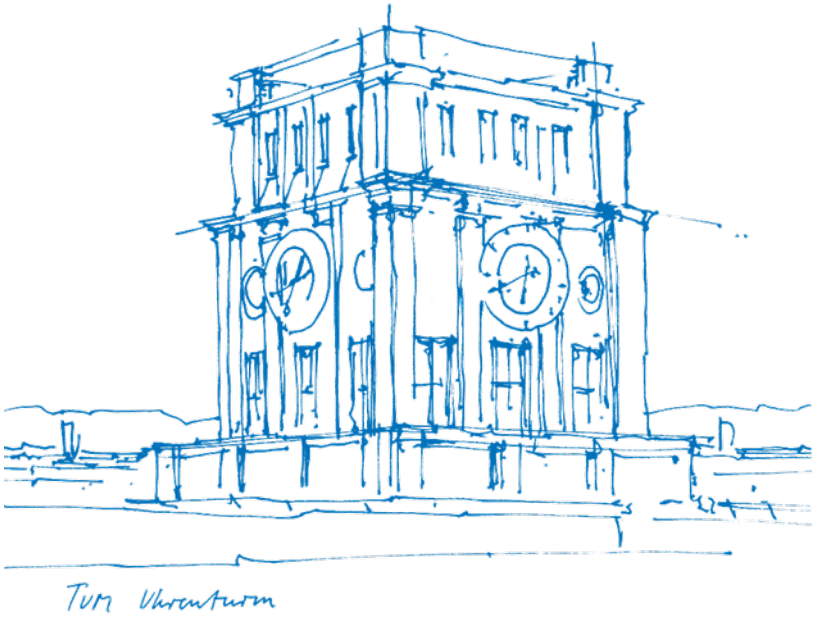
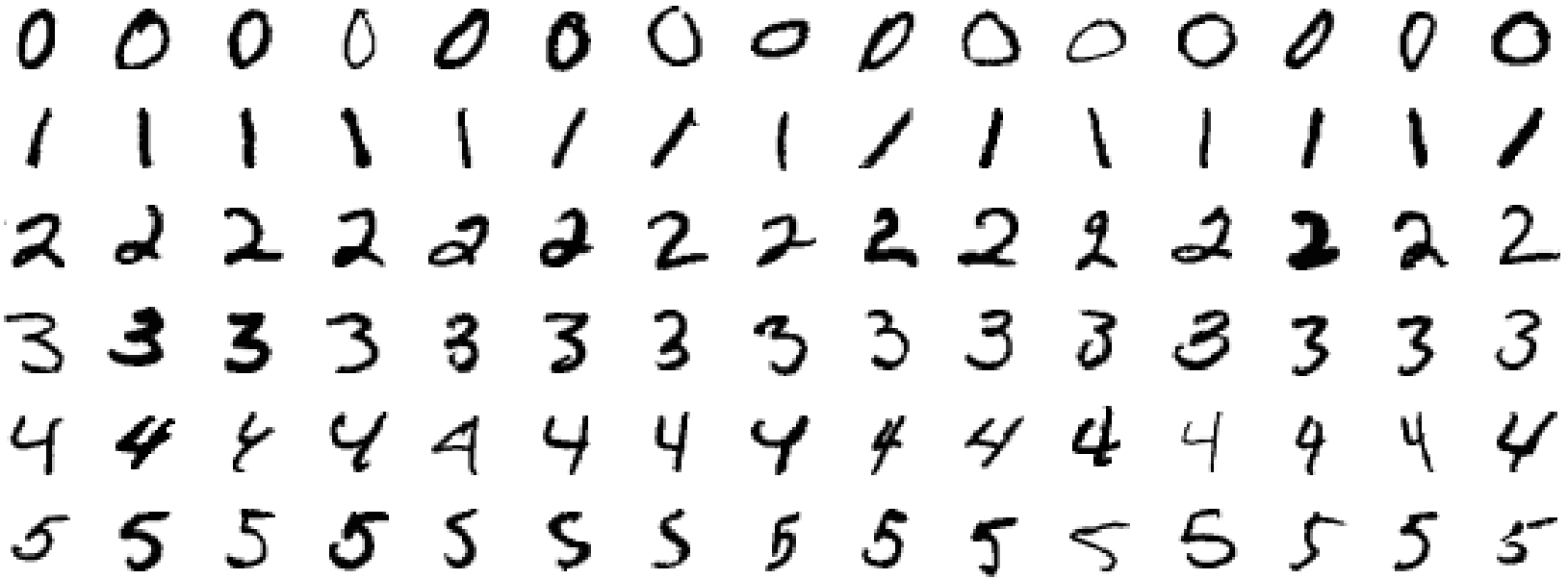
Bayesian Deep Learning – A Stochastic Dynamics Perspective

Niklas Bühler
Technical University of Munich
TUM School of Computation, Information and Technology
Computer Vision Group
Munich, 17 January 2023



Niklas Bühler
Technical University of Munich
TUM School of Computation, Information and Technology
Computer Vision Group
Munich, 17 January 2023

Who has trained a MNIST classifier before?



