Gradient-based Variational Inference

STATS 305C: Applied Statistics

Scott Linderman

May 9, 2023

Where are we?

Model	Algorithm	Application
Multivariate Normal Models	Conjugate Inference	Bayesian Linear Regression
Hierarchical Models	MCMC (MH & Gibbs)	Modeling Polling Data
Probabilistic PCA & Factor Analysis	MCMC (HMC)	Images Reconstruction
Mixture Models	EM & Variational Inference	Image Segmentation
Mixed Membership Models	Coordinate Ascent VI	Topic Modeling
Variational Autoencoders	Gradient-based VI	Image Generation
State Space Models	Message Passing	Segmenting Video Data
Bayesian Nonparametrics	Fancy MCMC	Modeling Neural Spike Trains

Taking stock of posterior inference algorithms thus far

We've covered a number of posterior inference algorithms thus far:

- **Exact inference:** for simple models (e.g. conjugate exponential family models) where the posterior is available in closed form.
- **Gibbs sampling:** an MCMC algorithm that iteratively samples conditional distributions for one variable at a time. This works well for conditionally conjugate models with weak correlations.
- **Metropolis-Hastings:** a very general MCMC algorithm to sample the posterior, and the building block for many other MCMC techniques.
- **Hamiltonian Monte Carlo:** an MCMC algorithm to draw samples from the posterior by leveraging gradients of the log joint probability. This works well for more general posteriors over continuous variables.
- **Coordinate Ascent Variational Inference:** Minimizes the KL divergence between an approximate distribution (e.g., the a mean-field distribution) and the true posterior. Yields biased estimates, but potentially lower variance for fixed computational budget.

Coordinate Ascent Variational Inference (CAVI)

- *What parametric family should we use?*
  - The **mean-field family**.
- *How should we measure closeness?*
  - The **Kullback-Leibler (KL)** divergence.
- *How do we find the closest distribution in that family?*
  - **Coordinate ascent**, assuming we have a conditionally conjugate model.

Gradient-based Variational Inference

- *What parametric family should we use?*
  - Pretty much any q, as long as we can sample from it and evaluate the log density.
- *How should we measure closeness?*
  - The **Kullback-Leibler (KL)** divergence.
- *How do we find the closest distribution in that family?*
  - **Stochastic gradient ascent** using Monte Carlo estimates of the ELBO and its gradient.

Gradient-based VI methods go under a few different names: black-box VI (BBVI), automatic differentiation VI (ADVI), fixed-form VI...

Setup

Let $\boldsymbol{\theta}$ denote parameters to be inferred and $p(\boldsymbol{\theta} \mid \boldsymbol{X})$ the true posterior to be approximated.

Let $\mathcal{Q} = \{q(\boldsymbol{\theta}; \boldsymbol{\lambda}) : \boldsymbol{\lambda} \in \boldsymbol{\Lambda}\}$ denote the variational family – a parametric family of distributions indexed by $\boldsymbol{\lambda}$ that we will optimize over in variational inference.

The optimal approximation is,

$$q^\star(\boldsymbol{\theta}; \boldsymbol{\lambda}) = \arg\min_{q \in \mathcal{Q}} D_{\mathrm{KL}}\left(q(\boldsymbol{\theta}; \boldsymbol{\lambda}) \parallel p(\boldsymbol{\theta} \mid \boldsymbol{X})\right) \tag{1}$$

or equivalently

$$\boldsymbol{\lambda}^\star = \arg\min_{\boldsymbol{\lambda} \in \boldsymbol{\Lambda}} D_{\mathrm{KL}}\left(q(\boldsymbol{\theta}; \boldsymbol{\lambda}) \parallel p(\boldsymbol{\theta} \mid \boldsymbol{X})\right) \tag{2}$$

$$= \arg\max_{\boldsymbol{\lambda} \in \boldsymbol{\Lambda}} \mathscr{L}(\boldsymbol{\lambda}) \tag{3}$$

where $\mathscr{L}(\boldsymbol{\lambda})$ denotes the **evidence lower bound (ELBO)**,

$$\mathscr{L}(\boldsymbol{\lambda}) = \mathbb{E}_{q(\boldsymbol{\theta}; \boldsymbol{\lambda})}\left[\log p(\boldsymbol{X}, \boldsymbol{\theta}) - \log q(\boldsymbol{\theta}; \boldsymbol{\lambda})\right] \tag{4}$$

