Least Squares

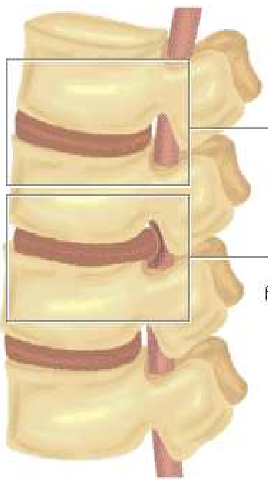
Numerical Analysis

Profs. Gianluigi Rozza- Luca Heltai

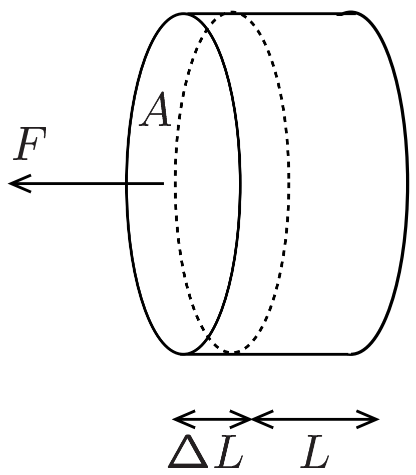
2019-SISSA mathLab Trieste

Examples and motivations

Example 1. We consider a mechanical test to establish the link between stresses ($MPa = 100N/cm^2$) and relative deformations (cm/cm) of a sample of biological tissue (an intervertebral disc, taken from P. Komarek, Chapt. 2 of *Biomechanics of Clinical Aspects of Biomedicine*, 1993, J. Valenta ed., Elsevier).



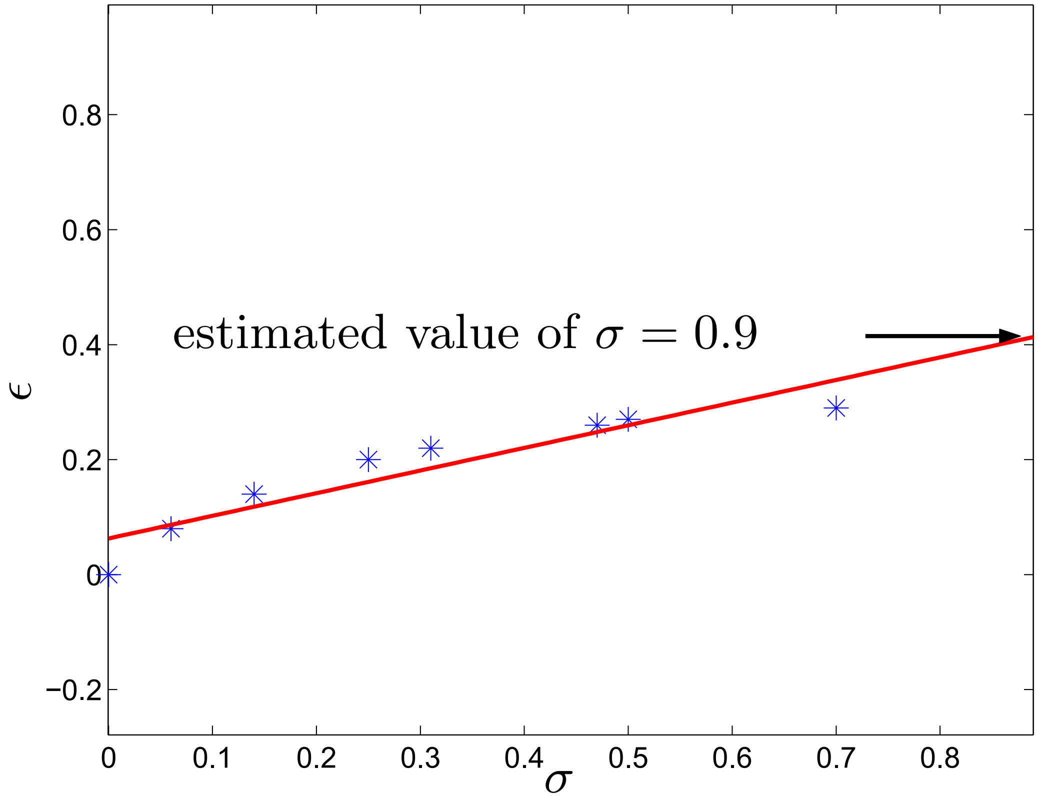
test i	pressure σ	deformation ϵ
1	0.00	0.00
2	0.06	0.08
3	0.14	0.14
4	0.25	0.20
5	0.31	0.23
6	0.47	0.25
7	0.60	0.28
8	0.70	0.29



