# Métodos Monte-Carlo

Vicente Martín

FIM-UPM

v 0.0
Feb. 2016

## Monte-Carlo Methods: The Big Picture

- „**Monte-Carlo** methods are **very bad methods**... but  sometimes is the best method available -or the only feasible-“
  - To the potentiall pitfalls of all numerical methods...
  - … the very high inefficiency of a Monte-Carlo Method is added.
  - They must be used when all other numerical methods are even worse.

# Monte-Carlo Methods: The Big Picture

- The convergence rate of a Monte-Carlo Method in *n* nodes is typically:

$$\varepsilon \sim \frac{1}{\sqrt{n}}$$

  - This is an almost inescapable consequence of the central limit theorem.
  - The proportionality can be optimized (sometimes by a factor of *millions*) but the asymptotic behavior not.
    - Note that *asymptotic* might be too far away for any practical purpose/meaning.
  - Use Monte-Carlo only when all other (specially *deterministic*) methods are even worse than 1/√n
    - Ej.: Many degrees of freedom, far from perfurbative regime (i.e no good approximation exist)

# Monte-Carlo Methods: The Big Picture

- Some numbers/explanations:
    - Note that *asymptotic* might be too far away for any practical purpose/meaning:
        - Simpson's rule for integration in *d* dimensions converge asymptotically as $n^{-4/d}$
            - If d<8, then Simpson is probably better than Monte-Carlo
        - If d=10, how many points do you need to get close to „asymptotic“?
            - The number of evaluations for a grid of 10 points in each of the 10 dimension is $10^{10}$... clearly prohibitive for a barely moderate precision calculation...

## So, what is a Monte-Carlo Method?

- Let's have a configuration space S.
- Let's have a probability measure π on S.
- Then, generate many ***random samples from π*** to evaluate some average (expected) property.
    - The volume of an odd-shaped object.
    - The estimated value of a market share.
    - The magnetization of a material.
    - The convergence properties of some algorithm..

## So, sampling?

- *Static* and *Dynamic* Monte-Carlo:
- Static Monte-Carlo:
    - Directly generate a sequence of statistically independent samples from $\pi$.
    - This is ok if $\pi$ is well known or not too complex.
    - e.g.: Monte-Carlo integration in spaces of not very high dimension.

## So, sampling?

- Dynamic (pseudo)Monte-Carlo:
    - Invent a stochastic process within S that has $\pi$ as its ***unique equilibrium distribution***
        - Also called stationary or limiting distribution.
    - The process is simulated till the system is in equilibrium. Then, the time averages converge to $\pi$-averages.
    - The simulation process is just a computational device; no correspondence with reality is required.
    - This process is taken to be a ***Markov process***.

- Assume S is discrete: Markov Chain in S:
  - A sequence of random variables $X_0$, $X_1$, $X_2$ … in S such that the transitions $X_t \to X_{t+1}$ are statistically independent.
- A Markov chain is specified by:
  - A initial probability distribution α in S. $P(X_0 = x) = \alpha_x$
  - A transition probability matrix $P = \{p_{xy}\}_{x,y \in S} = \{p(x \to y)\}$
    - $p_{xy} \geq 0$ for all x,y
    - $\sum_y p_{xy} = 1$ for all x
    - $\mathbf{P}(X_{t+1} = y \mid X_t = x) = p_{xy}$
- The Markov Chain is specified by the joint probabilities:

$$\mathbf{P}(X_0 = x_0, X_1 = x_1, \ldots, X_n = x_n) = \alpha_{x0} p_{x0x1} p_{x1x2} \ldots p_{xn-1xn}$$

- The n-step transition probabilities:

$$p^{(n)}_{xy} = \mathbf{P}(X_{t+n} = y \mid X_t = x)$$

- $p^{(0)}_{xy} = \delta_{xy}$, $p^{(1)}_{xy} = p_{xy}$,... in general $\{p^{(n)}_{xy}\}$ are the matrix elements of $P^n$.
- A Markov chain is ***irreducible*** if any state can be reached from any other (ergodicity) i.e.:

  For each $x,y \in S$ there is $n \geq 0$ for which $p^{(n)}_{xy} > 0$.
- The ***period*** $d_x$ of a state x is the gcd of the numbers n for which $p^{(n)}_{xx} > 0$.
  - If $d_x=1$ the state x is ***aperiodic***.
  - In an irreducible chain all states have the same period.

# Convergence to equilibrium.

- A probability measure $\pi = \{\pi_x\}_{x\in S}$ is a ***stationary distribution*** for the Markov Chain P if:

$$\sum_x \pi_x p_{xy} = \pi_y \text{ for all } y$$

- Let P be the transition probability matrix of an irreducible Markov chain of period d, then:
  - If a stationary probability distribution exists, then it is unique and $\pi_x > 0$ for all x.
  - If P is aperiodic $\mathit{lim}_{n\to\infty}\, p^{(n)}_{xy} = \pi_y$

- Now, **to set up a dynamic (pseudo) Monte-Carlo** method to sample from $\pi$:

**Design a transition probability matrix** $\{p_{xy}\}$ satisfying:

- ***Irreducibility***. (For each $x,y \in S$ there is $n \geq 0$ for which $p^{(n)}_{xy} > 0$)
- ***Stationarity*** of $\pi$. ($\sum_x \pi_x p_{xy} = \pi_y$ for all $y \in S$. Stronger alternative: detailed balance: For all $x,y \in S$ $\pi_x p_{xy} = \pi_y p_{yx}$)

Then the Markov chain P produces a legitimate way of estimating $\pi$-averages.

Any starting point will do and the system is guaranteed to converge to equilibrium when $t \to \infty$. „Time“ averages will converge to $\pi$-averages with convergence $\sqrt{n}$.

- This might be a method to build the algorithm but ***nothing is said about its efficiency***.
- Problem: Correlation.

  The ***successive states of the chain $X_0$, $X_1$,... can be correlated***. In principle, much more correlated than the independently sampled stationary Monte-Carlo.

- Idea: a well designed method will jump from one state to other *qualitatively different* in the sense that the purpose is to *sample from the population that is significant for the problem (typical set),* instead of sampling many times from parts of S that do not contribute.

## Monte-Carlo Methods: Back to Basics. Random Numbers

The first mandate in a Monte-Carlo Method is:

„Generate random numbers according to a distribution π“

- A set of random numbers is typically characterized by a set of numbers called ***moments.*** The most important are***:***
  - The ***mean (average value)*** of a set of samples $\{x_i\}$, i=1...n

$$\mu_n \equiv \frac{1}{n}\sum_{i=1}^{n} x_i$$

- The ***variance*** of the set of samples, that measures the variation from the mean value $\mu_n$ (or from the *true mean*, $\mu$, if known):

$$\sigma_n^2 \equiv \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu_n)^2$$

(numerator is taken to be n-1 when the samples come from a normal distribution)

- A system from which random samples are taken is described by a ***probability density function f(x).*** The probability that a sample from *f(x)* is in a set $\alpha$ is given by:

$$\int_\alpha f(x)\,dx$$

Probabilities are assumed normalized over its definition domain $\Omega$: $\int_\Omega f(x)\,dx = 1$

- Mean and variance of continuous distributions are given by integrals:

$$\mu = \int_{\Omega} x\, f(x)\, dx$$
$$\sigma^2 = \int_{\Omega} (x-\mu)^2 f(x)\, dx$$

-

- Example: Uniform distribution over [0,m]. The probability density function, mean and variance:

$$f(x)=\frac{1}{m}$$

$$\mu=\int_0^m\left(\frac{x}{m}\right)dx=\frac{m}{2}$$

$$\sigma^2=\int_0^m\left(\frac{1}{m}\right)\left(x-\frac{m}{2}\right)^2dx=\frac{m^2}{12}$$

- Example: Normal distribution with mean μ and variance σ. The probability density function is:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/(2\sigma^2)}$$

## The Central limit theorem or why we like so much normal distributions

- Normal distributions are quite usual in nature.
- It is easy to build samples from a normal distribution starting from almost any distribution:

**Central limit Theorem:** let f(x) be a distribution with mean $\mu$ and variance $\sigma^2$. If $\mu_n$ is the mean of a sample of size n taken from f(x), then the distribution of

$$y_n = \sqrt{n}\,\frac{\mu_n - \mu}{\sigma}$$

Approaches the normal distribution with mean 0 and variance 1.

## The generation of (pseudo)random numbers

- We want sequences of numbers that show the same properties than true random numbers.
    - … but we also want them to be reproducible: i.e. To be able to generate again the same exact sequence.
    - … and we want them fast.
    - … and in very large quantities.

This is why we use ***deterministic pseudorandom generators***: they appear to be random since its statistical analysis (mean, variance...) produces the same results than a truly random sequence, but they are generated in a reproducible way starting from a seed using a short sequence of operations.

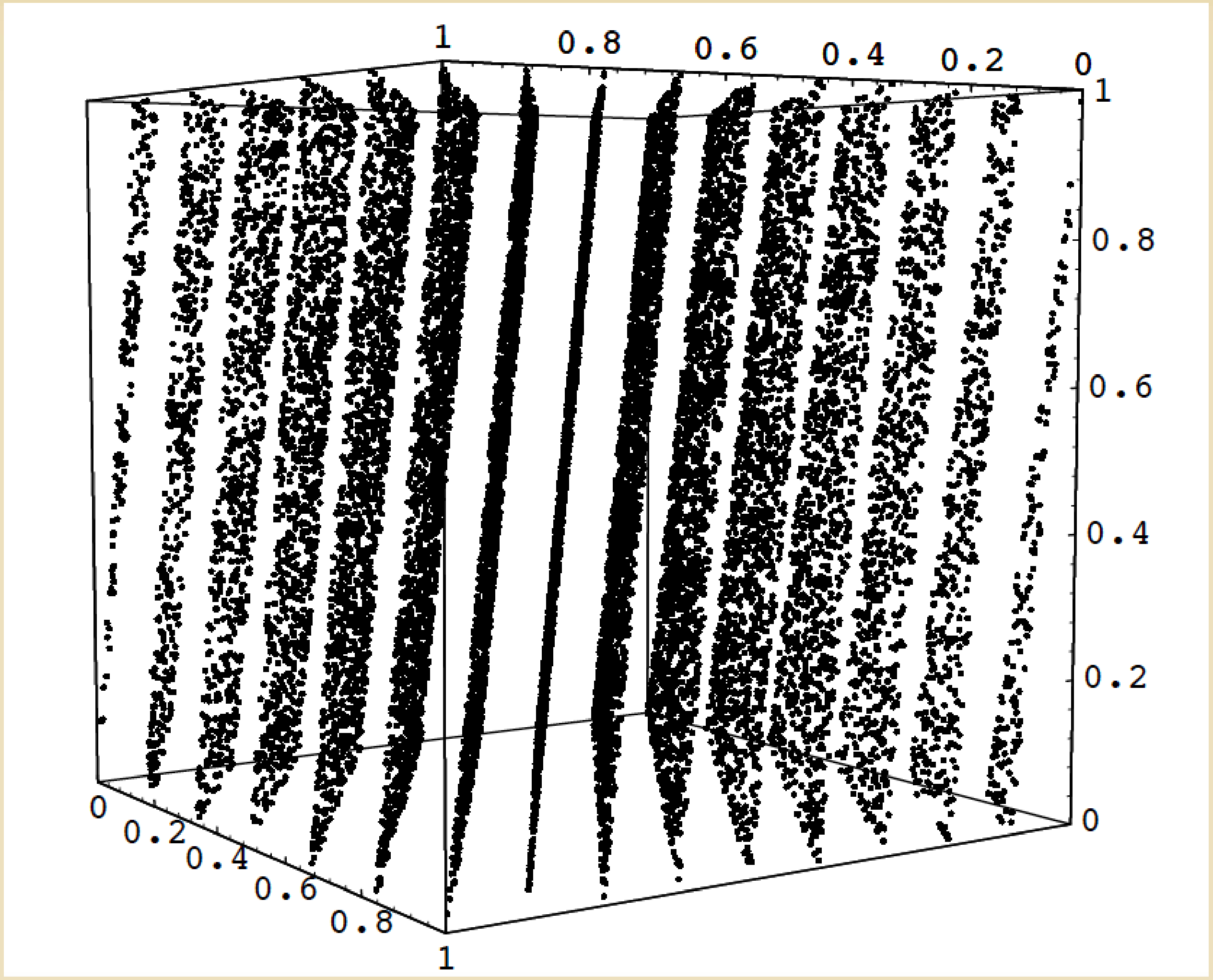
- **Beware:** bad pseudorandom number generators abound.
- But it is not difficult to find good ones.
  - Be aware that their purpose might be different: random numbers for cryptography are required to typically met more stringent standards than those for usual numerical simulations.
  - … and also take more time.
  - … and sometimes other properties are also important (quasi-random numbers. e.g.: space filling sequences)

- Historically, the generation of pseudorandom numbers has been dominated by the LCG: ***linear congruential generator***:

$$I_{j+1} = aI_j + c \quad (\textit{mod}\ m)$$

Where **m** is the ***modulus***, **a** is a positive integer ***multiplier*** and **c** is a nonnegative integer ***increment***. When c=0 it is called a Multiplicative LCG (MCLG).

