No - U - Turn Sampler

based on M.D.Hoffmann and A.Gelman (2011)

Marco Banterle

Presented at the "Bayes in Paris" reading group

11 / 04 / 2013

Outline

Outline

Hamiltonian dynamic

Hamiltonian MC techniques have a nice foundation in physics

Describe the total energy of a system composed by a frictionless particle sliding on a (hyper-)surface

Position $q \in \mathbb{R}^d$ and momentum $p \in \mathbb{R}^d$ are necessary to define the total energy $H(q, p)$, which is usually formalized through the sum of a potential energy term $U(q)$ and a kinetic energy term $\mathcal{K}(p)$.

Hamiltonian dynamics are characterized by

$$\frac{\partial q}{\partial t} = \frac{\partial H}{\partial p}, \qquad \frac{\partial p}{\partial t} = -\frac{\partial H}{\partial q}$$

that describe how the system changes though time.

Properties of the Hamiltonian

- **Reversibility** : the mapping from $(q,p)_t$ to $(q,p)_{t+s}$ is 1-to-1
- **Conservation of the Hamiltonian** : $\frac{\partial H}{\partial t} = 0$
  total energy is conserved
- **Volume preservation** : applying the map resulting from time-shifting to a region R of the (q,p)-space do not change the volume of the (projected) region
- **Symplecticness** : stronger condition than volume preservation.

What about MCMC?

$$H(q, p) = U(q) + \mathcal{K}(p)$$

We can easily interpret $U(q)$ as minus the log target density for the variable of interest q, while p will be introduced artificially.

What about MCMC?

$$H(q,p) = U(q) + \mathcal{K}(p) \rightarrow -\log\left(\pi(q|y)\right) - \log(f(p))$$

Let's examine its properties under a statistical lens:

- **Reversibility** : MCMC updates that use these dynamics leave the desired distribution invariant
- **Conservation of the Hamiltonian** : Metropolis updates using Hamiltonians are always accepted
- **Volume preservation** : we don't need to account for any change in volume in the acceptance probability for Metropolis updates (no need to compute the determinant of the Jacobian matrix for the mapping)

the Leapfrog

Hamilton's equations are not always[1] explicitly available, hence the need for time discretization with some small step-size ε.

A numerical integrator which serves our scopes is called Leapfrog

Leapfrog Integrator

For $j = 1, \ldots, L$

1. $p_{t+\varepsilon/2} = p_t - (\varepsilon/2)\frac{\partial U}{\partial q}(q_t)$
2. $q_{t+\varepsilon} = q_t + \varepsilon \frac{\partial \mathcal{K}}{\partial p}(p_{t+\varepsilon/2})$
3. $p_{t+\varepsilon} = p_{t+\varepsilon/2} - (\varepsilon/2)\frac{\partial U}{\partial q}(q_{t+\varepsilon})$

   $t = t + \varepsilon$

[1] unless they're quadratic form

A few more words

The kinetic energy is usually (for simplicity) assumed to be $\mathcal{K}(p) = p^T M^{-1} p$ which correspond to $p|q \equiv p \sim \mathcal{N}(0, M)$.
This finally implies that in the leapfrog $\frac{\partial \mathcal{K}}{\partial p}(p_{t+\varepsilon/2}) = M^{-1} p_{t+\varepsilon/2}$

Usually M, for which few guidance exists, is taken to be diagonal and often equal to the identity matrix.

The leapfrog method preserves volume exactly and due to its symmetry it is also reversible by simply negating p, applying the same number L of steps again, and then negating p again[2].
It does not however conserve the total energy and thus this deterministic move to (q', p') will be accepted with probability

$$\min\left[1, \exp(-H(q', p') + H(q, p))\right]$$

[2] negating ε serves the same purpose

HMC algorithm

We now have all the elements to construct an MCMC method based on the Hamiltoninan dynamics:

HMC

Given q_0, ε, L, M

For $i = 1, \ldots, N$

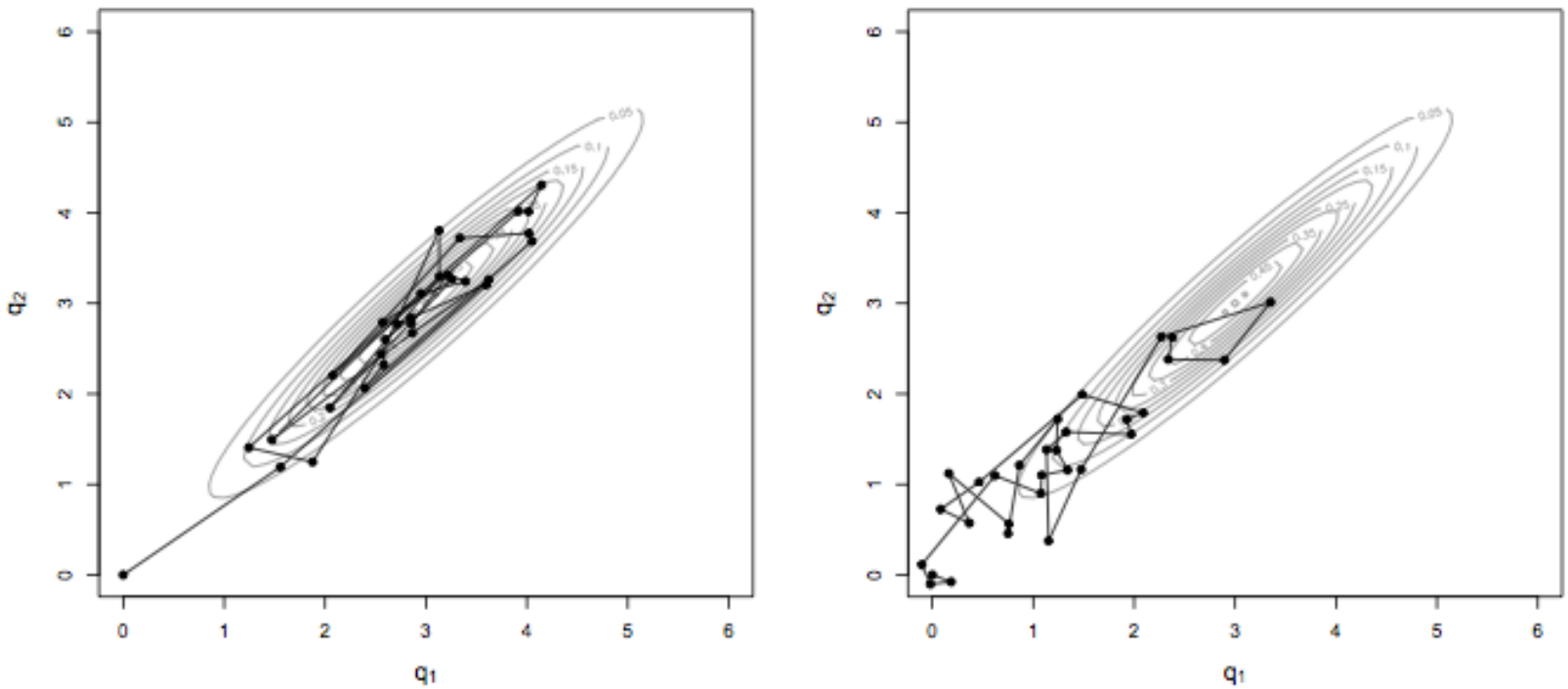
1. $p \sim \mathcal{N}(0, M)$
2. Set $q_i \leftarrow q_{i-1}, \quad q' \leftarrow q_{i-1}, \quad p' \leftarrow p$
3. for $j = 1, \ldots, L$

   $update(q', p')$ *through leapfrog*
4. Set $q_i \leftarrow q'$ with probability

$$\min\left[1, \exp(-H(q', p') + H(q_{i-1}, p_{i-1}))\right]$$

HMC benefits

HMC make use of gradient information to move across the space and so its typical trajectories do not resemble random-walks. Moreover the error in the Hamiltonian stays bounded[3] and hence we also have high acceptance rate.



[3] even if theoretical justification for that are missing

HMC benefits

Random-walk:

- require changes proposed with magnitude comparable to the sd in the most constrained direction (square root of the smallest eigenvalue of the covariance matrix)
- to reach a almost-independent state need a number of iterations mostly determined by how long it takes to explore the less constrained direction
- proposals have no tendency to move consistently in the same direction.

For HMC the proposal move accordingly to the gradient for L step, even if ε is still constrained by the smallest eigenvalue.

Connected difficulties

Performance depends strongly on choosing suitable values for ε and L.

- ε too large $\rightarrow$ inaccurate simulation & high reject rate
- ε too small $\rightarrow$ wasted computational power (small steps)
- L too small $\rightarrow$ random walk behavior and slow mixing
- L too large $\rightarrow$ trajectories retrace their steps $\rightarrow$ **U-TURN**

Connected difficulties

Performance depends strongly on choosing suitable values for ε and L.

- ε too large $\rightarrow$ inaccurate simulation & high reject rate
- ε too small $\rightarrow$ wasted computational power (small steps)
- L too small $\rightarrow$ random walk behavior and slow mixing
- L too large $\rightarrow$ trajectories retrace their steps $\rightarrow$ **U-TURN**

even worse: PERIODICITY!!

Periodicity

Ergodicity of HMC may fail if a produced trajectory of length (Lε) is an exact periodicity for some state.

Example

Consider q $\sim \mathcal{N}(0,1)$, then $H(q,p) = q^2/2 + p^2/2$ and the resulting Hamiltonian is

$$\frac{dq}{dt} = p \quad \frac{dp}{dt} = -q$$

and hence has an exact solution

$$q(L\varepsilon) = r\cos(a + L\varepsilon), \quad p(L\varepsilon) = -r\sin(a + L\varepsilon)$$

for some real a and r.

