Markov chain Monte Carlo

— Markov chain Monte Carlo (MCMC)
— Gibbs and Metropolis–Hastings
— Slice sampling
— Practical details

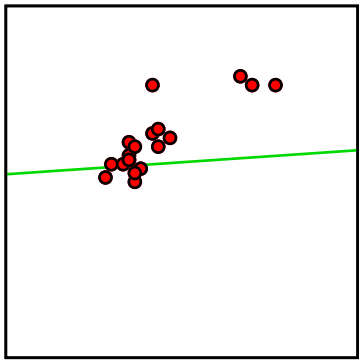
Iain Murray
http://iainmurray.net/

Reminder

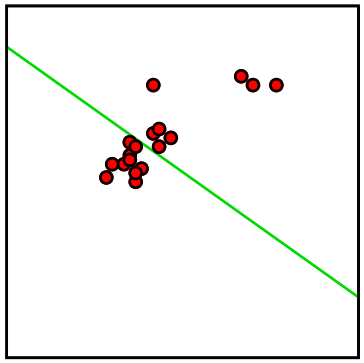
Need to sample large, non-standard distributions:

$$P(x|\mathcal{D}) \approx \frac{1}{S}\sum_{s=1}^{S} P(x|\theta), \quad \theta \sim P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)\,P(\theta)}{P(\mathcal{D})}$$

