Stat 202C
Monte Carlo Methods

Lecture 1: Introduction to MC methods

Background: choices of modeling & computing paradigms

- Approximate modeling + Exact computing (e.g. Dynamic programming)
- Exact modeling + Local computing (e.g. Gradient descent)
- Exact modeling + Global computing (MCMC, Here we are !)

Approximate model means you simplify the model, such as removing some edges in a graph to make it a tree or a chain, and thus removing certain energy terms.

Local computing means you may only find a local minimum (or maximum) and rely on heuristics to find a "good" one. Unfortunately most of the interesting function, like in deep learning, has astronomic number of local minima !

Introduction to MC methods

The essence is to represent a *target probability* by a set of "fair" samples.

Two large categories:

1, Sequential Monte Carlo
 -- Maintains and propagates a "*population*" through reweighting.

2, Markov chain Monte Carlo
 -- Simulates a *Markov chain* whose state follows the probability

- Discrete states (Gibbs sampler, Metropolis "walks" and "Jumps")
- Continuous States (Hamiltanion and Langevin "diffusions").

What is Monte Carlo?

Monte Carlo is a small hillside town in Monaco (near Italy) with casino since 1865 like Los Vegas in the US. It was picked by a physicist Fermi (Italian born American) who was among the first using the sampling techniques in his effort building the first man-made nuclear reactor in 1942.

The casino business is, literally, driven by tossing dice to simulate random events. Monte Carlo computing is to simulate samples from arbitrary probabilities by a single random function x=rand() which returns a pseudo-random number in the interval [0,1].

So, MC means a type of operation or business model.

Monte Carlo casino

Tasks in Monte Carlo computing: in increasing complexity

Task 1: Simulation: draw fair (typical) samples from a probability which governs a system.

$$x \sim \pi(x)$$

X is a typical state of the system.

Task 2: Integration / computing in very high dimensions, i.e. to compute

$$c = E[f(x)] = \int \pi(x) f(x) ds$$

Task 3: Optimization with an annealing scheme

$$x^* = \text{argmax } \pi(x)$$

Task 4: Learning and Bayesian hierarchical modeling from samples.

$$\Theta^* = \text{argmax } \ell(\Theta); \quad \ell(\Theta) = \sum_{i=1}^{m} \log p(x_i; \Theta)$$

Task 5: Visualizing the whole landscape of the probability

Task 1: Sampling and simulation

For many systems, their states are governed by some probability models. e.g. in statistical physics, the microscopic states of a system follows a Gibbs model given the macroscopic constraints. The fair samples generated by MCMC will show us what states are *typical* of the underlying system. In computer vision, this is often called "*synthesis*" --- the visual appearance of the simulated images, textures, and shapes, and it is a way to *verify* the sufficiency of the underlying model.

Suppose a system state x follows some global constraints.

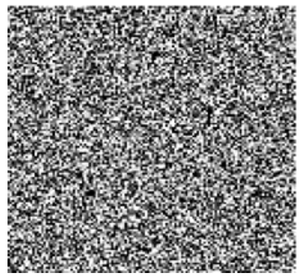
$$x \in \Omega = \{x : H_i(x) = h_i, i = 1, 2, ..., K\}$$

$H_i(s)$ can be a hard (logic) constraints (e.g. the 8-queen problem), macroscopic properties (e.g. a physical gas system with fixed volume and energy), or statistical observations (e.g the Julesz ensemble for texture).

