ENM531: Data-driven modeling and probabilistic scientific computing

Lecture #16: Sampling Methods

Paris Perdikaris
March 31, 2020

Monte Carlo approximation

$$\mathbb{E}_{x\sim p(x)}[f(x)] = \int f(x)p(x)dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i),$$

where x_i are drawn iid from $p(x)$

Markov Chain Monte Carlo

from *SIAM News*, Volume 33, Number 4

The Best of the 20th Century: Editors Name Top 10 Algorithms

By Barry A. Cipra

Algos is the Greek word for pain. *Algor* is Latin, to be cold. Neither is the root for *algorithm*, which stems instead from al-Khwarizmi, the name of the ninth-century Arab scholar whose book *al-jabr wa'l muqabalah* devolved into today's high school algebra textbooks. Al-Khwarizmi stressed the importance of methodical procedures for solving problems. Were he around today, he'd no doubt be impressed by the advances in his eponymous approach.

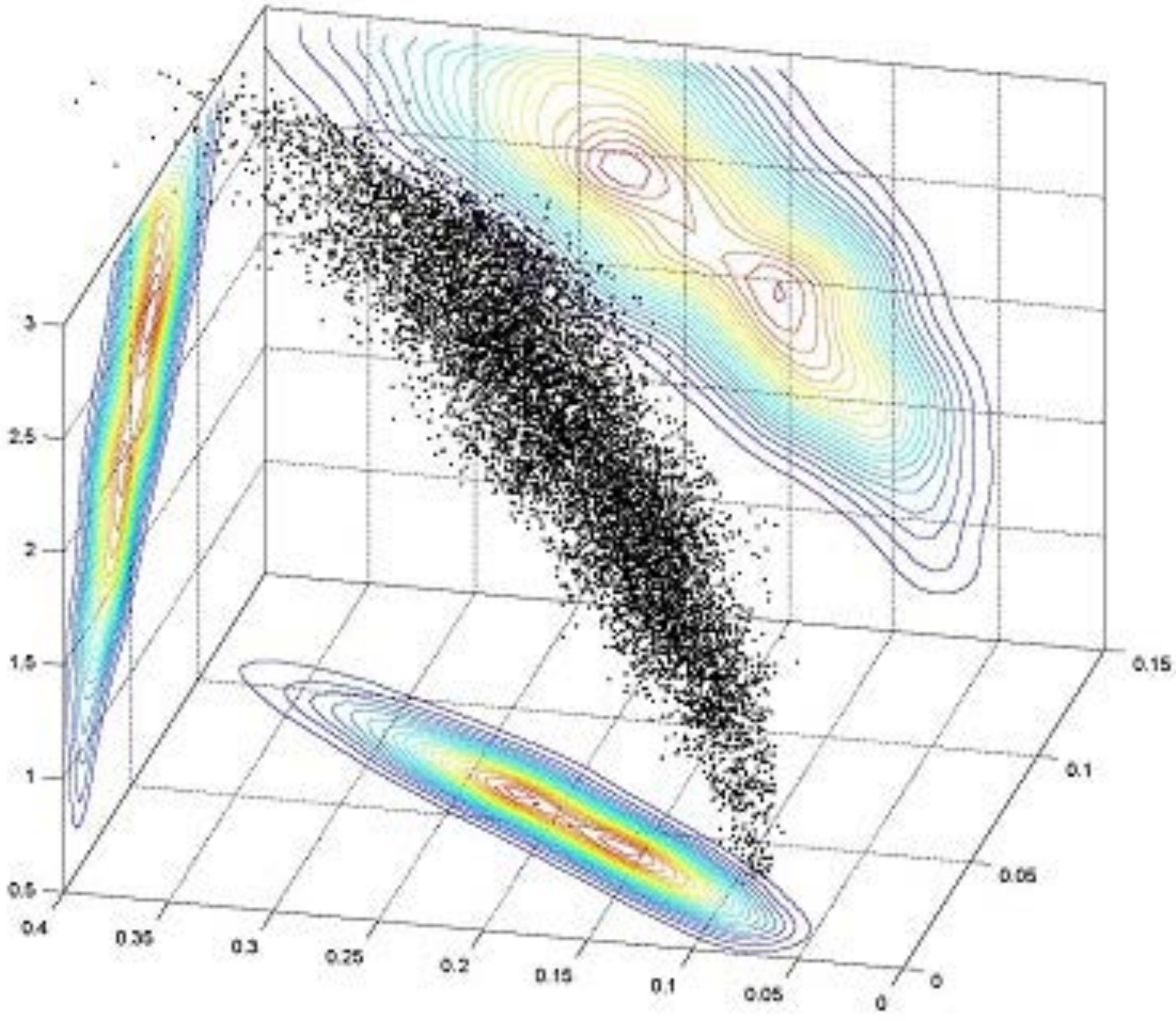
Some of the very best algorithms of the computer age are highlighted in the January/February 2000 issue of *Computing in Science & Engineering*, a joint publication of the American Institute of Physics and the IEEE Computer Society. Guest editors Jack Don-garra of the University of Tennessee and Oak Ridge National Laboratory and Fran-cis Sullivan of the Center for Comput-ing Sciences at the Institute for Defense Analyses put togeth-er a list they call the "Top Ten Algorithms of the Century."

