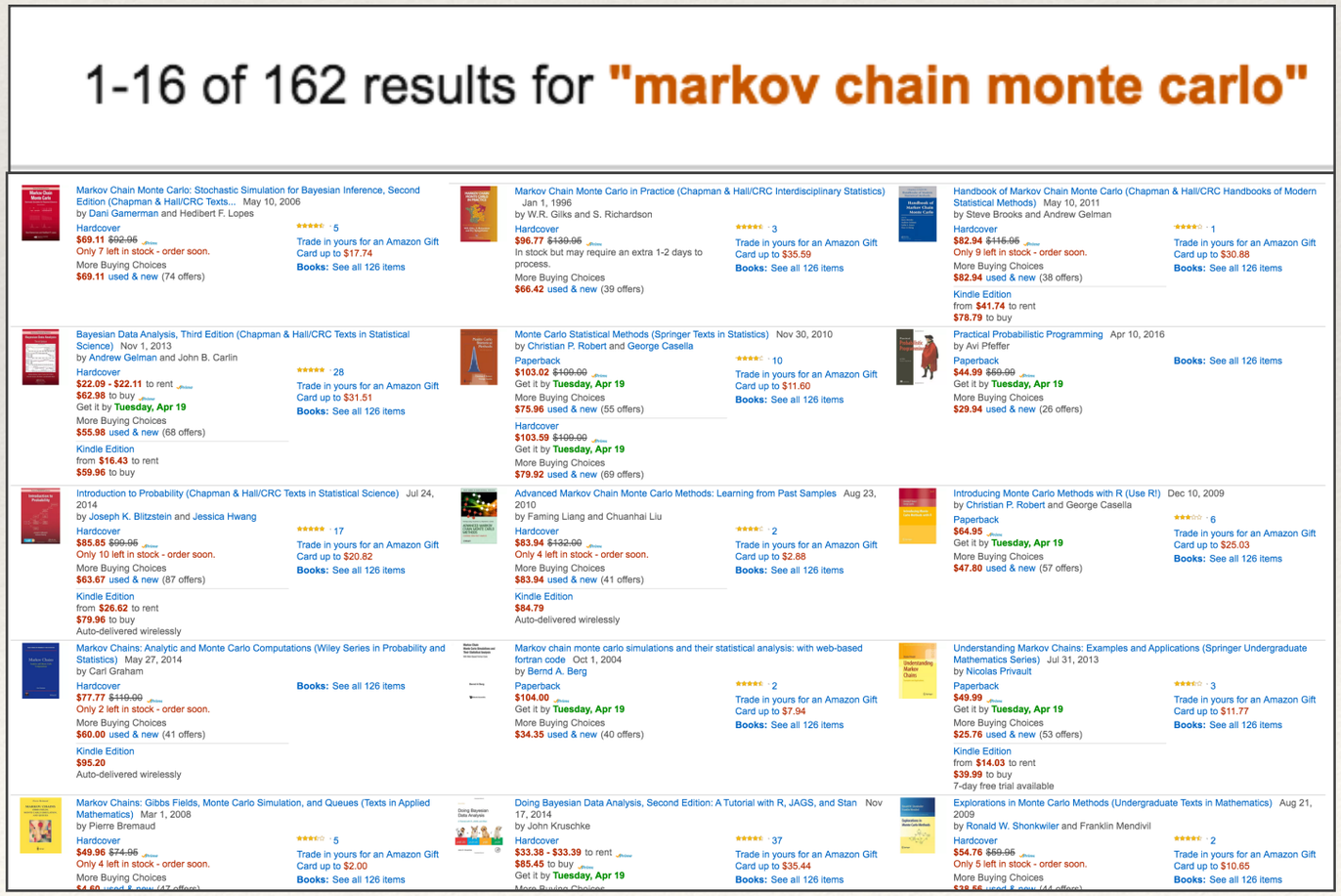
STT 7020: Applied Stochastic Processes

Exploring Markov Chain Monte Carlo Optimization

Matt Piekenbrock

Introduction

- Markov Chain Monte Carlo (MCMC) refers to **an entire field** of algorithms that sample from a probability distribution…
- … where the aforementioned desired distribution is the equilibrium distribution of a reversible, irreducible, Markov Chain
- Used all around the world (at least in CS):
  - STATNET package in R uses MCMC in fitting ERGMs
  - One of BioConductor's most popular packages is MCMCpack



Definition

From 4.9: "Let X be a discrete random vector whose set of possible values is": $x_j, j \geq 1$

Where the PMF of X is: $P(X = x_j), j \geq 1$

And you want to calculate:

$$\theta = E[h(X)] = \sum_{j=1}^{\infty} h(x_j) P(X = x_j)$$

So just use SLLN?

For some function h ...

$$\lim_{n \to \infty} \sum_{i=1}^{n} \frac{h(X_i)}{n} = \theta$$

Definition (cont.)

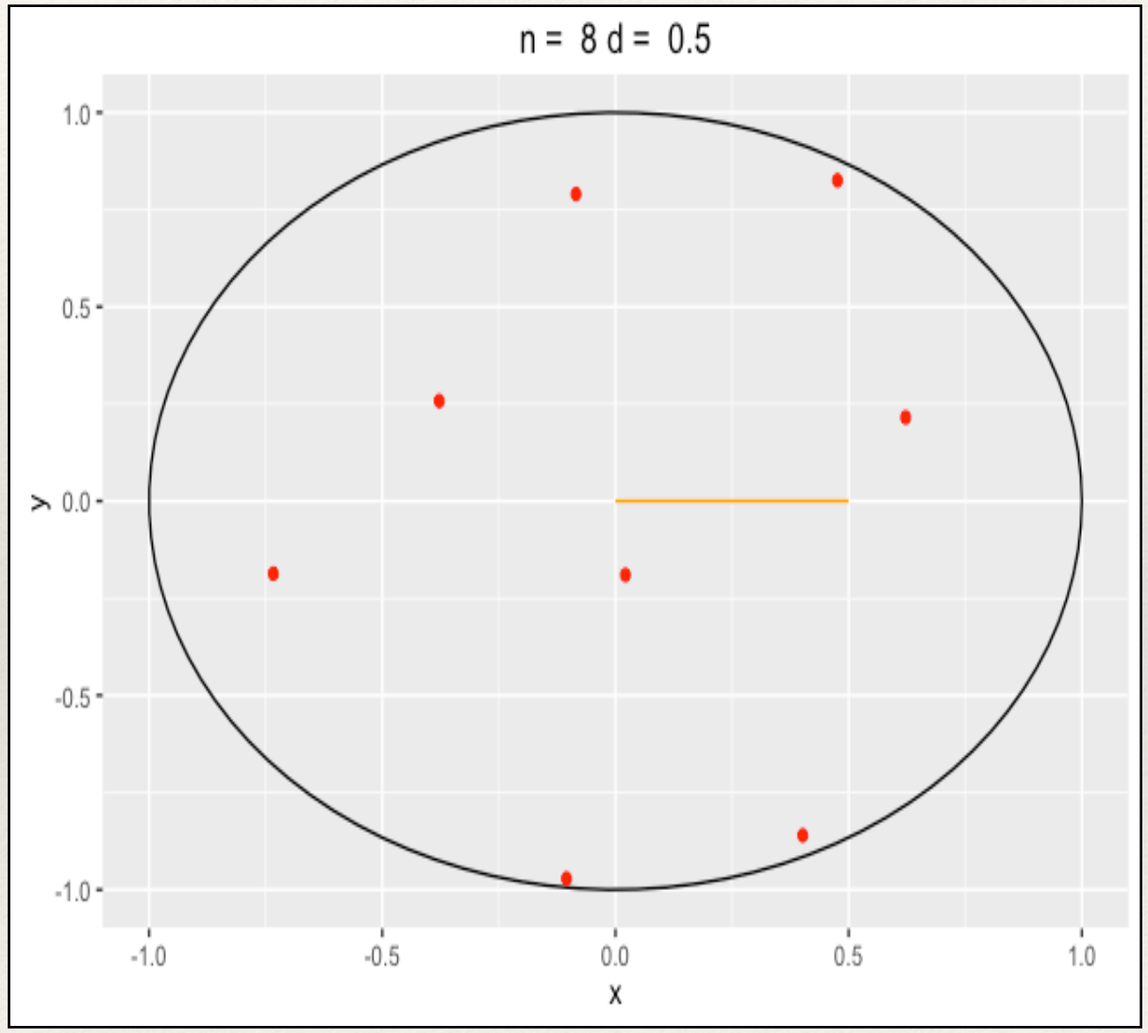
- Problem: It might be extremely difficult to generate a random vector X having a PMF $P(X = x_j)$
- Solution?
- Markov Chains are easy to make
- Generate a sequence of successive state of a vector-valued Markov Chain $(X_1, X_2, \dots)$ ...
- ... whose stationary probability are: $P(X = x_j)$

