Particle algorithms for maximum likelihood estimation of latent variable models

Juan Kuntz (joint work with Jen Ning Lim and Adam Johansen)
The University of Warwick

More info: Kuntz, J., Lim, J. N., and Johansen, A. M. Particle algorithms for maximum likelihood training of latent variable models. AISTATS 2023 (notable paper); PMLR 206:5134–5180.

Empirical Bayes

Problem setting

Goal: Given some data y, infer some unobserved or latent variables x.

We use a probabilistic model $p_\theta(x, y)$ relating x and y, that is defined in terms of a vector of parameters θ:

$$p_\theta(x, y) \geq 0 \quad \forall \theta \in \Theta, \;\; x \in \mathcal{X}, \;\; y \in \mathcal{Y},$$
$$\int_{\mathcal{X}} \int_{\mathcal{Y}} p_\theta(x, y) dx dy = 1 \quad \forall \theta \in \Theta.$$

Simplification: x and θ take values in Euclidean spaces.

Problem setting

Goal: Given some data y, infer some unobserved or latent variables x.

We use a probabilistic model $p_\theta(x, y)$ relating x and y, that is defined in terms of a vector of parameters θ:

$$p_\theta(x, y) \geq 0 \quad \forall \theta \in \Theta, \;\; x \in \mathcal{X}, \;\; y \in \mathcal{Y},$$

$$\int_{\mathcal{X}} \int_{\mathcal{Y}} p_\theta(x, y) dx dy = 1 \quad \forall \theta \in \Theta.$$

Simplification: x and θ take values in Euclidean spaces.

Example: Toy hierarchical model

Data $y = (y_1, \dots, y_D) \in \mathbb{R}^D$ and latent variables $x = (x_1, \dots, x_D) \in \mathbb{R}^D$, where y_d is a noisy observation of x_d. Model defined by

$$Y_d \sim \mathcal{N}(X_d, 1), \quad X_d \sim \mathcal{N}(\theta, 1), \quad \forall d = 1, \dots, D,$$

$$\Rightarrow p_\theta(x, y) := \prod_{d=1}^{D} \frac{1}{2\pi} \exp\left(-\frac{(x_d - \theta)^2}{2} - \frac{(y_d - x_d)^2}{2} \right).$$

Many supervised and unsupervised learning techniques involve latent variable models.

Example: Generator networks

Non-linear extensions of factor analysis.

Assume that each point in a dataset is generated by:

(A) sampling latent variables from an isotropic Gaussian prior,
(B) mapping them through a neural network,
(C) and adding Gaussian noise.

We consider their use for image datasets (MNIST and CelebA), where

- observed variables y: 1024 pixels per image,
- latent variables x: 64 per image,
- parameters θ: the network's parameters (dimension $\approx 350,000$).

Empirical Bayes

Problem: Given data y, use model $p_\theta(x, y)$ to infer latent variables x.

We approach it using the empirical Bayes (EB) paradigm:

(EB1) we search for parameters θ_* that explain the data y well;
(EB2) we use θ_* to infer, and quantify the uncertainty in, x.

More technically,

(EB1) we find a θ_* maximizing the marginal likelihood,

$$p_\theta(y) := \int p_\theta(x, y)dx;$$

(EB2) we obtain the corresponding posterior distribution,

$$p_{\theta_*}(x|y) := \frac{p_{\theta_*}(x, y)}{p_{\theta_*}(y)}.$$

Empirical Bayes

Problem: Given data y, use model $p_\theta(x, y)$ to infer latent variables x.

We approach it using the empirical Bayes (EB) paradigm:

(EB1) we search for parameters θ_* that explain the data y well;
(EB2) we use θ_* to infer, and quantify the uncertainty in, x.

More technically,

(EB1) we find a θ_* maximizing the marginal likelihood,

$$p_\theta(y) := \int p_\theta(x, y)dx;$$

(EB2) we obtain the corresponding posterior distribution,

$$p_{\theta_*}(x|y) := \frac{p_{\theta_*}(x, y)}{p_{\theta_*}(y)}.$$

Maximum marginal likelihood: In some cases, our main interest is θ_*.

Expectation maximization

A well-known method for solving (EB1,2) is the expectation maximization (EM) algorithm: starting from (θ_0, q_0), alternate

$$\text{(E) } q_{k+1} := p_{\theta_k}(\cdot|y), \qquad \text{(M) } \theta_{k+1} := \arg\max_{\theta\in\Theta} \int \ell(\theta, x) q_{k+1}(x) dx,$$

where $\ell(\theta, x) := \log(p_\theta(x, y))$ denotes the log-likelihood and Θ the parameter space.

Under general conditions,

$$\theta_k \to \theta_* \quad \text{and} \quad q_k \to p_{\theta_*}(\cdot|y) \quad \text{as} \quad k \to \infty,$$

where θ_* is a stationary point of $\theta \mapsto p_\theta(y)$ (i.e. $\nabla_\theta p_{\theta_*}(y) = 0$).

Expectation maximization

A well-known method for solving (EB1,2) is the expectation maximization (EM) algorithm: starting from (θ_0, q_0), alternate

$$\text{(E)}\ q_{k+1} := p_{\theta_k}(\cdot|y), \qquad \text{(M)}\ \theta_{k+1} := \arg\max_{\theta\in\Theta} \int \ell(\theta, x) q_{k+1}(x) dx,$$

where $\ell(\theta, x) := \log(p_\theta(x, y))$ denotes the log-likelihood and Θ the parameter space.

Under general conditions,

$$\theta_k \to \theta_* \quad \text{and} \quad q_k \to p_{\theta_*}(\cdot|y) \quad \text{as} \quad k \to \infty,$$

where θ_* is a stationary point of $\theta \mapsto p_\theta(y)$ (i.e. $\nabla_\theta p_{\theta_*}(y) = 0$).

Issue: The (E,M) steps are intractable for many models.

Typical solutions:

- Approximate (E) using Monte Carlo.
- Approximate (M) using numerical optimization.

EM is a well-known optimization algorithm applied to the free energy:

$$F(\theta, q) := \int \log\left(\frac{q(x)}{p_\theta(x, y)}\right) q(x)dx \quad \forall\theta \in \Theta, \;\; q \in \mathcal{P}(\mathcal{X}),$$

where $\mathcal{P}(\mathcal{X}) :=$ {probability distributions on the latent space $\mathcal{X}$}.

Theorem (Neal and Hinton [1998])

$p_\theta(y)$ has a maximum at θ iff F has a minimum at $(\theta, p_\theta(\cdot|y))$.

Proposition (Neal and Hinton [1998])

For any θ in Θ, the posterior $p_\theta(\cdot|y)$ minimizes $q \mapsto F(\theta, q)$.

