**Title**

Methods and applications in large-scale variational inference



**Author**

Liu, Runjing

**Publication Date**

2021

Peer reviewed|Thesis/dissertation

# Methods and applications in large-scale variational inference

by

Runjing Liu

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Statistics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Jon D. McAuliffe, Chair
Professor Michael I. Jordan
Professor Martin J. Wainwright

Summer 2021

Methods and applications in large-scale variational inference

Abstract

Methods and applications in large-scale variational inference

by

Runjing Liu

Doctor of Philosophy in Statistics

University of California, Berkeley

Professor Jon D. McAuliffe, Chair

This dissertation can be viewed as a collection of case studies in applying variational inference to analyze large data problems. The largest, most computationally difficult data problem we tackle is the task of cataloging light sources imaged by astronomical surveys (Chapter 2). For some surveys, the size of the raw data is on the scale of petabytes. To do inference, we employ two recent advances in variational inference: amortization and the wake-sleep algorithm.

The data analysis problems in Chapters 3, 4, and 5 are more tractable. The analyses in these chapters are exploratory in nature. However, such exploratory data analyses often require fitting either several related models or fitting the same model on subsamples of the data. Repeatedly solving for variational optima after each model or data perturbation may be unnecessarily expensive, particularly for exploratory settings where approximate optima might suffice. In these chapters, we present the notion of *local sensitivity* which we use to quickly extrapolate, from an initial variational optimum, the posterior quantities that would be obtained after a model or data perturbation. Our approach is particularly apt for sensitivity analysis, where the goal is to understand how conclusions might change should a different model be specified or should different data be observed.

Finally, Chapter 6 considers probabilistic machine learning problems where the training objective is an expectation over a discrete latent variable. The standard reparameterization and backpropagation method for computing stochastic gradients do not apply in this setting. Many alternative stochastic gradient estimators have been proposed specifically for this problem. Chapter 6 outlines a technique to lower the variance of any gradient estimator by employing a general statistical method called Rao-Blackwellization.

To Chuangmin, Haiying, Lindsay, and Helen

# Contents

# List of Figures

# List of Tables

# Acknowledgments

I am extraordinarily grateful to all of my committee members, Jon McAuliffe, Michael Jordan, and Martin Wainwright, for their guidance during my time at UC Berkeley.

Jon, as my PhD advisor, was the primary driver of my growth over the past five years. I had the opportunity to work with Jon on a diverse array of research projects. In each case, Jon was never short of helpful insights when I needed advice. I gained much technical knowledge in working with Jon, and I hope that in some small way, I am able to emulate his thoughtfulness in approaching statistical problems. Our work together appears in Chapters 2 and 6

Mike to me is the embodiment of academic scholarship. He inspires me to embrace all that academic research has to offer by reading literature far and wide while boldly seeking new ideas. Mike oversaw the work that appears in Chapters 3 through 6.

The inception of this dissertation is due to Ryan Giordano, who introduced me to variational inference. I could not have asked for a more patient mentor—Ryan took me under his wing as a first year student who barely knew how to code, and it was through his repeated code-reviews that I became passably competent in the end. Ryan always pushed me to set the highest standards for myself and my work; I would not be the researcher I am today without him. I would also like to mention the generosity of Tamara Broderick, who despite not being my advisor (and has only ever met me virtually!), has always found time to give feedback on my work. She patiently coached me through my first NeurIPS presentation, and those sessions alone raised my bar for future talks. Ryan and Tamara are co-authors on work that appears in Chapters 3, 4, and 5.

I am also indebted to Jeffrey Regier, who introduced me to astronomy applications of variational inference. He took a chance on me after I emailed him out of the blue, inquiring about his research interests. Since then, Jeff has been like another PhD advisor to me, and he is one of the first people I turn to for advice. The fruits of our collaboration can be found in Chapters 2 and 6.

There have been countless other individuals who have supported me over the years. I would not have survived the first-year curriculum at UC Berkeley without the camaraderie of Eli Ben-Michael and Jake Soloff. Late nights in Evans Hall with take-out from La Burrita were a staple of those early PhD years.

This thesis is dedicated to my parents who are my role models in their courage, tenacity, and work ethic. And to my sister, whom I love for her youthful spirit and moxie.

Finally, a dedication to my partner Helen Liu, who has made my life at UC Berkeley wonderful in more ways than I could have imagined. I came to the Bay Area as a lonely student, and will leave with a best friend. The warmth of her companionship is enough for a lifetime, almost surely.

# Chapter 1

# Introduction

Graphical models provide a flexible framework for reasoning about complex systems. Applications range from processing images collected by astronomical surveys, to inferring taxonomy using genomic data, to forecasting the state of financial markets. Graphical models represent relationships between variables as either joint or conditional distributions, and they form the starting point for many statistical models. Some variables in the graph are observed as data, while other variables are latent, or unobserved. Let $\mathbf{x} = (x_1, ..., x_n)$ be observed variables and $\mathbf{z} = (z_1, ..., z_m)$ be latent variables. In Bayesian statistics, the central quantity of interest is the posterior distribution $p(\mathbf{z}|\mathbf{x})$, the distribution of latent variables conditional on observed data.

Algorithms for computing the exact posterior distribution, or posterior summaries such as means and modes, are available for only a small subset of graphical models. Examples include the Kalman filter for linear state space models (Kalman, 1960; Shumway and Stoffer, 2017) and the Viterbi algorithm for discrete-state hidden Markov models (Forney, 1973). The Viterbi algorithm can be generalized to the *max-product* algorithm for inference on tree-structured graphs (of which hidden Markov models are a special case).

For general graphical models, approximate methods are required. Markov Chain Monte Carlo (MCMC) methods construct a sequence of random variables that converge in distribution to the exact posterior. The sequence of random variables is defined by a transition kernel specially chosen such that the stationary distribution of the chain is the exact posterior. Many recent advances in MCMC involve developing "black-box" methods which are model agnostic and hence can be applied to generic graphical models with little additional modification. Hamiltonian Monte Carlo (HMC) and the related no-U-turn sampler (NUTS), both of which apply to any model with a differentiable joint probability density, are two such examples (Neal, 2012; Betancourt and Girolami, 2013; Hoffman and Gelman, 2014).

However, sampling from the exact posterior requires a theoretically infinite number of steps. In practice, a finite number of steps are run. Various diagnostics have been proposed to evaluate convergence (Gelman and Rubin, 1992; Gelman et al., 2013), some of which require several runs of MCMC. In many applications, the chain may be slow to mix, and the runtime is prohibitive for large datasets of interest (Chapter 2).

Alternatively, variational methods use optimization for inference, rather than sampling. *Variational inference* defines a family of of approximating distributions $\mathcal{Q}$ and seeks the distribution $q^* \in \mathcal{Q}$ that is as close as possible to the exact posterior in KL divergence (Jordan et al., 1999; Wainwright and Jordan, 2008; Blei et al., 2017). When the distributions in $\mathcal{Q}$ are parameterized by a real-valued vector $\eta \in \mathbb{R}^d$, posterior inference amounts to solving the numerical optimization problem,

$$\eta^* = \underset{\eta \in \mathbb{R}^d}{\operatorname{argmin}} \Big\{ \mathrm{KL}\big[\, q_\eta(\mathbf{z}) \,\|\, p(\mathbf{z}|\mathbf{x}) \,\big] \Big\}. \tag{1.1}$$

Minimizing the KL divergence is equivalent to maximizing the evidence lower bound, or ELBO (Blei et al., 2017):

$$\mathcal{L}(\eta; \mathbf{x}) := \mathbb{E}_{q_\eta(\mathbf{z})} \Big[ \log p(\mathbf{x}, \mathbf{z}) - \log q_\eta(\mathbf{z}) \Big]. \tag{1.2}$$

Notice that the ELBO does not depend on the marginal distribution $p(\mathbf{x})$ or the posterior distribution $p(\mathbf{z}|\mathbf{x})$, both of which are intractable.

The class of distributions $\mathcal{Q}$ is chosen such the KL minimization problem is straightforward to solve using standard numerical optimization techniques. Typically, the family of distributions $\mathcal{Q}$ is constructed by breaking statistical dependencies. In the extreme case, *mean-field variational inference*, the posterior distribution is approximated by a distribution that fully factorizes over the latent variables,

$$q_\eta(\mathbf{z}) = \prod_{i=1}^{m} q_\eta(z_i).$$

By casting the posterior inference problem as a numerical optimization problem, the variational approach can leverage the vast literature on large-scale numerical optimization (Nocedal and Wright, 2006). On large data sets, solving the optimization problem (Equation 1.1) can be orders of magnitude faster than running MCMC.

The approaches we develop in this dissertation all have a common goal of reducing the runtime of variational methods in order to make inference feasible for their respective data applications. However, with the exception of Chapter 6, our contributions lie not in developing novel optimization techniques or improving existing optimization routines *per se*. In fact, for all the data problems encountered in this dissertation, the optimization problem (Equation 1.1) was solved with well-studied algorithms such as stochastic gradient descent or quasi-second order methods such as limited-memory BFGS.

Rather, Chapters 2 through 5 of this dissertation consider situations where optimization must be re-run either when new data is observed, the model is perturbed, or the existing data is subsampled. Chapter 2 employs variational inference to catalog astronomical surveys. These surveys scan the sky over many nights. With the arrival of each new data point, a traditional variational approach requires a new solution to Equation 1.1. To avoid repeated, expensive optimization, we instead use an *amortized* approach. Here, a neural network maps

data points to variational distributions. After the initial cost of fitting the neural network, producing an approximate posterior on new data requires only a forward pass through the network. The amortization enables our method, called *StarNet*, to scale variational inference to the petabytes of data collected by some modern surveys.

In analyzing a data set, it is generally good statistical practice to understand how conclusions drawn from the data might be affected by potentially subjective modeling choices. In Bayesian statistics, this includes determining the sensitivity of resulting inferences to possible prior choices. A conceptually straightforward way to evaluate prior sensitivity would be to refit the variational posterior for each plausible choice of prior. However, each refit requires a new solution to Equation 1.1. To avoid expensive refitting, we employ *local sensitivity* (Gustafson, 1996) to linearly approximate the refitting process. Local prior sensitivity in Bayesian models has been adopted specifically for variational methods in Giordano et al. (2018). Forming the linear approximation incurs a one-time cost of computing and inverting the Hessian matrix of the ELBO. In our applications, we solve the linear system with the conjugate gradient algorithm (Nocedal and Wright, 2006), which requires only Hessian-vector-products and allows us to avoid instantiating the full Hessian in memory. Chapter 3 applies these ideas to a Dirichlet process mixture model, and we evaluate the sensitivity of posterior inferences to the Dirichlet process stick-breaking distribution.

Finally, a frequentist approach to data analysis requires consideration of the randomness inherent in the data collection process. The bootstrap is a technique to assess the variability of the model fit with respect to randomness in data collection, while cross-validation evaluates the ability of the model to extrapolate to unseen data. In the context of variational inference, both the bootstrap and cross-validation require solving Equation 1.1 for each subsample of the data. When a single solve is already computationally expensive, the linear growth of computation with the number of bootstrap samples or cross-validation folds may be undesirable. In lieu of refitting the model for each bootstrap sample or cross-validation fold, Chapters 4 and 5 present an application of the the *infinitesimal jackknife* (IJ) (Jaeckel, 1972; Efron, 1982), which linearly approximates the results from repeated optimizations. As with evaluating prior sensitivity, forming the linear approximation incurs a one-time cost of computing and inverting a Hessian matrix. However, subsequent IJ approximations to each bootstrap sample or cross-validation fold require only a single matrix-vector multiplication followed by a vector-vector addition. On the applications studied in Chapters 4 and 5, the computational trade-off is extremely favorable, with the IJ running up to an order of magnitude faster than computing the refits.

The final chapter of this dissertation (Chapter 6) presents a technique to lower the variance of stochastic gradients of the ELBO. Lower variance stochastic gradients leads to faster convergence of stochastic gradient descent and thus improves the runtime of variational methods. We consider the case where $\mathbf{z}$ is a discrete random vector (or has some components that are discrete). In such cases, the reparameterization trick (Kingma and Welling, 2013; Rezende et al., 2014) does not apply. The stochastic gradient estimators that do apply, such as the REINFORCE estimator (Williams, 1992), are often too high-variance to be used in practice without additional modification. We propose a variance reduction

technique that applies to any discrete-distribution stochastic gradient estimate, such as REINFORCE. We prove the variance reduction by showing that our technique is an instance of Rao-Blackwellization.

The remainder of the introduction presents a conceptual overview of the methodologies we use and develop. Each has the purpose of scaling variational inference to the data problems encountered in this dissertation.

## Amortization scales inference to data with replicated structure

A common class of models encountered in Bayesian statistics and statistical machine learning involve a collection of $n$ observations, $x_1, ..., x_n$, each with a *local* latent variable $z_i$. Optionally, the model may have a *global* latent variable $\beta$. The joint distribution factorizes as

$$p(\mathbf{x}_{1:n}, \mathbf{z}_{1:n}, \beta) = p(\beta) \prod_{i=1}^{n} p(x_i, z_i|\beta). \tag{1.3}$$

In mean-field variational inference, each latent variable is independent in the approximate posterior. The variational distribution fully factorizes:

$$q_{\lambda,\boldsymbol{\phi}}(\beta, \mathbf{z}_{1:n}) = q_\lambda(\beta) \prod_{i=1}^{n} q_{\phi_i}(z_i). \tag{1.4}$$

The global variational parameter $\lambda$ governs the distribution of $\beta$; the local variational parameters $\boldsymbol{\phi} = (\phi_1, ..., \phi_n)$ govern the distribution of the local latent variables $z_1, ..., z_n$.

With the factorization of the joint likelihood (Equation 1.3) and the mean-field assumption (Equation 1.4), the ELBO decomposes as

$$\begin{aligned}\mathcal{L}(\lambda, \boldsymbol{\phi}; x_{1:n}) &= \mathbb{E}_{q_{\lambda,\boldsymbol{\phi}}}\Big[\log p(x_{1:n}, z_{1:n}, \beta) - \log q_{\lambda,\boldsymbol{\phi}}(\beta, z)\Big] \\ &= \mathbb{E}_{q_{\lambda,\boldsymbol{\phi}}}\Big[\log \frac{p(\beta)}{q_\lambda(\beta)} + \sum_{i=1}^{n} \log \frac{p(x_i, z_i|\beta)}{q_{\phi_i}(z_i)}\Big].\end{aligned} \tag{1.5}$$

Variational inference optimizes Equation 1.5 for $\lambda$ and $\boldsymbol{\phi}$.

In the presence of a global latent variable, the optimization is not separable – each $\phi_i$ is coupled to $\lambda$ by the terms $\mathbb{E}_q[\log p(x_i, z_i|\beta)]$ in the objective. Therefore, when a new observation $x_{n+1}$ is added to the dataset, the optimization of $\phi_{n+1}$ requires all previous $\phi_1, ..., \phi_n$ and $\lambda$ to be updated. In applications where data is collected online, re-optimizing the ELBO for all parameters, global and local, whenever a data point arrives may be inefficient.

Online methods for variational inference have been proposed (Hoffman et al., 2010; Wang et al., 2011; Hoffman et al., 2013). These methods keep a weighted running average of the global variational parameter $\lambda^{(n)}$ that is updated with the arrival of each data point $x_n$. When $x_{n+1}$ is observed, the ELBO (Equation 1.5) is optimized for $\phi_{n+1}$ with the global variational

parameter fixed at $\lambda^{(n)}$. With the global variational parameter fixed, the problem becomes separable: the optimal $\phi_{n+1}$ does not depend on the previous local variational parameters $\phi_1, ..., \phi_n$ or the previous data points $x_1, ..., x_n$. As $n \to \infty$, $\lambda^{(n)}$ converges to a stationary point of the ELBO (Hoffman et al., 2010, 2013).

However, numerical optimization is still required for each newly observed data point; instead of a full re-optimization of the ELBO over the entire dataset $x_1, ..., x_{n+1}$, we need an online update of $\phi_{n+1}$ using only fresh data $x_{n+1}$. A fundamental issue is that the the number of variational parameters grows linearly with the number of data points $n$.

*Amortized variational inference* (Kingma and Welling, 2013; Zhang et al., 2019) avoids direct optimization of the $\phi_i$'s, and instead defines a function $f_\omega$ that maps each data point $x_i$ to its local variational parameter $\phi_i$. $\omega$ is a real-valued vector that parameterizes the function. For example, $f_\omega$ may be a neural network, in which case $\omega$ are network weights. The amortized variational distribution is

$$q_{\lambda,\omega}(\beta, \mathbf{z}_{1:n}|\mathbf{x}_{1:n}) = q_\lambda(\beta) \prod_{i=1}^{n} q_\omega(z_i|x_i)$$
$$\text{where} \quad q_\omega(z_i|x_i) := q_{\phi_i}(z_i)\Big|_{\phi_i = f_\omega(x_i)}. \tag{1.6}$$

Unlike Equation 1.4, the variational distribution of $z_i$ now has explicit dependence on data point $x_i$.

Instead of optimizing $\phi_1, ..., \phi_n$, the amortized approach optimizes $\omega$, which is shared across all data points. Recalling the ELBO from Equation 1.5, the amortized variational approach solves

$$\underset{\lambda,\omega}{\text{argmax}} \; \mathcal{L}\Big(\lambda, \big(f_\omega(x_i)\big)_{i=1}^n; \mathbf{x}_{1:n}\Big). \tag{1.7}$$

Because $\omega$ is shared for all data points, the number of parameters to be optimized is constant with respect to the number of data points $n$.

After $f_{\omega^*}$ is fit by solving Equation 1.7 at $x_1, ..., x_n$, the local parameter for a new data point $x_{n+1}$ is estimated to be $\phi_{n+1} = f_{\omega^*}(x_{n+1})$. Only an evaluation of $f_{\omega^*}$ is needed; re-optimization is not required.

The computational speedup of the amortized approach may be substantial. Chapter 2 of this dissertation applies amortized variational inference to catalog light sources in astronomical surveys. Here, $x_1, ..., x_n$ are disjoint subimages of the sky; $z_i$ are the latent characteristics (location, flux, color) of any light sources appearing in subimage $x_i$. The mapping $f_\omega$ is a convolutional neural network. In our implementation, finding an approximate variational posterior required 20 minutes on a $40 \times 40$ arcsecond patch of the sky. For comparison, the Sloan Digital Sky Survey scans an $80 \times 7200$ arcminute region of the sky in a single night. Without amortization, repeatedly optimizing the variational objective on each $40 \times 40$ arcsecond patch would require several months to process one night of data collection.

On the other hand, in an amortized approach, inference after the initial 20 minute optimization requires only a forward pass through the neural network, a computation which

takes a tenth of a second on each $40 \times 40$ arcsecond region. Even over large regions the inference can be fast, as it is straightforward to parallelize neural network evaluations on a GPU.

## Wake-sleep optimization takes advantage of complete data

We chose to employ neural networks for cataloging astronomical surveys partly due to their success on other image classification tasks such as ImageNet (Russakovsky et al., 2015). On such image classification tasks, the network is commonly fit in a supervised fashion with large amounts of labeled observations. The lack of available labels may be a bottleneck to neural network training in some settings. However, in astronomy applications, knowledge of the physical system often give rise to reasonably realistic simulated data. For example, Lanusse et al. (2017) and Hezaveh et al. (2017) trained neural networks on simulated images to detect Einstein rings, a rare object found in astronomical surveys and important for dark matter research. Because training data was generated from a simulator, ground truth labels (i.e. ring exists or not) are known. Moreover, simulated data is useful because the network can be trained on an essentially unlimited number of examples – the bottleneck is the amount of computational resources, not the availability of labeled observations.

In our work, we also use simulated data to fit the amortized variational distribution $q_\omega(z|x)$, which is encoded by a neural network $f_\omega$. Instead of directly optimizing the ELBO, we fit $q_\omega(z|x)$ using the *wake-sleep algorithm* (Hinton et al., 1995; Bornschein and Bengio, 2014; Le et al., 2020). The wake-sleep optimization objective takes advantage of simulated, complete data $(x, z)$ sampled from the model $p$. By fitting $f_\omega$ in a supervised fashion on complete pairs $(x, z)$ rather than on observations $(x_1, ..., x_n)$ alone (as is the case in ELBO optimization), we find that the resulting approximate posterior is better able to avoid shallow local optima in the KL (Chapter 2).

The wake-sleep algorithm applies to a special case of global/local models (Equation 1.3) where $\beta$ is a fixed (potentially unknown) model parameter, rather than a random variable. In this case, we write the likelihood as

$$p_\beta(\mathbf{x}_{1:n}, \mathbf{z}_{1:n}) = \prod_{i=1}^{n} p_\beta(x_i, z_i),$$

and the exact posterior also factorizes:

$$p_\beta(\mathbf{z}_{1:n}|\mathbf{x}_{1:n}) = \prod_{i=1}^{n} p_\beta(z_i|x_i).$$

With $\beta$ a model parameter, the ELBO (Equation 1.5) is a summation over $n$ terms,

$$\mathcal{L}(\beta, \omega; \mathbf{x}_{1:n}) = \sum_{i=1}^{n} \mathbb{E}_{q_\omega(z_i|x_i)}\Big[\log \frac{p_\beta(x_i, z_i)}{q_\omega(z_i|x_i)}\Big] \tag{1.8}$$

$$= \sum_{i=1}^{n} \Big\{ \log p_\beta(x_i) - \mathrm{KL}\big(q_\omega(z_i|x_i) \parallel p_\beta(z_i|x_i)\big)\Big\}. \tag{1.9}$$

For a fixed $\beta$, optimizing the ELBO for $\omega$ minimizes the KL divergence between $q_\omega(z_i|x_i)$ and $p_\beta(z_i|x_i)$ averaged over the observed data points $x_1, ..., x_n$.

One can optimize Equation 1.8 jointly for the model parameter $\beta$ and the variational parameter $\omega$.[1] Variational expectation-maximization (EM) (Neal and Hinton, 2000; Beal and Ghahramani, 2002) is a technique that optimizes Equation 1.8 by block coordinate ascent: it alternates between an "E-step" that optimizes $\omega$ and an "M-step" that optimizes $\beta$.

The wake-sleep algorithm is closely related to variational EM in that it alternates between optimizing the model parameter $\beta$ and the variational parameter $\omega$. The "wake" phase optimizes $\beta$ and is equivalent to the M-step of variational EM. The "sleep" phase optimizes $\omega$, but targets a different objective than the ELBO. The sleep phase minimizes

$$\mathcal{L}^{sleep}(\omega; \beta) = \mathbb{E}_{p_\beta(x)}\Big[\mathrm{KL}\big(p_\beta(z|x) \parallel q_\omega(z|x)\big)\Big], \tag{1.10}$$

for a fixed $\beta$.

Note the differences between Equation 1.9 and Equation 1.10. Firstly, the arguments to the KL divergence are transposed – recall that the KL, being a divergence, is not symmetric. Secondly, for a fixed $\beta$, $\omega$ is optimized such that divergence between $p_\beta(z|x)$ and $q_\omega(z|x)$ is small on average over all possible data points $x \sim p_\beta$. Previously in Equation 1.9, the average is over the $n$ observed data points.

Chapter 2 shows that minimizing Equation 1.10 simplifies to solving

$$\underset{\omega}{\mathrm{argmin}} \ \mathbb{E}_{p_\beta(x,z)}\Big[ - \log q_\omega(z|x)\Big]. \tag{1.11}$$

We minimize Equation 1.11 with stochastic gradient descent (SGD). This amounts to sampling complete data $(x, z) \sim p_\beta(x, z)$ at each step and evaluating the loss $-\log q_\omega(z|x)$. This loss encourages $q_\omega(z|\cdot)$ to map data $x$ to distributions that place large mass on the corresponding value of the latent variable $z$. Fitting with simulated, complete data enables the neural network to see many examples, and in the problem of cataloging astronomical surveys (Chapter 2), the resulting variational posterior is able to avoid shallow local optima in the ELBO.

[1] This can be viewed as a special case of variational inference where the variational distribution on $\beta$ is a point mass.

## Local sensitivity extrapolates posterior statistics to model perturbations

Statistical analysis typically involves fitting multiple models and assessing goodness-of-fit or predictive performance. Even when a candidate model is chosen, it remains important to evaluate the sensitivity of the resulting inferences to possibly arbitrary model choices.

A Bayesian approach requires the user to specify a prior distribution. In some applications, the prior is chosen for computational convenience. In other applications, a range of priors may be plausible, and it may be unclear why some choices would be preferable over others. Therefore, evaluating the sensitivity of posterior inferences to prior choices is a key step in Bayesian data analysis.

A conceptually straightforward way to demonstrate prior sensitivity would be to compute the posterior distribution (or posterior summaries of interest such as modes, medians, modes) for various choices of the prior. In the context of variational inference, this amounts to resolving the optimization problem Equation 1.1 for each prior choice, which may be expensive.

Instead of refitting the variational approximation at each possible prior choice, we employ *local sensitivity* (Gustafson, 2000) to approximate the refitting process. We use the local sensitivity of a model fitted under some initial prior to predict the posterior summaries under alternative priors. Local prior sensitivity in Bayesian models have been adopted specifically for variational methods in Giordano et al. (2018).

To define local sensitivity, let $\alpha$ be a real-valued hyperparameter vector in a model; in the context of prior sensitivity, $\alpha$ is a parameter of the prior distribution. Generically, let $p_\alpha(\mathbf{x}, \mathbf{z})$ be the joint likelihood and $\eta^*(\alpha)$ be the optimal variational parameters for the model with hyperparameter $\alpha$. $\eta^*$ depends on $\alpha$ through optimization,

$$\eta^*(\alpha) := \underset{\eta}{\text{argmax}}\ \mathcal{L}(\eta, \alpha), \tag{1.12}$$

where $\mathcal{L}(\eta, \alpha)$ is the ELBO as a function of the variational parameter $\eta$ and model parameter $\alpha$,

$$\mathcal{L}(\eta, \alpha) = \mathbb{E}_{q_\eta(\mathbf{z})}\Big[\log p_\alpha(\mathbf{x}, \mathbf{z}) - \log q_\eta(\mathbf{z})\Big]. \tag{1.13}$$

Any approximated posterior quantity of interest is a function of the variational parameter $\eta$. Let $G(\eta)$, a real-valued function, denote the posterior quantity. For example, the posterior quantity can often be expressed as an expectation over some function $g$ of the latent variables, in which case $G(\eta) = \mathbb{E}_{q_\eta}[g(\mathbf{z})]$. The local sensitivity of posterior statistic $G$ at prior parameter $\alpha_0$ is defined as

$$S^G_{\alpha_0} := \frac{d}{d\alpha} G(\eta^*(\alpha))\bigg|_{\alpha=\alpha_0}, \tag{1.14}$$

provided that appropriate derivatives of $G$ and the ELBO exist, and the optimizer $\eta^*(\alpha_0)$ is strict (for details, see Chapter 3, or assumptions 1-4 in Giordano et al. (2018)). The

magnitude of this derivative captures the sensitivity of the posterior quantity $G$ to the hyperparameter $\alpha$.

In our work, we go beyond treating Equation 1.14 as a simply a measure of sensitivity and use the derivative to extrapolate $G$ to different priors by employing a first order approximation. Because we are specifically interested interested in extrapolation, we only linearize the mapping from $\alpha \mapsto \eta^*(\alpha)$. Define the *linearized variational parameter* as

$$\eta^{lin}(\alpha) := \eta^*(\alpha_0) + \frac{\partial \eta^*(\alpha)}{\partial \alpha}\Big|_{\alpha=\alpha_0} (\alpha - \alpha_0).$$

We then approximate

$$G(\eta^*(\alpha)) \approx G(\eta^{lin}(\alpha)).$$

We retain the nonlinearity in the mapping from $\eta \mapsto G(\eta)$. Only the computationally expensive operation $\alpha \mapsto \eta^*(\alpha)$, which requires an optimization solve, is linearized.

Computing the linearized variational parameter requires the initial variational parameter $\eta^*(\alpha_0)$ and the derivative $\frac{\partial \eta^*}{\partial \alpha}$ at $\alpha_0$. With these two quantities, $\eta^{lin}(\alpha)$ can be quickly evaluated at any $\alpha$ with simple matrix-vector operations. No re-optimization is required.

We briefly outline the computation of the derivative $\frac{\partial \eta^*}{\partial \alpha}$. Assuming the appropriate derivatives exist and that the optimization domain of $\eta$ in Equation 1.13 is unconstrained, we have for every $\alpha$ that

$$\frac{\partial \mathcal{L}(\eta^*(\alpha), \alpha)}{\partial \eta} = 0. \tag{1.15}$$

Viewing both sides of Equation 1.15 as functions of $\alpha$ and taking derivatives with respect to $\alpha$, we obtain

$$\left( \frac{\partial^2 \mathcal{L}(\eta, \alpha)}{\partial \eta \partial \eta^T} \frac{\partial \eta^*(\alpha)}{\partial \alpha} + \frac{\partial^2 \mathcal{L}(\eta, \alpha)}{\partial \alpha \partial \eta^T} \right) \Bigg|_{\eta=\eta^*(\alpha)} = 0.$$

Finally, solving for $\frac{\partial \eta^*}{\partial \alpha}$ implies that

$$\frac{\partial \eta^*(\alpha)}{\partial \alpha} = - \left[ \frac{\partial^2 \mathcal{L}(\eta, \alpha)}{\partial \eta \partial \eta^T} \right]^{-1} \frac{\partial^2 \mathcal{L}(\eta, \alpha)}{\partial \alpha \partial \eta^T} \Bigg|_{\eta=\eta^*(\alpha)}, \tag{1.16}$$

assuming that the Hessian matrix $\frac{\partial^2 \mathcal{L}}{\partial \eta \partial \eta^T}$ is nonsingular. The necessary derivatives can be computed with modern automatic differentiation tools, and Equation 1.16 can thus be used to compute $\frac{\partial \eta^*(\alpha)}{\partial \alpha}$ at any $\alpha_0$. In our work, we use `Autograd` (Maclaurin et al., 2015) and its more recent iteration, `Jax` (Bradbury et al., 2018). Both are publicly available automatic differentiation packages in Python.

Note that the Hessian matrix is square with dimensions equal to that of the variational parameter $\eta$. When the dimension of $\eta$ is large, calculating the Hessian is time-consuming,

and it may even be impossible to store in memory. In our applications, we solve the linear system in Equation 1.16 using the conjugate gradient algorithm (Nocedal and Wright, 2006), which requires only Hessian-vector-products, so the full Hessian is not instantiated in memory. When the model has global/local structure (Equation 1.3), we take additional advantage of sparsity in the Hessian by making use of Schur complements (see Chapter 3). In any case, the Hessian inversion is a one-time computational cost that is required only at $\alpha_0$; in our applications, this cost can be an order of magnitude faster than a new optimization solve. After the one-time Hessian inversion, evaluating $\eta^{lin}(\alpha)$ at many choices of $\alpha$ is fast.

Chapter 3 demonstrates these techniques on a Dirichlet process mixture model used to infer latent population structure from genetic data. One posterior quantity of interest is the number of latent populations in data set. We evaluate the sensitivity of the approximate posterior to both the concentration parameter of the Dirichlet process and the functional form of the stick-breaking distribution.

## The infinitesimal jackknife approximates data resampling

Frequentist statistical analysis focuses on accounting for the randomness inherent in the data collection process. On top of measuring sensitivity to modeling choices as discussed previously, a frequentist perspective allows us to evaluate how resulting inferences would change should different data sets be observed. Common techniques to approximate the randomness in data collection include cross-validation and the bootstrap. These techniques estimate the true, unknown data generating distribution via the empirical distribution; proxy datasets are then formed by re-sampling data points from the empirical distribution (the bootstrap) or deterministically removing data points from the original data set (cross-validation).

In Chapters 4 and 5, we cluster time-course gene expression data with a Bayesian mixture model. We wish to use cross-validation to select the complexity of the time series model, while we use bootstrap sampling to assess the stability of the inferred clusters. However, in our variational inference approach, each bootstrap sample or cross-validation fold requires re-optimization of the ELBO.

To avoid expensive re-optimization, we adapt the linear approximation from the previous subsection to data resampling, so that the model (and the corresponding linear approximation) needs to only be calculated once. Historically, this method is known as the infinitesimal jackknife (IJ) (Jaeckel, 1972). The old idea is revived here as the derivatives necessary for the infinitesimal jackknife are straightforward to compute with modern automatic differentiation tools, without the need for manual derivations.

We present the infinitesimal jackknife in the context of variational inference for the global / local model (Equation 1.3) with a mean-field variational distribution (Equation 1.4). Let $\eta := (\lambda, \boldsymbol{\phi})$ be the variational parameters. We augment the ELBO in Equation 1.5 with a weight vector $\boldsymbol{w} = (w_1, ..., w_n)$, one for each data point $x_i$:

$$\boldsymbol{\mathcal{L}}(\eta, \boldsymbol{w}) = \mathbb{E}_{q_\eta}\Big[\log \frac{p(\beta)}{q_\lambda(\beta)} + \sum_{i=1}^{n} w_i \log \frac{p(x_i, z_i|\beta)}{q_{\phi_i}(z_i)}\Big].$$

The optimal variational parameter is now a function of $\boldsymbol{w}$,

$$\eta^*(\boldsymbol{w}) := \underset{\eta}{\text{argmax}}\ \mathcal{L}(\eta, \boldsymbol{w}).$$

The initial variational posterior is evaluated with weights $w_i = 1$ for all $i$. Let $\mathbf{1}_n$ be the ones vector of length $n$. A bootstrap resample can be represented by setting $w_i$ to non-negative integers such that $\sum_i w_i = n$; a leave-$k$-out cross-validation fold can be represented by setting $w_i = 0$ for $k$ observations and $w_i = 1$ otherwise. For any weight vector $\boldsymbol{w}$, we approximate the optimal variational parameters with

$$\eta^{\text{IJ}}(\boldsymbol{w}) = \eta^*(\mathbf{1}_n) + \frac{\partial \eta^*(\boldsymbol{w})}{\partial \boldsymbol{w}}\Big|_{\boldsymbol{w}=\mathbf{1}_n} (\boldsymbol{w} - \mathbf{1}_n)$$

From the discussion in the previous subsection, the derivative $\frac{\partial \eta^*}{\partial \boldsymbol{w}}$ can be computed using Equation 1.16 with $\boldsymbol{w}$ in lieu of $\alpha$.

As discussed before, the derivative computation $\frac{\partial \eta^*}{\partial \boldsymbol{w}}$ incurs a one-time cost of computing and inverting a Hessian matrix. However, the cost is extremely favorable in comparison to repeatedly optimizing the ELBO for each bootstrap sample or cross-validation fold. In our experiments, the IJ well-approximates a bootstrap analysis of cluster stability (Chapter 4), and it selected the same complexity parameter as exact cross-validation (Chapter 5).

## Inference about discrete random variables

Discrete random variables appear in several applications discussed above. In cataloging astronomical surveys, the number of light sources present in an image is a discrete latent variable. In our clustering applications, a discrete latent variable encodes the cluster memberships in a mixture model.

The final chapter of this dissertation (Chapter 6) presents a method for lowering the variance of stochastic gradients of the ELBO when the latent vector contains a discrete component. Our method considers objective functions of the form

$$\mathcal{L}(\eta) = \mathbb{E}_{q_\eta(z)}[f_\eta(z)], \tag{1.17}$$

where $z$ is a univariate discrete random variable with $K \leq \infty$ categories.[2] The ELBO is of this form, with $f_\eta(z) = \log p(x, z) - \log q_\eta(z)$.

Because $z$ is discrete, the expectation in Equation 1.17 and its derivatives can be written as a summation of $K$ terms. However, when $K$ is large or infinite, the exact summation is intractable. To optimize $\mathcal{L}(\eta)$, we seek low-variance stochastic gradients and run stochastic gradient descent.

[2] A multivariate $z$ can be treated as a single discrete variable over the Cartesian product of the sample spaces. The applications discussed in Chapter 6 also include situations where only some components of $z$ are discrete and others are continuous.

A general purpose stochastic gradient is the REINFORCE estimator (Williams, 1992). This estimator samples $z \sim q_\eta$, and estimates the gradient $\nabla \mathcal{L}(\eta)$ with

$$g_{\text{rf}}(z) := f_\eta(z) \nabla \log q_\eta(z) + \nabla f_\eta(z). \tag{1.18}$$

While the REINFORCE estimator is unbiased for the true gradient, meaning that $\nabla \mathcal{L}(\eta) = \mathbb{E}_{q_\eta}[g_{\text{rf}}(z)]$, the variance of this estimator is often too large to be used in practice. Thus, *control variate* methodology (Mnih and Gregor, 2014; Gu et al., 2016; Tucker et al., 2017; Grathwohl et al., 2018) has been proposed to reduce the variance by adding a zero-mean random variable to Equation 1.18 with a chosen correlation structure.

Chapter 6 presents a method for achieving further variance reduction given any discrete-distribution stochastic-gradient estimator $g$ (e.g., REINFORCE or REINFORCE with control variates). Our estimator is constructed by deterministically evaluating $g$ at the categories on which $q_\eta$ places its largest mass. Let $\mathcal{C}_k$ be the categories with the $k$ largest probabilities under $q_\eta$. Our modified gradient estimator is

$$\hat{g}(v) = \sum_{z \in \mathcal{C}_k} q_\eta(z) g(z) + (1 - q_\eta(\mathcal{C}_k)) g(v), \tag{1.19}$$

where $v$ is a sample from $q_\eta$, conditional on the event that $v \notin \mathcal{C}_k$.

Note that the first term of this estimator is deterministic and only the second term is random. The second term is scaled by the probability of not belonging in $\mathcal{C}_k$, which is small when $q_\eta$ is concentrated on a small number of categories. Multiplying the random term by a small factor reduces the variance; most of the variance has been "absorbed" by the deterministic term.

Our method is especially beneficial in scenarios where $q_\eta$ places its mass on only a few categories out of many. This is the case in our applications, because $q_\eta$ is an approximate posterior, which typically becomes more concentrated as more data is observed.

We guarantee the variance reduction by showing that our technique is an instance of Rao-Blackwellization. More precisely, we show that given a budget of $N$ evaluations of $g$, a variant of Equation 1.19 reduces the variance of $g(z)$ by at least a factor of $1/N$, and hence Equation 1.19 improves on the variance reduction obtained by averaging $N$ independent samples of $g(z)$.

# Chapter 2

# Variational inference for deblending crowded starfields

Astronomical images record the arrival of photons from distant light sources. Astronomical catalogs are constructed from these images. Catalogs label light sources as stars, galaxies, or other objects; they also list the physical characteristics of light sources such as flux, color, and morphology. These catalogs are the starting point for many downstream analyses. For example, Bayestar used a catalog of stellar fluxes and colors to construct the 3D distribution of interstellar dust (Green et al., 2019). Catalogs of galaxy morphologies have been used to validate theoretical models of dark matter and dark energy (Abbott et al., 2018).

A light source, be it a star or a galaxy, produces a peak intensity of brightness in an image. When light sources are well separated, catalog construction is straightforward: characteristics of each light source, such as flux, can be estimated by analyzing intensities at the peak and surrounding pixels. However, in images crowded with many light sources, observed intensities may result from the combined light of multiple sources. Source separation, or *deblending*, is the task of differentiating and characterizing individual light sources from a mixture of intensities on an image. A key challenge in deblending is inferring whether an observed intensity is in fact blended, that is, whether it is composed of a single bright source or multiple dimmer sources.

Deblending is challenging for several reasons. First, it is an unsupervised problem without ground truth labeled data. Second, it is a problem with a sample size of one: there is only one night sky, which is imaged many times, and the collected survey images capture overlapping regions of it. Third, for blended fields, the properties of light sources are ambiguous; therefore, providing calibrated uncertainties for catalog construction is as important as making accurate predictions. Finally, the scale of the data is immense. The upcoming Rubin Observatory Legacy Survey of Space and Time (LSST), scheduled to begin data collection in 2022, is expected to produce 60 petabytes of astronomical images over its lifetime (LSST).

As more powerful telescopes are developed, and their ability to detect more distant light sources improves, the density of light sources in the images they capture will only increase. For instance, Bosch et al. (2018) estimates that 58% of light sources are blended in images

captured by the Subaru Telescope's Hyper Suprime-Cam, and that percentage is expected to increase for LSST. Therefore, developing a method that reliably characterizes light sources, even in cases of significant blending, advances any astronomical research in which conclusions about the physical universe are derived from estimated catalogs.

We focus on cataloging applications where all light sources are well modeled as points without spatial extent. Point-source-only models are applicable to surveys such as DECam (Schlafly et al., 2018), which imaged the center of the Milky Way, and WISE (Wright et al., 2010), whose telescope resolution did not allow for differentiation between stars and galaxies. In this work, we use the globular cluster M2, a region imaged by the Sloan Digital Sky Survey (SDSS) that is densely populated with stars, as a test bed for our method.

**From software pipelines to probabilistic cataloging**

Traditionally, most cataloging has been performed using software pipelines. These pipelines are algorithms that usually involve the following stages: locating the brightest peaks, estimating fluxes, and subtracting the estimated light source. These stages may be performed iteratively. Pipelines do not normally produce statistically calibrated error estimates that propagate the uncertainty that accumulates in each of the steps. Failure to properly accumulate error at each step results in unreliable catalogs for images in which ambiguity exists in identifying sources and estimating their characteristics. For example, PHOTO (Lupton et al., 2001), the default cataloging pipeline used by SDSS, failed to produce a catalog on the globular cluster M2 (Portillo et al., 2017).

In contrast, *probabilistic* cataloging posits a statistical model consisting of a likelihood for the observed image given a catalog and a prior distribution over possible catalogs (Portillo et al., 2017; Brewer et al., 2013; Feder et al., 2020). Instead of deriving a single catalog, probabilistic cataloging produces a posterior distribution over the set of all possible catalogs. Uncertainties are quantified by the posterior distribution. For example, in an image with an ambiguously blended bright peak, some catalogs sampled from the posterior would contain multiple dim light sources while others would contain one bright source. The relative density the posterior distribution places on one explanation over another represents the statistical confidence in that explanation. Moreover, a distribution over the set of all catalogs induces a distribution on any estimate derived from a catalog. Therefore, calibrated uncertainties can be propagated to downstream analyses.

Previous work on probabilistic cataloging employed Markov chain Monte Carlo (MCMC) to sample from the posterior distribution. The MCMC procedure in Portillo et al. (2017) and Feder et al. (2020) was called PCAT, short for "Probabilistic CATaloging."[1] A difficulty in any probabilistic cataloging approach is that the number of sources in a catalog is unknown and random, so the latent variable space is transdimensional. PCAT sampled transdimensional catalogs with reversible jump MCMC (Green, 1995), in which auxiliary variables are added to encode the "birth" and "death" of light sources in the Markov chain.

[1] We use "probabilistic cataloging" to refer to any method that produces a posterior over possible catalogs, whereas "PCAT" refers specifically to the MCMC procedure in Portillo et al. (2017) and Feder et al. (2020).

The computational cost of MCMC for this model is problematic for large-scale astronomical surveys. Early implementations of PCAT required a day to process a $100 \times 100$ pixel image of the M2 globular cluster imaged by SDSS (Portillo et al., 2017). More recent implementations running inexact MCMC brought the runtime down to 30 minutes (Feder et al., 2020). In any case, a $100 \times 100$ pixel image covers only a $0.66 \times 0.66$ arcminute patch of the sky. For comparison, in one night, SDSS scans a region on the order of $100 \times 1000$ arcminutes. Extrapolating the 30-minute runtime, PCAT would take on the order of ten years to process a nightly SDSS run.

As an alternative to MCMC, Regier et al. (2019) produced an approximate posterior using variational inference. Variational inference (VI) considers a family of candidate approximate posteriors and employs numerical optimization to find the distribution in the family closest in KL divergence to the exact posterior (Jordan et al., 1999; Wainwright and Jordan, 2008; Blei et al., 2017). With a sufficiently constrained family of distributions, the VI optimization problem can be solved orders of magnitude faster than MCMC runs.

However, Regier et al. (2019) is limited in that the number of light sources in a given image is treated as known and fixed – it had to be set using a preprocessing routine. They avoided the transdimensional latent variable space in order to have a tractable objective for numerical optimization.

**Our contribution**

We propose *StarNet*, an approach to deblending that employs several recent VI innovations (Zhang et al., 2019; Le et al., 2020). Unlike Regier et al. (2019), our VI approach is able to handle arbitrary probabilistic models, including a transdimensional model with an unknown number of sources. Section 2.1 introduces the statistical model, which is similar to the model used in PCAT.

Secondly, again unlike Regier et al. (2019), we employ *amortization*, which enables StarNet to scale inference to large astronomical surveys. In amortized variational inference, a neural network maps input images to an approximate posterior. Following a one-time cost to fit the neural network, inference on new images requires just a single forward pass. Rapid inference is available without the need to re-run MCMC or numerically optimize VI for each new image. For StarNet, the forward pass on a $100 \times 100$ pixel image takes 0.2 seconds (vs. 30 minutes for inference using PCAT). Section 2.2 details the variational distribution and neural network architecture in StarNet.

Finally, and critically, StarNet is fit using the wake-sleep algorithm (Hinton et al., 1995), which does not target the same KL divergence traditionally used in variational inference. Traditionally, variational inference minimizes the "reverse" KL divergence between the approximate posterior $q$ and the exact posterior $p$. Reverse KL is defined as the $q$-weighted average difference between $\log q$ and $\log p$. Wake-sleep instead fits the approximate posterior using the "forward" KL divergence, which weights the difference between $\log p$ and $\log q$ by $p$. Section 2.3 details the wake-sleep procedure.

In this application, optimizing the forward KL produces more reliable approximate posteriors than optimizing the traditional reverse KL (Section 2.4). In particular, optimizing

the forward KL involves sampling *complete* data—images and their corresponding catalogs—from their joint likelihood and fitting the network in a supervised fashion. Taking advantage of complete data allows the network to better avoid shallow local minima where the approximate posterior is far from the exact posterior in terms of KL divergence.

The wake-sleep algorithm has been used in previous research to train deep generative models (Hinton et al., 1995; Bornschein and Bengio, 2014; Le et al., 2020). However, to the best of our knowledge, this is the first application of wake-sleep for scientific purposes. Specifically, we use wake-sleep for inference to find a latent space that is interpretable: it is the set of all possible astronomical catalogs.

We applied StarNet to the M2 globular cluster as imaged by SDSS. StarNet was more accurate than both the MCMC-based cataloger PCAT and traditional cataloging approaches while running $100,000$ times faster than the former (Section 2.5). Code to reproduce our results is publicly available in a GitHub repository.[2]

## 2.1 The generative model

In crowded starfields, such as globular clusters and the Milky Way, the vast majority of light sources are stars. An astronomical image records the number of photons that reached a telescope and arrived at each pixel. Typically, photons must pass through one of several filters, each selecting photons from a specified band of wavelengths, before being recorded.

For a given $H \times W$ pixel image with $B$ filter bands, our goal is to infer a catalog of stars. The catalog specifies the number of stars in an image; for each such star, the catalog records its location and its flux, or brightness, in each band. The space of latent variables $\mathcal{Z}$ is the collection of all possible catalogs of the form

$$z := \{N, (\ell_i, f_{i,1}, ..., f_{i,B})_{i=1}^{N}\},$$

where the number of stars in the catalog is $N \in \mathbb{N}$; the location of the $i$th star is $\ell_i \in \mathbb{R}^2$; and the flux of the $i$th star in the $b$th band is $f_{i,b} \in \mathbb{R}^+$.

A Bayesian approach requires specification of a prior over catalog space $\mathcal{Z}$ and a likelihood for the observed images. Our likelihood and prior, detailed below, are similar to previous approaches (Brewer et al., 2013; Portillo et al., 2017; Feder et al., 2020).

### 2.1.1 The prior

The prior over $\mathcal{Z}$ is a marked spatial Poisson process. To sample the prior, first draw the number of stars contained in the $H \times W$ image as

$$N \sim \text{Poisson}(\mu HW), \tag{2.1}$$

[2] `https://github.com/Runjing-Liu120/DeblendingStarfields`.

where $\mu$ is a hyperparameter specifying the average number of sources per pixel. Next, draw locations

$$\ell_1, ..., \ell_N | N \overset{iid}{\sim} \text{Uniform}([0, H] \times [0, W]). \tag{2.2}$$

The fluxes in the first band are from a power law distribution with slope $\alpha$:

$$f_{1,1}, ..., f_{N,1} | N \overset{iid}{\sim} \text{Pareto}(f_{min}, \alpha). \tag{2.3}$$

Fluxes in other bands are described relative to the first band. Like Feder et al. (2020), we define the log-ratio of flux relative to the first band as "color." Colors are drawn from a Gaussian distribution

$$c_{1,b}, ..., c_{N,b} | N \overset{iid}{\sim} \mathcal{N}(\mu_c, \sigma_c^2), \quad b = 2, ..., B. \tag{2.4}$$

Given the flux in the first band $f_{i,1}$ and color $c_{i,b}$, the flux in band $b$ is $f_{i,b} = f_{i,1} \times 10^{c_{i,b}/2.5}$.

Also like Feder et al. (2020), we set the power law slope $\alpha = 0.5$ and use a standard Gaussian for the color prior ($\mu_c = 0$, $\sigma_c^2 = 1$). Section 2.7 evaluates the sensitivity of the resulting catalog to choices of the prior parameters.

### 2.1.2 The likelihood

Let $x_{hw}^b$ denote the observed number of photoelectrons at pixel $(h, w)$ in band $b$. For each band, at every pixel, the expected number of photoelectron arrivals is $\lambda_{hw}^b(z)$, a deterministic function of the catalog $z$. Motivated by the Poissonian nature of photon arrivals and the large photon arrival rate in SDSS and LSST images, observations are drawn as

$$x_{hw}^b | z \overset{ind}{\sim} \mathcal{N}(\lambda_{hw}^b, \lambda_{hw}^b), \quad b = 1, ..., B; \; h = 1, ..., H; \; w = 1, ..., W, \tag{2.5}$$

$$\text{where} \quad \lambda_{hw}^b = I^b(h, w) + \sum_{i=1}^{N} f_{i,b} \mathcal{P}^b\big(h - \ell_{i,1}, w - \ell_{i,2}\big). \tag{2.6}$$

Here, $\mathcal{P}^b$ is the point spread function (PSF) for band $b$ and $I^b$ is the background intensity. The PSF is a function $\mathcal{P}^b : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}^+$, describing the appearance of a stellar point source at any 2D position of the image (but ignoring pixelation). Our PSF model is a weighted average between a Gaussian "core" and a power-law "wing," as described in Xin et al. (2018). For each band, the PSF has the form

$$\mathcal{P}(u, v) = \frac{\exp(\frac{-(u^2+v^2)}{2\sigma_1^2}) + \zeta \exp(\frac{-(u^2+v^2)}{2\sigma_2^2}) + \rho(1 + \frac{v^2+u^2}{\gamma\sigma_P^2})^{-\gamma/2}}{1 + \zeta + \rho}. \tag{2.7}$$

The PSF parameters vary by band. Let the collection of PSF parameters across all bands be denoted $\pi := (\sigma_1^{(b)}, \sigma_2^{(b)}, \sigma_P^{(b)}, \gamma^{(b)}, \zeta^{(b)}, \rho^{(b)})_{b=1}^B$.

