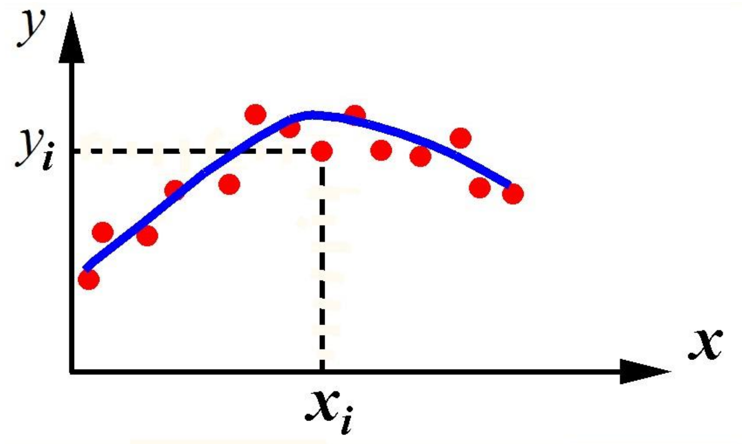
Chapter 7. Least-Squares approximation

(Least-Squares fitting procedure)

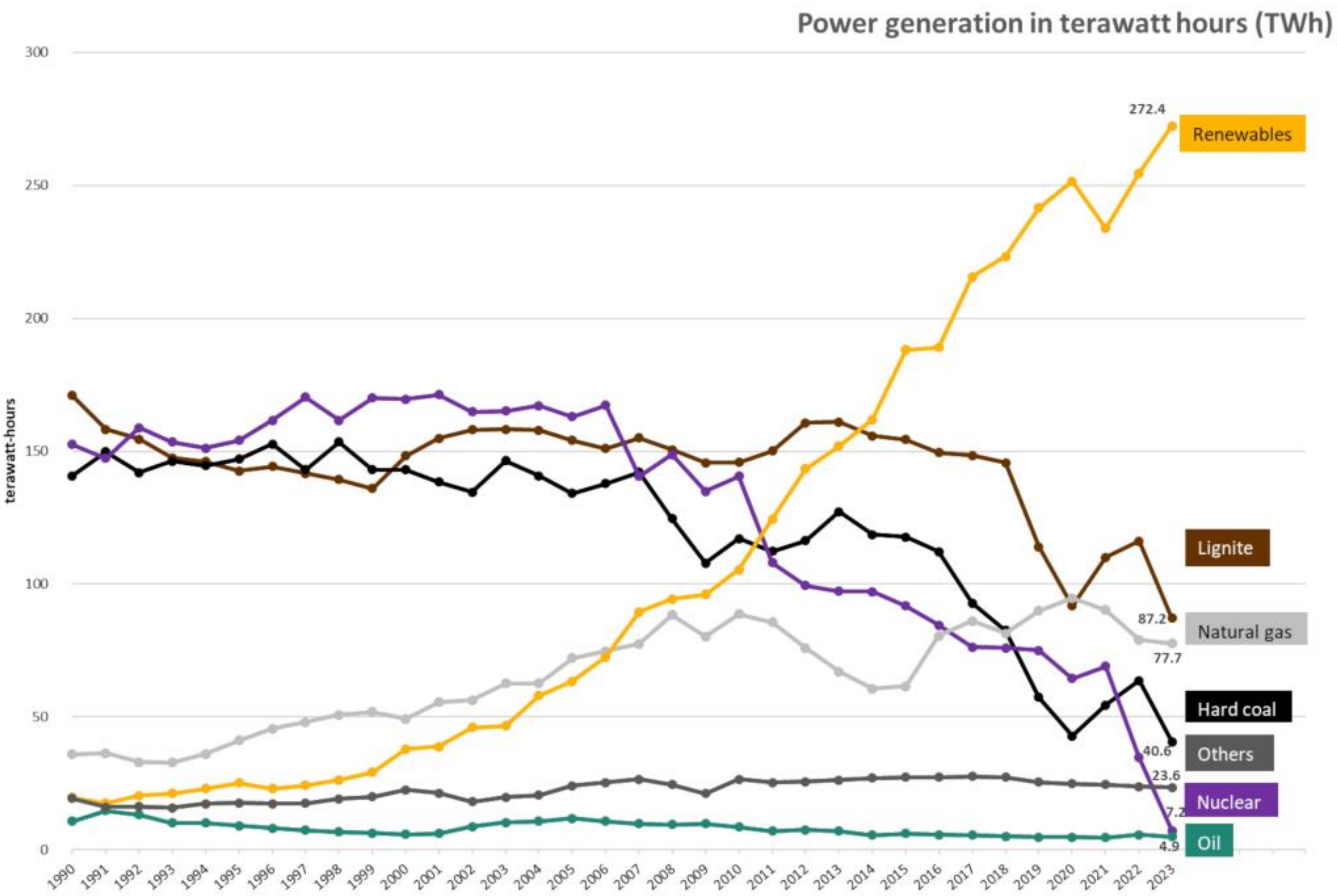


Suppose that given values x_i , y_i
show fluctuations, deviations, from some smooth curve (for example, because of influence of random factors).

