Numerical Simulations Applied to Engineering: 1 - Interpolation

Jordi José & Yuri Cavecchi
(Original Slides by Domingo García-Senz)

Departament Física, UPC

2024

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Plan of the Course

To understand the basic numerical foundations of several computational algorithms of interest in science and engineering. We will try to decipher how these algorithms work by building our own computational programs. We will apply these ideas to several problems in physics and engineering.

- **Part I. Basic concepts of numerical analysis**:

  • Interpolation techniques: Lagrange interpolation versus cubic spline interpolation. Examples and proposed classroom problems.

  • Geometry 1. Description of the position and orientation of reference frames and solid bodies. The homogeneous transformation matrix. Examples and proposed classroom problems.

  • Geometry 2. Some concepts of fractal theory and its potential applications to engineering.

  • Solving linear systems of equations. Inverting a matrix. Numerical foundations and comparative analysis of different analytical, semi-analytical and numerical methods. Examples and proposed classroom problems.

  • Numerical differentiation. Numerical solution of first and second ordinary differential equations (ODE) using basic schemes. Examples and proposed classroom problems.

Plan of the Course

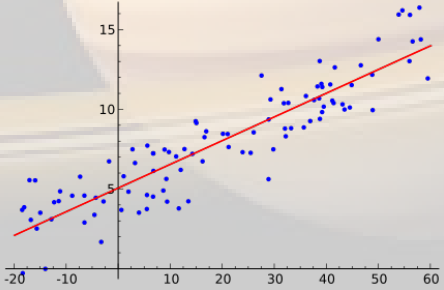
- **Part II. Applications**
  - Fourier Transforms and timing analysis.
  - Numerical solution of the Laplace equation: (I) Solving the Laplace equation to find the steady electrostatic potential within four electrodes. (II) Numerical approach to the temperature distribution in a room in steady conditions.
  - Partial Differential Equations (PDE). Numerical solution of the heat transport equation in 1D.
  - Partial Differential Equations (PDE). Conservation laws and the numerical solution of easy problems in fluid mechanics.

Interpolation

We should be aware of the main features of several related concepts: **Fitting, Interpolation and Extrapolation**.

Interpolation: Fitting

- **Fitting**: "Given a discrete set of points (sparse or not) find a simple meaningful function passing the closest to all of them"
- The standard approach to solve this problem is the **least squares** (LS) technique.
- The easiest case of LS is the linear regression fitting:



Interpolation: Fitting: Least Squares

- You have a function f that *you think* models the data, it depends on some data x and a set of parameters $\vec{p}$ (Note that y and x could be vectors).

$$y = f(x|\vec{p}) \tag{1}$$

- Then you look for the set $\vec{p}$ that **minimizes** the sum of the residuals between prediction and data

$$r_i = y_i - f(x_i|\vec{p}) \tag{2}$$

$$R(\vec{p}) = \sum_i r_i^2 = \sum_i |y_i - f(x_i|\vec{p})|^2 \tag{3}$$

Find the minimum of $R(\vec{p})$

Interpolation: Fitting: Least Squares

- The underlaying assumption is that the difference between theory and practice comes only from measument errors.

$$y_i = f(x_i|\vec{p}) + \epsilon_i \tag{4}$$

- The extrema of (3) are found by setting its gradient to 0:

$$\partial_{p_j} R(\vec{p}) = 0 \tag{5}$$

$$\partial_{p_j} R(\vec{p}) = 2 \sum_i r_i \cdot \partial_{p_j} r_i \tag{6}$$

$$\partial_{p_j} R(\vec{p}) = -2 \sum_i (y_i - f(x_i|\vec{p})) \cdot \partial_{p_j} f(x_i|\vec{p}) \tag{7}$$