Periodicity

Example

$$q \sim \mathcal{N}(0,1),\ H(q,p) = q^2/2 + p^2/2$$

$$\frac{dq}{dt} = p \quad \frac{dp}{dt} = -q$$

$$q(L\varepsilon) = r\cos(a + L\varepsilon), \quad p(L\varepsilon) = -r\sin(a + L\varepsilon)$$

If we chose a trajectory length such that $L\varepsilon = 2\pi$ at the end of the iteration we will return to the starting point!

Periodicity

Example

$$q \sim \mathcal{N}(0,1),\ H(q,p) = q^2/2 + p^2/2$$

$$\frac{dq}{dt} = p \quad \frac{dp}{dt} = -q$$

$$q(L\varepsilon) = r\cos(a + L\varepsilon), \quad p(L\varepsilon) = -r\sin(a + L\varepsilon)$$

If we chose a trajectory length such that $L\varepsilon = 2\pi$ at the end of the iteration we will return to the starting point!

- $L\varepsilon$ near the period makes HMC ergodic but practically useless
- interactions between variables prevent exact periodicities, but near periodicities might still slow HMC considerably!

Other useful tools - Windowed HMC

The leapfrog introduces "random" errors in H at each step and hence the acceptance probability may have an high variability. Smoothing (over windows of states) out this oscillations could lead to higher acceptance rates.

Other useful tools - Windowed HMC

The leapfrog introduces "random" errors in H at each step and hence the acceptance probability may have an high variability. Smoothing (over windows of states) out this oscillations could lead to higher acceptance rates.

Windows map

We map $(q, p) \to [(q_0, p_0), \ldots, (q_{W-1}, p_{W-1})]$ by writing

$$(q, p) = (q_s, p_s), \; s \sim \mathcal{U}(0, W-1)$$

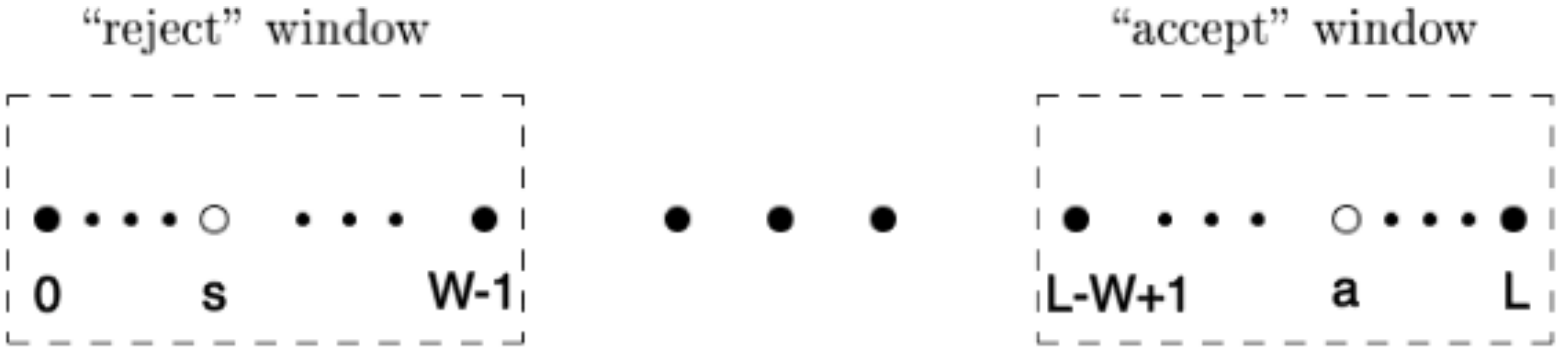
and deterministically recover the other states. This windows has probability density

$$P([(q_0, p_0), \ldots, (q_{W-1}, p_{W-1})]) = \frac{1}{W} \sum_{i=0}^{W-1} P(q_i, p_i)$$

Windowed HMC

Similarly we perform L - W + 1 leapfrog steps starting from (q_{W-1}, p_{W-1}), up to (q_L, p_L) and then accept the window $[(q_{L-W+1}, -p_{L-W+1}), \ldots, (q_L, -p_L)]$ with probability

$$\min\left[1, \frac{\sum_{i=L-W+1}^{L} P(q_i, p_i)}{\sum_{i=0}^{W-1} P(q_i, p_i)}\right]$$



and finally select $(q_a, -p_a)$ with probability

$$\frac{P(q_a, p_a)}{\sum_{i \in \text{accepted window}} P(q_i, p_i)}$$

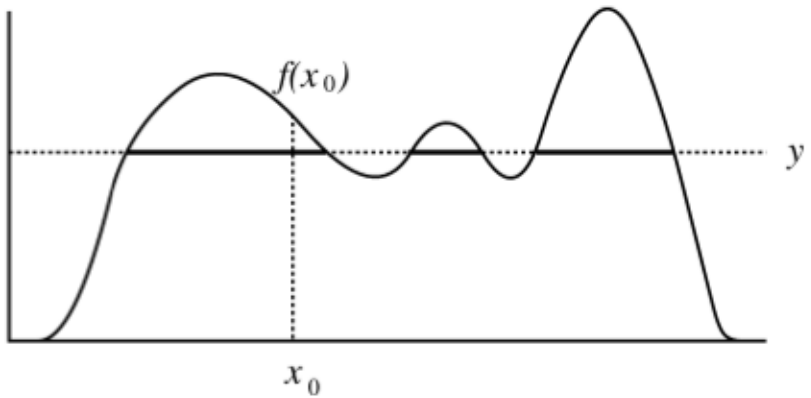
Slice Sampler

Slice Sampling idea

Sampling from $f(x)$ is equivalent to uniform sampling from
$$\mathcal{SG}(f) = \{(x, y) | 0 \leq y \leq f(x)\}$$
the subgraph of f

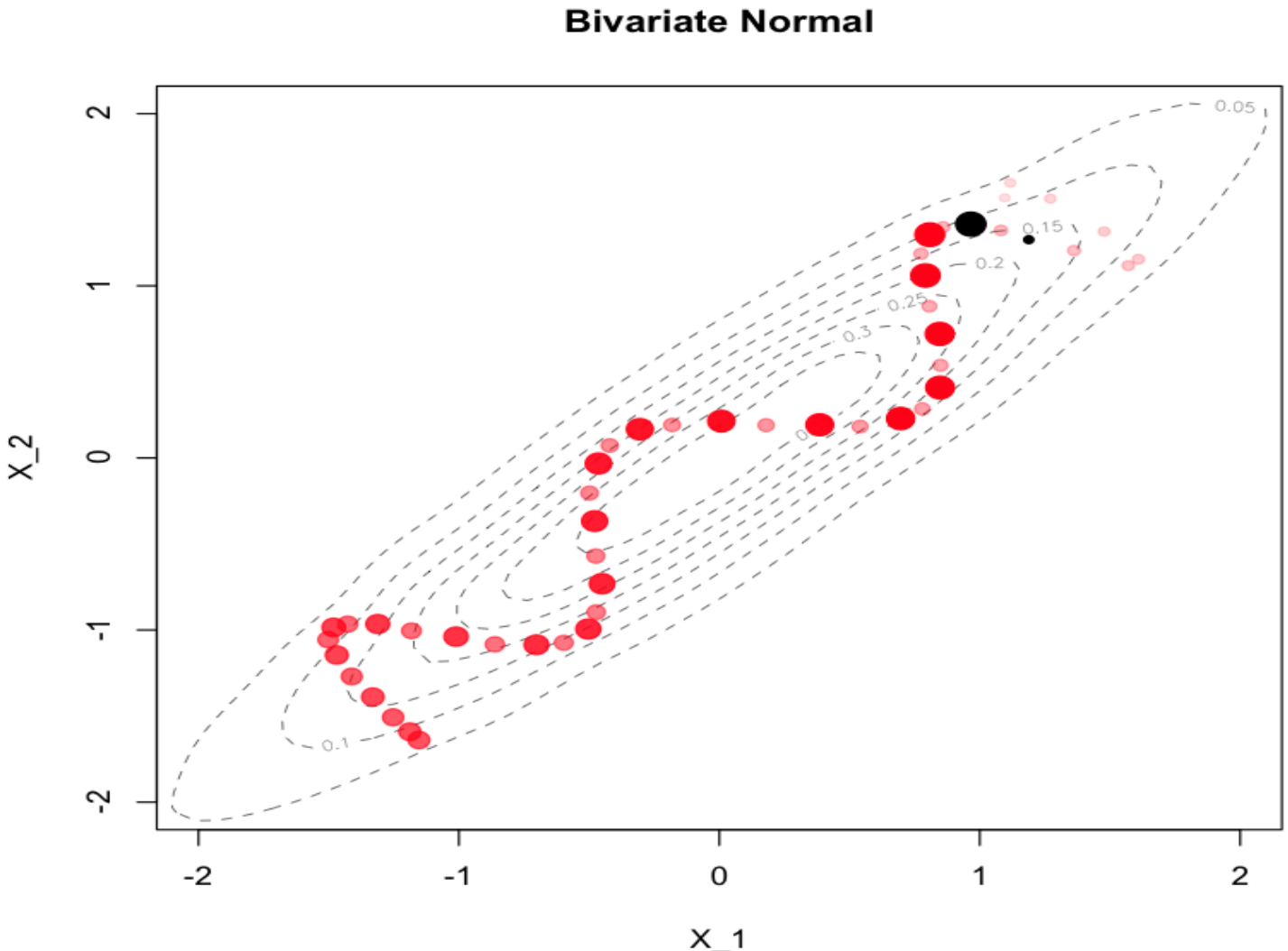
We simply make use of this by sampling from f with an auxiliary variable Gibbs sampling:

- $y|x \sim \mathcal{U}(0, f(x))$
- $x|y \sim \mathcal{U}_{S_y}$ where $S_y = \{x|y \leq f(x)\}$ is the slice



Outline

What we (may) have and want to avoid



Opacity and size grows with leapfrog steps made
In black start and end-point

Main idea

Define a criterion that helps us avoid these U-Turn, stopping when we simulated "long enough": instantaneous distance gain

$$C(q, q') = \frac{\partial}{\partial t}\frac{1}{2}(q'-q)^T(q'-q) = (q'-q)^T\frac{\partial}{\partial t}(q'-q) = (q'-q)^T p$$

Simulating until $C(q, q') < 0$ lead to a non-reversible MC, so the authors devised a different scheme.

