Machine Learning using Approximate Inference

Variational and Sequential Monte Carlo methods

Christian Andersson Naesseth

COLUMBIA UNIVERSITY

Junior Bayes Beyond the Borders (JB^3)
July, 2020

Exchange rate data

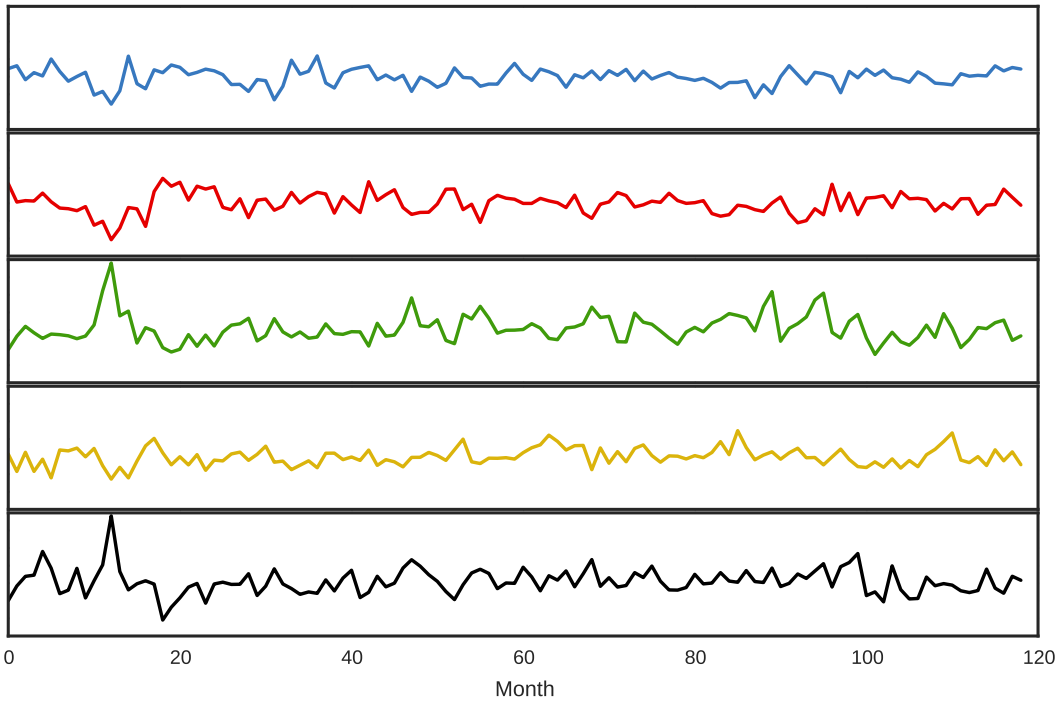
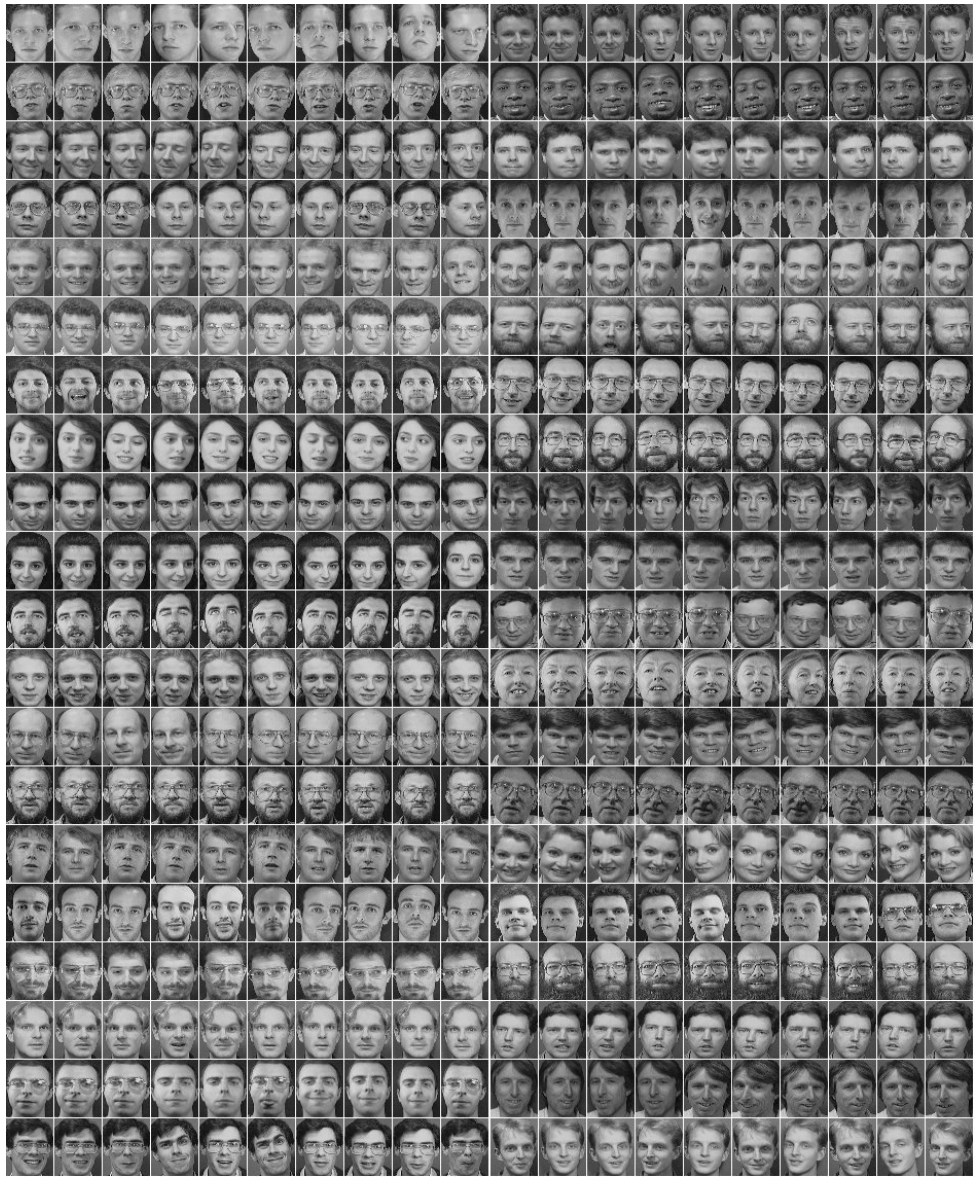


Image data



Bayesian inference

- $\mathbf{y}$ - data
- $\mathbf{x}$ - latent (unknown) variables
- $p(\mathbf{x}, \mathbf{y})$ - probabilistic (generative) model

Posterior distribution:

$$p(\mathbf{x} \mid \mathbf{y}) = \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{y})}$$

Expectations:

$$\mathbb{E}_{p(\mathbf{x} \mid \mathbf{y})}\left[\varphi(\mathbf{x})\right] = \int \varphi(\mathbf{x}) p(\mathbf{x} \mid \mathbf{y}) \mathrm{d}\mathbf{x}$$

Examples of probabilistic generative models

- *Exchangeable data models*

$$p(\mathbf{x}, \mathbf{y}_{1:n}) = \mu(\mathbf{x}) \prod_{i=1}^{n} g(\mathbf{y}_i|\mathbf{x})$$

  Applications: Regression, classification, ...

