


Figure 5.5: Predictive marginal likelihood for full AESEP with random initialization using a. Small Dataset b. Large dataset. The red line indicated true true marginal likelihood.

Experiments revealed AESEP to be difficult to tame; a behavior generally shared across EP-like algorithms. In the light of observed results, in training of AESEP, we propose the following starting points for further investigation. First, it is hypothesized that the two model constraints might be interfering: Does SEP and the recognition model have to learn at a similar rate (damping hyper-parameters)? Second, the theoretical claims of AESEP ought to be verified for larger and established datasets (e.g. MNIST). Third, lack of a global objective is restrictive in sensible visualizations of convergence. Predictive marginal likelihood are the recommended metrics. However, Energy function formulations are expected to ascribe an additional support for the theoretical claims. Fourth, profiling of the code reveals gradient computations as the bottleneck in time complexity, and hence alternative optimization strategies should be explored.

6 | Conclusion

This thesis has introduced AESEP for jointly performing Bayesian learning and inference in large latent variable models. We theoretically proved the existence of a local objective function that is minimized in the EP-fashion. With recent advances in stochastic optimization, we further developed a Monte Carlo approximation of the theoretical objective, and incorporated control variate schemes. Experiments indicate learning and inference on a factor analysis model. Applicability of AESEP to real-life scenarios is an open-ground and is left to future investigations.

References

Christopher M Bishop. Pattern recognition and machine learning, 2006.

Ernest Fokoué. Bayesian computation of the intrinsic structure of factor analytic models. *Copyright© 2002-2009 Journal of Data Science*, 2009. (Page 19)

Andrew Gelman, Aki Vehtari, Pasi Jylänki, Christian Robert, Nicolas Chopin, and John P Cunningham. Expectation propagation as a way of life. *arXiv preprint arXiv:1412.4869*, 2014. (Page 10)

Matthew D Hoffman, David M Blei, Chong Wang, and John William Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013. (Page 2)

Diederik Kingma. Stochastic gradient variational bayes. `http://dpkingma.com/wordpress/wp-content/uploads/2014/05/2014-01_talk_ias.pdf`, 2014. (Page 7)

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. (Page 21)

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. (Pages 2 and 5)

Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner. Stochastic expectation propagation. In *Advances in Neural Information Processing Systems*, pages 2323–2331, 2015. (Pages 2 and 19)

Yingzhen Li, Mark Rowland, Thang Bui, Daniel Hernandez-Lobato, Richard Turner, et al. Black-box alpha divergence minimization. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1511–1520, 2016. (Page 2)

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003. (Page 1)

Thomas P Minka. The ep energy function and minimization schemes. *See www. stat. cmu. edu/˜ minka/papers/learning. html*, 2001a. (Page 11)

Thomas P Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc., 2001b. (Page 19)

Thomas P Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001c. (Page 2)

John Paisley, David Blei, and Michael Jordan. Variational bayesian inference with stochastic search. *arXiv preprint arXiv:1206.6430*, 2012. (Pages 5 and 6)

Omiros Papaspiliopoulos, Gareth O Roberts, and Martin Sköld. A general framework for the parametrization of hierarchical models. *Statistical Science*, pages 59–73, 2007. (Page 5)

Tim Salimans, Diederik P Kingma, Max Welling, et al. Markov chain monte carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*, pages 1218–1226, 2015. (Page 2)

Richard Turner and Maneesh Sahani. Probabilistic amplitude and frequency demodulation. In *Advances in Neural Information Processing Systems*, pages 981–989, 2011. (Page 2)

Richard E Turner and Maneesh Sahani. Two problems with variational expectation maximisation for time-series models. In *Workshop on Inference and Estimation in Probabilistic Time-Series Models*, volume 2, 2008. (Page 2)

A | Derivation of Variational Lowerbound

$$
\begin{aligned}
\log p(\boldsymbol{x}) &= \log \int p_\theta(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} \\
&= \log \int p_\theta(\boldsymbol{x}, \boldsymbol{z}) \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z} \\
&= \log \Big(\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \Big[\frac{p_\theta(\boldsymbol{x}, \boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \Big] \Big) \\
&\geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \Big[\log p_\theta(\boldsymbol{x}, \boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x}) \Big] = \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x})
\end{aligned}
$$