[Deng, 2012]

What would your model output?

notMNIST Dataset

Bayesian Deep Learning – A Stochastics Dynamics Perspective

Bayesian Deep Learning

- Alternative approach for training and evaluating neural networks
- Results in probability distribution instead of fixed set of parameters

Stochastic Dynamics

- Study of systems that evolve over time and are subject to random fluctuations
- Framework for modeling stochastic training processes of Bayesian neural networks

Rev. Thomas Bayes
Terence O'Donnell,

History of Life Insurance in Its Formative Years, 1936

Outline

Bayesian Deep Learning

Application of Bayesian statistics to deep learning.

Bayes' Theorem

$$P(w \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid w)P(w)}{P(\mathcal{D})} \tag{1}$$

Benefits

- Assess degree of certainty of predictions
- Richer representations through cheap model averaging
- Regularization
- Inject expert knowledge via priors

Frequentist and Bayesian Neural Networks

Frequentist Learning

- MLE: $\arg\max_w P(\mathcal{D} \mid w)$
- Single "optimal" set of values $w \in \mathbb{R}^d$
- Predictions using fixed parameters
- Often overfits the data
- Overly confident decisions

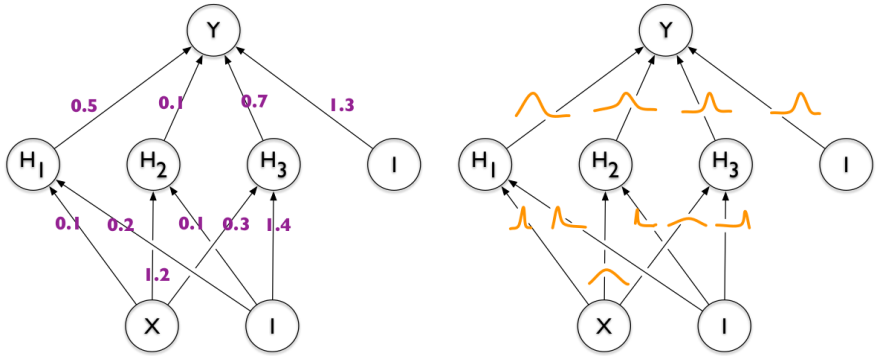


Figure taken from [Blundell et al., 2015].

Bayesian Neural Networks

- Bayesian inference: $P(w \mid \mathcal{D})$
- Weights are represented by probability distributions $w = P(w \mid \mathcal{D})$
- Predictions via uncountably infinite ensemble of networks f

$$\hat{y}_* = \mathbb{E}_{P(w|\mathcal{D})}[f(x_*, w)] = \int f(x_*, w) P(w \mid \mathcal{D}) dw \qquad (2)$$

Motivation

Goal

Train Bayesian neural networks.

Problem

Exact Bayesian inference of $P(w \mid \mathcal{D})$ is typically intractable!

Possible Solutions

1. Variational inference via **Bayes by Backprop** algorithm
2. Sample from posterior via **Hamiltonian Monte Carlo** algorithm

Outline

Variational Bayesian Learning: Bayes by Backprop

Variational inference via *Bayes by Backprop* algorithm [Blundell et al., 2015].

Goal

Variational approximation of posterior $P(w \mid \mathcal{D}) \approx q(w \mid \theta)$.

Find parameters θ^* of this variational distribution $q(w \mid \theta)$.

Solution

Minimize KL divergence from the true Bayesian posterior of the weights:

$$\begin{aligned}\theta^* &= \arg\min_\theta \mathrm{KL}[q(w \mid \theta) \mid\mid P(w \mid \mathcal{D})] \\ &= \arg\min_\theta \int q(w \mid \theta) \log \frac{q(w \mid \theta)}{P(w)P(\mathcal{D} \mid w)} dw \\ &= \arg\min_\theta \mathrm{KL}[q(w \mid \theta) \mid\mid P(w)] - \mathbb{E}_{q(w|\theta)}[\log P(\mathcal{D} \mid w)] = \arg\min_\theta \mathcal{F}(\mathcal{D}, \theta)\end{aligned} \tag{3}$$

Resulting cost function is called *Variational Free Energy* or *Expected Lower Bound (ELBO)*.

Unbiased Monte Carlo Gradients

Evaluate $\mathcal{F}(\mathcal{D}, \theta)$ using Monte Carlo method and a generalization of the Gaussian reparameterization trick.

Monte Carlo approximation of $\mathcal{F}(\mathcal{D}, \theta)$:
Draw Monte Carlo samples $w^{(i)} \sim q(w^{(i)} \mid \theta)$ and approximate

$$\begin{aligned} \mathcal{F}(\mathcal{D}, \theta) &= \int q(w \mid \theta) \log \frac{q(w \mid \theta)}{P(w)} dw - \mathbb{E}_{q(w|\theta)}[\log P(\mathcal{D} \mid w)] \\ &\approx \sum_{i=1}^{n} \log q(w^{(i)} \mid \theta) - \log P(w^{(i)}) - \log P(\mathcal{D} \mid w^{(i)}) \end{aligned} \tag{4}$$

Gaussian Variational Posterior

Assume Gaussian weights and parameterize them using independent noise $\epsilon \sim \mathcal{N}(0, I)$:

$$\begin{aligned} w &= \mu + \sigma \circ \epsilon \\ &= \mu + \log(1 + \exp(\rho)) \circ \epsilon \end{aligned} \tag{5}$$

Standard deviation parameterized as $\sigma = \log(1 + \exp(\rho))$.
Variational posterior parameters: $\theta = (\mu, \rho)$.