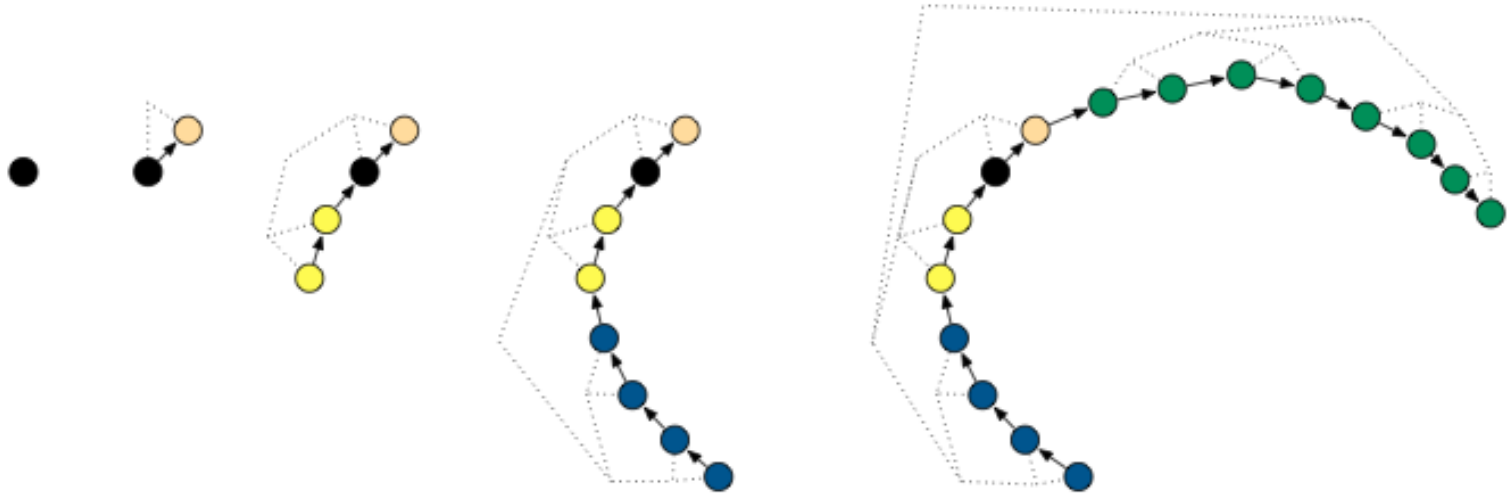
Main Idea

- NUTS augments the model with a slice variable u
- Add a finite set $\mathcal{C}$ of candidates for the update
- $\mathcal{C} \subseteq \mathcal{B}$ deterministically chosen,
  with $\mathcal{B}$ the set of all leapfrog steps

Main Idea

- NUTS augments the model with a slice variable u
- Add a finite set $\mathcal{C}$ of candidates for the update
- $\mathcal{C} \subseteq \mathcal{B}$ deterministically chosen,
  with $\mathcal{B}$ the set of all leapfrog steps

(At a high level) $\mathcal{B}$ is built by doubling and checking $C(q, q')$ on sub-trees



High level NUTS steps

1. resample momentum $p \sim \mathcal{N}(0, I)$
2. sample $u|q, p \sim \mathcal{U}[0, \exp(-H(q_t, p))]$
3. generate the proposal from $p(\mathcal{B}, \mathcal{C}|q_t, p, u, \varepsilon)$
4. sample $(q_{t+1}, p) \sim T(\cdot|q_t, p, \mathcal{C})$

$T(\cdot|q_t, p, \mathcal{C})$ s.t. leave the uniform distribution over $\mathcal{C}$ invariant
($\mathcal{C}$ contain (q', p') s.t. $u \leq \exp(-H(q', p'))$ and sat. reversibility)

Justify $T(q', p'|q_t, p, \mathcal{C})$

Conditions on $p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon)$:

- C.1: Elements in $\mathcal{C}$ chosen in a volume-preserving way
- C.2: $p((q, p) \in \mathcal{C}|q, p, u, \varepsilon) = 1$
- C.3: $p(u \leq exp(-H(q', p'))|(q', p') \in \mathcal{C}) = 1$
- C.4: If $(q, p) \in \mathcal{C}$ and $(q', p') \in \mathcal{C}$ then

$$p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon) = p(\mathcal{B}, \mathcal{C}|q', p', u, \varepsilon)$$

by them

$$\begin{aligned} p(q, p|u, \mathcal{B}, \mathcal{C}, \varepsilon) &\propto p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon)p(q, p|u) \\ &\propto p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon)\mathbb{I}_{\{u \leq \exp(-H(q', p'))\}} && C.1 \\ &\propto \mathbb{I}_{\mathcal{C}} && C.2 \& C.4 - C.3 \end{aligned}$$

$p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon)$ - Building $\mathcal{B}$ by doubling

Build $\mathcal{B}$ by repeatedly doubling a binary tree with (q,p) leaves.

- Chose a random "direction in time" $\nu_j \sim \mathcal{U}(\{-1, 1\})$
- Take 2^j leap(frog)s of size $\nu_j\varepsilon$ from
$$(q^-, p^-)\mathbb{I}_{\{-1\}}(\nu_j) + (q^+, p^+)\mathbb{I}_{\{+1\}}(\nu_j)$$
- Continue until a stopping rule is met

$p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon)$ - Building $\mathcal{B}$ by doubling

Build $\mathcal{B}$ by repeatedly doubling a binary tree with (q,p) leaves.

- Chose a random "direction in time" $\nu_j \sim \mathcal{U}(\{-1, 1\})$
- Take 2^j leap(frog)s of size $\nu_j\varepsilon$ from
$$(q^-, p^-)\mathbb{I}_{\{-1\}}(\nu_j) + (q^+, p^+)\mathbb{I}_{\{+1\}}(\nu_j)$$
- Continue until a stopping rule is met

Given the start (q, p) and ε : 2^j equi-probable height-j trees
Reconstructing a particular height-j tree from any leaf has prob 2^{-j}

$p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon)$ - Building $\mathcal{B}$ by doubling

Build $\mathcal{B}$ by repeatedly doubling a binary tree with (q,p) leaves.

- Chose a random "direction in time" $\nu_j \sim \mathcal{U}(\{-1, 1\})$
- Take 2^j leap(frog)s of size $\nu_j\varepsilon$ from
$$(q^-, p^-)\mathbb{I}_{\{-1\}}(\nu_j) + (q^+, p^+)\mathbb{I}_{\{+1\}}(\nu_j)$$
- Continue until a stopping rule is met

Given the start (q, p) and ε : 2^j equi-probable height-j trees
Reconstructing a particular height-j tree from any leaf has prob 2^{-j}

Possible stopping rule: at height j

- for one of the $2^j - 1$ subtrees

$$(q^+, q^-)^T p^- < 0 \qquad or \qquad (q^+, q^-)^T p^+ < 0$$

- the tree includes a leaf s.t. $\log(u) - H(q, p) > \Delta_{max}$

Satisfying Detailed-Balance

We've defined $p(\mathcal{B}|q, p, u, \varepsilon) \rightarrow$ deterministically select $\mathcal{C}$

Remember the conditions:

C.1: Elements in $\mathcal{C}$ chosen in a volume-preserving way

C.2: $p((q, p) \in \mathcal{C}|q, p, u, \varepsilon) = 1$

C.3: $p(u \leq exp(-H(q', p'))|(q', p') \in \mathcal{C}) = 1$

C.4: If $(q, p) \in \mathcal{C}$ and $(q', p') \in \mathcal{C}$ then

$$p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon) = p(\mathcal{B}, \mathcal{C}|q', p', u, \varepsilon)$$

Satisfying Detailed-Balance

We've defined $p(\mathcal{B}|q, p, u, \varepsilon) \rightarrow$ deterministically select $\mathcal{C}$

Remember the conditions:

C.1: ✓Satisfied because of the leapfrog

C.2: $p((q, p) \in \mathcal{C}|q, p, u, \varepsilon) = 1$

C.3: $p(u \leq exp(-H(q', p'))|(q', p') \in \mathcal{C}) = 1$

C.4: If $(q, p) \in \mathcal{C}$ and $(q', p') \in \mathcal{C}$ then

$$p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon) = p(\mathcal{B}, \mathcal{C}|q', p', u, \varepsilon)$$

Satisfying Detailed-Balance

We've defined $p(\mathcal{B}|q, p, u, \varepsilon) \rightarrow$ deterministically select $\mathcal{C}$

Remember the conditions:

C.1: ✓Satisfied because of the leapfrog

C.2: ✓Satisfied if $\mathcal{C}$ includes the initial state

C.3: $p(u \leq exp(-H(q', p'))|(q', p') \in \mathcal{C}) = 1$

C.4: If $(q, p) \in \mathcal{C}$ and $(q', p') \in \mathcal{C}$ then

$$p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon) = p(\mathcal{B}, \mathcal{C}|q', p', u, \varepsilon)$$

Satisfying Detailed-Balance

We've defined $p(\mathcal{B}|q, p, u, \varepsilon) \rightarrow$ deterministically select $\mathcal{C}$

Remember the conditions:

C.1: ✓Satisfied because of the leapfrog

C.2: ✓Satisfied if $\mathcal{C}$ includes the initial state

C.3: ✓Satisfied if we exclude points outside the slice u

C.4: If $(q, p) \in \mathcal{C}$ and $(q', p') \in \mathcal{C}$ then

$$p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon) = p(\mathcal{B}, \mathcal{C}|q', p', u, \varepsilon)$$

Satisfying Detailed-Balance

We've defined $p(\mathcal{B}|q, p, u, \varepsilon) \rightarrow$ deterministically select $\mathcal{C}$

Remember the conditions:

C.1: ✓Satisfied because of the leapfrog

C.2: ✓Satisfied if $\mathcal{C}$ includes the initial state

C.3: ✓Satisfied if we exclude points outside the slice u

C.4: As long as $\mathcal{C}$ given $\mathcal{B}$ is deterministic

$$p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon) = 2^{-j} \qquad \textit{or} \qquad p(\mathcal{B}, \mathcal{C}|q, p, u, \varepsilon) = 0$$

Satisfying Detailed-Balance

We've defined $p(\mathcal{B}|q, p, u, \varepsilon) \rightarrow$ deterministically select $\mathcal{C}$

Remember the conditions:

C.1: ✓Satisfied because of the leapfrog

C.2: ✓Satisfied if $\mathcal{C}$ includes the initial state

C.3: ✓Satisfied if we exclude points outside the slice u

C.4: ✓Satisfied if we exclude states that couldn't generate $\mathcal{B}$

Satisfying Detailed-Balance

We've defined $p(\mathcal{B}|q, p, u, \varepsilon) \rightarrow$ deterministically select $\mathcal{C}$

Remember the conditions:

C.1: ✓Satisfied because of the leapfrog

C.2: ✓Satisfied if $\mathcal{C}$ includes the initial state

C.3: ✓Satisfied if we exclude points outside the slice u

C.4: ✓Satisfied if we exclude states that couldn't generate $\mathcal{B}$

Finally we select (q',p') at random from $\mathcal{C}$

Algorithm 2 Naive No-U-Turn Sampler

Given θ^0, ϵ, $\mathcal{L}$, M:
for $m = 1$ to M **do**
  Resample $r^0 \sim \mathcal{N}(0, I)$.
  Resample $u \sim \text{Uniform}([0, \exp\{\mathcal{L}(\theta^{m-1} - \frac{1}{2} r^0 \cdot r^0\}])$
  Initialize $\theta^- = \theta^{m-1}$, $\theta^+ = \theta^{m-1}$, $r^- = r^0$, $r^+ = r^0$, $j = 0$, $\mathcal{C} = \{(\theta^{m-1}, r^0)\}$, $s = 1$.
  while $s = 1$ **do**
    Choose a direction $v_j \sim \text{Uniform}(\{-1, 1\})$.
    if $v_j = -1$ **then**
      $\theta^-, r^-, -, -, \mathcal{C}', s' \leftarrow \text{BuildTree}(\theta^-, r^-, u, v_j, j, \epsilon)$.
    else
      $-, -, \theta^+, r^+, \mathcal{C}', s' \leftarrow \text{BuildTree}(\theta^+, r^+, u, v_j, j, \epsilon)$.
    end if
    if $s' = 1$ **then**
      $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{C}'$.
    end if
    $s \leftarrow s' \mathbb{I}[(\theta^+ - \theta^-) \cdot r^- \geq 0] \mathbb{I}[(\theta^+ - \theta^-) \cdot r^+ \geq 0]$.
    $j \leftarrow j + 1$.
  end while
  Sample θ^m, r uniformly at random from $\mathcal{C}$.
end for

function BuildTree$(\theta, r, u, v, j, \epsilon)$
if $j = 0$ **then**
Base case—take one leapfrog step in the direction v.
$\theta', r' \leftarrow \text{Leapfrog}(\theta, r, v\epsilon)$.
$\mathcal{C}' \leftarrow \begin{cases} \{(\theta', r')\} & \text{if } u \leq \exp\{\mathcal{L}(\theta') - \frac{1}{2} r' \cdot r'\} \\ \emptyset & \text{else} \end{cases}$
$s' \leftarrow \mathbb{I}[u < \exp\{\Delta_{\max} + \mathcal{L}(\theta') - \frac{1}{2} r' \cdot r'\}]$.
return $\theta', r', \theta', r', \mathcal{C}', s'$.
else
Recursion—build the left and right subtrees.
$\theta^-, r^-, \theta^+, r^+, \mathcal{C}', s' \leftarrow \text{BuildTree}(\theta, r, u, v, j-1, \epsilon)$.
if $v = -1$ **then**
$\theta^-, r^-, -, -, \mathcal{C}'', s'' \leftarrow \text{BuildTree}(\theta^-, r^-, u, v, j-1, \epsilon)$.
else
$-, -, \theta^+, r^+, \mathcal{C}'', s'' \leftarrow \text{BuildTree}(\theta^+, r^+, u, v, j-1, \epsilon)$.
end if
$s' \leftarrow s' s'' \mathbb{I}[(\theta^+ - \theta^-) \cdot r^- \geq 0] \mathbb{I}[(\theta^+ - \theta^-) \cdot r^+ \geq 0]$.
$\mathcal{C}' \leftarrow \mathcal{C}' \cup \mathcal{C}''$.
return $\theta^-, r^-, \theta^+, r^+, \mathcal{C}', s'$.
end if

Efficient NUTS

The computational cost per-leapfrog step is comparable with HMC (just $2^{j+1} - 2$ more inner products)
However:

- it requires to store 2^j position-momentum states
- long jumps are not guaranteed
- waste time if a stopping criterion is met during doubling

Efficient NUTS

The authors address these issues as follow:

- waste time if a stopping criterion is met during doubling

as soon as a stopping rule is met *break out of the loop*

Efficient NUTS

The authors address these issues as follow:

- long jumps are not guaranteed

Consider the kernel:

$$T(w'|w,\mathcal{C}) = \begin{cases} \frac{\mathbb{I}[w' \in \mathcal{C}^{new}]}{|\mathcal{C}^{new}|} & \text{if } |\mathcal{C}^{new}| > |\mathcal{C}^{old}|, \\ \frac{\mathbb{I}[w' \in \mathcal{C}^{new}]}{|\mathcal{C}^{new}|} \frac{|\mathcal{C}^{new}|}{|\mathcal{C}^{old}|} + (1 - \frac{|\mathcal{C}^{new}|}{|\mathcal{C}^{old}|})\mathbb{I}[w' = w] & \text{if } |\mathcal{C}^{new}| \leq |\mathcal{C}^{old}| \end{cases},$$

where w is short for (q,p) and let $\mathcal{C}^{new}$ and $\mathcal{C}^{old}$ be respectively the set $\mathcal{C}'$ introduced during the final iteration and the older elements already in $\mathcal{C}$.

Efficient NUTS

The authors address these issues as follow:

- it requires to store 2^j position-momentum states
- long jumps are not guaranteed

Consider the kernel:

$$T(w'|w,\mathcal{C}) = \begin{cases} \frac{\mathbb{I}[w' \in \mathcal{C}^{new}]}{|\mathcal{C}^{new}|} & \text{if } |\mathcal{C}^{new}| > |\mathcal{C}^{old}|, \\ \frac{\mathbb{I}[w' \in \mathcal{C}^{new}]}{|\mathcal{C}^{new}|}\frac{|\mathcal{C}^{new}|}{|\mathcal{C}^{old}|} + (1 - \frac{|\mathcal{C}^{new}|}{|\mathcal{C}^{old}|})\mathbb{I}[w' = w] & \text{if } |\mathcal{C}^{new}| \leq |\mathcal{C}^{old}| \end{cases},$$

where w is short for (q,p) and let $\mathcal{C}^{new}$ and $\mathcal{C}^{old}$ be respectively the set $\mathcal{C}'$ introduced during the final iteration and the older elements already in $\mathcal{C}$.

Iteratively applying the above after *every* doubling moreover require that we store only $O(j)$ position (and sizes) rather than $O(2^j)$

Outline

Adaptive MCMC

Classic adaptive MCMC idea:
stochastic optimization on the parameter with vanishing adaptation:

$$\theta_{t+1} \leftarrow \theta_t - \eta_t H_t$$

where $\mathbb{R} \ni \eta_t \to 0$ with the "correct" pace and
$H_t = \delta - \alpha_t$ defines a characteristic of interest of the chain.

