STATS 230: Computational Statistics
Advanced MCMC Methods

Babak Shahbaba

Department of Statistics, UCI

Background

- Simple Metropolis algorithm and Gibbs sampler explore the posterior distribution using a random walk.
- While this strategy might work well for low-dimensional distributions, it could become very inefficient (e.g., high autocorrelation, missing isolated modes) for high-dimensional distributions.
- In this lecture, we discuss two useful strategies to improve the efficiency of Markov chain Monte Carlo methods.
- We are going to briefly discuss
  - Overrelaxation for Gibbs sampling
  - Hamiltonian Monte Carlo
- For more information on overrelaxation refer to Neal (1998). For Hamiltonian Monte Carlo, refer to Neal (2010).

Overrelaxation

Overrelaxation

- This method is used to improve Gibbs samplers by reducing the random walk behavior.
- Adler (1981) first introduced this method for situations where all the conditional distributions are Gaussian.
- Suppose we want to sample from the distribution of $x_1, \ldots, x_N$, where

$$x_j | x_{-j} \quad \sim \quad N(\mu_j, \sigma_j^2)$$

  Here, x_{-j} represents all random variables excluding x_j, and (μ_j, σ_j^2) are the parameters of the conditional distribution given all other parameters.
- Note that in Bayesian statistics the random variable of interest are model parameters and our goal is to sample from their joint posterior distribution.

Overrelaxation

- Recall that Gibbs sampling sequentially updates $\{x_j\}$ by sampling from their conditional distribution $N(\mu_j, \sigma_j^2)$ given the current values of all other parameters

$$\begin{aligned} x_j^{(i+1)} &= \mu_j + \sigma_j z \\ z &\sim N(0,1) \end{aligned}$$

- In Adler's method, we instead update x_j as follows:

$$\begin{aligned} x_j^{(i+1)} &= \mu_j + \alpha(x_j^{(i)} - \mu_j) + \sigma_j(1-\alpha^2)^{1/2} z \\ z &\sim N(0,1) \end{aligned}$$

  where α is a constant usually slightly bigger than -1, for example -0.9. (In general, to make this method work we must have $-1 < \alpha < 1$.)

- Note that when $\alpha = 0$ this method reduces to the simple Gibbs sampling.

Overrelaxation

- Let's look at a simple example to see how overrelaxation works.
- Assume that we want to sample from the following distribution:

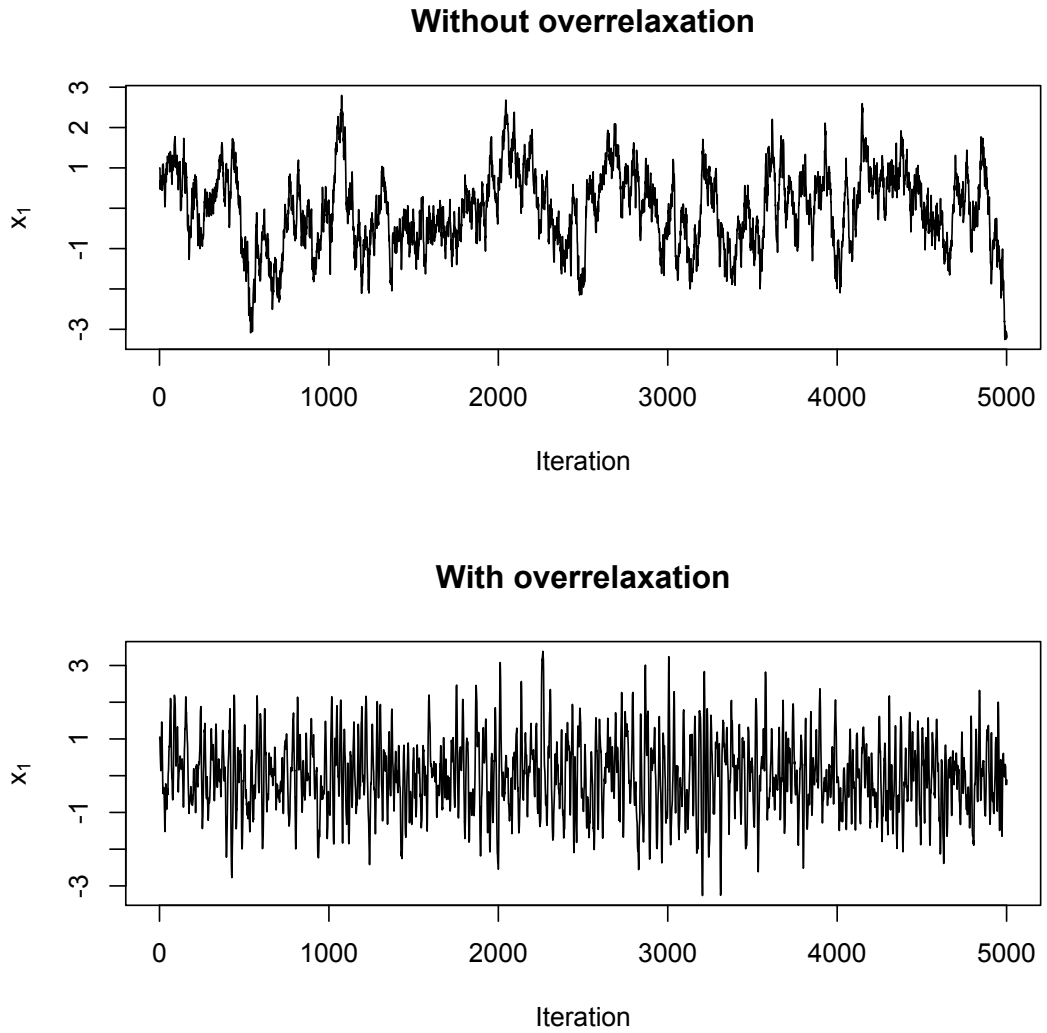
$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.99 \\ 0.99 & 1 \end{pmatrix} \right)$$

- Based on what we know about multivariate normal distributions, we have

$$\begin{aligned} x_1|x_2 &\sim N(0.99x_2, (1 - 0.99^2)) \\ x_2|x_1 &\sim N(0.99x_1, (1 - 0.99^2)) \end{aligned}$$