The last step uses Jensen's inequality. The final RHS is the variational lowerbound. VI considers the KL-divergence between the approximate and the exact posterior:

$$
\begin{aligned}
KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x})) &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \log \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x})}{p_\theta(\boldsymbol{z}|\boldsymbol{x})} \\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \Big[-\log p(\boldsymbol{x}, \boldsymbol{z}) + \log p(\boldsymbol{x}) + \log q_\phi(\boldsymbol{z}|\boldsymbol{x}) \Big] \\
&= -\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \Big[\log p_\theta(\boldsymbol{x}, \boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x}) \Big] + \log p(\boldsymbol{x}) \\
&= -\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) + \log p(\boldsymbol{x})
\end{aligned}
$$

B | Moment Matching of EP

A distribution in the exponential family can be expressed in the following form:

$$p(\boldsymbol{z}|\boldsymbol{\eta}) = h(\boldsymbol{z})g(\boldsymbol{\eta})exp\{\ \boldsymbol{\eta}^{\boldsymbol{T}}u(\boldsymbol{z})\}$$

$\boldsymbol{\eta}$ are the natural parameters, $g(\boldsymbol{\eta})$ ensures integration to unity:

$$g(\boldsymbol{\eta})\int h(\boldsymbol{z})exp\{\boldsymbol{\eta}^{\boldsymbol{T}}u(\boldsymbol{z})\}d\boldsymbol{z} = 1$$

Differentiating this with respect to $\boldsymbol{\eta}$ gives:

$$-\triangledown \ln g(\boldsymbol{\eta}) = \mathbb{E}_{q(\boldsymbol{\eta})}[u(\boldsymbol{z})] \tag{B.1}$$

Taking the KL divergence between the exact, p and approximate distribution, q, gives:

$$\begin{aligned}
KL(\boldsymbol{p}||\boldsymbol{q}) &= \int p(\boldsymbol{z})log\frac{p(\boldsymbol{z})}{q(\boldsymbol{z})}d\boldsymbol{\theta} \\
&= \mathbb{E}_{p(\boldsymbol{z})}[\log p(\boldsymbol{z}) - \log q(\boldsymbol{z})] \\
&= \mathbb{E}_{p(\boldsymbol{z})}[\log q(\boldsymbol{z})] + logp(\boldsymbol{z}) \\
&= \mathbb{E}_{p(\boldsymbol{z})}[-\ln\Big(h(\boldsymbol{z})exp\{\boldsymbol{\eta}^{\boldsymbol{T}}u(\boldsymbol{z})\}\Big)] + logp(\boldsymbol{z}) \\
&= \mathbb{E}_{p(\boldsymbol{z})}[-\ln g(\boldsymbol{\eta} - \boldsymbol{\eta}^{\boldsymbol{T}}u(\boldsymbol{z}))] - \ln h(\boldsymbol{z}) + \log p(\boldsymbol{z}) \\
&= -\ln g(\boldsymbol{\eta}) - \boldsymbol{\eta}^{\boldsymbol{T}}\mathbb{E}_{p(\boldsymbol{z})}[u(\boldsymbol{z})] + Constant
\end{aligned}$$

Taking the gradient

$$-\triangledown \ln g(\boldsymbol{\eta}) = \mathbb{E}_{p(\boldsymbol{z})}[u(\boldsymbol{z})] \tag{B.2}$$

Equation B.1 and B.2 imply moment matching:

$$\mathbb{E}_{q(\boldsymbol{z})}[u(\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{z})}[u(\boldsymbol{z})]$$