Hence, EM is coordinate descent applied to F:

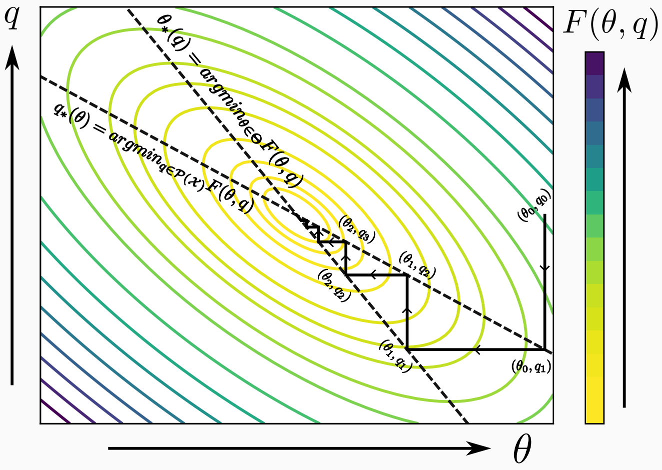
$$\text{(E) } q_{k+1} := p_{\theta_k}(\cdot|y), \qquad \text{(M) } \theta_{k+1} := \arg\max_{\theta\in\Theta} \int \ell(\theta, x) q_{k+1}(x)dx,$$

equals

$$\text{(E) } q_{k+1} := \arg\min_{q\in\mathcal{P}(\mathcal{X})} F(\theta_k, q), \qquad \text{(M) } \theta_{k+1} := \arg\min_{\theta\in\Theta} F(\theta, q_{k+1}).$$

Starting from (θ_0, q_0), alternate

(E) $q_{k+1} := \arg\min_{q \in \mathcal{P}(\mathcal{X})} F(\theta_k, q)$, (M) $\theta_{k+1} := \arg\min_{\theta \in \Theta} F(\theta, q_{k+1})$.



Issue: The (E,M) steps are intractable for many models.

Possible solution: Apply a different optimization algorithm to F!

Observations

- Coordinate descent often outperforms first order methods.
- Unclear whether this is still the case when the coordinate descent updates can only be computed approximately.
- Methods with joint rather than coordinate-wise updates?

Joint updates

Issue: The (E,M) steps are intractable for many models.

Possible solution: Apply a different optimization algorithm to F!

Observations

- Coordinate descent often outperforms first order methods.
- Unclear whether this is still the case when the coordinate descent updates can only be computed approximately.
- Methods with joint rather than coordinate-wise updates?

From [Neal and Hinton, 1998]:

> *"... justifying... as well as algorithms in which the maximization is done with respect to q and θ simultaneously."*

This idea has been taken up enthusiastically in the VI literature where $\mathcal{P}(\mathcal{X})$ is restricted to tractable parametric subset $(q_\phi)_{\phi\in\Phi}$.

Joint updates

Issue: The (E,M) steps are intractable for many models.

Possible solution: Apply a different optimization algorithm to F!

Observations

- Coordinate descent often outperforms first order methods.
- Unclear whether this is still the case when the coordinate descent updates can only be computed approximately.
- Methods with joint rather than coordinate-wise updates?

From [Neal and Hinton, 1998]:

> *"... justifying... as well as algorithms in which the maximization is done with respect to q and θ simultaneously."*

This idea has been taken up enthusiastically in the VI literature where $\mathcal{P}(\mathcal{X})$ is restricted to tractable parametric subset $(q_\phi)_{\phi\in\Phi}$.

Question: Can we directly optimize over $\Theta \times \mathcal{P}(\mathcal{X})$ using, for example, gradient descent (GD)?

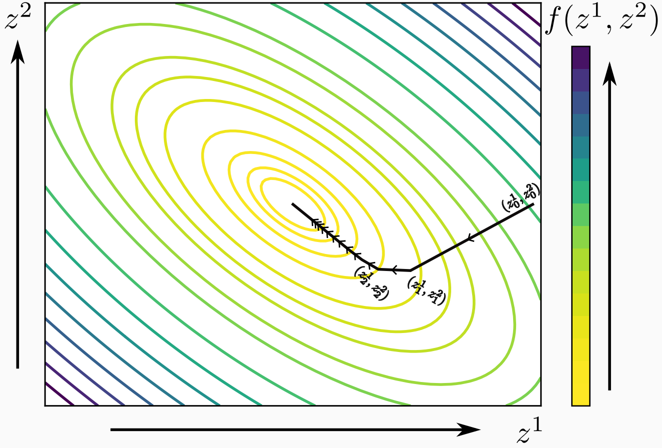
Particle gradient descent

Gradient descent for f

Recall the GD algorithm for optimizing a function $f : \mathbb{R}^n \to \mathbb{R}$,

$$z_{k+1} = z_k - h\nabla f(z_k) \qquad \forall k = 0, 1, \ldots,$$

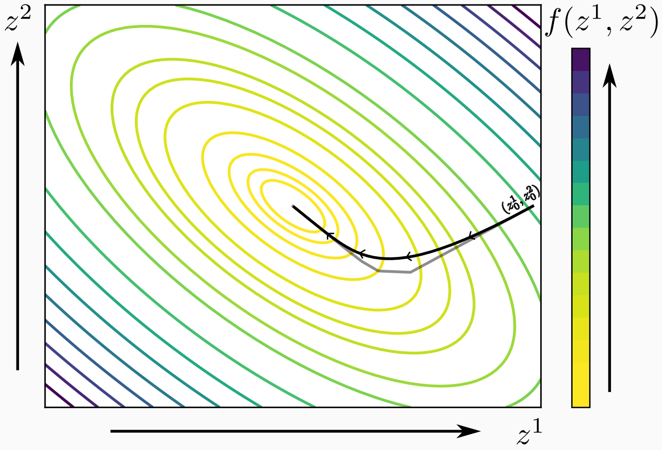
where $h > 0$ denotes the step size.



Gradient flow for f

GD is the Euler discretization of the gradient flow:

$$\dot{z}_t = -\nabla f(z_t) \qquad \forall t \geq 0.$$



Gradient flow for F

We start with a gradient flow,

$$(\dot{\theta}_t, \dot{q}_t) = -\nabla F(\theta_t, q_t) \qquad \forall t \geq 0,$$

and discretize to obtain a practical algorithm.

Gradient flow for F

We start with a gradient flow,

$$(\dot{\theta}_t, \dot{q}_t) = -\nabla F(\theta_t, q_t) \qquad \forall t \geq 0,$$

and discretize to obtain a practical algorithm.

Defining a notion of gradient for a functional on $\Theta \times \mathcal{P}(\mathcal{X})$ requires a metric d. For practical reasons, we use

$$d((\theta_1, q_1), (\theta_2, q_2)) = d_2(\theta_1, \theta_2) + d_{W_2}(q_1, q_2),$$

with

- d_2 denoting the Euclidean metric on Θ and
- d_{W_2} the Wasserstein-2 metric on $\mathcal{P}(\mathcal{X})$.