  - The generated sequence is repeated with a maximal period of m but, with badly chosen parameters it can be much less.



- RANDU: LCG($m=2^{31}$, $a=2^{16}+3$, $c=0$, $I_0=1$). Great-grandfather of many „modern“ rngs. IBM System 360

- If k numbers in the sequence are interpreted as a point in k-dimensions, they do not tend to fill the space, but to cluster in k-1 dimensional hyper-planes.
- In 32-bit machines, if m is chosen as large as possible but still fitting in one register, the largest m is $2^{32}$, and the (maximum) number of planes when three numbers in the sequence are used to represent a point in 3D is $(2^{32})^{1/3}=1600$.
  - With badly chosen m and a, it can be much worse... and you might end up sampling a small subset.
  - To avoid HW limitations, bad m and a were not so rare. e.g.: use a power of 2 for m to avoid the need of extra-lenght register for the multiplication and modulo operations. See infamous RANDU from IBM S/360.

## Some general advice from NR: The don'ts

- Do not use congruential generators... alone.
- Do not use generators with period less than $2^{64}$
- Do not use generators that warns against use of its low-order bits (that is a sign of an underlying LCG used in isolation)
- Do not use the built in generators of C or C++ languages (rand, srand).
- Do not use  generators that take more than ~50 operations per 64 bit result (or double precision)... there are cheaper ones.
- Do not use algorithms designed for crypto... unless you are doing crypto, that is.
- Do not use generators with periods larger than $10^{100}$ they are not useful... it is overkill.

## Some general advice: The do's

- Use a combined generator.
  - At least with two independent generators, algorithmically disparate, that share no state and have different periods.
  - Each of the components should be reasonably well studied mathematically and pass the typical statistical tests (Diehard or NIST, for example. Note: requiring to the *combined* generator to pass these tests is not enough)
  - Pay attention to its efficiency: number of operations, size of the state to keep it in the registers.
  - The output of the combined generator should not affect the independent evolution of each generator.
  - The methods should be combined using binary operators that do not reduce the randomness of the inputs, that is, use + or ^, but not a multiplication (multiply by a power of 2 produces trailing zeros, no matter how random is the other.

There is a further possibility to mix random number generators: instead of combine their results through + or ^, take the output of one of the generators and run it through the other generator (without altering the state of the first one). This is known as a *successor relation* and the mix is known as a *composed generator*.

- Composition is as costly as combination and does not increase the size of the state or the lenght of the period... but mixes very well different bit positions

Example: a **good uniform pseudo random-numbers generator** (see NR `Ran` function) would be:

- Compose the output of an LCG mod $2^{64}$ into a 64 bit Marsaglia Xorshift.
- Combine its output (+) with another 64 bit Marsaglia Xorshift with different parameters.
- Combine its output (^) with a Multiply With Carry with base $2^{32}$

Check the NR book for specific choices of the parameters of each generator

Marsaglia Xorshift: Three XORs and three shifts produces a full period (2.

## Sampling from other distributions (general Method)

- Assume that we have a uniform deviate x and calculate a function of it y(x).
- The probability distribution of y: p(y) dy is given by the fundamental transformation law of probabilities:

$$|p(y)\,dy| = |p(x)\,dx|$$

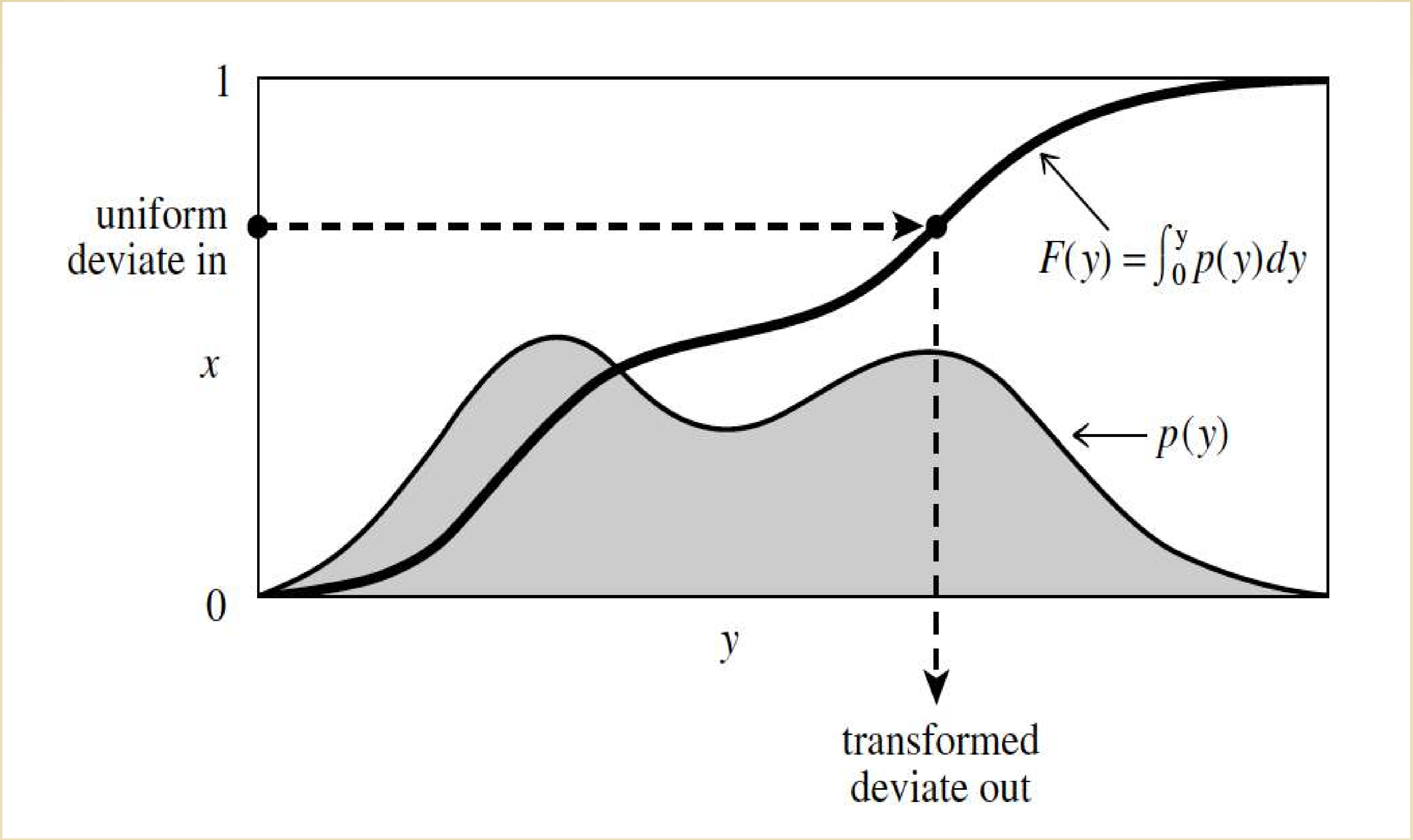
$$p(y) = p(x)\left|\frac{dx}{dy}\right|$$

-

- Then, if we want to generate a distribution of y with p(y)=f(y) from a uniform deviate x (f positive with normalized total probability -integral- to 1) we have to solve for y(x):

$$\frac{dx}{dy} = f(y)$$
$$x = \int f(y)\,dy = F(y)$$
$$y(x) = F^{-1}(x)$$

- Which has the obvious problem of having to calculate the inverse of an integral function.



Geometrical interpretation of the general transformation method to obtain a random deviate from a probability distribution p(y) starting from an uniform deviate x. (From NR, chapter 7)

In general, to generate random numbers from a given distribution is neither easy nor efficient. We will see other ways later. Now we are coming back to see some Monte-Carlo applications.

Exercise: plot the histograms of a uniform and gaussian random number generator for 100, 1000, 10.000 and 100.000 numbers.

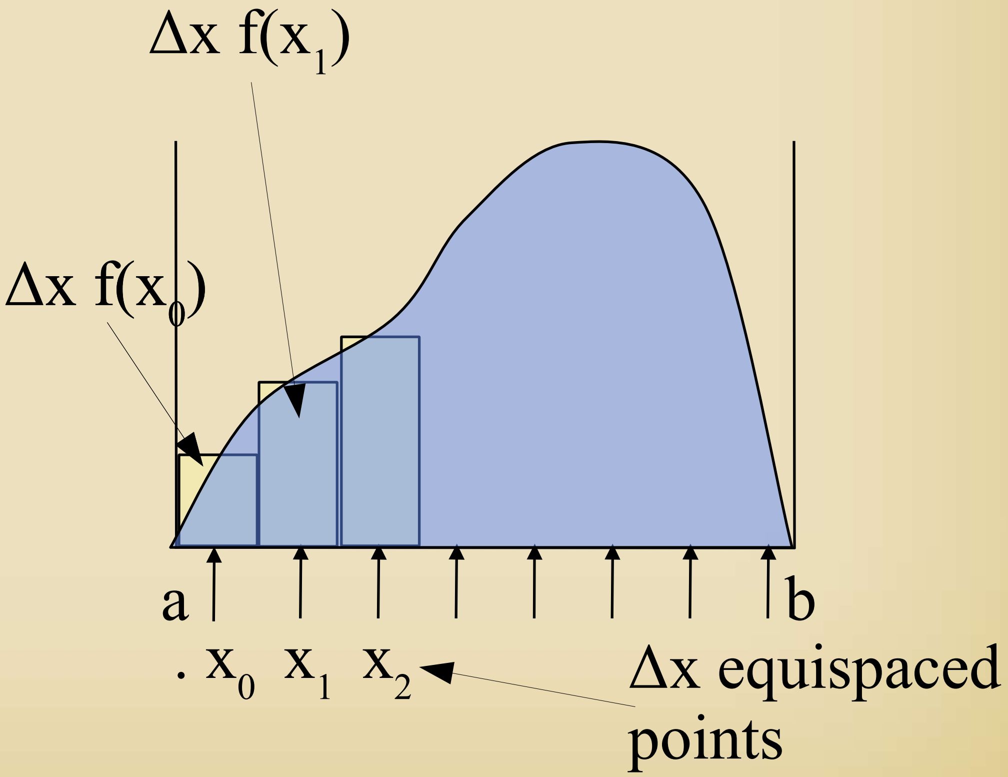
# Monte-Carlo and Numerical Integration

- The typical approaches to the numerical calculation of low dimensional integrals are based on weighted sums of function values obtained at certain fixed abcissas (equispaced in Newton-Cotes formulas or the zeroes of orthogonal polinomials in Gauss formulas)