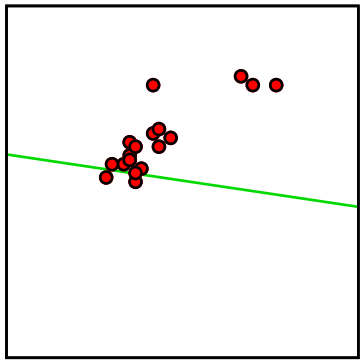
Importance sampling weights



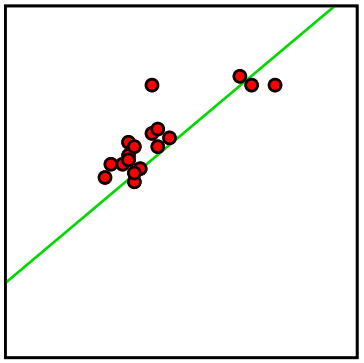
w =0.00548



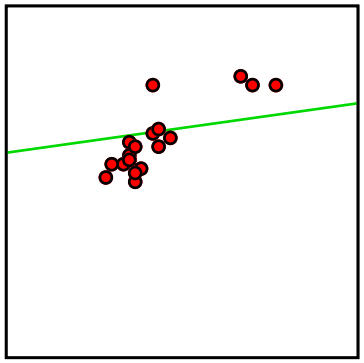
w =1.59e-08



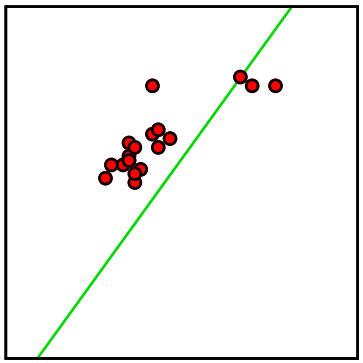
w =9.65e-06



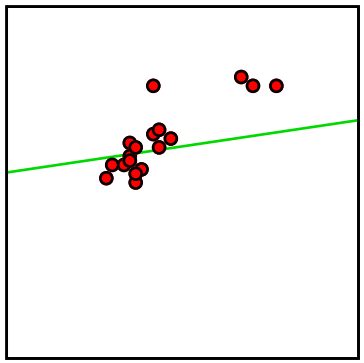
w =0.371



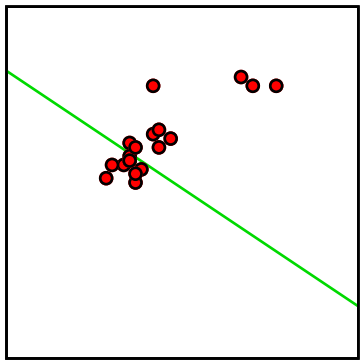
w =0.103



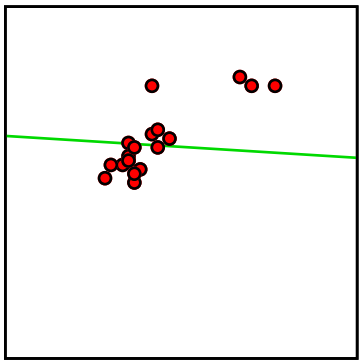
w =1.01e-08



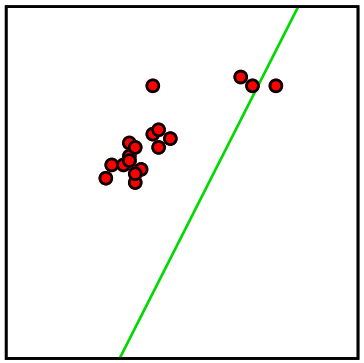
w =0.111



w =1.92e-09

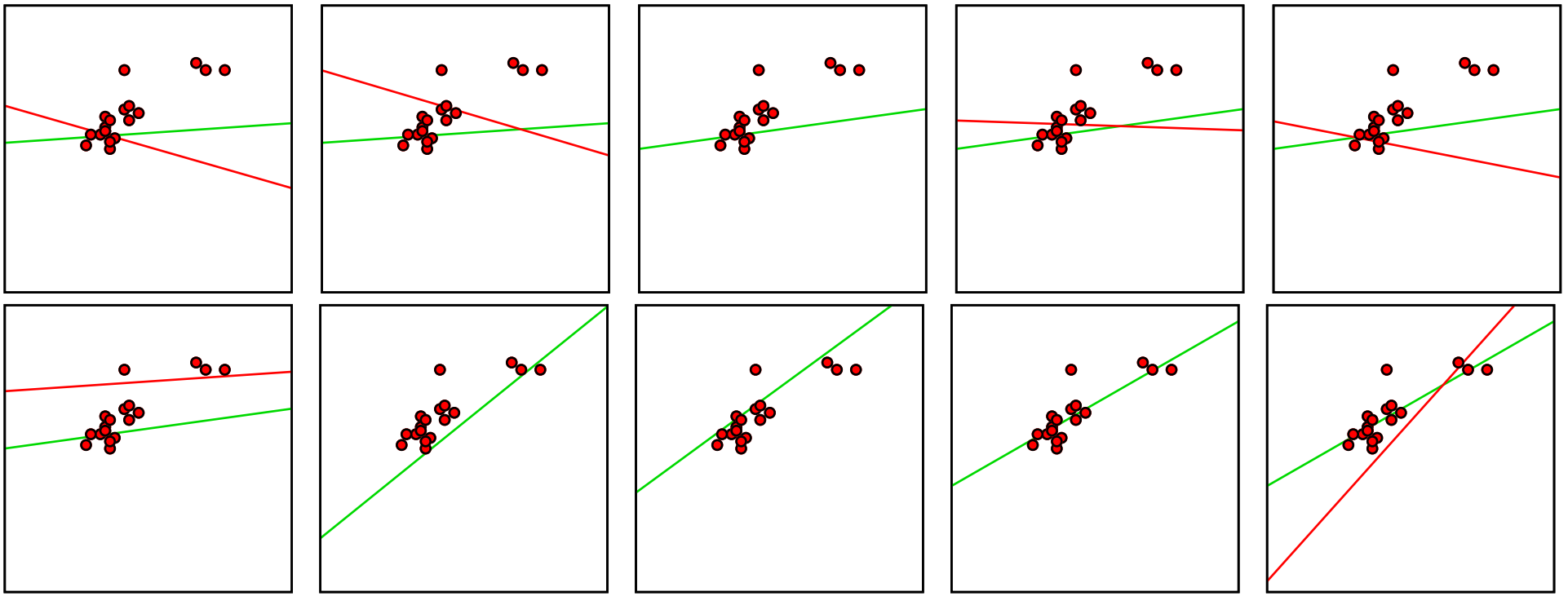


w =0.0126

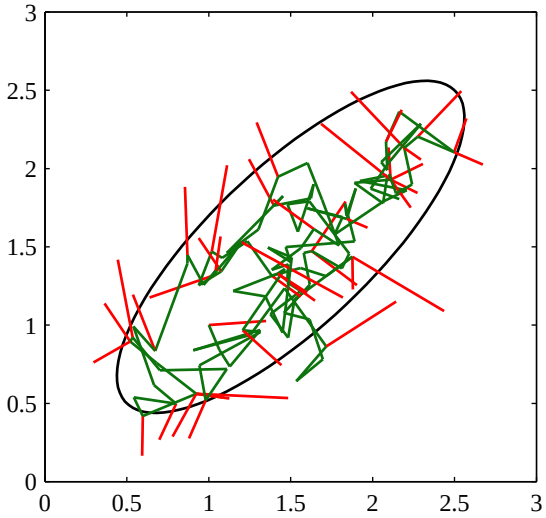


w =1.1e-51

Metropolis algorithm



- Perturb parameters: $Q(\theta'; \theta)$, e.g. $\mathcal{N}(\theta, \sigma^2)$
- Accept with probability $\min\left(1, \dfrac{\tilde{P}(\theta'|\mathcal{D})}{\tilde{P}(\theta|\mathcal{D})}\right)$
- Otherwise **keep old parameters**



This subfigure from PRML, Bishop (2006)

Detail: Metropolis, as stated, requires $Q(\theta'; \theta) = Q(\theta; \theta')$

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

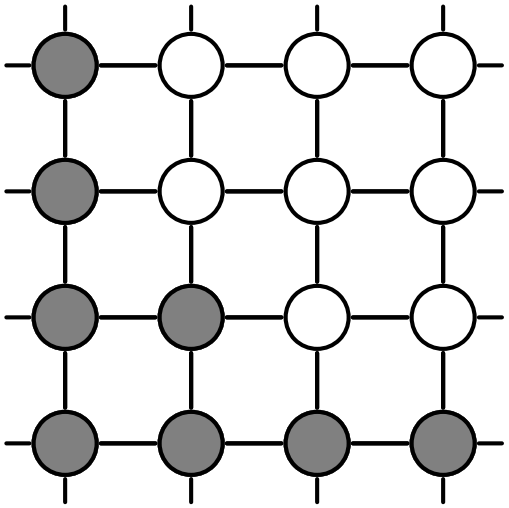
EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*

(Received March 6, 1953)

THE purpose of this paper is to describe a general method, suitable for fast electronic computing machines, of calculating the properties of any substance which may be considered as composed of interacting individual molecules. Classical statistics is assumed,

