# MARKOV CHAIN MONTE CARLO

Machine Learning

IIIy CSE IDP and IDDMP

Refer : Stephen Marsland

# Motivation

1) Sampling from complex distributions

   We often need to draw samples from a target distribution p(x) that is difficult to sample from directly.

2) Random walk inefficiency

   A simple random walk on a state space moves away from the start only as sqrt(t), making exploration very slow.

3) Desirable convergence properties

   We want our sampler to converge to p(x) regardless of the starting state.

4) Chain requirements

   **Irreducible**: every state must be reachable from every other state.

   **Ergodic (aperiodic and recurrent)**: ensures every state is revisited infinitely often without being stuck in cycles.

# What is Markov Chain Monte Carlo ?

1. **Markov Chain**
   a. A sequence of states $s_0, s_1, s_2, \ldots$
   b. **Markov property**: $P(s_t | s_{t-1}, s_{t-2}, \ldots) = P(s_t | s_{t-1})$
   c. Transition probabilities between states are given by a (constant) matrix T.
2. **Monte Carlo via random walk**
   d. Start from any state and move according to T
   e. If T is constructed so its stationary distribution is p(x), the empirical distribution of visited states converges to p(x).
3. **No decision-making**
   f. Unlike Markov Decision Processes, there are no actions—only stochastic transitions governed by T.
4. **Key properties for sampling**
   g. **Irreducible** and **ergodic** ensures convergence to the target distribution from any starting point.

# Invariance & Detailed Balance

**Invariant distribution**
We require that the target distribution p(x) remains unchanged by one step of the chain:

$$p(\mathbf{x}) = \sum_{\mathbf{y}} T(\mathbf{y}, \mathbf{x}) p(\mathbf{y})$$

**Reversibility ⇒ Detailed balance**
**To keep** p(x) invariant, transitions must be “balanced” so forward/backward moves cancel out:

$$p(\mathbf{x})T(\mathbf{x}, \mathbf{x}') = p(\mathbf{x}')T(\mathbf{x}', \mathbf{x})$$

**Consequence:** ergodicity + invariance
**Detailed balance plus row-normalized** T guarantees $\sum_{\mathbf{y}} p(\mathbf{y})T(\mathbf{y}, \mathbf{x}) = p(\mathbf{x})$, making

p(x) the chain’s unique stationary distribution.

**MCMC sampling**
Construct a reversible M… ose stationary distribution is p(x).
**Final equation:**

$$u(\mathbf{x}^*|\mathbf{x}^{(i)}) = \min\left(1, \frac{\tilde{p}(\mathbf{x}^*)q(\mathbf{x}^{(i)}|\mathbf{x}^*)}{\tilde{p}(\mathbf{x}^{(i)})q(\mathbf{x}^*|\mathbf{x}^{(i)})}\right)$$

# The Metropolis–Hastings Algorithm

**The Metropolis–Hastings Algorithm**

- Given an initial value $x_0$
- Repeat
  - sample $\mathbf{x}^*$ from $q(\mathbf{x}_i|\mathbf{x}_{i-1})$
  - sample $u$ from the uniform distribution
  - if $u <$ Equation (15.14):
    - set $\mathbf{x}[i+1] = \mathbf{x}^*$
  - otherwise:
    - set $\mathbf{x}[i+1] = \mathbf{x}[i]$
- Until you have enough samples

# Why Metropolis–Hastings Works

1. Proposal & acceptance
   Sample candidate $x^*$ from proposal $q(x^*|x^{(i)})$
   Accept moves that favor higher-probability regions of $p(x)$, otherwise reject.

2. Reversibility ensures correct exploration
   **Detailed balance $\Rightarrow$ chain visits states in proportion to** $p(x)$

3. Generality and simplicity
   Metropolis–Hastings is the most widely used MCMC method.
   Choice of proposal qqq is critical but the algorithm remains straightforward.