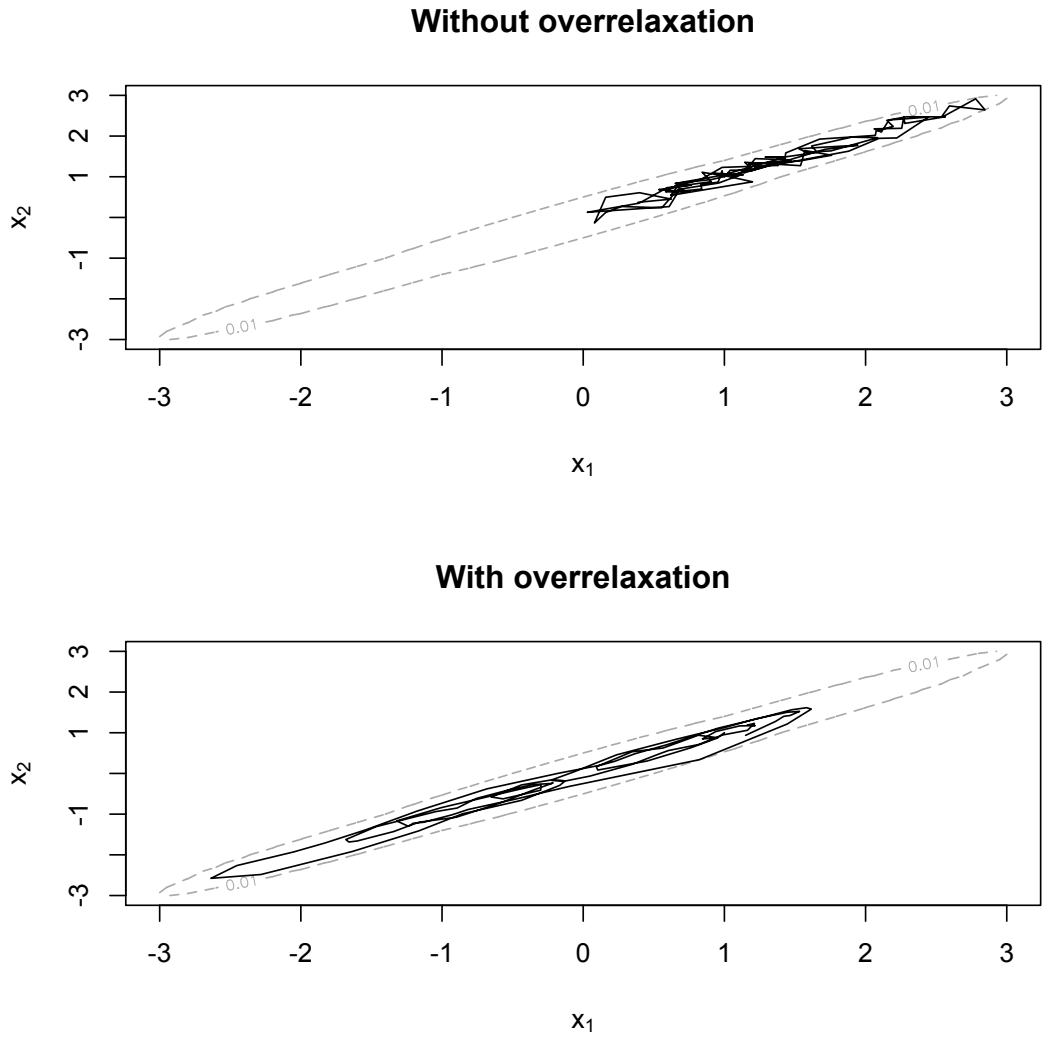
Overrelaxation

- The following graph shows the trace plots of samples for x_1 with and without overrlaxation



Overrelaxation

- The following graph shows the first 100 iterations, starting from (1, 1), with and without overrlaxation



Overrelaxation

- Why overrelaxation works?
- It clearly leaves the distribution invariant: if

$$x_j^{(i)}|. \sim N(\mu_j, \sigma_j^2)$$

then

$$x_j^{(i+1)}|. \sim N(\mu_j, \sigma_j^2)$$

- Overrelaxation tends to propose states on the opposite side of the conditional mean compared to the current state.
- This causes the chain to move in a consistent direction so it explores the posterior distribution better.
- See Neal (1995) for more general forms of overrelaxation (`http://arxiv.org/pdf/bayes-an/9506004.pdf`)

