STSCI 4780
Bayesian computation — Beyond the basics
(A selective survey)

Tom Loredo, CCAPS & SDS, Cornell University

Notation

$$
\begin{aligned}
p(\theta|D, M) &= \frac{p(\theta|M)p(D|\theta, M)}{p(D|M)} \\
&= \frac{\pi(\theta)\mathcal{L}(\theta)}{Z} = \frac{q(\theta)}{Z}
\end{aligned}
$$

- $M =$ model specification
- D specifies observed data
- $\theta =$ model parameters
- $\pi(\theta) =$ prior pdf for θ
- $\mathcal{L}(\theta) =$ likelihood for θ (likelihood function)
- $q(\theta) = \pi(\theta)\mathcal{L}(\theta) =$ "quasiposterior"
- $Z = p(D|M) =$ (marginal) likelihood for the model

Key themes in advanced algorithms

- Combining multiple update algorithms
- Adaptation—*gently* breaking the Markov property
- Augmenting the parameter space (*increasing* dimensionality)

Combining MH updates

No one class of proposal distributions works well for all problems $\rightarrow$ consider combining multiple proposals hoping they'll have complimentary strengths (esp. in a "black box" toolkit)

Two valid ways to combine reversible updates

- Composing updates: follow one update by another
- Mixing updates: randomly choose an update mechanism

Implementations

- Fixed/cyclic scan (or sweep)
- Random scan
- Random sequence scan—combines composition and mixing

For theory & examples, see Geyer's 1995 and 1998 MCMC notes

Adaptive MCMC

A proposal distribution for MH sampling typically has *tuning parameters*, ψ: we draw a candidate from $k_\psi(y; x)$.

- Random-walk Metropolis: Proposal width in each direction
- Independent Metropolis: Shape of proposal (location, covariance. . .)

For MH, we can't have ψ depend on the chain history—the chain wouldn't be Markov!

Simple approach: We can tune ψ using pilot runs (perhaps during burn-in), and then fix it to preserve detailed balance

Adaptive MCMC finds ways to adjust ψ continuously that preserves asymptotic sampling properties

Main idea: *Vanishing adaptation*

Example: Robust adaptive Metropolis (RAM)

Motivation: Consider random-walk metropolis (RWM), but with a proposal distribution that is multivariate normal, so it can take steps along directions aligned with the posterior

This requires:

- Finding a good covariance matrix for the MVN
- Drawing a vector of correlated steps from a MVN

MVN draws: Write the covariance matrix as $C = SS^T$, where S is the *Cholesky factorization* of C — a lower-diagonal matrix with positive elements

Then from current position X_{n-1}, we can propose a candidate position Y_n by drawing a vector U_n of *independent* standard normal variates, and shifting and correlating them:

$$Y_n = X_{n-1} + SU_n$$

Now the challenge is choosing S

RAM algorithm

Use Metropolis updates with a correlated multivariate proposal, *altering the covariance matrix along the chain to target a desired mean acceptance rate,* α_*:

1. Propose $Y_n = X_{n-1} + S_{n-1}U_n$, where $U_n \sim q$ is an independent random vector, and S_{n-1} is a lower-diagonal matrix with positive elements
2. With probability $\alpha_n \equiv \min\{1, \pi(Y_n)/\pi(X_{n-1})\}$ the step is accepted, and $X_n = Y_n$; otherwise the step is rejected and $X_n = X_{n-1}$
3. Compute an updated lower-diagonal matrix S_n via

$$S_n S_n^T = S_{n-1} \left(I + \eta_n(\alpha_n - \alpha_*) \frac{U_n U_n^T}{\| U_n \|^2} \right) S_{n-1}^T \tag{1}$$