The background intensity at pixel $(h, w)$ is modeled with an affine function:

$$I^b(h, w) = \beta_0^b + \beta_1^b \times h + \beta_2^b \times w. \tag{2.8}$$

The background parameters $(\beta_0^b, \beta_1^b, \beta_1^b)$ are specific to the band.

StarNet estimates these parameters jointly with the approximate posterior (Section 2.3). Prior work on probabilistic cataloging relied on estimates from the SDSS software pipeline and found the PSF estimates to be suboptimal in crowded starfields (Feder et al., 2020).

## 2.2 Variational inference

The central quantity in Bayesian statistics is the posterior distribution $p(z|x)$. However, in many nontrivial probabilistic models, including our own, the posterior distribution is intractable to calculate. Calculation of the posterior requires us to compute the marginal likelihood, $p(x)$, which involves integrating over the latent variable $z$. In our model, the latent variable space is high dimensional: it is the set of all catalogs. Approximate methods such as MCMC and variational inference are therefore required.

Variational inference (Jordan et al., 1999; Wainwright and Jordan, 2008; Blei et al., 2017) posits a family of distributions $\mathcal{Q}$ and seeks the distribution $q^* \in \mathcal{Q}$ that is closest to the exact posterior in KL divergence. $\mathcal{Q}$ is chosen such that $q^*$ will not be too difficult to find via optimization. We index the distributions in $\mathcal{Q}$ using a real-valued vector $\eta$, then seek $\eta^*$ satisfying

$$\eta^* = \underset{\eta}{\operatorname{argmin}} \operatorname{KL}\Big[\, q_\eta(z|x) \,\|\, p(z|x) \,\Big]. \tag{2.9}$$

Minimizing the KL divergence in Equation 2.9 is equivalent to maximizing the evidence lower bound (ELBO) (Blei et al., 2017):

$$\mathcal{L}_{elbo}(\eta) = \mathbb{E}_{q_\eta(z|x)}\Big[\log p(x, z) - \log q_\eta(z|x)\Big]. \tag{2.10}$$

Computing the ELBO does not require computing the marginal distribution $p(x)$, which is intractable, or the posterior distribution $p(z|x)$, which would be circular.

### 2.2.1 The variational distribution

Traditionally in variational inference, the posterior approximation $q_\eta$ depends on the data $x$ only implicitly, in that $\eta^*$ is chosen according to Equation 2.9. In this case, $q_\eta(z|x)$ is usually written $q_\eta(z)$, suppressing the dependence on $x$. When a new data point $x^{new}$ arrives, finding a variational approximation to the posterior $p(z^{new}|x^{new})$ requires re-solving Equation 2.9 with $x = x^{new}$.

On the other hand, in *amortized* variational inference (Kingma and Welling, 2013; Rezende et al., 2014), $q_\eta$ explicitly depends on the data. A flexible, parameterized function maps input $x$, in this case an observed image, to a real-valued vector characterizing a distribution on

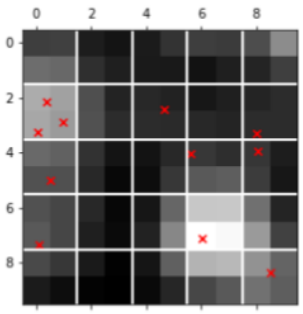



Figure 2.1: Tiling a $10 \times 10$ pixel image into $2 \times 2$ tiles.

the latent space $\mathcal{Z}$. Typically, the function is a neural network, in which case the variational parameters $\eta$ are the neural network weights. After the neural network is fitted with Equation 2.9 using a collection of observed $x$'s, the approximate posterior $q_\eta(z^{new}|x^{new})$ for a new data point $x^{new}$ can be evaluated with a single forward pass through the neural network. No additional run of an optimization routine is needed for a new data point $x^{new}$.

The following subsections detail the construction of our variational distribution.

### The factorization

To make the objective in Equation 2.9 tractable, the family $\mathcal{Q}$ is normally restricted to probability distributions without conditional dependencies between some latent variables. In the most extreme case, known as mean-field variational inference, the variational distribution completely factorizes across all latent variables.

Our factorization has a spatial structure. First, we partition the full $H \times W$ image into disjoint $R \times R$ tiles. $R$ will be chosen such that the probability of having three or more stars in one tile is small. In this way, the cataloging problem decomposes to inferring only a few stars at a time (Section 2.2.1).

Let $S = {}^{H}/_{R}$ and $T = {}^{W}/_{R}$ and assume without loss of generality that $H$ and $W$ are multiples of $R$. For $s = 1, ..., S$ and $t = 1, ..., T$, the tile $\tilde{x}_{st}$ is composed of the pixels,

$$\tilde{x}_{st} = \{x_{hw} : Rs \leq h \leq R(s+1) \text{ and } Rt \leq w \leq R(t+1)\}. \tag{2.11}$$

Figure 2.1 gives an example with $R = 2$.

Let $\tilde{N}^{(s,t)}$ be the number of stars in tile $(s, t)$. Because $\tilde{N}^{(s,t)}$ is random, the cardinality of the set of locations and fluxes in each tile is also random. To handle the trans-dimensional parameter space, we consider a *triangular array* of latent variables on each tile:

$$\tilde{\ell}^{(s,t)} = (\tilde{\ell}^{(s,t)}_{N,i} : i = 1, ..., N; N = 1, 2, ...); \tag{2.12}$$

$$\tilde{f}^{(s,t)} = (\tilde{f}^{(s,t)}_{N,i} : i = 1, ..., N; N = 1, 2, ...), \tag{2.13}$$

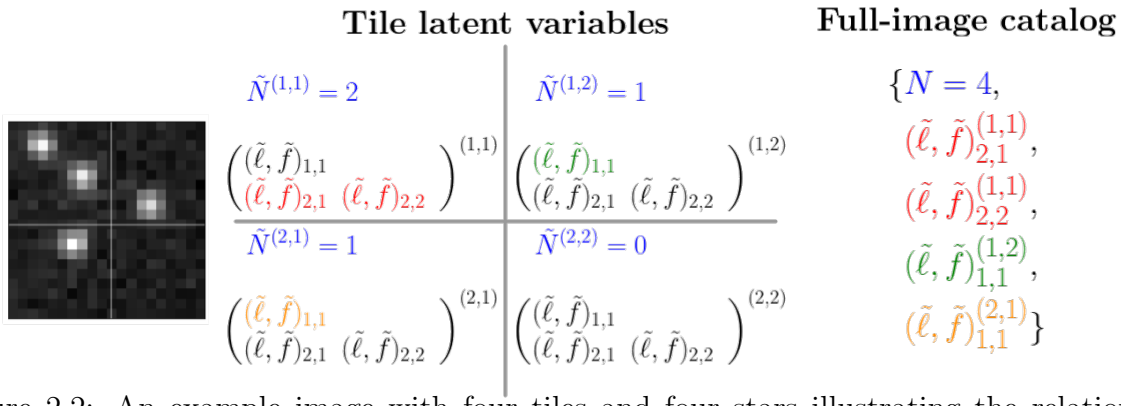



Figure 2.2: An example image with four tiles and four stars illustrating the relationship between the tile latent variables and the full-image catalog. To construct the full-image catalog, we index into the appropriate row of the triangular array on each tile.

where $\tilde{\ell}^{(s,t)}_{N,i}$ and $\tilde{f}^{(s,t)}_{N,i}$ are the elements of the triangular array corresponding to location and fluxes, respectively. Tile locations $\tilde{\ell}^{(s,t)}_{N,i} \in [0, R] \times [0, R]$ give the location of stars within a tile. The fluxes $\tilde{f}^{(s,t)}_{N,i}$ are vectors in $\mathbb{R}^B_+$ (one flux for each band).

Call $(\tilde{N}^{(s,t)}, \tilde{\ell}^{(s,t)}, \tilde{f}^{(s,t)})^{S,T}_{s=1,t=1}$ the *tile latent variables*. The distribution on tile latent variables factorize over image tiles:

$$\tilde{q}_\eta\big((\tilde{N}^{(s,t)}, \tilde{\ell}^{(s,t)}, \tilde{f}^{(s,t)})^{S,T}_{s=1,t=1}|x\big) = \prod_{s=1}^{S}\prod_{t=1}^{T} \tilde{q}_\eta\big(\tilde{N}^{(s,t)}, \tilde{\ell}^{(s,t)}, \tilde{f}^{(s,t)}|x\big). \tag{2.14}$$

Succinctly denote tile latent variables as $\tilde{z}$. The ultimate latent variable of interest is $z = \{N, (\ell_i, f_{i,1}, ..., f_{i,B})^N_{i=1}\}$, the catalog for the full image. There is a natural mapping from $\tilde{z}$ to $z$. First, the number of stars in the full catalog is given by the sum of the stars in each tile, $N = \sum_{s,t} \tilde{N}^{(s,t)}$. Then, for every tile $(s, t)$, we index into the $\tilde{N}^{(s,t)}$-th row of the triangular array of tile latent variables $\tilde{f}^{(s,t)}$ and $\tilde{\ell}^{(s,t)}$. The union of these fluxes and locations over all tiles form the full catalog (tile locations are shifted by the position of the tile in the full image to obtain locations in the full image). See Figure 2.2 for a schematic.

If $\tau$ is the mapping from $\tilde{z}$ to $z$, then the variational distribution on catalogs $z$ is

$$q_\eta(z|x) := \tilde{q}_\eta(\tau^{-1}(z)|x), \tag{2.15}$$

where $\tau^{-1}(z)$ is the pre-image of $z$ under $\tau$.

### Variational distributions on image tiles

We describe the variational distribution on each tile, $\tilde{q}_\eta\big(\tilde{N}^{(s,t)}, \tilde{\ell}^{(s,t)}, \tilde{f}^{(s,t)}|x\big)$. The latent variables also fully factorize within each tile. Dropping the index $(s,t)$ in this subsection,

$$\tilde{N} \sim \text{Categorical}(\omega; 0, ..., N_{max}); \tag{2.16}$$

$$\tilde{\ell}_{\tilde{N},i}/R \sim \text{LogitNormal}(\mu_{\ell_{\tilde{N},i}}, \text{diag}(\nu_{\ell_{\tilde{N},i}})); \tag{2.17}$$

$$\tilde{f}^b_{\tilde{N},i} \sim \text{LogNormal}(\mu_{f^b_{\tilde{N},i}}, \sigma^2_{f^b_{\tilde{N},i}}), \tag{2.18}$$

independently for $i = 1, ..., \tilde{N}$; $\tilde{N} = 1, ..., N_{max}$. $\omega$ is a $(\tilde{N}_{max} + 1)$-dimensional vector on the simplex. $\mu_{\ell_{\tilde{N},i}}$ and $\nu_{\ell_{\tilde{N},}}$ are two-dimensional vectors – the covariance on locations is diagonal. Note that in the exact posterior, $\tilde{N}$ has support on the nonnegative integers; in the variational distribution, we truncate at some large $N_{max}$.

These distributions were taken to match the constraints of the latent variables: fluxes are positive and right skewed, suggesting a log-normal; locations are between zero and $R$, suggesting a scaled logit-normal.

### Evaluating the variational distribution

We detail the computation of $q_\eta(z|x)$ for any given catalog $z = \{N, (\ell_i, f_{i,1}, ..., f_{i,B})_{i=1}^N\}$. This will be required for the wake-sleep optimization procedure (Section 2.3). By Equation 2.15, it suffices to evaluate $\tilde{q}_\eta(\tau^{-1}(z)|x)$.

Here, $\tau^{-1}(z)$ is a *set* of tile latent variables because the mapping from tile latent variables $\tilde{z}$ to catalogs $z$ is not injective, as we now explain.

Locations in the catalog $\{\ell_i\}_{i=1}^N$ determine the number of stars on tile $(s,t)$. The number of stars $\tilde{N}^{(s,t)}$ is simply the count of the locations that reside within that tile:

$$\tilde{N}^{(s,t)} = \sum_{i=1}^{N} \mathbf{1}\Big\{\ell_i \in [Rs, R(s+1)] \times [Rt, R(t+1)]\Big\}, \tag{2.19}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function, equal to one if true and zero if false.

Now, consider $\tilde{\ell}^{(s,t)}$ and $\tilde{f}^{(s,t)}$, the triangular array of locations and fluxes on tile $(s,t)$. For each $(s,t)$, the $\tilde{N}^{(s,t)}$-th row of the triangular array of fluxes and locations is determined by the locations and fluxes of stars imaged in tile $(s,t)$; they are determined by the catalog $z$. However, the other rows of the triangular arrays are not determined by the catalog $z$; they are free to take any value in their domain. Therefore, the mapping $\tau$ is not injective.

Thus, evaluating the probability of $\tau^{-1}(z)$ under $\tilde{q}_\eta$ requires marginalizing over the rows of the triangular arrays $\ell^{(s,t)}$ and $\tilde{f}^{(s,t)}$ that are not determined by $z$. However, because $\tilde{q}_\eta$ fully factorizes, the terms where $n \neq \tilde{N}^{(s,t)}$ do not enter the product after marginalization.

On each tile $(s, t)$,

$$\tilde{q}_\eta\big(\tilde{N}^{(s,t)}, \tilde{\ell}^{(s,t)}, \tilde{f}^{(s,t)}|x\big) = \tilde{q}_\eta(\tilde{N}^{(s,t)}|x) \prod_{n=1}^{N_{max}} \prod_{i=1}^{n} \tilde{q}_\eta\big(\tilde{\ell}_{n,i}^{(s,t)}|x\big)\tilde{q}_\eta\big(\tilde{f}_{n,i}^{(s,t)}|x\big) \tag{2.20}$$

$$= \tilde{q}_\eta(\tilde{N}^{(s,t)}|x) \prod_{i=1}^{\tilde{N}^{(s,t)}} \tilde{q}_\eta\big(\tilde{\ell}_{\tilde{N}^{(s,t)},i}^{(s,t)}|x\big)\tilde{q}_\eta\big(\tilde{f}_{\tilde{N}^{(s,t)},i}^{(s,t)}|x\big). \tag{2.21}$$

In words, given a catalog $z$, first convert $z$ to tile latent variables; then on each tile, it suffices to evaluate $\tilde{q}_\eta$ only at the rows of triangular arrays determined by the number of stars falling in each tile.

The last technical detail is computing a given row of a triangular array. Because catalogs are *sets*, each star in the tile must be matched with corresponding variational parameters. Let $(\tilde{\ell}_i, \tilde{f}_i)_{i=1}^n$ generically denote the tile latent variables in the $n$-th row of a triangular array, on some tile. We find the permutation of the tile latent variables that maximize its log-probability under $q$. Recalling the variational distribution in Equations 2.16, 2.17 and 2.18, we permute the stars by finding

$$\underset{\pi}{\text{argmax}} \Big\{ \prod_{i=1}^{n} \text{LogitNormal}(\tilde{\ell}_{\pi(i)}; \mu_{\ell_i}, \nu_{\ell_i}) \times \text{LogNormal}(\tilde{f}_{\pi(i)}; \mu_{f_i}, \sigma_{f_i}^2) \Big\} \tag{2.22}$$

where the argmax is taken over all permutations on $\{1, ..., n\}$. This is feasible because on each tile $N_{max} = 3$, so we only need to search through $3! = 6$ possibilities.

### Neural network architecture

In each tile, the distributional parameters in Equations 2.16, 2.17 and 2.18, are the output of a neural network. The input to the neural network is an $R \times R$ tile, padded with surrounding pixels. Padding enables the neural network to produce better predictions inside the tile. For example, a bright source in the vicinity of but outside the tile will affect the pixel values inside the tile. Padding the tiles allows the neural network access to this information. The appropriate amount of padding will depend on the PSF width in the analyzed image. To catalog the crowded starfield M2 (Section 2.5), we set $R = 2$ and padded the tile with a three-pixel-wide boundary. Thus, while the distribution on tile latent variables factorize over tiles, the neural network is able to use information from neighboring tiles in producing the distributional parameters.

In amortized inference, the variational parameters $\eta$ to be optimized are neural network weights. The architecture consists of several convolutional layers followed by several fully connected layers (Figure 2.3). This architecture has been successful on image classification challenges such as ImageNet (Russakovsky et al., 2015). The optimization of the architecture for this specific application is left for future work.

Note that the output dimension of the neural network is quadratic in $N_{max}$: the outputs are parameters for a triangular array consisting of $\frac{1}{2}(N_{max}^2 + N_{max})$ sources. Factorizing the

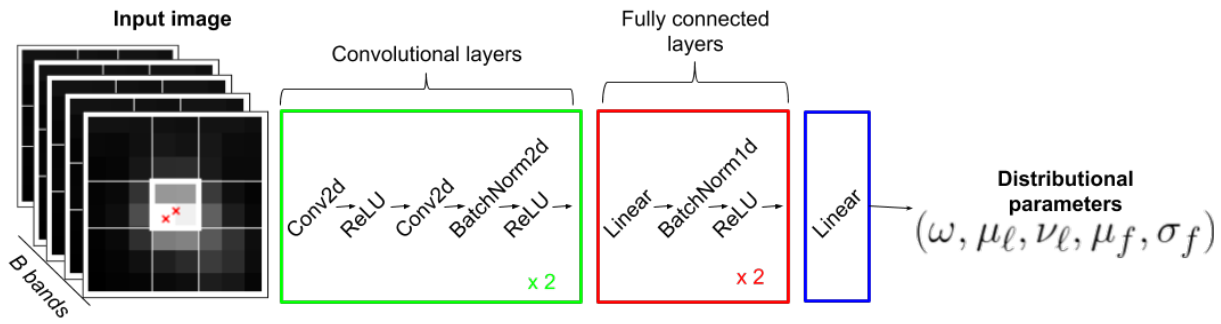



Figure 2.3: The neural network architecture. For cataloging M2, the input is an $8 \times 8$ padded tile, and the network returns distributional parameters for latent variables contained in the center $2 \times 2$ tile.

variational distribution spatially keeps the output dimension manageable. While the full image may contain many stars (on the region of M2 we consider, the number of stars is on the order of thousands), we set $N_{max} = 3$ on each $2 \times 2$ tile. Thus, the network is responsible for inferring only a few stars at once—a much easier task than inferring all $\sim 1000$ stars simultaneously.

We emphasize that while the variational distribution factorizes over $2 \times 2$ tiles, our method does not break the inference problem for the full image into isolated subproblems. The evaluation of the likelihood, e.g., when computing the ELBO in Equation 2.10, is always on the full image. Light from a star within a $2 \times 2$ tile spills over into neighboring tiles, so the likelihood should not and does not decouple across image tiles.

## 2.3 The wake-sleep algorithm

Procedures such as black-box variational inference (BBVI) (Ranganath et al., 2013) and automatic-differentiation variational inference (ADVI) (Kucukelbir et al., 2017) optimize the ELBO without the need for deriving analytic expressions for the expectation over $q_\eta$. These approaches all employ stochastic gradient descent (SGD); they sample latent variables from $q_\eta$ and produce an unbiased estimate for the gradient of the ELBO by taking advantage of modern automatic differentiation tools. ADVI is closely related to the reparameterization trick (Kingma and Welling, 2013; Rezende et al., 2014), which is often used to fit variational autoencoders and applies when the latent variables are continuous.

In our model, the number of stars $N$ is discrete. As an alternative, the REINFORCE estimator (Williams, 1992) produces an unbiased stochastic gradient for both continuous and discrete latent variables. However, REINFORCE gradients without additional modification often suffer from high variance in practice, resulting in slow convergence of stochastic optimization. We find this to be true in our empirical study. (Section 2.4).

The key difficulty in constructing stochastic gradients of the ELBO is that the integrating distribution depends on the optimization parameter $\eta$. The *wake-sleep* algorithm, originally

proposed by Hinton et al. (1995), replaces the ELBO objective with

$$\mathcal{L}_{sleep}(\eta) := -\mathbb{E}_{x\sim p(x)}\Big[\mathrm{KL}(p(z|x)\|q_\eta(z|x))\Big], \tag{2.23}$$

known as the *sleep objective*. Section 2.3.1 details a simple gradient estimator for Equation 2.23 that does not require reparameterization or REINFORCE.

There are two key differences between the sleep objective (Equation 2.23) and the ELBO (Equation 2.10). First, recall that maximizing the ELBO is equivalent to minimizing $\mathrm{KL}(q\|p)$; in the sleep objective, the KL arguments are transposed. Second, the outer expectation over $p(x)$ also gives different meaning to the sleep objective. The ELBO objective seeks $\eta$ to minimize the KL between $q_\eta(z|x)$ and $p(z|x)$ *for fixed, observed data* $x$, in this case the $H \times W$ image. In contrast, the sleep objective seeks to minimize the KL *on average over all possible data* $x$, as weighted by $p(x)$. The target is no longer an approximate posterior for the observed data, but rather an approximate posterior that is "good on average" over possible data under the model $p(x)$.

Therefore, it is imperative that the model $p(x)$ approximates the true underlying data-generating mechanism well. Thus, the wake-sleep algorithm also incorporates a "wake phase" to estimate model parameters. In our application, these model parameters include PSF parameters $\pi$ and background parameters $\beta$. Define $\phi := (\pi, \beta)$, and denote the dependence of the generative model on $\phi$ using subscripts, $p_\phi$.

To estimate model parameters, one would ideally optimize the marginal log-likelihood $\log p_\phi(x)$. However, since $\log p_\phi(x)$ is intractable, the wake-phase optimizes for $\phi$ using the ELBO (Equation 2.10) as a surrogate for the intractable log-likelihood. The ELBO is a lower bound of $\log p_\phi(x)$; it is equal to $\log p_\phi(x)$ when $q_\eta(z|x) = p_\phi(z|x)$.

The *wake-sleep* algorithm alternates between the two objectives:

$$\textbf{Sleep phase: } \eta_t = \operatorname*{argmax}_\eta -\mathbb{E}_{x\sim p_\phi(x)}\Big[\mathrm{KL}(p_{\phi_{t-1}}(z|x)\|q_\eta(z|x))\Big]; \tag{2.24}$$

$$\textbf{Wake phase: } \phi_t = \operatorname*{argmax}_\phi \mathbb{E}_{q_{\eta_t}(z|x)}\Big[\log p_\phi(x,z) - \log q_{\eta_t}(z|x)\Big], \tag{2.25}$$

for iterations $t = 1, ..., T$.

Stochastic gradients of the expectation in the wake-phase are simple to compute. Because the integrating distribution does not depend on the optimization parameter $\phi$ in the wake phase, unbiased stochastic gradients are simply computed as

$$\nabla_\phi \log p_\phi(x,z) \quad \text{for } z \sim q_\eta. \tag{2.26}$$

Section 2.3.1 shows that a similarly simple gradient estimator exists for the sleep objective.

The wake-sleep algorithm is closely related to variational EM (Jordan et al., 1999; Neal and Hinton, 2000; Beal and Ghahramani, 2002), which alternates between an *expectation* step (E-step) and a *maximization* step (M-step). Variational EM can be viewed as block coordinate ascent on the ELBO objective, with the E-step optimizing variational parameters

$\eta$ and the M-step optimizing the model parameters $\phi$. The wake phase of the wake-sleep algorithm is equivalent to the M-step of variational EM. As discussed above, the optimization of the ELBO with respect to $\eta$ is not conducive to using simple stochastic gradient estimators. Thus, the sleep phase replaces the ELBO objective of the E-step with the expected KL in Equation 2.23.

### 2.3.1 Decomposing the sleep objective

In this subsection, we decompose the sleep objective in Equation 2.23 for closer study. We take $\phi$ as fixed in this section and drop the explicit dependence of $p$ on $\phi$.

First, observe that optimizing the sleep objective does not require computing the intractable term $p(x)$:

$$\begin{aligned}
\operatorname*{argmax}_{\eta} \ \mathcal{L}_{sleep}(\eta) &= \operatorname*{argmin}_{\eta} \ \mathbb{E}_{x\sim p(x)}\Big[\mathrm{KL}(p(z|x)\|q_\eta(z|x)\Big] && (2.27)\\
&= \operatorname*{argmin}_{\eta} \ \mathbb{E}_{p(x)}\Big[\mathbb{E}_{p(z|x)}\Big(\log p(z|x) - \log q_\eta(z|x)\Big)\Big] && (2.28)\\
&= \operatorname*{argmin}_{\eta} \ \mathbb{E}_{p(x,z)}\Big[-\log q_\eta(z|x)\Big]. && (2.29)
\end{aligned}$$

Crucially, the integrating distribution, $p(x, z)$, does not depend on the optimization parameter $\eta$. Thus, unbiased stochastic gradients can be obtained simply as

$$g = -\nabla_\eta \log q_\eta(z|x) \quad \text{for } (x, z) \sim p(x, z). \tag{2.30}$$

In other words, at each iteration of SGD, the sleep phase simulates *complete* data $(x, z)$ from the generative model and evaluates the loss $-\log q_\eta(z|x)$. Here, "complete data" refers to the image along with its catalog. This loss encourages the neural network to map an image $x$ to a distribution $q_\eta(\cdot|x)$ that places large mass on the image's catalog $z$.

We decompose the loss $-\log q_\eta(z|x)$ further. Recall that $q_\eta$ fully factorizes over tile latent variables, and thus $-\log q_\eta(z|x)$ is a summation over all tile latent variables. To evaluate $-\log q_\eta(z|x)$ for some $(x, z) \sim p$, first convert $z$ to its tile parameterization $(\tilde{N}^{(s,t)}, \tilde{\ell}^{(s,t)}, \tilde{f}^{(s,t)})_{s=1,t=1}^{(S,T)}$, as detailed in Section 2.2.1. On each tile $(s, t)$, the variational distribution on the number of stars $\tilde{N}^{(s,t)}$ is categorical with probability vector $\omega^{(s,t)} \in \Delta^{N_{max}}$ (recall Section 2.2.1). The loss function for the number of stars becomes

$$-\log q_\eta(\tilde{N}^{(s,t)}|x) = -\sum_{n=0}^{\tilde{N}_{max}} 1\{\tilde{N}^{(s,t)} = n\} \log \omega_n^{(s,t)}. \tag{2.31}$$

The vector $\omega^{(s,t)}$ is the output of the neural network, and Equation 2.31 is the usual cross-entropy loss for a multi-class classification problem.

Next, recall that location coordinates are logit-normal and fluxes are log-normal in the variational distribution. Let $y$ generically denote either the logit-location or log-flux for a

star in the sampled catalog $z$; let $(\hat{\mu}, \hat{\sigma}^2)$ be the Gaussian mean and variance returned by the neural network. Then the loss for these latent variables is,

$$-\log q_\eta(y|x) = \frac{1}{2\hat{\sigma}^2}(y - \hat{\mu})^2 + \frac{1}{2}\log(2\pi\hat{\sigma}^2). \tag{2.32}$$

The first term encourages network predictions $\hat{\mu}$ to be close to the sampled latent variable $y$, while $\hat{\sigma}^2$ encodes the uncertainty of the network: the second term encourages small uncertainties, but is balanced by the scaling of the error $(y - \hat{\mu})^2$ in the first term.

The losses in Equations 2.31 and 2.32 show that the sleep objective results in a supervised learning problem on complete data sampled from our generative model: the objective function for the number of stars is the usual cross-entropy loss for classification, while the objective function for log-fluxes and logit-locations are $L_2$ losses in the mean parameters.

## 2.4 Empirical comparison of ELBO and sleep objectives

A simple example demonstrates that there exist shallow local optima in the ELBO where the fitted approximate posterior is far in KL divergence from the exact posterior. These local optima result in unreliable catalogs. The sleep objective, by taking advantage of complete data, appears to better avoid these local optima. For this example, the data were simulated with known PSF and background, and the wake phase is not needed. The simulated $20 \times 20$ single-band image $x_{test}$ is shown in Figure 2.4(d).

We compare three approaches to deblending. The first two approaches directly optimize the test ELBO,

$$\mathcal{L}_{elbo}(\eta; x_{test}) = \mathbb{E}_{q_\eta(z|x_{test})}\Big[\log p(x_{test}, z) - \log q_\eta(z|x_{test})\Big], \tag{2.33}$$

while the third approach optimizes the sleep objective (Equation 2.23). In each case, $q_\eta$ is the inference network from Section 2.2.1; the input to the network is a $10 \times 10$ tile with no padding.

Note that the sleep objective does not depend on $x_{test}$. Optimizing the sleep objective only requires sampling catalogs from the prior and simulating images conditional on each catalog. The prior on the number of stars was set to Poisson with mean $\mu = 4$.

Figure 2.4 (top row) charts the test ELBO (Equation 2.33) as the optimization proceeds in our three approaches. The first approach optimizes the ELBO with SGD and the REINFORCE gradient estimator. This optimization did not converge, likely due to the high variance of the REINFORCE estimator (Figure 2.4a). For a lower variance gradient estimator, the second approach employed the reparameterized gradient. To employ this gradient estimator, we analytically integrated the ELBO with respect to the number of stars $N$ to remove the discrete random variable. Using reparameterized gradients instead of REINFORCE gradients enabled the optimization to converge to stationary points (Figure 2.4b).

However, for two randomly initialized restarts, the optimization found local optima where the negative ELBO is higher than other restarts.

In contrast, optimizing the sleep objective consistently converged to a similar ELBO across all restarts and appeared to avoid shallow local optima (Figure 2.4c). Recall that sleep phase optimization does not directly optimize the test ELBO. However, the test ELBO increases nonetheless, because the variational posterior better approximates the exact posterior as the optimization proceeds.

Shallow local optima in the ELBO result in unreliable catalogs. The bottom row of Figure 2.4 displays the estimated locations, defined as the mode of the fitted variational distribution. Figure 2.4(e) shows these locations after converging to a shallow local optimum. Here, the upper left tile was correctly estimated to have two stars, though both estimated stars were placed at the same location. For correct detections, one of the locations should be placed on the second star. However, in order to move one of the estimated locations to the second star, the optimization path must traverse a region where the log-likelihood is lower than the current configuration (Figure 2.5). The displayed configuration is a local optimum where the gradient with respect to its locations is approximately zero. In contrast, the sleep-phase variational distribution consistently placed its mode around the four true stars. The sleep objective is quadratic in the logit-location estimate $\mu_\ell$ (Equation 2.32), and the gradient does not vanish in the sleep objective. An example of correct detection after sleep-phase optimization is shown in Figure 2.4(f).

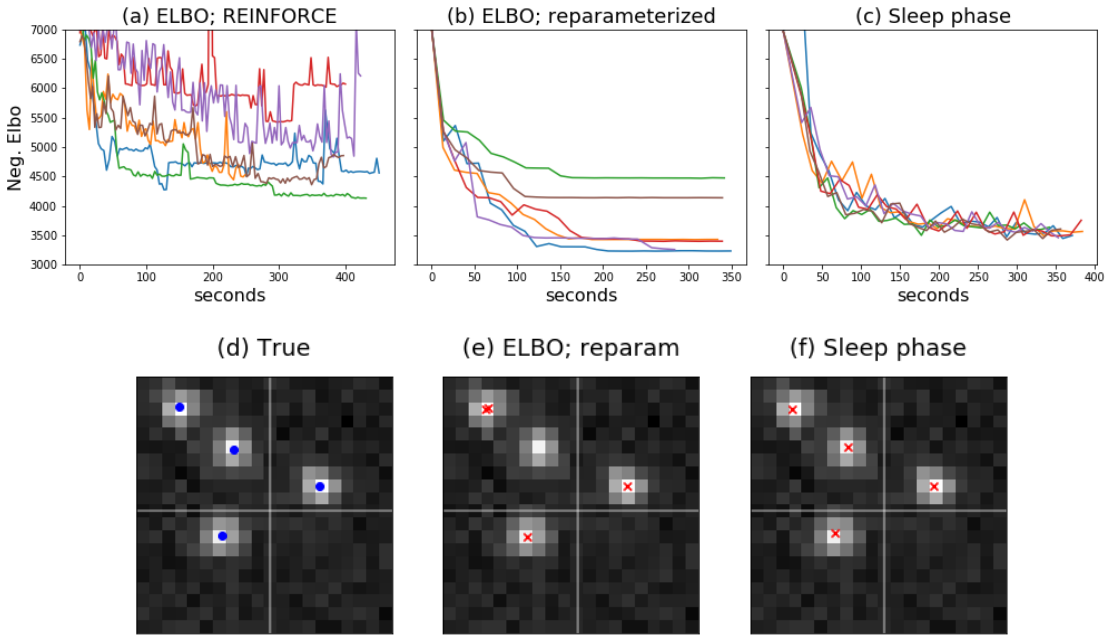



Figure 2.4: (Top row) The ELBO as the optimization progresses for six random restarts. (Bottom row) In red, modal locations from ELBO-optimized and sleep-optimized variational posteriors, for one of the six restarts. In blue, the true locations.

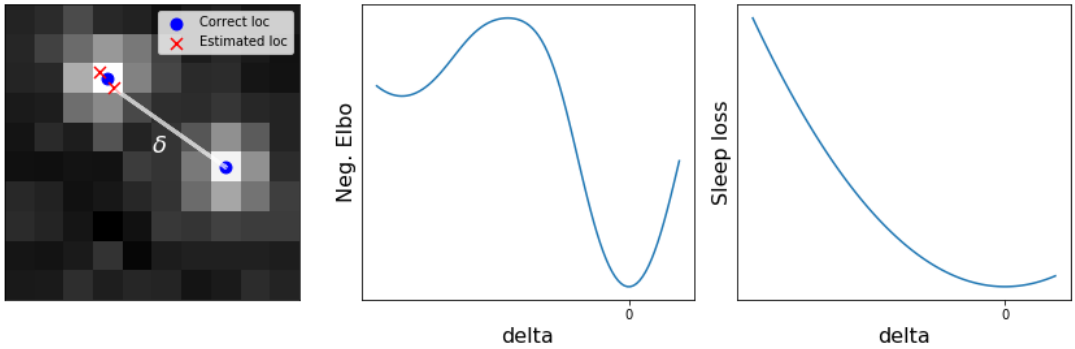



Figure 2.5: An illustration of local optima in the ELBO objective. To move an estimated location to a correct location, the optimization path must traverse a region where the negative ELBO is larger than the current configuration. In contrast, the sleep objective is quadratic in the estimated location, and the gradient does not vanish.

## 2.5 Results on the M2 globular cluster

We validated StarNet using an SDSS image of the Messier 2 (M2) globular cluster, a crowded starfield found in field 136 of camera column 2 in run 2583. M2 was also imaged in the ACS Globular Cluster Survey (Sarajedini et al., 2007) using the Hubble Space telescope (HST), which has greater resolution than the Sloan telescope. The resolution of the HST wide-field channel is 0.05 arcseconds per pixel versus 0.4 arcseconds per pixel in SDSS (ESAHubble; SDSS). The catalog from this Hubble survey (henceforth the "HST catalog") serves as ground truth for validating our results.

We focus on the $100 \times 100$ pixel subimage of M2 that Portillo et al. (2017) and Feder et al. (2020) analyzed with PCAT. This subimage is located approximately two arcseconds from the heavily saturated core of the cluster; even in this subimage, the HST catalog contains over 1000 stars with F606W-band magnitudes less than 22. We include two bands in our model, the SDSS $r$-band and $i$-band. The SDSS $r$-band and the Hubble F606W band are centered at roughly the same wavelength, but the wavelength range of the Hubble F606W band is slightly broader.

### 2.5.1 Runtime

We factorized our variational distribution into $2 \times 2$ pixel tiles. The neural network inputs were $8 \times 8$ pixel padded tiles: $2 \times 2$ tiles along with three surrounding pixels of padding (see Figure 2.3). The SDSS estimates for the PSF and background were used in the first sleep phase. This phase ran for 200 epochs; at each epoch, 200 images were sampled from the generative model. Optimization was done with Adam (Kingma and Ba, 2014). On a single NVIDIA GeForce RTX 2080 Ti GPU, the initial sleep phase took 15.2 minutes.

Two additional wake-sleep cycles followed the first sleep phase, after which the model parameters and inferred catalog appeared stable. The subsequent sleep phases were shorter

(10 epochs with 200 images each), and the wake phase employed SGD, with the gradient estimator in Equation 2.26. In total, the two further wake-sleep cycles took three minutes.

After fitting the model and variational posterior, calculating the approximate posterior (that is, producing the distributional parameters of the variational approximation) for the $100 \times 100$ pixel image of M2 took 30 milliseconds. By comparison, the reported runtime of PCAT, which uses MCMC, is 30 minutes on the same $100 \times 100$ pixel image (Feder et al., 2020). After the initial fit, StarNet provided nearly a $10^5$-fold speed increase.

The speed at inference time (which excludes training time) gives StarNet the scaling characteristics necessary for processing large astronomical surveys. A single SDSS image is $1489 \times 2048$ pixels. Based on the reported 30-minute runtime of PCAT for a $100 \times 100$ pixel subimage, we project that the runtime to process the full image would be $30 \text{ min} \times 14 \times 20 = 8400$ minutes, or almost six days. The SDSS survey consists of nearly one million images. Scaling PCAT to the entire SDSS survey would be infeasible. The upcoming LSST survey will be 300 times larger than SDSS.

In contrast, if we assume the PSF and background are homogeneous across the full SDSS image (which is also assumed in PCAT), we can fit StarNet using wake-sleep on a small $100 \times 100$ pixel subimage (while simultaneously obtaining estimates of the PSF and background), a one time computational cost of 18.2 minutes. Producing a catalog with the full $1489 \times 2048$ pixel image requires $30 \text{ msec} \times 14 \times 20 = 8.4$ seconds. In practice, inference can be made even faster by batching the image tiles to run in parallel on a GPU.

### 2.5.2 Inference

The cataloging accuracy of StarNet is compared with PCAT and DAOPHOT (Stetson, 1987). DAOPHOT is an algorithmic routine for detecting stars in crowded starfields which does not use a generative model. This software convolves the observed image with a Gaussian kernel and scans for peaks above a given threshold. The DAOPHOT catalog of M2 was reported in An et al. (2008).

To evaluate the three methods, the HST catalog was used as ground truth. We filtered the HST catalog to stars with magnitudes smaller than 22.5 in the Hubble F606W band, because stars with lower apparent brightness cannot be detected in SDSS images.

Estimated catalogs are evaluated on three metrics: the true positive rate (TPR), the positive predicted value (PPV), and the F1 score. The TPR is the proportion of stars in the HST catalog matched with a star in the estimated catalog; the PPV is the proportion of stars in the estimated catalog matched with a star in the HST catalog. The F1 score summarizes the two metrics as the harmonic mean of the PPV and the TPR.

Like Portillo et al. (2017) and Feder et al. (2020), a "match" between an estimated star and an HST star is defined as follows: (1) the estimated location and the HST location are within 0.5 SDSS pixels, and (2) the estimated SDSS $r$-band flux and the HST F606W band flux are within half a magnitude.

In probabilistic cataloging (PCAT and StarNet), the posterior defines a distribution over catalogs. For StarNet, the TPR, PPV, and F1 score were computed for the catalog

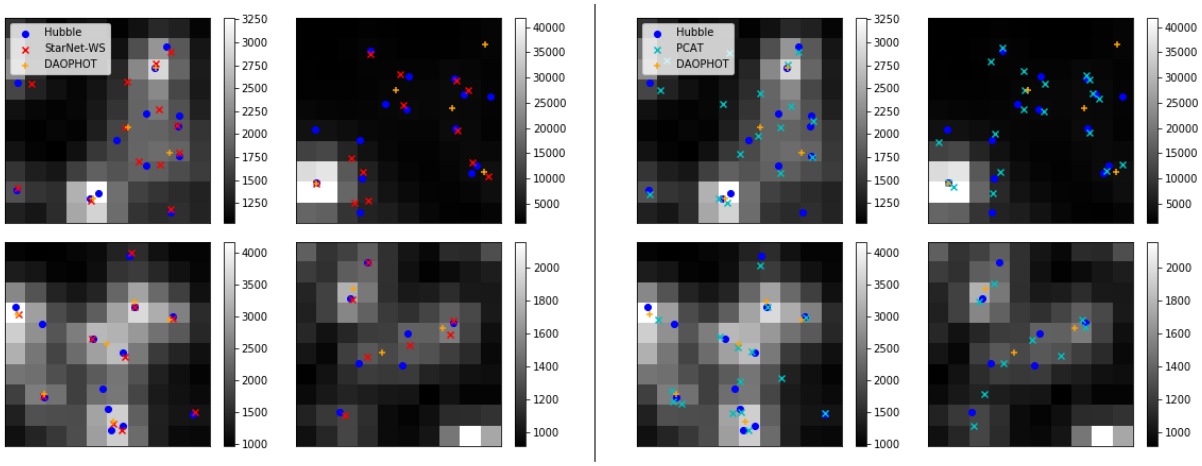



Figure 2.6: Estimated catalogs on four 10×10 subimages from M2. Blue dots are stars from the HST catalog used as ground truth. Starnet-WS, PCAT, and DAOPHOT estimated stars are shown as red, cyan, and orange crosses, respectively.

corresponding to the mode of the variational distribution (henceforth, the StarNet catalog). For PCAT, 300 catalogs were sampled using MCMC; the metrics were computed for each sampled catalog and averaged.

Fitting StarNet using wake-sleep (StarNet-WS) resulted in a catalog that outperforms DAOPHOT and PCAT in F1 score (Table 2.1). Figure 2.6 shows the StarNet-WS catalog alongside PCAT, DAOPHOT, and HST catalogs. Fitting StarNet without the wake phase and using only the default SDSS background and PSF (StarNet-S) produced a catalog with a TPR similar to that of StarNet-WS but with a smaller PPV. PCAT estimated the most stars of all methods; it therefore had a large TPR but a small PPV. On the other hand, DAOPHOT estimated less than half the number of stars when compared to the other methods. It therefore had a large PPV but a small TPR. The StarNet-WS catalog had about the same PPV as DAOPHOT while having nearly the same TPR as PCAT.

Table 2.1 also prints the number of stars inferred by each method. For probabilistic methods (StarNet and PCAT), we display the mean number of stars under the approximate posterior, along with the 5-th and 95-th quantiles. There are 1114 stars in the HST catalog. Neither PCAT nor StarNet captured the true number of stars in their 90% credible interval, though StarNet-WS came the closest. The StarNet credible intervals were three times wider than the PCAT intervals. The small PCAT credible intervals may be indicative of an MCMC sampler that failed to mix well.

The improvement of StarNet-WS over PCAT in PPV was most pronounced for dim stars (Figure 2.7). The TPR for StarNet-WS was uniformly better than DAOPHOT across almost all magnitudes. Of all methods, StarNet-WS best approximated the HST flux distribution (Figure 2.8).

The difference between the flux distributions (Figure 2.8) produced by StarNet-WS and PCAT is partly due to the fact that StarNet-WS estimates the background with maximum

Table 2.1: Performance metrics on M2. For probabilistic methods (StarNet and PCAT) the "#stars" column refers to the mean number of stars under the (approximate) posterior, while the right-most column displays the 5-th and 95-th percentiles under the posterior.

| Method | TPR | PPV | F1 score | #stars | (q-5%, q-95%) |
|---|---|---|---|---|---|
| DAOPHOT | 0.20 | 0.63 | 0.30 | 295 | – |
| PCAT | 0.56 | 0.40 | 0.47 | 1672 | (1664, 1680) |
| Sleep-only | 0.51 | 0.47 | 0.49 | 1292 | (1260, 1324) |
| Wake-sleep | 0.51 | 0.60 | 0.55 | 1014 | (987, 1041) |

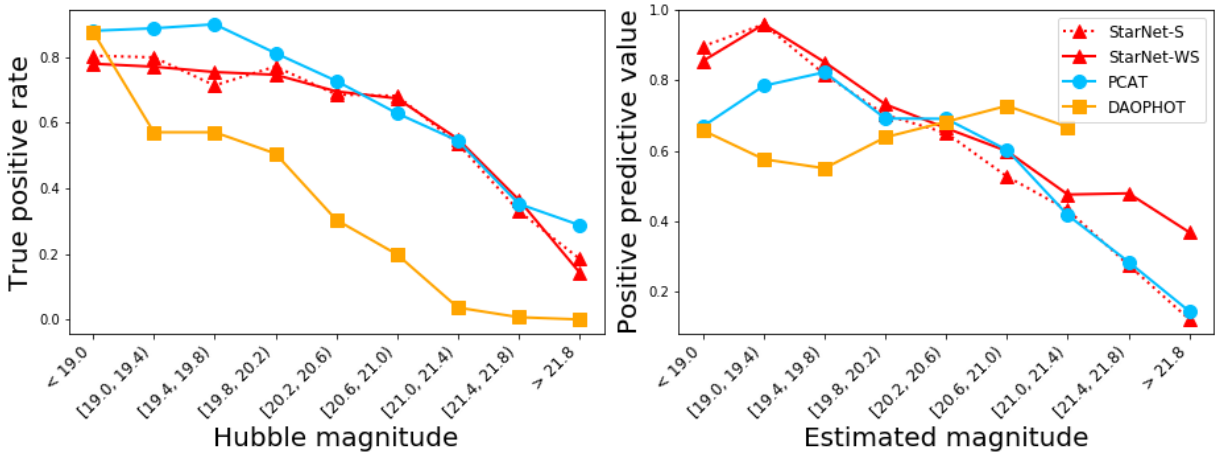



Figure 2.7: True positive rate (left) and positive predicted value (right) of various cataloging procedures on M2, plotted against $r$-band magnitude. Smaller magnitudes correspond to brighter stars.

likelihood, while the background in PCAT is fixed. As discussed below, the SDSS estimate of the background (which PCAT uses) is too dim, and PCAT compensates for the model mis-match by estimating more dim stars. The StarNet-WS estimate increases the intensity of the background, and hence does not over-estimate dim stars.

To examine the uncertainty calibration of StarNet-WS, we evaluated the approximate posterior conditional on the true number of stars in the HST catalog. Then, each star in the StarNet-WS catalog was matched with exactly one Hubble star by finding the permutation of Hubble stars that had the largest log-likelihood under our variational distribution $q_\eta$. For each star, we computed the z-score $(y - \hat{\mu})/\hat{\sigma}$, where $y$ is the HST log-flux or logit-location; $\hat{\mu}$ and $\hat{\sigma}$ are the mean and the standard deviation, respectively, of the Gaussian variational distribution for the log-flux or logit-location. The empirical z-score distributions are close to a standard Gaussian with some discrepancy in the tails and some evidence of skewness, suggesting the uncertainties are not too mis-estimated (Figure 2.9).

Finally, we evaluate the quality of the wake-phase estimates for the PSF and background.

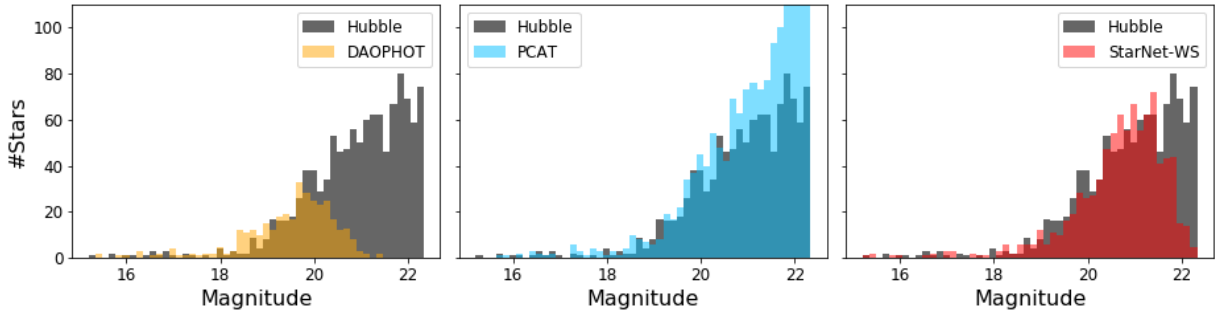



Figure 2.8: Flux distributions for the $r$-band observations of M2. The flux distribution of the HST catalog in grey. Estimated distributions by DAOPHOT, PCAT, and StarNet-WS catalogs overlaid. For PCAT, the flux distribution is from a single catalog sample.

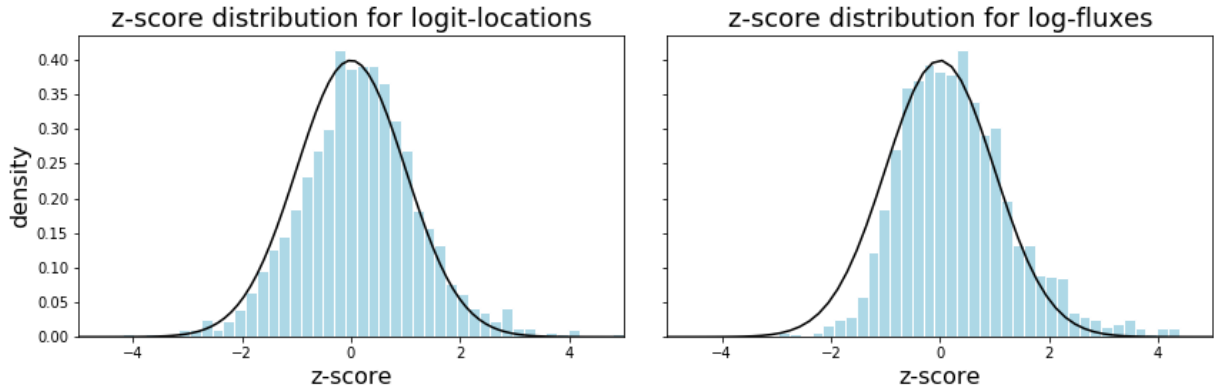



Figure 2.9: The empirical distribution of z-scores for logit-locations (left) and log-fluxes (right) under the StarNet-WS variational posterior.

Let $z_h$ denote the HST catalog, and given some model parameters $\phi$, let

$$\mathcal{L}^{Hubble}(\phi) := -\log\ p_\phi(x^{(r)}|z_h) \tag{2.34}$$

be the negative log-likelihood of the SDSS $r$-band image conditional on the HST catalog (recall that the Hubble absorption range most closely resembles the SDSS $r$-band). The log-likelihood of model parameters estimated by the wake phase is about two times larger than the log-likelihood of parameters estimated by SDSS (Table 2.2). The largest improvement in log-likelihood came from the wake-estimated background, which was brighter than the SDSS background, suggesting that the SDSS background was too dim for this starfield.

Section 2.7 presents the results of StarNet, PCAT, and DAOPHOT on a neighboring $100 \times 100$ pixel subimage of M2. The results remain qualitatively similar in that StarNet has the best F1 score of all methods. Importantly, StarNet was applied to the new region without further wake-sleep optimization; StarNet produced a catalog on this new region in $\approx 30$ milliseconds. On the other hand, PCAT required a new 30 minute run of MCMC.

