MCMC for Hierarchical Bayesian Models Using Non-reversible Langevin Methods

Radford M. Neal

University of Toronto, Vector Institute Affiliate

radford@stat.utoronto.ca
https://www.cs.utoronto.ca/∼radford
https://radfordneal.wordpress.com

Virtual seminar, University of Oxford, 14 May 2020

I. Motivation — Hierarchical models and discrete variables

II. MCMC background

III. Non-reversible Langevin methods

IV. Application to hierarchical models and discrete variables

Hierarchical Bayesian Models

Bayesian inference for realistic problems usually involves a prior that is most conveniently specified hierarchically.

A simple multivariate regression example with k covariates:

$$\begin{aligned}
y_i \mid x_i, \alpha, \beta &\sim N(\alpha + \beta^T x_i, \sigma^2), \quad i = 1, \ldots, n \\
\beta_j \mid \tau &\sim N(0, \tau^2), \quad j = 1, \ldots, k \\
\alpha &\sim N(0, 100^2) \\
1/\sigma^2 &\sim \text{Gamma}(0.5) \,/\, 0.05^2 \\
1/\tau^2 &\sim \text{Gamma}(0.5) \,/\, 0.1^2
\end{aligned}$$

Here, we don't know enough to specify a direct normal prior for the regression coefficients, such as $\beta_j \sim N(0, 7^2)$. Maybe the covariates are actually nearly useless for predicting y, and so all the β_j should be near zero! We can express this uncertainty by making the prior variance of the β_j be a higher-level hyperparameter.

More Complex Hierarchical Bayesian Models

More complex models may have many groups of related parameters, with each group having a prior whose variance is an unknown higher-level hyperparameter.

For example, in a neural network model, the weights on connections between a pair of layers might be a group. Or the weights from each input might be separate groups — allowing an input to be mostly ignored if its variance hyperparameter turns out to be close to zero.

Latent variables / random effects look the same in a Bayesian context — they also have a distribution controlled by an unknown variance parameter. For example, a one-way random effects model:

$$
\begin{aligned}
y_{ij} \mid \mu_i &\sim N(\mu_i, \sigma^2), \quad i = 1, \ldots, n, \quad j = 1, \ldots, m \\
\mu_i \mid \tau &\sim N(0, \tau^2) \quad i = 1, \ldots, n \\
1/\sigma^2 &\sim \text{Gamma}(0.5) \,/\, 0.1^2 \\
1/\tau^2 &\sim \text{Gamma}(0.5) \,/\, 0.1^2
\end{aligned}
$$

Problems with MCMC for Hierarchical Models

Inference for hierarchical models is often done with Markov chain Monte Carlo. Gibbs samplings works OK for simple models. For more complex models, a more general random-walk Metropolis (RWM) or Hamiltonian Monte Carlo (HMC) method could be tried.

There's a problem, however.

If the posterior distribution of a variance hyperparameter is broad, the Markov chain needs to sample both regions where there are tight constraints on lower-level parameters (eg, the β_j, when τ is small), and regions where they are much less constrained (τ large), at least in directions allowed by the data.

A suitable proposal width for RWM, or suitable leapfrog stepsize for HMC, will be different in these different regions. Using a proposal or stepsize suitable for the less-constrained region can result in the more-constrained region never being visited in a reasonable-length run. (See the "funnel" example in my slice sampling paper.)

Possible Solutions

- Randomly choose a proposal width or stepsize each MCMC iteration from a distribution broad enough to contain values suitable for both highly-constrained and less-constrained regions.

  — Much time may now be wasted on unsuitable values.

  — But see my "short-cut" method for a way to alleviate this.

- Reparameterize the model as $\beta_j = \tau\zeta_j$, with $\zeta_j \sim N(0,1)$. Now the prior constraint on ζ_j doesn't vary.

  — However, the constraint on ζ_j from the data now varies.

- Alternate iterations that update only the lower-level parameters with iterations that update only the higher-level hyperparameters.

  — It's valid to tune lower-level updates using higher-level values.

  — Higher level updates may be simple (eg, Gibbs sampling).

  — If lower-level updates are lengthy (eg, HMC), higher-level updates will be done infrequently, slowing exploration.

Handling Discrete Variables

Many models involve discrete variables, as well as continuous parameters and latent variables. Examples:

— Mixture models expressed using indicators of which component each data item came from.
— Imputed values when some categorical data are missing.
— Indicators for covariates included in a sparse regression model.

Discrete variables are not a problem for Gibbs sampling or RWM. But they can't be handled directly by a gradient-based method such as HMC.

One approach is to alternate HMC updates for the continuous variables with some other update for the discrete variables.

But we again have the problem that if HMC uses long trajectories (as is usually desirable), the discrete variables will be updated only infrequently (relative to movement of the continuous variables).

Can we find some HMC-like method that avoids this problem?

I. Motivation — Hierarchical models and discrete variables

II. MCMC background

III. Non-reversible Langevin methods

IV. Application to hierarchical models and discrete variables

Markov Chain Monte Carlo (MCMC)

The MCMC approach to Bayesian inference:

- Specify a model (defining a likelihood function) and a prior. Both are assumed computable up to some possibly unknown normalizing constant, so that the posterior density for parameters / latent variables is also computable apart from its normalizing constant.
- Define a Markov chain that will converge to this posterior distribution from any starting point.
- Run the chain from one or many starting points, for long enough for it to approximately converge, then use subsequent points to estimate expectations with respect to the posterior (eg, posterior means of parameters, predictions for new data points).

Invariance and Reversibility

For a Markov chain to converge to a desired distribution, which has probability density $\pi(x)$, it is necessary for it to leave π *invariant*:

$$\text{For all } x, \quad \int \pi(x)\, T(x'|x)\, dx \;=\; \pi(x')$$

where $T(x'|x)$ is probability density for the Markov chain to move to state x' when it is currently in state x. (Convergence also requires that the Markov chain not get trapped in some subset of the state space.)

Invariance is implied by *reversibility* (also called "detailed balance") with respect to π:

$$\text{For all } x \text{ and } x', \quad \pi(x)\, T(x'|x) \;=\; \pi(x')\, T(x|x')$$

Just integrate both sides over x to see this.

But reversibility is not *necessary* — non-reversible Markov chains that leave π invariant exist and are useful.

The Metropolis Algorithm

[Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller, 1953]

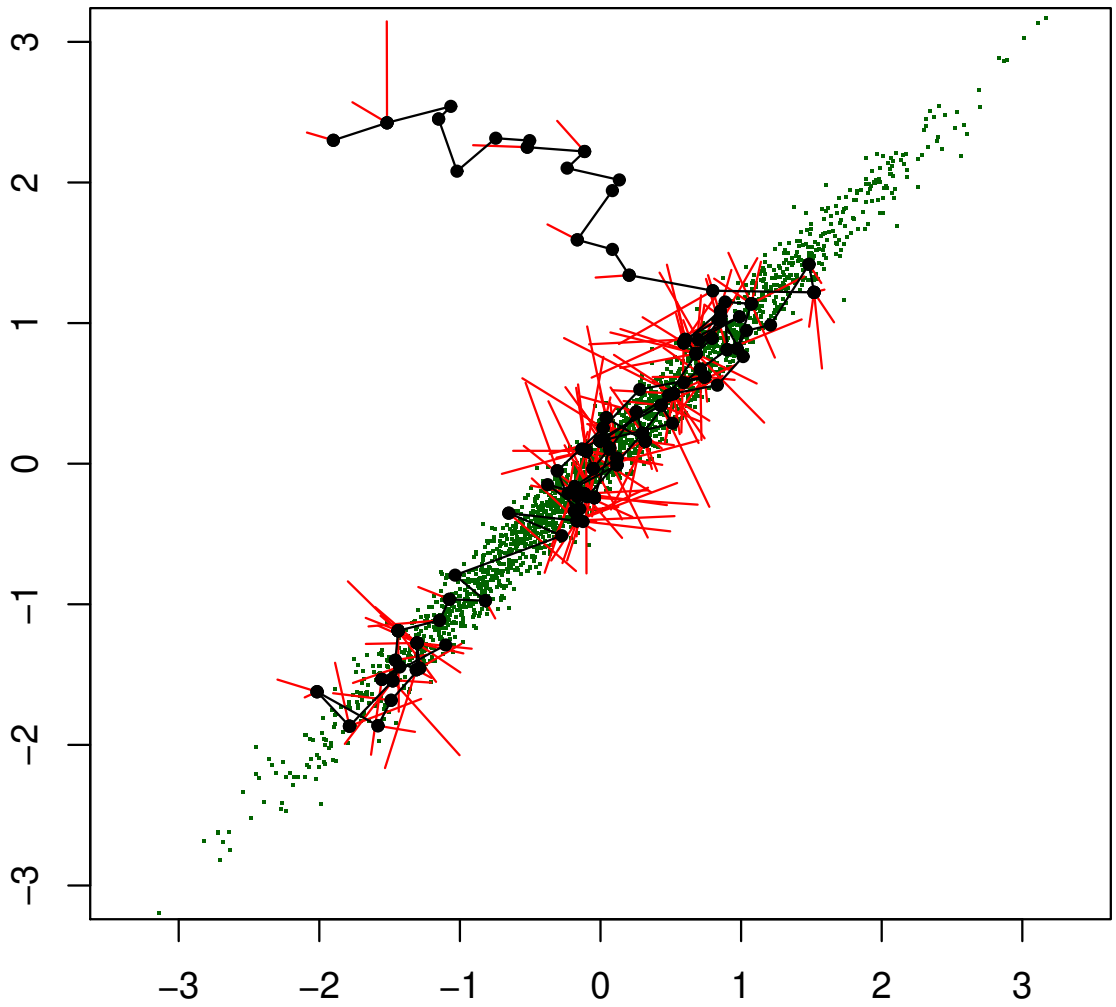
A very general way of defining a transition that's reversible for π was devised by Metropolis, et. al. — *propose* a state, x^*, to move to from x, and then *accept* or *reject* the proposal based on the ratio $\pi(x^*)/\pi(x)$. If we reject the proposal, the new state is the same as the old state.

