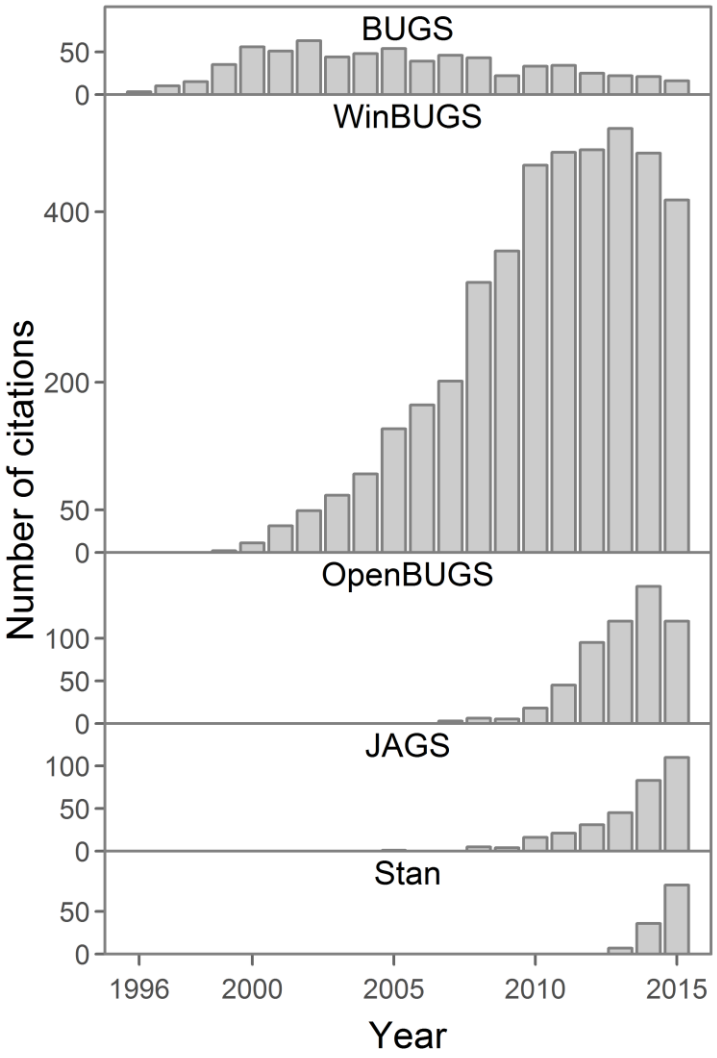
Faster estimation of Bayesian models with Stan

Modelos Bayesianos con aplicaciones ecológicas

Dr. Cole Monnahan
University of Concepción, Chile

Enero, 2019

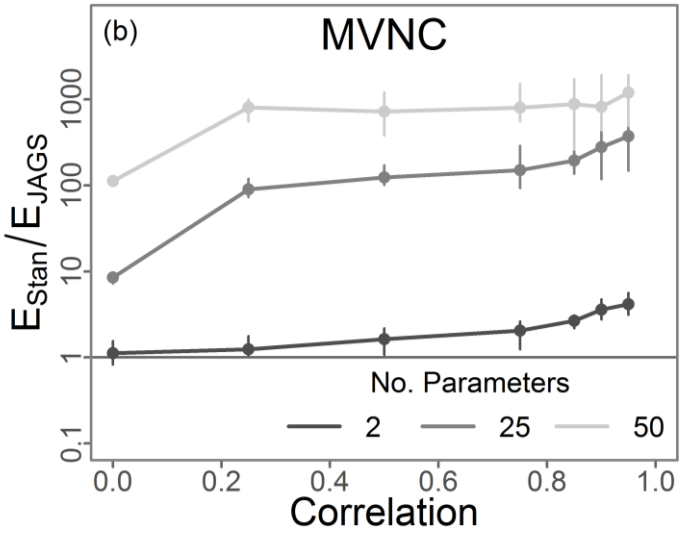
Stan: the new frontier in Bayesian analysis

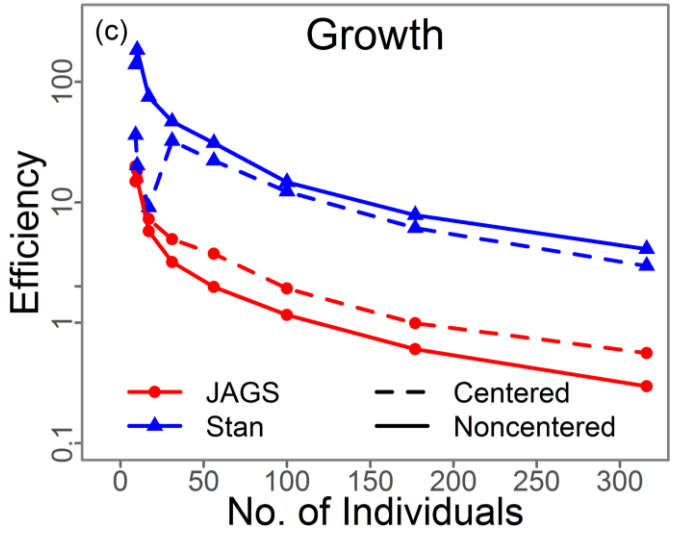


- Stan is growing rapidly compared to other programs
- Stan is more than software:
  - Valuable resources
  - Helpful community
  - Development of methods
  - Suite of supporting software

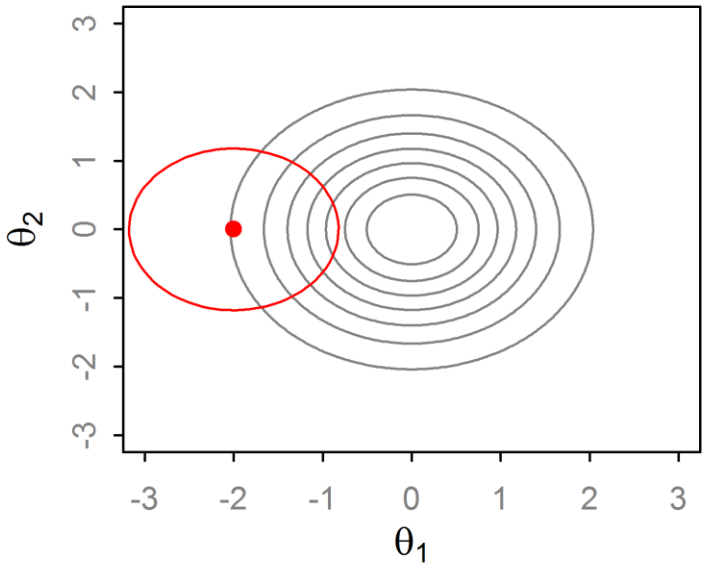
Stan: the new frontier in Bayesian analysis

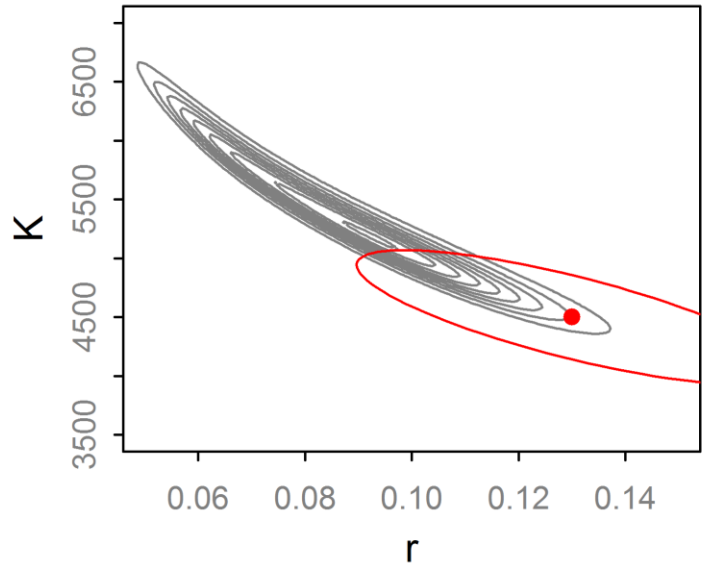
- It is FAST.
- Sometimes hundreds or thousands of times faster than JAGS
- Scales well with dimensionality & complexity
- It expands the possible models that can be fit





Random Walk Metropolis (RWM)





