CMPE 58N - Lecture 0.
Monte Carlo methods

Introduction, Course structure, Motivating Examples, Applications

Department of Computer Engineering,

Boğaziçi University, Istanbul, Turkey

Instructor: A. Taylan Cemgil

Goals of this Course

- Provide a basic understanding of underlying principles of Monte Carlo computation
- Orientation in the literature
- Focus on computational techniques rather than technical details,
  - ... the focus is not on proofs
  - ... but there will be some maths
    - Probability Theory
    - Statistics
    - Calculus and Linear Algebra
- Sharpening your intuition

Topics

- Markov Chain Monte Carlo
- Sequential Monte Carlo
- Probability theory
  - General background
  - Applications

Main study materials

- Handouts, Papers
- Jun S. Liu, Monte Carlo Strategies in Scientific Computing, 2001, Springer.
- Adam M. Johansen and Ludger Evers (edited by Nick Whiteley), Monte Carlo Methods, Lecture notes, University of Bristol

  `http://www.maths.bris.ac.uk/~manpw/teaching/notes.pdf`

- Information Theory, Inference, and Learning Algorithms
  David MacKay, Cambridge University Press – fourth printing (March 2005)

  `http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html`

General background about probability theory

- Geoffrey Grimmet and David Stirzaker, Probability and Random Processes, (3rd Ed), Oxford, 2006
  - Companion book containing 1000 exercises and solutions
- Grinstead and Snell, Introduction to probability available freely online!

`http://www.dartmouth.edu/~chance/teaching_aids/books_articles/probability_book/book.htm`

Main Book on Monte Carlo techniques

- Jun S. Liu,
  Monte Carlo Strategies for Scientific computing,
  Springer 2004
  - Short book
  - Covers almost everything we will mention on MCMC and SMC + more
  - Rather dense and Is not very easy to read

Other Books on Monte Carlo techniques

- Gilks, Richardson, Spiegelhalter, *Markov Chain Monte Carlo in Practice*, Chapman Hall, 1996
- Doucet, de Freitas, Gordon, *Sequential Monte Carlo Methods in Practice*, Springer, 2001

Tutorials and overviews (check course web page)

- Andrieu, de Freitas, Doucet, Jordan. *An Introduction to MCMC for Machine Learning*, 2001
- Andrieu. *Monte Carlo Methods for Absolute beginners*, 2004
- Doucet, Godsill, Andrieu. ”On Sequential Monte Carlo Sampling Methods for Bayesian Filtering”, Statistics and Computing, vol. 10, no. 3, pp. 197-208, 2000

Course Structure

- Web page

  `http://www.cmpe.boun.edu.tr/courses/cmpe58N/2009spring/`
- Required Work
  - Weekly Assignments (Reading, Programming, Analytic Derivations)
  - A project proposal and outline
  - Final Project: Implementation and Report
- Testing
  - 1 Midterm (in class), 1 Final (take home)
- Grading
  - Relative weights
    - % 25 Midterm
    - % 25 Take home final exam
    - % 50 Assignments, Quizzes and Final Project

Possible Topics

- In one application area (including but not limited to)
  - Scientific data analysis (DNA, Bioinformatics, Medicine, Seismology)
  - Robotics, Navigation, Self Localisation
  - Signal, Speech, Audio, Music Processing
  - Computer Vision (Object tracking)
  - Information Retrieval, Data mining, Text processing, Natural Language Processing
  - Sports, Finance, User Behaviour, Cognitive Science e.t.c.
- Reading a paper and writing a tutorial-like summary in own words and self designed examples
- Implementation and comparative study of inference algorithms on synthetic data

Remarks

- If you have already chosen a research topic
  - Use the project of this course to implement and write up a component of your work!
- If you have **not** chosen research/thesis topic but roughly have something in mind or simply don't know yet
  - Come and talk to me to clarify a topic/technique
  - Study/learn a few inference techniques more in depth
    - Never underestimate the insight gained from a well designed toy example
  - Investigate the feasibility/suitability of Monte Carlo techniques for your purposes

Remarks

- Ideally, a good report could be presented with some extensions at a national or international conference
  - Some well-known methods were master theses once,
  - Occasions when a forth year project report was published (and cited later!)
- Use TEX or LATEX.
  - If you are serious with research in computer science, statistics or engineering but are using other ways of document preparation, it is very likely that you are wasting some of your valuable time.

Remarks

- Any programming language or other system for computation and visualisation
  - Matlab (preferred)
  - Octave
  - Java,
  - C/C++, BLAS, ATLAS, GNU Scientific Library

Applications of Monte Carlo

- Statistics, Bioinformatics
- Signal Processing, Machine learning
- Seismology, Acoustics
- Computer Science, Analysis of algorithms, Randomized algorithms
- Networks, System simulation
- Robotics, Tracking, Navigation
- Econometrics, Finance
- Operations Research, Combinatorics, Optimisation
- Physics, Chemistry, Computational Geometry
- Environmental sciences, monitoring

Bayesian Statistics

- Computation of analytically intractable high dimensional integrals
- Inference, Model selection

Probabilistic Inference

- **expectations** of functions under probability distributions: **Integration**

$$\langle f(x) \rangle = \int_{\mathcal{X}} dx p(x) f(x) \qquad \langle f(x) \rangle = \sum_{x \in \mathcal{X}} p(x) f(x)$$

- **modes** of functions under probability distributions: **Optimization**

$$x^* = \operatorname*{argmax}_{x \in \mathcal{X}} p(x) f(x)$$

- However, computation of multidimensional integrals is hard

Combinatorics

- Counting

Example : What is the probability that a solitaire laid out with 52 cards comes out successfully given all permutations have equal probability ?

$$|A| = \sum_{x \in \mathcal{X}} [x \in A] \qquad [x \in A] \equiv \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

$$p(x \in A) = \frac{|A|}{|\mathcal{X}|} = \frac{?}{\approx 2^{225}}$$

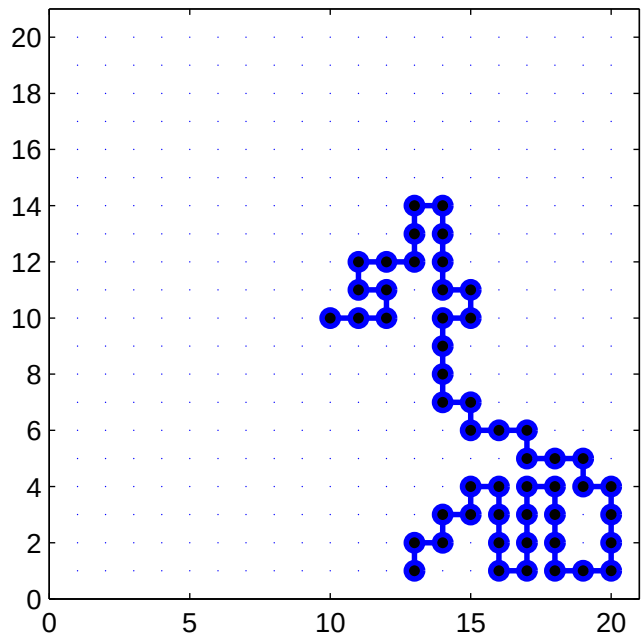
Random Combinatorial Objects

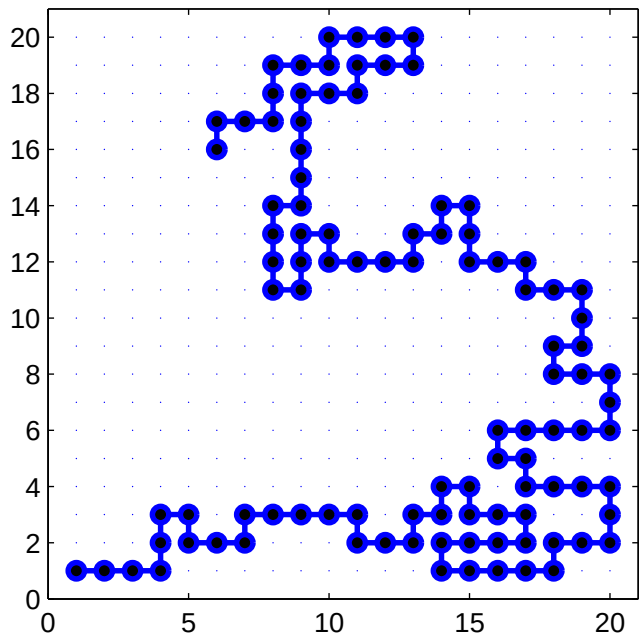
Generate uniformly from

- Self avoiding random walks on a $N \times N$ grid
- All spanning trees of a graph
- Binary trees with N nodes
- Directed Acyclic Graphs

Self avoiding random walks

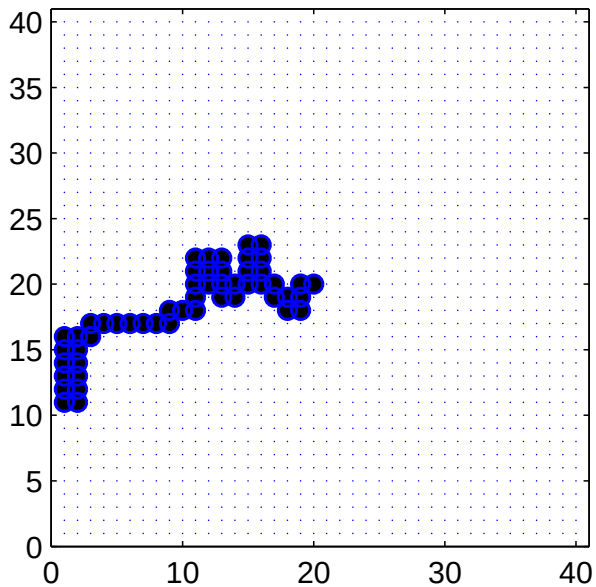
- How many different ways are there to place a chain with M nodes on an $N \times N$ 2-D rectangular grid ?
- In 3-D, problem is relevant for understanding protein folding

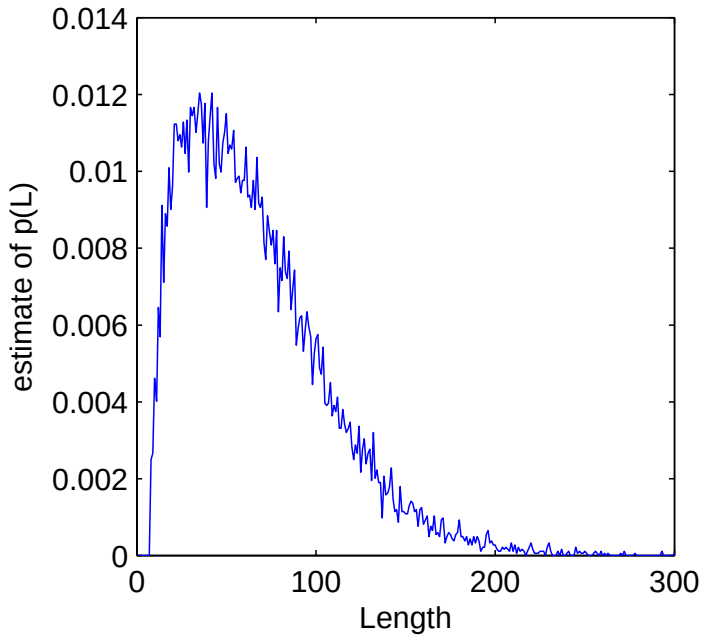




Self avoiding random walks

- S





Random Spanning Trees

- Given an undirected graph, generate a spanning tree **uniformly** from the set of all spanning trees
- (Aldous and Fill):

  Run a random walk on the graph until all verticies have been visited,

  Include the edge that the walk first visited v

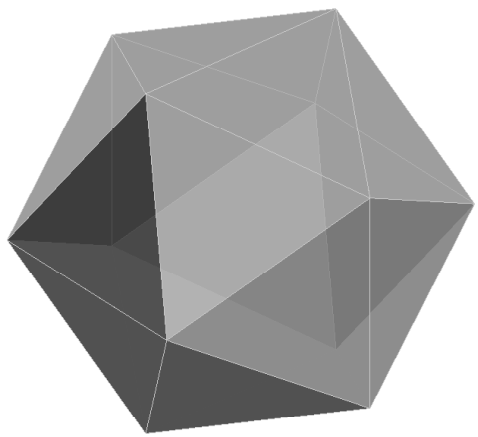
  It turns out that the spanning tree generated like this is an uniform draw.

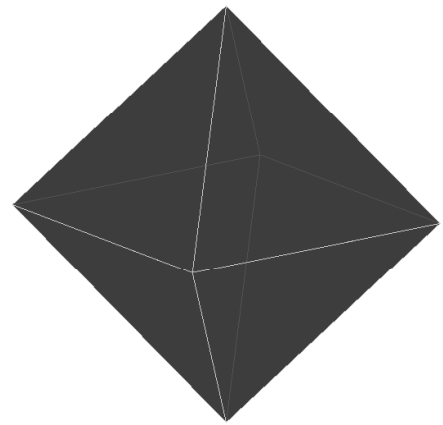
Geometry

- Given a simplex S in N dimensional space by

$$S = \{x : Ax \leq b, \quad x \in \mathbb{R}^N\}$$

find the Volume $|S|$





Sampling uniformly from a set S

- Suppose we have a black box implementation of an indicator function $[x \in S]$
- How can we generate uniform samples from S?
- It turns out that the following algorithm works (in principle)

  Choose an arbitrary $x^{(0)} \in S$
  For $i = 1, 2, \ldots$

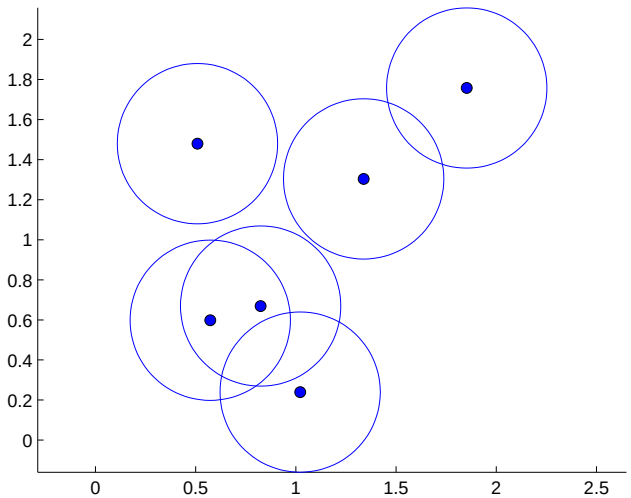
    Propose:
    $\epsilon_i \sim \mathcal{N}(0, V)$
    $x' \leftarrow x^{(i-1)} + \epsilon_i$
    Accept/Reject
    if $[x' \in S]$ then $x^{(i)} \leftarrow x'$ else $x^{(i)} \leftarrow x^{(i-1)}$ endif

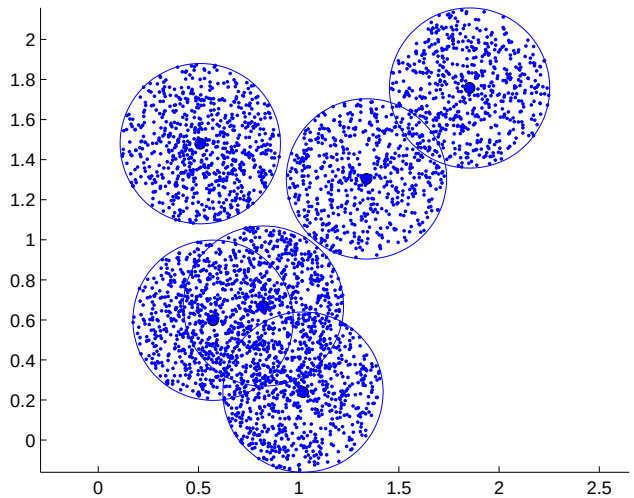
  EndFor

  - $x^{(i)}$ are the desired samples! Why?

Sampling uniformly from a set S

$$S = \{x : \|c_i - x\| \leq \rho\}$$



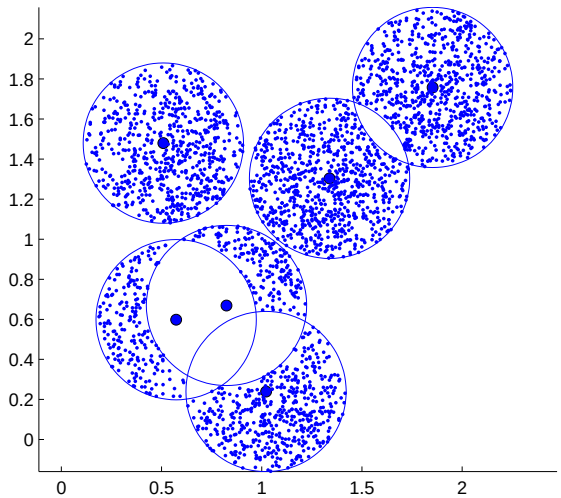


Sampling uniformly from a set S

```
x = c(:,1);
for i=1:5000,
    xhat = x + 0.2*randn(2,1);
    % Inclusion test
    e = c - repmat(xhat, [1 N]);
    d = sqrt(sum(e.^2,1));
    if any(d<rho),
      x = xhat;
      line(x(1), x(2), 'marker', '.');
    end;
end;
```

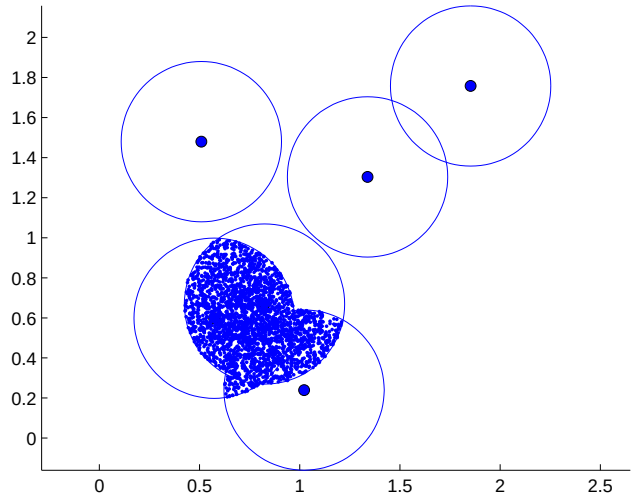
Sampling uniformly from a set S

Set of points that are close only a single center.
$S = \{x : \|c_i - x\| \leq \rho \text{and} \|c_j - x\| \geq \rho \text{for} i \neq j\}$



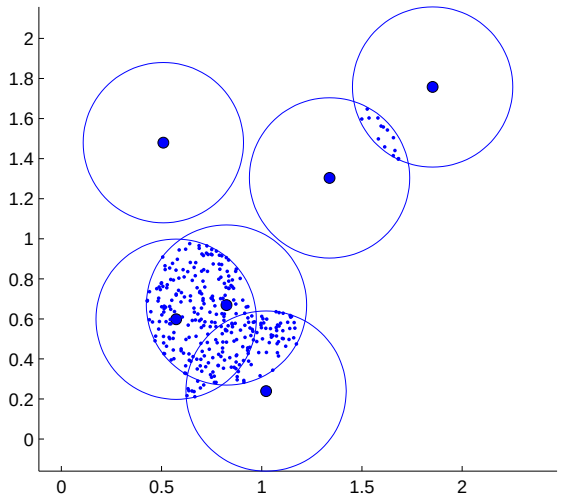
Sampling uniformly from a set S

Set of points that are close to two or more centers.



Sampling uniformly from a set S

```
%     xhat = x + 0.2*randn(2,1);
      xhat = x + 0.9*randn(2,1);
```



Matrix Permanent

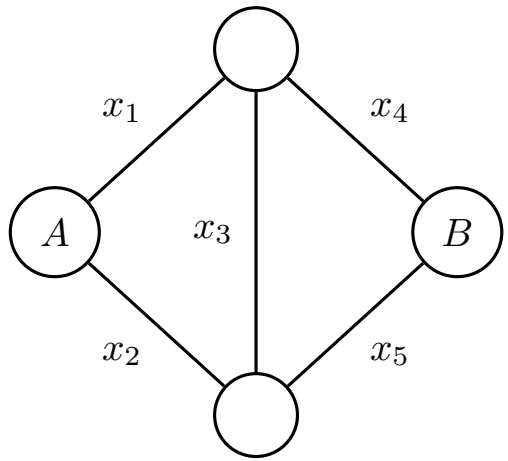
- We define a so-called *restriction matrix* A where $A_{i,j} \in \{0, 1\}$.
- We think of A as an adjacency matrix of a bipartite graph $\mathcal{G}_A = (\mathcal{V}_s, \mathcal{V}_t, \mathcal{E})$
- $A_{i,j} = 1 \Leftrightarrow s_i \in \mathcal{V}_s, t_j \in \mathcal{V}_t, (i,j) \in \mathcal{E}$
- permanent(A) = total number of perfect matchings on $\mathcal{G}_A$
- (Vailant 1977) Problem is $\sharp P$ (harder than NP!). But Jerrum et.al. developed a polynomial time randomised algorithm based on simulating a Markov chain with known mixing time!

Network analysis, Rare Events

- Given a graph with random edge lengths

$$x_i \sim p(x_i)$$

Find the probability that the **shortest path from A to B** is larger than γ.



CMPE 58N - Lecture 1.
Monte Carlo methods

Introduction to Monte Carlo method, Motivating Examples, Law of Large Numbers, Central Limit Theorem

Department of Computer Engineering,

Boğaziçi University, Istanbul, Turkey

Instructor: A. Taylan Cemgil

Outline

- Introduction to Monte Carlo method,
- Motivating Examples,
- Law of Large Numbers,
- Central Limit Theorem
- Example

Monte Carlo Methods

- Represent the solution of a problem as a parameter of a hypothetical population,
- use a pseudo-random sequence of numbers to construct a sample of a population, from which statistical estimates of the parameter can be obtained
- Stochastic Simulation or Sampling methods

History of Monte Carlo methods

1733 Buffon's needle problem.

1812 Laplace suggests using Buffon's needle experiment to estimate π.

1946 ENIAC (Electronic Numerical Integrator And Computer) built.

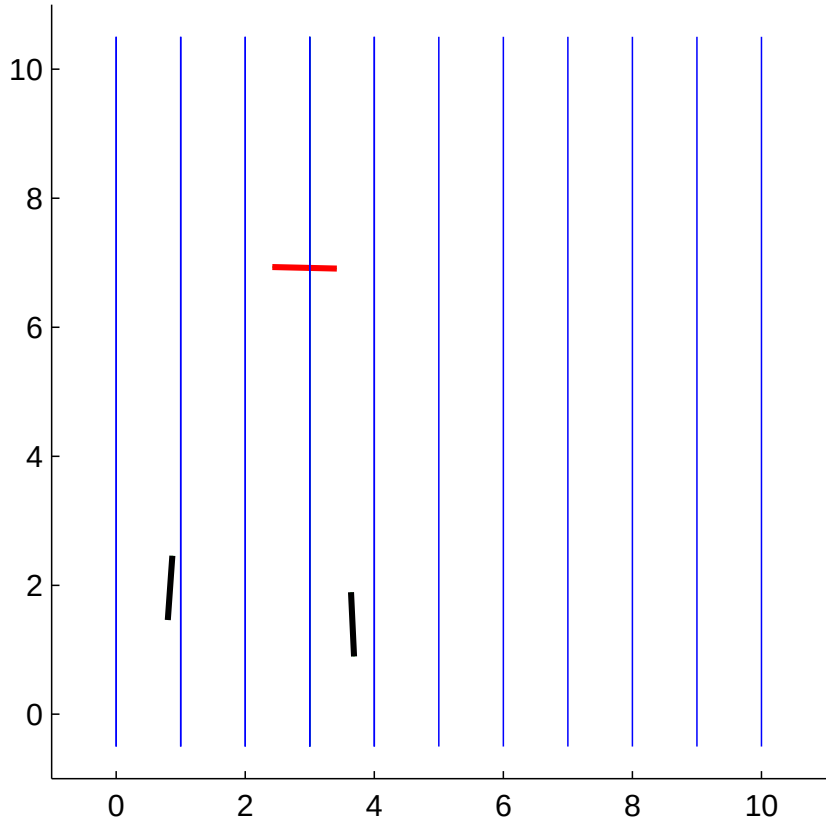
1947 John von Neuman and Stanislaw Ulam propose a computer simulation to solve the problem of neutron diffusion in fissionable material.

1949 Metropolis and Ulam publish their results in the Journal of the American Statistical Association.

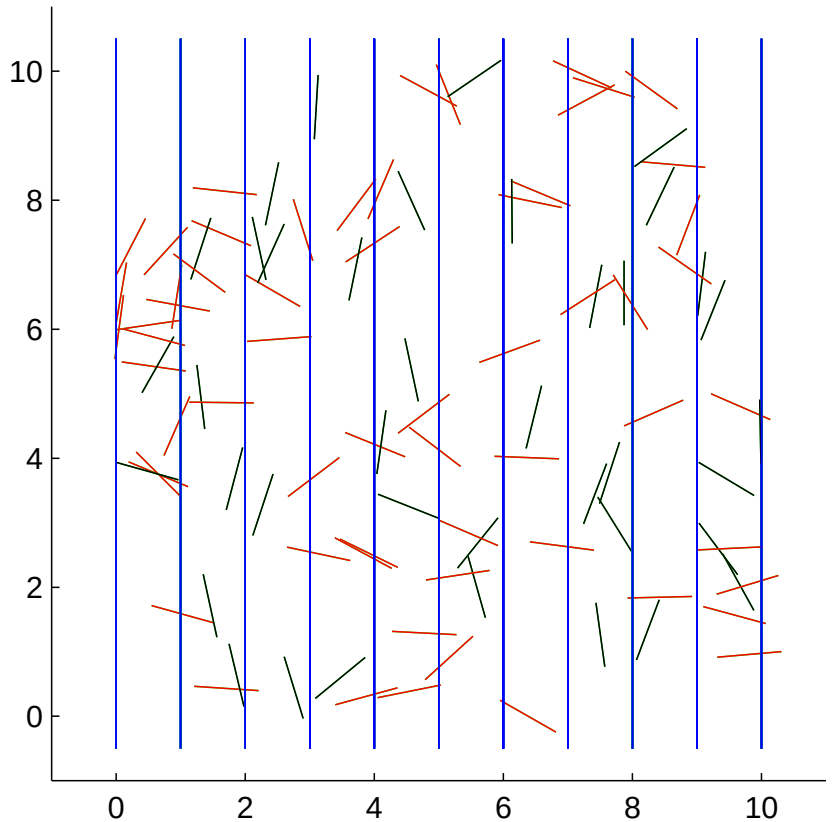
1984 Geman & Geman publish their paper on the Gibbs sampler

. . . continuously growing interest with increases in computational power

Buffon's needle



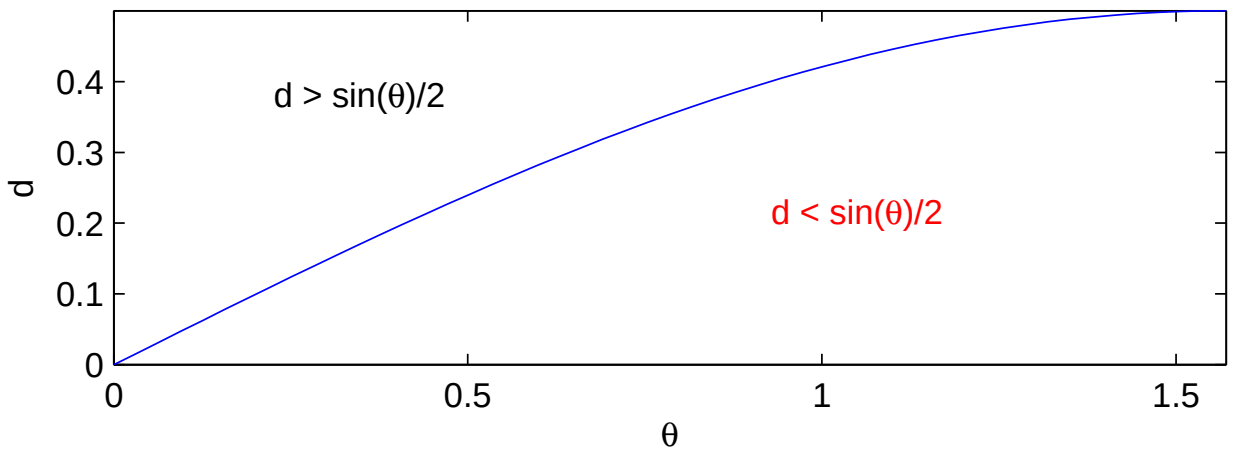
Buffon's needle



Buffon's needle

- d : Distance from the middle of the needle to the nearest line
- θ : Acute angle between the parallel lines and the needle
- A needle touches a line iff

$$\frac{d}{\sin\theta} < \frac{1}{2}$$



Buffon's needle

- The area of the rectangle is

$$S = \frac{1}{2}\frac{\pi}{2}$$

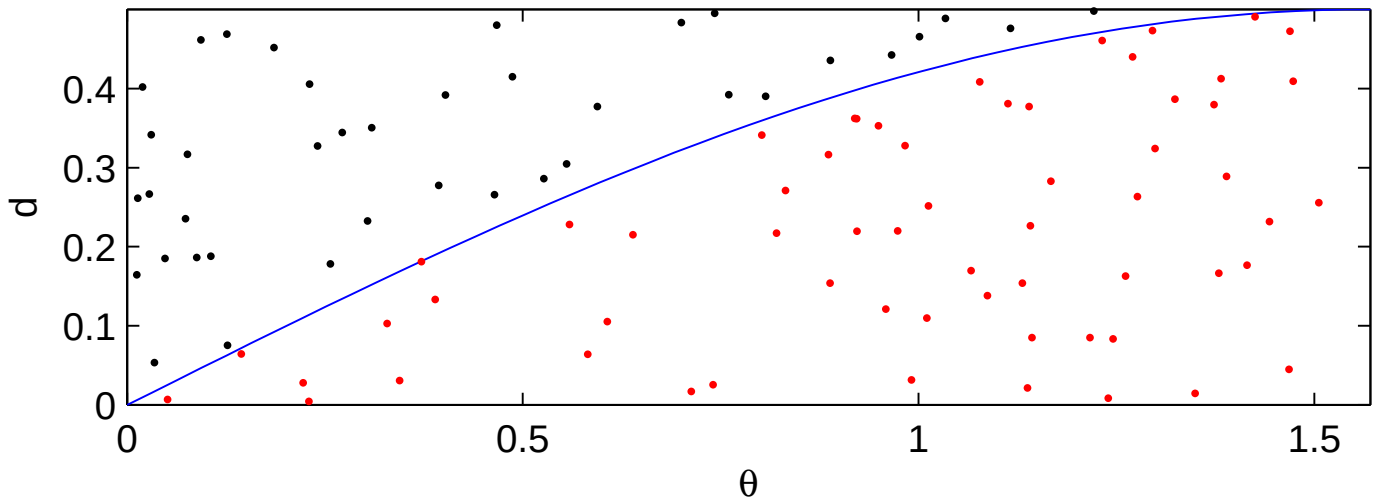
- The area under the sin is

$$\int_0^{\pi/2} \sin(\theta)/2 \quad = \quad \frac{1}{2}$$

-

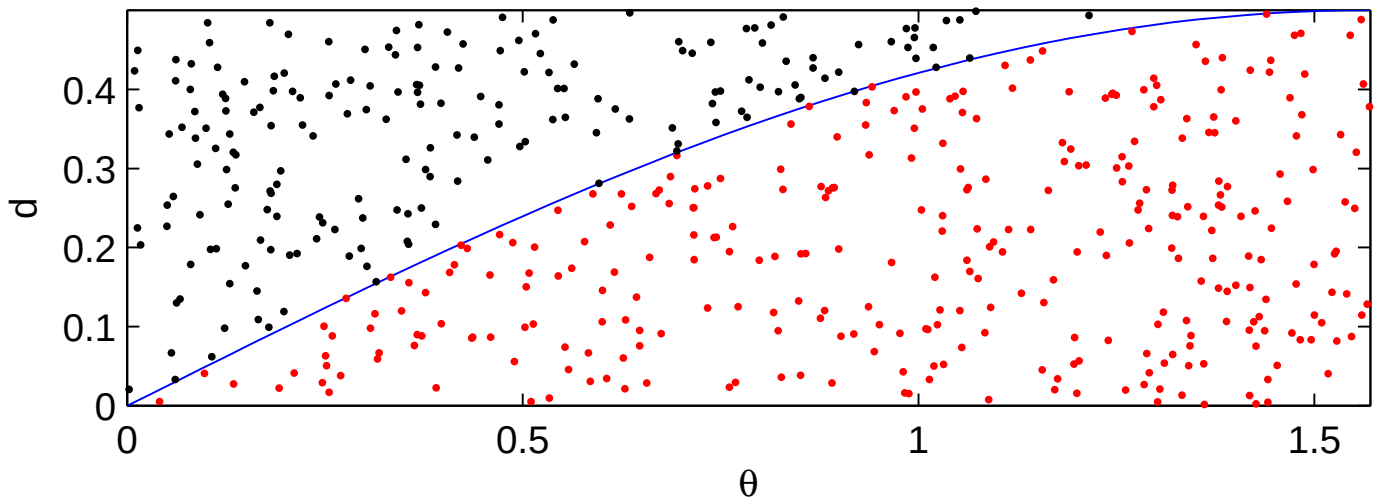
$$\Pr\{d < \sin(\theta)/2\} \quad = \quad \frac{1/2}{\pi/4} = \frac{2}{\pi}$$

Buffon's needle



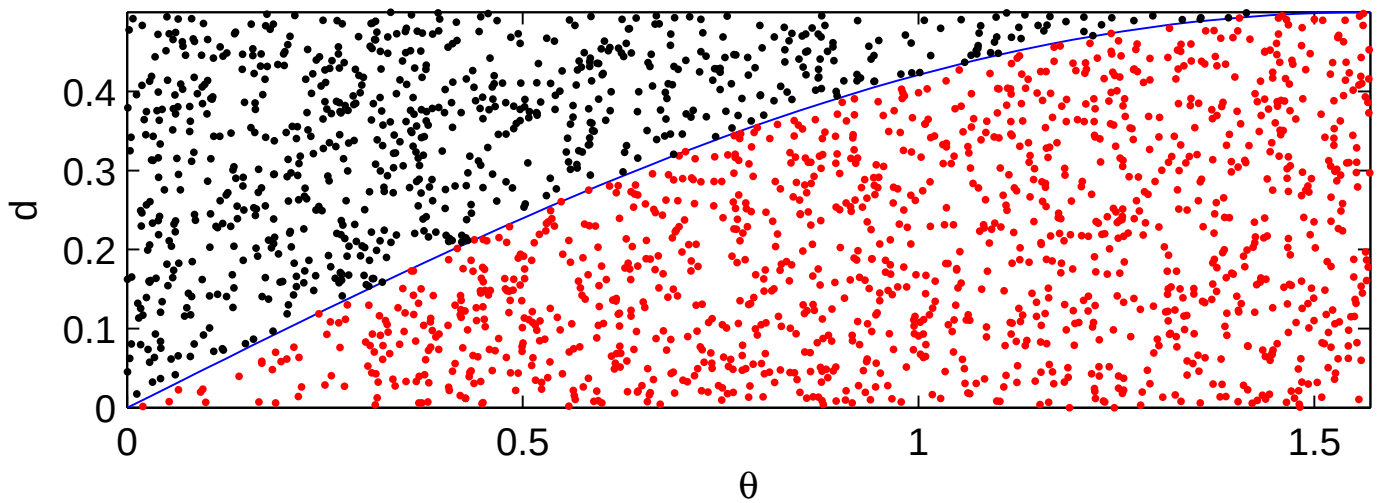
$$\pi \approx 3.2787$$

Buffon's needle



$$\pi \approx 3.1949$$

Buffon's needle



$$\pi \approx 3.1596$$

Indicator function

$$\mathbb{I}\{x\} = \begin{cases} 1 & x \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

Alternative notation: Iverson convention

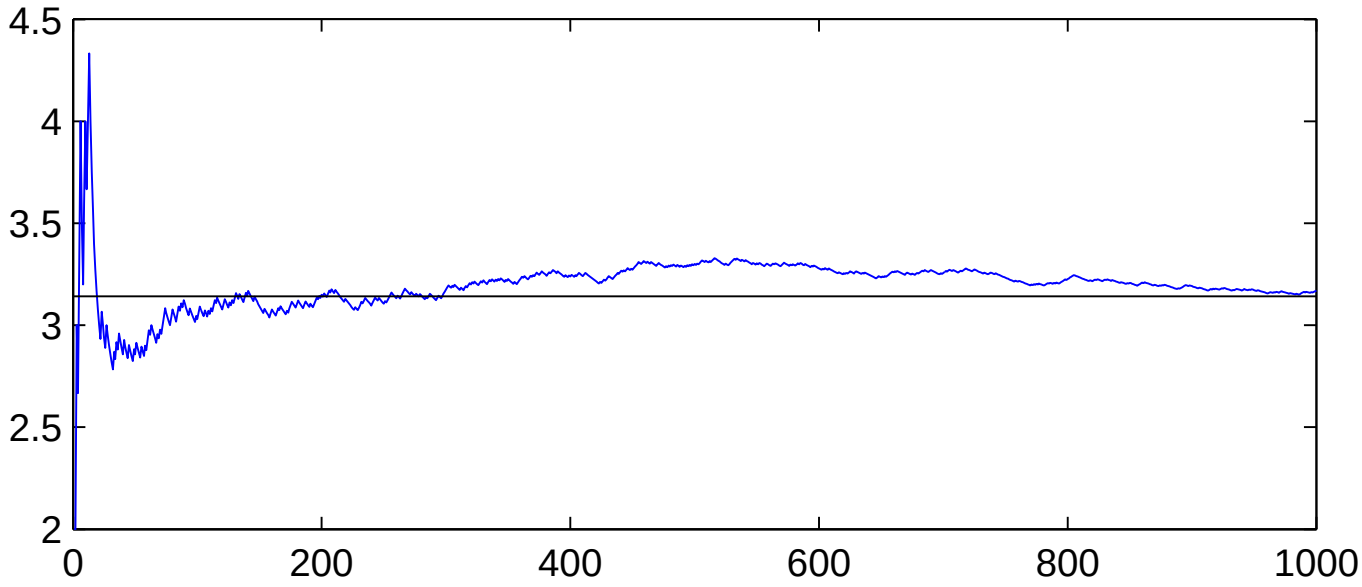
$$[x] = \begin{cases} 1 & x \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

Buffon's needle

- Draw $(d^{(n)}, \theta^{(n)}) \sim U_S$ and estimate π via

$$\begin{aligned} \pi &= \frac{2}{\Pr\{d < \sin(\theta)/2\}} \approx \frac{2\text{\# of all dots}}{\text{\# of red dots}} \\ &= \frac{2N}{\sum_{n=1}^{N} \mathbb{I}\{d^{(n)} < \sin(\theta^{(n)})/2\}} \end{aligned}$$

Buffon's needle



Speed of convergence

- Monte Carlo integration: error behaves as $n^{-1/2}$.
- Numerical integration of a one-dimensional function by Riemann sums: error behaves as n^{-1}.
- For one-dimensional problems Riemann is better; however deteriorates with increasing dimension: curse of dimensionality.
- Order of convergence of Monte Carlo integration is **independent of the dimension of the problem**.
  $\rightsquigarrow$ Monte Carlo methods can be a good choice for high-dimensional integrals.

Convergence of random variables

(Liu, Appendix A.1.4.)

$$y_n \sim p_n(y_n) \qquad F_n(y_n) = \int_{-\infty}^{y_n} p_n(\tau)d\tau$$

1. Convergence in distribution

$$\lim_{n\to\infty} F_n(y_n) = F(y)$$

2. Convergence in probability

$$\lim_{n\to\infty} \Pr(|y_n - y| > \epsilon) = 0$$

3. Convergence almost surely

$$\Pr(\lim_{n\to\infty} |y_n - y| = 0) = 1$$

- $3 \Rightarrow 2 \Rightarrow 1$

Convergence of Random variables

- Convergence of random variables is a delicate subject
- Important to get a deeper understanding
- Not get intimidated while reading the literature; remember the definitions and different modes of convergence
- See, e.g., Grimmet and Stirzaker, Ch. 7

Law of Large Numbers

$X_1, \ldots, X_n, \ldots$ are i.i.d.