Interpolation: Fitting: Least Squares

- If $f(x_i|\vec{p})$ is *linear* in p_j then $\partial_{p_j} f(x_i|\vec{p})$ is independent of p_j and (7) boils down to a linear system of equations: the p_js are then uniquely determined.
- If $f(x_i|\vec{p})$ is *not linear* in p_j then you are in trouble. But various methods exist to minimize a nonlinear function.
- One common solution is to *linearize* f and iterate. Write:

$$f(x_i|\vec{p}) \simeq f(x_i|\vec{p^0}) + \partial_{p_j} f(x_i|\vec{p})|_{\vec{p^0}} \cdot \Delta\vec{p} \tag{8}$$

Guess an initial value $\vec{p^0}$, solve for $\Delta\vec{p}$ (this is now linear), update your guess to $\vec{p^1} = \vec{p^0} + \Delta\vec{p}$ and repeat until *convergence*.

Interpolation: Fitting: Least Squares

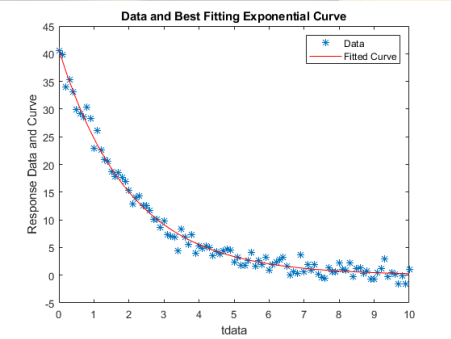
- You can also *weight* your residuals to give more "importance" to some data. A typical case is dividing by the error/variance of each measurement:

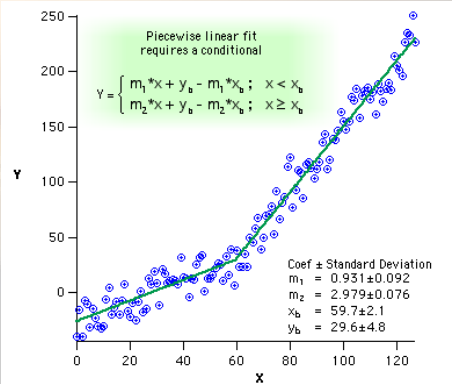
$$r_i = [y_i - f(x_i|\vec{p})]\, w_i \tag{9}$$

The rest is basically the same.

Interpolation: Fitting: Least Squares

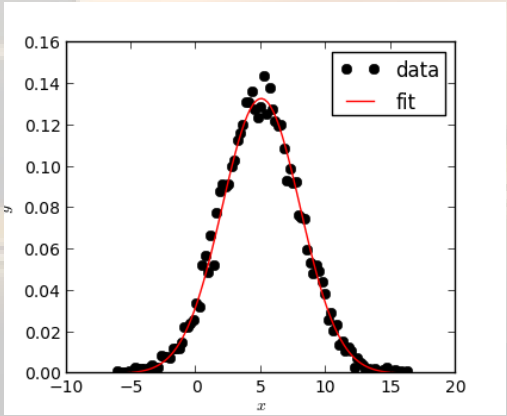
- Linear regression is a powerful technique. Through mathematical manipulation it can be adapted to fit non-linear functions
- For example $y = A\exp(bx)$ is changed to $\ln y = bx + \ln A$ which is linear (left).
- Multiple regressions in different regions of the sample are also allowed (right).





Interpolation: Fitting: Least Squares

- Other distributions can be optimally fitted using the LS technique. Here you have an example of Gaussian fitting.