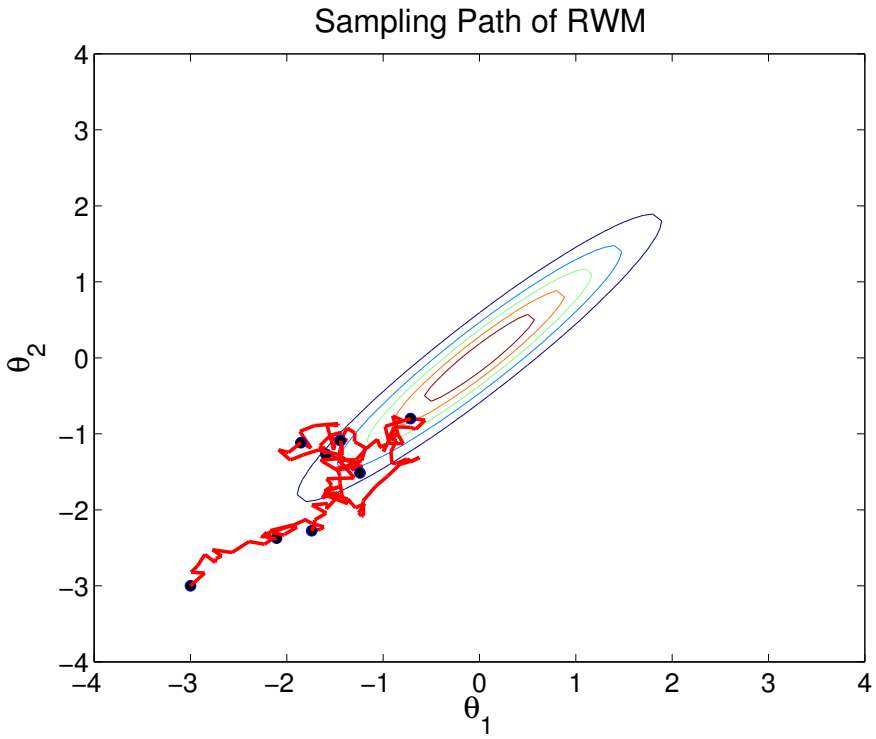
Hamiltonian Monte Carlo

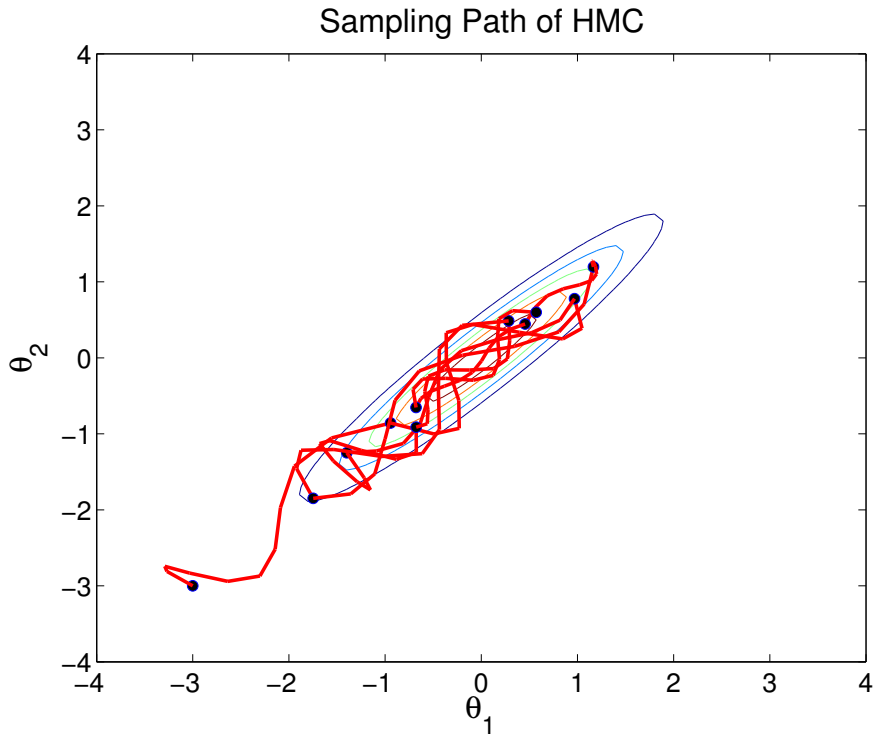
Hamiltonian dynamics

- Hamiltonian dynamics is used to improve the efficiency of the Metropolis algorithm.
- It does this by reducing the random walk behavior through proposing distant states (from the current state) with high probability of acceptance (in theory, with probability 1).
- A little bit of history...
- In 1953, Metropolis et. al. introduced MCMC to simulate the distribution of states for a system of idealized molecule.
- In 1959, Alder and Wainwright introduced a system called *Hamiltonian dynamics* to simulate motion of molecules deterministically based on Newton's law of motion.
- In 1987, Duane et. al. combine the MCMC and the Hamiltonian dynamics. They called their method *Hybrid Monte Carlo* (HMC).
- The abbreviation HMC has also been used for *Hamiltonian Monte Carlo*.

Hamiltonian Monte Carlo

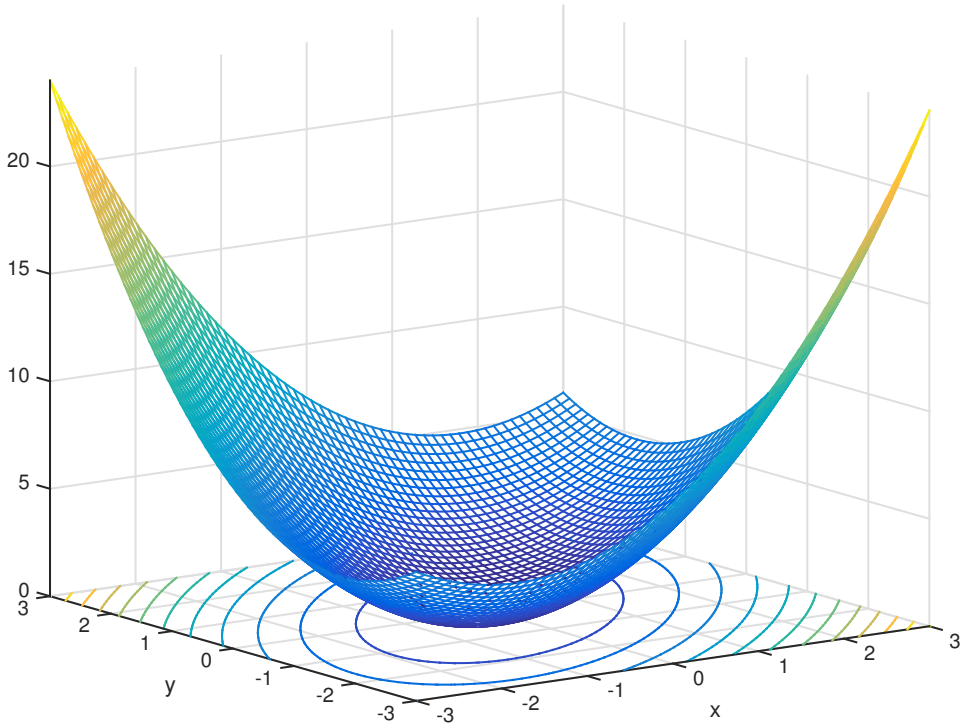
- HMC proposes states that are distant from the current state, but nevertheless have a high probability of acceptance.





Hamiltonian Monte Carlo

- The sampler is viewed as a dynamic system moving on a surface defined by the *energy* function U: negative log density of the target distribution



- Distant proposals are found by numerically simulating Hamiltonian dynamics for some specified amount of fictitious time

Posterior sampling

- For Bayesian inference, posterior distribution is the target distribution

Potential energy

$$U(\theta) = -\sum_{i=1}^{N} \log P(y_i|\theta) - \log P(\theta)$$

- We augment the parameter space with fictitious momentum variables

Kinetic energy

$$K(p) = \frac{1}{2} p^\top M^{-1} p$$

- Define the Hamiltonian function $H(\theta, p) = U(\theta) + K(p)$
- The joint density of (θ, p) is

$$P(\theta, p) \propto \exp\{-H(\theta, p)\} = \exp\{-U(\theta)\} \cdot \exp\{-K(p)\}$$

- The marginal distribution of θ is the posterior distribution

Hamiltonian Dynamics

- We can generate a proposal by starting from the current state at time 0 and moving to the state at time t:

$$(\theta, p) = (\theta^{(0)}, p^{(0)}) \xrightarrow{\text{HD}} (\theta^{(t)}, p^{(t)}) = (\theta^*, p^*)$$

- *Hamilton's equations* determine how θ and p change over [fictitious] time

Hamilton's equations

$$\begin{aligned} \frac{d\theta_j}{dt} &= \frac{\partial H}{\partial p_i} = [M^{-1}p]_i \\ \frac{dp_j}{dt} &= -\frac{\partial H}{\partial \theta_j} = -\frac{\partial U}{\partial \theta_j} \end{aligned}$$

- Important properties (see `http://arxiv.org/pdf/1206.1901.pdf`):
  - **Reversibility**: the target distribution remain invariant
  - **Volume preservation**: the Jacobin determinant is 1
  - **Conservation of Hamiltonian**: the acceptance rate is one; θ^* is the next sample if HD is analytically solvable