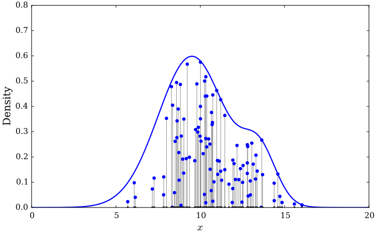
where I is an identity matrix, and $\eta_n = n^{-2/3}$ controls the adpativity

See: Vihola (2012): Robust adaptive Metropolis algorithm with coerced acceptance rate

Auxiliary/augmented variables

The accept/reject method for sampling a d-D density:

- Sample from a *uniform* $(d+1)$-D density (with a complicated boundary):



- Report the marginal samples for the d original dimensions

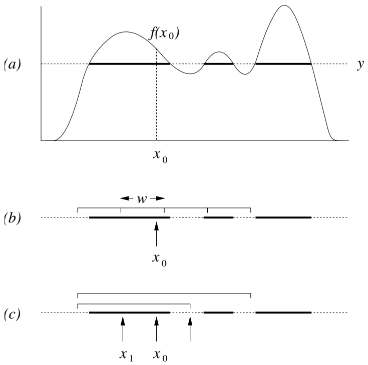
A paradoxical notion motivating some advanced MCMC methods is that making the problem "harder" (higher-dimensional) may actually make it *easier*

Slice Sampling

Add a vertical dimension (like rejection), and make a chain that samples uniformly under $p(\theta)$:

- Sample y uniformly over $[0, p(\theta_i)]$ (y given θ)
- Sample θ_{i+1} from dist'n for θ given y

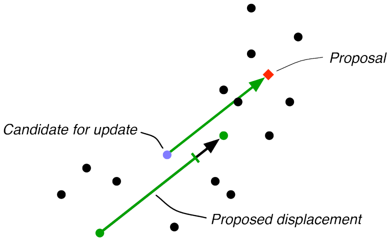
Latter is done sampling uniformly over $\{\theta : y < p(\theta)\}$



Differential Evolution MCMC

Combine evolutionary computing & MCMC (ter Braak 2006)

Follow a *population* of states, where a randomly selected state is considered for updating via the (scaled) vector difference between two other states.



Behaves roughly like RWM, but with a proposal distribution that automatically adapts to shape & scale of posterior

Step scale: Optimal $\gamma \approx 2.38/\sqrt{2d}$, but occassionally switch to $\gamma = 1$ for mode-swapping

Original DE-MCMC uses these simple moves and pop'n size $N \sim 3d$; works well if given a "smart start"

Later version (ter Braak & Vrugt 2008) adds new moves and can sample effectively with just $N = 3$ in up to a few dozen dimensions, without a smart start

Random Walks

Metropolis random walk (MRW) and Gibbs sampler updates execute a *random walk* through parameter space:

- Moves are local, with a characteristic scale l
- Total distance traversed over time $t \propto \sqrt{t}$

This is a relatively slow (albeit steady) rate of exploration

Multimodality $\rightarrow$ even slower exploration; only rare large jumps can move between modes

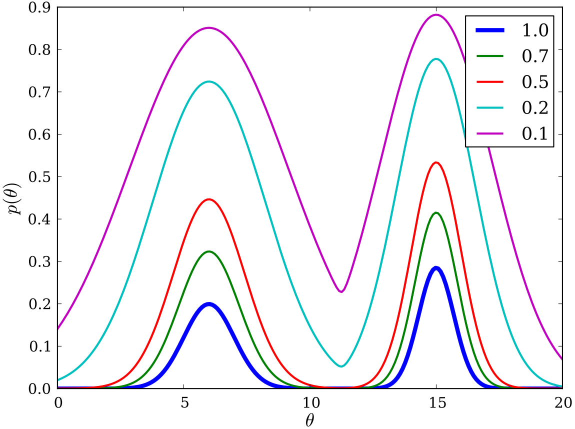
We need methods designed to make large moves