In which case, $\nabla F(\theta, q) = (\nabla_\theta F(\theta, q), \nabla_q F(\theta, q))$, where

$$\nabla_\theta F(\theta, q) = -\int \nabla_\theta \ell(\theta, x) q(x) dx,$$

$$\nabla_q F(\theta, q) = \nabla_x \cdot \left[q \nabla_x \log \left(\frac{p_\theta(\cdot, y)}{q} \right) \right].$$

The corresponding gradient flow $(\dot{\theta}_t, \dot{q}_t) = -\nabla F(\theta_t, q_t)$ reads

$$\dot{\theta}_t = \int \nabla_\theta \ell(\theta_t, x) q_t(x) dx, \quad \dot{q}_t = \nabla_x \cdot \left[q_t \nabla_x \log \left(\frac{q_t}{p_{\theta_t}(\cdot, y)} \right) \right].$$

Sanity checks [Kuntz et al., 2023]

Theorem (F's stationary points relate to $p_\theta(y)$'s)

$\nabla F(\theta, q) = 0$ if and only if $\nabla_\theta p_\theta(y) = 0$ and $q = p_\theta(\cdot|y)$.

Theorem (Exponential convergence under strong concavity)

If the log likelihood ℓ is strongly concave, for some $\lambda > 0$

$$\nabla^2 \ell(\theta, x) \preceq -\lambda I_{D_x + D_\theta} \quad \forall (\theta, x) \in \Theta \times \mathcal{X},$$

then the marginal likelihood has a unique maximizer θ_* and

$$||\theta_t - \theta_*|| = \mathcal{O}(e^{-\lambda t}) \text{ and } ||q_t - p_{\theta_*}(\cdot|y)||_{L^1} = \mathcal{O}(e^{-\lambda t}).$$

Discretizing the gradient flow, we obtain particle gradient descent:

(1) Choose the step size $h > 0$ and particle number $N > 0$, and run

$$\Theta_k = \Theta_{k-1} + \frac{h}{N}\sum_{n=1}^{N} \nabla_\theta \ell(\Theta_{k-1}, X_{k-1}^n), \quad \forall k \in [K] := \{1, \ldots, K\},$$

$$X_k^n = X_k^n + h\nabla_x \ell(\theta_{k-1}, X_{k-1}^n) + \sqrt{2h}W_{k-1}^n, \quad \forall n \in [N], \; k \in [K],$$

where $(W_k^n)_{k\in[K-1], n\in[N]}$ denote independent standard normal r.v.s., in which case

$$\Theta_k \approx \theta_{kh}, \qquad \frac{1}{N}\sum_{n=1}^{N} \delta_{X_k^n} \approx q_{kh}, \qquad \forall k > 0.$$

(2) Estimate a stationary point θ_* of the marginal likelihood $\theta \mapsto p_\theta(y)$ and its corresponding posterior $p_{\theta_*}(\cdot|y)$ using

$$\theta_* \approx \Theta_K, \qquad p_{\theta_*}(\cdot|y) \approx \frac{1}{N}\sum_{n=1}^{N} \delta_{X_K^n}.$$

Particle gradient descent:

- runs $N > 0$ ULA chains in tandem with an SGD-like recursion;
- avoids accept-reject steps;
- only requires evaluating gradients of $\ell(\theta, x) = \log(p_\theta(x, y))$;
- its cost is $\mathcal{O}(N[\text{eval. cost of } \nabla\ell])$;
- computations can be vectorized across N;
- in big-data settings, we replace $\nabla\ell$ with estimates thereof;
- we improve performance by adapting step sizes.

In short, PGD trains large latent variable models without resorting to variational inference.

Particle gradient descent:

- runs $N > 0$ ULA chains in tandem with an SGD-like recursion;
- avoids accept-reject steps;
- only requires evaluating gradients of $\ell(\theta, x) = \log(p_\theta(x, y))$;
- its cost is $\mathcal{O}(N[\text{eval. cost of } \nabla\ell])$;
- computations can be vectorized across N;
- in big-data settings, we replace $\nabla\ell$ with estimates thereof;
- we improve performance by adapting step sizes.

In short, PGD trains large latent variable models without resorting to variational inference.

Example: Generator networks

We

- fit the model using PGD,
- $10,000$ (MNIST) and $40,000$ (CelebA) training images,
- and a Google Colab subscription.

MNIST CelebA

Figure 1: Synthesized images obtained using the generator trained with PGD.

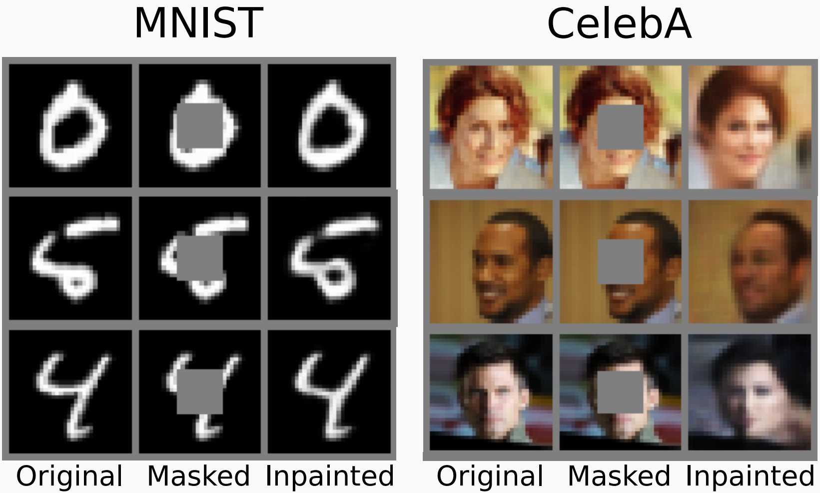


Figure 2: Images reconstructed using the generator trained with PGD.

Thank you for you time.

Questions?

Three intractabilities

Gradient flow:

$$\dot{\theta}_t = \int \nabla_\theta \ell(\theta_t, x) q_t(x) dx, \quad \dot{q}_t = -\nabla_x \cdot \left[q_t \nabla_x \log \left(\frac{p_{\theta_t}(\cdot, y)}{q_t} \right) \right]. \tag{1}$$

Three intractabilities:

(A) The continuous time axis.

(B) The integral over $\mathcal{X}$.

(C) The PDE with domain $\mathcal{X}$.

Three intractabilities

Gradient flow:

$$\dot{\theta}_t = \int \nabla_\theta \ell(\theta_t, x) q_t(x) dx, \quad \dot{q}_t = -\nabla_x \cdot \left[q_t \nabla_x \log \left(\frac{p_{\theta_t}(\cdot, y)}{q_t} \right) \right]. \tag{1}$$

Three intractabilities:

(A) The continuous time axis.

(B) The integral over $\mathcal{X}$.

(C) The PDE with domain $\mathcal{X}$.

Solution: (1) is the a McKean-Vlasov Fokker-Planck equation satisfied by the law of the following McKean SDE:

$$d\theta_t = \left[\int \nabla_\theta \ell(\theta_t, x) q_t(x) dx \right] dt, \quad dX_t = \nabla_x \ell(\theta_t, X_t) dt + \sqrt{2} dW_t,$$

where q_t denotes X_t's law and W a standard Brownian motion.