Problem: Find a function $p(x)$ that is smooth and well approximates x_i , y_i .

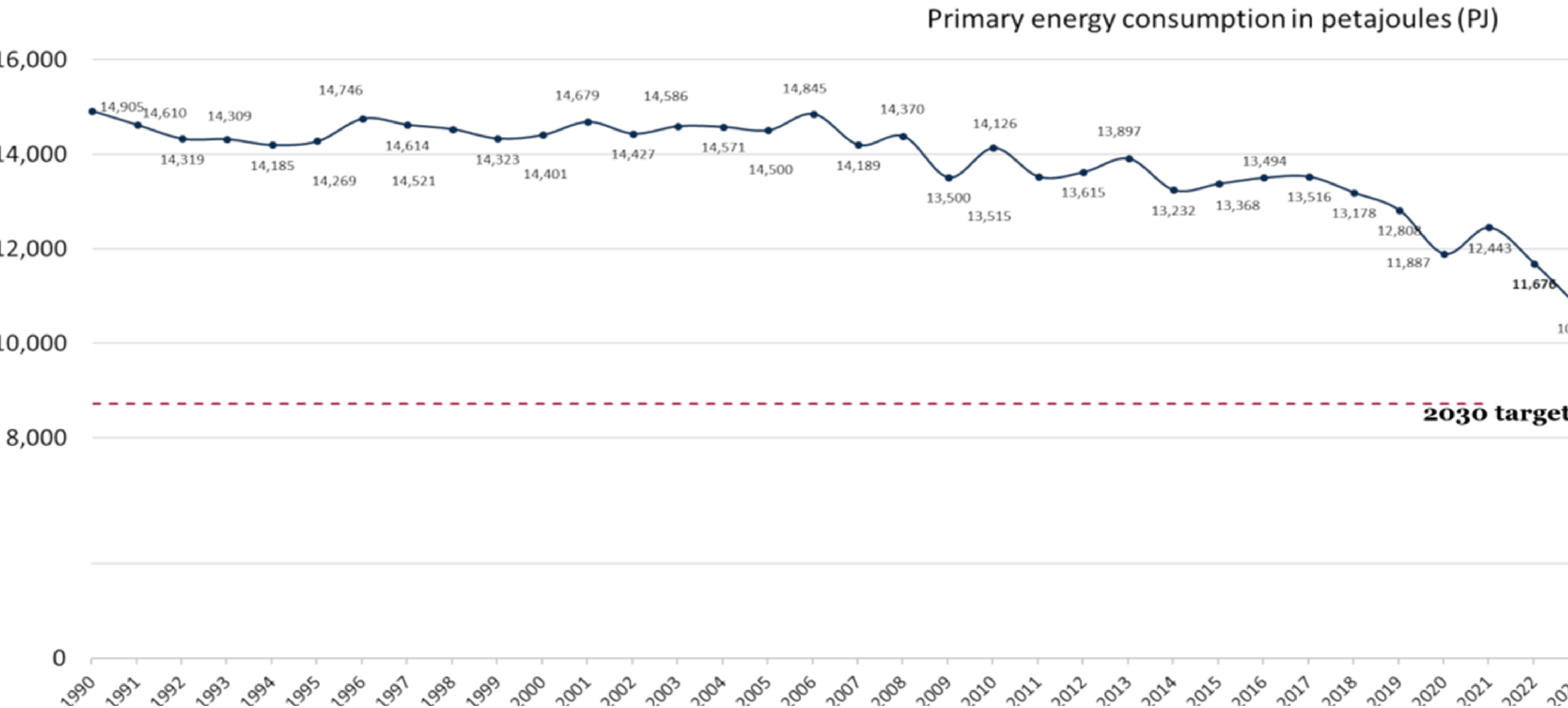
Gross power production in Germany 1990 - 2023, by source.

Data: AGEB 2024.



Development of Germany's primary energy consumption 1990 - 2023.

Data: AG Energiebilanzen 2024.