Each weight has two degrees of freedom (in μ and ρ), doubling the number of parameters.

Optimization process in Gaussian case

1. Sample $\epsilon \sim \mathcal{N}(0, I)$.
2. Let $w = \mu + \log(1 + \exp(\rho)) \circ \epsilon$.
3. Let $\theta = (\mu, \rho)$.
4. Let $f(w, \theta) = \log q(w \mid \theta) - \log P(w)P(\mathcal{D} \mid w)$.
5. Calculate the gradient w.r.t. the mean

$$\Delta_\mu = \frac{\partial f(w, \theta)}{\partial w} + \frac{\partial f(w, \theta)}{\partial \mu}. \tag{6}$$

6. Calculate the gradient w.r.t. the standard deviation parameter ρ

$$\Delta_\rho = \frac{\partial f(w, \theta)}{\partial w} \frac{\epsilon}{1 + \exp(-\rho)} + \frac{\partial f(w, \theta)}{\partial \rho}. \tag{7}$$

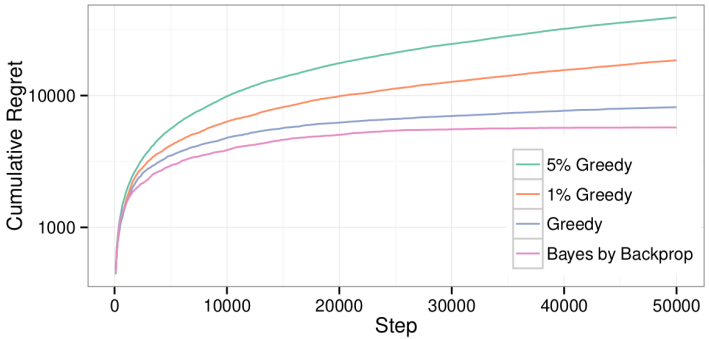
7. Update the variational parameters

$$\begin{aligned} \mu &\leftarrow \mu - \alpha \Delta_\mu \\ \rho &\leftarrow \rho - \alpha \Delta_\rho. \end{aligned} \tag{8}$$

Performance of Bayesian Neural Networks

Reinforcement Learning

- Regret: measure of difference between reward achievable by oracle and reward achieved by agent
- ϵ-greedy policy: with probability ϵ uniformly random, otherwise proposed solution



[Blundell et al., 2015]

Classification on MNIST

- Test error of 1.32% instead of 1.6% for FFNN
- Similar to dropout (1.3%)
- With 5% of parameters still 1.29% instead of 1.24%

Proportion removed	# Weights	Test Error
0%	2.4m	1.24%
50%	1.2m	1.24%
75%	600k	1.24%
95%	120k	1.29%
98%	48k	1.39%

[Blundell et al., 2015]

Outline

Hamiltonian Monte Carlo

Problem

Posterior distributions for Bayesian deep learning typically don't allow direct sampling.

Solutions

- *Markov Chain Monte Carlo (MCMC)* methods
  - Indirect sampling from target distribution
- *Metropolis-Hastings (MH)*
  - Simple MCMC method
  - Random walk approach
- *Hamiltonian Monte Carlo (HMC)*
  - More efficient state space exploration
  - Hamiltonian dynamics for distant, high-quality state proposals

Outline

Markov Chain Monte Carlo Methods

MCMC Methods

- Alternative to variational inference for sampling from posterior
- Generate Markov chain that converges to target distribution

Metropolis-Hastings Algorithm

- Can draw samples from any distribution for which the density can be evaluated (up to normalizing constant)
- Propose new sample x' given current sample x
- New sample x' is accepted or rejected dependent on

$$\frac{P(x')}{P(x)} \tag{9}$$

Metropolis-Hastings Algorithm

Parameters

- Let $f(x) \propto P(x)$ function proportional to target distribution $P(x)$
- Pick arbitrary starting point x_t
- Proposal density $g(x \mid y)$ suggests candidate
  e.g. $g(x \mid y) = \mathcal{N}(y, \sigma) \Rightarrow$ random walk