C | Exact EP on local latent variables of FA

The goal is to infer the latent variables $\boldsymbol{z_n}$ with the following approximate distribution:

$$q^*(\boldsymbol{z_{1:N}}) = \prod_n f(\boldsymbol{z_n})p(\boldsymbol{z_n}) = \prod_n \alpha_n \mathcal{N}(\boldsymbol{z_n}; \boldsymbol{\mu_n}, \boldsymbol{\Sigma})\mathcal{N}(\boldsymbol{z_n}; \boldsymbol{0}, \boldsymbol{I}) = \prod_n q(\boldsymbol{z_n})$$

EP steps

1. **Cavity**:
$$\frac{q^*(\boldsymbol{z_{1:N}})}{f_n(\boldsymbol{z_n})} = q^{\setminus n}(\boldsymbol{z_{1:N}})$$
2. **Tilted**:
$$q^{\setminus n}(\boldsymbol{z_{1:N}})p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{\theta})$$
3. **Moment matching**:
$$\begin{aligned}
&KL\Big(q^{\setminus n}(\boldsymbol{z_{1:N}})p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{\theta})||q^{\setminus n}(\boldsymbol{z_{1:N}})f(\boldsymbol{z_n})\Big) \\
&= \int q^{\setminus n}(\boldsymbol{z_{1:N}})p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{\theta}) log \frac{q^{\setminus n}(\boldsymbol{z_{1:N}})p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{\theta})}{q^{\setminus n}(\boldsymbol{z_{1:N}})f(\boldsymbol{z_n})} \\
&= \int q^{\setminus n}(\boldsymbol{z_{1:N}})p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{\theta}) log \frac{p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{\theta})}{f(\boldsymbol{z_n})} \\
&\implies f_n(\boldsymbol{z_n}) = p(\boldsymbol{x_n}|\boldsymbol{z_n}, \boldsymbol{\theta})
\end{aligned}$$

From the generative model of factor analysis we know the form of the data likelihood, and as a result of exact moment matching we get the following result:

$$\begin{aligned} f_n(\boldsymbol{z_n}) &= \mathcal{N}(\boldsymbol{x_n}; \boldsymbol{W z_n}, \sigma_x^2 \boldsymbol{I}) \\ &= \frac{1}{det(2\pi\sigma_x^2 \boldsymbol{I})} e^{\frac{1}{2\sigma_x^2}(\boldsymbol{x_n} - \boldsymbol{W z_n})^{\boldsymbol{T}}(\boldsymbol{x_n} - \boldsymbol{W z_n})} \\ &= \frac{1}{det(2\pi\sigma_x^2 \boldsymbol{I})} e^{-\frac{1}{2\sigma_x^2}\boldsymbol{x_n^T x_n}} e^{-\frac{1}{2\sigma_x^2}\boldsymbol{z_n^T W^T W z_n} + \frac{1}{\sigma_x^2}\boldsymbol{z_n^T W^T x_n}} \\ &= \alpha_n \mathcal{N}(\boldsymbol{z_n}; \boldsymbol{\mu_n}, \boldsymbol{\Sigma}) \end{aligned}$$

By solving for $\boldsymbol{\mu_n}$, $\boldsymbol{\Sigma}$ and the likelihood scaling factor α_n, a distribution over $\boldsymbol{z_n}$ can be formed. By inspection, the covariance becomes:

$$\boldsymbol{\Sigma} = \Big(\frac{\boldsymbol{W^T W}}{\sigma_x^2}\Big)^{-1}$$

In a similar manner, the mean is calculated:

$$\begin{aligned} \boldsymbol{z_n^T \Sigma^{-1} \mu} &= \frac{1}{\sigma_x^2} \boldsymbol{z_n^T W^T x_n} \\ \frac{\boldsymbol{z_n^T W^T W \mu}}{\sigma_x^2} &= \frac{\boldsymbol{z_n^T W^T x_n}}{\sigma_x^2} \\ \implies \mu &= \boldsymbol{(W^T W)^{-1} W^T x_n} \end{aligned}$$