"We tried to assemble the 10 al-gorithms with the greatest influence on the development and practice of science and engineering in the 20th century," Dongarra and Sullivan write. As with any top-10 list, their selections—and non-selections—are bound to be controversial, they acknowledge. When it comes to picking the algorithmic best, there seems to be no best algorithm.

Without further ado, here's the CiSE top-10 list, in chronological order. (Dates and names associated with the algorithms should be read as first-order approximations. Most algorithms take shape over time, with many contributors.)

1946: John von Neumann, Stan Ulam, and Nick Metropolis, all at the Los Alamos Scientific Laboratory, cook up the Metropolis algorithm, also known as the **Monte Carlo method**.

The Metropolis algorithm aims to obtain approximate solutions to numerical problems with unmanageably many degrees of freedom and to combinatorial problems of factorial size, by mimicking a random process. Given the digital computer's reputation for deterministic calculation, it's fitting that one of its earliest applications was the generation of random numbers.


3
2.5
2
1.5
1
0.5
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.05
0.1
0.15

Markov Chain Monte Carlo

John von Neumann

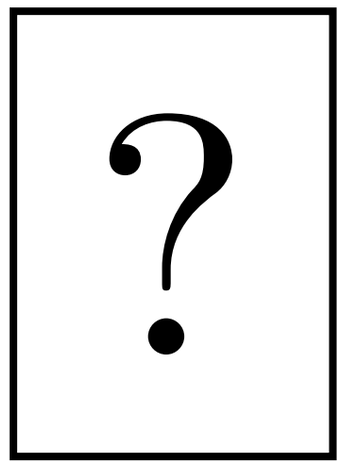
Stan Ulam

Edward Teller

Nicholas Metropolis

Marshall Rosenbluth

Augusta Teller



Arianna Rosenbluth

THE JOURNAL OF CHEMICAL PHYSICS VOLUME 21, NUMBER 6 JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

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

Markov Chain Monte Carlo

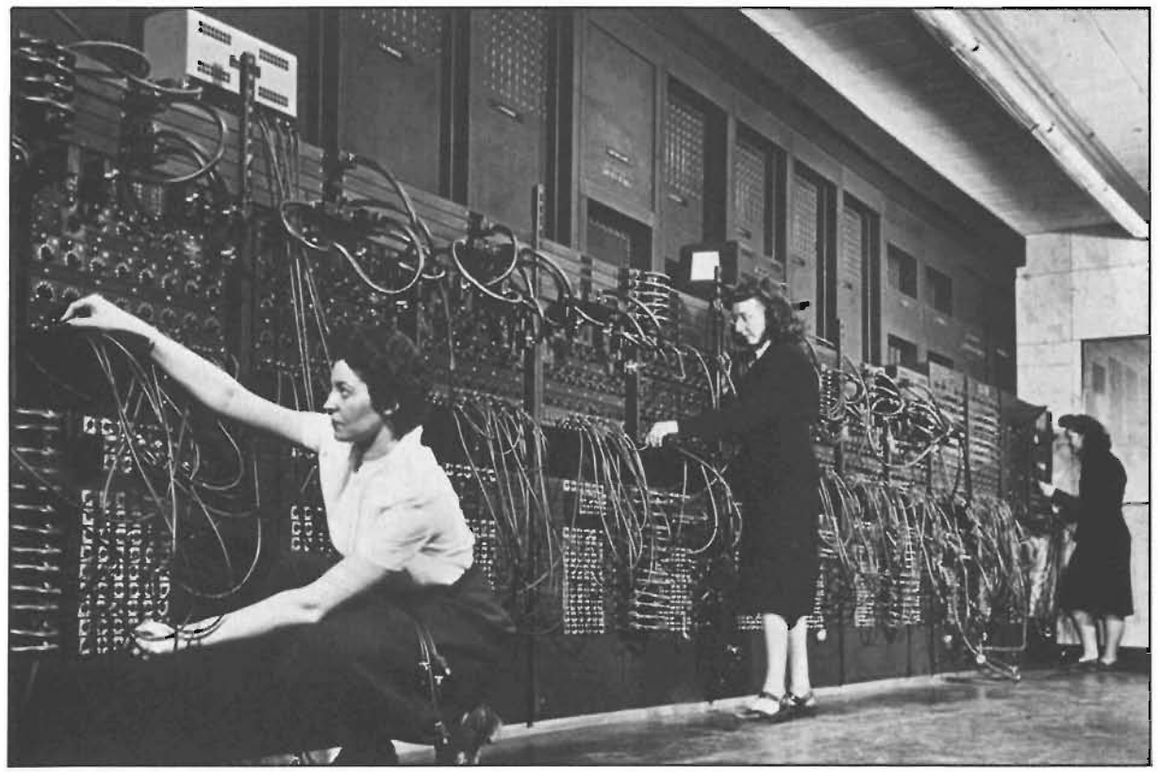


Fig. 1. Programming panels and cables of the ENIAC.

Pion-proton phase-shift analysis (Fermi, Metropolis; 1952)

Phase-shift analysis (Bethe, deHoffman, Metropolis; 1954)

Nonlinear coupled oscillators (Fermi, Pasta, Ulam; 1953)

Genetic code (Gamow, Metropolis; 1954)

Equation of state: Importance sampling (Metropolis, Teller; 1953)

Two-dimensional hydrodynamics (Metropolis, von Neumann; 1954)

Universalities of iterative functions (Metropolis, Stein, Stein; 1973)

Nuclear cascades using Monte Carlo (Metropolis, Turkevich; 1954)

Anti-clerical chess (Wells; 1956)

The lucky numbers (Metropolis, Ulam; 1956)

Fig. 6. Scientific triumphs achieved with the MANIAC. Nick Metropolis was a co-author of the publications resulting from these studies except for the ones on nonlinear coupled oscillators and anti-clerical chess.