4. Symmetric proposals = Metropolis
   **If** $q(x^*|x)=q(x|x^*)$, the acceptance ratio simplifies

# Results of Metropolis-Hastings Algo.

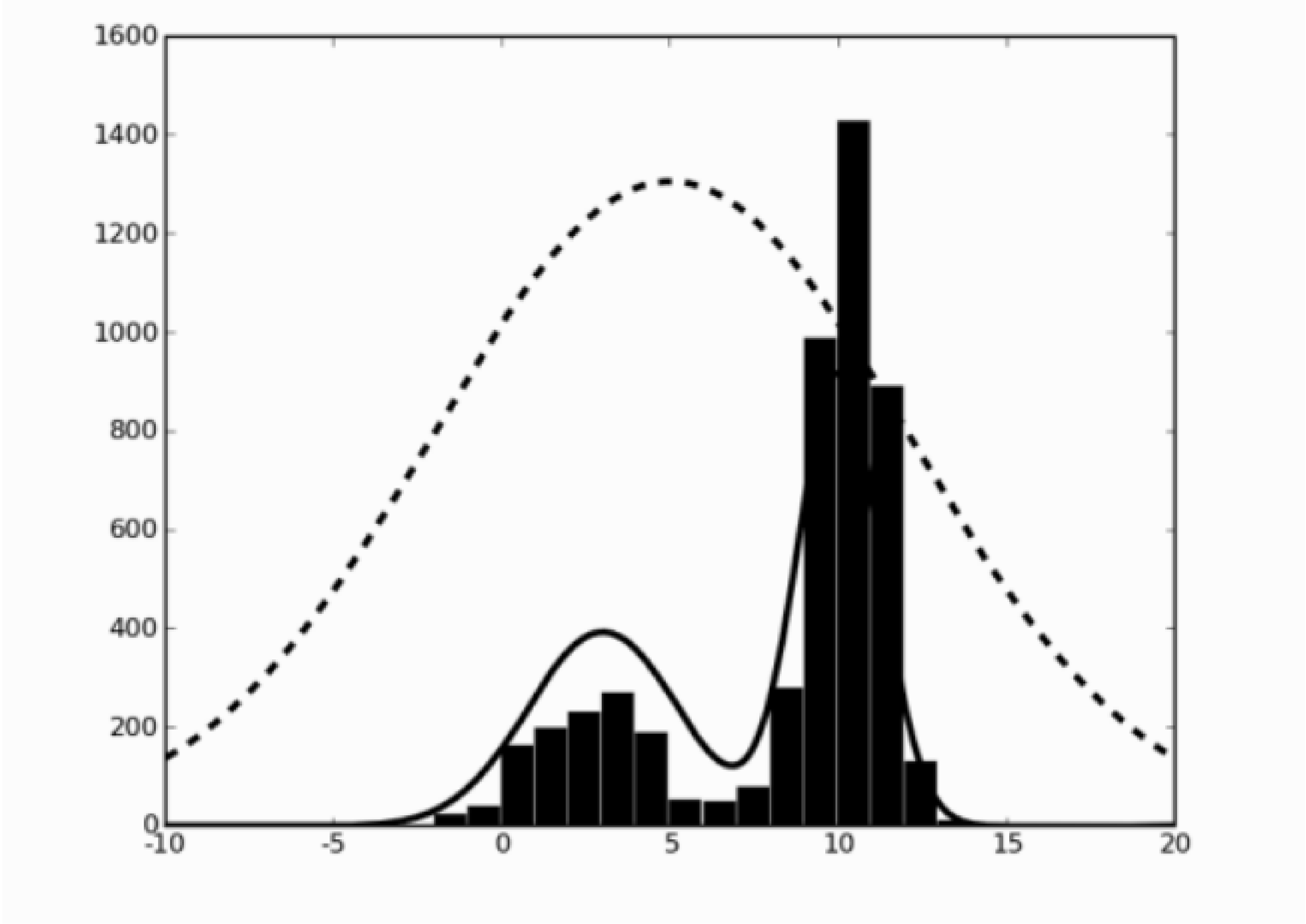


FIGURE 15.5 The results of the Metropolis–Hastings algorithm when the true distribution is a mixture of two Gaussians (shown by the solid line) and the proposal distribution is a single Gaussian (the dotted line).