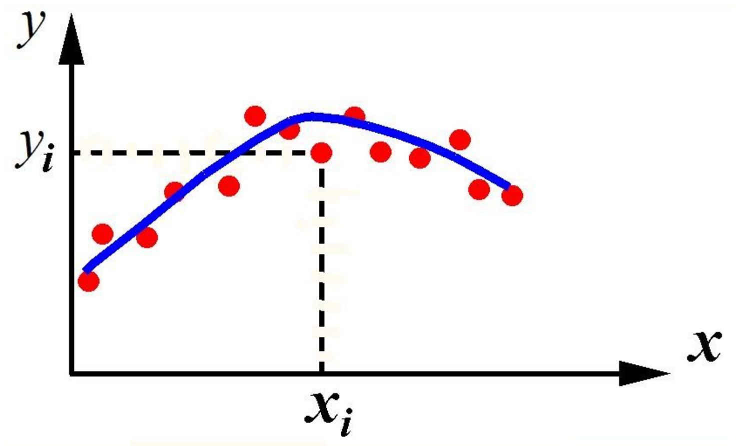


Primary energy is the energy found in nature that has not been subjected to any human engineered conversion process. wiki

Consider a polynomial

$p(x)=a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m$, $m<n$

We can use either condition



$$\sum_{i=0}^{n} | p(x_i) - y_i| \rightarrow \text{min}$$

or

$$\sum_{i=0}^{n} [\, p(x_i) - y_i \,]^2 / (n+1) \rightarrow \text{min}$$ (title of method)

$$\sum_{i=0}^{n} [\, p(x_i) - y_i \,]^2 \;\rightarrow\; \min$$

$$\sum_{i=0}^{n} [a_0 + a_1 x_i + \ldots + a_k x_i^k + \ldots + a_m x_i^m - y_i \,]^2 \rightarrow \min$$

The sum involves *n*+1 square brackets

This is the condition for finding a_i

Let us denote the left-hand side by

$$J(a_0, a_1, \ldots, a_k, \ldots, a_m) = \sum_{i=0}^{n} [\, p(x_i) - y_i \,]^2$$

This is a function of *m+1* variables a_j

Necessary condition of a minimum: $\partial J/\partial a_k = 0$

$$\partial J/\partial a_k = \sum_{i=0}^{n} 2[\,p(x_i) - y_i] \cdot \partial p(x_i)/\partial a_k = 0$$

$$\sum_{i=0}^{n} [\,p(x_i) - y_i] \cdot x_i^k = 0 \qquad k=0,1,\ldots m$$

$$\sum_{i=0}^{n} [a_0 + a_1 x_i + \ldots + a_m x_i^m\,] \cdot x_i^k = \sum y_i\, x_i^k$$

$$\sum_{i=0}^{n} [a_0 x_i^k + a_1 x_i^{k+1} + \ldots + a_m x_i^{k+m}\,] = \sum y_i\, x_i^k$$

and we get the system of $m+1$ **linear equations,** $k=0,1,\ldots m$**:**

$$a_0 \sum_{i=0}^{n} x_i^k + a_1 \sum_{i=0}^{n} x_i^{k+1} + \ldots + a_m \sum_{i=0}^{n} x_i^{k+m} = \sum_{i=0}^{n} y_i\, x_i^k \qquad (*)$$

<u>Theorem</u> If $m \le n$, then there exists a unique solution of the system of equations (*) with respect to coefficients a_k of the polynomial $p(x)$ (proof is omitted).

In particular, at m=n this polynomial is equivalent to Lagrange's polynomial (which passes through each node).

We used the necessary condition of minimum $\partial J/\partial a_k = 0$.

The fact that obtained a_k determine indeed a minimum, not maximum, is evident from the quadratic dependence of

$$J(a_0, a_1, + \dots + a_m) = \sum_{i=0}^{n} [\, p(x_i) - y_i \,]^2 \quad \textbf{on} \quad a_k,$$

$$p(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_m x^m$$

Scilab

[a,err]=datafit(J, xy, a0)

xy - matrix of given x_i , y_i

J - function that defines the approximating polynomial and the difference between its values and y_i

a - obtained coefficients of the approximating polynomial

a0 – initial approximation

err - obtained sum of squared differences

Scilab

```
function [fun]=J(a, xy)
fun=xy(2)-a(1)-a(2)*xy(1)-a(3)*xy(1)^2-a(4)*xy(1)^3
endfunction
x=[1.3   1.4   1.5      1.6    1.7      1.8];
y=[3.3   3.4   3.85   4.25   4.50    4.85];
xy=[x;y];
plot(x,y,'o','LineWidth',3);
//  initial approximation:
a0=[0;0;0;0]
// Solution:
[a,err]=datafit(J,xy,a0)
t=1.3:0.01:1.8;
poly=a(1)+a(2)*t+a(3)*t.^2+a(4)*t.^3;
plot(t,poly,'r');
xgrid
disp(t(41), poly(41))
```

Notice. When you choose the degree ***m***, you should take into consideration the physics of the dependence $y_i(x_i)$ and the number of bends you want in fitted line.

