SORTING

tasks: - sort in e.g ascending order the elements of a vector
- sort more vectors according to one vector in ascending order
- calculate index or rank vectors

computational cost: $\leq N^2$ (at small N, it is not important)

Straight insertion $\approx N^2$ $\underline{a}$ is the vector to sort

two cycles: (loops)
outer cycle: a_j $j = 2, 3, \ldots N$
inner cycle: a_i $i = j-1, j-2, j-3 \ldots 1$
if $a_j \leq a_i$, then $a_{i+1} = a_i$
else $a_{i+1} = a_j$, next j
(do it also, if $i = 0$)

Sorting with swap

outer loop a_i $i = 1, 2, 3 \ldots, (N-1)$
inner loop a_j $j = (i+1), (i+2), \ldots N$
if $a_i > a_j$ change a_i and a_j
(replace them with each other)

bubble sort

outer loop a_i $i = 2, 3, \ldots N$
inner loop a_j $j = N, (N-1), (N-2), \ldots, i$
if $a_{j-1} > a_j$ then change $a_{j-1} \leftrightarrow a_j$
(replace them with each other)

Shell's method

prepare groups at first

e.g for N = 16

$\Delta = 8$ $(a_1, a_9), (a_2, a_{10}) \ldots (a_8, a_{16})$ → then sort within the groups with e.g. straight insertion

unify groups for new larger groups

$\Delta = 4$ $(a_1, a_5, a_9, a_{13}), \ldots (a_4, a_8, a_{12}, a_{16})$ — " — (do the same ↑)

unify them

$\Delta = 2$ $(a_1, a_3, \ldots a_{15}) \ldots (a_2, a_4 \ldots a_{16})$ — " —

unify to $\Delta = 1$

$(a_1, a_2, a_3 \ldots a_{16})$

effective replacement of the elements is smaller:

$$N^{1,25} \leq \leq N^{1,5}$$

Δ can be a 2^{k-1}, $k = 1, 2, 3 \ldots$ or

$(3^k - 1)/2$ $(\ldots 40, 13, 4, 1)$

Quich sort

Choose a given a_j |——————|

Start from two ends of the vector:

$i \neq j$ $i = 1, 2 \ldots$ →

$k \neq j$ $k = N, N-1, N-2$ ←

do as long as find one-one on the two sides as

$a_i \geq a_j$ $a_k \leq a_j$

then stop for a change of $a_i \leftrightarrow a_k$, and

then continue further.

If $i = k$, put there a_j ← its final position.

Repeat it separately for section A and B,

do it as long as every one on the right place.

$\leq N^2$, but for unordered it is in average very good!

Index and rank

a	index	r	asorted
15	2	4	2
2	3	1	3
3	4	2	9
9	1	3	15

original vector — index vector — rank vector — sorted vector

Calculation of rank, Cloyd's method (step: set all r_i to 1)

outer loop $i = 1, 2, \ldots, (N-1)$

inner loop $j = i+1, i+2, \ldots N$

if $a_i \leq a_j$ then $r_j = r_j + 1$

else $r_i = r_i + 1$

In the case of more vectors (sorting them according to one vector)

poss. 1: - do one of the previous methods parallel with the other vectors

poss 2: - calculate index vector and rearrange all of them simultaneously

- same for rank vector

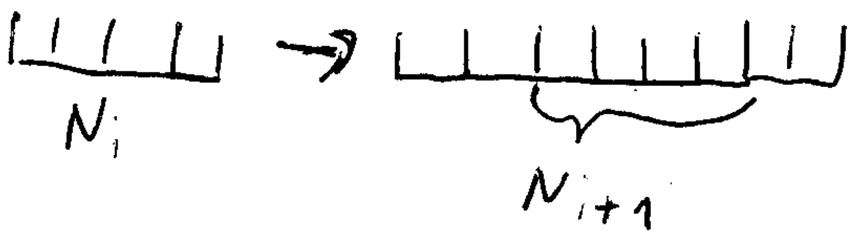
Random Numbers

in reality pseudo random numbers, because deterministic algorithm (heuristic equations)