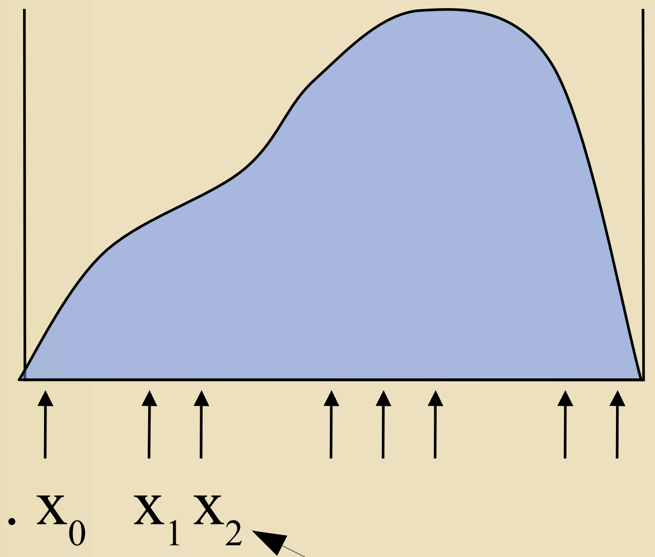
$$I=\int_{x_0=a}^{x_n=b} f(x)\,dx \simeq$$

$$\sum_{i=0}^{n} \Delta x\, f(x_i) = \frac{b-a}{n} \sum_{i=0}^{n} f(x_i)$$

$$(\textit{trapezoidal formula})$$



Let's use a Monte-Carlo style approach: Instead of using fixed abcissas, choose them from a probability distribution π(x) e.g.: a uniform one.



$$I = \int_{x_0=a}^{x_n=b} f(x)\,dx \simeq \frac{b-a}{n} \sum_{i=0}^{n} f(x_i)$$

$$(MC\ formula, x_i\ stochastically\ chosen)$$

Chosen from π(x)= uniform distribution.

## How do these methods compare?

- Equispaced points used as a base for interpolation (Newton-Cotes formulae)
  - Error term of composite Trapezoidal on n points: $O(n^{-2})$
  - Error term of composite Simpson on n points: $O(n^{-4})$
- Error term of MC integration: $O(n^{-1/2})$

## So, is MC integration worth it?

## Let's explore higher dimensional spaces.

- Imagine that we use Newton-Cotes like ideas to produce a multidimensional integration formula.
- e.g.: A Simpson formula in 10 dimensions: A polynomial of degree 2 in each of the 10 variables is fitted in each box of dimension 10.
    - Each polynomial has terms $x_1^{\square}x_2^{\square}x_3^{\square}x_4^{\square}x_5^{\square}x_6^{\square}x_7^{\square}x_8^{\square}x_9^{\square}x_{10}^{\square}$ where each $^{\square}$ is either 0,1 or 2. So, the polynomial has $3^{10}$=59.049 coeficients.
    - If we are doing the integral in the hyper-volume $[0,1]^{x10}$ and decide to divide each interval in 10 pieces, we have $10^{10}$ hyper-boxes
    - And in each hyper-box we have to evaluate a polynomial of 59.049 coefficients.
- This is obviously unfeasible

- The basic theorem of Monte-Carlo integration using **n** uniformly distributed random points $x_0 \ldots x_{n-1}$ over a volume V:

$$\int_V f\, dV \simeq V \langle f \rangle \mp V \sqrt{\frac{\langle f^2 \rangle - \langle f \rangle^2}{n}}$$

$$\langle f \rangle \equiv \frac{1}{N} \sum_{i=0}^{n-1} f(x_i)$$

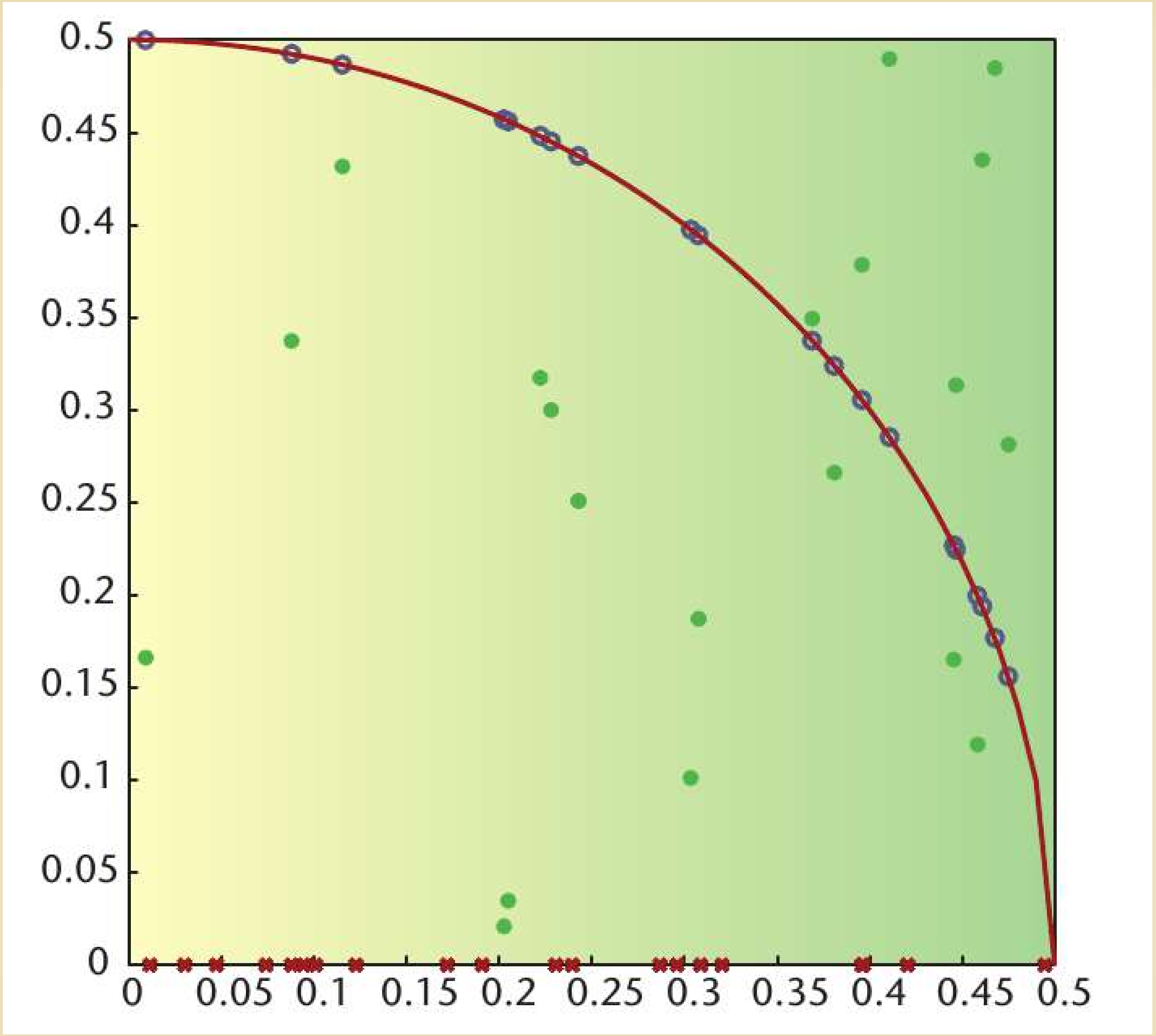
$$\langle f^2 \rangle \equiv \frac{1}{N} \sum_{i=0}^{n-1} f^2(x_i)$$

- What's important here?: The error (that is just a one-sigma estimate) goes like $n^{-1/2}$ … and it **does not depend on the dimension.**

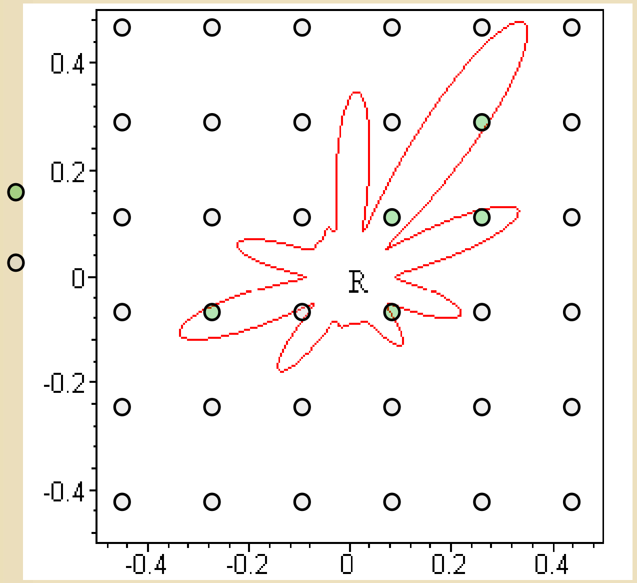
Note how the error is an statistical estimate. Better chosen $x_0$... $x_{n-1}$ can reduce the variance and improve the precision.

- Importance Sampling: later.

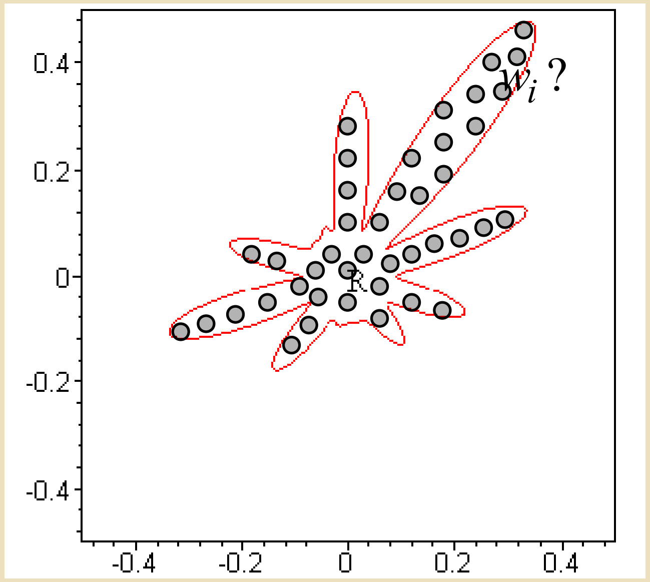
- Exercise: Calculate the area of a circle or radius 0.5
  - Using a Newton-Cotes formula: e.g.: mid-point Trapezoidal formula
  - Using a Monte-Carlo method to estimate the ratio of areas inside and outside the circle.
  - Using the simple Monte-Carlo method seen above (basic MC theorem).
  - Study the convergence and cost of the different methods.


0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5

## Estimate an integral over a region of odd shape.



Suboptimal sampling



$$r^2 = \frac{\int_{-0.5}^{0.5} dx \int_{-0.5}^{0.5} (x^2+y^2)\,\delta(x,y)\,dy}{\int_{-0.5}^{0.5} dx \int_{-0.5}^{0.5} \delta(x,y)\,dy}$$

$$\delta(x,y) = 1\ if\ (x,y) \in R;\ 0\ otherwise$$

## Back to a More Formal View: The Problems of Monte-Carlo:

- How to sample from $P(x)$?
  - Correct samples from $P(x)$ should come from places where $P(x)$ is big. How do we know which places are these without actually sampling $P(x)$ at every point?
    - If we choose to sample in a regular grid, the number of sample points grow as the power of the dimension of the space: If we use a basic size of 50 sample points in each dimension and have a space of dimension 100, $50^{100}$ points have to be sampled... more than the total number of atoms in the universe.
    - And systems of such dimensions are quite common...