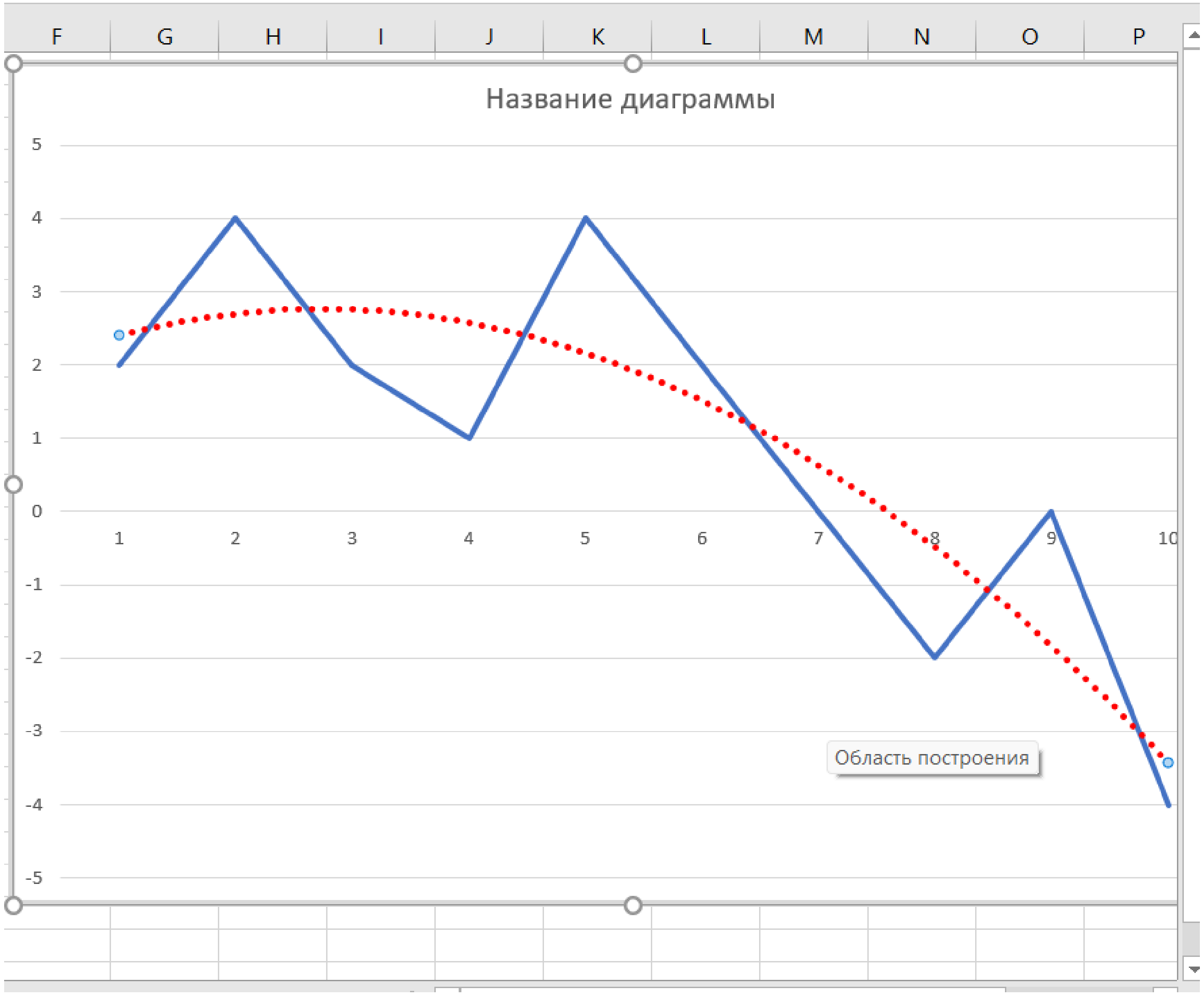
Each increase in the degree ***m*** produces one more bend in the fitted line.