Ex. 1 Simulating noise image

We define a "noise" pattern as a set of images with fixed mean and variance.

$$\text{noise} = \Omega(\mu, \sigma^2) = \{ I_\Lambda : \lim_{\Lambda \to Z^2} \frac{1}{|\Lambda|} \sum_{(i,j)\in\Lambda} I(i,j) = \mu, \quad \lim_{\Lambda \to Z^2} \frac{1}{|\Lambda|} \sum_{(i,j)\in\Lambda} (I(i,j) - \mu)^2 = \sigma^2 \}$$

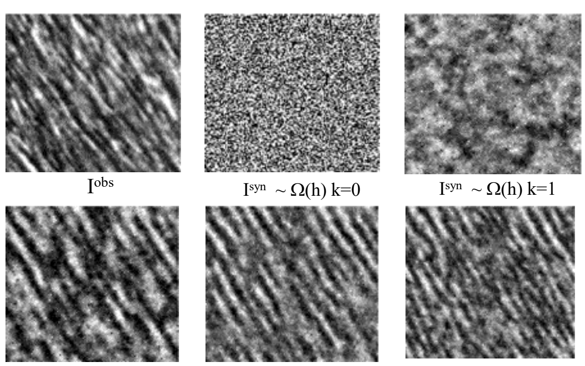


This is said to be a "*typical image*" of the Gaussian model.

Ex. 2 Simulating typical textures by MCMC in Stat232A

$$\text{a texture} = \Omega(h_c) = \{ I : \lim_{\Lambda \to Z^2} \frac{1}{|\Lambda|} \sum_{(i,j)\in\Lambda} h(I(i,j)) = h_c, \quad |h_c| = k \}$$

H_c are histograms of Gabor filters, i.e. marginal distributions of $f(I)$



I^{obs} $\quad I^{syn} \sim \Omega(h)$ k=0 $\quad I^{syn} \sim \Omega(h)$ k=1

$I^{syn} \sim \Omega(h)$ k=3 $\quad I^{syn} \sim \Omega(h)$ k=4 $\quad I^{syn} \sim \Omega(h)$ k=7

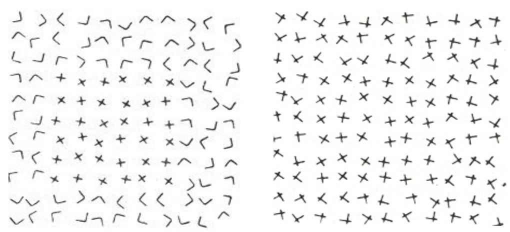
(Zhu et al, 1996-01)

Ex. 2 Simulating typical textures

Julesz's quest 1960-80s

early vision
(0.1-0.4sec)

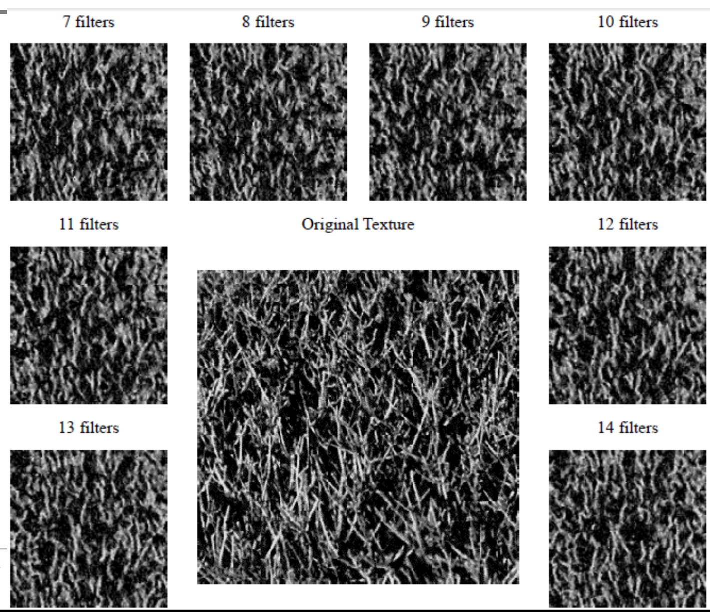
"What features and statistics are characteristics of a texture pattern, so that texture pairs that share the same features and statistics cannot be told apart by pre-attentive human visual perception?"



His quest was not answered partly due to the lack of general techniques for generating fair texture pairs that share the same features and statistics, no more no less.

--- To visualize the typical state of a probability in the high-dimensional space.

An example simulated by student from Stat232A



Ex 3: Simulating typical protein structures

We are interested in the *typical configurations*, of protein folding given some known properties. The set of typical configurations is often huge !