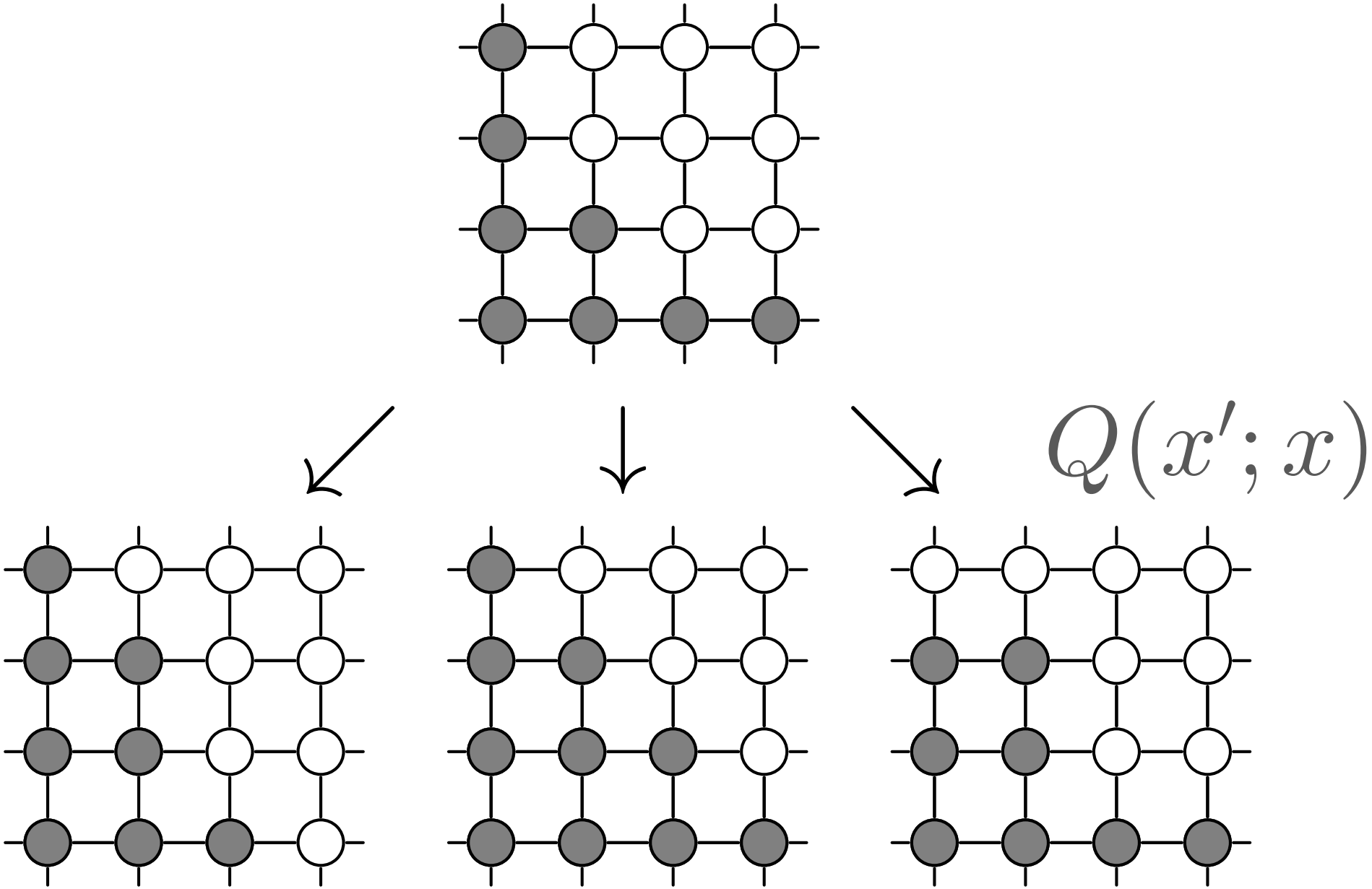
Target distribution

$$P(\mathbf{x}) = \frac{1}{Z} e^{-E(\mathbf{x})}$$

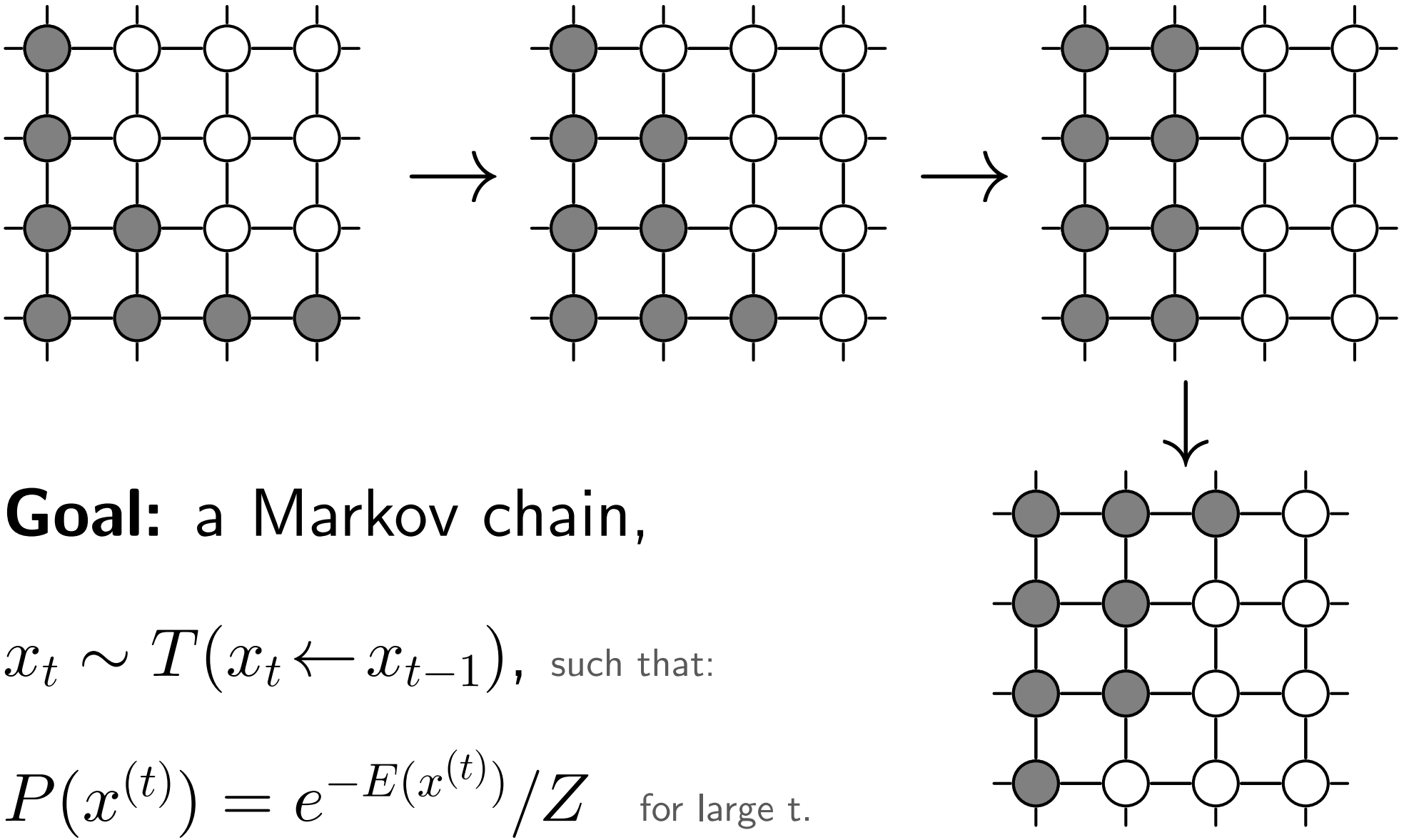
e.g. $\mathbf{x} =$ 

$$E(\mathbf{x}) = \sum_{(i,j) \in \text{edges}} W_{ij} x_i x_j, \quad x_i \in \{\pm 1\}$$

Local moves



Markov chain exploration



Goal: a Markov chain,

$x_t \sim T(x_t \leftarrow x_{t-1})$, such that:

$P(x^{(t)}) = e^{-E(x^{(t)})}/Z$ for large t.

Invariant/stationary condition

If $x^{(t-1)}$ is a sample from P,

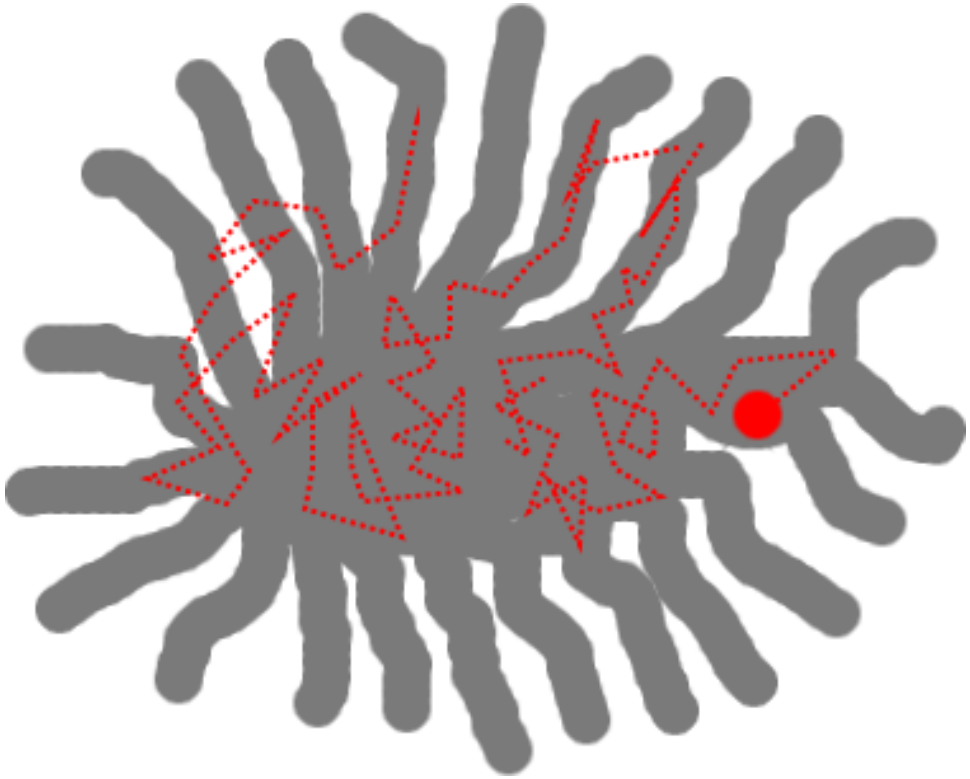
$x^{(t)}$ is also a sample from P.