- *State space models*

$$p(\mathbf{x}_{1:T}, \mathbf{y}_{1:T}) \triangleq \mu(\mathbf{x}_1) \prod_{t=2}^{T} f(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}) \prod_{t=1}^{T} g(\mathbf{y}_t \,|\, \mathbf{x}_t)$$

  Applications: Filtering, smoothing, forecasting, ...

Approximate Bayesian inference

Because $p(\mathbf{x} \mid \mathbf{y})$ is typically intractable we approximate:

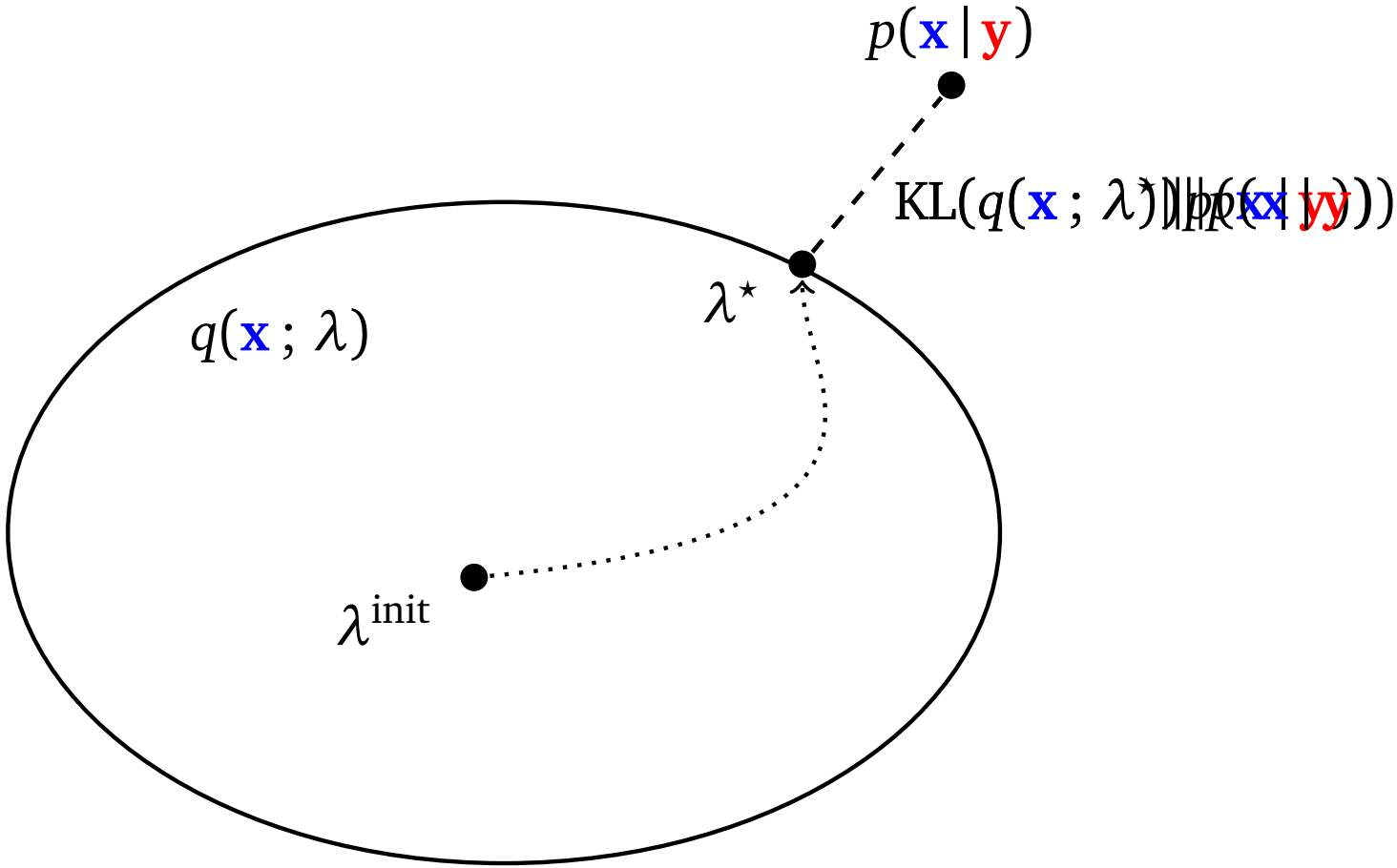
Variational methods - Turn integration into optimization

$$\mathbb{E}_{p(\mathbf{x} \mid \mathbf{y})}\left[\varphi(\mathbf{x})\right] \approx \mathbb{E}_{q(\mathbf{x}\,;\,\lambda^\star)}\left[\varphi(\mathbf{x})\right]$$

Monte Carlo methods - Random sampling

$$\mathbb{E}_{p(\mathbf{x} \mid \mathbf{y})}\left[\varphi(\mathbf{x})\right] \approx \frac{1}{N}\sum_{i=1}^{N} \varphi(\mathbf{x}^i), \quad \mathbf{x}^i \sim p(\mathbf{x} \mid \mathbf{y})$$

Variational inference



Evidence lower bound

$$\mathrm{KL}(q(\mathbf{x}\,;\,\lambda)\|p(\mathbf{x}\,|\,\mathbf{y})) = \log p(\mathbf{y}) - \mathbb{E}_q\left[\log p(\mathbf{x},\mathbf{y}) - \log q(\mathbf{x}\,;\,\lambda)\right]$$

$$\Rightarrow \log p(\mathbf{y}) \geq \underbrace{\mathbb{E}_q\left[\log p(\mathbf{x},\mathbf{y}) - \log q(\mathbf{x}\,;\,\lambda)\right]}_{\triangleq \mathscr{L}(\lambda)}$$

- Maximizing the *ELBO* $\mathscr{L}(\lambda)$ minimizes KL divergence between $q(\mathbf{x}\,;\,\lambda)$ and $p(\mathbf{x}\,|\,\mathbf{y})$
- *Coordinate ascent* when conditionally conjugate
- *Stochastic optimization* using noisy $\nabla_\lambda \mathscr{L}(\lambda)$ estimated with Monte Carlo methods

Bayesian logistic regression

- Data are pairs $(\mathbf{y}_i, \mathbf{z}_i)$
  - $\mathbf{z}_i$ is a covariate
  - $\mathbf{y}_i$ is binary label
  - θ is regression coefficient
- The generative model is

$$p(\theta, \mathbf{y}_{1:n} | \mathbf{z}_{1:n}) = \mathcal{N}(\theta\,;\, 0, 1) \prod_{i=1}^{N} \text{Bernoulli}\left(\mathbf{y}_i\,;\, \sigma(\theta \mathbf{z}_i)\right),$$

where $\sigma(\cdot)$ is the logistic function mapping reals to $(0, 1)$.