Stochastic gradient ascent on the local ELBO

- **Idea:** Assume the variational parameters $\boldsymbol{\Lambda}$ are unconstrained (i.e., $\boldsymbol{\Lambda} = \mathbb{R}^D$), then perform (stochastic) gradient ascent.
- If the parameters are unconstrained and the ELBO is differentiable, we can use **gradient ascent**. Repeat:

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \alpha \nabla_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda}) \tag{5}$$

  with **step size** α. Typically, you decrease the step size over iterations so that $\alpha_1 \geq \alpha_2 \geq \ldots$
- More generally, we can use ***stochastic*** **gradient ascent** with an estimate of the gradient, $\widehat{\nabla}_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda})$, as long as it is unbiased,

$$\mathbb{E}[\widehat{\nabla}_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda})] = \nabla_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda}). \tag{6}$$

Stochastic gradient ascent on the local ELBO

- **Idea:** Assume the variational parameters $\boldsymbol{\Lambda}$ are unconstrained (i.e., $\boldsymbol{\Lambda} = \mathbb{R}^D$), then perform (stochastic) gradient ascent.
- If the parameters are unconstrained and the ELBO is differentiable, we can use **gradient ascent**. Repeat:

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \alpha \nabla_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda}) \tag{5}$$

  with **step size** α. Typically, you decrease the step size over iterations so that $\alpha_1 \geq \alpha_2 \geq \ldots$
- More generally, we can use ***stochastic* gradient ascent** with an estimate of the gradient, $\widehat{\nabla}_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda})$, as long as it is unbiased,

$$\mathbb{E}[\widehat{\nabla}_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda})] = \nabla_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda}). \tag{6}$$

Stochastic gradient ascent on the local ELBO

- **Idea:** Assume the variational parameters $\boldsymbol{\Lambda}$ are unconstrained (i.e., $\boldsymbol{\Lambda} = \mathbb{R}^D$), then perform (stochastic) gradient ascent.
- If the parameters are unconstrained and the ELBO is differentiable, we can use **gradient ascent**. Repeat:

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \alpha \nabla_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda}) \tag{5}$$

  with **step size** α. Typically, you decrease the step size over iterations so that $\alpha_1 \geq \alpha_2 \geq \ldots$
- More generally, we can use ***stochastic*** **gradient ascent** with an estimate of the gradient, $\widehat{\nabla}_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda})$, as long as it is unbiased,

$$\mathbb{E}[\widehat{\nabla}_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda})] = \nabla_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda}). \tag{6}$$

Monte Carlo gradient estimation

No problem! We'll just use ordinary Monte Carlo to estimate the gradient. But we run into a problem...

$$\nabla_{\boldsymbol{\lambda}} \mathscr{L}(\boldsymbol{\lambda}) = \nabla_{\boldsymbol{\lambda}} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})} \left[\log p(\boldsymbol{x}, \boldsymbol{\theta}) - \log q(\boldsymbol{\theta}; \boldsymbol{\lambda}) \right] \tag{7}$$

$$\neq \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})} \left[\nabla_{\boldsymbol{\lambda}} \left(\log p(\boldsymbol{x}, \boldsymbol{\theta}) - \log q(\boldsymbol{\theta}; \boldsymbol{\lambda}) \right) \right]. \tag{8}$$

Problem: Why can't we simply bring the gradient inside the expectation?

The “score function” gradient estimator I

The basic problem is that the variational parameters $\boldsymbol{\lambda}$ determine the distribution we are taking an expectation under. However, there are a few ways to obtain unbiased estimates of the gradient.