$$\sum_x T(x' \leftarrow x)\, P(x) = P(x')$$

Ergodicity

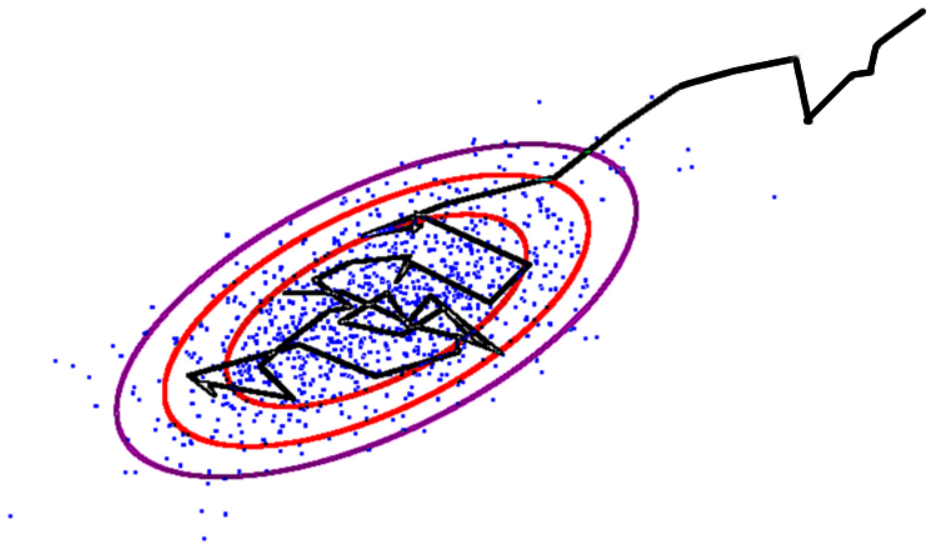
Unique invariant distribution

if 'forget' starting point, $x^{(0)}$



Quick review

MCMC: biased random walk exploring a target dist.



Markov steps,

$$x^{(s)} \sim T\left(x^{(s)} \leftarrow x^{(s-1)}\right)$$

MCMC gives approximate, correlated samples

$$\mathbb{E}_P[f] \approx \frac{1}{S}\sum_{s=1}^{S} f(x^{(s)})$$

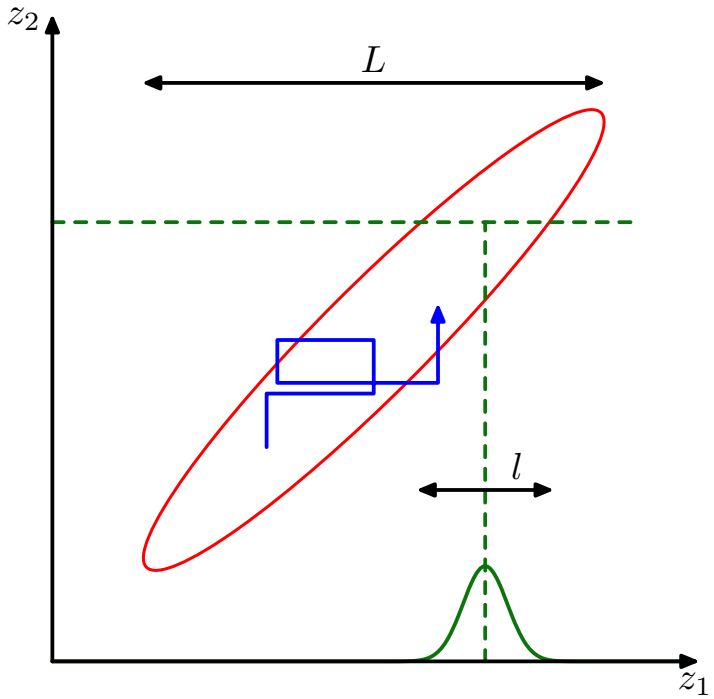
T must leave target invariant

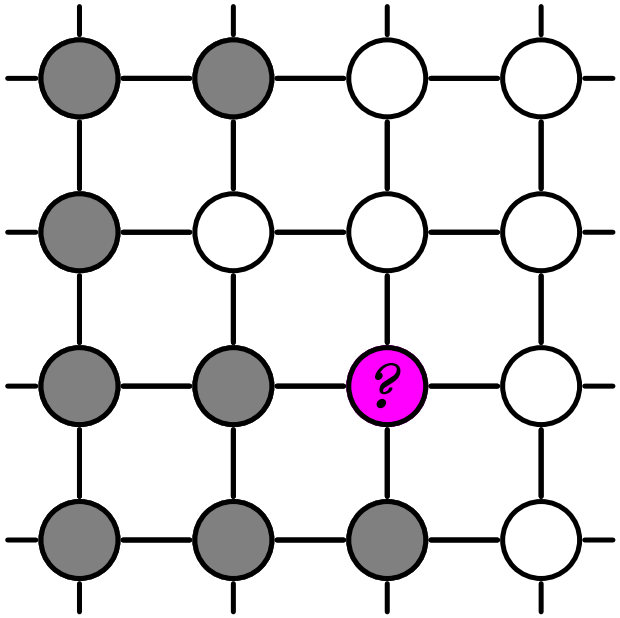
T must be able to get everywhere in K steps (for some K)

Gibbs sampling

Pick variables in turn or randomly,

and resample $P(x_i|\mathbf{x}_{j\neq i})$





$$T_i(\mathbf{x}' \leftarrow \mathbf{x}) = P(x'_i \,|\, \mathbf{x}_{j\neq i})\, \delta(\mathbf{x}'_{j\neq i} - \mathbf{x}_{j\neq i})$$

Gibbs sampling correctness

$$P(\mathbf{x}) = P(x_i \,|\, \mathbf{x}_{\setminus i})\, P(\mathbf{x}_{\setminus i})$$

Simulate by drawing $\mathbf{x}_{\setminus i}$, then $x_i \,|\, \mathbf{x}_{\setminus i}$

Draw $\mathbf{x}_{\setminus i}$: sample $\mathbf{x}$, throw initial x_i away

Reverse operators

If T leaves $P(x)$ stationary, define a *reverse operator*

$$R(x \leftarrow x') = \frac{T(x' \leftarrow x)\, P(x)}{\sum_x T(x' \leftarrow x)\, P(x)} = \frac{T(x' \leftarrow x)\, P(x)}{P(x')}.$$