- Propose $\boldsymbol{\theta}^*$ with distribution $q \sim N(\boldsymbol{\theta}_t, \Sigma)$.
- Then set:

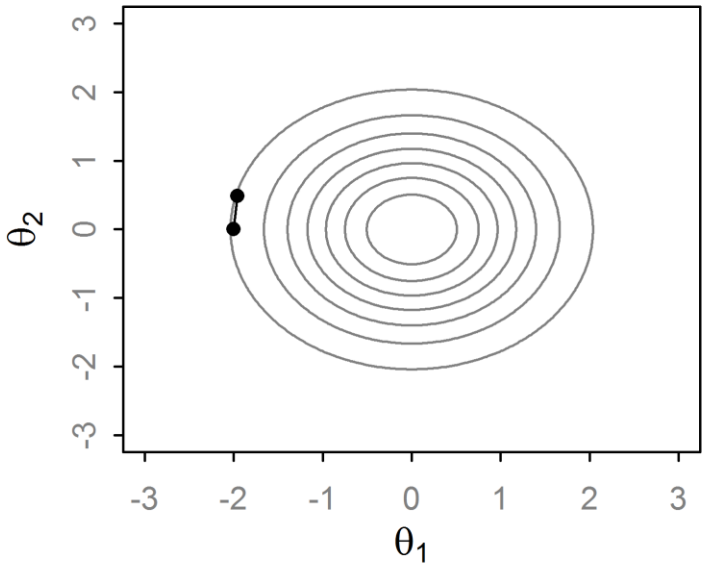
If *q* is symmetric this cancels out

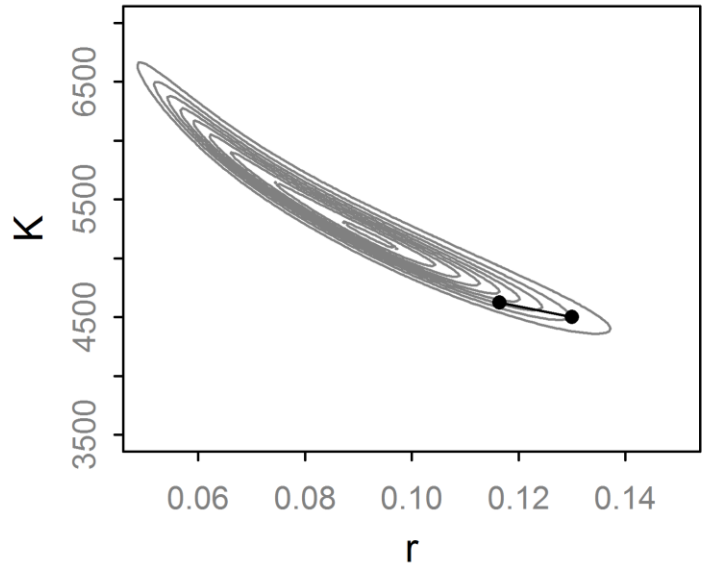
$$\theta_{t+1} = \theta^* \text{ if runif}(1) \leq \frac{f(\theta^*)q(\theta^* \mid \theta_t)}{f(\theta_t)q(\theta_t \mid \theta^*)}$$

$$= \theta_t \text{ otherwise}$$

- *q* affects efficiency of RWM so it needs to be 'tuned'

Random Walk Metropolis (RWM)





- Propose $\boldsymbol{\theta}^*$ with distribution $q \sim N(\boldsymbol{\theta}_t, \Sigma)$.
- Then set:

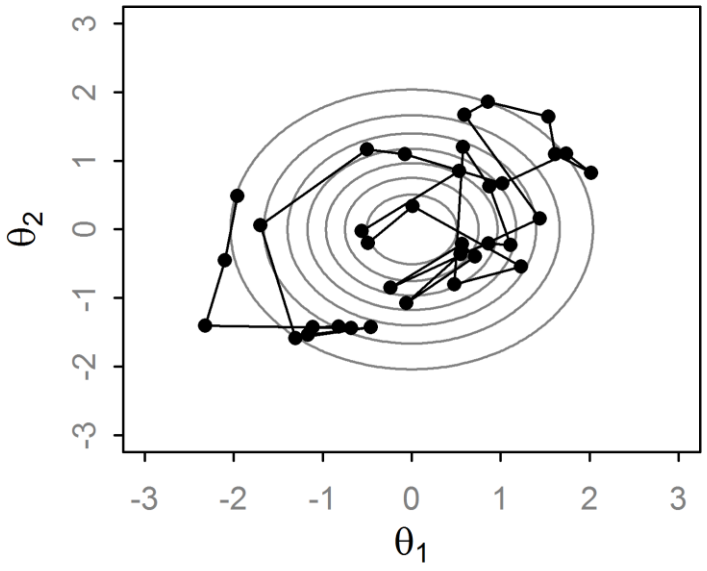
If *q* is symmetric this cancels out

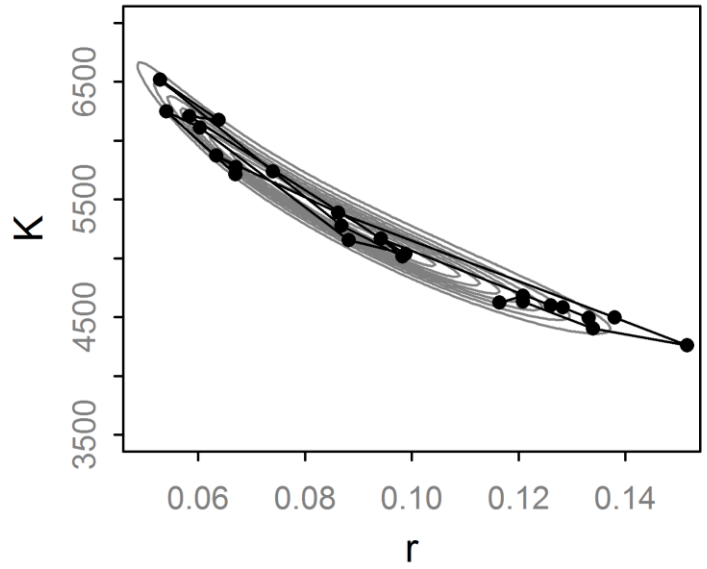
$$\theta_{t+1} = \theta^* \text{ if runif}(1) \leq \frac{f(\theta^*)q(\theta^* \mid \theta_t)}{f(\theta_t)q(\theta_t \mid \theta^*)}$$

$$= \theta_t \text{ otherwise}$$

- *q* affects efficiency of RWM so it needs to be 'tuned'

Random Walk Metropolis (RWM)





- Propose $\boldsymbol{\theta}^*$ with distribution $q \sim N(\boldsymbol{\theta}_t, \Sigma)$.
- Then set:

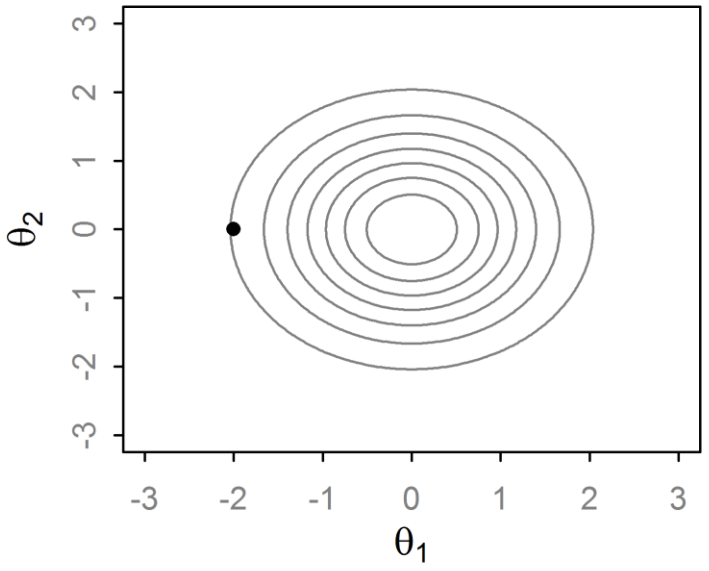
If *q* is symmetric this cancels out

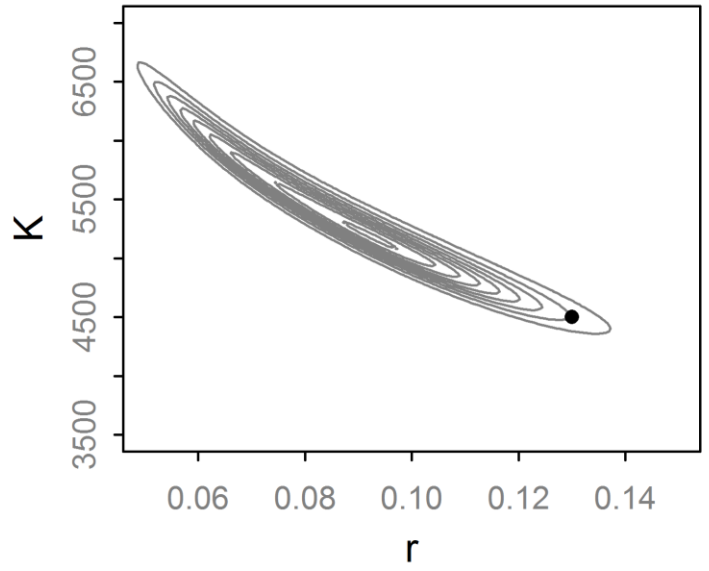
$$\theta_{t+1} = \theta^* \text{ if runif}(1) \leq \frac{f(\theta^*)q(\theta^* \mid \theta_t)}{f(\theta_t)q(\theta_t \mid \theta^*)}$$

$$= \theta_t \text{ otherwise}$$

- *q* affects efficiency of RWM so it needs to be 'tuned'

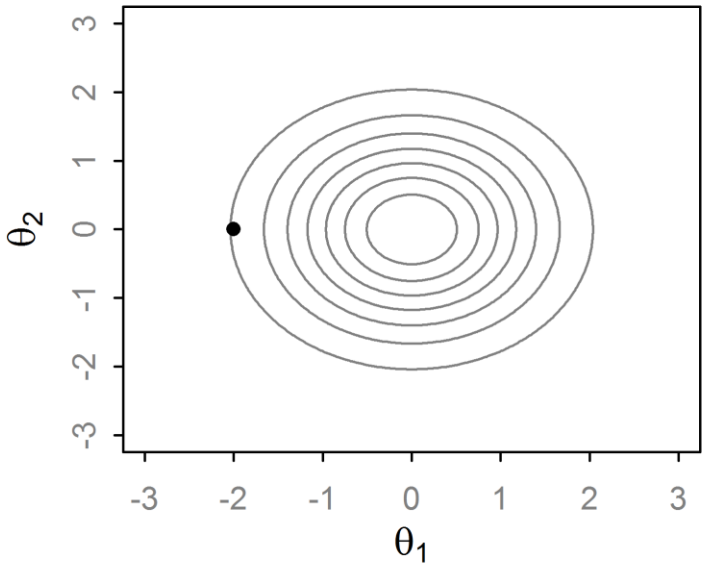
Gibbs Sampler

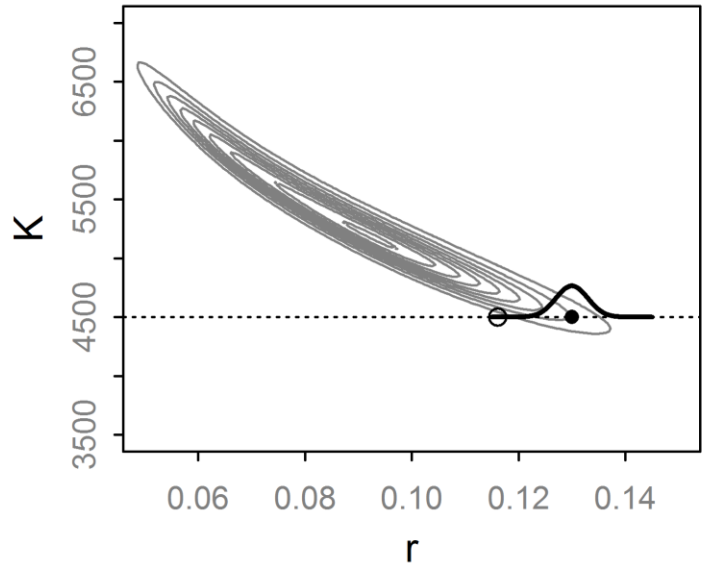




- Condition on all but first variable, find conjugate form
- Generate a value from this "full conditional" distribution.
- Repeat for all variables. That is a single step.
- If not conjugate, do Metropolis-within-Gibbs
- No tuning necessary, but poor efficiency for correlated parameters

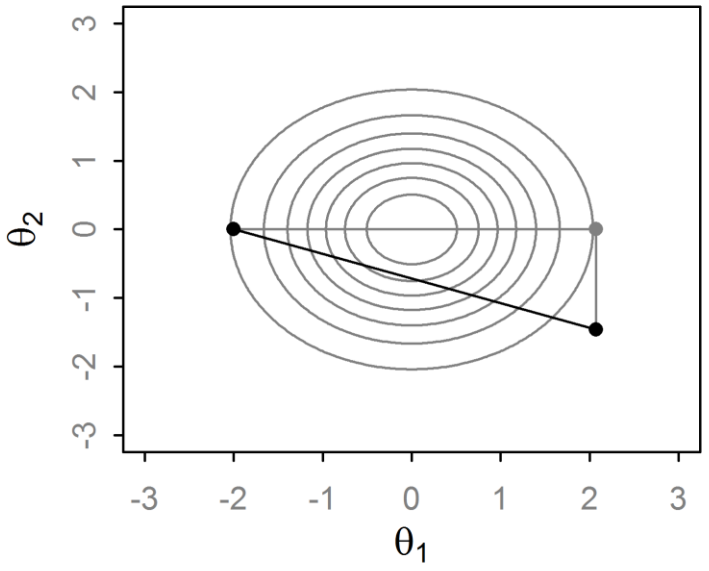
Gibbs Sampler

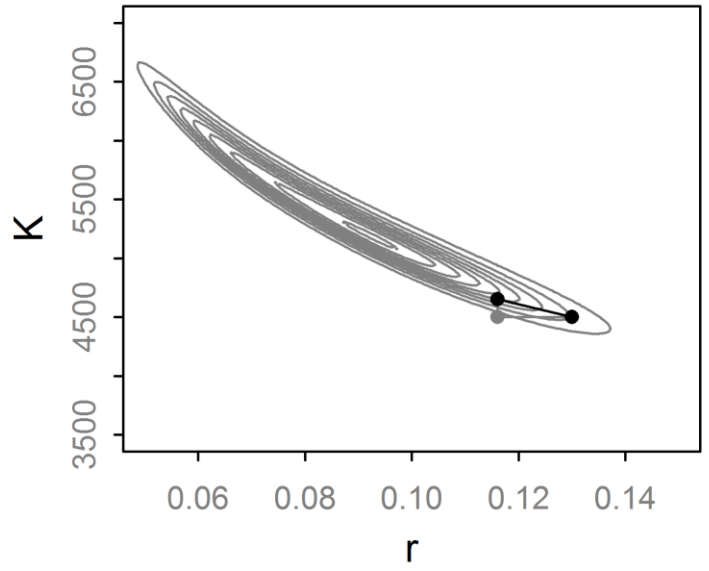




- Condition on all but first variable, find conjugate form
- Generate a value from this "full conditional" distribution.
- Repeat for all variables. That is a single step.
- If not conjugate, do Metropolis-within-Gibbs
- No tuning necessary, but poor efficiency for correlated parameters