One approach is called the **score function gradient estimator** or the **REINFORCE estimator** [Williams, 1992]. It is based on the following identity,

$$\nabla_{\boldsymbol{\lambda}} \log q(\boldsymbol{\theta}; \boldsymbol{\lambda}) = \frac{\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\theta}; \boldsymbol{\lambda})}{q(\boldsymbol{\theta}; \boldsymbol{\lambda})} \tag{9}$$

where the l.h.s. is called the score function of distribution q.

The "score function" gradient estimator II

We can use this identity to obtain an unbiased estimate of the gradient of an expectation,

$$\nabla_{\boldsymbol{\lambda}} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta})\right] = \nabla_{\boldsymbol{\lambda}} \int q(\boldsymbol{\theta};\boldsymbol{\lambda})\, h(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta} \tag{10}$$

$$= \int \left(\nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\theta};\boldsymbol{\lambda})\right) h(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta} \tag{11}$$

$$= \int \left(q(\boldsymbol{\theta};\boldsymbol{\lambda}) \nabla_{\boldsymbol{\lambda}} \log q(\boldsymbol{\theta};\boldsymbol{\lambda})\right) h(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta} \tag{12}$$

$$= \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[\left(\nabla_{\boldsymbol{\lambda}} \log q(\boldsymbol{\theta};\boldsymbol{\lambda})\right) h(\boldsymbol{\theta})\right] \tag{13}$$

From this identity, we can obtain an unbiased Monte Carlo estimate,

$$\widehat{\nabla}_{\boldsymbol{\lambda}} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}[h(\boldsymbol{\theta})] = \frac{1}{M} \sum_{m=1}^{M} \left[\nabla_{\boldsymbol{\lambda}} \log q(\boldsymbol{\theta}^{(m)};\boldsymbol{\lambda})\, h(\boldsymbol{\theta}^{(m)})\right]; \qquad \boldsymbol{\theta}^{(m)} \overset{\text{iid}}{\sim} q(\boldsymbol{\theta};\boldsymbol{\lambda}) \tag{14}$$

The “score function” gradient estimator III

Notes:

1. The exchange of the gradient and the integral is allowed as long as the dominated convergence theorem holds, and it usually does for ML applications.
2. The score function gradient estimator is broadly applicable; e.g. it works for discrete and continuous latent variables $\boldsymbol{\theta}$. We just need the log density to be continuously differentiable wrt $\boldsymbol{\lambda}$ and to be able to sample from q.
3. If h is a function of both $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$, you need to apply the product rule. This gives another term,

$$\nabla_{\boldsymbol{\lambda}} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta},\boldsymbol{\lambda})\right] = \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[\left(\nabla_{\boldsymbol{\lambda}} \log q(\boldsymbol{\theta};\boldsymbol{\lambda})\right) h(\boldsymbol{\theta},\boldsymbol{\lambda})\right] + \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[\nabla_{\boldsymbol{\lambda}} h(\boldsymbol{\theta},\boldsymbol{\lambda})\right] \tag{15}$$

Control variates

Though broadly applicable, the score function estimator is often too high variance to be useful. This problem can often be mitigated with **control variates**.

Recall that the expectation of the score is zero,

$$\mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[\nabla_{\boldsymbol{\lambda}} \log q(\boldsymbol{\theta};\boldsymbol{\lambda})\right] = \int q(\boldsymbol{\theta};\boldsymbol{\lambda}) \nabla_{\boldsymbol{\lambda}} \log q(\boldsymbol{\theta};\boldsymbol{\lambda}) \,\mathrm{d}\boldsymbol{\theta} \tag{16}$$

$$= \int \nabla_{\boldsymbol{\lambda}} q(\boldsymbol{\theta};\boldsymbol{\lambda}) \,\mathrm{d}\boldsymbol{\theta} \tag{17}$$