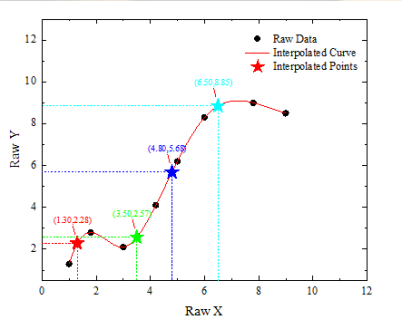


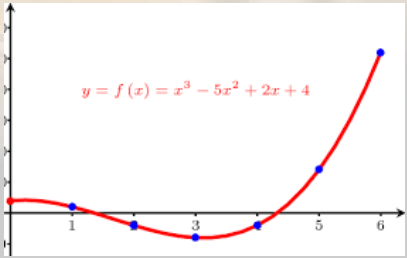
Interpolation: Fitting

- To make an optimal fit it is advisable to have a previous knowledge of the functional form of the function.
- It is therefore important to **understand the physical features** of the system we want to fit in order to select a suitable fitting function $f(\ldots)$.
- It is also of capital importance to define a second statistical parameter giving an idea of the quality of the fitting. A **parameter of confidence** linked to the variance of the data dispersion around $f(\ldots)$ is needed.

Interpolation

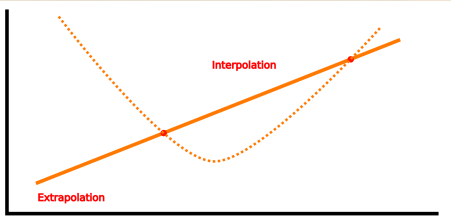
- **Interpolation** In 1D, given a set of points in the interval $[a, b]$ find a suitable function $f(x)$ which passes through all the data points. Once $f(x)$ is known we are able to confidently make an estimate of the value of *true* $f(x)$ at any point belonging to $[a, b]$ (it is therefore a particular case of fitting).
- **Polynomial interpolation** is the most used in practical applications.

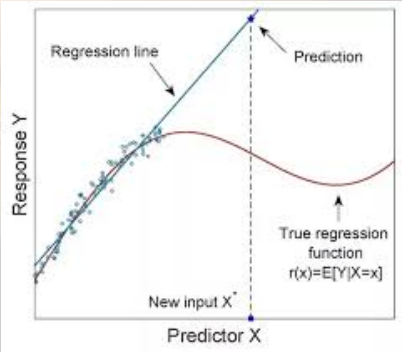




Interpolation: Extrapolation

- **Extrapolation**: Makes a guess of the value of the *estimated* function $f(x)$ beyond the interval $[a, b]$.
- Be aware that extrapolation is a risky technique, especially if the function does not behave smoothly and/or the point is far from the sample interval $[a, b]$. A good knowledge of the physical system is required to trust the extrapolated result.





Interpolation: Lagrange

- **Lagrange Interpolation**
- Our target, $f(x)$, is just a polynomial of degree $(n-1)$, passing through each point of the discrete set of data:

$$\{x_1, x_2, ...x_n\} \longrightarrow \{f(x_1), f(x_2), ..., f(x_n)\}$$

Let us take the easiest case with only two points. To find $f(x)$, first make a Taylor expansion:

$$f(x_1) = f(x) + (x_1 - x)f^{'}(x) + ... + ...$$
$$f(x_2) = f(x) + (x_2 - x)f^{'}(x) + ... + ...$$

Now we introduce the function $p(x)$, which is the Lagrange-polynomial we are looking for:

$$f(x_1) = p(x) + (x_1 - x)p^{'}(x) + ... + ...$$
$$f(x_2) = p(x) + (x_2 - x)p^{'}(x) + ... + ...$$

Interpolation: Lagrange

- Which is a set of two equations and two unknowns ($p(x), p^{'}(x)$). The solution is:

$$p(x) = \frac{(x - x_2)}{(x_1 - x_2)} f(x_1) + \frac{(x - x_1)}{(x_2 - x_1)} f(x_2) \tag{10}$$

  $p(x)$ is a linear function, as expected.

