GENERAL ASSEMBLY

MARKOV CHAIN MONTE CARLO (MCMC) METHODS

Matt Brems

DSI+

LEARNING OBJECTIVES

- Understand and be able to describe how MCMC works.
- Identify situations where MCMC is beneficial.

OPENING: BAYES' THEOREM

$$f(\theta|y) = \frac{f(y|\theta)f(\theta)}{f(y)} \propto f(y|\theta)f(\theta)$$

- Why would we want the posterior distribution?

OPENING: BAYES' THEOREM

$$f(\theta|y) = \frac{f(y|\theta)f(\theta)}{f(y)} \propto f(y|\theta)f(\theta)$$

- Why would we want the posterior distribution?
- What is conjugacy?

OPENING: BAYES' THEOREM

$$f(\theta|y) = \frac{f(y|\theta)f(\theta)}{f(y)} \propto f(y|\theta)f(\theta)$$

- Why would we want the posterior distribution?
- What is conjugacy?
- What happens when we don't have conjugacy?

OPENING: BAYES' THEOREM

$$f(\theta|y) = \frac{f(y|\theta)f(\theta)}{f(y)} \propto f(y|\theta)f(\theta)$$

- Why would we want the posterior distribution?
- What is conjugacy?
- What happens when we don't have conjugacy?
- If we can't find a "closed-form" solution, can we at least approximate one?

MCMC METHODS

MCMC COMPONENTS

- There are three main components to MCMC methods:
  - Monte Carlo Simulations
  - Markov Chains
  - Acceptance-Rejection Sampling

MCMC COMPONENTS

- There are three main components to MCMC methods:
  - **Monte Carlo Simulations**
  - Markov Chains
  - Acceptance-Rejection Sampling

MONTE CARLO

- Monte Carlo methods allow us to approximate a complicated system with a statistical sample.

- This is a way of generating random numbers.

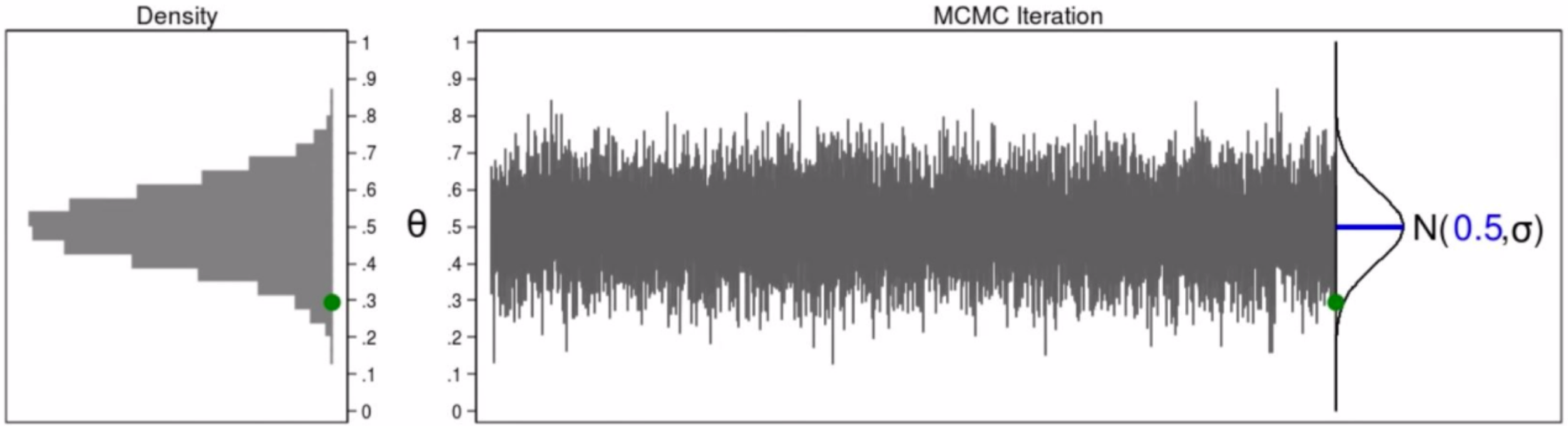
$$\theta_t \sim N(0.5, \sigma)$$

- We can use this to solve a number of problems – like approximating the value of π!

