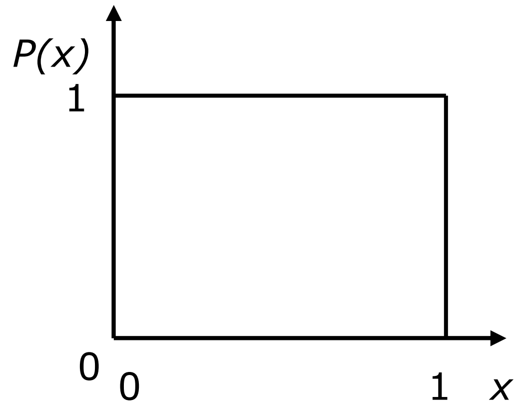
Monte Carlo:

Simulated Experiments

Random numbers: uniform distribution

$$0 \le x \le 1 \quad P(x)dx = dx$$



Basic generator of random numbers

Present in any computer language

```
FORTRAN:        iseed=1.
                x=ran(iseed)
```

Based upon linear congruent generators:

$I_{j+1} = aI_j + b \pmod{m}$

Known problems:

1. Given *k* random numbers, they are distirbuted inside a *k-1 hyperplane*.
2. Less significant bits are more correlated than more significant bits.

It is highly recommended to use specialized routines. Like *Numerical Recipes*, for instance:

ran0: has a period of $2\text{x}10^9$ and exhibits the problems mentioned.

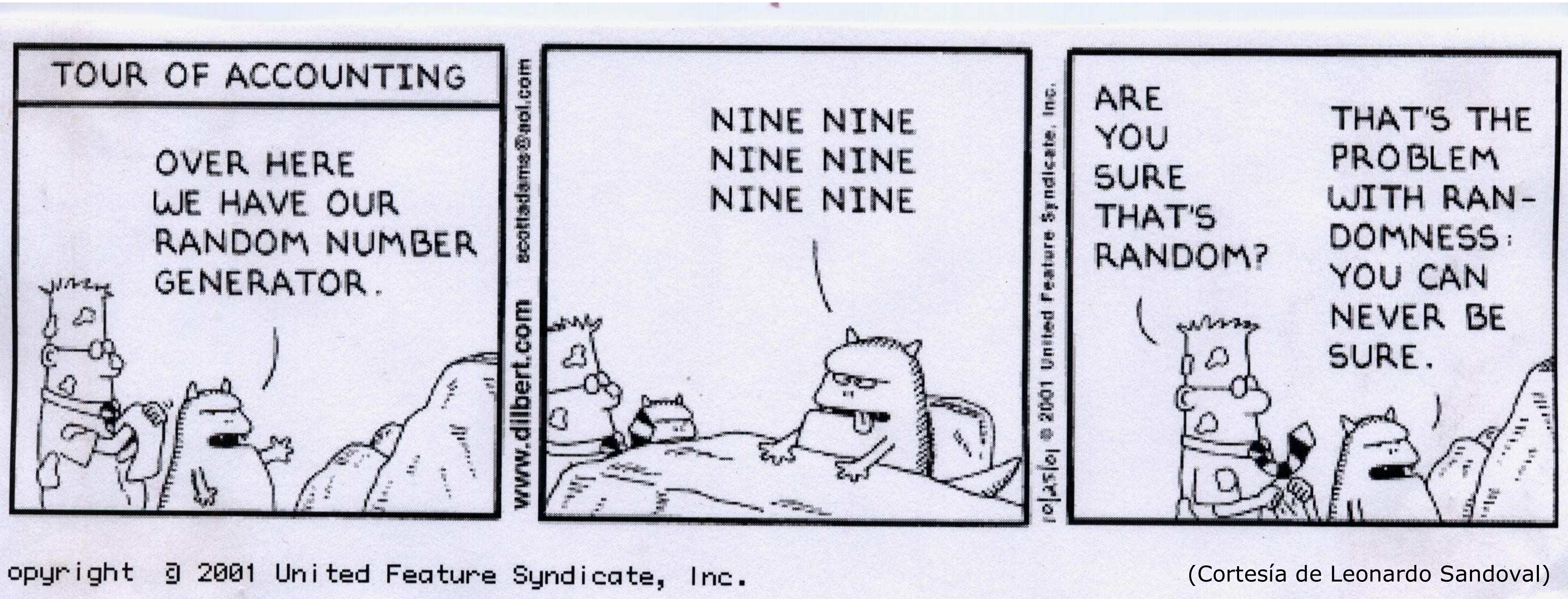
ran1: similar to ran0 but without the problems of dimensionality and correlation. It is one of the most recommended random generators.