Example

Consider the case of a random-walk MH with $x'|x_t \sim \mathcal{N}(0, \theta_t)$ adapted on the acceptance rate (α_t) in order to get to the desired optimum $\delta = 0.234$

Double Averaging

This idea has however (in particoular) an intrinsic flaw:

- the diminishing step sizes η_t give more weight to the early iterations

Double Averaging

This idea has however (in particoular) an intrinsic flaw:

- the diminishing step sizes η_t give more weight to the early iterations

The authors rely then on the following scheme[3]:

$$\varepsilon_{t+1} = \mu - \frac{1}{\gamma}\frac{\sqrt{t}}{t+t_0}\sum_{i=1}^{t} H_i \quad \tilde{\varepsilon}_{t+1} = \eta_t \varepsilon_{t+1} + (1-\eta_t)\tilde{\varepsilon}_t$$

adaptation	sampling
γ shrinkage $\to \mu$	$\eta_t = t^{-k}$
t_0 early iteration	$k \in (0.5, 1]$

[3] introduced by Nesterov (2009) for stochastic convex optimization

Select the criterion

The authors advice to use $H_t = \delta - \alpha_t$ with $\delta = 0.65$ and

$$\alpha_t^{HMC} = \min\left[1, \frac{P(q', -p')}{P(q_t, -p_t)}\right]$$

$$\alpha_t^{NUTS} = \frac{1}{|\mathcal{B}_t^{final}|} \sum_{(q_t', p_t') \in \mathcal{B}_t^{final}} \min\left[1, \frac{P(q_t', -p_t')}{P(q_{t-1}, -p_t)}\right]$$

Alternative: random ε

Periodicity is still a problem and the "optimal" ε may differ in different region of the target! (e.g. mixtures)
The solution might be to randomize ε around some ε_0

Example

$$\varepsilon \sim \mathcal{E}(\varepsilon_0) \qquad \varepsilon \sim \mathcal{U}(c_1\varepsilon_0, c_2\varepsilon_0)$$

$$c_1 < 1, \quad c_2 > 1$$

Alternative: random ε

Periodicity is still a problem and the "optimal" ε may differ in different region of the target! (e.g. mixtures)
The solution might be to randomize ε around some ε_0

Helpfull especially for HMC, care is needed for NUTS as L is usually inversely proportional to ε

What said about adaptation can be combined using $\varepsilon_0 = \varepsilon_t$

Outline

Numerical Examples

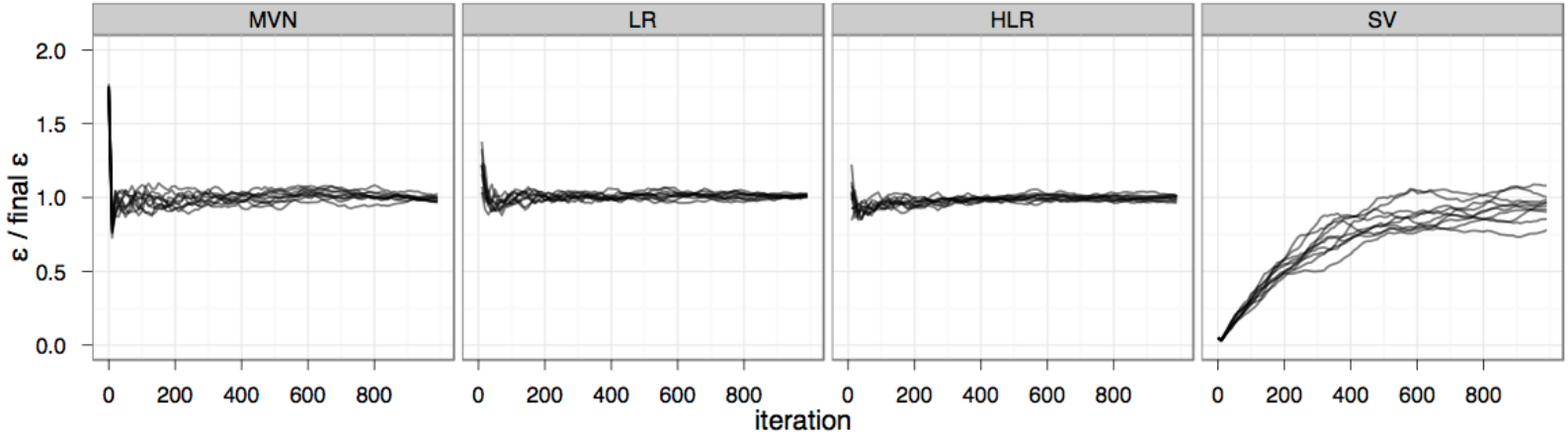
"It at least as good as an 'optimally' tuned HMC."
Tested on:

1. **(MVN)** : 250-d Normal whose precision matrix was sampled from Wishart with identity scale matrix and 250 df
2. **(LR)** : (24+1)-dim regression coefficient distribution on the German credit data with weak priors
3. **(HLR)** : same dataset - exponentially distributed variance hyper parameters and included two-way interactions $\rightarrow$ (300+1)*2-d predictors
4. **(SV)** : 3000 days of returns from S&P 500 index $\rightarrow$ 3001-d target following this scheme:

$$\tau \sim \mathcal{E}(100); \quad \nu \sim \mathcal{E}(100); \quad s_1 \sim \mathcal{E}(100);$$

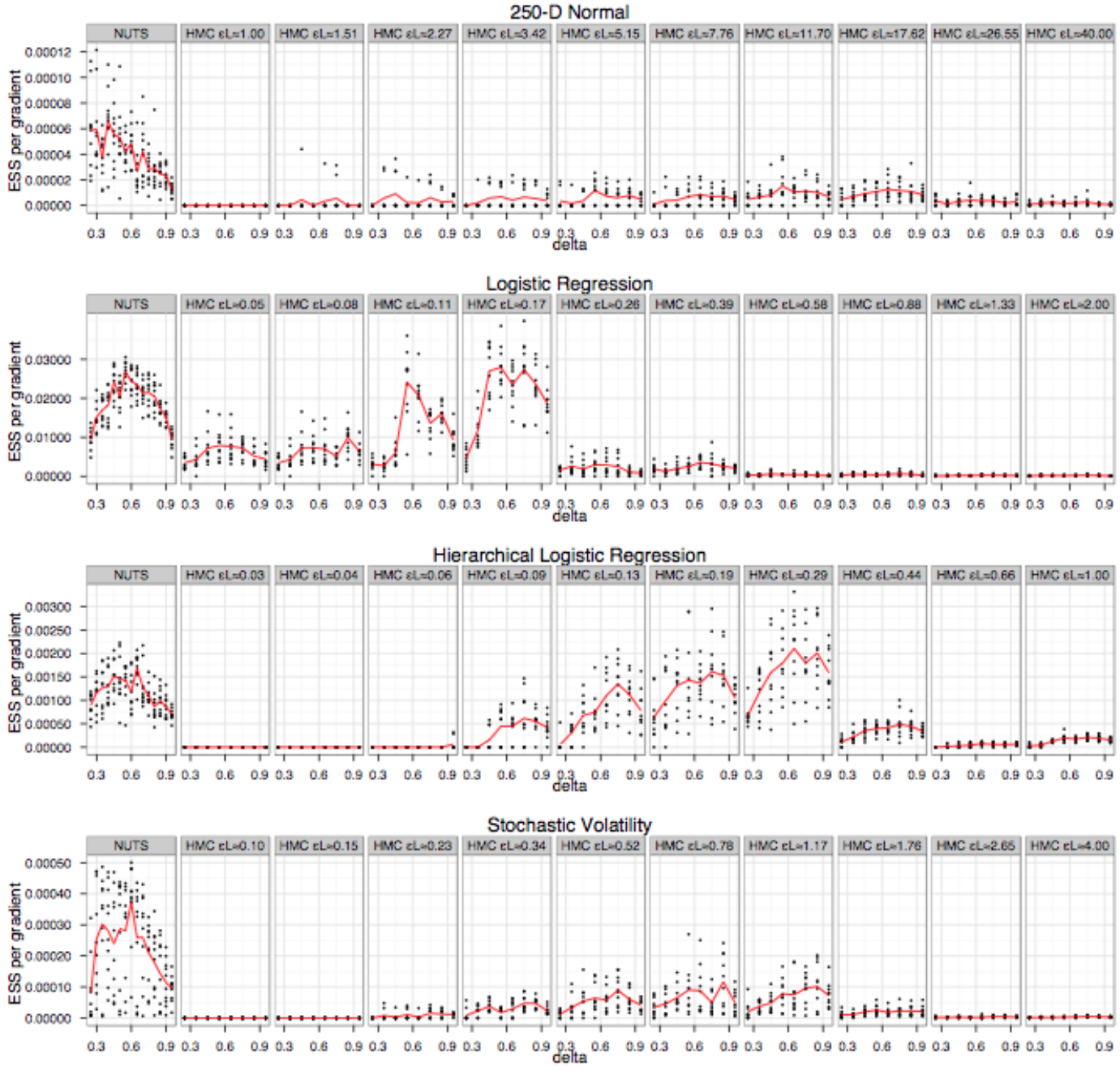
$$\log s_i \sim \mathcal{N}(\log s_{i-1}, \tau^{-}1); \quad \frac{\log y_i - \log y_{i-1}}{s_i} \sim t_\nu.$$