MONTE CARLO

- This is a way of generating random numbers.

$$\theta_t \sim N(0.5, \sigma)$$



MCMC COMPONENTS

- There are three main components to MCMC methods:
  - Monte Carlo Simulations
  - **Markov Chains**
  - Acceptance-Rejection Sampling

MARKOV CHAINS

- The Markov property:

$$\mathbb{P}(X_{n+1} = x_i | X_n = x_j) = \mathbb{P}(X_{n+1} = x_i | X_n = x_j, X_{n-1} = x_k, \ldots, X_1 = x_z)$$

  - Knowing what happens at time n tells me everything I can learn about time $n + 1$; knowing what happened at time $n - 1$, $n - 2$, and so on doesn't provide me with any additional information.

MARKOV CHAINS: EXAMPLES

- Genetics
- Monopoly
- Baseball
- Google's PageRank Algorithm

MARKOV CHAINS: LONG-RUN BEHAVIOR

- Let's look at the long-run behavior of Markov chains.

MARKOV CHAINS: LONG-RUN BEHAVIOR

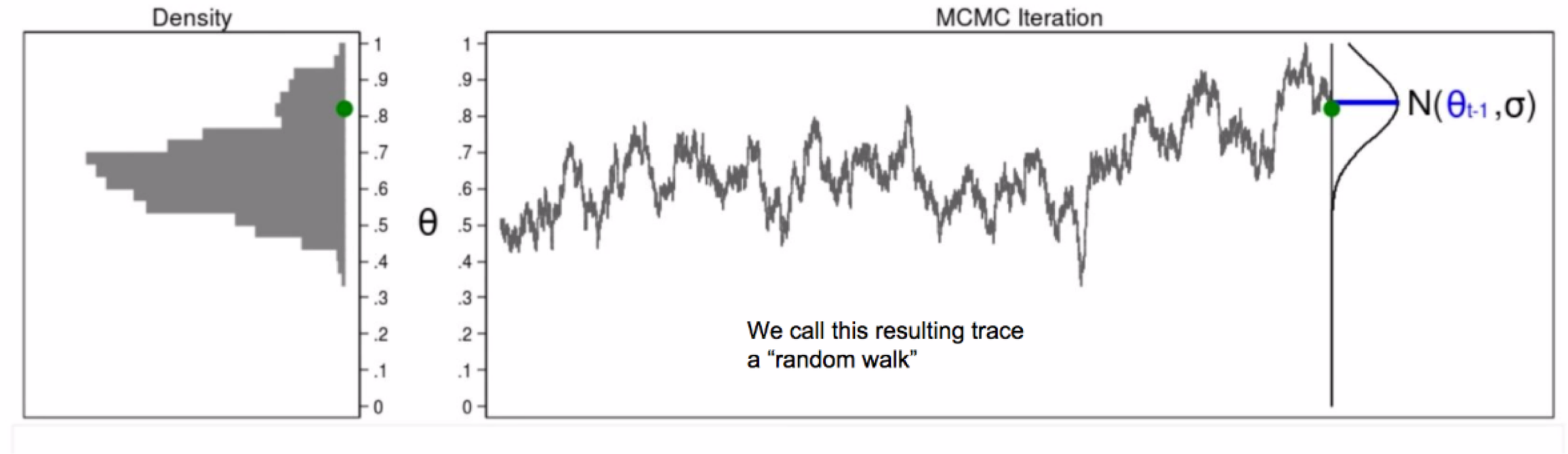
- Markov chains, under simple assumptions, will always converge to the same long-run behavior.

- Markov chains, under simple assumptions, will always converge to the same long-run behavior **independent of the starting point!**

MARKOV CHAINS

- We'll apply ideas from Markov chains here.

$$\theta_t \sim N(\theta_{t-1}, \sigma)$$



MCMC COMPONENTS

- There are three main components to MCMC methods:
  - Monte Carlo Simulations
  - Markov Chains
  - **Acceptance-Rejection Sampling**

ACCEPTANCE-REJECTION SAMPLING

- Acceptance-rejection sampling is a specific type of Monte Carlo sampling.

ACCEPTANCE-REJECTION SAMPLING

- Acceptance-rejection sampling is a specific type of Monte Carlo sampling.

- We're going to sample some observation, then decide whether to keep it (accept it) or discard it (reject it).