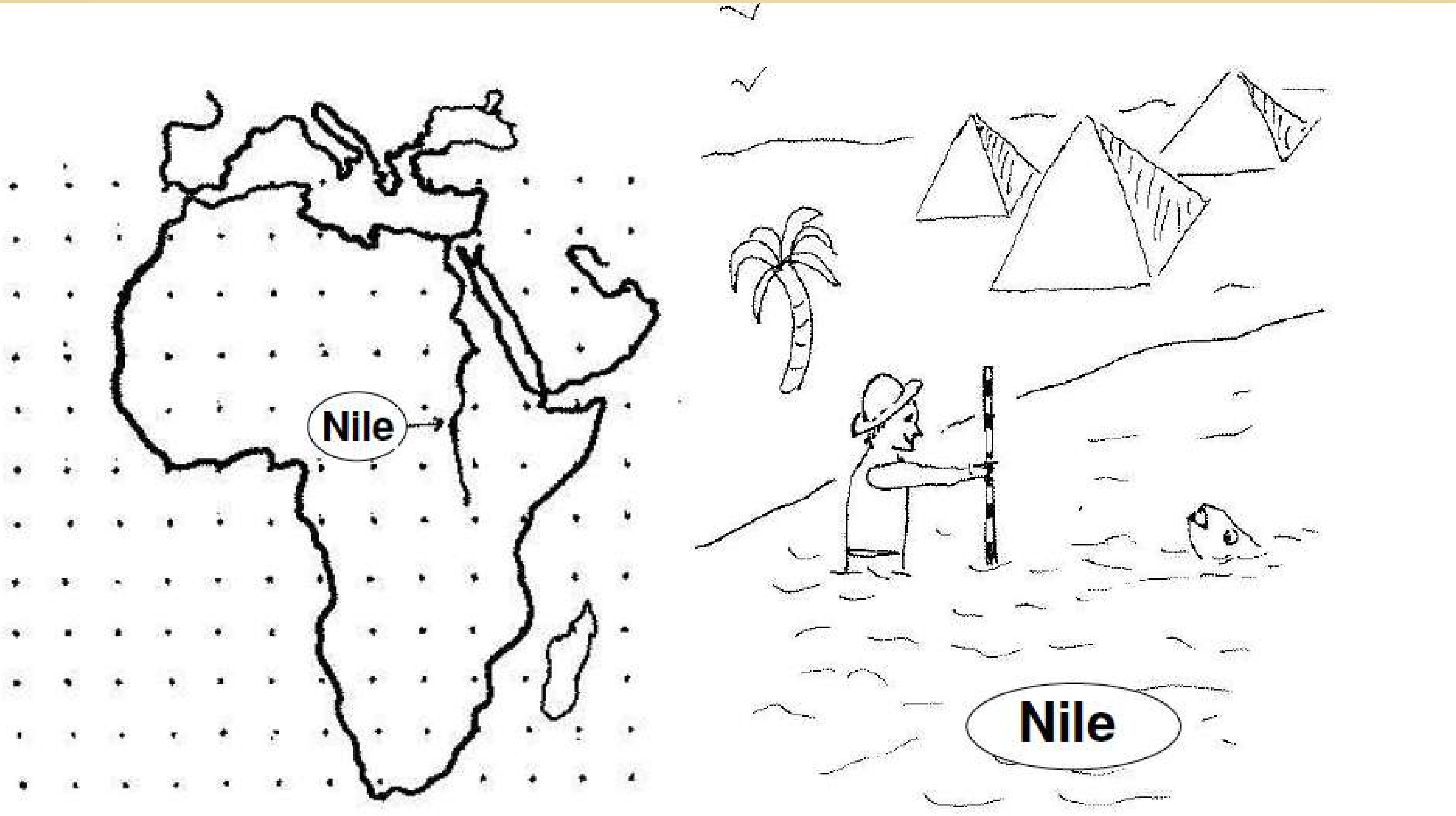


Figure 1. Measuring the depth of the Nile: a comparison of conventional quadrature (left), with the Metropolis scheme (right).

From D. Frenkel, „Introduction to Monte-Carlo Methods“
NIC Series, Vol. 23, ISBN 3-00-012641-4, pp. 29-60, 2004

**Example:** Calculate the average concentration of plankton in a lake assuming that the concentration at a point ($x,y$) is $\varphi(x,y)$ and that the depth of the lake is $P(x,y)$.

The required value is:

$$\langle \phi(x,y) \rangle \equiv \frac{1}{Z} \int dx\, dy\, P(x,y)\, \phi(x,y)$$

$$\textit{with}$$

$$Z = \int dx\, dy\, P(x,y)$$

- How to sample from $P(x,y)$?
  - If we know nothing about the depth of the lake, we have to sample at every possible ($x,y$) [within a given spatial resolution]... we have to discover every submerged valleys even in order to estimate Z.

**Sampling techniques:**

- Uniform sampling.
- Importance sampling.
- Rejection sampling.
- Metropolis-Hastings: Markov Chain Monte Carlo MCMC

## Uniform Sampling:

- Let us try to estimate the expectation value of $\phi(x)$
- Since we cannot visit all possible points in space state, lets do it in the simplest way by drawing random samples uniformly and evaluating P(***x***) in those points:
    - The normalizing constant could be approximated by:

$$Z_R = \sum_{r=1}^{R} P(\boldsymbol{x}^{(r)})$$

- And then the estimate of $\Phi = \langle \phi(\boldsymbol{x}) \rangle \equiv \int d^N \boldsymbol{x}\, P(\boldsymbol{x}) \phi(\boldsymbol{x})$ would be calculated as:

$$\hat{\Phi} = \sum_{r=1}^{R} \phi(\boldsymbol{x}^{(r)}) \frac{P(\boldsymbol{x}^{(r)})}{Z_R}$$

## Uniform Sampling:

- Is this of any help?
    - Only if the samples hit the “typical set”, where the $\boldsymbol{x}$ corresponding to the highest valued $P(\boldsymbol{x})$ lie, the estimate is good.
        - For highly dimensional $P(\boldsymbol{x})$, the probability of drawing x from the typical set is very low... unless $P(\boldsymbol{x})$ is uniform itself.
        - Uniform sampling, hence it is not likely to be very useful.

## Importance Sampling:

- Lets assume that we are able to ***evaluate*** the target density $P(\boldsymbol{x})$, at least to within a multiplicative constant $Z$, i.e.: we can evaluate $P^*(\boldsymbol{x})$ such that:

$$P(x) = P^*(x)/Z$$

- But we cannot sample from it. Instead we can sample from a simpler *sampler density function* $Q(x)$ and that we can evaluate it up to a constant, i.e: we can evaluate $Q^*(\boldsymbol{x})$ such that:

$$Q(x) = Q^*(x)/Z_Q$$

## Importance Sampling:

- Then we generate samples $\{x^{(r)}\}_{r=1}^{R}$ L from $Q(\boldsymbol{x})$ Were they obtained from $P(\boldsymbol{x})$, then we could estimate $\boldsymbol{\Phi}$ directly as:

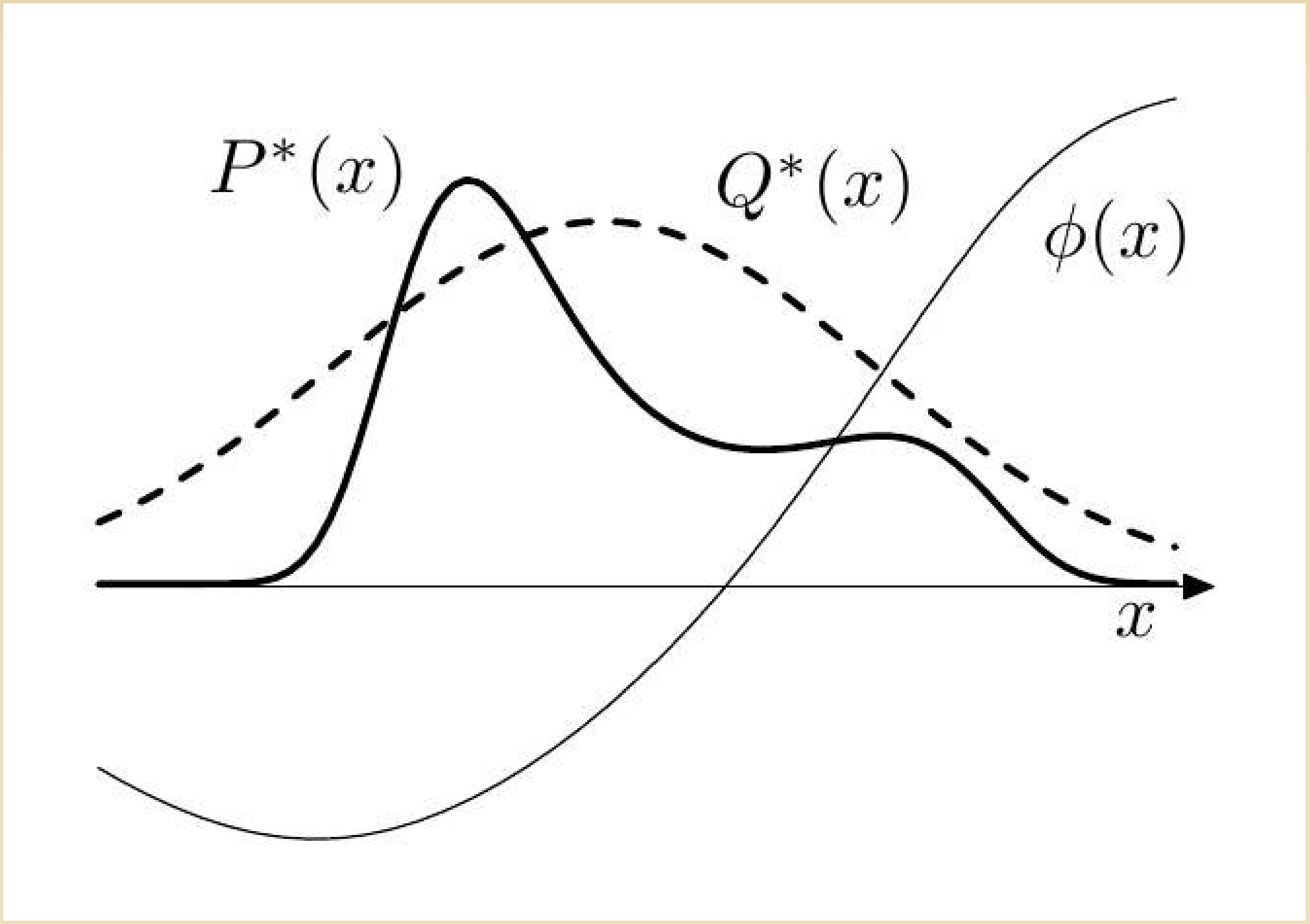
$$\Phi = \frac{1}{R}\sum_{r=1}^{R} \phi(\boldsymbol{x}^{r})$$

- However, by sampling $Q(\boldsymbol{x})$, now we will have cases in which $Q(\boldsymbol{x}) > P(\boldsymbol{x})$ and those would be over weighed if we use directly the formula above. Cases in which $P(\boldsymbol{x})>Q(\boldsymbol{x})$ would be under weighed. So, now we define the weights:

$$w_r = \frac{P^*(x^{(r)})}{Q^*(x^{(r)})}$$

- These weights adjust the importance of each sample and we can use as the estimate:

$$\hat{\Phi} = \frac{\sum_r w_r \phi(x^{(r)})}{\sum_r w_r}$$



D. Mackay.

Functions involved in importance sampling. The function for which we have to estimate the average ϕ($x$) over the probability function $P(\boldsymbol{x})$. The function $P^*(\boldsymbol{x}) = Z\,P(\boldsymbol{x})$ that we can evaluate and the function $Q^*(\boldsymbol{x}) = Z_Q\, Q(\boldsymbol{x})$, where $Q(\boldsymbol{x})$ is the sampler density function that we can Sample from.