Steps

1. Draw proposal $x' \sim g(x' \mid x_t)$
2. Acceptance ratio

$$\alpha = \frac{f(x')}{f(x_t)} = \frac{P(x')}{P(x_t)}, \tag{10}$$

3. Accept x' with probability α, otherwise $x_{i+1} = x_t$

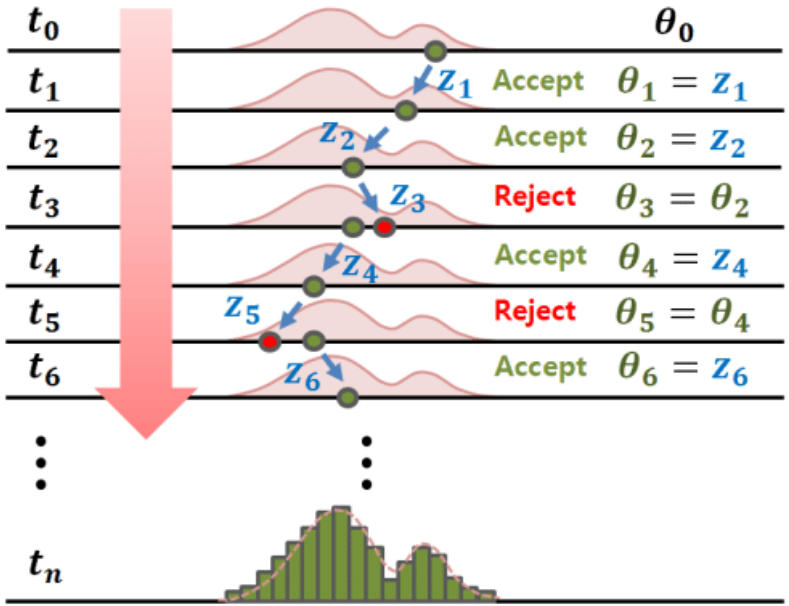


Figure taken from [Jin et al., 2019].

Outline

Hamiltonian Dynamics: Motivation

Problem with MH algorithm

- Random walk approach blindly proposes candidates from vicinity of current sample
- Inefficient as rejection rate is high

Solution

- Hamiltonian Monte Carlo algorithm simulates Hamiltonian dynamics
- Generates better-informed candidates, thus more efficient exploration

Hamiltonian Dynamics: 2D Analogy

Frictionless puck on surface of varying height with state

1. **Position** $q \in \mathbb{R}^2$ of the puck,
2. **Momentum** $p \in \mathbb{R}^2$ of the puck (mass times velocity).

Potential energy $U(q)$ proportional to height of surface at position q.
Kinetic energy $K(p)$ equal to $|p|^2/(2m)$ with m equal to mass.

Level surface: Constant velocity equal to p/m.
Rising slope: Potential energy increases, kinetic energy decreases.
Turning point: Potential energy decreases, kinetic energy increases.

Position variables are variables of interest.
Momentum variables are artificially introduced auxiliary variables.

©Taylor Friehl, www.unsplash.com

Hamiltonian Dynamics

Equations of motion

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i} \qquad \frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i} \tag{11}$$

Hamiltonian H can usually be written as sum of potential and kinetic energy:

$$H(q,p) = U(q) + K(p). \tag{12}$$

Hamilton's equations simplify:

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i} = \frac{\partial K}{\partial p_i} \qquad \frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i} = -\frac{\partial U}{\partial q_i}. \tag{13}$$

Discretization of Hamiltonian Dynamics

Problem

Simulation requires discretization of dynamics.

Solution

Discretize time using small stepsize ϵ.

Multiple methods:

1. Euler's method
2. Modification of Euler's method
3. **Leapfrog method**

The Leapfrog Method

Update equations:

$$
\begin{aligned}
p_i(t+\tfrac{\epsilon}{2}) &= p_i(t) - \frac{\epsilon}{2}\frac{\partial U}{\partial q_i}(q(t)) \\
q_i(t+\epsilon) &= q_i(t) + \epsilon \frac{\partial K}{\partial p_i}(p(t+\tfrac{\epsilon}{2})) \\
p_i(t+\epsilon) &= p_i(t+\tfrac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial U}{\partial q_i}(q(t+\epsilon)).
\end{aligned}
\tag{14}
$$

Steps:

1. Half step for momentum variables.
2. Full step for position variables, using new momentum values.
3. Half step for momentum variables, using new position variables.

This method **preserves volume exactly**.

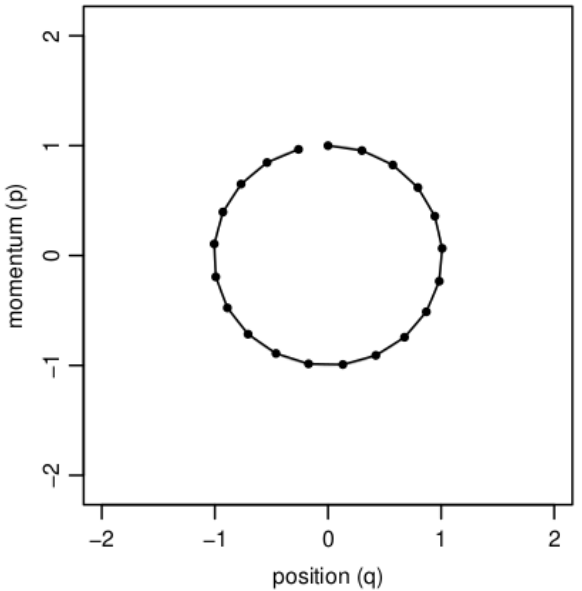


Figure adapted from [Neal et al., 2011].

Outline

MCMC from Hamiltonian Dynamics

Goal

Sample efficiently from target distribution.

Solution

Generate distant high-quality MH state proposals using Hamiltonian dynamics.