Numerical Integration

- Numerical integration is employed when analytic solution is not available

Leapfrog

$$
\begin{aligned}
p_j(t+\varepsilon/2) &= p_j(t) - (\varepsilon/2)\frac{\partial U}{\partial \theta_j}(\theta(t)) \\
\theta_j(t+\varepsilon) &= \theta_j(t) + \varepsilon\frac{\partial K}{\partial p_j}(p(t+\varepsilon/2)) \\
p_j(t+\varepsilon) &= p_j(t+\varepsilon/2) - (\varepsilon/2)\frac{\partial U}{\partial \theta_j}(\theta(t+\varepsilon))
\end{aligned}
$$

- Important properties:
  - **Stability**: numerically stable if ε is appropriately chosen
  - **Reversibility and Volume preservation**: still hold
  - **Conservation of Hamiltonian**: broken, but can be corrected by MH correction step with acceptance rate

$$\alpha = \min[1, \exp(-H(\theta^*, p^*) + H(\theta, p))]$$

Algorithm 1 HMC algorithm

Initialize $\theta^{(0)} = \text{current } \theta$

Sample new momentum $p^{(0)} \sim \mathcal{N}(0, M = I)$

Calculate current $H(\theta^{(0)}, p^{(0)}) = U(\theta^{(0)}) + \frac{1}{2}(p^{(0)})^\top p^{(0)}$

for $\ell = 1$ to L (leapfrog steps) **do**

$p^{(\ell+\frac{1}{2})} = p^{(\ell)} - \varepsilon/2 \nabla_\theta U(\theta^{(\ell)})$

$\theta^{(\ell+1)} = \theta^{(\ell)} + \varepsilon p^{(\ell+\frac{1}{2})}$

$p^{(\ell+1)} = p^{(\ell+\frac{1}{2})} - \varepsilon/2 \nabla_\theta U(\theta^{(\ell+1)})$

end for

Accept or reject according to the Metropolis acceptance probability

Example: logistic regression with $N(0, \sigma^2 I)$ prior

$$\nabla_{\beta_j} U(\beta) = -\sum_{i=1}^{N} [y_i - \frac{\exp(x_i\beta)}{1+\exp(x_i\beta)}] x_{ij} + \beta_j/\sigma^2$$

A special case: Langevin Monte Carlo

- A special case: $L = 1$ and $M = I$
- This is called Langevin Monte Carlo,

Langevin dynamics

$$\begin{aligned} \theta^* &= \theta - \frac{\varepsilon^2}{2}\nabla_\theta U(\theta) + \varepsilon p \\ p^* &= p - \frac{\varepsilon}{2}\nabla_\theta U(\theta) - \frac{\varepsilon}{2}\nabla_\theta U(\theta^*) \end{aligned}$$

- Alternatively, we could ignore the momentum variable p and use the following asymmetrical proposal with MH acceptance probability

$$\theta^* \sim N(\theta - \frac{\varepsilon^2}{2}\nabla_\theta U(\theta), \varepsilon^2 I)$$

- Dropping the accept/reject step leads to an approximate Langevin method (see Neal, 1993)

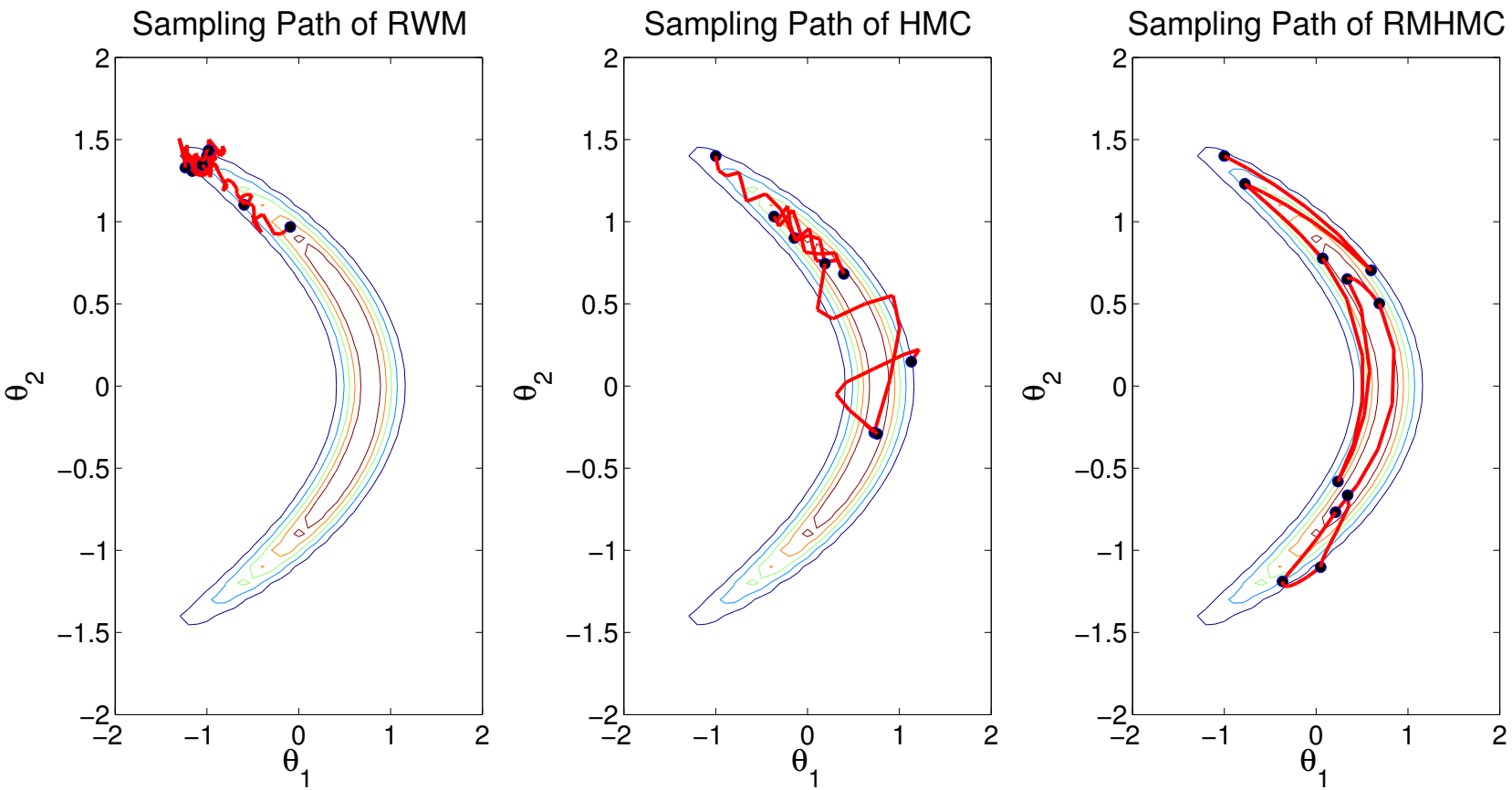
A more general case: Riemannian Manifold HMC

