In the name of $\mathcal{GOD}$

Computational Physics

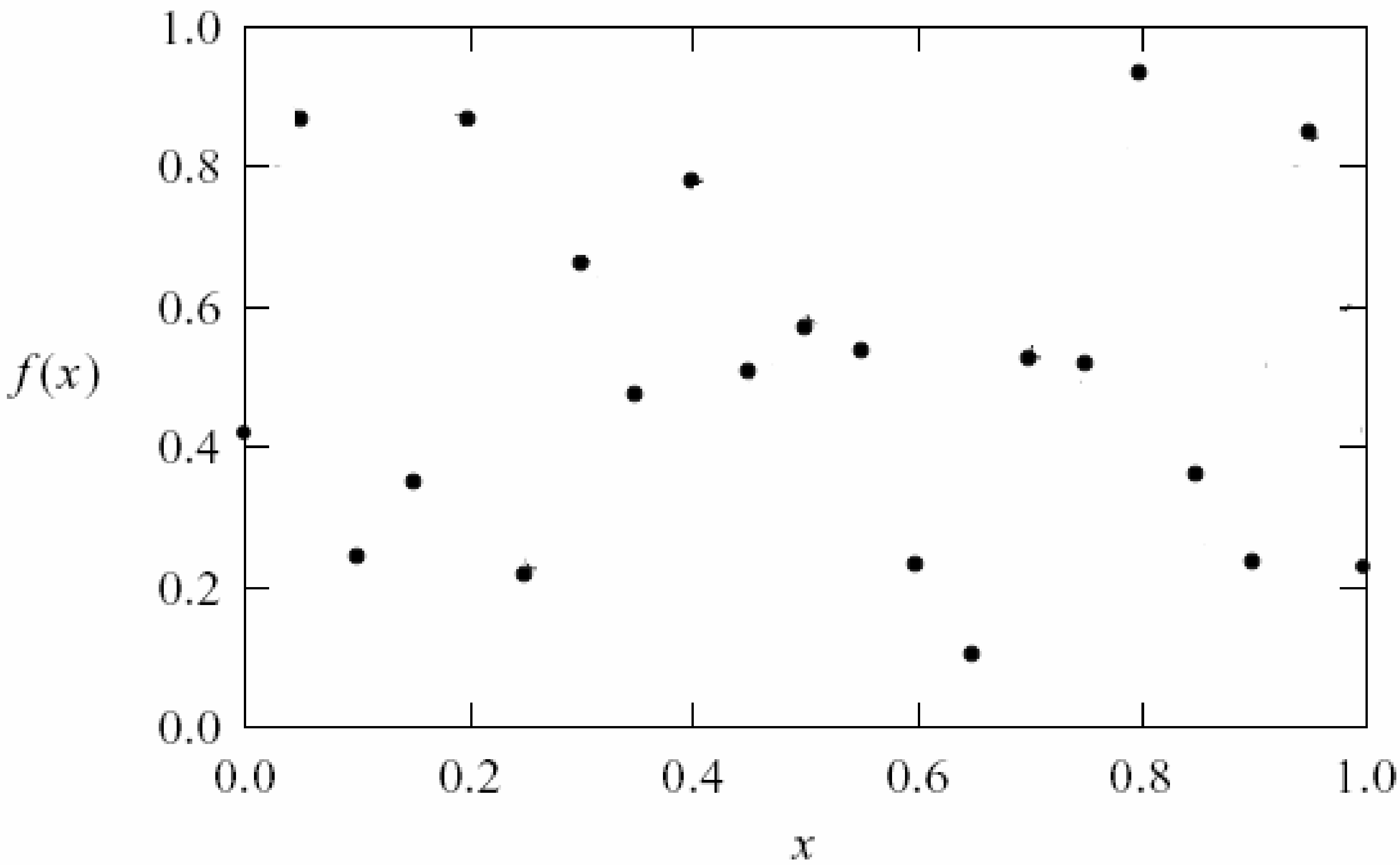
Mohammad Reza Mozaffari
Physics Group
University of Qom

Outline

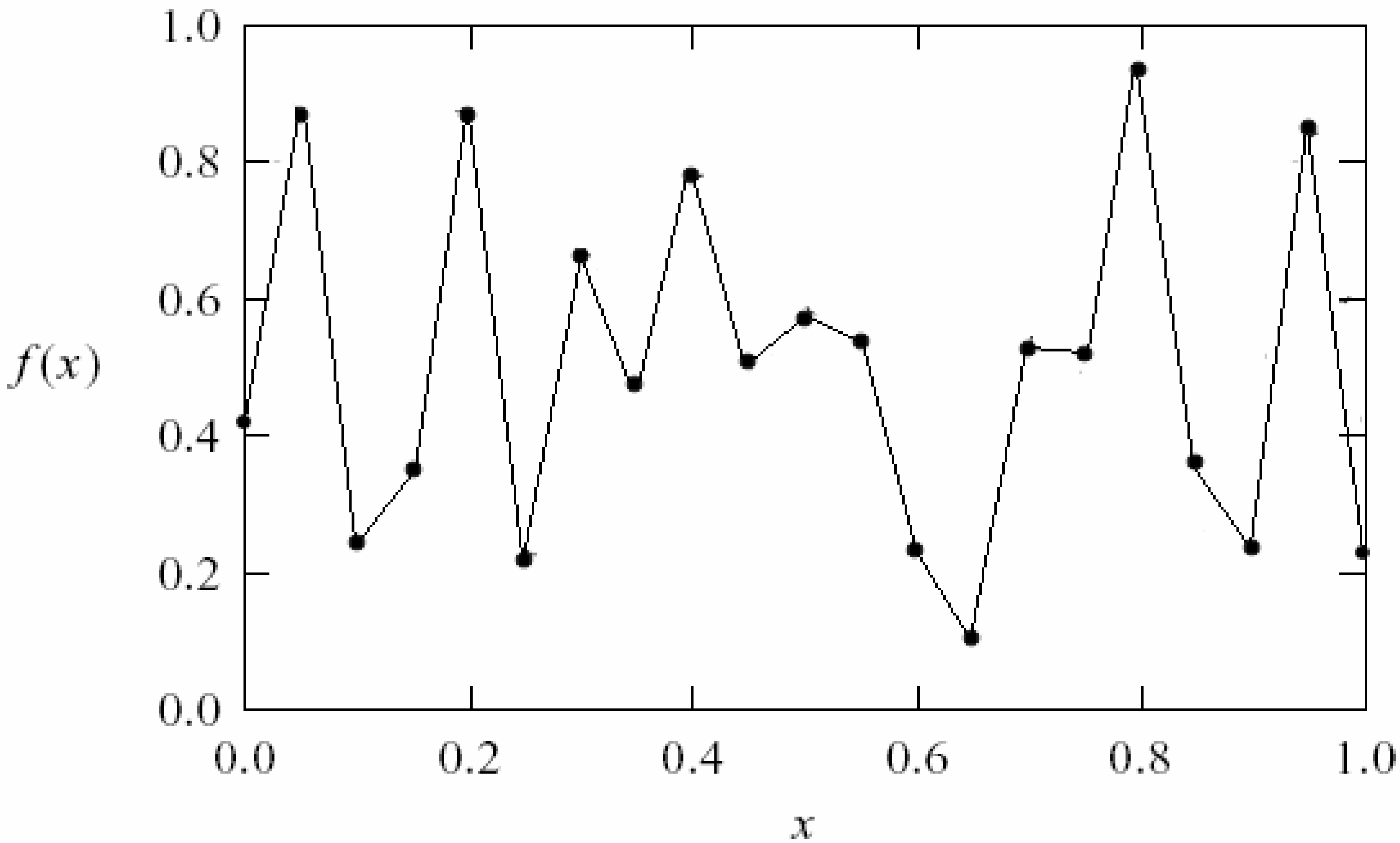
- Fortran-90 Programming
- Numerical calculus
- Approximation of a function
- Numerical methods for matrices
- Ordinary differential equations
- Partial differential equations
- Monte Carlo simulations
- Molecular dynamics simulations