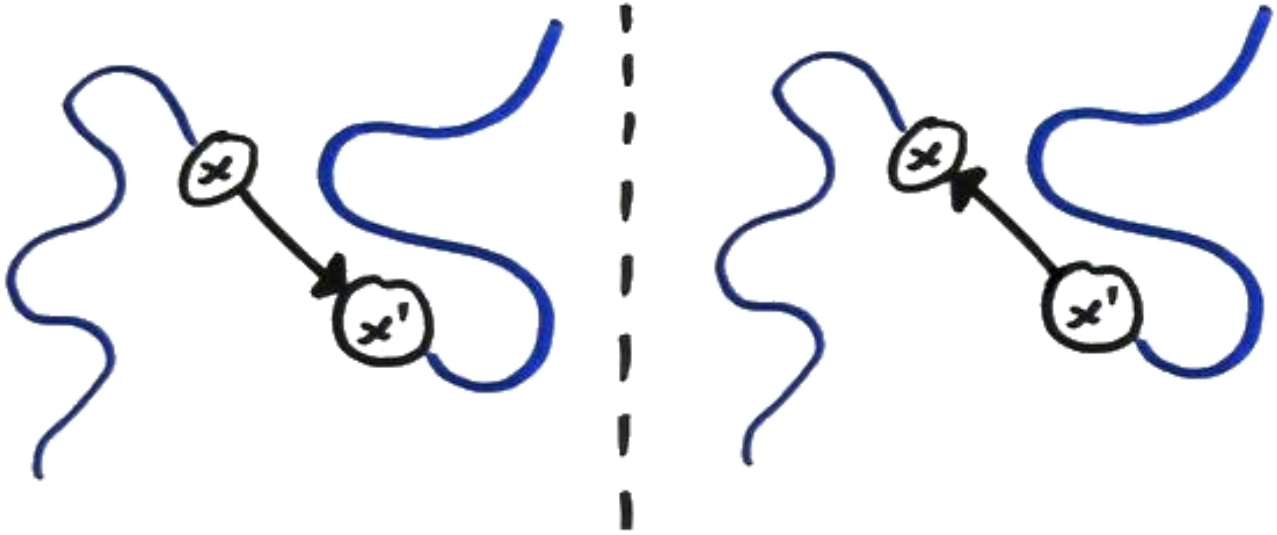
A necessary condition: there exists R such that:

$$T(x' \leftarrow x)\, P(x) \;=\; R(x \leftarrow x')\, P(x'), \qquad \forall x, x'.$$

If $R = T$, known as **detailed balance** (not necessary)

Balance condition

$$T(x' \leftarrow x)\, P(x) \;=\; R(x \leftarrow x')\, P(x')$$



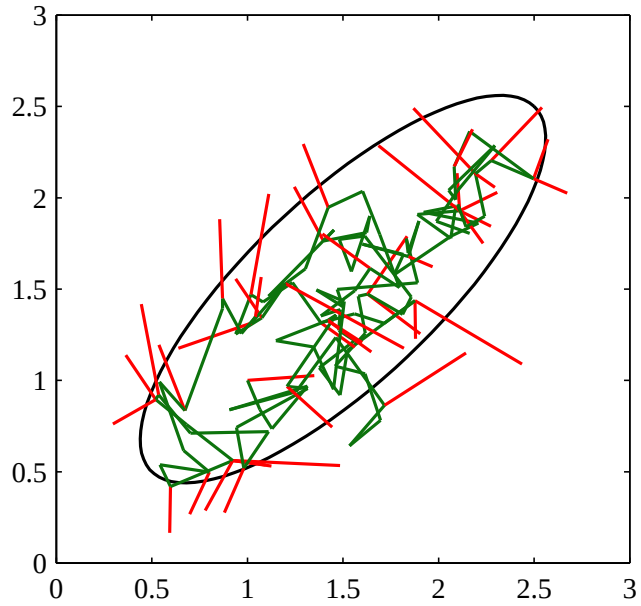
Implies that $P(x)$ is left invariant:

$$\sum_x T(x' \leftarrow x)\, P(x) \;=\; P(x') \sum_x R(x \leftarrow x') \;=\; 1$$

Metropolis–Hastings

Arbitrary proposals $\sim Q$:

$$Q(x';x)\,P(x) \;\neq\; Q(x;x')\,P(x')$$



PRML, Bishop (2006)

Satisfies detailed balance by rejecting moves:

$$T(x' \leftarrow x) = \begin{cases} Q(x';x)\min\left(1,\ \frac{P(x')\,Q(x;x')}{P(x)\,Q(x';x)}\right) & x' \neq x \\ \ldots & x' = x \end{cases}$$

Metropolis–Hastings

Transition operator

- Propose a move from the current state $Q(x';x)$, e.g. $\mathcal{N}(x, \sigma^2)$
- Accept with probability $\min\left(1,\ \frac{P(x')Q(x;x')}{P(x)Q(x';x)}\right)$
- Otherwise next state in chain is a copy of current state

Notes

- Can use $P^* \propto P(x)$; normalizer cancels in acceptance ratio
- Satisfies detailed balance (shown below)
- Q must be chosen so chain is ergodic

$$P(x) \cdot T(x' \leftarrow x) = P(x) \cdot Q(x';x) \min\left(1,\ \frac{P(x')Q(x;x')}{P(x)Q(x';x)}\right) = \min\Big(P(x)Q(x';x),\ P(x')Q(x;x')\Big)$$

$$= P(x') \cdot Q(x;x') \min\left(1,\ \frac{P(x)Q(x';x)}{P(x')Q(x;x')}\right) = P(x') \cdot T(x \leftarrow x')$$

Matlab/Octave code for demo

```
function samples = dumb_metropolis(init, log_ptilde, iters, sigma)

D = numel(init);
samples = zeros(D, iters);

state = init;
Lp_state = log_ptilde(state);
for ss = 1:iters
    % Propose
    prop = state + sigma*randn(size(state));
    Lp_prop = log_ptilde(prop);
    if log(rand) < (Lp_prop - Lp_state)
        % Accept
        state = prop;
        Lp_state = Lp_prop;
    end
    samples(:, ss) = state(:);
end
```

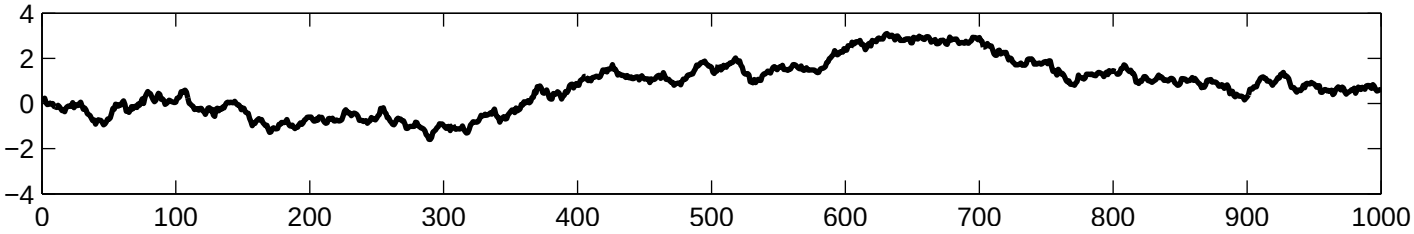
Step-size demo

Explore $\mathcal{N}(0,1)$ with different step sizes σ

```
sigma = @(s) plot(dumb_metropolis(0, @(x)-0.5*x*x, 1e3, s));
```