ACCEPTANCE-REJECTION SAMPLING

- Acceptance-rejection sampling is a specific type of Monte Carlo sampling.

- We're going to sample some observation, then decide whether to keep it (accept it) or discard it (reject it).

- As our random walk moves about, we need some way to make sure that we're moving in the right direction.
  - If we are moving in the right direction, we'll accept that sample.
  - If we aren't moving in the right direction, we may reject that sample.

ACCEPTANCE-REJECTION SAMPLING

- Acceptance-rejection sampling is a specific type of Monte Carlo sampling.

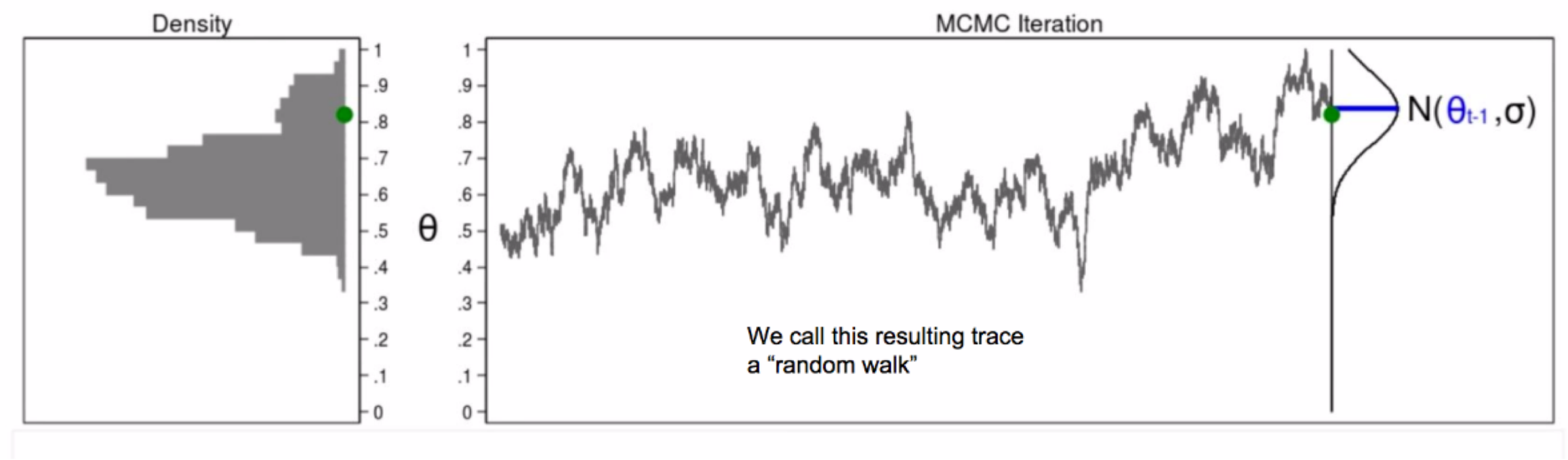
- We're going to sample some observation, then decide whether to keep it (accept it) or discard it (reject it).

- As our random walk moves about, we need some way to make sure that we're moving in the right direction.
  - If we are moving in the right direction, we'll accept that sample.
  - If we aren't moving in the right direction, we may reject that sample.

- We'll use a specific algorithm called the Metropolis-Hastings algorithm to accept/reject.

METROPOLIS-HASTINGS ALGORITHM

- Thus far, we've been generating all of these random numbers but have no idea if the numbers we're generating are headed in the right direction.



METROPOLIS-HASTINGS ALGORITHM

- Thus far, we've been generating all of these random numbers but have no idea if the numbers we're generating are headed in the right direction.

- The Metropolis-Hastings algorithm allows us to, at each step, identify whether we are getting "hotter" or "colder."

METROPOLIS-HASTINGS ALGORITHM

- Thus far, we've been generating all of these random numbers but have no idea if the numbers we're generating are headed in the right direction.

- The Metropolis-Hastings algorithm allows us to, at each step, identify whether we are getting "hotter" or "colder."

- Recall that we're simulating $\theta_t \sim N(\theta_{t-1}, \sigma)$.