- Interpolation is needed when we want to infer some ***local information*** from a set of incomplete or discrete data.

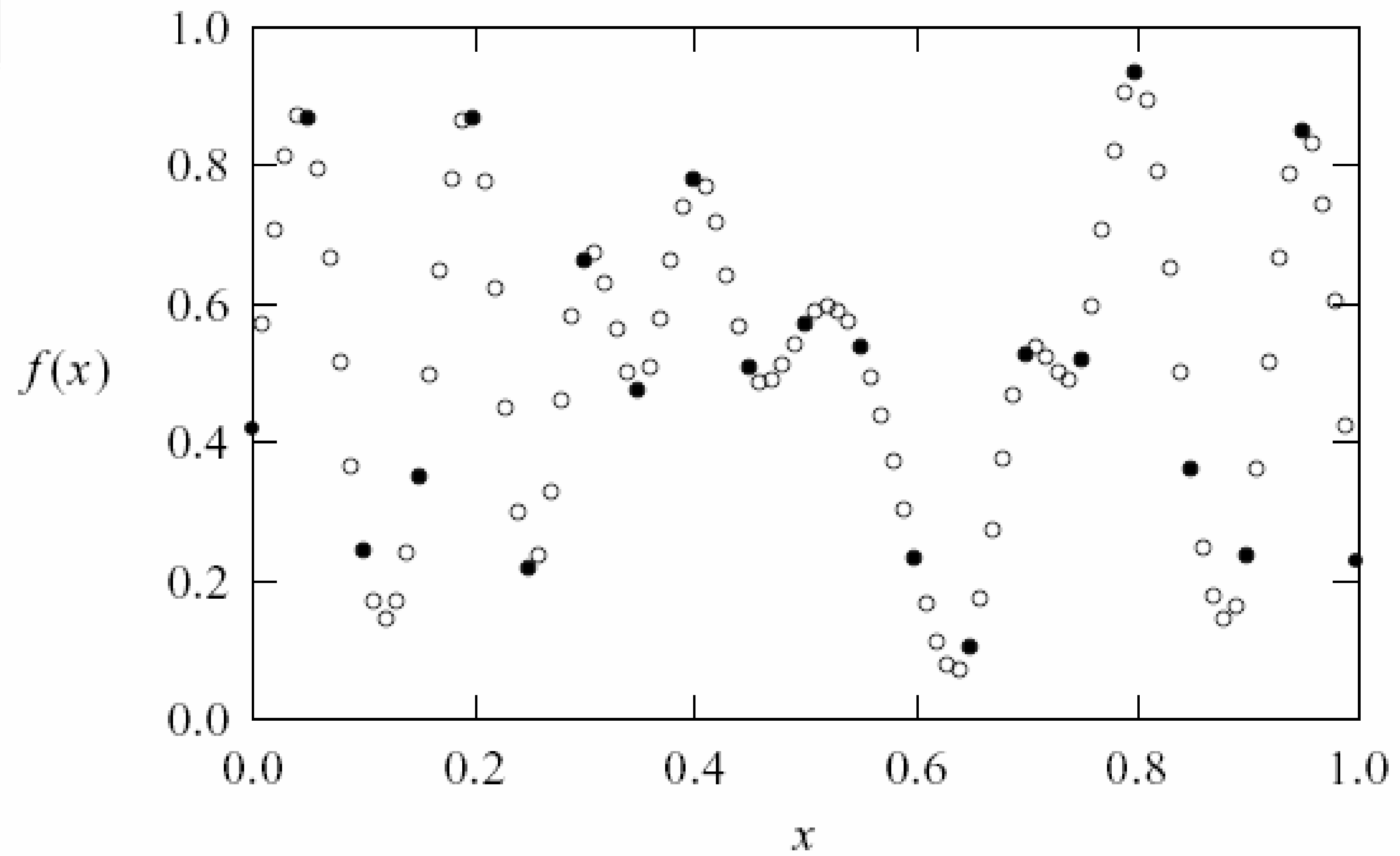
- Overall approximation or fitting is needed when we want to know the ***general or global behavior*** of the data.