- Weak Law: $\langle X_i \rangle = \mu$

$$\frac{X_1 + \cdots + X_n}{n} \to \mu \quad \text{in probability}$$

- Strong Law: $\langle X_i \rangle = \mu$ and X_i **with finite variance**

$$\frac{X_1 + \cdots + X_n}{n} \to \mu \quad \text{a. s.}$$

Central Limit Theorem

X_i are i.i.d. with mean μ and variance σ^2

- $$\bar{X}_n = \frac{X_1 + \cdots + X_n}{n}$$

$$\frac{\sqrt{n}(\bar{X}_n - \mu)}{\sigma} \to \mathcal{N}(0,1)$$

- We have approximately

$$\bar{X}_n \sim \mathcal{N}(\mu, \sigma^2/n)$$

Chevalier de Méré

- The famous letters between Pascal and Fermat (start of probability) mention a request for help from a French nobleman and gambler, Chevalier de Méré.
- Méré bets for:

  in four rolls of a die, at least one six would turn up

- Later he bets for:

  in 24 rolls of two dice, a pair of sixes would turn up.

  but he was not happy with the latter schema

Chevalier de Méré

- Setup a computer simulation for a single die

```
K = 4; % Number of dice throws
N = 1000; % Number of games
for trial=1:10,
    D = ceil(rand(N,K)*6);
    disp(sum(sum(D==6, 2) > 0)/N)
end
```

- Per game, Méré won

```
0.4950, 0.4950, 0.5090, 0.5210, 0.5460
0.5420, 0.5360, 0.5160, 0.5210, 0.5010
```

Chevalier de Méré

The analytical solution

$$\begin{aligned} \Pr\{\text{Méré wins}\} &= 1 - \Pr\{\text{Méré loses}\} \\ &= 1 - (5/6)^4 = 0.5177 \end{aligned}$$

Chevalier de Méré

- Setup a computer simulation for a pair of dice

```
K = 24; % Number of dice throws
N = 1000; % Number of games
for trial=1:10,
    D = ceil(rand(N,K,2)*6);
    sum(sum(D(:,:,1)==6 & D(:,:,2)==6,2) > 0)/N
end
```

- Per game, Mere wins

```
0.502,  0.486, 0.497, 0.533, 0.521
0.474,  0.451, 0.508, 0.470, 0.481 ...
```

- Accurate results by simulation require a large number of experiments

Chevalier de Méré

The analytical solution

$$\Pr\{\text{Méré wins}\} \quad = \quad 1 - (35/36)^{24} = 0.4914$$

Therefore, 24 times is not a good bet. But with 25 (Pascal)

$$\Pr\{\text{Méré wins}\} \quad = \quad 1 - (35/36)^{25} = 0.5055$$

Chevalier de Méré

- What is the distribution of the estimate for N games ?
- V_n the outcome that Méré wins the n'th game

$$\begin{aligned} V_n &\sim \mathcal{BE}(V_n; p) \\ S_n &= \frac{V_1 + \cdots + V_n}{n} \end{aligned}$$

- Evoke the law of large numbers $\langle V_n \rangle = p$

$$S_n \rightarrow p \qquad n \rightarrow \infty$$

Chevalier de Méré

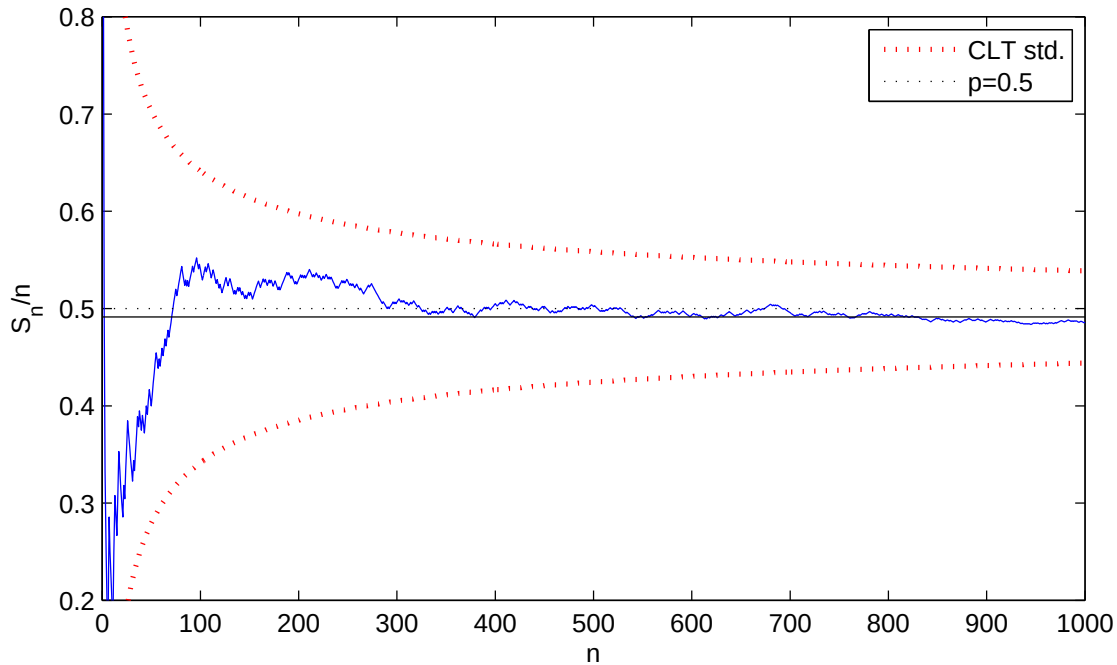
- Accuracy is given by the Central Limit Theorem

$$\begin{aligned} \langle V_n \rangle &= p \\ Var\{V_n\} &= p(1-p) \\ \sqrt{\frac{n}{p(1-p)}}(S_n - p) &\rightarrow \mathcal{N}(0,1) \end{aligned}$$

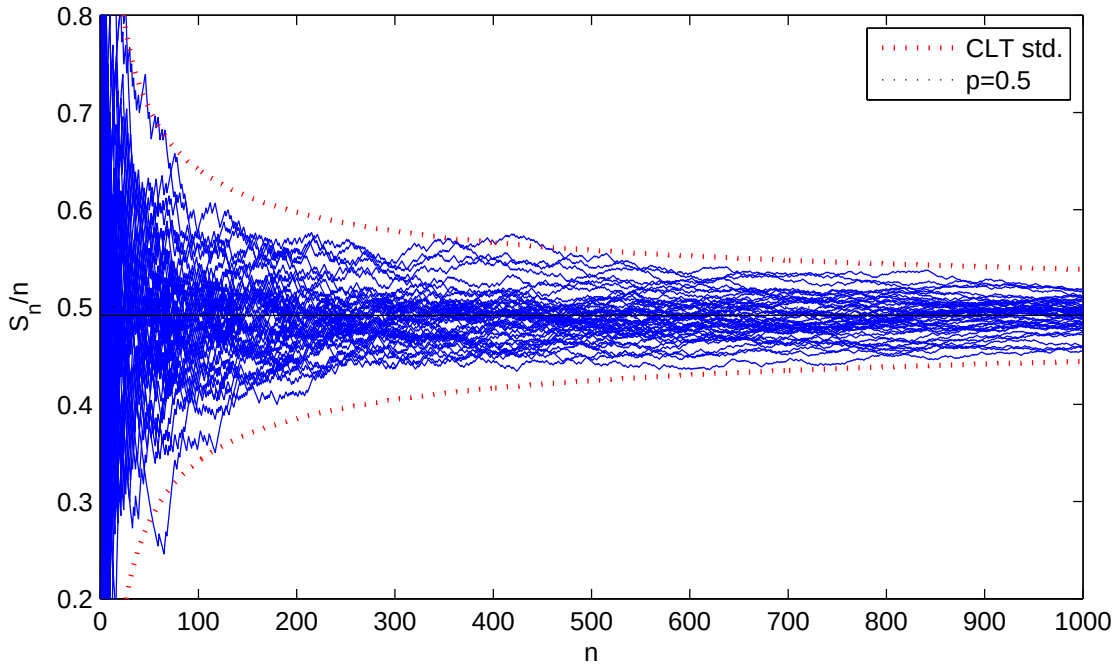
- Approximately

$$S_n \sim \mathcal{N}(p, p(1-p)/n)$$

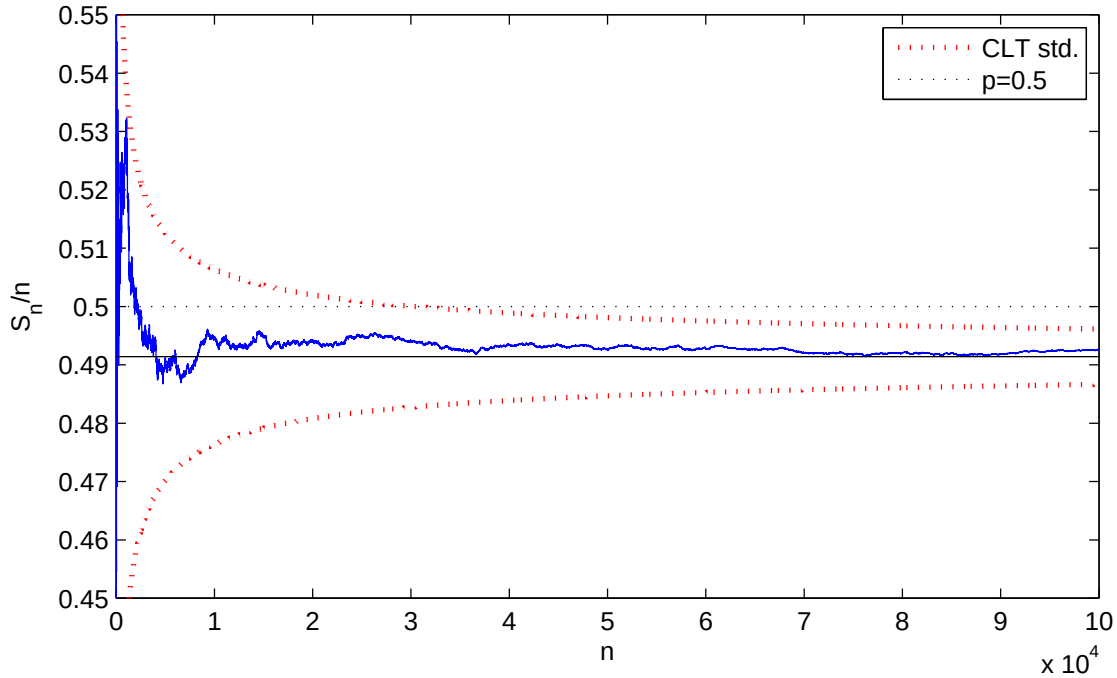
Chevalier de Méré



Chevalier de Méré (cont.)



Chevalier de Méré



- We need around 30000 games to say with about $\%99$ confidence that the game with 24 throws is truly unfavorable.

Summary

- Law of large numbers: Consistency.
- CLT: Provides information about the rate of convergence
- If we can draw N independent and identically distributed samples from a distribution $p(x)$, we can estimate expectations $E_p\left(\varphi(x)\right)$ with an error $O(N^{1/2})$, **independent** of the dimensionality of x.

CMPE 58N - Lecture 2
Monte Carlo methods

Random Number Generation, Rejection, Importance Sampling

Department of Computer Engineering,

Boğaziçi University, Istanbul, Turkey

Instructor: A. Taylan Cemgil

Fall 2009

Outline

- Random Number Generation
  - Pseudo-random numbers
  - Source of random bits: Generating Bernoulli Random Variables
- Inversion
- Transformation
- Rejection Sampling
- Importance Sampling

Random Number Generation

- Physical methods
  - throw dice, flip coins, shuffle playing cards, roulette wheel
  - thermal noise in Zener diodes or other analog circuits
  - Listen to atmospheric noise (www.Random.org)
  - Run a hash function against a frame of a video stream
  - . . .
- A random number generator (deterministic computation) to obtain numbers that “look” random
  - Efficient
  - Repeatable (seeds) – good for debugging

Pseudo-random number generator

- Linear congruential generator

$$\begin{aligned} Z_i &= (aZ_{i-1} + b) \mod M \\ X_i &= Z_i / M \end{aligned}$$

main flaw: the crystalline nature

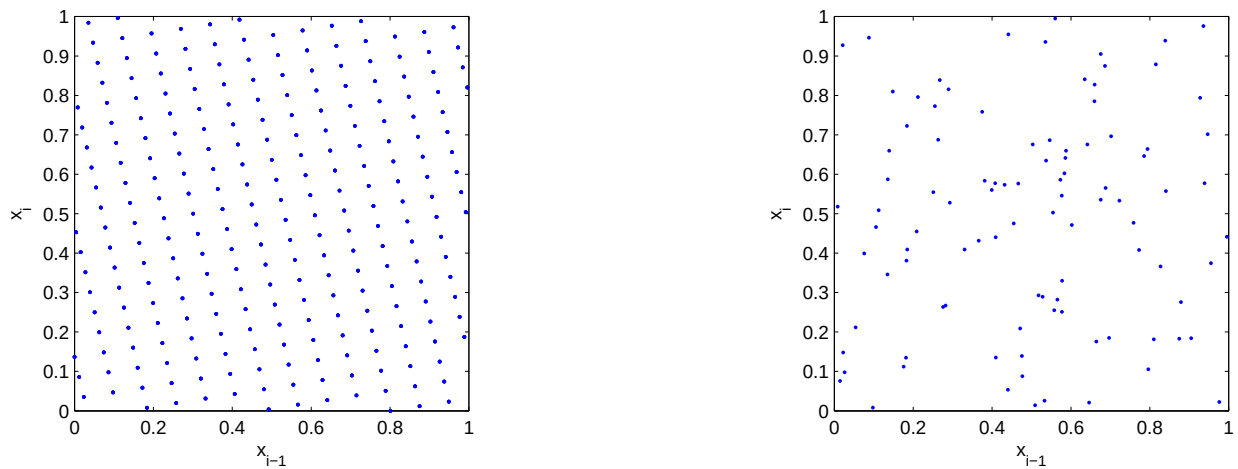


Figure: (left) $a = 81$; $b = 35$; $M = 256$;, (right) Matlab's `rand`

Remarks

A poor design which was popular during 1970's

$a = 2^{16} + 3; b = 0; M = 2^{31}$

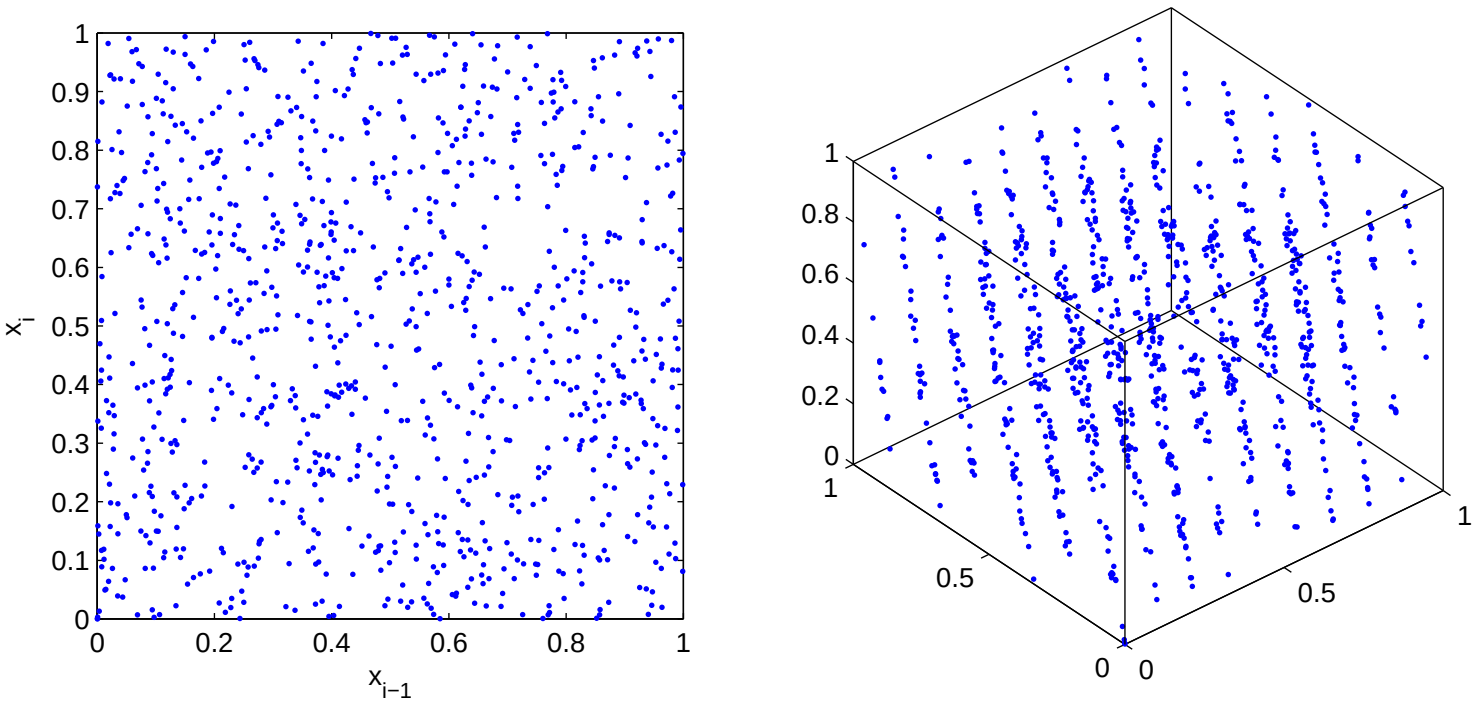


Figure: (left) (X_{i-1}, X_i), (right) (X_{i-2}, X_{i-1}, X_i)

Bernoulli Random Variables

- $x \in \{0, 1\}$

$$x \sim \mathcal{BE}(x; p) = p^x (1 - p)^{(1-x)},$$

- How to sample from a Bernoulli distribution on a computer given $p \in \mathbb{R}$ using samples from the uniform distribution $u \sim \mathcal{U}(u; 0, 1)$?

Bernoulli Random Variables

- $x \in \{0, 1\}$

$$x \sim \mathcal{BE}(x; p) = p^x (1-p)^{(1-x)},$$

$$u \sim \mathcal{U}(u; 0, 1)$$

$$x = u < p$$

Note that this is an idealisation as we can not represent irrational numbers on a computer.

The Knuth-Yao algorithm

- How to sample exactly from a Bernoulli distribution $x \sim \mathcal{BE}(x; p)$, $x \in \{0, 1\}$ on a computer given $p \in \mathbb{R}$ using a random bit source $\omega \sim \mathcal{BE}(\omega; 1/2)$?

The Knuth-Yao algorithm

Represent p in binary $p = 0.p_1 p_2 p_3 \ldots$

$i \leftarrow 0$

Repeat

$i \leftarrow i + 1$

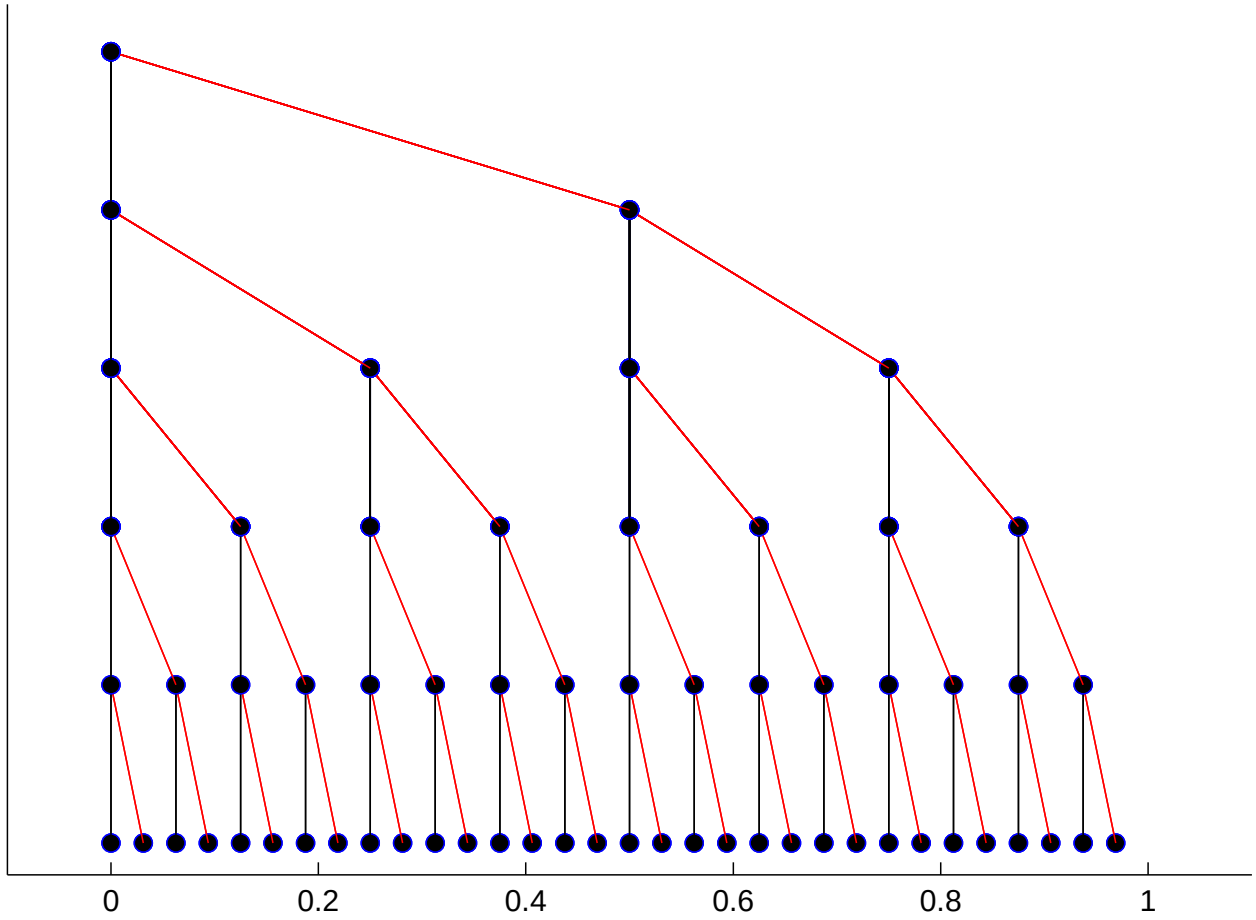
$\omega \sim \mathcal{BE}(\omega; 1/2)$

Until $\omega \neq p_i$

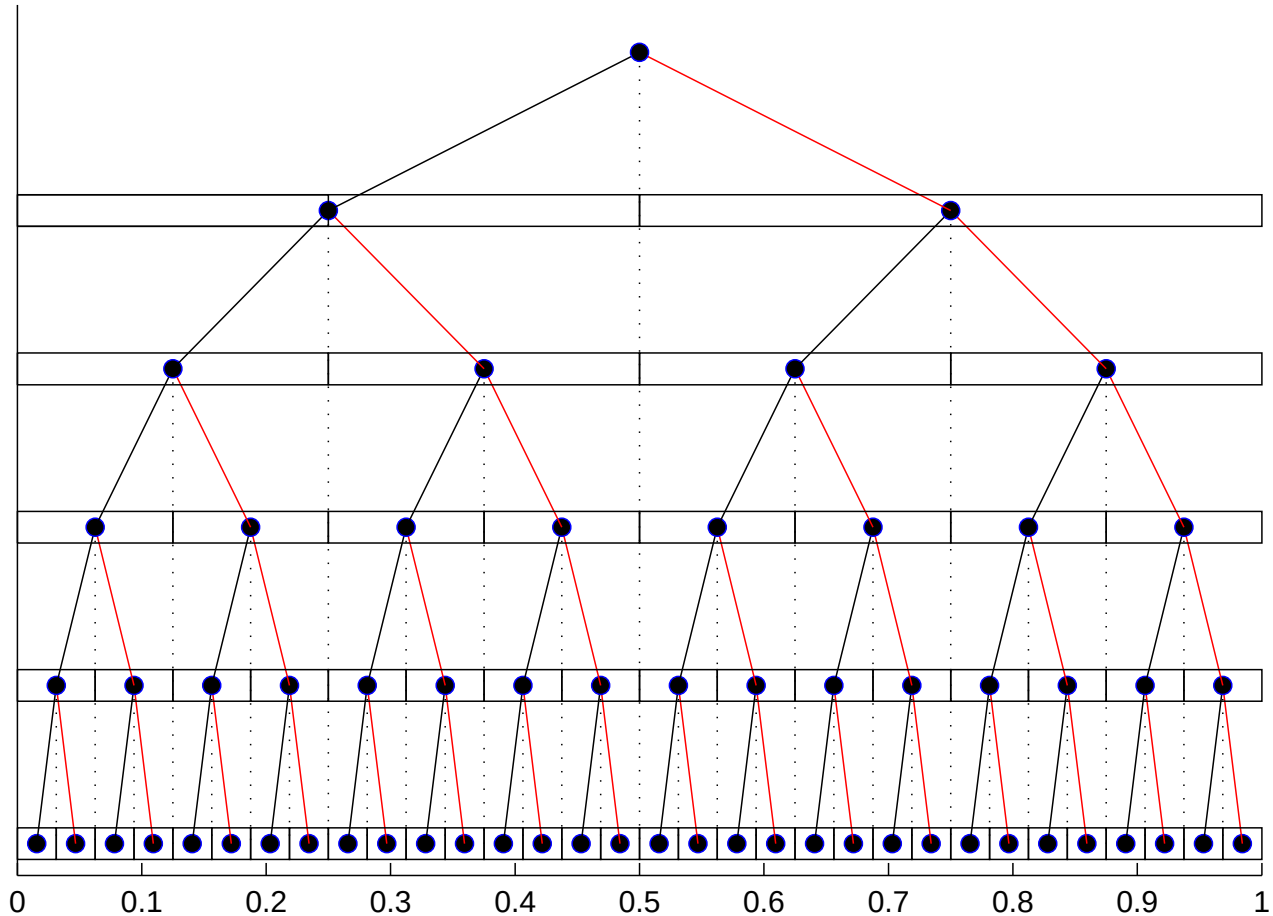
$x \leftarrow \omega < p_i$

See Luc Devroye, Non uniform random variate generation, available online, Ch. 15

The Knuth-Yao algorithm (cont.)



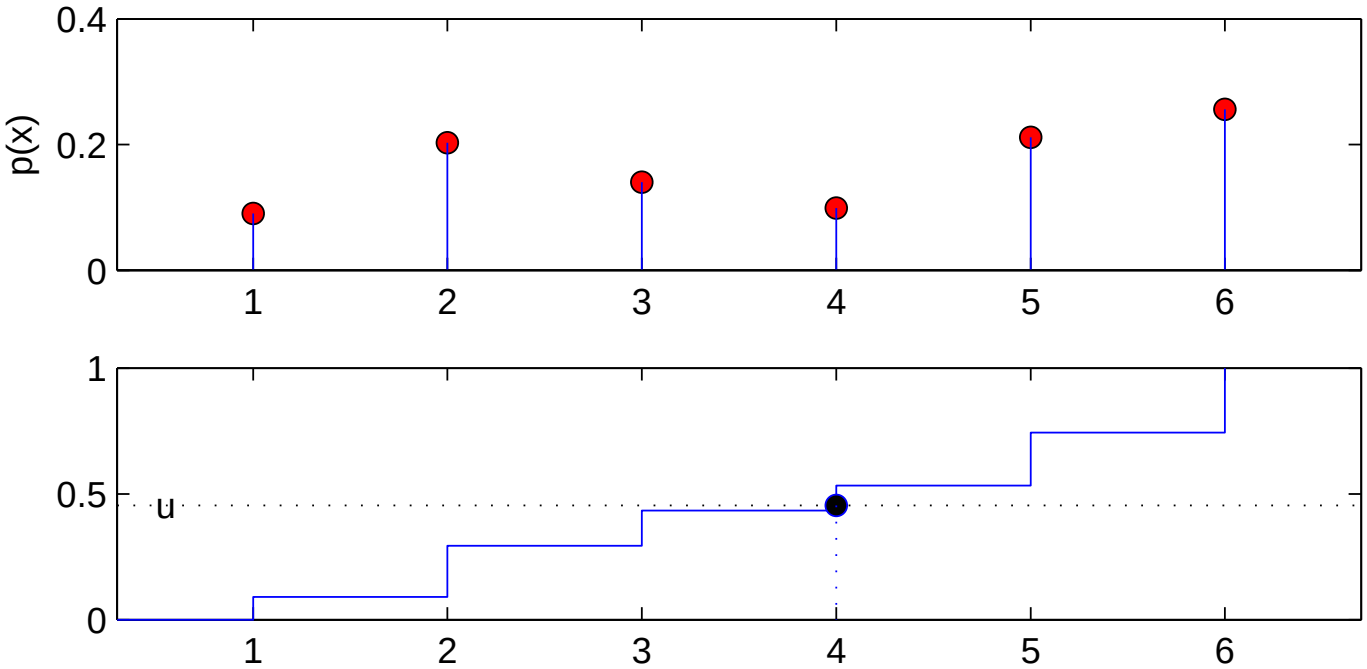
Alternative view



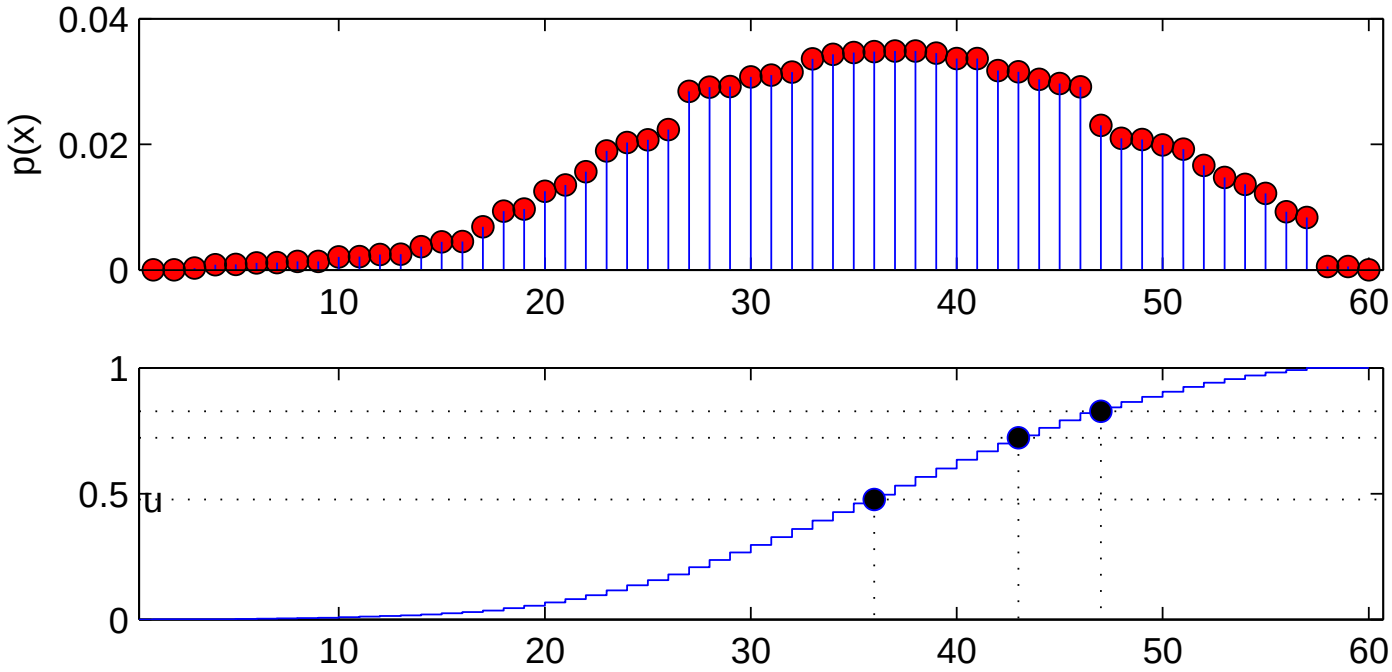
Generating random numbers from given distributions

- Inversion
- Transformation
- Rejection
- Reweighting – Importance sampling

Inversion



Inversion (cont.)



Generalised Inverse

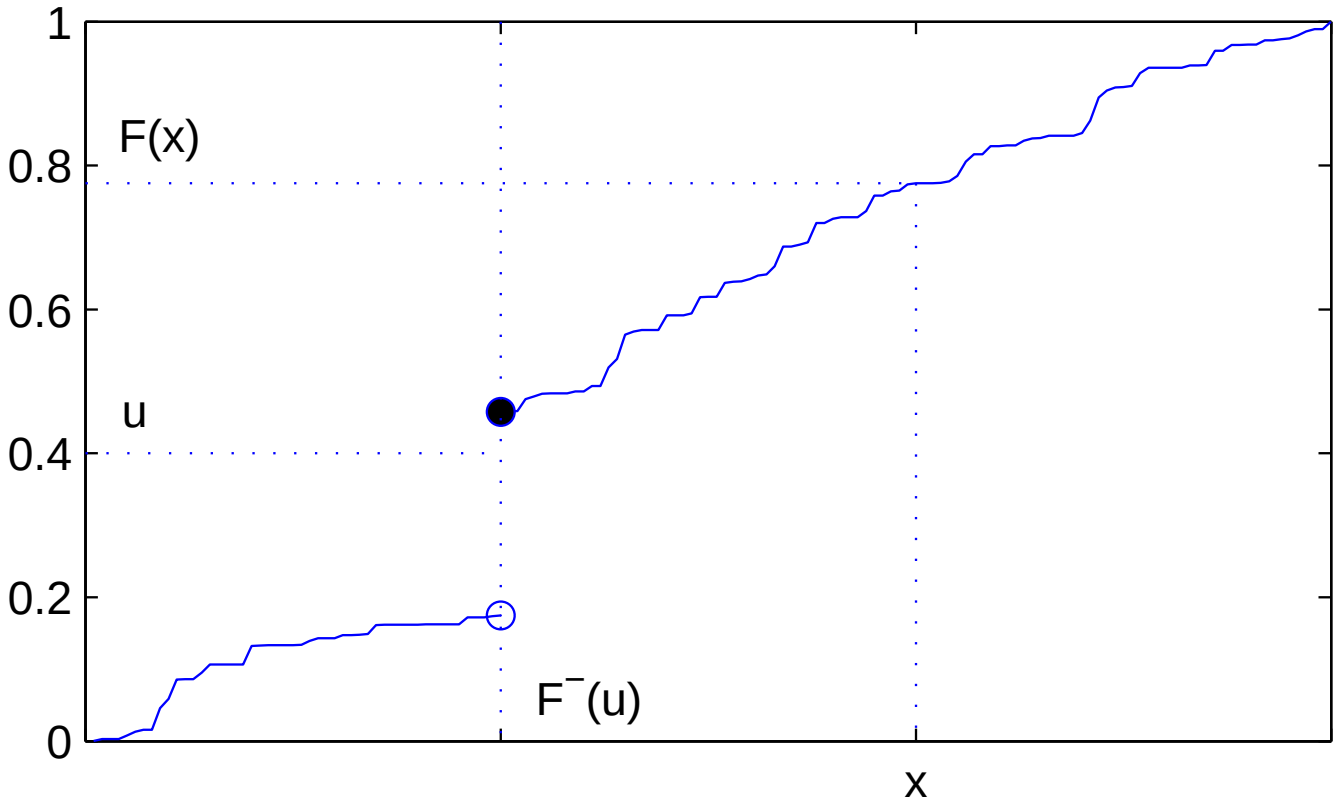
- Define the cumulative distribution function

$$\Pr\{X(\omega) \leq x\} \quad = \quad F(x)$$

- Generalised Inverse CDF

$$F^{-}(u) \quad \equiv \quad \inf\{x : F(x) \geq u\}$$

Generalised Inverse (cont.)



Example: Exponential Distribution

$$
\begin{aligned}
\mathcal{E}(x;\lambda) &= \lambda \exp(-\lambda x) \\
F(x) &= \int_0^x \lambda \exp(-\lambda \tau) = -\exp(-\lambda x) + 1 \\
u &= F(x) = 1 - \exp(-\lambda x) \\
x &= -\log(1-u)/\lambda = F^{-}(u)
\end{aligned}
$$

Or

$$
\begin{aligned}
u &\sim \mathcal{U}[0,1] \\
1-u &= u' \sim \mathcal{U}[0,1] \\
x &= -\log(u')/\lambda
\end{aligned}
$$

Transform Method

The generalised inverse of the CDF is just one possible transformation method

- Generate $\xi_i \sim p(\xi_i)$
- Set $x_j = h_j(\xi_1, \ldots, \xi_N)$
- Example: Box-Müller Method for generating Gaussian RV's
  - Generate a polar coordinate $(\sqrt{a}, \theta)$

$$\begin{aligned} \theta &\sim \mathcal{U}(\theta; 0, 2\pi) \\ a &\sim \mathcal{E}(a; 1/2) \end{aligned}$$

  - Transform into cartesian coordinates

$$\begin{aligned} x_1 &= \sqrt{a}\cos(\theta) && (\sim \mathcal{N}(0,1)) \\ x_2 &= \sqrt{a}\sin(\theta) && (\sim \mathcal{N}(0,1)) \end{aligned}$$

Change of Variables

Example:

- We are given X with density $f_X(x)$

$$Y = aX + b$$

- We find the CDF of Y analytically as follows

$$\Pr\{Y \leq y\} = \Pr\{aX + b \leq y\} = \begin{cases} \Pr\{X \leq (y-b)/a\}, & a > 0 \\ \Pr\{X \geq (y-b)/a\}, & a < 0 \end{cases}$$

- The density $f_Y(y)$ is the derivative w.r.t. y

$$f_Y(y) = \frac{1}{|a|} f_X((y-b)/a)$$

Change of variables

More general case (see Gri. & Sti. pp108):

- Define a one-to-one mapping

$$g : (x_1, x_2) \quad \rightarrow (y_1, y_2)$$

- Invert the mapping, i.e. find

$$\begin{aligned} x_1 &= x_1(y_1, y_2) \\ x_2 &= x_2(y_1, y_2) \end{aligned}$$

- Define the *Jacobian* determinant

$$J = \begin{vmatrix} \partial x_1/\partial y_1 & \partial x_2/\partial y_1 \\ \partial x_1/\partial y_2 & \partial x_2/\partial y_2 \end{vmatrix} = J(y_1, y_2)$$

Change of variables (cont.)

- The density of the transformed variable is

$$f_Y(y_1, y_2) = f_X(x_1(y_1, y_2), x_2(y_1, y_2))|J(y_1, y_2)|$$

- Keep in mind

$$f_Y(y)dy = f_X(x(y))\left|\frac{dx}{dy}\right|dy$$

Example: Box-Muller

- Generate

$$\begin{aligned} x_1 &\sim \mathcal{E}(x_1; 1/2) = \frac{1}{2}\exp(-x_1/2) \\ x_2 &\sim \mathcal{U}(x_2; 0, 2\pi) = \frac{1}{2\pi}\mathbb{I}\{0 \leq x_2 \leq 2\pi\} \end{aligned}$$

- Transform

$$\begin{aligned} y_1 &= \sqrt{x_1}\cos(x_2) \\ y_2 &= \sqrt{x_1}\sin(x_2) \end{aligned}$$

- Find the inverse mapping (the difficult bit)

$$\begin{aligned} y_1^2 + y_2^2 &= x_1 \\ \arctan(y_2/y_1) &= x_2 \end{aligned}$$

Example: Box-Muller (cont.)

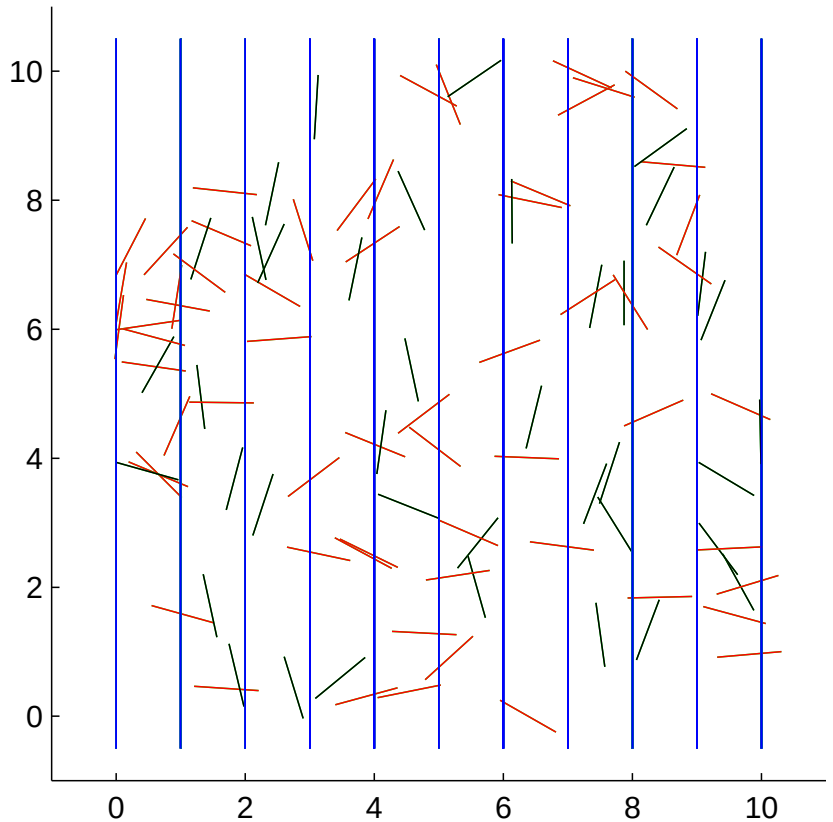
- Find the Jacobian determinant

$$J = \begin{vmatrix} \partial x_1/\partial y_1 & \partial x_2/\partial y_1 \\ \partial x_1/\partial y_2 & \partial x_2/\partial y_2 \end{vmatrix} = \begin{vmatrix} 2y_1 & \frac{1}{1+(y_2/y_1)^2}\frac{-y_2}{y_1^2} \\ 2y_2 & \frac{1}{1+(y_2/y_1)^2}\frac{1}{y_1} \end{vmatrix} = 2$$

- The density is

$$\begin{aligned} f_Y(y_1,y_2) &= \mathcal{E}(x_1(y_1,y_2);1/2)\mathcal{U}(x_2(y_1,y_2);0,2\pi)|J(y_1,y_2)| \\ &= \frac{1}{2}\exp(-(y_1^2+y_2^2)/2)\frac{1}{2\pi}2 \\ &= \frac{1}{\sqrt{2\pi}}\exp(-y_1^2/2)\frac{1}{\sqrt{2\pi}}\exp(-y_2^2/2) \\ &= \mathcal{N}(y_1;0,1)\mathcal{N}(y_2;0,1) \end{aligned}$$

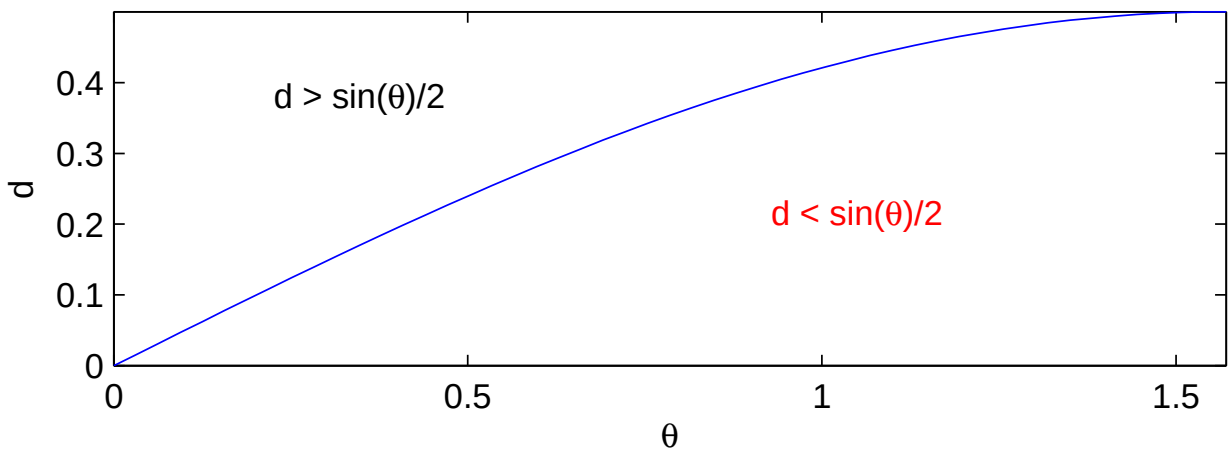
Rejection Sampling : Recall Buffon's needle



Buffon's needle

- d : Distance from the middle of the needle to the nearest line
- θ : Acute angle between the parallel lines and the needle
- A needle touches a line iff

$$\frac{d}{\sin\theta} < \frac{1}{2}$$



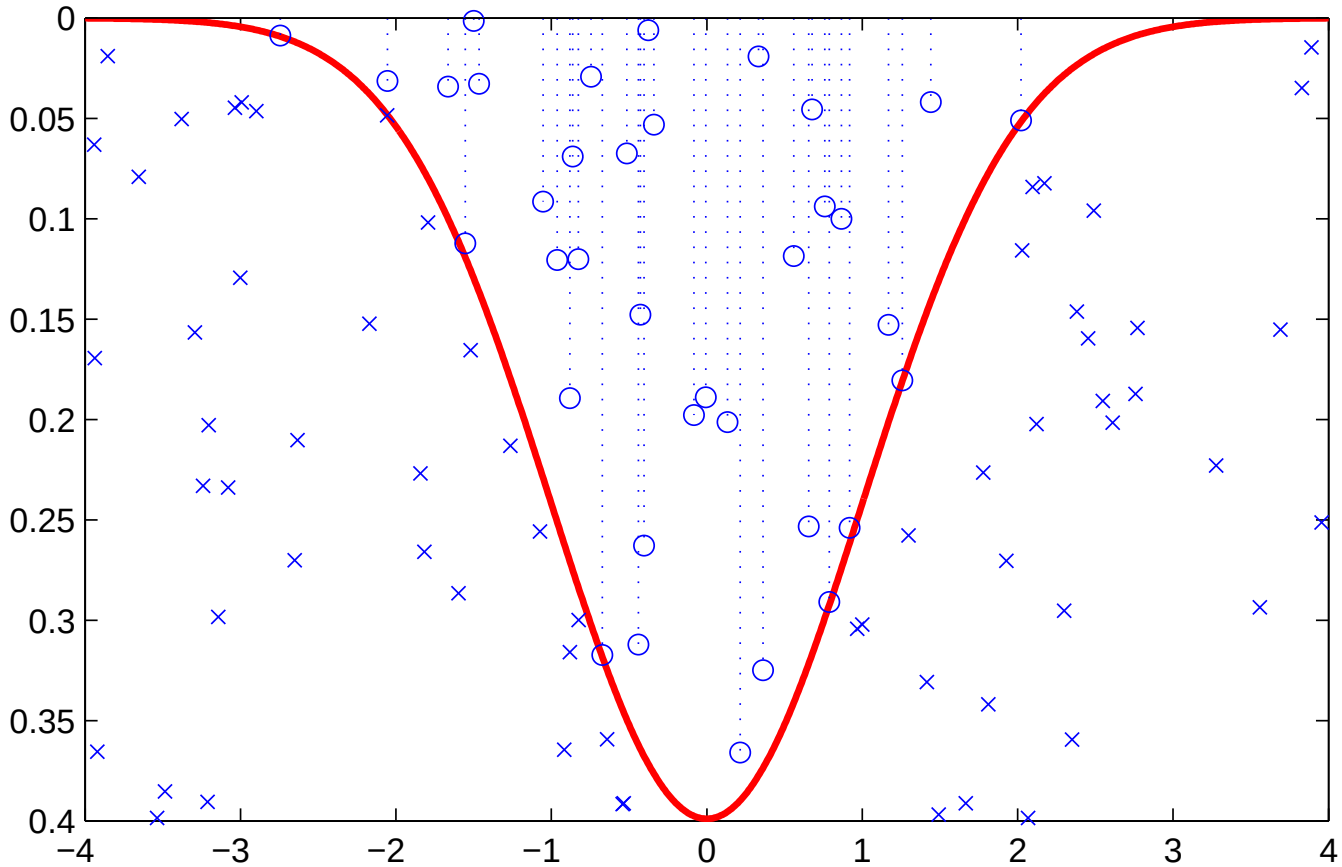
Rejection Sampling

- The solution to the Buffon needle problem is an instance of **rejection sampling**
- Basic idea

$$p(x) \quad = \quad \int_0^{p(x)} 1 d\tau = \int \mathbb{I}\{0 \leq \tau \leq p(x)\} d\tau \equiv \int p(x,\tau) d\tau$$

$\rightsquigarrow p(x)$ is a *marginal density* of the uniform distribution on the area under the curve $p(x)$. (Think of fishes in a lake)

Fishes in a lake



Rejection Sampling. Version 1.