Let $S(x^*|x)$ be the probability density for proposing to move to x^* when in state x. We require that $S(x^*|x) = S(x|x^*)$.

We accept the proposal x^* with probability $\min[1, \pi(x^*)/\pi(x)]$.
It's easy to show that the resulting transition is reversible with respect to π, and hence leaves π invariant.

Note: We only need the ratio $\pi(x^*)/\pi(x)$, which we can get even if we can only compute an unnormalized density function.

Illustration: Metropolis for Bivariate Gaussian



$\mathrm{Var}(x_1) = \mathrm{Var}(x_2) = 1$

$\mathrm{Cov}(x_1, x_2) = 0.99$

$S(x^*|x) = N(x^*; x, 0.3^2\, I)$

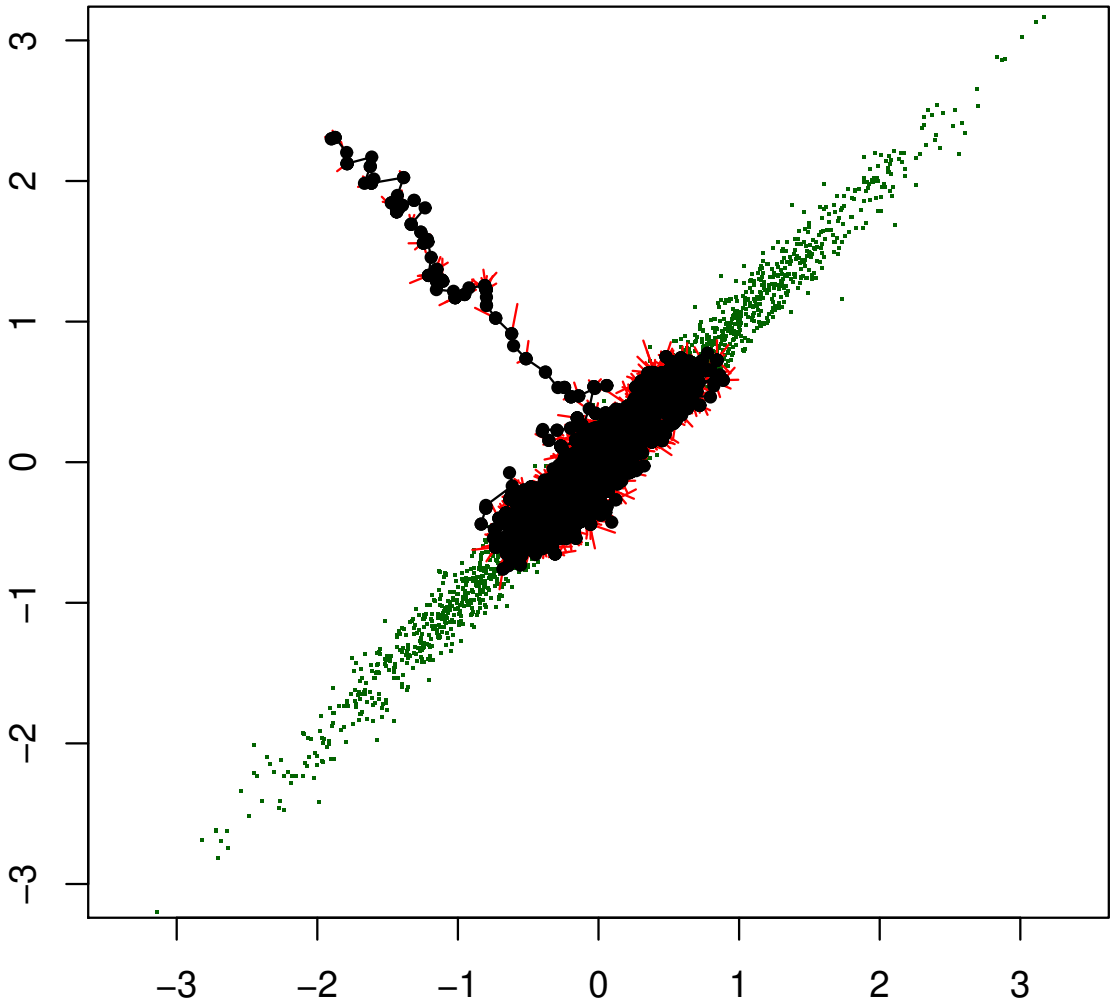
Green points are an i.i.d. sample from $\pi(x)$.

Black points show 250 transitions of the Markov chain

Rejection rate (last 90%) is 0.68. Red lines point to rejected proposals

When started from a low-probability point, the chain moves steadily towards the high-probability region. But once there, it wanders about the distribution in a random walk, often doubling back on itself.

Metropolis for Replicated Bivariate Gaussian



$\mathrm{Var}(x_i) \;=\; 1, \;\; i = 1, \ldots, 20$

$\mathrm{Cov}(x_{2j-1}, x_{2j}) \;=\; 0.99$

$S(x^*|x) \;=\; N(x^*; x,\, 0.07^2\, I)$

Green points are an i.i.d. sample from $\pi(x_1, x_2)$.

Black points show 4500 transitions of the Markov chain

Rejection rate (last 90%) is 0.71. Red lines point to rejected proposals

To get a similar rejection rate with 20 dimensions, a smaller proposal standard deviation is needed. So the random walk takes smaller steps. About 18 times more transitions are needed to move a similar distance.

The Inefficiency of Random Walks

Following an initial period of approach to convergence, reversibility implies that Metropolis transitions that each move only a small distance will explore high-probability regions via a *random walk*, with no tendency to keep going in the same direction.

This is inefficient.

A simple example: Suppose $x_{t+1} = x_t + n_t$, where n_t is a random draw from $N(0, 1)$, independently for each t. Then x_{t+K} is likely to be only about $\sqrt{K}$ away from x_t — not about K away, as one might expect if the n_t all had the same sign.

Enormously faster exploration of the distribution can result from avoiding this inefficiency, by either:

- Using transitions that make big rather than small changes, or
- Not doing a random walk (using non-reversible transitions).

Combining Transitions in Sequence

If we have several Markov transitions, $T_1, T_2, \ldots, T_k$, all of which leave the distribution π invariant, then the combined transition that applies each of these T_i in sequence will also leave π invariant.

But even if $T_1, T_2, \ldots, T_k$ are all reversible w.r.t. π, the combination will generally not be reversible.

Gibbs Sampling: For a multivariate state, $x = (x_1, \ldots, x_k)$, each T_i might update only component x_i, replacing it with a random value from its conditional distribution given the other components.

Gibbs sampling is generally not reversible, but the non-reversibility seems to have no important consequences. But in other situations, non-reversible transitions constructed by sequential combinations can be much better than reversible methods.

Hamiltonian Monte Carlo (HMC)

[Duane, Kennedy, Pendleton, and Roweth, 1987]

Simple Metropolis proposals (eg, Gaussian) lead to slow exploration via a random walk. Much better is to make distant proposals by simulating Hamiltonian dynamics for some period of fictitious "time".

We augment the variable of interest, x, with a "momentum" variable, p, of equal dimension, with a Gaussian distribution, independent of x.