1.0
0.8
0.6
0.4
0.2
0.0
f(x)
0.0
0.2
0.4
0.6
0.8
1.0
x

linear



Non-linear

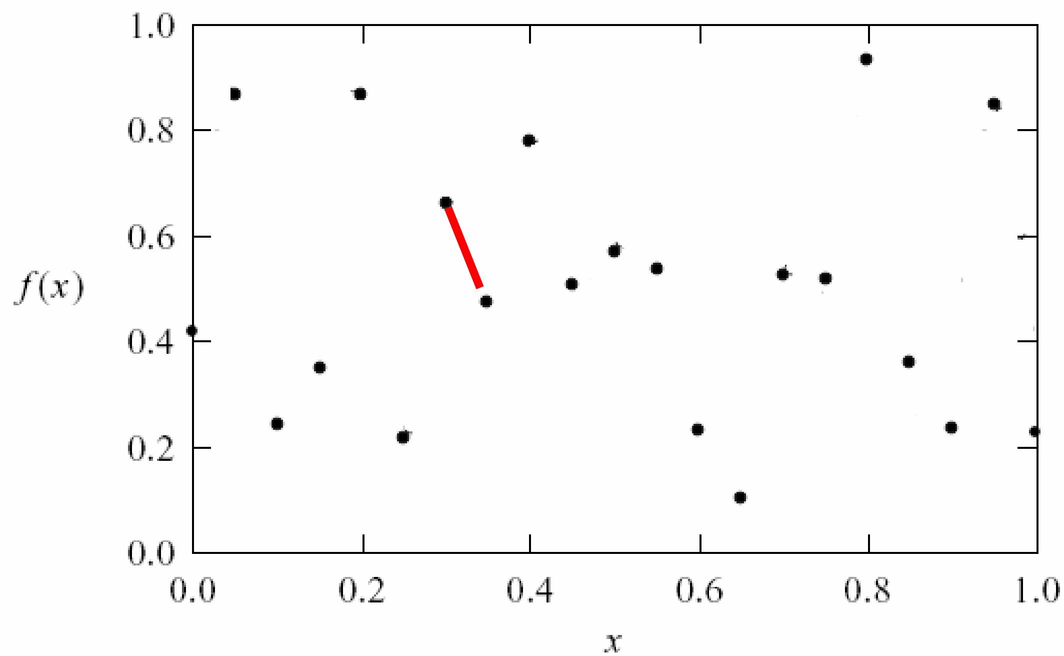


Linear interpolation

$$\boxed{x_1, x_2, x_3, \ldots, x_i, \ldots, x_n}$$

$$f_1 = f(x_1)$$
$$f_2 = f(x_2)$$
$$f_3 = f(x_3)$$
$$\vdots$$
$$f_i = f(x_i)$$
$$\vdots$$
$$f_n = f(x_n)$$

$$\boxed{f_1, f_2, f_3, \ldots, f_i, \ldots, f_n}$$



simplest way to obtain the approximation of ***f*(*x*)**

$$x \in [x_i, x_{i+1}]$$

$$f(x) = f_i + \frac{x - x_i}{x_{i+1} - x_i}(f_{i+1} - f_i)$$

is not accurate enough in most cases but serves as a good start in understanding other interpolation schemes.

$$f(x) = f_i + \frac{x - x_i}{x_{i+1} - x_i}(f_{i+1} - f_i)$$

$$f(x) = f_{i+1} + \frac{x - x_{i+1}}{x_i - x_{i+1}}(f_i - f_{i+1})$$

$$f(x) = \frac{x - x_i}{x_{i+1} - x_i} f_{i+1} + \frac{x - x_{i+1}}{x_i - x_{i+1}} f_i$$

$$f(x) = f_i + \frac{x - x_i}{x_{i+1} - x_i}(f_{i+1} - f_i) + \Delta f(x)$$

$$\Delta f(x) = ?$$

$$\begin{cases} f(x_i) = f_i \Rightarrow \Delta f(x_i) = 0 \\ f(x_{i+1}) = f_{i+1} \Rightarrow \Delta f(x_{i+1}) = 0 \end{cases}$$

$$\Delta f(x) \sim (x - x_i)(x - x_{i+1})$$

Taylor expansion

$$f(x) = f(a) + (x-a)\frac{f'(a)}{1!} + (x-a)^2\frac{f''(a)}{2!} + \cdots$$

$$+ (x-a)^n\frac{f^{(n)}(a)}{n!} + \cdots$$

$$\Delta f(x) \sim (x - x_i)(x - x_{i+1})$$

$$\Downarrow$$

$$\Delta f(x) = \frac{\gamma}{2}(x - x_i)(x - x_{i+1})$$

$$\gamma = f''(a) \qquad a \in [x_i, x_{i+1}]$$

$$\Delta f(x) = \frac{\gamma}{2}(x - x_i)(x - x_{i+1})$$

maximum error

$$\frac{d}{dx}\Delta f(x) = \frac{\gamma}{2}\{2x - (x_i + x_{i+1})\}$$

$$\frac{d}{dx}\Delta f(x) = 0 \Rightarrow x_{extremum} = \tfrac{1}{2}(x_i + x_{i+1})$$

$$\Delta f(x_{extremum}) = \frac{\gamma}{8}(x_i - x_{i+1})^2$$

The Lagrange interpolation

$$f(x) = \frac{x - x_i}{x_{i+1} - x_i} f_{i+1} + \frac{x - x_{i+1}}{x_i - x_{i+1}} f_i + \Delta f(x)$$

$$\Delta f(x) = \frac{\gamma}{2}(x - x_i)(x - x_{i+1})$$

$$f(x) = \frac{(x - x_i)(x - x_{i+1})}{(x_{i-1} - x_i)(x_{i-1} - x_{i+1})} f_{i-1} +$$

$$\frac{(x - x_{i-1})(x - x_{i+1})}{(x_i - x_{i-1})(x_i - x_{i+1})} f_i +$$

$$\frac{(x - x_{i-1})(x - x_i)}{(x_{i+1} - x_{i-1})(x_{i+1} - x_i)} f_{i+1} + \Delta f(x)$$

$$\Delta f(x) = \frac{\gamma}{6}(x - x_{i-1})(x - x_i)(x - x_{i+1})$$

$$f(x)=\sum_{i=1}^{n} f_i P_i^n(x)+\Delta f(x)$$

$$P_i^n(x)=\prod_{\substack{j=1\\ j\neq i}}^{n}\frac{(x-x_j)}{(x_i-x_j)}$$

$$\Delta f(x)=\frac{\gamma}{(n+1)!}(x-x_1)(x-x_2)(x-x_3)\cdots(x-x_n)$$

$$\gamma = f^{(n+1)}(a) \quad , \quad a\in[x_1,x_n]$$

$$|\Delta f(x)| \leq \frac{\gamma_n}{4(n+1)} h^{n+1}$$

Least-squares approximation

In many situations in physics we need to know the global behavior of a set of data in order to understand the trend in a specific measurement or observation. A typical example is a polynomial fit to a set of experimental data with error bars.

