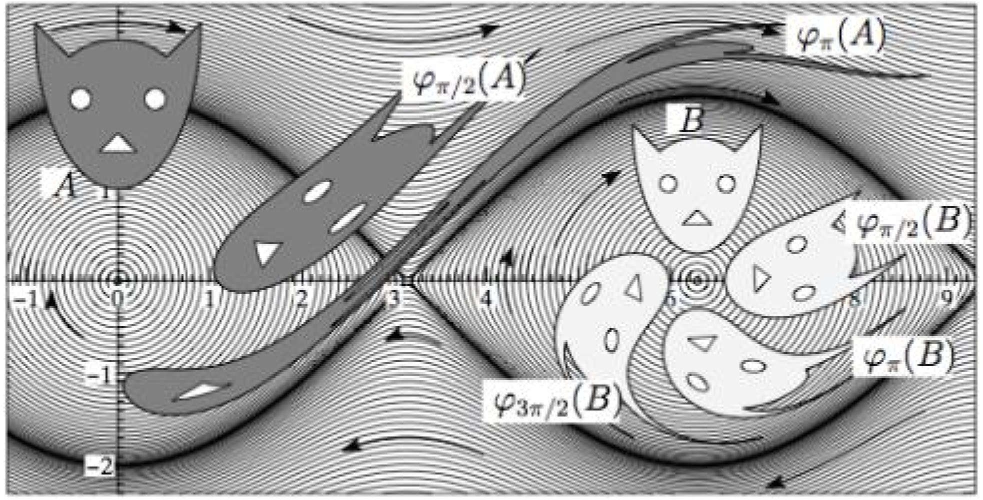
The Magic of NUTS

By Ryan Levy

For AIG 12/14/20

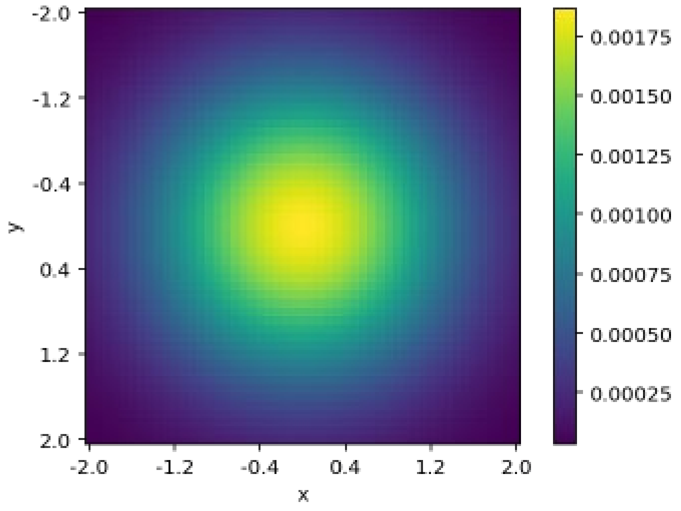


https://statmodeling.stat.columbia.edu/2017/01/12/conceptual-introduction-hamiltonian-monte-carlo/

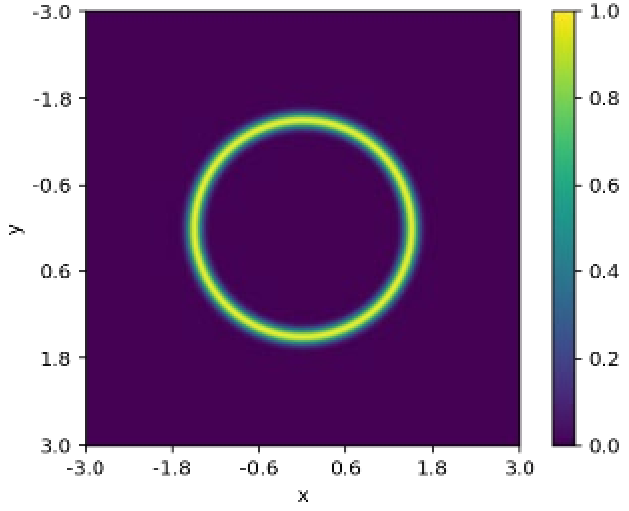
Goal: Sample from a distribution

- I want to get samples from a distribution $\pi(s)$, to for example take an integral
$$\int Q(s)\pi(s)ds$$