VI for Bayesian logistic regression

- Let's consider a single data pair $(\mathbf{y}, \mathbf{z})$
- Variational approximation $q(\theta\,;\,\lambda) = \mathcal{N}(\theta\,;\,\mu, \sigma^2)$
- The ELBO is given by

$$\mathcal{L}(\mu, \sigma^2) = \mathbb{E}_q\left[\log p(\theta) + \log p(\mathbf{y}|\theta, \mathbf{z}) - \log q(\theta\,;\,\mu, \sigma^2)\right]$$

VI for Bayesian logistic regression

$$
\begin{aligned}
\mathcal{L}(\mu, \sigma^2) &= \mathbb{E}_q\left[\log p(\theta) + \log p(\mathbf{y}|\theta, \mathbf{z}) - \log q(\theta\,;\, \mu, \sigma^2)\right] \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log\sigma^2 + \mathbb{E}_q\left[\log p(\mathbf{y}|\theta, \mathbf{z})\right] + \text{const.} \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log\sigma^2 + \mathbb{E}_q\left[\mathbf{yz}\theta - \log\left(1 + \exp(\mathbf{z}\theta)\right)\right] \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log\sigma^2 + \mathbf{yz}\mu - \mathbb{E}_q\left[\log\left(1 + \exp(\mathbf{z}\theta)\right)\right]
\end{aligned}
$$

We are stuck because of the expectation. What can we do?

Stochastic Optimization

Gradient-based optimization:

$$\nabla \mathscr{L}(\lambda) = 0$$

Stochastic approximation: [Robbins and Monro, 1951]

$$\lambda^{k+1} = \lambda^k + \alpha^k \widehat{\nabla} \mathscr{L}(\lambda^k),$$

with $\mathbb{E}\left[\widehat{\nabla} \mathscr{L}(\lambda^k)\right] = \nabla \mathscr{L}(\lambda^k)$.

VI for Bayesian logistic regression

$$\mathcal{L}(\mu, \sigma^2) = -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log\sigma^2 + \mathbf{yz}\mu - \mathbb{E}_q\left[\log\left(1 + \exp(\mathbf{z}\theta)\right)\right]$$

$$\begin{aligned}
\nabla\mathbb{E}_q\left[\log\left(1 + \exp(\mathbf{z}\theta)\right)\right] &= \int \nabla q(\theta\,;\,\mu,\sigma^2)\log\left(1 + \exp(\mathbf{z}\theta)\right)\mathrm{d}\theta \\
&= \mathbb{E}_q\left[\nabla \log q(\theta\,;\,\mu,\sigma^2)\log\left(1 + \exp(\mathbf{z}\theta)\right)\right] \\
&\approx \frac{1}{N}\sum_{i=1}^{N} \nabla \log q(\theta^i\,;\,\mu,\sigma^2)\log\left(1 + \exp(\mathbf{z}\theta^i)\right) \triangleq \widehat{\nabla}\mathcal{L}(\mu,\sigma^2),
\end{aligned}$$

with $\theta^i \sim q(\theta\,;\,\mu,\sigma^2)$. Unbiased MC estimator of the gradient!

Stochastic gradient variational inference

Monte Carlo estimates of gradients to optimize the ELBO.

Score function: [Paisley et al., 2012, Ranganath et al., 2014]

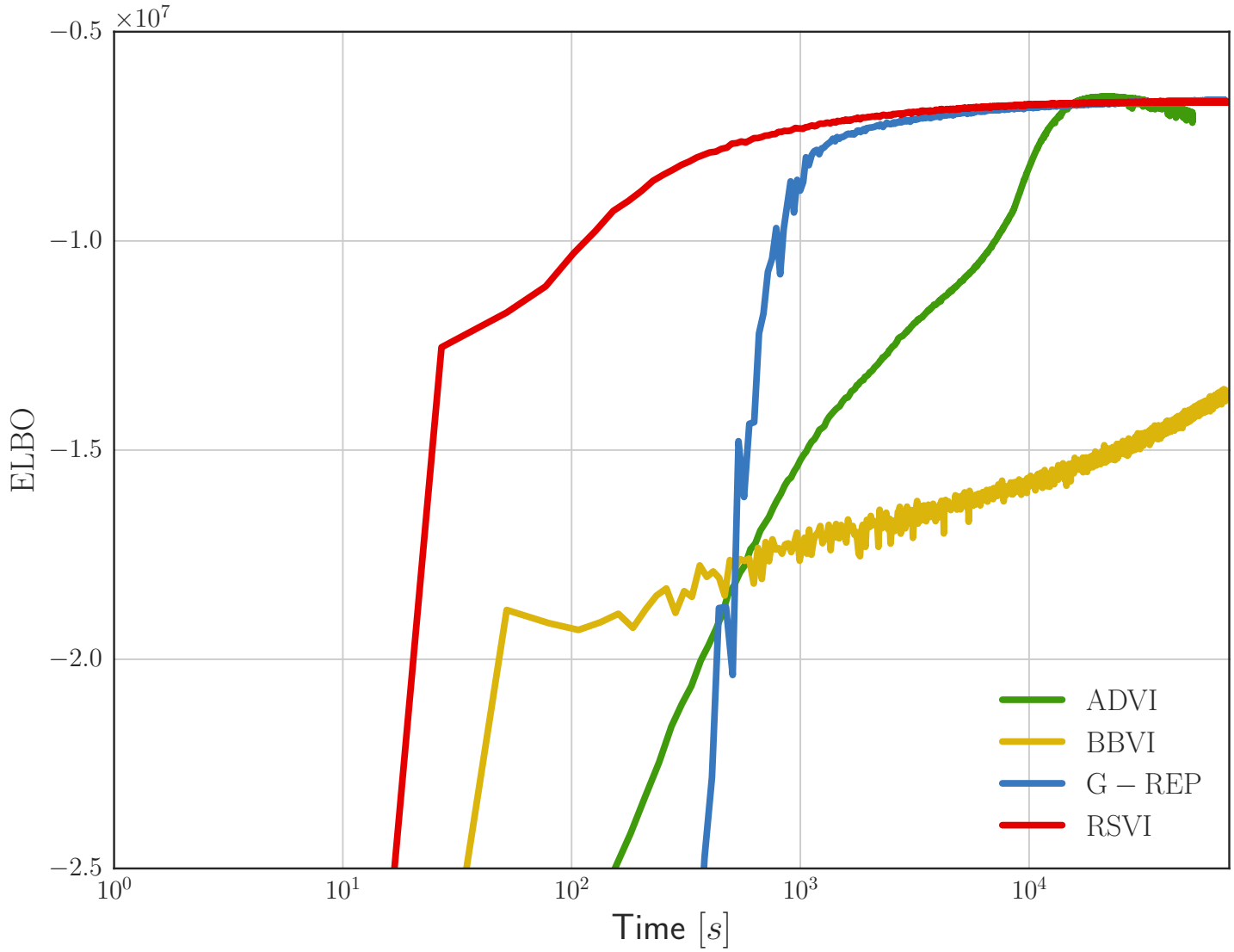
$$\begin{aligned}\nabla \mathcal{L}(\lambda) &= \nabla \mathbb{E}_q \left[\log p(\mathbf{x}, \mathbf{y}) - \log q(\mathbf{x}\,;\,\lambda) \right] \\ &= \mathbb{E}_q \left[\nabla \log q(\mathbf{x}\,;\,\lambda) \cdot \left(\log p(\mathbf{x}, \mathbf{y}) - \log q(\mathbf{x}\,;\,\lambda) \right) \right]\end{aligned}$$