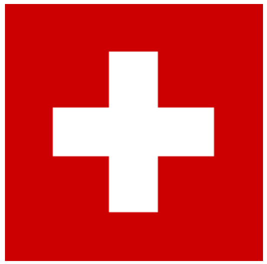
$$\sigma = F/A$$
$$\epsilon = \Delta L/L$$

From these data, we want to estimate the deformation corresponding to a stress $\sigma = 0.9$ MPa.

By the method of least squares, we get that best approximation for the data is $p(x) = 0.3938x - 0.0629$. The approximation can be used (called *a linear regression*) to estimate ϵ when $\sigma = 0.9$ MPa: We get $p(0.9) \simeq 0.4$.



Example 2. The result of census of the population of Switzerland between 1900 and 2010 (in thousands):

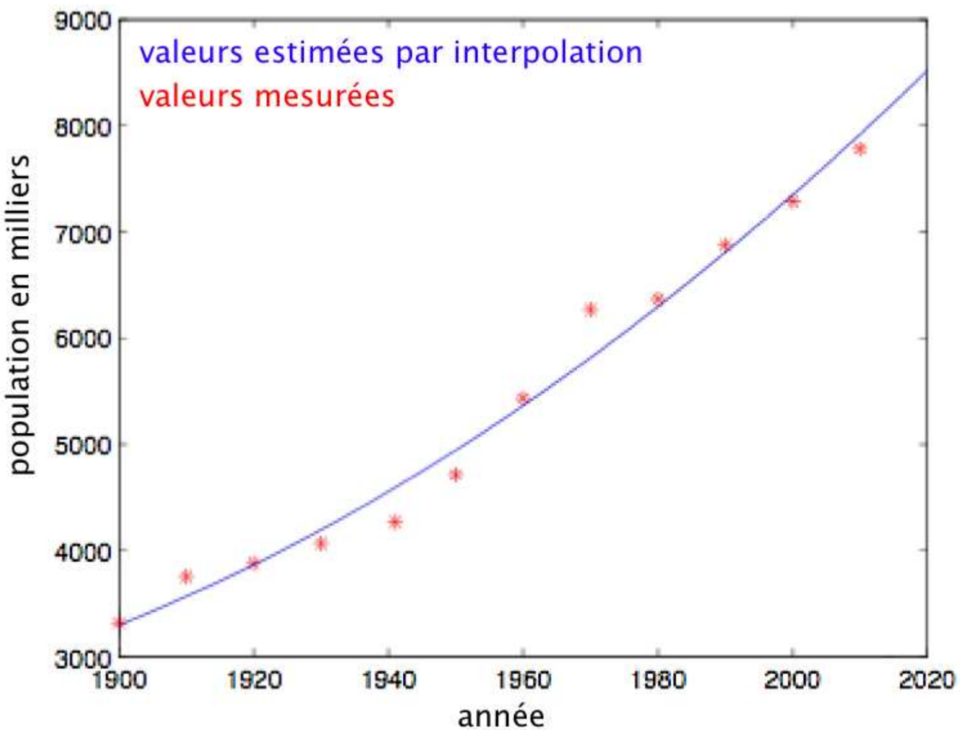


year	1900	1910	1920	1930	1941	1950
population	3315	3753	3880	4066	4266	4715

year	1960	1970	1980	1990	2000	2010
population	5429	6270	6366	6874	7288	7783

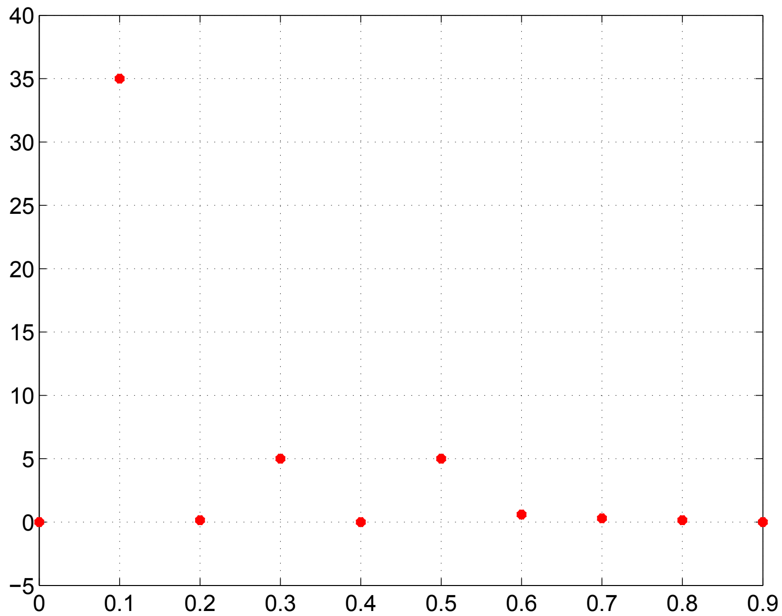
- Is it possible to estimate the number of inhabitants of Switzerland during the year when there has not been census, for example in 1945 and 1975?
- Is it possible to predict the number of inhabitants of Switzerland in 2020?