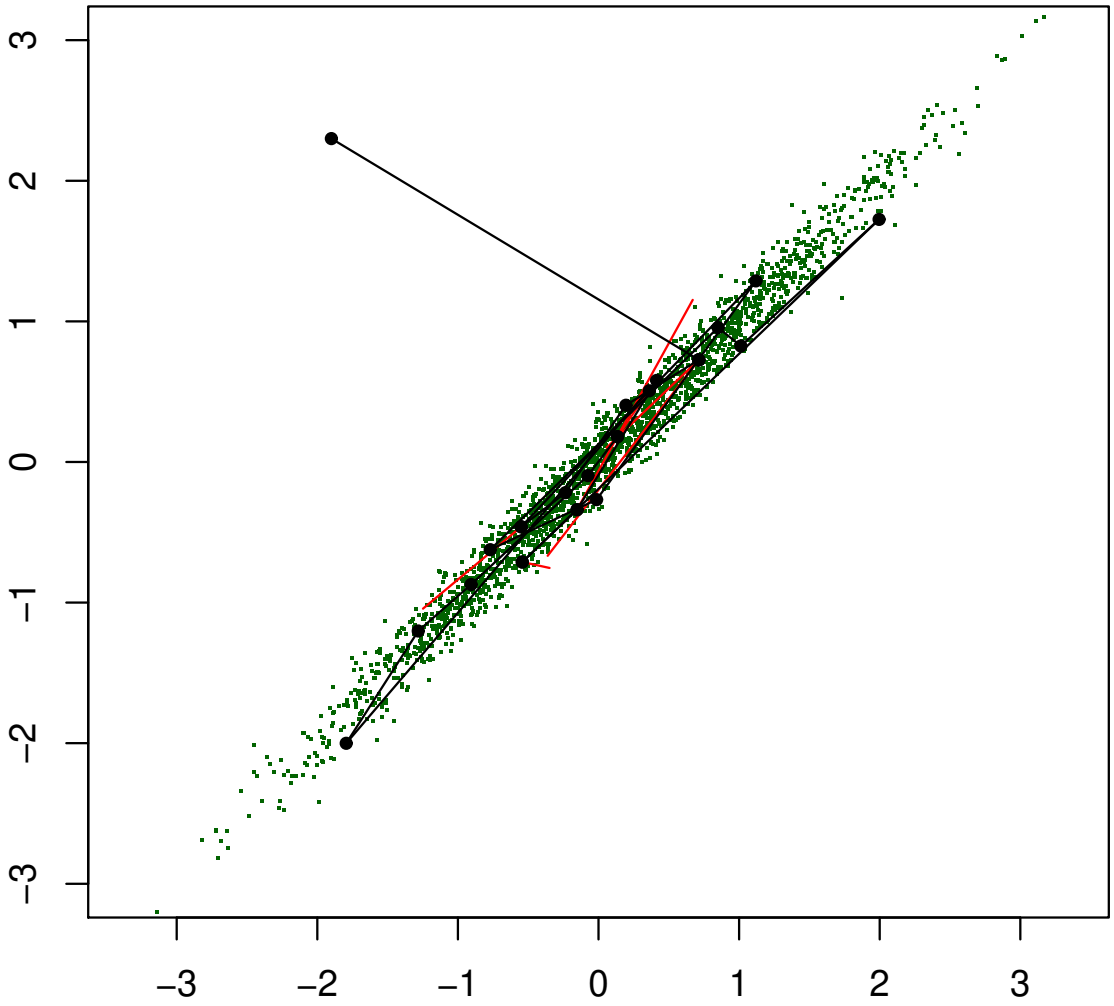
An HMC transition has two parts:

1) Sample p from its distribution (eg, $N(0, I)$).

2) Do a Metropolis update, with proposal found by simulating Hamiltonian dynamics from (x, p) for some time τ (then negating p so the proposal is symmetrical).

If the dynamical simulation were exact, the proposal would always be accepted — the dynamics preserves the log of the joint density of (x, p). In practice, we simulate the dynamics with L "leapfrog" steps, each for a time $\eta = \tau/L$. Since these steps are not exact, rejection is possible.

Illustration: HMC for Bivariate Gaussian



$\text{Var}(x_1) = \text{Var}(x_2) = 1$
$\text{Cov}(x_1, x_2) = 0.99$

$\eta = 0.16$, $L = 10$

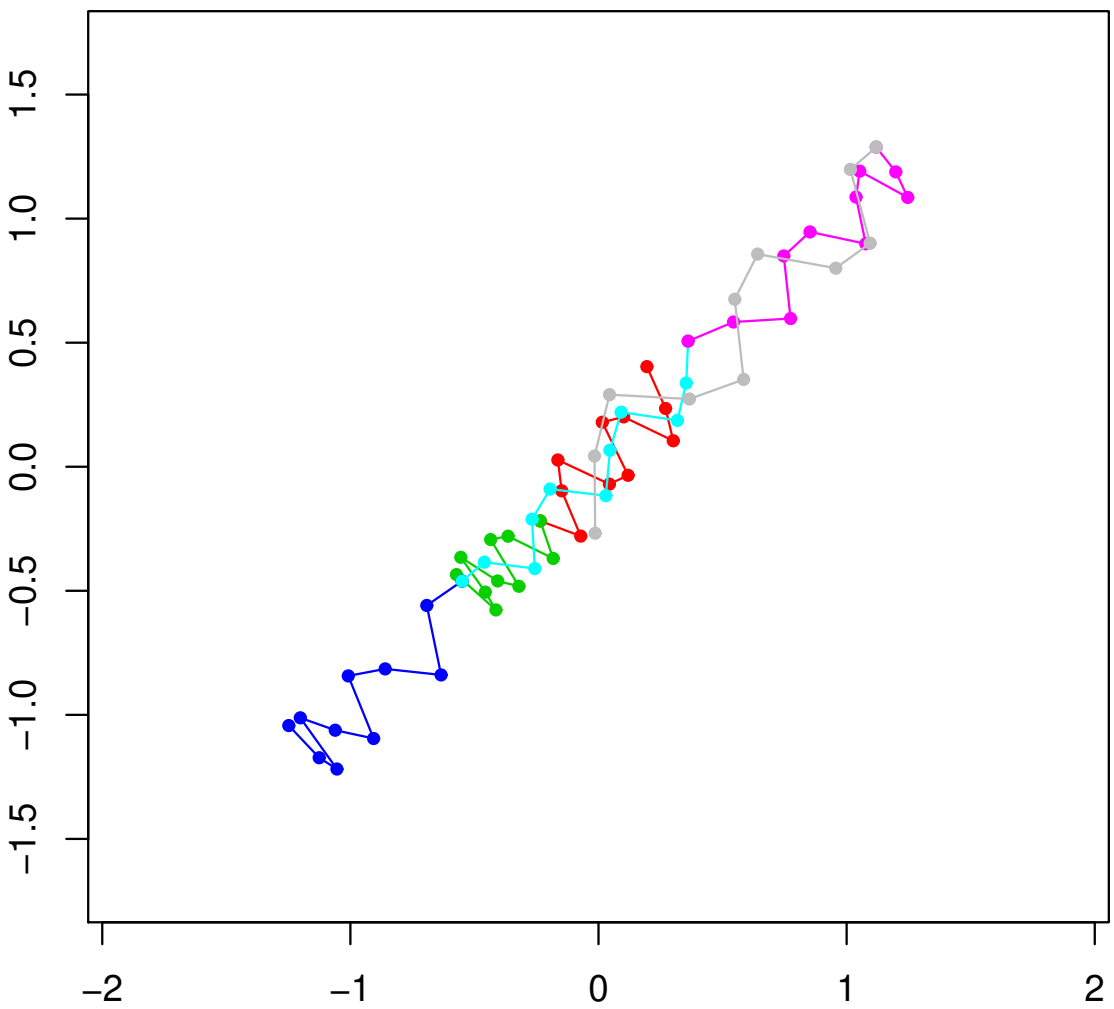
Green points are an i.i.d. sample from $\pi(x)$.

Black points show 25 transitions of the Markov chain

Rejection rate (last 90%) is 0.22. Red lines point to rejected proposals

HMC proposals (as found with suitably long trajectories) are often to states distant from the current state. So even though HMC is reversible, random walks are not a problem.