x	$f(x)$
x_0	f_0
x_1	f_1
x_2	f_2
$\vdots$	$\vdots$
x_n	f_n

$$P = P(x)$$

$$\begin{gathered}
\left(P(x_0) - f_0\right) \to 0 \\
\left(P(x_1) - f_1\right) \to 0 \\
\left(P(x_2) - f_2\right) \to 0 \\
\vdots \\
\left(P(x_i) - f_i\right) \to 0 \\
\vdots \\
\left(P(x_n) - f_n\right) \to 0
\end{gathered}$$

$$\ell = \left(P(x_0) - f_0\right) + \left(P(x_1) - f_1\right) + \left(P(x_1) - f_1\right) + \cdots$$
$$+ \left(P(x_i) - f_i\right) + \cdots + \left(P(x_n) - f_n\right)$$

Necessary condition to $\ell \to 0$ is :

$$\boxed{\begin{array}{c} \left(P(x_0) - f_0\right) \to 0 \\ \left(P(x_1) - f_1\right) \to 0 \\ \left(P(x_2) - f_2\right) \to 0 \\ \vdots \\ \left(P(x_i) - f_i\right) \to 0 \\ \vdots \\ \left(P(x_n) - f_n\right) \to 0 \end{array}}$$

$$\ell = \left(P(x_0) - f_0\right)^2 + \left(P(x_1) - f_1\right)^2 + \left(P(x_2) - f_2\right)^2 + \cdots$$
$$+ \left(P(x_i) - f_i\right)^2 + \cdots + \left(P(x_n) - f_n\right)^2$$

$$\ell = \left(P(x_0) - f_0\right)^2 + \left(P(x_1) - f_1\right)^2 + \left(P(x_2) - f_2\right)^2 + \cdots$$
$$+ \left(P(x_i) - f_i\right)^2 + \cdots + \left(P(x_n) - f_n\right)^2$$

$$\ell = \sum_{i=0}^{n} \left(P(x_i) - f_i\right)^2$$

least square

$$P(x) = ?$$

$$\boxed{\begin{gathered} P_m(x) = \sum_{k}^{m} c_k x^k \\ P_5(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3 + c_4 x^4 + c_5 x^5 \end{gathered}}$$

$$\boxed{\begin{gathered} P_m(x) = \sum_{k}^{m} c_k \sin\left(\frac{kx}{m}\right) \\ P_3(x) = c_0 + c_1 \sin\left(\frac{x}{3}\right) + c_2 \sin\left(\frac{2x}{3}\right) + c_3 \sin(x) \end{gathered}}$$

$$\ell = \sum_{i=0}^{n} \left(P(x_i) - f_i\right)^2$$

Example:

$$\ell = \sum_{i=0}^{n} \left(P(x_i) - f_i\right)^2 \qquad P_m(x) = \sum_{k=0}^{m} c_k x^k$$

$$\{c_0, c_1, c_2, \cdots, c_m\}$$

$$\ell_m = \sum_{i=0}^{n} \left(\sum_{k}^{m} c_k x_i^k - f_i \right)^2$$

The least-squares approximation is obtained with ℓ_m minimized with respect to all the m + 1 coefficients through

$$\frac{\delta \ell_m}{\delta c_l} = 0$$

$$\boxed{\ell_m = \sum_{i=1}^{n}\left(\sum_{k}^{m} c_k x_i^k - f_i\right)^2 \qquad \frac{\delta \ell_m}{\delta c_l} = 0}$$

$$\frac{\delta \ell_m}{\delta c_l} = 2\sum_{i=1}^{n}\left(\sum_{k}^{m}\frac{\delta c_k}{\delta c_l} x_i^k\right)\left(\sum_{k}^{m} c_k x_i^k - f_i\right)$$

$$= 2\sum_{i=1}^{n}\left(\sum_{k}^{m}\delta_{kl} x_i^k\right)\left(\sum_{k}^{m} c_k x_i^k - f_i\right)$$

$$= 2\sum_{i=1}^{n} x_i^l\left(\sum_{k}^{m} c_k x_i^k - f_i\right)$$

$$\boxed{l : \sum_{i=1}^{n} x_i^l\left(\sum_{k}^{m} c_k x_i^k - f_i\right) = 0}$$

$$l \quad : \quad \sum_{i=0}^{n} x_i^l \left(\sum_{k=0}^{m} c_k x_i^k - f_i \right) = 0$$

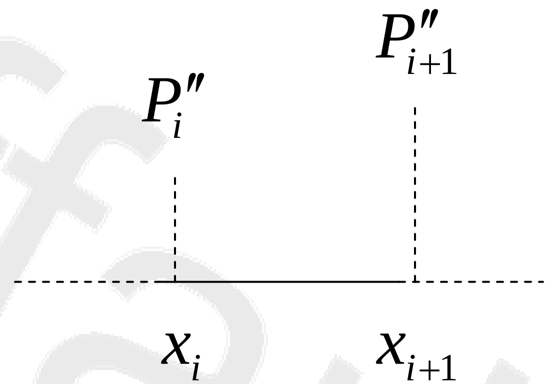
$$l \quad : \quad \sum_{k=0}^{m} c_k \left(\sum_{i=0}^{n} x_i^k x_i^l \right) = \sum_{i=0}^{n} f_i x_i^l$$