Steps

1. Translate target density to potential energy function.
2. Introduce auxiliary momentum variables.
3. Simulate Hamiltonian dynamics to obtain candidate states.
4. Generate Markov Chain by performing Metropolis-Hastings updates.

Sampling via Hamiltonian Dynamics

Goal

Sample vectors from target distribution: $q_t \sim P(q \mid \mathcal{D})$.

Define

$$U(q) = -\log P(q \mid \mathcal{D}). \tag{15}$$

Choose $K(p)$ arbitrarily, e.g.

$$K(p) = \frac{1}{2}|p|^2 \qquad \Rightarrow \qquad p \sim \mathcal{N}(0, I). \tag{16}$$

Hamiltonian H is thus defined as

$$H(q, p) = U(q) + \frac{1}{2}|p|^2. \tag{17}$$

Hamilton's equations become

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i} = p_i \qquad \frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i} = -\frac{\partial U}{\partial q_i}. \tag{18}$$

Hamiltonian H as energy function defines joint probability distribution over phase space (q, p):

$$\begin{aligned} P(p, q) \propto \exp(-H(q, p)) &= \exp(-U(q) - K(p)) \\ &= \exp(-U(q)) \exp(-K(p)) \\ &= P(q \mid \mathcal{D}) \exp(-K(p)). \end{aligned} \tag{19}$$

Marginal distribution for q is again proportional to target distribution $P(q \mid \mathcal{D})$.

To sample from target distribution:
Sample from joint distribution for q and p and just ignore values for p.

The Hamiltonian Monte Carlo Algorithm

A variant of the Metropolis-Hastings algorithm, with better-informed state proposals.

Step 1

Draw new momentum vector $p \sim K(p)$.

Leaves $P(q, p)$ and H invariant, as p is drawn from its correct marginal distribution.

Step 2

Simulate Hamiltonian dynamics for L steps via leapfrog method to obtain candidate state (q^*, p^*).

Acceptance probability:

$$\alpha = \min[1, \frac{P(q^*, p^*)}{P(q, p)}] = \min[1, \exp(-H(q^*, p^*) + H(q, p))]. \tag{20}$$

If candidate lowers energy H (increases $P(q, p)$), then $\alpha = 1$.
If candidate increases energy, then $\alpha = \exp(-\Delta H)$.

Exact simulation: $\alpha = 1$.
Leapfrog discretization, H can sometimes increase: $\alpha \leq 1$.

Outline

Stochastic Dynamics

Describes dynamics of a system that is subject to random fluctuations.

Problem

Calculating gradients, e.g. $\nabla U(q)$, on massive datasets is computationally infeasible.

Solution

Derive stochastic variations of Bayesian learning algorithms.

Replace gradient $\nabla_\theta \mathcal{L}(\theta)$ with stochastic approximation $\nabla_\theta \tilde{\mathcal{L}}(\theta)$ of minibatch $\tilde{\mathcal{D}}$.
Assume $x \in \mathcal{D}$ are independent and appeal to central limit theorem:

$$\nabla_\theta \tilde{\mathcal{L}}(\theta) \approx \nabla_\theta \mathcal{L}(\theta) + \mathcal{N}(0, \mathbb{V}[\theta]). \tag{21}$$

Stochastic Gradient Descent (SGD):

$$\Delta\theta = -\epsilon \nabla_\theta \tilde{\mathcal{L}}(\theta). \tag{22}$$

Outline

Constant SGD as Approximate Bayesian Inference

Run SGD with constant learning rate to simulate Markov chain with $P(w \mid \mathcal{D})$ as stationary distribution [Mandt et al., 2017].

Conventional SGD optimizes a function by following noisy gradients.
Decreasing step size allows to attain the (local) optimum.

Constant SGD moves towards an optimum, but never reaches it.
Constant step size makes the iterates bounces around the optimum.
Resulting stationary distribution can be made to approximate the posterior.

Benefits

- Stochastic approach
- Approximates full posterior
- Minimal implementation effort

Stochastic Gradient Langevin Dynamics (SGLD)

Central idea

Combine stochastic gradient descent with *Langevin dynamics* [Welling and Teh, 2011].

SG + LD

Simply add random Gaussian noise to stochastic gradients, balanced with step size, which goes to zero: $\eta_t \sim \mathcal{N}(0, \epsilon_t)$:

$$\Delta\theta_t = -\frac{\epsilon_t}{2}\nabla\tilde{\mathcal{L}}(\theta) + \eta_t. \tag{23}$$

Two sources of stochasticity:

1. Injected Gaussian noise η_t with variance ϵ_t
2. Noise in stochastic gradient with variance $(\frac{\epsilon}{2})^2\mathbb{V}[\theta_t]$

Initial phase: stochastic gradient noise dominates and algorithm imitates efficient SGD.

Later phase: $\epsilon_t \to 0$, injected noise dominates, so algorithm imitates Langevin dynamics.

Algorithm will transition smoothly between the two phases.

Outline

Stochastic Gradient Hamiltonian Monte Carlo

Problem

HMC limited by gradient computation over whole dataset.

Solution

Replace exact gradient with stochastic gradient [Chen et al., 2014].