Table 2.2: The negative log-likelihood Equation 2.34 for various model estimates. PHOTO provides estimates of background and PSF for every SDSS data release. We compare PHOTO estimates with StarNet estimates obtained after two cycles of wake-sleep.

| Model estimate | background<br>PSF | PHOTO<br>PHOTO | PHOTO<br>StarNet | StarNet<br>PHOTO | StarNet<br>StarNet |
|---|---|---|---|---|---|
| | Neg.Loglik | 8.671e+05 | 8.665e+05 | 3.651e+05 | 3.395e+05 |

## 2.6 Conclusion

StarNet employs variational inference and outperforms both an MCMC-based probabilistic cataloger and a non-model-based approach in terms of accuracy and runtime. Under the framework of probabilistic modeling, StarNet produces catalogs in which uncertainties are captured by a posterior over the set of all catalogs. Importantly, unlike MCMC, StarNet also has the capacity to scale probabilistic cataloging to process large astronomical surveys.

The quality of StarNet detections is the result of optimizing the forward KL, a different objective than the one traditionally used in variational inference. Optimizing the forward KL allows the variational posterior to be fit on large amounts of *complete* data – the image along with its latent catalog – generated from the statistical model. While labeled data from simulators has been used in other astronomy applications to train deep neural networks (see for example Lanusse et al. (2017) and Huang et al. (2020)), StarNet is the first training procedure to employ simulated data in a statistical framework: the neural network specifies an approximate Bayesian posterior.

This variational approach, unlike previous MCMC approaches, enables StarNet to estimate model parameters such as the PSF and sky background. While the current work focuses on PSF models, our methodology can be extended to more general sources such as galaxies. Unsurprisingly, the current performance of StarNet is sub-optimal for cataloging regions of the sky that contain both stars and galaxies, due to model misfit (Section 2.7).

One promising direction is to use a deep generative model for galaxies (Regier et al., 2015; Reiman and Göhre, 2019; Lanusse et al., 2020; Arcelin et al., 2021). Here, a neural network encodes a conditional likelihood of galaxy images given a low-dimensional galaxy representation. Using a neural network to encode a likelihood extends the flexibility of galaxy models beyond the simple models used here.

The statistical framework in this research lays the foundation for building flexible models to incorporate the cataloging of all celestial objects. Future astronomical surveys will only expand in terms of the volume of data they are able to amass. As telescopes peer deeper into space, fields will reveal more sources and images will become more crowded. The uncertainties in crowded fields necessitate a probabilistic approach. Our method holds the promise of providing a scalable inference tool that can meet the challenges of future surveys.

Table 2.3: Performance metrics on the M2 test subregion. StarNet-init is the network fit on the original M2 subregion (the same network as StarNet-WS in the main text). StarNet-refit ran two further wake-sleep cycles on the M2 test subregion.

| Method | TPR | PPV | F1 score | #stars | (q-5%, q-95%) |
|---|---|---|---|---|---|
| DAOPHOT | 0.13 | 0.53 | 0.21 | 338 | – |
| PCAT | 0.44 | 0.37 | 0.41 | 1793 | (1799, 1805) |
| StarNet-init | 0.47 | 0.47 | 0.47 | 1466 | (1431, 1499) |
| StarNet-refit | 0.47 | 0.50 | 0.48 | 1396 | (1362, 1432) |

## 2.7 Supplemental results

### 2.7.1 Results on a test M2 image

We evaluate StarNet on another subregion of M2. The initial $100 \times 100$ subregion of M2 considered in our main paper was located at pixel coordinates (630, 310) in SDSS run 2583, field 136, camera column 6. After fitting StarNet on this initial region $x_0$, we evaluate StarNet on a neighboring region $x_{test}$. See Figure 2.10 for locations of the considered subregions.

The subregion $x_{test}$ has approximately 25% more stars than $x_0$ (1413 Hubble detections in $x_{test}$ versus 1114 detections in $x_0$). Due the increased density, all methods suffered an $\approx 10$ percentage point decrease in F1 score on $x_{test}$ compared to the F1 score on $x_0$ (compare Table 2.1 and 2.3).

In Table 2.3 and Figure 2.11, "StarNet-init" refers to the wake-sleep trained network on $x_0$. Using StarNet-init and its fitted background and PSF as an initialization, we ran two further cycles of wake-sleep on $x_{test}$ (StarNet-refit). StarNet-refit improved the PPV over Starnet-init by two percentage points. The TPR appeared to be nearly identical across all magnitudes (Figure 2.11).

These results suggest that StarNet-init extrapolates well to neighboring regions, and re-running wake-sleep is not necessary. Evaluating StarNet-init on $x_{test}$ took 30 milliseconds. On the other hand, re-running PCAT takes another 30 minutes. Even if StarNet did require refitting, the subsequent wake-sleep cycles takes only an additional three minutes. The amortization enables StarNet inference to have much better scaling characteristics than PCAT.

### 2.7.2 Sensitivity to prior parameters

We examine the sensitivity of StarNet to prior parameters $\mu$ and $\alpha$ on the image M2. Recall $\mu$ is the prior mean number of stars per pixel Equation 2.1; $\alpha$ is the power law slope on the $r$-band fluxes Equation 2.3. In the results of Section 2.5, $\mu = 0.15$ and $\alpha = 0.5$.

The model appears reasonably robust. As expected, as $\mu$ increases the TPR increases while the PPV decreases – the prior encourages more detections (Figure 2.12).

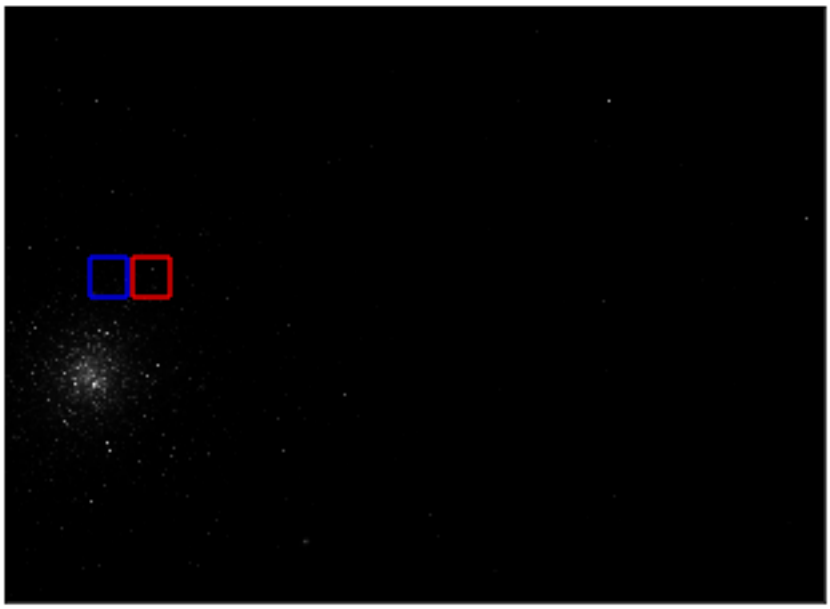


Figure 2.10: The SDSS field containing M2. In red, the subregion $x_0$ considered in the main text. The subregion $x_{test}$ in blue.

As $\alpha$ increases, StarNet estimates more dim sources: the prior distribution on fluxes places more mass near $f_{min}$ (Figure 2.13).

While the TPR increases across $\alpha$ values, the PPV suffers at $\alpha = 0.75$ (Figure 2.14). At $\alpha = 0.75$, we see that the TPR improves at dimmer sources at the expense of brighter sources, suggesting that the brigher sources become over-split. The stronger prior on dimmer stars also manifests in the flux distribution of the resulting StarNet catalog (Figure 2.15).

### 2.7.3 Results on Stripe-82

Most regions of the sky are much less densely populated than M2. We test StarNet on run 94, camcol 1, field 12 of SDSS, an image with light source density more typical of SDSS images. After 10 minutes of sleep training, StarNet produced a catalog on the full $1489 \times 2048$ image in $\approx 2$ seconds. For comparison, the projected runtime of PCAT on an image of this size is 6 days.

Since this region of the sky is more sparse, we tile the image into $50 \times 50$ tiles; $N_{max}$ on tiles is three. Because this region of the sky also contains galaxies, only the sleep phase was employed to fit StarNet; the wake phase would optimize the PSF to explain both stars and galaxies.

This image is contained in Stripe 82, a region of the sky repeatedly imaged by SDSS. Averaging images from different runs boosts the signal to noise ratio, resulting in a "co-

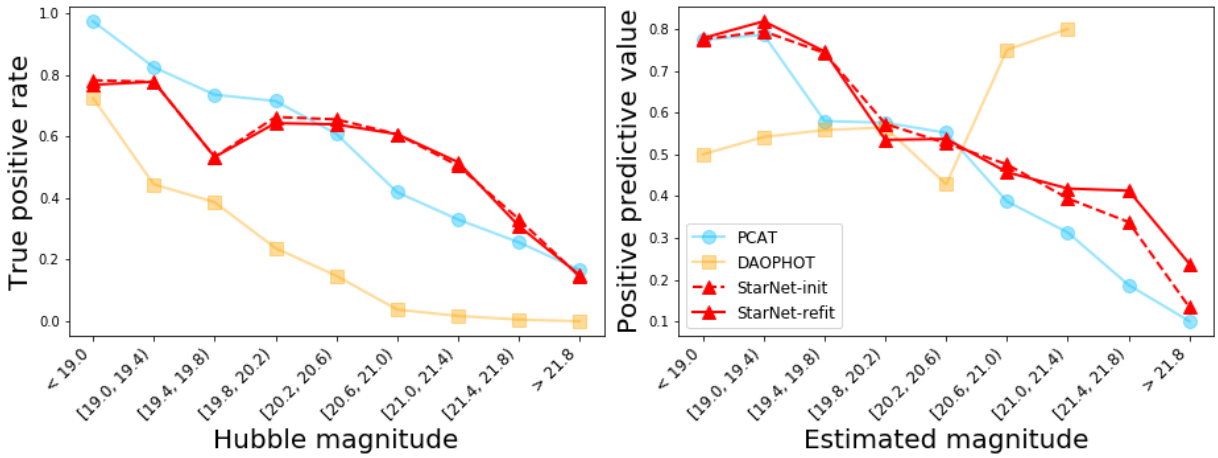
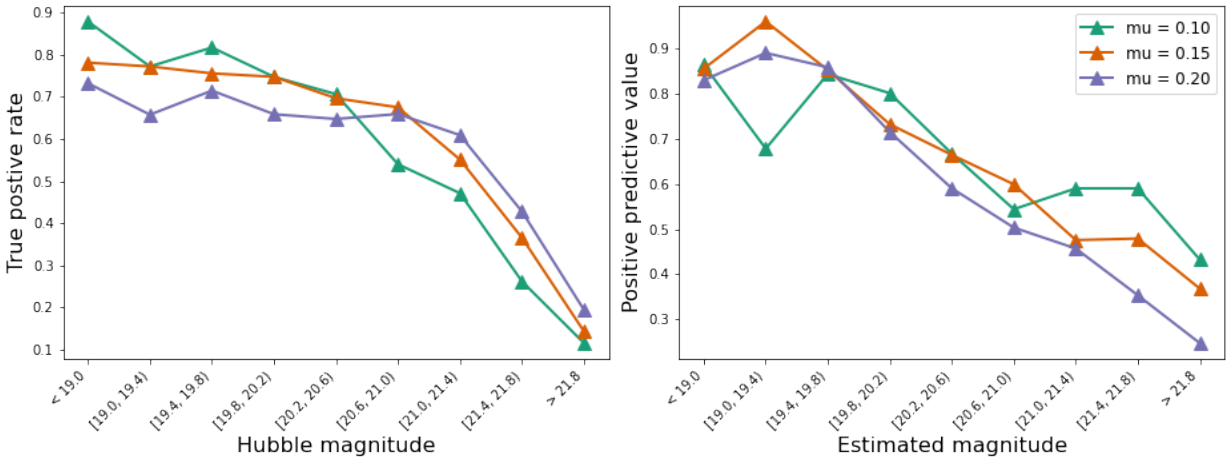



Figure 2.11: True positive rate (left) and positive predicted value (right) of various methods on the M2 test subregion. StarNet-init is the network fit on the original M2 subregion (the same network as StarNet-WS in the main text). StarNet-refit ran two further wake-sleep cycles on the M2 test subregion.

added" image. We compare the performance of StarNet and PHOTO on the *non* co-added image. The PHOTO catalog of the co-added image was used as ground truth.

The TPR of StarNet is comparable with the TPR of the PHOTO catalog on the original (non co-added) image (Figure 2.16). StarNet accrues false detections, namely galaxies, and thus we cannot compare the PPV. On some tiles, missed detections occur when a large galaxy within a tile causes all $N_{max}$ detections to be placed around the galaxy – the remaining stars in the image go undetected. Incorporating a galaxy model would boost our performance.




| mu | TPR | PPV |
|---|---|---|
| 0.10 | 0.46 | 0.66 |
| 0.15 | 0.51 | 0.60 |
| 0.20 | 0.51 | 0.49 |

Figure 2.12: Sensitivity of performance metrics to Poisson mean parameter on number of stars ($\mu$ in Equation 2.1).

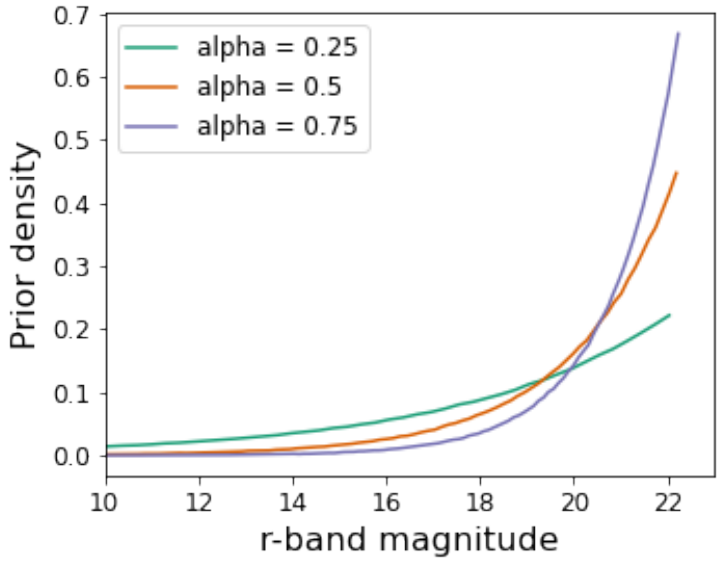
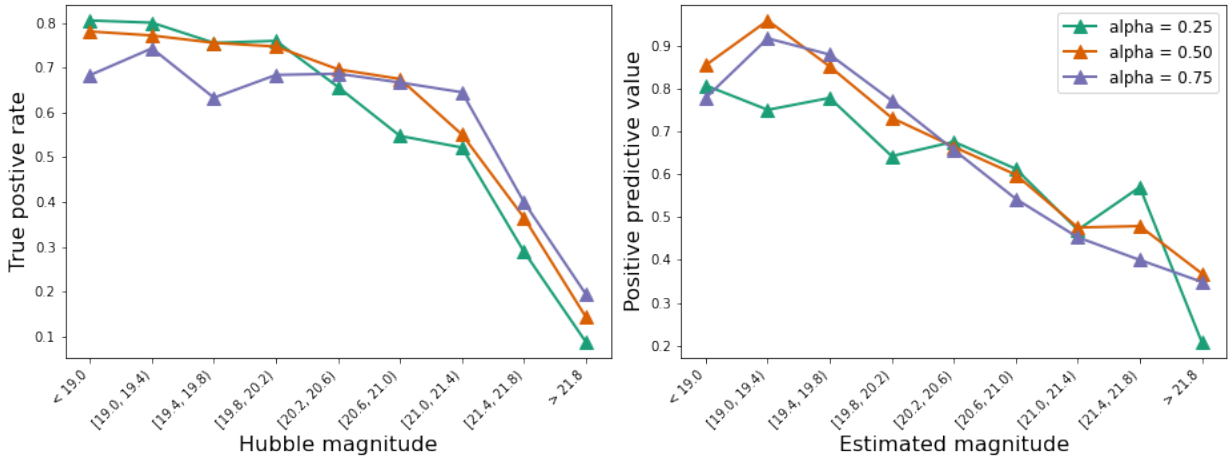



Figure 2.13: The prior density of $r$-band magnitude for various power law slopes $\alpha$.




| alpha | TPR | PPV |
|---|---|---|
| 0.25 | 0.46 | 0.60 |
| 0.50 | 0.51 | 0.60 |
| 0.75 | 0.52 | 0.54 |

Figure 2.14: Sensitivity of performance metrics to flux prior parameter $\alpha$.

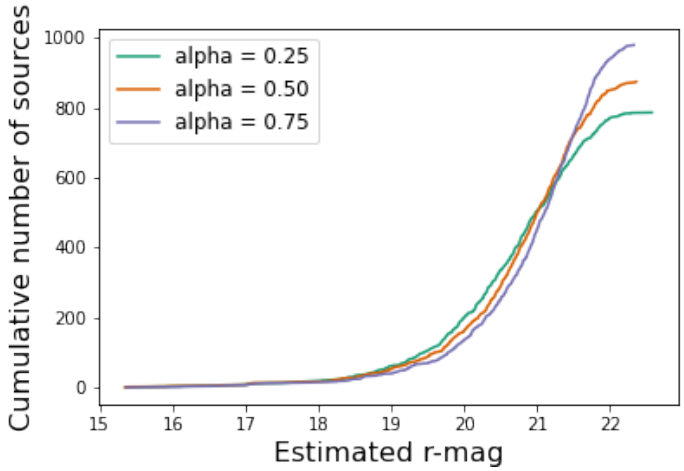



Figure 2.15: Sensitivity of estimated flux distribution to flux prior parameter $\alpha$.

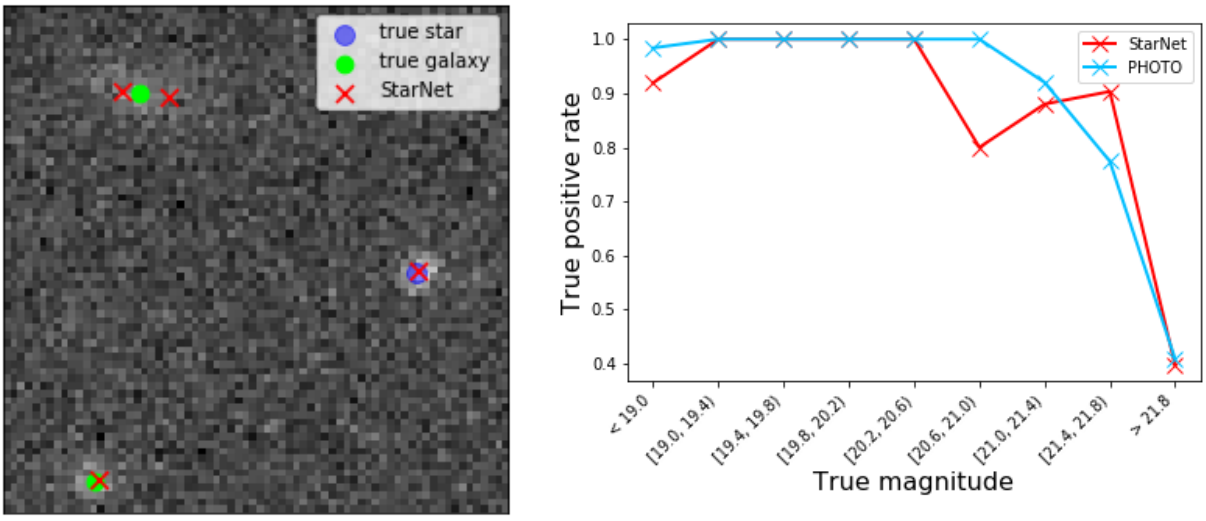



Figure 2.16: (Left) Detections on a sparse field. In this example, StarNet correctly identifies the star (blue). Without a galaxy model, StarNet also classifies galaxies (green) as one or multiple stars. (Right) The true positive rate of StarNet and PHOTO as a function of true magnitude.

# Chapter 3

# Local sensitivity in Bayesian nonparametrics

Two central questions in many probabilistic clustering problems is how many distinct clusters are present in a particular dataset, and which observations cluster together. Bayesian nonparametrics (BNP) addresses this question by placing a generative process on cluster assignment, making the number of distinct clusters present amenable to Bayesian inference. However, like all Bayesian approaches, BNP requires the specification of a prior, and this prior may favor a greater or fewer number of distinct clusters. In practice, it is important to establish that the prior is not too informative, particularly when—as is often the case in BNP—the particular form of the prior is chosen for mathematical convenience rather than because of a considered subjective belief.

We derive local sensitivity measures for assessing the impact of the prior on posterior inferences drawn using a variational Bayes (VB) approximation. In VB, the approximate posterior is characterized by a real-valued vector which is the solution to a numerical optimization problem. The optimization objective is defined as the Kullback-Leibler (KL) divergence between the approximating variational posterior and the true posterior. Thus, the variational posterior depends on prior parameters through the KL optimization. Local sensitivity measures approximate this dependence, which may be nonlinear, with a local Taylor series approximation (Gustafson, 1996; Giordano et al., 2018).

Using a stick-breaking representation of a Dirichlet process (Sethuraman, 1994), we consider perturbations both to the scalar concentration parameter and to the functional form of the stick-breaking distribution. To evaluate sensitivity to functional perturbations, we follow Gustafson (1996) and embed the stick-breaking density in the $L_p$ vector space of integrable functions and parameterize a path between the original prior and the perturbed prior.

We apply our methods to several real-world datasets, estimating the sensitivity of key posterior quantities to the BNP prior specification. Notably, we go beyond previous work on local Bayesian sensitivity (e.g. Basu et al. (1996)) which treated sensitivity as a measure of robustness *per se*. Rather, in the design and evaluation of our local sensitivity measures we pay special attention to our ability to accurately extrapolate posterior inferences to

different priors. We show the accuracy of our local approximation both for parametric and nonparametric perturbations by comparing against the much more expensive process of refitting the variational posterior. The speed of the local approximation allows rapid exploration of a wide range of possible perturbations.

Even with the speed of our approximation, testing sensitivity for *all* possible prior perturbations is impossible. However, we also show that the local sensitivity takes the form of an inner product, in an appropriate Hilbert space, between an *influence function* and a prior perturbation; we demonstrate how to use the influence function to guide our search for prior perturbations that result in high sensitivity. In particular, we can use the influence function to find maximally influential alternative priors.

Section 3.1 details the stick-breaking construction of Dirichlet process priors. Section 3.2 outlines our truncated variational approximation. Section 3.3 and Section 3.4 presents our local approximation for parametric and functional perturbations, respectively. Section 3.5 discusses how computing the sensitivity is done in practice.

In our results (Section 3.6), we first demonstrate our local sensitivity methodology on a toy example, a Gaussian mixture model of Fisher's iris data set. Then, we apply local sensitivity to real data analysis problems, which include a regression model of time-course gene expression data, and a topic model for studying population structure in a genetic database. We find that posterior quantities can be sensitive or non-sensitive, depending on both the application and the quantity of interest. In most cases, the local approximation well-approximates the results found under refitting. We empirically observe that computing the local sensitivity can be an magnitude faster than refitting. We also discuss some limitations of local sensitivity and present scenarios where it fails to be a good approximation to refitting in Section 3.7. Section 3.8 concludes.

## 3.1 Stick-breaking Dirichlet processes

A discrete Bayesian nonparametric (BNP) generative model draws data points $x_n$ from one of an infinite number of components indexed by $k = 1, 2, \ldots \infty$. Each component is characterized by a vector $\beta_k \in \Omega_\beta \subseteq \mathbb{R}^{D_\beta}$, with $\mathcal{P}(x_n|\beta_k)$ denoting the distribution of data arising from component $k$. We model the $\beta_k$ as arising IID from a known prior, or *base distribution*, denoted $\mathcal{P}_{\text{base}}(\beta_k)$, and write $\beta = (\beta_1, \beta_2, \ldots)$.

Assignment of data point $n$ to a mixture component is represented by an (infinite dimensional) vector $z_n = (z_{n1}, z_{n2}, \ldots)$ whose elements $z_{nk} = 1$ for exactly one $k$ and 0 otherwise. With $z_n$ defined in this way, we can write

$$\mathcal{P}(x_n|z_n, \beta) = \prod_{k=1}^{\infty} \mathcal{P}(x_n|\beta_k)^{z_{nk}}.$$

The prior probabilities of assignments $z_n$ are generated according to the following "stick-breaking" process. Fix a density $\mathcal{P}_{\text{stick}}(\cdot)$, with respect to the Lebesgue measure, over stick-breaking proportions $\nu_k \in (0, 1)$ and draw $\nu_k \overset{iid}{\sim} \mathcal{P}_{\text{stick}}(\nu_k)$ for $k = 1, 2, \ldots \infty$. Given these

stick lengths, construct probabilities using the following formula:

$$\pi_k := \nu_k \prod_{k'<k} (1 - \nu_{k'}), \tag{3.1}$$

where the empty product is taken to be equal to 1. By construction, $\sum_{k=1}^{\infty} \pi_k = 1$. Given the probability vector $\pi := (\pi_1, \pi_2, \ldots)$, the $z_n$ are drawn according to

$$\mathcal{P}(z_n|\pi) = \prod_{k=1}^{\infty} \pi_k^{z_{nk}}.$$

Since $\pi$ is a deterministic function of the stick-breaking proportions $\nu := (\nu_1, \nu_2, \ldots)$, we can also write $\mathcal{P}(z_n|\nu)$ with no ambiguity.

The stick-breaking distribution $\mathcal{P}_{\text{stick}}$ can be thought of as inducing a a distribution on the vector of probabilities $\pi$. Different stick-breaking distributions will different favor assignment probabilities, each with different implied degrees of concentration. A particularly common choice for $\mathcal{P}_{\text{stick}}(\nu_k)$ is the $\text{Beta}(\nu_k|1, \alpha)$ distribution,

$$\text{Beta}(\nu_k|1, \alpha) = \frac{\Gamma(1 + \alpha)(1 - \nu_k)^{\alpha-1}}{\Gamma(\alpha)}.$$

When $\mathcal{P}_{\text{stick}}$ is $\text{Beta}(\nu_k|1, \alpha)$, the resulting distribution on $\pi$ is known as the *GEM distribution*, and we write $\pi \sim \text{GEM}(\alpha)$.

The GEM distribution is closely related to the Dirichlet process (DP). Define a measure on $\Omega_\beta$ as

$$\mathcal{M} = \sum_{k=1}^{\infty} \pi_k \delta_{\beta_k},$$

which places atoms at points $\beta_k$ with weight $\pi_k$. When $\pi \sim \text{GEM}(\alpha)$ and $\beta_k \overset{iid}{\sim} \mathcal{P}_{\text{base}}(\beta_k)$, $\mathcal{M}$ is a random measure is distributed according to Dirichlet process with concentration parameter $\alpha$ and base measure $\mathcal{P}_{\text{base}}$ (Ferguson, 1973; Sethuraman, 1994).

We keep the generic notation $\mathcal{P}_{\text{stick}}$ for stick-breaking distributions because in our sensitivity analysis, we will consider stick-breaking distributions that are outside the family of Beta distributions.

In the generative process we have just outlined, the joint distribution of the observed data and latent variables in a basic BNP mixture model is

$$\begin{aligned} \log \mathcal{P}(x, \beta, z, \nu) = &\sum_{n=1}^{N} \sum_{k=1}^{\infty} z_{nk} \left( \log \mathcal{P}(x_n|\beta_k) + \log \pi_k \right) \\ &+ \sum_{k=1}^{\infty} \left( \log \mathcal{P}_{\text{stick}}(\nu_k) + \log \mathcal{P}_{\text{base}}(\beta_k) \right). \end{aligned} \tag{3.2}$$

**Example 1** (Gaussian mixture model)**.** The observations are vectors $x_n \in \mathbb{R}^d$, and we model each component with a multivariate Gaussian. In this model, $\beta_k = (\mu_k, \Lambda_k)$, where $\mu_k \in \mathbb{R}^d$, $\Lambda_k$ is a $d \times d$ positive definite information matrix, and

$$\begin{aligned} \mathcal{P}(x_n|\beta_k) &= \mathcal{N}\left(x_n|\mu_k, \Lambda_k^{-1}\right) \\ \log \mathcal{P}(x_n|\beta_k) &= -\frac{1}{2}(x_n - \mu_k)^T \Lambda_k (x_n - \mu_k) + \frac{1}{2}\log|\Lambda_k| + C. \\ &\quad (C \text{ does not depend on } \beta_k) \end{aligned}$$

We let $\mathcal{P}_{\text{base}}(\beta_k)$ be the conjugate prior, which in this case is normal-Wishart:

$$\begin{aligned} \mathcal{P}_{\text{base}}(\beta_k) &= \mathcal{NW}\left(\beta_k|\tau_0, n_0, p_0, V_0\right) \\ \log \mathcal{P}_{\text{base}}(\beta_k) &= -\frac{\tau_0}{2}(\mu_k - \mu_0)^T \Lambda_k (\mu_k - \mu_0) \\ &\quad + \frac{n_0 - p_0 - 1}{2}\log|\Lambda_k| - \frac{1}{2}\text{Tr}(V_0\Lambda_k) + C, \end{aligned}$$

where $(\tau_0, n_0, p_0, V_0)$ are fixed prior parameters.

△

We start with a Gaussian mixture model (GMM) because it conforms cleanly to the generative process culminating in Equation 3.2. In Section 3.6, we fit a GMM to Fisher's iris data set (Anderson, 1936; Fisher, 1936) and cluster irises into latent species based on morphological measurements. The next two examples, which we will apply to real data sets, require more careful modeling considerations, and we adjust the factorization in Equation 3.2 to suit our purposes.

**Example 2** (Regression mixture model)**.** We cluster time-course gene expression data. An observation $x_n \in \mathbb{R}^M$ is a vector of expression levels at $M$ time points. Let $A$ be an $M \times d$ regressor matrix. In our case, we will use cubic B-splines to smooth the time-course observations, so the $ij$-th entry of $A$ will be the $j$-th B-spline basis vector evaluated at the $i$-th time point (Section 3.6.2).

Each component is characterized by a vector of regression coefficients $\mu_k$ and a variance $\tau_k^{-1}$, so in this model, $\beta_k = (\mu_k, \tau_k)$. The distribution of the data arising from component $k$ is

$$\mathcal{P}(x_n|\beta_k, b_n) = \mathcal{N}\left(x_n|A\mu_k + b_n, \tau_k^{-1} I_{M\times M}\right),$$

where $b_n$ is a gene-specific additive offset and $I$ is the identity matrix. We include the additive offset because we are interested in clustering gene expressions based on their patterns over time, not their absolute level.

The joint distribution can be written in the same form as Equation 3.2, except that the conditional data likelihood now depends on $b_n$ as well as $\beta_k$, and we include an additional prior term for $b_n$. △

Our last example is a Bayesian topic model applied to genetic data. Genotypes at genetic markers take the place of words in a document; in lieu of inferring "topics," we infer latent populations.

**Example 3** (A topic model for population structure)**.** We consider genetic data where the data set consists of $N$ individuals genotyped at $L$ loci. For diploid organisms, there are two observations at each loci, one at each chromosome. Let $x_{nli} \in \{1, \ldots, J_l\}$ be the observed genotype for individual $n$ at locus $l$ and chromosome $i$; $J_l$ is the number of possible genotypes at locus $l$. For example, if the measurements are single nucleotides (A, T, C or G) then $J_l = 4$ for all $l$.

A latent population is characterized by the collection $\beta_k = (\beta_{k1}, \ldots, \beta_{kL})$ where $\beta_{kl} \in \Delta^{J_l - 1}$ are the latent frequencies for the $J_l$ possible genotypes at locus $l$. Let $z_{nli}$ be the assignment of observation $x_{nli}$ to a latent population. Notice that for a given individual $n$, different loci, or even different chromosomes at a given locus, may have different population assignments. The distribution of $x_{nli} \in \{1, \ldots, J_l\}$ arising from population $k$ is

$$\mathcal{P}(x_{nli}|\beta_k) = \text{Categorical}\left(x_{nli}|\beta_{kl}\right).$$

Unlike the previous models, we now have a stick-breaking process for each individual. Draw sticks

$$\nu_{nk} \overset{iid}{\sim} \mathcal{P}_{\text{stick}}(\nu_{nk}) \quad \forall n = 1, \ldots, N; k = 1, 2, \ldots \infty.$$

The prior assignment probability vector $\pi_n = (\pi_{n1}, \pi_{n2}, \ldots)$, now unique to each individual, is formed by the same stick-breaking construction as before,

$$\pi_{nk} = \nu_{nk} \prod_{k' < k} (1 - \nu_{nk'}).$$

The population assignment $z_{nli}$ is drawn from the usual multinomial distribution

$$p(z_{nli}|\pi_n) = \prod_{k=1}^{\infty} \pi_{nk}^{z_{nlik}}.$$

In this genetics application, we call $\pi_n$ the *admixture* of individual $n$.

The joint log-likelihood decomposes as

$$\begin{aligned}\log \mathcal{P}(x, \beta, z, \nu) = &\sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{i=1}^{2} \sum_{k=1}^{\infty} z_{nlik} \left(\log \mathcal{P}(x_{nli}|\beta_k) + \log \pi_{nk}\right) \\ &+ \sum_{n=1}^{N} \sum_{k=1}^{\infty} \log \mathcal{P}_{\text{stick}}(\nu_{nk}) + \sum_{k=1}^{\infty} \log \mathcal{P}_{\text{base}}(\beta_k).\end{aligned}$$

This model is identical to STRUCTURE, a model proposed in Pritchard et al. (2000); Raj et al. (2014), except that we replace the Dirichlet prior in STRUCTURE with an infinite stick-breaking process. The result is a model similar to a hierarchical Dirichlet process for topic modeling, (Teh et al., 2006), but without the top-level Dirichlet process. △

## 3.2 Variational approximation for Dirichlet processes

Let $\zeta$ denote the full vector of latent variables. In the GMM and topic model (Examples 1 and 3), $\zeta := (\beta, z, \nu)$; in the regression example (Example 2), $\zeta$ includes the additive shifts, $\zeta := (\beta, z, \nu, b)$. In each model, the exact posterior distribution $\mathcal{P}(\zeta|x)$ is intractable. Variational Bayes (VB) is an approach that seeks an approximate posterior through solving a numerical optimization problem (Jordan et al., 1999; Wainwright and Jordan, 2008; Blei et al., 2017).

VB specifies a family of approximating distributions $\mathcal{Q}(\zeta|\eta)$ parameterized by a finite-dimensional vector $\eta \in \Omega_\eta \subseteq \mathbb{R}^{D_\eta}$ and solves for $\mathcal{Q}(\zeta|\hat{\eta})$ that is closest to the posterior $\mathcal{P}(\zeta|x)$ in Kullback-Leibler (KL) divergence:

$$\hat{\eta} := \operatorname*{argmin}_{\eta\in\Omega_\eta} \mathrm{KL}\left(\mathcal{Q}(\zeta|\eta)||\mathcal{P}(\zeta|x)\right) \quad \text{where} \tag{3.3}$$
$$\mathrm{KL}\left(\mathcal{Q}(\zeta|\eta)||\mathcal{P}(\zeta|x)\right) = \mathop{\mathbb{E}}_{\mathcal{Q}(\zeta|\eta)}\left[\log \mathcal{Q}(\zeta|\eta) - \log \mathcal{P}(x,\zeta)\right] + \log \mathcal{P}(x).$$

As we discuss below, we will choose $\mathcal{Q}(\zeta|\eta)$ so that we can easily evaluate (or approximate) the above expectation with respect to $\mathcal{Q}(\zeta|\eta)$ as a closed-form function of $\eta$. Notice that the intractable $\log \mathcal{P}(x)$ term does not depend on $\eta$, and so can be neglected in the optimization.

In practice, forming an approximating posterior for BNP can be challenging since the latent variables $\nu$ and $\beta$ are (countably) infinite dimensional. We would like to keep dimension of the variational parameter $\eta$ finite in order for the optimization in Equation 3.3 to be tractable. In the present paper, we will follow Blei and Jordan (2006) and use a truncated stick-breaking representation in the variational distribution. We choose a truncation parameter $K_{\max}$ large but finite, and we set $\mathcal{Q}(\nu_k = 1|\eta) = 1$ for all $k > K_{\max}$. This implies that under $\mathcal{Q}$, $\pi_k = 0$ with probability one for all $k > K_{\max}$ (Equation 3.1). Correspondingly, we also set $\mathcal{Q}(z_{nk} = 0|\eta) = 1$ for $k > K_{\max}$.

For the generic BNP mixture model in Equation 3.2, we propose a mean-field variational approximating family of the following form:

$$\mathcal{Q}(\zeta|\eta) = \left(\prod_{k=1}^{K_{\max}-1} \mathcal{Q}(\nu_k|\eta)\right)\left(\prod_{k=1}^{K_{\max}} \mathcal{Q}(\beta_k|\eta)\right)\left(\prod_{n=1}^{N} \mathcal{Q}(z_n|\eta)\right). \tag{3.4}$$

Because $\pi_k = 0$ for all $k > K_{\max}$, we can ignore the latent variables $\beta_k$ for $k > K_{\max}$ in defining our variational approximation.

Notice that only our variational approximation is truncated—the model (Equation 3.2) itself is not finite. We set $K_{\max}$ large enough in our variational approximation to ensure that a large proportion of the components are unoccupied with high probability under $\mathcal{Q}$, in which case the truncation approximates the fully nonparametric model with $K_{\max} = \infty$.

The variational approximation for the topic model (Example 3) is similarly mean-field: the distribution on stick-breaking proportions $\nu$ factorizes over both individuals $n$ and components $k$, while the assignments $z$ factorize over individuals $n$, loci $l$, and chromosomes

$i$. For the regression model (Example 2), all terms in the variational approximation fully-factorize except for the cluster assignments $z$ and additive shifts $b$. While we assume $(z, b)$ to be independent from all other latent variables under $\mathcal{Q}$, we will allow conditional dependence between $z$ and $b$ (Section 3.9.1).

### Conditional conjugacy

For $z$ and $\beta$ in all models we consider, we will take advantage of conditional conjugacy to choose distributions $\mathcal{Q}(z_n|\eta)$ and $\mathcal{Q}(\beta_k|\eta)$, unless otherwise stated. This means that we will take $\mathcal{Q}(z_n|\eta)$ to be multinomial, matching $\mathcal{P}(z_n|x, \beta, \nu)$, and we will take $\mathcal{Q}(\beta_k|\eta)$ to match the distribution of $\mathcal{P}(\beta_k|x, z, \nu)$.

**Example 4** (VB approximation for $\beta_k$ in a GMM)**.** To evaluate the expectation in Equation 3.3, we need to compute the expected joint log-likelihood

$$\mathop{\mathbb{E}}_{\mathcal{Q}(\beta_k|\eta)} \left[\log \mathcal{P}(x_n, \beta_k)\right].$$

In Example 1, $\beta_k = (\mu_k, \Lambda_k)$, and the likelihoods are Gaussian, $\mathcal{P}(x_n|\beta_k) = \mathcal{N}\left(x_n|\mu_k, \Lambda_k^{-1}\right)$. The prior $\mathcal{P}_{\text{base}}(\beta_k)$ a normal-Wishart. Using the log densities displayed in Example 1, observe that $\beta_k$ enters the expected joint log-likelihood only through the expected moments

$$\mathop{\mathbb{E}}_{\mathcal{Q}(\beta_k|\eta)} \left[\Lambda_k\right], \quad \mathop{\mathbb{E}}_{\mathcal{Q}(\beta_k|\eta)} \left[\log|\Lambda_k|\right], \quad \mathop{\mathbb{E}}_{\mathcal{Q}(\beta_k|\eta)} \left[\Lambda_k \mu_k\right], \quad \mathop{\mathbb{E}}_{\mathcal{Q}(\beta_k|\eta)} \left[\mu_k \Lambda_k \mu_k\right].$$

The conditionally conjugate variational distribution on $\beta_k$ is normal-Wishart, which we denote as $\mathcal{Q}(\beta_k|\eta) = \mathcal{NW}(\beta_k|\eta)$. With this choice of $\mathcal{Q}(\beta_k|\eta)$, all the preceding expected moments can be provided as closed-form functions of $\eta$. △

Under the mean-field factorization (Equation 3.4), the vector $\eta$ will partition into parameters governing $\nu$, $\beta$, and $z$. Let the parameters governing a particular latent variable or latent vector be denoted with a subscript: for example, $\mathcal{Q}(\beta|\eta) = \mathcal{Q}(\beta|\eta_\beta)$, $\mathcal{Q}(z_n|\eta) = \mathcal{Q}(z|\eta_{z_n})$, and so on. With conditionally conjugate distributions and our mean-field assumption, the parameters $\eta_{z_n}$ can be optimally set as a function of parameters $\eta_\beta$ and $\eta_\nu$. The next example details this point.

**Example 5** (VB approximation for $z_n$ in a GMM)**.** The conditionally conjugate variational distribution for $z_n$ is multinomial. Our variational approximation is truncated at $K_{\max}$ so $z_{nk} = 0$ for all $k > K_{\max}$; the multinomial distribution under $\mathcal{Q}$ has $K_{\max}$ discrete categories.

We parameterize the the multinomial distribution using its natural parameterization in exponential family form. That is, we let $\eta_{z_n} = (\rho_{n1}, \rho_{n2}, ..., \rho_{n(K_{\max}-1)})$ be an unconstrained vector in $\mathbb{R}^{K_{\max}-1}$; in this parameterization, the multinomial expectations are

$$p_{nk} := \mathop{\mathbb{E}}_{\mathcal{Q}(z_n|\eta_z)} \left[z_{nk}\right] = \frac{\exp(\rho_{nk})}{1 + \sum_{k'=1}^{K_{\max}-1} \exp(\rho_{nk})}$$

We use the exponential family parameterization because we will require the optimal variational parameters $\hat{\eta}$ to be interior to $\Omega_\eta$ in our sensitivity analysis (Section 3.3). In the mean parameterization, $\sum_{k=1}^{K_{\max}} p_{nk} = 1$, so the optimal mean parameters $\hat{p}_n$ cannot be interior to $\Delta^{K_{\max}-1}$. On the other hand, $\eta_{z_n}$ as defined is unconstrained in $\mathbb{R}^{K_{\max}-1}$.

Moreover, with the distributions $\mathcal{Q}(\beta|\eta_\beta)$ and $\mathcal{Q}(\nu|\eta_\nu)$ fixed, the parameter vector $\eta_{z_n}$ that minimizes Equation 3.3 has a closed form. Fixing $\mathcal{Q}(\beta|\eta_\beta)$ and $\mathcal{Q}(\nu|\eta_\nu)$, the optimal $\hat{\eta}_{z_n}$ must satisfy

$$\begin{aligned}\mathcal{Q}(z_n|\hat{\eta}_{z_n}) &\propto \exp\left(\tilde{\rho}_{nk}\right) \\ \text{where} \quad \tilde{\rho}_{nk} &:= \mathop{\mathbb{E}}_{\mathcal{Q}(\beta,\nu|\eta)}\left[\log \mathcal{P}(x_n|\beta_k) + \log \pi_k\right].\end{aligned}$$

See Bishop (2006) and Blei et al. (2017) for details. To satisfy this optimality condition, we set the optimal $\hat{\eta}_{z_n}$ to be

$$\hat{\eta}_{z_n} = \left(\log \frac{\tilde{\rho}_{n1}}{\tilde{\rho}_{nK_{\max}}}, \log \frac{\tilde{\rho}_{n2}}{\tilde{\rho}_{nK_{\max}}}, \ldots, \log \frac{\tilde{\rho}_{n(K_{\max}-1)}}{\tilde{\rho}_{nK_{\max}}}\right).$$

Thus, as long as the expectation $\tilde{\rho}_{nk}$ has a closed-form as a function of $(\eta_\beta, \eta_\nu)$, the optimal $\hat{\eta}_{z_n}$ can be also be set in closed-form as a function of $(\eta_\beta, \eta_\nu)$. △

For fixed $(\eta_\beta, \eta_\nu)$, the option of setting $\eta_z$ at its optimum extends beyond the GMM example and will play a key role in computing our local sensitivity measures in practice (Section 3.5). In greater generality, each of our example models has latent variables that factorize in a global/local structure. In the GMM example discussed above, we call the variables $(\beta, \nu)$ "global" because they are shared across all data points; the $z$ is "local" because each $z_n$ is unique to a single data point. In the regression model (Example 2), the global variables are again $(\beta, \nu)$, but the local variables comprise of both the cluster assignments $z$ and additive shifts $b$. In these two models, notice that the dimension of global variables scale with $K_{\max}$, while the dimension of local variables scale with the number of observations $N$.

In the topic model (Example 3), we still call $(\beta, \nu)$ the global latent variables, even though they scale with the number of individuals $N$; they do not, however, scale with both the number of individuals and the number of loci like $z$ does. In the topic model, we call $z$ the local latent variables.

Let $\gamma = (\beta, \nu)$ be the global latent variables and let $\eta_\gamma = (\eta_\beta, \eta_\nu)$ be their variational parameters. Let $\eta_\ell$ be the local variational parameters. In Example 1 and Example 3, $\eta_\ell = \eta_z$, while in Example 2, $\eta_\ell = (\eta_z, \eta_b)$.

In each model we consider, for $\eta_\gamma$ fixed, the optimal $\eta_\ell$ that minimizes the KL can be set in closed form as a function of $\eta_\gamma$. The multinomial parameters for $\eta_{z_n}$ in the regression and topic models can be set in the same way as described in Example 5. For more details concerning the optimal shift parameters $\eta_b$ in the regression model, see Section 3.9.1.

**Evaluating stick expectations**

To evaluate the KL in Equation 3.3, we also need the expectations over stick-breaking proportions,

$$\mathop{\mathbb{E}}_{\mathcal{Q}(\nu_k|\eta)}\left[\log \nu_k\right], \quad \mathop{\mathbb{E}}_{\mathcal{Q}(\nu_k|\eta)}\left[\log(1-\nu_k)\right], \quad \text{and} \quad \mathop{\mathbb{E}}_{\mathcal{Q}(\nu_k|\eta)}\left[\log \mathcal{P}_{\text{stick}}(\nu_k)\right]. \tag{3.5}$$

(The discussion in this subsection applies to the topic model as well, with stick-breaking proportions indexed by $nk$). The first two expectations appear in the KL when decomposing the mixture weights $\mathbb{E}\left[\log \pi\right]$ into its component stick-breaking proportions (Equation 3.1).

If the prior $\mathcal{P}_{\text{stick}}(\nu_k)$ were Beta-distributed like in the GEM construction, then the conditionally conjugate distribution for $\mathcal{Q}(\nu_k|\eta)$ would also be Beta. In this case, all the displayed expectations in Equation 3.5 can be computed analytically as a function of the Beta parameters in the variational approximation.

However, we will be considering stick-breaking distributions $\mathcal{P}_{\text{stick}}(\nu_k)$ that are outside the family of Beta distributions. To accommodate a generic prior $\mathcal{P}_{\text{stick}}(\nu_k)$, we approximate the expectations in Equation 3.5 numerically. Each expectation is a univariate integral. A particularly easy approximation method is Gauss-Hermite (GH) quadrature, which we now describe.

To take advantage of GH quadrature, we first logit transform the stick-breaking proportion $\nu_k$ so that the transformed variable

$$\tilde{\nu}_k := \log\left(\frac{\nu_k}{1-\nu_k}\right)$$

is not constrained to be between $(0,1)$ and can take values in all of $\mathbb{R}$. Let $s$ be the sigmoid function, which provides the inverse transformation,

$$\nu_k = s(\tilde{\nu_k}) := \frac{\exp(\tilde{\nu}_k)}{1+\exp(\tilde{\nu}_k)}.$$

We choose $\mathcal{Q}(\tilde{\nu}_k|\eta)$ to be normally distributed with location parameter $\eta_k^\mu$ and scale parameter $\eta_k^\sigma$. This then induces a logit-normal distribution on our original variable of interest, $\nu_k$. To compute expectations of a smooth function $f(\nu_k)$ (such as $f(\nu_k) = \mathcal{P}_{\text{stick}}(\nu_k)$), the law of the unconscious statistician states that,

$$\mathop{\mathbb{E}}_{\mathcal{Q}(\nu_k|\eta)}\left[f(\nu_k)\right] = \mathop{\mathbb{E}}_{\mathcal{Q}(\tilde{\nu}_k|\eta)}\left[f \circ s\left(\tilde{\nu}_k\right)\right].$$

By choosing $\mathcal{Q}(\tilde{\nu}_k|\eta)$ to be Gaussian, the right-hand side is a Gaussian integral, which we approximate using GH quadrature with $N_{\text{GH}}$ knots, located at $\xi_g$, weighted by $\omega_g$:

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathcal{Q}(\tilde{\nu}_k|\eta)}\left[f \circ s\left(\tilde{\nu}_k\right)\right] &\approx \sum_{g=1}^{N_{\text{GH}}} \omega_g f \circ s\left(\eta_k^\sigma \xi_g + \eta_k^\mu\right) \\ &=: \mathop{\widehat{\mathbb{E}}}_{\mathcal{Q}(\nu_k|\eta)}\left[f(\nu_k)\right].\end{aligned} \tag{3.6}$$

Using GH quadrature to approximate the expectation is similar to the "reparameterization trick," only using GH points rather than standard normal draws. Conveniently, the approximation $\widehat{\mathbb{E}}_{\mathcal{Q}(\nu_k|\eta)}[f(\nu_k)]$ is a deterministic, differentiable function of $\eta_k^\mu$ and $\eta_k^\sigma$, and so also of $\eta$. This will be useful in our sensitivity computations in the next section.

### 3.2.1 Posterior quantities

In a VB approach, all posterior quantities of interest can be expressed as functions of the variational parameter $\eta$. We will use $g(\eta)$ to denote such quantities. Often, $g(\eta)$ takes the form of an expectation over $\mathcal{Q}$,

$$g(\eta) = \mathop{\mathbb{E}}_{q(\zeta|\eta)}[f(\zeta)]$$

for some function $f(\eta)$.

In the next few examples, we define some posterior quantities that we will consider in Section 3.6. We will evaluate the sensitivity of these quantities to the prior specification $\mathcal{P}_{\text{stick}}$.