- We can repeat the same reasoning for three points, n=3.

$$f(x_1) = p(x) + (x_1 - x)p^{'}(x) + \frac{1}{2}(x_1 - x)^2 p^{''} + ... + ...$$

$$f(x_2) = p(x) + (x_2 - x)p^{'}(x) + \frac{1}{2}(x_2 - x)^2 P^{''} + ... + ...$$

$$f(x_3) = p(x) + (x_3 - x)p^{'}(x) + \frac{1}{2}(x_3 - x)^2 p^{''} + ... + ...$$

Interpolation: Lagrange

- Which is a set of three equations and three unknowns $[p(x), p^{'}(x), p^{''}(x)]$. The solution is:

$$p(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} f(x_1) + \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)((x_2 - x_3)} f(x_2) + \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)} f(x_3) \tag{11}$$

  which is the equation of a parabola. Therefore we refer to quadratic interpolation, as expected.
- As it can be seen, there is a clear trend in the interpolating formula so that it can be generalized to work with any number of data points, n, which define a polynomial $p(x)$ of degree $(n - 1)$.

Interpolation: Lagrange

$$p(x) = \sum_{j=1}^{n} l_{j,n} f(x_j) \tag{12}$$

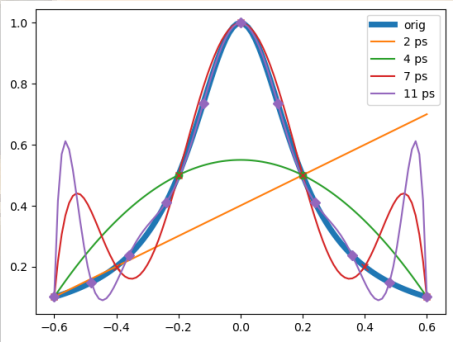
where:

$$l_{j,n} = \frac{(x - x_1)(x - x_2)...(x - x_{j-1})(x - x_{j+1}).....(x - x_n)}{(x_j - x_1)(x_j - x_2)...(x_j - x_{j-1})(x_j - x_{j+1}).....(x_j - x_n)} = \Pi_{i=1, i\neq j}^{n} \frac{x - x_i}{x_j - x_i} \tag{13}$$

- One problem with this kind of interpolation is that for a large collection of points the degree of the polynomial is excessively large to be practical.
- Another problem is that there is a weak control to the fitting of first and second derivatives at the limits of the data set. The correct handling of the first and second derivatives is important to simulate many physical and engineering problems.

Interpolation: Lagrange

- Especially for equispaced data, you would see large oscillations near the edges (Runge phenomenon - avoidable choosing better control points, but it can be cumbersome).



Interpolation: Lagrange

- **Example**: The following Table has been built according to the function $y = \sin x$ in the interval $[0, \pi/2]$.

x	0	$\frac{\pi}{3}$	$\frac{\pi}{2}$
f(x)	0	$\frac{\sqrt{3}}{2}$	1

- We directly apply the expression (11) to the values of the Table:

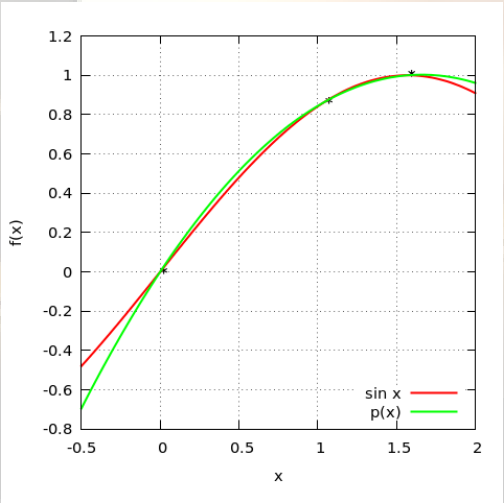
$$p(x) = \frac{(x-\frac{\pi}{3})(x-\frac{\pi}{2})}{(0-\frac{\pi}{3})(0-\frac{\pi}{2})}\mathbf{0} + \frac{(x-0)(x-\frac{\pi}{2})}{(\frac{\pi}{3}-0))(\frac{\pi}{3}-\frac{\pi}{2})}\frac{\sqrt{\mathbf{3}}}{\mathbf{2}} + \frac{(x-0)(x-\frac{\pi}{3})}{(\frac{\pi}{2}-0))(\frac{\pi}{2}-\frac{\pi}{3})}\mathbf{1} \tag{14}$$