Example 4.40

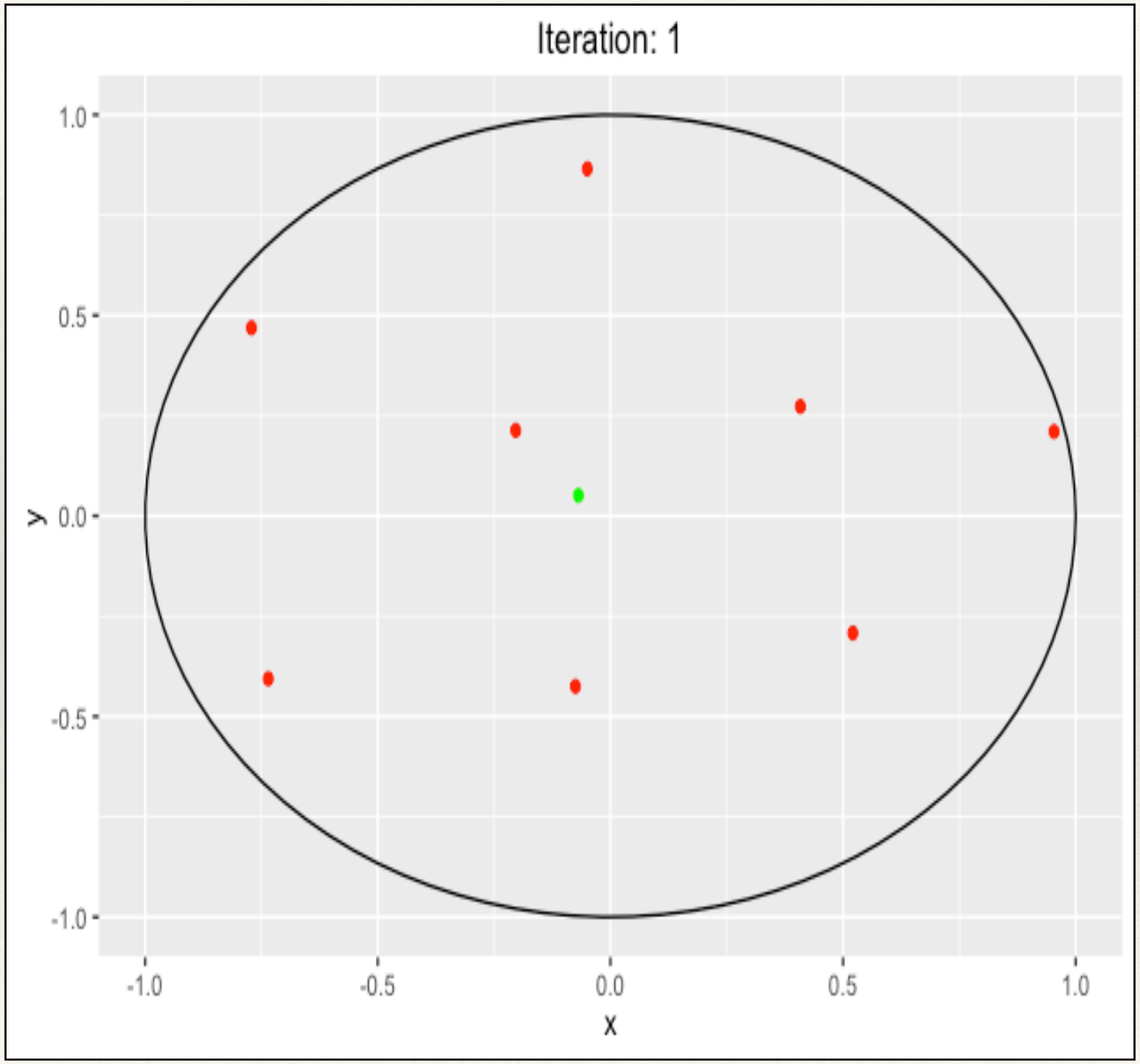
- "Suppose that we want to generate *n* uniformly distributed points in a circle with radius 1 centered at the origin, conditional on the event that no two points are within a distance *d* of each other, when the probability of this conditioning event is small. This can be accomplished by using the Gibbs sampler as follows. Start with any *n* points $x_1, \ldots, x_n$ in the circle that have the property that no two of them are within *d* of the other; then generate the value of *I*, equally likely to be any of the values $1, \ldots, n$. Then continually generate a random point in the circle until you obtain one that is not within *d* of any of the other *n-1* points excluding x_I. At this point, replace x_I by the generated point and then repeat the operation. After a large number of iterations of this algorithm, the set of *n* points will approximately have the desired distribution."

Example 4.40: Demonstration

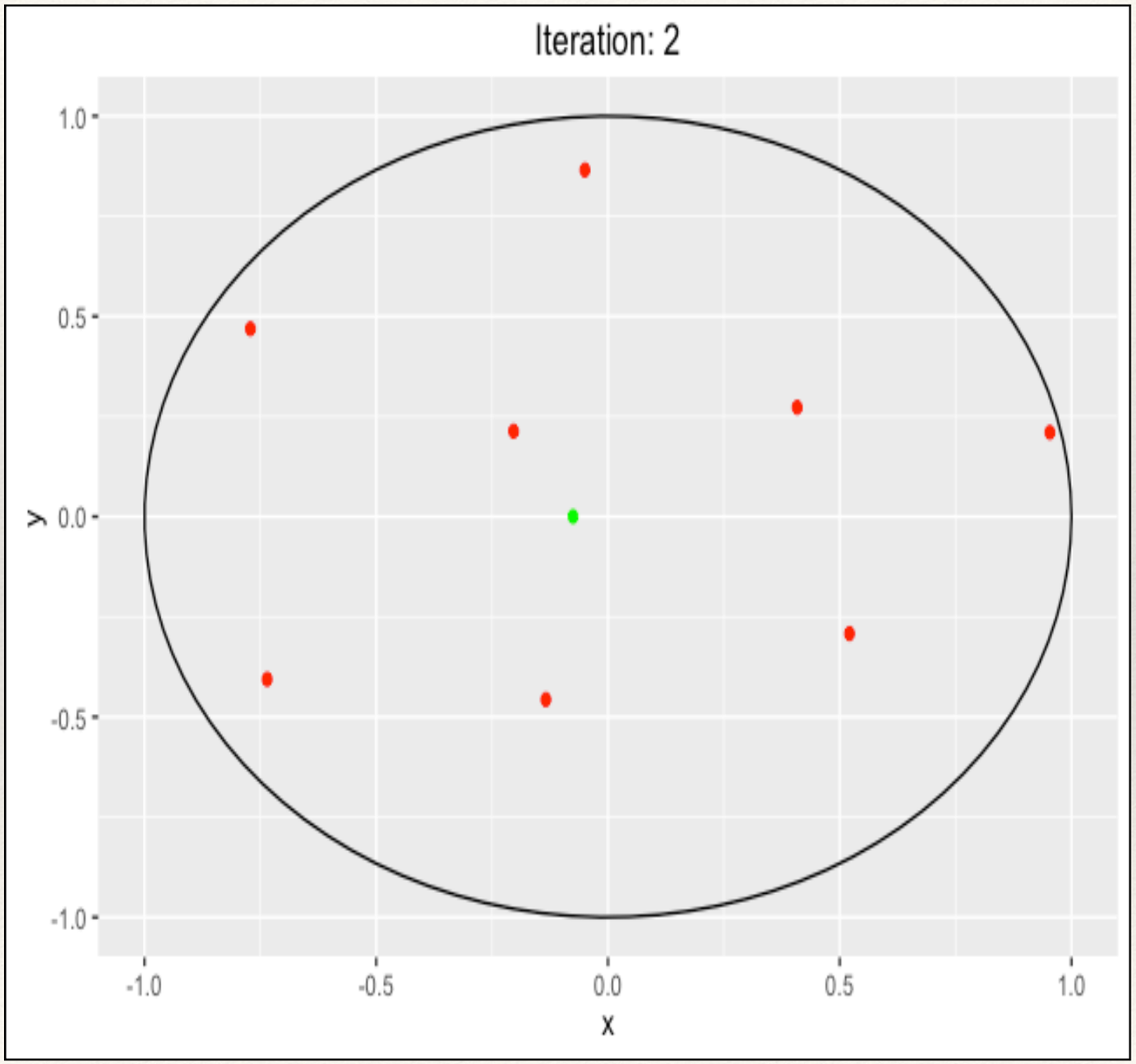
- Red dots: n=8 uniformly distributed points all at least d away from each other
- Orange line: minimal distance that must exists between every point
- Circle at origin with radius 1.0