Two intractabilities

McKean SDE:

$$d\theta_t = \left[\int \nabla_\theta \ell(\theta_t, x) q_t(x) dx\right] dt, \quad dX_t = \nabla_x \ell(\theta_t, X_t) dt + \sqrt{2} dW_t,$$

Two intractabilities:

(1) The continuous time axis.

(2) The integral over $\mathcal{X}$.

Solution: Generate $N > 0$ particles $X_t^1, \ldots, X_t^N$ with law q_t by solving

$$dX_t^n = \nabla_x \ell(\theta_t, X_t^n) dt + \sqrt{2} dW_t^n \quad \forall n \in [N] := \{1, \ldots, N\},$$

with $W^1, \ldots, W^N$ denoting N independent Brownian motions, and use

$$q_t \approx \frac{1}{N} \sum_{n=1}^{N} \delta_{X_t^n} \quad \Rightarrow \quad \int \nabla_\theta \ell(\theta_t, x) q_t(x) dx \approx \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta \ell(\theta_t, X_t^n).$$

One intractability

SDE:

$$d\Theta_t = \left[\frac{1}{N}\sum_{n=1}^{N}\nabla_\theta \ell(\Theta_t, X_t^n)\right] dt,$$

$$dX_t^n = \nabla_x \ell(\Theta_t, X_t^n) dt + \sqrt{2} dW_t^n \;\; \forall n \in [N].$$

One intractability: The continuous time axis.

Solution: Discretize using Euler-Maruyama with step size $h > 0$,

$$\Theta_{k+1} = \theta_k + \frac{h}{N}\sum_{n=1}^{N}\nabla_\theta \ell(\Theta_k, X_k^n), \quad \forall k \in [K],$$

$$X_{k+1}^n = X_k^n + h\nabla_x \ell(\Theta_k, X_k^n) + \sqrt{2h} W_k^n \;\; \forall n \in [N], \;\; k \in [K],$$

where $(W_k^n)_{k\in[K-1], n\in[N]}$ denote independent standard normal r.v.s.

Choose a tractable parametric family $\mathcal{Q} := (q_\phi)_{\phi \in \Phi} \subseteq \mathcal{P}(\mathcal{X})$ and solve

$$(\theta_*, \phi_*) = \operatorname*{arg\,min}_{(\theta,\phi)\in\Theta\times\Phi} F(\theta, q_\phi)$$

using an appropriate optimization algorithm.

General idea: If $\mathcal{Q}$ is rich, then (θ_*, q_{ϕ_*}) will be close to an optimum of $(\theta, q) \mapsto F(\theta, q)$ if (θ_*, ϕ_*) is an optimum of $(\theta, \phi) \mapsto F(\theta, q_\phi)$.

Issues

- How rich does $\mathcal{Q}$ need to be?
- We are interested in optimizing over (θ, q_ϕ) in $\Theta \times \mathcal{Q}$ rather than (θ, ϕ) in $\Theta \times \Phi$. Hence, naively applying an optimization algorithm to $(\theta, \phi) \mapsto F(\theta, q_\phi)$ can lead to trouble.

Particle gradient descent: Behaviour

Given the analogy between PGD and (stochastic) gradient descent, we expect that:

(C1) If the step size h is set too large, PGD will be unstable.

(C2) Otherwise, after a transient phase,

$$\theta_k \approx \theta_*, \quad q_k \approx p_{\theta_*}(\cdot|y), \quad \lim_{k\to\infty} \bar{\theta}_k = \theta_*, \quad \lim_{k\to\infty} \bar{q}_k = p_{\theta_*}(\cdot|y),$$

for some stationary point θ_* of $p_\theta(y)$, where

$$\bar{\theta}_K := \frac{1}{K}\sum_{k=1}^{K} \theta_k, \quad \bar{q}_K := \frac{1}{K}\sum_{k=1}^{K} q_k, \quad \textit{with} \;\; q_k := \frac{1}{N}\sum_{n=1}^{N} \delta_{X_k^n}.$$

(C3) Small hs lead to long transient phases but low estimator variance in the stationary phase.

Particle gradient descent: Behaviour

Given the analogy between PGD and (stochastic) gradient descent, we expect that:

(C1) If the step size h is set too large, PGD will be unstable.
(C2) Otherwise, after a transient phase,

$$\theta_k \approx \theta_*, \quad q_k \approx p_{\theta_*}(\cdot|y), \quad \lim_{k\to\infty} \bar{\theta}_k = \theta_*, \quad \lim_{k\to\infty} \bar{q}_k = p_{\theta_*}(\cdot|y),$$

for some stationary point θ_* of $p_\theta(y)$, where

$$\bar{\theta}_K := \frac{1}{K}\sum_{k=1}^{K}\theta_k, \quad \bar{q}_K := \frac{1}{K}\sum_{k=1}^{K} q_k, \quad \textit{with} \;\; q_k := \frac{1}{N}\sum_{n=1}^{N}\delta_{X_k^n}.$$

(C3) Small hs lead to long transient phases but low estimator variance in the stationary phase.

Asymptotic bias: $(\bar{\theta}_k, \bar{q}_k)$ does not converge exactly to $(\theta_*, p_{\theta_*}(\cdot|y))$, but instead to a point in its vicinity.

- The bias vanishes as $N \to \infty$ and $h \to 0$.

Toy hierarchical model

Recall our starting example:

$$Y_d \sim \mathcal{N}(X_d, 1), \quad X_d \sim \mathcal{N}(\theta, 1), \quad \forall d = 1, \ldots, D,$$

$$p_\theta(x, y) := \prod_{d=1}^{D} \frac{1}{2\pi} \exp\left(-\frac{(x_d - \theta)^2}{2} - \frac{(y_d - x_d)^2}{2}\right).$$

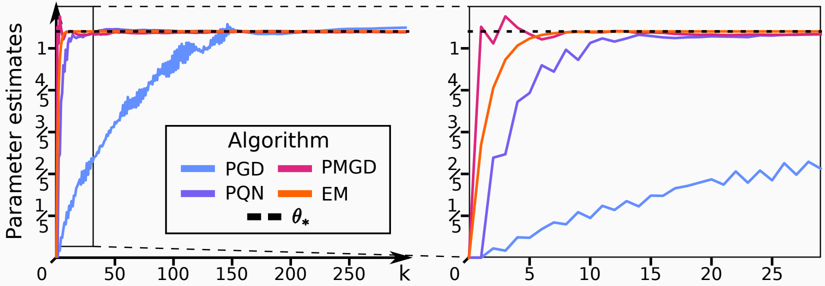


Figure 3: Parameter estimates with $D = 100$ latent variables and $N = 10$ particles. **(LHS)** PGD, PQN, PMGD, and EM estimates with well chosen h and averaging over k once the estimates reach stationarity. **(RHS)** First 30 steps.