$$p(x) = -\left(x^2 - \frac{\pi}{2}x\right)1.57944 + \left(x^2 - \frac{\pi}{3}x\right)1.21585 \tag{15}$$

$$p(x) = -0.36359x^2 + 1.2077x \tag{16}$$

Interpolation: Lagrange

- We can plot $f(x) = \sin x$ versus $p(x) = -(x^2 - \frac{\pi}{2}x)1.57944 + (x^2 - \frac{\pi}{3}x)1.21585$



Interpolation: Lagrange

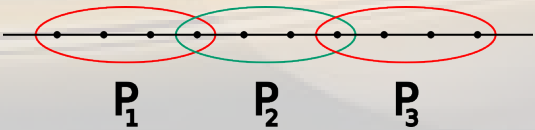
- The fit is pretty good in the interval $[0, \pi/2]$.
- Beyond that interval (extrapolation) the error grows. For $x = -0.5$ the error is:

$$\varepsilon(x_0 = -0.5) = \left| \frac{f(x_0) - p(x_0)}{f(x_0)} \right| = 0.449 (\simeq 45\%!!)$$

- Note also that the derivative is not estimated so well. For example, at $x_0 = 0 \longrightarrow f^{'} = \cos(x_0) = 1; p^{'} = 1.2077$, the error being around 20%.
- Large errors in the derivatives at the interpolation points can be the source of problems in engineering, because the system does not behave smoothly enough.
- The **cubic-spline** interpolation leads to a smoother behavior of the representation of physical systems. The first derivative is, generally, much better reproduced at the connecting points.

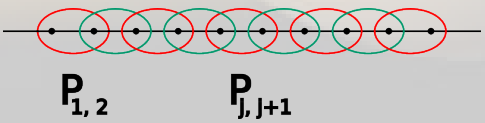
Interpolation: a family of lower order polynomials

- For a large set of points, n, the Lagrange interpolation may lead to a wild polynomial because the polynomial degree, $(n-1)$, becomes **too large**.
- A possible solution is to split the sample in small groups of four points finding a family of third-order polynomials covering the whole sample.



Interpolation: cubic-splines

- The problem is how to connect the different $p(x)$s smoothly.
- The cubic-spline method connects polynomials of order 3 smoothly, defining these polynomials over intervals of **only two** points.
- In the following, we will find the numerical algorithm to perform cubic-spline interpolations. Understanding the mains steps of the proof will help you better grasp the technique. I will highlight the more important steps while skipping some other, more tedious, points.



Interpolation: cubic-splines

- Let us find the polynomial of order 3 connecting points j to $j+1$: $p_{j,j+1}$.

$$p_{j,j+1}(x) = a_j(x - x_j)^3 + b_j(x - x_j)^2 + c_j(x - x_j) + d_j \tag{17}$$

thus, at $x = x_j$:

$$p(x_j) = f(x_j) = d_j \tag{18}$$

and at $x = x_{j+1}$:

$$p_{j,j+1}(x_{j+1}) = f(x_{j+1}) = a_j(x_{j+1} - x_j)^3 + b_j(x_{j+1} - x_j)^2 + c_j(x_{j+1} - x_j) + d_j \tag{19}$$

Using expression (18) and calling: $f_j = f(x_j)$, $h_j = x_{j+1} - x_j$ we obtain:

$$f_{j+1} = a_j h_j^3 + b_j h_j^2 + c_j h_j + f_j \tag{20}$$

Equation (20) has still three unknowns: a_j, b_j, c_j. Two more equations are needed to find $p_{j,j+1}(x)$. These equations come from the derivatives of $p_{j,j+1}(x)$.