ε and L in NUTS



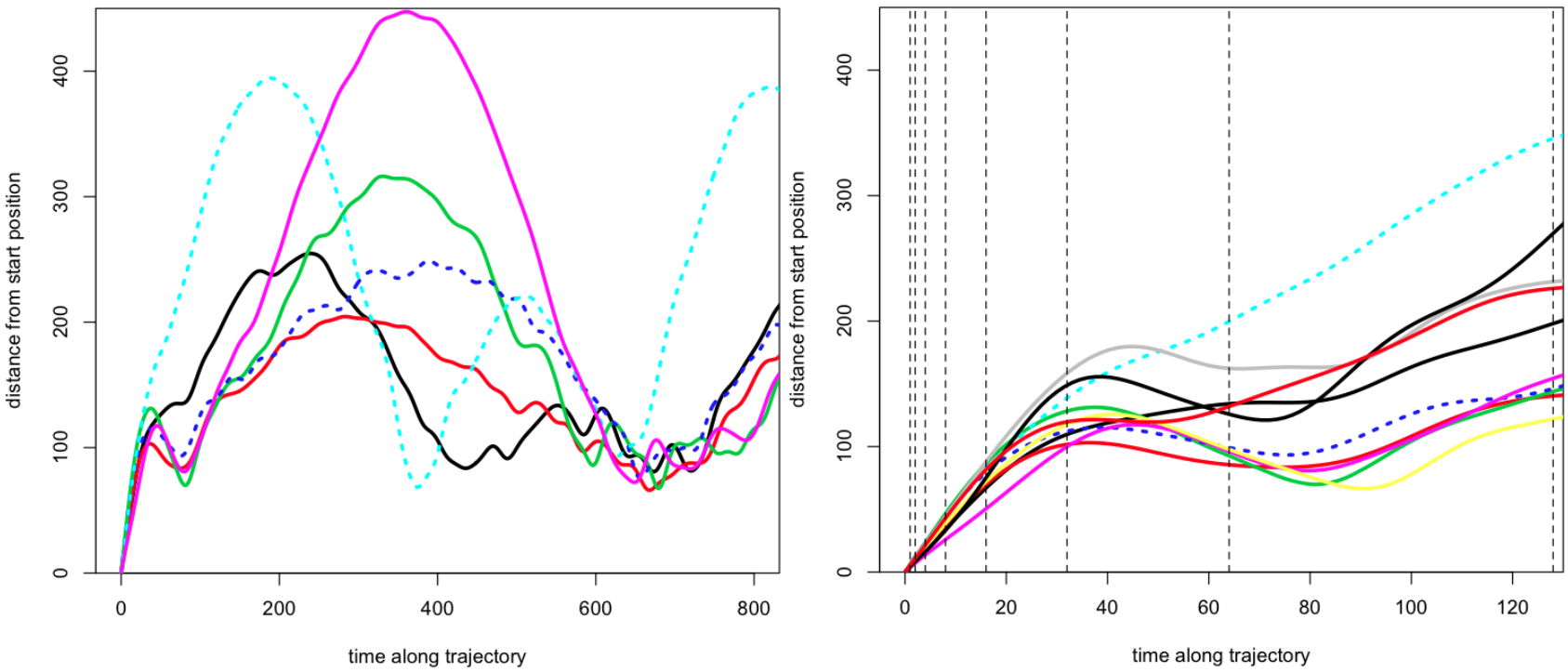
Also the dual averaging usually does a good job of coercing the statistic H to its desired value.

Most of the trajectory lengths are integer powers of two, indicating that the U-turn criterion is usually satisfied only after a doubling is completed, which is desirable since it means that we only occasionally have to throw out entire half-trajectories to satisfy DB.


250-D Normal
NUTS
HMC εL=1.00
HMC εL=1.51
HMC εL=2.27
HMC εL=3.42
HMC εL=5.15
HMC εL=7.76
HMC εL=11.70
HMC εL=17.62
HMC εL=26.55
HMC εL=40.00
ESS per gradient
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
0.3 0.6 0.9
delta
Logistic Regression
NUTS
HMC εL=0.05
HMC εL=0.08
HMC εL=0.11
HMC εL=0.17
HMC εL=0.26
HMC εL=0.39
HMC εL=0.58
HMC εL=0.88
HMC εL=1.33
HMC εL=2.00
ESS per gradient
0.03000
0.02000
0.01000
0.00000
0.3 0.6 0.9
delta
Hierarchical Logistic Regression
NUTS
HMC εL=0.03
HMC εL=0.04
HMC εL=0.06
HMC εL=0.09
HMC εL=0.13
HMC εL=0.19
HMC εL=0.29
HMC εL=0.44
HMC εL=0.66
HMC εL=1.00
ESS per gradient
0.00300
0.00250
0.00200
0.00150
0.00100
0.00050
0.00000
0.3 0.6 0.9
delta
Stochastic Volatility
NUTS
HMC εL=0.10
HMC εL=0.15
HMC εL=0.23
HMC εL=0.34
HMC εL=0.52
HMC εL=0.78
HMC εL=1.17
HMC εL=1.76
HMC εL=2.65
HMC εL=4.00
ESS per gradient
0.00050
0.00040
0.00030
0.00020
0.00010
0.00000
0.3 0.6 0.9
delta

Multivariate Normal

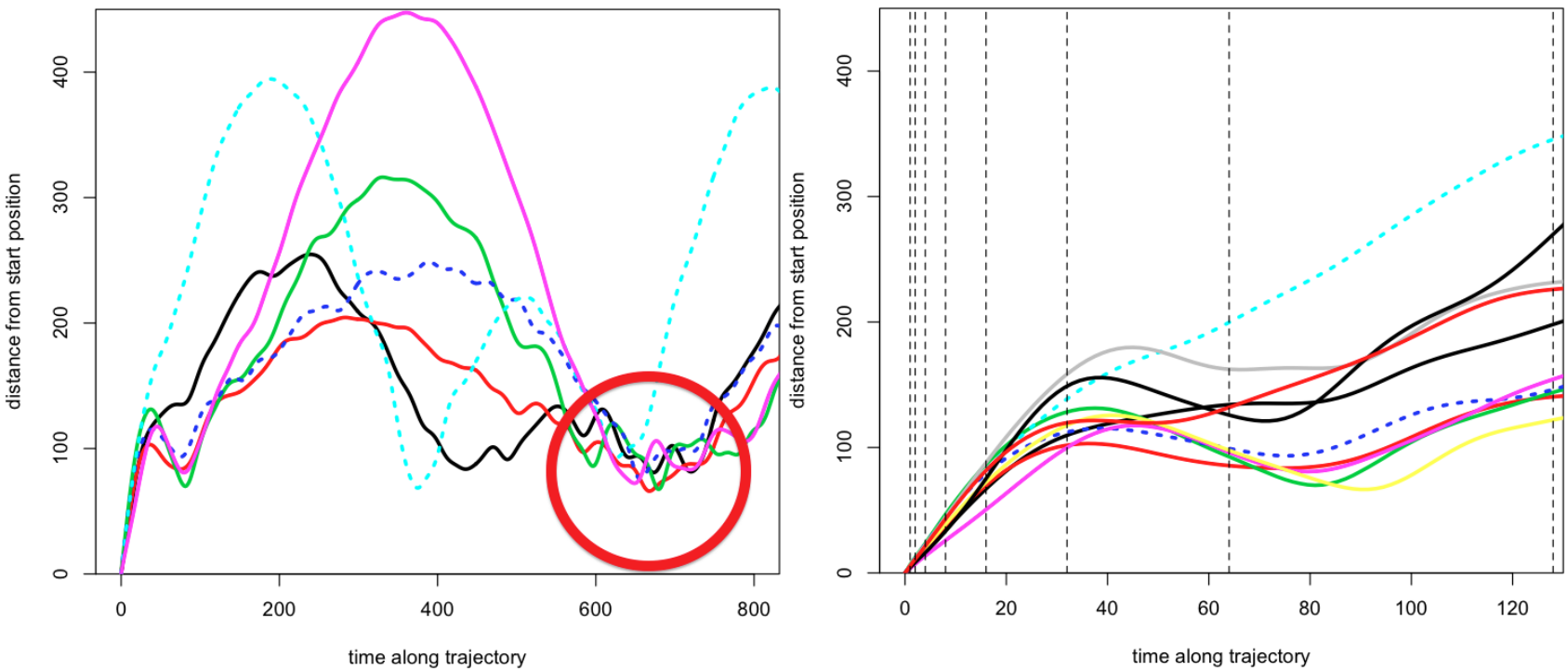
Highly correlated, 30-dim Multivariate Normal



To get the full advantage of HMC, the trajectory has to be long enough (but not much longer) that in the least constrained direction the end-point is distant from the start point.