$$= \nabla_{\boldsymbol{\lambda}} \int q(\boldsymbol{\theta};\boldsymbol{\lambda}) \,\mathrm{d}\boldsymbol{\theta} \tag{18}$$

$$= \nabla_{\boldsymbol{\lambda}} 1 = 0. \tag{19}$$

Thus, we can subtract off any **baseline** from the function of interest without changing the expectation, but potentially reducing variance substantially,

$$\mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta}) \nabla_{\boldsymbol{\lambda}} \log q(\boldsymbol{\theta};\boldsymbol{\lambda})\right] = \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[(h(\boldsymbol{\theta}) - b) \nabla_{\boldsymbol{\lambda}} \log q(\boldsymbol{\theta};\boldsymbol{\lambda})\right]. \tag{20}$$

The pathwise gradient estimator

- For example, suppose $q(\boldsymbol{\theta}; \boldsymbol{\lambda}) = \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$, where $\boldsymbol{\lambda} = (\boldsymbol{\mu}, \log \boldsymbol{\sigma}^2)$ are the (unconstrained) variational parameters. Then,

$$\boldsymbol{\theta} \sim q(\boldsymbol{\theta}; \boldsymbol{\lambda}) \quad \Longleftrightarrow \quad \boldsymbol{\theta} = r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}), \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{21}$$

  where $r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}) = \boldsymbol{\mu} + \boldsymbol{\sigma}\boldsymbol{\epsilon}$ is a **reparameterization** of $\boldsymbol{\theta}$ in terms of parameters $\boldsymbol{\lambda}$ and "noise" $\boldsymbol{\epsilon}$.

- We can use the **law of the unconscious statistician** to rewrite the expectations as,

$$\mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta}, \boldsymbol{\lambda})\right] = \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})}\left[h(r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}), \boldsymbol{\lambda})\right] \tag{22}$$

  The distribution that the expectation is taken under no longer depends on the parameters $\boldsymbol{\lambda}$, so we can simply take the gradient inside the expectation,

$$\nabla_{\boldsymbol{\lambda}} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta}, \boldsymbol{\lambda})\right] = \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})}\left[\nabla_{\boldsymbol{\lambda}} h(r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}), \boldsymbol{\lambda})\right] \tag{23}$$

- Now we can **use Monte Carlo to obtain an unbiased estimate of the final expectation.**

$$\widehat{\nabla}_{\boldsymbol{\lambda}} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta}, \boldsymbol{\lambda})\right] = \frac{1}{M} \sum_{m=1}^{M} \nabla_{\boldsymbol{\lambda}} h(r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}_m), \boldsymbol{\lambda}); \qquad \boldsymbol{\epsilon}_m \overset{\text{iid}}{\sim} \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{24}$$

The pathwise gradient estimator

- For example, suppose $q(\boldsymbol{\theta};\boldsymbol{\lambda}) = \mathcal{N}(\boldsymbol{\theta};\boldsymbol{\mu},\mathrm{diag}(\boldsymbol{\sigma}^2))$, where $\boldsymbol{\lambda} = (\boldsymbol{\mu}, \log\boldsymbol{\sigma}^2)$ are the (unconstrained) variational parameters. Then,

$$\boldsymbol{\theta} \sim q(\boldsymbol{\theta};\boldsymbol{\lambda}) \quad \Longleftrightarrow \quad \boldsymbol{\theta} = r(\boldsymbol{\lambda},\boldsymbol{\epsilon}), \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\boldsymbol{I}) \tag{21}$$

  where $r(\boldsymbol{\lambda},\boldsymbol{\epsilon}) = \boldsymbol{\mu} + \boldsymbol{\sigma}\boldsymbol{\epsilon}$ is a **reparameterization** of $\boldsymbol{\theta}$ in terms of parameters $\boldsymbol{\lambda}$ and "noise" $\boldsymbol{\epsilon}$.