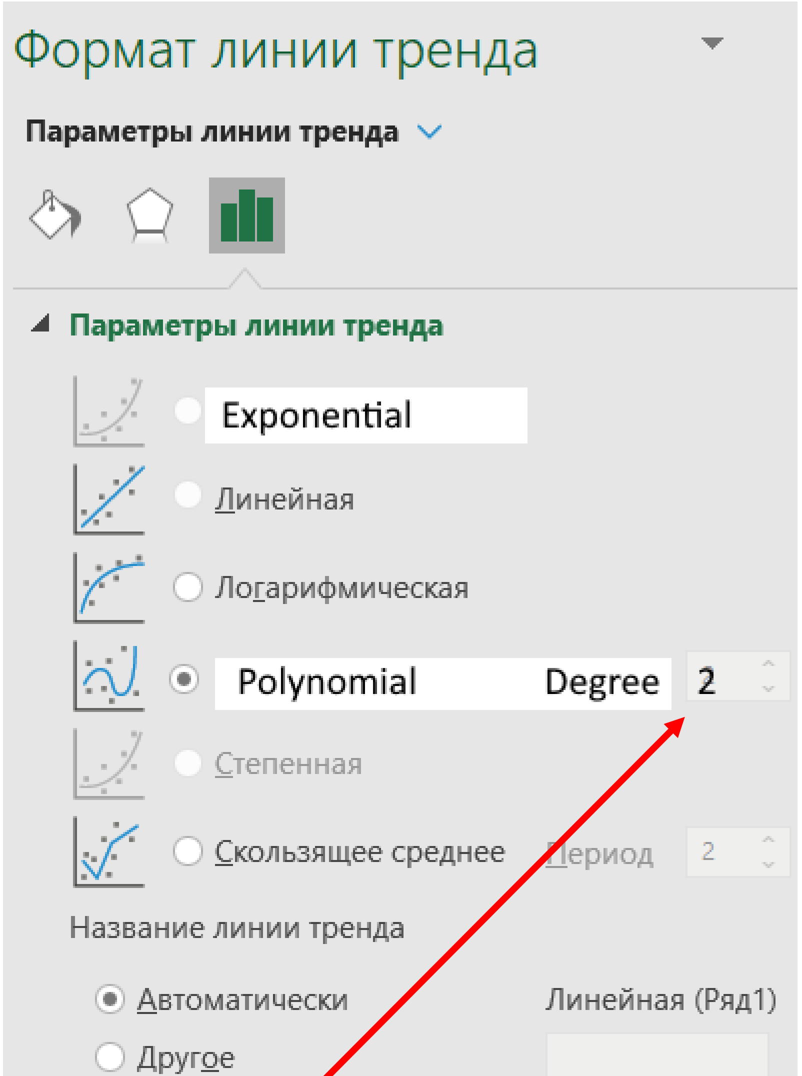
$$p(x) = a_0 + a_1x + a_2x^2 + \ldots + a_mx^m$$

Sometimes it is reasonable to use an exponential fitting curve ae^{bx} ***or*** ab^x **,**

or logarithm $a + \ln(bx)$

EXCEL





It is recommended to try several values of degree

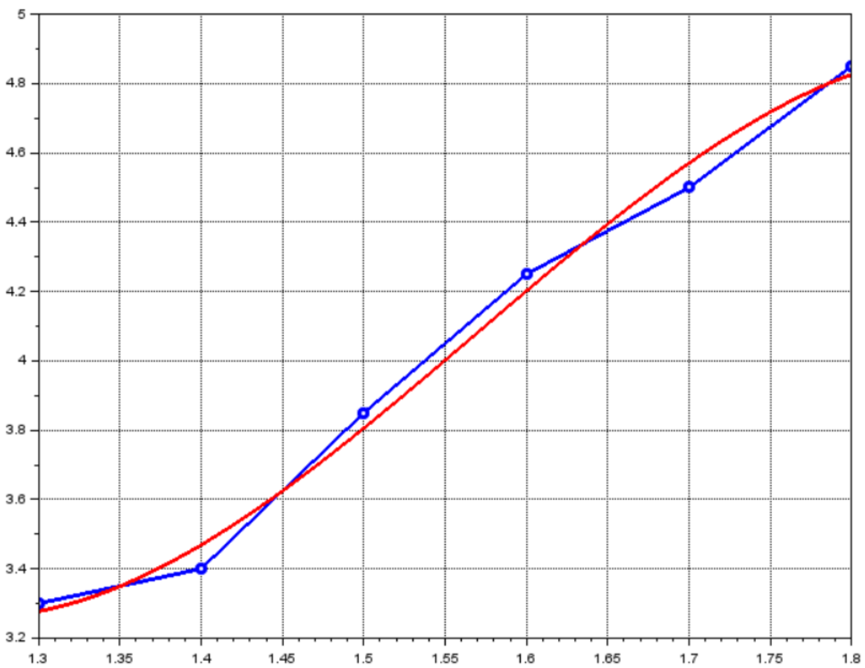
3 pages above, we considered the example of fitting the function

