We are going to need a lot of Random numbers.

Where do we get them? QM? Throwing dice?

Pseudo random numbers

Are they random?
How to generate them?
What happens if they are not "high Quality"?

e.g. Linear congruential generator, LCG

$$X_{n+1} = (a \cdot X_n + c) \bmod m$$

$0 < X_i < m$ can convert to: $[0,1]$ $x/(m-1)$; $(0,1)$ $\frac{x+1}{m+1}$; $[0,1)$ x/m ; $(0,1]$ $\frac{x+1}{m}$

X_0 = seed. reproducible results

fast, easy, has some problems

cycle $< m$

correlations

e.g. $a = 1277$ $m = 2^{17}$ $c = 0$ too simple

$a = 5DEECE66D$ $m = 2^{48}$ $c = 11$ drand48() in C

$a = 5851F42D4C957F2D$ $m = 2^{64}$ $c = 14057B7EF767814F$

m = power of 2 $\Rightarrow$ easy to implement

"Mersenne Twister" used in python

CMRG (combined LCGs) www.iro.umontreal.ca/~lecuyer/myftp/papers/streams00.pdf

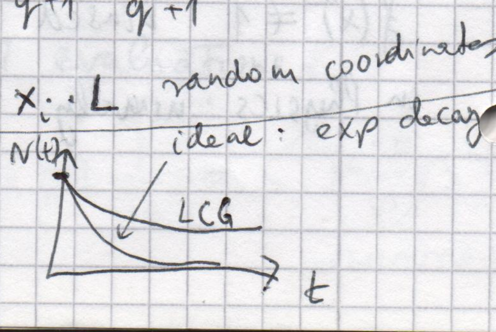
Tests for Pseudo Random Number Generators:

Take uniform $[0,1]$
$$\int_0^1 x^k\,dx = \frac{1}{k+1}$$

$$C_{qq'} = \int_0^1 x^q \int_0^1 x'^{q'}\,dx'dx = \frac{1}{q+1}\cdot\frac{1}{q'+1}$$

$\boxed{\text{Marsaglia effect}}$: fill L^d lattice $x_i = X_i \cdot L$ random coordinates

$N(t)$ = # of points with no hit after t random points



Random numbers with non uniform distribution

1. Tranformation
2. Inversion
3. rejection

Transformation. Construct a map between uniform and desired

e.g $x_1, x_2 \in [0,1]$ uniform

$$z_1 = \sqrt{-2\ln x_1} \cdot \cos(2\pi \cdot x_2) \qquad (\text{Box-Müller})$$

$$z_2 = \sqrt{-2\ln x_1} \cdot \sin(2\pi \cdot x_2)$$

z_1, z_2 are independent, Gaussian, distributed $P(z_i) \sim e^{-z^2/2}$

Inversion we want $\rho(y)$ if we know $\int_{-\infty}^{y} \rho(y)\,dy = F(y)$

$x \in [0,1]$ uniform

$y = F^{-1}(x)$ $\qquad P(y, y+dy) = P(x, x+dx) = dx = \frac{dF(y)}{dy} dy = \rho(y)\, dy$

with $y = F^{-1}(x)$

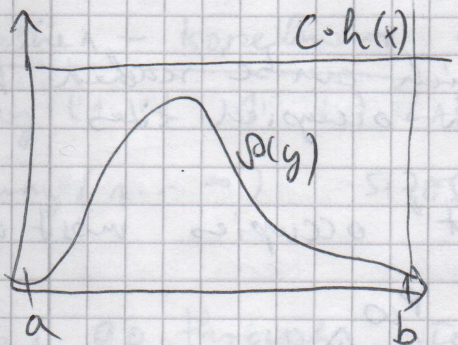
$$F(y) = x$$

$$\frac{dF(y)}{dy} dy = dx$$

1. 2. works if we are lucky

3. is general: **Rejection method**

we want $\rho(y)$. $\quad h(x) \cdot C \geq \rho(y)$, Take $h(x)$ = uniform first

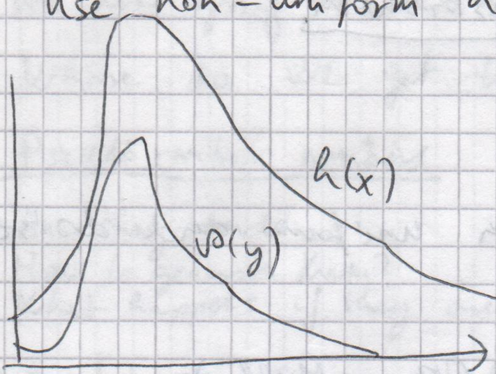


1. $y \in [a,b]$ uniform $(h(y))$ $\left(\begin{array}{l} a + x_i(b-a) \\ x_i \in [0,1] \end{array} \right)$
2. accept with prob. $\rho(y) / C \cdot h(y)$
3. if rejected, goto 1

$$P(y, y+dy) = \cancel{P_{a}} = h(y) \cdot dy \cdot \frac{\rho(y)}{C \cdot h(y)} \sim \rho(y) \cdot dy$$

if $\rho(y)$ is peaked rejection rate will be high: inefficient

Use non-uniform $h(x)$: We must have $C h(x) \geq \rho(x)$



rejection rate is decreased

$h(x)$ can be e.g.

