Markov Chain Monte Carlo for Deep Latent Gaussian Models

Matthew D. Hoffman
Adobe Research
mathoffm@adobe.com

Abstract

Deep latent Gaussian models (DLGMs) are powerful generative models of high-dimensional data. DLGMs assume a set of Gaussian latent variables z that are intimately involved in the process that generates each observation x. To fit DLGMs using maximum-likelihood estimation, one must somehow compute or approximate the posterior distribution $p(z \mid x)$ of these latent z variables given the data. Typically, this posterior distribution is approximated using variational inference, but it has been clearly established that the standard naive mean-field approximation leads to suboptimal results. There are now many papers exploring the merits of more powerful variational approximations that can be used to fit DLGMs. In this work, we instead explore the use of Markov chain Monte Carlo (MCMC) to approximate $p(z \mid x)$ in DLGMs. While the resulting algorithm loses some of the computational advantages of variational methods, in theory it should also eliminate any bias that variational methods introduce into DLGM parameter estimation. By comparing DLGMs fit using MCMC and variational inference, we can get new qualitative insights into how that bias manifests itself. In particular, we see that DLGMs fit with MCMC use all available latent dimensions, whereas DLGMs fit with variational inference tend to "prune out" less-important latent dimensions.

1 Background

Deep latent Gaussian models (DLGMs; Rezende et al., 2014; Kingma and Welling, 2014) model a set of i.i.d. vectors $x_{1:N}$ as being drawn from the following generative process:

$$z_n \sim \mathcal{N}(0, I); \quad x_n \sim f(g(z_n; \theta)), \tag{1}$$

where f is a distribution with some parameters γ, and those parameters are obtained by passing z_n through a nonlinear function g, which in turn is given by a neural network controlled by some parameters θ. For example, if the observations x are binary, f might be a Bernoulli distribution with mean γ; if the observations are real-valued, f might be a normal distribution whose mean is γ. For simplicity, we will assume that z_n is only fed into the bottom layer of the neural network.

Learning in DLGMs consists of finding a set of parameters $\hat{\theta}$ that maximize the marginal likelihood of the observed data, integrating over all possible z:

$$\hat{\theta} = \arg\max_\theta \tfrac{1}{N} \textstyle\sum_n \log p(x_n) = \arg\max_\theta \tfrac{1}{N} \textstyle\sum_n \log \int_{z_n} p(z_n, x_n) dz_n. \tag{2}$$

The integral over z is not analytically tractable, so it is usually approximated by a lower bound

$$\mathcal{L} \equiv \tfrac{1}{N} \textstyle\sum_n \mathbb{E}_q[\log p(x_n \mid z_n)] + \mathbb{E}_q[\log \tfrac{p(z_n)}{q(z_n)}] \leq \tfrac{1}{N} \textstyle\sum_n \log \int_{z_n} p(z_n, x_n) dz_n. \tag{3}$$

q is typically chosen to have some tractable form, often one parameterized by another neural network. This bound holds for any q, and is tight when $q(z_n) = p(z_n \mid x_n)$ (Jordan et al., 1999), so the

Workshop on Bayesian Deep Learning, NIPS 2016, Barcelona, Spain.

Algorithm 1 MCMC-based Generalized Stochastic EM

Initialize $z_{1:N}$, θ; choose L, ϵ, $\eta(t)$, S.
for t in 1:T **do**
  Sample a minibatch $\mathcal{S}$ of S indices between 1 and N.
  Apply an L-step HMC update with step size ϵ to each $z_s | s \in \mathcal{S}$.
  Update $\theta \leftarrow \theta - \eta(t) \nabla_\theta \frac{1}{S} \sum_{s \in \mathcal{S}} \log p(x_s \mid z_s)$.
end for.

gradient of $\mathcal{L}$ with respect to θ should approximate the gradient of $\log p(x)$ increasingly well as $q(z_n)$ approaches $p(z_n \mid x_n)$.

The quality of the learning signal for θ depends on how well we can approximate the posterior with q. This is a major motivation for developing more powerful q distributions that remain tractable (e.g., Rezende and Mohamed, 2015; Tran et al., 2016; Ranganath et al., 2016b; Burda et al., 2016; Kingma et al., 2016; Ranganath et al., 2016a; Salimans et al., 2015). These approaches make different tradeoffs, but all retain some amount of bias insofar as their q distributions cannot perfectly approximate the posterior.