**Example 6** (The in-sample number of clusters)**.** One might ask, *how many clusters are present in the data set*? For example, in the iris data set, answering this question has the interpretation of counting the number of iris species present. To estimate the number of clusters in the context of a BNP model, define the random variable

$$G_\tau := \sum_{k=1}^{K_{\max}} \mathbb{I}\left(\left(\sum_{n=1}^{N} z_{nk}\right) > \tau\right),$$

where $\mathbb{I}(\cdot)$ is the indicator function. $G_\tau$ counts the number of clusters with at least $\tau$ observations in a set of assignments $z$. The expected number of clusters under the variational posterior is

$$g_{\text{cl},\tau}(\eta) := \mathop{\mathbb{E}}_{\mathcal{Q}(z|\eta)}[G_\tau].$$

When $\tau = 0$, $g_{\text{cl},0}$ can be written as a function with respect to the assignment probabilities

$$g_{\text{cl},0}(\eta) = \sum_{k=1}^{K_{\max}}\left(1 - \prod_{n=1}^{N}\left(1 - \mathop{\mathbb{E}}_{\mathcal{Q}(z_{nk}|\eta)}[z_{nk}]\right)\right).$$

△

**Example 7** (The predictive number of clusters)**.** In the Bayesian approach, we can formulate the posterior predictive question, *how many clusters would be present if a new data set were collected*? In the iris example, this can interpreted as predicting the number of species one

might see if a fresh sample of iris flowers were collected. Under the BNP model, the expected number of predictive clusters is defined as

$$g_{\text{p.cl},\tau}(\eta) := \mathop{\mathbb{E}}_{\mathcal{Q}(\pi|\eta)} \left[ \mathop{\mathbb{E}}_{\mathcal{P}(z|\pi)} [G_\tau] \right].$$

Notice that the inner expectation conditions on $\pi$ and the randomness is over $z$ sampled from the generative model $z \sim \mathcal{P}(z|\pi)$. We can write out the inner expectation:

$$g_{\text{p.cl},\tau}(\eta) = \mathop{\mathbb{E}}_{\mathcal{Q}(\pi|\eta)} \left[ \sum_{k=1}^{K_{\max}} \left( 1 - \sum_{i=0}^{\lfloor \tau \rfloor} \binom{N}{i} (1-\pi_k)^N \right) \right],$$

where we use the convention that $\binom{N}{0} = 1$. △

**Example 8** (Co-clustering)**.** Finally, in a clustering problem, we are often interested in understanding which observations group with each other. One way to visualize the clusters is to construct the co-clustering matrix,

$$g_{\text{cc}}(\eta) := \mathop{\mathbb{E}}_{\mathcal{Q}(z|\eta)} \left[ z z^T \right],$$

where we view $z$ as a $N \times K_{\max}$ matrix of cluster assignments. Unlike the quantities in Examples 6 and 7, $g_{\text{cc}}$ is a matrix quantity, not a scalar quantity.

△

For some posterior quantities, the expectation over $\mathcal{Q}$ will not be a simple closed-form function of $\eta$. For example, computing $g_{\text{cl},\tau}$ with a threshold $\tau > 0$ requires forming all $\binom{N}{\tau}$ combination of $\tau$-length products $\mathbb{E}[z_{n_1 k}] \times \ldots \times \mathbb{E}[z_{n_\tau k}]$ for each $k$. In such cases, we resorted to Monte-Carlo approximations of the expectation. Specifically, we used the "reparameterization trick" to sample from the variational distribution. In the case of $g_{\text{cl},\tau}$, we constructed an $\eta$-dependent transformation $f(\cdot, \eta)$ that satisfies

$$u \overset{iid}{\sim} \text{Uniform}(0,1)^{NK_{\max}} \implies f(u,\eta) \overset{d}{=} z \sim \mathcal{Q}(\cdot|\eta).$$

To form a Monte Carlo estimate of $g_{\text{cl},\tau}$, we sampled $u_1, \ldots, u_m$ uniformly, and then averaged the expression inside the expectation evaluated at points $f(u_1,\eta), \ldots, f(u_m,\eta)$. The uniform draws $u_1, \ldots, u_m$ can be fixed beforehand. This is important for two reasons. First, we will be evaluating the same $g$ at different parameter vectors $\eta$; conditional on the fixed $m$ uniform draws, $g$ will be a deterministic function of $\eta$, and we can compare how $g$ changes without stochasticity. Secondly, in the construction of our "influence function" (Section 3.4), it will be useful to evaluate the gradient of $g$ with respect to $\eta$; conditional on the random draws, $g$ (or more precisely, our Monte Carlo approximation of $g$), will be differentiable with respect to $\eta$.

## 3.3 Local sensitivity

We now turn to the optimization problem Equation 3.3 where the priors $\mathcal{P}_{\text{stick}}(\nu_k)$ which enter the term $\log \mathcal{P}(\zeta)$ depend on a real-valued parameter, $t \in \Omega_t \subseteq \mathbb{R}$, writing $\mathcal{P}_{\text{stick}}(\nu_k|t)$. Starting now, we will state general results in terms of a generic parameter $\theta$, specializing to the BNP problem in examples and in the experiments below.

**Definition 1.** Let $\mu$ denote a sigma finite measure, and let $\mathcal{P}(\theta|t)$ denote a class of probability densities relative $\mu$. Let $\mathcal{P}(\theta|t)$ be defined for $t$ in an open set $\mathcal{B}_t \subseteq \mathbb{R}$ containing 0. Assume that the variational densities $\mathcal{Q}(\theta|\eta)$ are also defined relative to $\mu$.

Let $\tilde{\mathcal{Q}}$ and $\tilde{\mathcal{P}}$ refer to potentially unnormalized (but normalizable) versions of the respectively corresponding $\mathcal{Q}$ and $\mathcal{P}$, so that,

$$\mathcal{Q}(\theta|\eta) := \frac{\tilde{\mathcal{Q}}(\theta|\eta)}{\int \tilde{\mathcal{Q}}(\theta'|\eta)\mu(d\theta')} \quad \text{and} \quad \mathcal{P}(\theta|t) := \frac{\tilde{\mathcal{P}}(\theta|t)}{\int \tilde{\mathcal{P}}(\theta'|t)\mu(d\theta')}.$$

⊡

The prior depends on $t$; in turn, the posterior $\mathcal{P}(\theta|x,t)$ depends on $t$; in turn, the KL divergance depends on $t$; in turn, the optimal variational parameters depend on $t$. Define the shorthand notation

$$\text{KL}(\eta,t) := \text{KL}(\mathcal{Q}(\theta|\eta)||\mathcal{P}(x|\zeta,t)) \quad \text{and} \quad \hat{\eta}(t) := \underset{\zeta\in\Omega_\eta}{\text{argmin}}\, \text{KL}(\eta,t), \tag{3.7}$$

where we write $\hat{\eta}(t)$ to emphasize the dependence of the optimum on $t$. In Definition 1, we take $t = 0$ at the "original" problem, Equation 3.3, without loss of generality. We will thus continue to use $\hat{\eta}$ with no argument to refer to $\hat{\eta}(0)$.

**Example 9.** When drawing from the classical GEM($\alpha$) distribution, we take $\mu$ to be the Lebesgue measure on $[0,1]$ and model

$$\begin{aligned} \mathcal{P}_{\text{stick}}(\nu_k|\alpha) &= \text{Beta}(\nu_k|1,\alpha) \Rightarrow \\ \log \mathcal{P}_{\text{stick}}(\nu_k|\alpha) &= (\alpha-1)\log(1-\nu_k) + C. \qquad (C \text{ does not depend on } \nu_k) \end{aligned}$$

Fix some "original" $\alpha_0$. In this case, we represent deviations from the choice $\alpha_0$ by identifying $t$ with $\alpha - \alpha_0$:

$$\log \mathcal{P}_{\text{stick}}(\nu_k|t) = (t+\alpha_0-1)\log(1-\nu_k) + C.$$

The prior on the full $\nu$ vector is given by

$$\log \mathcal{P}_{\text{stick}}(\nu|t) = \sum_{k=1}^{K_{\max}-1} (t+\alpha_0-1)\log(1-\nu_k) + C.$$

△

Assuming for the moment that $t \mapsto \hat{\eta}(t)$ is continuously differentiable (we will consider its differentiability in detail below), and that we have already computed the solution $\hat{\eta}$ to the "original" problem Equation 3.3 with $t = 0$, then we can form a Taylor series approximation to $\hat{\eta}(t)$. Specifically, we define

$$\hat{\eta}^{\text{lin}}(t) := \hat{\eta} + \left.\frac{d\hat{\eta}(t)}{dt}\right|_{t=0} t. \tag{3.8}$$

Evaluating $\hat{\eta}(t)$ requires solving a new optimization problem, but, given $d\hat{\eta}(t)/dt|0$, evaluating $\hat{\eta}^{\text{lin}}(t)$ involves only multiplication and addition. When $|t|$ is small, by continuous differentiability of $\hat{\eta}(t)$, we might hope that $\hat{\eta}(t) \approx \hat{\eta}^{\text{lin}}(t)$, and so we can use $\hat{\eta}^{\text{lin}}(t)$ to quickly approximate a time-consuming optimization problem.

Futhermore, or functions of interest $g(\eta)$ which are themselves differentiable, we can use the chain rule to compute

$$\left.\frac{dg(\hat{\eta}(t))}{dt}\right|_{t=0} \left.\frac{\partial g(\eta)}{\partial \eta^T}\right|_{\hat{\eta}} \left.\frac{d\hat{\eta}(t)}{dt}\right|_{t=0} \tag{3.9}$$

$$g^{\text{lin}}(t) := g(\hat{\eta}) + \left.\frac{dg(\hat{\eta}(t))}{dt}\right|_{t=0} t. \tag{3.10}$$

For non-differentiable functions of $\eta$, we can still form the approximation

$$g^{\eta.\text{lin}}(t) := g(\hat{\eta}^{\text{lin}}(t)).$$

The advantage of $g^{\text{lin}}(t)$ relative to $g^{\eta.\text{lin}}(t)$ is that for the former we can compute influence functions and worst-case perturbations, as we discuss below. Converse, $g^{\eta.\text{lin}}(t)$ may be expected to provide a better approximation in some cases since it retains non-linearities in the map $\eta \mapsto g(\eta)$, linearizing only the computationally intensive map $t \mapsto \hat{\eta}(t)$.

When, then, is $\hat{\eta}(t)$ continuously differentiable? We will now state some sufficient conditions under which we can apply the implicit function theorem (e.g., Krantz and Parks (2012)) to prove the continuous differentiability of $\hat{\eta}(t)$.

The first key assumption, Assumption 1, states sufficient conditions for which we can apply the dominated convergence theorem to variational expectations of some generic function $\psi(\theta, t)$, allowing us to translate continuity of the variational and model densitites into continuity of the variational objective. The details can be found in Lemmas 3 and 4 of Section 3.9.1. In case Assumption 1 seems forbidding, observe that, in lieu of Assumption 1 we might equivalently have said that we can exchange limits and variational expectations whenever needed. Assumption 1 is simply a precise catalogue of what is needed.

**Assumption 1.** *Assume that the map $\eta \mapsto \log \tilde{\mathcal{Q}}(\theta|\eta)$ is twice continuously differentiable. Let $\psi(\theta, t)$ be a scalar-valued $\mu$-measurable function of $\theta$ and $t$. Assume that the map $t \mapsto \psi(\theta, t)$ is continuously differentiable.*

*Define the following shorthand notation:*

$$\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta) := \left.\frac{\partial \log \tilde{\mathcal{Q}}(\theta|\eta)}{\partial \eta}\right|_\eta$$
$$\nabla^2_\eta \log \tilde{\mathcal{Q}}(\theta|\eta) := \left.\frac{\partial^2 \log \tilde{\mathcal{Q}}(\theta|\eta)}{\partial \eta \partial \eta^T}\right|_\eta$$
$$\nabla_t \psi(\theta, t) := \left.\frac{\partial \psi(\theta, t)}{\partial t}\right|_t.$$

*For a given $t_0$ and $\eta_0$, assume there exists some neighborhood of $t_0$, $\mathcal{B}_t$, some neighborhood of $\eta_0$, $\mathcal{B}_\eta$, and a $\mu$-integrable $M_\psi(\theta)$ with $\int M_\psi(\theta)\mu(d\theta) < \infty$ such that the following bounds hold for all $\eta, t \in \mathcal{B}_\eta \times \mathcal{B}_t$:*

1. $\tilde{\mathcal{Q}}(\theta|\eta)\psi(\theta, t) \leq M_\psi(\theta)$.
2. $\tilde{\mathcal{Q}}(\theta|\eta) \left\|\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)\right\|_2 \psi(\theta, t) \leq M_\psi(\theta)$.
3. $\tilde{\mathcal{Q}}(\theta|\eta) \left\|\nabla^2_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)\right\|_2 \psi(\theta, t) \leq M_\psi(\theta)$.
4. $\tilde{\mathcal{Q}}(\theta|\eta) \left\|\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)\right\|_2 \nabla_t \psi(\theta, t) \leq M_\psi(\theta)$.
5. $\tilde{\mathcal{Q}}(\theta|\eta) \left\|\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)\right\|_2^2 \psi(\theta, t) \leq M_\psi(\theta)$.

**Lemma 1.** *Under Assumption 1, we can exchange the order of integration and differentiation in*

$$\left.\frac{\partial \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta, t)]}{\partial \eta}\right|_{\hat{\eta}, t=0}, \quad \left.\frac{\partial^2 \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta, t)]}{\partial \eta \partial \eta}\right|_{\hat{\eta}, t=0}, \text{ and } \left.\frac{\partial^2 \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta, t)]}{\partial \eta \partial t}\right|_{\hat{\eta}, t=0}.$$

*(See also Lemmas 3 and 4 of Section 3.9.1 for a more detailed statement.)*

Next, in Assumption 2, we first require some regularity conditions on the KL divergence and its optimum.

**Assumption 2.** *Let the following conditions on the variational approximation hold.*

1. *The map $\eta \mapsto \mathrm{KL}(\eta, 0)$ is twice continuously differentiable at $\hat{\eta}$.*
2. *The optimal $\hat{\eta}$ is interior to $\Omega_\eta$.*
3. *The Hessian matrix $\left.\frac{\partial^2 \mathrm{KL}(\eta,0)}{\partial \eta \partial \eta^T}\right|_{\hat{\eta}}$ is positive definite.*

4. *The unnormalized variational densities $\tilde{\mathcal{Q}}(\theta|\eta)$ satisfy Assumption 1 with $\psi(\theta, t) \equiv 1$ (no $\theta$ dependence).*

Observe that Assumption 1, applied with $\log \mathcal{P}(x|\theta)$, is one way to prove Assumption 2 (Item 1). Since our primary focus is on the prior $\log \mathcal{P}(\theta|t)$, we prefer to simply state Assumption 2 (Item 1) directly and reserve our detailed attention for the prior $\log \mathcal{P}(\theta|t)$.

Finally, in Assumption 3 we draw the needed connection between the class of prior perturbations and the variational approximation.

**Assumption 3.** *Under Definition 1, assume that the variational densities $\mathcal{Q}(\theta|\eta)$ satisfy Assumption 1 with both $\psi(\theta, t) \equiv 1$ (no $\theta$ dependence) and with and $\psi(\theta, t) = \log \mathcal{P}(\theta|t) - \log \mathcal{P}(\theta|t = 0)$.*

**Theorem 1.** *Under the conditions of Definition 1, let Assumption 2 hold at $\eta_0 = \hat{\eta}$. Further, let Assumption 1 hold with*

$$\psi(\theta, t) = \log \mathcal{P}(\theta|t) - \log \mathcal{P}(\theta|t = 0)$$

*at $t_0 = 0$. Define*[1]

$$\begin{aligned}
\overline{\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\hat{\eta}\right)} &:= \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\hat{\eta}\right) - \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\hat{\eta})}\left[\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\hat{\eta}\right)\right], \\
\hat{J} &:= \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\hat{\eta})}\left[\overline{\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\hat{\eta}\right)} \left.\frac{\partial \log \mathcal{P}(\theta|t)}{\partial t}\right|_{t=0}\right] \quad \textit{and} \\
\hat{H} &:= \left.\frac{\partial^2 \mathrm{KL}\left(\eta, 0\right)}{\partial \eta \partial \eta^T}\right|_{\hat{\eta}}.
\end{aligned}$$

*Then the map $t \mapsto \hat{\eta}(t)$ is continuously differentiable at $t = 0$ with derivative*

$$\left.\frac{d\hat{\eta}(t)}{dt}\right|_0 = -\hat{H}^{-1}\hat{J} \tag{3.11}$$

*(For a proof, see Section 3.9.1.)*

We conclude this section by showing that Theorem 1 applies to the setting of BNP stick-breaking.

**Lemma 2.** *In the setting of Assumption 1, let $\theta \in \mathbb{R}$, let $\mu$ be the Lebesgue measure, and let $\mathcal{Q}(\theta|\eta) = \mathcal{N}(\theta|\eta)$ be a normal distribution parameterized by its natural exponential family parameters.*

[1]Note that if $\tilde{\mathcal{Q}}(\theta|\eta)$ is already normalized ($\tilde{\mathcal{Q}} = \mathcal{Q}$), then $\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)\right] = 0$ for all $\eta$ and $\overline{\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\hat{\eta}\right)} = \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\hat{\eta}\right)$. Since it can sometimes be more convenient to work with unnormalized distributions, we keep the notation general.

*Let $\sigma(\eta)$ denote the standard deviation of the normal distribution. Fix $\eta_0$ such that $\sigma(\eta_0) > 0$, and let $\mathcal{B}_\eta$ be an open set containing $\eta_0$ such a that $\sigma_{max} := \sup_{\eta\in\mathcal{B}_\eta} \sigma(\eta) < \infty$.*

*If there exists a neighborhood $\mathcal{B}_t$ of $t_0$ such that $|\psi(\theta, t)|$ and $|\nabla_t \psi(\theta, t)|$ are uniformly bounded by some multiple of $\exp(-|\theta|)$ for all $t \in \mathcal{B}_t$, then $\mathcal{Q}$ and $\psi$ satisfy Assumption 1.*

*Proof.* By properties of the exponential family,

$$\nabla_\eta \log \tilde{\mathcal{Q}}(\theta, \eta) = (\theta, \theta^2)^T \quad \text{and} \quad \nabla^2_\eta \log \tilde{\mathcal{Q}}(\theta, \eta) = 0_{2\times 2} \Rightarrow$$
$$\left\|\nabla_\eta \log \tilde{\mathcal{Q}}(\theta, \eta)\right\|_2^2 = \theta^2 + \theta^4 \quad \text{and} \quad \left\|\nabla^2_\eta \log \tilde{\mathcal{Q}}(\theta, \eta)\right\|_2 = 0.$$

Let $\overline{\mathcal{B}}_\eta$ denote the closure of $\mathcal{B}_\eta$, and let

$$\eta^* := \operatorname*{argmax}_{\eta\in\overline{\mathcal{B}}_\eta} \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)} \left[\exp(|\theta|)\right].$$

By standard properties of the normal and the boundedness of $\sigma(\eta)$, the right hand side of the preceding display is finite. Then

$$\begin{aligned}\int \mathcal{Q}(\theta|\eta)\psi(\theta, t)\mu(d\theta) &\leq \left(\sup_\theta \sup_{t\in\mathcal{B}_t} |\psi(\theta, t)| \exp(-|\theta|)\right) \int \mathcal{Q}(\theta|\eta)\exp(\theta)\mu(d\theta) \\ &\leq C \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta^*)} \left[\exp(|\theta|)\right]. \quad (C \text{ does not depend on } \eta, t)\end{aligned}$$

Therefore, for Assumption 1 (Item 1), we can take $M(\theta) \propto \mathcal{Q}(\theta|\eta^*)\exp(|\theta|)$. The other terms follow similarly, since each multiplier of $\mathcal{Q}(\theta|\eta)$ is dominated by $\exp(-|\theta|)$. The final $M(\theta)$ simply takes the largest of the five constants. □

**Example 10.** To analyze Example 9, we take $\theta$ in Lemma 2 be the unconstrained stick-breaking proportion $\tilde{\nu}_k$, which recall from Section 3.2 was normally distributed under $\mathcal{Q}$. Let $\mu$ be the Lebesgue measure on $\mathbb{R}$.

In Example 9, the parameterization of the stick-breaking distribution was given by

$$\log \mathcal{P}_{\text{stick}}(\nu_k|t) - \log \mathcal{P}_{\text{stick}}(\nu_k|t = 0) = t \log(1 - \nu_k).$$

Expressing the perturbation in terms of $\tilde{\nu}_k$,

$$\begin{aligned}\log \mathcal{P}_{\text{stick}}(\tilde{\nu}_k|t) - \log \mathcal{P}_{\text{stick}}(\tilde{\nu}_k|t = 0) &= t \log\left(1 - \frac{\exp(\tilde{\nu}_k)}{1 + \exp(\tilde{\nu}_k)}\right) \\ &= -t \log\left(1 + \exp(\tilde{\nu}_k)\right).\end{aligned}$$

So, by Lemma 2, Assumption 3 is satisfied with the VB approximation given in Section 3.2 and the parametric perturbation given in Example 9. △

**Corollary 1.** *For the variational approximation of Section 3.2 and perturbation given in Example 9, $\alpha \mapsto \hat{\eta}(\alpha)$ is continuously differentiable.*

*Proof.* We have already shown in Example 10 that Assumption 3 is satisfied. Given that the variational approximations to $\mathcal{P}(z|x,\beta,\nu)$ and $\mathcal{P}(\beta|x,\nu,z)$ are conjugate exponential family approximations, $\eta \mapsto \mathrm{KL}\left(\eta, 0\right)$ is continuous. It remains only to numerically find $\hat{\eta}$ and verify Assumption 2 (Item 3), i.e. that the Hessian is positive definite at the optimum. □

We conclude this section with a brief remark about computing the expectation $\hat{J}$ in our BNP sensitivity analysis. We are interested in sensitivity to the stick-breaking distribution, so only the prior terms on stick-breaking proportions $\nu = (\nu_1, ..., \nu_{K_{\max}-1})$ depends on $t$. Because the elements of $\nu$ fully factorize under both the prior and the variational distributions, $\hat{J}$ decomposes as

$$\begin{aligned}\hat{J} &= \sum_{k=1}^{K_{\max}-1} \mathop{\mathbb{E}}_{\mathcal{Q}(\nu_k|\eta)}\left[\nabla_\eta \log \tilde{\mathcal{Q}}\left(\nu_k|\hat{\eta}\right) \left.\frac{\log \mathcal{P}_{\text{stick}}(\nu_k|t)}{\partial t}\right|_{t=0}\right] \\ &= \sum_{k=1}^{K_{\max}-1} \nabla_\eta \mathop{\mathbb{E}}_{\mathcal{Q}(\nu_k|\eta)}\left[\left.\frac{\log \mathcal{P}_{\text{stick}}(\nu_k|t)}{\partial t}\right|_{t=0}\right]\Bigg|_{\eta=\hat{\eta}(0)}, \end{aligned} \tag{3.12}$$

where we assumed that $\mathcal{Q}(\theta|\eta)$ is normalized, so $\overline{\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\hat{\eta}\right)} = \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\hat{\eta}\right)$, and that the assumptions of Theorem 1 hold, so we can freely exchange derivatives with expectations.

We approximate the expectation using GH quadrature (Equation 3.6), with $f(\nu_k) = \left.\frac{\log \mathcal{P}_{\text{stick}}(\nu_k|t)}{\partial t}\right|_{t=0}$. In all the functional forms for $t \mapsto \mathcal{P}_{\text{stick}}(\nu_k|t)$ considered below, $f(\nu_k)$ can be provided in either closed-form or computed with automatic differentiation. The resulting GH approximation is a deterministic function of $\eta$, and thus the gradient in Equation 3.12 can be computed with another application of automatic differentiation. Note that $\hat{J}$ is sparse in Equation 3.12: it is zero for all entries of $\eta$ other than those that parameterize the sticks.

## 3.4 Function-valued prior perturbations

In Corollary 1 of Section 3.3, we showed that we can form a Taylor series approximation to the dependence of a variational optimum on the parameter $\alpha$ in a Beta prior. However, there is typically no *a priori* reason to believe that the stick breaking prior lies within the parametric Beta family.

Let us fix a base prior density $\mathcal{P}_0(\theta)$ (e.g. a Beta distribution) at which we have computed a VB approximation, and suppose we wish to ask what the variational optimum would have been had we used some alternative prior density, $\mathcal{P}_1(\theta)$. Let us write $\hat{\eta}(\mathcal{P}_0)$ and $\hat{\eta}(\mathcal{P}_1)$ for these two approximations, respectively. To approximately answer this question using the local sensitivity approach of Section 3.3, we must somehow define a continuous path from $\mathcal{P}_0(\theta)$ to $\mathcal{P}_1(\theta)$ parameterized, say, by $t \in [0, 1]$.

To construct this path, we consider *multplicatve* perturbations of the form

$$\mathcal{P}(\theta|t) \propto \mathcal{P}_0(\theta)\exp(t\phi(\theta)) \quad \text{with} \quad \phi(\theta) = \log \mathcal{P}_1(\theta) - \log \mathcal{P}_0(\theta) \tag{3.13}$$

Then $t \mapsto \mathcal{P}(\theta|t)$ parameterizes a path from $\mathcal{P}_0$ to $\mathcal{P}_1$ for $t \in [0,1]$.

When Theorem 1 can be applied, it turns out that the derivative takes the form of an integral against $\phi$. The integrand is known as the "influence function," and characterizes the sensitivity of a function of interest to nonparametric prior perturbations for all $\phi$ to which Theorem 1 can be applied.

**Corollary 2.** *Let Assumption 2 hold at $\eta_0 = \hat{\eta}$. Fix a multiplicative perturbation $\phi$ and let $g(\eta) : \Omega_\eta \mapsto \mathbb{R}$ denote a continuously differentiable real-valued function of interest. Define the influence function*

$$\Psi_p(\theta) := -\left.\frac{dg(\eta)}{d\eta^T}\right|_{\hat{\eta}} \hat{H}^{-1}\overline{\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\hat{\eta}\right)}\ \mathcal{Q}(\theta|\hat{\eta}). \tag{3.14}$$

*Let Assumption 1 be satisfied at $t_0 = 0$ with $\psi(\theta, t) = \log \mathcal{P}(\theta|t)$ as given in Equation 3.13. Then the map $t \mapsto g(\hat{\eta}(t))$ is continuously differentiable at $t = 0$ with derivative*

$$\left.\frac{dg(\hat{\eta}(t\phi))}{dt}\right|_0 = \int \Psi(\theta)\phi(\theta)\mu(d\theta). \tag{3.15}$$

*Proof.* This follows immediately from Theorem 1, noting that

$$\log \mathcal{P}(\theta|t) - \log \mathcal{P}_0(\theta) = t\phi(\theta) + C,$$

where $C$ is independent of $t$; thus,

$$\left.\frac{\partial \log \mathcal{P}(\theta|t)}{\partial t}\right|_{t=0} = \phi(\theta).$$

Substituting into Equation 3.11 gives the desired form.

□

Corollary 2 suggests an intriguing result if it is taken to hold for all $\phi$ in some ball of radius $\delta$ around the null perturbation $\phi(\theta) := 0$. Consider for example, the $L_\infty$ ball of size $\delta$,

$$\begin{aligned}\mathcal{B}_\infty(\delta) &:= \{\phi : \|\phi\|_\infty \leq \delta\},\\ &\text{where } \|\phi\| := \sup_\theta |\phi(\theta)|.\end{aligned}$$

Then observe that, by Hölder's inequality (Dudley, 2018, Theorem 5.1.2 and subsequent disscussion),

$$\begin{aligned}\sup_{\phi \in \mathcal{B}_\infty(\delta)} \left.\frac{dg(\hat{\eta}(t\phi))}{dt}\right|_0 &= \sup_{\phi \in \mathcal{B}_\infty(\delta)} \int \Psi(\theta)\phi(\theta)\lambda(d\theta)\\ &= \delta \int |\Psi(\theta)|\, \lambda(d\theta)\end{aligned}$$

with equality at the "worst-case" perturbation

$$\phi^*(\theta) = \delta \times \text{sign}(\Psi(\theta)).$$

For the most negative "worst-case", simply apply the preceding result to $-g$. These "worst-case" perturbations are the variational Bayes analogues of the corresponding "worst-case" for exact Bayesian posteriors in Gustafson (1996).

The use of the $L_\infty$ ball is motivated by the fact that the finiteness of $\phi$ in $L_\infty$-norm implies that the perturbed prior Equation 3.13 is normalizable (Gustafson, 1996). More fundamentally, Giordano (2019), Chapter 7.2 argues that among all possible nonlinear perturbations considered by Gustafson (1996), the mapping from the space of perturbations $\phi$ to VB parameters $\hat{\eta}$ is Fréchet differentiable only when considering multiplicative perturbations and the $L_\infty$ norm. Fréchet differentiability is important because in Section 3.6, we will use the influence function to search for potentially influential perturbations, and Fréchet differentiability ensures that for all $\phi \in \mathcal{B}_\infty(\delta)$, the local approximation is uniformly good. This is a minimal requirement that allows us to safely search the space of functional perturbations.

## 3.5 Computing the sensitivity

Before computing any sensitivity measures, we first need to find $\hat{\eta}(0)$ by optimizing the KL objective (Equation 3.3). In the optimization, as well as the Hessian inversion discussed below, we take advantage of the global/local structure in our models (Section 3.2). Recall that we partitioned the variational parameter vector $\eta$ into global variational parameters $\eta_\gamma$ and local variational parameters $\eta_\ell$. The global variational parameters govern the distribution on latent variables shared across multiple data points, such as stick-breaking proportions $\nu$ and component variables $\beta$; the local variational parameters govern the collection of latent variables unique to each data point, such as the cluster assignments $z$.

In Section 3.2, we noted that the optimal local variational parameters $\hat{\eta}_\ell$ can be written as a closed-form function of the global variational parameters $\eta_\gamma$. Let $\hat{\eta}_\ell(\eta_\gamma; t)$ denote this mapping; that is,

$$\hat{\eta}_\ell(\eta_\gamma; t) := \underset{\eta_\ell}{\text{argmin}}\, \text{KL}\left((\eta_\gamma, \eta_\ell), t\right).$$

With this minimizer available in closed-form, we can define

$$\text{KL}_{\text{glob}}(\eta_\gamma, t) := \text{KL}\Big((\eta_\gamma, \hat{\eta}_\ell(\eta_\gamma; t)), t\Big), \tag{3.16}$$

which returns the KL as a function of only global parameters by implicitly setting local parameters at their optimum.

Rather than optimizing the KL over all variational parameters, both global and local, we optimize $\text{KL}_{\text{glob}}$, which is a function of global parameters only. Minimizing $\text{KL}_{\text{glob}}(\eta_\gamma)$ keeps the dimension of the optimization parameter independent of the number of data points.

After the optimization terminates at an optimal $\hat{\eta}_\gamma$, the optimal local parameters $\hat{\eta}_\ell$ can be set in closed form to produce the entire vector of optimal variational parameters $\hat{\eta} = (\hat{\eta}_\gamma, \hat{\eta}_\ell)$. The construction of $\mathrm{KL}_{\mathrm{glob}}$ will play a role in computing our sensitivity measures in practice, as we detail below.

### The Hessian

Typically, it is the computation and inversion of the Hessian matrix that is the most computationally intensive part of Equation 3.11, especially when the dimension of $\eta$, is large. The dimension of $\eta_\gamma$ scales with $K_{\max}$, while the dimension of $\eta_\ell$ scales with $K_{\max} \times N$. (the scaling of the topic model is slightly different; but the point remains that the dimension of $\eta_\ell$ grows faster than the dimension of $\eta_\gamma$). Depending on the size of $\eta$, instantiating the full Hessian in memory may be impossible.

We again take advantage of the fact that the latent variables factorize into a set of global and local latent variables. For generic latent variables $a$ and $b$, let $H_{ab}$ denote $\partial^2 \mathrm{KL}\left(\eta, t\right) / \partial \eta_a \eta_b^T \big|_{\hat{\eta}(0), t=0}$, the Hessian with respect to the variational parameters governing $a$ and $b$. We decompose the Hessian matrix $\hat{H}$ into four blocks:

$$\hat{H} = \left.\frac{\partial^2 \mathrm{KL}\left(\eta, t\right)}{\partial \eta \partial \eta^T}\right|_{\hat{\eta}(0), t=0} = \begin{pmatrix} H_{\gamma\gamma} & H_{\gamma\ell} \\ H_{\ell\gamma} & H_{\ell\ell} \end{pmatrix}.$$

Next, let $\hat{J}_\gamma$ be the components of $\hat{J}$ corresponding to the variational parameters $\eta_\gamma$—that is, $\hat{J}_\gamma$ is given by replacing the operator $\nabla_\eta$ with $\nabla_{\eta_\gamma}$ in Equation 3.12. The analogous quantity for the local parameters, $\hat{J}_\ell$, is zero, since no local variables enter the expectation in Equation 3.12. We can thus write

$$\hat{J} = \begin{pmatrix} \hat{J}_\gamma \\ 0 \end{pmatrix}.$$

In this notation,

$$\left.\frac{d\hat{\eta}(t)}{dt}\right|_{t=0} = -\begin{pmatrix} H_{\gamma\gamma} & H_{\gamma\ell} \\ H_{\ell\gamma} & H_{\ell\ell} \end{pmatrix}^{-1} \begin{pmatrix} \hat{J}_\gamma \\ 0 \end{pmatrix},$$

and an application of the Schur complement gives

$$\left.\frac{d\hat{\eta}(t)}{dt}\right|_{t=0} = -\begin{pmatrix} I_{\gamma\gamma} \\ H_{\ell\ell}^{-1} H_{\ell\gamma} \end{pmatrix} \left(H_{\gamma\gamma} - H_{\gamma\ell} H_{\ell\ell}^{-1} H_{\ell\gamma}\right)^{-1} \hat{J}_\gamma,$$

where $I_{\gamma\gamma}$ is the identity matrix with the same dimension as $\eta_\gamma$.

Specifically, observe that the sensitivity of the global parameters is given by

$$\left.\frac{d\hat{\eta}_\gamma(t)}{dt}\right|_{t=0} = -\hat{H}_\gamma^{-1} \hat{J}_\gamma \quad \text{where} \quad \hat{H}_\gamma := \left(H_{\gamma\gamma} - H_{\gamma\ell} H_{\ell\ell}^{-1} H_{\ell\gamma}\right). \tag{3.17}$$

Each term of $\hat{H}_\gamma$ can be easily computed using automatic differentiation. In fact, the $H_{\ell\ell}$ is block-diagonal and has a closed form inverse.

Alternatively, using the optimality of $\hat{\eta}_z(\eta_\gamma; t)$ and applying the chain rule, one can check that

$$\hat{H}_\gamma = \frac{\partial^2}{\partial \eta_\gamma \partial \eta_\gamma^T} \mathrm{KL}_{\mathrm{glob}}(\hat{\eta}_\gamma, 0) \tag{3.18}$$

Equation 3.18 is how we calculate $\hat{H}_\gamma$ in practice: we implement $\mathrm{KL}_{glob}$, which returns the KL as a function of only global parameters by implicitly setting local parameters at their optimum; we then use automatic differentiation to compute the Hessian of $\mathrm{KL}_{glob}$ with respect to the global parameters $\eta_\gamma$.

Crucially, the size of $\hat{H}_\gamma$ scales with $(K_{\max})^2$, while the full Hessian scales with both $(K_{\max})^2$ and the squared number of data points $N^2$ (in the topic model, the Hessian scale as $(NK_{\max})^2$ and $(NK_{\max}L)^2)$). Hence, $\hat{H}_\gamma$ is more memory efficient to store and faster to invert than the full Hessian of all the variational parameters.

Now that we can compute the sensitivity of the global parameters in Equation 3.17, we produce its linear approximation:

$$\hat{\eta}_\gamma^{\mathrm{lin}}(t) := \hat{\eta}_\gamma + \left.\frac{d\hat{\eta}_\gamma(t)}{dt}\right|_{t=0} t. \tag{3.19}$$

Finally, given a posterior quantity $g$, we again take advantage of the fact that the optimal local parameters can be found in closed form given global parameters. In the same way that $\mathrm{KL}_{\mathrm{glob}}$ implicitly sets the local parameters at their optimum and is a function of only global parameters and the prior parameter $t$, we can construct an analogous mapping for $g$,

$$(t, \eta_\gamma) \mapsto g\Big(\big(\eta_\gamma, \hat{\eta}_z(\eta_\gamma, t)\big)\Big). \tag{3.20}$$

This mapping can be used for any posterior quantity. Therefore, linearizing the global parameters using Equations 3.17 and 3.19 is sufficient; we do not need to invert the full Hessian and linearize the entire set of variational parameters, global and local. As a shorthand, we will use $g(\hat{\eta}_\gamma^{\mathrm{lin}}(t))$ to denote the posterior computed under our linearized global parameters at prior parameter $t$ using the mapping Equation 3.20.

## 3.6 Results

We evaluate the prior sensitivity in BNP models applied to three distinct data analysis examples. We first fit a Gaussian mixture model (Example 1) to the canonical iris data set. Secondly, we cluster time-course gene expression data using our regression model (Example 2) and study the resulting co-clustering matrix. Finally, we fit our topic model (Example 3) on a data set of sampled genotypes in an endangered thrush species; from our model fit,

we estimate the number of latent populations in this bird species and reconstruct ancestral migration patterns.

In each of our models, we consider both the effects of varying the parameter $\alpha$ in the $\text{Beta}(\nu_k|1,\alpha)$ stick distribution, as well as the effects of changing the functional form the Beta prior itself. For a given prior perturbation, we validate the performace of the linear approximation against the more expensive process of re-fitting the model.

In each data example, the initial variational parameters were fit to a model with Beta-distributed sticks at some chosen parameter $\alpha = \alpha_0$. We optimized the initial variational parameters with the BFGS algorithm run with a loose convergence tolerance followed by trust-region Newton conjugate gradient (trust-ncg) to find a high-quality optimum. Unless otherwise noted, all subsequent refits after a prior pertrubation were found using trust-ncg with the variational parameters at $\alpha = \alpha_0$ as an initialization.

In our results below, we use the approximation defined by Equations 3.17 and 3.19, where only the global variational parameters are linearized; the local parameters are set implicitly as part of the computation of the posterior statistic $g$. We solved the linear system in Equation 3.17 using the conjugate gradient algorithm, which requires only Hessian-vector products; this avoids instatiating the full Hessian matrix in memory. If memory were not limited, we could have computed the full Hessian and either factorized it (e.g. with a Cholesky decomposition) or found its inverse directly. Then, the Hessian inverse (or its Cholesky decomposition) can be re-used for different perturbations (since different choices of prior perturbation require solving different $\hat{J}$). By using conjugate gradient, we need to re-solve the linear system (Equation 3.17) for each choice of prior perturbation. On our data sets, the conjugate gradient algorithm was at least an order of magnitude faster than refitting, and did not pose a meaningful bottleneck to exploring different perturbations.

The conjugate gradient algorithm along with the optimizers BFGS and trust-ncg are implemented in SciPy (Virtanen et al., 2020). All derivatives were computed using the Python automatic differentiation libarary Jax (Bradbury et al., 2018).

### 3.6.1 Gaussian mixture modeling on iris data

We demonstrate the local sensitivity computations on a Gaussian mixture model (GMM) of Fisher's iris data set. The data set is a sample of 150 iris flowers with four measurements taken per flower: sepal length, sepal width, petal length, and petal width. The GMM was detailed in Example 1; here, the data dimension $d = 4$. In the variational approximation, we set the truncation parameter $K_{\max} = 15$. Figure 3.1 shows the inferred clusters at $\alpha = 6$. The data set contains three iris species, and the BNP model correspondingly identifies three dominant clusters.

#### Parametric sensitivity

We evaluate the sensitivity of the posterior number of clusters to the prior parameter $\alpha$. The expected in-sample number of clusters $g_{\text{cl},\tau}$ and the expected predictive number of clusters

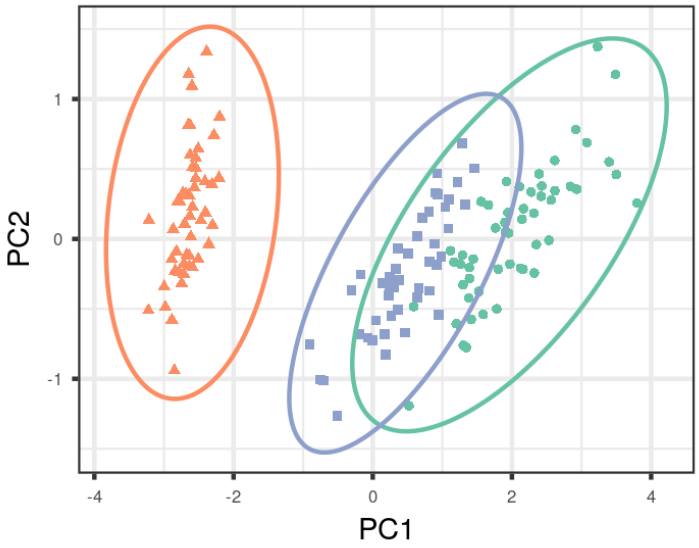



Figure 3.1: The iris data in principal component space and GMM fit at $\alpha = 6$. Colors denote inferred memberships and ellipses represent estimated covariances.

$g_{\text{p.cl},\tau}$ were defined in Examples 6 and 7, respectively; we do not use thresholding in computing the number of clusters (i.e. we set $\tau = 0$), so we simply write $g_{\text{cl}}$ and $g_{\text{p.cl}}$ for the in-sample and predictive quantities, respectively. We formed the linear approximation at the initial $\alpha_0 = 6$ fit. Without further optimization, we use the linear approximation (Equations 3.17 and 3.19) to quickly evaluate $\hat{\eta}_\gamma^{\text{lin}}(\alpha)$ for all $\alpha = 1, \ldots, 16$, and produce posterior quantities $g(\hat{\eta}_\gamma^{\text{lin}}(\alpha))$.

The in-sample quantity appears less sensitive to changes in the $\alpha$ parameter than the predictive quantity (Figure 3.2). As $\alpha$ varies from $\alpha = 1, \ldots, 16$, the quantity $g_{\text{cl}}(\hat{\eta}_\gamma^{\text{lin}}(\alpha))$ varies only from 3.0 to 3.4 (recall that the true number of iris species is three). On the other hand, over the same range of $\alpha$, the predictive quantity $g_{\text{p.cl}}(\hat{\eta}_\gamma^{\text{lin}}(\alpha))$ varies from 3.6 to 8.1.

Subsequent refits at $\alpha \neq \alpha_0$ confirm the sensitivity conclusions of our linearized variational parameters. Over the range $\alpha = 1, \ldots, 16$, the values $g(\hat{\eta}_\gamma^{\text{lin}}(\alpha))$ closely mimic the values $g(\hat{\eta}(\alpha))$ found by refitting the variational parameters at each $\alpha$ (Figure 3.2). Importantly, the linear approximation is an order of magnitude faster than refitting. Forming the linear approximation at $\alpha = \alpha_0$, required 0.02 seconds. After forming the linear approximation, computing $\hat{\eta}_\gamma^{\text{lin}}(\alpha)$ for all $\alpha = 1, \ldots, 16$ took another 0.02 seconds. On the other hand, to refit $\hat{\eta}(\alpha)$ for the same range of $\alpha$'s took a total of 9 seconds, with a median refit time of 0.6 seconds.

### Functional perturbations and the influence function

We demonstrate the ability of the influence function to provide guidance on the anticipated effect of potential perturbations on the expected number of in-sample clusters. Recall from

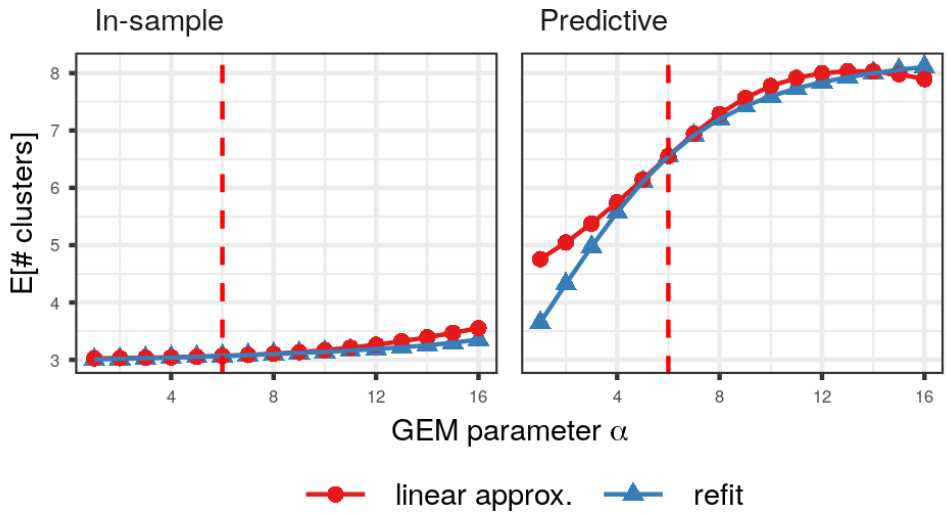



Figure 3.2: The expected number of clusters as $\alpha$ varies in the the GMM fit of the iris data. On the left is in-sample quantity $g_{\text{cl}}$. On the right is the the predictive quantity $g_{\text{p.cl}}$. We formed the linear approximation at $\alpha = 6$. In red, the expected number of clusters computed under the linearly approximated variational parameters (red). In blue, the expected number of clusters obtained by refitting the model at each $\alpha$.

Equation 3.15 that the local sensitivity can be represented as an inner product between the influence function $\Psi$ and a multiplicative perturbation $\phi$; the inner product takes the form,

$$\left.\frac{dg(\hat{\eta}(t\phi))}{dt}\right|_0 = \int \Psi_p(\tilde{\nu}_k)\phi(\tilde{\nu}_k)\mu(d\tilde{\nu}_k), \tag{3.21}$$

where we expressed both the influence function and the perturbation in logit-stick space, $\tilde{\nu}_k := \log(\nu_k) - \log(1 - \nu_k)$.

In order to illustrate this inner product, we consider functional perturbations $\phi_\mu$ which are Gaussian bumps, with each perturbation centered at a different $\mu$ on the real line (Figure 3.3 left column). The perturbed priors are $p(\tilde{\nu}_k|t) = p_0(\tilde{\nu}_k)\exp(t\phi_\mu(\tilde{\nu}_k))$. The middle column of Figure 3.3 displays the initial density $p_0(\nu_k) = \text{Beta}\,(\nu_k|1, \alpha_0)$ along with the perturbed densities $p(\nu_k|t = 1)$ for different choices of $\mu$, in the original $(0, 1)$ constrained space.

Each perturbation $\phi_\mu$ with a different choice of $\mu$ produces distinct changes in the expected number of in-sample clusters $g_{\text{cl}}$. The right column of Figure 3.3 plots the differences $\Delta g_{\text{cl}}(\eta(t)) := g_{\text{cl}}(\eta(t)) - g_{\text{cl}}(\hat{\eta}(0))$ for $t \in [0, 1]$. Depending on the perturbation, $\Delta g_{\text{cl}}$ can be positive, negative, or nearly zero. In each case, the approximation $\Delta g_{\text{cl}}(\hat{\eta}^{\text{lin}}_\gamma(t))$ is able to mirror the qualitative behavior of $\Delta g_{\text{cl}}(\hat{\eta}(t))$ observed by refitting.

While each perturbation produces distinct changes in $g_{\text{cl}}$, it is unclear how to anticipate the effect of a perturbation by comparing the original and perturbed densities alone. On the other hand, the sign and magnitude of the change in $g_{\text{cl}}$ after prior perturbation are

well-explained by its influence function, plotted in purple in the left column of Figure 3.3, and the representation of local sensitivity as an inner product (Equation 3.21). When $\phi_\mu$ is centered at a location where the influence function is negative, the effect of the perturbation on $g_{\text{cl}}$ is negative (Figure 3.3 top row); conversely, when $\phi_\mu$ is centered at a location where the influence function is positive, its effect on $g_{\text{cl}}$ is positive (bottom row); finally, when $\phi_\mu$ is centered at a location where the influence transitions from negative to positive, its effect on $g_{\text{cl}}$ is roughly zero (middle row). In the last case, $\phi_\mu$ placed approximately equal weight on the negative and positive portions of the prior-weighted influence function, resulting in an approximately zero inner-product. In the other data applications below, the influence function will guide our choice of functional perturbation and explain why some perturbations result in greater sensitivity than others.

Finally, we consider the worst-case multiplicative perturbation with unit $L_\infty$ norm. Recall that the worst-case perturbation with unit $L_\infty$ norm is a step-function taking on values $\pm 1$ corresponding to the sign of the influence function (Figure 3.4 left). The middle column of Figure 3.4 shows the prior density perturbed by the worst-case perturbation; the right column shows the effect on $g_{\text{cl}}$. This worst-case perturbation has a much larger effect on $g_{\text{cl}}$ compared to the other unit $L_\infty$ norm perturbations in Figure 3.3. However, even with the worst-case perturbation, the change in $g_{\text{cl}}$ is still small. We conclude that on the iris data set, $g_{\text{cl}}$ appears to be a quantity insensitive to the prior under a Gaussian mixture model.

Computing the linearized variational parameters $\hat{\eta}_\gamma^{\text{lin}}(t)$ at $t = 1$, including the necessary Hessian solve, for a given functional perturbation required 0.02 seconds. A refit at $t = 1$ requires about one second. While a second for a refit is not exceedingly large, the order-of-magnitude difference in timing between the linear approximation and refit will continue to hold for larger data analysis problems below. In general, the speed of the linear approximation allows us to quickly explore many different potential functional perturbations when refitting for each perturbation becomes prohibitive.

### 3.6.2 Regression mixture modeling

We consider the problem of clustering time-course gene expression data. While thousands of genes might be simultaneously measured in a given genomics experiment, many genes may exhibit similar expression patterns. Clustering gene expressions is one way to reduce the dimensionality of a complex data set and to facilitate scientific interpretations of intricate biological processes. Often, such dimensionality reduction is used for exploratory analysis and is a first step before further downstream investigation. It is important, therefore, to ascertain the stability of the discovered clusters.

We study a publicly available data set of mice gene expression (Shoemaker et al., 2015). Mice were infected with influenza virus, and expression levels of a set of genes were assessed at 14 time points after infection. Three measurements were taken at each time point (called biological replicates), for a total of $M = 42$ measurements per gene.