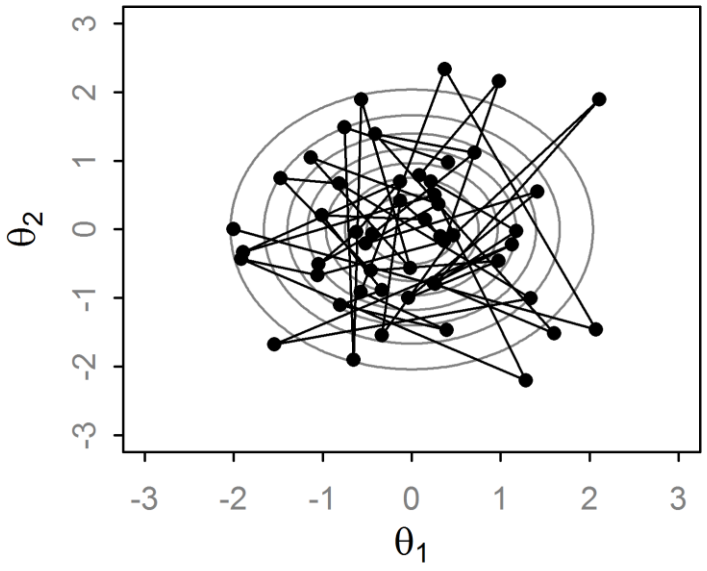
Gibbs Sampler

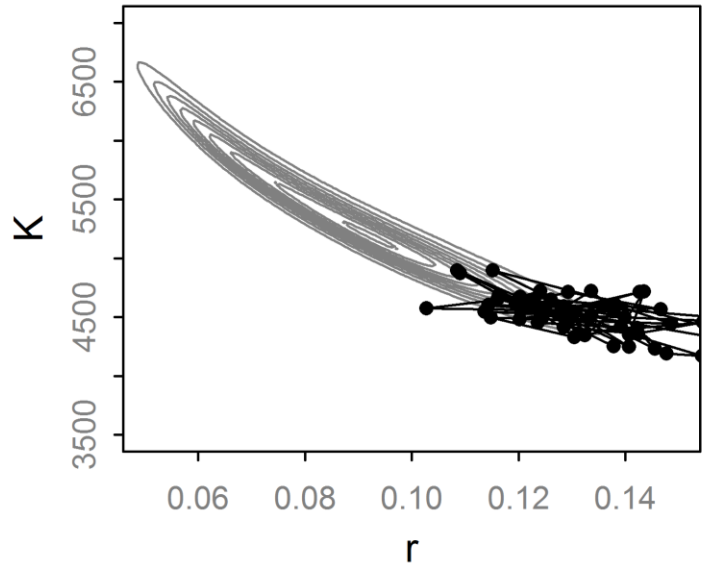




- Condition on all but first variable, find conjugate form
- Generate a value from this "full conditional" distribution.
- Repeat for all variables. That is a single step.
- If not conjugate, do Metropolis-within-Gibbs
- No tuning necessary, but poor efficiency for correlated parameters

Gibbs Sampler





- Condition on all but first variable, find conjugate form
- Generate a value from this "full conditional" distribution.
- Repeat for all variables. That is a single step.
- If not conjugate, do Metropolis-within-Gibbs
- No tuning necessary, but poor efficiency for correlated parameters

Beyond RWM and Gibbs

- RWM pros/cons:
  - Easy to implement and works well for many problems w/o conjugacy.
  - Must be tuned, can be very sensitive to this
- Gibbs pros/cons:
  - No tuning needed, if full conditionals are possible
  - Easy to implement (JAGS, BUGS, etc.)

- As the dimensionality and complexity increases, these algorithms can struggle.

Thought: We could use the gradient to quickly move between areas regardless of dimensionality

Hamiltonian Dynamics

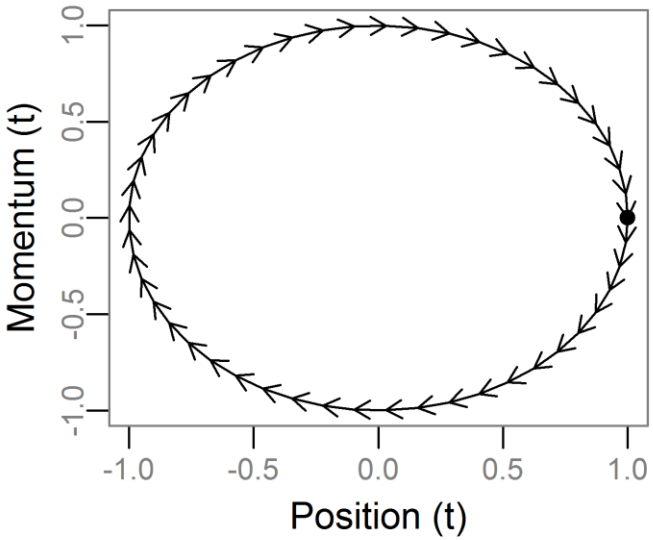
- Imagine a puck moving on a frictionless surface
- It has **position** θ with a potential energy U(θ)
- And **momentum** *r*, with kinetic energy K(*r*).
- The Hamiltonian [*H*(θ,*r*)] describes the behavior of the system over time. For MCMC:H=U(θ)+K(*r*)

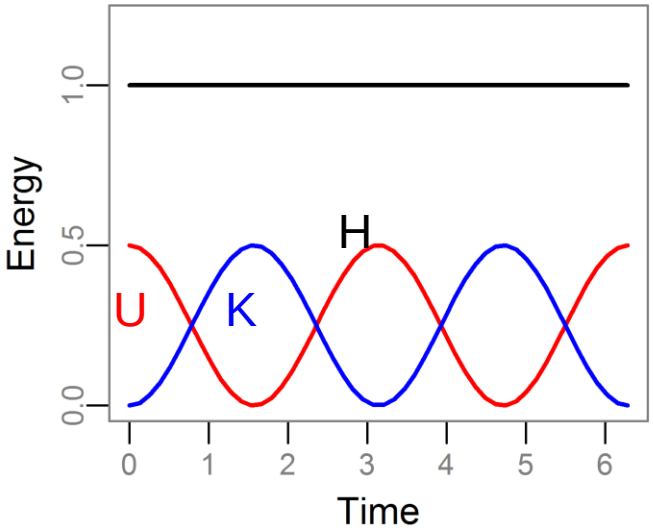
$$\frac{d\theta_i}{dt} = \frac{\delta H}{\delta r_i} = \frac{dK}{dr_i}; \quad \frac{dr_i}{dt} = \frac{\delta H}{\delta \theta_i} = -\frac{dU}{d\theta_i}$$

Trivial to calculate

Derivative of log-posterior

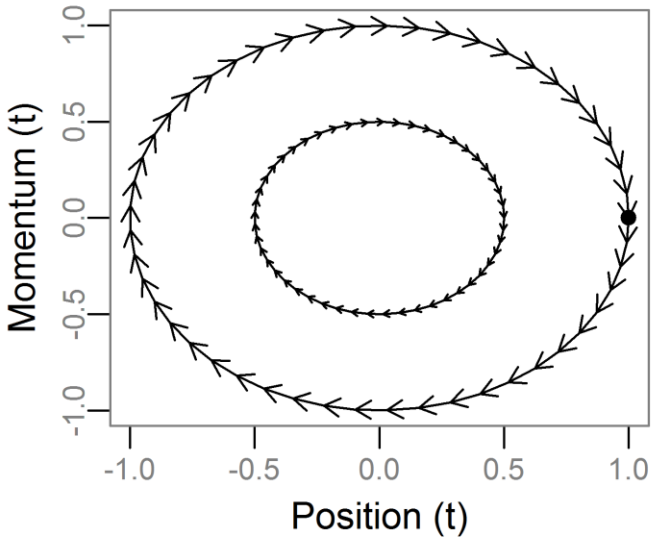
Hamiltonian Dynamics: Example

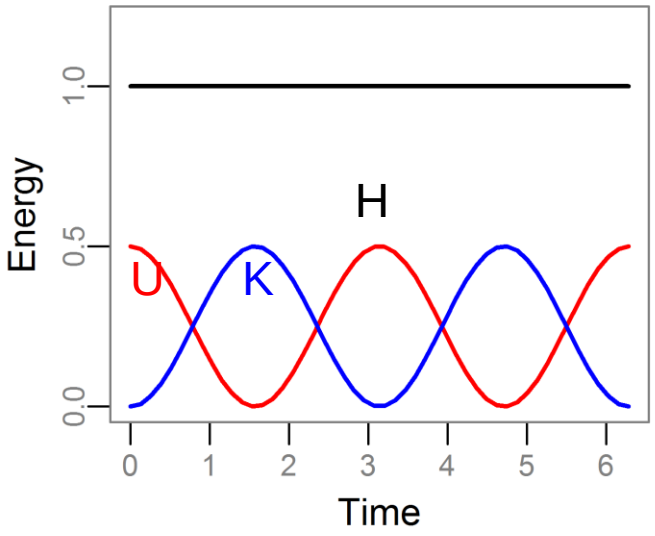




- See Neal (2010) for good review
- For MCMC we set U=log posterior and K=log N(0,Σ)
- Take a 1d example where:
  - $U=\theta^2/2$ [$\theta \sim N(0,1)$]
  - $K=r^2/2$ [$r \sim N(0,1)$]
- We can solve these equations analytically
- Note:
  - *H* is constant over time
  - Each *r* is a different contour
  - Most systems are not solvable

Hamiltonian Dynamics: Example





- See Neal (2010) for good review
- For MCMC we set U=log posterior and K=log N(0,Σ)
- Take a 1d example where:
  - $U=\theta^2/2$ [$\theta \sim N(0,1)$]
  - $K=r^2/2$ [$r \sim N(0,1)$]
- We can solve these equations analytically
- Note:
  - *H* is constant over time
  - Each *r* is a different contour
  - Most systems are not solvable

Static Hamiltonian Monte Carlo

1. Draw r~MVN(0,Σ) (Σ [1] is unit diagonal)
2. Project forward[2] L discrete steps of size ε.
3. The final value of trajectory is our **proposed value** (*q*!!).