- Construct an easy to sample density $q(x)$ such that $p(x) < Mq(x)$
- Draw $x^{(i)} \sim q(x)$, $i = 1 \ldots N$
- Generate the τ component

$$\begin{aligned} u &\sim \mathcal{U}[0,1] \\ \tau^{(i)} &= Mq(x^{(i)})u \end{aligned}$$

  The pair (x, τ) are uniformly distributed under the curve $Mq(x)$
- If $\tau^{(i)} < p(x^{(i)})$, accept: discard $\tau^{(i)}$, keep $x^{(i)}$, else reject: discard $(x^{(i)}, \tau^{(i)})$

Note that we don't need to explicitly generate $\tau^{(i)}$

Acceptance Probability

$$\begin{aligned} \tau^{(i)} &< p(x^{(i)}) \\ Mq(x^{(i)})u &< p(x^{(i)}) \\ u &< \frac{p(x^{(i)})}{Mq(x^{(i)})} \equiv \alpha(x^{(i)}) \end{aligned}$$

α is the acceptance probability.

Rejection Sampling. Final Version.

1. Construct an easy to sample density $q(x)$ such that $p(x) < Mq(x)$
2. Draw $x^{(n)} \sim q(x)$
3. Accept $x^{(n)}$ with probability

$$\alpha(x^{(n)}) = \frac{p(x^{(n)})}{Mq(x^{(n)})}$$

   - $p(x)$ can not have heavier tails than $q(x)$

Use in Bayesian Computation

- Suppose we don't know the normalisation constant Z

$$p(x|y) = \frac{1}{p(y)} p(x,y) \equiv \frac{1}{Z}\phi(x)$$

$$\begin{aligned} p(x|y) &< Mq(x) \\ \phi(x) &< ZMq(x) \equiv \bar{M}q(x) \end{aligned}$$

- We can still use the acceptance probability

$$\alpha(x^{(n)}) = \frac{\phi(x^{(n)})}{\bar{M}q(x^{(n)})}$$

- It may be very difficult to find $\bar{M}$ in higher dimensions.

Adaptive Rejection Sampling

- Idea: Adapt the rejection envelope adaptively to reduce the rejection ratio
- Suitable for log-concave ($\frown$) densities, which are analytically complex
- Gilks and Wild, 1992, Appl. Statist. Vol 41, No 2, pp. 337-348

Importance Sampling (IS)

- Consider a probability distribution $p(x)$, that is hard to sample.
- IS idea: Estimate expectations (or features) of $p(x)$ by a properly weighted sample, using a "simpler" proposal distribution q.

Importance Sampling (cont.)

- Change of measure with **weight function** $W(x) \equiv p(x)/q(x)$

$$\langle f(x) \rangle_{p(x)} = \int dx f(x) \frac{p(x)}{q(x)} q(x) = \left\langle f(x) \frac{p(x)}{q(x)} \right\rangle_{q(x)}$$

- The Expectation

$$\langle f(x) \rangle_{p(x)} \equiv \langle f(x) W(x) \rangle_{q(x)}$$

- Monte Carlo estimate

$$\tilde{\mu}_N = \frac{1}{N} \sum_{i=1}^{N} f(x^{(i)}) W(x^{(i)})$$

Unknown normalising constant

Consider a probability distribution with (possibly unknown) normalisation constant

$$p(x) = \frac{1}{Z}\phi(x) \qquad\qquad Z = \int dx \phi(x).$$

- Change of measure with **weight function** $W(x) \equiv \phi(x)/q(x)$

$$\langle f(x) \rangle_{p(x)} = \frac{1}{Z} \int dx f(x) \frac{\phi(x)}{q(x)} q(x) = \frac{1}{Z} \left\langle f(x) \frac{\phi(x)}{q(x)} \right\rangle_{q(x)}$$

Unknown normalising constant (cont.)

- Rewrite Z :

$$Z = \int dx \phi(x) = \int dx \frac{\phi(x)}{q(x)} q(x) = \langle W(x) \rangle_{q(x)}$$

- Ratio of two expectations

$$\langle f(x) \rangle_{p(x)} = \frac{\langle f(x) W(x) \rangle_{q(x)}}{\langle W(x) \rangle_{q(x)}}$$

- Monte Carlo estimate

$$\hat{\mu}_N = \frac{\sum_{i=1}^{N} W^{(i)} f(x^{(i)})/N}{\sum_{i'=1}^{N} W^{(i')}/N}$$

Normalised weights w

$$
\begin{aligned}
\hat{\mu}_N &= \frac{\sum_{i=1}^{N} W^{(i)} f(x^{(i)})}{\sum_{i'=1}^{N} W^{(i')}} \\
&= \sum_{i=1}^{N} \left(W^{(i)} / \sum_{i'=1}^{N} W^{(i')} \right) f(x^{(i)}) \\
&\equiv \sum_{i=1}^{N} \tilde{w}^{(i)} f(x^{(i)})
\end{aligned}
$$

$$
\tilde{w}^{(i)} \equiv W^{(i)} / \sum_{i'=1}^{N} W^{(i')}
$$

Importance Sampling

- Draw $i = 1, \ldots N$ independent samples from q

$$x^{(i)} \sim q(x)$$

- Calculate the **importance weights**

$$W^{(i)} \quad = \quad W(x^{(i)}) = \phi(x^{(i)})/q(x^{(i)})$$

- If $Z = 1$, i.e., $p(x) = \phi(x)$, construct the estimate $\tilde{\mu}$

$$\langle f(x) \rangle_{p(x)} \quad \approx \quad \tilde{\mu} = \frac{1}{N} \sum_{i=1}^{N} W(x^{(i)}) f(x^{(i)})$$

Importance Sampling (cont.)

- If Z is unknown, approximate the normalizing constant

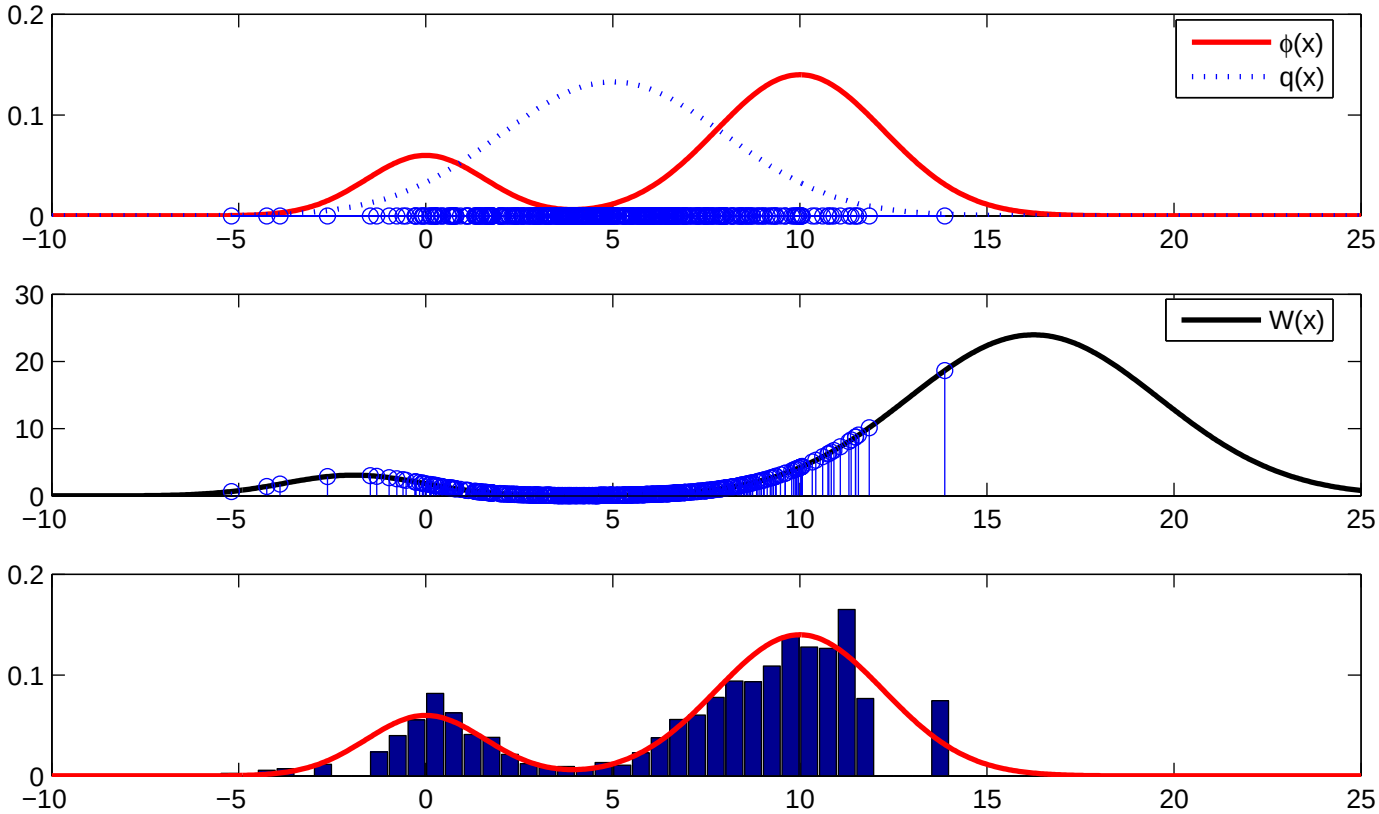
$$Z = \langle W(x) \rangle_{q(x)} \approx \frac{1}{N} \sum_{i=1}^{N} W^{(i)}$$

- Construct the estimate $\hat{\mu}$

$$\begin{aligned} \langle f(x) \rangle_{p(x)} &= \frac{\langle f(x) W(x) \rangle_{q(x)}}{\langle W(x) \rangle_{q(x)}} \approx \hat{\mu} \\ &= \frac{\sum_{i=1}^{N} W^{(i)} f(x^{(i)})/N}{\sum_{i=1}^{N} W^{(i)}/N} \equiv \sum_{i=1}^{N} \tilde{w}^{(i)} f(x^{(i)}) \equiv \hat{E}_N \end{aligned}$$

Here $\tilde{w}^{(i)} = W^{(i)} / \sum_{j=1}^{N} W^{(j)}$ are *normalized importance weights*.

Importance Sampling (cont.)



The Quality of the estimator

- Characterised by the (asymptotic) variance

$$
\begin{aligned}
Var\{f(x)W(x)\} &= \left\langle \left(f(x)W(x) - \langle f(x)W(x)\rangle\right)^2 \right\rangle_{q(x)} \\
&= \left\langle f^2(x)W^2(x)\right\rangle_{q(x)} - \langle f(x)W(x)\rangle^2_{q(x)} \\
&= \left\langle f^2(x)W^2(x)\right\rangle_{q(x)} - \langle f(x)\rangle^2_{p(x)} \\
&= \left\langle f^2(x)W^2(x)\right\rangle_{q(x)} - \mu_f^2
\end{aligned}
$$

- Alternatively, via Effective Sample Size (ESS)

Effective Sample Size (ESS)

For a weight function $W(x) = p(x)/q(x)$ we compute

$$ESS(N) = \frac{N}{1 + Var\{W\}_q}$$

- When $p(x) = q(x)$, the variance is zero, so $ESS(N) = N$.
- With increasing variance, the ESS decreases.

In practice, we estimate ESS via the coefficient of variation given samples $x^{(i)}$, $i = 1 \ldots N$ with weights $W^{(i)} = W(x^{(i)})$.

$$CV^2(W^{(1:N)}) = \frac{\sum_{i=1}^{N}(W^{(i)} - \bar{W})^2}{(N-1)\bar{W}^2}$$

$$\bar{W} = \frac{1}{N}\sum_{i=1}^{N} W^{(i)}$$

Example

- We will compute the expected value of the Beta distribution via importance sampling

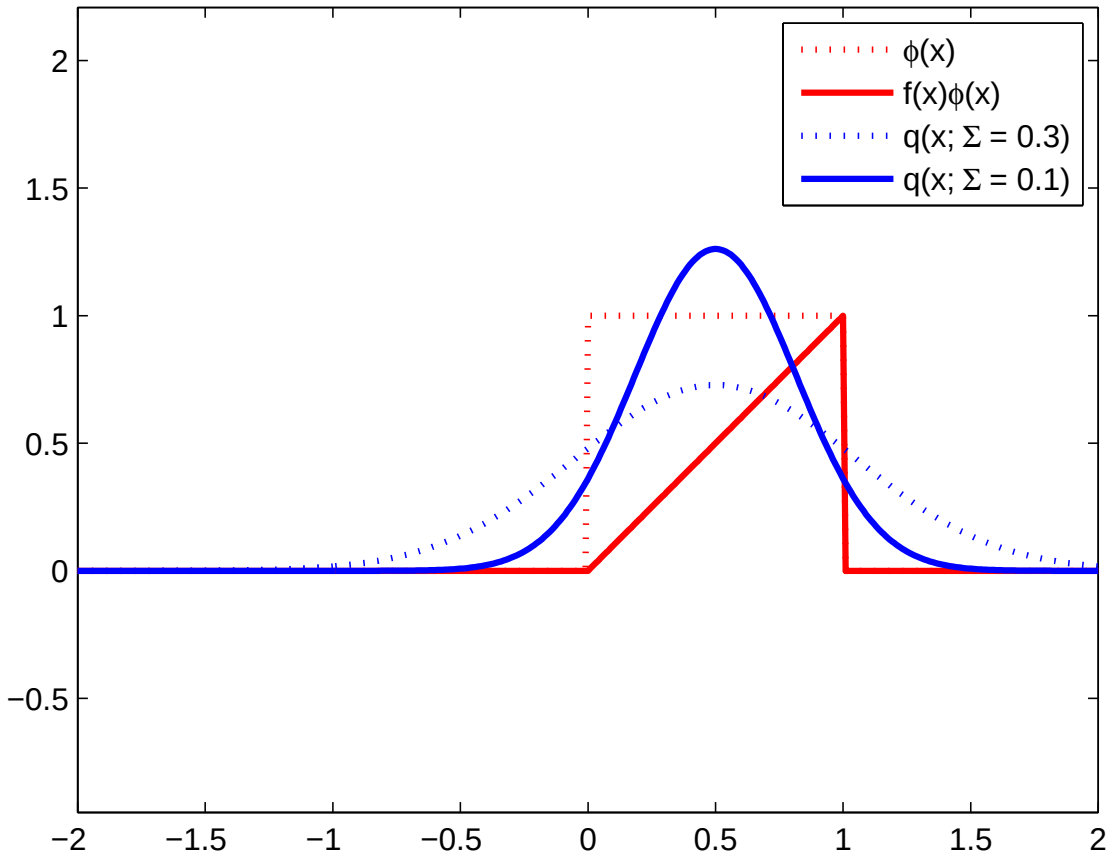
$$
\begin{aligned}
E &= \langle f(x) \rangle_{p(x)} \\
f(x) &\equiv x \\
p(x) &\equiv \mathcal{B}(x; a, b) = \phi(x)/Z \\
\phi(x) &\equiv \exp\left((a-1)\log x + (b-1)\log(1-x)\right)[0 \leq x \leq 1] \\
Z &\equiv \exp\left(-\log\Gamma(a+b) + \log\Gamma(a) + \log\Gamma(b)\right)
\end{aligned}
$$

- We will suppose that we don't know Z
- We will use a Gaussian as a proposal

$$
q(x) = \mathcal{N}(x; \mu, \Sigma)
$$

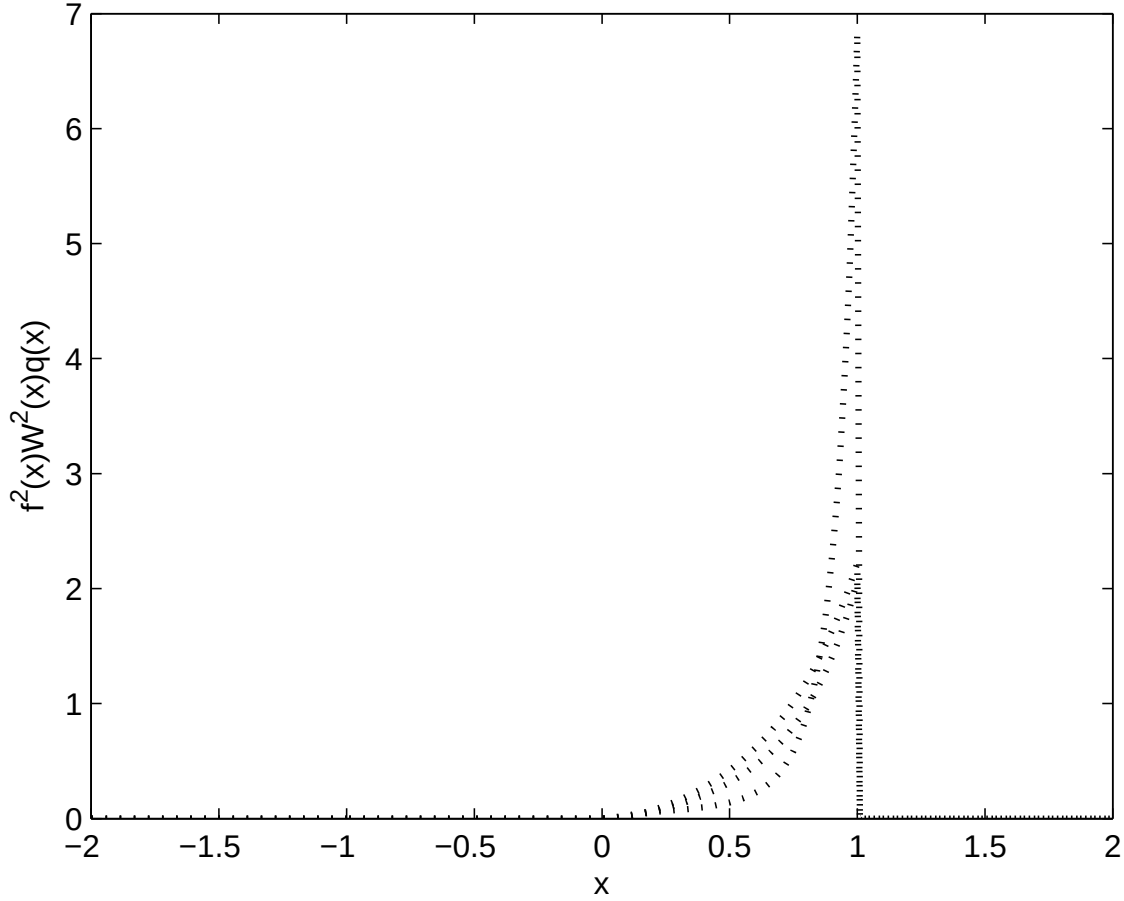
Example

- $\phi(x) \propto \mathcal{B}(x; 1, 1)$



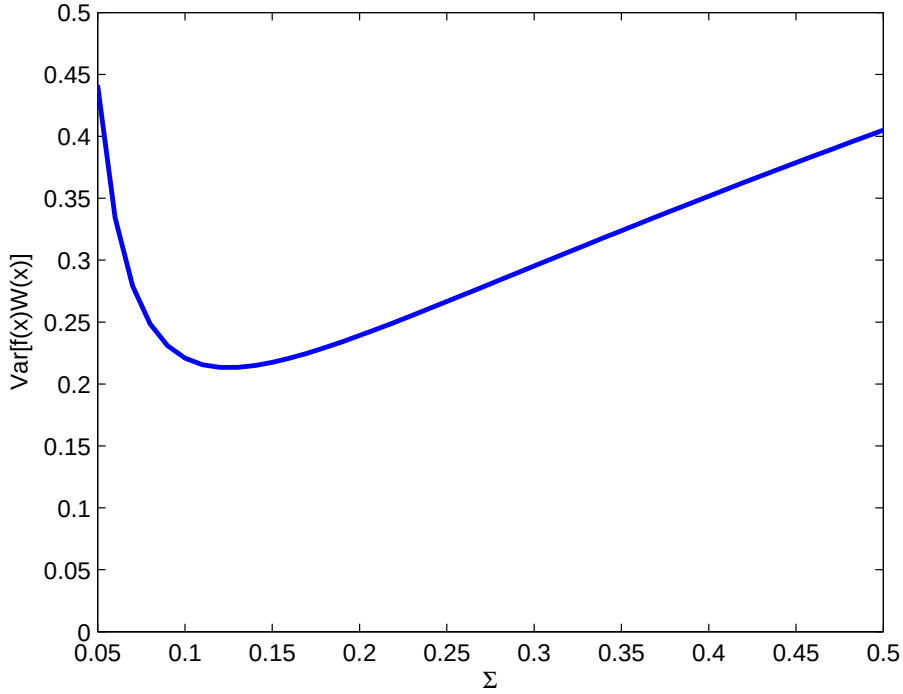
Example

- The variance is equal to the area under the curve $f(x)^2 W(x)^2 q(x) = f(x)^2 \phi(x)^2 / q(x)$

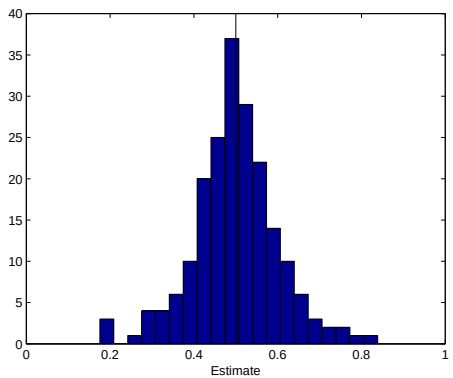


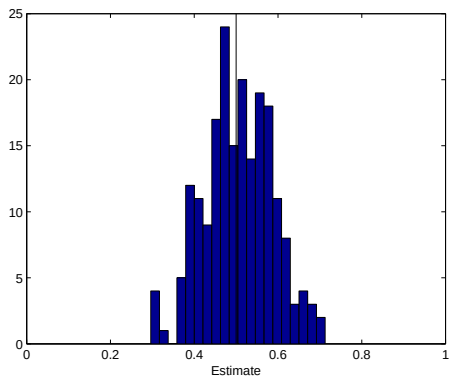
Example

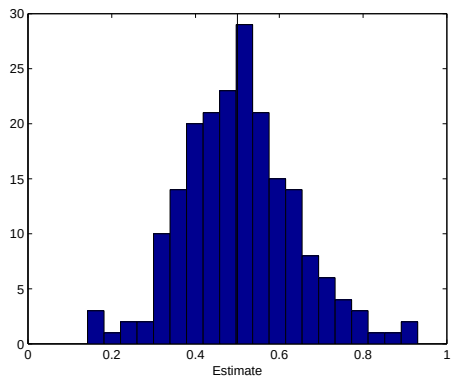
- The asymptotic variance as a function of the proposal density
- The variance is minimised when the proposal is 'close' to the target



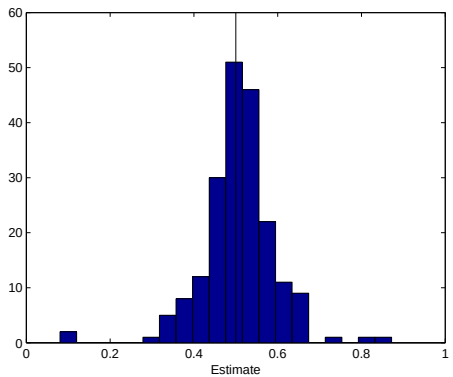
- Histogram of independent estimates with $N = 20$. $\Sigma = \{0.01, 0.1, 2\}$
- The experiment is repeated 200 times and the histogram is shown

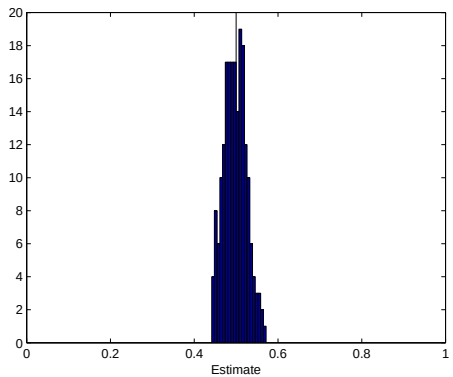


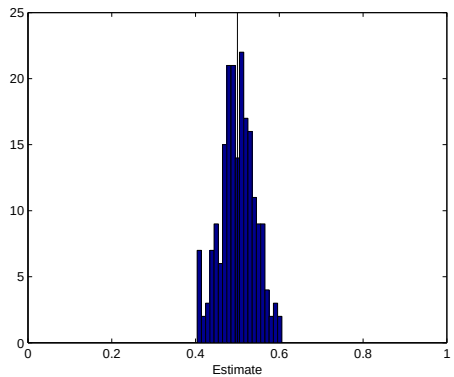




- Histogram of independent estimates with $N = 200$. $\Sigma = \{0.01, 0.1, 2\}$
- As expected, the variance scales with $N^{-1/2}$.







Example 2

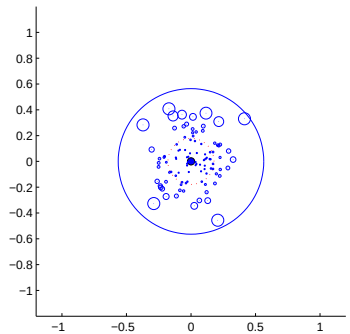
- We will compute the area of a circle centered at 0 with radius $\rho = 1/\sqrt{\pi}$ using importance sampling
- The proposal is an (isotropic) Gaussian

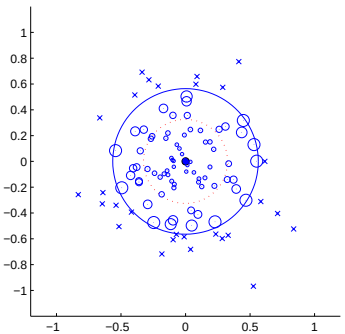
$$q(x) \quad = \quad \mathcal{N}(x; 0, \gamma\rho^2 I)$$

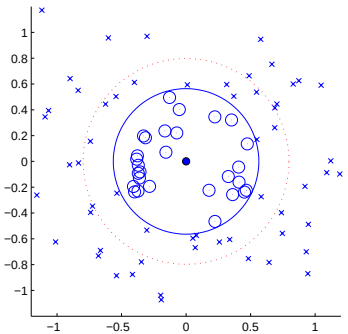
  here $\gamma > 0$ is a scalar that adjusts the variance
- We will investigate
  - The estimation quality
  - Variance of the weights - Effective sample size (ESS)
  - The effect of the proposal

Typical Draws

- Number of samples: $N = 100$
- $\gamma = \{1/10, 1/3, 2\}$, 1σ contour shown with red dotted ellipse

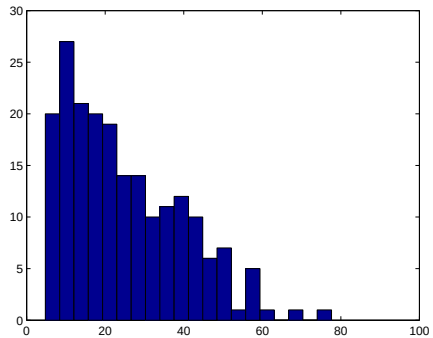


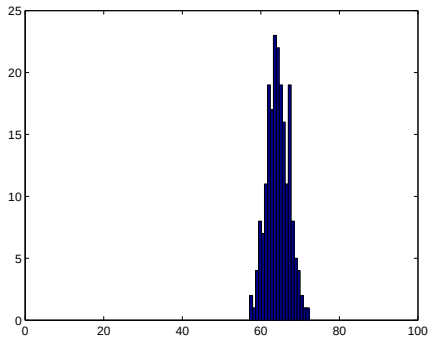


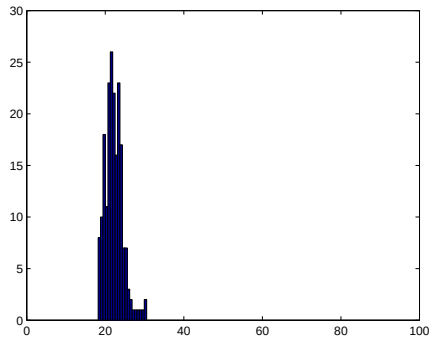


Histogram of ESS

- Number of samples: $N = 100$
- $\gamma = \{1/10, 1/3, 2\}$, 200 independent trials

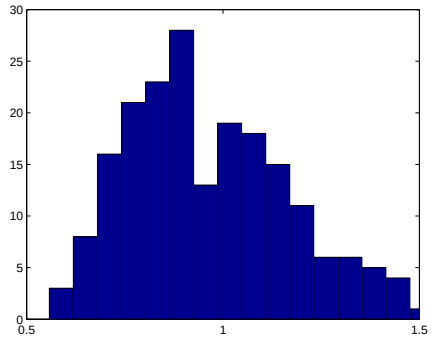


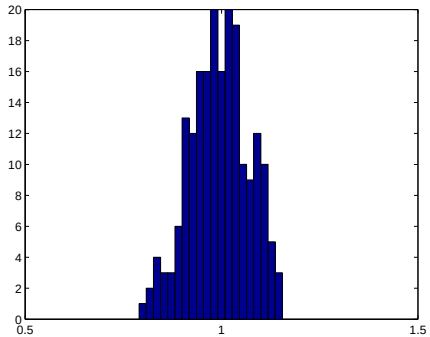


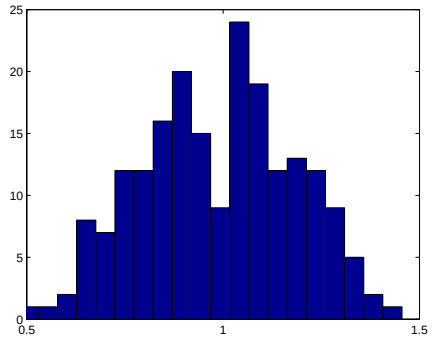


Histogram of Estimates

- Number of samples: $N = 100$
- $\gamma = \{1/10, 1/3, 2\}$, 200 independent trials

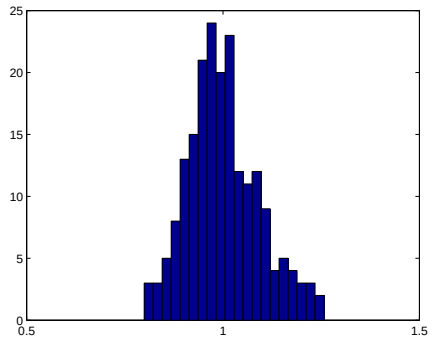


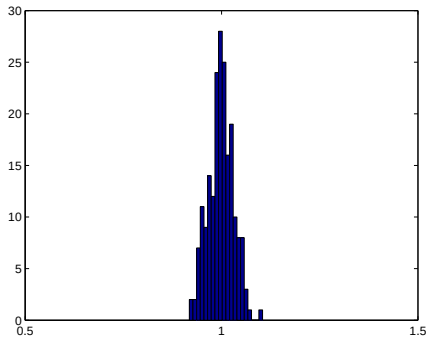


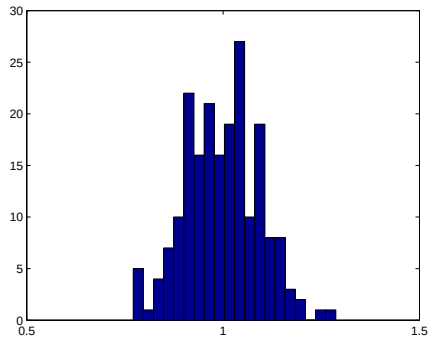


Histogram of Estimates

- Number of samples: $N = 500$
- $\gamma = \{1/10, 1/3, 2\}$, 200 independent trials







Variance reduction

There are many proposals,

- how do we choose a proposal?
- is there a “best” proposal distribution?

Optimal Proposal Distribution

Task: Estimate $\langle f(x) \rangle_{p(x)}$

- IS constructs the estimator $\mu_f = \langle f(x)W(x) \rangle_{q(x)}$
- Minimize the variance of the estimator

$$\begin{aligned} Var\{f(x)W(x)\} &= \left\langle \left(f(x)W(x) - \langle f(x)W(x) \rangle\right)^2 \right\rangle_{q(x)} \\ &= \left\langle f^2(x)W^2(x) \right\rangle_{q(x)} - \langle f(x)W(x) \rangle^2_{q(x)} \\ &= \left\langle f^2(x)W^2(x) \right\rangle_{q(x)} - \langle f(x) \rangle^2_{p(x)} \\ &= \left\langle f^2(x)W^2(x) \right\rangle_{q(x)} - \mu_f^2 \end{aligned}$$

- Minimize the first term: only it depends on q

Optimal Proposal Distribution

- (By Jensen's inequality) The first term is lower bounded:

$$\left\langle f^2(x)W^2(x)\right\rangle_{q(x)} \geq \left\langle |f(x)|W(x)\right\rangle^2_{q(x)} = \left(\int |f(x)|\, p(x)dx\right)^2$$

- Since this is a lower bound, we can't do better that this
- We will look for a distribution q^* that attains this lower bound. Take

$$q^*(x) = \frac{|f(x)|p(x)}{\int |f(x')|p(x')dx'}$$

Optimal Proposal Distribution (cont.)

- The weight function for this particular proposal q^* is

$$W_*(x) = p(x)/q^*(x) = \frac{\int |f(x')|p(x')dx'}{|f(x)|}$$

- We show that q^* attains its lower bound

$$\begin{aligned}
\left\langle f^2(x)W_*^2(x)\right\rangle_{q^*(x)} &= \left\langle f^2(x)\frac{\left(\int |f(x')|p(x')dx'\right)^2}{|f(x)|^2}\right\rangle_{q^*(x)} \\
&= \left(\int |f(x')|p(x')dx'\right)^2 = \langle |f(x)|\rangle^2_{p(x)} \\
&= \langle |f(x)|W_*(x)\rangle^2_{q^*(x)}
\end{aligned}$$

- $\Rightarrow$ There are distributions q^* that are even "better" than the distribution itself!

Reducing Variance of importance weights: A link to alpha divergences

The α-divergence between two distributions is defined as

$$D_\alpha(p||q) \equiv \frac{1}{\beta(1-\beta)}\left(1-\int dx p(x)^\beta q(x)^{1-\beta}\right)$$

where $\beta = (1+\alpha)/2$ and p and q are two probability distributions

- $\lim_{\beta\to 0} D_\alpha(p||q) = KL(q||p)$
- $\lim_{\beta\to 1} D_\alpha(p||q) = KL(p||q)$
- $\beta = 2$, $(\alpha = 3)$

$$D_3(p||q) \equiv \frac{1}{2}\int dx p(x)^2 q(x)^{-1} - \frac{1}{2} = \frac{1}{2}\left\langle W(x)^2\right\rangle_{q(x)} - \frac{1}{2}$$

Best q (in a constrained family) is typically a heavy-tailed approximation to p

Summary

- Random Number Generation
  - Pseudo-random numbers
- Inversion
- Transformation
- Rejection Sampling
- Importance Sampling

CMPE 58N - Lecture 3
Monte Carlo methods

Markov Chains, MCMC, Metropolis-Hastings Algorithm

Department of Computer Engineering,

Boğaziçi University, Istanbul, Turkey

Instructor: A. Taylan Cemgil

Fall 2009

Outline

- Motivations
- Markov Chains
- Metropolis Hastings algorithm

Motivating example: Sampling uniformly from a set S

- Suppose we have a black box implementation of an indicator function $[x \in S]$
- How can we generate uniform samples from S?
- It turns out that the following algorithm works (in principle)

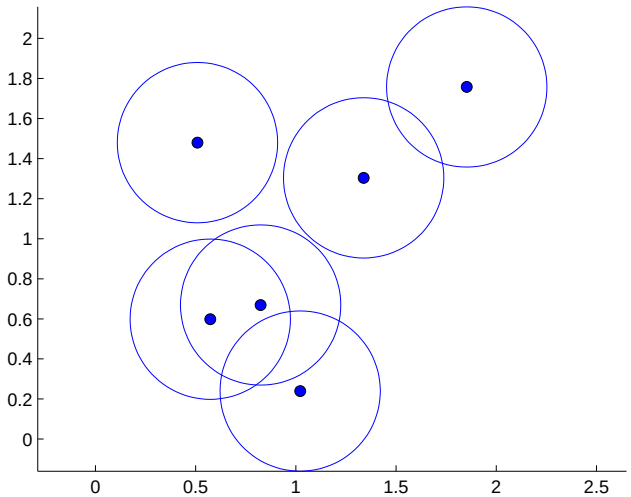
  Choose an arbitrary $x^{(0)} \in S$
  For $i = 1, 2, \ldots$

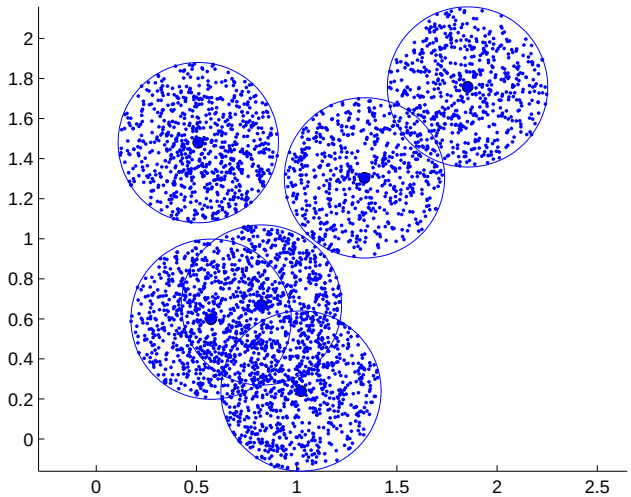
    Propose:
    $\epsilon_i \sim \mathcal{N}(0, V)$
    $x' \leftarrow x^{(i-1)} + \epsilon_i$
    Accept/Reject
    if $[x' \in S]$ then $x^{(i)} \leftarrow x'$ else $x^{(i)} \leftarrow x^{(i-1)}$ endif

  EndFor
  - $x^{(i)}$ are the desired samples! Why?