The polynomial of degree two (parabola) which approximates the data by the method of least squares is $p(x) = 0.15x^2 - 549.9x + 501600$.



Example 3. Points on the following figure represent the measurements of blood flow in a part of the common carotid artery during a heartbeat. The frequency of data collection is constant and equal to 10 / T where T = 1 sec. is the period of the beat.

We want to deduce from this discrete signal a continuous signal represented by a linear combination of known functions (eg. trigonometric functions - then we can adequately approximate a periodic signal).



Let $f(t)$ be a signal that we know. $N = 10$ is a size of a sample $[f(t_0), \ldots, f(t_{N-1})]$, where $t_j = jT/N$. We are looking for $\{c_k\} \in \mathbb{C}$, $k \in [0, N-1] \subset \mathbb{N}$ such that:

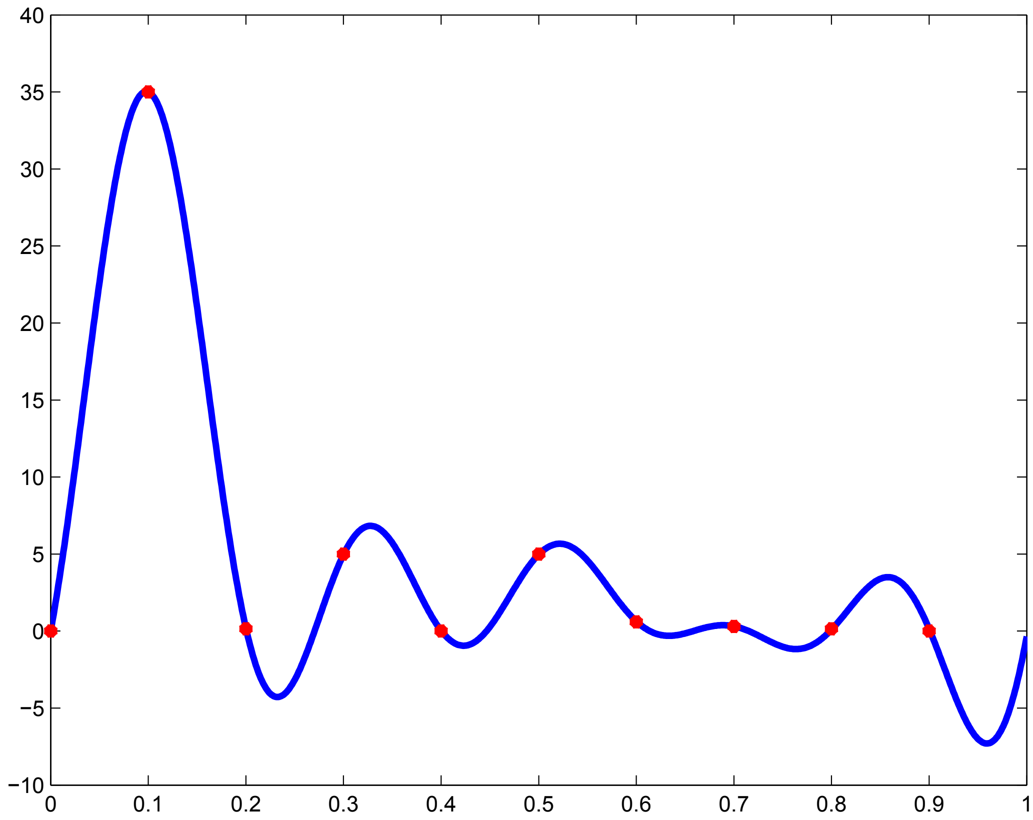
$$f(t_j) = \frac{1}{N} \sum_{k=0}^{N-1} c_k \omega_N^{-kj}, \quad j = 0, \ldots, N-1, \tag{1}$$