- Girolami and Calderhead (2011) have introduced a new method, called Riemannian Manifold HMC (RMHMC)
- They argue that it is more natural to put the Hamiltonian dynamic on Riemannian manifold of distributions rather than Euclidean space
- They follow Amari (2000) and use the Fisher information matrix, $G(\theta) = E\left[\nabla^2_\theta U(\theta)\right]$, as a metric on the manifold
- That is, they use position specific mass matrix, $M = G(\theta)$
- Example: logistic regression

$$G_{jk}(\beta) \quad = \quad \sum_{i=1}^{N} x_{ij} x_{ik} \frac{\exp(x_{i\beta})}{[1+\exp(x_i\beta)]^2}, \quad j \neq k$$

- We can explore the parameter space more efficiently by exploiting its geometric properties
- The resulting dynamics is non-separable so instead of the standard leapfrog method we need to use the *generalized* leapfrog method

HMC vs. RMHMC


Sampling Path of RWM
Sampling Path of HMC
Sampling Path of RMHMC
θ1
θ2

A main challenge: high computational cost

- For high-dimensional problems (big n and/or big d) and complex models, these methods tend to be computationally expensive
- We have proposed several variations of HMC:
  - Split HMC (S. et al., 2011)
  - Lagrangian Monte Carlo (Lan, et al., 2012)
  - Spherical HMC (Lan et al., 2013)
  - Wormhole HMC (Lan et al., 2013)
  - HMC with precomputing strategy (Zhang et al., 2015)
  - HMC with surrogate functions (Zhang et al., 2015)

Scalable HMC

Subsampling

- In recent years, computational methods based on mini-batches of data have been quite successful
  - The underlying assumption: there is redundancy in big data
  - The overall information can be retrieved from a small subset
  - We can approximate functions at low computational cost
- Welling and Teh (2011) used this approach (stochastic gradient) for Langevin dynamics using mini-batches of size n from N observations

$$\theta^* = \theta + \frac{\varepsilon^2}{2}\left(\nabla_\theta P(\theta) + \frac{N}{n}\sum_{i=1}^{n}\nabla_\theta \log P(x_i|\theta)\right) + \varepsilon p$$

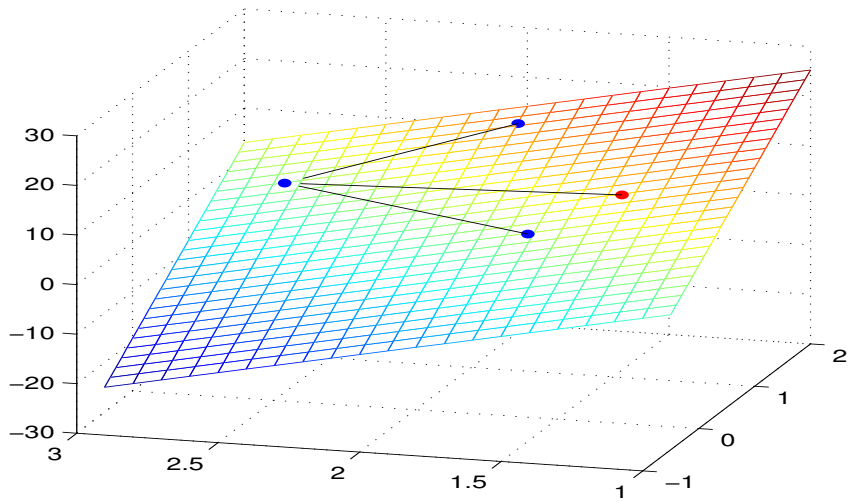
- They also dropped the accept/reject step

Precomputing strategies

- Finding optimum subsets by exploiting regularity in data space is difficult
- Using random subsets could lead to non-ignorable loss of information
- Therefore, (S. et al., 2011) previously proposed to identify a subset of influential points to split the Hamiltonian function (the likelihood part) into two parts
- Recently, Zhang et. al. (2015) have proposed to switch the focus of approximation from data space to parameter space (`http://arxiv.org/abs/1504.01418`)

A Toy Example – Gaussian Model

- Consider a Gaussian model with a Gaussian prior on the mean (no sampling required)
- The function value at any given point (red point) can be obtained by interpolation given a sample of three points (blue points)



$\nabla_\mu U$

Grid Approximation of HMC (GHMC)

$$\frac{dp_j}{dt} = -\frac{\partial U}{\partial \theta_j}$$
$$\frac{d\theta_j}{dt} = [M^{-1}p]_j$$

Denote

Force

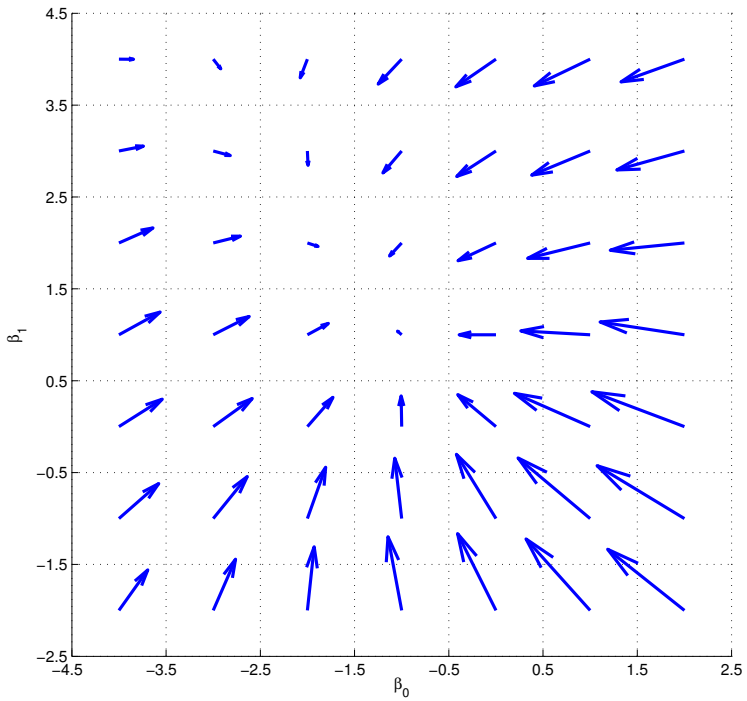
$$F = -\nabla U$$

- piecewise constant approximation

$$\tilde{F}(\theta) = F_{i,j} \stackrel{\triangle}{=} F(c_{i,j}), \quad \text{if } \theta \in C_{i,j}$$