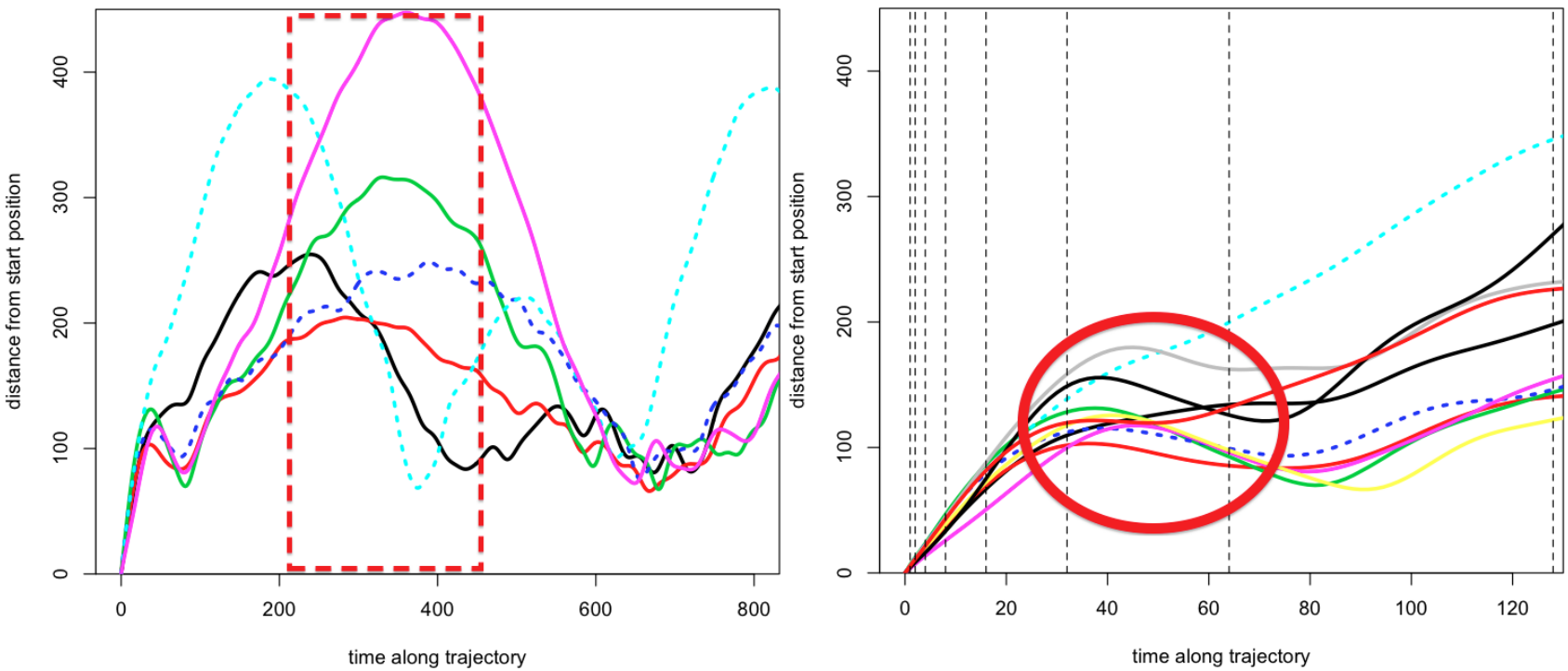
Multivariate Normal

Highly correlated, 30-dim Multivariate Normal



Multivariate Normal

Highly correlated, 30-dim Multivariate Normal

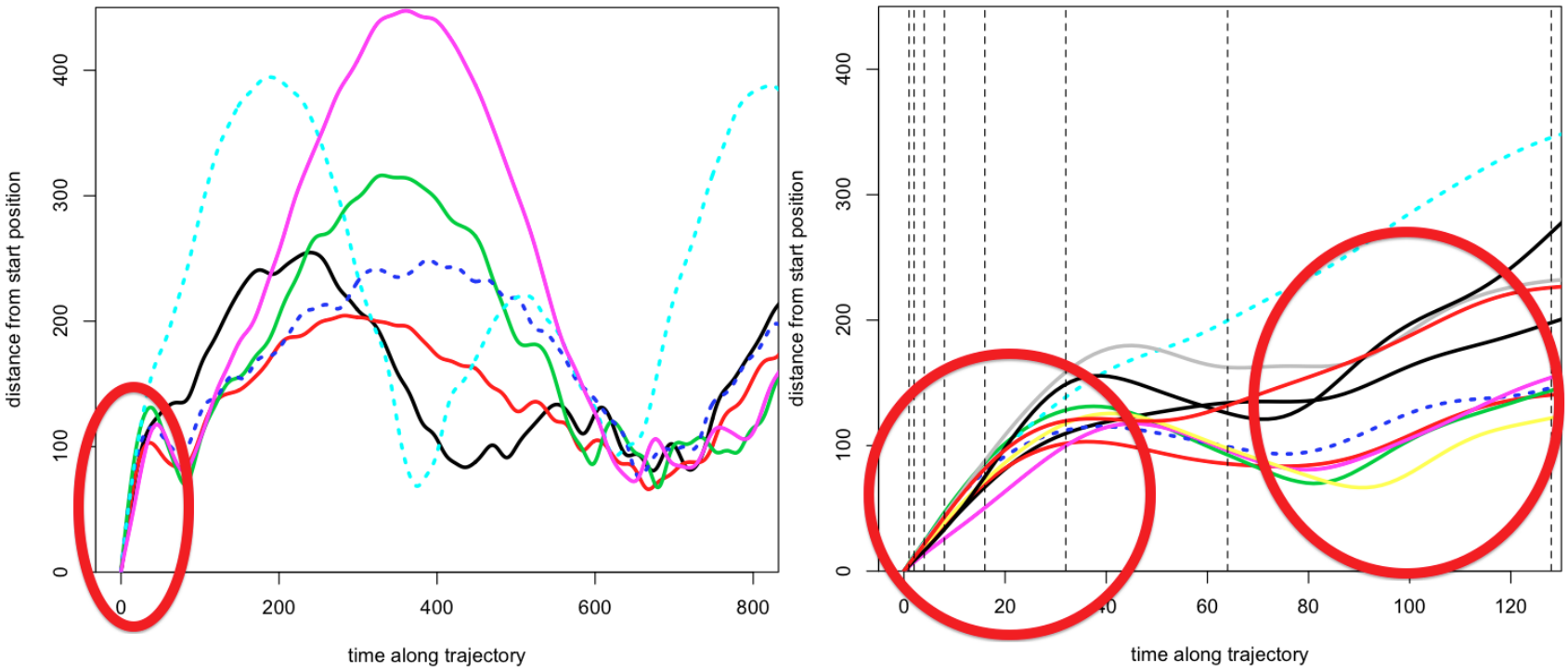


Trajectory reverse direction before the least constrained direction has been explored.

This can produce a U-Turn when the trajectory is much shorter than is optimal.

Multivariate Normal

Highly correlated, 30-dim Multivariate Normal

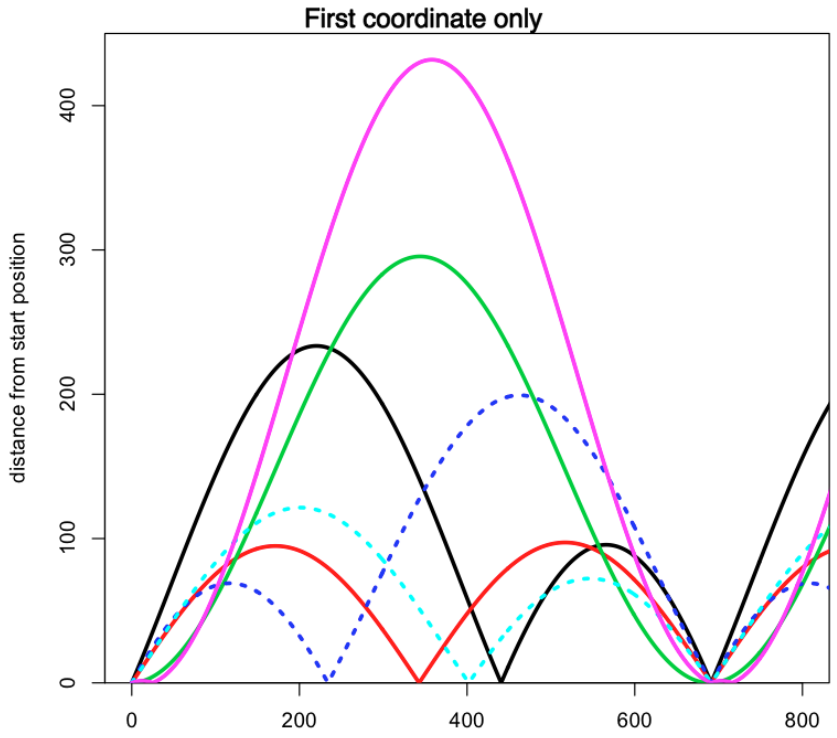


Fortunately, too-short trajectories are cheap relative to long-enough trajectories!

but it does not help in avoiding RW

Multivariate Normal

One way to overcome the problem could be to check for U-Turn only in the *least constrained*



But usually such an information is not known a-priori.
Maybe after a trial run to discover the magnitude order of the variables?

someone said periodicity?

Outline

Non-Uniform weighting system

Qin and Liu (2001) propose a similar (to Neal's) weighting procedure:

- Generate the whole trajectory from $(q_0, p_0) = (q, p)$ to (q_L, p_L)
- Select a point in the "accept window" $[(q_{L-W+1}, p_{L-W+1}), \ldots, (q_L, p_L)]$, say (q_{L-k}, p_{L-k})
- equivalent to select $k \sim \mathcal{U}(0, W)$

Non-Uniform weighting system

Qin and Liu (2001) propose a similar (to Neal's) weighting procedure:

- Generate the whole trajectory from (q_0, p_0) to (q_L, p_L)
- Select a point (q_{L-k}, p_{L-k})
- Take k step backward from (q_0, p_0) to (q_{-k}, p_{-k})
- Reject window : $[(q_{-k}, p_{-k}), \ldots, (q_{W-k-1}, p_{W-k-1})]$

Non-Uniform weighting system

Qin and Liu (2001) propose a similar (to Neal's) weighting procedure:

- Generate the whole trajectory from (q_0, p_0) to (q_L, p_L)
- Select a point (q_{L-k}, p_{L-k})
- Take k step backward from (q_0, p_0) to (q_{-k}, p_{-k})
- Accept (q_{L-k}, p_{L-k}) with probability

$$\min \left[1, \frac{\sum_{i=1}^{W} w_i P(q_{L-W+i}, -p_{L-W+i})}{\sum_{i=1}^{W} w_i P(q_{i-k-1}, -p_{i-k-1})} \right]$$

Non-Uniform weighting system

Qin and Liu (2001) propose a similar (to Neal's) weighting procedure:

- Accept (q_{L-k}, p_{L-k}) with probability

$$\min \left[1, \frac{\sum_{i=1}^{W} w_i P(q_{L-W+i}, -p_{L-W+i})}{\sum_{i=1}^{W} w_i P(q_{i-k-1}, -p_{i-k-1})} \right]$$

- With uniform weights $w_i = 1/W \ \forall i$
- Other weights may favor states further from the start!