x_i= 1.3 1.4 1.5 1.6 1.7 1.8

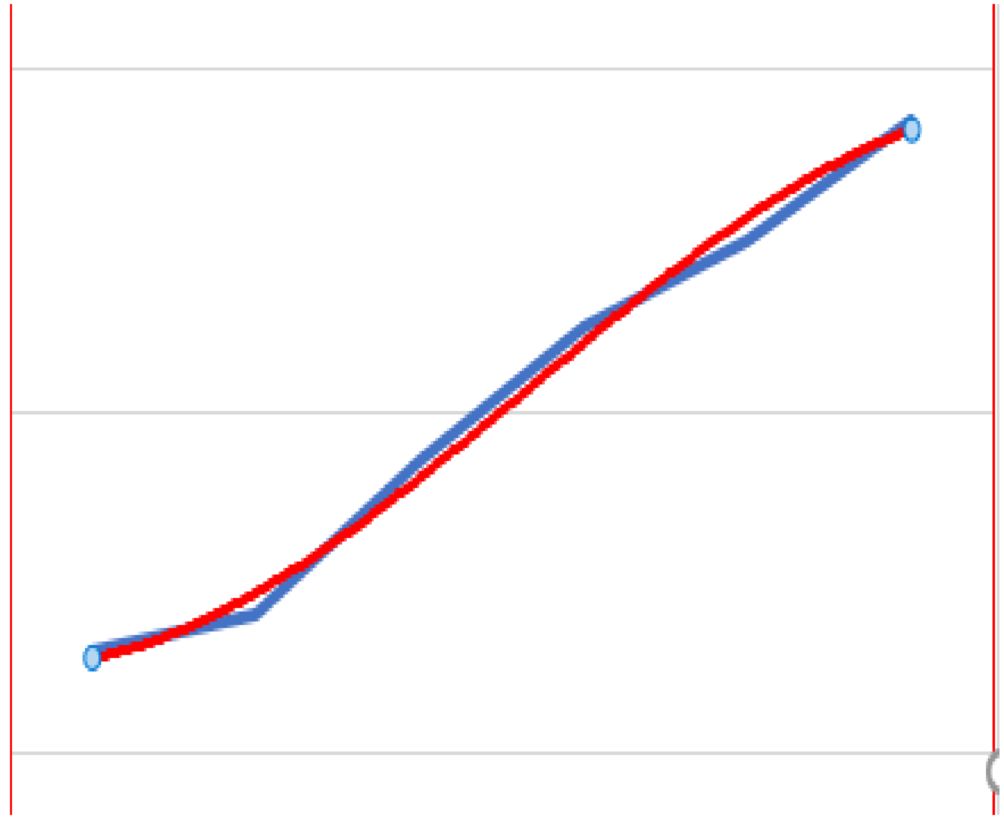
y_i= 3.3 3.4 3.85 4.25 4.50 4.85

with a polynomial of degree m=3

The result produced by Scilab was:



The result produced by EXCEL:



https://www.wolframalpha.com/input?i=curve+fitting

WolframAlpha

curve fitting

NATURAL LANGUAGE | MATH INPUT | EXTENDED KEYBOARD | EXAMPLES | UPLOAD | RANDOM

Examples for

Regression Analysis

Regression

Fit a line to two-dimensional data:

linear fit {1.3, 2.2},{2.1, 5.8},{3.7, 10.2},{4.2, 11.8}

Fit a line to sequential data:

linear fit 104, 117, 131, 145, 160, 171

linear fit

Fit a polynomial to given data:

quadratic fit {10.1,1.2},{12.6, 2.8},{14.8,7.6},{16.0,12.8},{17.5,15.1}

cubic fit 20.9,23.2,26.2,26.4,16.3,-12.2,-60.6,-128.9

Fitting by a logarithm *a + ln (bx)*

» data set of y values: {1.8, 2.4, 2.7, 3.1, 3.2, 3.4, :