- piecewise linear approximation

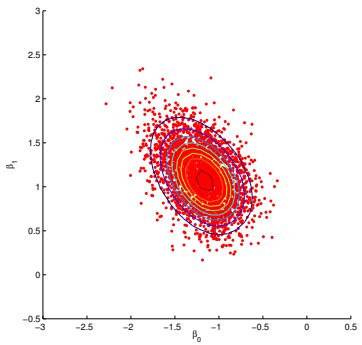
$$\tilde{F}(\theta) = F_{i,j} + \nabla F_{i,j} \cdot (q - c_{i,j}), \text{ if } \theta \in C_{i,j}$$



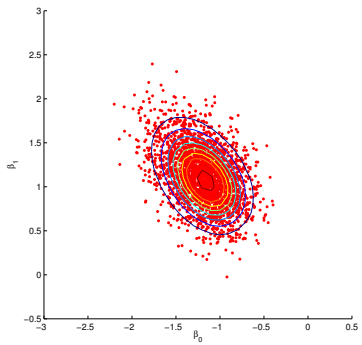
Force map of a logistic regression model

True $\beta = (-1, 1)$

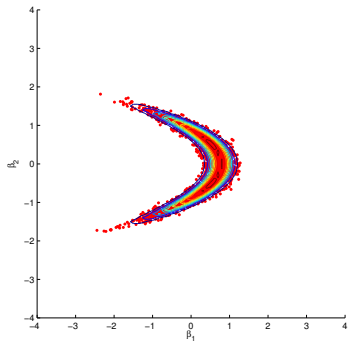
GHMC–Examples



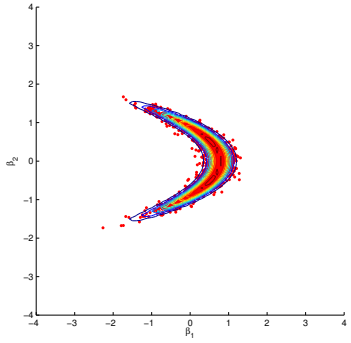
HMC



GHMC



HMC



GHMC

Logistic regression

Banana-shaped distribution

Experiment	Method	AR	s/Iteration	min(ESS)/s	Spped-up
LR	HMC	0.9225	7.0157E-4	1425.4	1
	GHMC	0.7981	3.318E-4	3013.9	**2.1**
BD	HMC	0.9353	3.8703E-4	962.1	1
	GHMC	0.6587	1.4498E-4	1651.6	**1.7**

HMC with Surrogate Functions

NNS-HMC

- In recent years, several methods have been proposed based on constructing *surrogate* Hamiltonians using Gaussian process models (Rasmussen, 2003; Meeds and Welling, 2015; Lan et. al., 2015)

- We have instead used a simple generalized additive model, which can be regarded as a shallow neural network,

$$\tilde{U}(\theta) = \sum_{i=1}^{s} v_i g(\boldsymbol{w_i} \cdot \theta + d_i) + d_0$$

with the softplus function: $g(z) = \log(1 + \exp(z))$

Extreme Learning Machine (ELM)

For training, we randomly assign input weights and biases, and then obtain the least-square estimates of the output weights v

ELM (Huang, 2006)

Given a training set $\mathcal{T} = \{(I_j, t_j) | I_j \in \mathbb{R}^n,\ t_j \in \mathbb{R}^m,\ j = 1, \ldots, N\}$, activation function $\sigma(x)$ and hidden node number s

1. Randomly assign input weight w_i and bias d_i, $i = 1, \ldots, s$
2. Calculate the hidden layer output matrix H

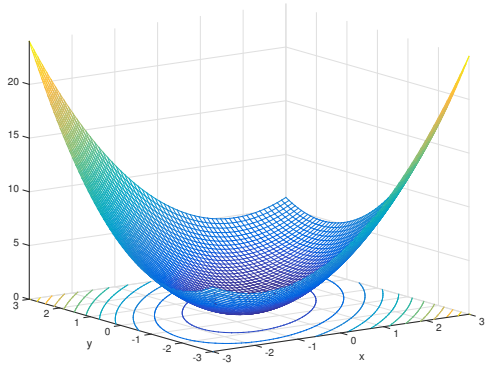
$$H_{ji} = \sigma(w_i I_j + d_i), \quad i = 1, \ldots, s,\ j = 1, \ldots, N$$

3. Calculate the output weight v

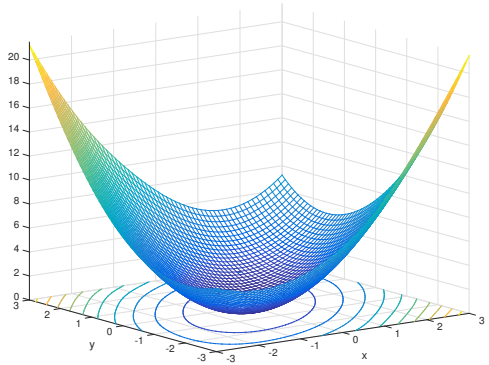
$$v = H^{\dagger} T, \quad T = [t_1, t_2, \ldots, t_N]^T$$