Motivating example: Sampling uniformly from a set S

$$S = \{x : \|c_i - x\| \leq \rho\}$$



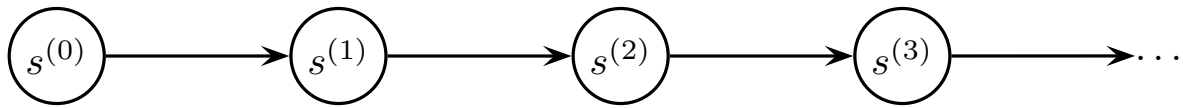


Markov Chains

- Markov Property

$$p(s^{(t)}|s^{(t-1)}, s^{(t-2)}, \ldots, s^{(0)}) = p(s^{(t)}|s^{(t-1)})$$

- Future and past are conditionally independent given current state

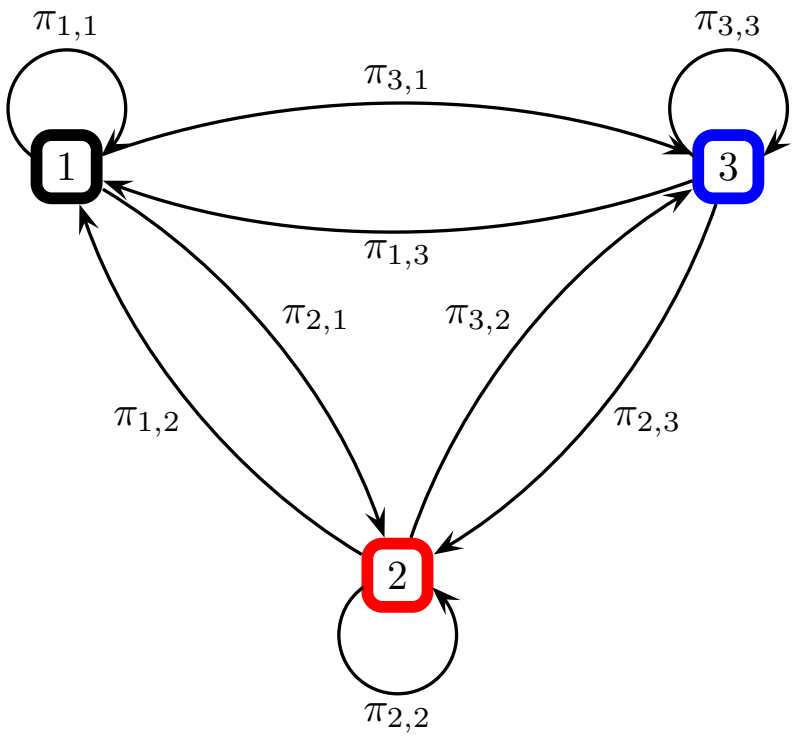


$$p(s^{(0)}, s^{(1)}, s^{(2)}, \ldots) = p(s^{(0)}) \prod_{t=1}^{\infty} p(s^{(t)}|s^{(t-1)})$$

Markov Chains

- Here, we look at Markov chains for analysis of inference algorithms, not for constructing a time series model
- Nature of the State space $\mathcal{X}$ matters ($s^{(t)} \in \mathcal{X}$)
  - $\mathcal{X}$ is Finite ($\leftarrow$ simplest, we look at this)
  - $\mathcal{X}$ is Countable
  - $\mathcal{X}$ is Uncountable ($\leftarrow$ actually needed but quite technical)

State Transition Diagram

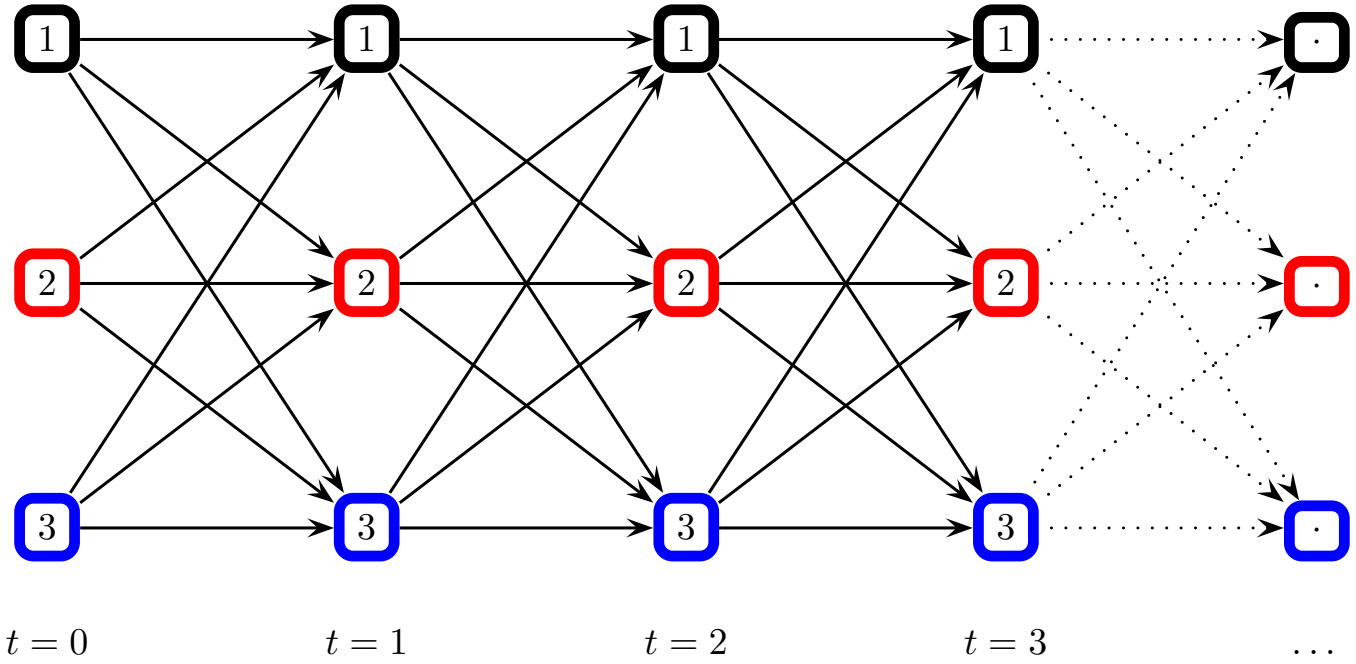


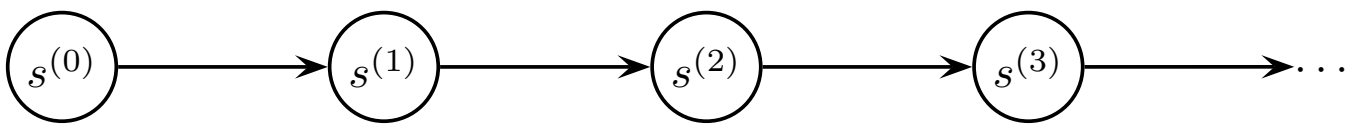
State Transition Diagram (cont.)

$$p(s^{(t)} = i | s^{(t-1)} = j) = \pi_{i,j}$$

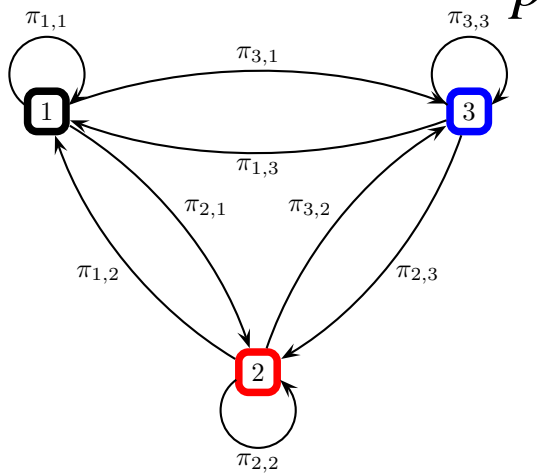
Caution: not a Graphical Model, a (nondeterministic) finite state machine

State Transition Diagram, alternative





Matrix Representation of the State Transition Diagram



$$p\left(s^{(t)}=i\middle|s^{(t-1)}=j\right)=\pi_{i,j}$$

$$p\left(s^{(t)}\middle|s^{(t-1)}\right)=\begin{pmatrix}\pi_{1,1} & \pi_{1,2} & \pi_{1,3}\\ \pi_{2,1} & \pi_{2,2} & \pi_{2,3}\\ \pi_{3,1} & \pi_{3,2} & \pi_{3,3}\end{pmatrix}\equiv\mathbf{T}$$

Computing Marginals

- Recursive

$$
\begin{aligned}
p(s^{(1)}) &= \sum_{s^{(0)}} p(s^{(1)}|s^{(0)})p(s^{(0)}) \qquad \left(\equiv p_i^{(1)} = \sum_j \pi_{i,j} p_j^{(0)} \right) \\
p(s^{(2)}) &= \sum_{s^{(1)}} p(s^{(2)}|s^{(1)})p(s^{(1)}) \\
&\vdots
\end{aligned}
$$

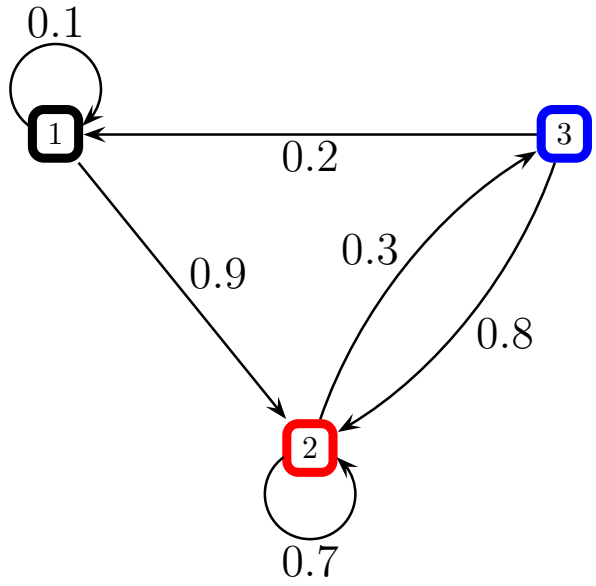
- Using matrix notation, by induction

$$
\begin{aligned}
p^{(1)} &= \mathbf{T}p^{(0)} \\
p^{(2)} &= \mathbf{T}p^{(1)} = \mathbf{T}^2 p^{(0)} \\
&\vdots \\
p^{(t)} &= \mathbf{T}^t p^{(0)}
\end{aligned}
$$

Chapman-Kolmogorov Equations

$$\mathbf{T}^{(n+m)} = \mathbf{T}^n \mathbf{T}^m$$

Numeric Example



$$\begin{pmatrix} 0.1 & 0 & 0.2 \\ 0.9 & 0.7 & 0.8 \\ 0 & 0.3 & 0 \end{pmatrix}$$

- Suppose the inital state is 1, we have

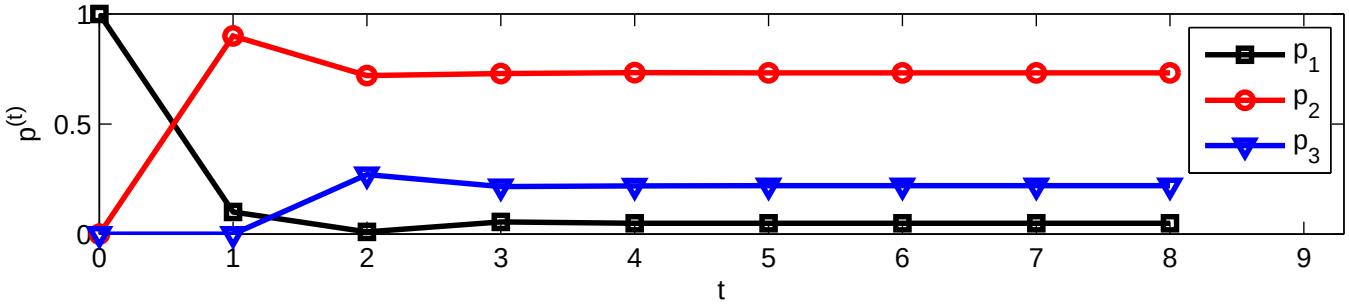
$$p^{(1)} = \mathbf{T}p^{(0)} = \begin{pmatrix} 0.1 & 0 & 0.2 \\ 0.9 & 0.7 & 0.8 \\ 0 & 0.3 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.1 \\ 0.9 \\ 0 \end{pmatrix}$$

Numeric Example

- Continue

$$p^{(2)} = \mathbf{T} \begin{pmatrix} 0.1 \\ 0.9 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.01 \\ 0.72 \\ 0.27 \end{pmatrix}$$

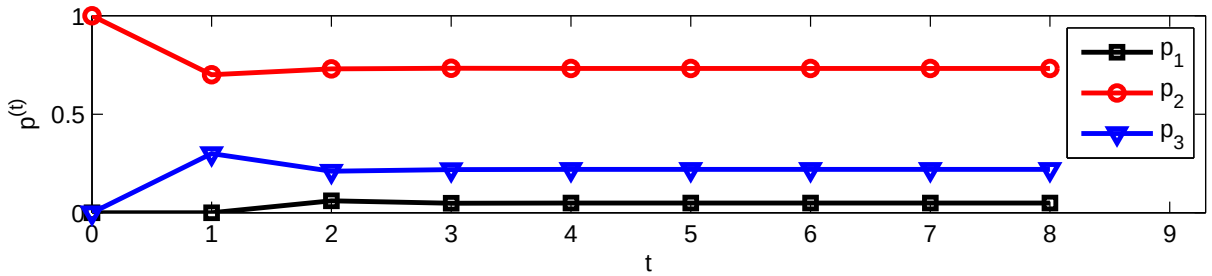
$$p^{(3)} = \mathbf{T} \begin{pmatrix} 0.01 \\ 0.72 \\ 0.27 \end{pmatrix} = \begin{pmatrix} 0.05 \\ 0.73 \\ 0.22 \end{pmatrix}$$



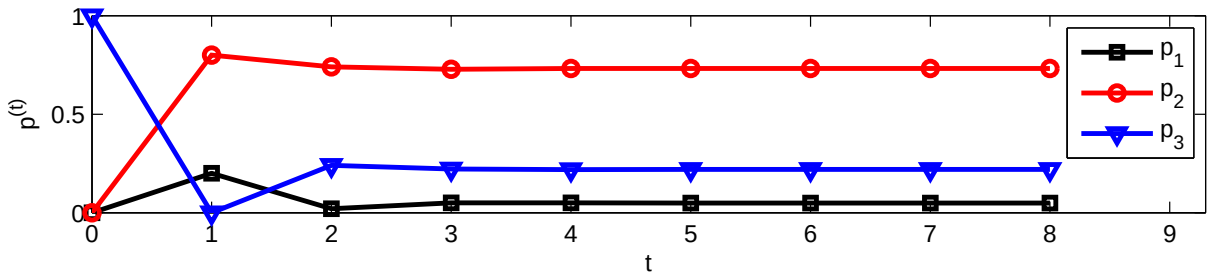
Convergence to a stationary distribution

Starting from other configurations does not alter the picture

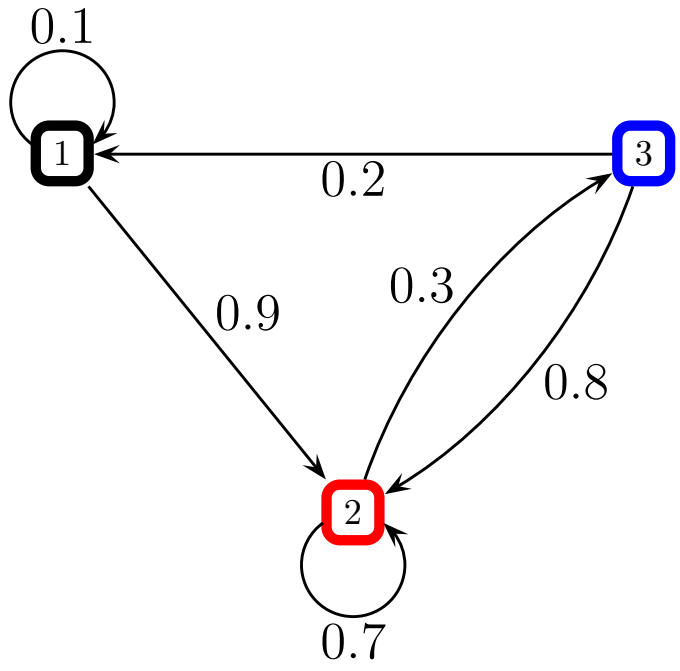
- $p^{(0)} = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^\top$



- $p^{(0)} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top$



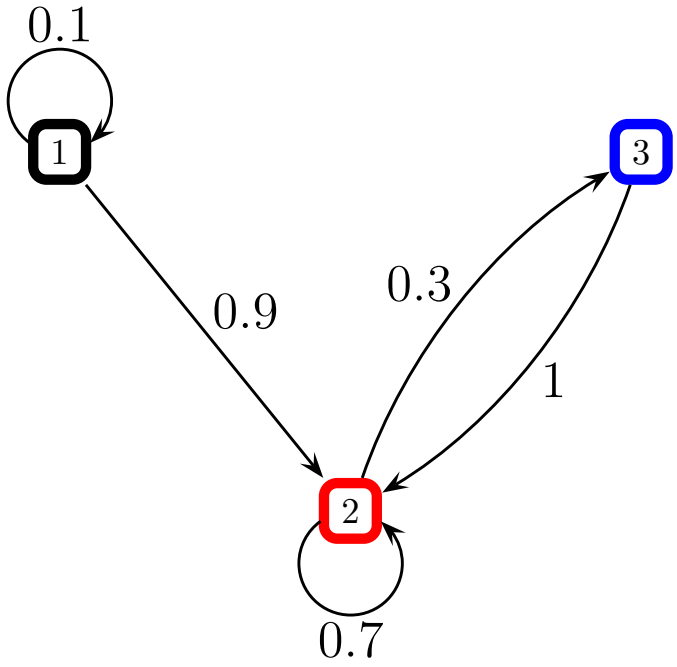
Examples: Irreducable chain



$$\begin{pmatrix} 0.1 & 0 & 0.2 \\ 0.9 & 0.7 & 0.8 \\ 0 & 0.3 & 0 \end{pmatrix}$$

- All states communicate $\Rightarrow$ Chain is said to be irreducable
- All states recurrent

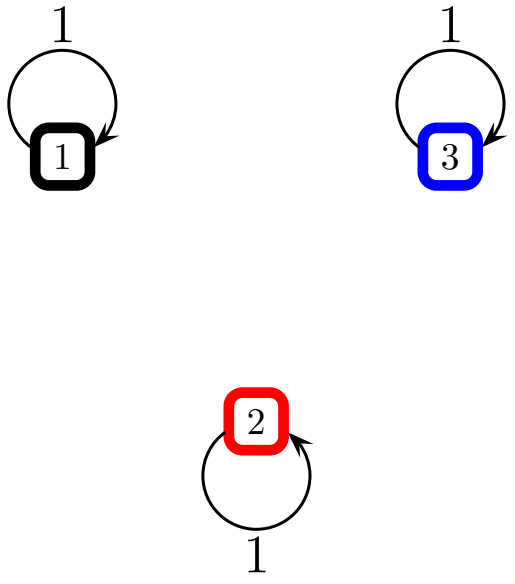
Examples: Transient states



$$\begin{pmatrix} 0.1 & 0 & 0 \\ 0.9 & 0.7 & 1 \\ 0 & 0.3 & 0 \end{pmatrix}$$

- When the chain leaves state 1, it never returns $\Rightarrow$ State 1 is transient

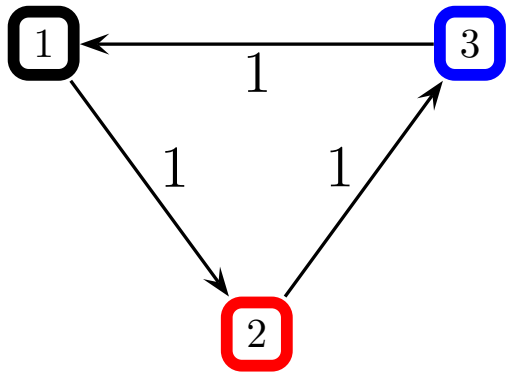
Examples: Reducable chains



$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Disconnected subgraphs in state transition diagram $\Rightarrow$ Chain is reducable
- No unique stationary distribution

Example: Periodic



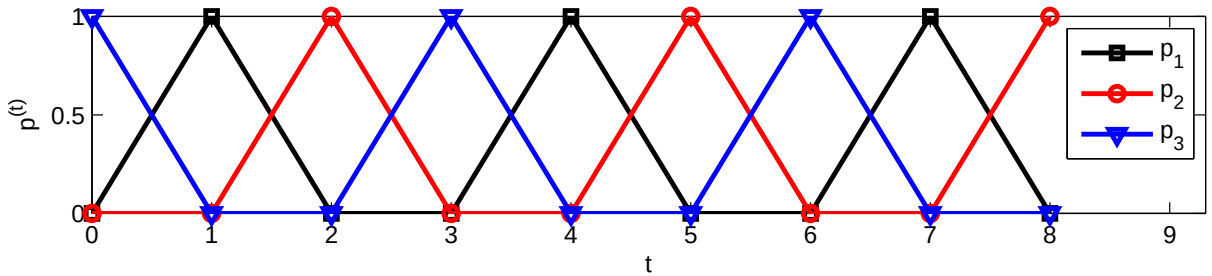
$$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

- All states communicate, but ...
- Effect of Initial distribution $p(s^0)$ on $p(s^t)$ does not diminish when $t \rightarrow \infty$

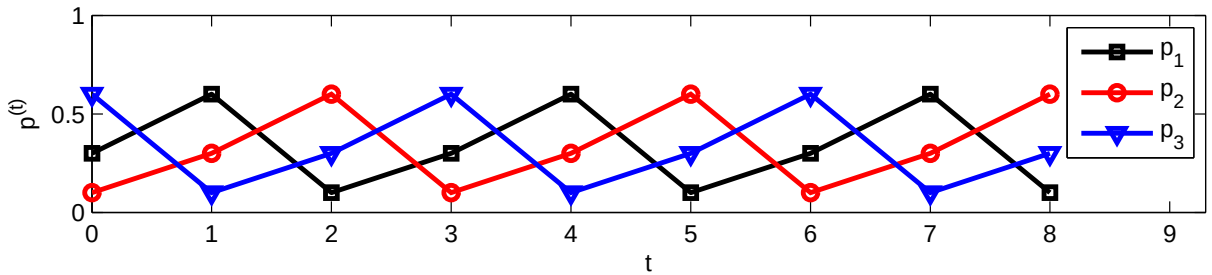
Example: Periodic

There is no stationary distribution

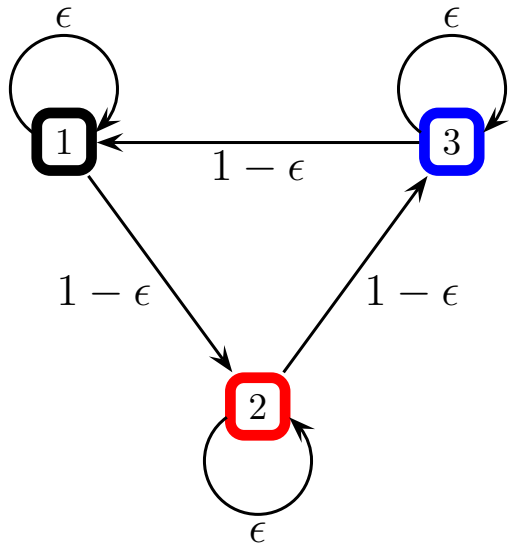
- $p^{(0)} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top$



- $p^{(0)} = \begin{pmatrix} 0.3 & 0.1 & 0.6 \end{pmatrix}^\top$



Example: Mixture

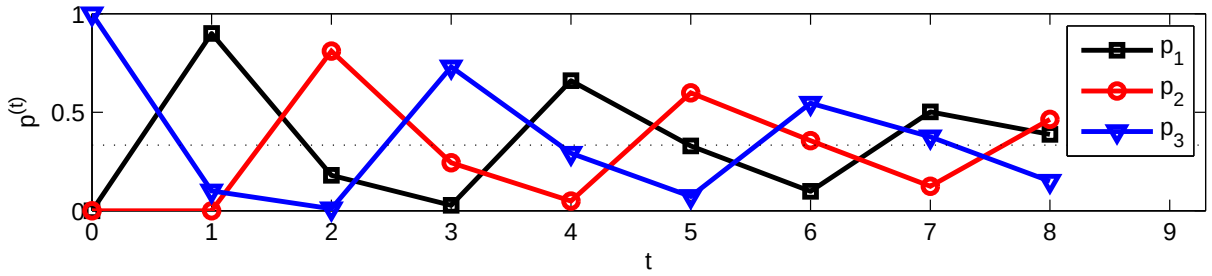


$$(1-\epsilon)\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} + \epsilon \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

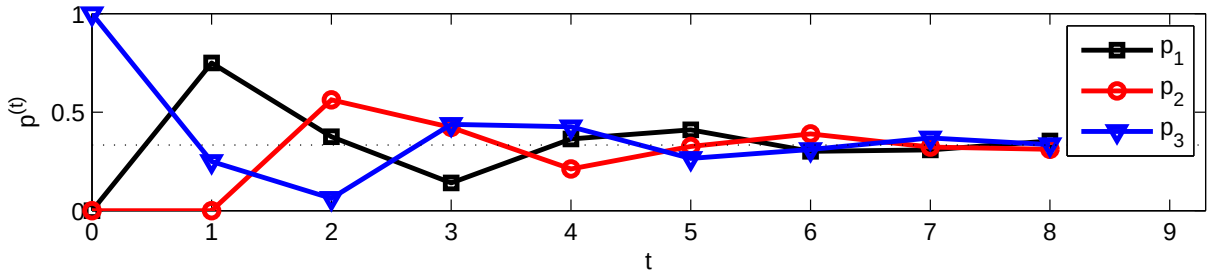
- All states communicate, not periodic
- Is there a unique stationary distribution?

Example: Mixture

- There is a stationary distribution
  $p^{(\infty)} = \begin{pmatrix} 1/3 & 1/3 & 1/3 \end{pmatrix}^\top$
- $\epsilon = 0.1$



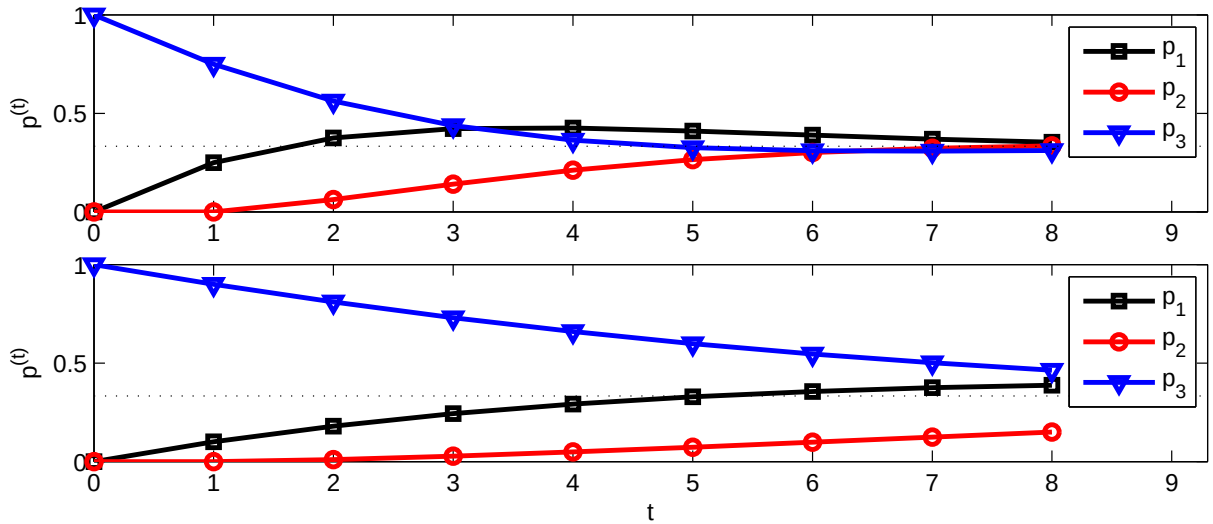
- $\epsilon = 0.25$



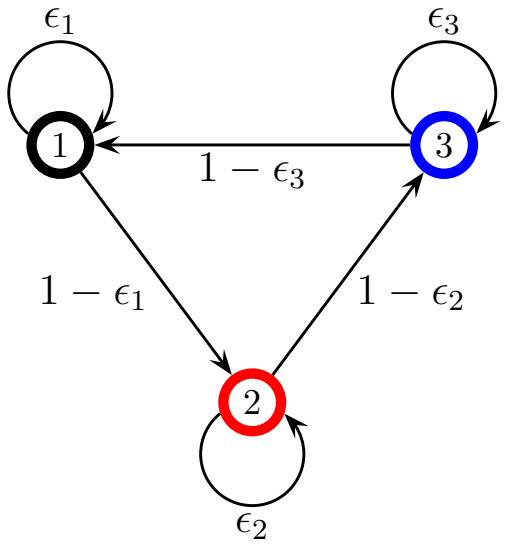
- Convergence rates are different

Example: Mixture

- There is a stationary distribution
  $p^{(\infty)} = \begin{pmatrix} 1/3 & 1/3 & 1/3 \end{pmatrix}^\top$
- $\epsilon = 0.75$
- $\epsilon =$



Example



$$\begin{pmatrix} \epsilon_1 & 0 & 1-\epsilon_3 \\ 1-\epsilon_1 & \epsilon_2 & 0 \\ 0 & 1-\epsilon_2 & \epsilon_3 \end{pmatrix}$$

- Self transition probabilities $\epsilon_1 > \epsilon_2 > \epsilon_3 \Rightarrow$ $p_1^{(\infty)} > p_2^{(\infty)} > p_3^{(\infty)}$, but the exact relationship is not trivial
- How can we find the stationary distribution ? How fast is the convergence ?
- How can we design a chain that will converge to a given target distribution ?

Stationary Distribution

- We compute an eigendecomposition

$$\begin{aligned} \mathbf{T} &= B\Lambda B^{-1} \\ \Lambda &= \mathbf{diag}(1, \lambda_2, \ldots, \lambda_K) \end{aligned}$$

- The stationary distribution is given by the limit

$$\begin{aligned} \lim_{t\to\infty} p^{(t)} &= \lim_{t\to\infty} \mathbf{T}^t p^{(0)} \\ \mathbf{T}^t &= B\Lambda B^{-1} B\Lambda \ldots \Lambda B^{-1} = B\Lambda^t B^{-1} \end{aligned}$$

- It turns out since $\mathbf{T}$ is a conditional probability matrix (columns sum up to one), the eigenvalues satisfy

$$1 = \lambda_1 \geq |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_K|$$

Stationary Distribution

- If and only if $|\lambda_2| < 1$, $\mathbf{T}^t$ goes to

$$B \begin{pmatrix} 1 & 0 & & 0 \\ 0 & \lambda_2^t & & 0 \\ & & \ddots & \\ 0 & & & \lambda_K^t \end{pmatrix} B^{-1} \xrightarrow{t \to \infty} B \begin{pmatrix} 1 & 0 & & 0 \\ 0 & 0 & & 0 \\ & & \ddots & \\ 0 & & & 0 \end{pmatrix} B^{-1}$$

$$= \begin{pmatrix} \pi_1 \\ \pi_2 \\ \vdots \\ \pi_K \end{pmatrix} \begin{pmatrix} 1 & 1 & \dots & 1 \end{pmatrix}$$

Matlab illustration

```
>> T = [0.1 0.9 0;0 0.7 0.3;0.2 0.8 0]';
>> [B L] = eig(T)
B =
   0.0637          -0.4714 - 0.3333i  -0.4714 + 0.3333i
   0.9559          -0.2357 + 0.3333i  -0.2357 - 0.3333i
   0.2868           0.7071             0.7071
L =
   1.0000                0                  0
        0          -0.1000 + 0.1414i        0
        0                0            -0.1000 - 0.1414i
```

Matlab illustration

```
>> inv(B)
ans =
   0.7655 + 0.0000i   0.7655 - 0.0000i   0.7655 + 0.0000i
  -0.1552 + 1.2073i  -0.1552 - 0.2927i   0.5519 + 0.7073i
  -0.1552 - 1.2073i  -0.1552 + 0.2927i   0.5519 - 0.7073i

>> B*L*inv(B)
ans =
   0.1000 + 0.0000i   0.0000 - 0.0000i   0.2000 + 0.0000i
   0.9000 + 0.0000i   0.7000 - 0.0000i   0.8000 + 0.0000i
  -0.0000 + 0.0000i   0.3000 + 0.0000i   0.0000

>> B*L^30*inv(B)
   0.0488 + 0.0000i   0.0488 - 0.0000i   0.0488 + 0.0000i
   0.7317 + 0.0000i   0.7317 - 0.0000i   0.7317 + 0.0000i
   0.2195 + 0.0000i   0.2195 - 0.0000i   0.2195 + 0.0000i
```

Rate of Convergence (for finite-state Markov Chains)

- Geometric Convergence property, there exist $c > 0$ s.t.

$$\|\mathbf{T}^t p^{(0)} - \pi\|_{\text{var}} \leq c|\lambda_2|^t$$

- However, it is hard to show algebraically that $|\lambda_2| < 1$. Fortunately, there is a...

Convergence Theorem (for finite-state Markov Chains)

- Finite State space $\mathcal{X} = \{1, 2, \ldots, K\}$
- $\mathbf{T}$ is irreducable and aperiodic, then there exist $0 < r < 1$ and $c > 0$ s.t.

$$\|\mathbf{T}^t p^{(0)} - \pi\|_{\text{var}} \leq c r^t$$

where π is the invariant distribution

$$\|P - Q\|_{\text{var}} \equiv \frac{1}{2} \sum_{s \in \mathcal{X}} |P(s) - Q(s)|$$

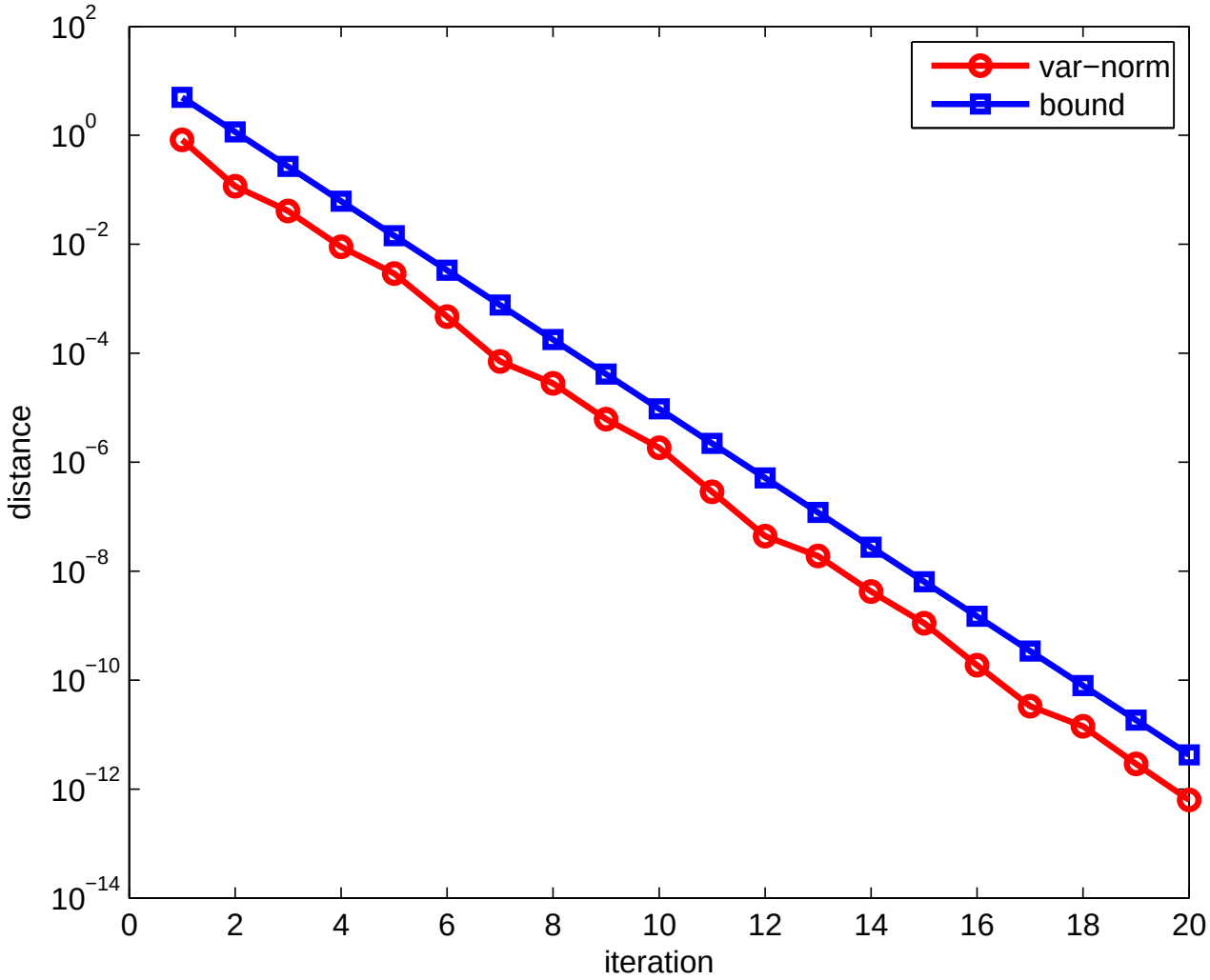
Example: Convergence in variation norm

```
N = 5;
% Generate a random transition matrix
T = rand(N); T = normalize(T,1);

% Compute the stationary distribution
% and second largest eigenvalue
[B L] = eig(T);
[dummy idx] = sort(abs(diag(L)), 'descend');
l2 = L(idx(2),idx(2));
sp = normalize(B(:,idx(1)));

% For the sake of the example
% choose the state with smallest probability
p = zeros(N,1); p(idx(end)) = 1;
```

Example: Convergence in variation norm

```
% iterate
EP = 20; d = zeros(1, EP); bound = zeros(1, EP);
for i=1:EP,
  d(i) = 0.5*sum(abs(p-sp));
  bound(i) = abs(N*l2^(i-1));
  p = T*p;
end;

% plot result
ln = semilogy(d, 'o-r'); set(ln, 'linew', 2);
hold on
ln = semilogy(bound, 's-b'); set(ln, 'linew', 2);
hold off
legend('var-norm', 'bound')
```

Example: Convergence in variation norm



Markov Chain Monte Carlo (MCMC)

- Construct a transition kernel $T(\mathbf{s}'|\mathbf{s})$ with the stationary distribution
  $\mathcal{P} = \phi(\mathbf{s})/Z_x \equiv \pi(\mathbf{s})$ for any initial distribution $r(\mathbf{s})$.

$$\pi(\mathbf{s}) = T^{\infty} r(\mathbf{s}) \tag{1}$$

- Sample $\mathbf{s}^{(0)} \sim r(\mathbf{s})$
- For $t = 1 \ldots \infty$, Sample $\mathbf{s}^{(t)} \sim T(\mathbf{s}|\mathbf{s}^{(t-1)})$
- Estimate any desired expectation by the average

$$\langle f(\mathbf{s}) \rangle_{\pi(\mathbf{s})} \approx \frac{1}{t - t_0} \sum_{n=t_0}^{t} f(\mathbf{s}^{(n)})$$

  where t_0 is a preset burn-in period.

But how to construct T and verify that $\pi(\mathbf{s})$ is indeed its stationary distribution ?

Proof Technique

- Show that the target distribution is a stationary distribution of the Markov chain
  - Verify detailed balance
- Show that the transition kernel T has a unique stationary distribution
  - Verify irreducibility and aperiodicity $\Rightarrow$ unique stationary distribution
    - Irreducibility (probabilisic connectedness): Every state s' can be reached from every s

$$T(s'|s) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$ is **not** irreducible

    - Aperiodicity : Cycling around is not allowed

$$T(s'|s) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$ is **not** aperiodic

Equilibrium condition = Detailed Balance

$$T(\mathbf{s}|\mathbf{s}')\pi(\mathbf{s}') \quad = \quad T(\mathbf{s}'|\mathbf{s})\pi(\mathbf{s})$$

If detailed balance is satisfied then $\pi(\mathbf{s})$ is a stationary distribution

$$\pi(\mathbf{s}) \quad = \quad \int d\mathbf{s}' T(\mathbf{s}|\mathbf{s}')\pi(\mathbf{s}')$$

If the configuration space is discrete, we have

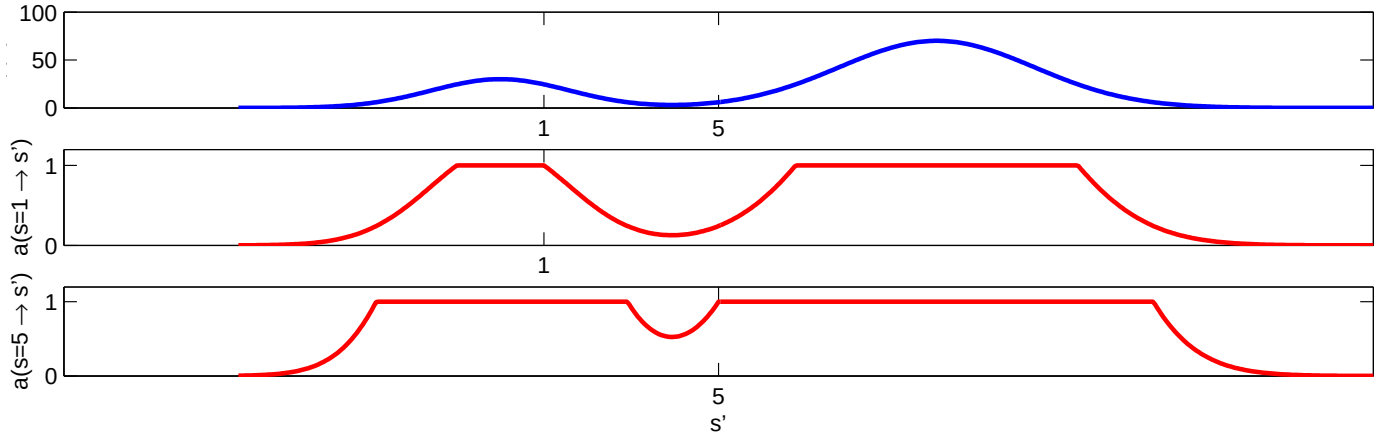
$$\begin{aligned} \pi(\mathbf{s}) &= \sum_{\mathbf{s}'} T(\mathbf{s}|\mathbf{s}')\pi(\mathbf{s}') \\ \pi &= T\pi \end{aligned}$$

π has to be a (right) eigenvector of T.