Reparameterization: [Kingma & Welling 2014, Rezende et al, 2014]

$$\begin{aligned}\nabla \mathcal{L}(\lambda) &= \nabla \mathbb{E}_q \left[\log p(\mathbf{x}, \mathbf{y}) - \log q(\mathbf{x}\,;\,\lambda) \right] \\ &= \nabla \mathbb{E}_{s(\varepsilon)} \left[\log p(h(\varepsilon, \lambda), \mathbf{y}) - \log q(h(\varepsilon, \lambda)\,;\,\lambda) \right] \\ &= \mathbb{E}_{s(\varepsilon)} \left[\nabla_{\mathbf{x}} \left(\log p(\mathbf{x}, \mathbf{y}) - \log q(\mathbf{x}\,;\,\lambda) \right) \nabla_{\lambda} h(\varepsilon, \lambda) \right]\end{aligned}$$

e.g. $\mathbf{x} = h(\varepsilon, \mu, \sigma) = \mu + \sigma\varepsilon$, $s(\varepsilon) = \mathcal{N}(0, 1)$.

Reparameterization vs. score function



Why doesn't reparameterization always work?

All random variables we simulate on our computers are ultimately transformations of uniforms.

Why can't we do reparameterization gradients for everything?

Often transformations $h(\cdot)$ used in practice are *not differentiable*.

Reparameterizing gamma

- Gamma($\lambda, 1$): Common distribution in VI
- Used to sample beta, Dirichlet, Student's t, chi-squared, ...
- *Simulation:*
  - Propose

$$h(\varepsilon, \lambda) = \left(\lambda - \frac{1}{3}\right)\left(1 + \frac{\varepsilon}{\sqrt{9\lambda - 3}}\right)^3, \quad \varepsilon \sim \mathcal{N}(0, 1)$$

  - Accept–reject

$$u \leq a(\varepsilon, \lambda)? \quad u \sim \mathcal{U}[0, 1]$$

- Tricky to reparameterize because of the accept–reject step!

A simple method for generating gamma variables, Marsaglia & Tsang, 2000

Our strategy: partial reparameterization

- *Idea:* Use $\varepsilon \sim \pi(\varepsilon\,;\,\lambda)$ (only *weakly dependent* on λ) s.t. $\mathbf{x} = h(\varepsilon, \lambda) \sim q(\mathbf{x}\,;\,\lambda)$
- *Then:*

$$\nabla \mathscr{L}(\lambda) = \nabla \mathbb{E}_{\pi(\varepsilon\,;\,\lambda)}\left[\log p(h(\varepsilon,\lambda),\mathbf{y}) - \log q(h(\varepsilon,\lambda)\,;\,\lambda)\right]$$

$$= \underbrace{\mathbb{E}_{\pi(\varepsilon\,;\,\lambda)}\left[\nabla_{\mathbf{x}}\left(\log p(\mathbf{x},\mathbf{y}) - \log q(\mathbf{x},\,;\,\lambda)\right)\nabla_{\lambda} h(\varepsilon,\lambda)\right]}_{\text{reparameterization}}$$

$$+ \underbrace{\mathbb{E}_{\pi(\varepsilon\,;\,\lambda)}\left[\left(\log p(h(\varepsilon,\lambda),\mathbf{y}) - \log q(h(\varepsilon,\lambda)\,;\,\lambda)\right)\nabla_{\lambda} \log \pi(\varepsilon\,;\,\lambda)\right]}_{\text{correction}}$$

- *Intuition:* Interpolating between *score function gradients* and *reparameterization gradients*.

Rejection sampling and reparameterization

- *Idea:*
  - Use $h(\varepsilon, \lambda)$ from the *proposal*
  - Let $\pi(\varepsilon\,;\,\lambda)$ be the distribution of the *accepted* ε
- *Rejection sampling:*
  - Propose $\mathbf{x} = h(\varepsilon, \lambda),\ \varepsilon \sim s(\varepsilon)$
  - Accept if $u \leq a(\varepsilon, \lambda) = \frac{q(h(\varepsilon,\lambda)\,;\,\lambda)\left|\frac{\mathrm{d}h}{\mathrm{d}\varepsilon}(\varepsilon,\lambda)\right|}{M_\lambda s(\varepsilon)},\ u \sim \mathcal{U}[0,1]$
- *Then:*

