Applications of Polynomial Interpolation

- Extrapolation

- Splines

Contents

Problem Formulation

We have a function $f : \mathbb{R} \to \mathbb{R}$ that can be evaluated for all $h > 0$, but f cannot be evaluated easily at $h = 0$. We are interested in computing

$$\lim_{h \to 0} f(h) \ .$$

Example: We want to evaluate the expression

$$\lim_{h \to 0} \frac{\exp(h) - 1}{\sin(h)} \ .$$

Here, we cannot simply substitute $h = 0$, as we would have to divide by $\sin(0) = 0$.

Problem Formulation

We have a function $f : \mathbb{R} \to \mathbb{R}$ that can be evaluated for all $h > 0$, but f cannot be evaluated easily at $h = 0$. We are interested in computing

$$\lim_{h \to 0} f(h) \ .$$

Example: We want to evaluate the expression

$$\lim_{h \to 0} \frac{\exp(h) - 1}{\sin(h)} \ .$$

Here, we cannot simply substitute $h = 0$, as we would have to divide by $\sin(0) = 0$.

L'Hospital's rule

There are several solution strategies for computing limites whose applicability depends on the situation. For example L'Hospital's rule gives

$$\lim_{h \to 0} \frac{\exp(h) - 1}{\sin(h)} \;=\; \frac{\exp(0)}{\cos(0)} \;=\; 1\,.$$

Thus, if we use algorithmic differentiation, we may be able to implement L'Hospital's rule for expressions of the form

$$\lim_{h \to 0} \frac{f_1(h)}{f_2(h)} \;=\; \lim_{h \to 0} \frac{f_1'(h)}{f_2'(h)}\,,$$

assuming $f_1(0) = f_2(0) = 0$ and that the derivatives of f_1 and f_2 can be computed easily. However, in general L'Hospital's rule may not be applicable.

L'Hospital's rule

There are several solution strategies for computing limites whose applicability depends on the situation. For example L'Hospital's rule gives

$$\lim_{h \to 0} \frac{\exp(h) - 1}{\sin(h)} \;=\; \frac{\exp(0)}{\cos(0)} \;=\; 1\,.$$

Thus, if we use algorithmic differentiation, we may be able to implement L'Hospital's rule for expressions of the form

$$\lim_{h \to 0} \frac{f_1(h)}{f_2(h)} \;=\; \lim_{h \to 0} \frac{f_1'(h)}{f_2'(h)}\,,$$

assuming $f_1(0) = f_2(0) = 0$ and that the derivatives of f_1 and f_2 can be computed easily. However, in general L'Hospital's rule may not be applicable.

L'Hospital's rule

There are several solution strategies for computing limites whose applicability depends on the situation. For example L'Hospital's rule gives

$$\lim_{h \to 0} \frac{\exp(h) - 1}{\sin(h)} = \frac{\exp(0)}{\cos(0)} = 1 .$$

Thus, if we use algorithmic differentiation, we may be able to implement L'Hospital's rule for expressions of the form

$$\lim_{h \to 0} \frac{f_1(h)}{f_2(h)} = \lim_{h \to 0} \frac{f_1'(h)}{f_2'(h)} ,$$

assuming $f_1(0) = f_2(0) = 0$ and that the derivatives of f_1 and f_2 can be computed easily. However, in general L'Hospital's rule may not be applicable.

Extrapolation

An alternative to L'Hospital's rule is to use extrapolation. For this aim, we evaluate f at a decreasing sequence of points $h_i > 0$, e.g., $h_1 = \frac{1}{4}$, $h_2 = \frac{1}{8}$, and $h_3 = \frac{1}{16}$.

Next, we compute a polynomial $p(x)$ interpolating the evaluation points

$$(h_1, f(h_1)),\ \ (h_2, f(h_2)),\ \ (h_3, f(h_3)),\ \ldots$$

and use the approximation $\lim_{h\to 0} \frac{f_1(h)}{f_2(h)} \approx p(0)$.

Extrapolation

An alternative to L'Hospital's rule is to use extrapolation. For this aim, we evaluate f at a decreasing sequence of points $h_i > 0$, e.g., $h_1 = \frac{1}{4}$, $h_2 = \frac{1}{8}$, and $h_3 = \frac{1}{16}$.

Next, we compute a polynomial $p(x)$ interpolating the evaluation points

$$(h_1, f(h_1)),\ \ (h_2, f(h_2)),\ \ (h_3, f(h_3)),\ \ldots$$

and use the approximation $\lim_{h\to 0}\ \frac{f_1(h)}{f_2(h)}\ \approx\ p(0)$.