Below, we will explore a largely overlooked[1] alternative: Markov chain Monte Carlo, and in particular Hamiltonian Monte Carlo (HMC; Neal, 2011).

2 MCMC for DLGMs

Algorithm 1 summarizes our proposed approach. The basic idea is to keep N *persistent* Markov chains running—one for each latent vector z_n. Each iteration, we sample a minibatch of examples and their corresponding latent variables, apply an MCMC update to the latent variables, and then apply a gradient update to the model parameters θ. The bias in this procedure depends on how quickly the Markov chains mix relative to the step size $\eta(t)$—as η approaches zero, so does the bias in the gradient estimates for θ.

We used this procedure to fit a DLGM to a binarized MNIST dataset. The DLGM had 100 latent dimensions, two fully connected hidden layers with 800 units each, and a final fully connected logistic layer giving the probability of each pixel being one or 0. For comparison, we also fit a DLGM using a mean-field inference network with the generative model initialized to the DLGM fit with MCMC—this allows us to examine what kind of biases the variational approximation introduces while hopefully avoiding confusing local optima issues.

To determine the most important directions in the latent space of the z variables, we applied PCA to the following matrix $\bar{J}$:

$$\tilde{z}_{m \in \{1,\ldots,M\}} \sim \mathcal{N}(0, I); \quad J_m \equiv \left.\frac{\partial \mathbb{E}[x|\tilde{z}_m]}{\partial z}\right|_{\tilde{z}_m}; \quad \bar{J} \equiv \frac{1}{M} \sum_m J_m. \tag{4}$$

That is, $\bar{J}$ is an estimate of the average over z of the Jacobian of the expected value of an observation x given z. The largest principal components of $\bar{J}$ correspond to the directions in which, on average, a small perturbation in z will lead to the largest change in x. Conversely, small principal components are directions in z space that can be varied widely without affecting the observation.

Figure 1 shows the singular value spectra of the expected Jacobian matrix $\bar{J}$ for the DLGMs fit with MCMC and mean-field variational inference. The model fit with variational inference has pruned out most latent dimensions, consistent with the observations of Burda et al. (2016). The model fit with MCMC seems to use all available latent dimensions, at least a little.

Figure 2 visualizes what these principal components encode. Each row shows the average reconstruction $\int_{\tilde{z}_{K+1:100}} \mathcal{N}(\tilde{z}_{K+1:100}; 0, I)\mathbb{E}[x \mid z = \tilde{z}]$; that is, we hold the first K most important components of z fixed and average over the remaining components. The first 30 components (those kept by the variational model) seem to encode large-scale structure, while higher-order components encode fine details and noise. This suggests that some of the blurriness sometimes associated with DLGMs (e.g., by Larsen et al., 2015) may be due to variational pruning.

[1]Note that the approach of Salimans et al. (2015) does use HMC as part of a variational approximation. This approach still retains some bias, however, since it does not run a Markov chain to convergence.

