No-U-Turn Sampler :
Adaptively Setting Path Lengths in HMC

김성민

서울대학교
통계학과, 베이즈통계 연구실

2021. 03. 18

CONTENTS

CONTENTS

HMC

- $\theta(t)$: location at t
- $r(t)$: momentum at t
- $U(\theta)$: potential energy

$$U(\theta) = -log(f(\theta))$$

- $K(r)$: kinetic energy

$$K(r) = \frac{r^T r}{2}$$

- $H(\theta, r) = U(\theta) + K(r)$

Leapfrog Method

- Hamilton's equation

$$\frac{\mathrm{d}r}{\mathrm{d}t} = -\frac{\partial H}{\partial \theta} = -\frac{\partial U}{\partial \theta}$$
$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = \frac{\partial H}{\partial r} = \frac{\partial K}{\partial r}$$

- Leapfrog integrator

$$r(t+\epsilon/2) = r(t) - (\epsilon/2)\frac{\partial U}{\partial \theta}(\theta(t))$$
$$\theta(t+\epsilon) = \theta(t) + \epsilon\frac{\partial K}{\partial \theta}(r(t+\epsilon/2))$$
$$r(t+\epsilon) = r(t+\epsilon/2) - (\epsilon/2)\frac{\partial U}{\partial \theta}(\theta(t+\epsilon))$$

Algorithm

Algorithm 1 Hamiltonian Monte Carlo

Given θ^0, ϵ, L, $\mathcal{L}, M$:
for $m = 1$ to M **do**
  Sample $r^0 \sim \mathcal{N}(0, I)$.
  Set $\theta^m \leftarrow \theta^{m-1}, \tilde{\theta} \leftarrow \theta^{m-1}, \tilde{r} \leftarrow r^0$.
  for $i = 1$ to L **do**
    Set $\tilde{\theta}, \tilde{r} \leftarrow \text{Leapfrog}(\tilde{\theta}, \tilde{r}, \epsilon)$.
  end for
  With probability $\alpha = \min\left\{1, \frac{\exp\{\mathcal{L}(\tilde{\theta}) - \frac{1}{2}\tilde{r}\cdot\tilde{r}\}}{\exp\{\mathcal{L}(\theta^{m-1}) - \frac{1}{2}r^0\cdot r^0\}}\right\}$, set $\theta^m \leftarrow \tilde{\theta}$, $r^m \leftarrow -\tilde{r}$.
end for

function Leapfrog(θ, r, ϵ)
Set $\tilde{r} \leftarrow r + (\epsilon/2)\nabla_\theta \mathcal{L}(\theta)$.
Set $\tilde{\theta} \leftarrow \theta + \epsilon\tilde{r}$.
Set $\tilde{r} \leftarrow \tilde{r} + (\epsilon/2)\nabla_\theta \mathcal{L}(\tilde{\theta})$.
return $\tilde{\theta}, \tilde{r}$.

Difficulties

- L : number of steps
- ϵ : step size

- L is too small : random walk behavior
- L is too large : loop back and retrace their steps
- ϵ is too small : computational waste
- ϵ is too large : inaccurate

CONTENTS

- P : transition probability matrix

$$P_{ij} = P(X_{m+1} = j | X_m = i)$$

- Detailed balance equation

$$\pi_i P_{ij} = \pi_j P_{ji}$$

- Time reversible

$$P(X_{m+1} = j | X_m = i) = P(X_m = i | X_{m+1} = j)$$

CONTENTS

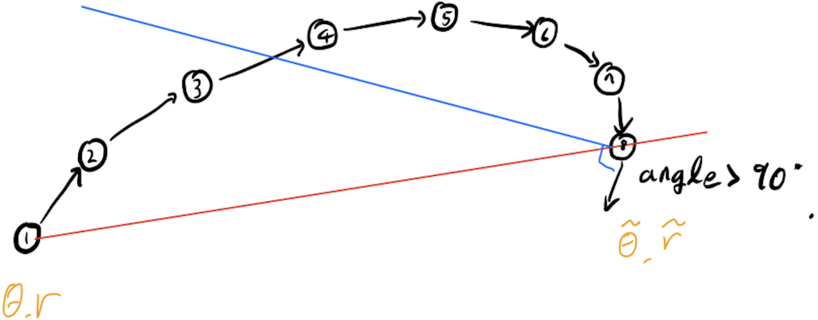
Main Idea

- Running the simulation for more steps until no longer increase the distance between the proposal $\tilde{\theta}$ and the initial value of θ.