Some HMC Trajectories for the Bivariate Gaussian



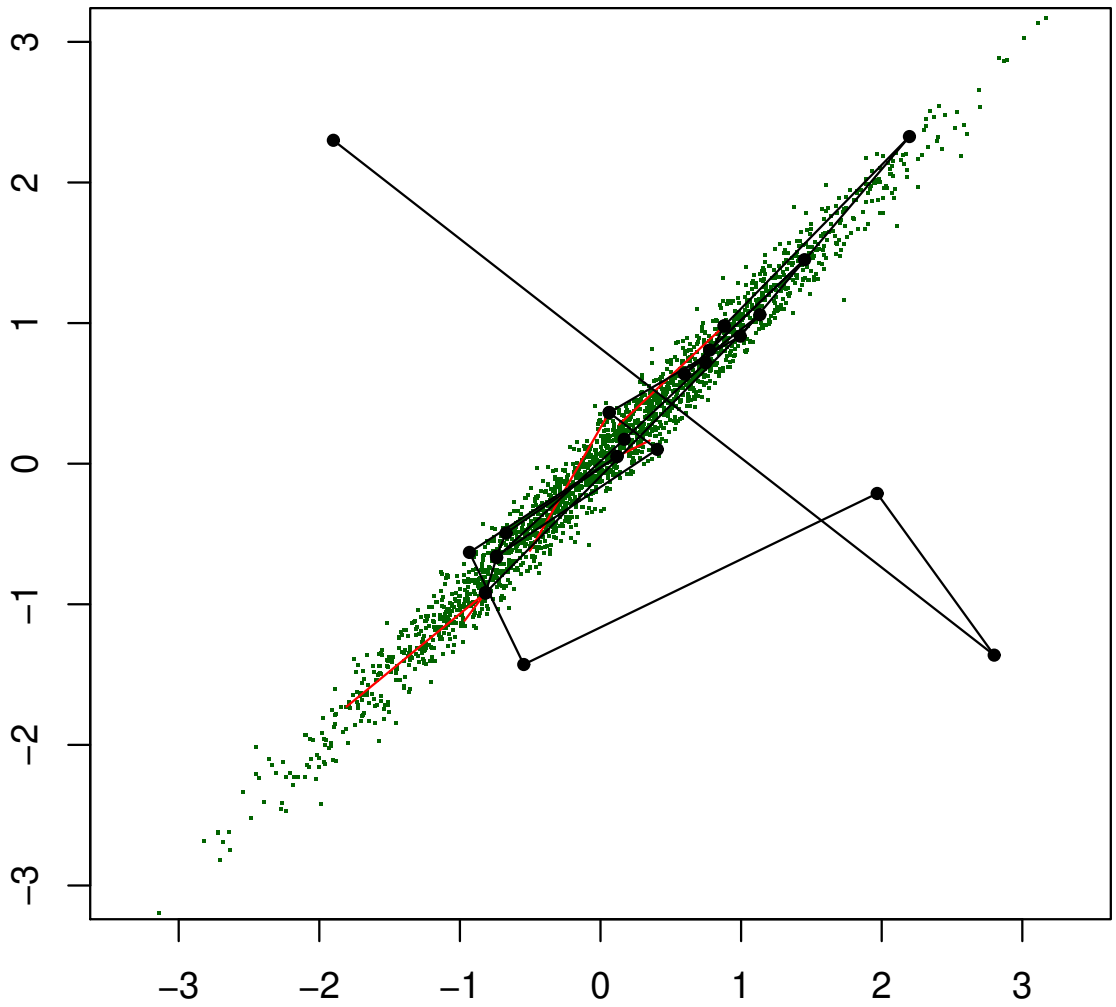
Six successive trajectories (red first) used to produce proposals for HMC (blue rejected, others accepted)

Only x is shown (not p)

$\eta = 0.16$, $L = 10$

The dynamics confines the trajectories to the high-probability region, while keeping them going in the same direction (except turning at an end). But each trajectory has a random initial direction.

HMC for Replicated Bivariate Gaussian



$\text{Var}(x_i) = 1,\ i = 1, \ldots, 20$

$\text{Cov}(x_{2j-1}, x_{2j}) = 0.99$

$\eta = 0.1,\ L = 16$

Green points are an i.i.d. sample from $\pi(x)$.

Black points show 25 transitions of the Markov chain

Rejection rate (last 90%) is 0.22. Red lines point to rejected proposals

With 20 dimensions, η needs to be smaller, and so L needs to be larger to compensate. But the scaling of HMC with dimensionality is substantially better than for simple Metropolis methods.

Langevin Monte Carlo

[Rossky, Doll, and Friedman, 1978; others later]

Doing only one leapfrog step in HMC is equivalent to "Langevin" Monte Carlo. A Langevin transition goes as follows:

- Sample p (of same dimension as x) from the $N(0, I)$ distribution.
- Compute a proposal (x^*, p^*) with one leapfrog step, as follows:

$$\begin{aligned} p^\circ &= p - (\eta/2)\,\nabla \log \pi(x) \\ x^* &= x + \eta\, p^\circ \\ p^* &= -\,[\,p^\circ - (\eta/2)\,\nabla \log \pi(x^*)\,] \end{aligned}$$

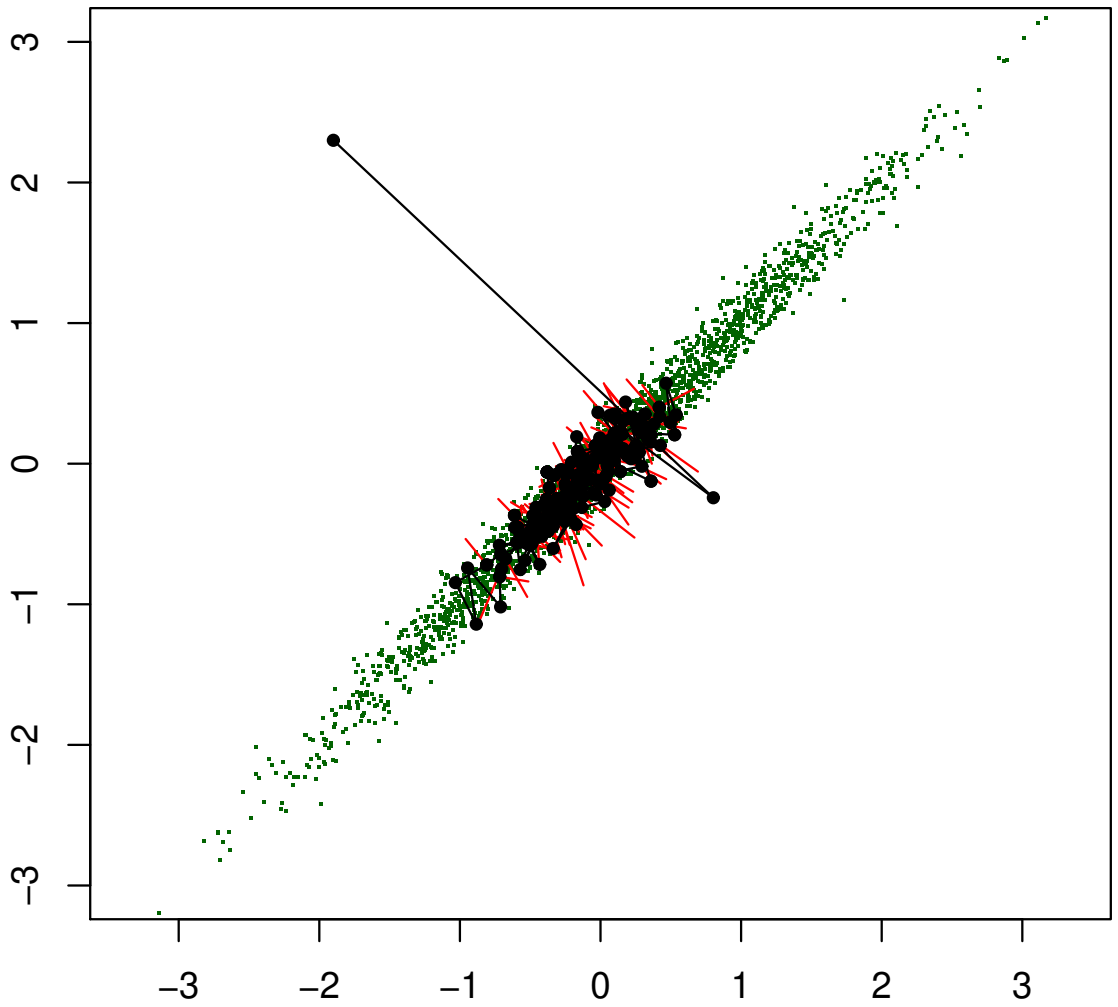
- Accept (x^*, p^*) as the new state with probability

$$\min\,[1,\ \pi(x^*)\phi(p^*)\,/\,\pi(x)\phi(p)]$$

  where $\phi(p)$ is the probability density for the $N(0, I)$ distribution. If (x^*, p^*) is not accepted, the new state is the same as the old.