Molecular dynamcs

Poteintial energy function $U(x)$

Kinetic energy $K(\dot{x})$

Total energy

$$H(x) = U(x) + K(\dot{x})$$

Statistical physics

$$x \sim \pi(x) = \frac{1}{Z} \exp\{-\frac{1}{KT} U(x)\}$$

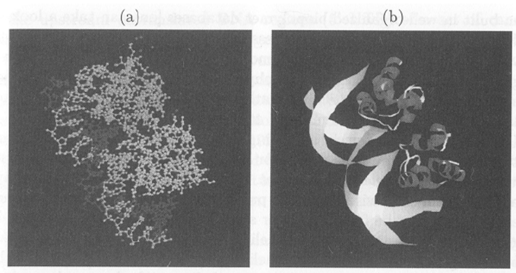
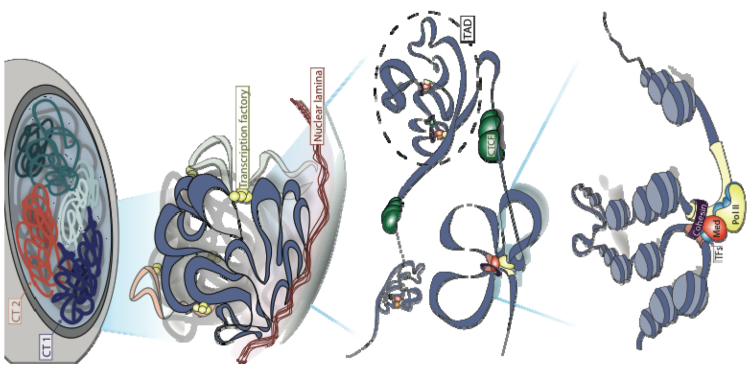


FIGURE 1.4. (a) A ball-and-stick plot of the interaction between a regulatory protein in yeast, 3CRO, and the DNA segment to which it binds. (b) The same structure as in (a), but expressed by a ribbon representation widely used in the protein structure modeling community.

[From Jun Liu]

3D genome representation in space and time

The real system is hierarchical and heterogeneous,
and the interactions (potentials) are in 3D space and time.



From Dr. Bin Ren, UCSD

Task 2: Scientific computing

In scientific computing, one often needs to compute the integral in very high dimensional space.

Monte Carlo integration,

e.g.

1. Estimating the expectation by empirical sample mean.
2. Importance sampling

Approximate counting

e.g.

1. How many non-self-intersecting paths are in a 2 n x n lattice of length N?
2. Estimate the value π by generating uniform samples in a unit square.

Ex 4: Monte Carlo integration

Often we need to estimate an integral in a very high dimensional space Ω,

$$c = \int_\Omega \pi(x) f(x) dx$$

We draw N samples from $\pi(x)$,

$$x_1, x_2, \ldots, x_N \sim \pi(x)$$

Then we estimate C by the sample mean

$$\hat{c} = \frac{1}{N} \sum_{i=1}^{N} f(x_i)$$

For example, we estimate some statistics for a Julesz ensemble $\pi(x;\theta)$,

$$C(\theta) = \int_\Omega \pi(x;\theta) H(x) dx$$

Ex 5: Approximate counting in polymer study

For example, what is the number K of Self-Avoiding-Walks in an n x n lattice?

Denote the set of SAWs by $\Omega_{n^2} = \{r\}$

An example of n=10. (Persi Diaconis)

The estimated number by Knuth was $(1.6 \pm 0.3) \times 10^{24}$

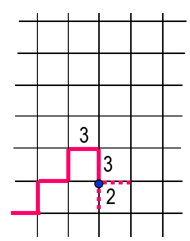
The truth number is 1.56875×10^{24}

Ex 5: Approximate counting in polymer study

Computing K by MCMC simulation

$$\begin{aligned} K &= \sum_{r \in \Omega_{n^2}} 1 = \sum_{r \in \Omega_{n^2}} \frac{1}{p(r)} p(r) \\ &= E[\frac{1}{p(r)}] \\ &\approx \frac{1}{M} \sum_{i=1}^{M} \frac{1}{p(r_i)} \end{aligned}$$