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{(\tilde{\theta}-\theta)\cdot(\tilde{\theta}-\theta)}{2} = (\tilde{\theta}-\theta)\cdot\frac{\mathrm{d}}{\mathrm{d}t}(\tilde{\theta}-\theta) = (\tilde{\theta}-\theta)\cdot\tilde{r}$$

- Simulating until $(\tilde{\theta}-\theta)\cdot\tilde{r} < 0$.

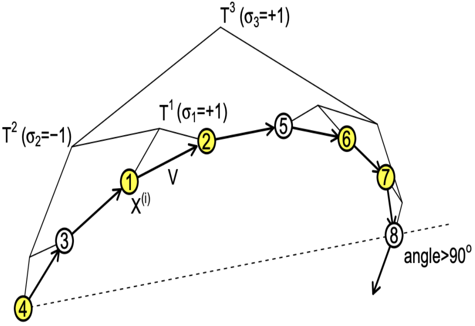
Main Idea



Problem

- This algorithm does not guarantee time reversibility.
- NUTS overcomes this issue by running the Hamiltonian simulation both forward and backward in time.
- Introduce a slice variable u for simplifying the NUTS.

Simple algorithm



- Choosing a random direction $\sigma_i \sim U(-1, 1)$
- Taking 2^j leapfrog steps of size $\sigma_j \epsilon$
- θ^-, r^- and θ^+, r^+ : the state of leftmost and rightmost leaves.
- Simulating until $(\theta^+ - \theta^-) \cdot r^- < 0$ or $(\theta^+ - \theta^-) \cdot r^+ < 0$

Slice sampling

- $\theta \sim \pi(\theta)$
  - $V = \{(\theta, u) : 0 < u < \pi(\theta)\}$
  - Sampling (θ, u) uniformly over the region V.
- Step 1 : $p(u|\theta) \propto \mathbb{I}[0 < u < \pi(\theta)]$
- Step 2 : $p(\theta|u) \propto \mathbb{I}[u < \pi(\theta)]$

- Slice sampling can suppress random walks.

Naive NUTS

- $\mathcal{B}$: all position-momentum states
- $\mathcal{C}$: candidate position-momentum states

- C.1 : All elements of $\mathcal{C}$ must be chosen in a way that preserves volume.
- C.2 : $P((\theta, r) \in \mathcal{C} | \theta, r, u, \epsilon) = 1$
- C.3 : $P(u \leq exp\{-U(\theta') - \frac{r'^T r'}{2}\} | (\theta', r') \in C) = 1$
- C.4 : If $(\theta, r) \in \mathcal{C}$ and $(\theta', r') \in \mathcal{C}$ then for any $\mathcal{C}$, $P(\mathcal{B}, \mathcal{C} | \theta, r, u, \epsilon) = P(\mathcal{B}, \mathcal{C} | \theta', r', u, \epsilon)$

Naive NUTS

- Step
  - 1. sample $r \sim N(0, I)$
  - 2. sample $u \sim U([0, exp(-U(\theta) - r \cdot r/2])$
  - 3. sample $\mathcal{B}, \mathcal{C}$ from $P(\mathcal{B}, \mathcal{C}|\theta(t), r, u, \epsilon)$
  - 4. sample $\theta(t+1), r \sim T(\theta_t, r, \mathcal{C})$
- T : transition kernel
- $u \leq exp\{-U(\theta') - r' \cdot r'/2\}$ then $(\theta', r') \in \mathcal{C}$
- Stop doubling if the tree includes a leaf node whose state θ, r satisfies $log(f(\theta)) - r \cdot r/2 - \log(u) < -\Delta_{max}$

Algorithm

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

Algorithm

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

- Naive NUTS algorithm requires that we store 2^j position and momentum vectors.
- Naive NUTS cannot guarantee large jumps.