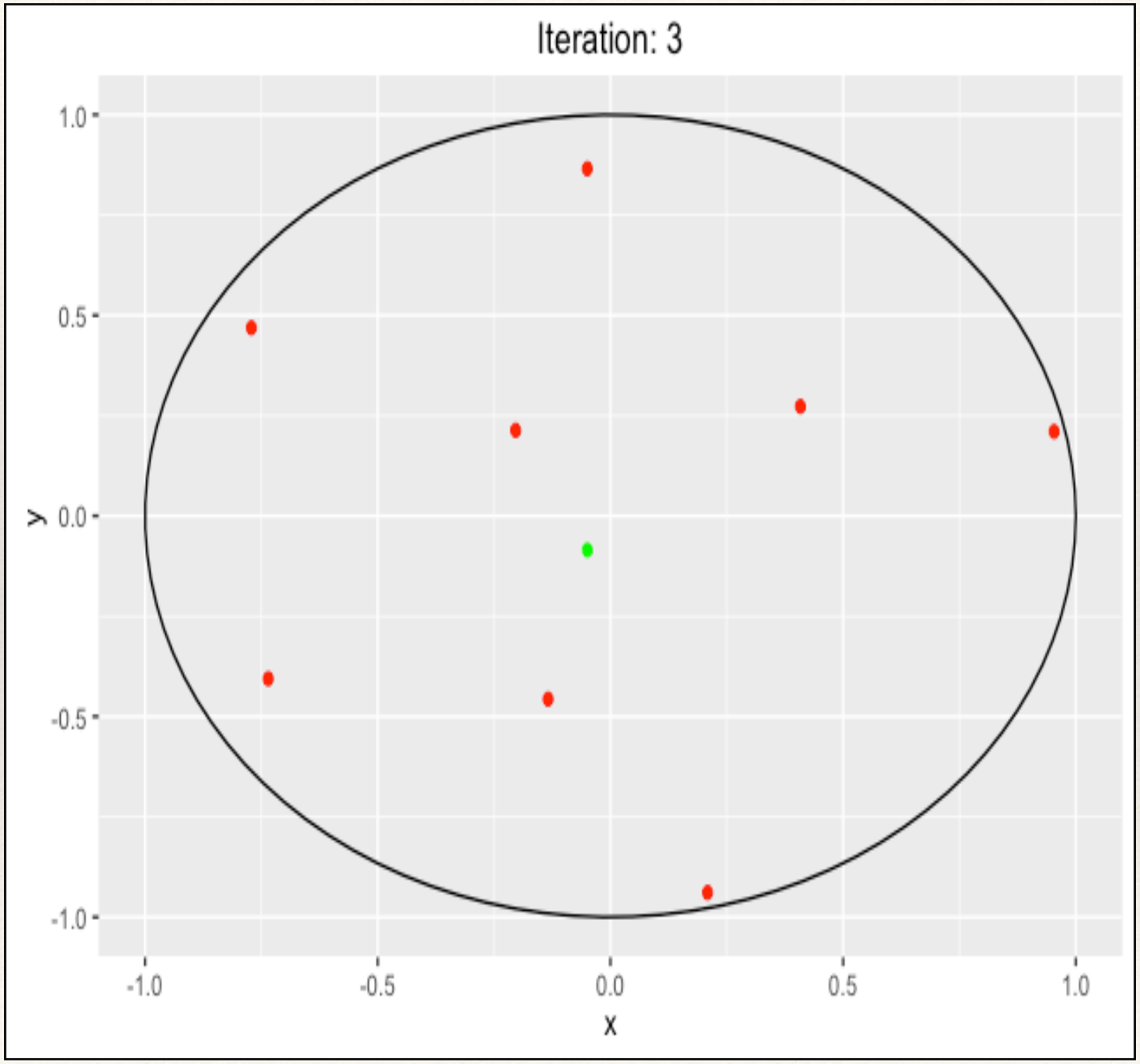
Example 4.40: Demonstration



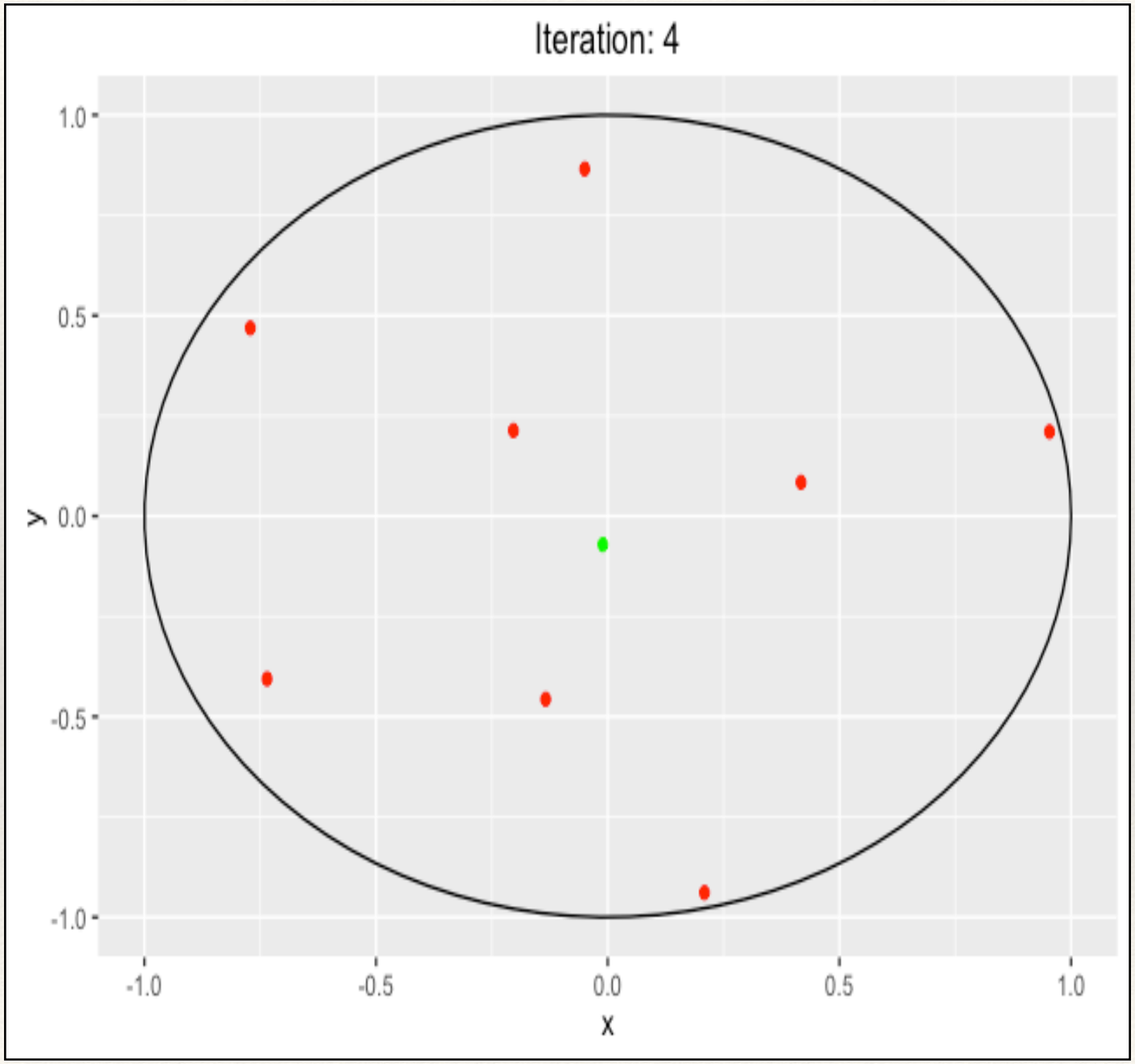
Example 4.40: Demonstration



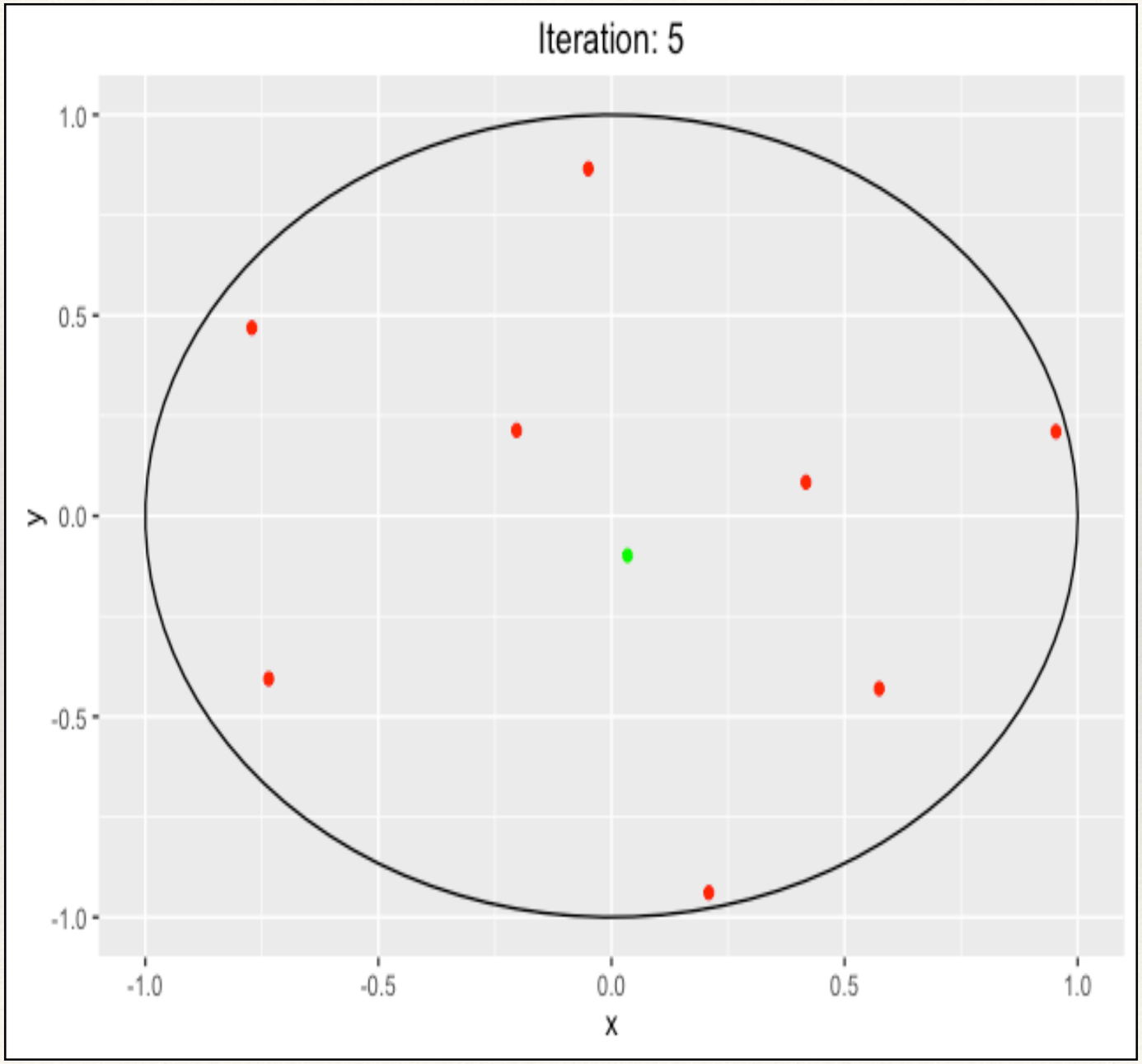
Example 4.40: Demonstration



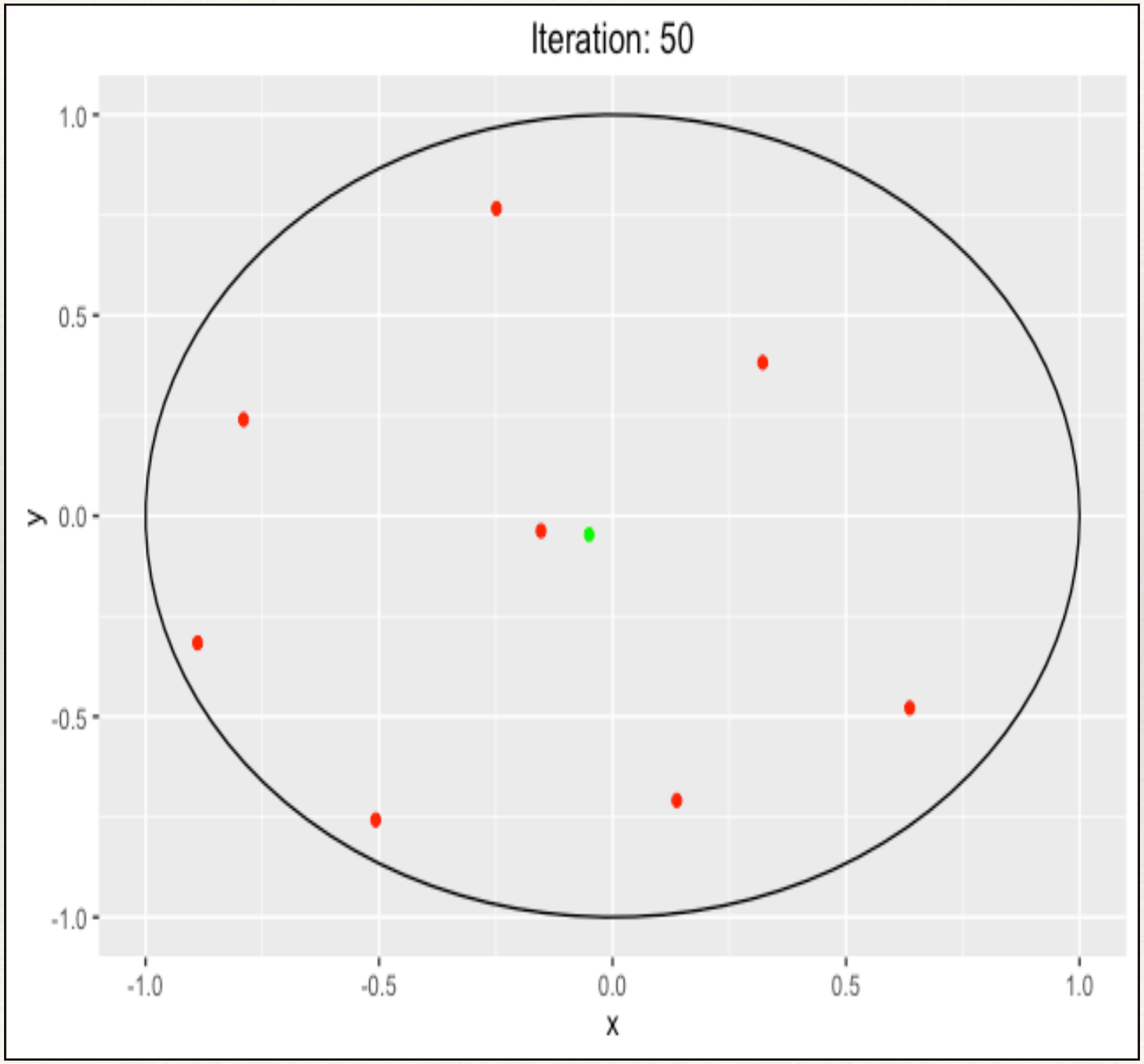
Example 4.40: Demonstration