# Simulated Annealing

**Objective:**
**Locate the mode** arg $\max_x p(x)$ rather than approximate the full distribution.

**Annealed invariant distribution:**
**Replace** $p(x)$ with $p(x)^{1/T_i}$, where $T_i$ (temperature) decreases to 0 as $i \to \infty$

**Annealing schedule:**
**Introduce a cooling schedule** $T_1 > T_2 > \cdots > T_N \to 0$ to progressively penalize worse moves.

**Modifications to Metropolis–Hastings:**

**Acceptance test** incorporates current temperature $T_i$

**Loop update**: decrement $T_i$ according to the schedule.

**Empirical result:**
Next Slide shows how simulated annealing concentrates samples around the global maximum over iterations.

# Simulated annealing Results

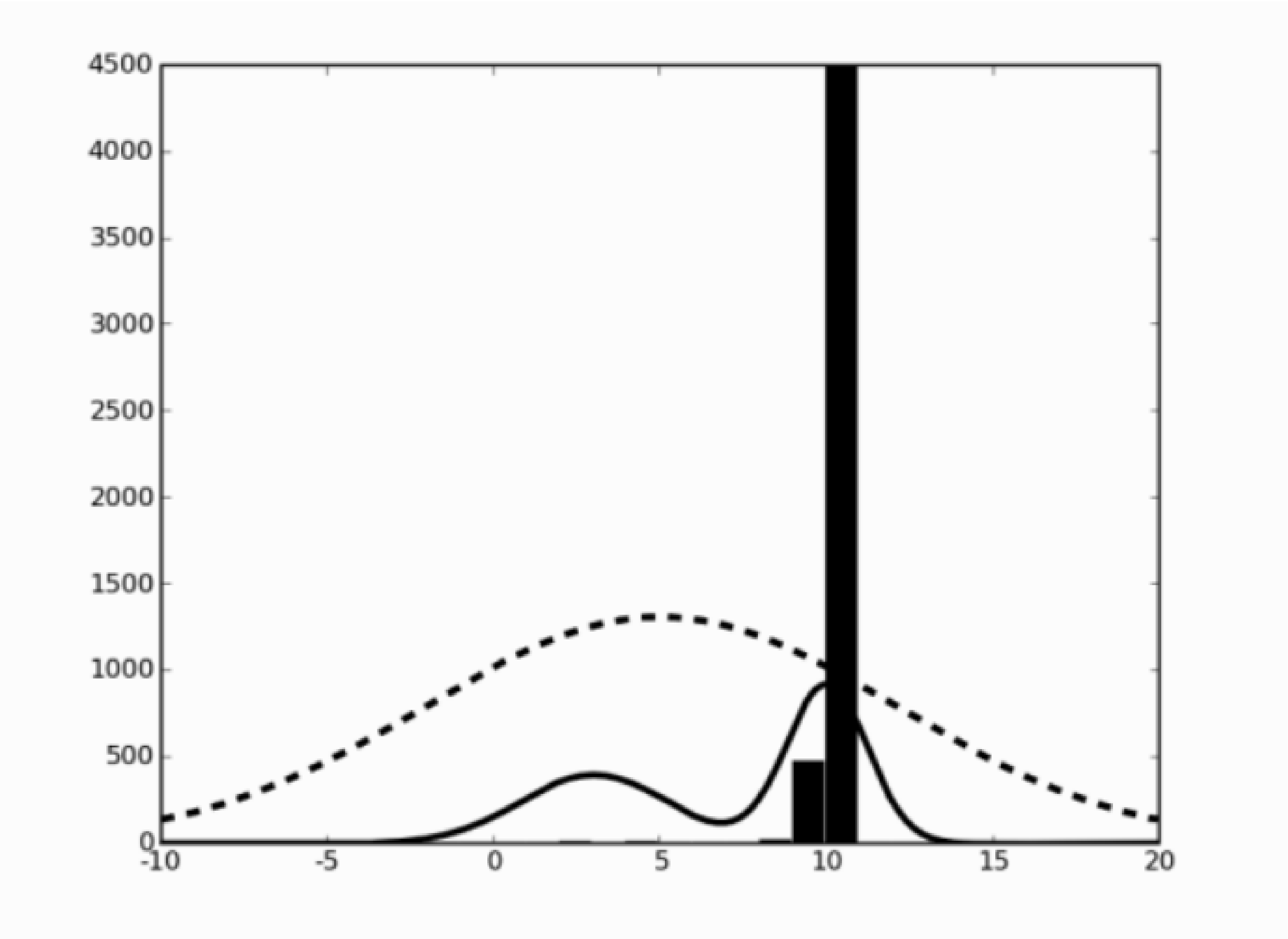


FIGURE 15.7 Using simulated annealing gives the maximum rather than an approximation to the distribution, as is shown here for the same example as in Figures 15.5 and 15.6.

# Gibbs Sampling

**When to use:**

**Full conditional distributions** $p(x_j|x_{-j})$ are known (common in Bayesian networks).

**Proposal choice:**

**Set** $q(x^* \mid x^{(i)}) = p\big(x_j^* \mid x_{-j}^{(i)}\big)$

(only $x_j$ changes; other variables fixed).

**Always accept:**

The acceptance probability simplifies to 1, so every proposed update is accepted.

**Algorithm outline:**

For each variable $x_j$, sample $x_j^{(i+1)} \sim p(x_j|x_{-j}^{(i)})$.

Cycle through variables in a fixed or random order. Continue until the joint distribution stabilizes.

**Applications:**

Foundation of the BUGS software (Bayesian Updating with Gibbs Sampling).

Widely used for inference in complex graphical models.

**Final equation:**

$$\text{Acceptance probability } P_a = \min(1, 1) = 1$$

# The Gibbs Sampler

**The Gibbs Sampler**

- For each variable $x_j$:
  - initialise $x_j^{(0)}$
- Repeat
  - for each variable $x_j$:
    * sample $x_1^{(i+1)}$ from $p(x_1|x_2^{(i)}, \ldots x_n^{(i)})$
    * sample $x_2^{(i+1)}$ from $p(x_2|x_1^{(i+1)}, x_3^{(i)}, \ldots x_n^{(i)})$
    * ...
    * sample $x_n^{(i+1)}$ from $p(x_n|x_1^{(i+1)}, \ldots x_{n-1}^{(i+1)})$
- Until you have enough samples