$\Longrightarrow$ Introduce other transition kernel

New transition kernel

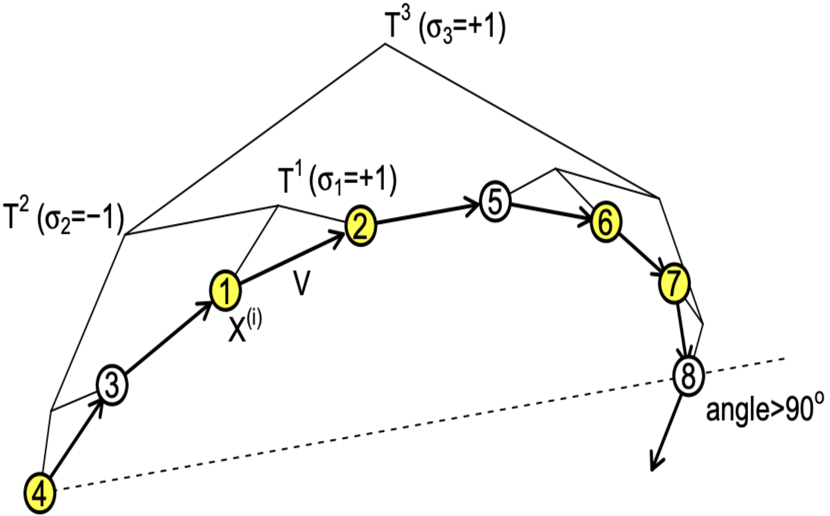
Consider the transition kernel

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

- $\mathcal{C} = \mathcal{C}^{old} \cup \mathcal{C}^{new}$, $w \in \mathcal{C}^{old}$
- T propose a move from $\mathcal{C}^{old}$ to a random state in $\mathcal{C}^{new}$ and accept the move with probabiltiy $\dfrac{|\mathcal{C}^{new}|}{|\mathcal{C}^{old}|}$
- It satisfies detailed balance with respect to the uniform distribution on $\mathcal{C}$, that is,

$$p(w|\mathcal{C})T(w'|w,\mathcal{C}) = p(w'|\mathcal{C})T(w|w',\mathcal{C})$$

New transition kernel



Algorithm

Algorithm 3 Efficient 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 $\theta^- = \theta^{m-1}$, $\theta^+ = \theta^{m-1}$, $r^- = r^0, r^+ = r^0, j = 0, \theta^m = \theta^{m-1}, n = 1, s = 1$.
  while $s = 1$ **do**
    Choose a direction $v_j \sim \text{Uniform}(\{-1, 1\})$.
    if $v_j = -1$ **then**
      $\theta^-, r^-, -, -, \theta', n', s' \leftarrow \text{BuildTree}(\theta^-, r^-, u, v_j, j, \epsilon)$.
    else
      $-, -, \theta^+, r^+, \theta', n', s' \leftarrow \text{BuildTree}(\theta^+, r^+, u, v_j, j, \epsilon)$.
    end if
    if $s' = 1$ **then**
      With probability $\min\{1, \frac{n'}{n}\}$, set $\theta^m \leftarrow \theta'$.
    end if
    $n \leftarrow n + n'$.
    $s \leftarrow s' \mathbb{I}[(\theta^+ - \theta^-) \cdot r^- \geq 0] \mathbb{I}[(\theta^+ - \theta^-) \cdot r^+ \geq 0]$.
    $j \leftarrow j + 1$.
  end while
end for

Algorithm

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

CONTENTS

Setting ϵ

- HMC : Optimal value of ϵ for a given simulation length ϵL is the one that produces an average Metropolis acceptance probability of approximately 0.65.

$$H_t^{HMC} = min\left\{1, \frac{p(\theta^t, r^t)}{p(\theta^{t-1}, r^{t,0})}\right\}; \quad h^{HMC}(\epsilon) = \mathbb{E}[H_t^{HMC}|\epsilon]$$

- NUTS : Since there is no single accept/reject step, we must define an alternative statistic to Metropolis acceptance probability.

