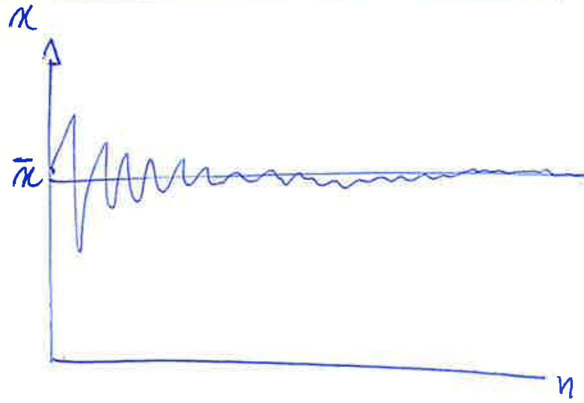
MONTE CARLO METHODS

- Convergence in probability

The sequence $\{x_1, \dots, x_n\}$ is said to converge in probability to $\bar{x}$ if, $\forall \varepsilon > 0$ and $\forall \eta > 0$, a value n_0 can be found such that:

$$\boxed{P(|x_n - \bar{x}| > \varepsilon) < \eta \quad \forall\, n \geq n_0}$$



- Law of large numbers

Assume to repeat the same measurement n times, whose outcome is a random variable x with a given PDF and STD σ.

The average will be: $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$

→ Weak law: If the mean μ exists, and if $\lim_{n\to\infty} \left[\frac{1}{n^2} \sum \sigma_i^2 \right] = 0$

Then $\bar{x}$ converges to μ in quadratic mean: $\lim_{n\to\infty} E\left[(\bar{x} - \mu)^2\right] = 0$

→ Strong law: If $\lim_{n\to\infty} \left[\sum_i \left(\frac{\sigma_i}{i} \right)^2 \right]$ is finite

Then $\bar{x}$ converges almost certainly to μ, which means that $P\left(\lim_{n\to\infty} \bar{x} = \mu \right) = 1$

Take home message: if the parent mean μ exists, the more you measure, the closer the sample mean $\bar{x}$ will go to μ.

→ We should be able to initialize the generator in such a way that it can reproduce exactly the same sequence of random numbers.
The inizialization is commonly done by passing a user-defined number called seed.
This is very useful for debugging code.

• Uniform random number generators

Most common ~~generator~~ and simple generator produces numbers in $[0;1[$

For example, the lrand48 (standard of C) uses the following algorithm:

$$x_{i+1} = (a x_i + c) \bmod (m)$$

where: $m = 2^{48}$

$a = 25214903917$

$c = 11$

The produced random numbers are uniformly distributed between 0 and $2^{48}-1$, and mapped into floating-point ~~nu~~ numbers between 0 and 1.

→ A uniform random number can be remapped to any other interval $[a,b[$ simply by doing:

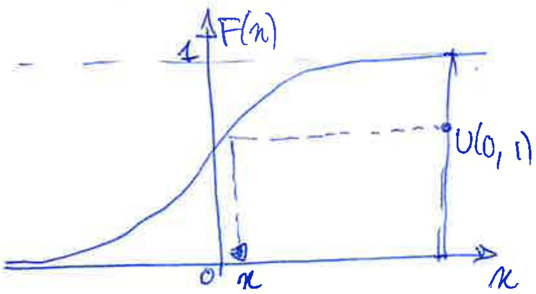
$$x \to x' = a + x(b-a)$$

• Non-uniform generators: inverse-transformation method

Suppose we want to generate a random number distributed as $f(x)$.

The cumulative $F(x)$ will map $x \to [0,1[$

Therefore we can invert (analitically or numerically) $F(x)$ to obtain a number distributed as $f(x)$.



1) Generate r uniformly in $[0,1[$
2) Invert F: $x = F^{-1}(r)$

- Combination of random ~~number generators~~ variables

For complicated cases, we can combine the transform-rejection method and the acceptance rejection method. ~~Or we can use~~

We can also use random number generators to produce binned PDFs of variables that are a combination of other ~~...~~ variables.

This is for example the case of the ratio of two variables.

Example: Change of variable → ratio

- Numerical integration with MC

The acceptance-rejection method estimates the integral $\int_{x_1}^{x_2} f(x)\,dx$ from the fraction of accepted events k over the number n of generated events:

$$I = \int_{x_1}^{x_2} f(x)\,dx \simeq (x_2 - x_1)\cdot \frac{k}{n}$$

~~This applies also for multi-dimensional integration.~~

The uncertainty is (we'll see it in some future lecture):

$$\sigma_{\hat{I}} = (x_2 - x_1)\sqrt{\frac{\hat{I}\,(1-\hat{I})}{N}}$$

Notice that there is no dependence on the dimensionality.

This makes MC method ~~...~~ advantageous ~~...~~ when computing the integrals ~~of~~ high-dimensionality PDFs.

The problem, however, ~~...~~ might be to find the maximum of $f(x)$.

Properties:

→ Metropolis-Hastings allows to ~~map~~ an n-dimensional PDF.

The condition (6) ensures ~~that~~ that if we move to a higher point, the move is always accepted, but ~~also move~~ at the same time we have a small but non-zero probability to accept also lower points.

→ Metropolis-Hastings does not ~~~~ always find the mode of the distribution!

It will find it if you ~~are in low~~ have few dimensions, but it won't if you have dim $\gtrsim 10$ (by experience).

<u>Examples</u>: → Drunk Markov in golf course

→ Random number correlation