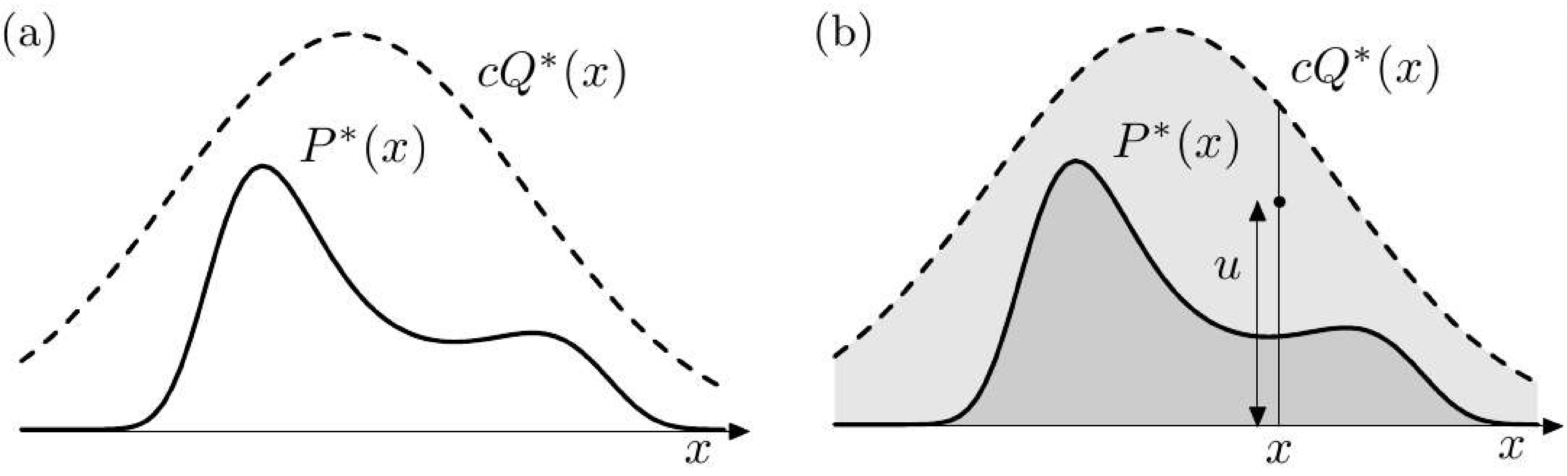
- In practise it is difficult to estimate the reliability of the result. The weighs are not necessarily a good guide.
- Unless $Q(x)$ is a good approximation of $P(x)$ most of the samples will have a negligible weight.

## Rejection Sampling:

- Again, we assume that we have a difficult to sample $P(x)$ and a simpler $Q(x)$ that we can sample and evaluate up to the usual multiplicative constant $Z_Q$ .
- We also assume that we know a constant $c$ such that:

$$c\,Q^*(x) > P^*(x)\ \forall\ x$$

- Procedure:
    - Generate one random number, $x$, from $Q(x)$ and evaluate $cQ^*(x)$.
    - Generate another random number, $u$, but now in $[0,cQ^*(x)]$.
    - Evaluate $P^*(x)$. If $u > P^*(x)$, then reject $x$, accept otherwise.
        - Accept means to add $x$ to $\{x^{(r)}\}$
        - $u$ is discarded.
    - Continue till a big enough number of samples $\{x^{(r)}\}$ has been obtained.
        - In high dimensional problems it is likely that $c$ will become large in order to make $cQ^*(x) > P^*(x)$ for all $x$ and the frequency of rejection will be large unless $Q(x)$ is a good approximation to $P(x)$
        - The rejection ratio is proportional to the volume of $P(x)$ over $cQ(x)$. For normalized $P(x)$ and $Q(x)$, this is $1/c$. Hence, if $c$ is big the rejection ratio is very high and the performance is poor.



D. Mackay

Functions involved in rejection sampling: $P^*(\boldsymbol{x})$ and $Q^*(\boldsymbol{x})$ are as in importance sampling. $c$ is the constant that guarantees that $cQ^*(x) > P^*(x)$. $x$ is the first random number generated from $Q(\boldsymbol{x})$ and $u$ the second, uniform in $[0, cQ^*(x)]$. $x$ is accepted or rejected depending on wheter is inside the dark grey region (accepted) or outside (rejected).

## •Metropolis-Hastings method (1).

Importance and rejection sampling work badly unless $Q(x)$ closely tracks $P(x)$. M-H adjusts $Q(x)$ at each step $t$ : $Q(x) \equiv Q(x;x^{(t)})$

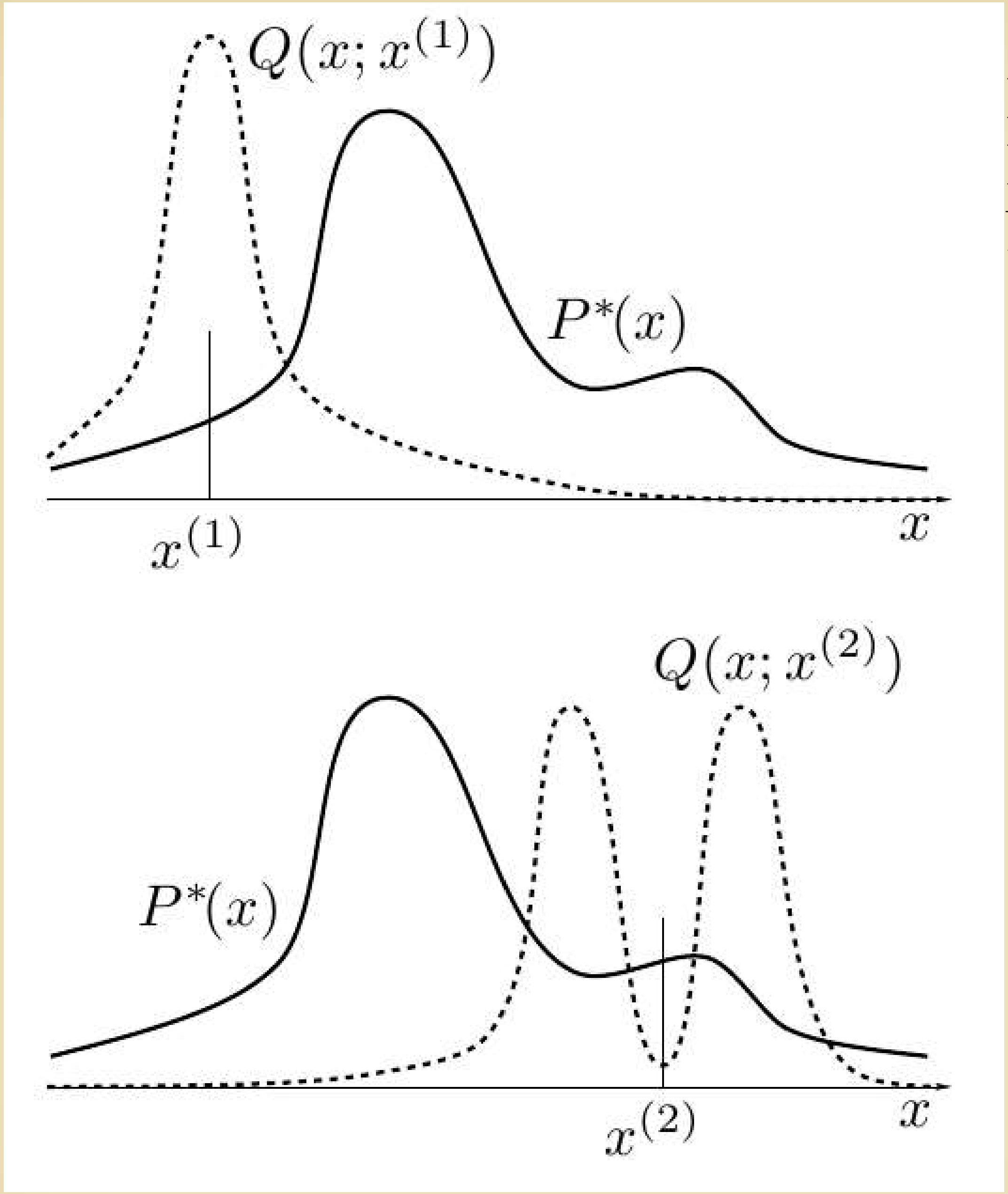
- $Q(x;x^{(t)})$ could be, for example, a gaussian centered around $x^{(t)}$

- As before, we assume that we can evaluate $P^*(\boldsymbol{x})$ for any $\boldsymbol{x}$. From $Q(x;x^{(t)})$ we obtain a new $x$.
- The new $x$ is accepted if $a \geq 1$, where

$$a = \frac{P^*(x)}{P^*(x^{(t)})} \times \frac{Q(x^{(t)};x)}{Q(x;x^{(t)})}$$

## •**Metropolis-Hastings method (2)**.

- Otherwise, it is accepted with probability *a*
  - If it is accepted $x^{(t+1)} = x$
  - And if not, $x^{(t+1)} = x^{(t)}$
  - Note that, differently from rejection sampling, rejected points do influence on the list of samples since they are not discarded if rejected. The current state is rewritten in the list if rejected.
  - If Q is symmetric, the second term in the quotient is 1. This is what has been traditionally called a Metropolis method.



Functions involved in the Metropolis-Hasting method. Note how Q changes with the current state.

Mackay.

- For any $Q(x';x) > 0$, for all $x,x'$, it can be demonstrated that the distribution of $x^{(t)}$ tends to $P(x)$ as $t \to \infty$
- M-H method is an example of a Markov Chain Monte Carlo (MCMC).
    - In rejection sampling the accepted points are independent samples of the desired distribution.
    - In MCMC a sequence of states is generated. Each state is dependent on the previous one, hence a MCMC must run for some time in order to generate samples that are effectively independent.
    - It is possible to greatly increase the efficiency of the MC methods described. See Hamiltonian MC, overrelaxation and Simulated annealing.

## Importance Sampling, back to examples

Let us go back to the evaluation of the integral. What is the best choice of function to sample from?

- When we use importance sampling with a function π(x) to evaluate the integral of a function f(x) what we are doing is to calculate:

$$\langle f(x)\rangle = \int_V f(x)\,dx \simeq$$

$$\simeq \langle f(x)\rangle_{\pi(x)} = \int_V \frac{f(x)}{\pi(x)}\pi(x)\,dx$$

- Whether we get something better will depend on how good is the sampling. If the new sampling using π(x) samples better the „typical set“ that has the largest contribution, then we will have a faster convergence.
- What about the error?

$$\sigma^2=\frac{1}{n}(\langle f(x)^2\rangle-\langle f(x)\rangle^2)$$

becomes under importance sampling in

$$\sigma^2_{\pi(x)}=\frac{1}{n}(\int_V (\frac{f(x)}{\pi(x)})^2 \pi(x)\,dx-(\int_V \frac{f(x)}{\pi(x)}\pi(x)\,dx)^2)$$

- The error is minimized (is cero) when  π(x)= cte.*f(x)

- **Exercise:** use importance sampling to estimate the area of the circle, like in the previous example.
  - Use a piecewise constant π(x)
  - Use a linear interpolation between a few sample points in the circumference.
    - Remember:

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

if we want $p(x) = ax$ on $x \in (0,1)$ then

$\int ax\,dx = \int p(y)\,dy$ If $p(y)$ is uniformly distributed then:

$$\frac{1}{2}ax^2 + c = y$$

since we want $y=0$ when $x=0$ and $y=1$ when $x=1$, then $a=2$ and $c=0$. So, if $y$ is uniform, then $\sqrt{y}$ is distributed linearly.

- But to choose π(x)= cte.*f(x) is not a possibility: Note that we have to be able to normalize the probability. i.e.: we are supposed to be able to calculate:

$$V = \int_V \pi(x)\,dx$$

but if $\pi(x) = cte. \times f(x)$ this is equivalent to calculate:

$\int_V f(x)\,dx$ which is what we want to calculate in first place

## Importance Sampling, back to examples

- Exercise: plot the different values of the area of the circle obtained with the different methods as a function of the number of samples.

## Can we sample without a explicit function? Dynamic (pseudo) Monte-Carlo.

- **Yes:** Using Markov Chains we can use the concept of a ***limiting distribution*** that appears as a result of a designed procedure that stocastically guides our selection of sample points over S.
- (Recall the concepts introduced in the first slides)
- We will exemplify those ideas here.