$$H_t^{NUTS} = \frac{1}{|\mathcal{B}_t^{final}|} \sum_{\theta, r \in \mathcal{B}_t^{final}} min\left\{1, \frac{p(\theta, r)}{p(\theta^{t-1}, r^{t,0})}\right\};$$

$$h^{NUTS}(\epsilon) = \mathbb{E}[H_t^{NUTS}|\epsilon]$$

Dual Averaging

- Assuming that we want to find a setting of a parameter $x \in \mathbb{R}$ such that $h(x) = \mathbb{E}_t[H_t|x] = 0$, we can apply the updates

$$x_{t+1} \leftarrow \mu - \frac{\sqrt{t}}{\gamma}\frac{1}{t+t_0}\sum_{i=1}^{t} H_i; \quad \bar{x}_{t+1} \leftarrow \eta_t x_{t+1} + (1-\eta_t)\bar{x}_t$$

- $\eta_t = t^{-\kappa}, \ \kappa \in (0.5, 1]$
- Assuming that h is nondecreasing as a function of x

$\Longrightarrow$ $\bar{x}_t$ is guaranteed to converge to a value such that $h(\bar{x}_t)$ converges to 0.

Algorithm(HMC with Dual Averaging)

Algorithm 5 Hamiltonian Monte Carlo with Dual Averaging

Given θ^0, δ, λ, $\mathcal{L}, M, M^{\text{adapt}}$:
Set $\epsilon_0 = \text{FindReasonableEpsilon}(\theta), \mu = \log(10\epsilon_0), \bar{\epsilon}_0 = 1, \bar{H}_0 = 0, \gamma = 0.05, t_0 = 10, \kappa = 0.75$.
for $m = 1$ to M **do**
  Reample $r^0 \sim \mathcal{N}(0, I)$.
  Set $\theta^m \leftarrow \theta^{m-1}, \tilde{\theta} \leftarrow \theta^{m-1}, \tilde{r} \leftarrow r^0, L_m = \max\{1, \text{Round}(\lambda/\epsilon_{m-1})\}$.
  for $i = 1$ to L_m **do**
    Set $\tilde{\theta}, \tilde{r} \leftarrow \text{Leapfrog}(\tilde{\theta}, \tilde{r}, \epsilon_{m-1})$.
  end for
  With probability $\alpha = \min\left\{1, \frac{\exp\{\mathcal{L}(\tilde{\theta}) - \frac{1}{2}\tilde{r}\cdot\tilde{r}\}}{\exp\{\mathcal{L}(\theta^{m-1}) - \frac{1}{2}r^0 \cdot r^0\}}\right\}$, set $\theta^m \leftarrow \tilde{\theta}, r^m \leftarrow -\tilde{r}$.
  if $m \leq M^{\text{adapt}}$ **then**
    Set $\bar{H}_m = \left(1 - \frac{1}{m+t_0}\right)\bar{H}_{m-1} + \frac{1}{m+t_0}(\delta - \alpha)$.
    Set $\log \epsilon_m = \mu - \frac{\sqrt{m}}{\gamma}\bar{H}_m, \log \bar{\epsilon}_m = m^{-\kappa} \log \epsilon_m + (1 - m^{-\kappa}) \log \bar{\epsilon}_{m-1}$.
  else
    Set $\epsilon_m = \bar{\epsilon}_{M^{\text{adapt}}}$.
  end if
end for

Algorithm(NUTS with Dual Averaging)