Compute

Assuming data set of y values | Use data set of {x,y} values

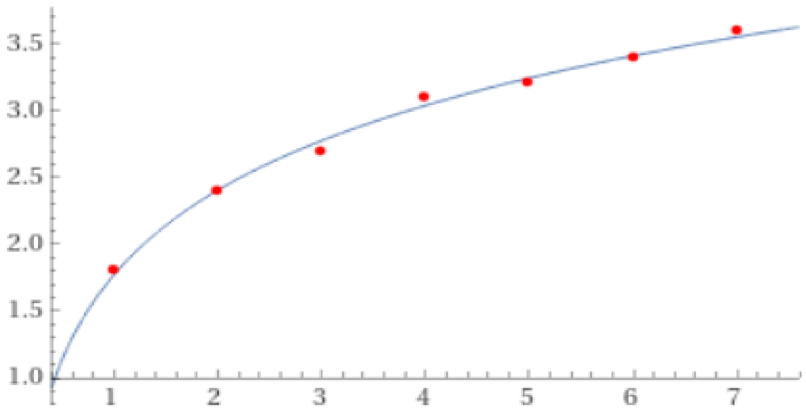
Input interpretation

fit		
	data	{1.8, 2.4, 2.7, 3.1, 3.2, 3.4, 3.6}
	model	logarithmic

Least-squares best fit

$0.914619 \log(6.93942\, x)$

Plot of the least-squares fit



Notice. Polynomials

$$p(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m \qquad (**)$$

where m>10 are difficult to use in practice, as determinant of the system (*) becomes very close to 0, and the system becomes ill-conditioned.

Then, for least squares approximation, one can use Chebyshev polynomials $T_k(x)$ instead of (**):

$$a_0 T_0 + a_1 T_1(x) + a_2 T_2(x) + \ldots + a_m T_m(x)$$

$$\sum_{i=0}^{n} [a_0 T_0 + a_1 T_1(x_i) + \ldots + a_k T_k(x_i) + \ldots + a_m T_m(x_i) - y_i]^2 \rightarrow$$