Extrapolation Error

If the function f is $(n+1)$-times continuously differentiable in a small neighborhood of 0 and $p(x)$ a polynomial of degree $\leq n$ which interpolates the points

$$(h_1, f(h)),\ (h_2, f(h_2)),\ \ldots, (h_n, f(h_n))$$

for small $h \geq h_1 > h_2 > \ldots > h_n > 0$, then we have

$$|p(0) - f(0)| \leq \mathbf{O}(h^{n+1})\ .$$

(the proof follows from Taylor's theorem)

Contents

Limitations of Polynomial Interpolation

We have learned in the previous lecture that the interpolation polynomial may not converge uniformly for $n \to \infty$.

Example: Interpolation of

$$f(x) = \frac{1}{x^2 + 1}$$

on the interval $[-5, 5]$ with high order polynomials leads to (unwanted) highly oscillatory interpolation.

Splines

One strategy to overcome this problem is to use splines. Here, we break up the whole interval $[x_{\min}, x_{\max}]$ into sub-intervals and interpolate the function f on each of these sub-intervals with a polynomial of moderate degree (often $n = 3$).

The most common splines are "piecewise linear interpolation" and "cubic splines".

In this lecture we have a closer look at cubic splines.

Splines

One strategy to overcome this problem is to use splines. Here, we break up the whole interval $[x_{\min}, x_{\max}]$ into sub-intervals and interpolate the function f on each of these sub-intervals with a polynomial of moderate degree (often $n = 3$).

The most common splines are "piecewise linear interpolation" and "cubic splines".

In this lecture we have a closer look at cubic splines.

Splines

One strategy to overcome this problem is to use splines. Here, we break up the whole interval $[x_{\min}, x_{\max}]$ into sub-intervals and interpolate the function f on each of these sub-intervals with a polynomial of moderate degree (often $n = 3$).

The most common splines are "piecewise linear interpolation" and "cubic splines".

In this lecture we have a closer look at cubic splines.

Cubic Splines using Hermite Interpolation

Let f be continuously differentiable. We divide the whole interval $[x_{\min}, x_{\max}]$ into n sub-intervals

$$x_{\min} = x_0 < x_1 < \ldots < x_n = x_{\max} \ .$$

Now, we search for a piecewise cubic approximation of the form

$$p(x) = \left\{ \begin{array}{ll} x_{\min} + f'(x_{\min})(x - x_{\min}) & \text{if } x < x_{\min} \\ p_i(x) & \text{if } x \in [x_i, x_{i+1}] \\ x_{\max} + f'(x_{\max})(x - x_{\max}) & \text{if } x > x_{\max} \end{array} \right\}$$

with $i \in \{0, \ldots, n-1\}$),where the cubic polynomials p_i satisfy

$$p_i(x_i) = f(x_i) \quad \text{and} \quad p_i'(x_i) = f'(x_i)$$

$$p_i(x_{i+1}) = f(x_{i+1}) \quad \text{and} \quad p_i'(x_{i+1}) = f'(x_{i+1}) \qquad \text{(Hermite interpolation!)}$$

for all $i \in \{0, \ldots, n\}$. The function p is called a cubic spline.

Cubic Splines using Hermite Interpolation

Let f be continuously differentiable. We divide the whole interval $[x_{\min}, x_{\max}]$ into n sub-intervals

$$x_{\min} = x_0 < x_1 < \ldots < x_n = x_{\max} \; .$$

Now, we search for a piecewise cubic approximation of the form

$$p(x) = \left\{ \begin{array}{ll} x_{\min} + f'(x_{\min})(x - x_{\min}) & \text{if } x < x_{\min} \\ p_i(x) & \text{if } x \in [x_i, x_{i+1}] \\ x_{\max} + f'(x_{\max})(x - x_{\max}) & \text{if } x > x_{\max} \end{array} \right\}$$

with $i \in \{0, \ldots, n-1\}$), where the cubic polynomials p_i satisfy

$p_i(x_i) = f(x_i) \quad \text{and} \quad p_i'(x_i) = f'(x_i)$

$p_i(x_{i+1}) = f(x_{i+1}) \quad \text{and} \quad p_i'(x_{i+1}) = f'(x_{i+1})$ (Hermite interpolation!)

for all $i \in \{0, \ldots, n\}$. The function p is called a cubic spline.