where

$$\omega_N = e^{(-2\pi i)/N} = \cos(2\pi/N) - i \sin(2\pi/N),$$

i is the imaginary unit. We can calculate a vector of the coefficients $\mathbf{c} = [c_1, \ldots, c_{10}]^T$ by the FFT algorithm (Fast Fourier Transform, Cooley & Tuckey, 1965), implemented in Matlab/Octave by the `fft` command.

Using the formula (1), there are several techniques for obtaining the value of f in each point $t \in [0, T]$.
Using one of techniques we obtain a result that is plotted on a following figure:



Approximation by the least square metho

(Chapt. 3.4 of the book)

Suppose we have $n + 1$ points $x_0, x_1, \ldots, x_n$ and $n + 1$ values $y_0, y_1, \ldots, y_n$. We have seen that if n is large, the interpolating polynomial may show large oscillations

Instead of interpolating the values, it is possible to define a polynomial of degree $m < n$ that approximates the data “at best”

Definition 1. *We call least squares polynomial approximation of degree m the polynomial $\tilde{f}_m(x)$ of degree m such that*

$$\boxed{\sum_{i=0}^{n} |y_i - \tilde{f}_m(x_i)|^2 \leq \sum_{i=0}^{n} |y_i - p_m(x_i)|^2 \qquad \forall p_m(x) \in \mathbb{P}_m}$$

Remark 4. *Where $y_i = f(x_i)$ (f is a continuous function) then $\tilde{f}_m$ is called the approximation of f in the least squares sense.*

In other words, the least squares polynomial approximation is the polynomial of degree m that minimizes the distance to the data.
Let note $\tilde{f}_m(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m$ and define the function

$$\Phi(a_0, a_1, \ldots, a_m) = \sum_{i=0}^{n} \left| y_i - \left(a_0 + a_1 x_i + a_2 x_i^2 + \ldots + a_m x_i^m\right) \right|^2$$

Then the coefficients of $\tilde{f}^m$ can be determined by the relation

$$\frac{\partial \Phi}{\partial a_k} = 0, \qquad k = 0, ..., m, \tag{5}$$

i.e., $m+1$ linear equations in with $m+1$ unknowns a_k, $k = 0, ..., m$.

Example 8. Let $n = 2$ and $m = 1$, nodes $x_0 = 1$, $x_1 = 3$, $x_2 = 4$ with values $y_0 = 0$, $y_1 = 2$, $y_2 = 7$. We want to compute the (*regression line*), i.e., the the least squares polynomial approximation of degree 1, $\tilde{f}_1(x) = a_0 + a_1 x$.