Gaussian

exponential $\rho(x) = \lambda \cdot e^{-\lambda x}$

$F(x) = 1 - e^{-\lambda x}$

Combined: $x_1, x_2 \in [0,1]$

$$s = x_1^2 + x_2^2 \leq 1 \quad \text{(reject if not)} \quad \text{uniform dist } [0,1]$$

$$\cos\theta = x_1/\sqrt{s} \qquad \sin\theta = x_2/\sqrt{s}$$

$$z_1 = \sqrt{-2\ln s} \cdot \frac{x_1}{\sqrt{s}} = x_1 \cdot \sqrt{\frac{-2\ln s}{s}}$$

$$z_2 = \qquad = x_2 \cdot \sqrt{-2\ln s / s}$$

Percolation - mixture of insulating and conducting grains

"trickle through" - line formation of polymers - when does coffe through filter it become interconnected? (gelation)

def:) Occupy sites of a square lattice with probability p, independently

(occupied / empty, conducting / insulating ...)

→ configuration

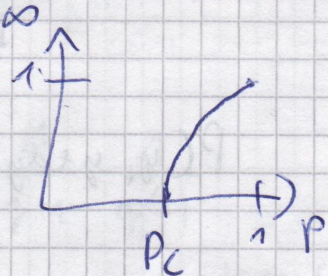
Cluster = set of points, which can be reached from each other through occupied sites

p small → small clusters

p large → "giant component" occupies most of the lattice

geometrical phase transition:

P_∞ = prob. of belonging to the largest cluster



n_s = # of clusters with size s per site

$$n_s = \frac{\text{\# of s-size clusters}}{L_s}$$

prob. of belonging to s-size clusters

$$p_s = s \cdot n_s$$

$$\sum_s p_s + P_\infty + (1-p) = 1$$

average size of finite clusters ⇒ $S = \dfrac{\sum_s s^2 n_s}{\sum_s s n_s}$

similar to phase transition of spin systems

$P_\infty \Leftrightarrow$ magnetization

$S \Leftrightarrow$ susceptibility

$p \Leftrightarrow$ temperature

two point function $G(r)$ = prob. that two occupied points with dist. r belong to same cluster ⇒ connectivity ξ = mean diameter

ghost site $\Leftrightarrow$ magnetic field

↑ single "super site" attached with prob $1-e^{-h}$ to any single site

around p_c

$$S \sim |p-p_c|^{-\gamma}$$

$$n_s(p_c) \sim s^{-\tau} \qquad \xi \sim (p-p_c)^{-\nu}$$

$$P_\infty \sim |p-p_c|^{\beta}$$

$$g_s(p) = \sum_{fink} n_s(p)$$

$$g_s(p,h) = \sum n_s\, e^{-hs}$$

in simulation: how to find n_s?

Hoshen – Kopelman alg. in 2d

0	x	0	x	x	0	0		0	1	0	2	2	0	0
0	x	0	0	0	0	x	→	0	1	0	0	0	0	3
0	0	0	0	x	0	x		0	0	0	0	3	0	3
0	0	0	0	x	x	x		0	0	0	0	3	3	3

Array(1..L, 1..L) LABEL

Array(1 → ∞) SIZES ← sizes of an identified cluster or −LABEL if it belongs to a proper cluster

Alg: 1, go through sites left to right up to down

2, site occupied – with unoccupied left and upper neigh.: new cluster label NL, set SIZES(NL) = 1

– occupied only left: take its label, SIZES(L) += 1

– occupied only above: find root label from L

while ~~SIZES(L) > 0~~

while SIZES(L) < 0 : L = − SIZES(L)

write L to the LABEL lattice, SIZES(L) += 1

− both are occupied :

find both root labels, take the smaller and unify them

$$\left(L_1 < L_2 : \quad \begin{array}{l} \text{SIZES}(L_1) \mathrel{+}= \text{SIZES}(L_2) + 1 \\ \text{SIZES}(L_2) = -L_1 \end{array} \right)$$

at the end the SIZES array has what we need.

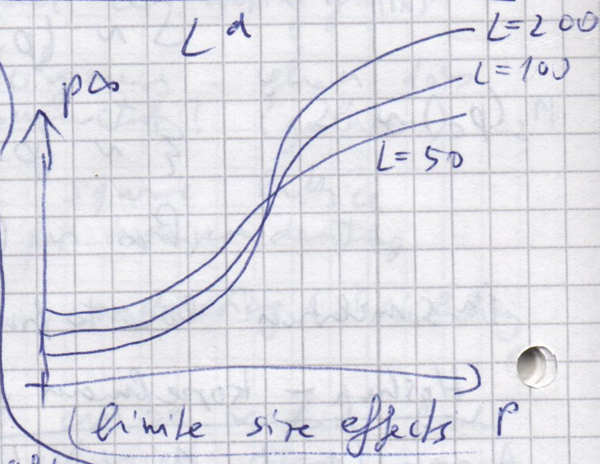
~~fill label~~ : TRICK : do this alg as occupations are generated.

. we can forget old rows. (we need L^{d-1} storage)

$$\left[\text{in } d=2 \quad \begin{array}{ll} p = \frac{1}{2} & \text{for triangular latt} \\ p = 0.5927 & \text{square latt} \end{array} \right) \quad \boxed{\nu = 4/3 \quad \beta = \frac{5}{36}}$$

for finite lattice e.g. $P_\infty = \dfrac{\text{points in largest cluster}}{L^d}$

(generalizations bond percolation
directed percolation)



Random walks Conformation of a polymer chain

$R \sim N^{\nu}$

simplest model : Random walk

$\rightarrow R \sim t^{1/2}$ disagrees with experiment.

no back track RW : do not ~~imedi~~ turn back

SAW self avoiding random walk : do not go to a site already visited (monomers repulse)

What should be the weight of a random walk?

RW : all walks of length N are equally probable

SAW : — " — as long as they are SAWs