Spline approximation

we want to fit the function locally and to connect each piece of the function smoothly. A *spline* is such a tool; it interpolates the data locally through a polynomial and fits the data overall by connecting each segment of the interpolation polynomial by matching the **function and its derivatives at the data points.**

x	$f(x)$
x_0	f_0
x_1	f_1
x_2	f_2
$\vdots$	$\vdots$
x_n	f_n

$$P = P(x)$$



$$P''_i(x) = \frac{1}{(x_{i+1} - x_i)}\left((x - x_i)P''_{i+1} - (x - x_{i+1})P''_i\right)$$

$$P_i''(x) = \frac{1}{(x_{i+1} - x_i)}\Big((x - x_i)P''_{i+1} - (x - x_{i+1})P''_i\Big)$$

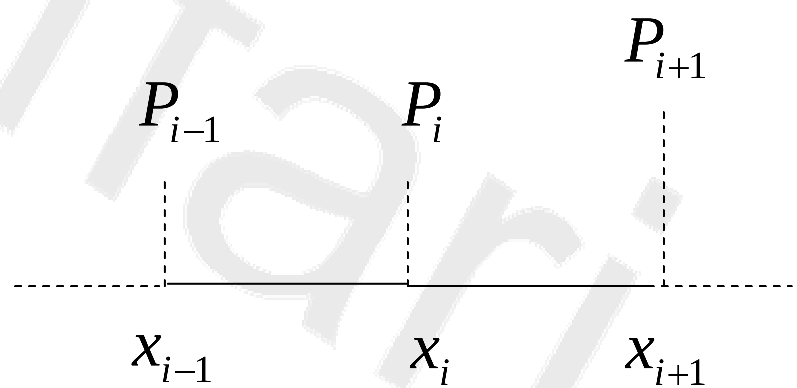
$$\boxed{P_i(x) = c_0(x - x_i) + c_1(x - x_{i+1}) + \frac{1}{6(x_{i+1} - x_i)}\Big((x - x_i)^3 P''_{i+1} - (x - x_{i+1})^3 P''_i\Big)}$$

$$h_i = x_{i+1} - x_i$$

$$\begin{matrix} P_i(x_i) = f_i \\ P_i(x_{i+1}) = f_{i+1} \end{matrix} \Longrightarrow \begin{cases} c_0 = \dfrac{f_{i+1}}{h_i} - \dfrac{h_i P''_{i+1}}{6} \\[2ex] c_1 = -\dfrac{f_i}{h_i} + \dfrac{h_i P''_i}{6} \end{cases}$$

$$P_i(x) = \left(\frac{f_{i+1}}{h_i} - \frac{h_i P''_{i+1}}{6}\right)(x - x_i) + \left(-\frac{f_i}{h_i} + \frac{h_i P''_i}{6}\right)(x - x_{i+1})$$
$$+ (x - x_i)^3 \frac{P''_{i+1}}{6h_i} - (x - x_{i+1})^3 \frac{P''_i}{6h_i}$$

$$P'_{i-1}(x_i) = P'_i(x_i)$$



$$P'_{i-1}(x)=\left(\frac{f_i}{h_{i-1}}-\frac{h_{i-1}P''_i}{6}\right)+\left(-\frac{f_{i-1}}{h_{i-1}}+\frac{h_{i-1}P''_{i-1}}{6}\right)+(x-x_{i-1})^2\,\frac{P''_i}{2h_i}-(x-x_i)^2\,\frac{P''_{i-1}}{2h_{i-1}}$$

$$P'_i(x)=\left(\frac{f_{i+1}}{h_i}-\frac{h_iP''_{i+1}}{6}\right)+\left(-\frac{f_i}{h_i}+\frac{h_iP''_i}{6}\right)+(x-x_i)^2\,\frac{P''_{i+1}}{2h_i}-(x-x_{i+1})^2\,\frac{P''_i}{2h_i}$$

$$P'_{i-1}(x_i)=P'_i(x_i)$$

$$h_{i-1}P''_{i-1}+2(h_{i-1}+h_i)P''_i+h_iP''_{i+1}=6\left(\frac{g_i}{h_i}-\frac{g_{i-1}}{h_{i-1}}\right)$$

$$\{P''_0,P''_1,P''_2,\cdots,P''_i,\cdots P''_n\}$$

$$h_{i-1}P''_{i-1} + 2(h_{i-1} + h_i)P''_i + h_i P''_{i+1} = 6\left(\frac{g_i}{h_i} - \frac{g_{i-1}}{h_{i-1}}\right)$$

$$\begin{pmatrix} d_1 & h_1 & 0 & \cdots & \cdots & 0 \\ h_1 & d_2 & h_2 & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & h_{n-3} & d_{n-2} & h_{n-2} \\ 0 & \cdots & \cdots & 0 & h_{n-2} & d_{n-1} \end{pmatrix} \begin{pmatrix} p''_1 \\ p''_2 \\ \vdots \\ p''_{n-2} \\ p''_{n-1} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-2} \\ b_{n-1} \end{pmatrix}.$$

$$d_i = 2(h_{i-1} + h_i) \quad , \quad b_i = 6\left(\frac{g_i}{h_i} - \frac{g_{i-1}}{h_{i-1}}\right)$$

Random-number generators

First, a good generator should have a long period.
Second, a good generator should have the best *randomness*.
Finally, a good generator has to be very fast.

simplest uniform random-number generator:

$$x_{i+1} = (ax_i + b) \bmod c$$

a, b, and c are magic numbers

$$a = 7^5 \quad , \quad b = 0 \quad , \quad c = 2^{31} - 1$$

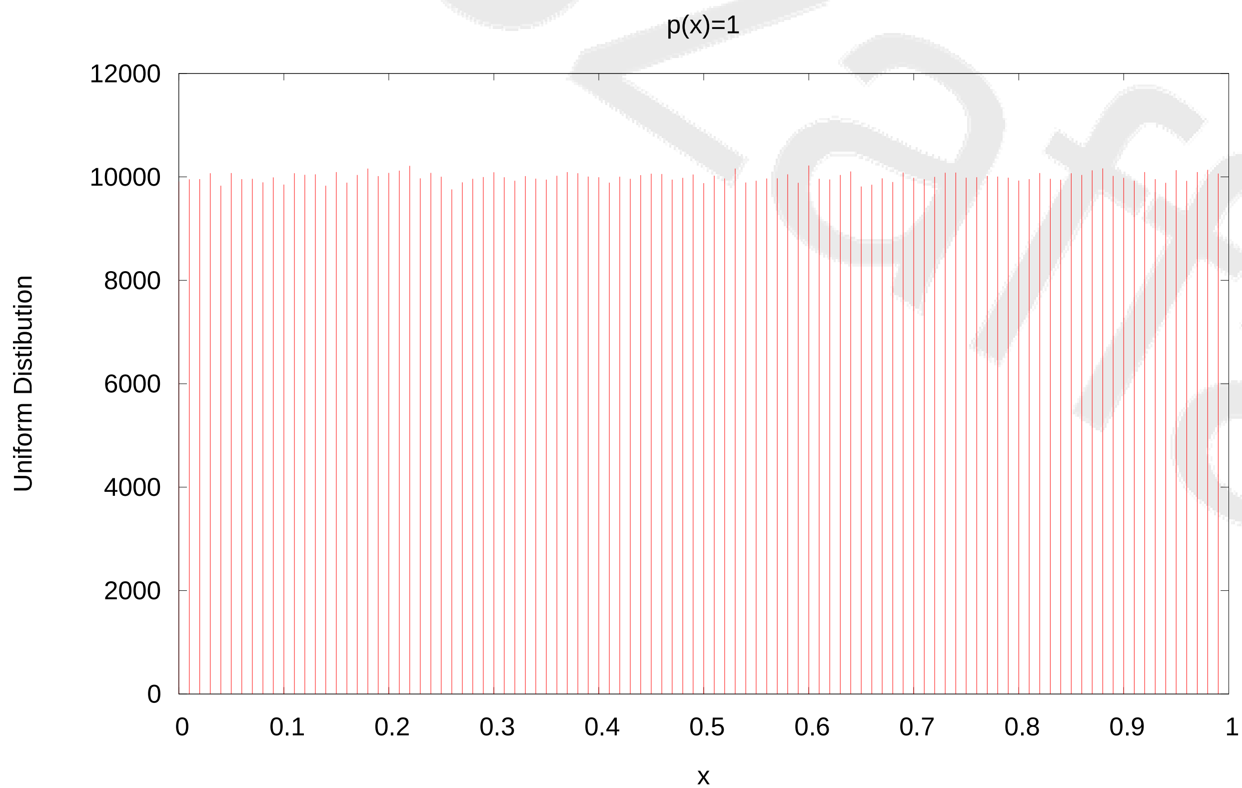
$$real :: ran$$

$$\vdots$$

$$call \quad random\_number(ran)$$

$$0 \le ran < 1$$

Uniform



$$0 \le x < 1$$

$$Number\ of\ \ randoms = 10^6$$

$$nt = 100$$

$$\Delta x = 1/nt$$

Uniform distribution

$$p(x) = \frac{1}{b-a} \quad , \quad x \in [a,b]$$

$$P(x) = \int_a^x p(x')dx' \Rightarrow P(x) = \frac{x-a}{b-a}$$

$$\downarrow$$

$$0 \le P(x) < 1$$

$$\downarrow$$

$$0 \le ran < 1 \longrightarrow \boxed{x = a + ran * (b-a)}$$

Exponential distribution

$$p(x) = e^{-x} \quad , \quad x \in [0, \infty]$$

$$P(x) = \int_a^x p(x')dx' = \int_a^x e^{-x'}dx' \Rightarrow P(x) = 1 - e^{-x}$$

$$\downarrow$$

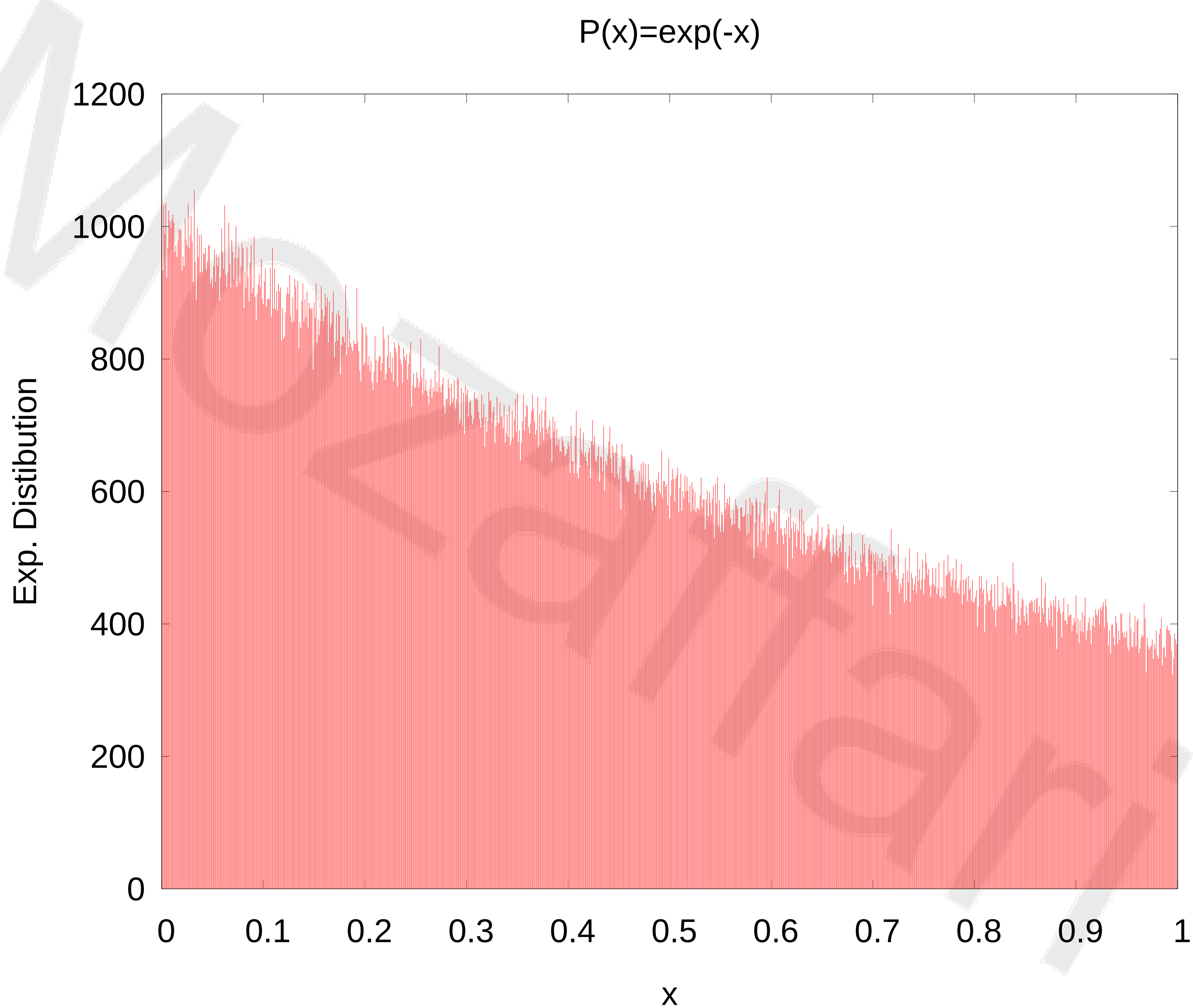
$$0 \le P(x) < 1$$

$$\downarrow$$

$$0 \le ran < 1$$

$$\downarrow$$