Note: we can write x^* as $x + (\eta^2/2)\nabla \log \pi(x) + \eta n$, with $n \sim N(0, I)$.

Illustration: Langevin for Bivariate Gaussian



$\mathrm{Var}(x_1) = \mathrm{Var}(x_2) = 1$
$\mathrm{Cov}(x_1, x_2) = 0.99$

$\eta = 0.17.$

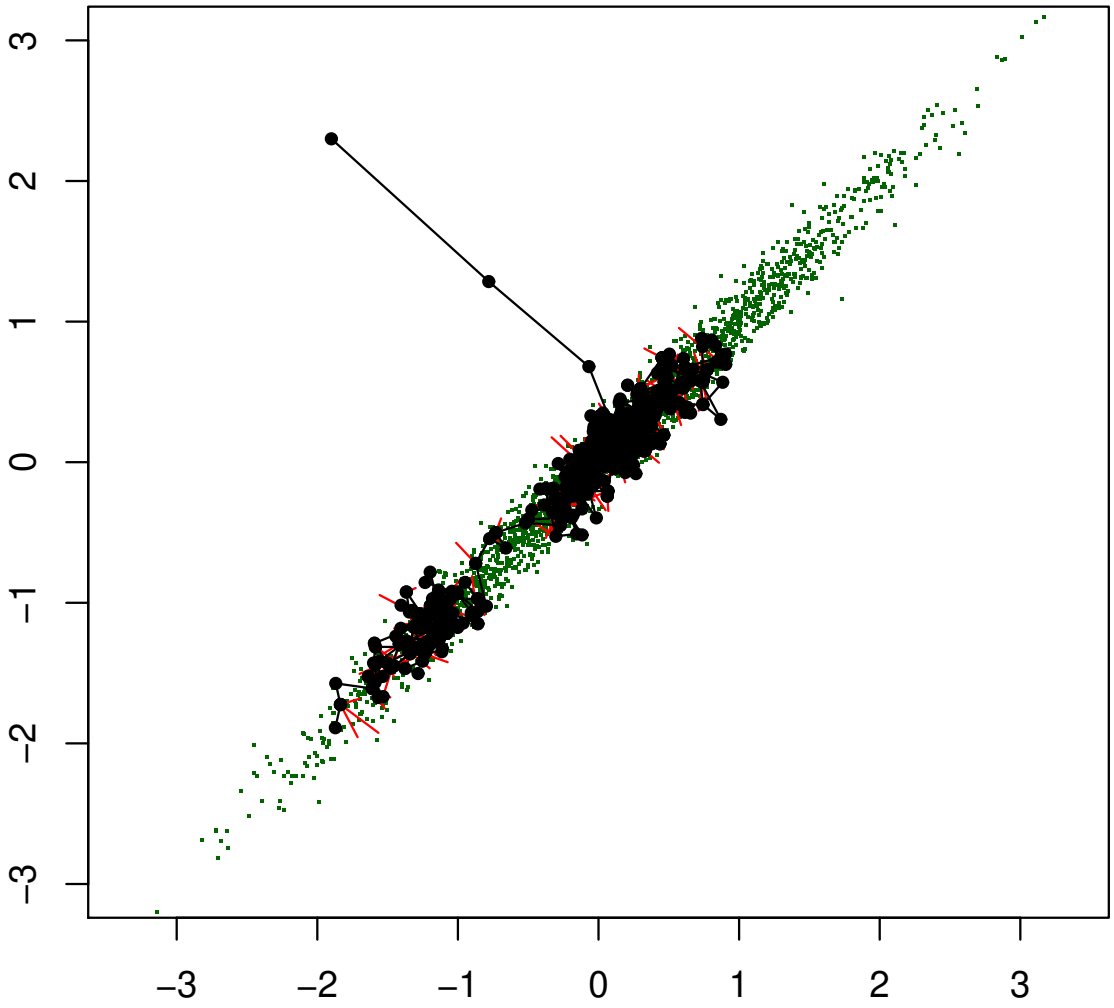
Green points are an i.i.d. sample from $\pi(x)$.

Black points show 250 transitions of the Markov chain

Rejection rate (last 90%) is 0.37. Red lines point to rejected proposals

Langevin benefits from using gradient information, but since it is reversible, and (typically) takes small steps, it still suffers from the inefficiency of a random walk.

Langevin for Replicated Bivariate Gaussian



$\mathrm{Var}(x_i) = 1, \; i = 1, \ldots, 20$

$\mathrm{Cov}(x_{2j-1}, x_{2j}) = 0.99$

$\eta = 0.11.$

Green points are an i.i.d. sample from $\pi(x)$.

Black points show 600 transitions of the Markov chain

Rejection rate (last 90%) is 0.39. Red lines point to rejected proposals

Langevin's scaling with dimensionality is better than for simple Metropolis, but worse than for HMC.

I. Motivation — Hierarchical models and discrete variables

II. MCMC background

III. Non-reversible Langevin methods

IV. Application to hierarchical models and discrete variables

Langevin Monte Carlo with Persistent Momentum

[Horowitz, 1991]

A transition from (x, p) to the next state has three steps:

1) Update p to $\alpha p + \sqrt{1-\alpha^2}\, n$, where α is slightly less than 1 and n is a $N(0, I)$ random variable.

2) Propose a new state by doing one leapfrog step from (x, p) and then negating p. Accept or reject this proposal the usual way.

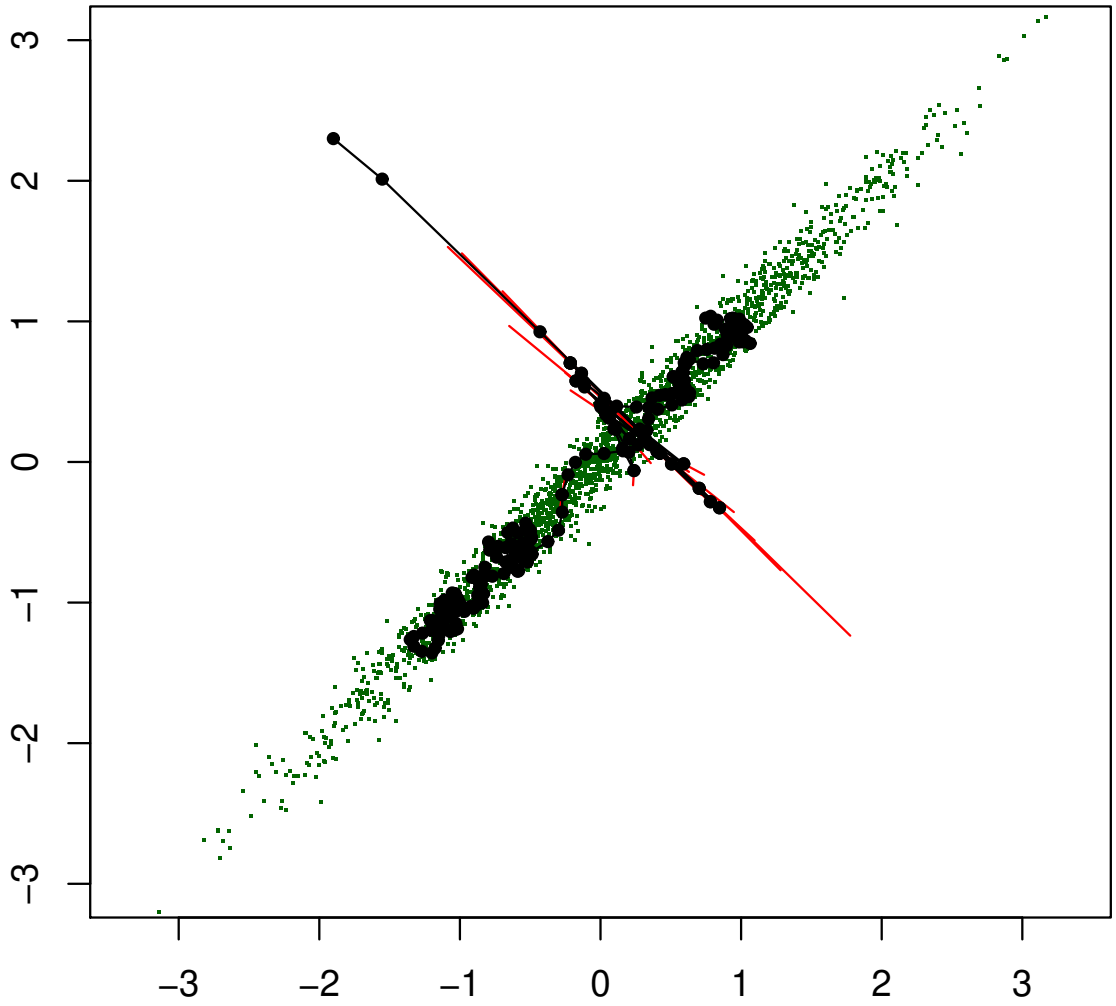
3) Negate p.