Interpolation: cubic-splines

- So far, we can solve for c_j:

$$c_j = \frac{f_{j+1} - f_j}{h_j} - b_j h_j - a_j h_j^2 \tag{21}$$

- The main trick is **we impose continuity up to the second derivative** at the common, internal points x_j ($2 \leq j \leq n-1$):

$$p'_{j-1,j}(x_j) = p'_{j,j+1}(x_j) = p'_j \tag{22}$$

$$p''_{j-1,j}(x_j) = p''_{j,j+1}(x_j) = p''_j \tag{23}$$

Interpolation: cubic-splines

- Consider the first and second derivatives of expression (17):

$$p'_{j,j+1}(x) = 3a_j(x - x_j)^2 + 2b_j(x - x_j) + c_j \tag{24}$$

$$p''_{j,j+1}(x) = 6a_j(x - x_j) + 2b_j \tag{25}$$

Evaluating the second derivative at $x = x_j$ and $x = x_{j+1}$ leads to:

$$b_j = \frac{p''_j}{2} \tag{26}$$

$$a_j = \frac{1}{6}\frac{p''_{j+1} - p''_j}{h_j} \tag{27}$$

Interpolation: cubic-splines

- The coefficient c_j is obtained from expression (21):

$$c_j = \frac{f_{j+1} - f_j}{h_j} - \frac{h_j p_{j+1}^{''} + 2h_j p_j^{''}}{6} \tag{28}$$

Putting all together in Eq.(17):

$$p_{j,j+1}(x) = f_j + \left[\frac{f_{j+1} - f_j}{h_j} - \frac{h_j p_{j+1}^{''}}{6} - \frac{h_j p_j^{''}}{3} \right] (x - x_j) + \frac{p_j^{''}}{2}(x - x_j)^2 + \frac{p_{j+1}^{''} - p_j^{''}}{6h_j}(x - x_j)^3 \tag{29}$$

which is valid in the interval $x_j \leq x \leq x_{j+1}$. The first derivative of (29) is:

$$p_{j,j+1}^{'}(x) = \left[\frac{f_{j+1} - f_j}{h_j} - \frac{h_j p_{j+1}^{''}}{6} - \frac{h_j p_j^{''}}{3} \right] + p_j^{''}(x - x_j) + \frac{p_{j+1}^{''} - p_j^{''}}{2h_j}(x - x_j)^2 \tag{30}$$

Interpolation: cubic-splines

- And now it comes the **more relevant** step in the demonstration. To obtain $p_j^{''}$ and $p_{j+1}^{''}$ at every point along the sample, we connect the current interval $(j, j+1)$ with the previous one $(j-1, j)$:

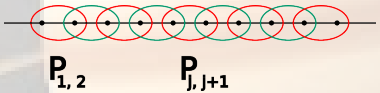
$$p_{j-1,j}^{'}(x_j) = p_{j,j+1}^{'}(x_j) \tag{31}$$

The first derivative of $p_{j-1,j}$ is similar to (30):

$$p_{j-1,j}^{'}(x) = \left[\frac{f_j - f_{j-1}}{h_{j-1}} - \frac{h_{j-1}p_j^{''}}{6} - \frac{h_{j-1}p_{j-1}^{''}}{3}\right] + p_{j-1}^{''}(x - x_{j-1}) + \frac{p_j^{''} - p_{j-1}^{''}}{2h_{j-1}}(x - x_{j-1})^2 \tag{32}$$

which is valid in the interval $x_{j-1} \leq x \leq x_j$.

Interpolation: cubic-splines



$$p'_{j-1,j}(x_j) = p'_{j,j+1}(x_j) \tag{33}$$

leads to the following equations ($j = 2, \ldots\ldots n-1$):

$$h_{j-1}p''_{j-1} + 2(h_{j-1} + h_j)p''_j + h_j p''_{j+1} = 6\left(\frac{f_{j+1} - f_j}{h_j} - \frac{f_j - f_{j-1}}{h_{j-1}}\right) \tag{34}$$

Interpolation: cubic-splines

- Which is a linear system with $(n-2)$ equations with n unknowns (the second derivatives $p_j^{''}$)
- Therefore we have to add two extra informations to solve the system. A very common simplification is to consider:

$$p_1^{''} = p_n^{''} = 0$$

which simply means that we suppose that the function is behaving linearly at the limits of the data set. In this case we talk of **natural splines**.