$$\boxed{x = -\log(1 - ran)}$$



P(x)=exp(-x)
1200
1000
800
600
400
200
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
x
Exp. Distibution

Gaussian distribution

$$p(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{x^2}{2\sigma^2}} \quad , \quad x\in[-\infty,+\infty]$$

$$\int_{-\infty}^{x}\int_{-\infty}^{y} e^{\frac{-x'^2}{2\sigma^2}} e^{\frac{-y'^2}{2\sigma^2}} dx'dy' = \int_0^r\int_0^\phi e^{\frac{-r'^2}{2\sigma^2}} r'dr'd\phi' = \left\{\int_0^\phi d\phi'\right\}\left\{\int_0^r e^{\frac{-r'^2}{2\sigma^2}} r'dr'\right\}$$

$$\begin{cases} x' = r'Cos(\phi') \\ y' = r'Sin(\phi') \end{cases} \quad , \quad \phi'\in[0,2\pi]$$

$$=\left\{\int_0^\phi d\phi'\right\}\left\{\int_0^t e^{-t'}dt'\right\}$$

$$r'=\sigma\sqrt{2t'} \quad , \quad t'\in[0,\infty]$$

$$\begin{cases} x = \sigma\sqrt{2t}Cos(\phi) \\ y = \sigma\sqrt{2t}Sin(\phi) \end{cases} \quad , \quad \phi \in [0,2\pi]$$