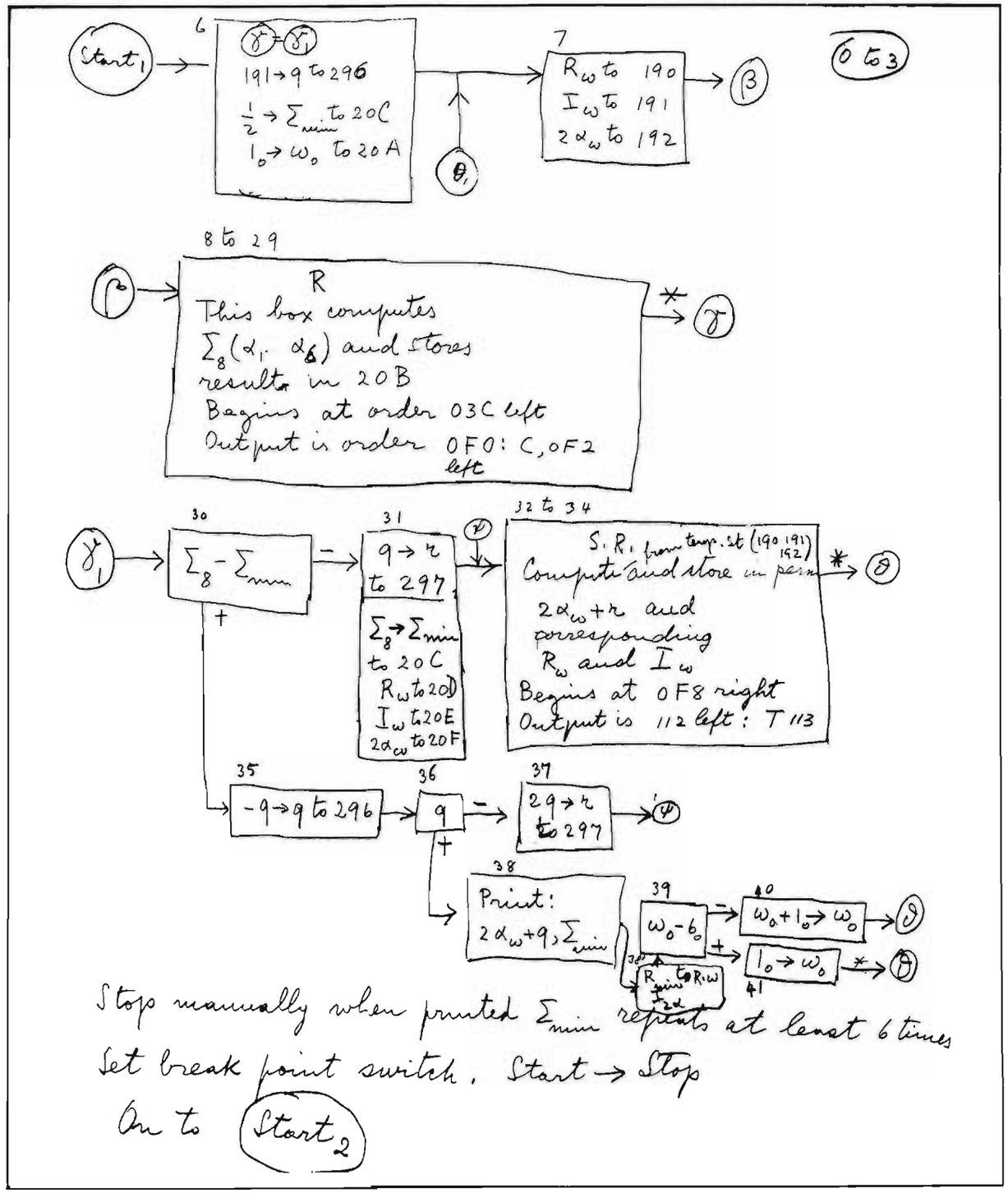


Fig. 4. A subprogram written by Fermi for calculating phase shifts by finding a minimum chi-squared in a fit to the data.

The Metropolis algorithm

- Choose a symmetric proposal matrix Q. So, $Q_{ab} = Q_{ba}$.
- Initialize $x_o \in X$.
- for $i \in 0, 1, 2, \ldots, n-1$:
  - Sample proposal x from $Q(x_i, x)$ if x is discrete, otherwise, $p(x \mid x_i)$.
  - Sample r from Uniform$(0, 1)$.
  - If

$$r < \frac{\tilde{\pi}(x)}{\tilde{\pi}(x_i)},$$

    accept and $x_{i+1} = x$.
  - Otherwise, reject and $x_{i+1} = x_i$.

Output: $x_o, x_1, \ldots, x_n$

- Symmetric proposals include:

$$J(\theta^* \mid \theta^{(s)}) = \text{Uniform}(\theta^{(s)} - \delta, \theta^{(s)} + \delta)$$

and

$$J(\theta^* \mid \theta^{(s)}) = \text{Normal}(\theta^{(s)}, \delta^2).$$

The Metropolis algorithm for Bayesian inference

Goal: We want to sample from

$$p(\theta \mid y) = \frac{f(y \mid \theta)\pi(\theta)}{m(y)}.$$

Typically, we don't know $m(y)$.

The notation is a bit more complicated, but the set up is the same.

We'll approach it a bit differently, but the idea is exactly the same.