- Using natural splines we are able to find the second derivatives of the family of cubic polynomials. The problem can be expressed in matrix form.

Interpolation: cubic-splines

$$\begin{bmatrix} 2(h_1+h_2) & h_2 & & & \\ h_2 & 2(h_2+h_3) & h_3 & & \\ & \cdots\cdots & & & \\ & \cdots\cdots\cdots & & & \\ & \cdots & h_{n-2} & 2(h_{n-2}+h_{n-1}) & \end{bmatrix} \cdot \begin{bmatrix} p_2'' \\ p_3'' \\ . \\ . \\ p_{n-1}'' \end{bmatrix} = \begin{bmatrix} \frac{6(f_3-f_2)}{h_2} - \frac{6(f_2-f_1)}{h_1} \\ \frac{6(f_4-f_3)}{h_3} - \frac{6(f_3-f_2)}{h_2} \\ . \\ . \\ \frac{6(f_n-f_{n-1})}{h_{n-1}} - \frac{6(f_{n-1}-f_{n-2})}{h_{n-2}} \end{bmatrix} \tag{35}$$

Which is a system of $n-2$ equations with $n-2$ unknowns. It has been assumed that $p_1'' = p_n'' = 0$ (**natural cubic splines**).

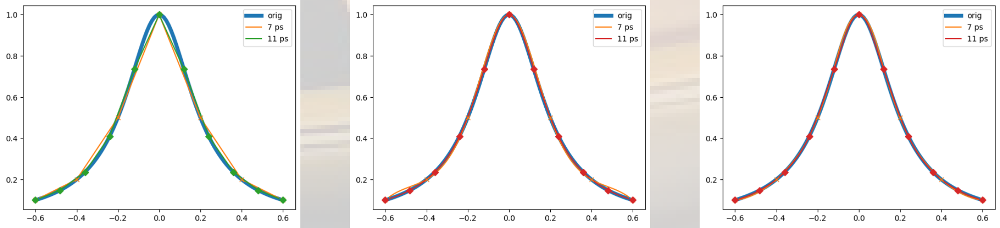
After finding the second derivatives we have to put them in Eqs. (29, 30) to obtain the spline functions, which work in each of the intervals $x_j \leq x \leq x_{j+1}$.

Interpolation: cubic-splines

- A system of equations such as (35) is called three-diagonal. There exist especial techniques to efficiently solve it.
- Algorithmic efficiency is very important whenever we interpolate in real life, because of the run-time of the calculations.
- Remember that cubic spline interpolation is not just one single cubic-spline but a family. The first and second derivatives are kept continuous at the common points so that the connection among each member is smooth, while keeping good accuracy at each point. This is the main advantage against Lagrange interpolation using cubic polynomials.
- But the cubic spline interpolation is more expensive in computational terms than the Lagrangian interpolation.

Interpolation: cubic-splines

- Back to Runge phenomenon - the spline version (order = 1, 3, 4)



Interpolation: cubic-splines

- **Example**: Find the two natural cubic-spline polynomials which fit the following discrete function.

x	1	2	4
f(x)	2	5	17

- We will find two polynomials: $p_{1,2}$ defined in the interval $1 \leq x \leq 2$ and $p_{2,3}$ in the interval $2 \leq x \leq 4$. Both polynomials share $x = 2$ as a common point.
- Firs of all we calculate $\mathbf{h_1 = x_2 - x_1 = 2 - 1 = 1}$ as well as $\mathbf{h_2 = x_3 - x_2 = 4 - 2 = 2}$
- We go to the matrix expression (35). In this simple problem there is only one linear equation:

$$2(h_1 + h_2)p_2'' = 6\frac{f_3 - f_2}{h_2} - 6\frac{f_2 - f_1}{h_1} \longrightarrow 6p_2'' = 18 \tag{36}$$