- Note:
  - H varies due to discretization, so use RWM step:

$$\theta_{t+1} = \theta^* \text{ if runif}(1) \leq \exp[H(\theta\ ,r\ ) - H(\theta^*, r^*)]$$

  - This generates joint samples (θ,r), so we discard (ignore) the *r* samples.

[1] This is known as the "mass matrix"

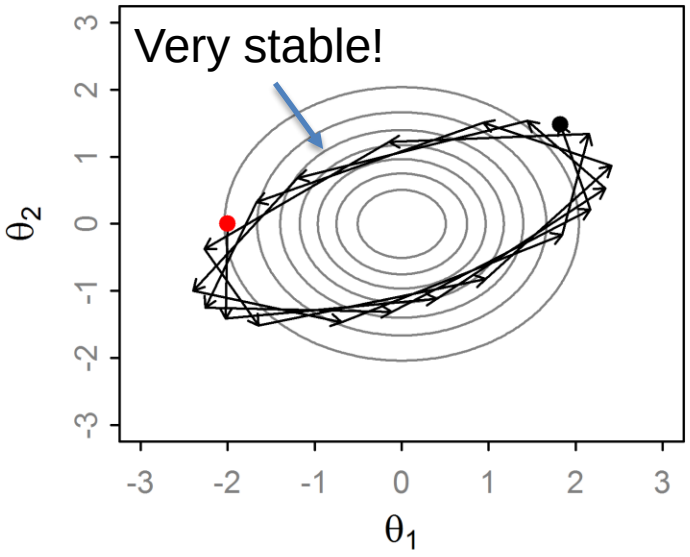
[2] Using the Leapfrog integrator which is more stable/robust than Euler's method

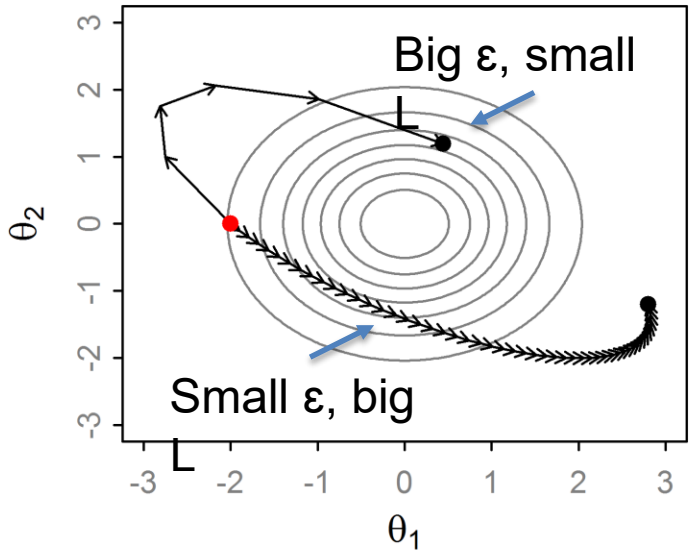
Hamiltonian Monte Carlo

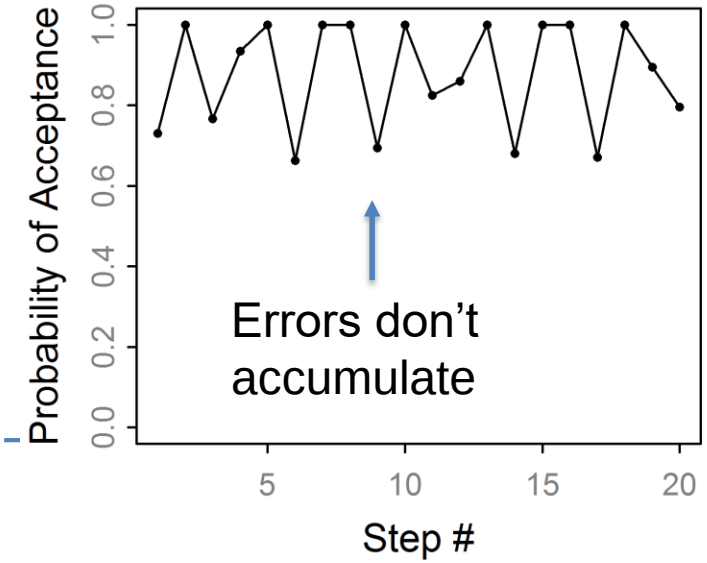
- Q: Why do we need to utilize a Hamiltonian system?
- A: Detailed balance!
- HMC has several mathematical properties advantageous for MCMC:
  - Reversible + Volume preserving.
  - Informally: the q cancels out. Impossible to calculate otherwise.
- Crucially, these hold under discretization
- Bottom line:

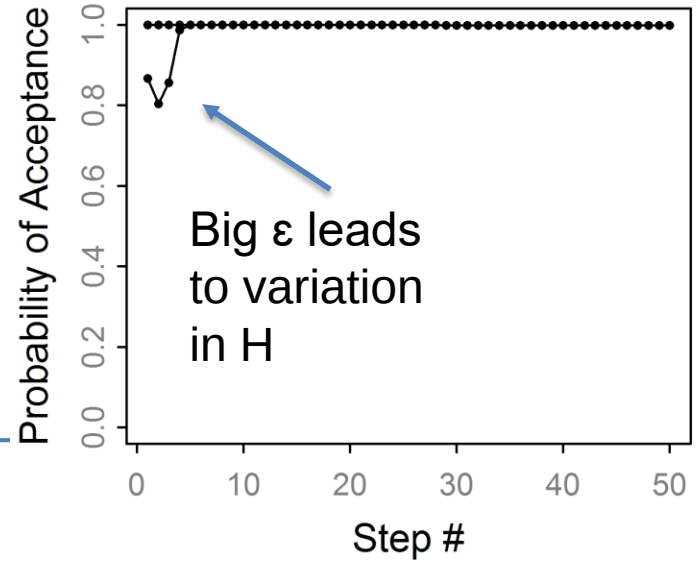
  The chain gives us samples from the posterior