## So, sampling?

- Dynamic (pseudo)Monte-Carlo:
  - Invent a stochastic process within S that has π as its ***unique equilibrium distribution***
    - Also called stationary or limiting distribution.
  - The process is simulated till the system is in equilibrium. Then, the time averages converge to π-averages.
  - The simulation process is just a computational device; no correspondence with reality is required.
  - This process is taken to be a ***Markov process***.

- Assume S is discrete: Markov Chain in S:
  - A sequence of random variables $X_0$, $X_1$, $X_2$ … in S such that the transitions $X_t \rightarrow X_{t+1}$ are statistically independent.
- A Markov chain is specified by:
  - A initial probability distribution α in S. $P(X_0 = x) = \alpha_x$
  - A transition probability matrix $P=\{p_{xy}\}_{x,y \in S}=\{p(x\rightarrow y)\}$
    - $p_{xy} \geq 0$ for all x,y
    - $\sum_y p_{xy} = 1$ for all x
    - $\mathbf{P}(X_{t+1}= y \mid X_t = x) = p_{xy}$
- The Markov Chain is specified by the joint probabilities:

$$\mathbf{P}(X_0=x_0,X_1=x_1,\ldots,X_n=x_n) = \alpha_{x0}p_{x0x1}p_{x1x2}\ldots p_{xn-1xn}$$

- The n-step transition probabilities:

$$p^{(n)}_{xy} = \mathbb{P}(X_{t+n} = y \mid X_t = x)$$

- $p^{(0)}_{xy} = \delta_{xy}$, $p^{(1)}_{xy} = p_{xy}$,... in general $\{p^{(n)}_{xy}\}$ are the matrix elements of $P^n$.

- A Markov chain is ***irreducible*** if any state can be reached from any other (ergodicity) i.e.:

  For each $x,y \in S$ there is $n \geq 0$ for which $p^{(n)}_{xy} > 0$.

- The ***period*** $d_x$ of a state x is the gcd of the numbers n for which $p^{(n)}_{xx} > 0$.
  - If $d_x=1$ the state x is ***aperiodic***.
  - In an irreducible chain all states have the same period.

## Convergence to equilibrium.

- A probability measure $\pi = \{\pi_x\}_{x\in S}$ is a ***stationary distribution*** for the Markov Chain P if:

$$\sum_x \pi_x p_{xy} = \pi_y \text{ for all } y$$

- Let P be the transition probability matrix of an irreducible Markov chain of period d, then:
  - If a stationary probability distribution exists, then it is unique and $\pi_x > 0$ for all x.
  - If P is aperiodic $\lim_{n\to\infty} p^{(n)}_{xy} = \pi_y$

- Now, **to set up a dynamic (pseudo) Monte-Carlo** method to sample from π:

**Design a transition probability matrix** $\{p_{xy}\}$ satisfying:

- ***Irreducibility***. (For each x,y ϵ S there is n ≥ 0 for which $p^{(n)}_{xy} > 0$)
- ***Stationarity*** of π. ($\sum_x \pi_x p_{xy} = \pi_y$ for all y ϵ S. Stronger alternative: detailed balance: For all x,y ϵ S $\pi_x p_{xy} = \pi_y p_{yx}$)

Then the Markov chain P produces a legitimate way of estimating π-averages.

Any starting point will do and the system is guaranteed to converge to equilibrium when $t \to \infty$. „Time“ averages will converge to π-averages with convergence √n.

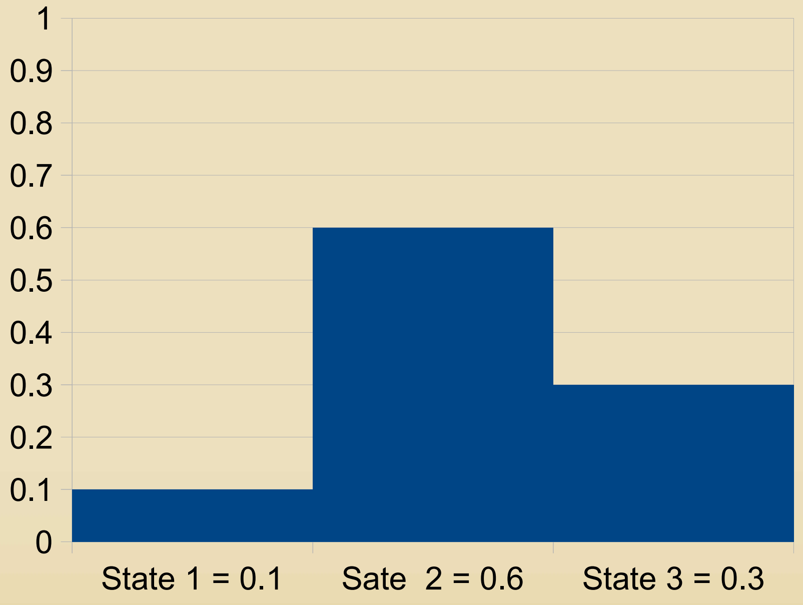
- A Markov process has no memory: The selection of the next state depends exclusively on the present one.
- It is fully specified by a set of transition probabilities $\pi_{ij}$ that dictate the probability to end in state j when starting in state i.
  - Example: a transition probability matrix (TPM) collecting all transitions probabilities in a system with three states is:

$$\Pi = \begin{pmatrix} \pi_{11} & \pi_{12} & \pi_{13} \\ \pi_{21} & \pi_{22} & \pi_{23} \\ \pi_{31} & \pi_{32} & \pi_{33} \end{pmatrix}$$

    - Note that all entries are probabilities, hence they are non-negative and less than 1 (normalization)
    - Note that the sum of each row (the total probability from a given initial estate to go to any possible final state, including itself) must be 1.

## The Limiting (stationary, equilibrium) Distribution.

- Imagine that we start in any of the possible states.
- Now we follow the Markov chain by selecting the next state from Π.
- Repeat the process many times and then plot the final distribution:



Limiting Distribution: $\pi=(\pi_1,\pi_2,\pi_3)=(0.1,0.6,0.3)$

- When we follow a Markov chain we are actually sampling from this limiting distribution.
- What we know in Monte-Carlo is this limiting distribution from which we want to sample.
- We need to build a Markov chain that leads us to the limiting distribution from which we want to sample.

- To understand how to build the desired limiting distribution, lets study the Transition Probability Matrix, Π.
- What is the meaning of $\Pi^2$?

$$\Pi^2 = \begin{pmatrix} \pi_{11} & \pi_{12} & \pi_{13} \\ \pi_{21} & \pi_{22} & \pi_{23} \\ \pi_{31} & \pi_{32} & \pi_{33} \end{pmatrix} \times \begin{pmatrix} \pi_{11} & \pi_{12} & \pi_{13} \\ \pi_{21} & \pi_{22} & \pi_{23} \\ \pi_{31} & \pi_{32} & \pi_{33} \end{pmatrix} =$$

$$= \begin{pmatrix} \pi_{11}\pi_{11} + \pi_{12}\pi_{21} + \pi_{13}\pi_{31} & \dots \\ \pi_{21}\pi_{11} + \pi_{22}\pi_{21} + \pi_{23}\pi_{31} & \dots \\ \pi_{31}\pi_{11} + \pi_{32}\pi_{21} + \pi_{33}\pi_{31} & \dots \end{pmatrix}$$

- The element (1,1) is actually the probability to stay in state 1 after 2 moves. (2,1) is the probability to move from state 2 to state 1 in two moves...

- In general $\Pi^n$ is the matrix that defines the probability to go from one state to another in exactly n steps.

$$\Pi^n = \begin{pmatrix} \pi_{11}^{(n)} & \pi_{12}^{(n)} & \pi_{13}^{(n)} \\ \pi_{21}^{(n)} & \pi_{22}^{(n)} & \pi_{23}^{(n)} \\ \pi_{31}^{(n)} & \pi_{32}^{(n)} & \pi_{33}^{(n)} \end{pmatrix}$$

- Note that this matrix defines $\pi_{ij}^{(n)}$, for example, the probability to go from state 3 to 2 in exactly 3 steps:

$$\begin{aligned} \pi_{32}^{(3)} = &\left(\pi_{31}\pi_{11} + \pi_{32}\pi_{21} + \pi_{33}\pi_{31}\right)\pi_{12} \\ &+ \left(\pi_{31}\pi_{12} + \pi_{32}\pi_{22} + \pi_{33}\pi_{32}\right)\pi_{22} \\ &+ \left(\pi_{31}\pi_{13} + \pi_{32}\pi_{23} + \pi_{33}\pi_{33}\right)\pi_{32} \end{aligned}$$

- If a ***limiting distribution*** exist it must be ***independent of the initial state.***
  - Otherwise, we cannot guarantee that the sampling is going to follow the desired distribution.
  - The goal is to build a Markov chain that samples the important regions (typical set) of our system.
  - For this to happen, the Markov process must be ergodic (irreducible) : any state is reachable from any other, although maybe in many steps.
    - And $\sum_x \pi_x p_{xy} = \pi_y$ for all $y \in S$. Stronger alternative: detailed balance: For all $x,y \in S$ $\pi_x p_{xy} = \pi_y p_{yx}$

## In our 3 states example.

- Starting from any of the three possible states (denoted by the superscript $^{(0)}$), namely: $\pi_1^{(0)}=(1,0,0)$, $\pi_2^{(0)}=(0,1,0)$, $\pi_3^{(0)}=(0,0,1)$.
- Then $\pi_i^{(n)}=\pi_i^{(0)}\Pi^n= (\pi_{i1}^{(n)},\pi_{i2}^{(n)},\pi_{i2}^{(n)})$ is a vector with the probabilities for ending at states 1,2 or 3, respectively, after n steps when starting in state i. e.g. starting in state 1:

$$\pi_1^{(n)}=\pi_1^{(0)}\Pi^n=(1,0,0)\begin{pmatrix}\pi_{11}^{(n)} & \pi_{12}^{(n)} & \pi_{13}^{(n)}\\ \pi_{21}^{(n)} & \pi_{22}^{(n)} & \pi_{23}^{(n)}\\ \pi_{31}^{(n)} & \pi_{32}^{(n)} & \pi_{33}^{(n)}\end{pmatrix}=(\pi_{11}^{(n)},\pi_{12}^{(n)},\pi_{13}^{(n)})$$

- The limiting distribution π is stationary, independent of the initial state and corresponds to $n \to \infty$. In our example: $\pi_1^{(\infty)} = \pi_2^{(\infty)} = \pi_3^{(\infty)} = \pi$
- Note that under stationary conditions, π is a left eigenvector of Π:

$$\pi = \lim_{n\to\infty}\left(\pi_i^{(0)}\Pi^n\right) = \left(\lim_{n\to\infty}\left(\pi_i^{(0)}\Pi^{n-1}\right)\right)\Pi = \pi\,\Pi$$