Our analysis focuses on mice treated with the "A/California/04/2009" strain. We normalize the data as described in Shoemaker et al. (2015) and then apply the differential

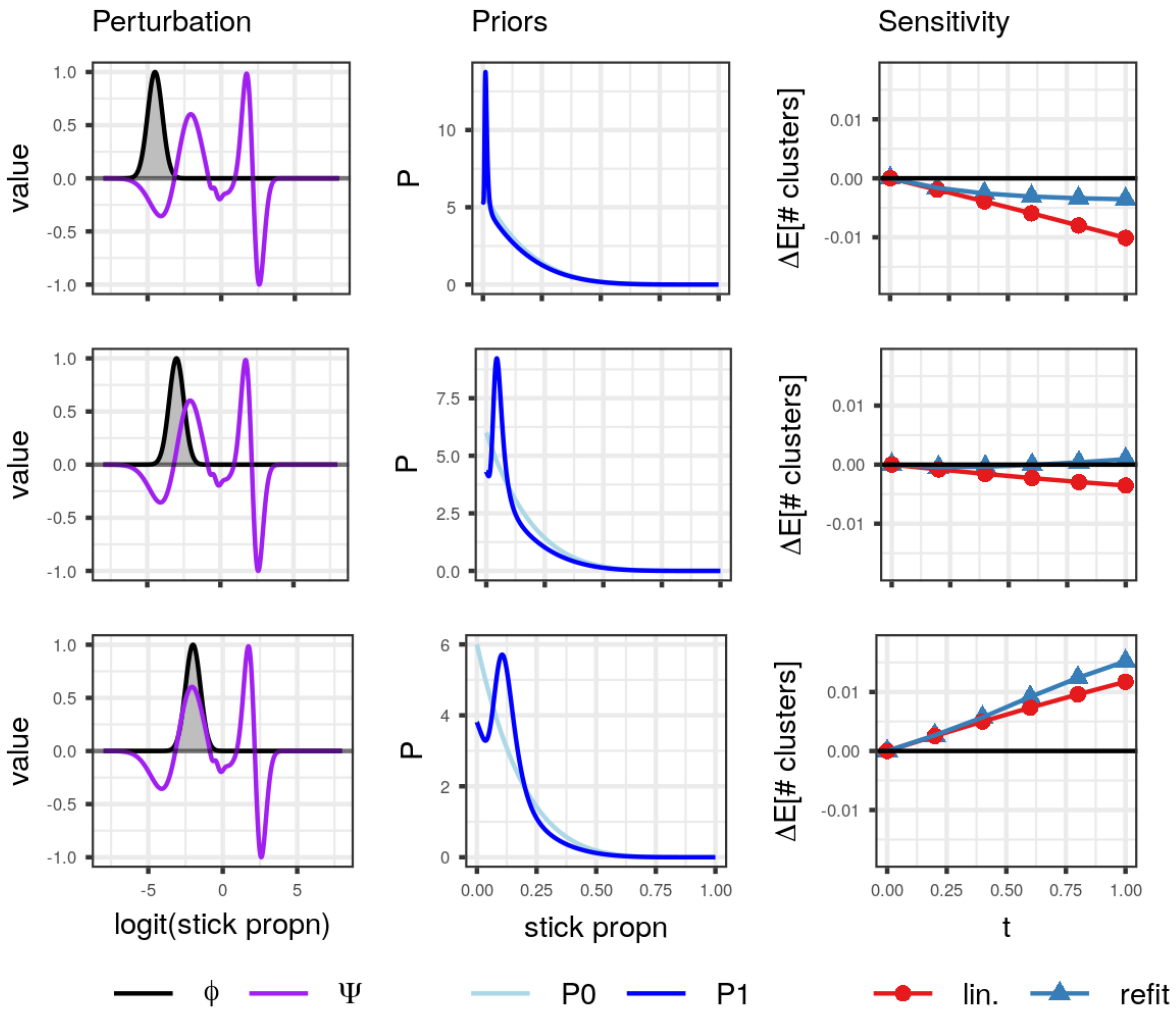



Figure 3.3: Sensitivity of the expected number of in-sample clusters in the iris data set to three multiplicative perturbations with unit $L_\infty$-norm. (Left) the multiplicative perturbation $\phi$ in grey. The influence function $\Psi$ in purple, scaled to also have unit $L_\infty$-norm. (Middle) the original prior density $\mathcal{P}_0$ and the perturbed prior density $\mathcal{P}_t = \mathcal{P}_0 \times \exp(t\phi)$ at $t = 1$. (Right) the effect of the perturbation on the change in expected number of in-sample clusters as $t \to 1$.

analysis tool EDGE (Storey et al., 2005) to rank the genes from most to least significantly differentially expressed. We fit the regression model described in Example 2 and run our analysis below on the top $N = 1000$ genes.

Figure 3.5 shows an example time-course for a single gene. Notice that the time points are unevenly spaced, with more frequent observations at the beginning. Following Luan and Li (2003) we apply cubic B-splines to smooth the time course expression data. Specifically,

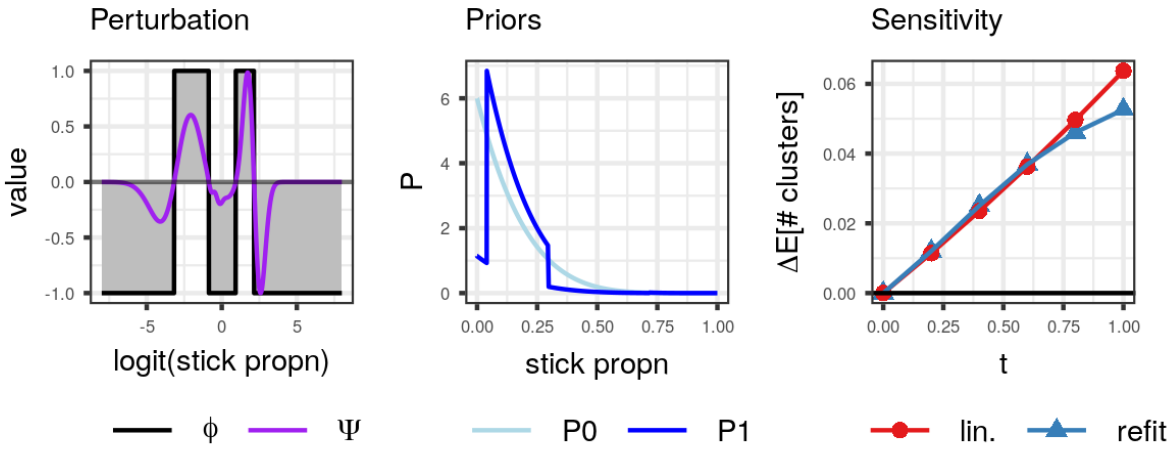



Figure 3.4: Sensitivity of the expected number of in-sample clusters in the iris data set to the worst-case multiplicative perturbation with unit $L_\infty$-norm.

we model the first 11 time points using cubic B-splines with 7 degrees of freedom. For the last three time points, $T = 72, 120, 168$ hours, we use indicator functions. That is, if $\tilde{A}$ is the design matrix where each column is a B-spline basis vector evaluated at the $M$ measurement times, we append to $\tilde{A}$ three additional columns: in these columns, entries are 1 if $T = 72, 120$, or 168, receptively, and 0 otherwise. The resulting matrix is the full design matrix $A$. We use indicators for the last three time points for numerical stability; without the indicator columns, the matrix $\tilde{A}^T\tilde{A}$ is nearly singular because the later time points are more spread out.

We fitted the initial approximate posterior at $\alpha_0 = 6$. Figure 3.6 shows the inferred smoothers $A\mathbb{E}_{\mathcal{Q}}[\beta_k]$ for selected clusters. Figure 3.7 displays the inferred co-clustering matrix $g_{\text{cc}}(\eta)$, whose $(i, j)$-th entry is the posterior probability that gene $i$ belongs to the same cluster as gene $j$ (Example 8).

Below, we evaluate the sensitivity of the inferred co-clustering matrix to both parametric and functional perturbations to the stick distribution.

### Parametric sensitivity

We first evaluate the sensitivity of the co-clustering matrix $g_{\text{cc}}$ to the choice of $\alpha$ in the Beta $(\nu_k|1, \alpha)$ stick-breaking distribution. Let $g^0_{\text{cc}} := g_{\text{cc}}(\hat{\eta}(\alpha_0))$ be the co-clustering matrix inferred at $\alpha_0$, and let $\Delta g_{\text{cc}}(\eta) := g_{\text{cc}}(\eta) - g^0_{\text{cc}}$. We formed the linear approximation at $\alpha_0$ and computed $\Delta g_{\text{cc}}(\hat{\eta}^{\text{lin}}_\gamma(\alpha))$, the the change in co-clustering under the linearized variational parameters, at $\alpha = 1$ and $\alpha = 11$. For either $\alpha$, the change in co-clustering matrix is minuscule (Figure 3.8): the largest entry of either matrix $\Delta g_{\text{cc}}(\hat{\eta}^{\text{lin}}_\gamma(1))$ or $\Delta g_{\text{cc}}(\hat{\eta}^{\text{lin}}_\gamma(11))$ is of order $10^{-2}$. Refitting the approximate posterior at $\alpha = 1$ and $\alpha = 11$ and computing $\Delta g_{\text{cc}}(\hat{\eta}(\alpha))$

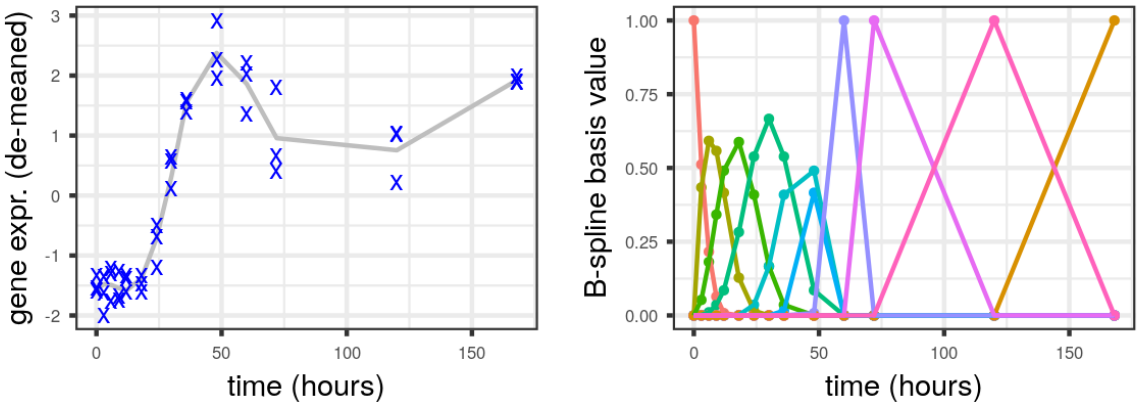



Figure 3.5: (Left) An example gene and its expression measured at 14 unique time points with three biological replicates at each time point. (Right) The cubic B-spline basis with 7 degrees of freedom, along with three indicator functions for the last three time points, $T = 72, 120, 168$.

confirms the insensitivity predicted by the linearized variational global parameters. Beyond capturing insensitivity, the linearized parameters were also able to approximate the sign and size of the changes in the individual entries of the coclustering matrix (these changes albeit small).

### Functional sensitivity

Insensitivity to $\alpha$ does not necessarily rule out insensitivity to other prior perturbations, however. As demonstrated in Section 3.6.1, the influence function can provide guidance on which functional perturbations may result in greater sensitivity for a chosen posterior quantity. However, the co-clustering matrix as a posterior quantity is $N^2$-dimensional and thus does not lend itself to an easily interpretable influence function. We therefore summarize the co-clustering matrix into a scalar quantity: we use the sum of the eigenvalues of the symmetrically normalized graph Laplacian. This quantity has close connection with the number of distinct components in a graph (von Luxburg, 2007). Let this posterior quantity be denoted $g_{\text{ev}}$, given by

$$g_{\text{ev}}(\eta) = \text{Tr}\left(I - D(\eta)^{-1/2} g_{\text{cc}}(\eta) D(\eta)^{-1/2}\right),$$

where $D(\eta)^{-1/2}$ is the diagonal matrix with entries $d_i = \sum_{j=1}^{N} [g_{\text{cc}}(\eta)]_{ij}$. (And recall that the trace of a matrix is equivalent to the sum of its eigenvalues).

Because $g_{\text{ev}}(\eta)$ is a scalar quantity, we can plot its influence function. We choose a functional perturbation $\phi_{\text{ev}}$ that has a large, positive inner-product with the influence function. In this case, we construct $\phi_{\text{ev}}$ using two Gaussian bumps aligned with the two largest modes

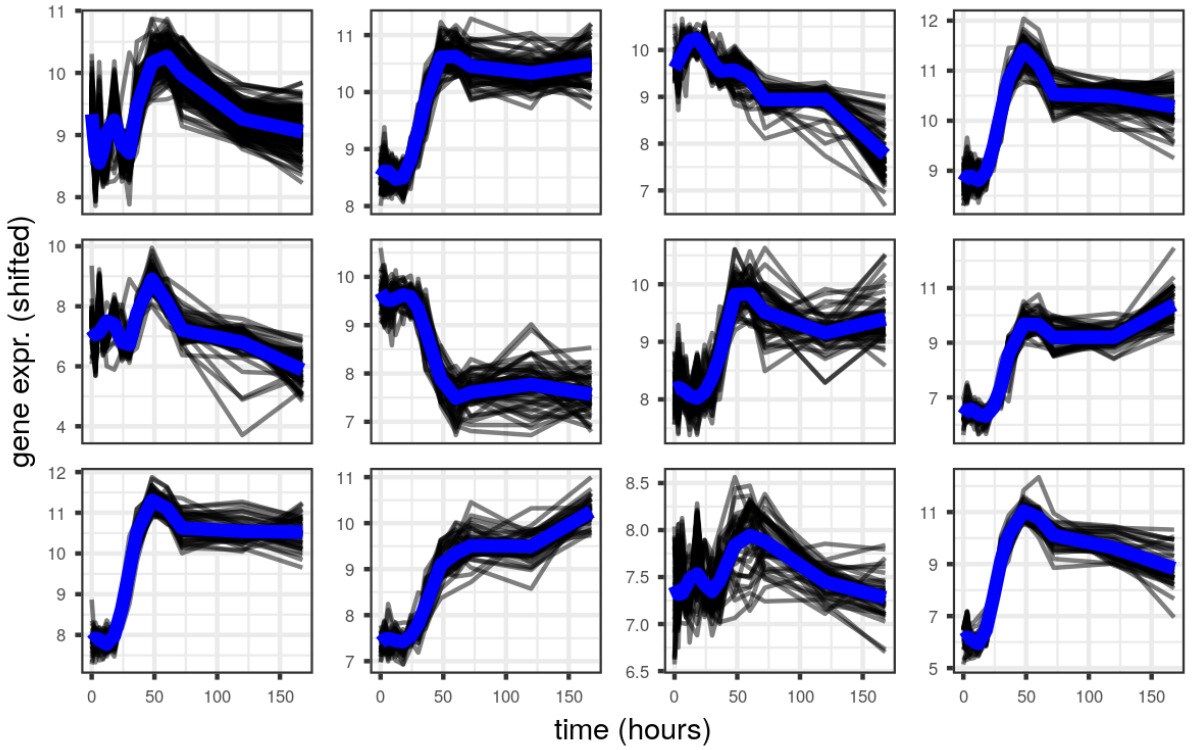



Figure 3.6: Inferred clusters in the mice gene expression dataset. Shown are the twelve most occupied clusters. In blue, the inferred cluster centroid. In grey, gene expressions averaged over replicates and shifted by their inferred intercepts.

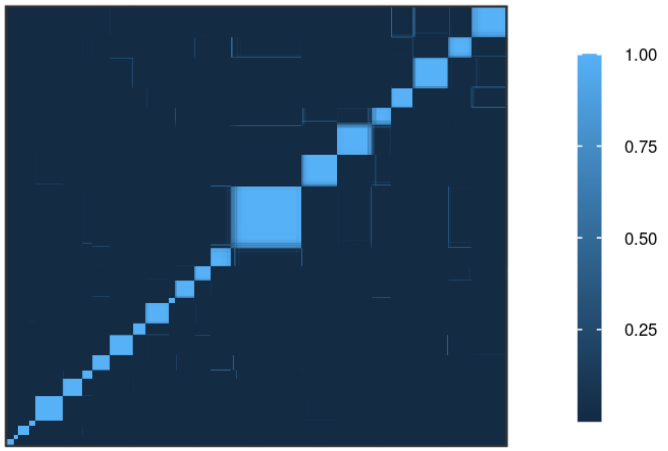



Figure 3.7: The inferred co-clustering matrix of gene expressions at $\alpha_0 = 6$.

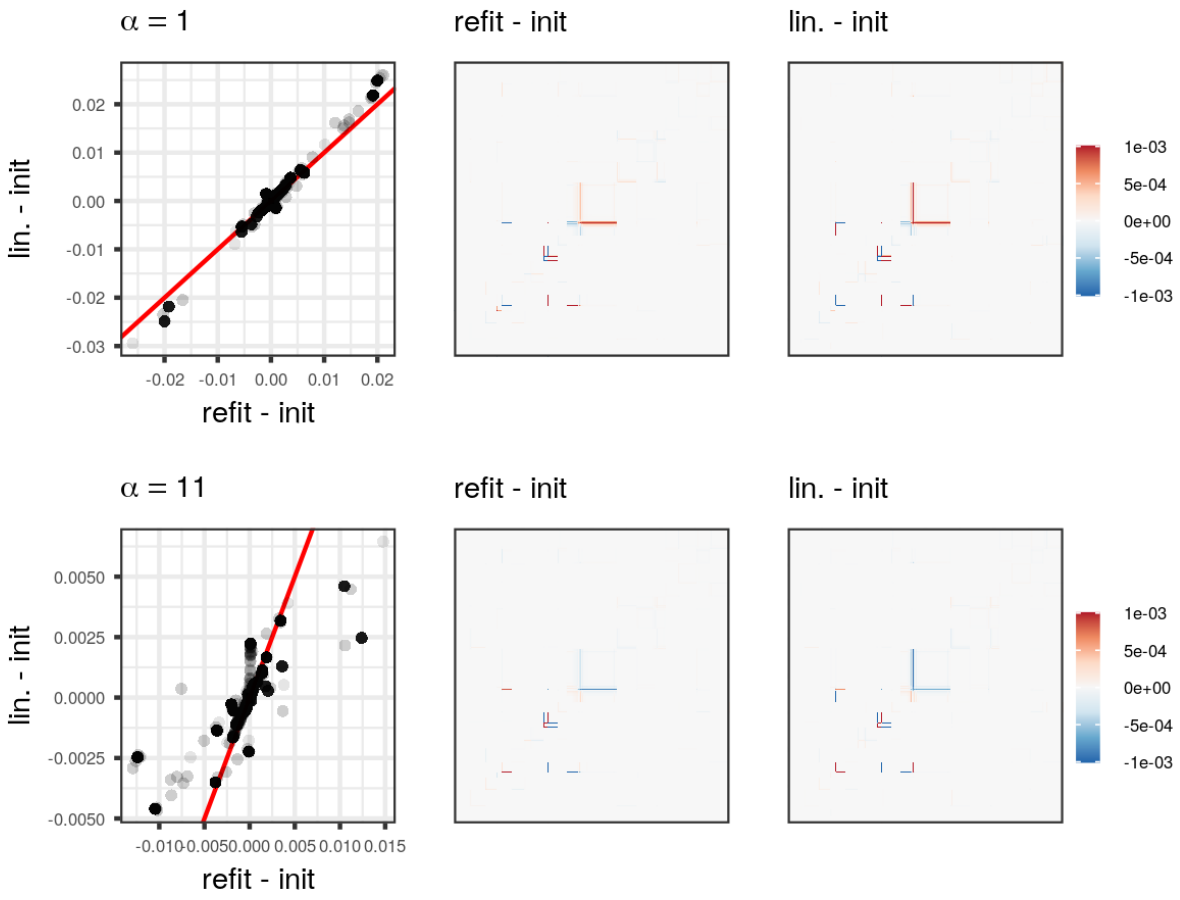



Figure 3.8: Differences in the co-clustering matrix at $\alpha = 1$ (top row) and $\alpha = 11$ (bottom row), relative to the co-clustering matrix at $\alpha_0 = 6$. We compare differences obtained with the linearly approximated variational parameters against changes observed after refiting. (Left) a scatter plot of differences under the linear approximation against differences after refitting, where each point represents an entry of the co-coclustering matrix. (Middle) the difference in co-clustering matrix observed after refitting. (Right) the difference observed under the linearly approximated variational parameters. For visualization, values in the heatmaps are clipped at $\pm 10^{-3}$.

of the prior-weighted influence function (Figure 3.9 top left). We anticipate $\phi_{\text{ev}}$ to have a large effect on $g_{\text{ev}}$. With $g_{\text{ev}}$ a proxy for our actual posterior quantity of interest, the full co-clustering matrix, we then expect that the co-clustering matrix will also experience large changes.

Our intuition is confirmed in Figure 3.9. After perturbing by $\phi_{\text{ev}}$, the largest changes in the co-clustering matrix are of now of order $10^{-1}$, compared with changes on the order

of $10^{-2}$ after the $\alpha$ perturbations. The linearized variational parameters are again able to capture the qualitative changes in the co-clustering matrix after refitting at the perturbed prior.

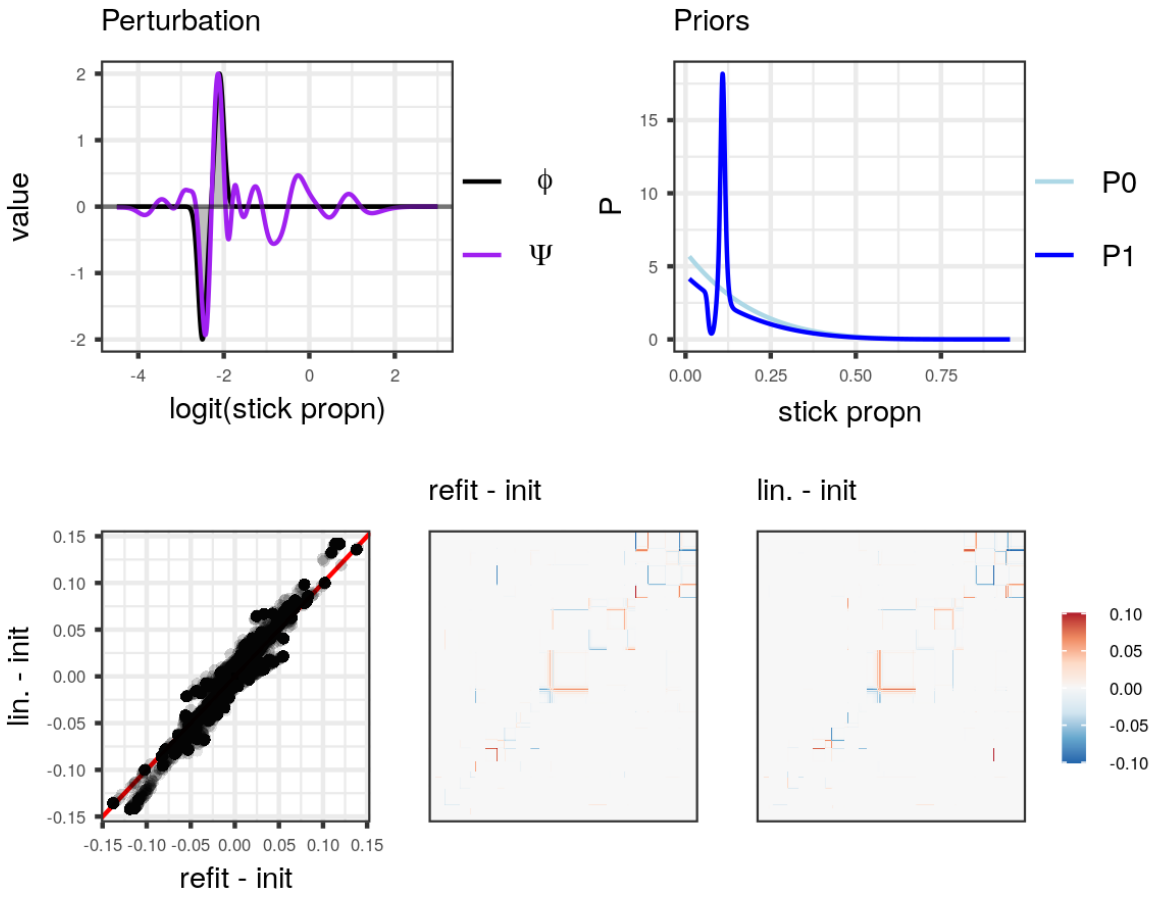



Figure 3.9: Effect on the co-clustering matrix after a multiplicative functional perturbation. The perturbation $\phi$ (top left, in grey) is a difference of two Gaussian bumps scaled to have $L_\infty$ norm equal to two. $\phi$ is chosen such that the Gaussian bumps roughly align with the two largest modes of the influence function (top left, purple; the influence function is scaled to also have $L_\infty$ norm equal to two). The effect of this perturbation on the prior density in the top right. The bottom row shows the effect of this perturbation on the coclustering matrix. For visualization, the differences in the heatmap are clipped at $\pm 10^{-1}$.

The influence function is able to explain why the co-clustering matrix is insensitive to $\alpha$. The functional perturbation that corresponds to a change in $\alpha$ is

$$\phi_\alpha(\nu_k) := \log \text{Beta}\left(\nu_k | 1, \alpha\right) - \log \text{Beta}\left(\nu_k | 1, \alpha_0\right).$$

The function $\phi_\alpha(\nu_k)$ is large when the influence function is small and vice-versa (Figure 3.10), resulting in a small inner-product between the influence function and $\phi_\alpha$. Thus, the linear approximation will predict small changes, and the refitted results confirms the predictions.

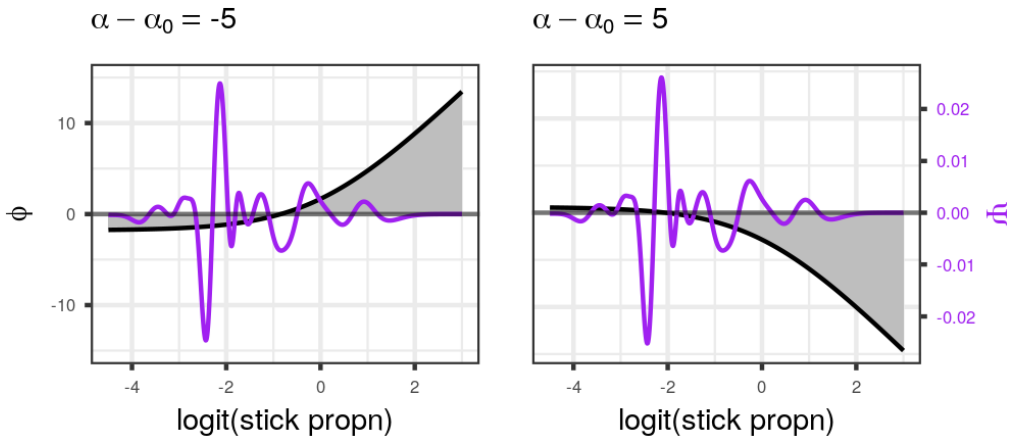



Figure 3.10: The multiplicative perturbations $\phi_\alpha(\cdot)$ that corresponds to decreasing (left) or increasing (right) the $\alpha$ parameter by five.

However, even with the selected functional perturbation, the size of the differences in the co-clustering matrix remains modest. It is unlikely that any conclusions derived from the co-clustering matrix would have changed after the functional perturbation. The co-clustering matrix appears insensitive to perturbations in the stick-breaking distribution.

Finally, the computational cost of linearizing the variational parameters is favorable compared with refitting (Table 3.1). Forming the linear approximation, which requires a Hessian inversion, took 3-4 seconds; subsequent evaluations of $\hat{\eta}_\gamma^{\text{lin}}$ take milliseconds. Conversely, refitting the model after a prior perturbation can take up to 20 seconds.

### 3.6.3 Genetic admixture modeling with STRUCTURE

Our final data analysis example is an application of population genetics. Given a database of individuals and their genotypes at selected genetic loci, population geneticists seek to identify the presence of latent populations, infer the population of origin for specific loci, and estimate the degree to which populations are admixed in each individual.

We consider a publicly available dataset from Galbusera et al. (2000) that contains genotypes from 155 samples of an endangered bird species, the Taita thrush. Individuals were collected from four regions in southeast Kenya (Chawia, Mbololo, Ngangao, Yale), and each individual was genotyped at seven micro-satellite loci. The four regions were once part of a cohesive cloud forest that has since been fragmented by human development. For this endangered bird species, understanding the degree to which populations have grown genetically

Table 3.1: Compute time of results on the mice data set.

| | time (seconds) |
|---|---|
| Initial fit | 30 |
| Hessian solve for $\alpha$ sensitivity | 3.9 |
| Linear approx. $\hat{\eta}_\gamma^{\text{lin}}(\alpha)$ for $\alpha = 1$ | 0.0013 |
| Linear approx. $\hat{\eta}_\gamma^{\text{lin}}(\alpha)$ for $\alpha = 11$ | 0.0012 |
| Refit $\hat{\eta}(\alpha)$ for $\alpha = 1$ | 14 |
| Refit $\hat{\eta}(\alpha)$ for $\alpha = 11$ | 13 |
| The influence function | 4.3 |
| Hessian solve for $\phi$ perturbation | 3.3 |
| Linear approx. $\hat{\eta}^{\text{lin}}(t)$ at $t = 1$ | 0.00099 |
| Refit $\hat{\eta}(t)$ at $t = 1$ | 22 |

distinct is important for conservation efforts: well-separated populations with little genetic diversity are particularly at risk of extinction.

We employ the topic model from Example 3. Figure 3.11 shows the inferred individual admixtures the initial fit with stick-breaking distribution is $\mathcal{P}_{\text{stick}}(\nu_{nk}) = \text{Beta}\,(\nu_{nk}|1, \alpha_0)$, $\alpha_0 = 6$. In this fit, there appears to be three dominant latent populations, which we arbitrarily label as populations 1, 2, and 3 (see Figure 3.11). The inferred populations generally correspond with the geographic regions: individuals from the Mbololo region have population 1 as their dominant admixture proportion; individuals from the Ngangao are dominantly population 2; individuals from the Chawia are admixed with populations 1, 2, and 3. The four individuals from Yale appear similar to individuals from the Ngangao; this is not surprising given that the Yale region is geographically located in proximity to the Ngangao region.

The resulting inference from our BNP model is qualitatively similar to the results reported in Pritchard et al. (2000), who employed a related model called STRUCTURE. STRUCTURE uses a Dirichlet prior for the individual admixtures with some fixed number of populations $K$ and does inference with MCMC. To select $K$, Pritchard et al. (2000) ran STRUCTURE with $K = 1, ..., 5$, and selected the $K$ that maximized a proxy for the posterior quantity $p(K|x)$, under the assumption of a uniformly distributed prior on $K$. Their algorithm selected $K = 3$. At $\alpha_0 = 3$, our BNP model agrees with Pritchard et al. (2000) in that we also uncover three dominant latent populations, with each having a loose correspondence with the Chawia, Ngangao, and Mbololo geographical regions.

**Sensitivity: number of populations**

We start by evaluating the sensitivity of the inferred number of populations to the $\alpha$ parameter in the $\text{Beta}\,(\nu_{nk}|1, \alpha)$ stick distribution. The expected number of in-sample clusters was defined in Example 6, except in this model, the summation is over the indices for individual $n$, locus $l$ and chromosome $i$. (We also often use "population" instead of "cluster", in this

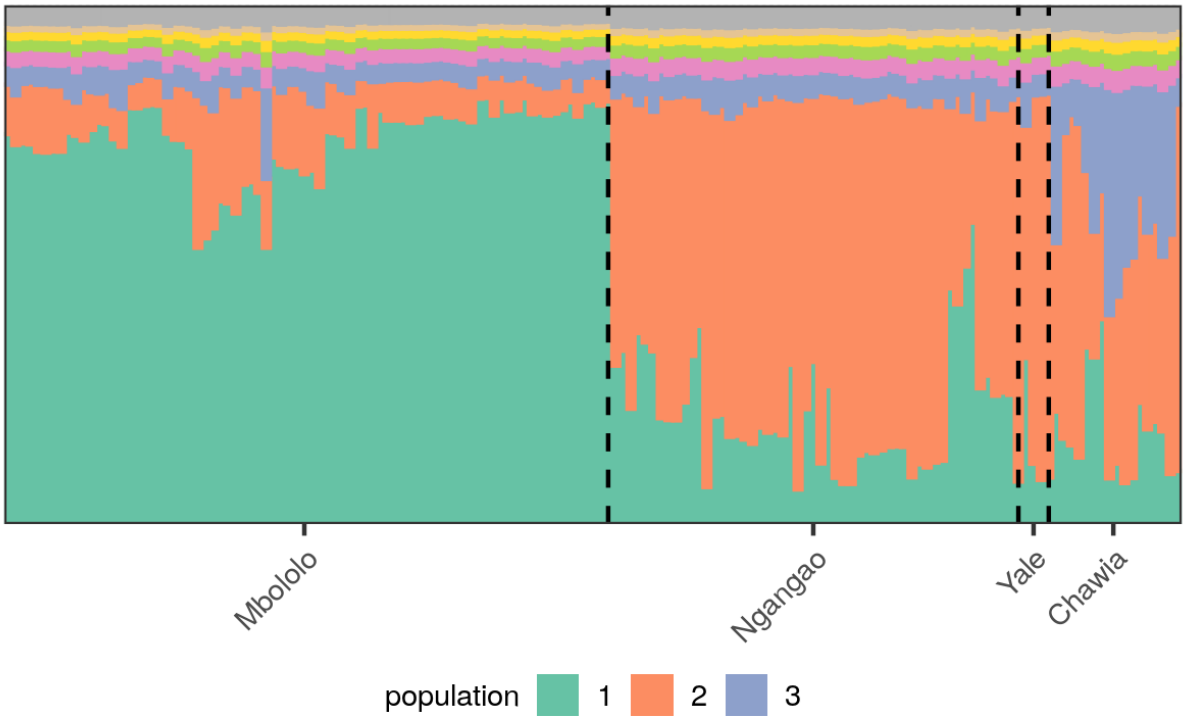



Figure 3.11: The inferred individual admixtures at $\alpha_0 = 3$. Each vertical strip is an individual and each color a latent population. Lengths of colored segments represent the inferred admixture proportions. Individuals are ordered by the geographic region from which they were sampled (Mbololo, Ngangao, Yale, and Chawia). In the text, we refer to the green, orange, and purple latent populations as population 1, 2, and 3, respectively.

application). We will allow the option of setting $\tau > 0$ in order to count only the populations that comprise a non-negligible fraction of the data set.

The expected number of latent populations is sensitive to $\alpha$ (Figure 3.12). Without any thresholding ($\tau = 0$), the expected number of populations quickly increases as $\alpha$ increases; in fact, it nearly saturates at $K_{\max} = 20$ when $\alpha = 7$. This sensitivity is likely due to the fact that the non-thresholded quantity is highly dependent on the behavior of small, nearly unoccupied populations; even though the probability of a single locus belonging to these rare populations is small, the probability that *none* of the $N \times L \times 2$ observed genotypes belong to these rare populations is non-negligible.

This motivates the use of thresholding in reporting the number of populations. We consider two thresholds, $\tau = 20$ and $\tau = 40$, corresponding to approximately 2% and 4% of the total number of loci in the data set, respectively. The thresholded estimates for the number of populations is still moderately sensitive to the value of $\alpha$. When refitting the variational approximation at $\alpha = 1, \ldots, 7$, the thresholded quantities vary between two and four latent populations.

The linearized variational parameters $\hat{\eta}_\gamma^{\text{lin}}(t)$ imperfectly captures the results observed by refitting. We formed the linear approximation at $\alpha_0 = 3$ and computed $g_{\text{cl},\tau}(\hat{\eta}_\gamma^{\text{lin}}(\alpha))$ for $\alpha = 1, \ldots, 7$. For this range of $\alpha$, the linearized parameters and the refitted parameters almost perfectly agree on values of $g_{\text{cl},\tau}$ with $\tau = 0$. However, when $\tau = 20$ and $\tau = 40$, the linearized parameters underestimated the true sensitivity of $g_{\text{cl},\tau}$ found by refitting. In particular, the linearized parameters failed to produce the reduction to two latent populations at $\alpha = 1$ observed in the refits.

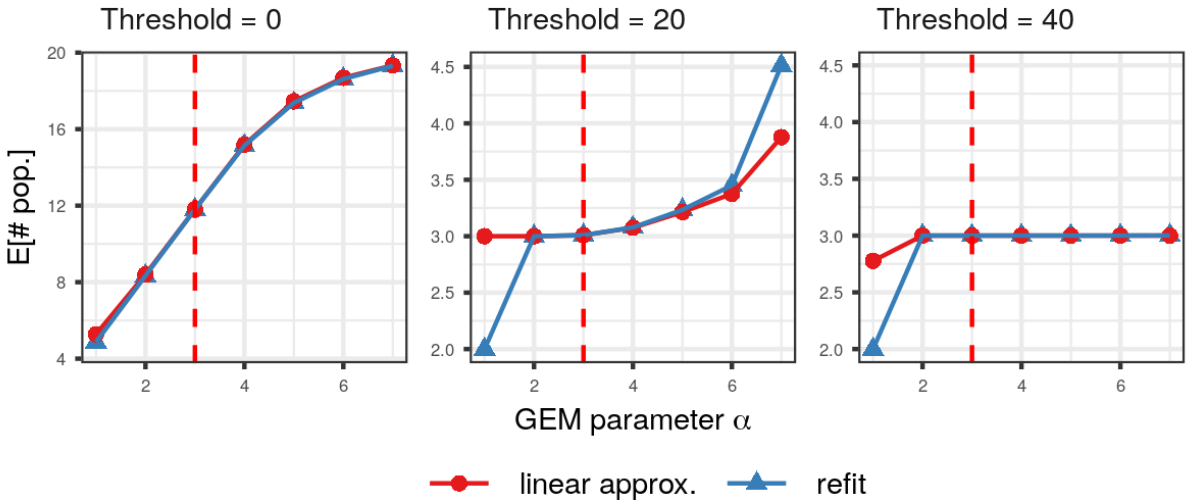



Figure 3.12: The expected number of (thresholded) populations in the thrush data as $\alpha$ varies. We computed the linear approximation at $\alpha_0 = 3$, and we compare the results under the linearly approximated variational parameters with the results observed after refitting. Thresholds at $\tau = 20$ and $\tau = 40$ corresponding to approximately 2% and 4% of the total number of loci in the data set, respectively.

We provide some more intuition concerning the thresholded estimate for the number of populations. The posterior quantity $g_{\text{cl},\tau}$ is closely related to the expected number of loci belonging to each population, defined as

$$g_{\text{loci}}(\eta; k) = \mathop{\mathbb{E}}_{\mathcal{Q}(z|\eta)} \left[ \sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{i=1}^{2} z_{nlik} \right].$$

Figure 3.13 plots $g_{\text{loci}}$ for the first six populations as $\alpha$ varies. The expected number of loci at the initial fit, $g_{\text{loci}}(\hat{\eta}(\alpha_0); k)$, is at least 100 for populations $k = 1, 2$, and 3 and less than 12 for the remaining populations. A sample of memberships $z \sim \mathcal{Q}(z|\hat{\eta}(\alpha_0))$ will almost always have at least $\tau$ loci allocated to populations 1, 2, and 3, while the allocations to each remaining population will almost always be below $\tau$, for either $\tau = 20$ or $\tau = 40$. Thus, at $\alpha = \alpha_0$ there then are clearly 3 populations by our definition of $g_{\text{cl},\tau}$, for either $\tau$.

At $\alpha = 7$, the expected number of loci belonging to population 4 increases to 20.7, a new population emerges above the threshold at $\tau = 20$. Both the linearized and the refitted variational parameters agree on this shift in allocation to population 4. On the other hand, under the refitted variational parameters at $\alpha = 1$, the expected number of loci belonging to population 3 decreases to 6.7, below the thresholds $\tau = 20$ and $\tau = 40$. Thus, the expected number of latent populations with allocations above either threshold decreases to two. The linearized parameters under-estimated this decrease in allocation to population 3, and therefore continued to estimate three latent populations even at $\alpha = 1$.

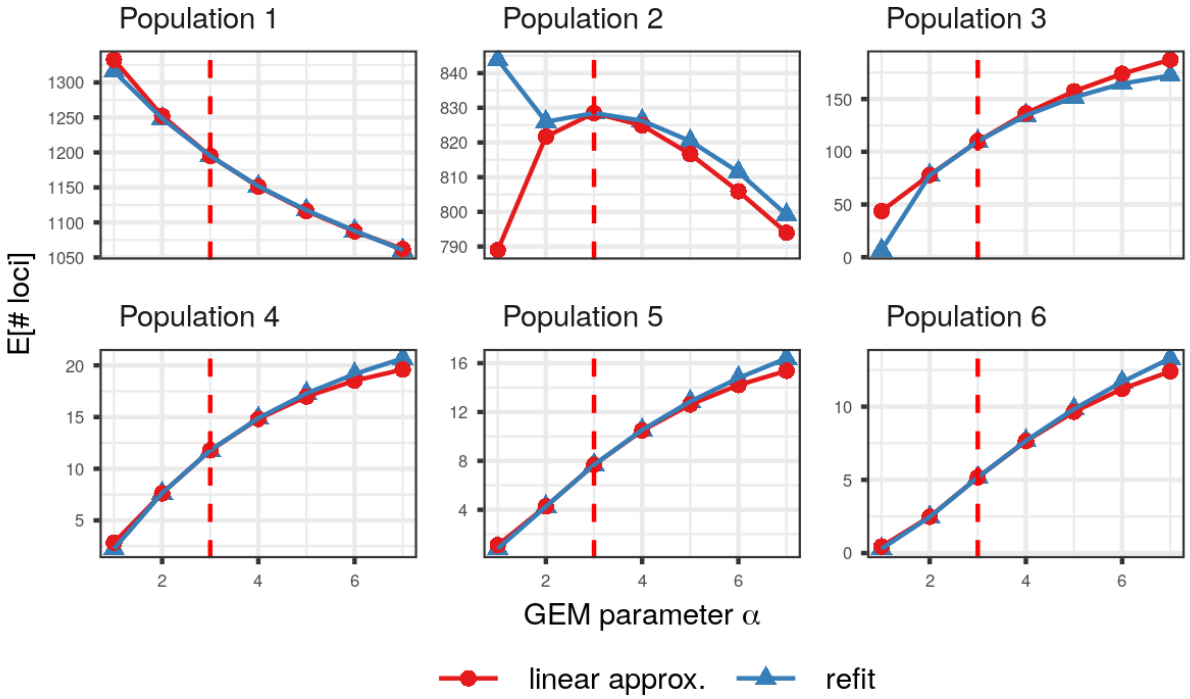



Figure 3.13: The expected number of loci per population as $\alpha$ varies.

### Sensitivity: individual admixtures

Examining the inferred admixtures in Figure 3.11 provides clues into the historical migration patterns of the genotyped individuals. For example, while individuals collected from the Mbololo region are inferred to be admixed primarily with population 1, several individuals from this region have abnormally large admixture proportions of population 2. Conversely, while individuals collected from the Ngangao region are admixed primarily with population 2, a few of these individuals have abnormally large admixture proportions of population 1. This suggests that some migration has occurred between the Mbololo and Ngangao regions.

We evaluate the sensitivity of this conclusion to possible prior perturbations. Consider the posterior statistic

$$g_{\text{admix}}(\eta; \mathcal{N}, k) = \mathop{\mathbb{E}}_{\mathcal{Q}(\pi|\eta)} \left[ \frac{1}{|\mathcal{N}|} \sum_{n \in \mathcal{N}} \pi_{nk} \right],$$

the average admixture proportion of population $k$ in a set of individuals $\mathcal{N}$.

We present results on three variations of $g_{\text{admix}}$: $\mathcal{N} = \{26, ..., 31\}$ and $k = 2$, corresponding to the six individuals from the Mbololo region with outlying proportions of population 2; $\mathcal{N} = \{125, ..., 128\}$ and $k = 1$, corresponding to the four individuals from the Ngangao region with outlying proportions of population 1; $\mathcal{N} = \{139, ..., 155\}$ and $k = 3$, corresponding to all individuals from the Chawia region. The first two posterior quantities relate to the inferred mixing between Mbololo and Ngangao. In the last case, we are studying the sensitivity of having a third latent population present, a population which primarily appears in Chawia individuals.

We construct the worst-case negative perturbation for each variant of $g_{\text{admix}}$. We consider the negative direction because we are interested in testing the robustness of these patterns' existence. For each worst-case perturbation, we examine the effect on its respective posterior quantity as $t \to 1$ in the multiplicatively perturbed prior $\mathcal{P}(\nu_{nk}|t) = \mathcal{P}_0(\nu_{nk}) \exp(t\phi_{\text{wc}}(\nu_{nk}))$ (Figure 3.14). Under the linearized variational parameters $\hat{\eta}^{\text{lin}}_{\gamma}(t)$, the admixture proportion of population 2 in the outlying Mbololo individuals is nearly halved at $t = 1$. The same quantity computed after refitting the model confirms the sensitivity predicted by the linearized variational parameters On the other hand, the presence of population 1 in the outlying Ngangao individuals appears to be insensitive even after this worst-case perturbation. The linearized and the refitted variational parameters again agree on this conclusion. Finally, the presence of population 3 in the Chawia individuals is anticipated to be sensitive by the linearized parameters, as this admixture proportion steadily decreases as $t \to 1$. However, under the refits, this admixture proportion does not decrease steadily but rather levels off after $t = 0.5$.

Our sensitivity analysis suggests that the inferred migration from Mbololo to Ngangao based on the outlying Ngangao admixtures appears robust to our stick-breaking prior. On the other had, the the outlying Mbololo admixtures appears to be more sensitive to the stick-breaking prior, and conclusions about migration from Ngangao to Mbololo may be dependent on prior choices.

Finally, the computational cost of linearizing the variational parameters is again favorable compared with the cost of refitting (Table 3.2). The linear approximation allows us to quickly explore all the functional perturbations presented here, and many more: computing the linearized variational parameters (including the Hessian inversion) $\hat{\eta}^{\text{lin}}_{\gamma}$ takes about half a second for a given perturbation. On the other hand, refitting the model after a prior perturbation can take more than ten seconds. While exploring *all* the possible functional perturbations is impossible, the linear approximation allows rapid exploration of the space of perturbations. Notice also that computing the influence function only takes half a second. As

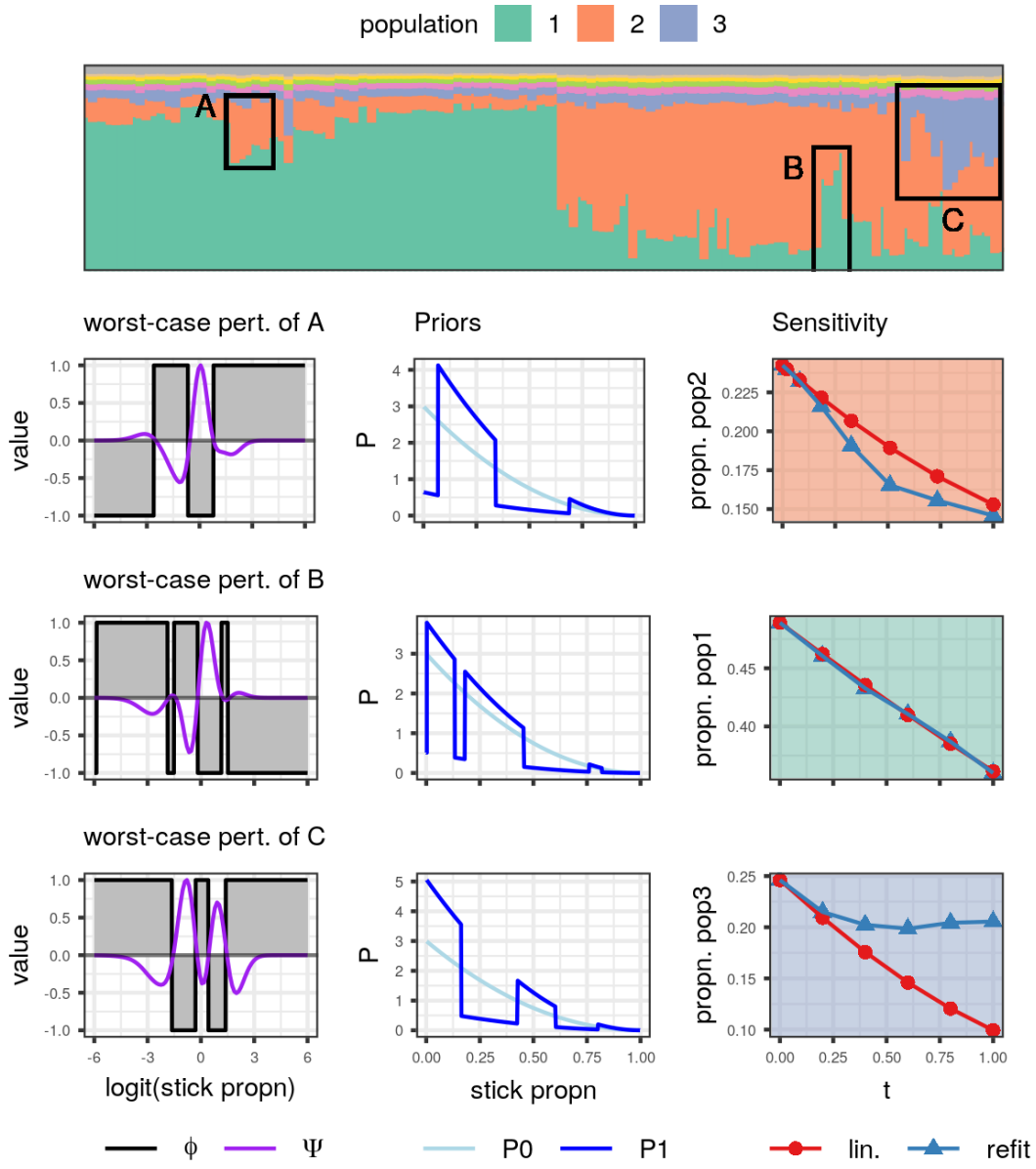



Figure 3.14: Sensitivity of inferred admixtures for several outlying individuals. For individuals A, we examine the sensitivity of the admixture proportion of population 2. For individuals B, we examine the population 1 admixture For the individuals C, we examine the population 3 admixture. (Left column) The worst-case negative perturbation with unit $L_\infty$-norm in grey, plotted against the influence function in purple (scaled to also have $L_\infty$ norm equal to 1). (Middle column) The effect of the perturbation on the prior density. (Right column) Effects on the inferred admixture.

Table 3.2: Compute time of results on the thrush dataset. Timing results on perturbation $\phi$ are reported for the worst-case perturbation "A" in Figure 3.14. Timing on other considered $\phi$ are similar.

| | time (seconds) |
|---|---|
| Initial fit | 7 |
| Hessian solve for $\alpha$ sensitivity | 0.32 |
| Linear approx. $\hat{\eta}_\gamma^{\text{lin}}(\alpha)$ for $\alpha = 1, ..., 10$ | 0.0059 |
| Refits $\hat{\eta}(\alpha)$ for $\alpha = 1, ..., 10$ | 34 |
| The influence function | 0.59 |
| Hessian solve for $\phi$ | 0.38 |
| Linear approx. $\hat{\eta}_\gamma^{\text{lin}}(\epsilon)\vert_{\epsilon=1}$ for $\phi$ | 0.00085 |
| Refit $\hat{\eta}(\epsilon)\vert_{\epsilon=1}$ for $\phi$ | 13 |

we have demonstrated, the influence function provides an informative guide for uncovering which perturbations might result in greater sensitivity than others. These perturbations can be more explored either by either linearizing in the direction of the perturbation, or by refitting.