Annealing and Parallel Tempering

PT, aka Metropolis-coupled MCMC

To enable large jumps, *anneal* or *temper* the posterior:

$$q_\beta(\theta) = [q(\theta)]^\beta, \qquad \beta \in [0,1]$$



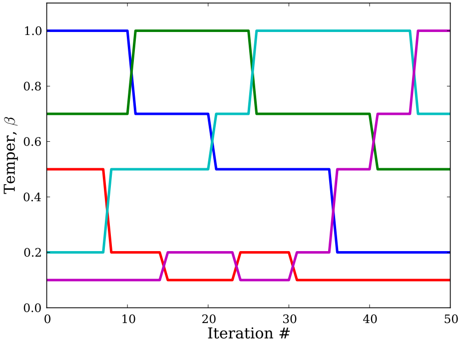
Consider a set of tempers ("inverse temperatures") $\{\beta_i\}$

Think of each $q_i = q_{\beta_i}$ as its own "model" with its own parameters, and construct a sampler for the joint distribution

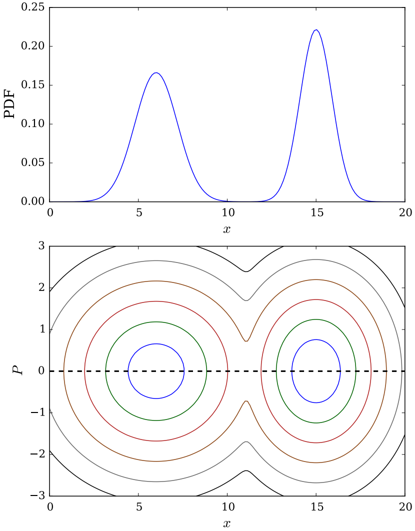
$$p(\theta_1, \ldots, \theta_m) = \prod_i q_i(\theta_i)$$

Alternate within-temper proposals and swap proposals between adjacent tempers

Swaps between tempered chains



Phase space: Doubling the dimensionality



$$p(x, P) \propto q(x) \times f(P)$$

$$p(x) = \int dP \, p(x, P) \propto q(x)$$

$$p(P) = \int dx \, p(x, P) \propto f(P)$$

- Pick $P \sim f(P)$
- Move along a contour in phase space
- Drop P, keep x

Will work if the phase space motion corresponds to sampling $p(x, P)$

Hamiltonian (Hybrid) Monte Carlo

Give samples "momentum" so moves tend to go in the same direction a while; use derivatives to guide the evolution $\rightarrow$ suppress random walks

Adds d additional variables, P, with a joint Gaussian dist'n:

$$\log p(\theta, P) = -\left[U(\theta) + \frac{1}{2}P^2\right]; \qquad U(\theta) \equiv -\log q(\theta)$$

Sample P from a Gaussian, and use it to generate proposals via

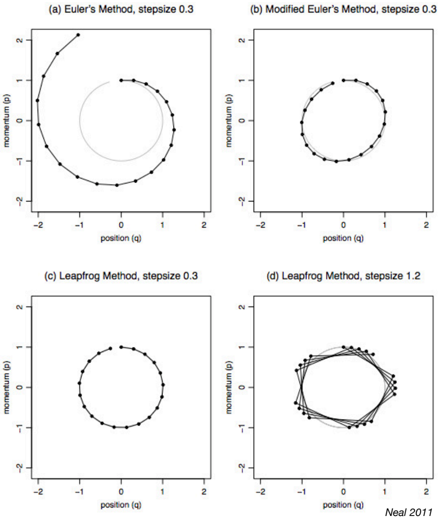
$$\dot{\theta} = P; \qquad \dot{P} = -\frac{\partial H}{\partial \theta}$$

Hamiltonian dynamics $\rightarrow$ reversible, preserves volume, keeps p constant (*exact* proposals always accepted, like Gibbs sampling)