- Rarely do we have $\pi(s)$ but instead $\mathrm{P}(s) \propto \pi(s)$ (unnormalized)



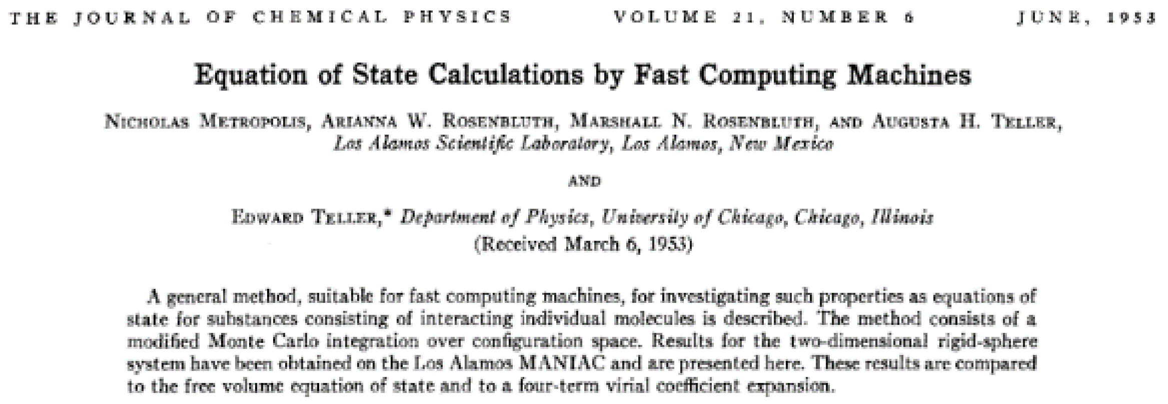
2D Gaussian (No Correlation)



Ring

Potential Solution

Metropolis-Hastings (1953/1970)


THE JOURNAL OF CHEMICAL PHYSICS VOLUME 21, NUMBER 6 JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*
(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

- Generate samples via random walk
- Algorithm:
  - Propose new state (e.g. Gaussian step about current spot)
  - Accept/reject with based on *probability proportional to target* $\pi(s)$

- Turns out:
  - Gaussian – Great! Fast!
  - Ring – Less great… Slow…

Live Code Break

Alternative to Metropolis

Hamiltonian Monte Carlo

- Leverage between random walks and wasting computational time
- Meet Hamiltonian (Hybrid) Monte Carlo (1987)
  - Idea: Introduce fictitious momentum and 'integrate out'
  - How? Randomly kick ball and follow potential of probability distribution
- Still Markov Chain Algorithm
  - Now need derivatives

Can your Markov chain do this?

Thank you Richard McElreath

Hamiltonian Monte Carlo (HMC) Algorithm

- (Gibbs) Sample momentum p from $\sim \mathcal{N}(0, I)$
- Run Hamiltonian Dynamics via Leapfrog for L steps @ dt to generate new sample x'

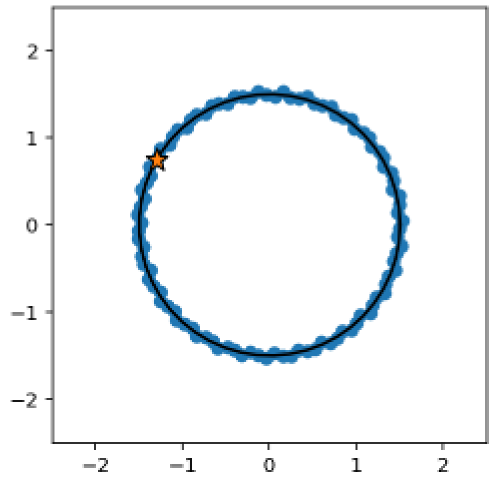
$$H = \frac{1}{2} p^T M^{-1} p - \log f(x)$$

  - Mass term rescales problem to take appropriate size steps
  - Leapfrog conserves energy/time (symplectic integrator)
  - Leapfrog needs $\nabla_x \log f(x)$
- Accept/Reject with probability $\min\{1, \frac{\exp[-H(x')]}{\exp[-H(x)]}\}$