We set $\Phi(a_0, a_1) = \sum_{i=0}^{2}[y_i - (a_0 + a_1 x_i)]^2$ and impose $\dfrac{\partial \Phi}{\partial a_0} = 0$ and $\dfrac{\partial \Phi}{\partial a_1} = 0$:

$$
\begin{aligned}
\frac{\partial \Phi}{\partial a_0} &= -2\sum_{i=0}^{2}[y_i - (a_0 + a_1 x_i)] = -2\left(\sum_{i=0}^{2} y_i - 3a_0 - a_1 \sum_{i=0}^{2} x_i\right) \\
&= -2(9 - 3a_0 - 8a_1) \\
\frac{\partial \Phi}{\partial a_1} &= -2\sum_{i=0}^{2} x_i[y_i - (a_0 + a_1 x_i)] = -2\left(\sum_{i=0}^{2} x_i y_i - a_0 \sum_{i=0}^{2} x_i - a_1 \sum_{i=0}^{2} x_i^2\right) \\
&= -2(34 - 8a_0 - 26a_1)
\end{aligned}
$$

Hence the coefficients a_0 and a_1 are the solution of the system

$$
\left\{
\begin{array}{rcrcr}
3a_0 & + & 8a_1 & = & 9 \\
8a_0 & + & 26a_1 & = & 34
\end{array}
\right.
$$

Ideally, for $\tilde{f}_m(x) = a_0 + a_1 x + a_2 x^2 + \ldots$ we would like to impose $\tilde{f}_m(x_i) = y_i$ pour $i = 0, \ldots, n$.
This can be written as a linear system with unknowns a_k, $k = 0, \ldots, m$: $B\mathbf{a} = \tilde{\mathbf{y}}$, where B is a matrix of dimension $(n+1) \times (m+1)$

$$B = \begin{pmatrix} 1 & x_0 & \ldots & x_0^m \\ 1 & x_1 & \ldots & x_1^m \\ \vdots & & & \vdots \\ 1 & x_n & \ldots & x_n^m \end{pmatrix}$$

Since $m < n$, the system is oversized. The solution to (5) is equivalent to the square system (*system of normal equations*)

$$\boxed{B^T B\mathbf{a} = B^T \tilde{\mathbf{y}}.}$$

Example 9. We have 8 measures (σ-ϵ):

```
>> sigma = [0.00 0.06 0.14 0.25 0.31 0.47 0.50 0.70];
>> epsilon = [0.00 0.08 0.14 0.20 0.22 0.26 0.27 0.29];
```

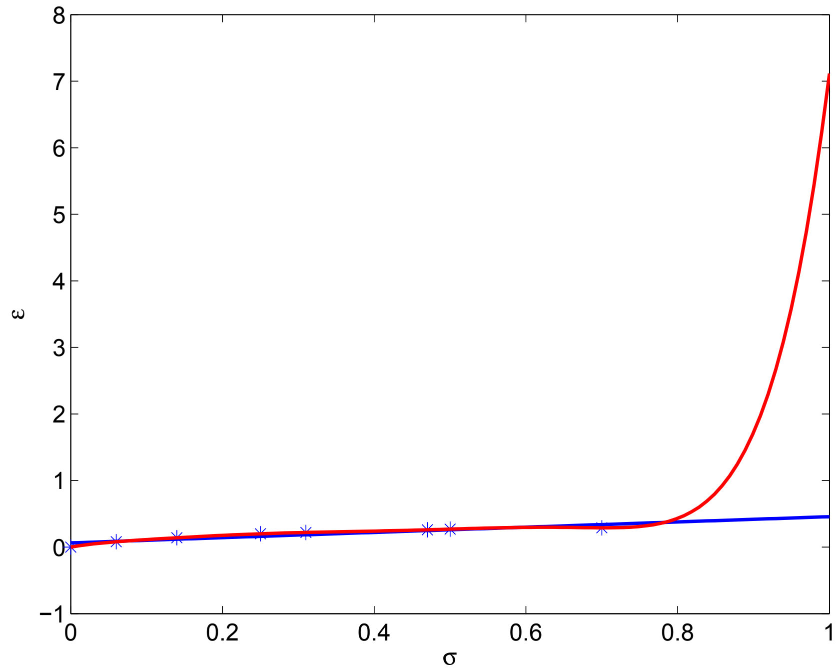
We want to extrapolate the value of ϵ for$\sigma = 0.4$. We consider two ways:

- compute the interpolating polynomial Π_7 of degree 7
- compute the least squares polynomial approximation of degree 1

On peut utiliser les commandes suivantes:

```
>>  plot(sigma, epsilon, '*'); hold on;  % plotting the known values
>>  sigma_sample = linspace(0,1.0,100);
>>  p7 = polyfit(sigma, epsilon, 7);
>>  pol = polyval(p7, sigma_sample);     % interpolating polynomial
>>  plot(sigma_sample,pol, 'r');
>>  p1 = polyfit(sigma,epsilon,1);
>>  pol_mc = polyval(p1, sigma_sample);  % least square
>>  plot(sigma_sample, pol_mc, 'g'); hold off;
```

- the interpolating polynomial Π_7 of degree 7 is in red unstable
- the least squares polynomial approximation of degree 1 is in blue stable



For $\sigma > 0.7$,the behavior of the two polynomials are very different.

In particular, for $\sigma = 0.9$ the values of $\epsilon(\sigma)$ extrapolated with the two methods are

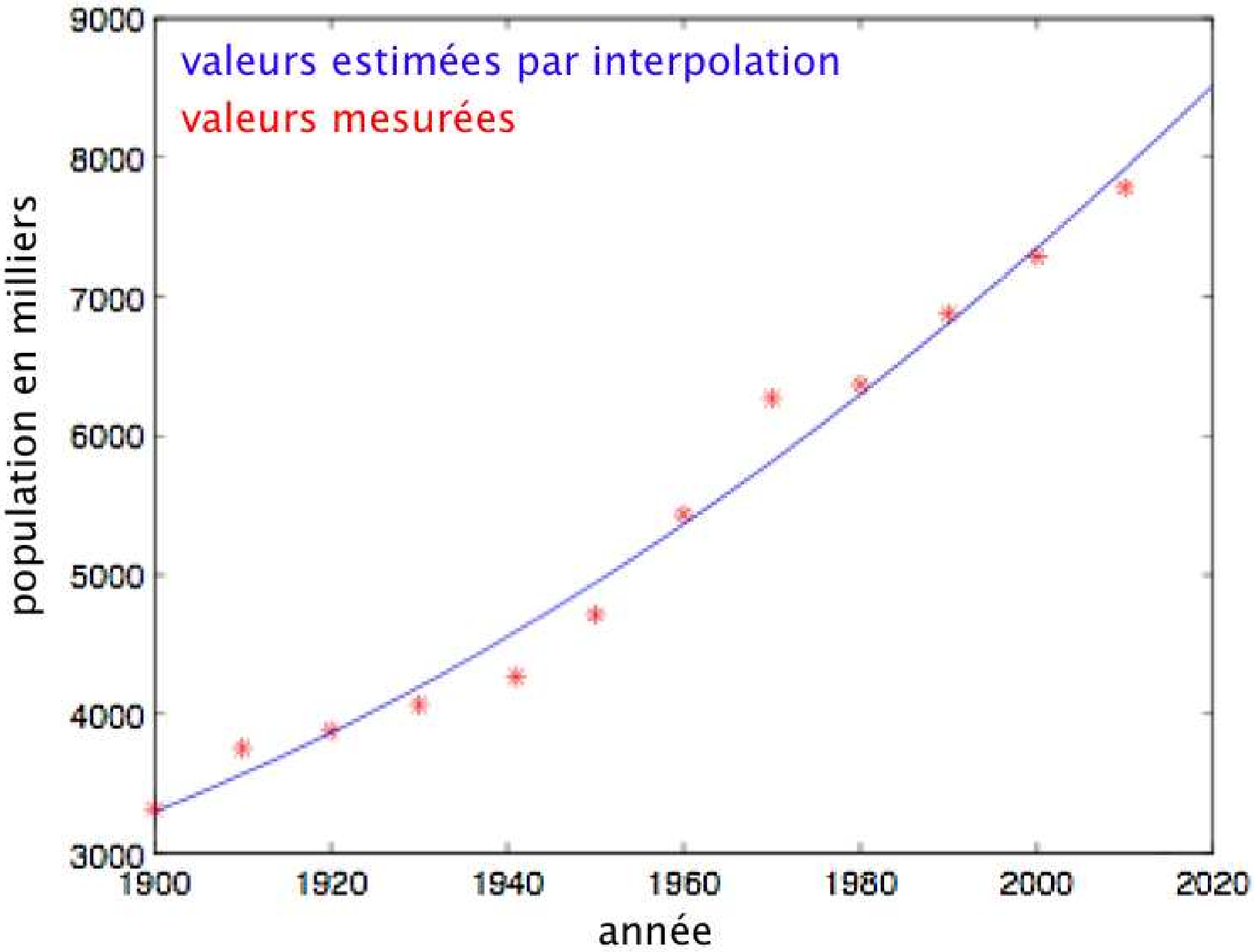
```
>> polyval(p7, 0.9)
ans =
   1.7221

>> polyval(p1, 0.9)
ans =
    0.4173
```