where $H^{\dagger}$ is the *Moore-Penrose generalized inverse* of matrix H

NNS-HMC



Target function



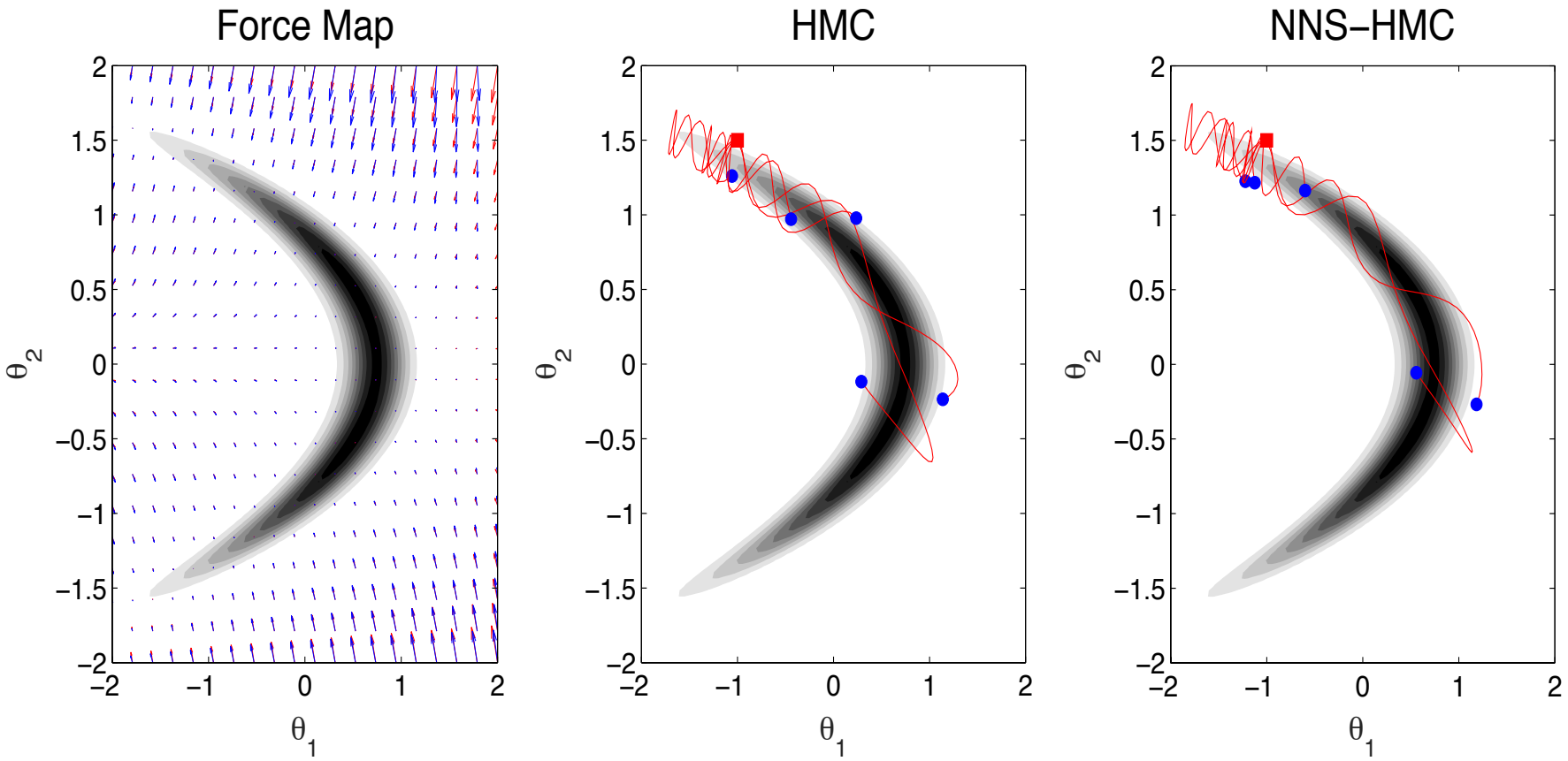
Neural network approximation

- The training process (using pre-convergence samples) and the approximation of functions in the sampling phase can be easily incorporated in HMC
- The approximate geometric information (e.g, gradient and Hessian) is obtained by differentiating the neural network directly,

$$\frac{\partial \tilde{U}}{\partial \theta} = \sum_{i=1}^{s} v_i g'(\boldsymbol{w}_i \cdot \boldsymbol{\theta} + d_i)\boldsymbol{w}_i$$

- Easy generalization to Riemannian Manifold HMC (NNS-RMHMC)

Surrogate Induced Hamiltonian Flow



Experiments

Experiment	Method	AP	s/Iter	min(ESS)/s	Spped-up
LR (Simulation)	HMC	0.6656	3.573E-01	1.45	1
	RMHMC	0.8032	3.794	0.06	0.04
	NNS-HMC	0.6726	1.364E-02	37.83	**26.09**
	NNS-RMHMC	0.8162	1.027E-01	2.17	1.50
LR (Bank Marketing)	HMC	0.8038	7.400E-02	6.51	1
	RMHMC	0.9210	6.305E-01	0.56	0.08
	NNS-HMC	0.7944	7.508E-03	58.22	**8.94**
	NNS-RMHMC	0.9064	2.741E-02	14.41	2.21
LR (Adult Data)	HMC	0.8300	7.898E-02	0.21	1
	RMHMC	0.8526	5.842E-01	1.06	4.81
	NNS-HMC	0.8096	9.914E-03	2.66	12.09
	NNS-RMHMC	0.8400	3.300E-02	18.68	**84.90**
Elliptic PDE	HMC	0.7077	1.568	0.061	1
	RMHMC	0.8014	4.388	0.228	3.74
	NNS-HMC	0.7138	7.419E-02	1.410	23.11
	NNS-RMHMC	0.6584	9.720E-02	4.375	**71.72**

Variational HMC

Free-form variational Bayes

- Alternatively, we can make inference based on an approximate distribution, similar to variational Bayes, but with a better and more flexible approximation (see for example, de Freitas et al., 2001; Salimans et al., 2015)
- For variational Bayes, we typically use a parametrized distribution $q_\eta(\theta)$ to approximate the target posterior $p(\theta|Y)$ by minimizing the KL divergence
- Here, we use the approximate distribution based our neural network model

$$Q_v(\theta) \propto \exp(-\tilde{U}(\theta)) = \exp[-\sum_{i=1}^{s} v_i g(\boldsymbol{w_i} \cdot \theta + d_i) + \phi(v)]$$

- This is simply a flexible exponential family model

Free-form variational Bayes

- To find Q_v, we follow Hyvarinen (2005) and minimize the score-matching distance

$$\tilde{D}_{SM}(P(\theta|Y)||Q_v(\theta)) = \frac{1}{2}\int Q_v(\theta)\|\nabla_\theta \tilde{U}(\theta) - \nabla_\theta U(\theta)\|^2 \, d\theta$$

- For this, we use HMC to generate samples from Q_v

$$\frac{d\theta}{dt} = \frac{\partial \tilde{H}}{\partial p} = M^{-1}p$$

$$\frac{dp}{dt} = -\frac{\partial \tilde{H}}{\partial \theta} = -\nabla_\theta \tilde{U}(\theta)$$

where the modified Hamiltonian is

$$\tilde{H}(\theta, p) = \tilde{U}(\theta) + K(p)$$

- Then minimize the regularized empirical distance

$$\hat{v} = \arg\min_v \frac{1}{2}\sum_{n=1}^{t} \|\nabla_\theta \tilde{U}(\theta_n) - \nabla_\theta U(\theta_n)\|^2 + \frac{\lambda}{2}\|v\|^2$$