All steps leave the desired distribution invariant and are reversible.

Their sequential combination leaves the desired distribution invariant but is not reversible.

For α near 1, Step (1) only slightly changes p. If Step (2) accepts, the negation in the proposal is canceled by the negation in Step (3). But a rejection will reverse p, and the chain will almost double back on itself.

Illustration: Persistent Langevin for Bivariate Gaussian



$\mathrm{Var}(x_1) = \mathrm{Var}(x_2) = 1$
$\mathrm{Cov}(x_1, x_2) = 0.99$

$\eta = 0.062$, $\alpha = 0.94$

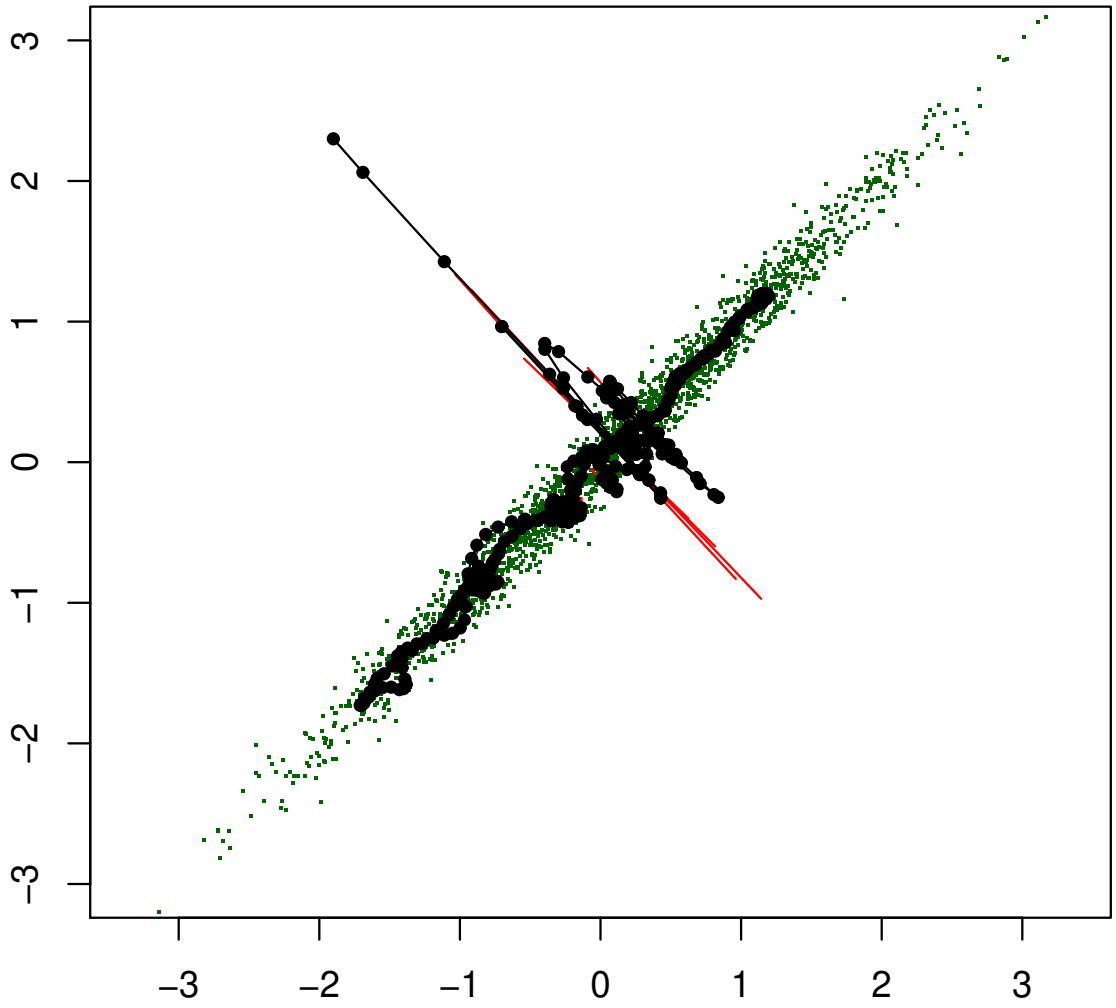
Green points are an i.i.d. sample from $\pi(x)$.

Black points show 250 transitions of the Markov chain

Rejection rate (last 90%) is 0.044. Red lines point to rejected proposals

By only slightly changing p each iteration, persistent Langevin can suppress random walk behaviour. Unfortunately, for this to work, the rejection rate must be small (comparable to or less than $1-\alpha$).

Persistent Langevin for Replicated Bivariate Gaussian



$\mathrm{Var}(x_i) = 1,\ i = 1, \ldots, 20$
$\mathrm{Cov}(x_{2j-1}, x_{2j}) = 0.99$

$\eta = 0.045,\ \alpha = 0.95$

Green points are an i.i.d. sample from $\pi(x)$.

Black points show 345 transitions of the Markov chain

Rejection rate (last 90%) is 0.035. Red lines point to rejected proposals

As dimensionality increases, η must be made even smaller in order to keep the rejection rate very small.

Avoiding Reversals $\Rightarrow$ Inefficient Small Stepsize

Unfortunately, though persistent Langevin can avoid random walks, it does so only *only* if the rejection rate is small. This requires a small η, which slows speed of exploration.

So it's not as good as Hamiltonian Monte Carlo, which can use a comparatively large η even when the number of leapfrog steps, L, needs to be quite large in order to avoid random walks.

The new innovation: As well as the non-reversibility from not completely replacing p, also introduce non-reversibility into the *acceptance decision*.

Non-Reversible Form of the Acceptance Decision

To decide on accepting a Metropolis proposal to move from x to x^*, we can check if $u < \pi(x^*)/\pi(x)$, with u a random uniform over $[0, 1]$.

Equivalently, we can check whether $\pi(x^*) > s$, where s is a random uniform over $[0, \pi(x)]$.

Rather than choosing s randomly, we can make it, or $u = s/\pi(x)$, part of the state, and update it in any way that leaves the joint distribution invariant.

One possible update: For some constant δ, add/subtract $\delta\pi(x)$ to s, reflecting off the boundaries at 0 and $\pi(x)$.

Implementation details: We let $s = \pi(x)|v|$, with v having the uniform(−1,+1) distribution. We update v by adding δ, and then subtracting 2 if $v > 1$. If x changes to x', we make a corresponding change of v to $v' = v\pi(x)/\pi(x')$, which keeps s unchanged.

Non-Reversible Acceptance Can Cluster Rejections, Avoid Random Walks at a Higher Rejection Rate

If the rejection rate is not high, $\pi(x^*)/\pi(x)$ will usually be close to one. So u will mostly change as a result of adding δ (reflecting off 0 and 1). If δ is small, u will be near 0 for a while, then near 1 for a while, etc.

Rejections will tend to be *clustered*, and acceptances will be too. (Overall rejection rate will be same as for the standard method.)

Clustering of rejections produces *less random walk behaviour.*

Compare: If each accept moves d in same direction, each reject randomizes direction, then $20K$ iterations of the following form:

4 accept, 1 reject, 4 accept, 1 reject, 4 accept, 1 reject, 4 accept, 1 reject, ...