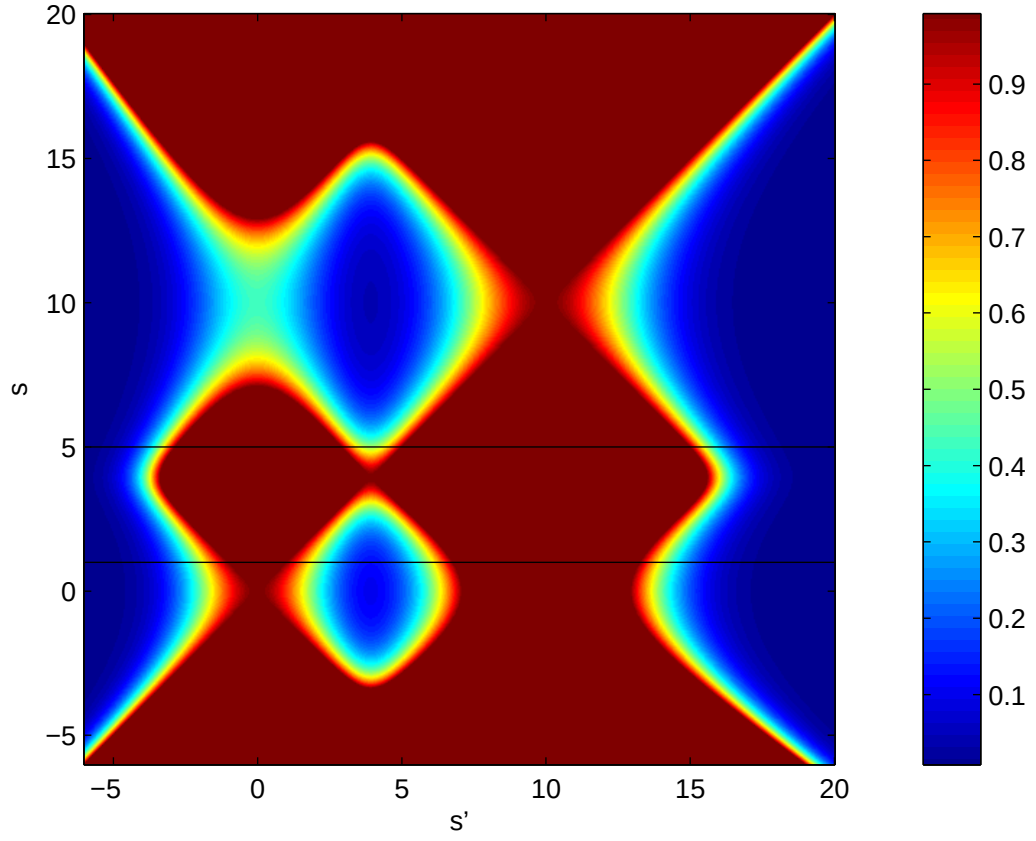
Metropolis-Hastings Kernel

- We choose an arbitrary proposal distribution $q(s'|s)$ (that satisfies mild regularity conditions).
  (When q is symmetric, i.e., $q(s'|s) = q(s|s')$, we have a Metropolis algorithm.)
- We define the *acceptance probability* of a jump from s to s' as

$$a(s \to s') \equiv \min\{1, \frac{q(s|s')\pi(s')}{q(s'|s)\pi(s)}\}$$



Acceptance Probability $a(s \rightarrow s')$



Basic MCMC algorithm: Metropolis-Hastings

1. Initialize: $s^{(0)} \sim r(s)$
2. For $t = 1, 2, \ldots$
   - Propose:

$$s' \sim q(s'|s^{(t-1)})$$

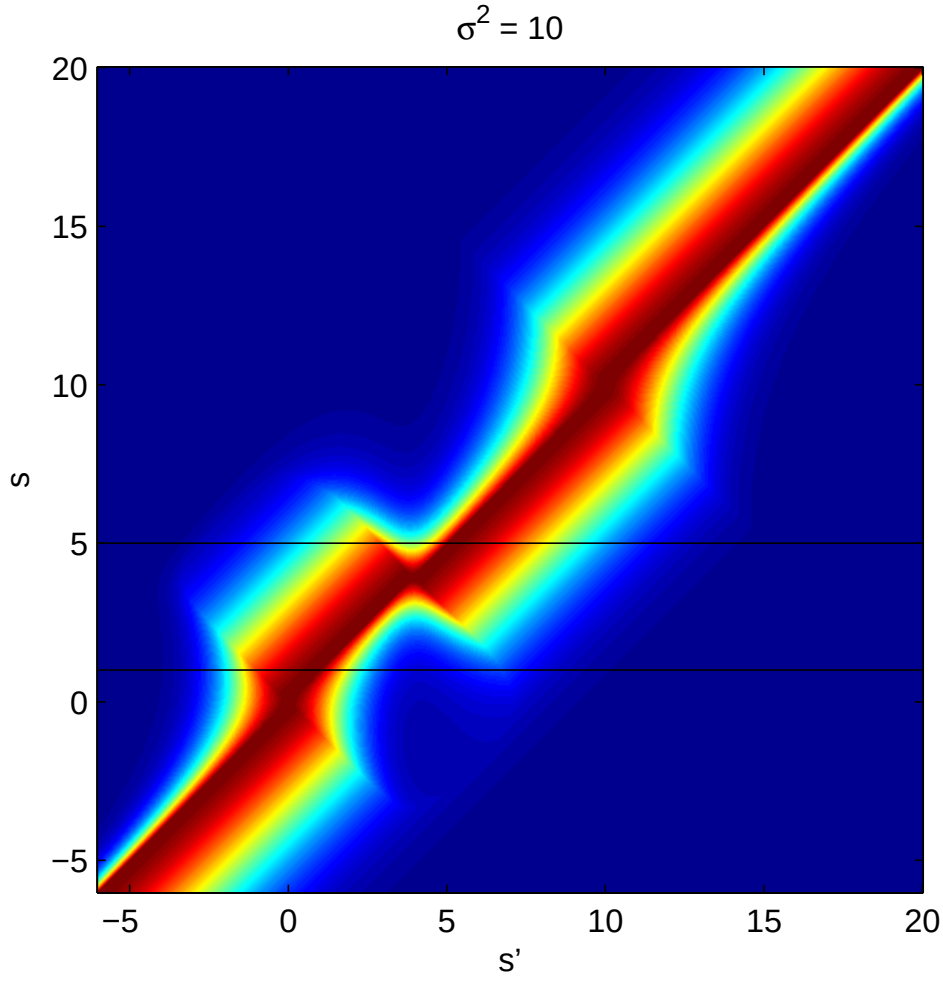
   - Evaluate Proposal: $u \sim \text{Uniform}[0, 1]$

$$s^{(t)} := \begin{cases} s' & u < a(s^{(t-1)} \to s') \quad \text{Accept} \\ s^{(t-1)} & \text{otherwise} \quad \text{Reject} \end{cases}$$

Transition Kernel of the Metropolis-Hastings

$$
\begin{aligned}
T(s'|s) &= \underbrace{q(s'|s)a(s \to s')}_{\text{Accept}} + \underbrace{\delta(s'-s)\rho(s)}_{\text{Reject}} \\
\rho(s) &\equiv \int ds' q(s'|s)(1 - a(s \to s'))
\end{aligned}
$$

Transition Kernel of the Metropolis-Hastings



Only Accept part for visual convenience

Verification of detailed balance for Metropolis

- Target Density

$$\pi(s) = \frac{1}{Z}\phi(s)$$

- Acceptance Probability

$$a(s \to s') = \min\{1, \frac{\pi(s')}{\pi(s)}\} = \min\{1, \frac{\phi(s')}{\phi(s)}\}$$

- Symmetric Proposal

$$q(s|s') = q(s'|s)$$

Verification of detailed balance for Metropolis

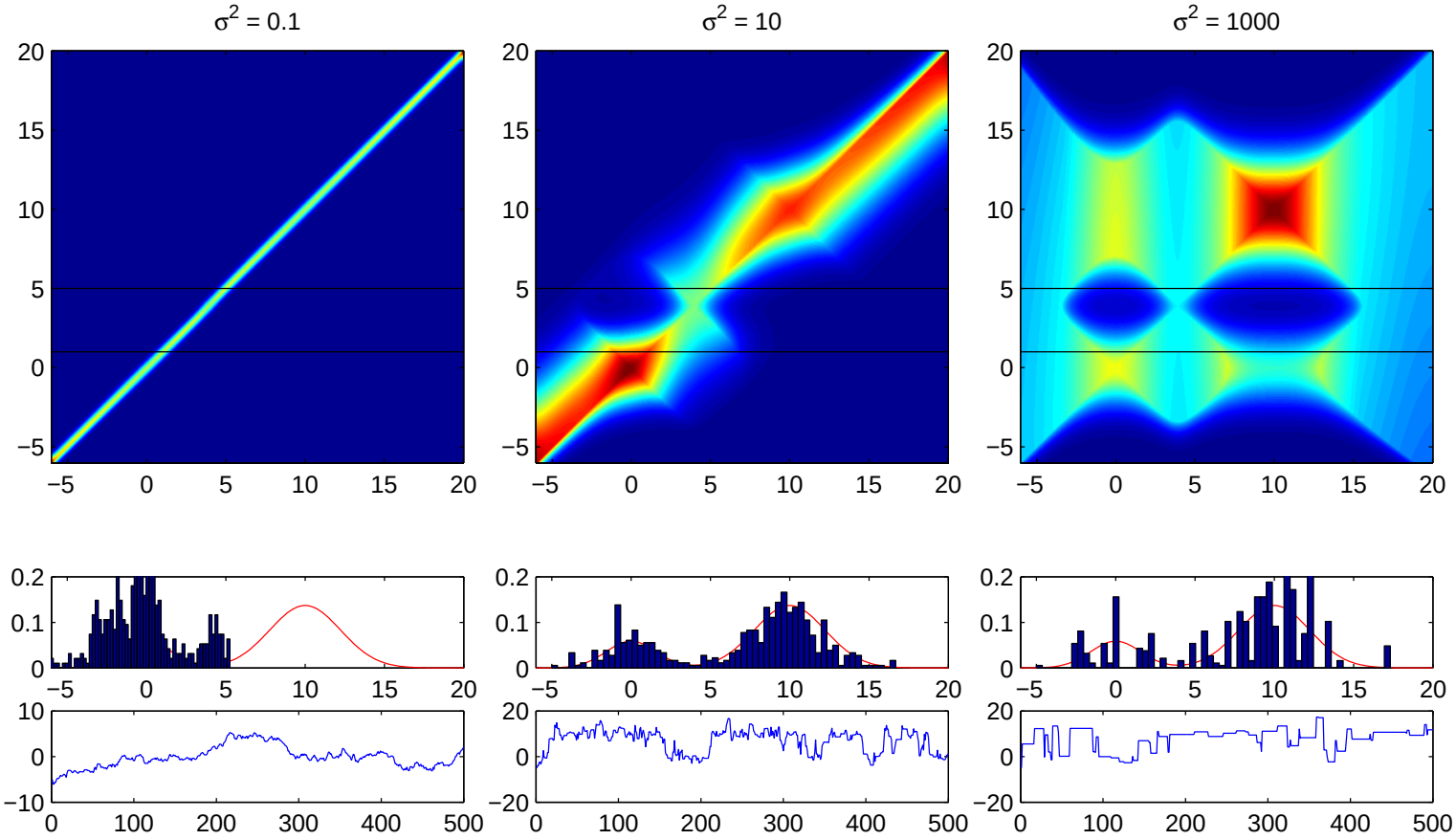
$$
\begin{aligned}
T(s'|s)\pi(s) &= q(s'|s)\min\{1, \frac{\phi(s')}{\phi(s)}\}\pi(s) \quad + \delta(s-s')\rho(s)\pi(s) \\
&= q(s'|s)\min\{\frac{\phi(s)}{Z}, \frac{\phi(s')}{\phi(s)}\frac{\phi(s)}{Z}\} + \delta(s-s')\rho(s)\pi(s) \\
&= q(s'|s)\min\{\frac{\phi(s)}{Z}, \frac{\phi(s')}{Z}\} + \pi(s')\rho(s')\delta(s'-s) \\
&= q(s|s')\frac{\phi(s')}{Z}\min\{\frac{\phi(s)/Z}{\phi(s')/Z}, 1\} + \frac{\phi(s')}{Z}\rho(s')\delta(s'-s) \\
&= T(s|s')\pi(s')
\end{aligned}
$$

Verification of detailed balance for Metropolis-Hastings

$$
\begin{aligned}
\pi(s) &= \frac{1}{Z}\phi(s) \\
a(s \to s') &= \min\{1, \frac{q(s|s')\pi(s')}{q(s'|s)\pi(s)}\} = \min\{1, \frac{q(s|s')\phi(s')}{q(s'|s)\phi(s)}\}
\end{aligned}
$$

$$
\begin{aligned}
T(s'|s)\pi(s) &= q(s'|s)\min\{1, \frac{q(s|s')\phi(s')}{q(s'|s)\phi(s)}\}\frac{\phi(s)}{Z} + \delta(s-s')\rho(s)\frac{\phi(s)}{Z} \\
&= \min\{q(s'|s)\frac{\phi(s)}{Z}, \frac{q(s|s')\phi(s')}{Z}\} + \frac{\phi(s')}{Z}\rho(s')\delta(s'-s) \\
&= T(s|s')\pi(s')
\end{aligned}
$$

Various Kernels with the same stationary distribution



$$q(s'|s) = \mathcal{N}(s'; s, \sigma^2)$$

Cascades and Mixtures of Transition Kernels

Let T_1 and T_2 have the same stationary distribution $p(s)$.
Then:

$$\begin{aligned} T_c &= T_1 T_2 \\ T_m &= \nu T_1 + (1-\nu) T_2 \quad 0 \leq \nu \leq 1 \end{aligned}$$

are also transition kernels with stationary distribution $p(s)$.
This opens up many possibilities to "tailor" application specific algorithms.
For example let

T_1 : global proposal (allows large "jumps")
T_2 : local proposal (investigates locally)

We can use T_m and adjust ν as a function of rejection rate.

CMPE 58N - Lecture 4
Monte Carlo methods

The Gibbs Sampler, Applications

Department of Computer Engineering,

Boğaziçi University, Istanbul, Turkey

Instructor: A. Taylan Cemgil

Fall 2009

Outline

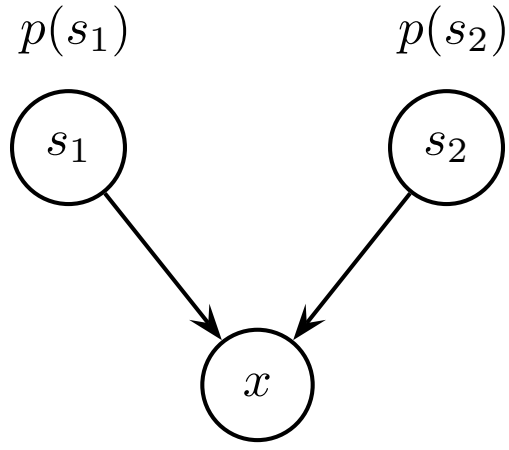
- Motivating Example
- The Gibbs sampler
- Gibbs Application: Change point model
- MH Application: Sensor fusion

The Gibbs sampler

Algorithm 1 Gibbs sampler

1: Initialize: $x^{(0)} \sim q(x)$
2: **for** $i = 1, 2 \ldots$ **do**
3: $\quad x_1^{(i)} \sim p(x_1 | X_2 = x_2^{(i-1)}, X_3 = x_3^{(i-1)}, \ldots, X_N = x_N^{(i-1)})$
4: $\quad x_2^{(i)} \sim p(x_2 | x_1 = x_1^{(i)}, X_3 = x_3^{(i-1)}, \ldots, X_N = x_N^{(i-1)})$
$\quad \vdots$
5: $\quad x_N^{(i)} \sim p(x_N | x_1 = x_1^{(i)}, X_2 = x_2^{(i)}, \ldots, X_{N-1} = x_{N-1}^{(i)})$
6: **end for**

Motivating Example : Gibbs Sampler

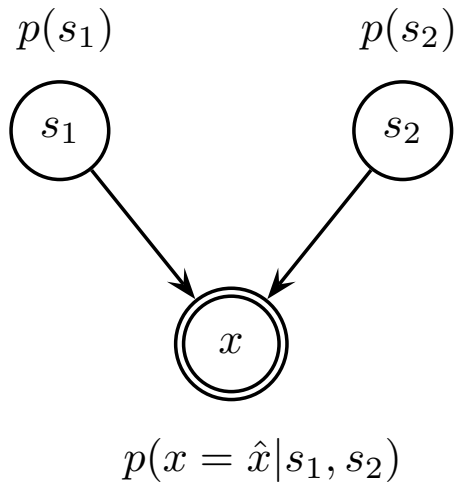


This graph encodes the joint: $p(x, s_1, s_2) = p(x|s_1, s_2)p(s_1)p(s_2)$

$$
\begin{aligned}
s_1 &\sim p(s_1) = \mathcal{N}(s_1; \mu_1, P_1) \\
s_2 &\sim p(s_2) = \mathcal{N}(s_2; \mu_2, P_2) \\
x|s_1, s_2 &\sim p(x|s_1, s_2) = \mathcal{N}(x; s_1 + s_2, R)
\end{aligned}
$$

Toy example

Suppose, we observe $x = \hat{x}$.



- By Bayes' theorem, the posterior is given by:

$$\mathcal{P} \equiv p(s_1, s_2|x = \hat{x}) = \frac{1}{Z_{\hat{x}}} p(x = \hat{x}|s_1, s_2) p(s_1) p(s_2) \equiv \frac{1}{Z_{\hat{x}}} \phi(s_1, s_2)$$

- The function $\phi(s_1, s_2)$ is proportional to the exact posterior. ($Z_{\hat{x}} \equiv p(x = \hat{x})$)

Toy example, cont.

$$
\begin{aligned}
\log p(s_1) &= \mu_1^T P_1^{-1} s_1 - \frac{1}{2} s_1^T P_1^{-1} s_1 + \text{const} \\
\log p(s_2) &= \mu_2^T P_2^{-1} s_2 - \frac{1}{2} s_2^T P_2^{-1} s_2 + \text{const} \\
\log p(x|s_1, s_2) &= \hat{x}^T R^{-1} (s_1 + s_2) - \frac{1}{2} (s_1 + s_2)^T R^{-1} (s_1 + s_2) + \text{const}
\end{aligned}
$$

Toy example, cont.

$$
\begin{aligned}
\log \phi(s_1, s_2) \quad &= \quad \log p(x = \hat{x} | s_1, s_2) \; + \log p(s_1) \; + \log p(s_2) \\
&=^{+} \quad \left(\mu_1^T P_1^{-1} + \hat{x}^T R^{-1} \right) s_1 + \left(\mu_2^T P_2^{-1} + \hat{x}^T R^{-1} \right) s_2 \\
&\qquad -\frac{1}{2} \mathbf{Tr} \left(P_1^{-1} + R^{-1} \right) s_1 s_1^T - \underbrace{s_1^T R^{-1} s_2}_{(*)} \\
&\qquad -\frac{1}{2} \mathbf{Tr} \left(P_2^{-1} + R^{-1} \right) s_2 s_2^T
\end{aligned}
$$

- The (*) term is the cross correlation term that makes s_1 and s_2 a-posteriori dependent.

Toy example, cont.

Completing the square

$$\log \phi(s_1, s_2) \quad =^+ \quad \begin{pmatrix} P_1^{-1}\mu_1 + R^{-1}\hat{x} \\ P_2^{-1}\mu_2 + R^{-1}\hat{x} \end{pmatrix}^\top \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$$
$$-\frac{1}{2} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}^\top \begin{pmatrix} P_1^{-1} + R^{-1} & R^{-1} \\ R^{-1} & P_2^{-1} + R^{-1} \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$$

Remember: $\log \mathcal{N}(s; m, \Sigma) \quad =^+ \quad (\Sigma^{-1} m)^\top s - \frac{1}{2} s^\top \Sigma^{-1} s$

$$\Sigma \quad = \quad \begin{pmatrix} P_1^{-1} + R^{-1} & R^{-1} \\ R^{-1} & P_2^{-1} + R^{-1} \end{pmatrix}^{-1} \qquad m = \Sigma \qquad \begin{pmatrix} P_1^{-1}\mu_1 + R^{-1}\hat{x} \\ P_2^{-1}\mu_2 + R^{-1}\hat{x} \end{pmatrix}$$

Gibbs sampler

- We define the following iterative schema to generate a Markov Chain

$$
\begin{aligned}
s_1^{(t+1)} &\sim p(s_1 | s_2^{(t)}, x = \hat{x}) &&\propto \phi(s_1, s_2^{(t)}) \\
s_2^{(t+1)} &\sim p(s_2 | s_1^{(t+1)}, x = \hat{x}) &&\propto \phi(s_1^{(t+1)}, s_2)
\end{aligned}
$$

- The desired posterior $\mathcal{P}$ is the stationary distribution of T

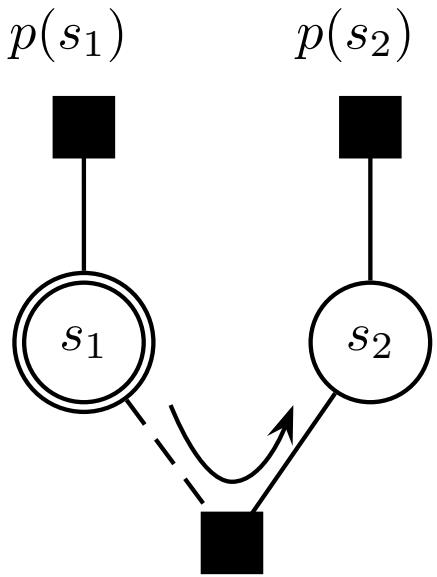
Gibbs sampler

- A remarkable fact is that we can estimate any desired expectation by ergodic averages

$$\langle f(\mathbf{s}) \rangle_{\mathcal{P}} \approx \frac{1}{t - t_0} \sum_{n=t_0}^{t} f(\mathbf{s}^{(n)})$$

- Consecutive samples $\mathbf{s}^{(t)}$ are dependent but we can "pretend" as if they are independent!

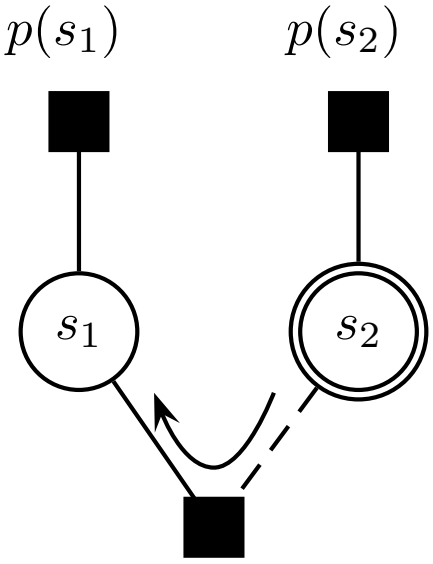
Gibbs Sampler



$$p(x = \hat{x}|s_1, s_2)$$

$$s_2^{(t+1)} \sim \mathcal{N}(s_2; m_2(s_1^{(t+1)}), S_2)$$

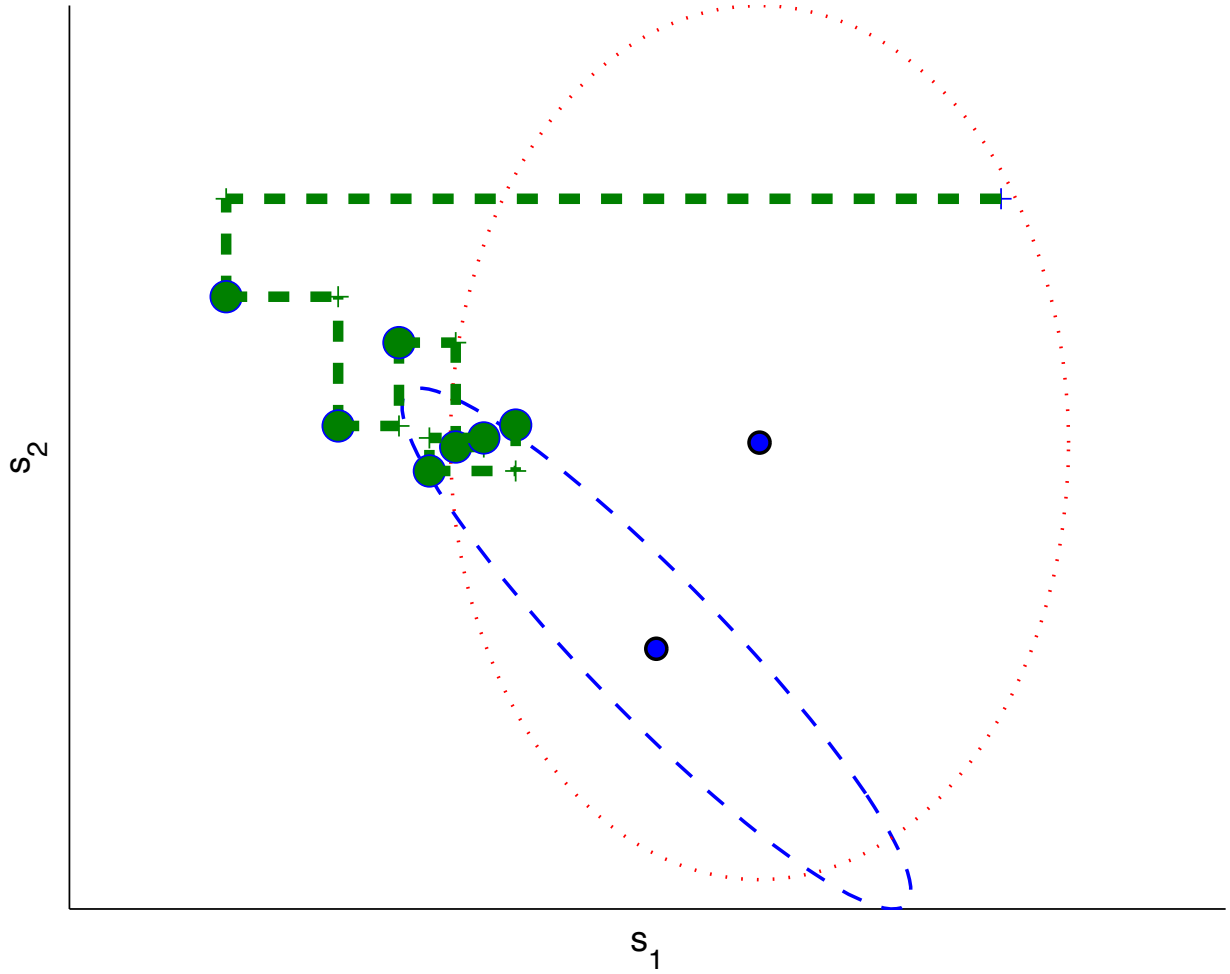
Gibbs Sampler



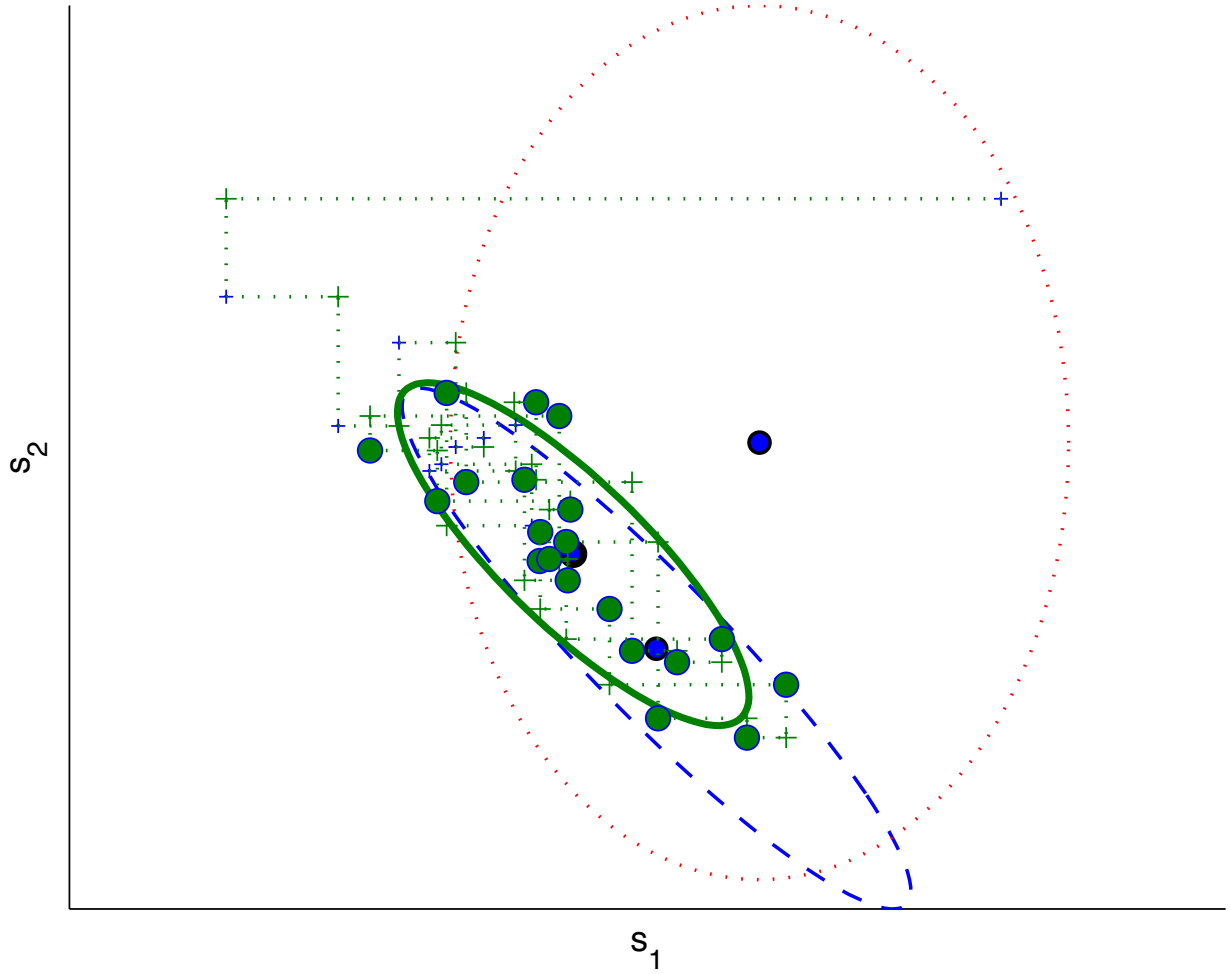
$$p(x = \hat{x}|s_1, s_2)$$

$$s_1^{(t+1)} \sim \mathcal{N}(s_1; m_1(s_2^{(t)}), S_1)$$

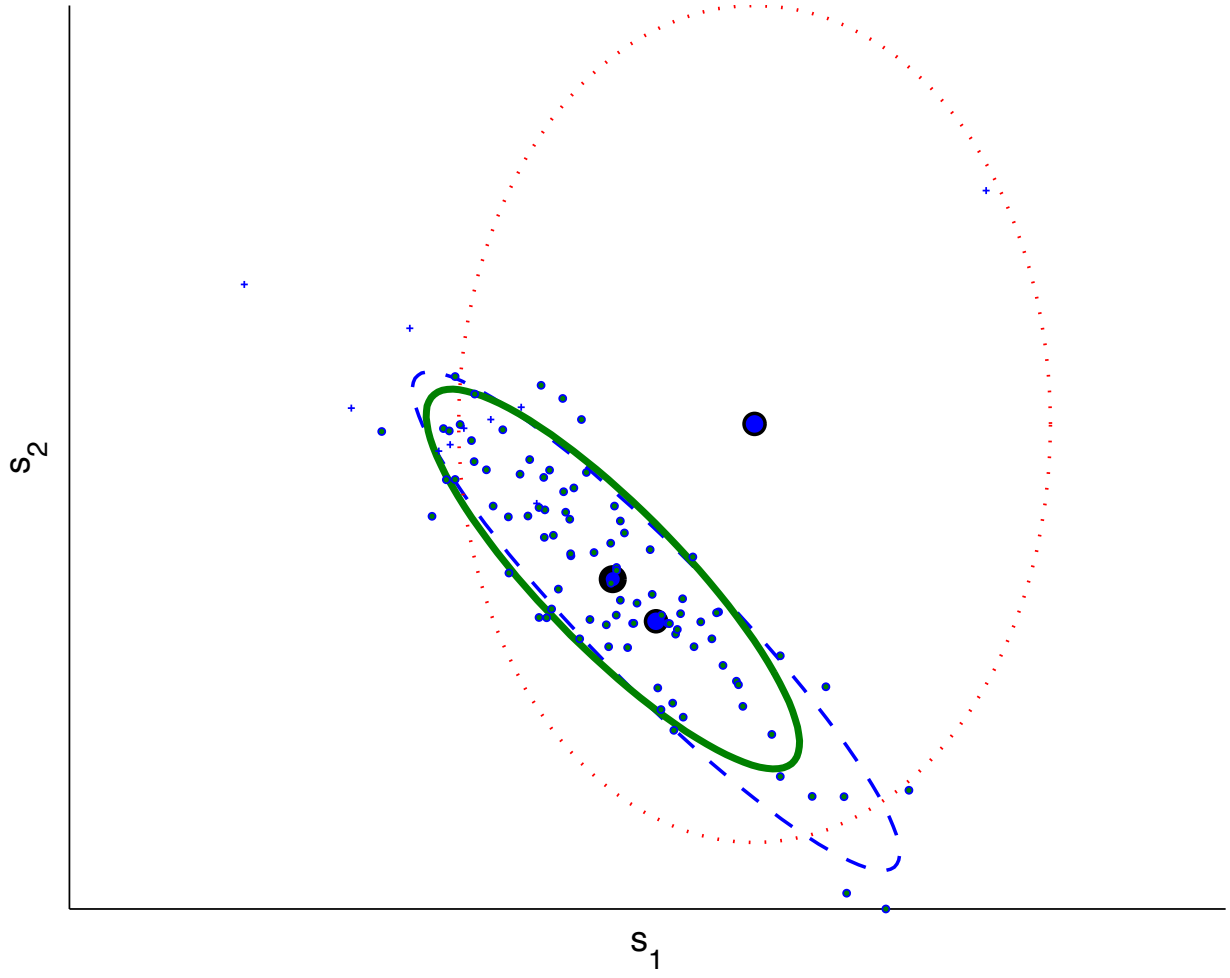
Gibbs Sampler



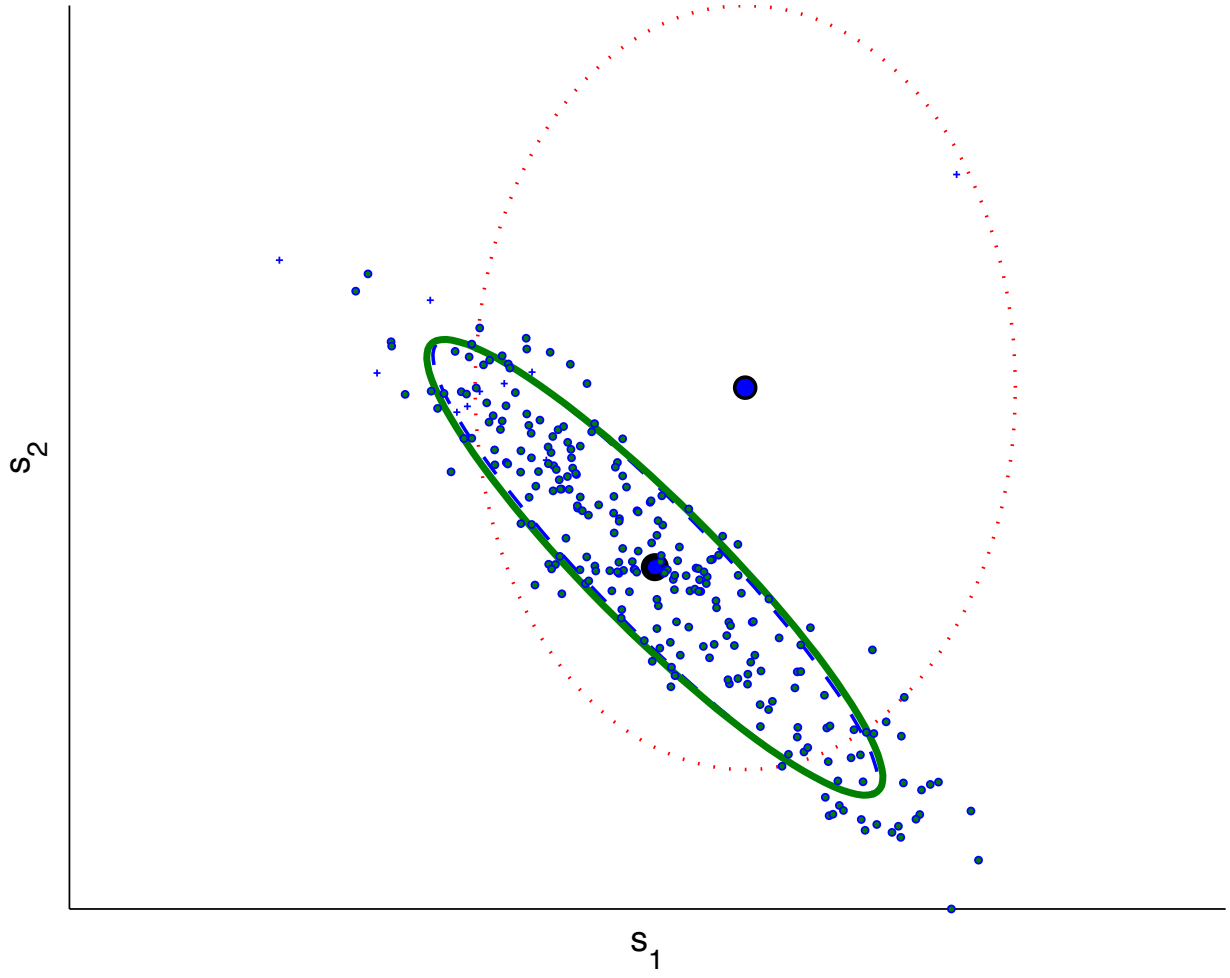
Gibbs Sampler, $t = 20$



Gibbs Sampler, $t = 100$



Gibbs Sampler, $t = 250$



Technicalities: Finding the full conditionals

$$s_1^{(t+1)} \sim p(s_1|s_2^{(t)}, x=\hat{x}) \propto \phi(s_1, s_2^{(t)}) \equiv \phi_1$$

Eliminate terms that don't depend on s_1

$$\log \phi_1 = \log p(x=\hat{x}|s_1, s_2^{(t)}) + \log p(s_1) + \log p(s_2^{(t)})$$

$$=^{+} \underbrace{\mu_1^\top P_1^{-1} s_1 - \frac{1}{2} s_1{}^\top P_1^{-1} s_1}_{\log p(s_1)} + \underbrace{\hat{x}^\top R^{-1}(s_1 + s_2^{(t)}) - \frac{1}{2}(s_1 + s_2^{(t)})^\top R^{-1}(s_1 + s_2^{(t)})}_{p(x=\hat{x}|s_1, s_2^{(t)})}$$

Technicalities: Finding the full conditionals (cont.)

$$\log \phi_1 \quad =^{+} \quad \left(\mu_1^\top P_1^{-1} + (\hat{x} - s_2^{(t)})^\top R^{-1}\right) s_1 - \frac{1}{2} \mathbf{Tr}\left(P_1^{-1} + R^{-1}\right) s_1 s_1{}^\top$$

$$p(s_1 | s_2^{(t)}, x = \hat{x}) \quad = \quad \mathcal{N}(s_1; m_1, S_1)$$

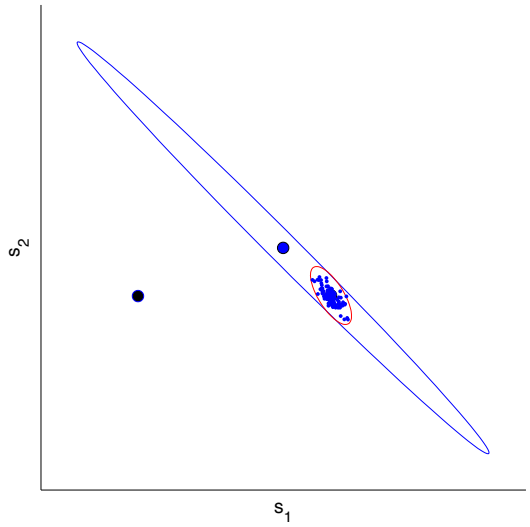
$$\begin{aligned} S_1 &= \left(P_1^{-1} + R^{-1}\right)^{-1} \\ m_1(s_2^{(t)}) &= S_1 \left(P_1^{-1}\mu_1 + R^{-1}(\hat{x} - s_2^{(t)})\right) \end{aligned}$$

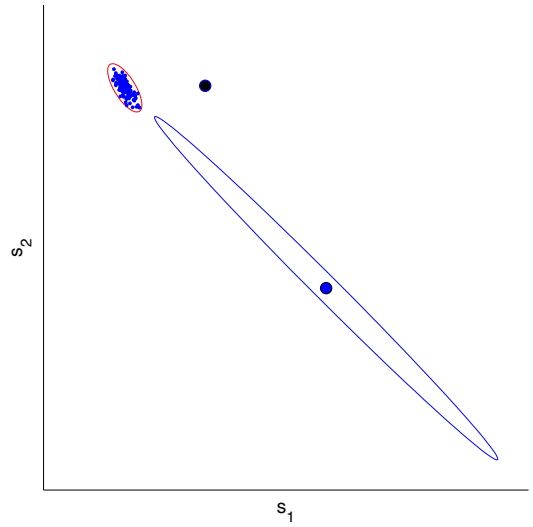
The transition kernel

$$
\begin{aligned}
T(s_{1:2}^{(t+1)}|s_{1:2}^{(t)}, s_2^{(t)}) &= T(s_2^{(t+1)}|s_1^{(t+1)}, s_1^{(t)}, s_2^{(t)}) T(s_1^{(t+1)}|s_1^{(t)}, s_2^{(t)}) \\
&= T(s_2^{(t+1)}|s_1^{(t+1)}) T(s_1^{(t+1)}|s_2^{(t)}) \\
&= \mathcal{N}(s_2^{(t+1)}; m_2(s_1^{(t+1)}), S_2) \mathcal{N}(s_1^{(t+1)}; m_1(s_2^{(t)}), S_1)
\end{aligned}
$$

Therefore, the transition kernel is also Gaussian.