Benchmark: SOUL algorithm [De Bortoli et al., 2021]

Alternating coordinate-wise cousin of PGD:

- Approximates the (E) step by running the unadjusted Langevin algorithm (ULA) targetting the current posterior:

$$p_{\theta_k}(\cdot|y) \approx \frac{1}{N}\sum_{n=1}^{N}\delta_{X_k^n}$$

  where

$$X_k^0 = X_{k-1}^N, \qquad X_k^{n+1} = X_k^n + h\nabla_x \ell(\theta_k, X_k^n) + \sqrt{2h}W_k^n \quad \forall n \in [N-1].$$

- Approximates the (M) step using a stochastic gradient step:

$$\theta_{k+1} = \theta_k + \frac{h}{N}\sum_{n=1}^{N}\nabla_\theta \ell(\theta_k, X_k^n).$$

MNIST classification with a Bayesian neural network

We consider

- a scaled-down version of the MNIST classification task,
- involving 1000 data points with labels $4, 9$ and an $80/20$ training/testing split.

We apply

- a two layer Bayesian neural network,
- with isotropic Gaussian priors on the network's weights.
- Latent variables: the network's weights (dimension ≈ 30000).
- Parameters: the prior variances (dimension 2).

Predictive performance

Table 1: Test errors achieved using the final particle cloud $X_{500}^{1:N}$ and corresponding computation times (averaged over 10 replicates).

	$N = 1$		$N = 10$		$N = 100$	
	Error (%)	Time (s)	Error (%)	Time (s)	Error (%)	Time (s)
PGD	7.45 ± 2.03	4.10 ± 0.26	3.20 ± 1.12	10.4 ± 1.2	2.45 ± 0.99	76.6 ± 0.4
PQN	7.45 ± 1.60	4.12 ± 0.21	3.45 ± 1.04	10.0 ± 0.2	2.34 ± 0.81	74.0 ± 0.3
PMGD	7.24 ± 1.75	3.27 ± 0.13	3.75 ± 1.38	9.12 ± 0.2	2.45 ± 0.81	72.1 ± 0.5
SOUL	6.25 ± 1.54	5.02 ± 0.20	7.25 ± 1.38	36.5 ± 0.1	6.85 ± 1.42	364.0 ± 5.3

Gap in performance might be due to:

- SOUL particles are sequentially correlated,

$$X_k^{n+1} = X_k^n + h\nabla_x \ell(\theta_k, X_k^n) + \sqrt{2h}W_k^n \quad \forall n \in [N-1].$$

- $\Rightarrow$ its posterior approximations are narrower than PGD's.

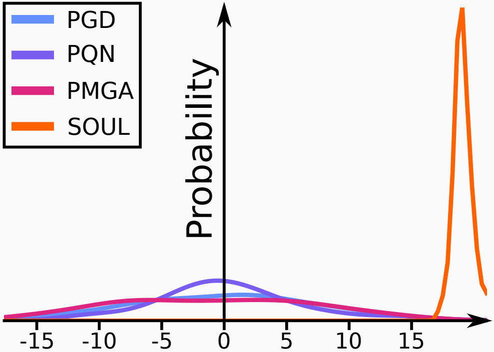


Figure 4: KDE of a randomly-chosen entry of the final cloud $X_{500}^{1:100}$.

Generator networks for image synthesis and inpainting

We consider

- MNIST and CelebA image datasets
- with $10,000$ (MNIST) and $40,000$ (CelebA) training images.

We apply a generator model. It assumes that the images are produced by:

(A) sampling latent variables from an isotropic Gaussian prior,

(B) mapping them through a convolutional neural network,

(C) and adding Gaussian noise.

- Latent variables: 64 per image (totals of $640,000$ and $2,560,000$)
- Parameters: the network's parameters (dimension $\approx 350,000$).

We fit the model using maximum likelihood and PGD (tweaked).

Image synthesis

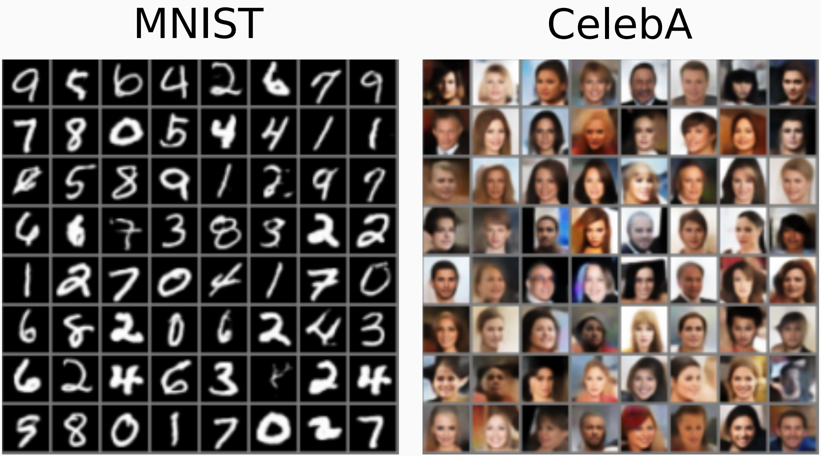


Figure 5: Synthesized images obtained using the generator trained with PGD.

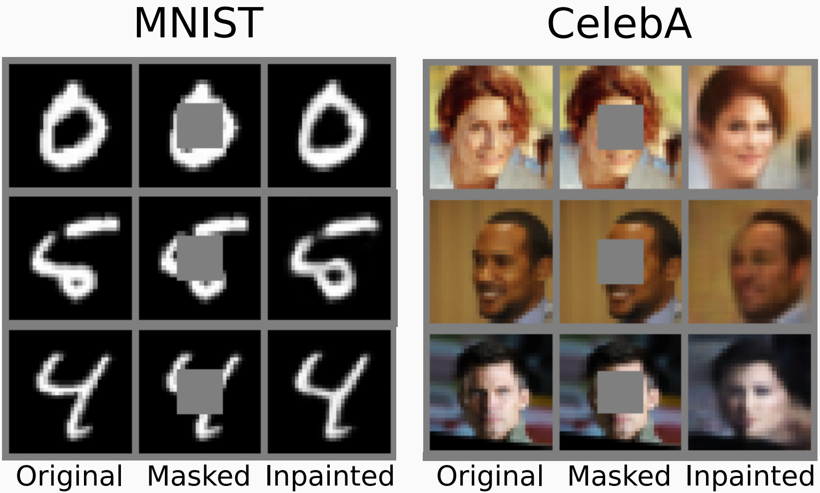


Figure 6: Reconstructed images obtained using the generator trained with PGD.

Advantages

- Applies to broad classes of models.
- Straightforward to implement and tune.
- Recycles posterior approximations.
- Scalable:
    - No accept-reject steps.
    - Low cost: $\mathcal{O}(KN[\text{eval. cost of } (\nabla_\theta \ell, \nabla_x \ell)])$.
    - Easy to parallelize/vectorize computations across particles.

Disadvantages

- Separate timescales: often, $|[\nabla_\theta \ell(\theta, x)]_i| \gg |[\nabla_x \ell(\theta, x)]_j|$ for all i, j.
  Solutions: Hack, particle quasi-Newton, particle marginal GD.
- Biased.
- Only returns stationary points.
- Requires Euclidean parameter and latent spaces.
- Requires differentiable densities.