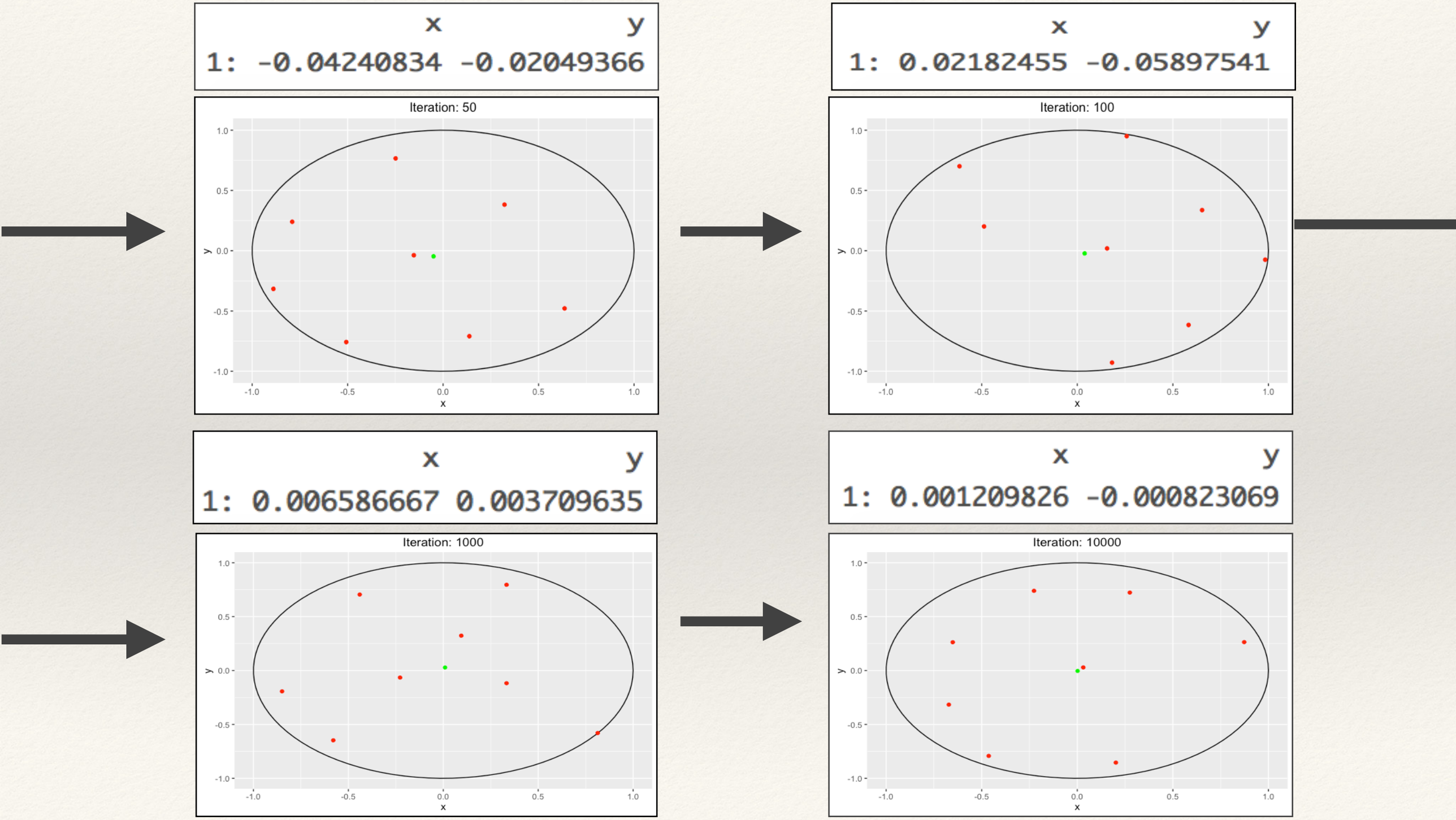
Example 4.40: Demonstration



Example 4.40: Demonstration

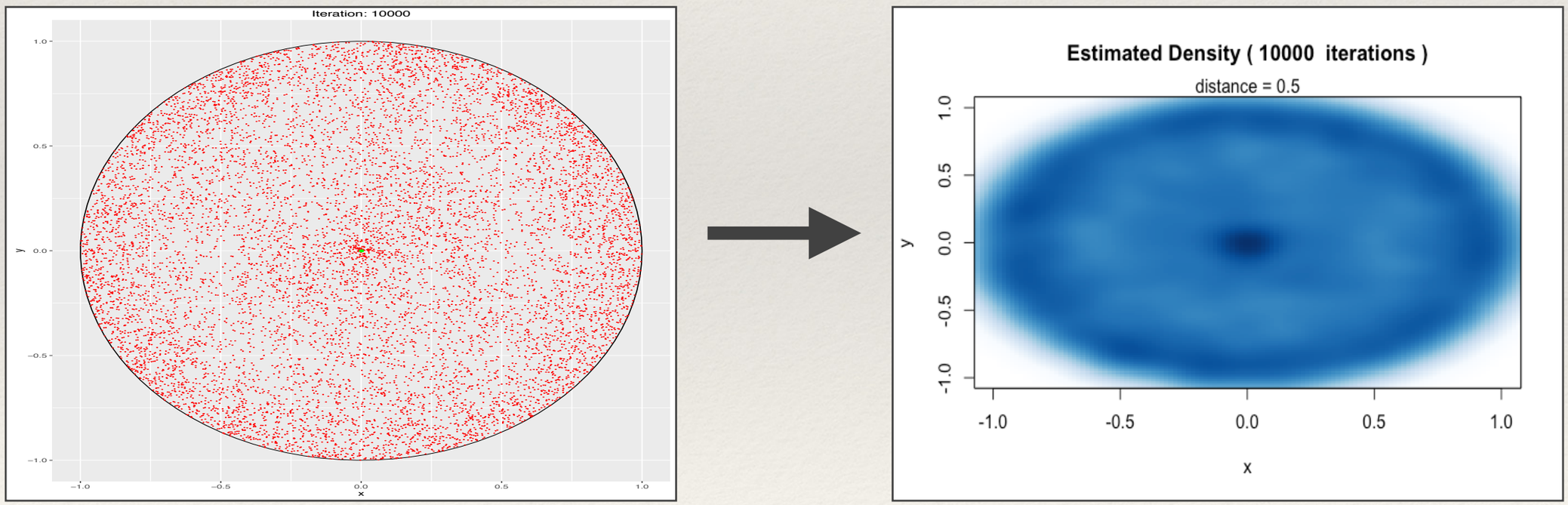


Example 4.40: Demonstration



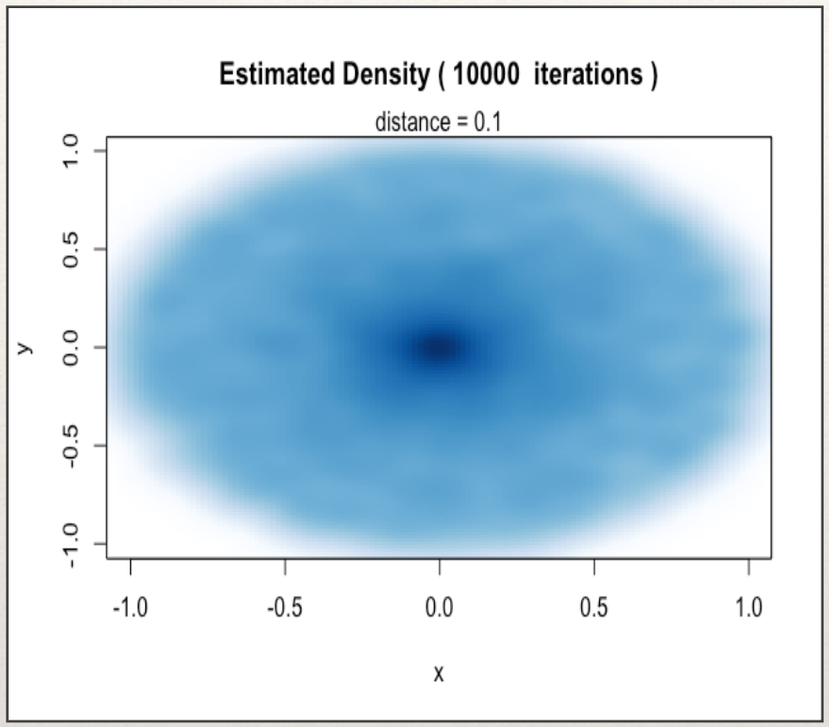
Another Use case of MCMC

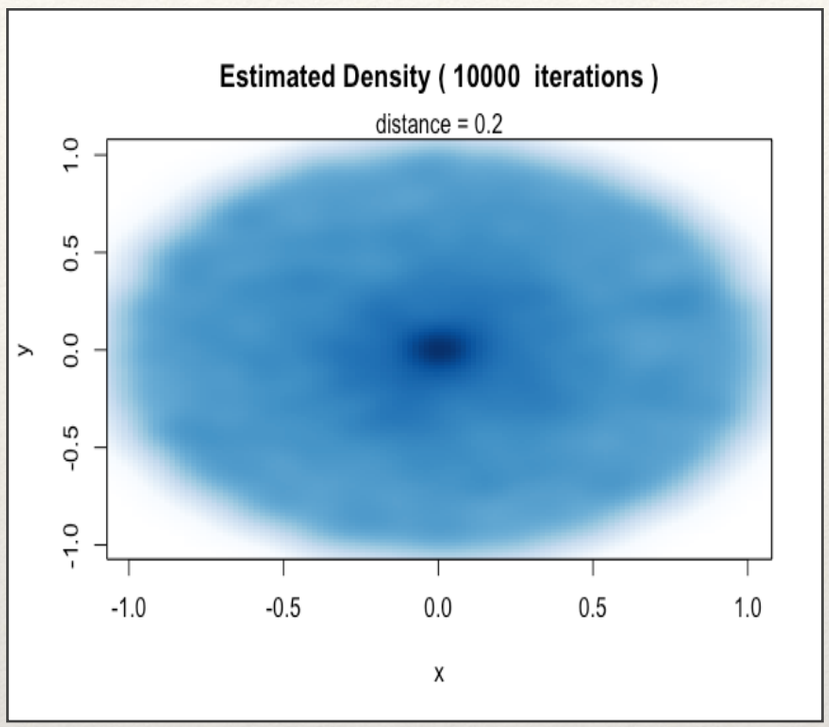
- So we can approximate the expected value of an intractable integral that's too difficult to find analytically… and?
  - What if we save the sample points we keep deleting in the previous example?
- The set of points that were sampled will approximately have the same distribution as the equilibrium distribution!

