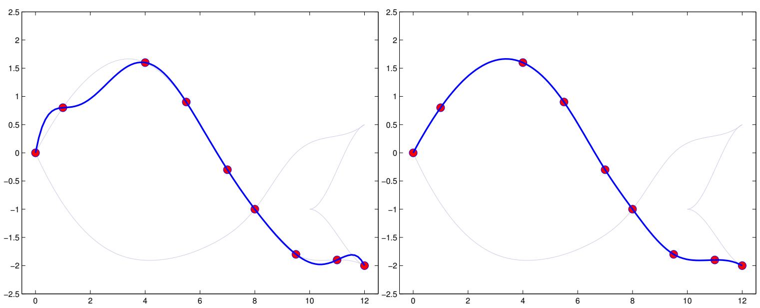
5 Cubic Splines

5.1 Introduction

Assume that we have a set of $n+1$ points $\{x_i, y_i\}_{i=0}^n$ and we want to find a curve interpolationg these points. One possibility is of course to use polynomial interplation, that is, find a polynomial $p_n \in \mathbb{P}_n$ so that

$$p_n(x_i) = y_i, \qquad i = 0, 1, \ldots, n.$$

This may be quite unsatisfactory, as the following picture demonstrate:



In the picture to the left, polynomial interpolation have been used, to the right, cubic splines. The idea of splines is to split the interval $[a, b]$ by $a = t_0 < t_1 < \cdots < t_n = b$, and let interpolating curve be a polynomial on each subinterval $[t_{i-1}, t_i)$. The points t_i, $i = 0, 1, \ldots, n$ are called *knots* (skjøter på norsk), and they may or may not correspond to the interpolation nodes x_i. The piecewise polynomials are then glued together by some smoothness conditions. More formally, the definition is:

Definition 5.1. *On some interval* $[a, b]$ *, suppose that* $n+1$ *points* $a = t_0 < t_1 < \cdots < t_n = b$ *has been specified. A spline of degree* k *is a function* S *satisfying*

1. On each interval $[t_{i-1}, t_i)$, S is a polynomial of degree k.
2. $S \in C^{(k-1)}[a, b]$.

We will write the spline by

$$S(x) = \begin{cases} S_0(x) & x \in [t_0, t_1) \\ S_1(x) & x \in [t_1, t_2) \\ \vdots & \\ S_{n-1}(x) & x \in [t_{n-1}, t_n] \end{cases} \tag{18}$$

where $S_i \in \mathbb{P}_k$.

Example 5.2. *The linear spline interpolating the the points* $\{t_i, y_i\}_{i=0}^n$ *is given by*

$$S_i(x) = y_i \frac{x - t_{i+1}}{t_i - t_{i+1}} + y_{i+1} \frac{x - t_i}{t_{i+1} - t_i}, \qquad x \in [t_i, t_{i+1}) \qquad i = 0, 1, \ldots, n-1, \tag{19}$$

the straight lines between the points.

5.2 Cubic splines

We will now construct an algorithm for finding the cubic splines, interpolating the points $\{t_i, y_i\}_{i=0}^n$. It means that

$$S_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i, \qquad x \in [t_i, t_{i+1}) \qquad i = 0, 1, \dots, n-1$$

which gives a total of $4n$ parameters to be determined. A cubic spline is two times continuous differentiable, thus it has has to satisfy

$$S_i(t_i) = y_i, \qquad S_i(t_{i+1}) = y_{i+1}, \qquad i = 0, \cdots, n-1 \tag{20}$$
$$S'_{i-1}(t_i) = S'_i(t_i), \qquad i = 1, 2, \dots, n-1 \tag{21}$$
$$S''_{i-1}(t_i) = S''_i(t_i), \qquad i = 1, 2, \dots, n-1 \tag{22}$$

a total of $4n - 2$ conditions, leaving two free parameters. Some common choices for those are

- Natural cubic splines: $S''(t_0) = S''(t_n) = 0$.
- Clamped cubic splines: $S'(t_0)$ and $S'(t_n)$ are specified.
- Not-a-knot condition: $S'''_0(t_1) = S'''_1(t_1)$ and $S'''_{n-2}(t_{n-1}) = S'''_{n-1}(t_n)$.
- Periodic conditions: $S'_0(t_0) = S'_{n-1}(t_n)$ and $S''_0(t_0) = S''_{n-1}(t_n)$.