Open directions

(A) Theoretical analysis.

(B) Variants:

- Big data versions with stochastic gradients à la SGD/SGLD.
- Decreasing h and/or increasing N with k:
  - Robbins-Monro type conditions.
  - Adaptive strategies à la Adagrad and its variants.
  - Line searches.
- Better approximations to the gradient flow.
- Versions for Riemannian manifolds.

(C) Other optimization-inspired methods:

- Different geometries:
  - Stein (leading to an extension of SVGD).
  - Wasserstein-Kalman (leading to an extension of EKS).
  - Better approximations to Newton's method.
- Non-gradient-descent methods:
  - Nesterov acceleration/momentum/underdamped Langevin.
  - Proximal algorithms for non-differentiable models.
  - Mirror descent.

(D) Other particle-based methods updating θ and q 'jointly':

- Metropolis-Hastings Algorithms.

References

V. De Bortoli, A. Durmus, M. Pereyra, and A. Fernandez Vidal. Efficient stochastic optimisation by unadjusted Langevin Monte Carlo. *Statistics and Computing*, 31, 2021.

J. Kuntz, J. N. Lim, and A. M. Johansen. Scalable particle-based alternatives to em. *arXiv preprint arXiv:2204.12965*, 2022.

R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. Springer Netherlands, 1998.

Generator networks for image synthesis and inpainting

Given a dataset of 32×32 images $y^{1:M} := (y^m)_{m=1}^M \subseteq \mathbb{R}^{32\times 32}$.

Model which assumes that each image y^m is independently generated by:

1. drawing a (64-dimensional) latent variable x^m from $\mathcal{N}(0, I)$;
2. mapping x^m to the image space via a generator $f_\theta : \mathbb{R}^{64} \to \mathbb{R}^{32\times 32}$;
3. adding noise: $y^m = f_\theta(x^m) + \epsilon^m$ where $(\epsilon^m)_{m=1}^M$ is a sequence of i.i.d. R.V.s with law $\mathcal{N}(0, 0.01^2 I)$.

In full, the model's density is given by

$$p_\theta(x^{1:M}, y^{1:M}) = \prod_{m=1}^{M} \mathcal{N}(y^m | f_\theta(x^m), 0.01^2 I)\mathcal{N}(x^m | 0, I);$$

and

- f_θ is a convolutional neural net with $\approx 350,000$ parameters,
- there are $640,000$–$2,560,000$ latent variables in total,
- we learn θ by maximizing the marginal likelihood $p_\theta(y^{1:M})$ with PGD (and a few tweaks).

LaSalle's Invariance Principle

$$\dot{\theta}_t = \int \nabla_\theta \ell(\theta_t, x) q_t(x) dx, \quad \dot{q}_t = -\nabla_x \cdot \left[q_t \nabla_x \log\left(\frac{p_{\theta_t}(\cdot, y)}{q_t} \right) \right].$$

Note that

- $\frac{dF(\theta_t, q_t)}{dt} = I(\theta_t, q_t)$ where
  $I(\theta, q) := \left|\left| \int \nabla_\theta \ell(\theta, x) q(x) dx \right|\right|^2 + \int \left|\left| \nabla_x \log\left(\frac{p_\theta(x,y)}{q(x)} \right) \right|\right|^2 q(x) dx.$
- $I \geq 0$. Hence, $t \mapsto F(\theta_t, q_t)$ is non-decreasing.
- Moreover, $I(\theta, q) = 0$ iff $\nabla_\theta p_\theta(y) = 0$ and $q = p_\theta(\cdot|y)$.
- $\Rightarrow$ if $p_\theta(x, y)$ is s.t. F's super-level sets are appropriately compact, an extension of LaSalle's Principle should yield that

$$(\theta_t, q_t) \to \{(\theta_*, p_\theta(\cdot|y)) : \nabla_\theta p_\theta(y) = 0\} \quad \text{as} \quad t \to \infty.$$

Assumption: the marginal likelihood's super-level sets are bounded.

Because $\log(p_\theta(y)) = F(\theta, p_\theta(\cdot|y))$,

$$\{\theta \in \Theta : p_\theta(y) \geq e^l\} \subseteq \{\theta : F(\theta, q) \geq l \text{ for some } q\}.$$

Two sources

(B1) h > 0. Discretizations of the Langevin diffusion do not preserve stationary distributions, c.f. mean field limits in App.G.

(B2) N < ∞. Finite populations, c.f. continuum limits in App.H.

B1 can be mitigated by decreasing h and B2 by increasing N:

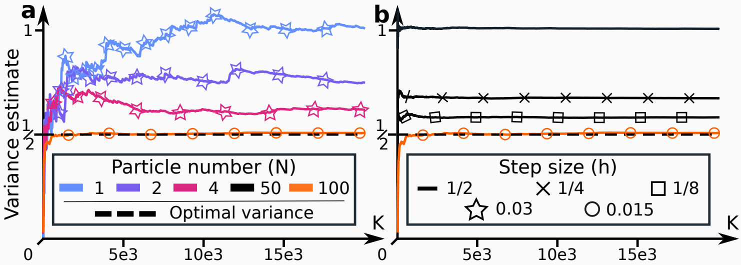


Figure 7: Toy hierarchical model, bias. PMGD posterior variance estimates with $D = 1$ using the time-averaged posterior approximation $\bar{q}_K$ and no burn-in ($k_b = 0$), as a function of K.

Time-scale separation and a hack

For the toy hierarchical model,

$$\nabla_\theta \ell(\theta, x) = \sum_{d=1}^{D} [x_d - \theta], \qquad [\nabla_x \ell(\theta, x)]_d = y_d - x_d - (x_d - \theta) \quad \forall d,$$
$$\Rightarrow |\nabla_\theta \ell(\theta, x)| \gg |[\nabla_x \ell(\theta, x)]_d| \quad \forall d.$$

This causes θ_k to evolve in a faster time-scale than the X_k^ns,

$$\theta_{k+1} = \theta_k + \frac{h}{N} \sum_{n=1}^{N} \sum_{d=1}^{D} [X_{d,k}^n - \theta_k],$$
$$X_{d,k+1}^n = X_{d,k}^n + h[y_d + \theta_k - 2X_{d,k}^n] + \sqrt{2h} W_{d,k}^n \quad \forall d, n,$$

and makes PGD ‘ill-conditioned’.

Time-scale separation and a hack

For the toy hierarchical model,

$$\nabla_\theta \ell(\theta, x) = \sum_{d=1}^{D} [x_d - \theta], \qquad [\nabla_x \ell(\theta, x)]_d = y_d - x_d - (x_d - \theta) \quad \forall d,$$

$$\Rightarrow |\nabla_\theta \ell(\theta, x)| \gg |[\nabla_x \ell(\theta, x)]_d| \quad \forall d.$$

This causes θ_k to evolve in a faster time-scale than the X_k^ns,

$$\theta_{k+1} = \theta_k + \frac{h}{DN} \sum_{n=1}^{N} \sum_{d=1}^{D} [X_{d,k}^n - \theta_k] = \theta_k + h\left[\bar{X}_k - \theta_k\right],$$

$$X_{d,k+1}^n = X_{d,k}^n + h[y_d + \theta_k - 2X_{d,k}^n] + \sqrt{2h} W_{d,k}^n \quad \forall d, n,$$

and makes PGD 'ill-conditioned', where $\bar{X}_k := \frac{1}{DN} \sum_{n=1}^{N} \sum_{d=1}^{D} X_{d,k}^n$.

