The Markov chain is a <u>weighted</u> (or biased) <u>random walk</u>.

<u>Example</u>

Metropolis:

$$\vec{p} = (p_1, \ldots, p_N)$$

$$\uparrow$$

$$p_\alpha.$$

The goal is to generate a sequence of numbers x_n with relative frequencies approaching $\text{Prob}(x_n = x_\alpha) = p_\alpha$.

Assume that after the n'th step,

$x_n = x_\alpha$.

Choose an x_β from the neighborhood of x_α by the rule

$$x_\beta = x_\alpha + \left(r - \tfrac{1}{2}\right)\Delta x$$

random number on [0, 1].

If $p_\beta = p(x_\beta)$ is such that $p_\beta \geq p_\alpha$, we set $x_{n+1} = x_\beta$.

If $p_\beta < p_\alpha$, we generate a new random number $\rho \in [0, 1]$.

* If $\rho < \frac{p_\beta}{p_\alpha}$, we set $x_{n+1} = x_\beta$.

* If $\rho \geq \frac{p_\beta}{p_\alpha}$, we set $x_{n+1} = x_\alpha$.

(i.e. we keep the old value.)

This is the Metropolis Monte Carlo method.

Note that we do not need to know the absolute values of p_α, only the relative values

$$\frac{p_\alpha}{p_\beta}.$$

Normalization factors are therefore not necessary.

This is a make or break point of the method.

Why?

Consider statistical mechanics:

Let $\vec{x}_n$ represent a point in phase space (position of $N \sim 10^{10}$ interacting particles for example).

In order to use the Metropolis Monte Carlo method to sample this phase space, we only need the Boltzmann factor $e^{-H(\vec{x})/k_B T}$, but not the normalization, which is the partition function itself,

$$Z = \sum_{\{\vec{x}\}}' e^{-H(\vec{x})/k_B T}.$$

Ordinary differential equations

Example:

$$y'' + g(x)\,y' = r(x)$$

This can be rewritten as

$$\begin{cases} y' = z \\ z' = r - gz \end{cases}$$

One second order equations $\Longleftrightarrow$

two first order equations.

For general:

Nth order differential equation

$\Longleftrightarrow$

N first order equations.

Hence, the general case is:

$$\boxed{\begin{gathered} y_i' = f_i(x; y_1, \ldots, y_N) \\ i = 1, \ldots, N \end{gathered}}$$

Boundary conditions:

Algebraic constraints on the variables y_i.

These must be fulfilled at discrete points x_j.

Initial value problem:

Constraints must be fulfilled at one point x_0.

Find $y_i(x)$ for $x > x_0$.

Two-point boundary value problem.

Constraints must be fulfilled at two points x_s and x_f.

Find $y_i(x)$ for $x_s < x < x_f$.

For the following, we consider the two-point boundary value problem.

Definition of the generic problem:

$n_1 < N$ constraints at x_s:

$$B_{sj}(x_s; y_1, \ldots, y_N) = 0 \quad j = 1, \ldots, n_1.$$

$n_2 < N$ constraints at x_f:

$$B_{fk}(x_f; y_1, \ldots, y_N) = 0, \quad k = 1, \ldots, n_2.$$

$$n_1 + n_2 = N$$

Example:

Solving the Laplace equation on the unit interval

– formulation of problem.

$$\frac{d^2\phi}{dx^2} = 0, \qquad \phi(0) = 0 \qquad \phi(1) = 1.$$

We rewrite the Laplace equation as two first order equations:

$$\begin{cases} \dfrac{dy_1}{dx} = y_2 & y_1(0) = 0 \\[2ex] \dfrac{dy_2}{dx} = 0 & y_1(1) = 1. \end{cases}$$

The constraints are thus:

$$\begin{cases} B_{s1}(x; y_1, y_2) = y_1(x) \\ B_{f1}(x; y_1, y_2) = y_1(x) - 1 \end{cases}$$

$$\begin{cases} B_{s1}(0, \vec{y}) = 0 \Rightarrow y_1(0) = 0 \\ B_{f1}(1, \vec{y}) = 0 \Rightarrow y_1(1) = 1. \end{cases}$$

There are two classes of methods:

(A) "Shooting" methods

(B) Relaxation methods.

(A) "Shooting" methods.