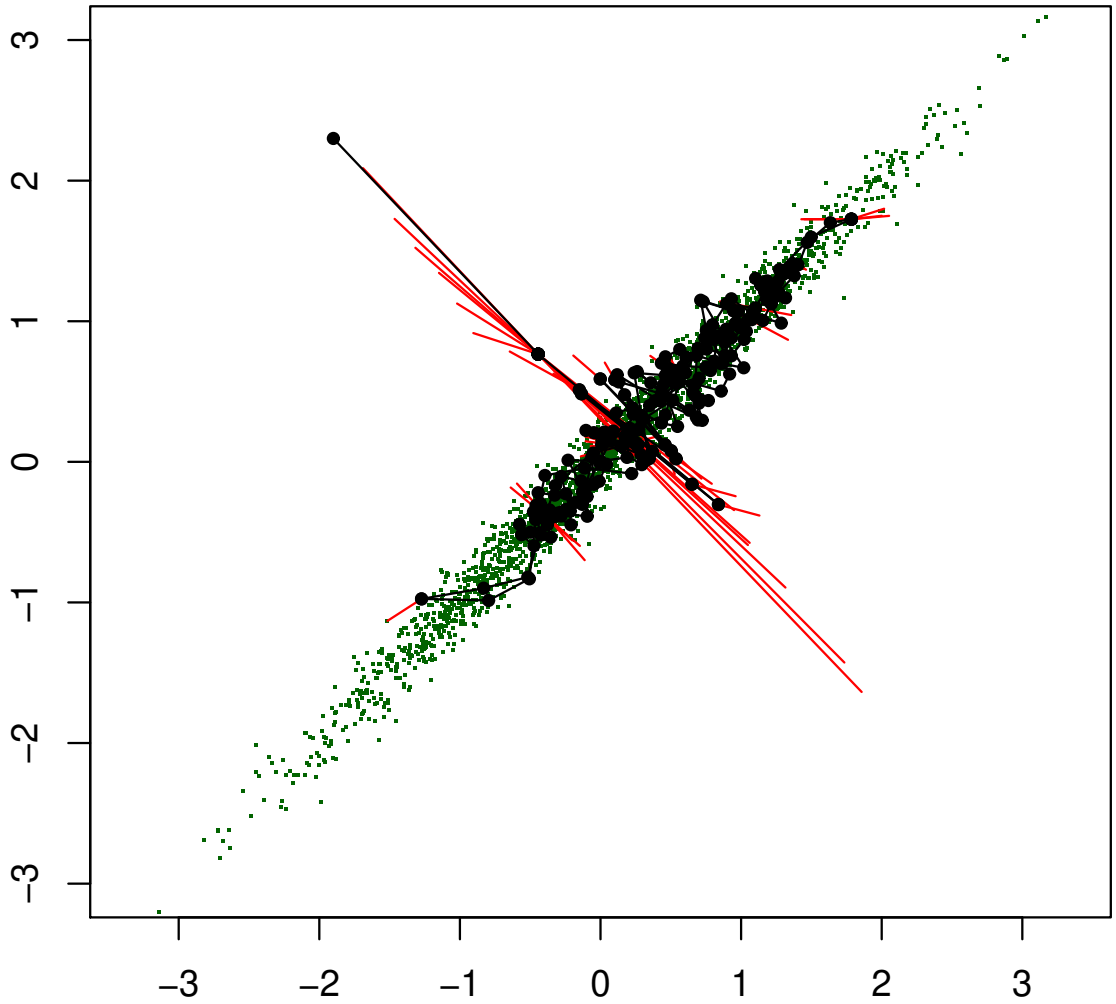
move on average a distance of $4d\sqrt{4K} \;=\; 8d\sqrt{K}$.

But $20K$ iterations of the form:

16 accept, 4 reject, ...

move on average a distance of $16d\sqrt{K}$.

Illustration: Persistent Langevin with Non-Reversible Acceptance for Bivariate Gaussian



$\mathrm{Var}(x_1) = \mathrm{Var}(x_2) = 1$
$\mathrm{Cov}(x_1, x_2) = 0.99$

$\eta = 0.12$, $\alpha = 0.92$, $\delta = 0.05$

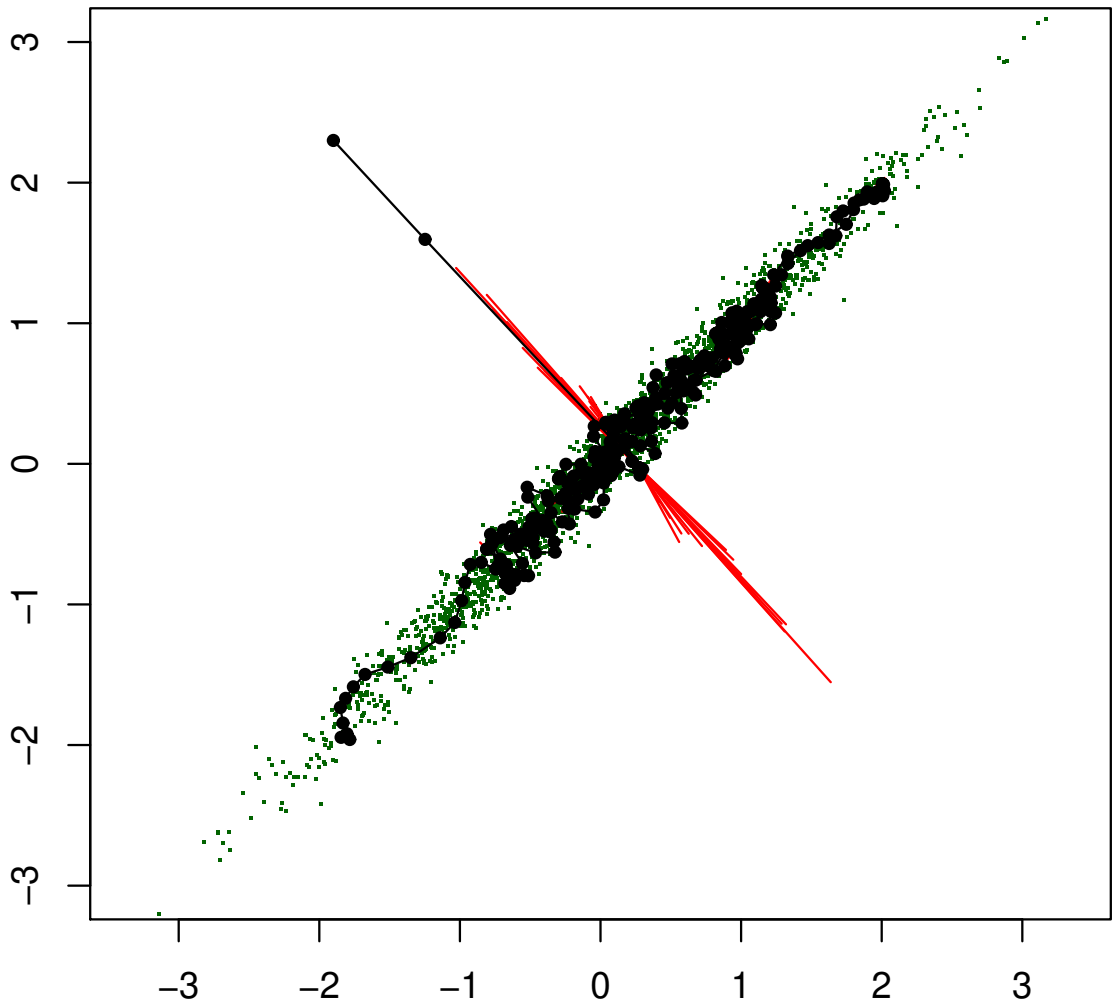
Green points are an i.i.d. sample from $\pi(x)$.

Black points show 250 transitions of the Markov chain

Rejection rate is 0.13; red lines point to rejected proposals

Compare to standard acceptance, with $\eta = 0.062$ and rejection rate of 0.044. Here, random walks are mostly suppressed despite a rejection rate of 0.13, and the larger η of 0.12 leads to more movement.

Persistent Langevin with Non-Reversible Acceptance for Replicated Bivariate Gaussian



$\mathrm{Var}(x_i) \;=\; 1,\; i = 1, \ldots, 20$
$\mathrm{Cov}(x_{2j-1}, x_{2j}) \;=\; 0.99$

$\eta = 0.08$, $\alpha = 0.94$, $\delta = 0.05$

Green points are an i.i.d. sample from $\pi(x)$.