Challenges for basic HMC

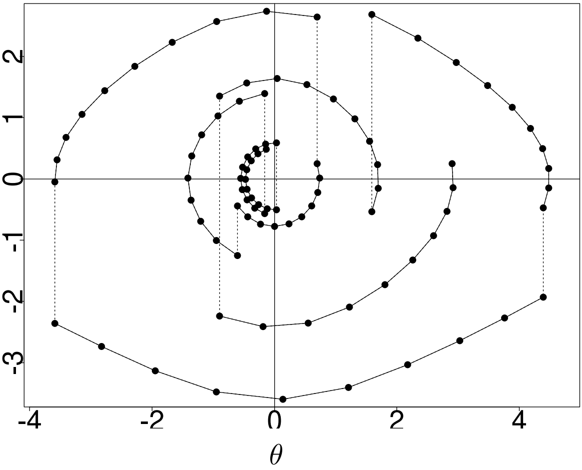
- Tuning parameters:
  - PDE integration time step size, ϵ, and integration length, L
  - Handling problems with very different scales along different dimensions ($\rightarrow$ need different momentum scales)
- Computing the needed derivatives

Numerical integration (1-D)



Introduce a M-H accept/reject step to account for integration error

Sampling a 1-D Student-t dist'n with dof$= 5$



HMC vs. random walk (2-D)

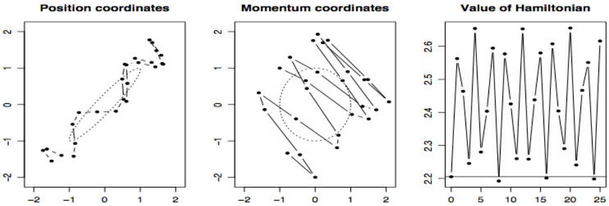
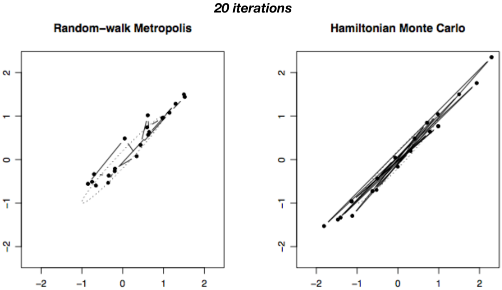


Figure 3: A trajectory for a 2D Gaussian distribution, simulated using 25 leapfrog steps with a stepsize of 0.25. The ellipses plotted are one standard deviation from the means. The initial state had $q = [-1.50, -1.55]^T$ and $p = [-1, 1]^T$.



Neal 2011

Tuning integration length

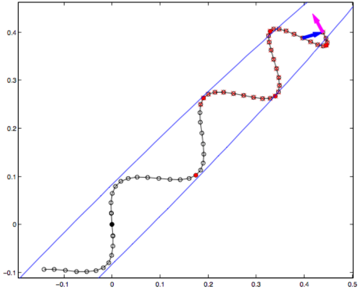
We want to move along a contour long enough to get far from the starting point, but not head back toward it

Examine

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{(\theta-\theta_i)\cdot(\theta-\theta_i)}{2} = (\theta-\theta_i)\cdot P$$

Stop integrating when this becomes negative

No-U-Turn Sampler (NUTS)

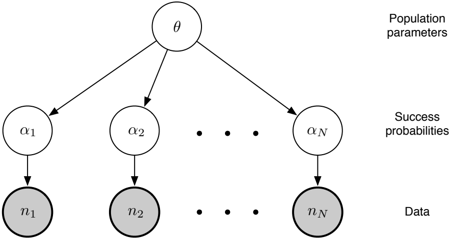


Hoffman & Gelman 2013

Multilevel models: parameter-dependent scales

Goal: Learn a population dist'n from noisy member measurements

Qualitative



$$p(\theta, \{\alpha_i\}, \{n_i\}) = \pi(\theta) \prod_i p(\alpha_i|\theta)\; p(n_i|\alpha_i)$$
$$= \pi(\theta) \prod_i p(\alpha_i|\theta)\; \ell_i(\alpha_i)$$