# The Gibbs Sampler Output

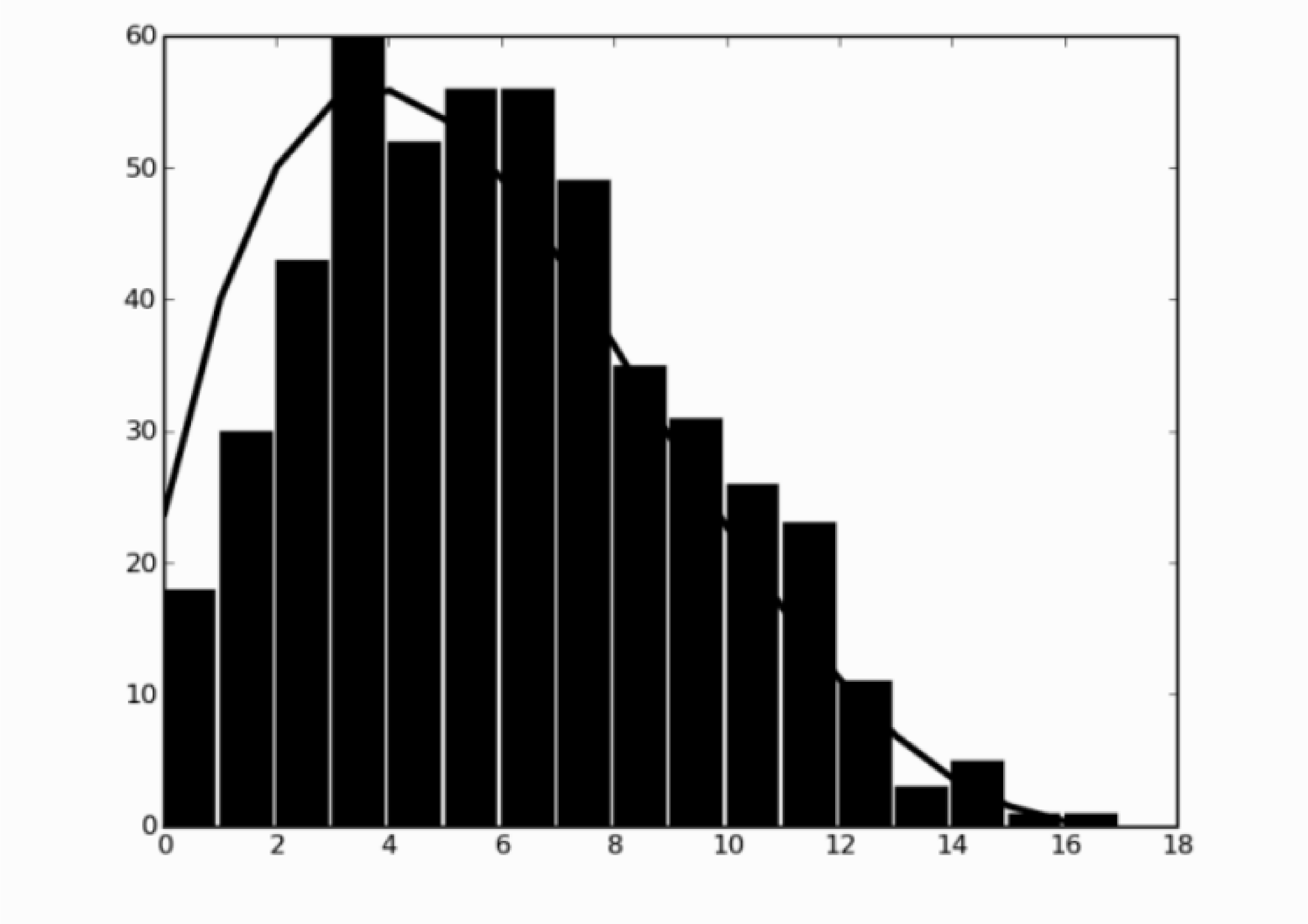


FIGURE 15.8 The Gibbs sampler output for the beta-binomial distribution.

# Summary & Conclusion

1. **Goal of MCMC:**
   Efficiently sample from complex target distributions when direct sampling is infeasible.
2. **Core principle:**
   Construct a reversible Markov chain whose stationary distribution matches the target.
3. **Key algorithms:**
   a. **Metropolis–Hastings:** Flexible, handles arbitrary proposals.
   b. **Metropolis (symmetric proposals):** Special case with simplified acceptance.
   c. **Simulated Annealing:** Adapts MCMC to locate global maxima.
   d. **Gibbs Sampling:** Uses known full-conditionals for automatic acceptance.
4. **Practical considerations:**
   e. Choose proposal distributions to balance exploration versus acceptance rate.
   f. Ensure irreducibility and ergodicity for convergence from any start.
   g. Tune parameters (step sizes, temperature schedules) for efficiency.
5. **Takeaway:**
   MCMC methods form a unified framework for both sampling and optimization, underpinning many applications in Bayesian inference, statistical physics, and beyond.