$$t = -\log(1 - ran1)$$

$$\phi = 2\pi * ran2$$

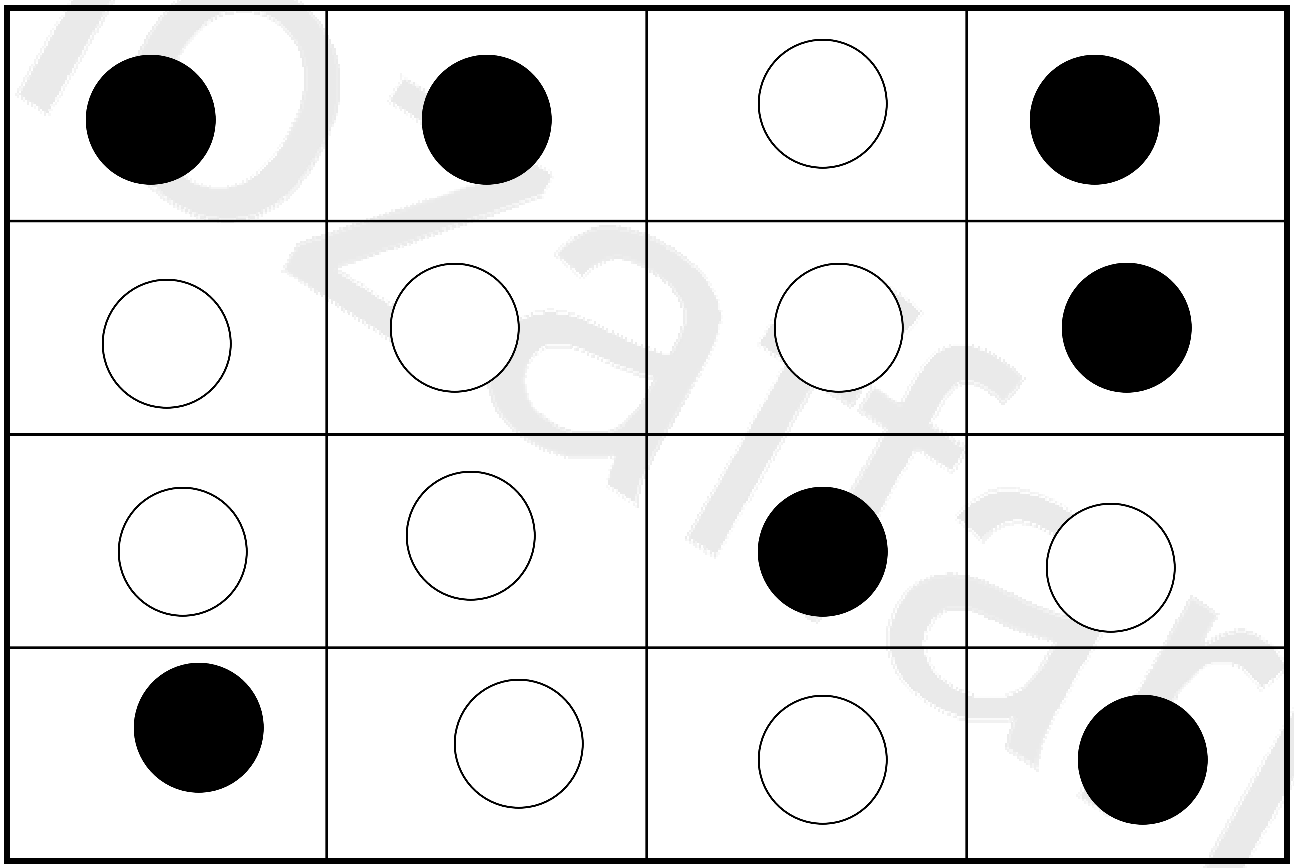
$$0 \le ran1 < 1$$

$$0 \le ran2 < 1$$

Percolation in two dimensions

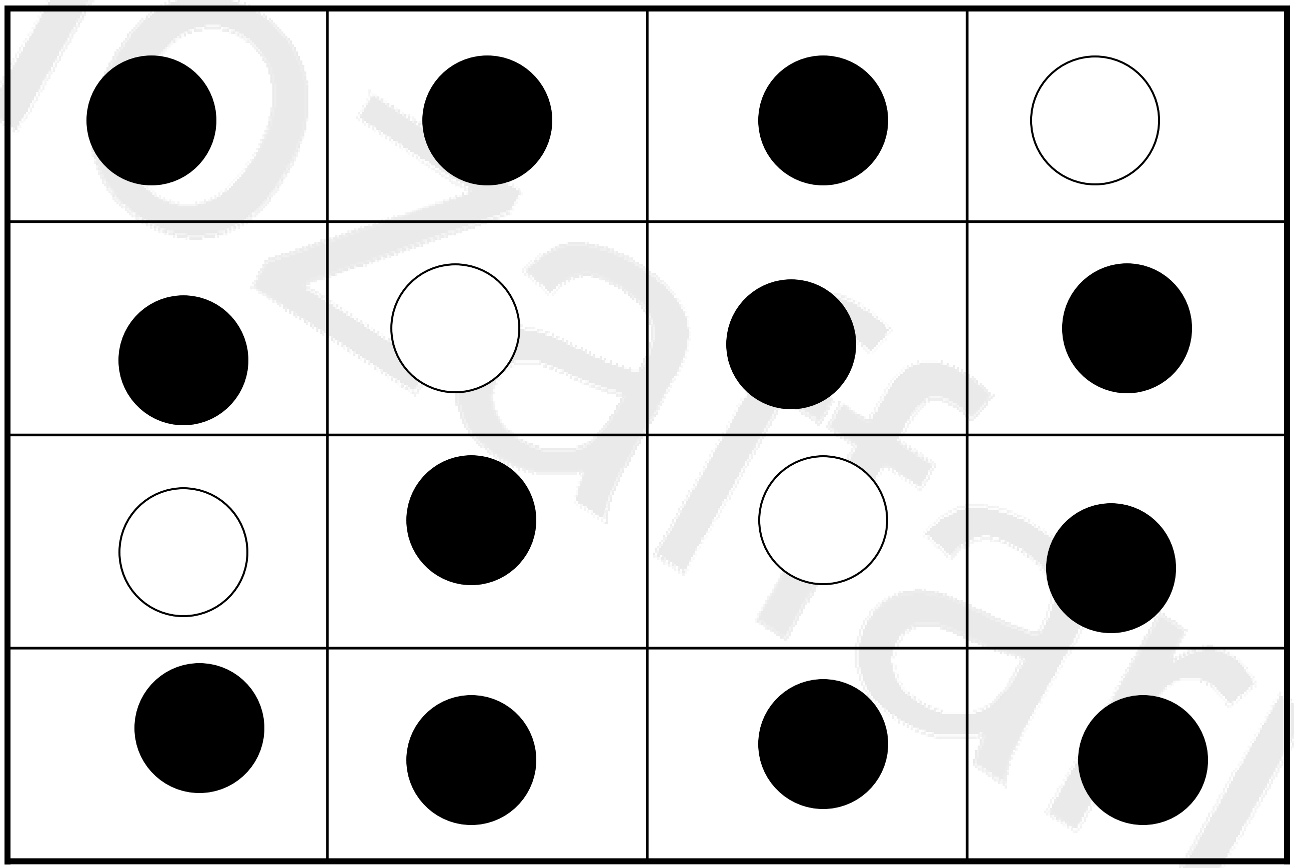
Assume that we have a two-dimensional square lattice with $n \times n$ lattice points.

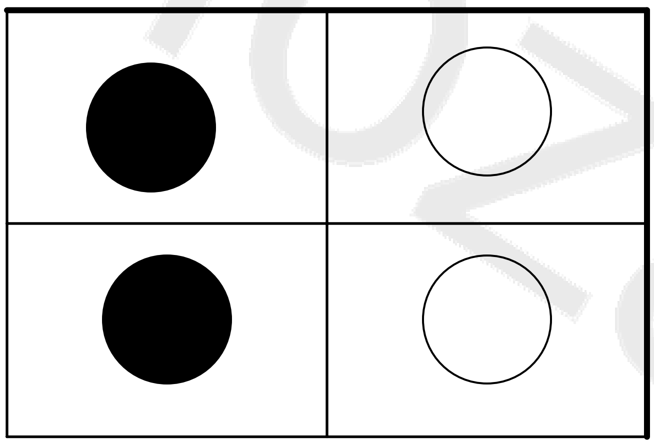
$$ran \leq p$$



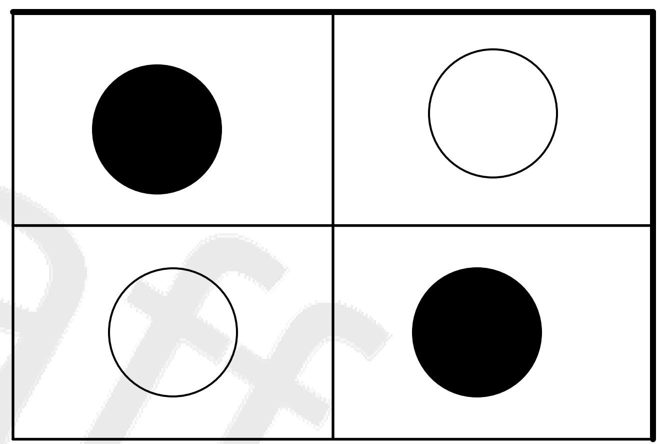
Threshold percolation

p_c





first neighbor



second neighbor