Sampling SAWs r_i by random walks (roll over when it fails).



$$p(r) = \prod_{j=1}^{m} \frac{1}{k(j)}$$

Task 3: Optimization and Bayesian inference

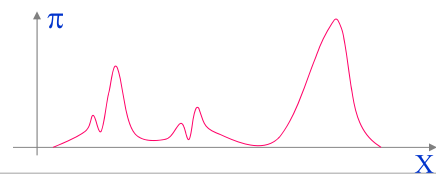
A basic assumption, since Helmholtz (1860), is that biologic and machine vision compute the most probable interpretation(s) from input images.

Let I be an image and X be a semantic representation of the world.

$$X^* = \arg\max \pi(X|I)$$

In statistics, we need to sample from the posterior and keep multiple solutions.

$$(X_1, X_2, ..., X_k) \sim \pi(X|I)$$

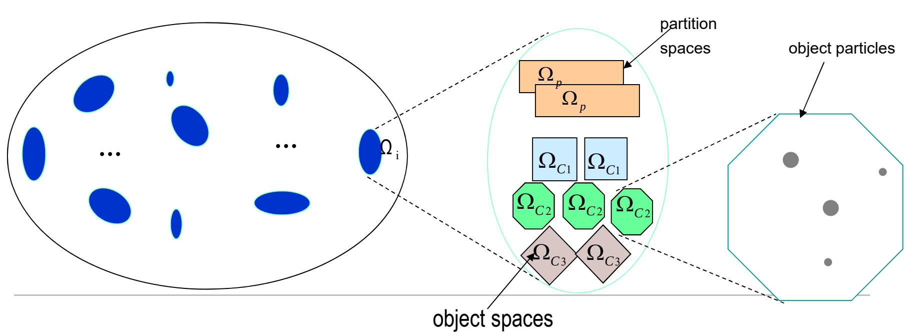


Traversing Complex State Spaces

1. The state space Ω in computer vision often has a large number of sub-spaces of varying dimensions and structures, because of the diverse visual patterns in images.

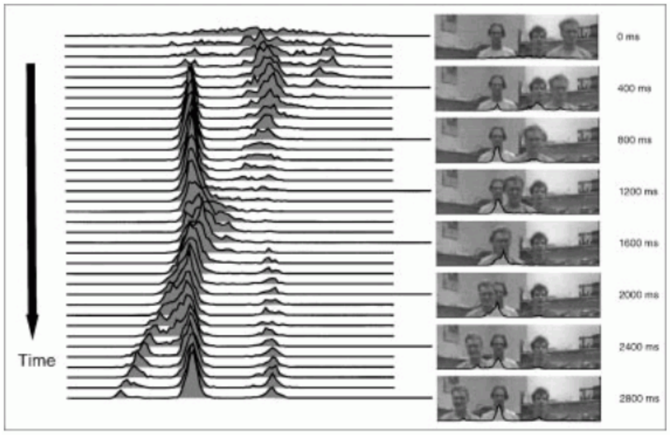
2. Each sub-space is a product of
   some *partition (coloring) spaces* ---- what go with what?
   some *object spaces* ---- what are what?

3. The posterior has low entropy, the *effective volume* of the search space is relatively sma



Ex. 6 Tracking objects by Sequential Monte Carlo

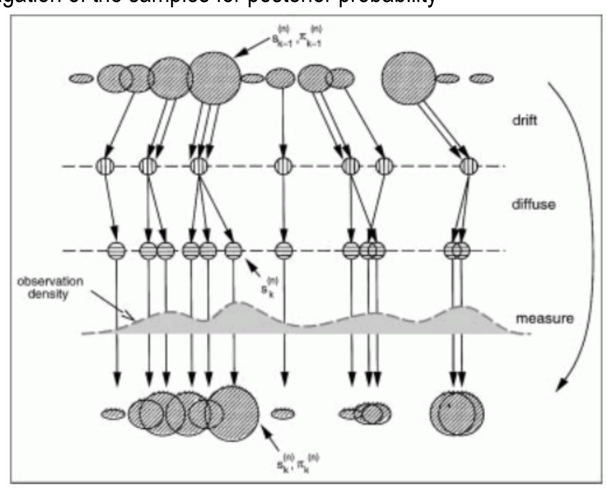
Main goal: preserving uncertainty over time.