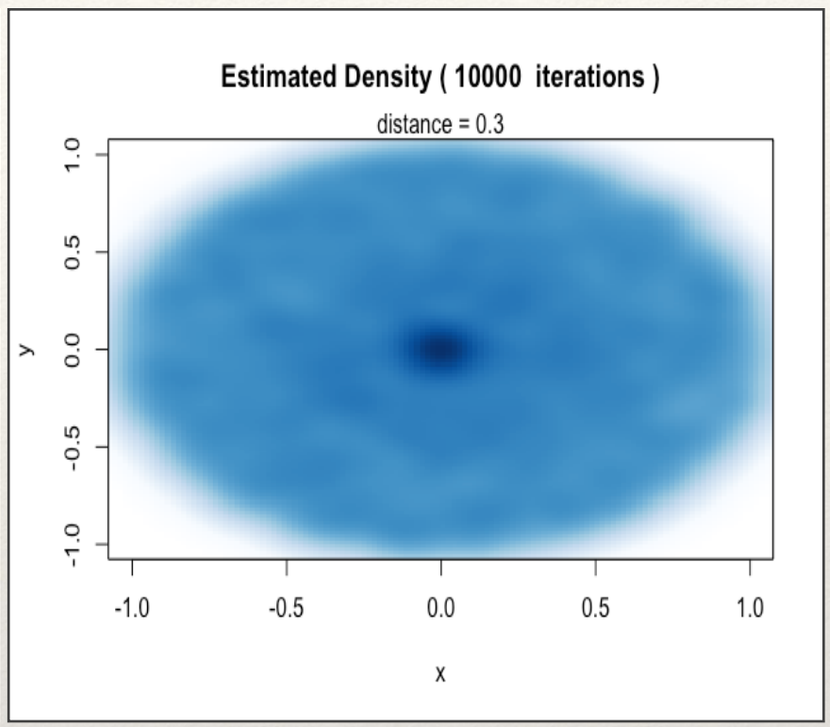


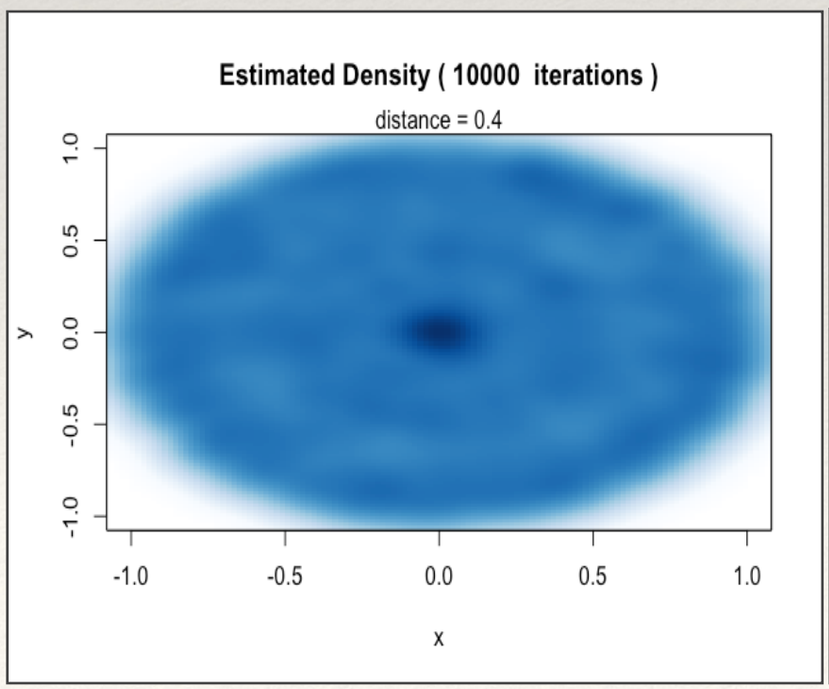
Remark: "As can be seen by Examples 4.40 and 4.41, although the theory for the Gibbs sampler was represented under the assumption that they distribution to be generated was discrete, it also holds when this distribution is continuous"

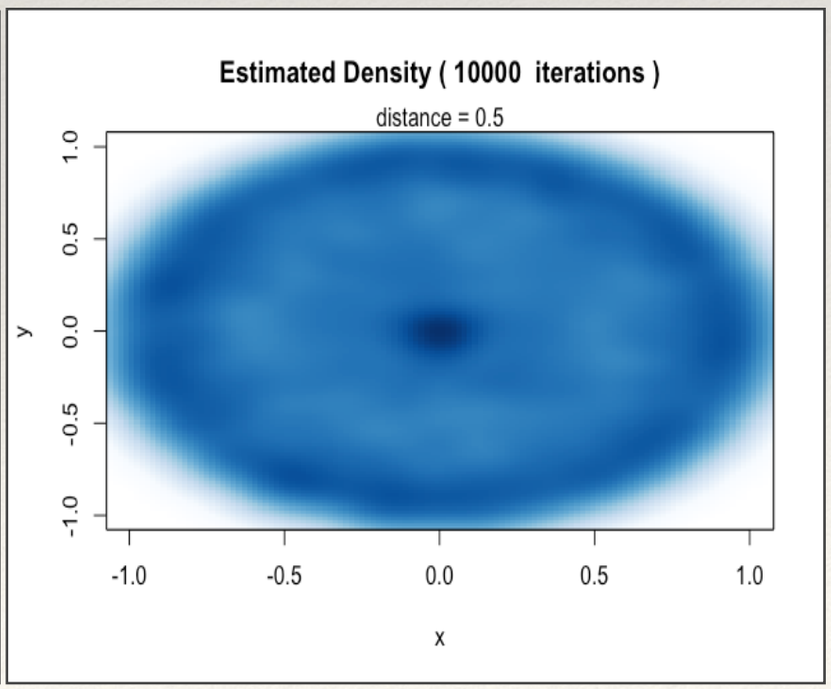
Hypothesis: Varying d

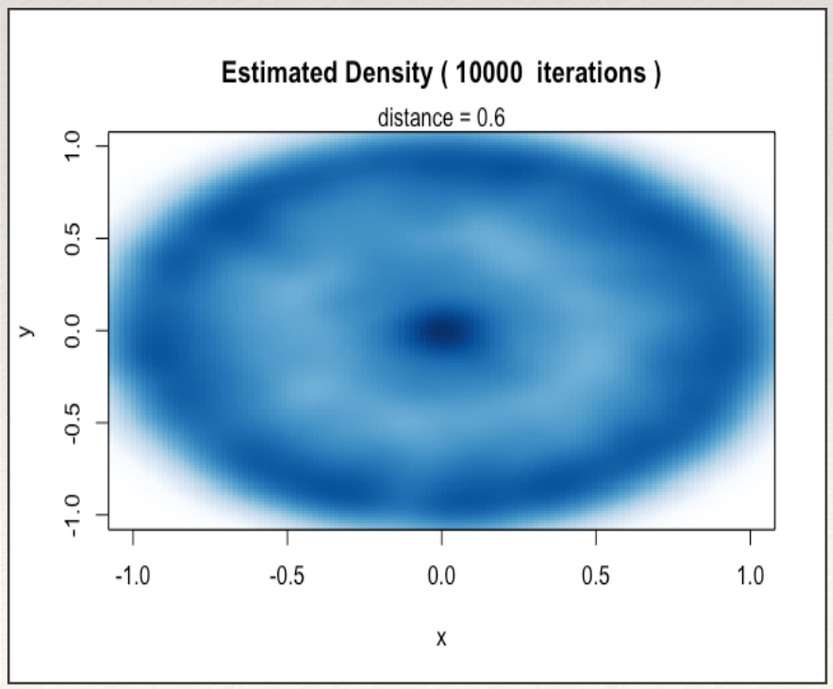












Terminology

- MCMC Chain := The samples that have been recorded as (*approximately*) representative sample values of the target (equilibrium) distribution
- Proposal Distribution := The distribution used to determine/*propose* the next state in the Markov Chain (distribution of the transition probabilities at the current state)
- Candidate State := The proposed next state in the chain
- Mixing := The rate of convergence* towards the stationary distribution

Hastings-Metropolis (Symmetric)

- Generate candidate state x' (sample) using symmetric proposal (such as MVN) centered at current state x
- Instead of automatically accepting the candidate state in the next state of the chain (Gibbs), compare it with the previous candidate x