## 3.7 Limitations of local sensitivity

In this final subsection, we discuss some examples where the linearized variational parameters $\hat{\eta}^{\text{lin}}(t)$ are a poor substitute for the refitted variational parameters $\hat{\eta}(t)$ (in this subsection, the only posterior quantities considered are functions of global parameters, so we will not make a distinction between $\hat{\eta}^{\text{lin}}$ and $\hat{\eta}_\gamma^{\text{lin}}$ here). Naturally, the linearized parameters $\hat{\eta}^{\text{lin}}(t)$ will be a poor substitute for $\hat{\eta}(t)$ when the mapping $t \mapsto \hat{\eta}(t)$ is highly nonlinear. The examples discussed below use the STRUCTURE model and dataset presented in Section 3.6.3, and we examine results after the worst-case perturbation "A" in Figure 3.14.

Recall from Figure 3.14 that the linearized parameters agreed with the refitted parameters in predicting the diminished admixture proportion of population 2 in the outlying Mbololo individuals (individuals "A") at the perturbed prior $\mathcal{P}_1 = \mathcal{P}_0 \exp(\phi_{\text{wc}})$. However, while the linearized parameters were able to capture the change in overall admixture proportion, it does not perform uniformly well over all individual admixtures (Figure 3.15). For example, the admixture proportion of population 2 in individual $n = 25$ dramatically increased after refitting with the perturbed prior $\mathcal{P}_1$; the linearized parameters failed to reproduce this change.

Figure 3.16 examines individual $n = 25$ more closely. The bottom row plots this individual's admixture proportions as $t$ varies from 0 to 1 in the perturbed prior $\mathcal{P}(\nu|t) = \mathcal{P}_0(\nu_k) \exp(t\phi_{\text{wc}}(\nu_k))$. The linearized parameters poorly captured the change in admixture proportions observed after refitting, particularly for populations 1 and 2, for values of $t$ close to 1. Even though we retain non-linearities in the mapping from variational parameters

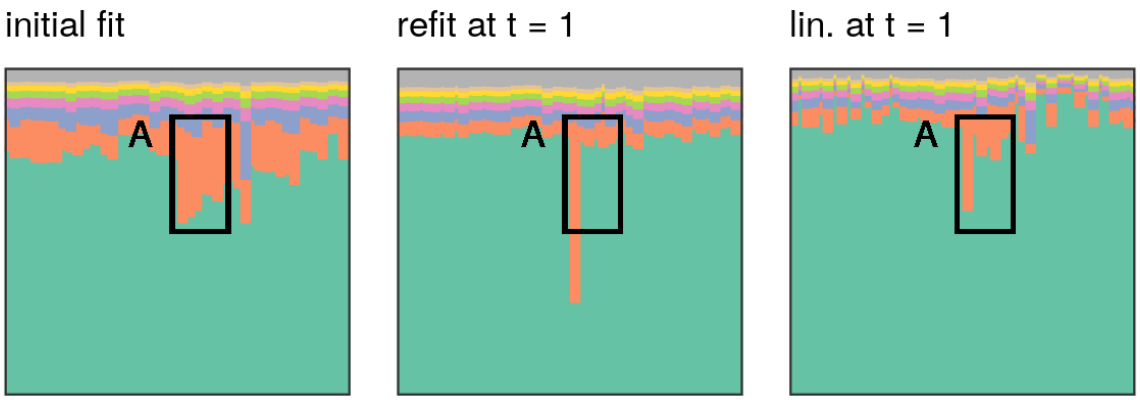



Figure 3.15: Inferred admixtures after the worst-case perturbation to individuals "A" (see Figure 3.14 for perturbation).

to the posterior statistic, for this perturbation, the mapping from prior parameter $t$ to the relevant variational parameters is highly non-linear. This latter mapping is what we linearize and what causes our approximation to fail in this case. Specifically, the variational location parameter on the first stick-breaking proportion is concave as a function of $t$ — the location parameter increases for small $t$, then decreases as $t \to 1$. However, $\hat{\eta}^{\text{lin}}(t)$ linearizes the relationship between the location parameter and $t$. Therefore, the corresponding admixture mixture proportion of population 1 is over-estimated under the linearized variational parameters. Furthermore, because our linearized variational parameters over-estimated the length of the first stick, and the second admixture proportion is a product of the remaining stick times the second stick-breaking proportion, the linearized variational parameters then under-estimates the admixture proportion of population 2.

Figure 3.17 shows a similar situation for individual $n = 74$. The linearized variational parameters grossly over-estimated the length of the first stick, resulting in the later admixture proportions being under-estimated. The third admixture proportion was particularly poorly approximated under the linearized variational parameters. Given the recursive nature of the relationship between admixtures and stick-breaking proportions, errors at early sticks affect later admixture proportions. Fully linearizing the mapping $t \mapsto g(\hat{\eta}(t))$ to form the approximation $g^{\text{lin}}(t)$ (Equation 3.9) avoids this problem. In this example, $g^{\text{lin}}(t)$ outperforms $g(\hat{\eta}^{\text{lin}}(t))$, with $g$ being the admixture proportion of population 3. In all the previous examples (including the example in Figure 3.16), computing $g(\hat{\eta}^{\text{lin}}(t))$, and thus retaining the non-linearities in the mapping from $\eta \mapsto g(\eta)$, does no worse, and is usually better, than the fully linearized version, $g^{\text{lin}}(t)$—as can be seen by drawing tangent lines of the refitted curve at $\alpha = \alpha_0$ or $t = 0$ for Figures 3.2, 3.12, 3.13 and 3.16. It is likely that $g(\hat{\eta}^{\text{lin}}(t))$ outperforms $g^{\text{lin}}(t)$ for most posterior quantities, though as we see in Figure 3.17, this is not guaranteed to always be true.

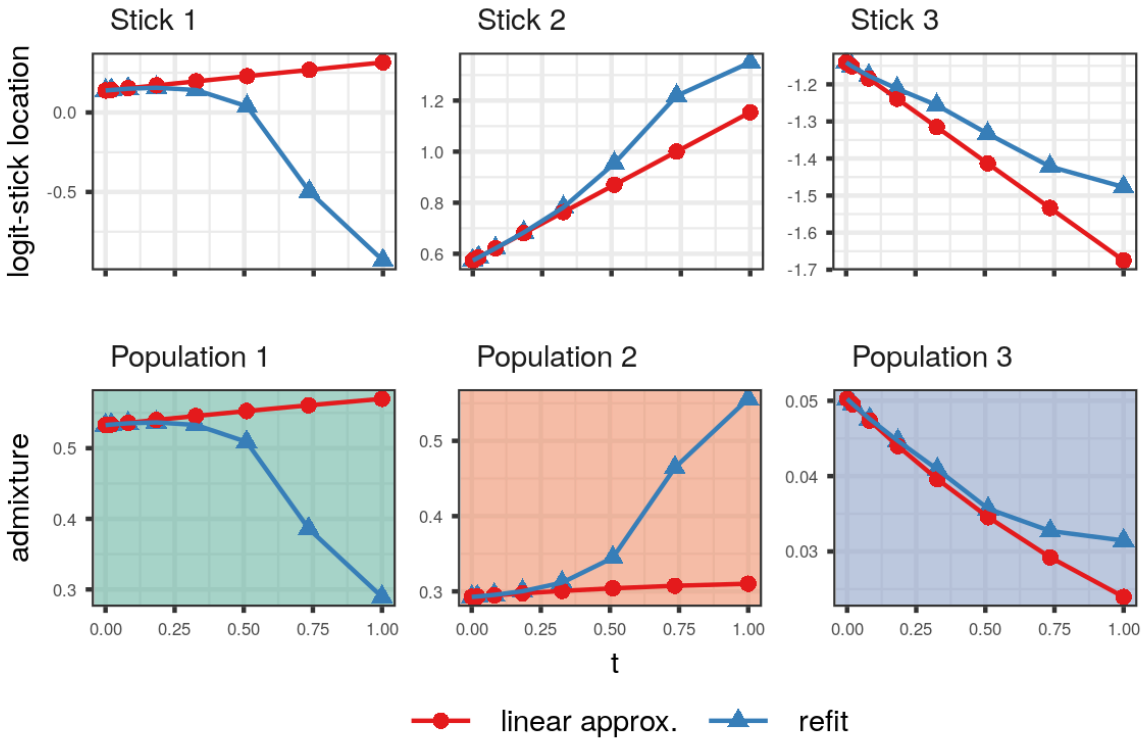



Figure 3.16: An individual ($n = 26$) for which the linearly approximated variational parameters poorly captured the change in admixture observed after refitting as $t \to 1$. (Top row) the change in location parameter of the normally distributed logit-sticks, for the first three sticks. The response here is a variational parameter, so the approximation (red) is necessarily linear with respect to $t$. (Bottom row) the change in the inferred admixtures for populations 1, 2, and 3.

## 3.8 Conclusion

The concept of local sensitivity in Bayesian nonparametric models is not novel (see, for example, Basu et al. (1996)). Historically however, the derivatives required for local sensitivity required analytic derivations, which are tedious to produce or perhaps unavailable altogether. With the availability of modern automatic differentiation tools, such difficulties are rendered obsolete. All that is necessary for computing derivatives is the implementation in computer code the KL objective as a function of variational parameters and the hyperparameter. Tools such as `JAX` (Bradbury et al., 2018) handles the derivative evaluations with either a backward or forward pass through the computation graph. We showed that computing these derivatives and linearizing the variational parameters can be an order of magnitude faster than refitting the KL objective at a perturbed prior.

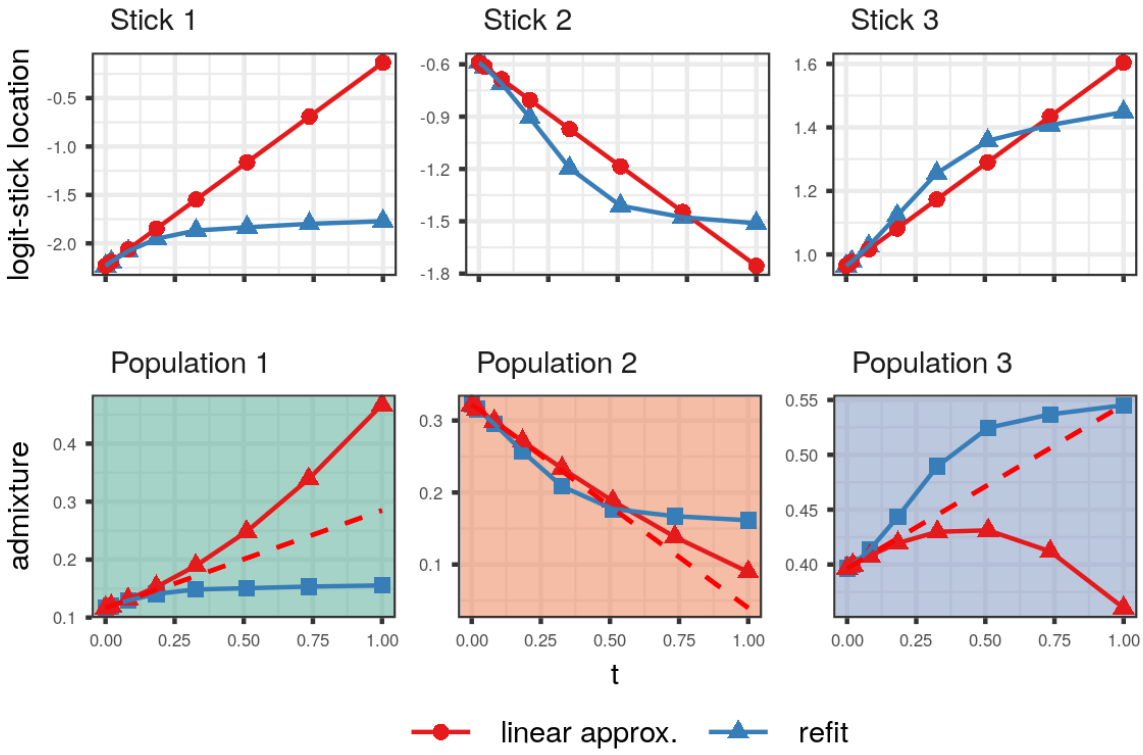



Figure 3.17: An example where linearizing the posterior quantity itself outperforms linearizing the variational parameters only. Shown are logit-stick location parameters (top row) and inferred admixtures (bottom row) for individual $n = 74$ and populations $k = 1, 2$ and 3. Dashed red is the approximation $g^{\text{lin}}(t)$ formed by linearizing the inferred admixture $\mathbb{E}_{\mathcal{Q}}[\pi_{nk}]$ with respect to prior parameter $t$. On the admixture proportion of population 3, $g^{\text{lin}}(t)$ outperforms $g(\hat{\eta}^{\text{lin}}(t))$ (solid red).

Computing the influence function is also just as fast as forming the linear approximation. Evaluating sensitivity to all possible perturbations is impossible, but we demonstrated how the influence function can guide our search for functional perturbations that result in high sensitivity. The high-influence perturbations can then be explored more closely either by our linear approximation or by refitting.

Finally, we remark that the framework of Theorem 1 for producing local sensitivity measures extend well-beyond inference in variational Bayes. The same conceptual set-up can be applied to study the sensitivity of *M-estimators*, which are vectored-valued estimators $\hat{y}$ characterized as a minimizer of some objective function. The optimal variational parameters $\hat{\eta}$, which are minimizers of the KL objective, are just one such example. The next two chapters further explore this generality, and apply similar local approximations to data perturbations, resulting in the *infinitesimal jackknife* and the *linear bootstrap*.

## 3.9 Supplemental details

### 3.9.1 continuity lemmas

A standard consequence of the dominated convergence theorem is the ability to exchange integration and differentiation. Since we will use this result frequently, we state it here in our own notation as Theorem 2.

**Theorem 2.** *(Billingsley, 1986, Theorem 16.8) Let $\mu$ be sigma-finite measure on $\Omega_\theta$, and let $S_t \subseteq \mathbb{R}$. Let $f : \Omega_\theta \times S_t \mapsto \mathbb{R}$.*

*If there exists a function $M(\theta)$ with $\int M(\theta)\mu(d\theta) < \infty$ such that $|f(\theta, t)| \leq M(\theta)$, $\mu$-almost surely, for all $t \in S_t$, then the map $t \mapsto \int f(\theta, t)\mu(d\theta)$ is continuous.*

*Further, suppose that the derivative $\frac{\partial f(\theta,t)}{\partial t}\Big|_t$ exist $\mu$-almost surely for $t \in S_t$. If there exists an $M'(\theta)$ such that $\int M'(\theta)\mu(d\theta) < \infty$ and $\left|\frac{\partial f(\theta,t)}{\partial t}\Big|_t\right| \leq M'(\theta)$, $\mu$-almost surely and for all $t \in S_t$, then*

$$\frac{\partial \int f(\theta, t)\mu(d\theta)}{\partial t}\Big|_t = \int \frac{\partial f(\theta, t)}{\partial t}\Big|_t \mu(d\theta).$$

**Lemma 3.** *Let Assumption 1 hold for some $\psi(\theta, t)$ as well as for $\psi(\theta, t) = 1$. Define*

$$\begin{aligned}
\overline{\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)} &:= \nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta) - \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)\right] \\
\overline{\nabla^2_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)} &:= \nabla^2_\eta \log \tilde{\mathcal{Q}}(\theta|\eta) - \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\nabla^2_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)\right].
\end{aligned}$$

*Then the following equalties hold:*

$$\left.\frac{\partial \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta,t)]}{\partial \eta}\right|_{\eta} = \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\overline{\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)}\ \left(\psi(\theta,t) - \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta,t)]\right)\right] \tag{3.22}$$

$$\left.\frac{\partial^2 \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta,t)]}{\partial \eta \partial t}\right|_{\eta,t} =$$
$$\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\overline{\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)}\ \left(\nabla_t \psi(\theta,t) - \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\nabla_t \psi(\theta,t))]\right)\right] \tag{3.23}$$

$$\left.\frac{\partial^2 \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta,t)]}{\partial \eta \partial \eta^T}\right|_{\eta} =$$
$$\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\overline{\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)}\ \overline{\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)}^{\,T}\left(\psi(\theta,t) - \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta,t)]\right)\right] +$$
$$\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\overline{\nabla^2_\eta \log \tilde{\mathcal{Q}}(\theta|\eta)}\ \left(\psi(\theta,t) - \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta,t)]\right)\right]. \tag{3.24}$$

*Proof.* The proof follows by repeatedly using Theorem 2 to interchange the order of integration and differentiation as in (Giordano et al., 2018, Theorem 1). For example,

$$\left.\frac{\partial \int \mathcal{Q}(\theta|\nu)\psi(\theta|t)\mu(d\theta)}{\partial \eta}\right|_{\eta} =$$
$$\int \left.\frac{\partial \mathcal{Q}(\theta|\nu)\psi(\theta|t)}{\partial \eta}\right|_{\eta} \mu(d\theta) = \qquad \text{(Assumption 1 (Item 1) and Theorem 2)}$$
$$\int \nabla_\eta \log \tilde{\mathcal{Q}}(\theta,\eta)\,\psi(\theta|t)\mathcal{Q}(\theta,\eta)\mu(d\theta) =$$
$$\mathop{\mathbb{E}}_{\mathcal{Q}(\theta,\eta)}\left[\nabla_\eta \log \tilde{\mathcal{Q}}(\theta,\eta)\,\psi(\theta|t)\right].$$

Applying analogous reasoning to the denominator of

$$\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta,t)] = \frac{\int \psi(\theta,t)\mathcal{Q}(\theta|\eta)\mu(d\theta)}{\int \mathcal{Q}(\theta|\eta)\mu(d\theta)}$$

and applying the chain rule gives Equation 3.22.

For Equation 3.23, by anaologously applying Assumption 1 (Item 1) and the DCT gives

$$\left.\frac{\partial \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta|\eta)]}{\partial t}\right|_{t} = \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\nabla_t \psi(\theta|\eta)\right].$$

Applying Assumption 1 (Item 2) and Theorem 2 gives

$$\left.\frac{\partial \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)\psi(\theta|\eta)\right]}{\partial t}\right|_t = \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)\nabla_t\psi\left(\theta|\eta\right)\right],$$

where we have used the fact that the absolute value of any component of the vector $\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)\psi(\theta|\eta)$ is bounded above by a constant times $\left\|\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)\psi(\theta|\eta)\right\|_2$. From the preceding two displays, Equation 3.23 follows.

Finally, for Equation 3.24, we need to differentiate Equation 3.22. In addition to quantities already considered above, Equation 3.22 involves terms of the following form, to which we can apply Theorem 2 using the corresponding assumptions:

$$\mathop{\mathbb{E}}_{\mathcal{Q}\theta|\eta}\left[\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)\right] \qquad \text{Assumption 1 (Item 2)}$$

$$\mathop{\mathbb{E}}_{\mathcal{Q}\theta|\eta}\left[\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)\psi(\theta,t)\right]. \qquad \text{Assumption 1 (Item 2)}$$

Equation 3.24 then follows by differentiating as above and collecting terms. □

**Lemma 4.** *(See Proof 3.9.1 on this page.) Let Assumption 1 hold for some $\psi$ as well as with $\psi(\theta,t)=1$. Then*

$$\eta, t \mapsto \left.\frac{\partial \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\psi(\theta,t)\right]}{\partial \eta}\right|_{\eta,t},$$

$$\eta, t \mapsto \left.\frac{\partial^2 \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\psi(\theta,t)\right]}{\partial \eta \partial t}\right|_{\eta,t}, \text{ and}$$

$$\eta, t \mapsto \left.\frac{\partial^2 \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\psi(\theta,t)\right]}{\partial \eta \partial \eta^T}\right|_{\eta}$$

*are continuous on $\mathcal{B}_\eta \times \mathcal{B}_t$.*

*Proof of Lemma 4.* By Lemma 3, the mixed partial $\eta, t \mapsto \left.\frac{\partial^2 \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}[\psi(\theta,t)]}{\partial\eta\partial t}\right|_{\eta,t}$ is a continuous combination of terms of the form

$$\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)\nabla_t\psi\left(\theta,t\right)\right] \qquad \text{Assumption 1 (Item 4)}$$

$$\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)\right] \qquad \text{Assumption 1 (Item 2)}$$

$$\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)}\left[\nabla_t\psi\left(\theta,t\right)\right]. \qquad \text{Assumption 1 (Item 2)}$$

By the corresponding assumptions, Theorem 2 applies to each of these terms, and by Assumption 1, each of the expressions in the preceding display are continuous. For example,

$$\begin{aligned}
&\left\| \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)} \left[ \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \nabla_t \psi\left(\theta, t\right) \right] - \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta')} \left[ \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta'\right) \nabla_t \psi\left(\theta, t'\right) \right] \right\|_2 = \\
&\quad \left\| \int \left( \mathcal{Q}(\theta|\eta) \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \nabla_t \psi\left(\theta, t\right) - \mathcal{Q}(\theta|\eta') \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta'\right) \nabla_t \psi\left(\theta, t'\right) \right) \mu(d\theta) \right\|_2 \leq \\
&\quad \int \left\| \mathcal{Q}(\theta|\eta) \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \nabla_t \psi\left(\theta, t\right) - \mathcal{Q}(\theta|\eta') \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta'\right) \nabla_t \psi\left(\theta, t'\right) \right\|_2 \mu(d\theta) \leq \\
&\quad \int \left\| \left( \mathcal{Q}(\theta|\eta) - \mathcal{Q}(\theta|\eta') \right) \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \nabla_t \psi\left(\theta, t\right) \right\|_2 \mu(d\theta) + \\
&\qquad \int \left\| \mathcal{Q}(\theta|\eta') \left( \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) - \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta'\right) \right) \nabla_t \psi\left(\theta, t\right) \right\|_2 \mu(d\theta) + \\
&\qquad \int \left\| \mathcal{Q}(\theta|\eta') \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta'\right) \left( \nabla_t \psi\left(\theta, t\right) - \nabla_t \psi\left(\theta, t'\right) \right) \right\|_2 \mu(d\theta).
\end{aligned}$$

By Assumption 1 (Item 4) we can apply Theorem 2 to each term in the final line of the preceding display, giving

$$\begin{aligned}
&\lim_{\eta' \to \eta} \lim_{t' \to t} \left\| \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)} \left[ \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \nabla_t \psi\left(\theta, t\right) \right] - \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta')} \left[ \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta'\right) \nabla_t \psi\left(\theta, t'\right) \right] \right\|_2 \leq \\
&\quad \int \lim_{\eta' \to \eta} \lim_{t' \to t} \left\| \left( \mathcal{Q}(\theta|\eta) - \mathcal{Q}(\theta|\eta') \right) \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \nabla_t \psi\left(\theta, t\right) \right\|_2 \mu(d\theta) + \\
&\qquad \int \lim_{\eta' \to \eta} \lim_{t' \to t} \left\| \mathcal{Q}(\theta|\eta') \left( \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) - \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta'\right) \right) \nabla_t \psi\left(\theta, t\right) \right\|_2 \mu(d\theta) + \\
&\qquad \int \lim_{\eta' \to \eta} \lim_{t' \to t} \left\| \mathcal{Q}(\theta|\eta') \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta'\right) \left( \nabla_t \psi\left(\theta, t\right) - \nabla_t \psi\left(\theta, t'\right) \right) \right\|_2 \mu(d\theta) = 0,
\end{aligned}$$

the final equality following from the continuity assumptions of Assumption 1.

Similarly, $\left. \frac{\partial^2 \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)} [\psi(\theta,t)]}{\partial \eta \partial \eta^T} \right|_\eta$ involves terms of the form

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)} \left[ \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)^T \psi(\theta, t) \right] && \text{Assumption 1 (Item 5)} \\
&\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)} \left[ \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \nabla_\eta \log \tilde{\mathcal{Q}}\left(\theta|\eta\right)^T \right] && \text{Assumption 1 (Item 5)} \\
&\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)} \left[ \nabla_\eta^2 \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \psi(\theta, t) \right] && \text{Assumption 1 (Item 3)} \\
&\mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)} \left[ \nabla_\eta^2 \log \tilde{\mathcal{Q}}\left(\theta|\eta\right) \right], && \text{Assumption 1 (Item 3)}
\end{aligned}$$

to which we can apply Theorem 2 by the corresponding assumption. Reasoning analogously to the other term, the conclusion follows. □

*Proof.* **Proof of Theorem 1.** For the duration of the proof, define

$$\rho(\eta, t) := \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\eta)} \left[\log \mathcal{P}(\theta|t) - \log \mathcal{P}(\theta|0)\right] \quad \text{and}$$
$$\rho_\eta(\eta, t) := \left.\frac{\partial \rho(\eta, t)}{\partial \eta}\right|_\eta .$$

Expanding $\log \mathcal{P}(\theta|t)$ in Equation 3.3, we see that

$$\mathrm{KL}(\eta, t) = \mathrm{KL}(\eta, 0) + \rho(\eta, t). \tag{3.25}$$

By Lemma 3, $\eta \mapsto \rho(\eta, t)$ is continuous, and by Lemma 4 $\eta \mapsto \rho(\eta, t)$ is continuously differentiable, for all $\eta, t \in \mathcal{B}$. So $\partial \mathrm{KL}(\eta, t)/\partial \eta$ is continuous for all $\eta, t \in \mathcal{B}$ and, by the first-order condition of Equation 3.7, $\hat{\eta}(t)$ satisfies

$$\left.\frac{\partial \mathrm{KL}(\eta, t)}{\partial \eta}\right|_{\hat{\eta}(t), t} = \left.\frac{\partial \mathrm{KL}(\eta, 0)}{\partial \eta}\right|_{\hat{\eta}(t)} + \rho_\eta(\hat{\eta}(t), t) = 0 \tag{3.26}$$

for all $t \in \mathcal{B}_t$.

We wish to apply the implicit function theorem to Equation 3.26, for which we must show that $\partial \mathrm{KL}(\eta, t)/\partial \eta$ is continuously differentiable in both $\eta$ and $t$. By Assumption 2 (Item 1), $\partial \mathrm{KL}(\eta, 0)/\partial \eta$ is continuously differentiable, so we need only consider $\rho_\eta(\theta, t)$.

By Lemma 4 and Assumption 3, we have that both $\partial \rho_\eta(\eta, t)/\partial \eta$ and $\partial \rho_\eta(\eta, t)/\partial t$ are continuous in both $\eta$ and $t$. Since both its partial derivatives are continuously differentiable, the joint map $\eta, t \mapsto \rho_\eta(\eta, t)$ is also continuously differentiable (e.g., Fleming (2012, Theorem 3.2)). Consequently, $\partial \mathrm{KL}(\eta, t)/\partial \eta$ is continuously differentiable in both $\eta$ and $t$.

Together with Assumption 2 (Item 3), which gives that $\partial^2 \mathrm{KL}(\eta, t)/\partial \eta \partial \eta^T$ is invertible at $\hat{\eta}$, the result then follows from the implicit function theorem Krantz and Parks (2012, Theorem 3.3.1). For convenience, Table 3.9.1 shows the correspondence between their notation and ours.

| Krantz & Parks notation | Our notation |
|---|---|
| $\Phi(x)$ | $\mathrm{KL}(\eta, t)$ |
| $Q$ | 1 |
| $M$ | $D_\eta$ |
| $U$ | $\mathcal{B}$ |
| $W$ | $\mathcal{B}_t$ |
| $x_1, \ldots, x_Q$ | $t$ |
| $x_{Q+1}, \ldots, x_N$ | $\eta$ |
| $f_1(x_a), \ldots, f_M(x_a)$ | $\hat{\eta}(t)$ |

The form of the derivative is given by combining Equation 3.25 with Equation 3.23 of Lemma 3, since

$$\left.\frac{\partial^2 \mathrm{KL}(\eta, t)}{\partial \eta \partial t}\right|_{\hat{\eta}, 0} = \mathop{\mathbb{E}}_{\mathcal{Q}(\theta|\hat{\eta})} \left[\overline{\nabla_\eta \log \tilde{\mathcal{Q}}(\theta|\hat{\eta})} \left.\frac{\partial \log \mathcal{P}(\theta|t)}{\partial t}\right|_{t=0}\right].$$

□

### 3.9.2 Variational approximation for the regression mixture model

We detail the variational approximation for the local latent variables in the regression mixture model (Example 2). All latent variables fully factorize, except for the cluster assignments $z$ and the addtive shifts $b$. Under $\mathcal{Q}$, their distribution factorizes as

$$\mathcal{Q}(z, b|\eta) = \prod_{n=1}^{N} \mathcal{Q}(z_n|\eta) \prod_{k=1}^{K_{\max}} \mathcal{Q}(b_n|z_{nk} = 1, \eta).$$

As discussed in Section 3.2, the optimal distribution $\mathcal{Q}(z_n|\eta)$ is multinomial whose parameters can be set in closed form as a function of the global variational parameters only. We allow the distribution of $\beta_n$ to depend on $z_{nk}$ so that the its optimal distribution can also be set in closed form as a function of global parameters as well.

The optimal distribution $q(b_n|z_{nk} = 1, \eta)$ is Gaussian,

$$q(b_n|z_{nk} = 1, \eta) = \mathcal{N}\left(b_n|\hat{\mu}_{b_{nk}}, \hat{\sigma}^2_{b_{nk}}\right).$$

Let

$$\begin{aligned}
\rho^{(1)}_{nk} &= \mathop{\mathbb{E}}_{\mathcal{Q}(\beta_k|\eta)}\left[\sum_{m=1}^{M} \tau_k(x_{nm} - A_m\mu_k)\right] + \tau_0\mu_0 \\
\rho^{(2)}_{nk} &= M \mathop{\mathbb{E}}_{\mathcal{Q}(\beta_k|\eta)}[\tau_k] + \tau_0,
\end{aligned}$$

where $\mu_0$ and $\tau_0$ are the prior mean and information on $b_n$, respectively.

The optimal parameters for the Gaussian distribution on $b_n$ is given by

$$\begin{aligned}
\hat{\mu}_{b_{nk}} &= \rho^{(1)}_{nk}/\rho^{(2)}_{nk} \\
\hat{\sigma}^2_{b_{nk}} &= 1/\rho^{(2)}_{nk}.
\end{aligned}$$

# Chapter 4

# Measuring cluster stability using the linear bootstrap

Clustering is the canonical unsupervised learning problem, in which we aim to find an assignment of data points to groups, or clusters, that represent meaningful latent structure in a data set. Bayesian nonparametric (BNP) models form a particularly popular set of Bayesian models for clustering due to their flexibility and coherent assessment of uncertainty. As with any Bayesian model of moderate complexity, typically the Bayesian posterior cannot be computed exactly for BNP clustering problems, and an approximation must be employed. Mean-field variational Bayes (MFVB) forms a posterior approximation by solving an optimization problem and is widely used due to its speed (Blei and Jordan, 2006).

An exact BNP posterior might, at least in theory, vary dramatically when presented with different data. Certainly we expect small, rare clusters—which are ubiquitous in BNP—to vary substantially based on the observed data. When reporting the summaries of the clustering for the purposes of scientific inquiry, it behooves us to understand how stable, or alternatively how sensitive, this report is relative to the data (Yu, 2013).

If one were to use the bootstrap to assess stability in this analysis pipeline, it would require a new run of MFVB for each simulated data set. This time cost is often prohibitively expensive, especially for exploratory data analyses. We instead propose to provide a fast, automatic approximation to a full bootstrap analysis based on the infinitesimal jackknife (Jaeckel, 1972; Efron, 1982), which can be seen as a linear approximation to the global stability measure provided by the full bootstrap. This locality can buy drastic time savings, with the infinitesimal jackknife sometimes running orders of magnitude faster than the bootstrap. We here demonstrate how to apply this idea to a data analysis pipeline consisting of an MFVB approximation to a BNP clustering posterior. We show that the necessary calculations can be done nearly automatically, without tedious derivations by a practitioner, using modern automatic differentiation software (Maclaurin et al., 2015). This automation suggests a generality to our methods beyond BNP clustering.

In the remainder of this chapter, we describe the BNP model and MFVB inference in more detail in Section 4.2 and Section 4.3. We review summaries for assessing the output of

our clustering, across which we can in turn assess stability, in Section 4.4. We describe our new stability assessment procedure in Section 4.5. And we demonstrate our ability to quickly and accurately quantify stability in Section 4.6 on an application to clustering time-course gene expression data (Shoemaker et al., 2015; Luan and Li, 2003).

## 4.1 Data

Unsupervised procedures often estimate which data points are clustered together, a quantity of primary importance in many analyses. It can be used to reduce the dimensionality, or to facilitate the interpretation of complex data sets. For example, a genomics experiment assesses cell activity genome-wide, but many genes behave the same way. Clustering genes allows dimensionality reduction that can facilitate interpretation. Finding robust and stable clusters is thus crucial for appropriate downstream analysis.

We focus on the specific task of clustering time course gene-expression data. The differences and evolution over time of gene expression yield important insight on gene regulation of the cell-cycle, or on how cells react to toxins, drugs or viruses. We use a publicly available data set of mice gene expression (Shoemaker et al., 2015). Mice were infected with different influenza viruses, and gene expression was assessed at 14 time points after infection. We focus on the influenza virus "A/California/04/2009", a mildly pathogenic virus from the 2009 pandemic season. We normalize the data as described in Shoemaker et al. (2015). We then apply the differential analysis tool EDGE between the influenza infected mice and control mice (Storey et al., 2005). EDGE yields for each gene a p-value assessing how differently the genes behave between the two conditions. We then rank the genes from most significantly differentially expressed, to least significantly expressed and perform all downstream analysis on the top 1000 genes.

The observed data consists of expression levels $y_{gt}$ for genes $g = 1, ..., n_g$ and time points $t = 1, ..., n_t$ (see Figure 4.1 for a single-gene time course data). The observations are unevenly spaced, with more frequent observations at the beginning. Each gene also has three measurements at each time point, called biological replicates.

## 4.2 Model

As described by Luan and Li (2003), we model the time series as a gene-specific constant additive offset plus a B-spline basis of degree 3 and 7 degrees of freedom. We denote the basis matrix by $X$.

By modeling gene expression as a smooth function, via a B-spline basis, we naturally model the time aspect of the data, as well as provide an easy framework for including biological replicates in the clustering. The reader may observe that the sparse observations at later times leads to apparent non-smoothness in the fitted time series at late times, though the B-splines enforce smoothness in actual calendar time as desired.

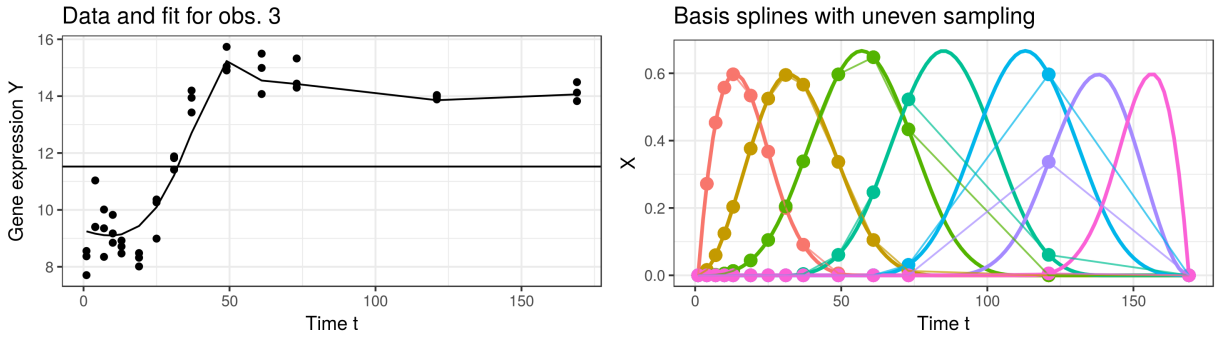



Figure 4.1: Mice gene expression observations over time (left) and B-spline basis of degree 3 and 7 degrees of freedom (right).

Let $y_g$ be the vector of observations $(y_{g1}, ..., y_{gT})^\top$ for gene $g$, $b_g$ denote its additive offset, and $\beta_g$ be its B-spline coeffcients. Denote the variance of the errors as $\sigma^2$ and let $I_T$ be the $n_t \times n_t$ identity matrix. We model the observations as

$$y_g | X, \beta_g, b_g, \sigma^2 \overset{iid}{\sim} \mathcal{N}\left(X\beta_g + b_g, I_T\sigma^2\right) \tag{4.1}$$

Priors on the offsets and error variance are,

$$b_g \overset{iid}{\sim} \mathcal{N}(0, 10) \quad \text{for } n = 1, ...n_g \tag{4.2}$$
$$\sigma^2 \sim \text{Inv.Gamma}(0.10, 0.10). \tag{4.3}$$

We model each gene's B-spline coefficients, $\beta_g$, using a a stick-breaking representation of a Bayesian nonparametric (BNP) Dirichlet process mixture model (Ferguson, 1973; Sethuraman, 1994). First, we draw sticks and coefficients,

$$\nu_k \overset{iid}{\sim} Beta(1, \alpha) \tag{4.4}$$
$$\beta_k \overset{iid}{\sim} \mathcal{N}(0.38, 10), \tag{4.5}$$

for $k = 1, ..., \infty$. We set $\alpha = 2$. Then for each gene, we draw its cluster membership:

$$z_g | \nu \overset{iid}{\sim} \text{Multinomial}(\pi(\nu)), \quad n = 1, ..., n_g, \tag{4.6}$$
$$\text{where } \pi_k(\nu) := \nu_k \prod_{j=1}^{k-1} (1 - \nu_j) \tag{4.7}$$

Finally, the B-spline coefficient for gene $g$ is given by

$$\beta_g | z_g = \sum_{k=1}^{\infty} \beta_k z_{gk}. \tag{4.8}$$

## 4.3 Variational inference

For brevity, use the single vector $\theta$ to represent all the unknown parameters $\nu$, $\beta_k$, $z_{gk}$, $\sigma^2$, and $b_g$, for all $k$ and $g = 1, ..., n_g$. We are interested in the posterior $p(\theta|Y)$, which is intractable. To approximate $p(\theta|Y)$, we form a variational approximation to $p(\theta|Y)$, denoted $q^*(\theta)$ and parameterized by a real-valued parameter $\eta$, using a truncated representation of the BNP prior with $K = 30$ components, which was large enough that more than half of the clusters were essentially unoccupied (Blei and Jordan, 2006). The variational distribution is chosen as a local minimum of the KL divergence from the true posterior:

$$q^*(\theta) := q(\theta|\eta^*) \text{ where } \eta^* := \operatorname{argmin} \text{ KL}(q(\theta|\eta) \,||p(\theta|Y)). \tag{4.9}$$

The variational approximation is

$$q(\theta|\eta) = \delta(\beta)\,\delta(\tau) \prod_{k=1}^{K} \left\{ q(\nu_k) \prod_g q(z_{gk})\, q(b_g|z_{gk}=1) \right\},$$

where $\delta(\cdot)$ denotes a point mass at a parameterized location (Neal and Hinton, 2000) [1], $q(\nu_k)$ is a beta distribution, $q(z_{gk})$ is a multinomial distribution, $q(b_g|z_{gk}=1)$ is a normal distribution, and $\eta$ denotes the vector of parameters for all these distributions.

Ideally, we would like a global minimum of Equation 4.9, but due to the non-convexity of the problem, we can only guarantee finding a local minimum. Importantly for the assessment of co-clustering, knowledge of $\eta^*$ allows us to approximate the posterior probability $\zeta_{gk}(\eta^*) := \mathbb{E}_{q^*}[z_{gk}]$, the posterior probability of gene $g$ belonging to cluster $k$ . We write $\zeta$ without subscripts to refer to the $n_g \times K$ matrix with entries $\zeta_{gk}$.

### 4.3.1 Optimization

Note that by parameterizing $q(b_g, z_g) = \prod_{k=1}^{K} q(b_g|z_{gk}=1)\, q(z_g)$, the updates for $q(b_g, z_g)$ have a closed form given $q(\beta, \tau, \nu)$. Denote the parameters for $q(b_g, z_g)$ as $\eta_{local}$ and the parameters for $q(\beta, \tau, \nu)$ as $\eta_{global}$, and write

$$\hat{\eta}_{local}(\eta_{global}) := \operatorname*{argmin}_{\eta_{local}} \text{KL}(\eta_{global}, \eta_{local}).$$

We can write the optimization problem in Equation 4.9 as a function of $\eta_{global}$ only:

$$\eta_{global} = \operatorname*{argmin}_{\eta_{global}} \text{ KL}(\eta_{global}, \hat{\eta}_{local}(\eta_{global})). \tag{4.10}$$

[1] Technically, a true point mass does not have a well-defined KL divergence with respect to the Lebesgue measure on $\beta$ and $\tau$. But $\delta(\beta; \eta_\beta)$ can be thought of as a density with constant entropy, and where $\mathbb{E}_\delta[\beta] \approx \eta_\beta$. Such a distribution can be approximated arbitrarily closely with a multivariate normal distribution with vanishing variance, for example.

This is valuable because the size of $\eta_{local}$ grows with the number of genes, but the size of $\eta_{global}$ does not.

To solve Equation 4.10, we use a combination of Newton and quasi-Newton methods. We first choose an initialization by fitting individual B-splines to each gene expression, and use K-means to cluster the coefficients; the centroids were used to initialize the variational means for $\beta_K$. From this initialization, we ran BFGS for 300 iterations; at the point where BFGS terminated, we computed the Hessian of the KL objective, Equation 4.10. This Hessian was used as a preconditioner for the final Newton trust region steps, which was iterated to convergence. Hessians were computed using `autograd` (Maclaurin et al., 2015), while BFGS and the newton trust-region routines were done with the `BFGS` and `trust-ncg` methods of `scipy-optimize` (Virtanen et al., 2020), respectively.

### 4.3.2 Auxillary variables

Finally, we introduce some additional notation related to data sensitivity that will be useful to describe the bootstrap and the infinitesimal jackknife in Section 4.5. To assess data sensitivity, we augment our model with scalar per-gene weights, $w_g \geq 0$, where $W = \left(w_1, ..., w_{n_g}\right)^\top$, where we define the weighted likelihood and corresponding optimal variational parameter:

$$\log p\left(Y|\theta, W\right) = \sum_{g=1}^{n_g} w_g \log p\left(y_{g\cdot}|\theta\right)$$

$$\Rightarrow \quad \eta^*\left(W\right) := \text{argmin } \text{KL}\left(q\left(\theta;\eta\right)||p\left(\theta|Y,W\right)\right). \tag{4.11}$$

Defining $W_1 := (1, ..., 1)^\top$ we recover the original variational posterior $\eta^* = \eta^*\left(W_1\right)$. By setting $W$ to other integer-valued vectors, we can produce the effect of removing or repeating datapoints, since $p\left(Y|\theta\right)$ is exchangeable in $y_g$. In particular, by drawing $n_b$ bootstrap weights $W_b \sim \text{Multinomial}\left(n_b, n_b^{-1}\right)$, for $b = 1, ..., n_b$, the bootstrap distribution of a function $\phi\left(\zeta\left(\eta^*\right)\right)$ can be approximated with the draws $\phi\left(\zeta\left(\eta^*\left(W\right)\right)\right)$. In the remainder of the paper, in a slight abuse of notation, we will write $\phi\left(W\right)$ in place of $\phi\left(\zeta\left(\eta^*\left(W\right)\right)\right)$ below when the meaning is clear from the context.

## 4.4 Clutering stability measures

To quantify the stability of a clustering procedure, we must first define measures of similarity between different clustering outputs. In particular, we consider the similarity between the clustering $\zeta = \zeta\left(W_1\right)$, which is clustering at the optimum $\eta^*(W_1)$, and $\tilde{\zeta} := \zeta\left(W_b\right)$ at bootstrap weights $W_b$. We adapt two standard clustering similarity measures, the Fowlkes-Mallows index (Fowlkes and Mallows, 1983) and the normalized mutual information (Vinh et al., 2010). Below, we detail these two measures and how we adapt them to our setting of posterior inference.

First, we take a closer look at the Fowlkes-Mallows index. Ignoring for the moment the variational distribution, consider a general clustering algorithm that outputs binary indicators $z_{gk}$ for gene $g$ belonging to cluster $k$. Suppose two different runs of the algorithm (e.g. runs with two different initializations) give two different outputs $z_{gk}$ and $\tilde{z}_{gk}$. Let $C_{g_1g_2} := \sum_{k=1}^{K} z_{g_1k} z_{g_2k}$ be the indicator that genes $g_1$ and $g_2$ are clustered together under the first clustering; and $\tilde{C}_{g_1g_2}$ denotes the same quantity under the second clustering. Then the Fowlkes-Mallows similarity index is defined as

$$\mathrm{FM}(C, \tilde{C}) = \frac{\sum_{g_1g_2} C_{g_1g_2} \tilde{C}_{g_1g_2}}{\sqrt{(\sum_{g_1g_2} C_{g_1g_2}^2) \cdot (\sum_{g_1g_2} \tilde{C}_{g_1g_2}^2)}}. \tag{4.12}$$

The numerator in Equation 4.12 then counts the number of gene pairs that were co-clustered by both two clustering results, and the denominator normalizes the index to be between 0 and 1; hence, values closer to 1 suggest a more similar clustering.

We modify this definition slightly for our case since we have more than just binary indicators: we have posterior probabilities for $z_{gk}$ approximated by the variational distribution. This then gives the probability of co-clustering under the variational distribution, $\mathbb{E}_{q^*}[C_{g_1g_2}]$. Having two different clustering results now corresponds to having two different variational distributions $q^*$ and $\tilde{q}^*$ for $z$. To measure clustering similarity here, we simply replace $C_{g_1g_2}$ and $\tilde{C}_{g_1g_2}$ in Equation 4.12 with $\mathbb{E}_{q^*}[C_{g_1g_2}]$ and $\mathbb{E}_{\tilde{q}^*}[C_{g_1g_2}]$, their expectations under two different variational distributions. The Folkes-Mallows score for clustering similarity relative to the initial clustering $\zeta$ is

$$\phi_{\mathrm{FM}}\left(\tilde{\zeta}\right) = \frac{\sum\limits_{g_1g_2} \left(\zeta_{g_1}^\top \zeta_{g_2}\right) \left(\tilde{\zeta}_{g_1}^\top \tilde{\zeta}_{g_2}\right)}{\sqrt{\sum\limits_{g_1g_2} \left(\zeta_{g_1}^\top \zeta_{g_2}\right)^2 \cdot \left(\tilde{\zeta}_{g_1}^\top \tilde{\zeta}_{g_2}\right)^2}} \tag{4.13}$$

Now, we turn to the normalized mutual information score. Again, let $q^*$ and $\tilde{q}^*$ be two different variational distributions, with $\mathbb{E}_{q^*}[z_{gk}] := \zeta_{gk}$ and $E_{\tilde{q}^*}[z_{gk}] := \tilde{\zeta}_{gkl}$. Suppose we consider the distribution on labels induced by drawing a random gene $g$, and then drawing the labels $k_1|g \sim q(z_g)$ and $k_2|z_g \sim \tilde{q}(z_g)$. Then define $P(k_1) = \frac{1}{n_g}\sum_g \zeta_{gk}$, the probability of cluster $k_1$ under the first variational distribution, and $\tilde{P}(k_2) = \frac{1}{n_g}\sum_g \tilde{\zeta}_{gk}$, the probability of cluster $k_2$ under the second variational distribution; also let $P(k_1,k_2) = \frac{1}{n_g}\sum_g \zeta_{gk_1}\tilde{\zeta}_{gk_2}$, the joint cluster probabilities. Then the normalized mutual information score for clustering similarity is given by

$$\phi_{\mathrm{MI}}(\tilde{\zeta}) = \frac{\sum_{k_1k_2} P(k_1,k_2) \log(\frac{P(k_1,k_2)}{P(k_1)\tilde{P}(k_2)})}{\sqrt{(\sum_k P(k)\log P(k)) \cdot (\sum_k \tilde{P}(k)\log \tilde{P}(k))}} \tag{4.14}$$

The numerator is the mutual information between the two clustering outputs defined by the variational distributions $q$ and $\tilde{q}$, with a larger mutual information representing more similar clusterings; the denominator then normalizes such that the score is between 0 and 1.

Both measures yield scores ranging between 0 and 1, where the higher the scores, the more similar the clusterings are. Note that while we focus on these two measures, the procedure described below can be applied to any similarity measure $\phi(\zeta)$.

## 4.5 The linear bootstrap

We now derive a local approximation to the bootstrap using the weight notation from Section 4.3. Noting that Equation 4.11 is well-defined even for non-integer values of $W$, and observing that $KL(q(\theta;\eta)||p(\theta|Y,W))$ is smooth in both $\eta$ and $W$ , it follows that $\eta^*(W)$ is smooth in $W$ in a neighborhood of $W_1$. Using the results from Giordano et al. (2018, Appendix D), and adopting the shorthand notation $KL(\eta,W) := KL(q(\theta;\eta)||p(\theta|Y,W))$, we can then calculate a "weight sensitivity matrix" $S$ as

$$S := \left.\frac{d\eta^*(W)}{dW}\right|_{W=W_1} = -\left(\frac{\partial^2 KL(\eta,W)}{\partial\eta\partial\eta^\top}\right)^{-1}\left.\frac{\partial^2 KL(\eta,W)}{\partial\eta\partial W}\right|_{W=W_1}. \tag{4.15}$$

Although Equation 4.15 would be tedious to calculate by hand, it can be calculated exactly using automatic differentiation in just a few lines of code.

Using $S$, and a single-term Taylor expansion, we can approximate $\eta^*(W)$ and, in turn, a clustering metric $\phi(W)$:

$$\eta^*(W) \approx \eta^*_{Lin}(W) := \eta^* + S(W - W_1) \tag{4.16}$$

$$\phi(W) = \phi(\zeta(\eta^*(W))) \approx \phi_{Lin}(W) := \phi(\zeta(\eta^*_{Lin}(W))). \tag{4.17}$$

Note that the quantities $\zeta$, which are probabilities and must lie between 0 and 1, can be expected to be extremely non-linear functions of $\eta^*$, but they can be calculated quickly for any given $\eta^*$. We take advantage of this fact to make a linear approximation only on $\eta^*$ rather than calculating $\frac{dp_{gk}}{dW}$ directly.