ran2: has a repeat period of $2\text{x}10^{18}$, but it is slower than ran0 or ran1.

...

(Press et al., "Numerical Recipes")

Números aleatorios



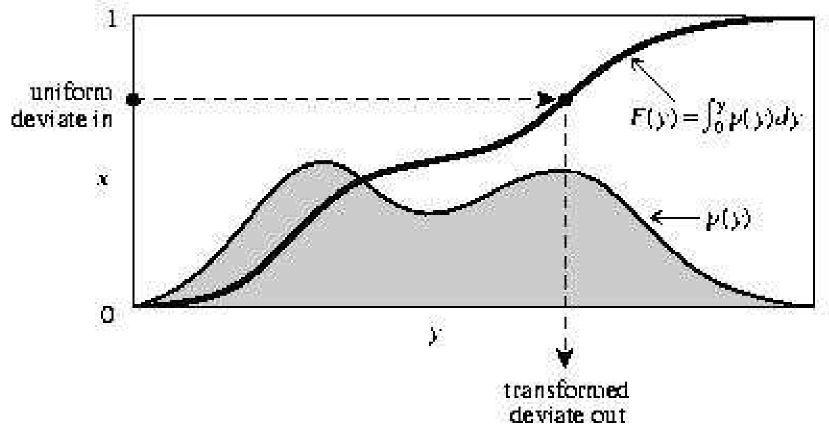
(Cortesía de Leonardo Sandoval)

DIEHARD, set of randomness tests (George Marsaglia, ftp://stat.fsu.edu/diehard/index.html)

Other randomness tests and alternate random generators (http://burtleburtle.net/bob/rand/testsfor.html)

Random numbers: méthod of transformation

Let's assume that we want to generate random numbers following a probability density *P(y)*, associated to a cumulated probability *F(y)*



(Fig. © "Numerical Recipes")

This distribution can be related to the uniform distribution *P(x)*

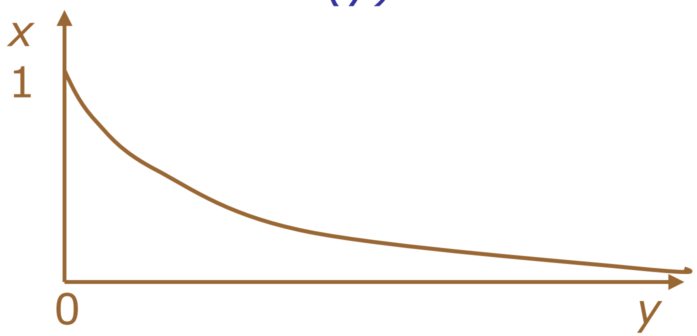
$$P(y)dy = P(x)dx \;\Rightarrow\; x = \int_a^y P(y')dy' \equiv F(y)$$

if *F(y)* can be inverted, then the random number *y=F⁻¹(x) is what we want*. Thus, uniform random numbers *x* are generated and from them are obtained random numbers *y* under the desired distribution *P(y)*.