Online updating of the weight vector

- Given the current weight vector $v^{(t)}$ and a new training data point $(\theta_{t+1}, \nabla_\theta U(\theta_{t+1}))$, the updating formula for the estimator is given by

$$v^{(t+1)} = v^{(t)} + W^{(t+1)}(\nabla_\theta U(\theta_{t+1}) - A_{t+1} v^{(t)})$$

where

$$W^{(t+1)} = C^{(t)} A'_{t+1} \left[I_d + A_{t+1} C^{(t)} A'_{t+1} \right]^{-1}$$
$$A_{t+1} = (A_1(\theta_{t+1}), \ldots, A_s(\theta_{t+1}))$$

with $A_i(\theta_{t+1}) := \sigma'(w_i \cdot \theta_{t+1} + d_i) w_i$, and $C^{(t)}$ can be updated by *Sherman-Morrison-Woodbury* formula:

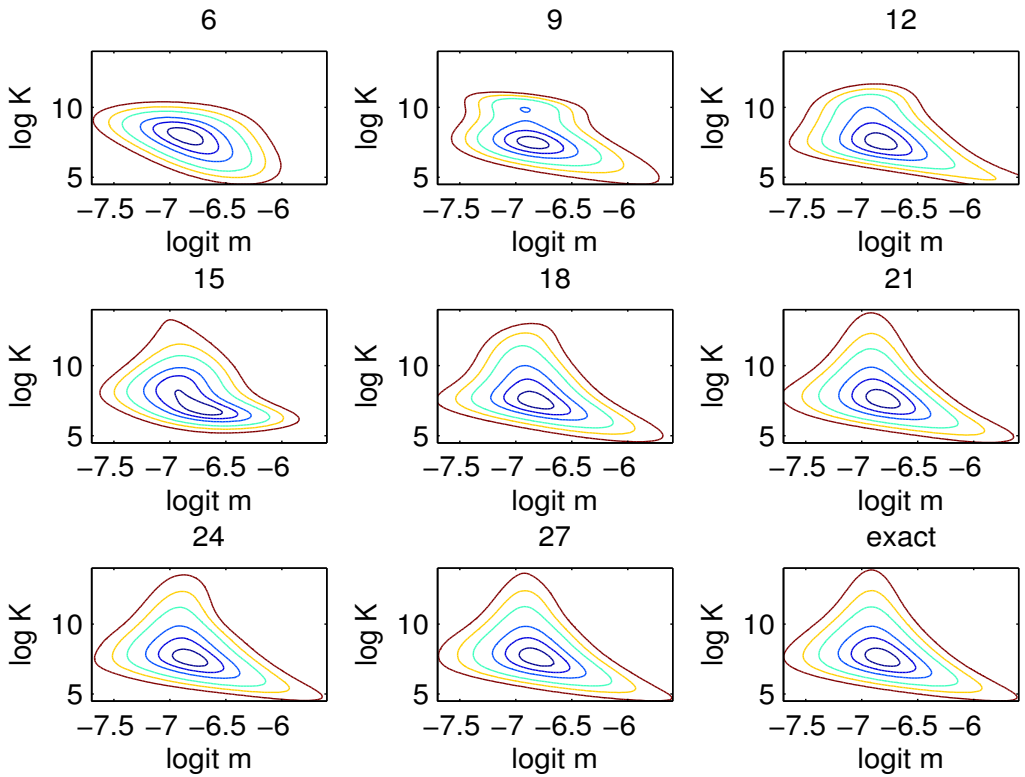
$$C^{(t+1)} = C^{(t)} - W^{(t+1)} A_{t+1} C^{(t)}$$

Example: a beta-binomial model

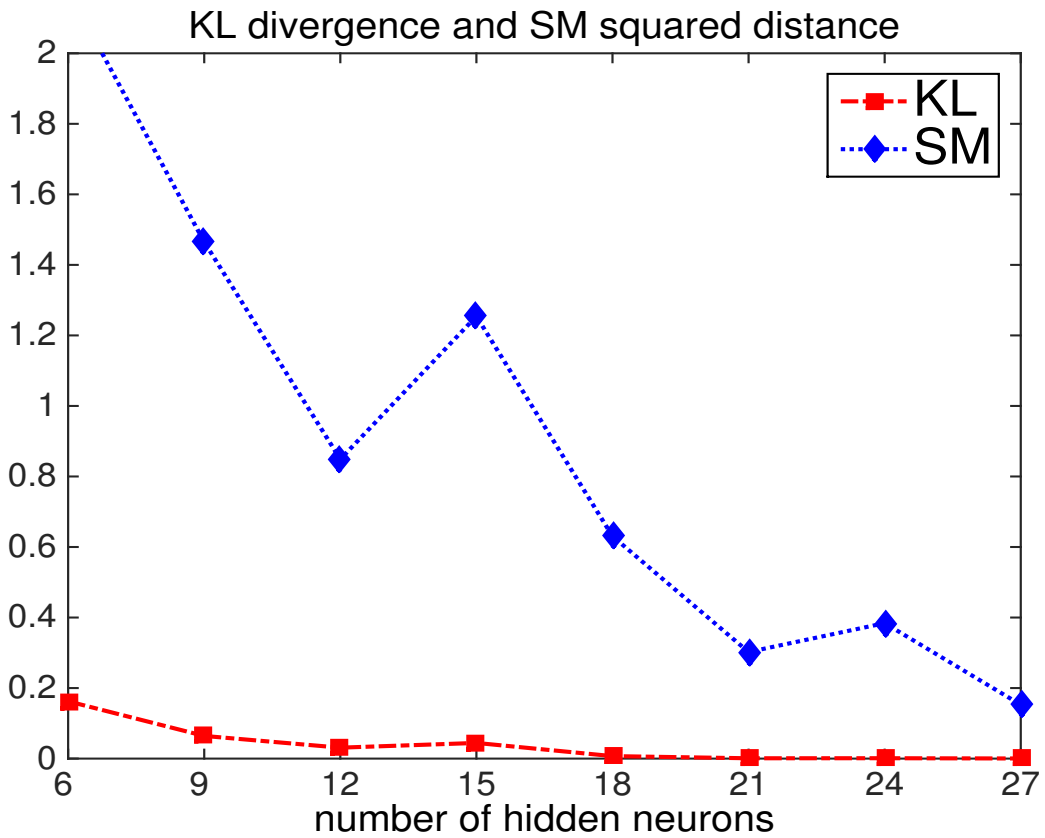
- For illustration, we consider the following beta-binomial model:

$$P(y_j|m,K) = \binom{n_j}{y_j} \frac{B(Km + y_j, K(1-m) + n_j - y_j)}{B(Km, K(1-m))}$$

- The following plot shows approximate posterior distributions for different numbers of hidden neurons (basis functions)



Example: beta-binomial model



Example: Independent Component Analysis

- In this example, we apply ICA to MEG data
- The following plot compares our method to HMC and SGLD (Welling and Teh, 2011) using the Amari distance (Amari et al., 1996), $d_A(\overline{W}, W_0)$, for the unmixing matrix W