We know $\pi(\theta)$ and $f(y \mid \theta)$, so we can can draw samples from these.

Our notation here will be that we assume parameter values
$\theta_1, \theta_2, \ldots, \theta_s$ which are drawn from $\pi(\theta)$.

We assume a new parameter value comes in that is θ^*.

The Metropolis algorithm for Bayesian inference

The Metropolis algorithm proceeds as follows:

1. Sample $\theta^* \sim J(\theta \mid \theta^{(s)})$.
2. Compute the acceptance ratio (r):

$$r = \frac{p(\theta^*|y)}{p(\theta^{(s)}|y)} = \frac{p(y \mid \theta^*)p(\theta^*)}{p(y \mid \theta^{(s)})p(\theta^{(s)})}.$$

3. Let

$$\theta^{(s+1)} = \begin{cases} \theta^* & \text{with prob min(r,1)} \\ \theta^{(s)} & \text{otherwise.} \end{cases}$$

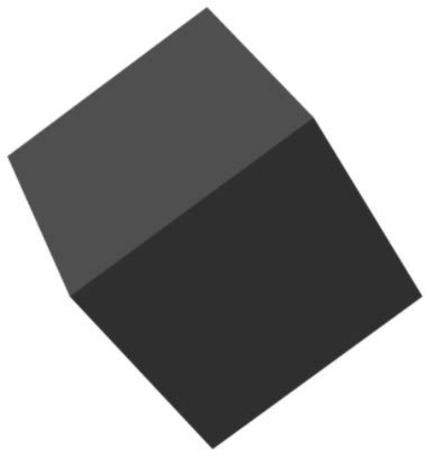
Remark: Step 3 can be accomplished by sampling $u \sim \text{Uniform}(0, 1)$ and setting $\theta^{(s+1)} = \theta^*$ if $u < r$ and setting $\theta^{(s+1)} = \theta^{(s)}$ otherwise.

Probabilistic programming

http://mc-stan.org/

https://github.com/pymc-devs/pymc3

Edward



http://edwardlib.org/

https://github.com/uber/pyro