The transition kernel





- But why does the chain converge to the target distribution?
- The Gibbs sampler is a special MH algorithm

Metropolis-Hastings

Algorithm 2 Metropolis-Hastings

1: Initialize: $x^{(0)} \sim q(x_0)$
2: **for** $i = 1, 2 \ldots$ **do**
3: Propose: $\xi^{(i)} \sim q(x|x^{(i-1)})$
4: Acceptance Probability:

$$\alpha(\xi^{(i)}|x^{(i-1)}) = \min\{1, \frac{q(x^{(i-1)}|\xi^{(i)})\pi(\xi^{(i)})}{q(\xi^{(i)}|x^{(i-1)})\pi(x^{(i-1)})}\}$$

5: Uniform: $u \sim \mathcal{U}(u; 0, 1)$
6: **if** $u < \alpha$ **then**
7: Accept: $x^{(i)} \leftarrow \xi^{(i)}$
8: **else**
9: Reject: $x^{(i)} \leftarrow x^{(i-1)}$
10: **end if**
11: **end for**

The acceptance probability for Gibbs

$$
\begin{aligned}
\xi &\sim p(x_n | x_{-n} = x_{-n}^{(i-1)}) \\
\alpha &\equiv \alpha\left(\xi, x_{-n}^{(i-1)} | x_n^{(i-1)}, x_{-n}^{(i-1)}\right)
\end{aligned}
$$

$$
\begin{aligned}
\alpha &= \min\left\{1, \frac{q(x_n^{(i-1)}, x_{-n}^{(i-1)} | \xi, x_{-n}^{(i-1)}) p(\xi, x_{-n}^{(i-1)})}{q(\xi, x_{-n}^{(i-1)} | x_n^{(i-1)}, x_{-n}^{(i-1)}) p(x_n^{(i-1)}, x_{-n}^{(i-1)})}\right\} \\
&= \min\left\{1, \frac{p(x_n^{(i-1)} | x_{-n}^{(i-1)}) p(\xi, x_{-n}^{(i-1)})}{p(\xi | x_{-n}^{(i-1)}) p(x_n^{(i-1)}, x_{-n}^{(i-1)})}\right\} \\
&= \min\left\{1, \frac{p(x_n^{(i-1)} | x_{-n}^{(i-1)}) p(\xi | x_{-n}^{(i-1)}) p(x_{-n}^{(i-1)})}{p(\xi | x_{-n}^{(i-1)}) p(x_n^{(i-1)} | x_{-n}^{(i-1)}) p(x_{-n}^{(i-1)})}\right\} = 1
\end{aligned}
$$

Example: A change-point model

- We have a sequence of counts x_j.
- The average of the counts is jumping to a new value after an unknown index

Generative model

- We can model the counts with Poisson random variables $x_j, j = 1 \ldots M$

$$\begin{aligned} \mathcal{PO}(x;\lambda) &= e^{-\lambda}\frac{\lambda^x}{x!} \\ &= \exp\left(+x\log\lambda - \lambda - \log(x!)\right) \end{aligned}$$

- The unknown intensity can be modelled by a Gamma random variable

$$\begin{aligned} \mathcal{G}(\lambda;a,b) &\equiv \frac{1}{\Gamma(a)}b^a\lambda^{a-1}\exp(-b\lambda) \\ &= \exp\left((a-1)\log\lambda - b\lambda - \log\Gamma(a) + a\log b\right) \end{aligned}$$

- The unknown intensity λ_1 is jumping to a new unknown value λ_2 after an unknown index m

Generative model

$$
\begin{aligned}
m &\sim \mathcal{U}\{1 \dots M\} \\
\lambda_i &\sim \mathcal{G}(\lambda_i; a, b) \\
x_j &\sim \begin{cases} \mathcal{PO}(x_j; \lambda_1) & 1 \leq j \leq m \\ \mathcal{PO}(x_j; \lambda_2) & m < j \leq M \end{cases}
\end{aligned}
$$

Goal: Compute $p(\lambda_1, \lambda_2, m | x_{1:M})$

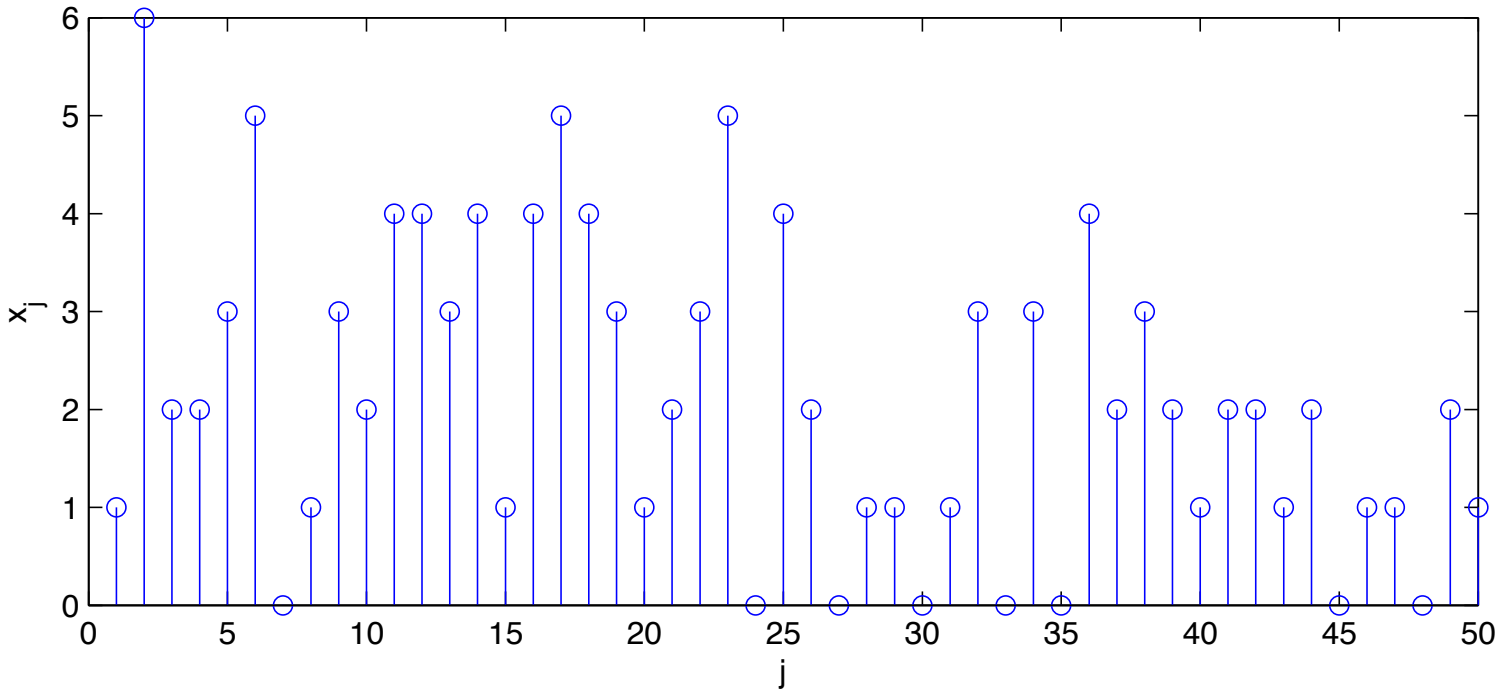
Generate data

```
% Fix the seeds for reproducibility
randn('seed', 1); rand('seed', 1);

% Hyperparameters
data.M = 50;
data.a = 2; data.b = 1;

% Changepoint
data.m = ceil(data.M*rand);

% Intensities, % !! Beware of the Gamma parametrisation !!
data.lambda = gamrnd(data.a, 1/data.b, [1 2]);

Lambda = zeros(1, M);
Lambda(1:data.m) = data.lambda(1);
Lambda((data.m+1):data.M) = data.lambda(2);

% Counts
data.x = poissrnd(Lambda);
```

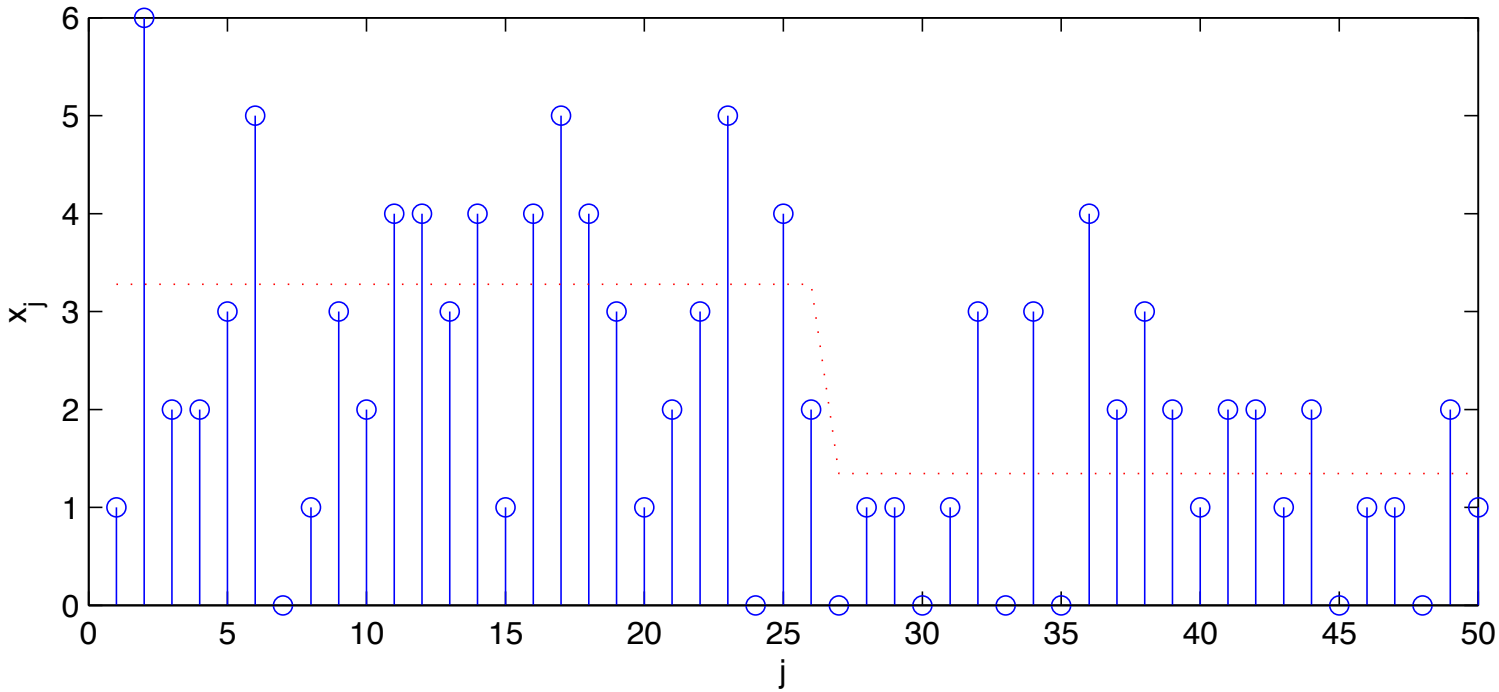
Generate data (cont.)

```
%% Visualise
stem(data.x); xlabel('j'); ylabel('x_j')
```

A Change-point model



A Change-point model



Full Joint Density

$$
\begin{aligned}
p(x_{1:M}, \lambda_1, \lambda_2, m) &= p(x_{1:M}|\lambda_1, \lambda_2, m)p(\lambda_1)p(\lambda_2)p(m) \\
&= \left(\prod_{j=1}^{m} p(x_j|\lambda_1)\right)\left(\prod_{j=m+1}^{M} p(x_j|\lambda_2)\right)p(\lambda_1)p(\lambda_2)p(m)
\end{aligned}
$$

Full Conditional Distributions

$$
\begin{aligned}
\mathcal{L} &= \log p(x_{1:M}|\lambda_1,\lambda_2,m) + \log p(\lambda_1) + \log p(\lambda_2) + \log p(m) \\
&= \sum_{j=1}^{m} (+x_j \log\lambda_1 - \lambda_1 - \log(x_j!)) \\
&\quad + \sum_{j=m+1}^{M} (+x_j \log\lambda_2 - \lambda_2 - \log(x_j!)) \\
&\quad +(a-1)\log\lambda_1 - b\lambda_1 - \log\Gamma(a) + a\log b \\
&\quad +(a-1)\log\lambda_2 - b\lambda_2 - \log\Gamma(a) + a\log b - \log M
\end{aligned}
$$

Full Conditional Distributions

$$
\begin{aligned}
\log p(\lambda_1 | m, \lambda_2, x_{1:M}) &=^{+} \sum_{j=1}^{m} (+x_j \log \lambda_1 - \lambda_1) \\
&\quad +(a-1)\log\lambda_1 - b\lambda_1 \\
&= \left(a + \sum_{j=1}^{m} x_j - 1 \right) \log\lambda_1 - (m+b)\lambda_1 \\
&=^{+} \log \mathcal{G}(a + \sum_{j=1}^{m} x_j, m+b)
\end{aligned}
$$

Full Conditional Distributions

$$
\begin{aligned}
\log p(\lambda_2 | m, \lambda_1, x_{1:M}) &=^{+} \sum_{j=m+1}^{M} (+x_j \log \lambda_2 - \lambda_2) \\
&\quad +(a-1)\log \lambda_2 - b\lambda_2 \\
&=^{+} \log \mathcal{G}(a + \sum_{j=m+1}^{M} x_j, M - m + b)
\end{aligned}
$$

Full Conditional Distributions

$$
\begin{aligned}
\log p(m|\lambda_1, \lambda_2, x_{1:M}) &= \sum_{j=1}^{m} (+x_j \log \lambda_1 - \lambda_1 - \log(x_j!)) \\
&\quad + \sum_{j=m+1}^{M} (+x_j \log \lambda_2 - \lambda_2 - \log(x_j!)) \\
&= + \left(\sum_{j=1}^{m} x_j \right) \log \lambda_1 - m\lambda_1 \\
&\quad + \left(\sum_{j=m+1}^{M} x_j \right) \log \lambda_2 - (M-m)\lambda_2
\end{aligned}
$$

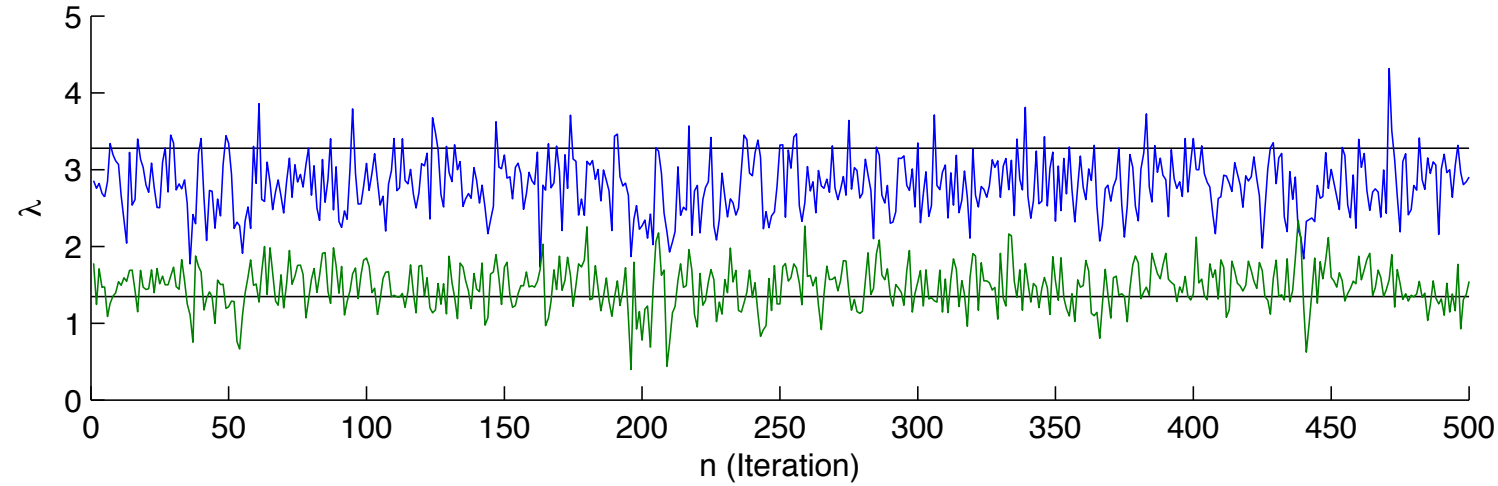
The Gibbs sampler

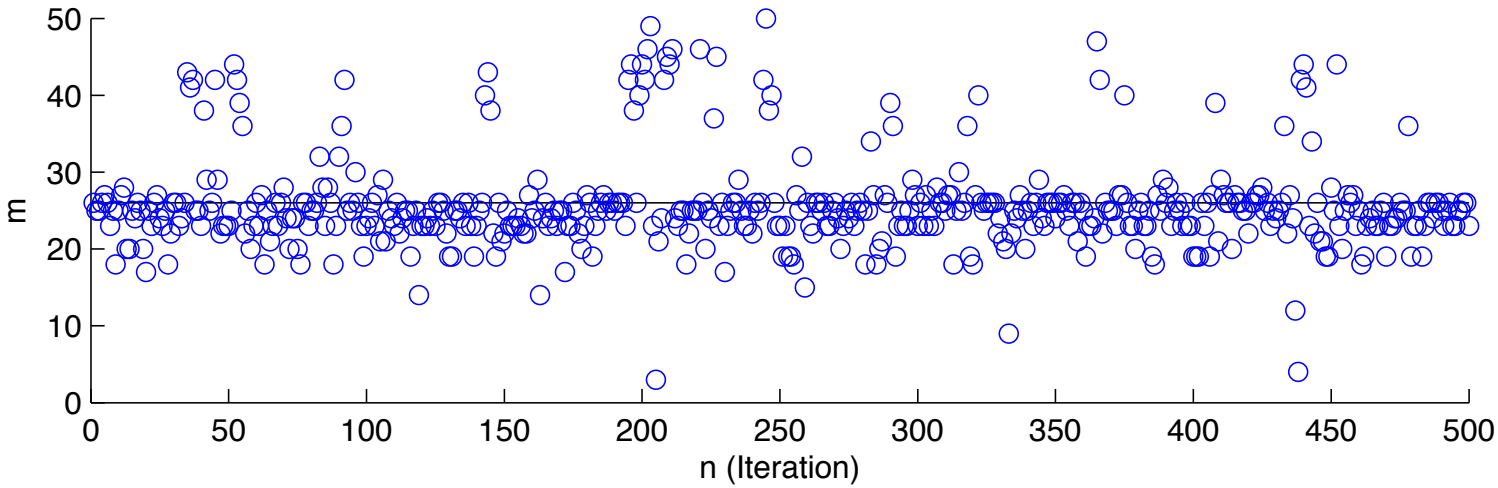
```
% Gibbs sampler
EP = 5200; % Number of epochs
BURN_IN = 200;

% Initial state of the Markov  chain
m = 10; lam = zeros(2,1);
% Storage
chain.m = zeros(1, EP-BURN_IN);
chain.lambda = zeros(2, EP-BURN_IN);
```

The Gibbs sampler (cont.)

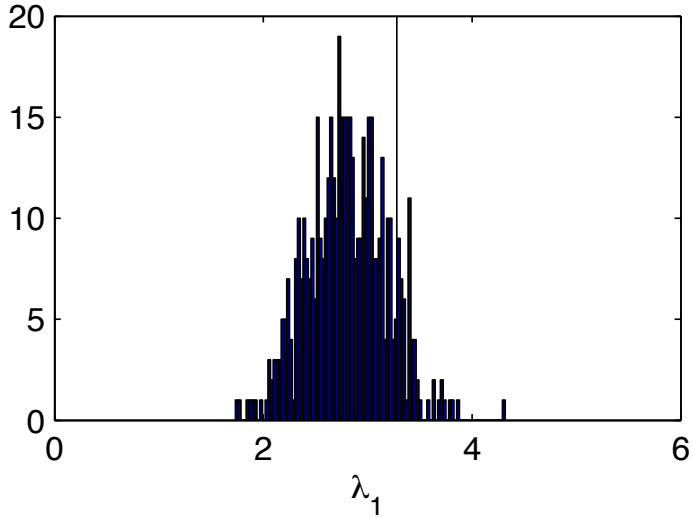
```
for e=1:EP,
  % lambda ~ p(lambda | x, m)
  lam(1) = gamrnd(data.a + sum(data.x(1:m)), 1/(m+data.b) );
  lam(2) = gamrnd(data.a + sum(data.x(m+1:end)), 1/(data.M-m+da

 % m ~ p(m | x, lambda)
  lp = zeros(1, data.M);
  for i=1:data.M,
    lp(i) = sum(data.x(1:i)).*log(lam(1)) - i*lam(1) +...
 sum(data.x((i+1):data.M)).*log(lam(2)) - (data.M-i)*lam(2) ;
  end;
  m = randgen(exp(lp-max(lp)));

  if e>BURN_IN,
    chain.lambda(:, e-BURN_IN) = lam;
    chain.m(1, e-BURN_IN) = m;
  end;
end;
```

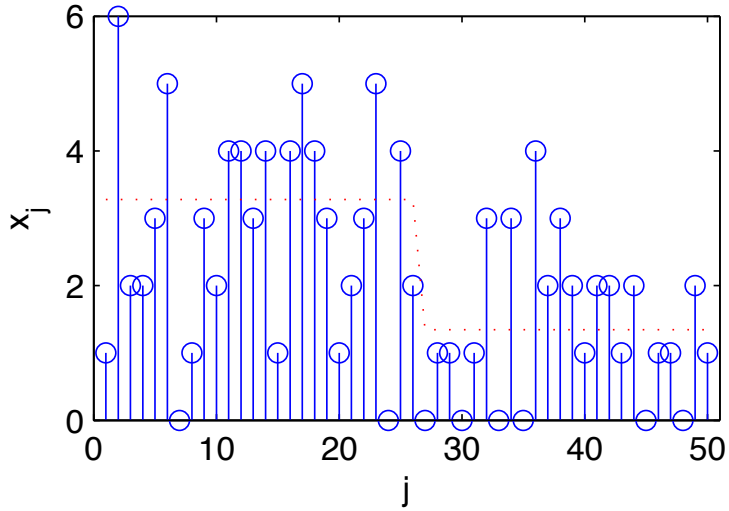
Results

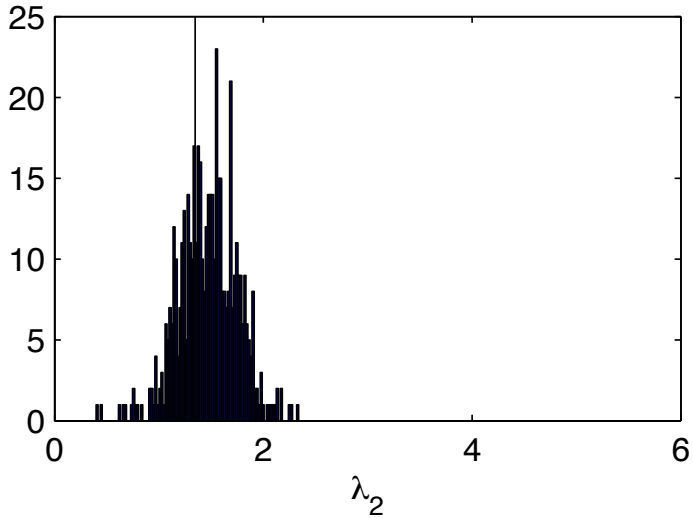


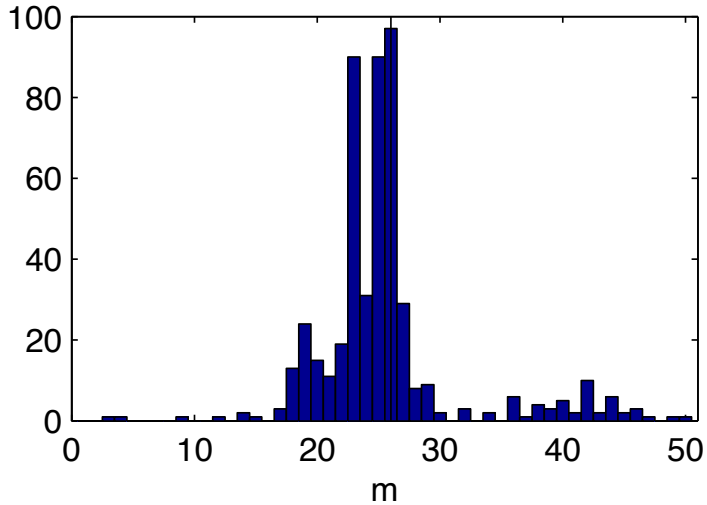


Results

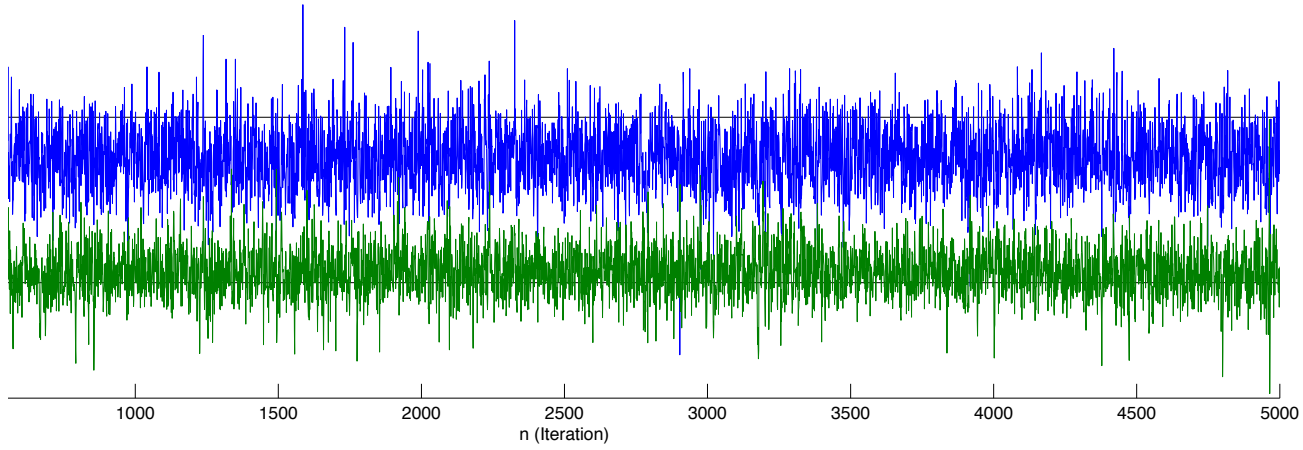

20
15
10
5
0
0
2
4
6
λ1

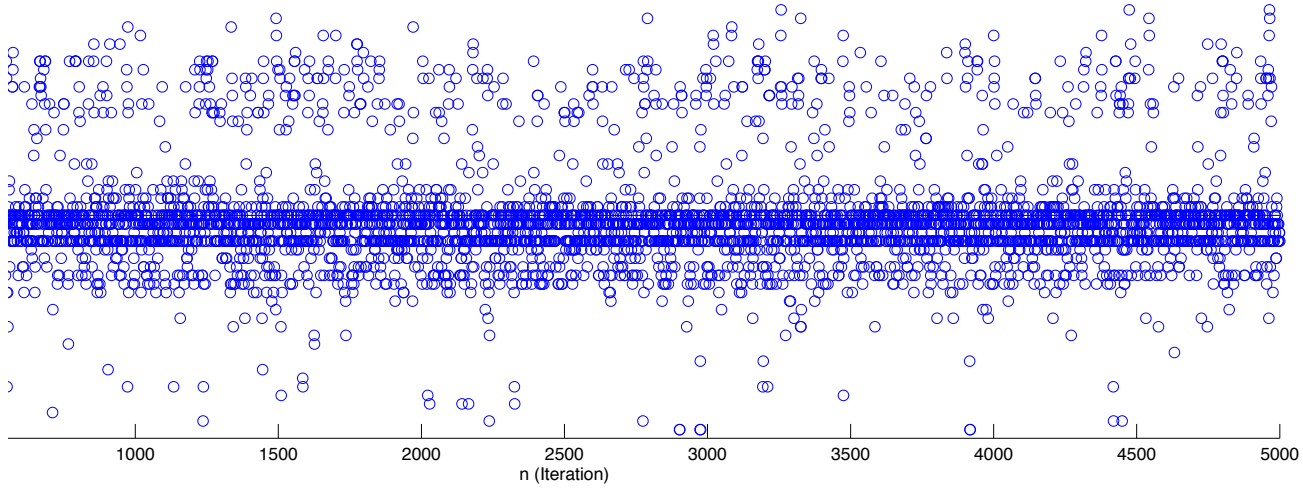

6
4
2
0
xj
0
10
20
30
40
50
j


25
20
15
10
5
0
0
2
4
6
λ2

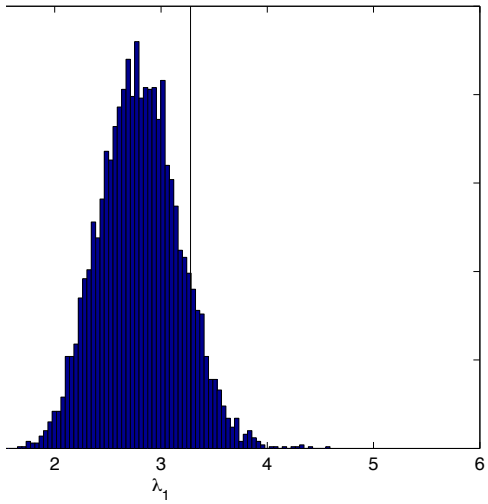

100
80
60
40
20
0
0
10
20
30
40
50
m

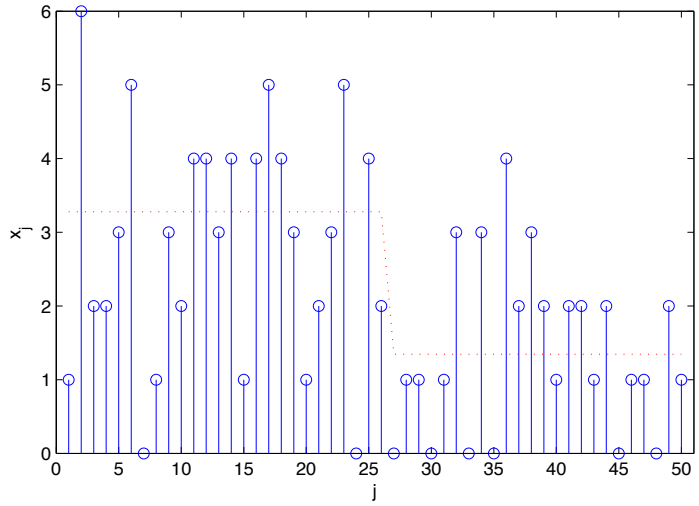
Results, Longer chain

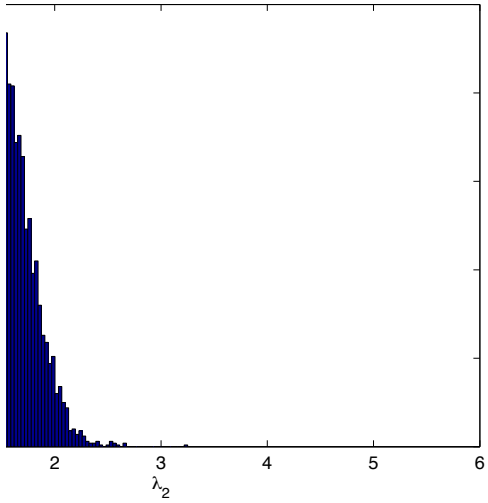


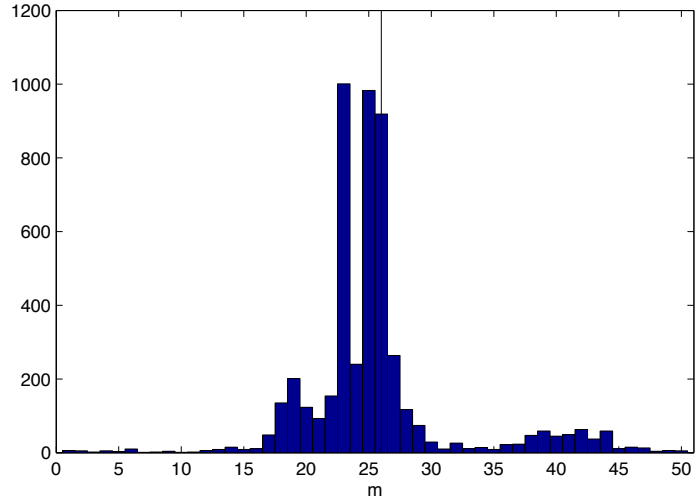


Results, Longer chain









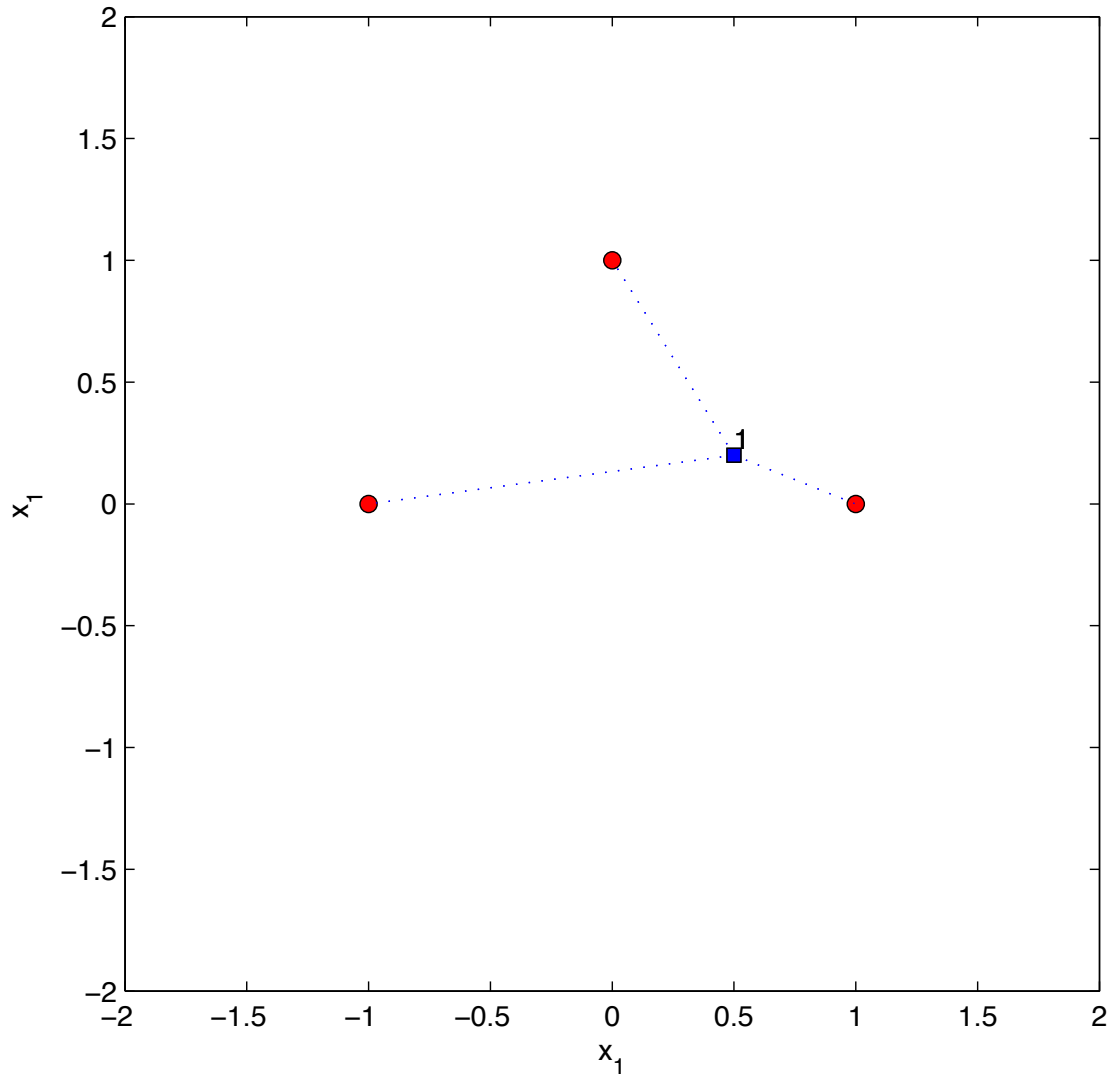
Metropolis Hastings Example

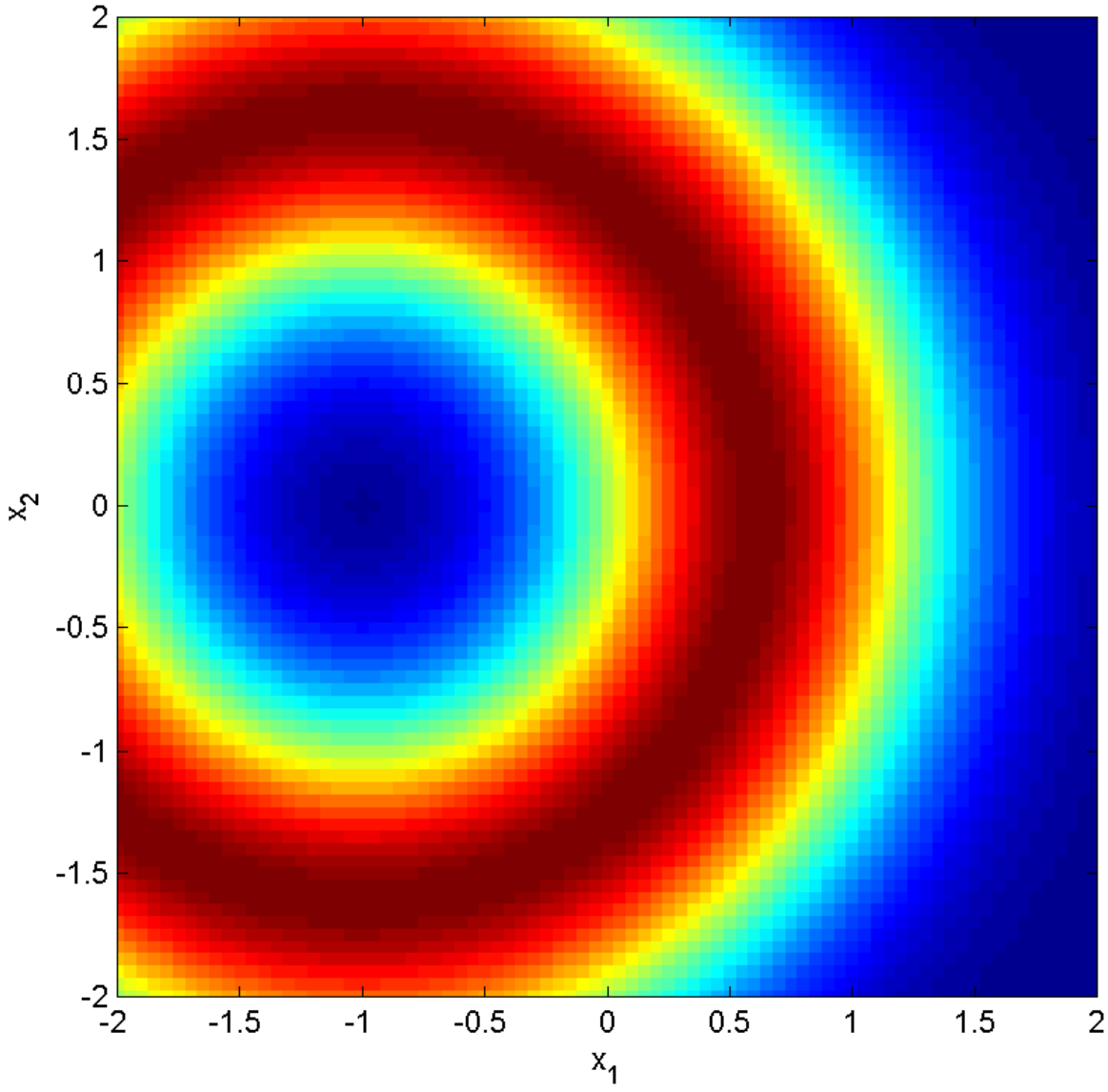
- Problem : Object position estimation from range measurements
- We have L sensors at known positions s_j, $j = 1 \ldots L$, each measuring the distance of a point object with noise

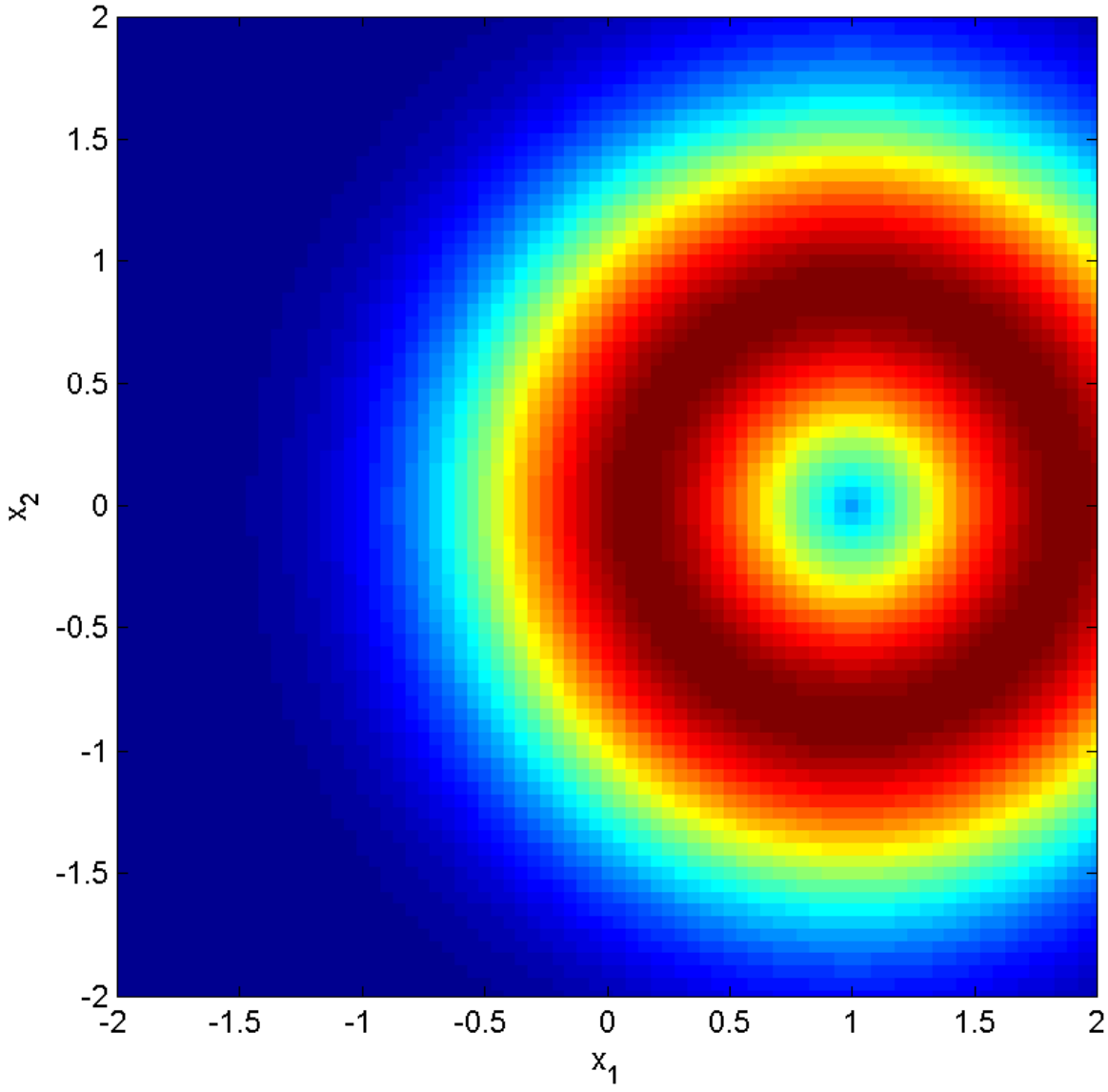
Range measurements

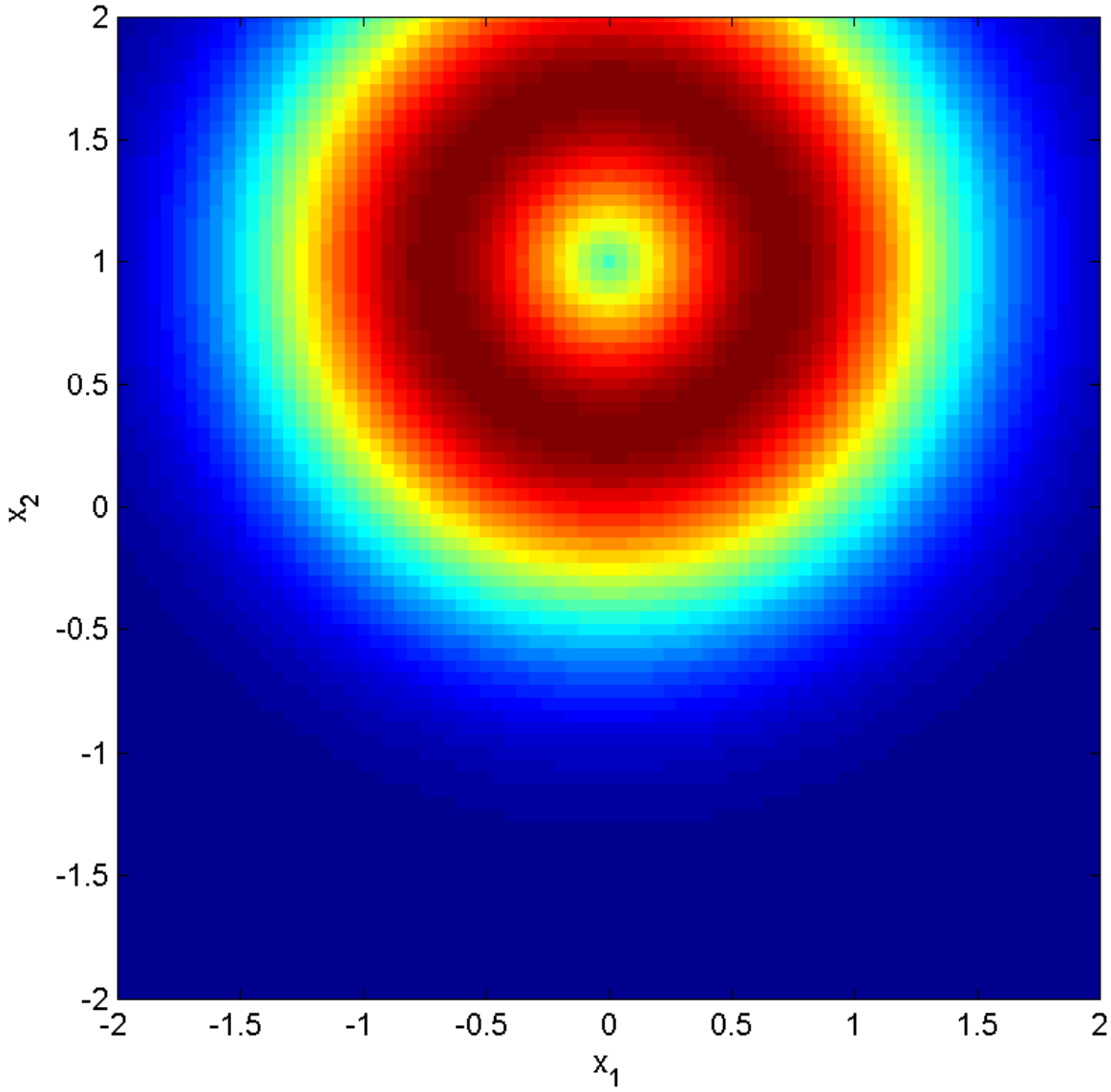
$$
\begin{aligned}
x &\sim p(x) \propto 1 \\
y_j | x, s_j &\sim \mathcal{N}(y_j; \|x - s_j\|, R)
\end{aligned}
$$

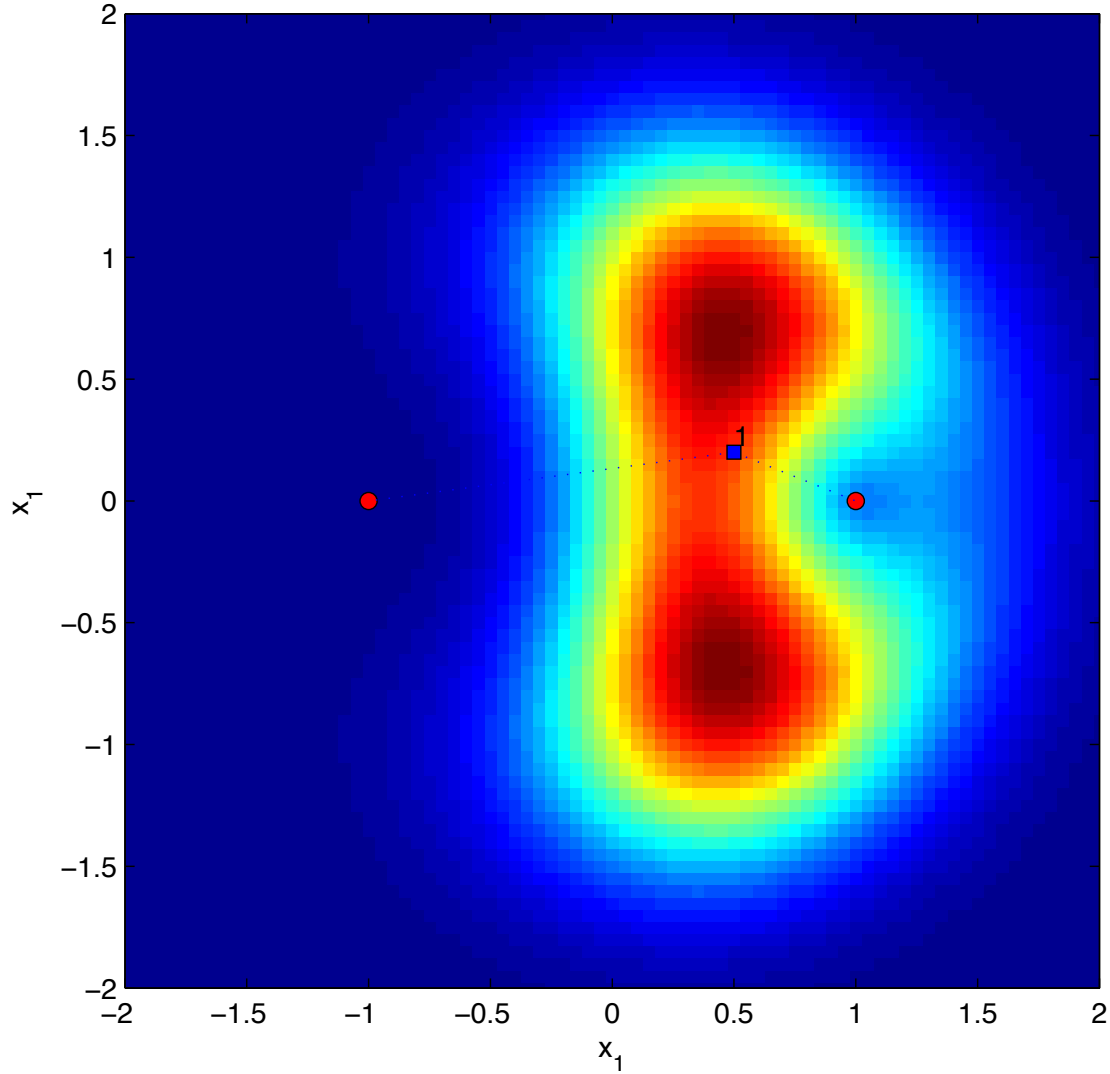
$$
\|x\| \equiv \left(\sum_k x_k^2 \right)^{1/2}
$$