Neumann János - (John von Neumann) 1951

N is a four digit number

$N_i \rightarrow N_i^2 \rightarrow$ cut the middle four digits $= N_{i+1}$



Linear congruential generators → uniform distributed random numbers

$$X_i = (a X_{i-1} + c)_{\mod m}$$

i - iteration number

$(5)_{\mod 3} = 2$

$(16)_{\mod 5} = 1$

$X_i \geq 0 \quad X_i \in \mathbb{R}$ or $X_i \in \mathbb{N}$

c can be $\emptyset$

$0 \leq \frac{X_i}{m} < 1$ requirements: - uniform in $[0; 1)$

- long recurrance time if $X_i \in \mathbb{N}$
- no correlation in the position of X_i and X_{i+1}

a good generator:

$X \in \mathbb{N}, c = 0, a = 16807$

$m = 2^{31} - 1 = 2147483647$

- not every a, c, m are good
- use built in routines
- there are many modified schemes
- initial value is necessary, called „seed"

Transformation to other distributions

for uniform, a number is between $(x; x+dx)$

the probability is $p(x)\cdot dx$ $\begin{cases} dx, \text{ if } 0 \le x < 1 \\ 0 \end{cases}$

↑ "density function of probability"

Let y be a function of $x \Rightarrow y(x)$

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

↑ dist. of y

example: $y(x) = -\ln x$

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

↑ derivate of the inverse function of $y(x) \Rightarrow x(y) = e^{-y}$, its derivate in abs. $= \left|-e^{-y}\right| = e^{-y}$

1.) generate $0 \le x < 1$

2.) calculate $y = -\ln x \Rightarrow y$ will be distributed as e^{-y}

Box-Miller algorithm for Gauss-distribution

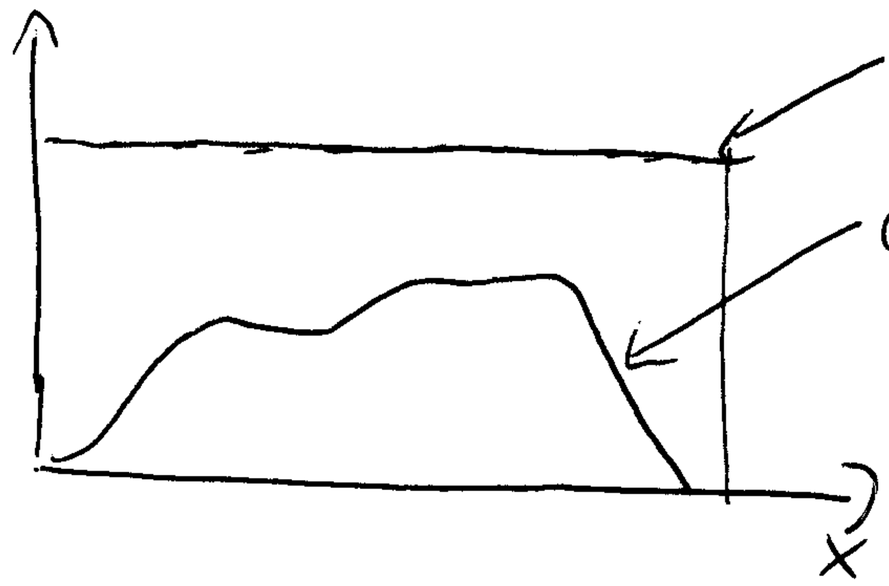
1.) Generate uniform dist. x_1, x_2 $\quad 0 \le x_1 < 1,\ 0 \le x_2 < 1$

2.) calculate y_1 an $y_2 \leftarrow$ normally distributed, if

$$y_1 = \sqrt{-2\ln x_1}\cdot\cos(2\pi x_2)$$

$$y_2 = \sqrt{-2\ln x_2}\cdot\sin(2\pi x_1)$$

Rejection method (Neumann 1951)



$f(x) \geq p(x)$ for all important x

1.) generate an x according to $f(x)$

2.) generate a ξ, uniform in $0 \leq \xi < 1$

3.) calculate $p(x)$ and $f(x)$

4.) if $\frac{p(x)}{f(x)} \geq \xi$ → accept the generated x

else → reject the generated x and try with a new x and ξ

For special cases there are spetial methods,
e.g. uniformly distributed points on a sphere.