For instance: exponential distribution $P(y)=e^{-y}$

$e^{-y}dy=dx$ ⇒ $x=1-e^{-y}$ ⇒ $y=-\ln(1-x)$

$0 \leq x \leq 1$ $\qquad$ $0 \leq y \leq \infty$



(Press et al., "Numerical Recipes")

Random numbers: gaussian distributions

Given a pair x,y of random numbers generated from normal distributions. If they are independent, their distribution on a plane will be given by

$$P(x,y) = \frac{1}{2\pi} \exp\left\{ -\frac{(x^2+y^2)}{2} \right\}$$

Or in polar coordinates (R,θ) with $d=R^2$

$$P(d,\theta) = \left| \frac{\partial(x,y)}{\partial(d,\theta)} \right| P(x,y) = \frac{1}{2\pi}\frac{1}{2}\exp(-d/2)$$

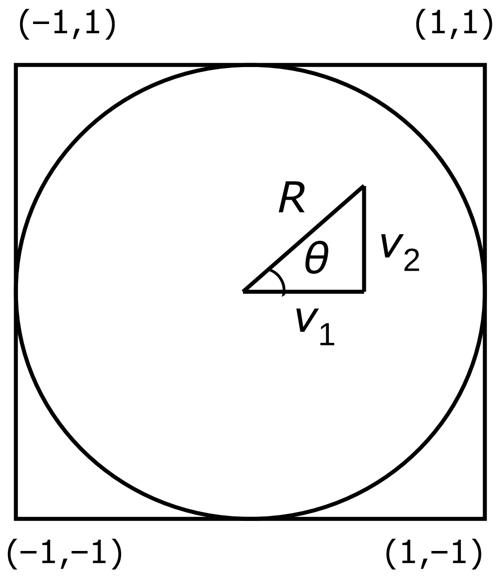
This is equivalent to the product of an exponential distribution of mean life 2 and an uniform distribution in the range [0,2π].
This is the base of the Box-Müller transformation:
Let's take two random numbers u_1, u_2 with uniform distribution.
We will perform the transformations:

$$\boxed{\begin{aligned} R^2 &= -2\ln u_1 \\ \theta &= 2\pi u_2 \end{aligned}} \qquad \boxed{\begin{aligned} x &= R\cos\theta = \sqrt{-2\ln u_1}\ \cos(2\pi u_2) \\ y &= R\sin\theta = \sqrt{-2\ln u_1}\ \sin(2\pi u_2) \end{aligned}}$$

That would yield two random numbers x,y cuya distributed according to a Gaussian distribution. Taking into account that the transformations involve trigonometric functions, they are not very efficient.

(Press et al., "Numerical Recipes")

Random numbers: gaussians



In order to accelerate the Box-Müller algorythm we define the following variables:

$v_1 = 2u_1 - 1$

$v_2 = 2u_2 - 1$

And random numbers are generated (v_1, v_2) picked such as they lie inside the unit circle $R=1$.