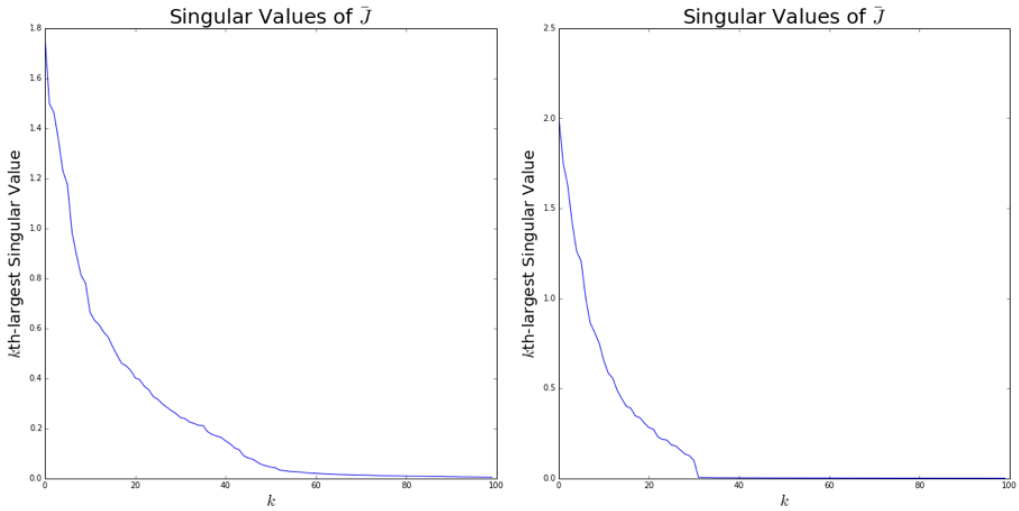

Figure 1: Left: Singular value spectrum of expected Jacobian matrix $\bar{J}$ for the DLGM fit to MNIST using MCMC. Right: Singular value spectrum of expected Jacobian matrix $\bar{J}$ for the DLGM fit to MNIST using mean-field variational inference. The model fit with variational inference has discarded all but 30 latent components.

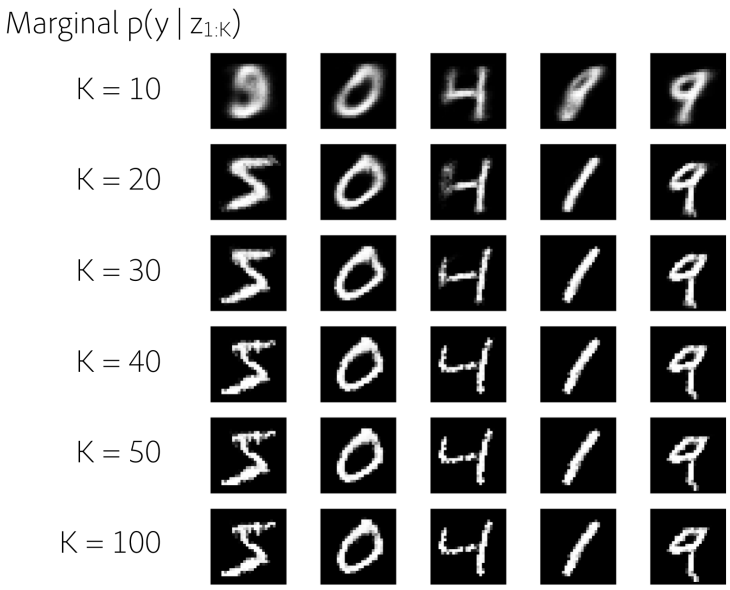

Figure 2: Reconstructions of five MNIST digits, holding fixed a varying number K of principal components. As K increases, the images become clearer.

References

Burda, Y., Grosse, R., and Salakhutdinov, R. (2016). Importance Weighted Autoencoders. In *International Conference on Learning Representations*.

Jordan, M., Ghahramani, Z., Jaakkola, T., and Saul, L. (1999). Introduction to variational methods for graphical models. *Machine Learning*, 37:183–233.

Kingma, D. and Welling, M. (2014). Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*.

Kingma, D. P., Salimans, T., and Welling, M. (2016). Improving Variational Inference with Inverse Autoregressive Flow. In *Neural Information Processing Systems*.

Larsen, A. B. L., Sønderby, S. K., and Winther, O. (2015). Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*.

Neal, R. (2011). *Handbook of Markov Chain Monte Carlo*, chapter 5: MCMC Using Hamiltonian Dynamics. CRC Press.

Ranganath, R., Altosaar, J., Tran, D., and Blei, D. M. (2016a). Operator variational inference. In *Neural Information Processing Systems*.

Ranganath, R., Tran, D., and Blei, D. M. (2016b). Hierarchical variational models. In *International Conference on Machine Learning*.

Rezende, D. J. and Mohamed, S. (2015). Variational Inference with Normalizing Flows. In *International Conference on Machine Learning*.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and variational inference in deep latent Gaussian models. In *International Conference on Machine Learning*.

Salimans, T., Kingma, D. P., and Welling, M. (2015). Markov chain Monte Carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*.

Tran, D., Ranganath, R., and Blei, D. M. (2016). The Variational Gaussian Process. In *International Conference on Learning Representations*.