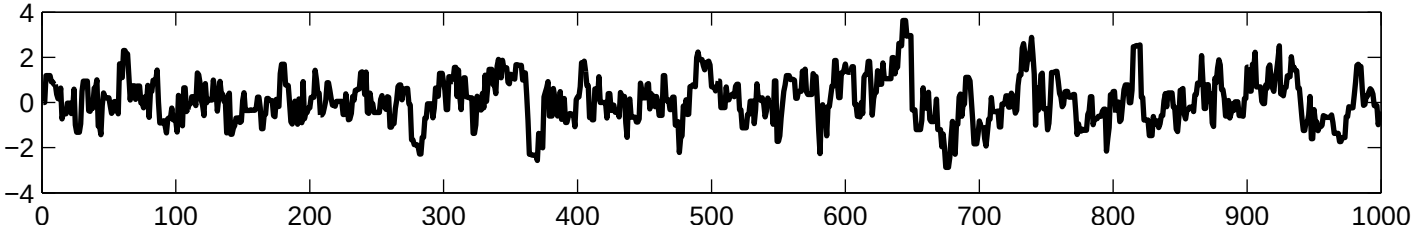
`sigma(0.1)`

99.8% accepts



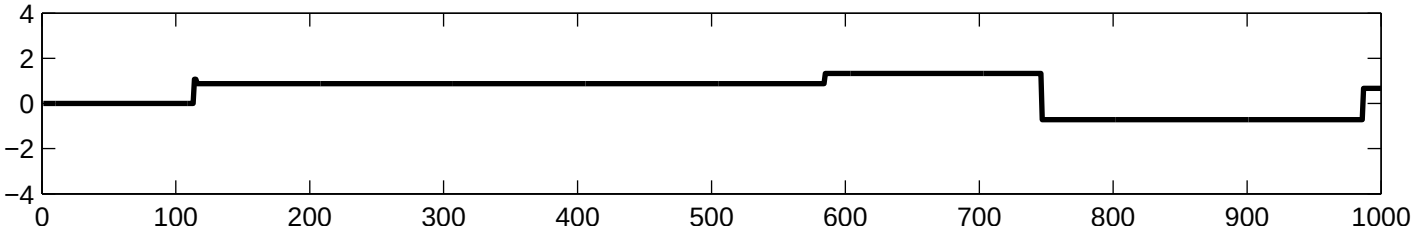
`sigma(1)`

68.4% accepts

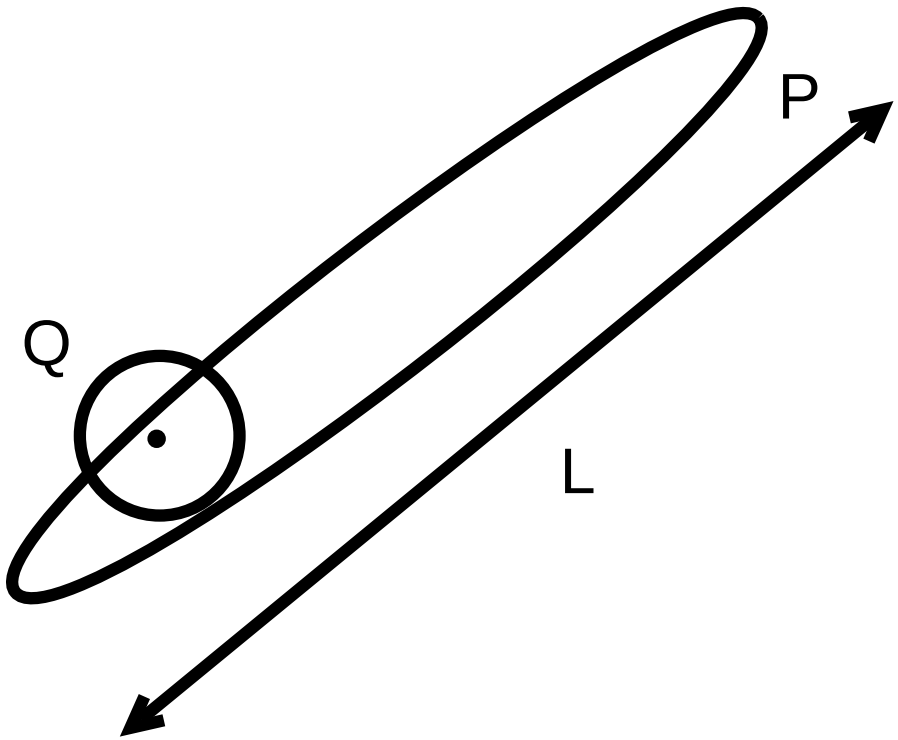


`sigma(100)`

0.5% accepts



Diffusion time



Generic proposals use
$Q(x'; x) = \mathcal{N}(x, \sigma^2)$

σ large $\rightarrow$ many rejections

σ small $\rightarrow$ slow diffusion:

$\sim (L/\sigma)^2$ iterations required

Adapted from MacKay (2003)

An MCMC strategy

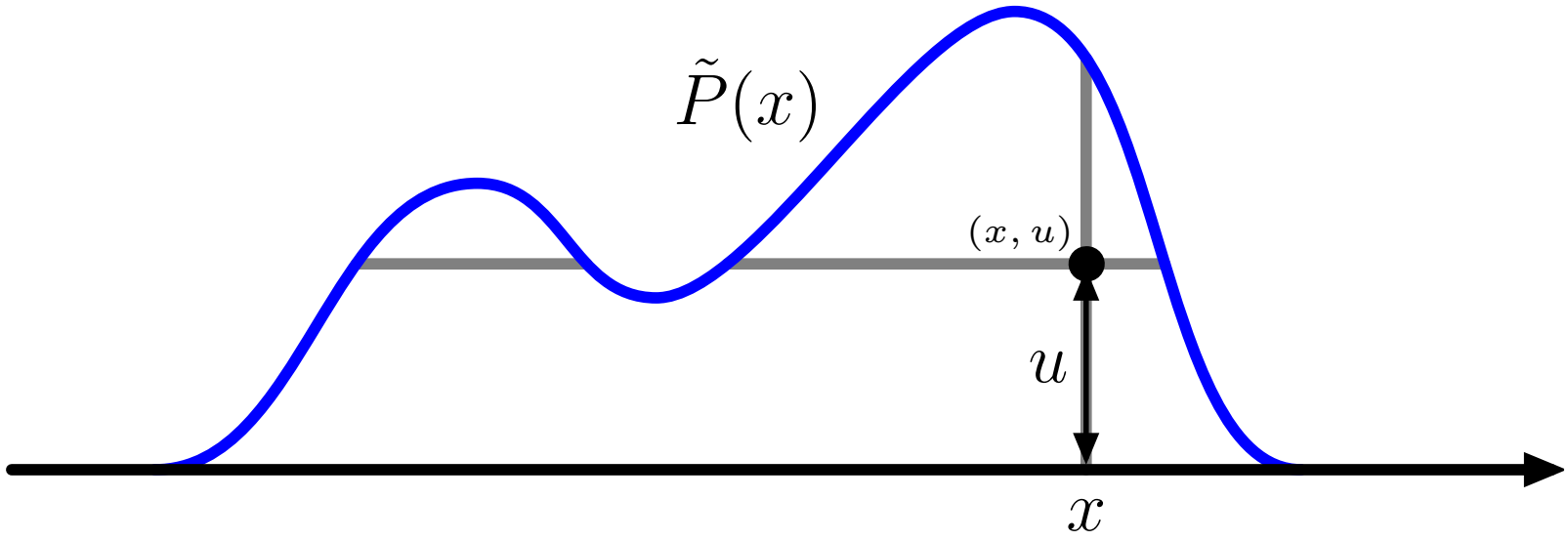
Come up with good proposals $Q(x'; x)$

Combine transition operators:

$$
\begin{aligned}
x_1 &\sim T_A(\cdot \leftarrow x_0) \\
x_2 &\sim T_B(\cdot \leftarrow x_1) \\
x_3 &\sim T_C(\cdot \leftarrow x_2) \\
x_4 &\sim T_A(\cdot \leftarrow x_3) \\
x_5 &\sim T_B(\cdot \leftarrow x_4) \\
&\ldots
\end{aligned}
$$

Slice sampling idea

Sample point uniformly under curve $P^*(x) \propto P(x)$

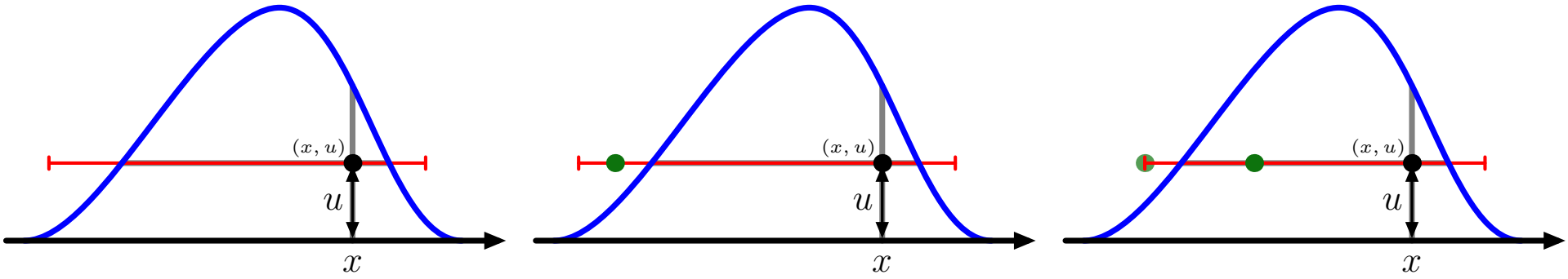


$$p(u|x) = \text{Uniform}[0, P^*(x)]$$

$$p(x|u) \propto \begin{cases} 1 & P^*(x) \geq u \\ 0 & \text{otherwise} \end{cases} = \text{“Uniform on the slice”}$$

Slice sampling

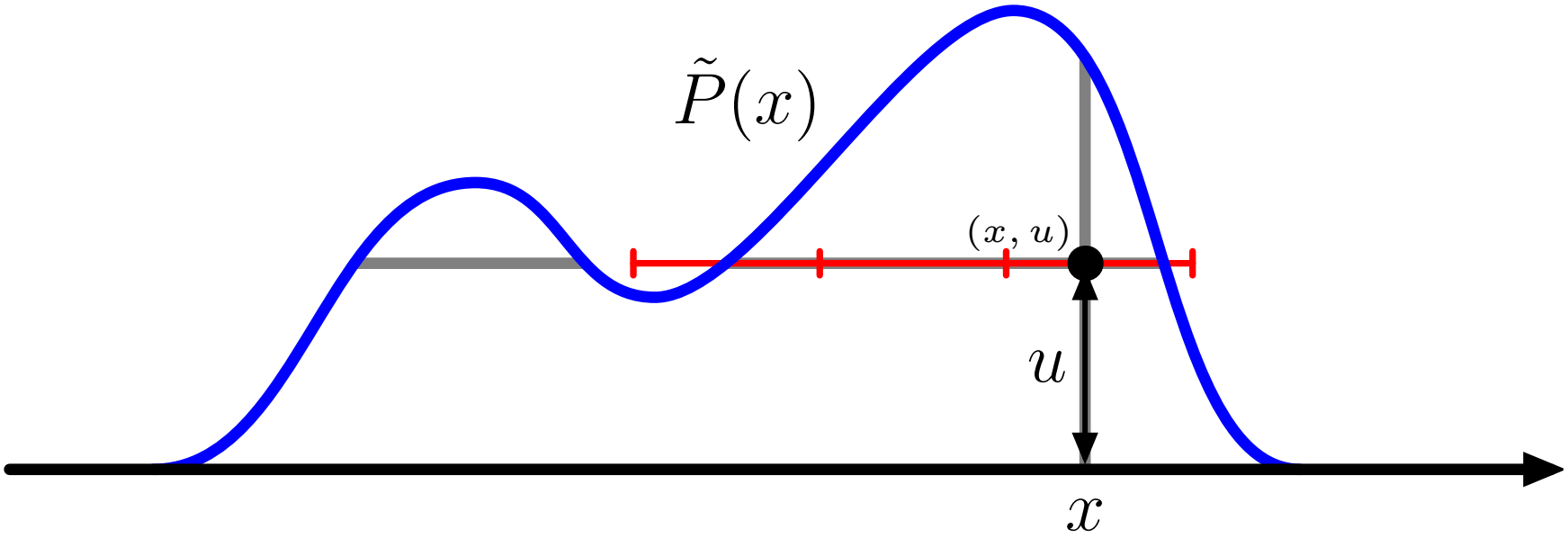
Unimodal conditionals



- bracket slice
- sample uniformly within bracket
- shrink bracket if $P^*(x) < u$ (off slice)
- accept first point on the slice

Slice sampling

Multimodal conditionals



- place bracket randomly around point
- linearly step out until bracket ends are off slice
- sample on bracket, shrinking as before

Satisfies detailed balance, leaves $p(x|u)$ invariant

Slice sampling

Advantages of slice-sampling:

- Easy — only requires $P^*(x) \propto P(x)$
- No rejections
- Tweak params not too important

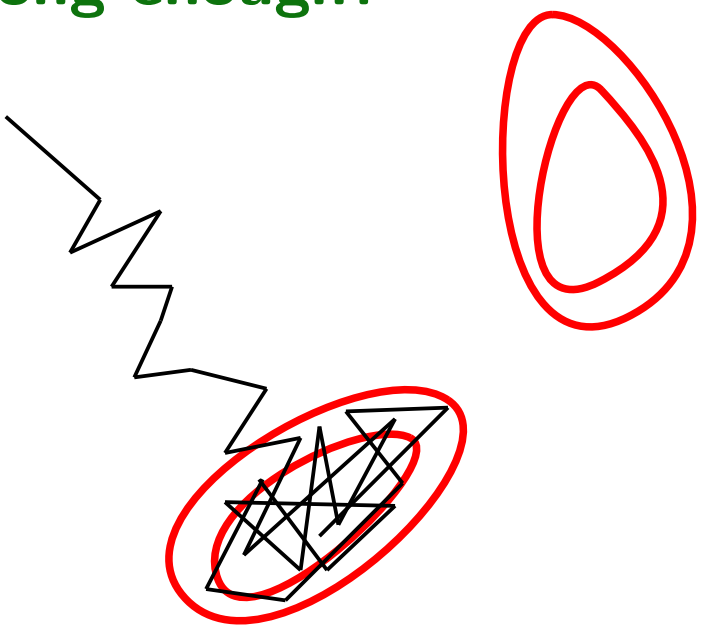
There are more advanced versions.
Neal (2003) contains *many* ideas.