Live Code Break

Why HMC isn’t the best solution

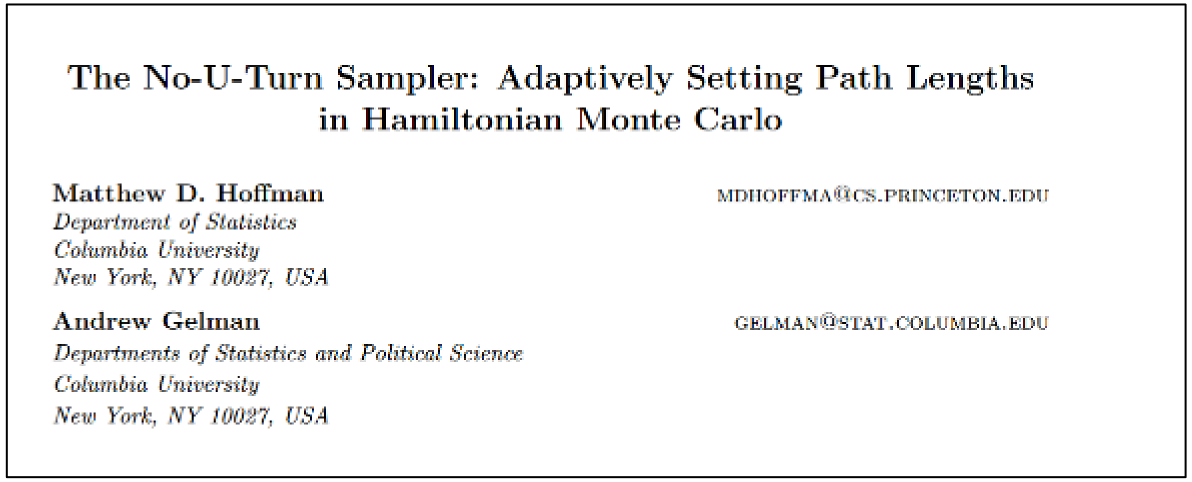
- Easy to waste computer time
  - dt:
    - Too small -> small steps
    - Too large -> inaccurate dynamics, low acceptance
    - Direction -> may break detailed balance
  - L
    - Too small -> highly correlated,
    - Too large -> The dreaded **U-turn**



A U-turn that haunts statisticians dreams

Improvement: NUTS

The No-U-Turn Sampler (2011/2014)



The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo

Matthew D. Hoffman MDHOFFMA@CS.PRINCETON.EDU
Department of Statistics
Columbia University
New York, NY 10027, USA

Andrew Gelman GELMAN@STAT.COLUMBIA.EDU
Departments of Statistics and Political Science
Columbia University
New York, NY 10027, USA

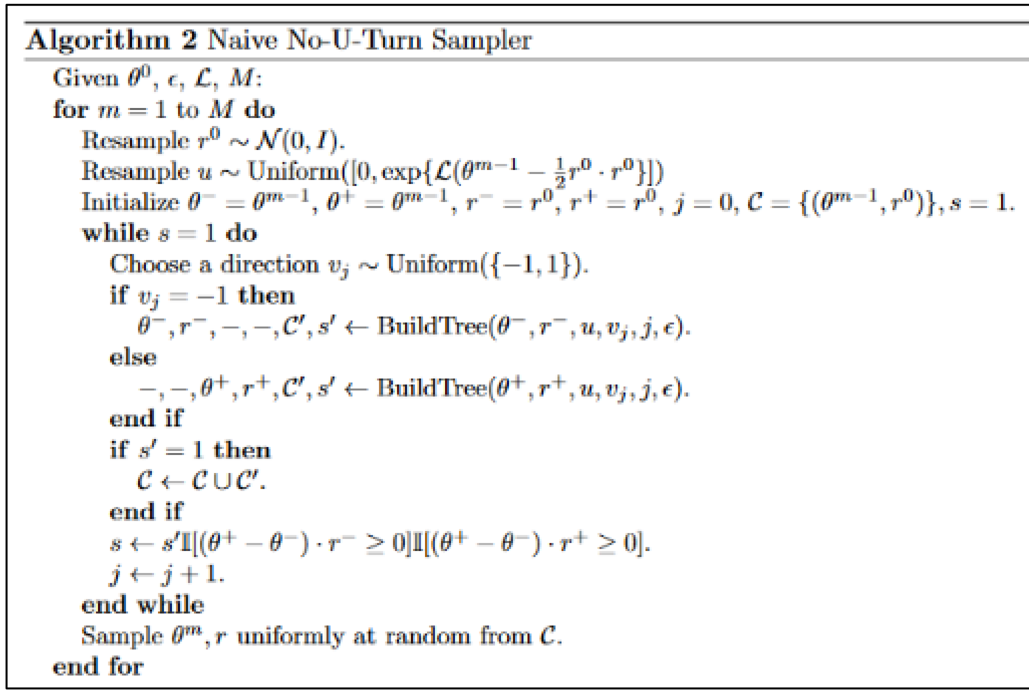
- Motivation: Remove need to tune L and dt