M. Isard and A. Blake, “CONDENSATION—Conditional Density Propagation for Visual Tracking,” Int’l J. of Computer Vision, 29(1), 5–28, 1998.

Ex. 6 Tracking objects by Sequential Monte Carlo

Propagation of the samples for posterior probability

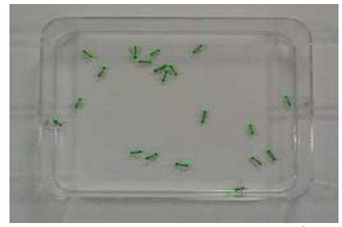


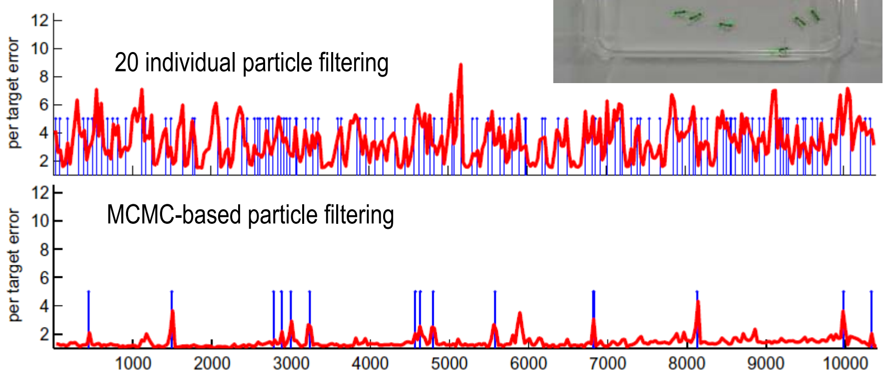
Ex. 7 MCMC-Based Particle Filters

From Khan, Balch & Dellaert

Running particle filters in large state spaces
(ants, bees, people, sports)

Blue: lose track occurs,
Red: pixel errors per target





Task 4: Learning and Model Estimation

In statistical learning and machine learning, a common problem is "point estimation" by Maximum likelihood (MLE): to learn the parameters Θ of a model $p(x;\ \Theta)$ from a set of Examples $D = \{x_i,\ i = 1,2,\dots,m\}$:

$$\Theta^* = \arg\max \ell(\Theta); \quad \ell(\Theta) = \sum_{i=1}^{m} \log p(x_i;\Theta)$$

When the probability is of the Gibbs form,

$$p(x;\ \Theta) = \frac{1}{Z}\exp^{-<\Theta,\ H(x)>}$$

The MLE $\frac{\partial \ell(\Theta)}{\partial \Theta} = 0$ will need to be computed by stochastic gradients,

$$\frac{d\Theta}{dt} = \eta(E_{p(x;\Theta_t)}[H(x)] - \overline{H}^{obs}), \quad \overline{H}^{obs} = \frac{1}{m}\sum_{i=1}^{m} H(x_i)$$

$E_{p(x;\Theta_t)}[H(x)] = \int p(x;\Theta_t)H(x)dx$
has to be approximated by samples $D_t = \{x_j,\ j = 1,2,\dots,n\} \sim p(x;\Theta)$.