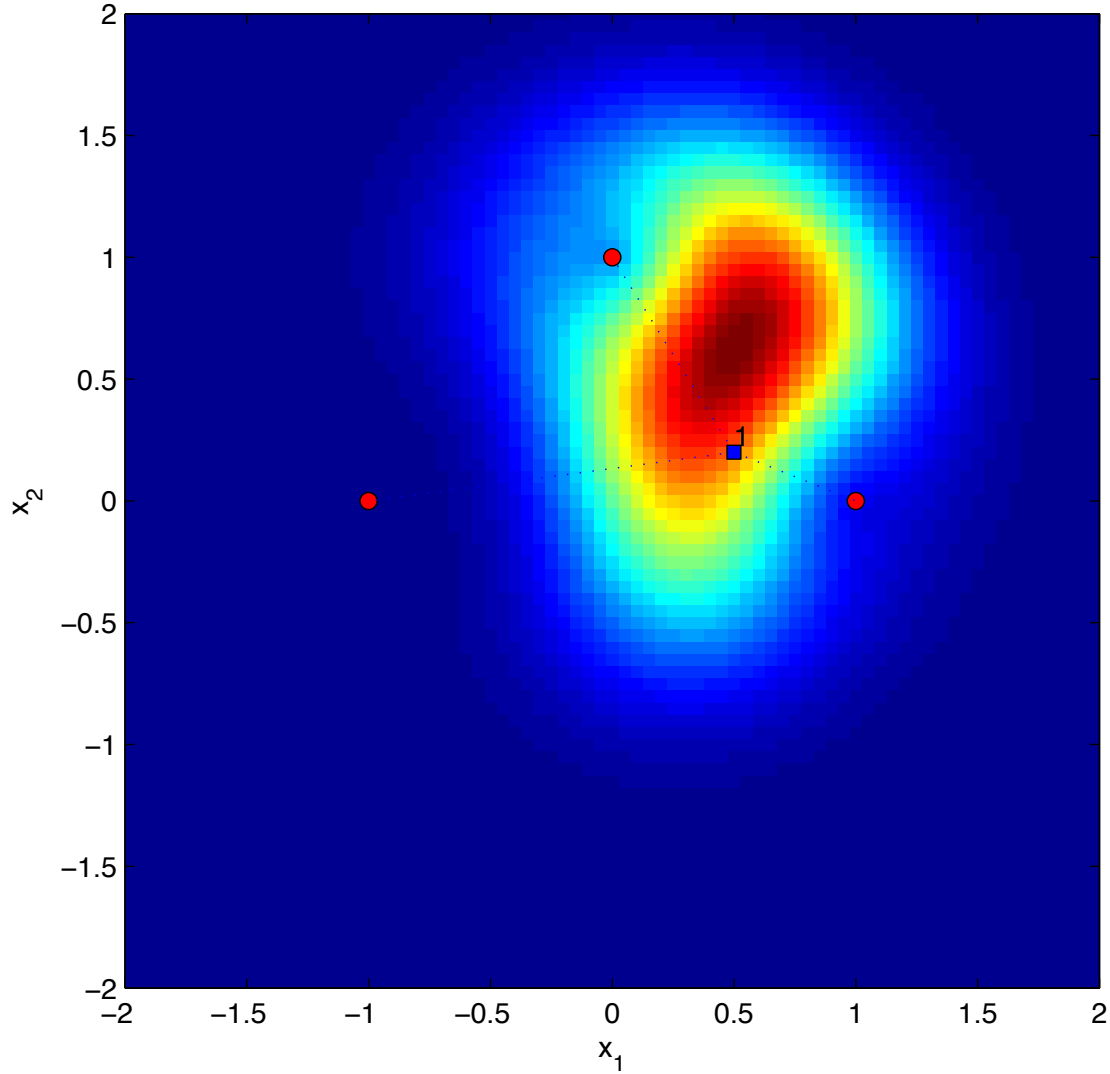

2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
−2
−1.5
−1
−0.5
0
0.5
1
1.5
2
x1
x1
1


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
x2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
x1


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
x_2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
x_1


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
x2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
x1


2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
x1
−2
−1.5
−1
−0.5
0
0.5
1
1.5
2
x1
1


2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
−2
−1.5
−1
−0.5
0
0.5
1
1.5
2
x2
x1
1

Generative Model

```
% Fix the seeds for reproducibility
randn('seed', 11); rand('seed', 1);

% Sensor positions
s = [-1 0; 1 0; 0 1]';
L = size(s, 2); % Number of sensors

M = 1; % Number of targets
x = [0.5;0.2]; % True Target position
R_r = 0.3; %% Range sensor noise variance

m_r = zeros(M, L);
for i=1:M, % For each target
  % Find the true distance from the sensor
  d = repmat(x(:,i), [1 L] ) - s;
  m_r(i, :) = sqrt(d(1,:).^2 + d(2,:).^2);
end;

y_r = sqrt(R_r)*randn(size(m_r)) + m_r;
```

Finding numerically the true posterior

```
% Prepare a uniform grid
xx = -2:0.05:2;
[X1 X2] = meshgrid(xx);

llk = zeros([size(X1) L]);

i = 1; %% Taget idx, assume M = 1;
for j=1:L,
  d = sqrt((X1-s(1,j)).^2 + (X2 - s(2, j)).^2);
  llk(:,:,j) = -0.5*(d-y_r(i, j)).^2/R_r -0.5*log(2*pi*R_r);
end;
lk = sum(llk, 3);

imagesc(xx, xx, exp(lk-max(lk(:))))
```

Metropolis

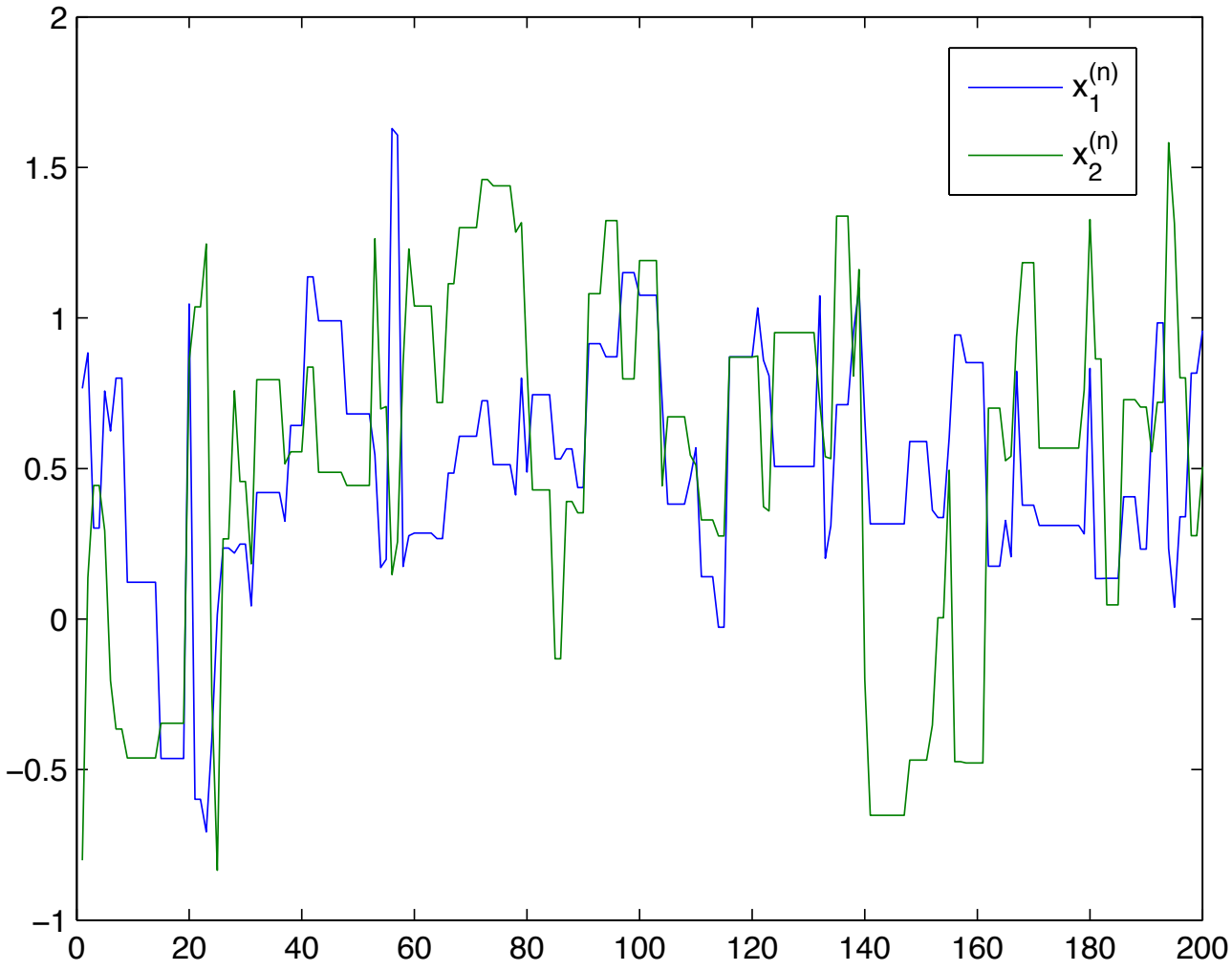
```
% Metropolis
EP = 50200; % Number of epochs
BURN_IN = 200;
R_q = 0.4; % proposal variance
i = 1; % Which target ?
% Storage
chain.x = zeros(2, EP-BURN_IN);
chain.lk = zeros(1, EP-BURN_IN);
```

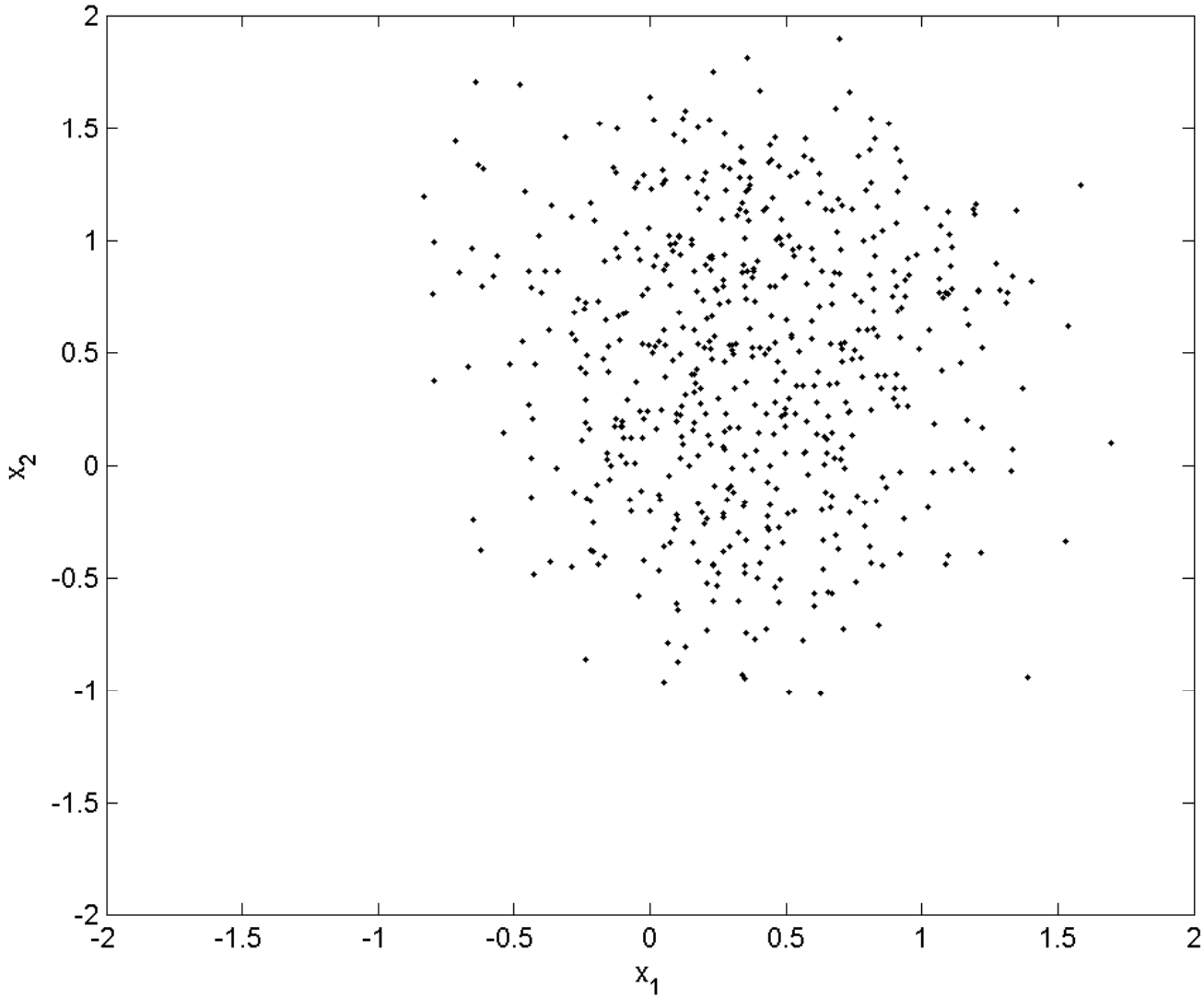
Metropolis (cont.)

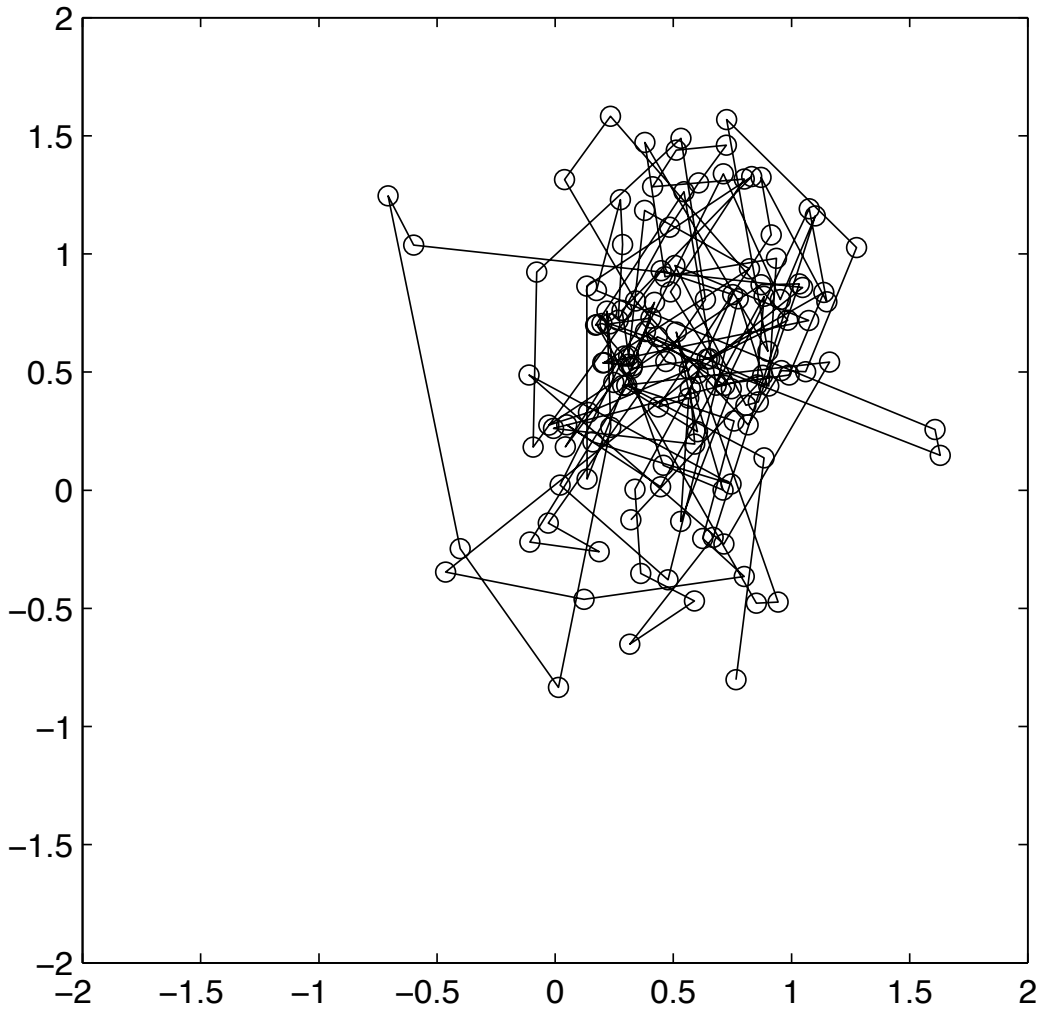
```
x_c = [0; 0]; lk = -Inf;
for e=1:EP,
 % propose a new position
  x_new = x_c + sqrt(R_q)*randn(size(x_c));
 % Evaluate the unnormalised joint posterior
  d = sqrt((x_new(1)-s(1,:)).^2 + (x_new(2) - s(2, :)).^2);
  lk_new = sum(-0.5*(d-y_r(i, :)).^2/R_r -0.5*log(2*pi*R_r));
 % Compute the log-acceptance probability
  log_a = min(0, lk_new - lk);

  if log(rand)<log_a, % Accept,
    x_c = x_new; lk = lk_new;
  end;   % else reject

  if e>BURN_IN,
    chain.x(:, e-BURN_IN) = x_c;
    chain.lk(1, e-BURN_IN) = lk;
  end;
end;
```

Metropolis (cont.)

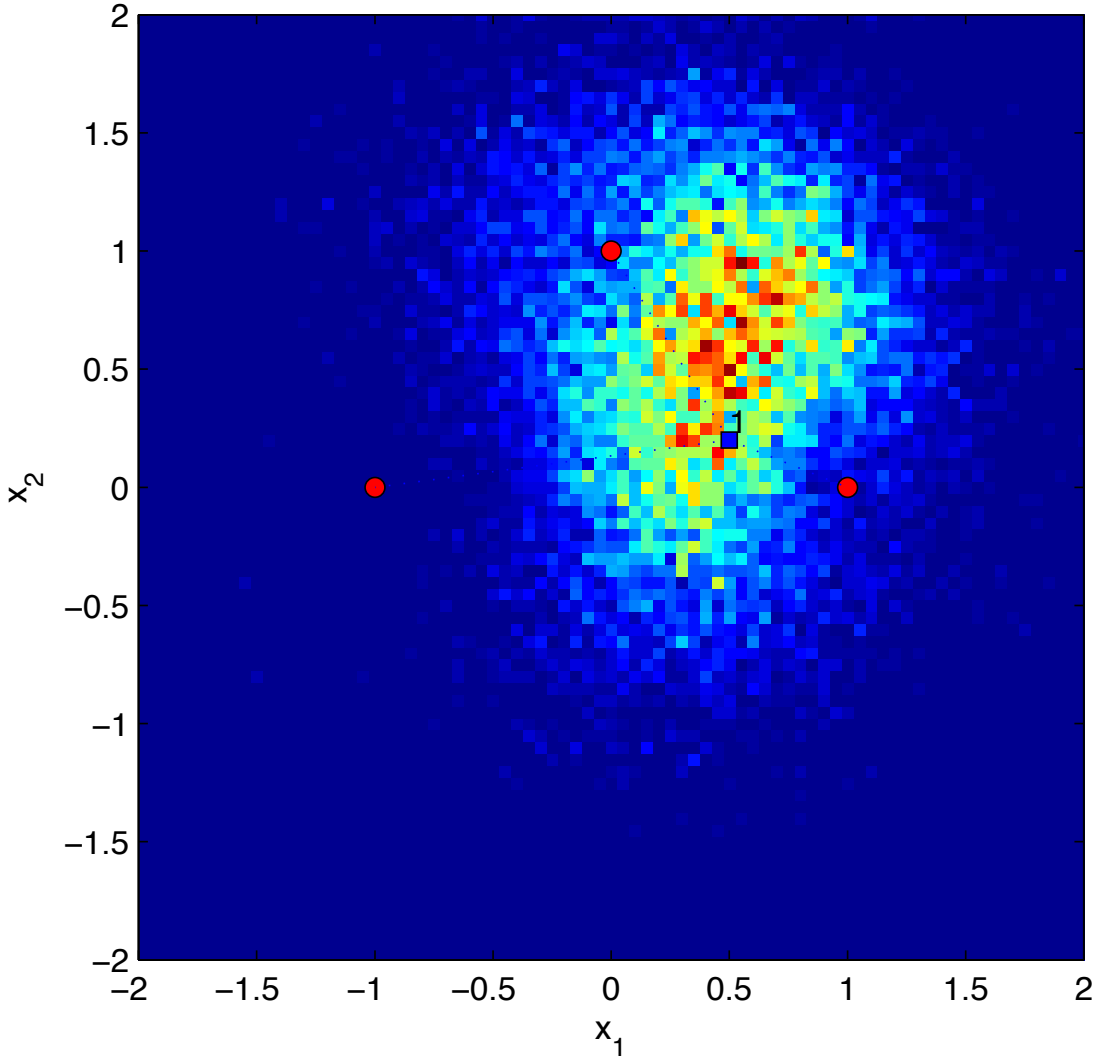
```
[z] = hist3(chain.x', 'ctrs', {xx',xx' });
imagesc(xx, xx, z'); set(gca, 'ydir', 'n')
xlabel('x_1');
ylabel('x_2');
```


2
1.5
1
0.5
0
−0.5
−1
0
20
40
60
80
100
120
140
160
180
200
$x_1^{(n)}$
$x_2^{(n)}$


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
x2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
x1


2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
−2
−1.5
−1
−0.5
0
0.5
1
1.5
2

A 2-D histogram from a longer chain



CMPE 58N - Lecture 5
Monte Carlo methods

Introduction to model selection

Department of Computer Engineering,

Boğaziçi University, Istanbul, Turkey

Instructor: A. Taylan Cemgil

Fall 2009

Example: Poisson change-point model revisited

- We have a sequence of counts x_j.
- Is the count distribution changing or not ?

Generative model

- We model the counts with Poisson random variables x_j, $j = 1 \ldots M$

$$\begin{aligned} \mathcal{PO}(x;\lambda) &= e^{-\lambda}\frac{\lambda^x}{x!} \\ &= \exp\left(+x\log\lambda - \lambda - \log(x!)\right) \end{aligned}$$

- The unknown intensity can be modelled by a Gamma random variable

$$\begin{aligned} \mathcal{G}(\lambda;a,b) &\equiv \frac{1}{\Gamma(a)}b^a\lambda^{a-1}\exp(-b\lambda) \\ &= \exp\left((a-1)\log\lambda - b\lambda - \log\Gamma(a) + a\log b\right) \end{aligned}$$

- The unknown intensity λ_1 is jumping to a new unknown value λ_2 after an unknown index m

Generative model

$$
\begin{aligned}
m &\sim \mathcal{U}\{1 \dots M\} \\
\lambda_i &\sim \mathcal{G}(\lambda_i; a, b) \\
x_j &\sim \begin{cases} \mathcal{PO}(x_j; \lambda_1) & 1 \leq j \leq m \\ \mathcal{PO}(x_j; \lambda_2) & m < j \leq M \end{cases}
\end{aligned}
$$

Goal: Compute $p(m|x_{1:M})$

Full Joint Density

$$
\begin{aligned}
p(x_{1:M}, \lambda_1, \lambda_2, m) &= p(x_{1:M}|\lambda_1, \lambda_2, m)p(\lambda_1)p(\lambda_2)p(m) \\
&= \left(\prod_{j=1}^{m} p(x_j|\lambda_1)\right)\left(\prod_{j=m+1}^{M} p(x_j|\lambda_2)\right) p(\lambda_1)p(\lambda_2)p(m)
\end{aligned}
$$

Integrating out λ_1 and λ_2

$$p(x_{1:M}, m) = \int d\lambda_1 d\lambda_2 p(x_{1:M}|\lambda_1, \lambda_2, m) p(\lambda_1) p(\lambda_2) p(m)$$

Compound Poisson

$$\mathcal{CP}(x;a,b) = \int d\lambda \mathcal{PO}(x;\lambda)\mathcal{G}(\lambda;a,b)$$

$$\begin{aligned}
\mathcal{CP}(x;a,b) &= \int d\lambda \exp((a+x-1)\log\lambda - (b+1)\lambda \\
&\quad - \log(x!) - \log\Gamma(a) + a\log b) \\
&= \int d\lambda \mathcal{G}(\lambda;a+x,b+1)\exp(\log\Gamma(a+x) - (a+x)\log(b+1) \\
&\quad - \log(x!) - \log\Gamma(a) + a\log b) \\
&= \frac{\Gamma(a+x)}{\Gamma(x+1)\Gamma(a)}\frac{b^a}{(b+1)^{(a+x)}}
\end{aligned}$$

Compound Poisson

$$
\begin{aligned}
p(x)p(\lambda|x) &= p(x|\lambda)p(\lambda) \\
\mathcal{CP}(x;a,b)\mathcal{G}(\lambda;a+x,b+1) &= \mathcal{PO}(x;\lambda)\mathcal{G}(\lambda;a,b)
\end{aligned}
$$

Full Joint Distribution

$$
\begin{aligned}
\mathcal{L} &= \log p(x_{1:M}|\lambda_1,\lambda_2,m) + \log p(\lambda_1) + \log p(\lambda_2) + \log p(m) \\
&= \sum_{j=1}^{m} (+x_j \log \lambda_1 - \lambda_1 - \log(x_j!)) \\
&\quad + \sum_{j=m+1}^{M} (+x_j \log \lambda_2 - \lambda_2 - \log(x_j!)) \\
&\quad + (a-1)\log\lambda_1 - b\lambda_1 - \log\Gamma(a) + a\log b \\
&\quad + (a-1)\log\lambda_2 - b\lambda_2 - \log\Gamma(a) + a\log b - \log M
\end{aligned}
$$

Full Joint Distribution

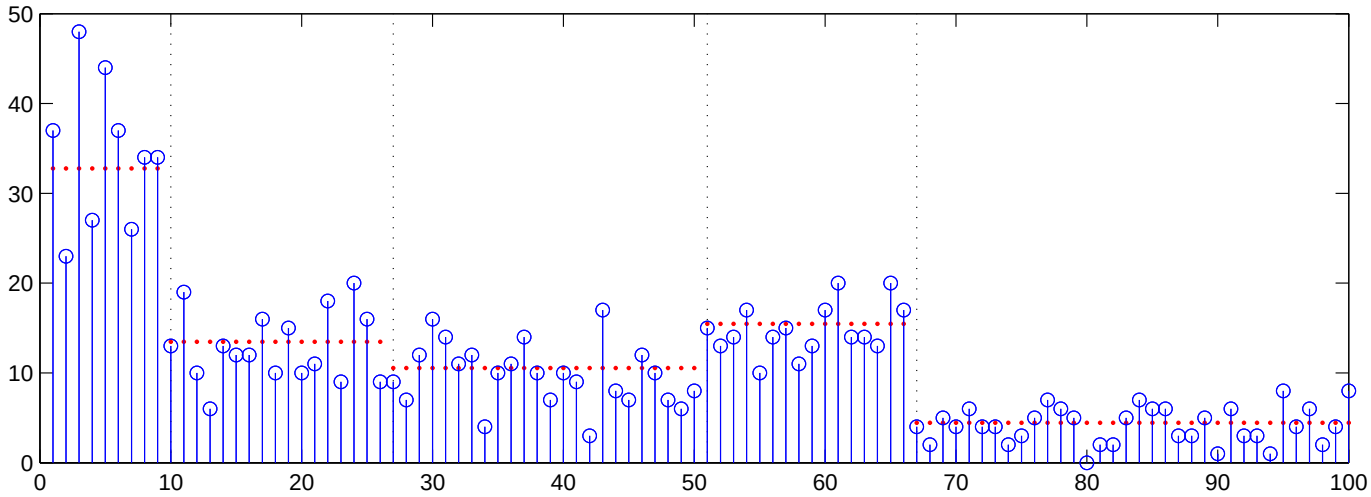
$$
\begin{aligned}
p(x_{1:M}, m) &= \frac{b^a \Gamma(a + \sum_{j=1}^{m} x_j)}{\Gamma(a)(m+b)^{(a+\sum_{j=1}^{m} x_j)} \prod_{j=1}^{m} \Gamma(x_j + 1)} \\
&\quad \times \frac{b^a \Gamma(a + \sum_{j=m+1}^{M} x_j)}{\Gamma(a)(M-m+b)^{(a+\sum_{j=m+1}^{M} x_j)} \prod_{j=m+1}^{M} \Gamma(x_j + 1)} \\
&\propto \frac{\Gamma(a + \sum_{j=1}^{m} x_j)\Gamma(a + \sum_{j=m+1}^{M} x_j)}{(m+b)^{(a+\sum_{j=1}^{m} x_j)}(M-m+b)^{(a+\sum_{j=m+1}^{M} x_j)}}
\end{aligned}
$$

Implementation

```
cx = cumsum(data.x);
cxi = cx(end) - cx;
tlp = gammaln(data.a + cx) +  gammaln(data.a + cxi) ...
 - (data.a + cx).*log((1:data.M) + data.b) ...
 - (data.a + cxi).*log(data.M - (1:data.M) + data.b);

bar(normalize_exp(tlp, 2));
set(gca, 'xlim', [0 data.M+1])
xlabel('m')
```

$$
\begin{aligned}
p(x_{1:M}, m=0) &= \frac{b^a \Gamma(a)}{\Gamma(a) b^a} \\
&\quad \times \frac{b^a \Gamma\left(a + \sum_{j=1}^{M} x_j\right)}{\Gamma(a)(M+b)^{\left(a+\sum_{j=1}^{M} x_j\right)} \prod_{j=1}^{M} \Gamma(x_j+1)}
\end{aligned}
$$

Multiple Change Points



A Multiple Change Point model

$$
\begin{aligned}
c_1 &= 1 \\
j &\geq 2 \\
c_j &= c_{j-1} + \mathcal{B}(r_j; w) \\
\lambda_c &\sim \mathcal{G}(\lambda_c; a, b) \\
x_j | \lambda_{c_j} &\sim \mathcal{PO}(x_j; \lambda_{c_j})
\end{aligned}
$$

Multiple Change Points – Indicator representation

$$
\begin{aligned}
\lambda_0 &\sim \mathcal{G}(\lambda_0; a, b) \\
r_j &\sim \mathcal{B}(r_j; w) \\
\lambda_j | r_j, \lambda_{j-1} &\sim [r_j = 0]\, \delta(\lambda_j - \lambda_{j-1}) + [r_j = 1]\, \mathcal{G}(\lambda_j; a, b) \\
x_j | \lambda_j &\sim \mathcal{PO}(x_j; \lambda_j)
\end{aligned}
$$

Observations

- This change point model is actually a hidden Markov model with the latent chain defined on joint variables (r_j, λ_j)
- Hence, we could in principle run the forward backward algorithm to find the posterior marginals
- Take home messages:
  - Complicated looking models can be entirely tractable (simple looking models can be entirely intractable)
  - Equivalent models can have substantially different looking specifications
  - One specification can be easier to work with than others

Assignment 3

1. Derive and Implement a MH algorithm for the single changepoint model to sample from $p(m|x_{1:M})$
2. Derive and Implement a Gibbs sampler for the multiple change point problem to sample from $p(r_{1:M}|x_{1:M})$
3. (Optional) Derive and Implement an exact algorithm for the multiple change point problem to compute $p(r_j|x_{1:M})$ for $j = 1 \ldots M$.
4. (Reading) pp122-124 from Liu
5. (Reading) Section 6 pp63-66 from Bristol notes

CMPE 58N - Lecture 6. Monte Carlo methods

Introduction to Model selection

Department of Computer Engineering,

Boğaziçi University, Istanbul, Turkey

Instructor: A. Taylan Cemgil

Spring 2009

Example: Poisson change-point model revisited

- We have a sequence of counts x_j.
- Is the count distribution changing or not ?

Generative model

- We model the counts with Poisson random variables x_j, $j = 1 \ldots M$

$$\begin{aligned} \mathcal{PO}(x;\lambda) &= e^{-\lambda}\frac{\lambda^x}{x!} \\ &= \exp\left(+x\log\lambda - \lambda - \log(x!)\right) \end{aligned}$$

- The unknown intensity can be modelled by a Gamma random variable

$$\begin{aligned} \mathcal{G}(\lambda;a,b) &\equiv \frac{1}{\Gamma(a)}b^a\lambda^{a-1}\exp(-b\lambda) \\ &= \exp\left((a-1)\log\lambda - b\lambda - \log\Gamma(a) + a\log b\right) \end{aligned}$$

- The unknown intensity λ_1 is jumping to a new unknown value λ_2 after an unknown index m

Generative model

$$
\begin{aligned}
m &\sim \mathcal{U}\{1 \ldots M\} \\
\lambda_i &\sim \mathcal{G}(\lambda_i; a, b) \\
x_j &\sim \begin{cases} \mathcal{PO}(x_j; \lambda_1) & 1 \leq j \leq m \\ \mathcal{PO}(x_j; \lambda_2) & m < j \leq M \end{cases}
\end{aligned}
$$

Goal: Compute $p(m|x_{1:M})$

Full Joint Density

$$
\begin{aligned}
p(x_{1:M}, \lambda_1, \lambda_2, m) &= p(x_{1:M}|\lambda_1, \lambda_2, m)p(\lambda_1)p(\lambda_2)p(m) \\
&= \left(\prod_{j=1}^{m} p(x_j|\lambda_1)\right)\left(\prod_{j=m+1}^{M} p(x_j|\lambda_2)\right) p(\lambda_1)p(\lambda_2)p(m)
\end{aligned}
$$

Integrating out λ_1 and λ_2

$$p(x_{1:M}, m) = \int d\lambda_1 d\lambda_2 p(x_{1:M}|\lambda_1, \lambda_2, m) p(\lambda_1) p(\lambda_2) p(m)$$

Compound Poisson

$$\mathcal{CP}(x;a,b) = \int d\lambda \mathcal{PO}(x;\lambda)\mathcal{G}(\lambda;a,b)$$

$$\begin{aligned}
\mathcal{CP}(x;a,b) &= \int d\lambda \exp((a+x-1)\log\lambda - (b+1)\lambda \\
&\quad - \log(x!) - \log\Gamma(a) + a\log b) \\
&= \int d\lambda \mathcal{G}(\lambda;a+x,b+1)\exp(\log\Gamma(a+x) - (a+x)\log(b+1) \\
&\quad - \log(x!) - \log\Gamma(a) + a\log b) \\
&= \frac{b^a\Gamma(a+x)}{(b+1)^{(a+x)}\Gamma(x+1)\Gamma(a)}
\end{aligned}$$

Full Joint Distribution

$$
\begin{aligned}
\mathcal{L} &= \log p(x_{1:M}|\lambda_1,\lambda_2,m) + \log p(\lambda_1) + \log p(\lambda_2) + \log p(m) \\
&= \sum_{j=1}^{m} (+x_j \log \lambda_1 - \lambda_1 - \log(x_j!)) \\
&\quad + \sum_{j=m+1}^{M} (+x_j \log \lambda_2 - \lambda_2 - \log(x_j!)) \\
&\quad +(a-1)\log\lambda_1 - b\lambda_1 - \log\Gamma(a) + a\log b \\
&\quad +(a-1)\log\lambda_2 - b\lambda_2 - \log\Gamma(a) + a\log b - \log M
\end{aligned}
$$

Full Joint Distribution

$$
\begin{aligned}
p(x_{1:M}, m) &= \frac{b^a \Gamma(a + \sum_{j=1}^{m} x_j)}{\Gamma(a)(m+b)^{(a+\sum_{j=1}^{m} x_j)} \prod_{j=1}^{m} \Gamma(x_j + 1)} \\
&\times \frac{b^a \Gamma(a + \sum_{j=m+1}^{M} x_j)}{\Gamma(a)(M-m+b)^{(a+\sum_{j=m+1}^{M} x_j)} \prod_{j=m+1}^{M} \Gamma(x_j + 1)} \\
&\propto \frac{\Gamma(a + \sum_{j=1}^{m} x_j)\Gamma(a + \sum_{j=m+1}^{M} x_j)}{(m+b)^{(a+\sum_{j=1}^{m} x_j)}(M-m+b)^{(a+\sum_{j=m+1}^{M} x_j)}}
\end{aligned}
$$

Implementation

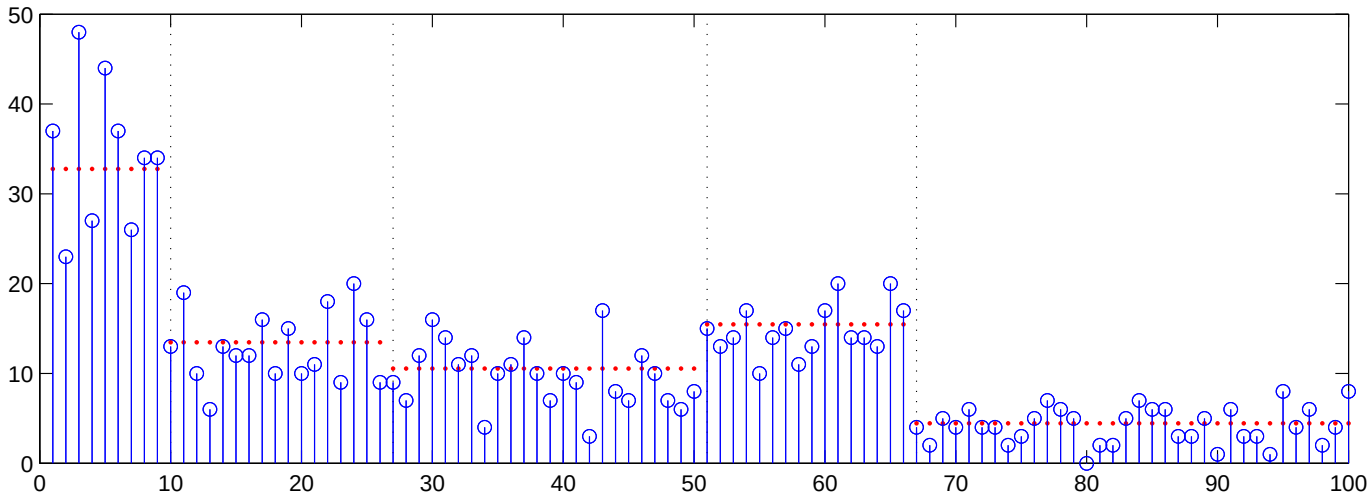
```
cx = cumsum(data.x);
cxi = cx(end) - cx;
tlp = gammaln(data.a + cx) +  gammaln(data.a + cxi) ...
 - (data.a + cx).*log((1:data.M) + data.b) ...
 - (data.a + cxi).*log(data.M - (1:data.M) + data.b);

bar(normalize_exp(tlp, 2));
set(gca, 'xlim', [0 data.M+1])
xlabel('m')
```

$$
\begin{aligned}
p(x_{1:M}, m=0) &= \frac{b^a \Gamma(a)}{\Gamma(a) b^a} \\
&\quad \times \frac{b^a \Gamma(a + \sum_{j=1}^{M} x_j)}{\Gamma(a)(M+b)^{(a+\sum_{j=1}^{M} x_j)} \prod_{j=1}^{M} \Gamma(x_j+1)}
\end{aligned}
$$

Multiple Change Points – Indicator representation

$$
\begin{aligned}
\lambda_0 &\sim \mathcal{G}(\lambda_0; a, b) \\
r_j &\sim \mathcal{B}(r_j; w) \\
\lambda_j | r_j, \lambda_{j-1} &\sim [r_j = 0]\, \delta(\lambda_j - \lambda_{j-1}) + [r_j = 1]\, \mathcal{G}(\lambda_j; a, b) \\
x_j | \lambda_j &\sim \mathcal{PO}(x_j; \lambda_j)
\end{aligned}
$$

Multiple Change Points



Multiple Change Points – Alternative model

$$
\begin{aligned}
c_1 &= 1 \\
j &\geq 2 \\
c_j &= c_{j-1} + \mathcal{B}(r_j; w) \\
\lambda_c &\sim \mathcal{G}(\lambda_c; a, b) \\
x_j | \lambda_{c_j} &\sim \mathcal{PO}(x_j; \lambda_{c_j})
\end{aligned}
$$

Assignment 3

1. Derive and Implement a MH algorithm for the single changepoint model to sample from $p(m|x_{1:M})$
2. Derive and Implement a Gibbs sampler for the multiple change point problem to sample from $p(r_{1:M}|x_{1:M})$
3. (Optional) Derive and Implement an exact algorithm for the multiple change point problem to compute $p(r_j|x_{1:M})$ for $j = 1 \ldots M$.
4. (Reading) pp122-124 from Liu
5. (Reading) Section 6 pp63-66 from Bristol notes

CMPE 58N - Lecture 7.
Monte Carlo methods

Reversible Jump MCMC

Department of Computer Engineering,

Boğaziçi University, Istanbul, Turkey

Instructor: A. Taylan Cemgil

Spring 2009

Model Selection

- Number of things you don't know is one of the things you don't know

“Model Selection” Example

Given an unknown number of fair dice with outcomes
$\lambda_1, \lambda_2, \ldots, \lambda_n$,

$$\mathcal{D} = \sum_{i=1}^{n} \lambda_i$$

How many dice are there when $\mathcal{D} = 9$?
Assume that any number n is equally likely *a-priori*