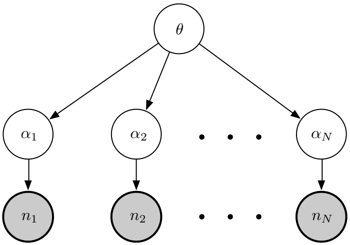
Quantitative

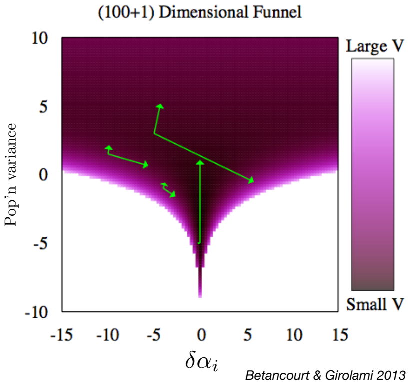
$$\theta = (a, b) \text{ or } (\mu, \sigma)$$

$$\pi(\theta) = \mathrm{Flat}(\mu, \sigma)$$

$$p(\alpha_i|\theta) = \mathrm{Beta}(\alpha_i|\theta)$$

$$p(n_i|\alpha_i) = \binom{N_i}{n_i} \alpha_i^{n_i} (1 - \alpha_i)^{N_i - n_i}$$





Betancourt & Girolami 2013

Mass matrix = metric

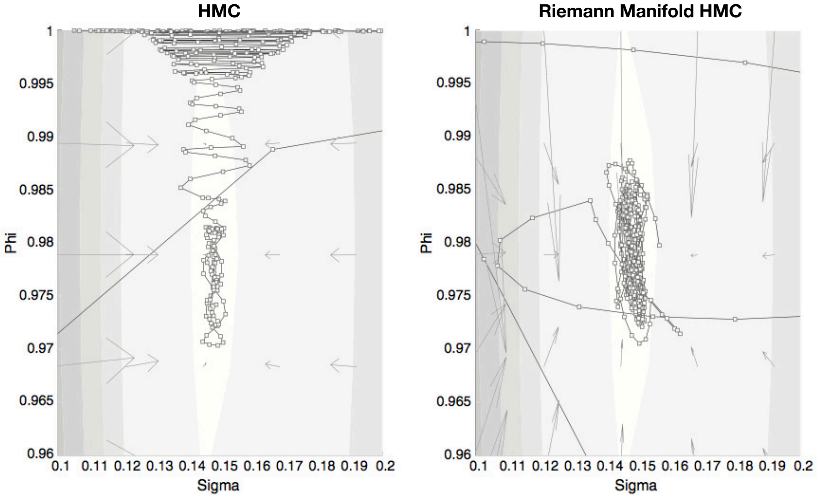
Add d additional variables, P, with a *correlated* Gaussian dist'n:

$$\log p(\theta, P) = -\left[U(\theta) + \frac{1}{2} P \cdot M^{-1} \cdot P\right]; \qquad U(\theta) \equiv -\log p(\theta)$$

M introduces d more tuning parameters!

- **Euclidean manifold HMC:** Use the Hessian at the mode
- **Riemannian manifold HMC:** Use position-dependent $M(\theta)$

HMC vs. Riemann manifold MC



Girolami & Calderhead 2011

Stan capabilities

- High-performance probabilistic model implementation
  - Stan code is compiled to a C++ library
  - Parameters transformed to unconstrained space; transformation & Jacobian handled automatically
  - Automatic differentiation (AD) used to compute derivatives of log-likelihood WRT parameters
- HMC No-U-Turn Sampler (NUTS)
  - PDE solver step size & number automatically tuned during burn-in
  - Mass matrix adaptively tuned during burn-in
- Optimization
  - BFGS and Newton's method
- Ongoing development — RMHMC, ensemble samplers in progress