METROPOLIS-HASTINGS ALGORITHM

- Thus far, we've been generating all of these random numbers but have no idea if the numbers we're generating are headed in the right direction.
- The Metropolis-Hastings algorithm allows us to, at each step, identify whether we are getting "hotter" or "colder."
- Recall that we're simulating $\theta_t \sim N(\theta_{t-1}, \sigma)$.
- Let's not move so quickly and call this θ_t. Let's call it $\theta_{proposal}$ instead.

$$\theta_{proposal} \sim N(\theta_{t-1}, \sigma)$$

METROPOLIS-HASTINGS ALGORITHM

- Well, how do we find θ_t?

$$r(\theta_{proposal}, \theta_{t-1}) = \frac{P(\theta_{proposal}|y)}{P(\theta_{t-1}|y)}$$

METROPOLIS-HASTINGS ALGORITHM

- Well, how do we find θ_t?

$$r\left(\theta_{proposal}, \theta_{t-1}\right) = \frac{P\left(\theta_{proposal}|y\right)}{P(\theta_{t-1}|y)}$$

$$\alpha_t = \min\{r, 1\}$$

METROPOLIS-HASTINGS ALGORITHM

- Well, how do we find θ_t?

$$r\left(\theta_{proposal}, \theta_{t-1}\right) = \frac{P\left(\theta_{proposal}|y\right)}{P(\theta_{t-1}|y)}$$

$$\alpha_t = \min\{r, 1\}$$

$$u_t \sim Uniform(0,1)$$

METROPOLIS-HASTINGS ALGORITHM

- Well, how do we find θ_t?

$$r\left(\theta_{proposal}, \theta_{t-1}\right) = \frac{P\left(\theta_{proposal}|y\right)}{P(\theta_{t-1}|y)}$$

$$\alpha_t = \min\{r, 1\}$$

$$u_t \sim Uniform(0,1)$$

$$\theta_t = \begin{cases} \theta_{t-1} & \text{if } u_t > \alpha_t \\ \theta_{proposal} & \text{if } u_t \leq \alpha_t \end{cases}$$

METROPOLIS-HASTINGS ALGORITHM

- Suppose $\theta_{proposal}$ is 10% likelier than θ_{t-1}, based on the posterior probability.
- $r(\theta_{proposal}, \theta_{t-1}) = \frac{P(\theta_{proposal}|y)}{P(\theta_{t-1}|y)}$
- $\alpha_t = \min\{r, 1\}$
- $u_t \sim Uniform(0,1)$
- $\theta_t = \begin{cases} \theta_{t-1} & \text{if } u_t > \alpha_t \\ \theta_{proposal} & \text{if } u_t \leq \alpha_t \end{cases}$

METROPOLIS-HASTINGS ALGORITHM

- Suppose $\theta_{proposal}$ is only 80% as likely as θ_{t-1}, based on the posterior probability.
- $r(\theta_{proposal}, \theta_{t-1}) = \frac{P(\theta_{proposal}|y)}{P(\theta_{t-1}|y)}$
- $\alpha_t = \min\{r, 1\}$
- $u_t \sim Uniform(0,1)$
- $\theta_t = \begin{cases} \theta_{t-1} & \text{if } u_t > \alpha_t \\ \theta_{proposal} & \text{if } u_t \leq \alpha_t \end{cases}$

METROPOLIS-HASTINGS ALGORITHM

- Wait… so sometimes we pick a worse value?