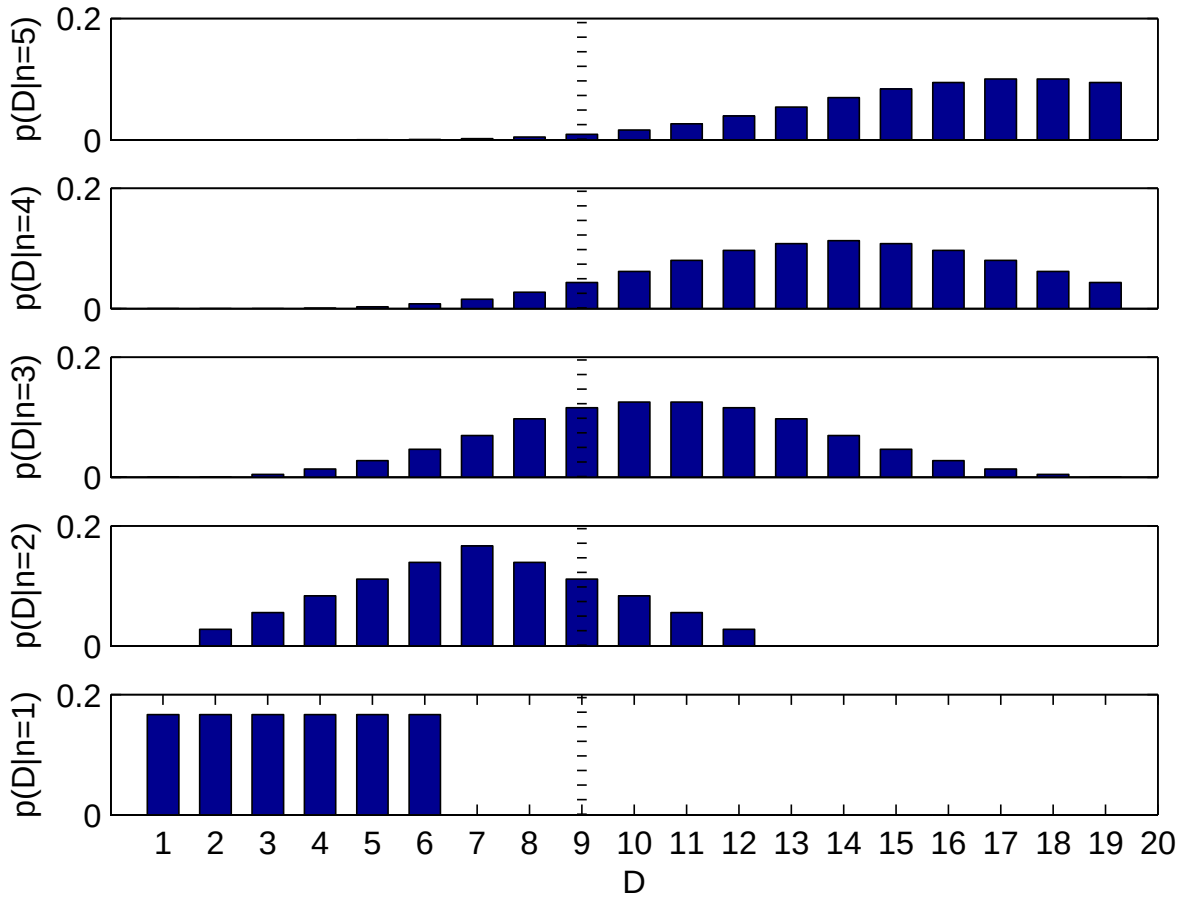
“Model Selection” Example

Given all n are equally likely (i.e., $p(n)$ is flat), we calculate (formally)

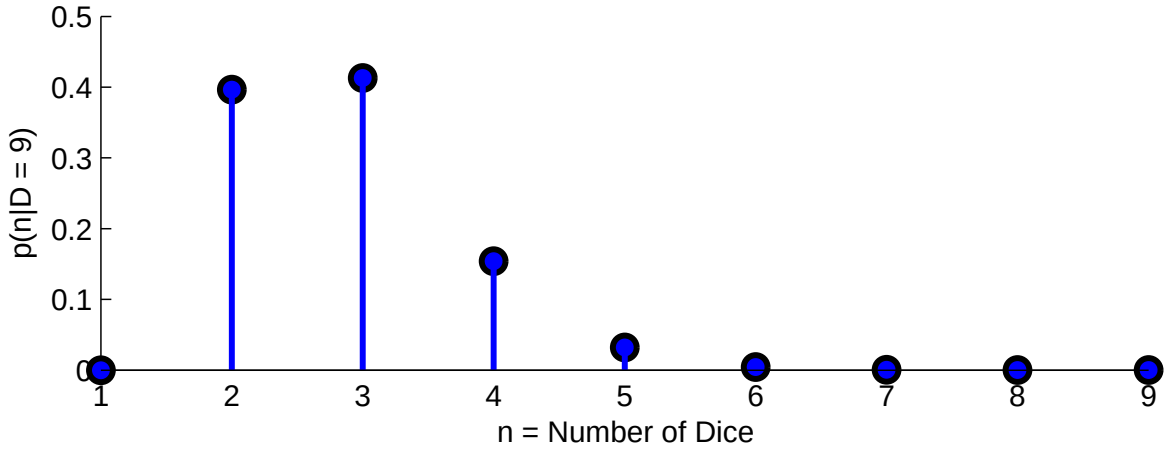
$$p(n|\mathcal{D}=9) = \frac{p(\mathcal{D}=9|n)p(n)}{p(\mathcal{D})} \propto p(\mathcal{D}=9|n)$$

$$\begin{aligned}
p(\mathcal{D}|n=1) &= \sum_{\lambda_1} p(\mathcal{D}|\lambda_1)p(\lambda_1) \\
p(\mathcal{D}|n=2) &= \sum_{\lambda_1}\sum_{\lambda_2} p(\mathcal{D}|\lambda_1,\lambda_2)p(\lambda_1)p(\lambda_2) \\
&\cdots \\
p(\mathcal{D}|n=n') &= \sum_{\lambda_1,\ldots,\lambda_{n'}} p(\mathcal{D}|\lambda_1,\ldots,\lambda_{n'}) \prod_{i=1}^{n'} p(\lambda_i)
\end{aligned}$$

$$p(\mathcal{D}|n) = \sum_{\boldsymbol{\lambda}} p(\mathcal{D}|\boldsymbol{\lambda}, n) p(\boldsymbol{\lambda}|n)$$

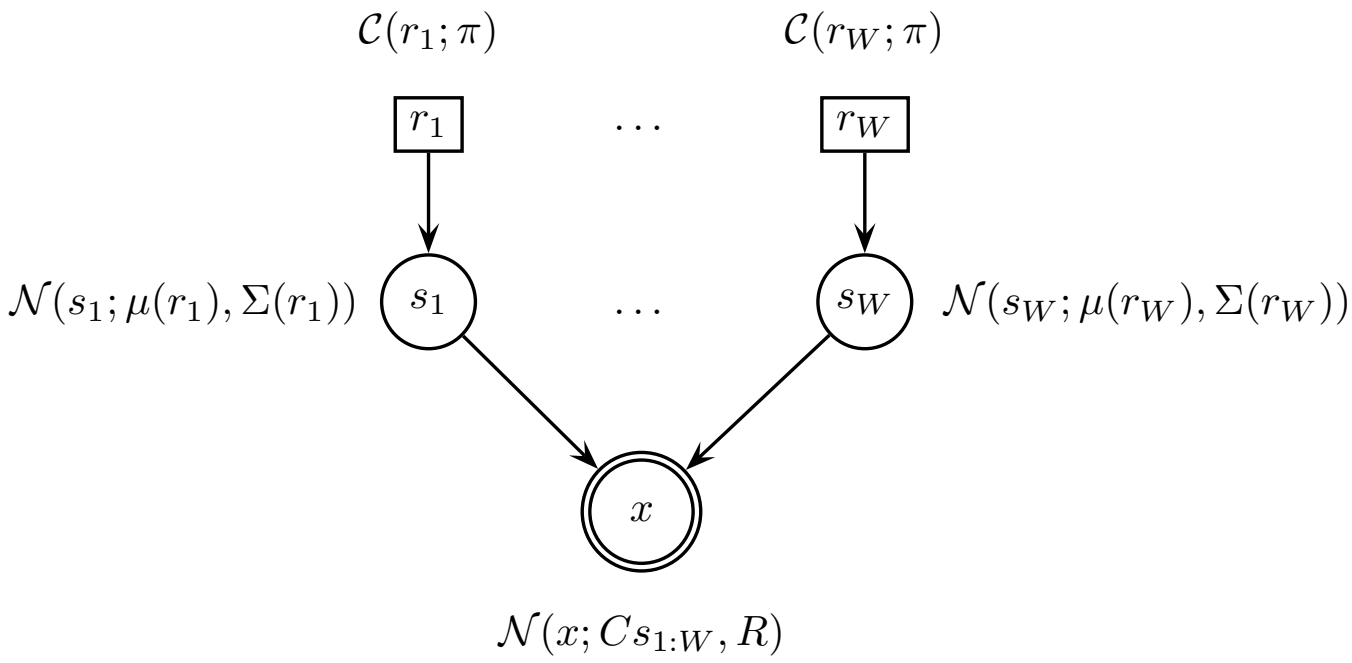


“Model Selection” Example



- Complex models are more flexible but they spread their probability mass
- Bayesian inference inherently prefers “simpler models” – Occam’s razor
- Computational burden: We need to sum over all parameters λ

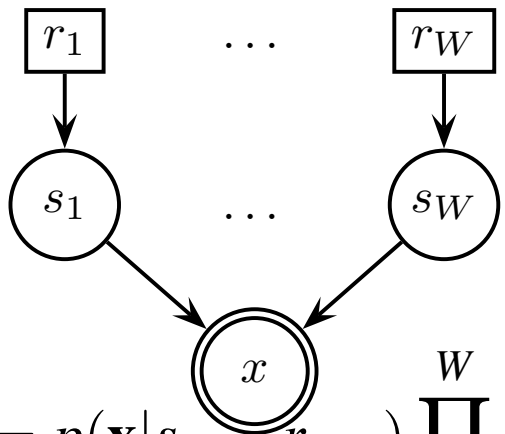
Bayesian Variable Selection



- Generalized Linear Model – Column's of C are the basis vectors
- The exact posterior is a mixture of 2^W Gaussians
- When W is large, computation of posterior features becomes intractable.

Generative model

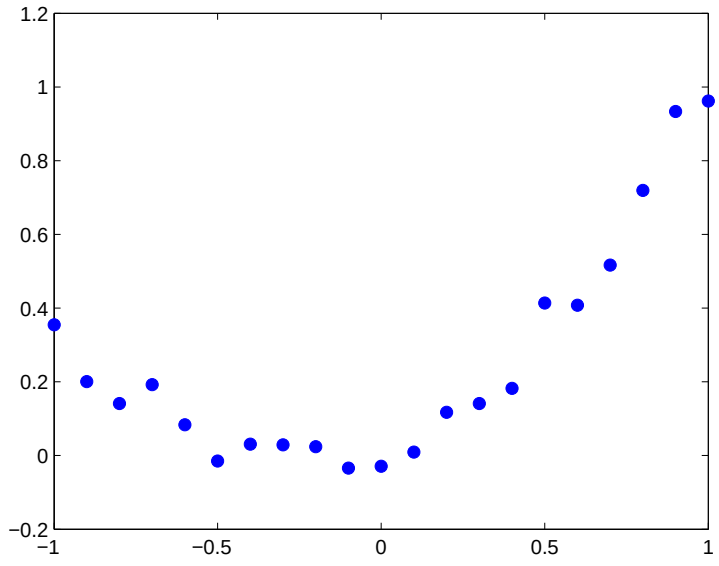
$$
\begin{aligned}
r_i &\sim \mathcal{C}(r_i;\pi) \\
s_i|r_i &\sim \mathcal{N}(s_i;\mu(r_i),\Sigma(r_i)) \\
\mathbf{x}|s_{1:W} &\sim \mathcal{N}(\mathbf{x};Cs_{1:W},R) \\
C &\equiv \begin{bmatrix} C_1 & \dots & C_i & \dots & C_W \end{bmatrix}
\end{aligned}
$$



$$
p(\mathbf{x}, s_{1:W}, r_{1:W}) = p(\mathbf{x}|s_{1:W}, r_{1:W}) \prod_{i=1}^{W} p(s_i|r_i)p(r_i)
$$

Example 1: Variable selection in Polynomial Regression

Given $\{t_j, x(t_j)\}_{j=1...J}$, what is the order N of the polynomial?



$$x(t) = \sum_{i=0}^{N} s_{i+1} t^i + \epsilon(t)$$

Ex1: Regression

$$\mathbf{t} = \begin{pmatrix} t_1 & t_2 & \dots & t_J \end{pmatrix}^\top$$
$$C \equiv \begin{pmatrix} \mathbf{t}^0 & \mathbf{t}^1 & \dots & \mathbf{t}^{W-1} \end{pmatrix}$$

```
>> C = fliplr(vander(0:4))  % Van der Monde matrix
     1     0     0     0     0
     1     1     1     1     1
     1     2     4     8    16
     1     3     9    27    81
     1     4    16    64   256
```

$$r_i \sim \mathcal{C}(r_i; 0.5, 0.5) \qquad r_i \in \{\text{on}, \text{off}\}$$
$$s_i | r_i \sim \mathcal{N}(s_i; 0, \Sigma(r_i))$$
$$\mathbf{x} | s_{1:W} \sim \mathcal{N}(\mathbf{x}; C s_{1:W}, R)$$

$$\Sigma(r_i = \text{on}) \gg \Sigma(r_i = \text{off})$$

Ex1: Regression

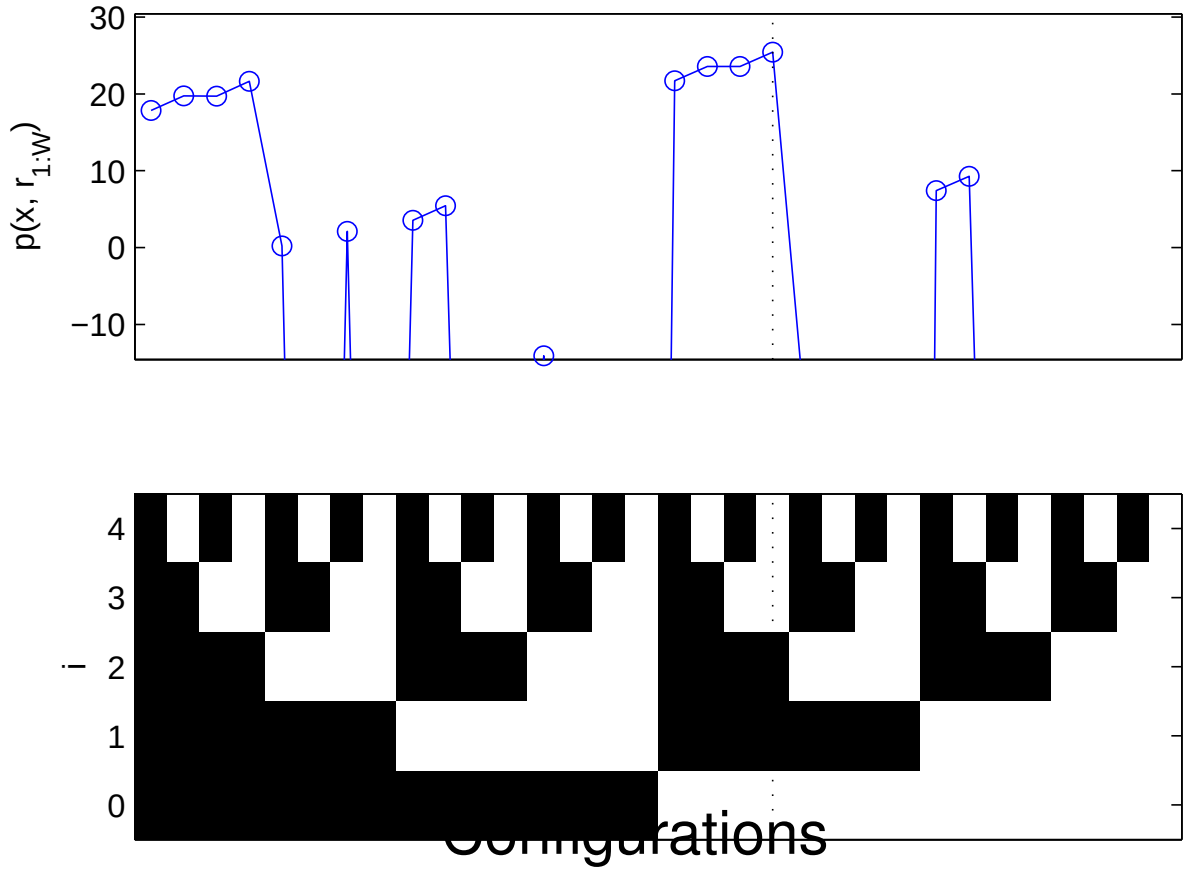
To find the "active" basis functions we need to calculate

$$r^*_{1:W} \equiv \underset{r_{1:W}}{\operatorname{argmax}}\, p(r_{1:W}|\mathbf{x}) \quad = \quad \underset{r_{1:W}}{\operatorname{argmax}} \int ds_{1:W} p(\mathbf{x}|s_{1:W}) p(s_{1:W}|r_{1:W}) p(r_{1:W})$$

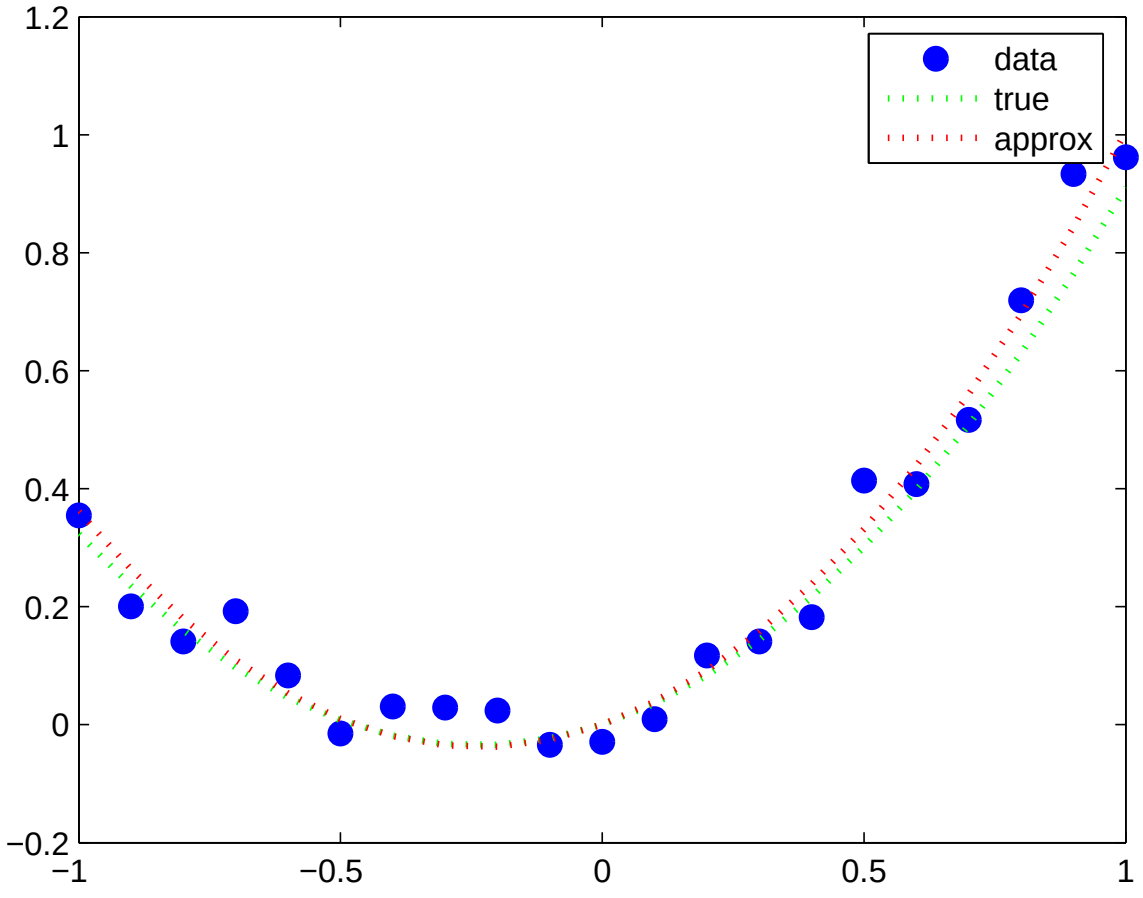
Then, the reconstruction is given by

$$\begin{aligned} \hat{x}(t) &= \left\langle \sum_{i=0}^{W-1} s_{i+1} t^i \right\rangle_{p(s_{1:W}|\mathbf{x}, r^*_{1:W})} \\ &= \sum_{i=0}^{W-1} \left\langle s_{i+1} \right\rangle_{p(s_{i+1}|\mathbf{x}, r^*_{1:W})} t^i \end{aligned}$$

Ex1: Regression



Ex1: Regression



Model Selection

- Number of things you don't know is one of the things you don't know
- k: Number of parameters
- $\theta_k \in \mathbb{R}^{n_k}$, n_k is model dimension

Model Selection via Reversible Jump

- "State Space"

$$(k, \theta_k) \in \bigcup_{k \in \mathcal{K}} \{\{k\} \times \mathbb{R}^{n_k}\}$$

- Reversible Jump $=$ Metropolis-Hastings on this general and "non-standard" state space
- We use Metropolis-Hastings to build a suitable reversible chain.
-

Metropolis-Hastings

- The target distribution is $\pi(s)$.
- We choose a suitable proposal distribution $q(s'|s)$
- We define the *acceptance probability* of a jump from s to s' as

$$a(s \rightarrow s') \equiv \min\{1, \frac{\pi(s')q(s|s')}{\pi(s)q(s'|s)}\}$$

- We have verified the detailed balance for the MH transition Kernel T

Metropolis-Hastings Transition Kernel

- Transition Kernel

$$T(s'|s) = \underbrace{q(s'|s)a(s \to s')}_{\text{Accept}} + \underbrace{\delta(s'-s)\int ds' q(s'|s)(1-a(s \to s'))}_{\text{Reject}}$$

- We know that T satisfies detailed balance, i.e.,

$$T(s|s')\pi(s') = T(s'|s)\pi(s)$$

Implications of Detailed Balance

- Consider now the probability that the chain is in $s \in S$ and jumps to $s' \in S'$

$$\Pr\{S \to S'\} \equiv \int_{(s,s') \in S \times S'} ds ds' T(s'|s) \pi(s)$$

- Detailed balance says simply that this probability is equal to the probability that the chain is in $s \in S$ and jumps to $s' \in S'$

$$\Pr\{S' \to S\} \equiv \int_{(s,s') \in S \times S'} ds ds' T(s|s') \pi(s')$$

- (Technicality: S and S' need to be "measurable")

Implications of Detailed Balance

- On $S \bigcap S'$, i.e., when the chain is not moving with $s' = s$, we have

$$\int_{(s,s') \in S \times S'} ds ds' \delta(s' - s) \int ds' q(s'|s)(1 - a(s \to s')) =$$
$$\int_{(s,s') \in S \times S'} ds ds' \delta(s' - s) \int ds q(s|s')(1 - a(s' \to s))$$

- This implies that ...

Equilibrium Probability of a Jump

$$\int_{(s,s')\in S\times S'} dsds' q(s'|s)\pi(s)a(s \to s') =$$
$$\int_{(s,s')\in S\times S'} dsds' q(s|s')\pi(s')a(s' \to s)$$

Metropolis-Hastings

- Consider how we implement MH in an abstract sense
  - We draw some random numbers $u \sim g(u)$
  - The new state is a deterministic function of the current state s and u, i.e.,

$$s' = h(s, u)$$

  - The reverse jump would be given by new random numbers $u' \sim g'(u')$

$$s = h'(s', u')$$

Equilibrium Probability of a Jump

- In other words, we implement the proposal as

$$\begin{aligned} q(s'|s) &= \int du \delta(s' - h(s,u)) g(u) \\ &= \int_{(s,s'=h(s,u))} du g(u) \end{aligned}$$

where $g(u)$ is the joint density of u.

Equilibrium Probability of a Jump

- The probability jumping from S to S' is

$$\Pr\{S \to S'\} = \int_{(s,s'=h(s,u))\in S\times S'} dsdug(u)\pi(s)a(s \to s')$$

Equilibrium Probability of a Jump

- The probability of jumping from S' to S is

$$\Pr\{S' \to S\} = \int_{(s=h'(s',u'),s') \in S \times S'} ds' du' g(u') \pi(s') a(s' \to s)$$

- By detailed balance, we have

$$\Pr\{S \to S'\} = \Pr\{S' \to S\}$$

- Detailed balance holds iff ...

Transformation $(s,u) \rightarrow (s',u')$

$$g(u)\pi(s)a(s \rightarrow s') = g(u')\pi(s')a(s' \rightarrow s)\left|\frac{\partial(s',u')}{\partial(s,u)}\right|$$

$$a(s \rightarrow s') = \min\left\{1, \frac{g(u')\pi(s')}{g(u)\pi(s)}\left|\frac{\partial(s',u')}{\partial(s,u)}\right|\right\}$$

- Provided that $s'(s,u)$ and $u'(s,u)$ (and their inverses) are differentiable, so that the Jacobian exists

Dimension Matching

- We need

$$\begin{pmatrix} s \\ u \end{pmatrix} \in \mathbb{R}^{n+m}$$

$$\begin{pmatrix} s' \\ u' \end{pmatrix} \in \mathbb{R}^{n'+m'}$$

$$n+m = n'+m'$$

- Need this to be able to define the Jacobian
- Necessary but not sufficient

Toy Example (from Green, 2002)

- The state space of the chain

$$s \in \mathbb{R}^1 \cup \mathbb{R}^2$$

- Probabilities

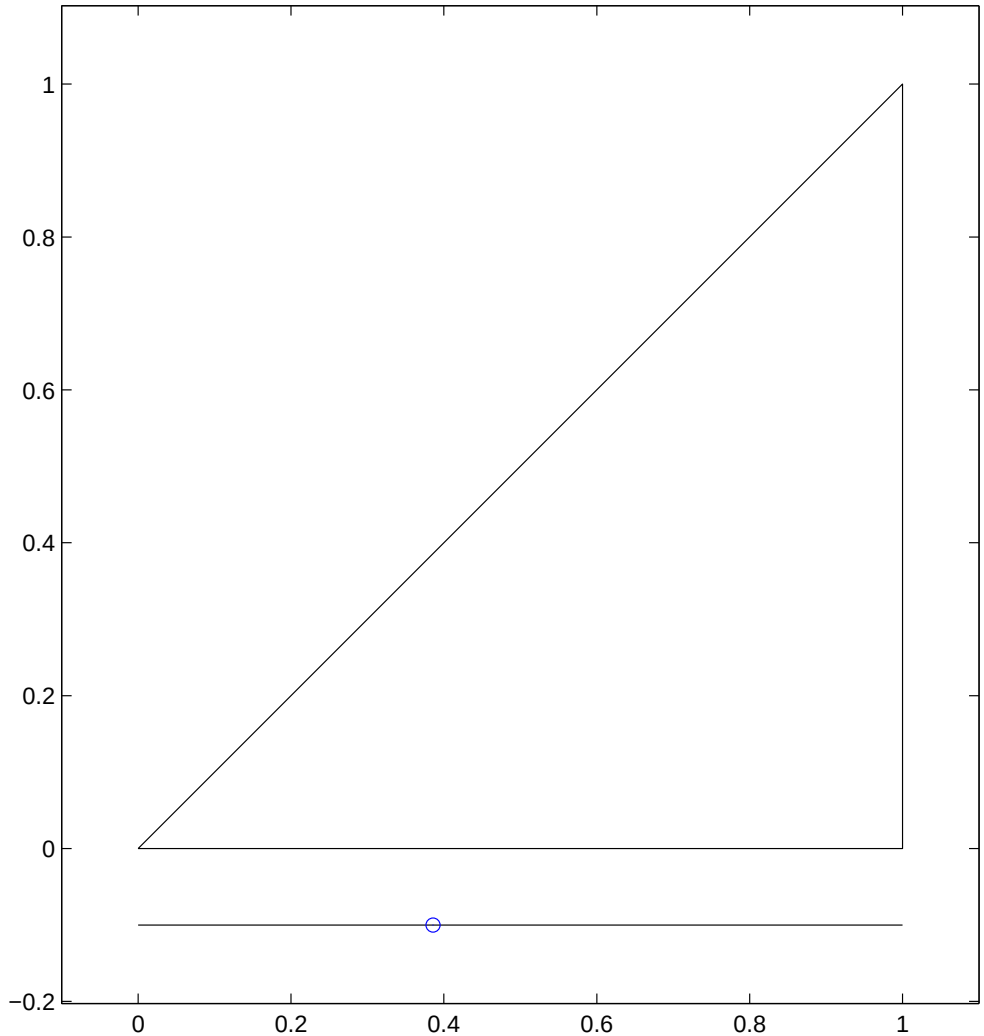
$$\begin{aligned} \Pr\{s \in \mathbb{R}\} &= p_1 \\ \Pr\{s \in \mathbb{R}^2\} &= p_2 \end{aligned}$$

- If $s \in \mathbb{R}$

$$s \sim \mathcal{U}([0,1])$$

- If $s = (s_1, s_2) \in \mathbb{R}^2$

$$s \sim \mathcal{U}(\{s : 0 < s_2 < s_1 < 1\})$$


1
0.8
0.6
0.4
0.2
0
−0.2
0
0.2
0.4
0.6
0.8
1

Possible Moves

$(1 \leftrightarrow 1)$ In $\mathbb{R}$

$$s' \sim \mathcal{U}(s - \epsilon, s + \epsilon)$$

$(2 \leftrightarrow 2)$ In $\mathbb{R}^2$

$$(s_1, s_2)' \leftarrow (1 - s_2, 1 - s_1)$$

$(1 \leftrightarrow 2)$ Transdimensional move:

$$\mathbb{R} \rightarrow \mathbb{R}^2$$

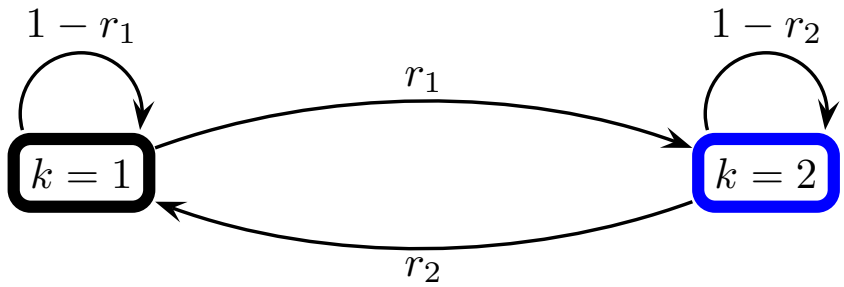
$$u \sim \mathcal{U}([0, 1])$$

$$(s_1', s_2') \leftarrow (s, u)$$

$$\mathbb{R}^2 \rightarrow \mathbb{R}$$

$$s \leftarrow s_1'$$

Moves



Designing the MH Kernel

- For each move $m = (k \to k')$, design a proposal distribution

$$q_m(s'|s) \equiv q(s'|s,m)q(m|s)$$

  - $j_m(s) \equiv q(m|s)$ is the probability taking the jump m when at s
  - Regular moves $(k \to k)$, no dimensionality change
  - Transdimensional moves $(k \to k') \;\; k \neq k'$, need dimension matching
- Calculate the acceptance probability α_m
  - Each move must be reversible!
- The accept part of the transition Kernel is

$$\sum_m \alpha_m(s \to s')q_m(s'|s)$$

Toy Example, Regular moves

- $\mathbb{R} \to \mathbb{R}$ with $j_{1\to 1} = 1 - r_1$

$$s' \sim \mathcal{U}(s-\epsilon, s+\epsilon) = q(s'|s, m = (1 \to 1))$$

- Acceptance probability

$$\begin{aligned} \alpha_{1\to 1}(s \to s') &= \min\left\{1, \frac{q_{1\to 1}(s|s')\pi(s')}{q_{1\to 1}(s'|s)\pi(s)}\right\} \\ &= \min\left\{1, \frac{[0 < s' < 1]}{1}\right\} \end{aligned}$$

Toy Example, Regular moves

- $\mathbb{R}^2 \to \mathbb{R}^2$ with $j_{2\to 2} = 1 - r_2$

$$(s_1', s_2') \sim \delta(s_1' - (1 - s_2))\delta(s_2' - (1 - s_1)) = q(s'|s, m = (2 \to 2))$$

- Acceptance probability

$$\begin{aligned} \alpha_{2\to 2}(s \to s') &= \min\left\{1, \frac{q_{2\to 2}(s|s')\pi(s')}{q_{2\to 2}(s'|s)\pi(s)}\right\} \\ &= \min\{1, 1\} = 1 \end{aligned}$$

Toy Example, Transdimensional moves

- $\mathbb{R} \rightarrow \mathbb{R}^2$ $j_{1\rightarrow 2} = r_1$

$$\begin{aligned} s_1' &= s_1 \\ s_2' &= u \end{aligned}$$

- $\mathbb{R}^2 \rightarrow \mathbb{R}$ $j_{2\rightarrow 1} = r_2$

$$s_1 = s_1'$$

- The Jacobian

$$\left| \frac{\partial(s_1', s_2')}{\partial(s_1, u)} \right| = \begin{vmatrix} \partial s_1'/\partial s_1 & \partial s_1'/\partial u \\ \partial s_2'/\partial s_1 & \partial s_2'/\partial u \end{vmatrix} = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} = 1$$

Toy Example, Transdimensional moves

- Proposals

$$\begin{aligned} g_{1\to 2}(u) &= \mathcal{U}(0,1) \\ g_{2\to 1}(u) &= 1 \end{aligned}$$

- Remember that we implement $q(s'|s,m)$ using $g_m(u)$
- We denote the reverse move of m by $-m$
- The proposal corresponding to the reverse move of m is denoted as g_{-m} so for example

$$g_{-(1\to 2)}(u') = g_{2\to 1}(u')$$

Acceptance probabilities

$$\alpha_m(s \to s') = \min\left\{1, \frac{g_{-m}(u')}{g_m(u)} \frac{j_{-m}(s')}{j_m(s)} \frac{\pi(s')}{\pi(s)} \left|\frac{\partial(s', u')}{\partial(s, u)}\right|\right\}$$

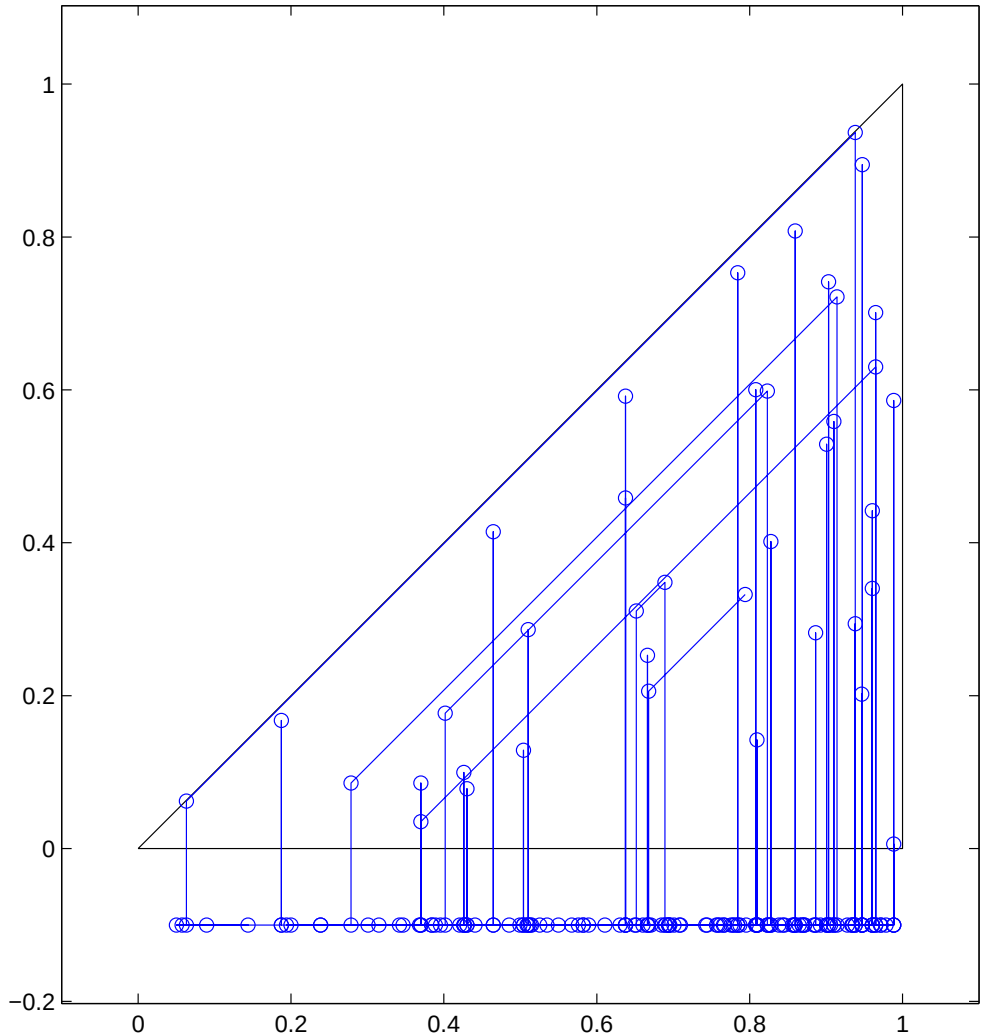
- Acceptance probability for $m = (1 \to 2)$

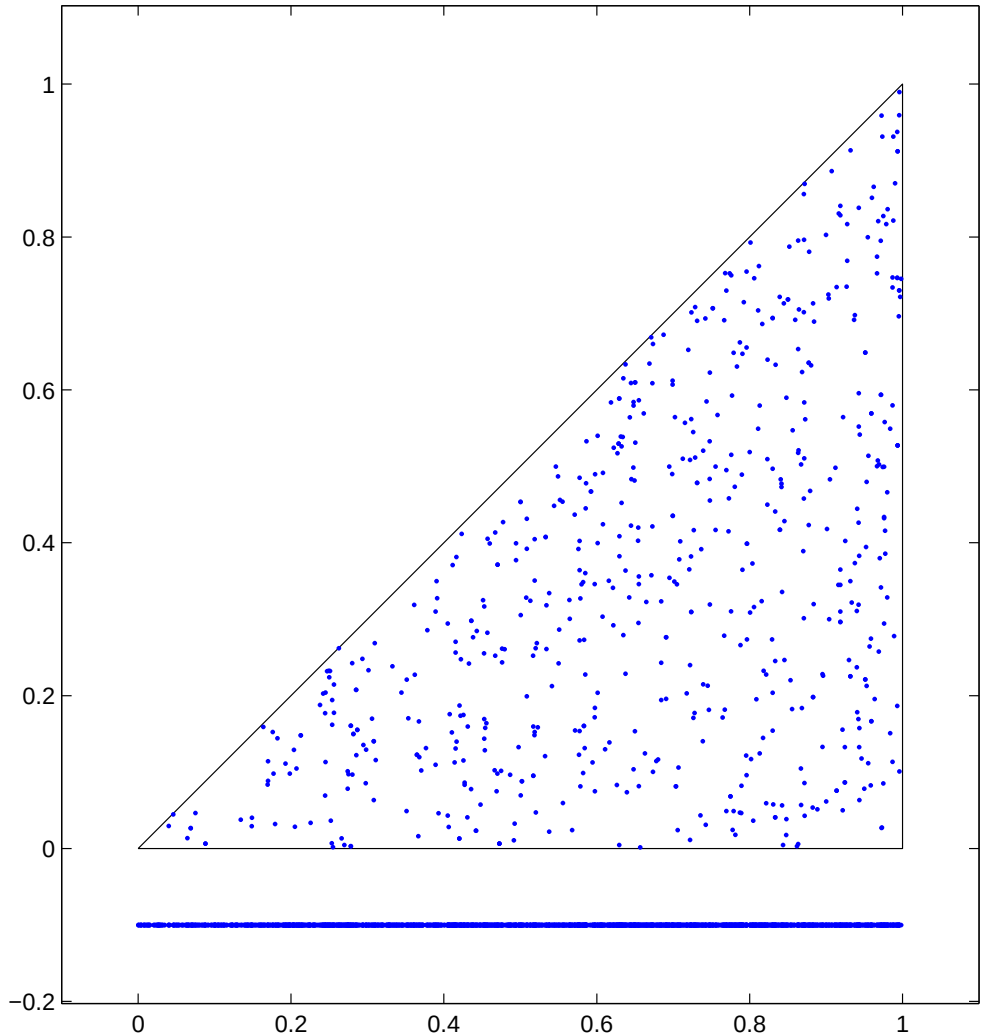
$$\begin{aligned}\alpha_{1\to 2}(s \to s') &= \min\left\{1, \frac{1}{1}\frac{r_2}{r_1}\frac{2\,[s_2' < s_1']\,p_2}{1p_1}\,|1|\right\} \\ &= \min\left\{1, \frac{r_2}{r_1}\frac{2p_2}{p_1}\right\}[u < s_1]\end{aligned}$$

- Acceptance probability for $m = (2 \to 1)$

$$\alpha_{2\to 1}(s \to s') = \min\left\{1, \frac{1}{1}\frac{r_1}{r_2}\frac{1p_1}{2p_2}\,|1|\right\} = \min\left\{1, \frac{r_1}{r_2}\frac{p_1}{2p_2}\right\}$$

```
% Probability x \in R and x \in R^2
p = [0.6 0.4];
% Probability of cross dimensional move
r = [0.3 0.9];
% Proposal width in R
epsilon = 0.2;
x = 0.2; k = 1;
```


1
0.8
0.6
0.4
0.2
0
−0.2
0
0.2
0.4
0.6
0.8
1


1
0.8
0.6
0.4
0.2
0
−0.2
0
0.2
0.4
0.6
0.8
1

```
% Prior Probability x \in R and x \in R^2
p = [0.6 0.4];

% Probability of cross dimensional move
r = [0.3 0.9];
% Proposal width in R
epsilon = 0.2;

T = 5000;
% initial state
x = 0.2; k = 1;

K = zeros(1, T); X = -0.1*ones(2, T);
```

```
for e=1:T,
 if k==1, % in R
 % Which move to choose
  if rand<r(k),       % Transdimensional move
    u = rand;
    x_new = [x u];
    alpha = min(1, 2*p(2)*r(2)/(p(1)*r(1)) )...
           *double(u < x);
  else
    x_new = 2*(rand-0.5)*epsilon + x;
    alpha = min(1,  double(( 0 < x_new & x_new < 1 )) );
  end;
 elseif k==2, % in R^2
  % Which move to choose
  if rand<r(k),         % Transdimensional move
    x_new = x(1);
    alpha = min(1, p(1)*r(1)/(2*p(2)*r(2)));
  else
    x_new = 1-x(2:-1:1);
    alpha = 1;
  end;
 end;
```

```
 if rand<alpha,
  x = x_new;
  k = length(x);
 end;
 X(1:k, e) = x';
 K(e) = k;
end;
```

Observations

- Many factors that determine the efficiency
  - The jump probabilities j_m
  - Proposals q_m
  - Too many choices: To design good moves is an art

Model Selection

- Setup a MH chain on the space

$$(k, \theta_k) \in \bigcup_{k \in \mathcal{K}} \{\{k\} \times \mathbb{R}^{n_k}\}$$

- The invariant target distribution is

$$p(k, \theta_k | Y)$$

- We need typically multiple types of moves to traverse the whole space $\mathcal{X}$.
- Each move leaves π invariant but we need different move types for ergodicity

Detailed Balance for Model Choice

- Detailed balance

$$\int_{(s,s')\in S\times S'} dsds'\pi(s)q_m(s'|s)\alpha_m(s\to s') = \int_{(s,s')\in S\times S'} dsds'\pi(s')q_m(s|s')\alpha_m(s'\to s)$$

- For each m, $q_m(s'|s)$ is the *joint* distribution of move type and destination
- Acceptance probability

$$\alpha_m(s\to s') = \min\left\{1, \frac{\pi(s')j_{-m}(s')g_{-m}(u')}{\pi(s)j_m(s)g_m(u)}\left|\frac{\partial(s',u')}{\partial(s,u)}\right|\right\}$$

- $j_m(s)$ Probability of choosing move m when in s

Moves