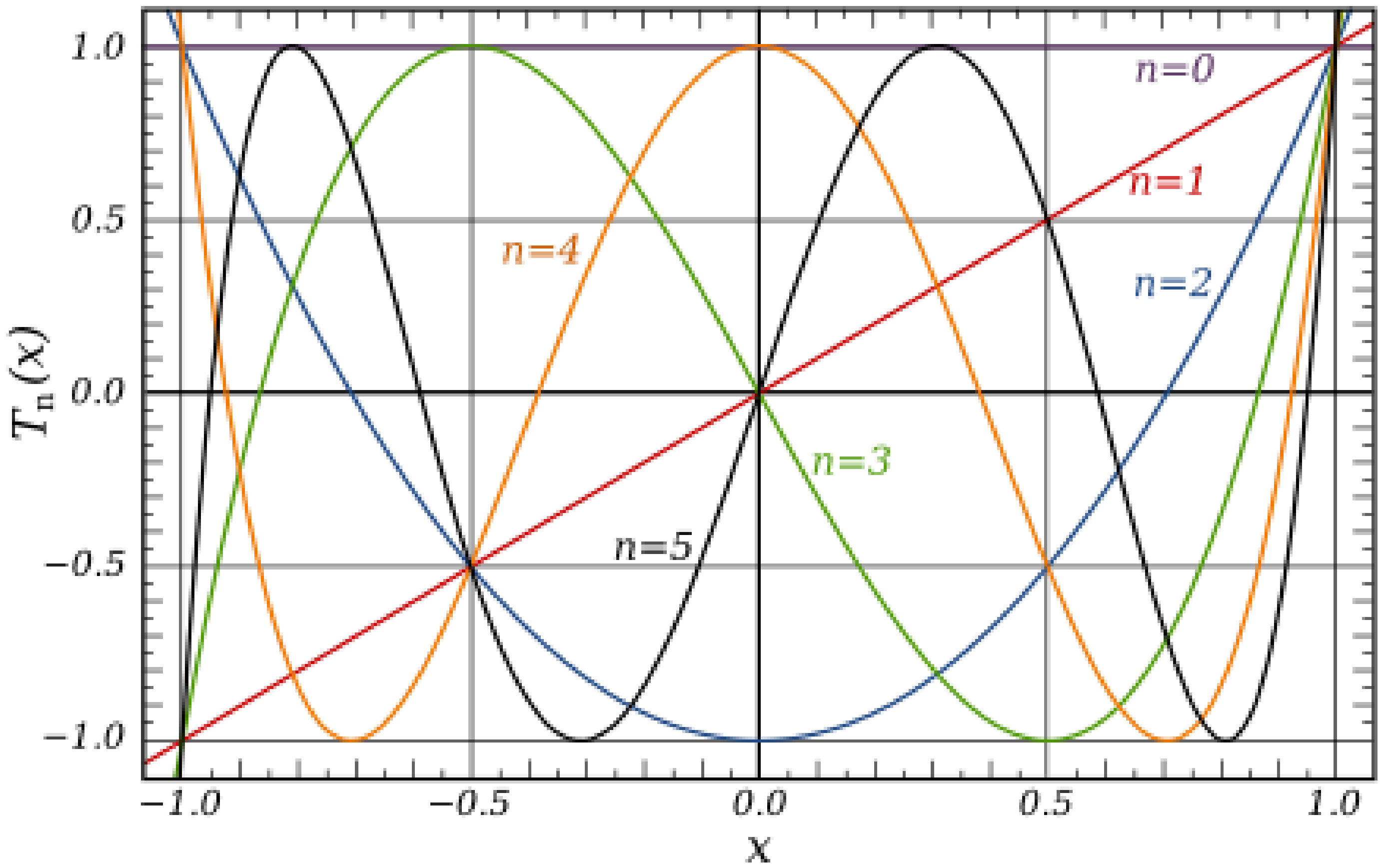
$$\rightarrow \min$$

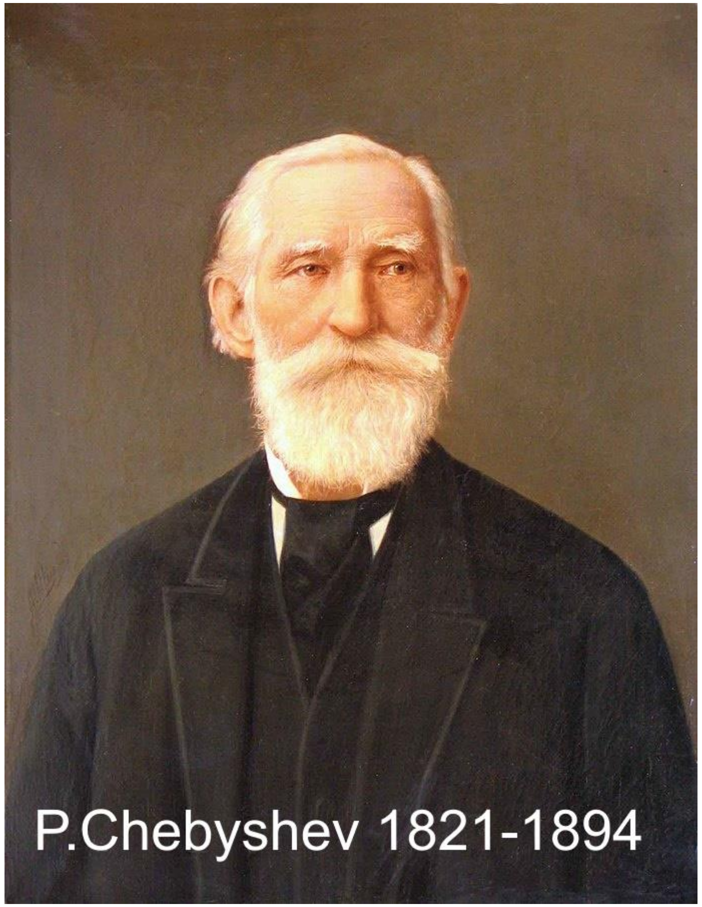
Definition of Chebyshev polynomials :

$$T_0=1 \qquad T_1(x)=x \qquad T_2(x)=2x^2-1$$

$$T_k(x)=2xT_{k-1}(x)-T_{k-2}(x), \qquad k=3,4,\ldots$$

$$-1 \leq x \leq 1$$





For a different interval, one can use the substitution

$$t=(2x-x_0-x_n)/(x_n-x_0)$$

Least Squares for a function of 2 variables

Let function $z=f(x,y)$ be given by a table

(x_0, y_0, z_0), (x_1, y_1, z_1), . . . , (x_n, y_n, z_n)

In the following example, a linear function

$p(x,y) = a_0 + a_1 x + a_2 y$

is used for the least-squares approximation of z .

We should choose a_k to minimize the sum:

$$J = \sum_{i=0}^{n} (z_i - a_0 - a_1 x_i - a_2 y_i)^2$$

$$\frac{\partial J}{\partial a_0} = -2\sum (z_i - a_0 - a_1 x_i - a_2 y_i) = 0 ,$$

$$\frac{\partial J}{\partial a_1} = -2\sum x_i (z_i - a_0 - a_1 x_i - a_2 y_i) = 0 ,$$

and

$$\frac{\partial J}{\partial a_2} = -2\sum y_i (z_i - a_0 - a_1 x_i - a_2 y_i) = 0 .$$

These equations simplify to

$$\left.\begin{aligned} (n+1)a_0 + a_1 \Sigma x_i + a_2 \Sigma y_i &= \Sigma z_i \\ a_0 \Sigma x_i + a_1 \Sigma x_i^2 + a_2 \Sigma x_i y_i &= \Sigma z_i x_i \\ a_0 \Sigma y_i + a_1 \Sigma y_i x_i + a_2 \Sigma y_i^2 &= \Sigma z_i y_i \end{aligned}\right\}$$

from which a_0, a_1 and a_2 can be determined.