This is nearly equivalent to the "infinitesimal jackknife" of Jaeckel (1972) (see also Efron (1982), Chapter 6), where $\eta^*$ is thought of as a statistic depending on the data $y_g$. The only difference is that we linearize $\eta^*$ rather than the full statistic $\phi$. In order to avoid confusion with the jackknife estimator of variance, we will refer to $\phi_{Lin}(W_b)$ as the "linear bootstrap." Note that the right-hand side of Equation 4.16 involves only a matrix multiplication once $S$ has been calculated, but evaluating $\eta^*(W)$ exactly for $W \neq W_1$ typically involves re-solving the optimization problem in Equation 4.11. So although $\phi_{Lin}(W_b)$ is only an approximation, it can generally be calculated much more quickly than $\phi(W)$ (as is shown in Table 4.1 below).

| | $\eta^*$ (200 inits) | $\eta^*_{Cold}$ (10 inits) | $\eta^*_{Warm}$ (1 init) | $\eta^*_{Lin}$ (given $S$) | $S$ |
|---|---|---|---|---|---|
| Time (s): | 16088 | 931 | 53 | 0.0003 | 12 |

Table 4.1: Median times to compute each bootstrap sample (or related quantities)

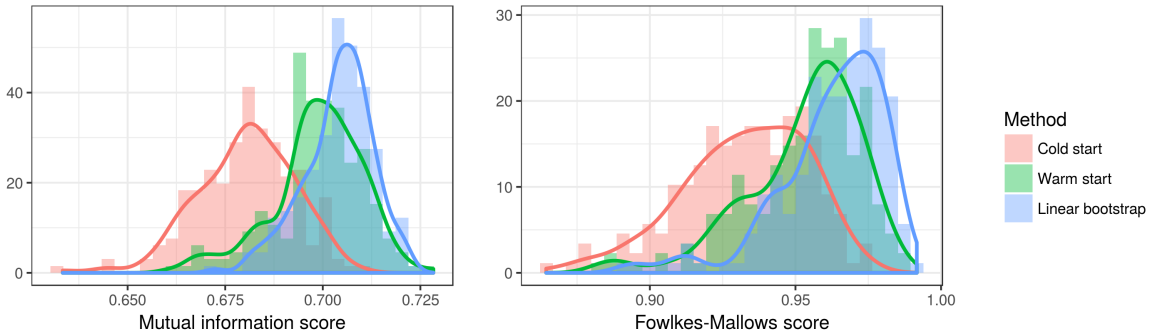



Figure 4.2: Cluster quality

## 4.6 Results

We optimized Equation 4.11 in Python using the `trust-ncg` method of `scipy-optimize` (Virtanen et al., 2020) using an initialization based on K-means. We calculated the necessary derivatives for the optimization and for Equation 4.15 using the automatic differentiation library `autograd` (Maclaurin et al., 2015).

We first found a high-quality optimum for the original dataset (that is, at $W = W_1$) by choosing lowest KL divergence achieved amongst 200 random restarts. We take this optimum to be $\eta^*$, the value at which we calculate the sensitivity $S$ in Equation 4.15. Then, for $n_b = 200$ different bootstrap weights $W_b$, we calculate three different estimates of $\eta^*(W_b)$: "warm starts", $\eta^*_{Warm}(W_b)$, which optimize $\eta^*(W_b)$ starting at $\eta^*$; "cold starts", $\eta^*_{Cold}(W_b)$ which optimize $\eta^*(W_b)$ taking the best of ten new random K-means initializations, and the linear bootstrap estimates, which are $\eta^*_{Lin}(W_b)$ of Equation 4.16. For each of these three estimates of $\eta^*(W_b)$, we compare the bootstrap distribution of the two stability measures detailed in Section 4.4. The median times to calculate each of these measures are given in Table 4.1.

Figure 4.2 shows the distribution of $\phi_{MI}(W_b)$ and $\phi_{FM}(W_b)$. Although the bootstrap based on is $\eta^*_{Lin}(W_b)$ biased slightly upwards relative to both of the actual bootstraps, it is a good approximation to the warm start bootstrap.

Finally, we look at the bootstrap standard deviation of the elements of the matrix $\zeta(W_b)$. Figure 4.3 shows the relationship between the co-clustering standard deviation as measured by $\eta^*_{Warm}(W_b)$ on the x-axis and $\eta^*_{Lin}(W_b)$ or $\eta^*_{Cold}(W_b)$ on the y-axes. Each point in the graph corresponds to a single value of $W_b$, so each graph contains $B = 200$ points. Because the vast majority pairs have very small standard deviation in both measures of the graph, we condition on at least one standard deviation being larger than 0.03. For both the cold start

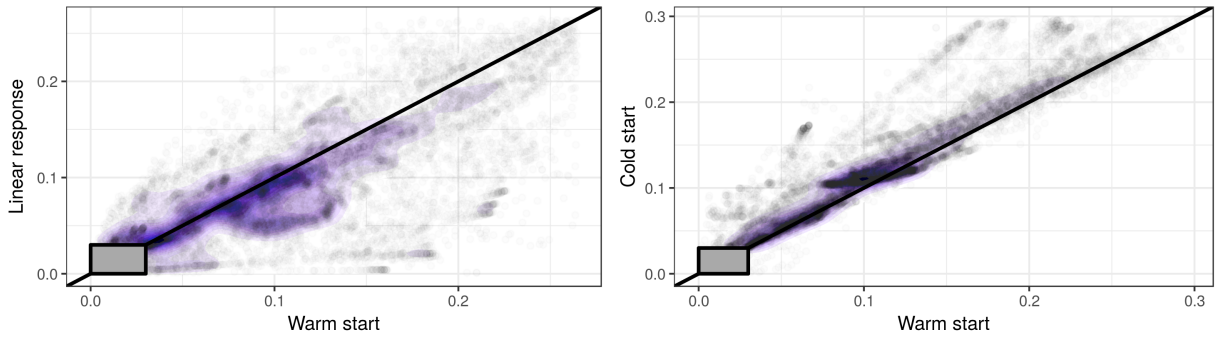



Figure 4.3: Standard deviations of elements of the co-clustering matrix for a randomly selected subset of genes. Pairs with standard deviations $< 0.03$ on both axes are not shown.

and the linear bootstrap, most of the mass lies on the diagonal, indicating a good qualitative correspondence with the warm start, though there is more frequent extreme deviation in the linear bootstrap.

### 4.6.1 Local optima

Many unsupervised clustering problems exhibit multiple local optima in the objective function, even for permutation-invariant quantities like co-clustering measures, and the problem described in the present work is no exception. Measures of uncertainty which are based on local information (like the infinitesimal jackknife) cannot be expected to capture the frequentist variability due to different initializations leading to substantively different local optima. The fact that the cold starts have lower-quality co-clustering than the warm starts in Figure 4.2 indicates that there exist different local optima relatively far from $\eta^*$. In this section, we briefly discuss two additional observations concerning local optima.

One might first ask whether the local optima found by the cold start are much worse than those found by the warm start. The distribution of KL divergences across the bootstrap samples is shown in Figure 4.4. Each point in Figure 4.4 corresponds to two different estimates at the same weights $W_b$, so there are $B = 200$ points in each graph. The linear response KL divergence, which is not evaluated at an actual optimum, is larger than the corresponding optimal value, as expected. Note that the cold start KL divergence is not actually noticeably worse than the warm start KL divergence, suggesting that there may be meaningful frequentist variability due to local optima that is not captured by either $\eta^*_{Warm}(W_b)$ nor $\eta^*_{Lin}(W_b)$.

Finally, we note that the results in Figures 4.2 and 4.3 depend in part on the fact that we are re-starting the optimization in our bootstrap samples at a high-quality optimum, $\eta^*$, chosen as the best out of 200 random restarts. If, instead, we set $\eta^*$ to be the best optimum found after only 10 random restarts, the results are not quite as good, as seen in Figure 4.5. This is probably due both to the base set of cluster assignments, $\zeta$ in Equation 4.13, is not

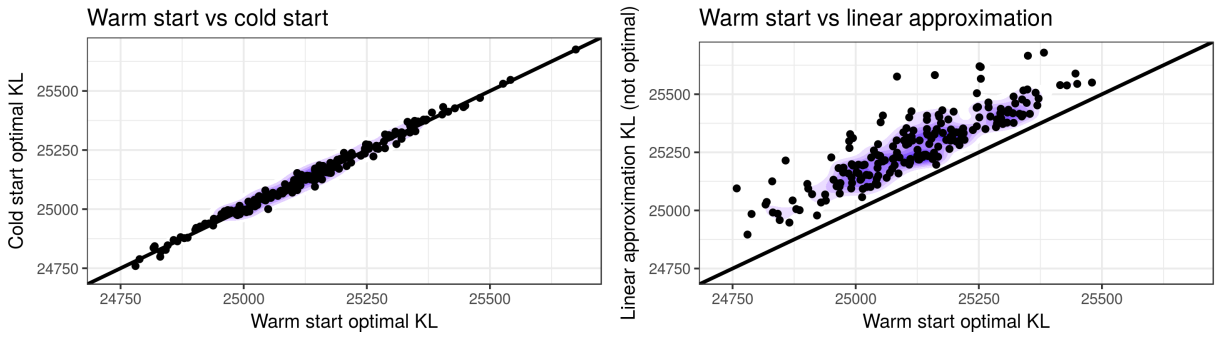



Figure 4.4: Distribution of KL divergence relative to the warm start

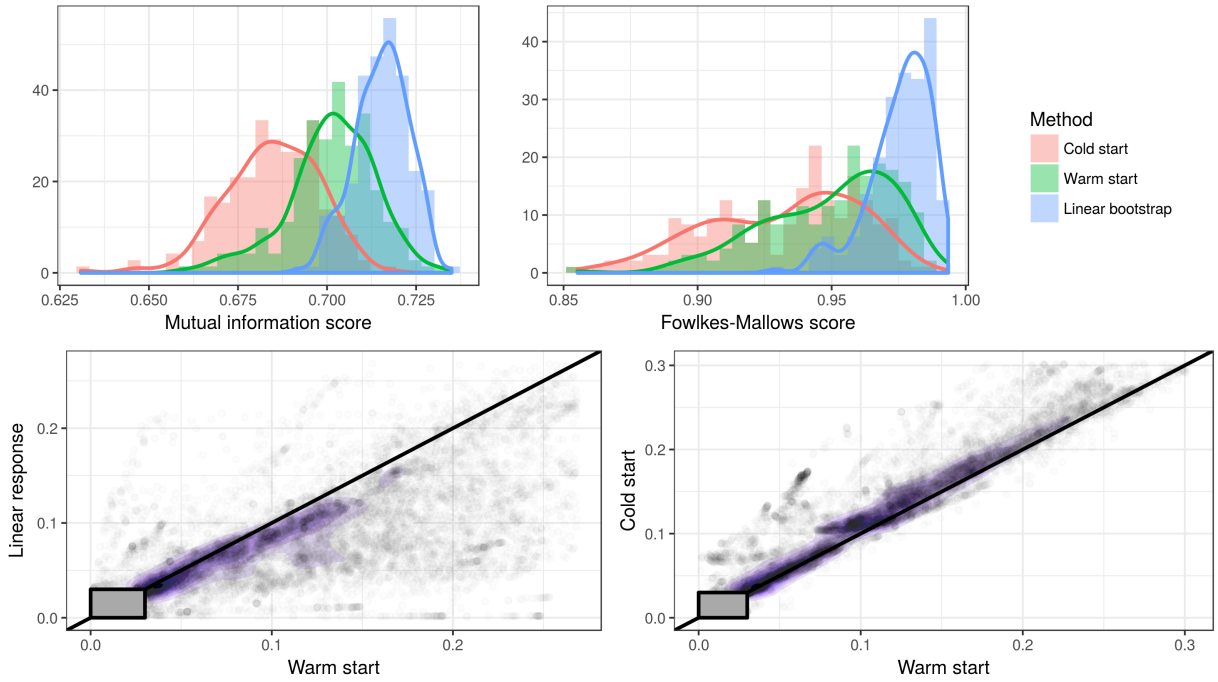



Figure 4.5: Results with an initial optimum based on only 10 random restarts rather than 200

as high-quality an optimum, and to the fact that optima near $\eta^*$, being of lower quality, is chosen less often during the bootstrap procedure.

## 4.7 Conclusion

In this work, we studied the stability of time-course gene expression clustering, using a BNP model and MFVB inference. We compared the bootstrap, a traditional but computationally intensive approach to assess stability with a fast, approximate stability assessment procedure,

the linear bootstrap. Instead of re-sampling the data and refitting the model a large number of times, the linear bootstrap leverages auto-differentiation tools to obtain a first order approximation of the re-sampling scheme. We show that the linear bootstrap is a fast and reasonably accurate alternative to the full bootstrap.

# Chapter 5

# Cross-validation with a Swiss army infinitesimal jackknife

This chapter is a continuation of the previous two chapters, where we used automatic differentiation to evaluate the robustness of the variational posterior. We again work with the mice gene expression data set from Chapter 4. The aim in this chapter is to run cross-validation (CV) in order to select the degrees of freedom for a spline smoother in modeling the time-series of gene expressions.

Cross-validation is a technique that requires repeatedly refitting the model on subsets of the data and evaluating the predictions on the corresponding held out sets. In this subsampling set up, the entries of the weight vector in Equation 4.16 are binary: it is one if the data point appears in the subset, and zero otherwise.

Repeatedly refitting the model can be expensive. In this chapter, we show that the linear approximation can be a viable replacement for exact CV—specifically, the approximation selects the same degrees of freedom as exact CV. Moreover, the linear approximation runs up to an order of magnitude faster than exact CV.

This linear approximation an instance of the infinitesimal jackknife (IJ), a precursor to cross-validation and the bootstrap (Jaeckel, 1972; Efron, 1982). In combination with modern automatic differentiation software, the IJ has found relevance in a wide-range of modern machine learning problems. We dub this method the "Swiss Army infinitesimal jackknife": like the Swiss army knife, the IJ provides several possible functionalities from a single tool. Code and instructions to reproduce our results can be found in the git repository `https://github.com/rgiordan/AISTATS2019SwissArmyIJ`.

## 5.1 Genomics experiment: modeling

We take a slightly different approach to modeling the mice gene expression data (Shoemaker et al., 2015) than in the previous chapter. Our analysis runs in two stages—first, we regress the genes on the spline basis, and then we cluster a transformed version of the regression fits.

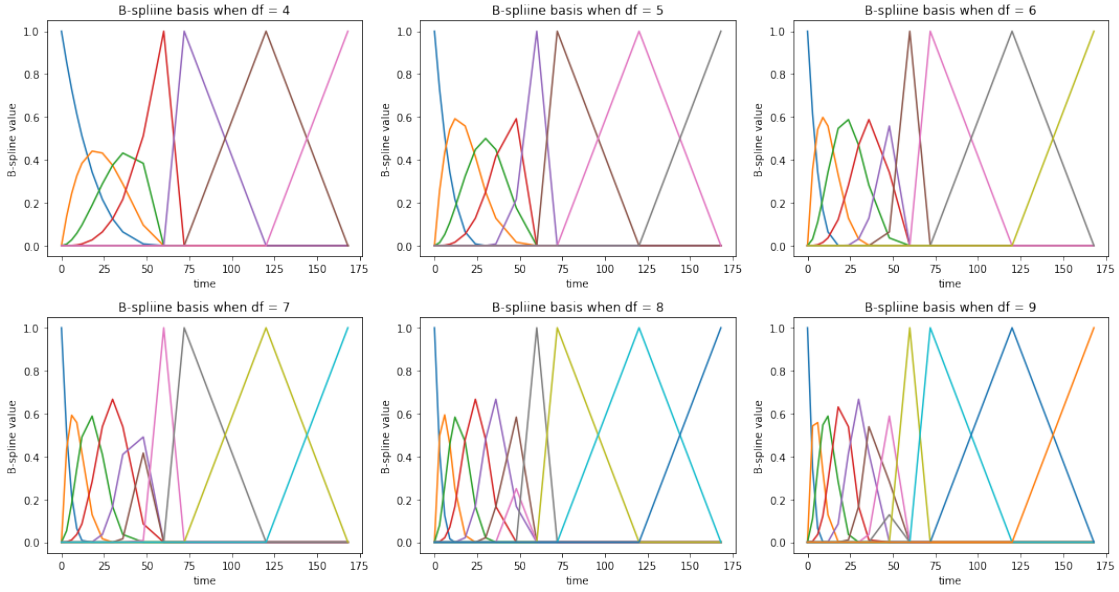



Figure 5.1: B-spline bases with various degrees of freedom. Time is measured in hours.

By modeling in two stages, we both speed up the clustering and allow for the use of flexible transforms of the fits. We are interested in choosing the smoothing parameter using CV on the time points. Both the time points and the smoothing parameter enter the regression objective directly, but they affect the clustering objective only through the optimal regression parameters. Because the optimization proceeds in two stages, the fit is not the optimum of any single objective function. However, it can still be represented as an M-estimator as we detail below.

### 5.1.1 First stage: regression

We model the time course using cubic B-splines (Figure 5.1). Let $\alpha$ be the degrees of freedom of the B-splines, and this is the parameter we seek to choose using cross-validation.

For a given degrees of freedom, the B-spline basis is given by an $n_t \times n_x$ matrix $X$, where the each column of $X$ is a B-spline basis vector evaluated at the $n_t$ timepoints. Note that $n_x$ increases with increasing degrees of freedom.

We only use B-splines to smooth the first 11 timepoints (Figure 5.1). For the last three timepoints, $t = 72, 120, 168$, we use indicator functions on each timepoint as three extra basis vectors. In other words, we append to the regressor matrix three columns, where each column is 1 if $t = 72, 120$, or 168, respectively, and 0 otherwise. We do this to avoid numerical issues in the matrix $X^T X$. Because the later timepoints are more spread out, the B-spline basis are close to zero at the later timepoints, leading to matrices close to being singular.

With the regressor $X$ defined for some B-spline of with degrees of freedom $\alpha$, for each gene $g$ we model $P\left(y_g|\beta_g, \sigma_g^2\right) = \mathcal{N}\left(y_g|X\beta_g, \sigma_g^2\right)$. In the second stage, we will cluster $\beta_g$ taking

into account its uncertainty on each gene. To do this, we wish to estimate the posterior mean $\mathbb{E}[\beta_g|y_g]$ and covariance $\text{Cov}(\beta_g|y_g)$ with flat priors for both $\beta_g$ and $\sigma_g^2$.

For each gene, we estimate the posterior with a mean field variational Bayes (MFVB) approximation $q\left(\sigma_g^2, \beta_g; \hat{\eta}_g\right)$ to the posterior $P\left(\beta_g, \sigma_g^2|y_g\right)$.

In particular, we take $q\left(\sigma_g^2, \beta_g; \hat{\eta}_g\right) = q^*\left(\sigma_g^2\right) q^*\left(\beta_g\right)$, where $q^*\left(\sigma_g^2\right)$ is a dirac delta function, and we optimize over its a location parameter; $q^*\left(\beta_g\right)$ is a Gaussian density and we optimize over its mean and covariance.

The optimal variational approximation has a closed form that is formally identical to the standard frequentist mean and covariance estimate for linear regression. Explicitly, the optimal variational distribution is,

$$\begin{aligned} q^*(\beta_g) &= \mathcal{N}\Big(\beta_g \;\Big|\; (X^TX)^{-1}X^Ty_g,\ \hat{\tau}_g(X^TX)^{-1}\Big) \\ q^*(\sigma_g^2) &= \delta\{\sigma_g^2 = \hat{\tau}_g\}, \end{aligned}$$

where

$$\hat{\tau}_g = \frac{1}{n_t - n_x}\|y_g - X(X^TX)^{-1}X^Ty_g\|_2^2.$$

The advantage of the MVFB construction is that $\hat{\eta}_g$ for $g = 1, ..., n_g$ satisfies set of $n_g$ independent M-estimation objectives, allowing us to apply our infinitesimal jackknife results. Specifically, defining $\theta_{reg} := \left(\eta_1, ..., \eta_{n_g}\right)$, we wish to minimize

$$\begin{aligned} F_{reg}\left(\theta_{reg}, \alpha\right) &= \sum_{g=1}^{n_g} KL\left(q\left(\sigma_g^2, \beta_g; \eta_g\right) ||P\left(\beta_g, \sigma_g^2|y_g\right)\right) \\ &= -\sum_{g=1}^{n_g} \mathbb{E}_q\left[\log P\left(\beta_g, \sigma_g^2|y_g\right)\right] + \mathbb{E}_q\left[\log q\left(\beta_g, \sigma_g^2|\eta_g\right)\right] \\ &:= \sum_{g=1}^{n_g} F_{reg,g}\left(\eta_g, \alpha\right). \end{aligned}$$

Our M-estimator for this first stage is

$$\frac{\partial F_{reg}\left(\theta_{reg}, \alpha\right)}{\partial \theta_{reg}} = 0.$$

### 5.1.2 Second stage: fit a mixture model

We transform the regression coefficients $\beta_g$ before clustering. We are interested in the pattern of gene expression, not the absolute level, so we wish to cluster $\hat{y}_g - \bar{\hat{y}}_g$, where $\bar{\hat{y}}_g$ is the average over time points. Noting that the $n_t \times n_t$ matrix $\text{Cov}_q\left(\hat{y}_g - \bar{\hat{y}}_g\right)$ is rank-deficient because

we have subtracted the mean, the final step is to rotate $\hat{y}_g - \bar{\hat{y}}_g$ into a basis where the zero eigenvector is a principle axis and then drop that component.

Call these transformed regression coefficients $\gamma_g$ and observe that $\mathrm{Cov}_q(\gamma_g)$ has a closed form and is full-rank. It is these $\gamma_g$s that we will cluster in the second stage.

We briefly note that the re-centering operation could have been equivalently achieved by making constant one of the regressors. We chose this implementation because it also allows the user to cluster more complex, non-linear transformations of the regression coefficients, though we leave this extension for future work.

If $T$ is the matrix that effects the transformation, then

$$\begin{aligned}\mathbb{E}_q[\gamma_g] &= T\mathbb{E}_q[\beta_g] \\ \mathrm{Cov}_q(\gamma_q) &= T\mathrm{Cov}_q(\beta_g)T^T.\end{aligned}$$

The transformed parameters are also regression parameters, just in a different space.

We now define a clustering problem for the $\gamma_g$. Let $n_k$ be the number of clusters, and $\mu_1, ..., \mu_{n_k}$ be the cluster centers. Also let $z_{gk}$be the binary indicator for the $g$th gene belonging to cluster $k$. We then define the following generative model

$$\begin{aligned}P(\pi) &= Dirichlet(\omega) \\ P(\mu_k) &= \mathcal{N}(\mu_k|0, \Sigma_0) \quad \text{for} \quad k = 1, ..., n_k \\ P(z_{gk} = 1|\pi_k) &= \pi_k \quad \text{for} \quad k = 1, ..., n_k;\ n = 1, ..., n_g \\ P(\gamma_g|z_{gk} = 1, \mu_k, \eta_g) &= \mathcal{N}(\gamma_g|\mu_k, \mathrm{Cov}_q(\gamma_g) + \epsilon I_{n_t - 1}) \quad \text{for} \quad k = 1, ..., n_k;\ n = 1, ..., n_g.\end{aligned}$$

where $\epsilon$ is a small regularization parameter, which helped our optimization produce more stable results.

We will estimate the clustering using the maximum a posteriori (MAP) estimator of $\theta_{clust} := (\mu, \pi)$. This defines an optimization objective that we seek to minimize:

$$F_{clust}(\theta_{clust}, \theta_{reg}) = -\sum_{g=1}^{n_g} E_{q_z^*}\Big\{\log P(\gamma_g|\eta_g, \mu, \pi, z_g) - \log P(z_g|\pi)\Big\} - \log P(\mu) - \log P(\pi)$$

which, for every value of $\theta_{reg}$, we expect to satisfy

$$\frac{\partial F_{clust}(\theta_{clust}, \theta_{reg})}{\partial \theta_{clust}} = 0.$$

Note that $\theta_{clust}$ involves only the "global" parameters $\mu$ and $\pi$. We did take a variational distribution for the $z_{gk}$s, represented by independent Bernoulli distribution, but the optimal $q_z^*$ can be written as a function of $\mu$ and $\pi$. Hence, our optimization objective only involves these global parameters.

In our experiment, the number of clusters $n_k$ was chosen to be 10. We set $\omega$ to be the ones vector of length $n_k$. The prior info for the cluster centers $\Sigma_0$ is 1e-5$\times I$. $\epsilon$ was set to be 0.1.

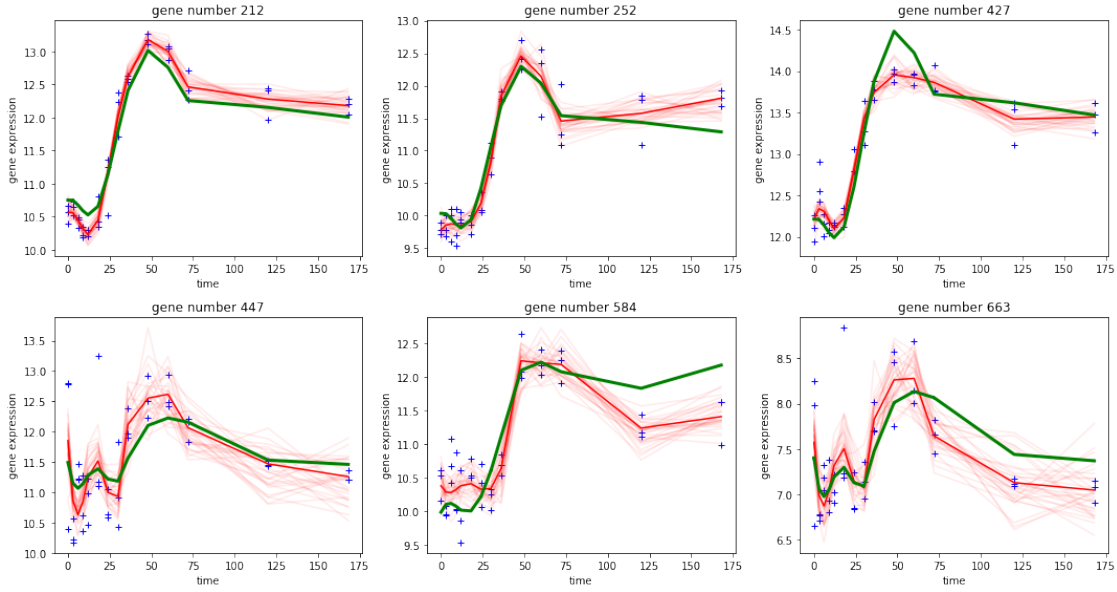



Figure 5.2: Observations from six genes in blue. In red, fits from the first stage regression; light red lines are samples from the approximate posterior. In green, the cluster centroid to which that each gene belongs.

We implemented the model in `scipy` (Virtanen et al., 2020) and computed all derivatives with `autograd` (Maclaurin et al., 2015). In the results below, the match between exact cross-validation and the IJ was considerably improved by using a high-quality second-order optimization method. In particular, for these experiments, we employed the Newton conjugate-gradient trust region method (Nocedal and Wright, 2006, Chapter 7.1) as implemented by the method `trust-ncg` in `scipy.optimize`, preconditioned by the Cholesky decomposition of an inverse Hessian calculated at an initial approximate optimum. The Hessian used for the preconditioner was with respect to the clustering parameters only and so could be calculated quickly, in contrast to the $H_1$ matrix used for the IJ, which includes the regression parameters as well (see below). We found that first-order or quasi-Newton methods (such as BFGS) often got stuck or terminated at points with fairly large gradients. At such points our method does not apply in theory nor, we found, very well in practice.

Figure 5.2 show our model fits. Each gene's regression line has an inferred cluster membership given by $\mathbb{E}_{q_z^*}[z_g]$, and an expected posterior centroid given by $\sum_k \mathbb{E}_{q_z^*}[z_{gk}]\mu_k$. This expected posterior centroid can be un-transformed to give a prediction for the observation (green line in figure 5.2). It is the difference between this prediction line — which is a function of the clustering — and the actual data that we consider to be the "error" of the model.

### 5.1.3 Calculating $H_1$ for the IJ

We seek to choose the degrees of freedom $\alpha$ for the B-splines using cross-validation. We leave out one or more timepoints, and fit using only the remaining timepoints. We then estimate the test error by predicting the value of the genes at the held out timepoints.

To do this, we define time weights $w_t$ by observing that, for each $g$, the term $\mathbb{E}_q\left[\log P\left(\beta_g, \sigma_g^2 | y_g\right)\right]$ decomposes into a sum over time points:

$$F_{reg,g}\left(\eta_g, \alpha, w\right) := -\sum_{t=1}^{n_t} w_t \left(-\frac{1}{2}\sigma_g^{-2}\left(y_{g,t} - \left(X\beta_g\right)_t\right)^2 - \frac{1}{2}\log\sigma_g^2\right) + \mathbb{E}_q\left[\log q\left(\beta_g, \sigma_g^2 | \eta_g\right)\right].$$

We naturally define $F_{reg}\left(\theta_{reg}, \alpha, w\right) := \sum_{g=1}^{n_g} F_{reg,g}\left(\eta_g, \alpha, w\right)$.

By defining $\theta = (\theta_{clust}, \theta_{reg})$, we then have an M-estimator

$$G\left(\theta, w, \alpha\right) := \begin{pmatrix} \frac{\partial F_{reg}(\theta_{reg}, w, \alpha)}{\partial \theta_{reg}} \\ \frac{\partial F_{clust}(\theta_{clust}, \theta_{reg})}{\partial \theta_{clust}} \end{pmatrix} = 0.$$

The corresponding "Hessian" can be computed in blocks:

$$H_1 = \begin{pmatrix} \nabla^2_{\theta_{reg}} F_{reg} & 0 \\ \nabla_{\theta_{reg}}\nabla_{\theta_{clust}} F_{clust} & \nabla^2_{\theta_{clust}} F_{clust} \end{pmatrix}.$$

(Note that what we call the "Hessian" for this two-step procedure is not really a Hessian, as it is not symmetric. More precisely, it is the Jacobian of $G$).

Calculating $H_1$ is the most time-consuming part of the infinitesimal jackknife, since the $H_1$ matrix is quite large (though sparse). However, once $H_1$ is computed, calculating each $\theta_{IJ}(w)$ is extremely fast.

## 5.2 Genomics experiment: results

Figure 5.3 shows that the IJ is a reasonably good approximation to the test set error.[1] In particular, both the IJ and exact CV capture the increase in test error for $df = 8$, which is not present in the training error. Thus we see that, like exact CV, the IJ is able to prevent overfitting. Though the IJ underestimates exact CV, we note that it differs from exact CV by no more than exact CV itself differs from the true quantity of iterest, the test error.

The timing results for the genomics experiment are shown in Figure 5.4. For this particular problem with approximately $39,000$ parameters (the precise number depends on the degrees of freedom), finding the initial optimum takes about 42 seconds. The cost of finding the initial optimum is shared by exact CV and the IJ, and, as shown in Figure 5.4, is a small proportion of both.

[1] In fact, in this case, the IJ is a better predictor of test set error than exact CV. However, the authors have no reason at present to believe that the IJ is a better predictor of test error than exact CV in general.

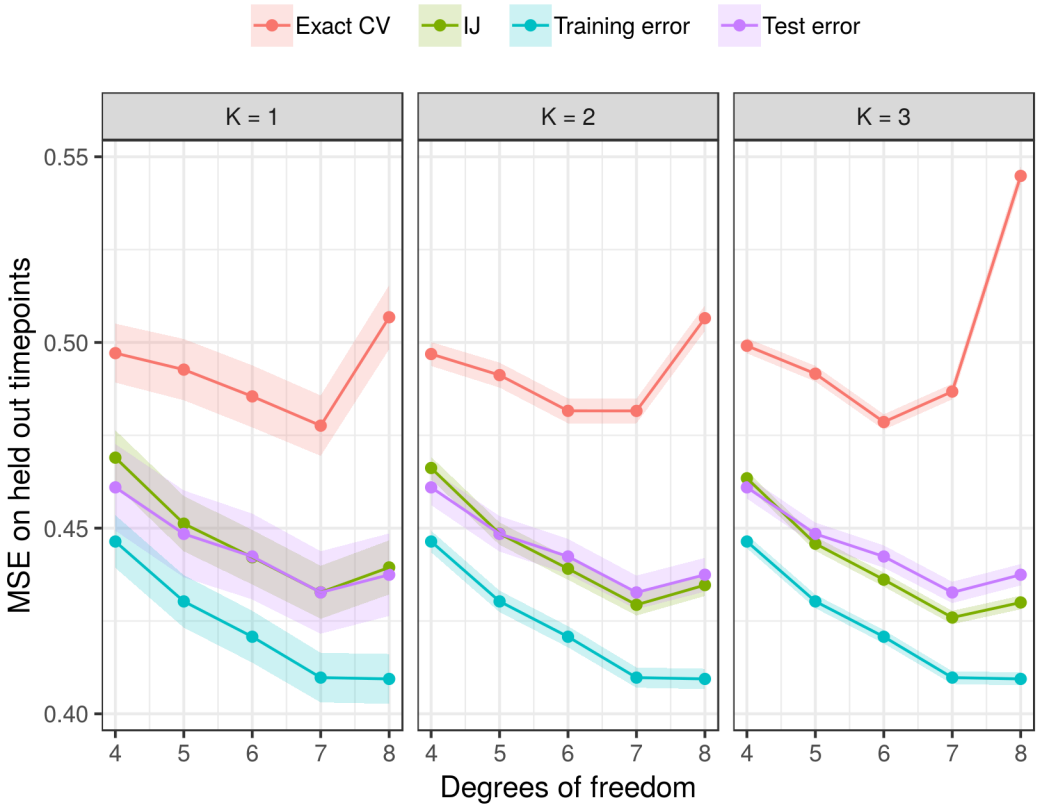



Figure 5.3: Comparison of held-out accuracies. Here $k$ refers to the number of data points left out.

The principle time cost of the IJ is the computation of $H_1$. Computing and inverting a dense matrix of size $39,000$ would be computationally prohibitive. But, for the regression objective, $H_1$ is extremely sparse and block diagonal, so computing $H_1$ in this case took only around 360 seconds. Inverting $H_1$ took negligible time. Once we have $H_1^{-1}$, obtaining the subsequent IJ approximations is nearly instantaneous.

The cost of refitting the model for exact CV varies by degrees of freedom (increasing degrees of freedom increases the number of parameters) and the number of left-out points (an increasing number of left-out datapoints increases the number of refits). As can be seen in Figure 5.4, for low degrees of freedom and few left-out points, the cost of re-optimizing is approximately the same as the cost of computing $H_1$. However, as the degrees of freedom and number of left-out points grow, the cost of exact CV increases to as much as an order of magnitude more than that of the IJ.

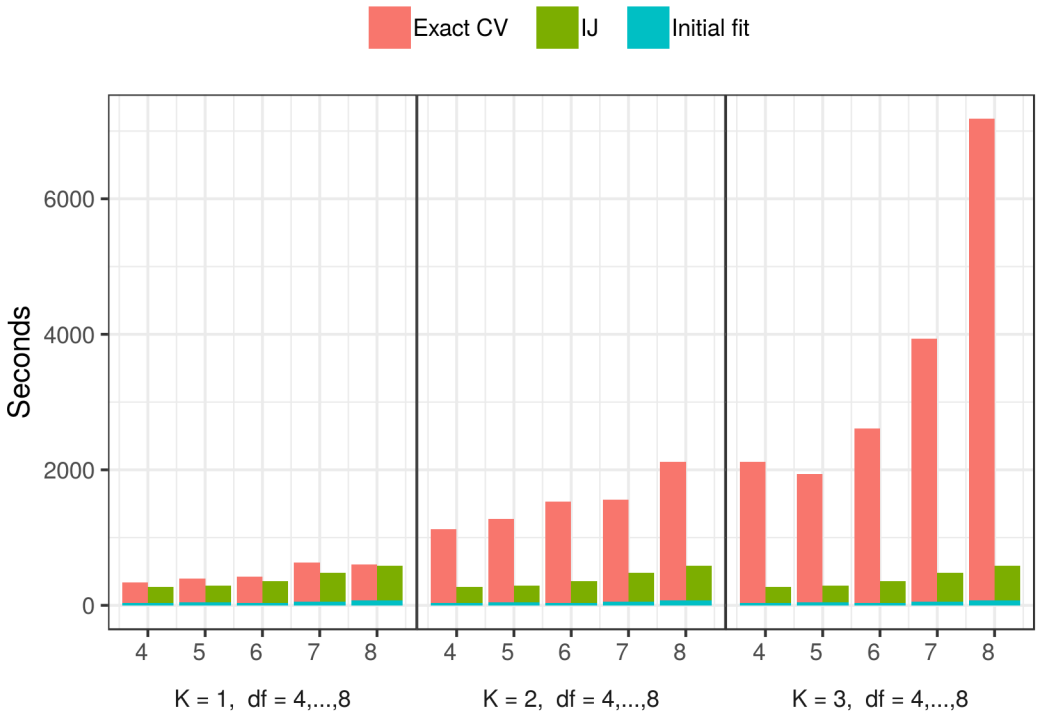



Figure 5.4: Comparison of compute time.

## 5.3 Conclusion

In this chapter, we demonstrated the performace of the Swiss army infinitesimal jackknife on a time series dataset of gene expressions, where we wish to use cross-validation to select the degrees of freedom for a spline smoother. The IJ provides a reasonably accurate approximation to exact CV, in that it selected the same degrees of freedom as exact CV. Importantly, the IJ runs up to an order of magnitude faster. Modern automatic differentiation renders many past practical difficulties with the IJ obsolete, and we recommend consideration of the Swiss Army infinitesimal jackknife for modern machine learning problems.

# Chapter 6

# Rao-Blackwellized stochastic gradients for discrete distributions

Models with discrete latent variables appear in several applications from previous chapters. In cataloging astronomical surveys, the number of light sources present in an image was a discrete latent variable. In clustering gene expressions, a discrete latent variable encoded the cluster memberships in a mixture model.

This chapter considers the problem of minimizing objectives with the form

$$\mathcal{L}(\eta) := \mathbb{E}_{z\sim q_\eta(z)}\left[f_\eta(z)\right] = \sum_{k=1}^{K} q_\eta(k) f_\eta(k), \tag{6.1}$$

where $z$ is a discrete random variable over $K \leq \infty$ categories, with distribution $q_\eta(z)$ parameterized by a real vector $\eta$ and differentiable in $\eta$. In general, we allow the real-valued integrand $f_\eta$ to also depend differentiably on $\eta$. The ELBO is of this form, with $f_\eta(z) = \log p(x, z) - \log q_\eta(z)$.

If $K$ is finite and small enough, we can compute the exact gradient as

$$\nabla_\eta \mathbb{E}_{q_\eta(z)}[f_\eta(z)] = \sum_{k=1}^{K} \Big\{ \left[\nabla_\eta q_\eta(k)\right] f_\eta(k) + q_\eta(k) \nabla_\eta f_\eta(k) \Big\}. \tag{6.2}$$

On the other hand, $K$ may be infinite, or large relative to the cost of evaluating $q_\eta \cdot f_\eta$. In either of these cases, which are the focus of this chapter, the exact gradient is computationally intractable. Thus, in order to optimize $\mathcal{L}(\eta)$, we seek low-variance stochastic approximations of the gradient.

The "reparametrization trick" (Spall, 2003; Kingma and Welling, 2013; Rezende et al., 2014) provides efficient stochastic gradients when $q_\eta$ is a continuous distribution, but it does not apply when $z$ is discrete. Two well-known possibilities in the discrete case are continuous relaxation (Maddison et al., 2017; Jang et al., 2017) and REINFORCE (Williams, 1992), also known as the score function estimator. The former replaces the discrete random variable

with a continuous relaxation so that the reparametrization trick can be applied. However, it results in biased gradient estimates. The latter is impractical for most purposes due to its high variance.

Control variate methodology provides a general framework for variance reduction. Specific examples include RELAX (Grathwohl et al., 2018), REBAR (Tucker et al., 2017), NVIL (Mnih and Gregor, 2014), and MuProp (Gu et al., 2016). These methods provide a mechanism for reducing the variance of REINFORCE, but unfortunately they do not reduce the variance enough for many applications.

In the current paper, we show how to achieve further variance reduction via a meta-procedure that can be applied to any discrete-distribution stochastic-gradient procedure (e.g., REINFORCE or REINFORCE with control variate). Our framework reduces variance without changing the bias. In particular, an unbiased stochastic gradient remains unbiased after application of our approach. Further, our approach is "anytime" in the sense that it can reduce stochastic-gradient variances given any computational budget—the larger the budget, the greater the variance reduction. Hence it is well suited to our chosen setting, where $K$ is infinite or very large, and/or $q_\eta \cdot f_\eta$ is expensive to evaluate.

Our method is particularly apt in the setting where the probability mass $q_\eta(z)$ is concentrated on only a few categories. For example, in extreme classification, only a few labels out of many are plausible. In reinforcement learning, only a few actions in the possible action space are advantageous. Neither control-variate methods nor continuous-relaxation techniques take advantage of this "sparsity," and we show that the variance reduction of our method in this setting can be dramatic.

We show that our variance-reduction meta-procedure is an instance of a general statistical method called Rao-Blackwellization (Casella and Robert, 1996). Rao-Blackwellization has been used in previous work to reduce the variance of stochastic gradients (Ranganath et al., 2013; Titsias and Lázaro-Gredilla, 2015), but in a setting orthogonal to ours, one with multivariate discrete random variables. Our focus here is on a univariate discrete random variable with many categories. Our method can be applied in conjunction with the former work to extend to the case of multivariate discrete random variables, each with a large number of categories. This extension is not discussed in the present work, and we leave it as an avenue of future exploration.

This chapter is organized as follows. We present our variance-reduction procedure in Section 6.1 and make the connection to Rao-Blackwellization in Section 6.2, demonstrating that our technique necessarily reduces stochastic-gradient variances. In Section 6.3 we discuss related work. In Section 6.4, we exhibit the benefits of our procedure on synthetic data, a semi-supervised classification problem, and a pixel attention task. We conclude in Section 6.5.

## 6.1 Method

We consider the situation where the number of categories $K$ is infinite, or very large in the sense that computing the exact gradient in Equation 6.2 is intractable. One possible stochastic estimator for the gradient is the REINFORCE estimator,

$$f_\eta(z)\nabla_\eta \log q_\eta(z) + \nabla_\eta f_\eta(z) \quad z \sim q_\eta(z), \tag{6.3}$$

which one can check is unbiased for the true gradient in Equation 6.2.

In practice, the REINFORCE estimator often has variance too large to be useful. Control variates have been proposed to decrease the variance of the REINFORCE estimator. The key observation is that the score function $\nabla_\eta \log q_\eta(z)$ has zero expectation under $q_\eta(z)$, so

$$[f_\eta(z) - C]\nabla_\eta \log q_\eta(z) + \nabla_\eta f_\eta(z) \quad z \sim q_\eta(z) \tag{6.4}$$

is still unbiased for the true gradient. Several proposals have been put forth for choosing $C$ to reduce the variance (Mnih and Gregor, 2014; Gu et al., 2016; Tucker et al., 2017).

In this paper, we present a meta-procedure that can be applied to any stochastic estimator for the gradient of a discrete expectation obtained by sampling from $q_\eta(z)$. Let $g(z)$ be any such estimator which is unbiased[1], i.e., satisfies $\mathbb{E}_{q_\eta(z)}[g(z)] = \nabla_\eta \mathbb{E}_{q_\eta(z)}[f_\eta(z)]$. An example is the REINFORCE estimator. We decompose the expectation $\mathbb{E}_{q_\eta(z)}[g(z)]$ into two components: one containing the high-probability atoms of $q_\eta$, and one containing the remaining atoms. We compute the exact contribution of the high-probability component to the expectation, and we use a stochastic estimator for the other component. The idea comes from observing that in many applications, $q_\eta(z)$ only puts significant mass on a few categories. If $g(z)$ is reasonably well behaved over $z$, then $q_\eta(z)g(z)$ is large when $q_\eta(z)$ attains its largest values and smaller elsewhere. By computing the high-probability component of the expectation exactly, we obtain a value already close to correct. A stochastic estimator is then added to correct, on average, for what error remains.

Formally, let $\mathcal{C}_k$ be the set of $z$ such that $q_\eta(z)$ assumes one of its $k$ largest values. Ties may be broken arbitrarily. Let $\bar{\mathcal{C}}_k$ denote the complement of $\mathcal{C}_k$. Then

$$\begin{aligned}
\nabla_\eta \mathbb{E}_{q_\eta(z)}[f_\eta(z)] &= \mathbb{E}_{q_\eta(z)}[g(z)] && (6.5)\\
&= \mathbb{E}_{q_\eta(z)}[g(z)\mathbb{I}\{z \in \mathcal{C}_k\} + g(z)\mathbb{I}\{z \in \bar{\mathcal{C}}_k\}] && (6.6)\\
&= \sum_{z \in \mathcal{C}_k} q_\eta(z)g(z) + \mathbb{E}_{q_\eta(z)}[g(z)\mathbb{I}\{z \in \bar{\mathcal{C}}_k\}]. && (6.7)
\end{aligned}$$

It remains to approximate the expectation in the second term. We use an importance-sampling approximation based on a single draw from an importance distribution. We choose a simple importance distribution: the distribution of $q_\eta$ conditional on the event $\bar{\mathcal{C}}_k$. We denote this importance distribution by $q_\eta|_{\bar{\mathcal{C}}_k}$. By construction, the importance weighting

[1] Our technique applies to biased estimators as well. For concreteness, we focus on the unbiased case.

function is identically equal to $q_\eta(\bar{\mathcal{C}}_k)$, regardless of which $z \sim q_\eta|_{\bar{\mathcal{C}}_k}$ we draw. (Note that the indicator inside the second term of Equation 6.7 always equals one, because we are only sampling from $z \in \bar{\mathcal{C}}_k$.)

Our estimator assumes that, given $k$, the set $\mathcal{C}_k$ can be identified at little cost. This certainly holds in the case of inference: using variational Bayes, $q(z)$ is a variational approximate posterior chosen from a set we designate.

In summary, we estimate the gradient as

$$\hat{g}(v) = \sum_{z \in \mathcal{C}_k} q_\eta(z) g(z) + q_\eta(\bar{\mathcal{C}}_k) g(v) \quad \text{where } v \sim q_\eta|_{\bar{\mathcal{C}}_k} \tag{6.8}$$

which also satisfies $\mathbb{E}_v[\hat{g}(v)] = \nabla_\eta \mathbb{E}_{q_\eta(z)}[f_\eta(z)]$.

We see that the first term of this estimator is deterministic and the second term is random, but scaled by $q_\eta(\bar{\mathcal{C}}_k)$, which is small when $q_\eta$ is concentrated on a small number of atoms. Therefore, we intuitively expect this estimator to have smaller variance than the original estimator, $g(z)$.

In the next section, we confirm this intuition by interpreting the construction of the estimator $\hat{g}(v)$ as Rao-Blackwellization (which always reduces variance). Hence, we call $\hat{g}(v)$ the *Rao-Blackwellized gradient estimator*.

## 6.2 Theory

We begin by describing how a suitable representation of the original discrete variable $z \sim q_\eta(z)$ allows us to interpret our estimator as an instance of Rao-Blackwellization. Let $q_\eta|_{\mathcal{C}_k}$ denote the distribution of $q_\eta$ conditional on the event $C_k$. Consider the three independent random variables

$$u \sim q_\eta|_{\mathcal{C}_k}, \tag{6.9}$$

$$v \sim q_\eta|_{\bar{\mathcal{C}}_k}, \tag{6.10}$$

$$\text{and} \quad b \sim \text{Bernoulli}\left(q_\eta(\bar{\mathcal{C}}_k)\right). \tag{6.11}$$

The triplet $(u, v, b)$ provides a distributionally equivalent representation of $z$:

$$T(u, v, b) \stackrel{d}{=} z, \tag{6.12}$$

where

$$T(u, v, b) := u^{1-b} v^b. \tag{6.13}$$

The estimator in Equation 6.8 can then be written as

$$\hat{g}(v) = \mathbb{E}\left[g(T(u, v, b))|v\right], \tag{6.14}$$

where $g(z)$ is the original unbiased gradient estimator. To see this, break the right-hand side of Equation 6.14 into two terms according to the value of $b$, then simplify. Equation 6.14 demonstrates directly that our estimator is an instance of Rao-Blackwellization. As such, it has the same expectation as the original estimator $g(z)$, a fact about Rao-Blackwellization that follows immediately from iterated expectation. In particular, if $g(z)$ is unbiased as we have assumed, so too is our estimator.

An application of the conditional variance decomposition gives

$$\text{Var}\left[g(z)\right] = \text{Var}\left[\hat{g}(v)\right] + \mathbb{E}\left\{\text{Var}\left[g(T(u,v,b))|v\right]\right\}, \tag{6.15}$$

showing that $\hat{g}$ has lower variance than $g$, by at least as much as the last term in Equation 6.15. This too is a standard result about Rao-Blackwellization.

Proposition 1 further quantifies this variance reduction, showing the variance of $\hat{g}(v)$ must be less then the variance of $g(v)$ by the multiplicative factor $q_\eta(\bar{\mathcal{C}}_k)$.

**Proposition 1.** *Let $g(z)$ be an unbiased gradient estimator as in Equation 6.5 and $\hat{g}(v)$ denote the Rao-Blackwellized estimator defined in Equation 6.8. Then*

$$\text{Var}[\hat{g}(v)] \leq q_\eta(\bar{\mathcal{C}}_k)\text{Var}[g(z)]. \tag{6.16}$$

*Proof.* We apply the conditional variance decomposition. Let $\epsilon = q_\eta(\bar{\mathcal{C}}_k)$ and recall the Bernoulli random variable $b$ defined in Equation 6.11. First,

$$\begin{aligned}
\text{Var}[g(z)] &= \mathbb{E}[\text{Var}[g(z)|b]] + \text{Var}[\mathbb{E}[g(z)|b]] && (6.17)\\
&\geq \mathbb{E}[\text{Var}[g(z)|b]] && (6.18)\\
&= \epsilon\text{Var}[g(z)|z \in \bar{\mathcal{C}}_k] + (1-\epsilon)\text{Var}[g(z)|z \in \mathcal{C}_k]\\
&\geq \epsilon\text{Var}[g(z)|z \in \bar{\mathcal{C}}_k].
\end{aligned}$$

But $\text{Var}[\hat{g}(v)] = \epsilon^2\text{Var}[g(z)|z \in \bar{\mathcal{C}}_k]$, which in combination with the above yields the result. □

The multiplicative factor of variance reduction guaranteed by Rao-Blackwellization can be significant if the probability mass of $q_\eta(z)$ is concentrated on just a few categories. But while Rao-Blackwellization reduces the variance of $g(z)$, this comes at the cost of evaluating $g(z)$ a total $k+1$ times (cf. Equation 6.8). An initial stochastic gradient $g(z)$ such as REINFORCE will only require a single evaluation of $g$.

There is an alternative approach to reducing the variance of an initial estimator $g(z)$ via multiple evaluations of $g(z)$: minibatching, i.e., simple Monte-Carlo averaging over independent draws of $z$. Thus, the question arises: given a budget of $N < K$ evaluations of $g(z)$, is it better to Rao-Blackwellize or minibatch? Computationally, our method is parallelizable in the same way that minibatching is parallelizable. The next proposition shows constructively that there is a choice of $k \leq N$ for which Rao-Blackwellization reduces variance at least as much as minibatching.