- Today: **Focus on removing L (Algorithms 2,3)**
- Not today: Dual averaging to eliminate dt (Alg 6)

Alg 2: Naïve NUTS

- Minor Details
  - Detailed Balance
  - Maintain time reversal invariance

- Main Idea: Build a (balanced) tree of doubled leapfrog steps 2^j
  - Walk forward until we hit a U-turn (or our tree is too big)
  - Introduce a new (Gibbs) sampled slice variable u
    - Helps avoid big trees and low probability states (stochastically)
- Choose one of the states that we visit (C)
  - Uniformly within C
  - Carefully preserve volume



Algorithm 2 Naive No-U-Turn Sampler

```
Given $\theta^0$, $\epsilon$, $\mathcal{L}$, $M$:
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 $\theta^m, r$ uniformly at random from $\mathcal{C}$.
end for
```

Naïve NUTS – Tree Building🌲

Leapfrog Paths ●

Balanced Binary Tree ●

Live Code Break

jupyter

But we can do better

- Normally the limiting factor is the gradient and $P(s)$ evaluations
- However, our Naïve algorithm stores $2^j - 1$ position & momentum vectors
- Also
    - what if we hit a U-turn while filling out the tree?
    - Why sample steps uniformly? *(fixed due to dt solutions)*

- Solutions:
    - Sample from subtrees as we go
        - Secret Metropolis Step assuming Uniform choice over C
    - Trade $O(2^j)$ memory for $O(j)$ memory and $O(2^j)$ random numbers