Lastly, the likelihood scaling factor becomes:

$$f_n(\boldsymbol{z_n}) = \frac{1}{det(2\pi\sigma_x^2\boldsymbol{I})} e^{-\frac{1}{2\sigma_x^2}\boldsymbol{x_n^T x_n}} e^{-\frac{1}{2\sigma_x^2}\boldsymbol{z_n^T W^T W z_n} + \frac{1}{\sigma_x^2}\boldsymbol{z_n^T W^T x_n}}$$

by defining the following variables

$$a = \frac{1}{det(2\pi\sigma_x^2\boldsymbol{I})} e^{-\frac{1}{2\sigma_x^2}\boldsymbol{x_n^T x_n}}$$

$$b = -\frac{1}{2\sigma_x^2}\boldsymbol{z_n^T W^T W z_n} + \frac{1}{\sigma_x^2}\boldsymbol{z_n^T W^T x_n}$$

$$= a \times e^b \times \frac{e^{-\frac{1}{2}\boldsymbol{\mu^T \Sigma^{-1} \mu}}}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} \times \frac{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}}{e^{-\frac{1}{2}\boldsymbol{\mu^T \Sigma^{-1} \mu}}}$$

$$f_n(\boldsymbol{z_n}) = \mathcal{N}(\boldsymbol{z_n}, \mu, \Sigma) \times \frac{a(2\pi)^{\frac{d}{2}}\boldsymbol{|\Sigma|^{\frac{1}{2}}}}{e^{-\frac{1}{2}\boldsymbol{\mu^T \Sigma^{-1} \mu}}}$$

$$\implies \alpha_n = \frac{a(2\pi)^{\frac{d}{2}}\boldsymbol{|\Sigma|^{\frac{1}{2}}}}{e^{-\frac{1}{2}\boldsymbol{\mu^T \Sigma^{-1} \mu}}}$$

Substituting in the calculated mean and covariance and assuming the dimensions of $\boldsymbol{z}$ is 2, the final likelihood factor takes the following tedious form.

$$\alpha_n = \frac{1}{det(2\pi\sigma_x^2\boldsymbol{I})} e^{-\frac{1}{2\sigma_x^2}\boldsymbol{x_n^T x_n}} \times \frac{(2\pi)^{\frac{d}{2}}\boldsymbol{|\Sigma|^{\frac{1}{2}}}}{e^{-\frac{1}{2}\boldsymbol{\mu^T \Sigma^{-1} \mu}}}$$

$$= \frac{(2\pi)^{\frac{d}{2}}\boldsymbol{|\Sigma|^{\frac{1}{2}}}}{det(2\pi\sigma_x^2\boldsymbol{I})} \times e^{-\frac{1}{2\sigma_x^2}\boldsymbol{x_n^T x_n} + \frac{1}{2}\boldsymbol{\mu^T \Sigma^{-1} \mu}}$$

$$\mu^{\mathbf{T}}\boldsymbol{\Sigma^{-1}}\mu = \Big((\boldsymbol{W^T W})^{-1}\boldsymbol{W^T x_n}\Big)^{\boldsymbol{T}} \frac{\boldsymbol{W^T W}}{\boldsymbol{\sigma_x^2}} \Big((\boldsymbol{W^T W})^{-1}\boldsymbol{W^T x_n}\Big)$$

$$= \frac{1}{\sigma_x^2}\Big((\boldsymbol{W^T W})^{-1}\boldsymbol{W^T x_n}\Big)^{\boldsymbol{T}} (\boldsymbol{W^T W})(\boldsymbol{W^T W})^{-1}\boldsymbol{W^T x_n}$$

$$= \frac{1}{\sigma_x^2}\boldsymbol{x_n^T W (W^T W)^{-1} W^T x_n}$$

$$\alpha_n = \frac{(2\pi)^{\frac{d}{2}}\boldsymbol{|\Sigma|^{\frac{1}{2}}}}{|2\pi\sigma_x^2\boldsymbol{I}|} e^{-\frac{1}{2\sigma_x^2}\boldsymbol{x_n^T x_n} + \frac{1}{2\sigma_x^2}\boldsymbol{x_n^T W (W^T W)^{-1} W^T x_n}}$$