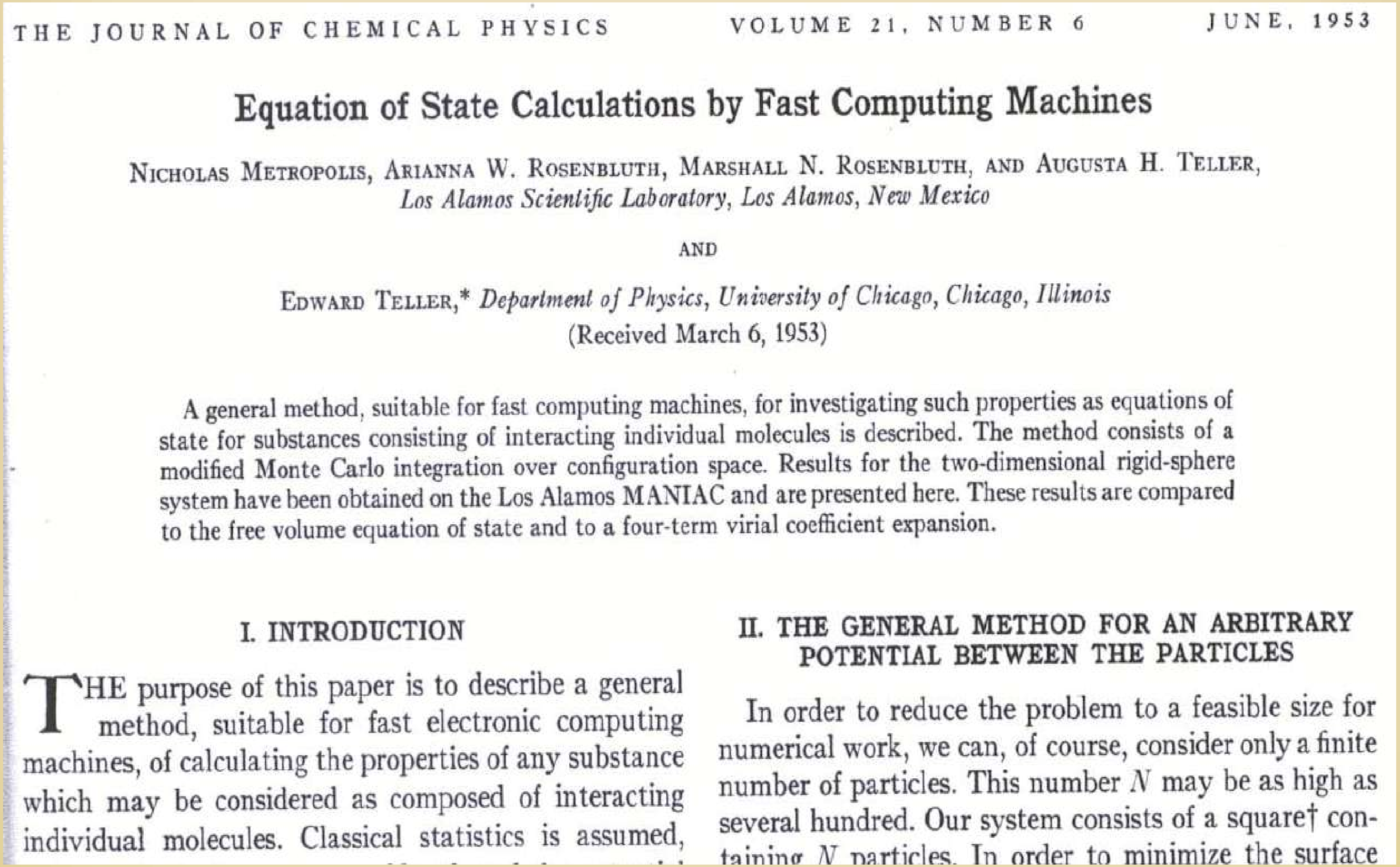
- Element by element: $\pi_i = \sum_j \pi_j \pi_{ji}$

- The eigenvector equation is satisfied if we impose:
  - ***Detailed balance*** $\pi_i \pi_{ij} = \pi_j \pi_{ji}$
  - This is also known as „microscopic reversibility“
    - A way to think about it: if we are in the limiting distribution, the Markov chain moves do not have to disturb it: the probabilities to leave one state must be equal to the probabilities to enter that state.
  - Detailed balance is actually not required (it is only a sufficient condition) in order to have a limiting distribution, but it is an easy way not to run into trouble.

$$\pi_i = \sum_j \pi_j \pi_{ji} \underset{\substack{\uparrow \\ \text{Detailed balance}}}{=} \sum_j \pi_i \pi_{ij} = \pi_i \sum_j \pi_{ij} = \pi_i \ \text{(i.e.: stationary)}$$

- Building a Transition Probability Matrix that satisfies detailed balance is very useful because it involves only two probabilities at a time, not the full eigenvector problem, that would require the evaluation of all transition probabilities for all states at each step.
- In this way, the calculation of the transition probabilities can be done as required in the specific step in Markov chain. Only the information for the states actually visited has to be calculated.
- One has still to make sure that the limiting distribution equals the desired one.

- The Metropolis Algorithm



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*

(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

### I. INTRODUCTION

THE purpose of this paper is to describe a general method, suitable for fast electronic computing machines, of calculating the properties of any substance which may be considered as composed of interacting individual molecules. Classical statistics is assumed,

### II. THE GENERAL METHOD FOR AN ARBITRARY POTENTIAL BETWEEN THE PARTICLES

In order to reduce the problem to a feasible size for numerical work, we can, of course, consider only a finite number of particles. This number $N$ may be as high as several hundred. Our system consists of a square† containing $N$ particles. In order to minimize the surface

- The Metropolis algorithm is a very convenient way to produce a Markov chain that satisfies detailed balance.
  - Instead of doing an „ordinary“ Monte-Carlo in which the decision to go to a certain state is directly determined by the Transition Probability Matrix.
  - The decision to go to a certain state is divided in two:
    - A move according to a probability to go from one state to other taken from an „underlying transition matrix“ that can be conveniently chosen.
    - An acceptance probability of the move. This acceptance is done in a way that, together with the underlying transition matrix, satisfies detailed balance.
    - There are many ways to specify the total transition probabilities, Metropolis is just one choice.

- Metropolis Recipe:
  - Starting in a given state i and with a certain probability $\tau_{ij}$ (given by the underlying probability distribution ) choose a trial state j for the move.
  - Accept j as the new state if $\pi_j > \pi_i$, else accept j as the new state with probability $\chi=\tau_{ji}\pi_j/\tau_{ij}\pi_i$
  - If j is rejected, the original starting state i is kept and used as the next step in the Markov chain ($\pi_{ii}\neq 0$)
  - **Note:** it is usual to choose a symmetric underlying probability $\tau_{ij}=\tau_{ji}$ This is not necessary if the assymmetry is taken into account in the acceptance criterion.

- Does it satisfy detailed balance? The probabilities are:

$$\pi_{ij} = \tau_{ij}\, min(1, \chi)$$
$$\pi_{ji} = \tau_{ji}\, min(1, 1/\chi)$$
$$\pi_{ii} = 1 - \sum_{j \neq i} \pi_{ij}$$

- Using them in the definition of detailed balance:

$$\pi_i \pi_{ij} = \pi_j \pi_{ji}$$
$$\pi_i \tau_{ij}\, min(1, \chi) = \pi_j \tau_{ji}\, min(1, 1/\chi)$$
$$\pi_i \tau_{ij} \chi = \pi_j \tau_{ji}$$

since $\chi = \pi_j \tau_{ji} / \pi_i \tau_{ij}$ it is obviously satisfied

- Example: Let us illustrate the Metropolis method with the calculation of an average over an odd shaped area A.

$$\langle r^2 \rangle = \frac{\int_{-0.5}^{0.5} dx \int_{-0.5}^{0.5} dy \left(x^2 + y^2\right) s(x,y)}{\int_{-0.5}^{0.5} dx \int_{-0.5}^{0.5} dy\, s(x,y)} = \frac{\langle r^2 s \rangle_A}{\langle s \rangle_A}$$

where $s(x,y) = 1$ if $(x,y) \in A$ and $0$ otherwise

- Case 1: Let us choose first as limiting distribution π a uniform one, i.e.:

$$\pi_1(x,y) = \frac{s(x,y)}{q_1}$$

where $q_1$ is the normalization constant of $\pi_1$ over $A$

- With this sampling distribution, the importance sampled integral is then:

$$\langle r^2 \rangle = \frac{\left\langle \frac{r^2 s}{\pi_1} \right\rangle_{\pi_1}}{\left\langle \frac{s}{\pi_1} \right\rangle_{\pi_1}} = \frac{\langle r^2 q_1 \rangle_{\pi_1}}{\langle q_1 \rangle_{\pi_1}} = \langle r^2 \rangle_{\pi_1}$$

- Note that this is exactly what we have requested: we evaluate the integral by just adding the $r^2$ values for the points that are uniformly generated inside the region A.

- Case 2: Let us choose another sampling that weighs more the points with higher value of $r^2$. In this case we can use as limiting distribution $\pi$:

$$\pi_2(x, y) = r^2 \frac{s(x, y)}{q_2}$$

Again $q_2$ is the normalization constant for $\pi_2$ over $A$

- In this case we have:

$$\langle r^2 \rangle = \frac{\left\langle \frac{r^2 s}{\pi_2} \right\rangle_{\pi_2}}{\left\langle \frac{s}{\pi_2} \right\rangle_{\pi_2}} = \frac{\langle q_2 \rangle_{\pi_2}}{\langle q_2 / r^2 \rangle_{\pi_2}} = \frac{q_2}{q_2 \langle 1/r^2 \rangle_{\pi_2}} = \frac{1}{\langle r^{-2} \rangle_{\pi_2}}$$

- This means that we calculate the average of the reciprocal of the $r^2$ for our sampled points to obtain the solution as its reciprocal

- How to apply the Metropolis algorithm for these cases?
  - Given a point in A, generate a new point in its proximity.
    - Let us do in a square region, with a size defined by a given $\delta$, around the initial point $(x,y)$. The new point would be:

$$x_{\text{new}} = x + \delta \times rand(-1,1)$$
$$y_{\text{new}} = y + \delta \times rand(-1,1)$$

  - Accept this point with a probability $\min(1, \pi_{\text{new}}/\pi_{\text{old}})$.
    - For the Case 1, $\pi=\pi_1$ This is: $\dfrac{\pi_{1\,\text{new}}}{\pi_{1\,\text{old}}} = \dfrac{s_{\text{new}}/q_1}{s_{\text{old}}/q_1} = \dfrac{s_{\text{new}}}{s_{\text{old}}}$
    - This means that we accept all moves that leave the state inside A and reject all that lie outside.

- For the Case 2, $\pi=\pi_2$ This is:
  - Reject all trials that move outside A.
  - If the move ends inside A, accept it with a probability $\min(1, r^2_{new}/r^2_{old})$.
  - Note how this gives more weight to points with larger r.
- Note that the symmetry of the underlying transition matrix is guaranteed in the way the selection of the state is done through the parameter $\delta$: This keeps the area of the square in which the new point is selected constant: The probability of selecting a point j from point i is inversely proportional to the area of this square ($4\delta^2$) and this is the same if we are at point j and go „back“ to point i.

- Note how conveniently the normalization constants (here $q_1$ and $q_2$) cancel. This allows us to evaluate average values over the given region. This is not the case if we want to evaluate the Area of region R (or (hyper)volume) itself. In this case:

$$A=\int_{-0.5}^{0.5} dx \int_{-0.5}^{0.5} dy\, s(x,y)=\langle s\rangle_R$$

- To fix ideas, imagine that we use Case 1: $\pi=\pi_1=s/q_1$ then

$$A=\langle \frac{s_1}{\pi_1}\rangle_{\pi_1}=\langle q_1\rangle_{\pi_1}=q_1$$

- So we have to know the value of $q_1$, but its definition is exactly the same that the integral that we want to calculate.

- Exercise:
  - Use a simple Monte-Carlo algorithm to solve the Travelling Salesman Problem (e.g. a move could be to take a pair of cities and interchange them if this lowers the total distance. Note: Matlab has a simple Monte-Carlo for the TSP.
  - Use a Simulated Annealing on the TSP and compare its performance with the simple Monte-Carlo.