HMC: Example trajectories

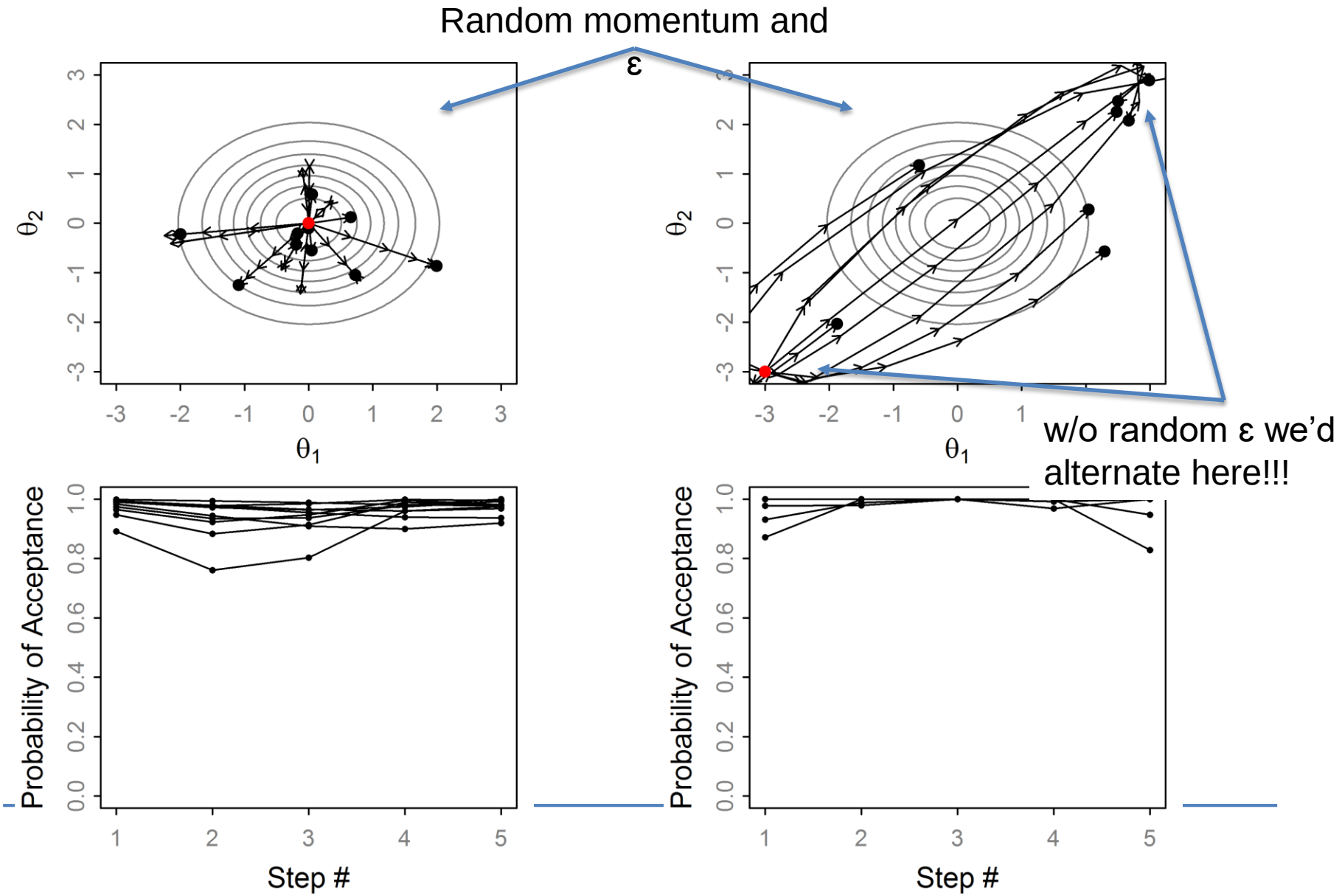




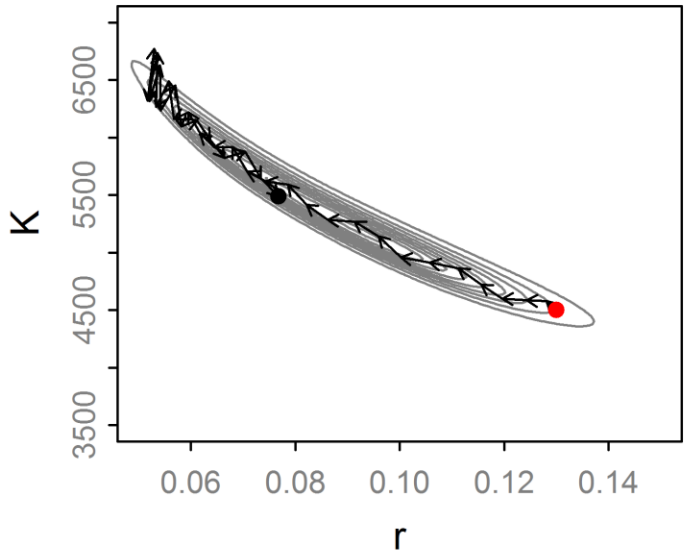


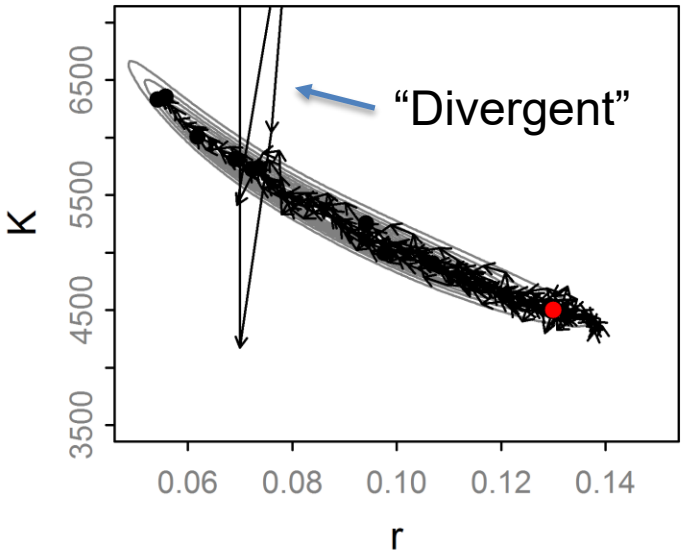


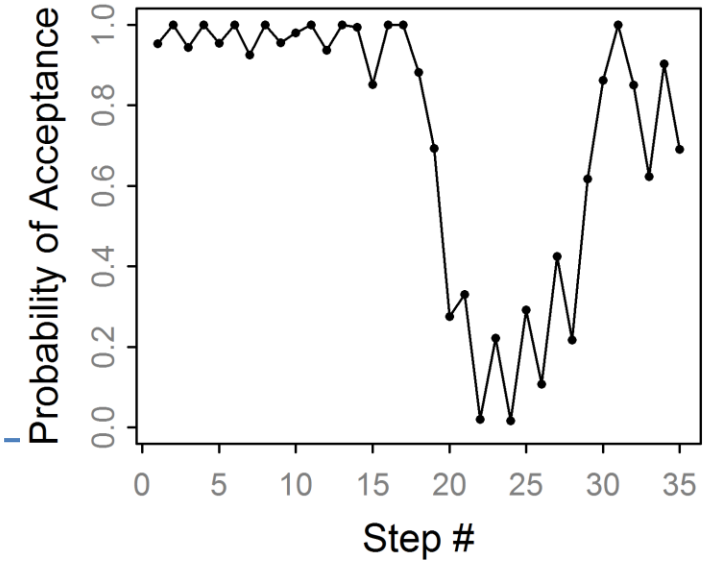
Effect of random momentum

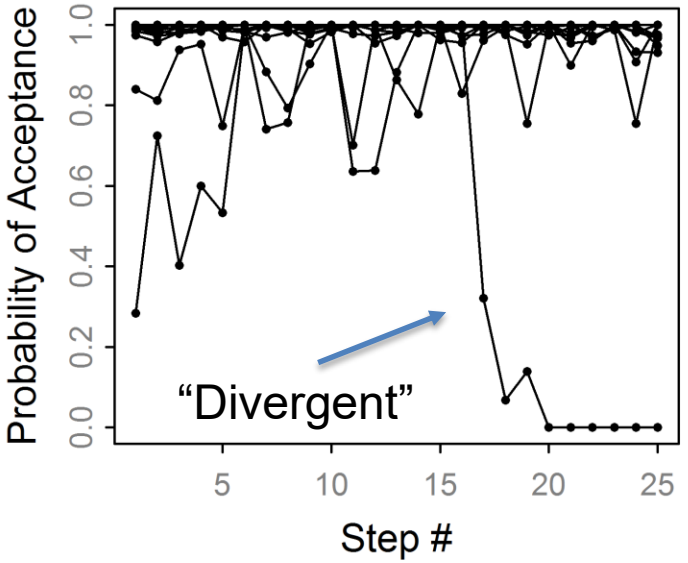


HMC: Example trajectories

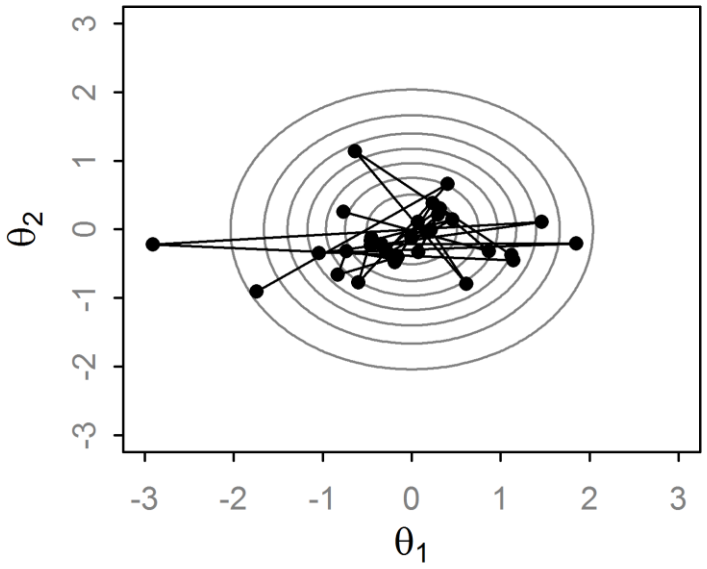


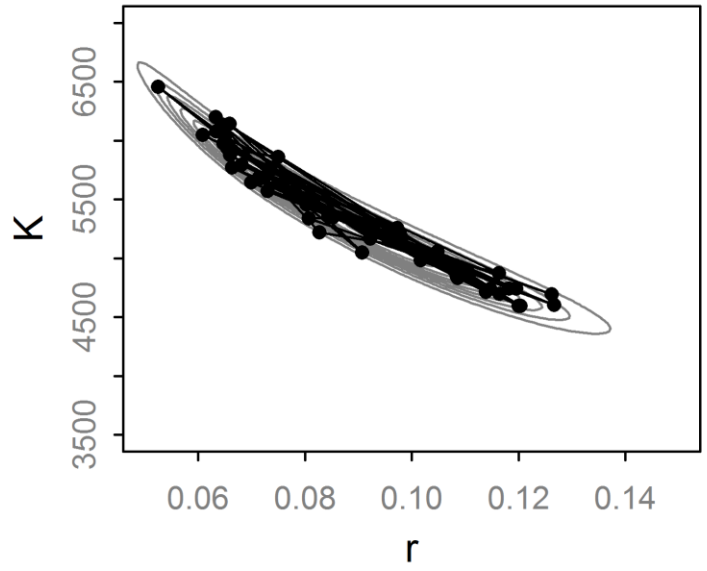






Hamiltonian Monte Carlo





- HMC eliminates inefficient random walk behavior
- Fancy way to propose values
- Often produces nearly independent samples (for large L)
- Has high computational cost (L ≈ to thinning)

Implementation Hurdles of HMC

- Introduced by Duane et al. (1987)… why uncommon?
- Some in the physics/stats literature[1], but it "*seems to be under-appreciated by statisticians*" (Neal, 2010).

Mainly for two reasons:

1. **Hard to calculate derivatives of log posteriors**
2. **Efficiency is notoriously sensitive to the tuning parameters: (L, ε, Σ)**