METROPOLIS-HASTINGS ALGORITHM

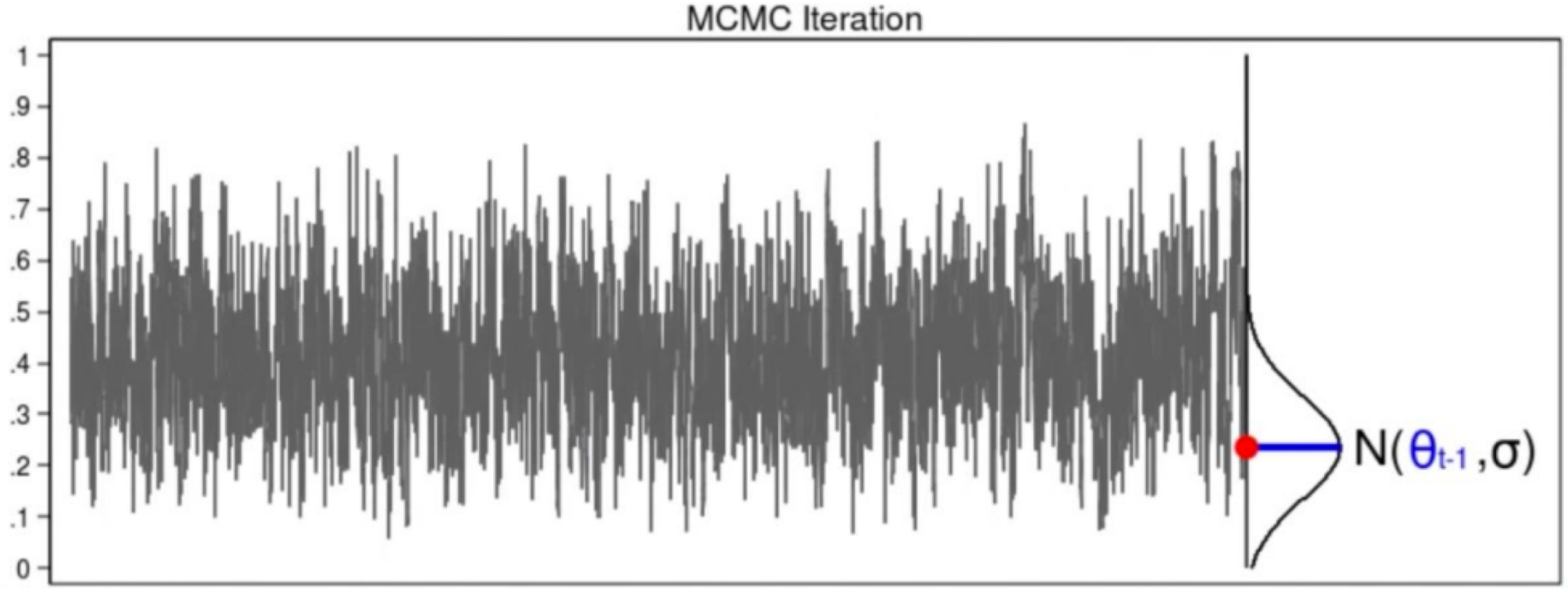
- Step 1: Generate $\theta_{proposal} \sim N(\theta_{t-1}, \sigma)$.
- Step 2: Calculate $r(\theta_{proposal}, \theta_{t-1})$, the ratio of posteriors of $\theta_{proposal}$ and θ_{t-1}.
- Step 3: Calculate the acceptance probability $\alpha_t = \min\{r, 1\}$.
- Step 4: Generate u_t from a $Uniform(0,1)$ distribution.
- Step 5: If the generated number u_t is greater than the acceptance probability α_t, then $\theta_t = \theta_{t-1}$. Otherwise, $\theta_t = \theta_{proposal}$.

METROPOLIS-HASTINGS ALGORITHM

- We do this many, many times.

METROPOLIS-HASTINGS ALGORITHM

- We do this many, many times.



Source: STATA

METROPOLIS-HASTINGS ALGORITHM

- We do this many, many times.

- Once we do this long enough and visually inspect plots to convince ourselves that we have "converged" to the posterior distribution of interest, we usually discard early samples (before we converged), then we store some large n of the later samples.

METROPOLIS-HASTINGS ALGORITHM

- We do this many, many times.

- Once we do this long enough and visually inspect plots to convince ourselves that we have "converged" to the posterior distribution of interest, we usually discard early samples (before we converged), then we store some large n of the later samples.

- Once we have our sample of size n, we can conduct whatever inference we want.
  - Find the mean.
  - Find the median.
  - Find the 95% 'highest posterior density.'
  - Find the variance.

CLOSING: BAYES' THEOREM

$$f(\theta|y) = \frac{f(y|\theta)f(\theta)}{f(y)} \propto f(y|\theta)f(\theta)$$

- Why would we want the posterior distribution?

CLOSING: BAYES' THEOREM

$$f(\theta|y) = \frac{f(y|\theta)f(\theta)}{f(y)} \propto f(y|\theta)f(\theta)$$

- Why would we want the posterior distribution?
- When is MCMC helpful?

CLOSING: BAYES' THEOREM

$$f(\theta|y) = \frac{f(y|\theta)f(\theta)}{f(y)} \propto f(y|\theta)f(\theta)$$

- Why would we want the posterior distribution?
- When is MCMC helpful?
- How does MCMC work?