Benefits

- Efficient state space exploration of HMC.
- Computational efficiency of SG methods.

Naïve Stochastic Gradient Hamiltonian Monte Carlo

Naïve Approach

Simply replace $\nabla U(q)$ in the momentum update with $\nabla \tilde{U}(q)$:

$$\frac{dp}{dt} = -\nabla U(q) \approx -\nabla \tilde{U}(q) \approx -\nabla U(q) + \mathcal{N}(0, \mathbb{V}[q]) \tag{24}$$

Hockey Puck Analogy

Puck on frictionless surface of varying height, but with a random wind blowing.

This no longer leads to Hamiltonian dynamics!

Requires frequent MH corrections or long simulation runs with low acceptance probabilities.

Stochastic Gradient HMC with Friction (SGHMC)

Problem

Naïve approach doesn't simulate Hamiltonian dynamics.

Solution

Add friction to the momentum update in order to maintain target distribution as stationary distribution:

$$\frac{dp}{dt} = -\nabla U(q) \approx -\nabla \tilde{U}(q) - \frac{1}{2}\mathbb{V}[q]p \approx -\nabla U(q) - \frac{1}{2}\mathbb{V}[q]p + \mathcal{N}(0, \mathbb{V}[q]). \tag{25}$$

Friction term: $\frac{1}{2}\mathbb{V}[q]p$.

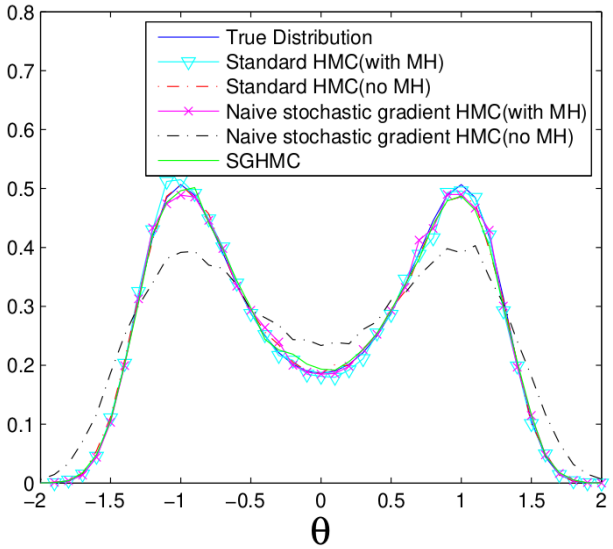
Hockey Puck Analogy

Street hockey instead of ice hockey, which introduces friction from asphalt.

Still random wind blowing, but friction of surface prevents puck from running far away.

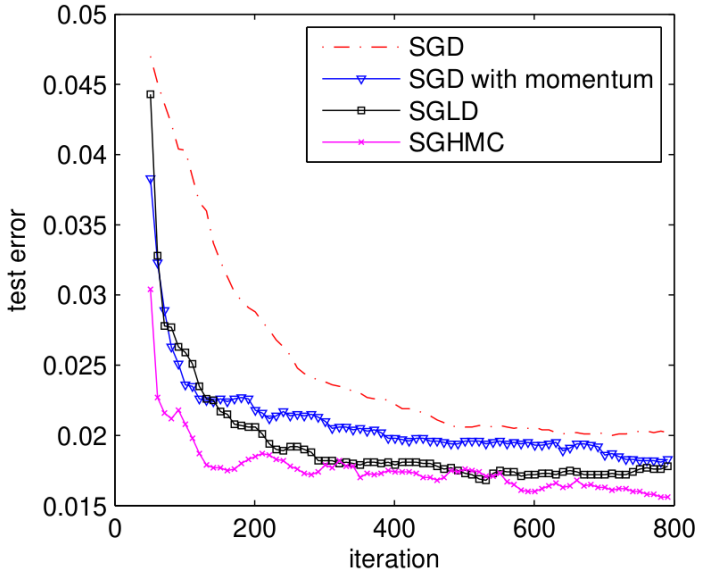
Note that when the additional noise is removed, SGHMC reduces to a stochastic gradient method with momentum.

Performance Comparison



Comparison of various sampling algorithms.

Figure taken from [Chen et al., 2014].



Convergence of test error on the MNIST dataset.

Figure taken from [Chen et al., 2014].

Outline

Summary

Goal

Perform Bayesian deep learning.

Problem 1

Direct sampling from posterior distribution impossible.

Solutions

1. Variational inference (*Bayes by Backprop*)
2. MCMC Methods
   - 2.1 Metropolis-Hastings algorithm
   - 2.2 HMC algorithm

Problem 2

Gradient computation infeasible for larger datasets.

Solutions

1. Bayesian variants of SGD
   - 1.1 Constant SGD
   - 1.2 Stochastic Gradient Langevin Dynamics
2. Stochastic variants of HMC
   - 2.1 Stochastic Gradient Monte Carlo

Thank you for your attention!

I’m happy to take questions.

References I

Andersen, H. C. (1980).
Molecular dynamics simulations at constant pressure and/or temperature.
The Journal of chemical physics, 72(4):2384–2393.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015).
Weight uncertainty in neural networks.
In *International Conference on Machine Learning*, pages 1613–1622. PMLR.