$$\cos\theta = \frac{v_1}{R} = \frac{v_1}{(v_1^2 + v_2^2)^{1/2}}$$

$$\sin\theta = \frac{v_2}{R} = \frac{v_2}{(v_1^2 + v_2^2)^{1/2}}$$

$$x = v_1\left(\frac{-2\ln d}{d}\right)^{1/2}$$

$$y = v_2\left(\frac{-2\ln d}{d}\right)^{1/2}$$

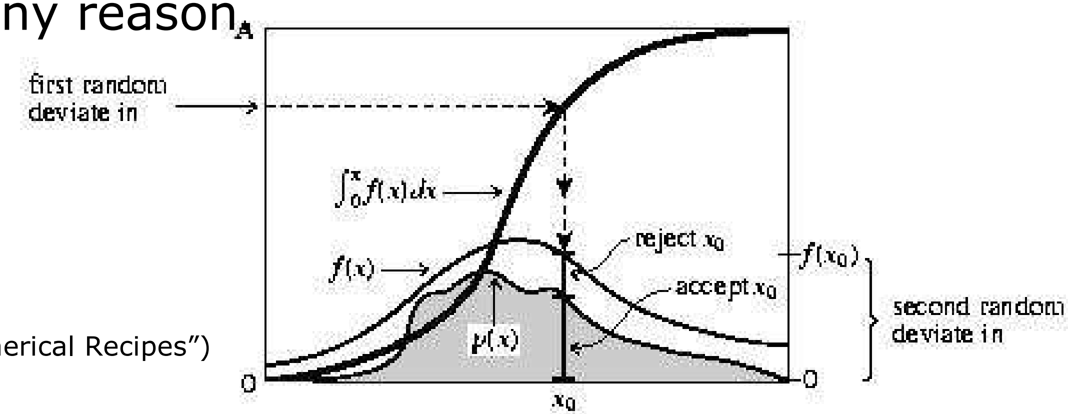
for $d \leq 1$

This modified transformations are computationally more efficient.

(Press et al., "Numerical Recipes")

Random numbers: rejection method

Let’s assume that we want random numbers that follow a particular probability density *P(y)* with cumulated density *F(y)*, that is neither no sea analytic or non invertible by any reason.



(Fig. © “Numerical Recipes”)

We look for the envelop f(y) with finite and invertible integral $I(y)=\int f(y)dy$. If then we take a random number *x* with uniform distribution in the interval (0,*A*), then $y=F^{-1}(x)$ is a random number distributed along the envelop function *f(y).* If now we generate a second random number distributed uniformly, x_2 in the interval $(0,f(y))$, then *y* is a random number distributed according to *P(y)* if $x_2 \leq P(y)$.

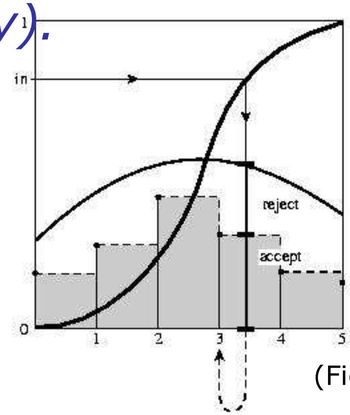
For instance: Poisson distribution where we expand the area as if it were a continuous distribution

$$P(j)=\frac{\mu^j e^{-\mu}}{j!}$$

$$f(y)=\frac{c_0}{1+(y-y_0)^2/a_0^2}$$

$$y=a_0\tan(\pi x)+y_0$$

(Press et al., “Numerical Recipes”)

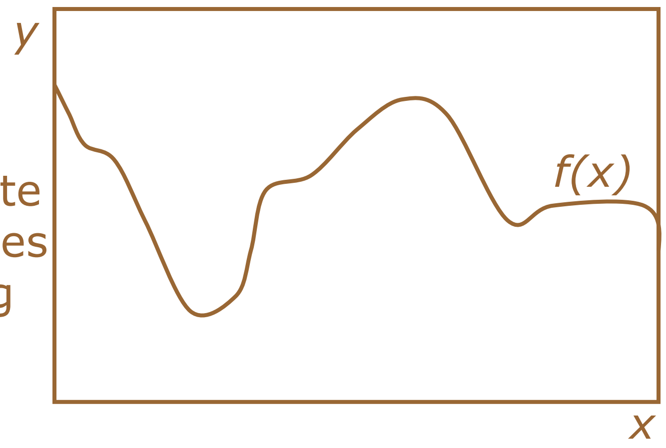


(Fig. © “Numerical Recipes”)

Monte Carlo: brute force approach

Results of an experiment are ***simulated*** employing a computer and a random number generator. This is often useful when the computations are too difficult or too vaguely defined to be solved by numeric or algebraic methods numéricos, or we are simply too lazy to think in more elegant solution strategies.

Classical example: area under a curve.
Given an area easy to measure that contains a difficult to integrate curve curve *f(x)* we can compute the area under the curve by generating many instances *of a pair of uniform random nubers (x,y)* representing coordinates. We simply count the number of pairs above and below the curve.



$$\int f(x)dx = A\,\frac{\text{n}^{\text{o}}\ \text{de puntos bajo la curva}}{\text{n}^{\text{o}}\ \text{de puntos}}$$

This same way of reasoning can also be applied to volumes.

The statistical error is proportional to $1/\sqrt{N}$