Summary

- We need approximate methods to solve sums/integrals
- Monte Carlo does not explicitly depend on dimension.
  Using samples from simple $Q(\mathbf{x})$ only works in low dimensions.
- Markov chain Monte Carlo (MCMC) can make local moves.
  Sample from complex distributions, even in high dimensions.
- simple computations $\Rightarrow$ "easy" to implement
  (harder to diagnose).

How should we run MCMC?

- The samples aren't independent. Should we **thin**, only keep every Kth sample?
- Arbitrary initialization means starting iterations are bad. Should we discard a **"burn-in" period**?
- Maybe we should perform **multiple runs?**
- How do we know if we have run for **long enough?**



Forming estimates

Approximately independent samples can be obtained by *thinning*.
However, **all the samples can be used.**
Use the simple Monte Carlo estimator on MCMC samples. It is:

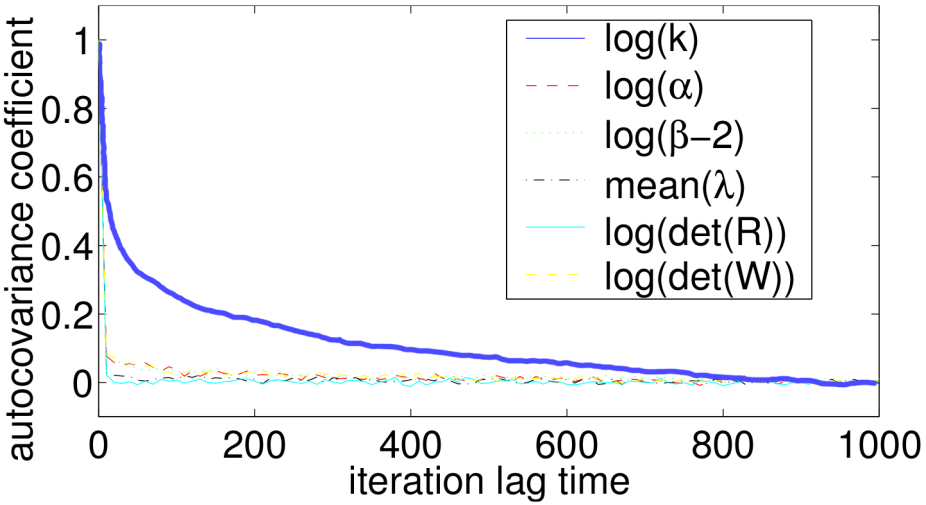
— consistent
— unbiased if the chain has "burned in"

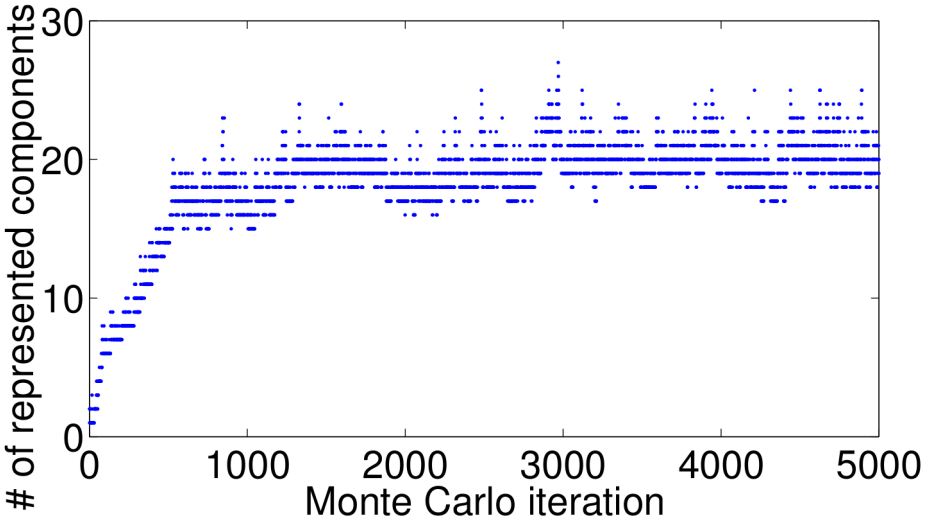
The correct motivation to thin: if computing $f(\mathbf{x}^{(s)})$ is expensive

In some special circumstances strategic thinning can help.

Steven N. MacEachern and Mario Peruggia, Statistics & Probability Letters, 47(1):91–98, 2000.
http://dx.doi.org/10.1016/S0167-7152(99)00142-X — Thanks to Simon Lacoste-Julien for the reference.

Empirical diagnostics





Rasmussen (2000)

Recommendations

For diagnostics:
Standard software packages like `R-CODA`

For opinion on thinning, multiple runs, burn in, etc.
Practical Markov chain Monte Carlo
Charles J. Geyer, *Statistical Science*. 7(4):473–483, 1992.

`http://www.jstor.org/stable/2246094`

Consistency checks

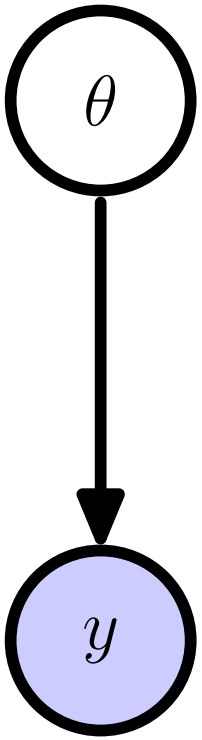
Do I get the right answer on tiny versions of my problem?

Can I make good inferences about synthetic data drawn from my model?

Getting it right: joint distribution tests of posterior simulators, John Geweke, $JASA$, 99(467):799–804, 2004.

Posterior Model checking: Gelman et al. Bayesian Data Analysis textbook and papers.

Getting it right



We write MCMC code to update $\theta \,|\, y$

Idea: also write code to sample $y \,|\, \theta$

Both codes leave $P(\theta, y)$ invariant

Run codes alternately. Check θ's match prior

Summary

Write down the probability of everything.

Condition on what you know,
sample everything that you don't.

Samples give plausible explanations:

— Look at them

— Average their predictions