- We can use the **law of the unconscious statistician** to rewrite the expectations as,

$$\mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta},\boldsymbol{\lambda})\right] = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[h(r(\boldsymbol{\lambda},\boldsymbol{\epsilon}),\boldsymbol{\lambda})\right] \tag{22}$$

  The distribution that the expectation is taken under no longer depends on the parameters $\boldsymbol{\lambda}$, so we can simply take the gradient inside the expectation,

$$\nabla_{\boldsymbol{\lambda}}\mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta},\boldsymbol{\lambda})\right] = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\nabla_{\boldsymbol{\lambda}}h(r(\boldsymbol{\lambda},\boldsymbol{\epsilon}),\boldsymbol{\lambda})\right] \tag{23}$$

- Now we can **use Monte Carlo to obtain an unbiased estimate of the final expectation.**

$$\widehat{\nabla}_{\boldsymbol{\lambda}}\mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta},\boldsymbol{\lambda})\right] = \frac{1}{M}\sum_{m=1}^{M}\nabla_{\boldsymbol{\lambda}}h(r(\boldsymbol{\lambda},\boldsymbol{\epsilon}_m),\boldsymbol{\lambda}); \qquad \boldsymbol{\epsilon}_m \overset{\text{iid}}{\sim} \mathcal{N}(\mathbf{0},\boldsymbol{I}) \tag{24}$$

The pathwise gradient estimator

- For example, suppose $q(\boldsymbol{\theta}; \boldsymbol{\lambda}) = \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$, where $\boldsymbol{\lambda} = (\boldsymbol{\mu}, \log \boldsymbol{\sigma}^2)$ are the (unconstrained) variational parameters. Then,

$$\boldsymbol{\theta} \sim q(\boldsymbol{\theta}; \boldsymbol{\lambda}) \quad \iff \quad \boldsymbol{\theta} = r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}), \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{21}$$

  where $r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}) = \boldsymbol{\mu} + \boldsymbol{\sigma}\boldsymbol{\epsilon}$ is a **reparameterization** of $\boldsymbol{\theta}$ in terms of parameters $\boldsymbol{\lambda}$ and "noise" $\boldsymbol{\epsilon}$.

- We can use the **law of the unconscious statistician** to rewrite the expectations as,

$$\mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta}, \boldsymbol{\lambda})\right] = \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})}\left[h(r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}), \boldsymbol{\lambda})\right] \tag{22}$$

  The distribution that the expectation is taken under no longer depends on the parameters $\boldsymbol{\lambda}$, so we can simply take the gradient inside the expectation,

$$\nabla_{\boldsymbol{\lambda}} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta}, \boldsymbol{\lambda})\right] = \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})}\left[\nabla_{\boldsymbol{\lambda}} h(r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}), \boldsymbol{\lambda})\right] \tag{23}$$

- Now we can **use Monte Carlo to obtain an unbiased estimate of the final expectation.**

$$\widehat{\nabla}_{\boldsymbol{\lambda}} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[h(\boldsymbol{\theta}, \boldsymbol{\lambda})\right] = \frac{1}{M} \sum_{m=1}^{M} \nabla_{\boldsymbol{\lambda}} h(r(\boldsymbol{\lambda}, \boldsymbol{\epsilon}_m), \boldsymbol{\lambda}); \qquad \epsilon_m \overset{\text{iid}}{\sim} \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{24}$$

Exercises

Exercise: Come up with a reparameterization of an exponential distribution, $q(\theta; \lambda) = \text{Exp}(\theta; \lambda)$

Question: Can you use the pathwise gradient estimator for a Bernoulli posterior, $q(\theta; \lambda) = \text{Bern}(\theta; \lambda)$?