Algorithm 6 No-U-Turn Sampler with Dual Averaging

Given θ^0, δ, $\mathcal{L}, M, M^{\text{adapt}}$:
Set $\epsilon_0 = \text{FindReasonableEpsilon}(\theta), \mu = \log(10\epsilon_0), \bar{\epsilon}_0 = 1, \bar{H}_0 = 0, \gamma = 0.05, t_0 = 10, \kappa = 0.75$.
for $m = 1$ to M **do**
  Sample $r^0 \sim \mathcal{N}(0, I)$.
  Resample $u \sim \text{Uniform}([0, \exp\{\mathcal{L}(\theta^{m-1} - \frac{1}{2} r^0 \cdot r^0\}])$
  Initialize $\theta^- = \theta^{m-1}$, $\theta^+ = \theta^{m-1}$, $r^- = r^0, r^+ = r^0, j = 0, \theta^m = \theta^{m-1}, n = 1, s = 1$.
  while $s = 1$ **do**
    Choose a direction $v_j \sim \text{Uniform}(\{-1, 1\})$.
    if $v_j = -1$ **then**
      $\theta^-, r^-, -, -, \theta', n', s', \alpha, n_\alpha \leftarrow \text{BuildTree}(\theta^-, r^-, u, v_j, j, \epsilon_{m-1}\theta^{m-1}, r^0)$.
    else
      $-, -, \theta^+, r^+, \theta', n', s', \alpha, n_\alpha \leftarrow \text{BuildTree}(\theta^+, r^+, u, v_j, j, \epsilon_{m-1}, \theta^{m-1}, r^0)$.
    end if
    if $s' = 1$ **then**
      With probability $\min\{1, \frac{n'}{n}\}$, set $\theta^m \leftarrow \theta'$.
    end if
    $n \leftarrow n + n'$.
    $s \leftarrow s' \mathbb{I}[(\theta^+ - \theta^-) \cdot r^- \geq 0] \mathbb{I}[(\theta^+ - \theta^-) \cdot r^+ \geq 0]$.
    $j \leftarrow j + 1$.
  end while
  if $m \leq M^{\text{adapt}}$ **then**
    Set $\bar{H}_m = \left(1 - \frac{1}{m+t_0}\right) \bar{H}_{m-1} + \frac{1}{m+t_0}(\delta - \frac{\alpha}{n_\alpha})$.
    Set $\log \epsilon_m = \mu - \frac{\sqrt{m}}{\gamma} \bar{H}_m, \log \bar{\epsilon}_m = m^{-\kappa} \log \epsilon_m + (1 - m^{-\kappa}) \log \bar{\epsilon}_{m-1}$.
  else
    Set $\epsilon_m = \bar{\epsilon}_{M^{\text{adapt}}}$.
  end if
end for

Algorithm(NUTS with Dual Averaging)

function $\text{BuildTree}(\theta, r, u, v, j, \epsilon, \theta^0, r^0)$
if $j = 0$ **then**
Base case—take one leapfrog step in the direction v.
$\theta', r' \leftarrow \text{Leapfrog}(\theta, r, v\epsilon)$.
$n' \leftarrow \mathbb{I}[u \le \exp\{\mathcal{L}(\theta') - \frac{1}{2}r' \cdot r'\}]$.
$s' \leftarrow \mathbb{I}[u < \exp\{\Delta_{\max} + \mathcal{L}(\theta') - \frac{1}{2}r' \cdot r'\}]$.
return $\theta', r', \theta', r', \theta', n', s', \min\{1, \exp\{\mathcal{L}(\theta') - \frac{1}{2}r' \cdot r' - \mathcal{L}(\theta^0) + \frac{1}{2}r^0 \cdot r^0\}\}, 1$.
else
Recursion—implicitly build the left and right subtrees.
$\theta^-, r^-, \theta^+, r^+, \theta', n', s', \alpha', n'_\alpha \leftarrow \text{BuildTree}(\theta, r, u, v, j-1, \epsilon, \theta^0, r^0)$.
if $s' = 1$ **then**
if $v = -1$ **then**
$\theta^-, r^-, -, -, \theta'', n'', s'', \alpha'', n''_\alpha \leftarrow \text{BuildTree}(\theta^-, r^-, u, v, j-1, \epsilon, \theta^0, r^0)$.
else
$-, -, \theta^+, r^+, \theta'', n'', s'', \alpha'', n''_\alpha \leftarrow \text{BuildTree}(\theta^+, r^+, u, v, j-1, \epsilon, \theta^0, r^0)$.
end if
With probability $\frac{n''}{n'+n''}$, set $\theta' \leftarrow \theta''$.
Set $\alpha' \leftarrow \alpha' + \alpha''$, $n'_\alpha \leftarrow n'_\alpha + n''_\alpha$.
$s' \leftarrow s''\mathbb{I}[(\theta^+ - \theta^-) \cdot r^- \ge 0]\mathbb{I}[(\theta^+ - \theta^-) \cdot r^+ \ge 0]$
$n' \leftarrow n' + n''$
end if
return $\theta^-, r^-, \theta^+, r^+, \theta', n', s', \alpha', n'_\alpha$.
end if