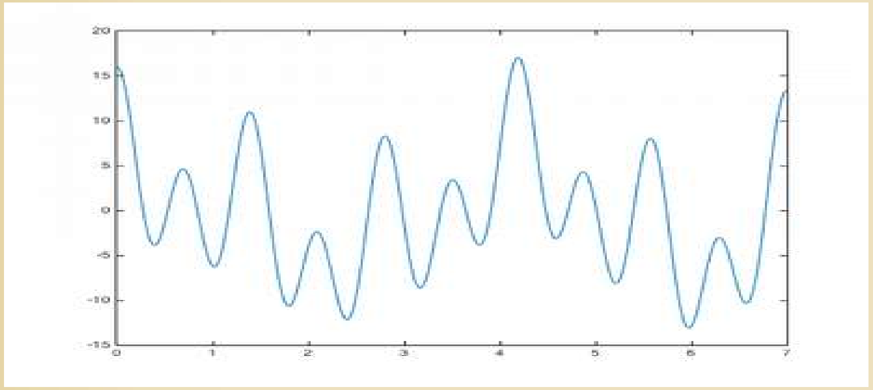
## A system based on ideas from physics: Simulated Annealing

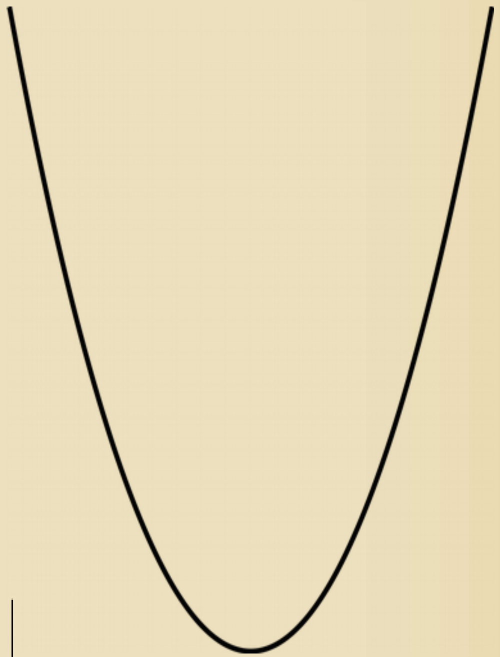
- Use an algorithm that emulates how a physical system reaches the state of minimum energy, as a series of equilibrium states very close together.
  - E.g.:  A melted solid cristalizes when cold enough. If cooling is done very slowly, a single crystal is obtained, that is: the minimum energy state..
- – A system in thermal equilibrium at a temperature T has its energy distributed among all possible states with a Boltzmann probability distribution:

$$Prob(E) \sim e^{\frac{-E}{kT}}$$

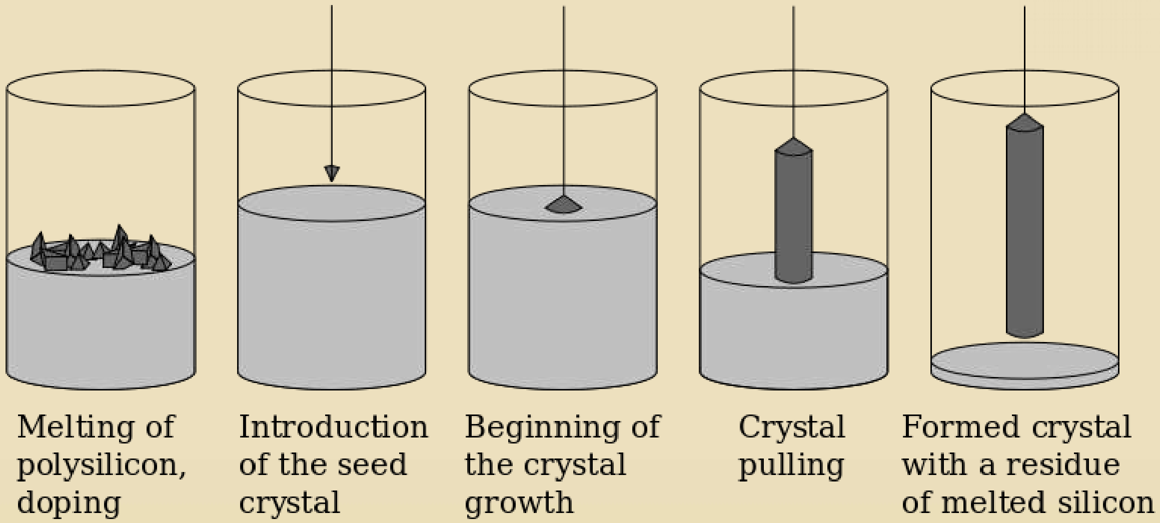
  - Hence, high energy states are possible, although with a very low probability, this happens even for systems with a very low temperature.

# Crystal growth





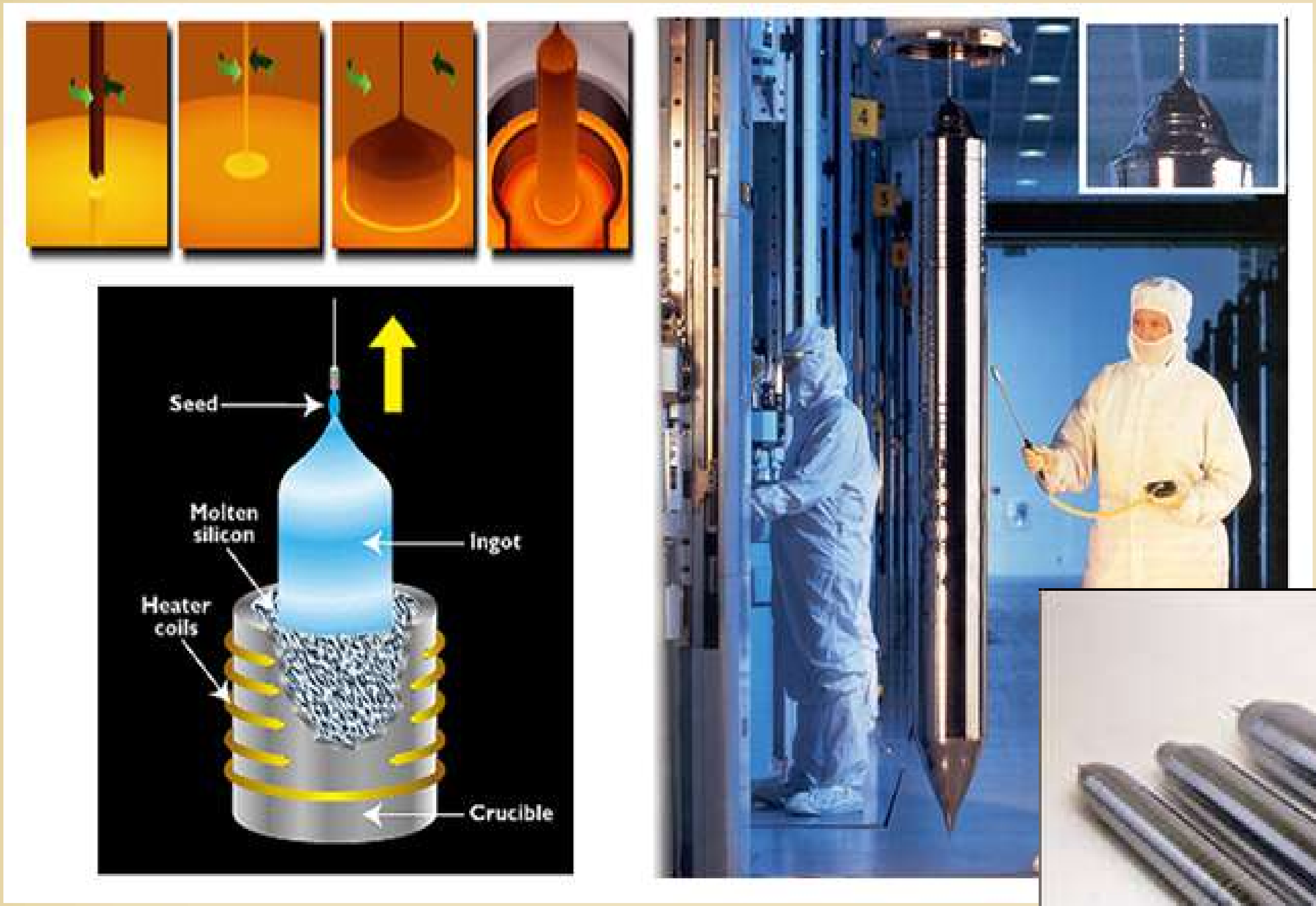
High T: Many close minima Not very deep



Melting of polysilicon, doping
Introduction of the seed crystal
Beginning of the crystal growth
Crystal pulling
Formed crystal with a residue of melted silicon

Low T: the Global minimum

# Silicon wafers



- Algorithm:
  - Start from an approximate solution. Evaluate the objective function („Energy“) to be minimized in that point.
  - Propose a change to the original solution and evaluate the objective function in the new point.
    - If the value of the objective function in the new point is smaller, accept it and change it again.
    - If the value of the objective function is bigger, accept the new point with probability:

$$e^{\frac{-\left(E_{new}-E_{old}\right)}{T}}$$

for a given „temperature“ value, *T*, defined ad hoc.

  - Continue till convergence is achieved.

- The procedure is a Markov chain with transition probabilities given by the Boltzmann distribution.
- Boltzmann's distribution is not the only possible:

$$Threshold\ accepting: P_{TA}(\Delta E) = \begin{Bmatrix} 1 & si & \Delta E \le T \\ 0 & si & \Delta E > T \end{Bmatrix}$$

$$Tsallis\ annealing: P_{TS}(\Delta E, q) = \begin{Bmatrix} 1 \quad si \quad \Delta E \le T \\ \left[1-(1-q)\dfrac{\Delta E}{T}\right]^{1/(1-q)} si \quad \Delta E > 0 \quad y \quad (1-q)\dfrac{\Delta E}{T} \le 1 \\ 0 \quad si \quad \Delta E > 0 \quad y \quad (1-q)\dfrac{\Delta E}{T} > 1 \end{Bmatrix}$$

In fact, for many problems, threshold accepting is the best strategy.

- The two main problems to solve when this algorithm is implemented is:
    - Find a good way to generate new states in which to apply the accept/reject test.
    - Find a good „cooling“ strategy: How to define adequately the „temperature“ and how to vary it during the calculation.

- Example:  Travelling Salesman.
  - Configuration: The list of the pairs of coordinates ($x_i$,$y_i$) for each city. A configuration is a permutation of the original list (cities are visited in the order of the list)
    - Generation of new configuration: As an example, a section of the old path is travelled in inverse order, or a piece of the old path is deleted from its position and inserted in a new one obtained randomly.
  - Objective function: Total length of the path.

$$L = \sum \sqrt{\left(x_i - x_{i+1}\right)^2 + \left(y_i - y_{i+1}\right)^2}$$

  - Cooling strategy: Selected heuristically. As an example, randomly generated configurations are used to determine the typical range of variations in length path $\Delta L$. Choose an initial T substantially higher than the biggest one previously found.  Produce a predetermined number of configurations with that value and them decrease the T value by a  10%.

- The objective function can be adequately weighed to avoid undesirable configurations.
  - Eg.: If we are considering the total length of the paths joining a set of Integrated Circuits that could be in different boards, we could assing a value μ=1 to those in one of the boards and μ=-1 to those in the other. To reduce the communication among boards, a possible objective function could be:

$$L=\sum\left[\sqrt{\left(x_i-x_{i+1}\right)^2+\left(y_i-y_{i+1}\right)^2}+\lambda\left(\mu_i-\mu_{i+1}\right)^2\right]$$

# Bibliography

- S. Kirpatrick, C.D. Gelatt, M.P. Vecchi. „Optimization by Simulated Annealing“. Science 220 (1983) p. 671
- D. Mackay “Information Theory, Inference, and Learning Algorithms”. Cambridge University Press. Non-printable version available online at: http://www.inference.phy.cam.ac.uk/itprnn/book.html
- D. O'Leary “Multidimensional Integration: Partition and conquer” Computing in Science and Enginering Nov./Dec. 2004 p. 58
- D. Kofke „Molecular Simulation“. U. Buffalo.
- D.P. O’leary. Scientific Computing with Case Studies. SIAM (2009)
- W.H. Press et al.  Numerical Recipes. Cambridge University Press. (2007)