We will now construct an efficient algorithm for solving finding the splines. The idea is as follows: Since S is a cubic spline, S'' is a linear spline. Let $z_i = S''(t_i)$, $i = 0, 1, \dots, n$ (to be found). Further, let $h_i = t_{i+1} - t_i$. Then, from 19 we have that

$$S''_i(x) = \frac{z_i}{h_i}(t_{i+1} - x) + \frac{z_{i+1}}{h_i}(x - t_i).$$

So, by this, (22) is satisfied. Integrating twice gives

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1} - x)^3 + \frac{z_{i+1}}{6h_i}(x - t_i)^3 + C_i x + D_i.$$

The integration constants C_i and D_i can be determined by (20), the result becomes

$$S_i(x) = \frac{z_i}{6h_i}(t_{i+1}-x)^3 + \frac{z_{i+1}}{6h_i}(x-t_i)^3 + \left(\frac{y_{i+1}}{h_i} - \frac{z_{i+1}h_i}{6}\right)(x-t_i) + \left(\frac{y_i}{h_i} - \frac{z_i h_i}{6}\right)(t_{i+1}-x), \tag{23}$$

We now activate the second condition (21). Notice that

$$S'_i(t_i) = -\frac{h_i}{3} z_i - \frac{h_i}{6} z_{i+1} - \frac{y_i}{h_i} + \frac{y_{i+1}}{h_i}$$

and

$$S''_{i-1}(t_i) = \frac{h_i}{6} z_{i-1} + \frac{h_{i-1}}{3} z_i - \frac{y_{i-1}}{h_{i-1}} + \frac{y_i}{h_{i-1}}$$

so these conditions will simply become

$$h_{i-1} z_{i-1} + 2(h_i + h_{i-1}) z_i + h_i z_{i+1} = \frac{6}{h_i}(y_{i+1} - y_i) - \frac{6}{h_{i-1}}(y_i - y_{i-1}), \qquad i = 1, \cdots, n-1.$$

Let us now assume $z_0 = z_n = 0$, the natural spline condition. The whole system now becomes a tridiagonal system of equations:

$$\begin{pmatrix} u_1 & h_1 & & & & \\ h_1 & u_2 & h_2 & & & \\ & h_2 & u_3 & h_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & h_{n-3} & u_{n-2} & h_{n-2} \\ & & & & h_{n-2} & u_{n-1} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{n-2} \\ z_{n-1} \end{pmatrix} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-2} \\ v_{n-1} \end{pmatrix}$$

with

$$h_i = t_{i+1} - t_i, \qquad u_i = 2(h_i + h_{i-1}), \qquad b_i = \frac{6}{h_i}(y_{i+1} - y_i), \qquad v_i = b_i - b_{i-1}.$$

Notice that the matrix is diagonal dominant, so the system can be solved by some direct methods for tridiagonal systems. The complete algorithm becomes:

Input: n, $(t_i, y_i)_{i=0}^n$
  for $i = 0, 1, \dots, n-1$ **do** ▷ Set up the linear system
    $h_i \leftarrow t_{i+1} - t_i$
    $b_i \leftarrow 6(y_{i+1} - y_i)$
  end for

  $u_1 \leftarrow 2(h_0 + h_1)$ ▷ The LU-factorization
  $v_1 \leftarrow b_1 - b_0$
  for $i = 2, 3, \dots, n-1$ **do**
    $u_i \leftarrow 2(h_i + h_{i-1}) - h_{i-1}^2/u_{i-1}$
    $v_i \leftarrow b_i - b_{i-1} - h_{i-1}v_{i-1}/u_{i-1}$
  end for

  $z_n \leftarrow 0$ ▷ Back substitution
  for $i = n-1, n-2, \dots, 1$ **do**
    $z_i = (v_i - h_i z_{i+1})/u_i$
  end for
  $z_0 \leftarrow 0$.

For natural cubic splines, we do have the following result:

Theorem 5.3. *Let $f \in C^2[a,b]$. If S is the natural cubic spline interpolating f in the knots $a = t_0 < t_1 < \cdots < t_n = b$ then*

$$\int_a^b \left(S''(x)\right)^2 dx \le \int_a^b \left(f''(x)\right)^2 dx.$$

Proof. Let $g = f - S$. Then

$$\int_a^b \left(f''(x)\right)^2 dx = \int_a^b \left(S''(x)\right)^2 dx + \int_a^b \left(g''(x)\right)^2 dx + 2\int_a^b g''(x)S''(x)dx.$$

The statement of the theorem is clearly true if we can prove that the last term is positive. Notice that S_i''' is constant on each each interval $[t_i, t_{i+1})$, and let us call this constant a_i. By partial integration we get

$$\int_a^b S''g''dx = \sum_{i=0}^{n-1} \int_{t_i}^{t_{i+1}} S''g''dx = \sum_{i=0}^{n-1} \left\{ (S''(t_{i+1})g'(t_{i+1}) - S''(t_i)g'(t_i) - \int_{t_i}^{t_{i+1}} S'''g'dx \right\}$$

$$= S''(t_n)g'(t_n) - S''(t_0)g'(t_0) - \sum_{i=0}^{n-1} c_i \int_{t_i}^{t_{i+1}} g'dx = \sum_{i=0}^{n-1} c_i(g(t_{i+1}) - g(t_i)) = 0.$$

□

The *curvature* of a function f is defined as $|f''|/\left(\sqrt{1+(f')^2}\right)^3$. If we assume that $|f'| \ll 1$ we are left with f'' as a approximate measure for the curvature. In this sense, the natural cubic spline is the smoothest possible function interpolating the given data.