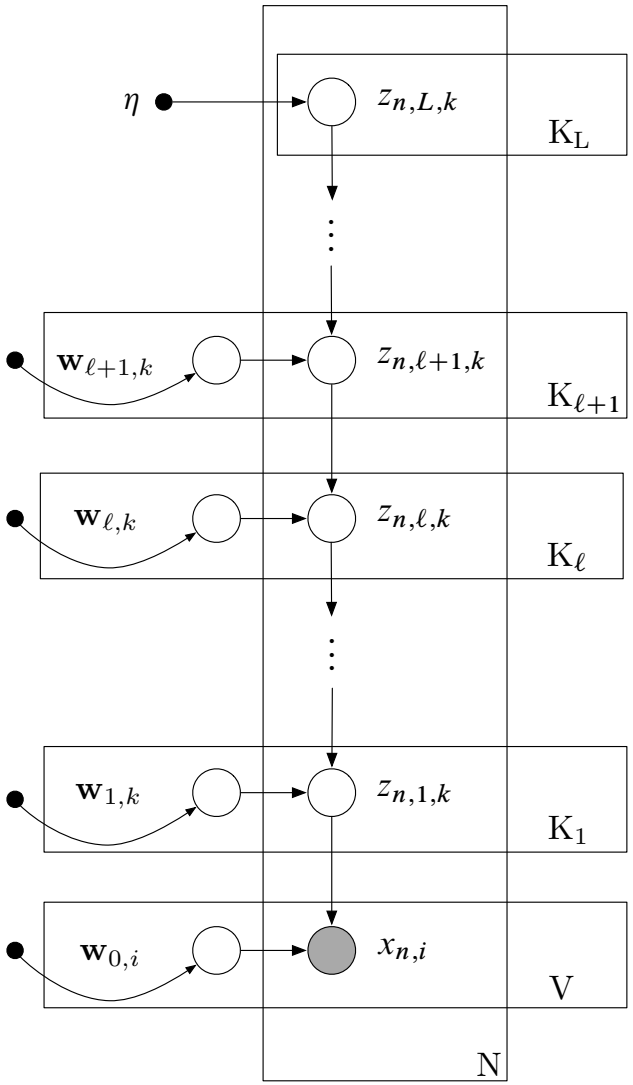
$$\begin{aligned}\pi(\varepsilon\,;\,\lambda) &= \int_0^1 \pi(\varepsilon, u\,;\,\lambda)\mathrm{d}u = M_\lambda s(\varepsilon)\int_0^1 \mathbf{1}[u \leq a(\varepsilon,\lambda)]\mathrm{d}u \\ &= q(h(\varepsilon,\lambda)\,;\,\lambda)\left|\frac{\mathrm{d}h}{\mathrm{d}\varepsilon}(\varepsilon,\lambda)\right|\end{aligned}$$

Sparse gamma deep exponential family

$$\mathbf{w}_{k,k'} \sim \text{Gamma}(0.1,\ 0.1)$$

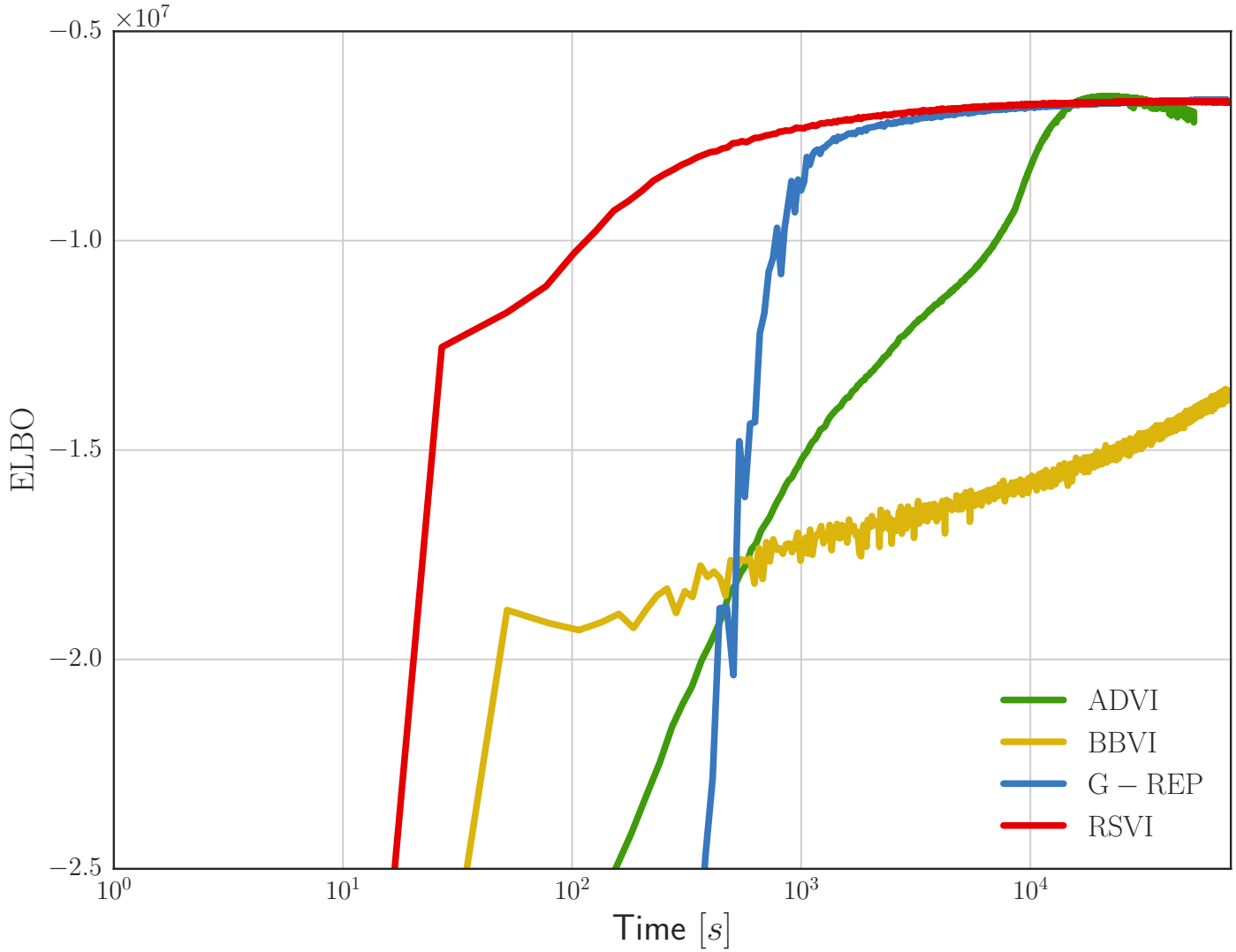
$$\mathbf{x}^{\ell}_{n,k} \sim \text{Gamma}\left(0.1, \frac{0.1}{\sum_{k'} \mathbf{w}^{\ell}_{k,k'} \mathbf{x}^{\ell+1}_{n,k'}}\right)$$

$$\mathbf{y}_{n,d} \sim \text{Poisson}\left(\sum_{k} \mathbf{w}^{0}_{k,d} \mathbf{x}^{1}_{n,k}\right)$$