Live Code Break

“Benchmark” Examples

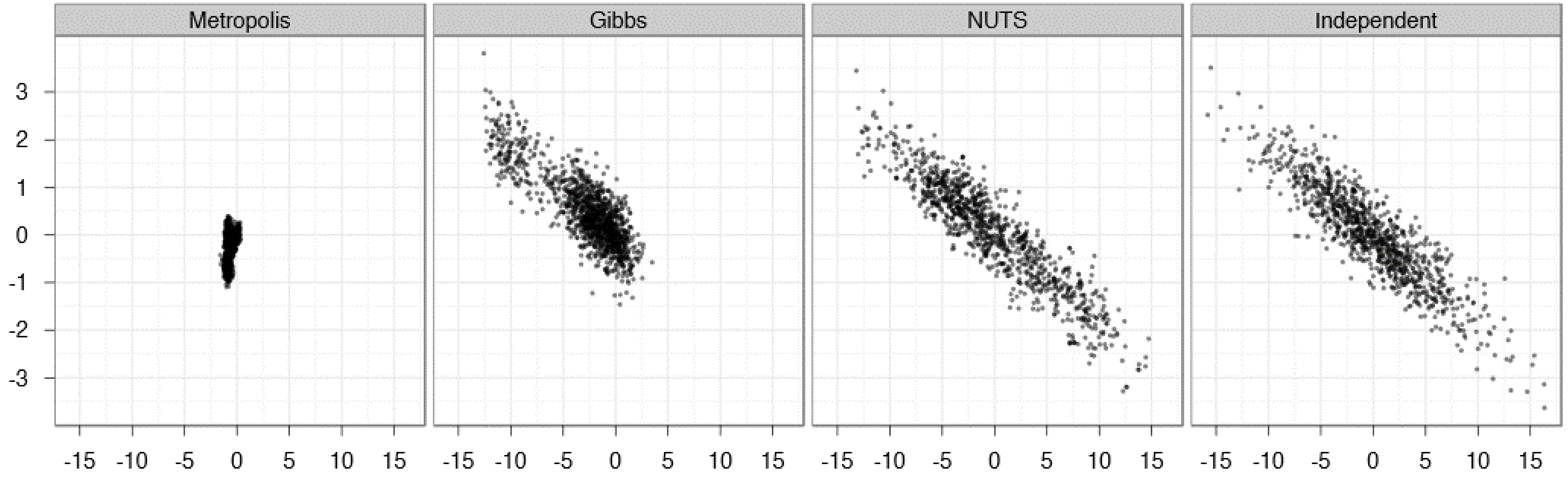


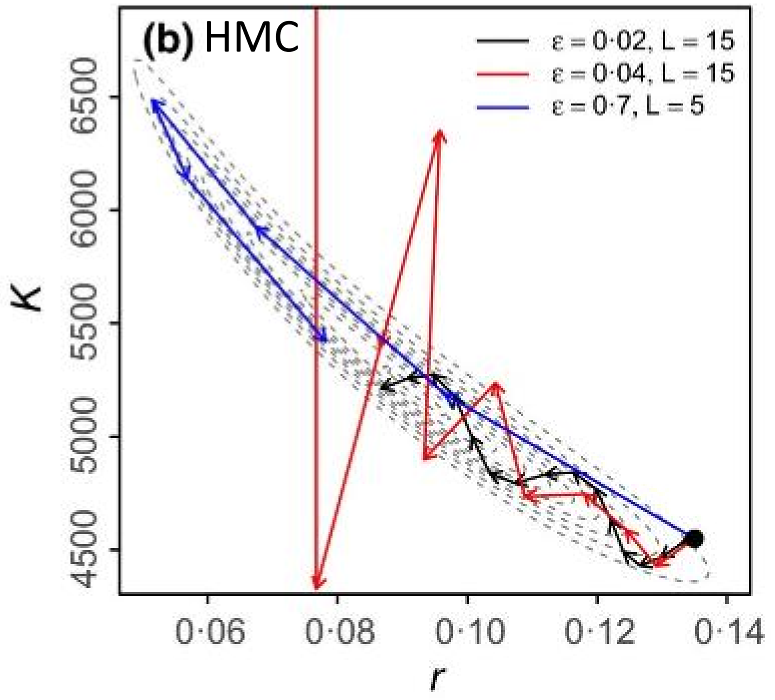
Figure 7: Samples generated by random-walk Metropolis, Gibbs sampling, and NUTS. The plots compare 1,000 independent draws from a highly correlated 250-dimensional distribution (right) with 1,000,000 samples (thinned to 1,000 samples for display) generated by random-walk Metropolis (left), 1,000,000 samples (thinned to 1,000 samples for display) generated by Gibbs sampling (second from left), and 1,000 samples generated by NUTS (second from right). Only the first two dimensions are shown here.