- Accept the new state with probability:

$$\alpha = min(\frac{f(x')}{f(x)}, 1)$$

Note: To get a random-ish initial state for the chain, there's usually a "burnin" period where a large number of states are generated but are discarded from being recorded

Optimizing MCMC

- There is a lot of ongoing research on how to "optimize" MCMC implementations
- Where "optimize" fundamentally means…:
  - Increasing the *convergence* rate* of approximating the stationary probabilities (also called *rapid mixing)*
  - Decreasing the algorithmic complexity of the different MCMC components (CS)
  - Proving theoretical convergence for more complex MCMC algorithms

Simple Optimization Examples (1)

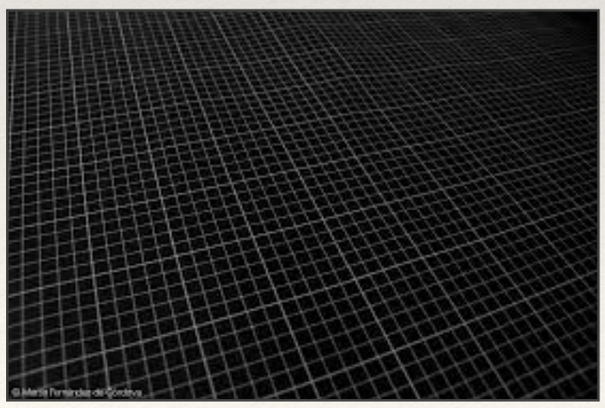
- Symmetric Hasting-Metropolis - each proposed candidate state is dependent on the previous state…

Because it's a Markov Chain

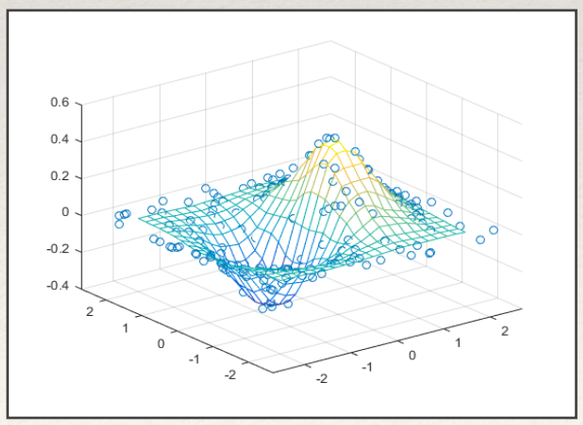
- … But this means the samples are **not independent**, thus they could be a biased representation of the stationary distribution
- Solution: Instead of 'recording' each accepted candidate, only accept 1 out of every *n* samples (discard the rest)
- Process known as "thining" or "spacing"