Empirically comparing estimator variances

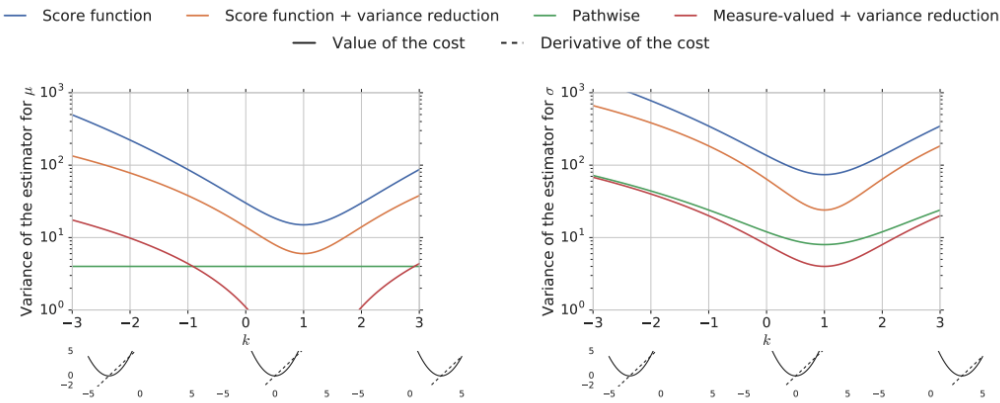

Figure 2: Variance of the stochastic estimates of $\nabla_\theta \mathbb{E}_{\mathcal{N}(x|\mu,\sigma^2)}\left[(x-k)^2\right]$ for $\mu = \sigma = 1$ as a function of k for three different classes of gradient estimators. Left: $\theta = \mu$; right: $\theta = \sigma$. The graphs in the bottom row show the function (solid) and its gradient (dashed) for $k \in \{-3, 0, 3\}$.

Empirical comparisons from Mohamed et al. [2020].

Working with mini-batches of data

Often, the ELBO involves a sum over data points,

$$\mathscr{L}(\boldsymbol{\lambda}) = \mathbb{E}_q[\log p(\boldsymbol{X}, \boldsymbol{\theta}) - \log q(\boldsymbol{\theta}; \boldsymbol{\lambda}] \tag{25}$$

$$= \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}\left[\sum_{n=1}^{N} \log p(\boldsymbol{x}_n \mid \boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) - \log q(\boldsymbol{\theta}; \boldsymbol{\lambda})\right] \tag{26}$$

$$= \sum_{n=1}^{N} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}[\log p(\boldsymbol{x}_n \mid \boldsymbol{\theta})] - D_{\mathrm{KL}}\left(q(\boldsymbol{\theta}; \boldsymbol{\lambda}) \,\|\, p(\boldsymbol{\theta})\right) \tag{27}$$

We can view the sum as an "expectation" over data indices,

$$\sum_{n=1}^{N} \mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}[\log p(\boldsymbol{x}_n \mid \boldsymbol{\theta})] = N\,\mathbb{E}_{n\sim\mathrm{Unif}(1,N)}[\mathbb{E}_{q(\boldsymbol{\theta};\boldsymbol{\lambda})}[\log p(\boldsymbol{x}_n \mid \boldsymbol{\theta})]], \tag{28}$$

and we can use Monte Carlo to approximate both expectations! (The same is true for Monte Carlo estimators of the gradient of the ELBO.)

SGD convergence and extensions

When does SGD work? This is a well studied problem in stochastic optimization [Bottou et al., 1998, Robbins and Siegmund, 1971].

Under relatively mild conditions, SGD converges to a **local minimum** if the step sizes obey the **Robbins-Monro condtions**,

$$\sum_{i=0}^{\infty} \alpha_i = \infty \qquad \text{and} \qquad \sum_{i=0}^{\infty} \alpha_i^2 < \infty \tag{29}$$

There have been dozens of extensions to basic SGD including,

- SGD with momentum
- AdaGrad [Duchi et al., 2011]
- RMSProp
- Adam [Kingma and Ba, 2014]

References I

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte Carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020.

Léon Bottou et al. Online learning and stochastic approximations. *On-line learning in neural networks*, 17(9):142, 1998.

Herbert Robbins and David Siegmund. A convergence theorem for non negative almost supermartingales and some applications. In *Optimizing methods in statistics*, pages 233–257. Elsevier, 1971.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

References II

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.