- Therefore $p_2'' = 3$ (and $p_1'' = p_3'' = 0$, because we are dealing with natural splines)

Interpolation: cubic-splines

- Now we make use of expression (29) to compute $p_{1,2}$ and $p_{2,3}$.

$$p_{1,2}(x) = f_1 + \left[\frac{f_2 - f_1}{h_1} - \frac{h_1 p_2^{''}}{6} - \frac{h_1 p_1^{''}}{3}\right](x - x_1) + \frac{p_1^{''}}{2}(x - x_1)^2 + \frac{p_2^{''} - p_1^{''}}{6h_1}(x - x_1)^3 \quad (37)$$

valid in the interval $(1 \leq x \leq 2)$.

$$p_{2,3}(x) = f_2 + \left[\frac{f_3 - f_2}{h_2} - \frac{h_2 p_3^{''}}{6} - \frac{h_2 p_2^{''}}{3}\right](x - x_2) + \frac{p_2^{''}}{2}(x - x_2)^2 + \frac{p_3^{''} - p_2^{''}}{6h_2}(x - x_2)^3 \quad (38)$$

valid in the interval $(2 \leq x \leq 4)$

Interpolation: cubic-splines

- Using the values:

$$x_1 = 1, x_2 = 2, x_3 = 4; f_1 = 2, f_2 = 5, f_3 = 17; h_1 = 1, h_2 = 2; p_1'' = 0, p_2'' = 3, p_3'' = 0$$

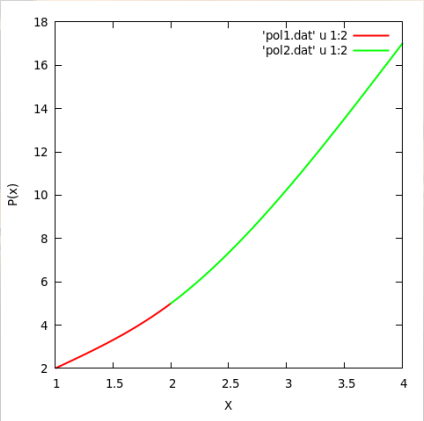
we get:

$$p_{1,2}(x) = \frac{1}{2}(x-1)^3 + \frac{5}{2}(x-1) + 2 \qquad 1 \leq x \leq 2 \qquad (39)$$

$$p_{2,4}(x) = -\frac{1}{4}(x-2)^3 + \frac{3}{2}(x-2)^2 + 4(x-2) + 5 \qquad 2 \leq x \leq 4 \qquad (40)$$

Interpolation: cubic-splines

- The figure shows that the two splines are passing over the sample of points. The connection is at $x = 2$.
- Check that the first derivative at the common point $x = 2$ is the same.



Interpolation: Classroom Problem 1.1

The position, x, of an object which comes into an aerodynamic tunnel has been monitored as a function of the elapsed time t. The initial velocity is v_0 and the air opposes a resistance force just proportional to the velocity of the object, $F_R = -bv$. The following table is written.

t (s)	x (km)
0	0
1	0.63
2	0.86
3	0.95

(41)

Fit $x(t)$ in the interval $0 \leq t \leq 3$ s by using three cubic spline polynomials and give a graph of the resulting function. Check the quality of the fitting by comparing the result with the analytical solution:

$$x(t) = \frac{mv_0}{b}\left(1 - e^{-\frac{b}{m}t}\right) \tag{42}$$

which gives the distance covered by a body with initial velocity v_0 under a resistant force $F_R = -bv$. Take $m = b = v_0 = 1$ to solve this problem. Analyze the behavior of the velocity.