Simple Optimization Examples (2)

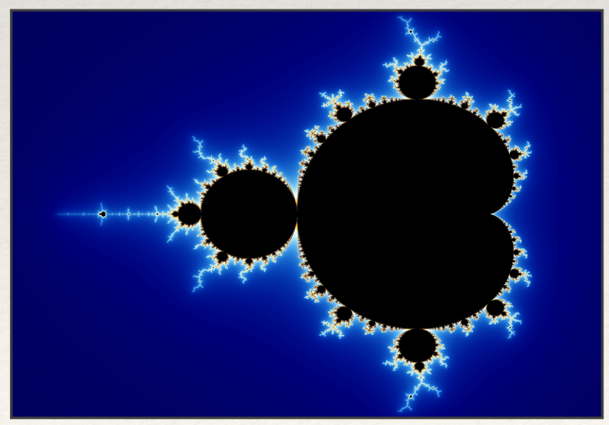
- Symmetric Hasting-Metropolis
  - Different state spaces converge* at different rates based on their composition
  - One random-walk MCMC variant may perform extremely well on some state spaces, but poorly with others



≠



≠



Simple Optimization Examples (2)

- Possible Solution:
  - Recall that a MVN, centered at the current state, is often used to *propose* the next candidate state (jump around the state space)
  - Also recall that there is a potentially **large burnin period** where samples are thrown away
    - Idea: Why not create a *time series* of the samples collected during the burnin period, measure the autocorrelation of each state dimension at different time points in the burnin…

$$R(s,t) = \frac{E[(X_t - \mu_t)(X_s - \mu_s)]}{\sigma_t \sigma_s}$$

  - … And then use this value to scale the variance-covariance matrix of the proposal distribution for each state-dimension

Improved MCMC Sampling Approaches

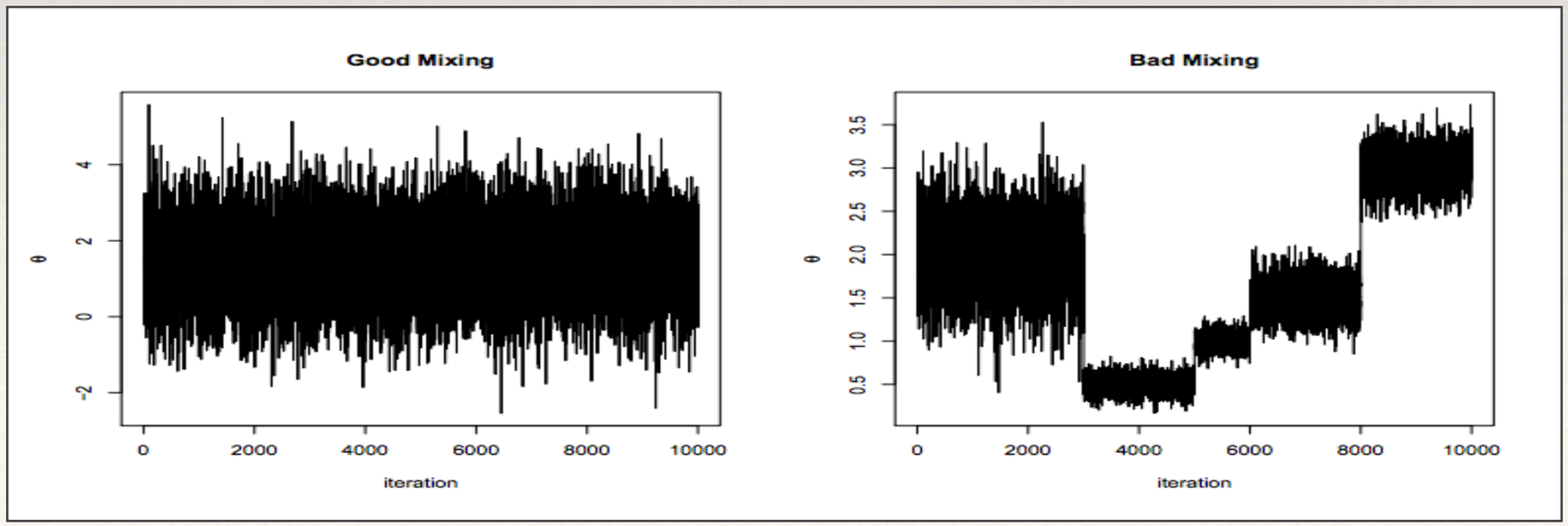
- Reversible Jump MCMC [5]
  - Like Hasting-Metropolis/Gibbs variant, but allows for jumps in differing dimensions, taking advantage of MC reversibility
    - Updates model parameters and some model indicator in each state change
- Birth/Death Process MCMC [6]
  - Added parameters represent births, deleted represent deaths
- Metropolis-Hastings-Green (MHG) & Metropolis-Hastings-Green with Jacobian (MHGJ) [4]

MCMC Convergence*

- It's difficult to assess whether or not a chain is mixing rapidly or not
- Many have quoted that you should aim at an acceptance rate of anywhere between 0.234 [2] - 0.44 [3] in HM MCMC approaches, but there is no perfect value
- A sign of convergence* would be the case where the MCMC chain doesn't erratically change behavior over time (random walk(s) means are relatively stable)

MCMC Diagnostics (1)

- Possible Solutions:
  - Run multiple chains in parallel via asynchronous, independent processors, compare their average differences in sample values across every chain over time via ANOVA [4]
  - Autocorrelation across a parameter's value in the R.W at multiple time points $\rho_k = \frac{\sum_{i=1}^{n-k}(x_i - \bar{x})(x_{i+k} - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$



MCMC Diagnostics (2)

- Geweke Diagnostic
  - Compares two non-overlapping partitions of the chain, computes mean and does a difference in means hypothesis test
- Gelman and Rubin Diagnostic
  - Compares within-chain and between-chain variance
  - Estimates a “Potential Scale Reduction Factor” for each parameter
  - PSRF nears 1 with convergence
- Raftery and Lewis Pseudo-Diagnostic
  - Given a quantile, quantile tolerance range, and probability of the MC chain being within said quantile, gives minimum iteration needed to run a pilot chain…
  - …pilot chain results can then be used to reveal number of burnin and regular iterations needed from MCMC chain to estimate the quantile of interest
  - Gives indicator as well to assess probability of chain never converging to the degree of accuracy given

Application: Non-convex Optimization

- Suppose you have a bounded function *f(x, y) = z* and you want to find the optimal parameter value(s) x' and y', if any, such that:

$$(x_1, \ldots, x_n) \text{ and } (y_1, \ldots, y_n) \geq f(x, y) \ \forall (x, y), x \in X, y \in Y$$

- Problems:
  - Can't formulate or solve analytically
  - Can't differentiate it + convexity not guaranteed (gradient descent is out)
- Possible Solution:
  - Use an optimized MCMC to estimate the distribution of *z* (and the distribution of which parameters produce relatively maximal *z* values) to eliminate large portions of the search space (and isolate local optima), then use an optimized hill-climbing algorithm (such as a binary search)

*Convergence

- Used the word “convergence” several times
- Can random sampling technically “converge” to an optimal value(s) with 100% confidence, truly?
- Many diagnostic checks report some measure of “convergence”
- There is always some residual effect of the starting state
- Some authors have pointed out that some of these diagnostics should be considered as “[hypothesis] tests for lack of divergence” rather than tests for convergence