Deep exponential families, Ranganath et al., 2015

Sparse gamma DEF – Olivetti faces



[Kucukelbir et al., 2015, Ranganath et al., 2014, Ruiz et al., 2016]

Variational Monte Carlo methods

- Monte Carlo methods can enable variational inference for more complex (non-conjugate) models $p(\mathbf{x}, \mathbf{y})$
- But what if we can use Monte Carlo methods to also define more flexible $q(\mathbf{x}\,;\,\lambda)$?
- I will illustrate how we can do this using sequential Monte Carlo for the state space model:

$$p(\mathbf{x}_{1:T}, \mathbf{y}_{1:T}) \triangleq \mu(\mathbf{x}_1) \prod_{t=2}^{T} f(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}) \prod_{t=1}^{T} g(\mathbf{y}_t \,|\, \mathbf{x}_t)$$

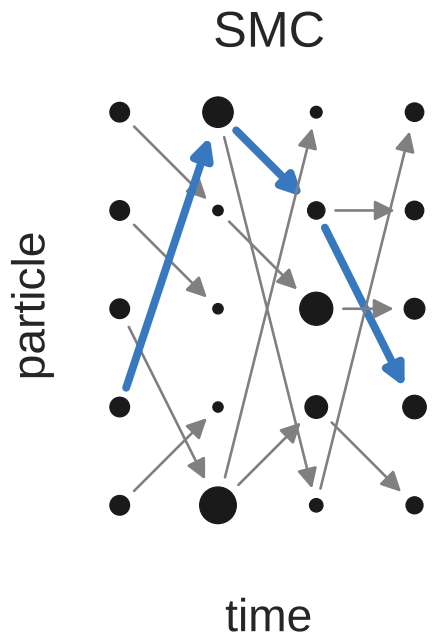
Sequential Monte Carlo

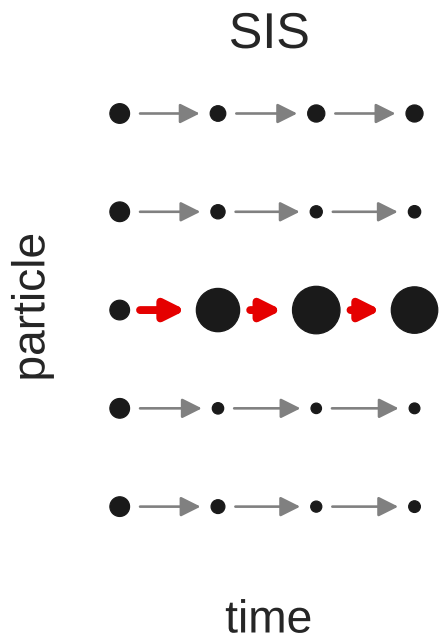
Approximation:

$$\widehat{p}(\mathbf{x}_{1:t-1} \mid \mathbf{y}_{1:t-1}) = \sum_{i=1}^{N} \frac{w_{t-1}^{i}}{\sum_{\ell} w_{t-1}^{\ell}} \delta_{\mathbf{x}_{1:t-1}^{i}}$$

Update:

- $a_{t-1}^{i} \sim \text{Discrete}\left(w_{t-1}^{j} / \sum_{\ell} w_{t-1}^{\ell}\right)$
- $\mathbf{x}_{t}^{i} \sim r(\mathbf{x}_{t} \mid \mathbf{x}_{t-1}^{a_{t-1}^{i}} ; \lambda)$
- $w_{t}^{i} = \frac{f(\mathbf{x}_{t}^{i} \mid \mathbf{x}_{t-1}^{a_{t-1}^{i}}) g(\mathbf{y}_{t} \mid \mathbf{x}_{t}^{i})}{r(\mathbf{x}_{t}^{i} \mid \mathbf{x}_{t-1}^{a_{t-1}^{i}} ; \lambda)}$



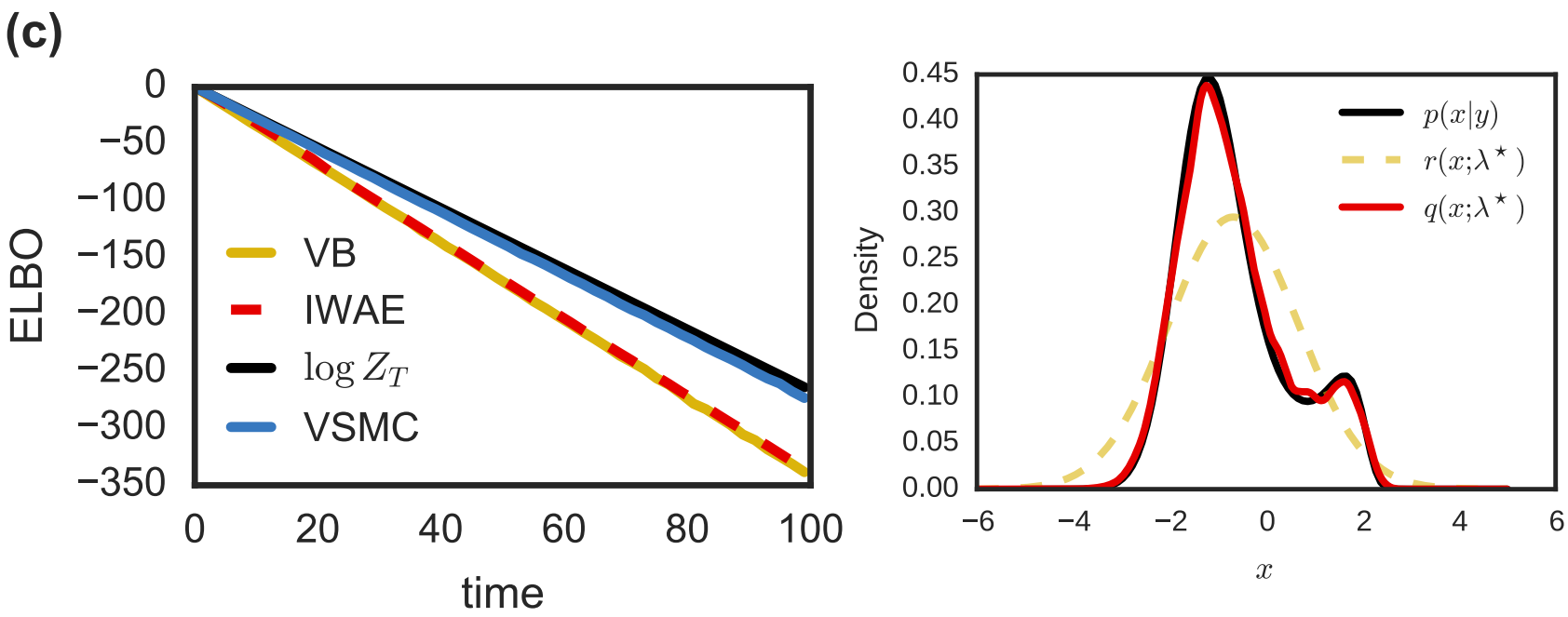


[Gordon et al., 1993, Naesseth et al., 2019]