CONTENTS

Model and Data Sets

- 250-dim MVN

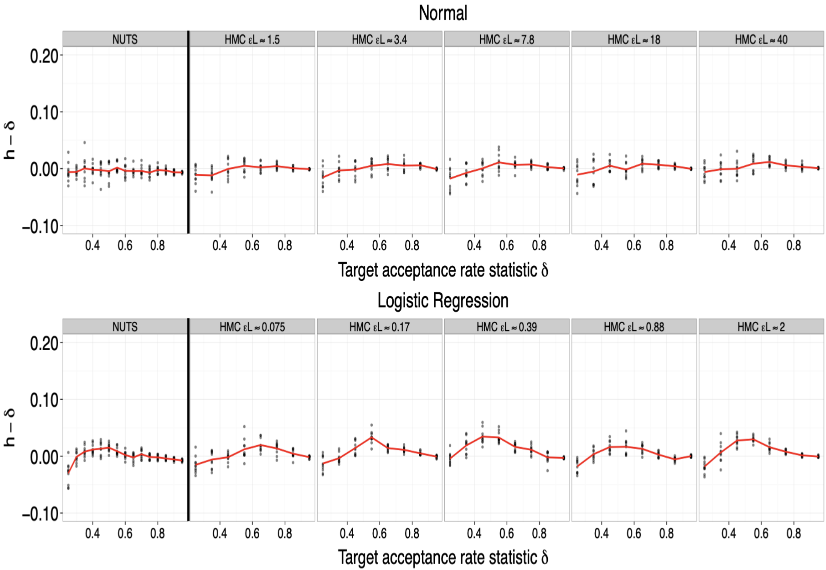
$$p(\theta) \propto exp\{-\frac{1}{2}\theta^T A\theta\}$$

- Bayesian logistic regression
    - α : intercept term
    - β : vector of 24 regression coeffients

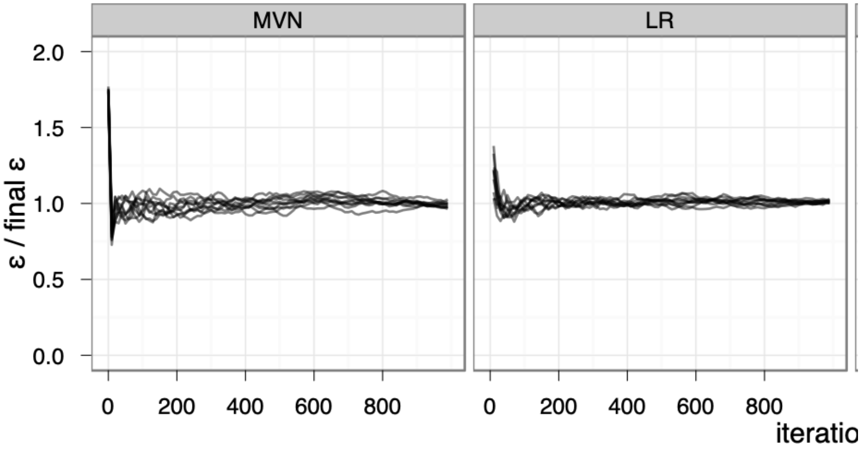
$$p(\alpha, \beta|x, y) \propto p(y|x, \alpha, \beta)p(\alpha)p(\beta)$$

$$\propto exp\Big\{ -\sum_i \log(1 + exp\{-y_i(\alpha + x_i \cdot \beta)\}) - \frac{\alpha^2}{2\sigma^2} - \frac{\beta \cdot \beta}{2\sigma^2} \Big\}$$