- The value obtained by Π_7 (172.21%) is unrealistic do not do extrapolation with p
- On the contrary, the value obtained with the regression line is more appropriate to compute the value at $\sigma = 0.9$.

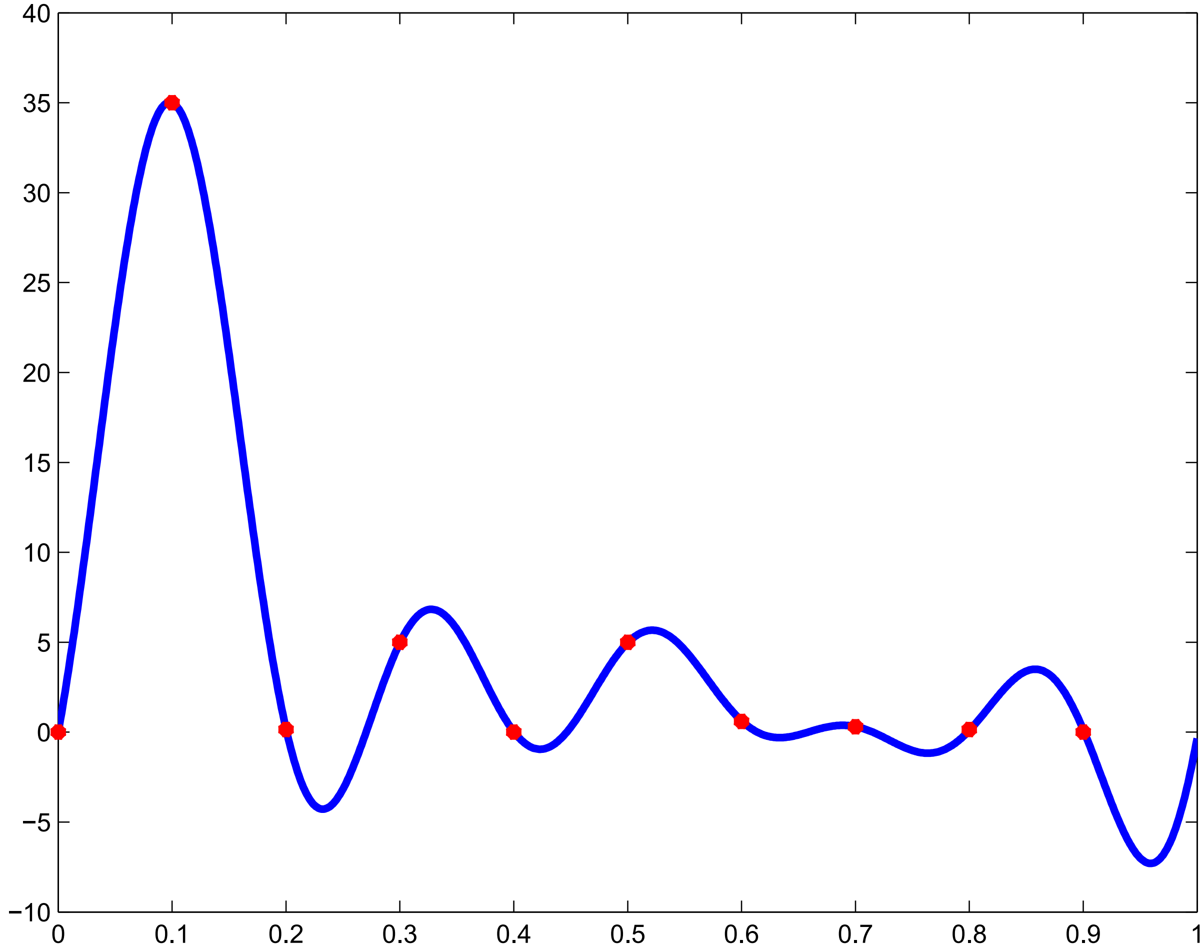
Example 10. Swiss population Starting from the population at the 20th century decades, we extrapolate the Swiss populmulation this year. We use a least square polynomial approximation of degree 2

```
>> year = [1900, 1910, 1920, 1930, 1941, 1950,...
           1960, 1970, 1980, 1990, 2000];
>> population = [3315, 3753, 3880, 4066, 4266, 4715,...
           5429, 6270, 6366, 6874, 7288];
>> p2 = polyfit(year, population, 2);
>> year_sample = [1900:2:2010];
>> vp2 = polyval(p2, year_sample);
>> plot(year, population, '*r', ...
       year_sample, vp2, 'b');
```