Variational Sequential Monte Carlo

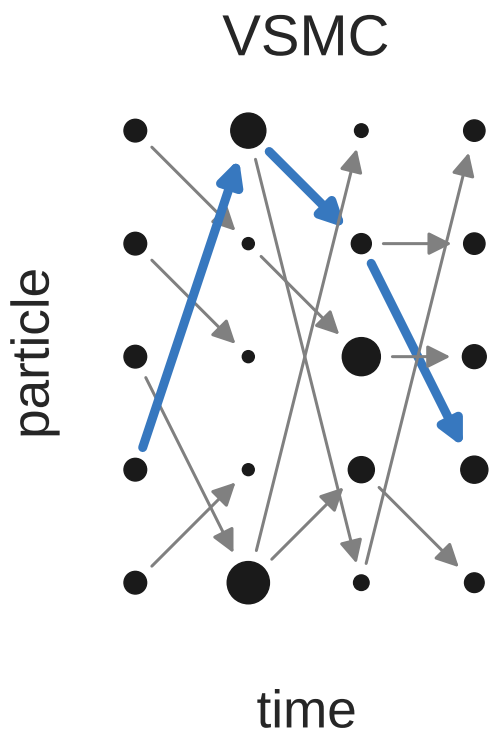
Key idea:

$q(\mathbf{x}_{1:T}; \lambda)$ defined by running SMC and sample $\widehat{p}(\mathbf{x}_{1:T} \,|\, \mathbf{y}_{1:T})$



Variational Sequential Monte Carlo Generative Process

1. Run SMC to get $\left\{\left(\mathbf{x}_{1:T}^{i}, w_T^i\right)\right\}_{i=1}^{N}$
2. $b_T \sim \text{Discrete}\left(w_T^i / \sum_\ell w_T^\ell\right)$
3. **return** $\mathbf{x}_{1:T} \triangleq \mathbf{x}_{1:T}^{b_T}$



Distribution of VSMC

The distribution q is the marginal of $\mathbf{x}_{1:T}^{b_T}$ over all random variables, $\mathbf{x}_{1:T}^{1:N}, a_{1:T-1}^{1:N}, b_T$, generated by VSMC

$$q(\mathbf{x}_{1:T} \mid \mathbf{y}_{1:T}\,;\,\lambda) = p(\mathbf{x}_{1:T}, \mathbf{y}_{1:T})\,\mathbb{E}\left[\widehat{p}(\mathbf{y}_{1:T})^{-1}\middle|\mathbf{x}_{1:T}\right],$$

where $\widehat{p}(\mathbf{y}_{1:T}) \triangleq \prod_{t=1}^{T} \frac{1}{N} \sum_{i=1}^{N} w_t^i$.

Surrogate ELBO

We can not directly use the standard ELBO, $\mathscr{L}(\lambda)$, because evaluating q pointwise is intractable.

Consider instead:

$$\widetilde{\mathscr{L}}(\lambda) \triangleq \mathbb{E}\left[\sum_{t=1}^{T} \log\left(\frac{1}{N}\sum_{i=1}^{N} w_t^i\right)\right] = \mathbb{E}\left[\log \widehat{p}(\mathbf{y}_{1:T})\right]$$

Theorem (Surrogate ELBO)

$$\log p(\mathbf{y}_{1:T}) \geq \mathscr{L}(\lambda) \geq \widetilde{\mathscr{L}}(\lambda).$$

Proof.

(Idea) Use the definition of $\mathscr{L}(\lambda)$ for $q(\mathbf{x}_{1:T} \,|\, \mathbf{y}_{1:T}\,;\, \lambda)$ and then Jensen's inequality. □

Stochastic Optimization

We optimize the surrogate ELBO $\widetilde{\mathscr{L}}(\lambda)$ using stochastic gradient ascent,

$$\nabla\widetilde{\mathscr{L}}(\lambda) = \nabla\mathbb{E}\left[\log\widehat{p}(\mathbf{y}_{1:T})\right] = \mathbb{E}\left[\nabla\log\widehat{p}(\mathbf{y}_{1:T}) + \log\widehat{p}(\mathbf{y}_{1:T})\nabla\log\widetilde{\phi}\right].$$

Variance reduction using the reparameterization trick,
Rao-Blackwellization, and control variates.

Theoretical Results

- For some $c(\lambda) < \infty$,

$$\mathrm{KL}\left(q(\mathbf{x}_{1:T}; \lambda) \middle\| p(\mathbf{x}_{1:T} \mid \mathbf{y}_{1:T})\right) \leq \frac{c(\lambda)}{N}.$$

- With $N = bT$,

$$\mathrm{KL}\left(q(\mathbf{x}_{1:T}; \lambda) \middle\| p(\mathbf{x}_{1:T} \mid \mathbf{y}_{1:T})\right) \leq -\mathbb{E}\left[\log \frac{\widehat{p}(\mathbf{y}_{1:T})}{p(\mathbf{y}_{1:T})}\right]$$

$$\xrightarrow[T \to \infty]{} \frac{\sigma^2(\lambda)}{2b},$$

  for $\sigma^2(\lambda) < \infty$.

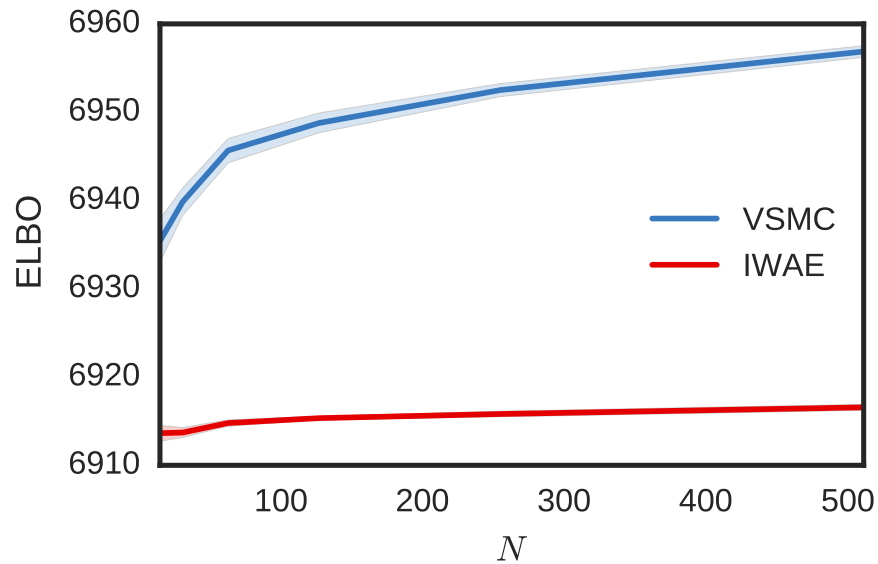
Stochastic volatility

Data: 10 years of monthly returns for the exchange rate of 22 currencies wrt to USD