[1] e.g., Neal (1996), Ishwaran (1999) and Schmidt (2009)

Solution #1: Automatic Differentiation

- AD is a numerical technique to get precise derivative of any continuous function.
- The computer applies the chain rule successively
- It is as precise as analytical derivatives up to computer precision.
- Available widely, e.g., ADMB, TMB, Stan
- Posterior must be continuously differentiable

Solution #2: No-U-Turn Sampler

- **Extends HMC to avoid specifying L and ε.**
- ε is adapted with ‘dual averaging’. Works for HMC too. Skipping this...
- L is set automatically with a sophisticated algorithm that detects a “U-turn” in the trajectory and stops.
- Thus L varies at each iteration, avoiding wasteful steps.

Hoffman and Gelman (2011)

No-U-Turn Trajectory

for j in 0:max_depth

Pick random direction (left or right)

Recursively build tree of size 2^j

If U-turn occur in subtree or divergence

break, excluding subtree

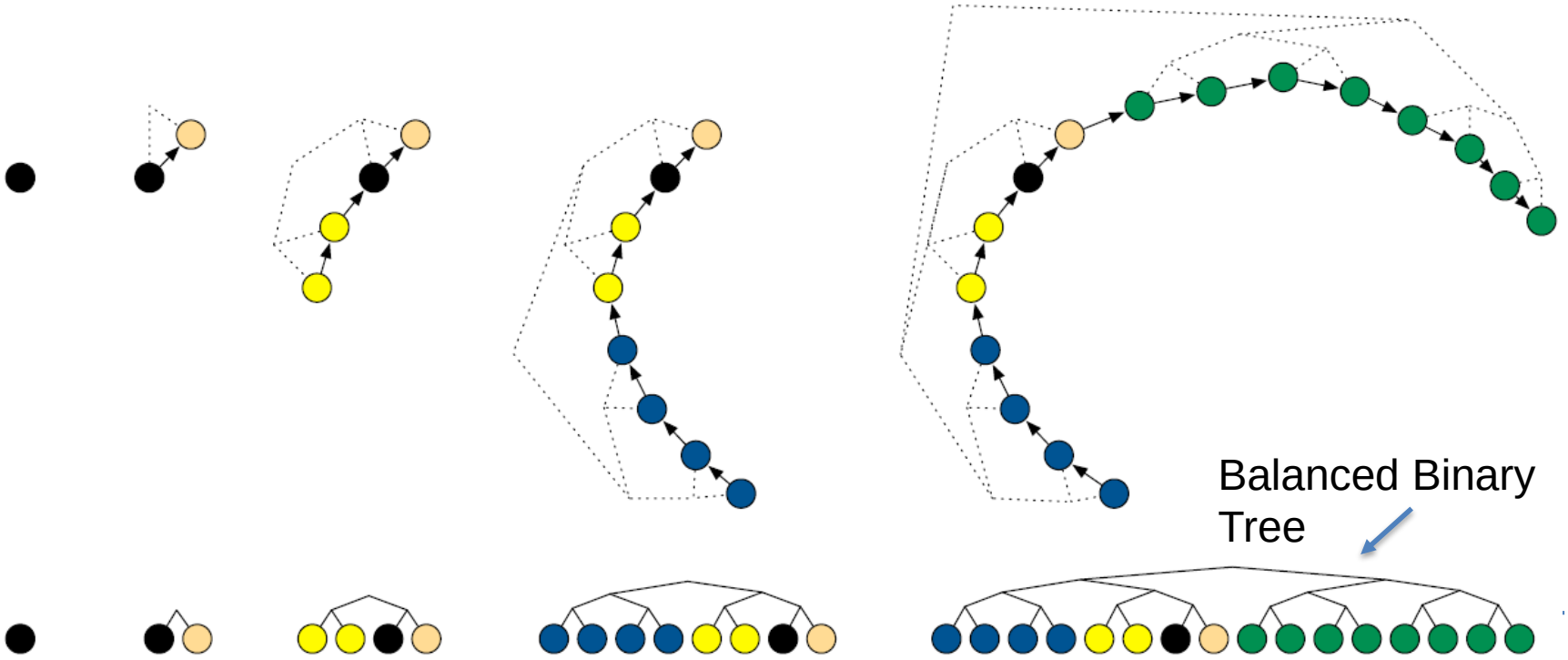


Fig 1, Hoffman and Gelman (2011)