valeurs estimées par interpolation
valeurs mesurées
population en milliers
année
9000
8000
7000
6000
5000
4000
3000
1900
1920
1940
1960
1980
2000
2020

Example. Carotide flow rate We define 10 couples of data (time vs flowrate) with the vectors `t` and `deb`. With the help of the command `interpft`, we compute the value of the interpolating function at 1000 equidistributed points in the periodicity interval [0,1]:

```
t = [0 .1 .2 .3 .4 .5 .6 .7 .8 .9];
deb = [0 35 .15 5 0 5 .6 .3 .15 0];
f = interpft(deb, 1000);
plot(linspace(0, 1, 1000), f); hold on;
plot(t, deb, 'r*')
```


40
35
30
25
20
15
10
5
0
−5
−10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Processing of polynomials

In Matlab/Octave, there are specific commands for doing calculations with polynomials. Let $\mathtt{x}$ be a vector of abscissas, $\mathtt{y}$ be a vector of ordinates and $\mathtt{p}$ (respectively $\mathtt{p}_i$) be the vector of coefficients of a polynomial $P(x)$ (respectively P_i); then, we have following commands:

command	action
`y=polyval(p,x)`	$\mathtt{y}$ = values of $P(x)$
`p=polyfit(x,y,n)`	$\mathtt{p}$ = coefficients of the interpolating polynomial Π_n
`z=roots(p)`	$\mathtt{z}$ = zeros of P such that $P(\mathtt{z}) = 0$
p=conv($\mathtt{p}_1$,$\mathtt{p}_2$)	$\mathtt{p}$ = coefficients of the polynomial $P_1 P_2$
[q,r]=deconv($\mathtt{p}_1$,$\mathtt{p}_2$)	$\mathtt{q}$ = coefficients of Q, $\mathtt{r}$ = coefficients of R such that $P_1 = QP_2 + R$
`y=polyderiv(p)`	$\mathtt{y}$ = coefficients of $P'(x)$
`y=polyinteg(p)`	$\mathtt{y}$ = coefficients of $\int P(x)\,dx$