Chen, T., Fox, E., and Guestrin, C. (2014).
Stochastic gradient hamiltonian monte carlo.
In *International Conference on Machine Learning*, pages 1683–1691. PMLR.

Deng, L. (2012).
The mnist database of handwritten digit images for machine learning research.
IEEE Signal Processing Magazine, 29(6):141–142.

Jin, S.-S., Ju, H., and Jung, H.-J. (2019).
Adaptive markov chain monte carlo algorithms for bayesian inference: recent advances and comparative study.
Structure and Infrastructure Engineering.

Mandt, S., Hoffman, M. D., and Blei, D. M. (2017).
Stochastic gradient descent as approximate bayesian inference.
arXiv preprint arXiv:1704.04289.

Neal, R. (1992).
Bayesian learning via stochastic dynamics.
Advances in Neural Information Processing Systems, 5.

Neal, R. M. et al. (2011).
MCMC using hamiltonian dynamics.
Handbook of Markov Chain Monte Carlo, 2(11):2.

Welling, M. and Teh, Y. W. (2011).
Bayesian learning via stochastic gradient langevin dynamics.
In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688.

Appendix

Hamiltonian Dynamics: One-dimensional Example

Let $H(q,p) = U(q) + K(p)$

$$\begin{aligned} U(q) &= q^2/2 \\ K(p) &= p^2/2 \end{aligned} \tag{26}$$

This corresponds to $q \sim \mathcal{N}(0,1)$.

Resulting dynamics:

$$\frac{dq}{dt} = p \qquad \frac{dp}{dt} = -q \tag{27}$$

Solutions (r, a constant):

$$q(t) = r\cos(a+t) \qquad p(t) = -r\sin(a+t). \tag{28}$$

The mapping T_s is thus rotation by s radians clockwise around origin in (p,q) plane.

Properties of Hamiltonian Dynamics

Reversibility

- Mapping T_s is injective
- $T_s^{-1} = T_{-s}$
- Linked to invariance of target distribution under Hamiltonian MCMC updates

Conservation of the Hamiltonian

- Leaves the Hamiltonian itself invariant, i.e. $\frac{dH}{dt} = 0$
- Acceptance probability of Metropolis updates is one
- Discretizations: H only approximately invariant

Volume Preservation

- Preservation of volume in (q, p) space
- No need to account for change in volume in acceptance probability for Metropolis updates

Symplecticness

- $B_s^T J^{-1} B_s = J^{-1}$, with B_s Jacobian of mapping T_s
- Implies volume preservation because $\det(B_s^T)\det(J^{-1})\det(B_s) = \det(J^{-1})$ implies that $\det(B_s)^2 = 1$.

Euler's Method

Basic method for approximating stochastic differential equations.

Update equations:

$$p_i(t+\epsilon) = p_i(t) + \epsilon \frac{dp_i}{dt}(t) = p_i(t) - \epsilon \frac{\partial U}{\partial q_i}(q(t))$$
$$q_i(t+\epsilon) = q_i(t) + \epsilon \frac{dq_i}{dt}(t) = q_i(t) + \epsilon \frac{\partial K}{\partial p_i}(p(t)). \quad (29)$$

Can produce divergent trajectories!

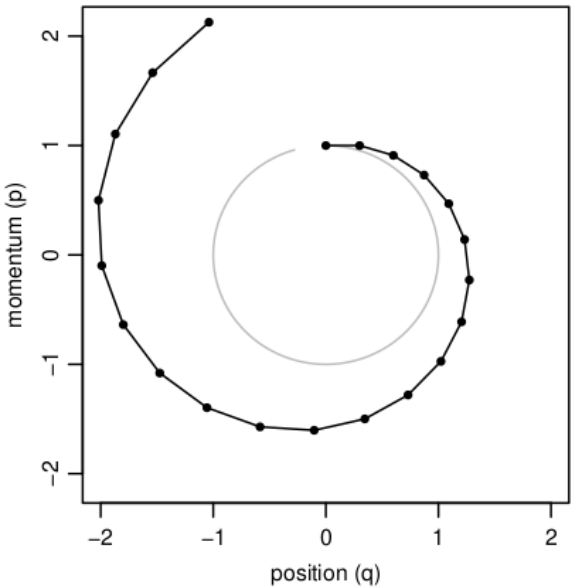


Figure adapted from [Neal et al., 2011].

Modification of Euler's Method

Modified update equations:

$$\begin{aligned} p_i(t+\epsilon) &= p_i(t) - \epsilon \frac{\partial U}{\partial q_i}(q(t)) \\ q_i(t+\epsilon) &= q_i(t) + \epsilon \frac{dq_i}{dt}(t) = q_i(t) + \epsilon \frac{\partial K}{\partial p_i}(p(t+\epsilon)). \end{aligned} \tag{30}$$

Modification: Use updated momentum values for updating position.

Describes "shear" transformations which preserve volume exactly, thus **no tendency to diverge to infinity**.