$$\mathbf{x}_t = \mu + \phi(\mathbf{x}_{t-1} - \mu) + v_t, \qquad v_t \sim \mathcal{N}(0, Q)$$

$$\mathbf{y}_t = \beta \exp\left(\frac{\mathbf{x}_t}{2}\right) e_t, \qquad e_t \sim \mathcal{N}(0, I)$$

	Method	ELBO
	Structured VI	6905.1
$N = 4$	IWAE	6911.2
	VSMC	**6921.6**
$N = 8$	IWAE	6912.4
	VSMC	**6935.8**
$N = 16$	IWAE	6913.3
	VSMC	**6936.6**

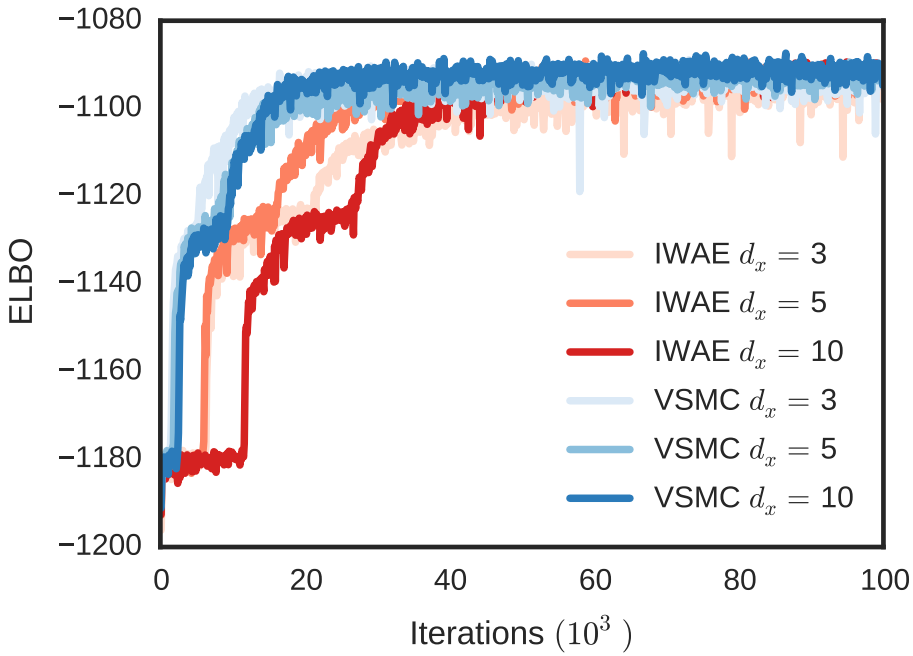


$$r(\mathbf{x}_t | \mathbf{x}_{t-1}\,;\, \lambda, \theta) \propto f(\mathbf{x}_t | \mathbf{x}_{t-1}\,;\, \theta)\, \mathcal{N}(\mathbf{x}_t\,;\, \mu_t, \Sigma_t)$$

Stochastic recurrent neural networks

Data: 105 motor cortex neurons simultaneously recorded in a macaque monkey

$$\mathbf{x}_t = \mu_\theta(\mathbf{x}_{t-1}) + \exp\left(\sigma_\theta(\mathbf{x}_{t-1})/2\right) v_t,$$
$$\mathbf{y}_t \sim \text{Poisson}\left(\exp\left(\eta_\theta(\mathbf{x}_t)\right)\right),$$



Proposal:

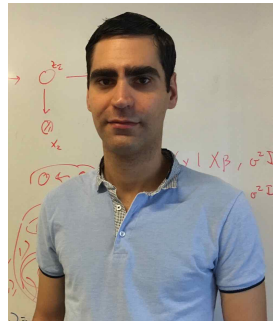
$$r(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_t) \propto f(\mathbf{x}_t|\mathbf{x}_{t-1}\,; \theta)$$
$$\mathcal{N}\left(\mathbf{x}_t\,; \mu_\lambda(\mathbf{y}_t), e^{2\sigma_\lambda(\mathbf{y}_t)}\right),$$

$\mu_\lambda(\cdot), \sigma_\lambda(\cdot)$ are *inference networks*.

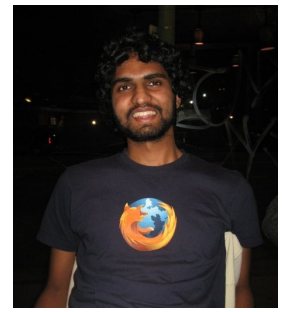
Collaborators

Scott Linderman

Francisco Ruiz



Rajesh Ranganath



David Blei

Fredrik Lindsten

Thomas Schön

Reparameterization Gradients through Acceptance-Rejection Sampling Algorithms, Naesseth et al., 2017

Variational Sequential Monte Carlo, Naesseth et al., 2018

Machine learning using approximate inference: Variational and sequential Monte Carlo methods, Naesseth, 2018

Take-home messages

- Stochastic optimization opens up for interesting new approximate Bayesian inference methods
- Monte Carlo methods ⇒ more flexible variational approximations with guarantees
- Variational methods ⇒ adaptation of Monte Carlo methods based on a global objective

Contact: christian.a.naesseth@columbia.edu
Website: naesseth.github.io

Subsampling and Variational EM

- Data subsampling $\mathbf{y}_{1:T} = \{\mathbf{y}^j_{1:T}\}_{j=1}^n$

$$\widetilde{\mathscr{L}}(\lambda) = \sum_{j=1}^{n} \mathbb{E}\left[\log \widehat{p}(\mathbf{y}^j_{1:T})\right]$$

  At each iteration sample a mini-batch of data.

- Variational EM

$$\log p_\theta(\mathbf{y}_{1:T}) \geq \widetilde{\mathscr{L}}(\lambda, \theta) = \mathbb{E}\left[\log \widehat{p}_\theta(\mathbf{y}_{1:T})\right]$$

  Maximize $\widetilde{\mathscr{L}}(\lambda, \theta)$ jointly with respect to λ, θ