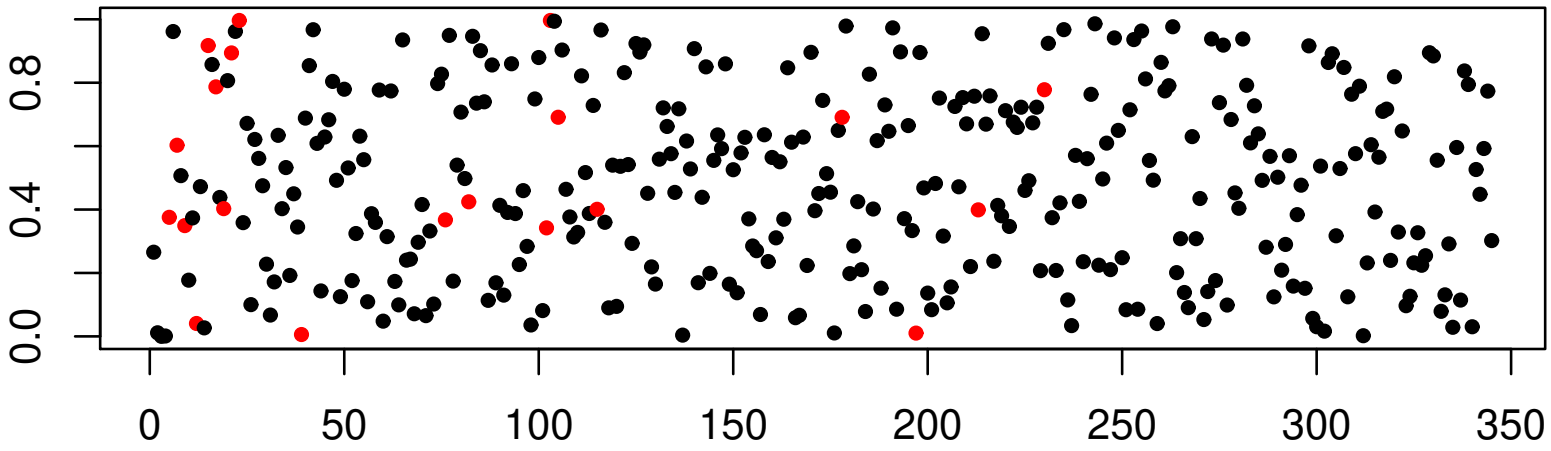
Black points show 345 transitions of the Markov chain

Rejection rate is 0.11; red lines point to rejected proposals

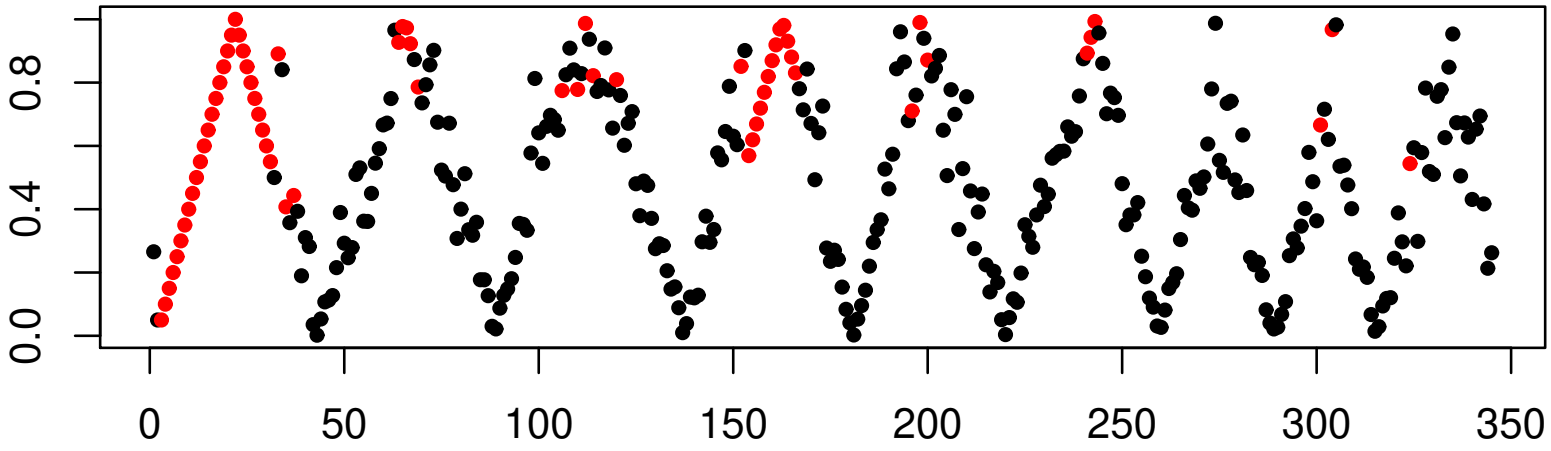
A smaller η is needed in higher dimensions, then α needs to be closer to 1 for the same random walk suppression. Choosing δ to be roughly $1-\alpha$ seems about right. Performance here seems comparable to HMC.

Changes to u and Clustering of Rejections

Plots of u values for each accept/reject decision, with values leading to rejection in red, for sampling from replicated bivariate Gaussian:



Persistent Langevin



Persistent Langevin with non-reversible update of u

I. Motivation — Hierarchical models and discrete variables

II. MCMC background

III. Non-reversible Langevin methods

IV. Application to hierarchical models and discrete variables

Test on a Model with Discrete Variables

I applied non-reversible Langevin and other methods to this problem:

$$\begin{aligned} u &\sim N(0,1) \\ v \mid u &\sim N(u,\, 0.04^2) \\ w_1 \mid u &\sim \text{Bernoulli}\,(1/(1+e^u)) \\ &\vdots \\ w_{20} \mid u &\sim \text{Bernoulli}\,(1/(1+e^u)) \end{aligned}$$

I used Gibbs sampling to update $w_1, \ldots, w_{20}$, and updated u and v with either HMC (alternating with Gibbs sampling) or non-reversible Langevin (with Gibbs sampling done every tenth iteration).

Assuming that updating the continuous variables is the dominant cost (typical for real problems this test is meant to mimic), non-reversible Langevin was 1.83 times more efficient than HMC, with optimal tuning.

Optimal tuning for HMC was $L = 40$ and $\eta = 0.035$ (best choice from $L \in \{30, 40, 60\}$ and $\eta \in \{0.025, 0.030, 0.035, 0.040, 0.045\}$).

Optimal tuning for non-reversible Langevin was $\delta = 0.01$, $\alpha = 0.995$, and $\eta = 0.03$ (best choice from $\delta \in \{0.003, 0.005, 0.010, 0.015\}$, $\alpha \in \{0.98, 0.99, 0.005, 0.0075, 0.9985, 0.9990\}$, and $\eta \in \{0.015, 0.020, 0.025, 0.030, 0.040, 0.050\}$.)

Bayesian Neural Network models

I've tried using non-reversible Langevin methods for Bayesian neural networks, of varying complexity, for regression or classification.

Some networks of increasing complexity:

- One hidden layer of tanh units. Three hyperparameters, controlling
  — input to hidden weights
  — hidden unit biases
  — hidden to output weights
- One hidden layer of tanh units. $k+3$ hyperparameters, controlling
  — input to hidden weights from input j, for $j = 1, \ldots, k$,
  plus a second-level hyperparameter controlling them
  — hidden unit biases
  — hidden to output weights
- $k+1$ hidden layers, not connected to each other, all connected to the output, with the first k looking at only one input, the last looking at all inputs. $3k + k + 3$ hyperparameters.

How Well do the Methods Work on These Models?

It's actually pretty hard to say! We don't know the correct answer.

We can look at the error for predictions on a test set, but it's possible for a method worse at sampling the posterior to predict better, and all methods ought to produce the same predictions if given enough time (but maybe more than is practical).

For a simple classification model — 5 inputs (only 2 relevant), one hidden layer of 12 tanh units, 300 training cases – I found that non-reversible Langevin had 1.25 times smaller autocorrelation time for the hyperparameters than HMC.

For a more complex regression problem — 6 inputs (all relevant to varying degrees), both additive and interaction aspects, 500 training cases — for the simplest models (3 hyperparameters), HMC did better at prediction than non-reversible Langevin, but for the more complex models, the best predictions with were made by non-reversible Langevin methods.

Bibliography

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987) "Hybrid Monte Carlo", *Physics Letters B*, vol. 195, pp. 216-222.

Gelfand, A. E. and Smith, A. F. M. (1990) "Sampling-based approaches to calculating marginal densities", *Journal of the American Statistical Association*, vol. 85, pp. 398-409.

Horowitz, A. M. (1991) "A generalized guided Monte Carlo algorithm", *Physics Letters B*, vol. 268, pp. 247-252.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953) "Equation of state calculations by fast computing machines", *Journal of Chemical Physics*, vol. 21, pp. 1087-1092.

Neal, R. .M. (1994) *Bayesian Learning for Neural Networks*, PhD thesis, University of Toronto.

Neal, R. M. (2003) "Slice sampling" (with discussion), *Annals of Statistics*, vol. 31, pp. 705-767.

Neal, R. M. (2005) "The short-cut Metropolis method", `arxiv.org/abs/math/0508060`

Neal, R. M. (2010) "MCMC using Hamiltonian dynamics", in the *Handbook of Markov Chain Monte Carlo*, S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng (editors), Chapman & Hall / CRC Press, pp. 113-162. Also available at `arxiv.org/abs/1206.1901`

Neal, R. M. (2020) "Non-reversibly updating a uniform [0,1] value for Metropolis accept/reject decisions", `arxiv.org/abs/2001.11950`

Rossky, P. J., Doll, J. D., and Friedman, H. L. (1978) "Brownian dynamics as smart Monte Carlo simulation", *Journal of Chemical Physics*, vol. 69, pp. 4628-4633.