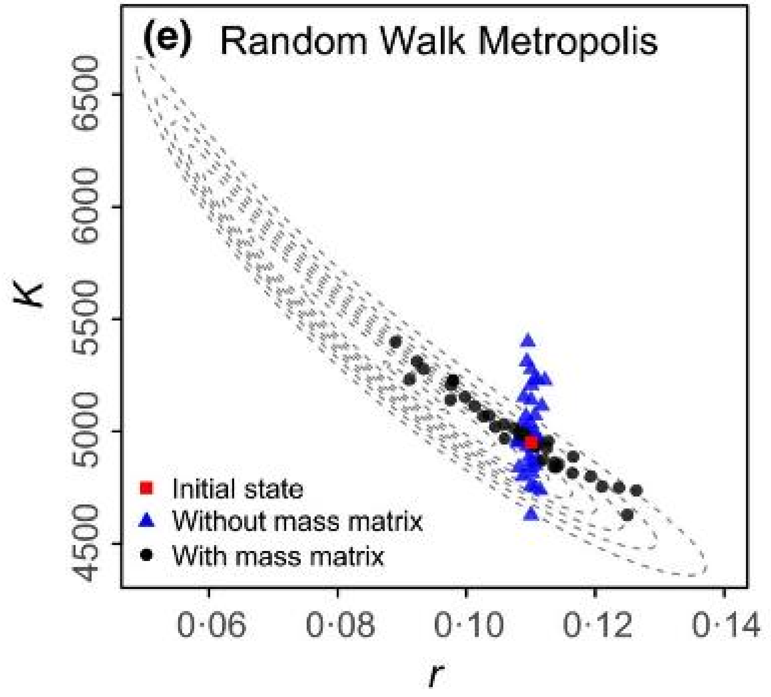
arXiv:1111.4246

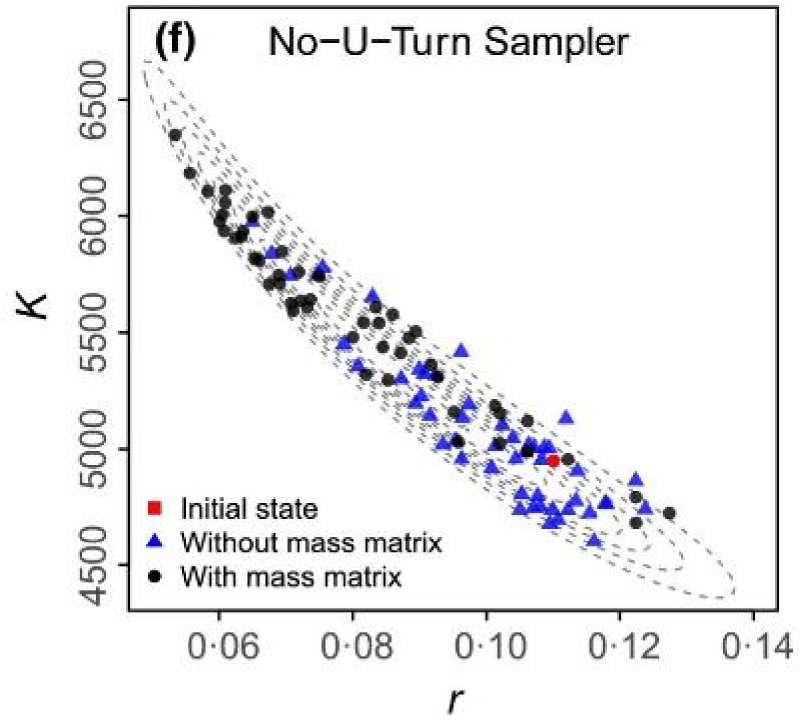
Even more “modern” updates (2013/2016)

- Modern implementations remove slice variable
  - Biased progressive sampling to favor away from current spot
  - Rao–Blackwell theorem tells us this is always as good/better
- Adaptively set dt
  - Dual averaging HMC
  - Heuristic seems good enough
- Generalized Stopping Criteria
  - Non-Identity mass matrix requires new stopping criteria
  - Fixes Banana/Twisted Gaussian problem

“Benchmark” Examples







Monnahan, C.C., Thorson, J.T. and Branch, T.A. (2017), Faster estimation of Bayesian models in ecology using Hamiltonian Monte Carlo. Methods Ecol Evol, 8: 339-348. https://doi.org/10.1111/2041-210X.12681

Things to consider

- NUTS requires continuous variables

- Need a lot of $\log(P(s))$ and gradients
  - Automatic Differentiation should help greatly here