No-U-Turn Example

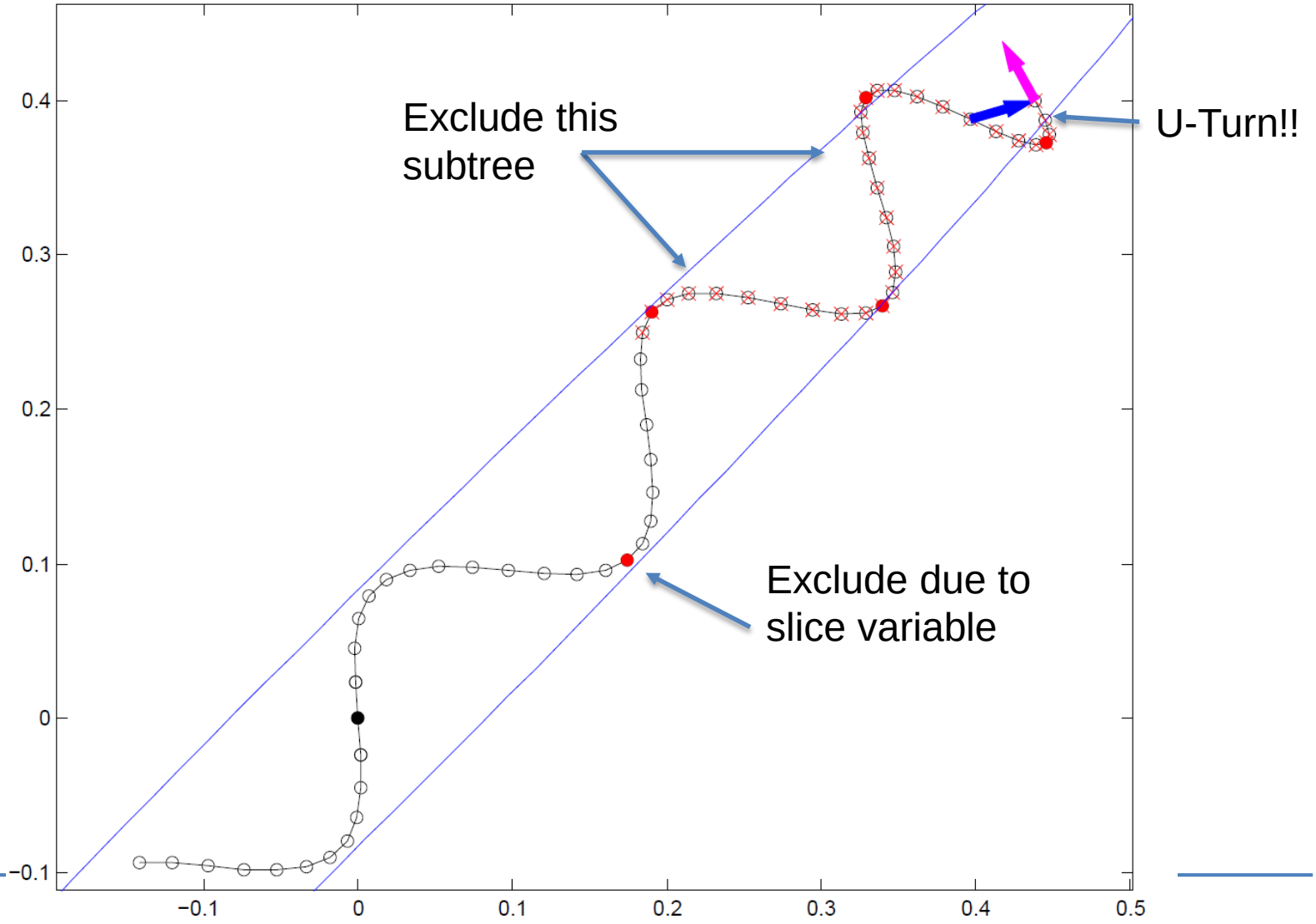


Fig 2, Hoffman and Gelman (2011)

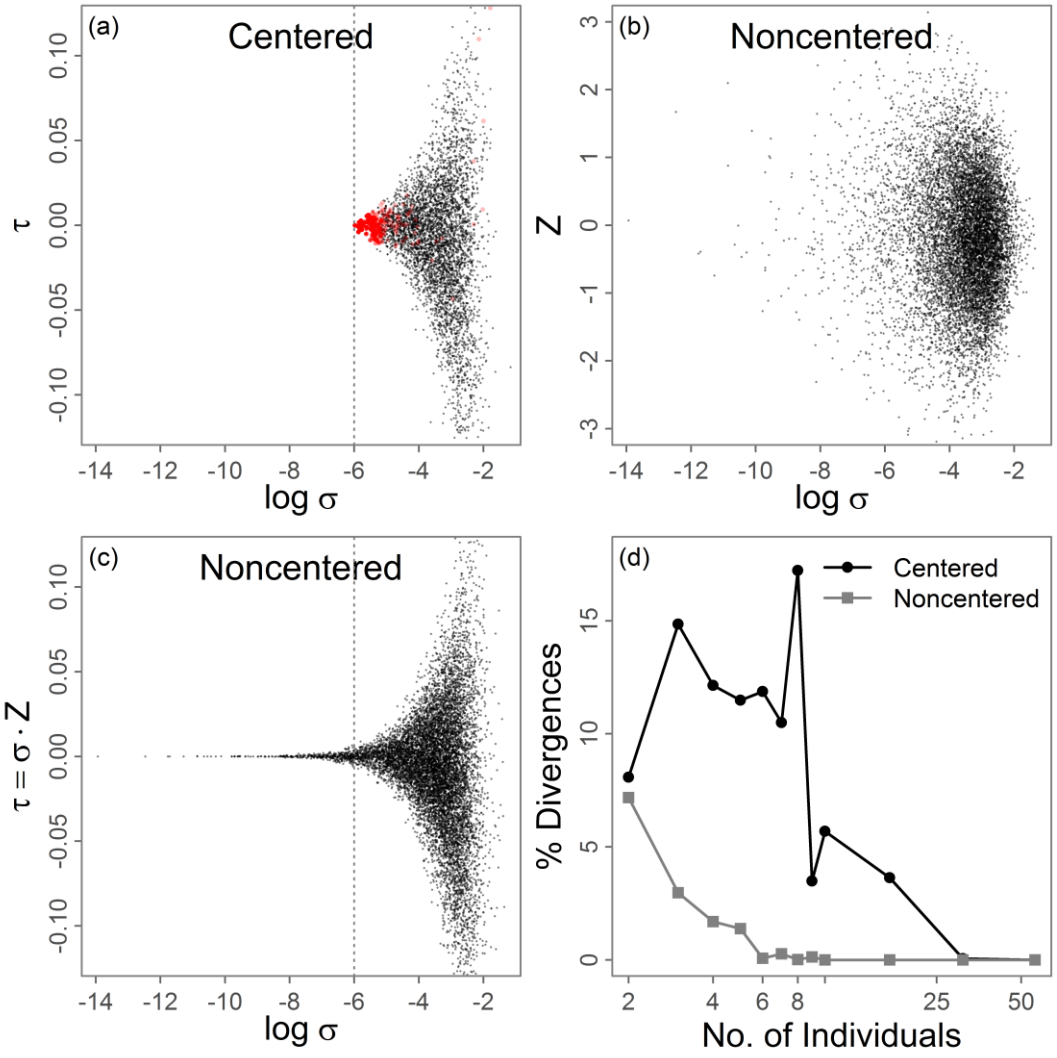
Tuning the No-U-Turn Sampler

- Eliminates the need to specify ε or L: ε is tuned during the warmup phase, L dynamically.
- But, introduces new tuning parameters:
  - max_treedepth=12: Maximum tree depth.
  - Delta=0.8: The target acceptance rate (increase toward 1 as needed)
- However, this seems to work smoothly without intervention (good for general use)

Divergent transitions indicate issues

- Divergences occur when a trajectory hits a region of high curvature and the total energy goes to infinity
- This diagnostic tells us the MCMC sampler may be biased
- Try rerunning with a higher adapt_delta
- Or reparametrize if possible

Divergent transitions indicate issues



- This is a non-linear mixed effects growth model
- There are two ways to parametrize the random effects: ‘centered’ and ‘non-centered’
- The non-centered version has divergences and bias
- Non-centering fixes this and is a recommended solution

Concluding thoughts

- HMC/NUTS are extremely sophisticated and powerful MCMC algorithms
- A basic understanding helps interpret and diagnose output
- Stan is replacing JAGS as a generic platform
- Stan's divergences warning of bias (good)

Advice: JAGS is good starting place. Switch to Stan and gradient-based MCMC if needed.

References

- Monnahan, C. C., J. T. Thorson, and T. A. Branch. 2017. Faster estimation of Bayesian models in ecology using Hamiltonian Monte Carlo. Methods in Ecology and Evolution **8:339-348.**
- Papaspiliopoulos, O., G. O. Roberts, and M. Skold. 2007. A general framework for the parametrization of hierarchical models. Statistical Science **22:59-73.**
- Betancourt, M., and M. Girolami. 2015. Hamiltonian Monte Carlo for hierarchical models. Current Trends in Bayesian Methodology with Applications:79.
- Hoffman, M. D., and A. Gelman. 2014. The no-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. Journal of Machine Learning Research **15:1593-1623.**
- Carpenter, B., A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell. 2017. Stan: A Probabilistic Programming Language. Journal of Statistical Software **76:1-29.**