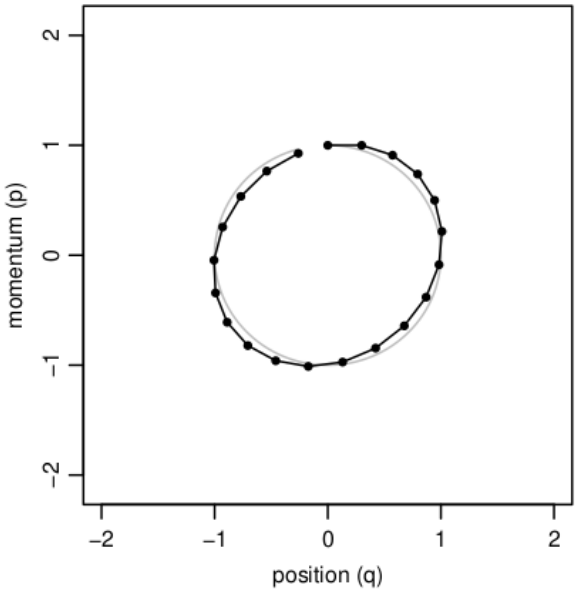


Figure adapted from [Neal et al., 2011].

Performance of Discretization Methods

Example: $H(q,p) = q^2/2 + p^2/2$ with initial state $(q,p) = (0,1)$.

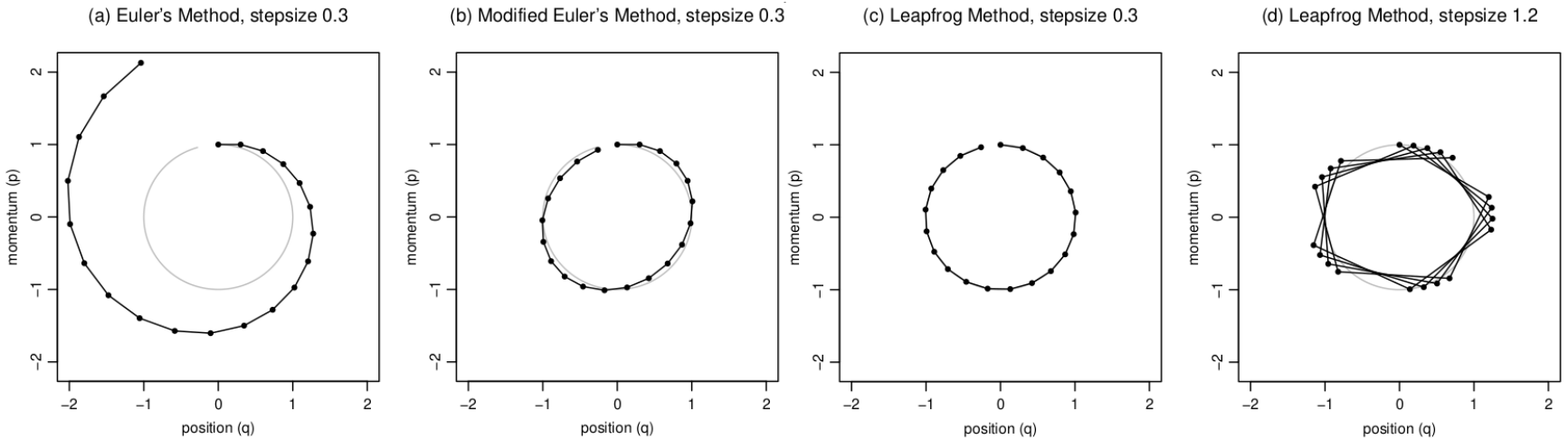


Figure adapted from [Neal et al., 2011].

Simple Stochastic Dynamics Variant of HMC

Proposed in [Andersen, 1980], [Neal, 1992].

Necessary to interleave leapfrog steps (which keep H approximately constant) with steps that can change H.
Convenient to only affect p, as it enters into H in a simple way.

Perform stochastic steps of the form

$$p' = \alpha p + (1 - \alpha^2)^{\frac{1}{2}} r, \tag{31}$$

where $0 \leq \alpha < 1$ and $r \sim \mathcal{N}(0, I)$.

Letting $\alpha \approx 1$ is best, as this reduces random walk aspect.
If $\alpha = 1$, process is equivalent to ordinary batch mode backpropagation with momentum.

Langevin Monte Carlo (LMC)

Special case of HMC algorithm, performing only a single leapfrog step per iteration, i.e. $L = 1$ [Neal et al., 2011].

Steps:

1. Sample momentum variables $p \sim \mathcal{N}(0, I)$.
2. Generate proposed values q^* and p^*:

$$q_i^* = q_i - \frac{\epsilon^2}{2}\frac{\partial U}{\partial q_i}(q) + \epsilon p_i \tag{32}$$

$$p_i^* = p_i - \frac{\epsilon}{2}\frac{\partial U}{\partial q_i}(q) - \frac{\epsilon}{2}\frac{\partial U}{\partial q_i}(q^*). \tag{33}$$

Proposed state q^* is accepted with probability

$$\min[1, \exp(-(U(q^*) - U(q)) - \frac{1}{2}\sum_i((p_i^*)^2 - p_i^2))]. \tag{34}$$