- Trade in quality over runtime

Two Robust Implementations

Stan
https://mc-stan.org/

PyMC3
https://docs.pymc.io/

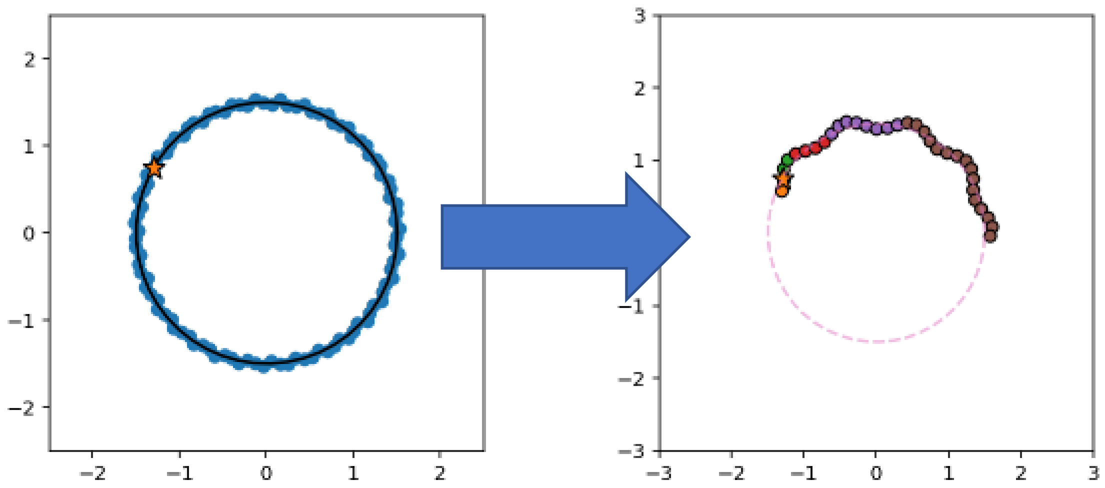
A Fiscally Sponsored Project of
NUMFOCUS
OPEN CODE = BETTER SCIENCE

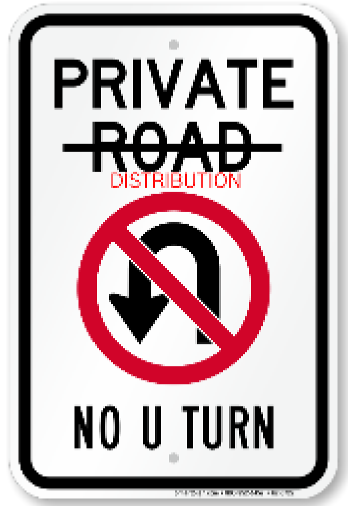
References – Thanks!

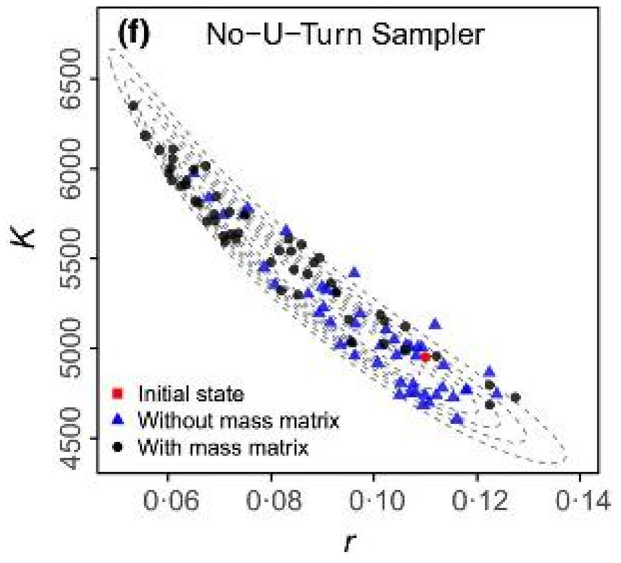
- https://elevanth.org/blog/2017/11/28/build-a-better-markov-chain/
- https://github.com/chi-feng/mcmc-demo
- https://arxiv.org/pdf/1111.4246.pdf
- https://arxiv.org/abs/1304.1920
- https://github.com/mfouesneau/NUTS
- https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/2041-210X.12681
  - Very accessible HMC/NUTS intro
- https://arxiv.org/pdf/1701.02434.pdf
  - Another great HMC Intro

Conclusion

- We sample continuous distributions better via the No-U-Turn Sampler (NUTS)
- The secret is the flow of Hamiltonian dynamics with subtle auxiliary variable tricks
- Still more to learn!






(f) No–U–Turn Sampler
Initial state
Without mass matrix
With mass matrix
K
r