**Proposition 2.** *Suppose we have a budget of $N$ evaluations of $g$. Consider the estimators*

$$\hat{g}_{N,k}(v) := \sum_{u \in \mathcal{C}_k} q_\eta(u) g(u) + \frac{q_\eta(\bar{\mathcal{C}}_k)}{N-k} \sum_{j=1}^{N-k} g(v_j), \tag{6.19}$$
$$v_1, ..., v_{N-k} \overset{iid}{\sim} q_\eta|_{\bar{\mathcal{C}}_k}$$

*and*

$$g_N(z) := \frac{1}{N} \sum_{j=1}^{N} g(z_j), \quad z_1, ..., z_N \overset{iid}{\sim} q_\eta. \tag{6.20}$$

*If we choose*

$$\hat{k} = \underset{k \in \{0,...,N\}}{\operatorname{argmin}} \frac{q_\eta(\bar{\mathcal{C}}_k)}{N-k} \tag{6.21}$$

*then* $\mathrm{Var}[\hat{g}_{N,\hat{k}}(v)] \leq \mathrm{Var}[g_N(z)]$.

*Proof.* Let $V_1 = \mathrm{Var}[g_1(z)]$. Then $\mathrm{Var}[g_N(z)] = (1/N)V_1$, while Proposition 1 shows that $\mathrm{Var}[\hat{g}_{N,k}(v)] \leq \frac{q_\eta(\bar{\mathcal{C}}_k)}{N-k} V_1$. Since $\frac{q_\eta(\bar{\mathcal{C}}_k)}{N-k} = \frac{1}{N}$ when $k = 0$, the result follows. □

Together, Propositions 1 and 2 imply the following:

- Rao-Blackwellization leads to a significant variance reduction if the mass of $q_\eta(z)$ is concentrated.
- Even when restricting minibatched versions of the initial and Rao-Blackwellized estimators to an equal number of evaluations of $g$, Rao-Blackwellization yields equal or lower variance, for a computable choice of $k$.

## 6.3 Related Work

Methods to reduce the variance of stochastic gradients for discrete distributions generally fall into two broad categories: continuous relaxation methods and control variate methods.

In the first category, the Concrete distribution (Maddison et al., 2017) approximates the discrete random variable with a reparametrizable continuous random variable so that the standard reparametrization trick can be applied. While this continuous relaxation reduces the variance of the stochastic gradient, the resulting estimators are biased. Thus the Gumbel-softmax procedure (Jang et al., 2017) introduced an annealing step into the optimization whereby the continuous relaxation converges towards the discrete random variable as the optimization path moves forward.

In the second category, control variate methods include black-box variational inference (BBVI) (Ranganath et al., 2013), NVIL (Mnih and Gregor, 2014), DARN (Gregor et al.,

2014), and MuProp (Gu et al., 2016). BBVI uses multiple samples at each step to estimate the 'optimal' control variate. NVIL introduces an observation dependent control variate learned by a separate neural network. DARN uses a Taylor expansion of $f_\eta(z)$ to compute a control variate, but this results in a biased estimator; MuProp proposes an estimate of this bias and corrects it.

Finally, RELAX (Grathwohl et al., 2018) and REBAR (Tucker et al., 2017) are a combination of the two broad methods and use a continuous relaxation to construct a control variate.

Section 6.4 compares both continuous relaxation and control variate methods to our Rao-Blackwellization.

A Rao-Blackwellization procedure for gradient estimation was also applied in BBVI and "local expectation gradients" (Titsias and Lázaro-Gredilla, 2015), but for a different purpose. In their setting, the expectation is decomposed over a multivariate (discrete or continuous) random variable using iterated expectation. BBVI approximates each conditional expectation by sampling (with a control variate), while local expectation gradients compute each conditional expectation analytically. This Rao-Blackwellization is orthogonal to our approach: while they consider multiple discrete random variables, our approach focuses on a univariate discrete with many categories.

The process of summing out a few terms and sampling the remainder for gradient estimation has appeared in the context of reinforcement learning (Titsias, 2014; Liang et al., 2018), though to our knowledge we are the first to make the connection with Rao-Blackwellization. In MAPO (Liang et al., 2018), a procedure to create a memory buffer of trajectories for policy optimization, the terms with high rewards (or small loss) are kept and summed. In contrast, we choose to sum terms with high probability. In our setting, it is the loss $f_\eta(z)$, not the probability, $q_\eta(z)$, that is expensive to evaluate for all categories $z$.

Finally, the problem of having a large number of categories also manifests in language models, and methods such as noise contrastive estimation (Gutmann and Hyvärinen, 2012) and hierarchical softmax (Morin and Bengio, 2005) have been introduced. However, these methods are applied when the normalizing constant for $q_\eta(z)$ is intractable. In our work, we restrict ourselves to scenarios where $q_\eta(z)$ is normalized.

## 6.4 Experiments

In our experiments, we will consider applying the Rao-Blackwellization procedure to either the REINFORCE estimator,

$$g(z) = f_\eta(z)\nabla_\eta \log q_\eta(z) + \nabla_\eta f_\eta(z), \quad z \sim q_\eta(z), \tag{6.22}$$

or REINFORCE with a control variate $C$,

$$g(z) = [f_\eta(z) - C]\nabla_\eta \log q_\eta(z) + \nabla_\eta f_\eta(z), \quad z \sim q_\eta(z). \tag{6.23}$$

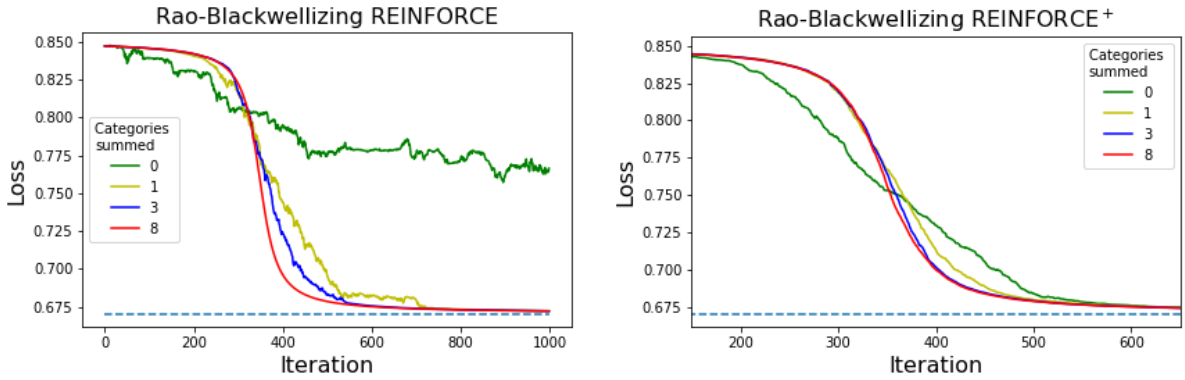



Figure 6.1: The loss function at each iteration in the Bernoulli experiments. Each line is an average over 20 trials from the same initialization. Zero categories summed is the original estimator, while eight categories summed returns the exact gradient.

A simple choice of control variate that works well in practice is to take $C = f_\eta(z')$ for an independent draw $z' \sim q_\eta$. We abbreviate this estimator as REINFORCE$^+$.

Note that in both REINFORCE and REINFORCE$^+$, $g(z)$ is unbiased for the true gradient. (In the second case, $g(z)$ is unbiased conditional on $z'$, and hence unconditionally unbiased as well.)

### 6.4.1 Bernoulli latent variables

We fix a vector $p = [0.6, 0.51, 0.48]^\top$ and seek to minimize the loss function

$$\mathbb{E}_{b_1,b_2,b_3 \overset{iid}{\sim} \mathrm{Bern}(\sigma(\eta))} \left\{ \sum_{i=1}^{3} (b_i - p_i)^2 \right\} \tag{6.24}$$

over $\eta \in \mathbb{R}$, where $\sigma(\eta)$ is the sigmoid function. Here, the discrete random vector $b = [b_1, b_2, b_3]^\top$ is supported over $K = 2^3 = 8$ categories. The optimal value of $\sigma(\eta)$ is 1, approached as $\eta \to \infty$.

Figure 6.1 shows the performance of Rao-Blackwellizing REINFORCE and REINFORCE$^+$. We initialized $\eta$ at $\eta = -4$, so the sampling distribution has large mass at $b = (0, 0, 0)$. The optimal distribution on the other hand should put all mass at $b = (1, 1, 1)$. In other words, we initialized the optimization procedure such that the mass is concentrated on the wrong point. The Rao-Blackwellized gradient is therefore initially slightly slower than the original gradient, since we are analytically summing the wrong category. However, Rao-Blackwellization improves the performance of both gradient estimators at the end of the path.

Figure 6.2 shows the variances of the gradient estimates at $\eta = 0$ and $\eta = -4$, as a function of $k$, the categories analytically summed. As expected, the variance decreases as more categories are analytically summed. At $\eta = 0$, the corresponding $q_\eta$ distribution is uniform, i.e., maximally anti-concentrated, so the variance reduction of Rao-Blackwellization

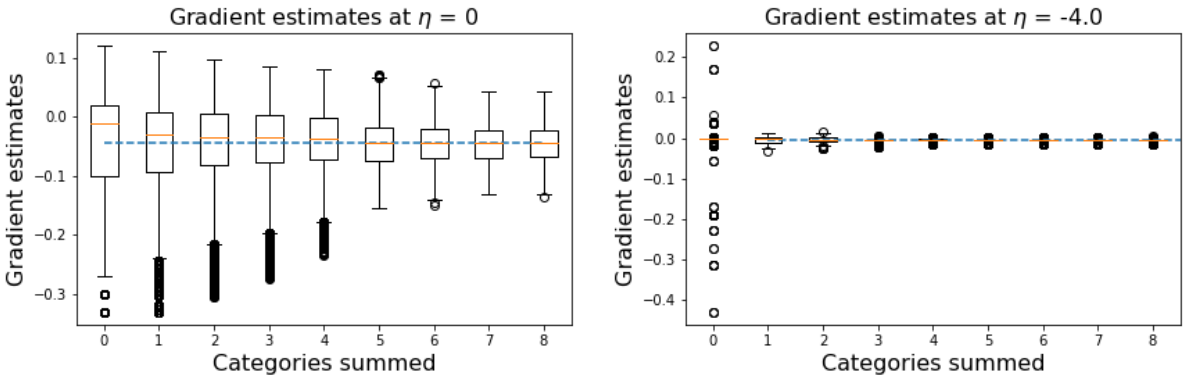



Figure 6.2: The distribution of gradient estimates from REINFORCE$^+$ in the Bernoulli experiments. We examine the gradients at $\eta = 0$ and $\eta = -4$, as a function of $k$, the number of categories summed. Summing out categories reduces variance. The reduction is large at $\eta = -4$ where the variational distribution is concentrated on just one category. (Note there is still some random noise when we sum out all 8 categories here, because of the random control variate.)

is not large. However, the gains are quite substantial at $\eta = -4$, where $q_\eta$ is concentrated around the point $b = (0, 0, 0)$. In this case, analytically summing out one category removes nearly all the variance.

### 6.4.2 Gaussian mixture model

For our next experiment, we draw $N = 200$ observations $(y_n)$ from a $d$-dimensional Gaussian mixture model with $K = 10$ components, taking $d = 2$. The generative model is,

$$z_n \overset{\text{iid}}{\sim} \text{Categorical}(\pi_{1:K}),\ n = 1, \ldots, N, \tag{6.25}$$

$$\mu_k \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_0^2 I_{d\times d}),\ k = 1, \ldots, K, \tag{6.26}$$

$$y_n | z_n, \mu \overset{\text{iid}}{\sim} \mathcal{N}(\mu_{z_n}, \sigma_y^2 I_{d\times d}),\ n = 1, \ldots, N. \tag{6.27}$$

Here each $\mu_k$ is a Gaussian centroid and each $z_n$ is a cluster membership indicator.

As exact inference of the posterior $p(\mu, z|y)$ is intractable, we approximate it variationally (Blei et al., 2017) with the mean-field family

$$q(\mu,z) = \prod_{k=1}^{K} q(\mu_k) \prod_{n=1}^{N} q(z_n). \tag{6.28}$$

Here

$$q(\mu_k) = \delta\{\mu_k = \hat{\mu}_k\}, \tag{6.29}$$

$$q(z_n) = \text{Categorical}\left(\hat{\pi}_n\right), \tag{6.30}$$

where $\delta\{\cdot = \hat{\mu}_k\}$ is the Dirac-delta function.

We then seek to minimize $\mathrm{KL}(q(\mu, z)\|p(\mu, z|y))$ over the variational parameters $\hat{\mu}$ and $\hat{\pi}$. This is equivalent to maximizing the ELBO

$$\sum_{n=1}^{N} \mathbb{E}_{q(z_n;\pi_n)}\Big[\log \frac{p(y_n|\hat{\mu}, z_n)p(z_n)}{q(z_n)}\Big] + \sum_{k=1}^{K} \log p(\hat{\mu}_k). \tag{6.31}$$

Note that the expectation over $z_n$ is a summation over $K = 10$ categories. Figure 6.3 compares the performance of unbiased stochastic gradients produced from REINFORCE$^+$ to the Rao-Blackwellization of REINFORCE$^+$ for optimization of the ELBO in Equation 6.31.

Unlike the Bernoulli example, we are also optimizing parameters inside the expectation; specifically, in this case we are jointly optimizing the variational mean parameters $\hat{\mu}_k$ alongside the $\hat{\pi}_n$. We expect that more quickly learning the latent categories $z_n$ aids the optimization process, since the mean parameters depend on the cluster memberships.

We initialized the optimization with $K$-means. Figure 6.3 shows that Rao-Blackwellization improves the convergence rate, with faster convergence when more categories are summed. With summing just three categories, we nearly recover the same ELBO trajectory of the exact gradient, which sums all ten categories. We chose $K = 10$ as an example so we can compare against the exact gradient; with larger $K$, computing the exact gradient will become intractable and stochastic methods such as ours will be required.

We also examine here the computational trade-off. Our Rao-Blackwellized estimator with $k$ categories summed requires $k+1$ evaluations of the original REINFORCE$^+$ estimator. For a fairer comparison, we also consider the benefits of variance reduction obtained from simple Monte-Carlo sampling, where $k + 1$ samples of the REINFORCE$^+$ estimator are averaged at each iteration. In this experiment, Rao-Blackwellization yields better performance than Monte-Carlo averaging. This is because for most observations, memberships are fairly unambiguous and so $q(z)$ is concentrated. This is the regime where our theory suggests significant variance reduction using Rao-Blackwellization.

### 6.4.3 An example with countably infinite $K$

We give an example to demonstrate our method when there is a countably infinite number of categories. Consider the N-mixture model,

$$N \sim \text{Poisson}(\lambda) \tag{6.32}$$

$$y_i \sim \text{Binomial}(N, p) \quad \text{for} \quad i = 1, ..., n, \tag{6.33}$$

a model used in ecological modeling of species counts (Royle, 2004).

In our experiment, we take $p$ and $\lambda$ to be known parameters. We want to infer $N$ given data $y_1, ..., y_n$. Since the support of $N$ in the exact posterior is the integers greater than or equal to $y_{max} := \max_n \{y_n\}$, we use a negative binomial distribution shifted by $y_{max}$

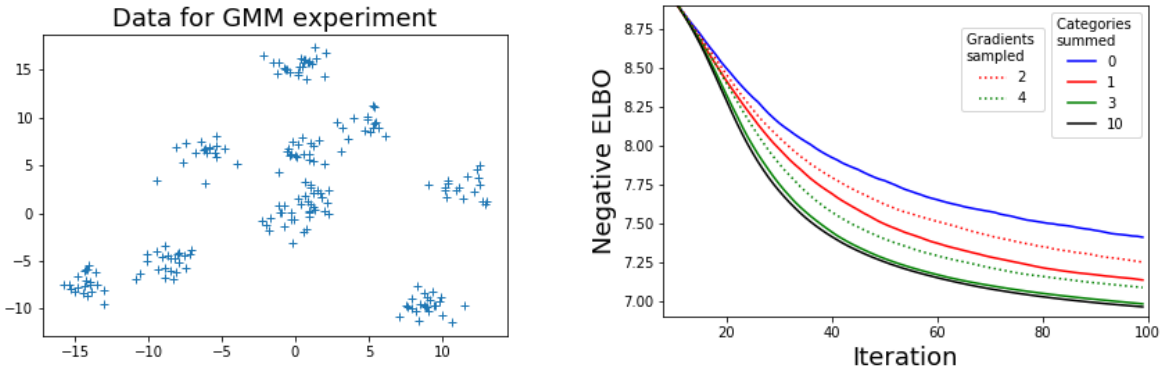



Figure 6.3: Results for Gaussian mixture model experiment. (Left) Simulated data. (Right) Solid lines display the negative ELBO per iteration using REINFORCE$^+$, for $k$ categories summed. Zero categories summed is the original REINFORCE$^+$ estimator, while 10 categories summed returns the analytic gradient. Dashed lines show performance when $n \in \{2, 4\}$ draws of the REINFORCE$^+$ estimator are averaged at each iteration to reduce variance. Each line is an average over 20 trials from the same initialization.

to approximate the posterior. Let $\hat{r}$ and $\hat{p}$ be the number of failures and the probability of success, respectively, for a negative binomial. We optimize the ELBO,

$$\mathcal{L}(\hat{r}, \hat{p}) = E_{q(N;\hat{r},\hat{p})}[\log p(y|N)p(N) - \log q(N; \hat{r}, \hat{p})] \tag{6.34}$$

This expectation is taken over $N$ and is given by an infinite sum. The exact expectation is intractable. However, we have a closed form variational distribution, and for any $\hat{r}$ and $\hat{p}$, it is easy to find the integers $N$ where $q(N; \hat{r}, \hat{p})$ places most of its mass. We therefore can apply our Rao-Blackwellization procedure to compute stochastic gradients of the ELBO.

In our experiment, we take the true $N = 10$ and $p = 0.2$. We drew 1000 data points from Equation 6.33. We set our Poisson prior with $\lambda = 10$.

We found that the REINFORCE estimator was too high variance to be useful in this example, so we start with REINFORCE$^+$. Figure 6.4 compares the REINFORCE$^+$ estimator with its Rao-Blackwellization, using either $k = 1$ or $k = 3$ categories summed.

We find that our Rao-Blackwellization improves the convergence rate of the ELBO. This is because our variational distribution eventually concentrates around the true $N$ (Figure 6.4, right), and only a few categories have significant mass.

### 6.4.4 Generative semi-supervised classification

#### Semi-supervised models

The goal of a semi-supervised classification task is to predict labels $y$ from $x$, but where the training set consists of both labeled data $(x, y) \sim \mathcal{D}_L$ and unlabeled data $x \sim \mathcal{D}_U$. The approach proposed by Kingma et al. (2014) uses a variational autoencoder (VAE) whose

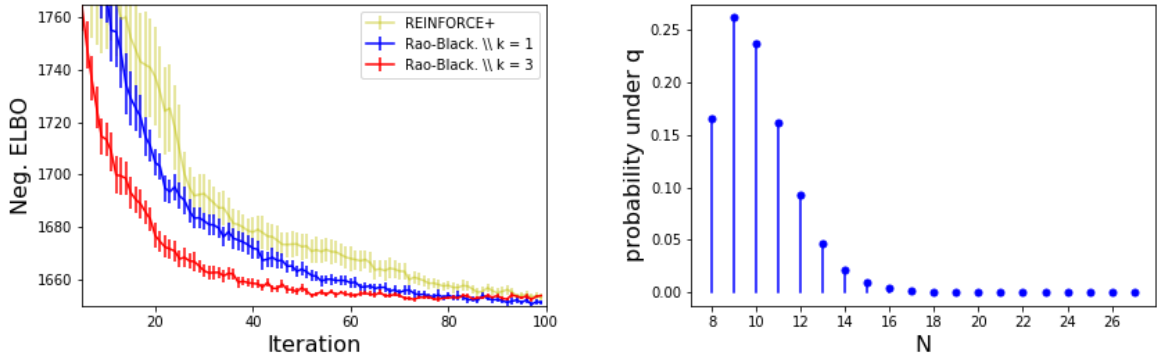



Figure 6.4: (Left) Negative ELBO per iteration in the N-mixture experiment. We compare the REINFORCE$^+$ estimator with its Rao-Blackwellization, using either $k = 1$ or $k = 3$ categories summed. Vertical lines denote standard errors over 10 trials from the same initialization. (Right) Negative binomial variational distribution $q$ at convergence.

latent space is joint over a Gaussian variable $z$ and the discrete label $y$. The training objective is to learn a classifier $q_\phi(y|x)$, an inference model $q_\phi(z|x, y)$, and a generative model $p_\theta(x|y, z)$. On labeled data, the variational lower bound is

$$\log p_\theta(x, y) \geq \mathcal{L}^L(x, y) \tag{6.35}$$

$$:= \mathbb{E}_{q_\phi(z|x,y)}[\log p_\theta(x|y, z) + \log p_\theta(z) + \log p_\theta(y) - \log q_\phi(z|x, y)] \tag{6.36}$$

On unlabeled data, the unknown label $y$ is treated as a latent variable and integrated out,

$$\log p_\theta(x) \geq \mathcal{L}^U(x) \tag{6.37}$$

$$:= \mathbb{E}_{q_\phi(z|x,y)q_\phi(y|x)}[\log p_\theta(x|y, z) + \log p_\theta(z) + \log p_\theta(y) - \log q_\phi(z|x, y) - \log q_\phi(y|x)] \tag{6.38}$$

$$= \mathbb{E}_{q_\phi(y|x)}[\mathcal{L}^L(x, y) - \log q_\phi(y|x)] \tag{6.39}$$

The full objective to be maximized is

$$\mathcal{J} = \mathbb{E}_{x\sim\mathcal{D}_U}[\mathcal{L}^U(x)] + \mathbb{E}_{(x,y)\sim\mathcal{D}_L}[\mathcal{L}^L(x, y)] + \alpha\mathbb{E}_{(x,y)\sim\mathcal{D}_L}[\log q_\phi(y|x)] \tag{6.40}$$

where the third term is added for the classifier $q_\phi(y|x)$ to also train on labeled data. $\alpha$ is a hyperparameter which we set to 1.0 in our experiments.

We take $z$ to be a continuous random variable with a standard Gaussian prior. Hence, gradients can flow through $z$ using the reparametrization trick. However, $y$ is a discrete label. The original approach proposed by Kingma et al. (2014) computed the expectation

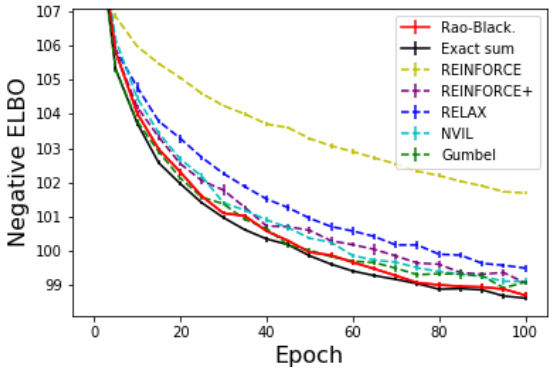



Figure 6.5: Results on the semisupervised MNIST task. Plotted is test set negative ELBO evaluated at the MAP label. Paths are averages over 10 runs from the same initialization. Vertical lines are standard errors. Our method (red) is comparable with summing out all ten categories (black).

in Equation 6.39 by exactly summing over the ten categories. However, most images are unambiguous in their classification, so $q_\phi(y|x)$ is often concentrated on just one category. We will show that applying our Rao-Blackwellization procedure with one category summed gives results comparable to computing the the full sum, more quickly.

### Experimental Results

We work with the MNIST dataset (Lecun et al., 1998). We used 50 000 MNIST digits in the training set, 10 000 digits in the validation set, and 10 000 digits in the test set. Among the 50 000 MNIST digits in the training set, 5 000 were randomly selected to be labeled, and the remaining 45 000 were unlabeled.

To optimize, we Rao-Blackwellized the REINFORCE estimator. We compared against REINFORCE without Rao-Blackwellization; the exact gradient with all 10 categories summed; REINFORCE$^+$; Gumbel-softmax (Jang et al., 2017); NVIL (Mnih and Rezende, 2016); and RELAX (Grathwohl et al., 2018).

For all methods, we used performance on the validation set to choose step-sizes and other parameters. See Section 6.6 for details concerning parameters and model architecture.

Figure 6.5 shows the negative ELBO, $-\mathcal{L}^L(x, y)$, on the test set evaluated at the MAP label as a function of epoch. In this experiment, our Rao-Blackwellization with one category summed (RB-REINFORCE) achieves the same convergence rate as the original approach where all ten categories are analytically summed. Moreover, our method achieves comparable test accuracy, at 97%. Finally, our method requires about 18 seconds per epoch, compared to 31 seconds per epoch when using the full sum (Table 6.1).

In comparing with other approaches, we clearly improve upon the convergence rate of REINFORCE. We slightly improve on RELAX. On this example, REINFORCE$^+$, NVIL,

Table 6.1: Accuracies and timing results on semi-supervised MNIST classification. Standard errors of test accuracies are over 10 runs of each method. Standard deviations of timing are over the 100 epochs of 10 runs. Training was run on a p3.2xlarge instance on Amazon Web Services.

| Method | test acc. (SE) | secs/epoch (SD) |
|---|---|---|
| RB-REINFORCE | 0.965 (0.001) | 17.5 (1.8) |
| Exact sum | 0.966 (0.001) | 31.4 (3.2) |
| REINFORCE | 0.940 (0.002) | 15.7 (1.6) |
| REINFORCE$^+$ | 0.953 (0.001) | 17.2 (1.7) |
| RELAX | 0.966 (0.001) | 29.8 (3.0) |
| NVIL | 0.956 (0.002) | 17.5 (1.8) |
| Gumbel-softmax | 0.954 (0.001) | 16.4 (1.7) |

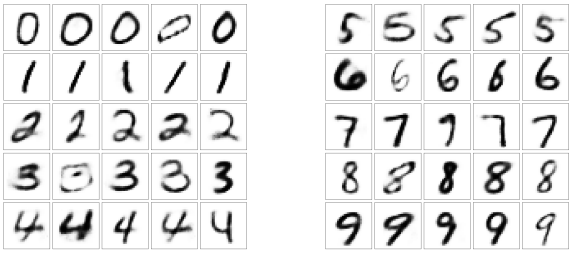


Figure 6.6: The conditional generation of MNIST digits. Each row displays five draws from the learned generative model $z \sim \mathcal{N}(0, I)$, $x \sim p_\theta(x|y, z)$, for a different digit $y$ in each row.

and Gumbel-softmax also give results comparable to ours.

Figure 6.6 displays the conditional generation of MNIST digits obtained after 100 epochs of running our Rao-Blackwellized gradient method.

### 6.4.5 Moving MNIST

In this section, we use a hard-attention mechanism (Mnih et al., 2014; Gregor et al., 2015) to model non-centered MNIST digits. We choose this problem because, as will be seen, the exact stochastic gradient is intractable due to the large number of categories. However, only a few of the categories will have significant probabilities.

Like the original VAE work (Kingma and Welling, 2013), we learn an inference model $q_\phi(z|x)$ and generative model $p_\theta(x|z)$, where $z$ is a low-dimensional, continuous representation of the MNIST digit $x$. Unlike the previous section, we are no longer using the class label. However, we now work with a non-centered MNIST digit, and in order to train the inference and generative models, we must also infer the pixel at which the MNIST digit is centered.

More precisely, our generative model is as follows. For each image, we sample a two-vector

representing the pixel at which to center the original $28 \times 28$ MNIST image:

$$\ell \sim \text{Categorical}(H \times W). \tag{6.41}$$

Here $H$ and $W$ are respectively the height and width, in pixels, of the larger image frame on which the MNIST digit will be placed. We take $H = W = 68$ in our experiments.

Next, we generate the non-centered MNIST digit as

$$z \sim \mathcal{N}(0, I_d), \tag{6.42}$$

$$x_{h,w}|\ell, z \overset{ind}{\sim} \text{Bernoulli}(\mu(z)[h - \ell_0, w - \ell_1]), . \tag{6.43}$$

for $h \in \{0, ..., H-1\}$ and $w \in \{0, ..., W-1\}$. Here $\mu$ is a neural network that maps $z \in \mathbb{R}^d$ to a grid of mean parameters $\mu(z) \in \mathbb{R}^{28 \times 28}$. In Equation 6.43, we take $\mu(z)[a, b] = 0$ if $(a, b) \notin [0, 28]^2$.

In this way, $x \in \mathbb{R}^{H \times W}$ is a random sample of an image containing a single non-centered MNIST digit on a blank background (Figure 6.7).

Hence, we need to learn not only the generative model for an MNIST digit, but also the pixel at which the digit is centered. Our two latent variables are $z_n$ and $\ell_n$. We find a variational approximation to the posterior using an approximating family of the form

$$\ell_n|x_n \sim \text{Categorical}(\zeta(x_n)), \tag{6.44}$$

$$z_n|x_n, \ell_n \sim \mathcal{N}(h_\mu(x_n, \ell_n), h_\Sigma(x_n, \ell_n)), \tag{6.45}$$

where $\zeta$, $h_\mu$, and $h_\Sigma$ are neural networks. Section 6.6 details the architecture for the neural networks.

REINFORCE was too high variance to be practical here, so we started with REINFORCE$^+$ and its Rao-Blackwellization. Here, we chose to sum the top five categories. We again compare with NVIL, Gumbel-softmax, and RELAX. For all the methods, we use a validation set to tune step-sizes and other parameters.

Figure 6.8 shows the negative ELBO on the test set evaluated at the MAP pixel location as a function of epoch. RELAX converged to a similar ELBO as our method, but did so at a slower rate. While NVIL also converged quickly, it converged to a worse negative ELBO than our method.

Gumbel-softmax did not appear to converge to a reasonable ELBO. We believe that the bias of this procedure was too high in this application. In particular, because we are constrained to sampling discrete values for the pixel attention, we must use the straight-through version of Gumbel-softmax (Bengio et al., 2013; Jang et al., 2017), which suffers from even higher bias.

Our method is more computationally expensive per epoch than the others (Table 6.2). However, the gains in convergence are still substantive: for example, it takes about 44 seconds for our method to reach a negative ELBO of 500, while it takes RELAX about 110 seconds.

Figure 6.9 displays (1) the original non-centered MNIST digit; (2) the reconstruction of the MNIST digits after passing through our attention mechanism and VAE; and (3) the

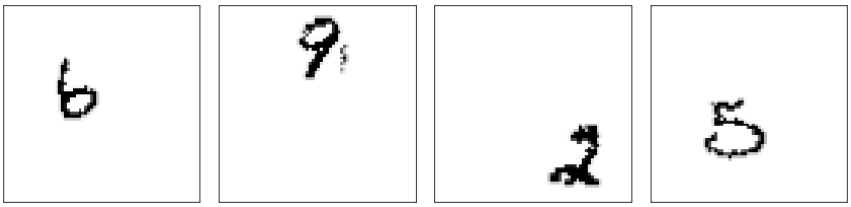


Figure 6.7: Examples of non-centered MNIST digits

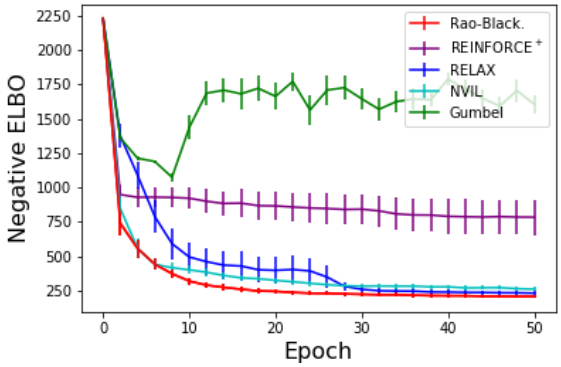



Figure 6.8: Results on the moving MNIST task. Plotted is test set negative ELBO evaluated at the MAP pixel location. Paths are averages over 10 runs from the same initialization. Vertical lines are standard errors. Our Rao-Blackwellization (red) with summing out the top five categories exhibits the fastest convergence and reaches a smaller negative ELBO than NVIL and REINFORCE$^+$.

Table 6.2: Timing results on the moving MNIST task. Standard deviations of timing are over the 50 epochs of 10 runs. Training was run on a p3.2xlarge instance on Amazon Web Services.

| Method | secs/epoch (SD) |
|---|---|
| RB-REINFORCE$^+$ | 15.4 (2.3) |
| REINFORCE$^+$ | 8.9 (1.3) |
| RELAX | 11.1 (1.6) |
| NVIL | 9.5 (1.4) |
| Gumbel-softmax | 8.7 (1.2) |

learned pixel locations. Our method performs best because it is the only one that takes advantage of the fact that only a few digit positions have high probabilities. Summing these positions analytically removes much of the variance.

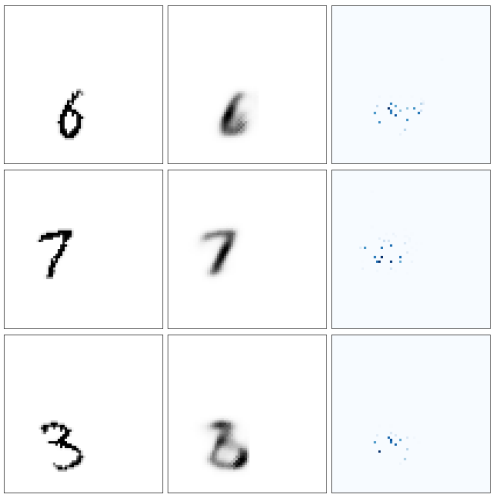


Figure 6.9: (Left column) The original MNIST digit. (Center column) The reconstructed MNIST digit. (Right column) The learned probability distribution over the grid of pixels. Brighter spots indicate higher probabilities.

## 6.5 Conclusion

Efficient stochastic approximation of the gradient $\nabla_\eta \mathbb{E}_{q_\eta(z)}[f_\eta(z)]$, where $z$ is discrete, is a basic problem that arises in many probabilistic modelling tasks. We have presented a general method to reduce the variance of stochastic estimates of this gradient, without changing the bias. Our method is grounded in the classical technique of Rao-Blackwellization. Experiments on synthetic data and two large-scale MNIST modeling problems show the practical benefits of our variance-reduced estimators.

We have focused on the particular setting where $z$ is a univariate discrete random variable, which is relevant for many applications. In other situations, multiple discrete variables will naturally appear in the expectations being optimized. Treating these as a single discrete variable over the Cartesian product of the sample spaces may make such problems amenable to our Rao-Blackwellization approach.

In addition, many multivariate discrete distributions arising in modeling applications will be structured (e.g., the discrete-space latent Markov chain of an HMM). Local expectation gradients (Titsias and Lázaro-Gredilla, 2015) reduce high-dimensional expectations over these multivariate discrete distributions to iterated univariate expectations through appropriate conditioning on variable sets. Our technique can then be applied for variance reduction in computing the univariate expectations. This is an avenue of future research.

## 6.6 Experiment technical details

Implementations of all methods in this chapter as well as code to reproduce our results can be found in the git repository `https://github.com/Runjing-Liu120/RaoBlackwellizedSGD`.

### 6.6.1 Generative semi-supervised classification

In this experiment, our classifier $q_\phi(y|x)$ consists of three fully connected hidden layers, each with 256 nodes and ReLU activations. The inference and generative models, $q_\phi(z|x, y)$ and $p_\theta(x|z, y)$, both have one hidden layer with 128 nodes and ReLU activations, similar to the MLPs used in Kingma et al. (2014). The latent variable $z$ is five dimensional and $q_\phi(z|x)$ is multivariate Gaussian with diagonal covariance.

For all methods, we used performance on a validation set to choose between the possible step-sizes, {5e-5, 1e-4, 5e-4, 1e-3, 5e-3}. For Gumbel-softmax, we also chose the annealing rate among {1e-5, 5e-5, 1e-4, 5e-4}. For RELAX, the relaxation temperature was chosen adaptively using gradients, while the scaling parameter was set at 1.0.

The step-size for REINFORCE was chosen to be 1e-4 and the step-size for RELAX was chosen to be 5e-4. The step-size for the remaining methods were chosen to be 1e-3. The annealing rate for Gumbel-softmax was chosen to be 5e-4.

Optimization was done with Adam (Kingma and Ba, 2014), with parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$. An initialization for $q_\phi(z|x, y)$ and $p_\theta(x|z, y)$ was obtained by first optimizing $\mathcal{L}^L(x, y)$ on the labeled data only. We also initialized $q_\phi(y|x)$ on the labeled data using cross-entropy loss. The results in the paper show the optimization of the semi-supervised ELBO starting from this initialization.

### 6.6.2 Moving MNIST

For the decoder $p(x|l, z)$ we use one fully connected hidden layer with 256 nodes and tanh activations, similar to the architecture described in (Kingma and Welling, 2013). Our $z$ is 5 dimensional.

The attention mechanism $q(l|x)$ contains four convolutional layers, each with 7 output channels and ReLU activations; the final layer is a fully connected layer with a softmax. The encoder network $q(z|x)$ has one fully connected hidden layer with 256 nodes and tanh activations, mirroring the decoder network.

We again used performance on the validation set to choose between the possible step-sizes and model parameters as described in the section above. The learning rate and annealing rate for Gumbel-sofmax was chosen to be 5e-5 and 5e-4, respectively. For RELAX, the learning rate was 5e-4. The step-sizes for the remaining procedures were chosen to be 1e-3. We again use Adam Kingma and Ba (2014) for optimization, and we set $\beta_1 = 0.9$, $\beta_2 = 0.999$.

# Bibliography

Timothy M.C. Abbott, Filipe B. Abdalla, Alex Alarcon, et al. Dark energy survey year 1 results: Cosmological constraints from galaxy clustering and weak lensing. *Physical Review D*, 98(4), 2018.

Deokkeun An, Jennifer A. Johnson, James L. Clem, et al. Galactic globular and open clusters in the Sloan Digital Sky Survey. I. crowded-field photometry and cluster fiducial sequences in ugriz. *The Astrophysical Journal Supplement Series*, 179(2):326–354, 2008.

Edgar Anderson. The species problem in iris. *Annals of the Missouri Botanical Garden*, 23 (3):457–509, 1936.

Bastien Arcelin, Cyrille Doux, Eric Aubourg, Cécile Roucelle, and LSST Dark Energy Science Collaboration. Deblending galaxies with variational autoencoders: a joint multiband, multi-instrument approach. *Monthly Notices of the Royal Astronomical Society*, 500(1): 531–547, 2021.

Sanjib Basu, Sreenivasa R. Jammalamadaka, and Wei Liu. Local posterior robustness with parametric priors: Maximum and average sensitivity. In *Maximum Entropy and Bayesian Methods*, pages 97–106. Springer, 1996.

Matthew Beal and Zoubin Ghahramani. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian Statistics*, 2002.

Yoshua Bengio, Nicholas Leonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013. `https://arxiv.org/abs/1308.3432`.

Michael J. Betancourt and Mark Girolami. Hamiltonian Monte Carlo for hierarchical models, 2013. `https://arxiv.org/abs/1312.0906`.

Patrick Billingsley. *Probability and Measure*. John Wiley and Sons, second edition, 1986.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

David M Blei and Michael I. Jordan. Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, 1(1):121–143, 2006.

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

Jörg Bornschein and Yoshua Bengio. Reweighted wake-sleep, 2014. `https://arxiv.org/abs/1406.2751`.

James Bosch, Robert Armstrong, Steven Bickerton, et al. The hyper suprime-cam software pipeline. *Publications of the Astronomical Society of Japan*, 70(SP1):1–39, 2018.

James Bradbury, Roy Frostig, Peter Hawkins, et al. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

Brendon J. Brewer, Daniel Foreman-Mackey, and David W. Hogg. Probabilistic catalogs for crowded stellar fields. *The Astronomical Journal*, 146(1):7–15, 2013.

George Casella and Christian P. Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

Richard M. Dudley. *Real analysis and probability*. CRC Press, 2018.

Bradley Efron. *The Jackknife, the Bootstrap, and Other Resampling Plans*, volume 38. Society for Industrial and Applied Mathematics, 1982.

ESAHubble. Hubble's instruments: ACS – advanced camera for surveys. `https://esahubble.org/about/general/instruments/acs/`, 2021. [Accessed: 2021-02-21].

Richard M Feder, Stephen K. N. Portillo, Tansu Daylan, and Douglas Finkbeiner. Multiband probabilistic cataloging: a joint fitting approach to point-source detection and deblending. *The Astronomical Journal*, 159(4):163–188, 2020.

Thomas S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, pages 209–230, 1973.

Ronald Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

Wendell Fleming. *Functions of several variables*. Springer Science & Business Media, 2012.

G. David Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

E. B. Fowlkes and C. L. Mallows. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, 78(383):553–569, 1983.

Peter Galbusera, Lens Lens, Tine Schenck, Edward Waiyaki, and Erik Matthysen. Genetic variability and gene flow in the globally, critically-endangered taita thrush. *Conservation Genetics*, 1:45–55, 2000.

Andrew Gelman and Donald B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4):457–472, 11 1992.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis, Third Edition*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis, 2013.

Ryan Giordano. *On the Local Sensitivity of M-Estimation: Bayesian and Frequentist Applications*. PhD thesis, University of California, Berkeley, 2019.

Ryan Giordano, Tamara Broderick, and Michael I. Jordan. Covariances, robustness and variational Bayes. *Journal of Machine Learning Research*, 19(51), 2018.

Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations*, 2018.

Gregory M. Green, Edward Schlafly, Catherine Zucker, Joshua S. Speagle, and Douglas Finkbeiner. A 3D dust map based on Gaia, Pan-STARRS 1, and 2MASS. *The Astrophysical Journal*, 887(1):93–120, 2019.

Peter J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

Karol Gregor, Andriy Mnih, and Daan Wierstra. Deep autoregressive networks. In *International Conference on Machine Learning*, 2014.

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. DRAW: a recurrent neural network for image generation. In *International Conference on Machine Learning*, 2015.

Shixiang Gu, Sergey Levine, Ilya Sutskever, and Andriy Mnih. MuProp: Unbiased backpropagation for stochastic neural networks. In *International Conference on Learning Representations*, 2016.

Paul Gustafson. Local sensitivity of posterior expectations. *Annals of Statistics*, 24(1): 174–195, 1996.

Paul Gustafson. *Local Robustness in Bayesian Analysis*, pages 71–88. Springer New York, New York, NY, 2000.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *International Conference on Artificial Intelligence and Statistics*, 2012.

Yashar D. Hezaveh, Laurence Perreault Levasseur, and Philip J. Marshall. Fast automated analysis of strong gravitational lenses with convolutional neural networks. *Nature*, 548 (7669):555–557, 2017.

Geoffrey E. Hinton, Peter Dayan, Brendan J. Frey, and Radford M. Neal. The wake-sleep algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

Matthew Hoffman, Francis Bach, and David Blei. Online learning for latent Dirichlet allocation. In *Advances in Neural Information Processing Systems*, volume 23, pages 856–864. Curran Associates, Inc., 2010.

Matthew D. Hoffman and Andrew Gelman. The no-u-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1): 1593–1623, 2014.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(4):1303–1347, 2013.

Xiaosheng Huang, Manuel Domingo, Andrew Pilon, et al. Finding strong gravitational lenses in the DESI DECam Legacy survey. *The Astrophysical Journal*, 894(1):78–106, 2020.

Louis A. Jaeckel. The infinitesimal jackknife, memorandum. Technical report, MM 72-1215-11, Bell Lab. Murray Hill, NJ, 1972.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-softmax. In *International Conference on Learning Representations*, 2017.

Michael I. Jordan, Zoubin Ghahramani, Tommi I. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

Rudalf E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

Diederik P. Kingma and Jimmy Ba. Adam: a method for stochastic optimization, 2014. `https://arxiv.org/abs/1412.6980`.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes, 2013. `https://arxiv.org/abs/1312.6114`.

Diederik P. Kingma, Danilo Jimenez Rezende, Shakir Mohamed, and Max Welling. Semi-supervised learning with deep generative models, 2014. `https://arxiv.org/abs/1406.5298`.

Steven G. Krantz and Harold R. Parks. *The implicit function theorem: History, theory, and applications*. Springer Science & Business Media, 2012.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M. Blei. Automatic differentiation variational inference. *Journal of Machine Learning Research*, 18 (14):1–45, 2017.

Franccois Lanusse, Rachel Mandelbaum, Siamak Ravanbakhsh, Chun-Liang Li, Peter Freeman, and Barnabas Poczos. Deep generative models for galaxy image simulations, 2020. `https://arxiv.org/abs/2008.03833`.

Franccois Lanusse, Quanbin Ma, Nan Li, et al. CMU DeepLens: deep learning for automatic image-based galaxy–galaxy strong lens finding. *Monthly Notices of the Royal Astronomical Society*, 473(3):3895–3906, 2017.

Tuan Anh Le, Adam R. Kosiorek, N. Siddharth, Yee Whye Teh, and Frank Wood. Revisiting reweighted wake-sleep for models with stochastic control flow. In *Uncertainty in Artificial Intelligence*, pages 1039–1049, 2020.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc Le, and Ni Lao. Memory augmented policy optimization for program synthesis with generalization. In *Neural Information Processing Systems*, 2018.

LSST. About LSST. `https://www.lsst.org/about/dm`, 2020. [Accessed: 2020-12-01].

Yihui Luan and Hongzhe Li. Clustering of time-course gene expression data using a mixed-effects model with B-splines. *Bioinformatics*, 19(4):474–482, 2003.

Robert Lupton, James E. Gunn, Zeljko Ivezic, Gillian R. Knapp, Stephen Kent, and Naoki Yasuda. The SDSS imaging pipelines, 2001. `https://arxiv.org/abs/astro-ph/0101420`.

Dougal Maclaurin, David Duvenaud, and Ryan P. Adams. Autograd: Effortless gradients in NumPy. In *International Conference on Machine Learning (ICML) AutoML Workshop*, 2015.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, 2014.

Andriy Mnih and Danilo Jimenez Rezende. Variational inference for Monte Carlo objectives. In *International Conference on Machine Learning*, 2016.

Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, 2014.

Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In *International Conference on Artificial Intelligence and Statistics*, 2005.

Radford Neal and Geoffrey Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. *Learning in graphical models*, 89:355–368, 2000.

Radford M. Neal. MCMC using Hamiltonian dynamics, 2012.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, second edition, 2006.

Stephen K. N. Portillo, Benjamin C. G. Lee, Tansu Daylan, and Douglas P. Finkbeiner. Improved point-source detection in crowded fields using probabilistic cataloging. *The Astronomical Journal*, 154(4):132–156, 2017.

Jonathan K. Pritchard, Matthew Stephens, and Peter Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959, 2000.

Anil Raj, Matthew Stephens, and Jonathan K. Pritchard. fastSTRUCTURE: Variational inference of population structure in large SNP data sets. *Genetics*, 197(2):573–589, 2014.

Rajesh Ranganath, Sean Gerrish, and David M. Blei. Black box variational inference, 2013. `https://arxiv.org/abs/1401.0118`.

Jeffrey Regier, Jon D. McAuliffe, and Prabhat. A deep generative model for astronomical images of galaxies. In *NIPS Workshop on Advances in Approximate Bayesian Inference*, 2015.

Jeffrey Regier, Andrew C. Miller, David Schlegel, Ryan P. Adams, Jon D. McAuliffe, and Prabhat. Approximate inference for constructing astronomical catalogs from images. *The Annals of Applied Statistics*, 13(3):1884–1926, 2019.

David M. Reiman and Brett E. Göhre. Deblending galaxy superpositions with branched generative adversarial networks. *Monthly Notices of the Royal Astronomical Society*, 485 (2):2617–2627, 2019.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models, 2014. `https://arxiv.org/abs/1401.4082`.

Andrew J. Royle. N-mixture models for estimating population size from spatially replicated counts. *Biometrics*, 60(1):108–115, 2004.

Olga Russakovsky, Jia Deng, Hao Su, et al. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

Ata Sarajedini, Luigi R. Bedin, Brian Chaboyer, et al. The ACS survey of galactic globular clusters. *The Astronomical Journal*, 133(4):1658–1672, 2007.

Edward F. Schlafly, Gregory M. Green, Dustin Lang, et al. The DECam plane survey: Optical photometry of two billion objects in the southern galactic plane. *The Astrophysical Journal Supplement Series*, 234(2):39–58, 2018.

SDSS. Scope. `https://www.sdss.org/dr16/scope/`, 2020. [Accessed: 2020-12-23].

Jayaram Sethuraman. A constructive definition of Dirichlet priors. *Statistica sinica*, pages 639–650, 1994.

Jason E. Shoemaker, Satoshi Fukuyama, Amie J. Eisfeld, et al. An ultrasensitive mechanism regulates influenza virus-induced inflammation. *PLoS Pathogens*, 11(6):1–25, 2015.

Robert H. Shumway and David S. Stoffer. *State space models*, chapter 6. Springer Texts in Statistics. Springer International Publishing, 2017.

James C. Spall. *Introduction to Stochastic Search and Optimization*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 2003.

Peter B. Stetson. DAOPHOT: A computer program for crowded-field stellar photometry. *Astronomical Society of the Pacific*, 99:191–222, 1987.

John D. Storey, Wenzhong Xiao, Jeffrey T. Leek, Ronald G. Tompkins, and Ronald W. Davis. Significance analysis of time course microarray experiments. *Proceedings of the National Academy of Sciences of the United States of America*, 102(36):12837–42, 2005.

Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

Michalis K. Titsias. Combine Monte Carlo with exhaustive search: Effective variational inference and policy gradient reinforcement learning. In *NIPS Workshop: Advances in Approximate Inference*, 2014.

Michalis K. Titsias and Miguel Lázaro-Gredilla. Local expectation gradients for black box variational inference. In *Neural Information Processing Systems*, 2015.

George Tucker, Andriy Mnih, Chris J. Maddison, John Lawson, and Jascha Sohl-Dickstein. REBAR: Low-variance, unbiased gradient estimates for discrete latent variable models. In *Neural Information Processing Systems*, 2017.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11:2837–2854, 2010.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, et al. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17:261–272, 2020.

Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2007.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.

Chong Wang, John Paisley, and David Blei. Online variational inference for the hierarchical Dirichlet process. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15, pages 752–760. JMLR Workshop and Conference Proceedings, 2011.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992.

Edward L. Wright, Peter R. M. Eisenhardt, Amy K. Mainzer, et al. The wide-field infrared survey explorer (WISE): Mission description and initial on-orbit performance. *The Astronomical Journal*, 140(6):1868–1881, 2010.

Bo Xin, Zeljko Ivezič, Robert H. Lupton, et al. A study of the point-spread function in SDSS images. *The Astronomical Journal*, 156(5):222–232, 2018.

Bin Yu. Stability. *Bernoulli*, 19(4):1484–1500, 2013.

Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41 (8):2008–2026, 2019.