It is straightforward to mitigate issue with a hack.

PMGD

For some models the (M) step is tractable:

- For each q, $\theta \mapsto F(\theta, q)$ has a unique stationary point $\theta_*(q)$.
- We can evaluate $\theta_*(x^{1:N}) := \theta_*(q)$ whenever $q = N^{-1} \sum_{n=1}^{N} \delta_{x^n}$ for some $x^{1:N} = (x^1, \ldots, x^N)$ in $\mathcal{X}^N$.

Consider the 'marginal objective': $F_*(q) := F(\theta_*(q), q)$ for all q.

Theorem (Kuntz et al. [2022])
$\theta = \theta_*(q)$ and $\nabla F_*(q) = 0$ if and only if $\nabla_\theta p_\theta(y) = 0$ and $q = p_\theta(\cdot|y)$.

PMGD

For some models the (M) step is tractable:

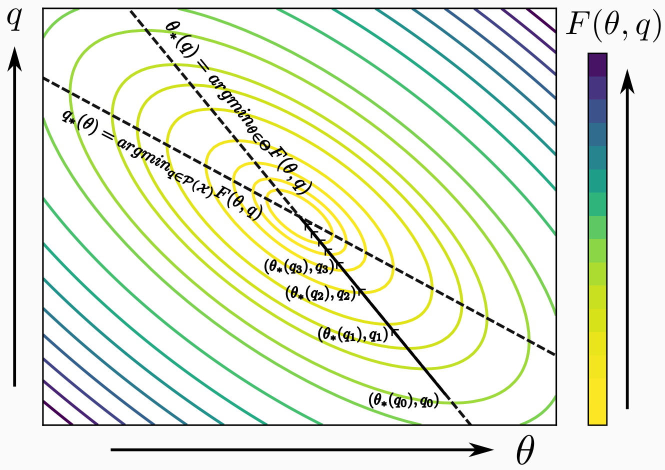
- For each q, $\theta \mapsto F(\theta, q)$ has a unique stationary point $\theta_*(q)$.
- We can evaluate $\theta_*(x^{1:N}) := \theta_*(q)$ whenever $q = N^{-1}\sum_{n=1}^{N}\delta_{x^n}$ for some $x^{1:N} = (x^1, \ldots, x^N)$ in $\mathcal{X}^N$.

Consider the 'marginal objective': $F_*(q) := F(\theta_*(q), q)$ for all q.

Theorem (Kuntz et al. [2022])
$\theta = \theta_*(q)$ and $\nabla F_*(q) = 0$ if and only if $\nabla_\theta p_\theta(y) = 0$ and $q = p_\theta(\cdot|y)$.

Idea: Using gradient descent, find q minimizing F_* and evaluate $\theta_*(q)$.

Idea: Using gradient descent, find q minimizing F_* and evaluate $\theta_*(q)$.



PMGD

Idea: Using gradient descent, find q minimizing F_* and evaluate $\theta_*(q)$. Using the Wasserstein-2 metric on $\mathcal{P}(\mathcal{X})$ leads to

$$\nabla F_*(q) = \nabla_x \cdot \left[q \nabla_x \log \left(\frac{p_{\theta_*(q)}(\cdot, y)}{q} \right) \right].$$

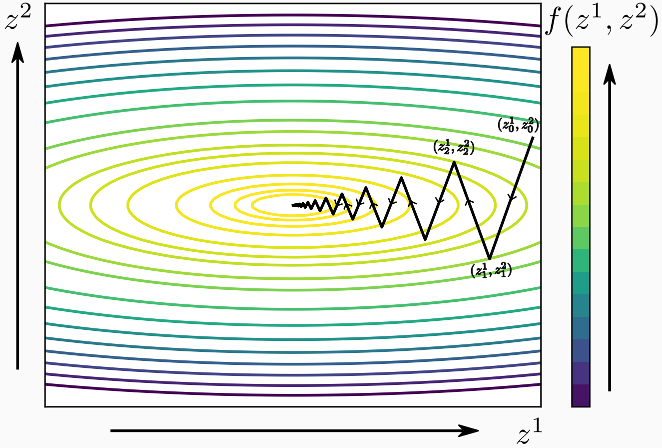
Approximating the corresponding gradient-flow similarly as for PGD then yields the particle marginal gradient descent (PMGD) algorithm:

$$X_{k+1}^n = X_k^n + h \nabla_x \ell(\theta_k, X_k^n) + \sqrt{2h} W_k^n \quad \forall n \in [N],$$

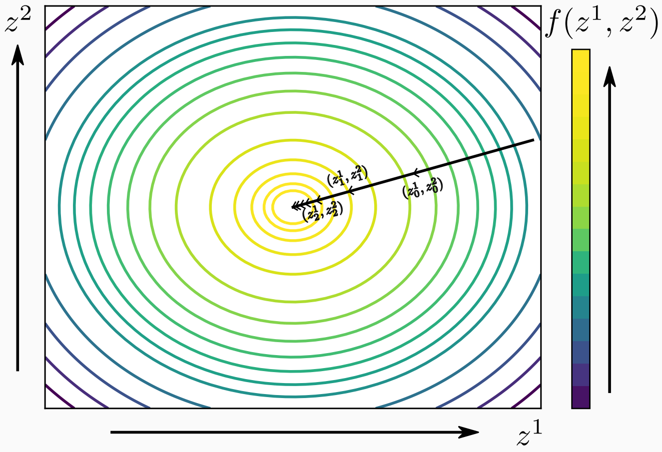
where

$$\theta_k := \theta_*(q_k) \quad \text{with} \quad q_k := \frac{1}{N} \sum_{n=1}^{N} \delta_{X_k^n}.$$

Gradient descent works badly if f is ill-conditioned.



Gradient descent works well if f is well-conditioned.