Riemann Manifold HMC

- In statistical modeling the parameter space is a manifold
- $d(p(y|\theta), p(y|\theta + \delta\theta))$ can be defined as $\delta\theta^T G(\theta)\delta\theta$
  $G(\theta)$ is the expected Fisher information matrix [Rao (1945)]
- $G(\theta)$ defines a position-specific Riemann metric.

Girolami & Calderhead (2011) used this to tune $\mathcal{K}(p)$ using local 2^{nd}-order derivative information encoded in $M = G(q)$

Riemann Manifold HMC

Girolami & Calderhead (2011) used this to tune $\mathcal{K}(p)$ using local 2^{nd}-order derivative information encoded in $M = G(q)$

The deterministic proposal is guided not only by the gradient of the target density but also exploits local geometric structure.
Possible optimality : paths that are produced by the solution of Hamiltoninan equations follow the geodesics on the manifold.

Riemann Manifold HMC

Girolami & Calderhead (2011) used this to tune $\mathcal{K}(p)$ using local 2^{nd}-order derivative information encoded in $M = G(q)$

Practically:

$p|q \sim \mathcal{N}(0, G(q))$ resolving the scaling issues in HMC

- tuning of ε less critical
- non separable Hamiltonian

$$H(q,p) = U(q) + \frac{1}{2} log\left\{(2\pi)^D |G(q)|\right\} + \frac{1}{2} p^T G(q)^{-1} p$$

- need for an (expensive) implicit integrator (generalized leapfrog + fixed point iterations)

RMHMC & Related

"In some cases, the computational overhead for solving implicit equations undermines RMHMCs benefits" . [Lan et al. (2012)]

Lagrangian Dynamics (RMLMC)

Replaces momentum p (mass$\times$velocity) in HMC by velocity v
volume correction through the Jacobian

- semi-implicit integrator for "Hamiltonian" dynamics
- different fully explicit integrator for Lagrangian dynamics
  two extra matrix inversions to update v

Adaptively Updated (AUHMC)

Replace $G(q)$ with $M(q, q') = \frac{1}{2}[G(q) + G(q')]$

- fully explicit leapfrog integrator ($M(q, q')$ constant through trajectory)
- less local-adaptive (especially for long trajectories)
- is it really reversible?

Split Hamiltonian

Variations on HMC obtained by using discretizations of Hamiltonian dynamics, splitting H into:

$$H(q,p) = H_1(q,p) + H_2(q,p) + \cdots + H_K(q,p)$$

This may allows much of the movement to be done at low cost.

- log of $U(q)$ as the log of a Gaussian plus a second term

  quadratic forms allow for explicit solution

- H_1 and its gradient can be eval. quickly, with only a slowly-varying H_2 requiring costly computations. e.g. splitting data

Split Hamiltonian

- log of $U(q)$ as the log of a Gaussian plus a second term
  quadratic forms allow for explicit solution
- H_1 and its gradient can be eval. quickly, with only a slowly-varying H_2 requiring costly computations. e.g. splitting data

Algorithm 1: Leapfrog for split Hamiltonian Monte Carlo with a partial analytic solution.

$R \leftarrow \Gamma e^{D\varepsilon}\Gamma^{-1}$

Sample initial values for p from $N(0, I)$

for $\ell = 1$ to L **do**

$p \leftarrow p - (\varepsilon/2)\frac{\partial U_1}{\partial q}$

$q^* \leftarrow q - \hat{q}$

$X_0 \leftarrow (q^*, p)$

$(q^*, p) \leftarrow RX_0$

$q \leftarrow q^* + \hat{q}$

$p \leftarrow p - (\varepsilon/2)\frac{\partial U_1}{\partial q}$

end for

Algorithm 2: Nested leapfrog for split Hamiltonian Monte Carlo with splitting of data.

Sample initial values for p from $N(0, I)$

for $\ell = 1$ to L **do**

$p \leftarrow p - (\varepsilon/2)\frac{\partial U_1}{\partial q}$

for $m = 1$ to M **do**

$p \leftarrow p - (\varepsilon/2M)\frac{\partial U_0}{\partial q}$

$q \leftarrow q + (\varepsilon/M)p$

$p \leftarrow p - (\varepsilon/2M)\frac{\partial U_0}{\partial q}$

end for

$p \leftarrow p - (\varepsilon/2)\frac{\partial U_1}{\partial q}$

end for

split-wRMLNUTS & co.

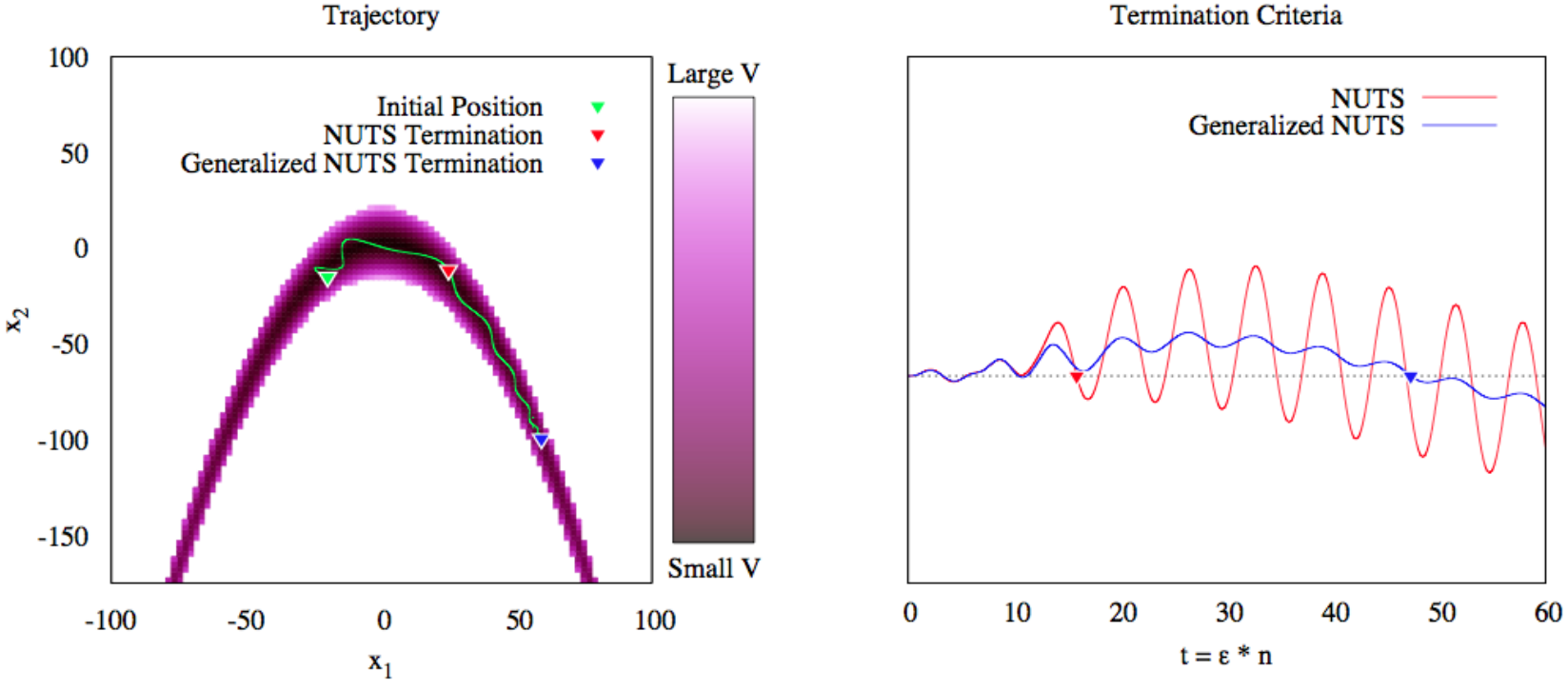
- NUTS itself provide a nice automatic tune of HMC
- very few citation to their work!
- integration with RMHMC (just out!) may overcome ε-related problems and provide better criteria for the stopping rule!

HMC may not be the definitive sampler, but is definitely equally useful and unknown.
It seems the direction taken may finally overcome the difficulties connected with its use and spread.

Betancourt (2013b)

- RMHMC has smoother movements on the surface but..
- .. $(q' - q)^T p$ has no more a meaning

The paper address this issue by generalizing the stopping rule to other M mass matrices.



References I

Introduction:

- M.D. Hoffmann & A. Gelman (2011) - *The No-U-Turn Sampler*
- R.M. Neal (2011) - *MCMC using Hamiltonian dynamics* in *Handbook of Markov Chain Monte Carlo*
- Z. S. Qin, and J.S. Liu (2001) - *Multipoint Metropolis method with application to hybrid Monte Carlo*
- R.M. Neal (2003) - *Slice Sampling*

References II

Further Readings:

- M. Girolami, B. Calderhead (2011) - *Riemann manifold Langevin and Hamiltonian Monte Carlo methods*
- Z. Wang, S. Mohamed, N. de Freitas (2013) - *Adaptive Hamiltonian and Riemann Manifold Monte Carlo Samplers*
- S. Lan, V. Stathopoulos, B. Shahbaba, M. Girolami (2012) - *Lagrangian Dynamical Monte Carlo*
- M. Burda, JM. Maheu (2013) - *Bayesian Adaptively Updated Hamiltonian Monte Carlo with an Application to High-Dimensional BEKK GARCH Models*
- Michael Betancourt (2013) - *Generalizing the No-U-Turn Sampler to Riemannian Manifolds*