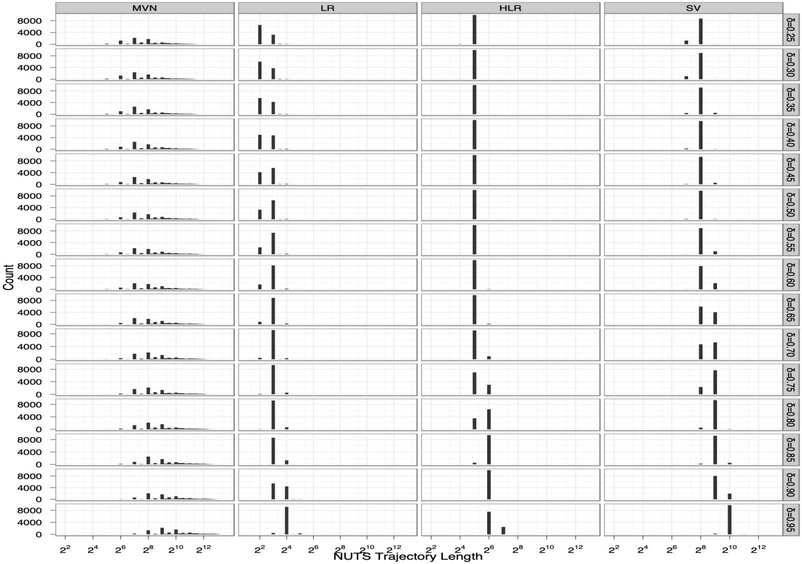
Convergence of Dual Averaging



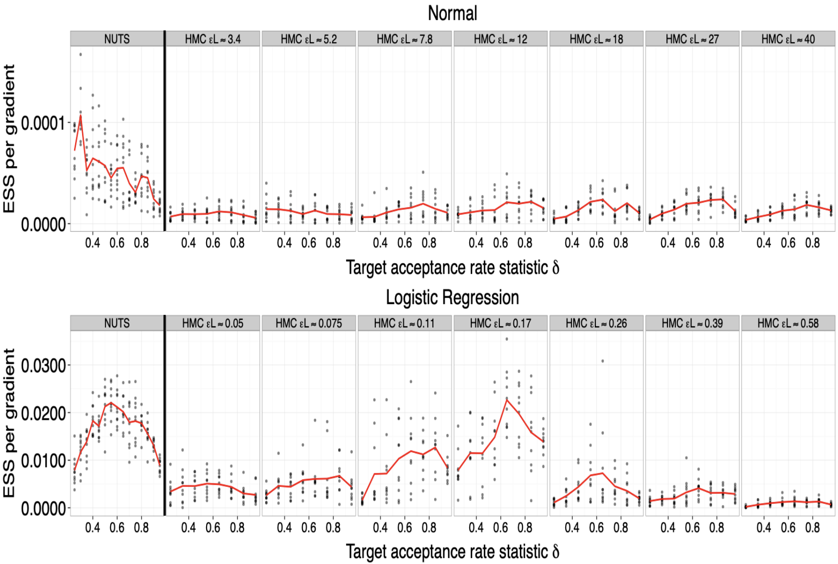
Convergence of Dual Averaging



NUTS Trajectory Lengths

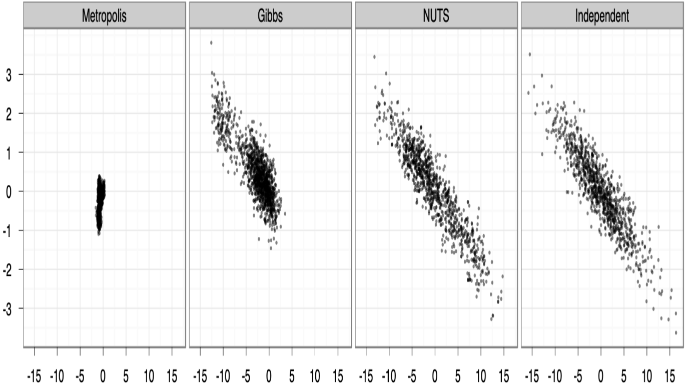


Efficiency of HMC and NUTS



Qualitative Comparison of NUTS,HMC and Gibbs

- Sampling on the 250-dim MVN.(projected onto the first two dimensions)
- 1,000,000 iteration (Gradient)



Reference I

Betancourt, M. (2018).
A Conceptual Introduction to Hamiltonian Monte Carlo
arXiv Preprint arXiv:1701.02434.

Hoffman, M.D. and Gelman, A. (2014).
The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo.
Journal of Machine Learning Research, 31(3), 363-396.

Neal, R. (2003).
Slice sampling.
Annals of Statistics, 31(3), 705-741.

Park, J. and Atchadá, Y. (2018).
Markov chain Monte Carlo algorithms with sequential proposals.
Statistics and Computing, 30, 1325-1345.