Task 4: Learning and Model Estimation

One special example is the Restricted Bolzmann Machine (RBM) with binary input v and output h (hidden):

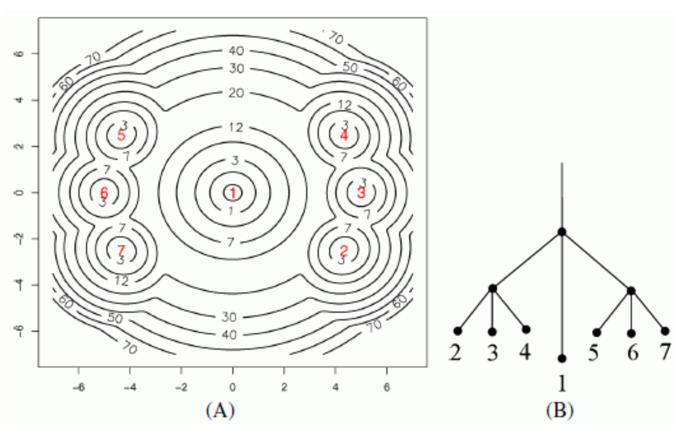
$$p(\boldsymbol{v}, \mathbf{h}; \Theta) = \frac{1}{Z} \exp(-E(\boldsymbol{v}, \mathbf{h}))$$

$$E(\boldsymbol{v}, \mathbf{h}; \Theta) = -\boldsymbol{a}^T \boldsymbol{v} - \boldsymbol{b}^T \mathbf{h} - \boldsymbol{v}^T W \mathbf{h}.$$

$$\Theta^* = (W, \boldsymbol{a}, \boldsymbol{b})^* = \operatorname{argmax} \sum_{i=1}^{n} \log \int p(\boldsymbol{v}_i, \mathbf{h}; \Theta) d\mathbf{h}$$

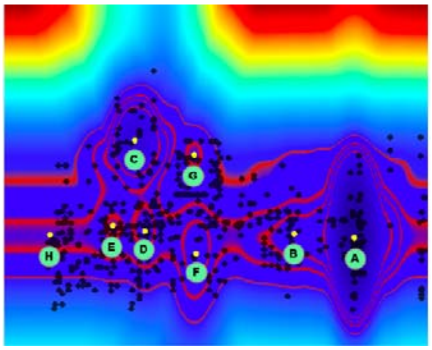
As the algorithm iterates in infinite number of steps, and thus the network of computing is *infinite number of layers.* This RBM was actually the original "deep learning", which is quite different from the current multi-layer neural network.

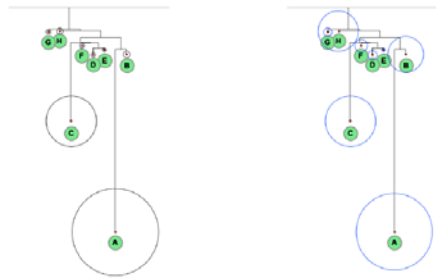
Task 5: Visualizing the landscape of an energy/probability



Q. Zhou and W. Wong, "Reconstructing the energy landscape of a distribution from Monte Carlo sample," Annals of Applied Statistics, 2008.

Ex. 8 A 2D landscape for a K-mean clustering problem





Probability mass Volume

The circles represent the relative sizes

By Maria Pavlovskaia, UCLA

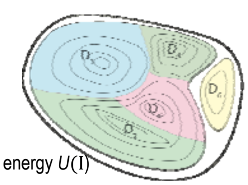
Ex. 9 Visualizing in the landscape of Image models

Many complex systems are governed by a probability model and represented by energy landscape U.

Minima of energy $U(\mathrm{I})$ are maxima of probability $p(\mathrm{I};\omega)$

- Physical states (magnetic states, molecular states, folding states of a protein chain)
- Memories/concepts learned from training data (focus of our application)

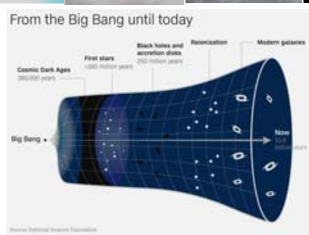
Partition of 2D landscape into basins of attraction for local minima.