Newton's method

f is well-conditioned if

$$f(z+hv) \approx f(z) + h\left\langle \nabla f(z), v \right\rangle + \frac{h^2}{2}\left\langle v, v \right\rangle + o(h^2).$$

Newton's method

f is well-conditioned if

$$f(z+hv) \approx f(z) + h\,\langle \nabla f(z), v\rangle + \frac{h^2}{2}\,\langle v, v\rangle + o(h^2).$$

Idea: If f is ill-conditioned, change the inner product $\langle \cdot, \cdot \rangle$ so that it becomes well-conditioned! By Taylor's Theorem,

$$\begin{aligned} f(z+hv) &= f(z) + h\,\langle \nabla f(z), v\rangle + \frac{h^2}{2}\,\left\langle \mathcal{H}_f(z)v, v\right\rangle + o(h^2) \\ &= f(x) + h\,\left\langle [\mathcal{H}_f(z)]^{-1}\nabla f(z), v\right\rangle_z + \frac{h^2}{2}\,\langle v, v\rangle_z + o(h^2), \\ &= f(x) + h\,\left\langle \tilde{\nabla} f(z), v\right\rangle_z + \frac{h^2}{2}\,\langle v, v\rangle_z + o(h^2), \end{aligned}$$

where

$$\mathcal{H}_f = (\partial^2 f/\partial z_i \partial z_j)_{ij}, \quad \langle v, v\rangle_z := \left\langle \mathcal{H}_f(z)v, v\right\rangle, \quad \tilde{\nabla} f(z) := [\mathcal{H}_f(z)]^{-1}\nabla f(z).$$

f is well-conditioned if

$$f(z+hv) \approx f(z) + h\,\langle \nabla f(z), v\rangle + \frac{h^2}{2}\,\langle v, v\rangle + o(h^2).$$

Idea: If f is ill-conditioned, change the inner product $\langle\cdot,\cdot\rangle$ so that it becomes well-conditioned! By Taylor's Theorem,

$$\begin{aligned} f(z+hv) &= f(z) + h\,\langle \nabla f(z), v\rangle + \frac{h^2}{2}\,\left\langle \mathcal{H}_f(z)v, v\right\rangle + o(h^2) \\ &= f(x) + h\,\left\langle [\mathcal{H}_f(z)]^{-1}\nabla f(z), v\right\rangle_z + \frac{h^2}{2}\,\langle v, v\rangle_z + o(h^2), \\ &= f(x) + h\,\left\langle \tilde{\nabla} f(z), v\right\rangle_z + \frac{h^2}{2}\,\langle v, v\rangle_z + o(h^2), \end{aligned}$$

where

$$\mathcal{H}_f = (\partial^2 f/\partial z_i \partial z_j)_{ij}, \quad \langle v, v\rangle_z := \left\langle \mathcal{H}_f(z)v, v\right\rangle, \quad \tilde{\nabla} f(z) := [\mathcal{H}_f(z)]^{-1}\nabla f(z).$$

Newton's: Take steps following $\tilde{\nabla} f$ (i.e. the gradient under the geometry that makes f isotropic).

Particle Quasi-Newton

(A) Using a 2nd order expansion of F, identify an analogous inner product and the corresponding gradient.

(B) Approximate until we get a tractable gradient.

(C) Define the corresponding gradient flow.

(D) Approximate the flow similarly as for PGD and PMGD.

Particle Quasi-Newton (PQN) Algorithm:

$$\theta_{k+1} = \theta_k + h \left[\sum_{n=1}^{N} \mathcal{H}_\theta(X_k^n) \right]^{-1} \sum_{n=1}^{N} \nabla_\theta \ell(\theta_k, X_k^n), \quad \forall k \in [K],$$

$$X_{k+1}^n = X_k^n + h \nabla_x \ell(\theta_k, X_k^n) + \sqrt{2h} W_k^n \;\; \forall n \in [N], \;\; k \in [K].$$

Discretizing the flow

Gradient flow:

$$\dot{\theta}_t = \int \nabla_\theta \ell(\theta_t, x) q_t(x) dx, \quad \dot{q}_t = -\nabla_x \cdot \left[q_t \nabla_x \log \left(\frac{p_{\theta_t}(\cdot, y)}{q_t} \right) \right]. \tag{2}$$

Intractabilities

(A) The continuous time axis.
(B) The integral over $\mathcal{X}$.
(C) The PDE with domain $\mathcal{X}$.

Discretizing the flow

Gradient flow:

$$\dot{\theta}_t = \int \nabla_\theta \ell(\theta_t, x) q_t(x) dx, \quad \dot{q}_t = -\nabla_x \cdot \left[q_t \nabla_x \log \left(\frac{p_{\theta_t}(\cdot, y)}{q_t} \right) \right]. \qquad (2)$$

Intractabilities

(A) The continuous time axis.
(B) The integral over $\mathcal{X}$.
(C) The PDE with domain $\mathcal{X}$.

Solutions

(C) Eq. (2) is satisfied by the law of a McKean SDE:

$$d\theta_t = \left[\int \nabla_\theta \ell(\theta_t, x) q_t(x) dx \right] dt, \quad dX_t = \nabla_x \ell(\theta_t, X_t) dt + \sqrt{2} dW_t,$$

where q_t denotes X_t's law and W a standard Brownian motion.

(B) Generate N i.i.d. copies $X_t^1, \ldots, X_t^N$ of X_t so that $q_t \approx \frac{1}{N} \sum_{n=1}^N \delta_{X_t^n}$.
(A) Discretize using Euler-Maruyama.

Discretizing the flow

Gradient flow:

$$\dot{\theta}_t = \int \nabla_\theta \ell(\theta_t, x) q_t(x) dx, \quad \dot{q}_t = -\nabla_x \cdot \left[q_t \nabla_x \log \left(\frac{p_{\theta_t}(\cdot, y)}{q_t} \right) \right]. \tag{2}$$

Intractabilities

(A) The continuous time axis.
(B) The integral over $\mathcal{X}$.
(C) The PDE with domain $\mathcal{X}$.

Solutions

(C) Eq. (2) is satisfied by the law of a McKean SDE:

$$d\theta_t = \left[\int \nabla_\theta \ell(\theta_t, x) q_t(x) dx \right] dt, \quad dX_t = \nabla_x \ell(\theta_t, X_t) dt + \sqrt{2} dW_t,$$

where q_t denotes X_t's law and W a standard Brownian motion.
(B) Generate N i.i.d. copies $X_t^1, \ldots, X_t^N$ of X_t so that $q_t \approx \frac{1}{N} \sum_{n=1}^N \delta_{X_t^n}$.
(A) Discretize using Euler-Maruyama.

Discretizing the flow

Gradient flow:

$$\dot{\theta}_t = \int \nabla_\theta \ell(\theta_t, x) q_t(x) dx, \quad \dot{q}_t = -\nabla_x \cdot \left[q_t \nabla_x \log \left(\frac{p_{\theta_t}(\cdot, y)}{q_t} \right) \right]. \qquad (2)$$

Intractabilities

(A) The continuous time axis.

(B) The integral over $\mathcal{X}$.

(C) The PDE with domain $\mathcal{X}$.

Solutions

(C) Eq. (2) is satisfied by the law of a McKean SDE:

$$d\theta_t = \left[\int \nabla_\theta \ell(\theta_t, x) q_t(x) dx \right] dt, \quad dX_t = \nabla_x \ell(\theta_t, X_t) dt + \sqrt{2} dW_t,$$

where q_t denotes X_t's law and W a standard Brownian motion.

(B) Generate N i.i.d. copies $X_t^1, \ldots, X_t^N$ of X_t so that $q_t \approx \frac{1}{N} \sum_{n=1}^N \delta_{X_t^n}$.

(A) Discretize using Euler-Maruyama.