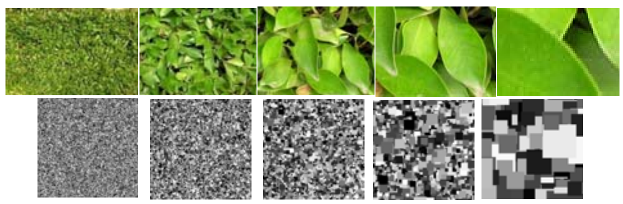
energy $U(\mathrm{I})$

probability mass of $p(\mathrm{I};\theta)$

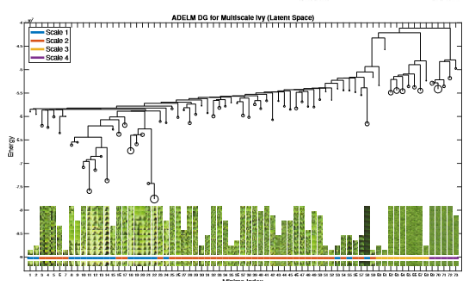
Building a Telescope to looking into high dim spaces



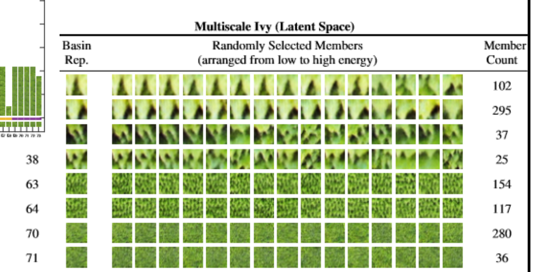
Scaling (zoom-out) → entropy rate decreases



Ex. 9 Visualizing in the landscape of Image models

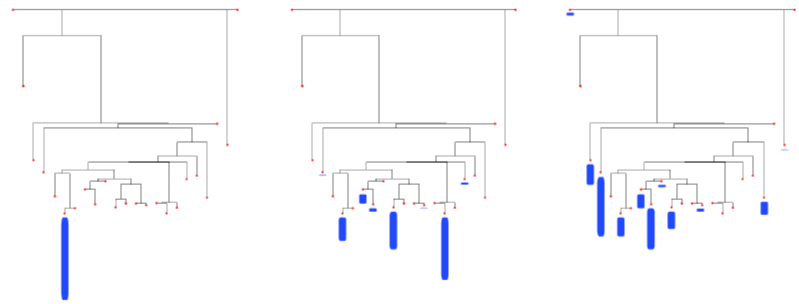


By Mitch Hill and Erik Nijkamp, UCLA, 2018



Ex. 10 Visualizing the behavior of algorithms in the landscape

The bars show the relative frequency that an algorithm visits the local energy basins.



(a) SW-cut (b) EM (c) k-means

By Maria Pavlovskaia, UCLA

Summary

MC is a general purpose technique for sampling from complex probabilistic models.

In high dimensional space, sampling is a key step for

(a) modeling (simulation, synthesis, verification, visualization)
(b) learning (estimating parameters)
(c) estimation (Monte Carlo integration, importance sampling)
(d) optimization (together with simulated annealing).
(e) imputation (Bayesian hierarchical model).
(f) visualization (landscape and complexity of the problem).

It can achieve global optimal solution for complex models!

A Brief History of MCMC

1942-46: Real use of MC started during the WWII
--- study of atomic bomb (neutron diffusion in fissile material)

1948: Fermi, Metropolis, Ulam obtained MC estimates for the eigenvalues
of the Schrodinger equations.

1950s: Formating of the basic construction of MCMC, e.g. the Metropolis method
--- applications to statistical physics model, such as Ising model

1960-80: Using MCMC to study phase transition; material growth/defect,
macro molecules (polymers), etc.

1980s: Gibbs samplers, Simulated annealing, data augmentation, Swendsen-Wang, etc
global optimization; image and speech; quantum field theory,

1990s: Applications in genetics; computational biology, vision etc.

2000s: Application in vision, graphics, robotics simulation etc.

2010s: Applications in machine learning, deep learning etc.

Some MCMC developments related to vision