Cubic Splines using Hermite Interpolation

Let f be continuously differentiable. We divide the whole interval $[x_{\min}, x_{\max}]$ into n sub-intervals

$$x_{\min} = x_0 < x_1 < \ldots < x_n = x_{\max} \; .$$

Now, we search for a piecewise cubic approximation of the form

$$p(x) = \left\{ \begin{array}{ll} x_{\min} + f'(x_{\min})(x - x_{\min}) & \text{if } x < x_{\min} \\ p_i(x) & \text{if } x \in [x_i, x_{i+1}] \\ x_{\max} + f'(x_{\max})(x - x_{\max}) & \text{if } x > x_{\max} \end{array} \right\}$$

with $i \in \{0, \ldots, n-1\}$),where the cubic polynomials p_i satisfy

$$p_i(x_i) = f(x_i) \quad \text{and} \quad p_i'(x_i) = f'(x_i)$$

$$p_i(x_{i+1}) = f(x_{i+1}) \quad \text{and} \quad p_i'(x_{i+1}) = f'(x_{i+1}) \qquad \text{(Hermite interpolation!)}$$

for all $i \in \{0, \ldots, n\}$. The function p is called a cubic spline.

Natural Cubic Splines

Another way to construct cubic splines is by imposing the following conditions on the piecewise cubic function p:

1. The function p satisfies $p(x_i) = f(x_i)$, $i \in \{0, \ldots, n\}$, ($2n$ conditions)
2. The function p is twice continuously differentiable, ($2(n-1)$ conditions)
3. We have $p''(x_{\min}) = p''(x_{\max}) = 0$, ($2$ conditions)

In total, we have $4n$ conditions determining the coefficients of the n cubic polynomials.

Natural Cubic Splines

Notation: $p_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$, assume $x_{i+1} - x_i = h$.

1. Interpolation ($i \in \{1, \ldots, n\}$):

$$a_i = f(x_i) \quad \text{and} \quad a_i - b_i h + c_i h^2 - d_i h^3 = f(x_{i-1})$$

2. First derivatives ($i \in \{1, \ldots, n-1\}$):

$$b_i = b_{i+1} - 2c_{i+1}h + 3d_{i+1}h^2$$

3. Second derivatives ($i \in \{1, \ldots, n-1\}$):

$$c_i = c_{i+1} - 3d_{i+1}h$$

4. Boundaries: $c_n = 0 \qquad c_1 - 3d_1 h = 0$.

Natural Cubic Splines

Notation: $p_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$, assume $x_{i+1} - x_i = h$.

1. Interpolation ($i \in \{1, \ldots, n\}$):

$$a_i = f(x_i) \quad \text{and} \quad a_i - b_i h + c_i h^2 - d_i h^3 = f(x_{i-1})$$

2. First derivatives ($i \in \{1, \ldots, n-1\}$):

$$b_i = b_{i+1} - 2c_{i+1}h + 3d_{i+1}h^2$$

3. Second derivatives ($i \in \{1, \ldots, n-1\}$):

$$c_i = c_{i+1} - 3d_{i+1}h$$

4. Boundaries: $c_n = 0 \qquad c_1 - 3d_1 h = 0.$

Natural Cubic Splines

Notation: $p_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$, assume $x_{i+1} - x_i = h$.

1. Interpolation ($i \in \{1, \ldots, n\}$):

$$a_i = f(x_i) \quad \text{and} \quad a_i - b_i h + c_i h^2 - d_i h^3 = f(x_{i-1})$$

2. First derivatives ($i \in \{1, \ldots, n-1\}$):

$$b_i = b_{i+1} - 2c_{i+1}h + 3d_{i+1}h^2$$

3. Second derivatives ($i \in \{1, \ldots, n-1\}$):

$$c_i = c_{i+1} - 3d_{i+1}h$$

4. Boundaries: $c_n = 0 \qquad c_1 - 3d_1 h = 0$.

Natural Cubic Splines

Notation: $p_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$, assume $x_{i+1} - x_i = h$.

1. Interpolation ($i \in \{1, \ldots, n\}$):

$$a_i = f(x_i) \quad \text{and} \quad a_i - b_i h + c_i h^2 - d_i h^3 = f(x_{i-1})$$

2. First derivatives ($i \in \{1, \ldots, n-1\}$):

$$b_i = b_{i+1} - 2c_{i+1}h + 3d_{i+1}h^2$$

3. Second derivatives ($i \in \{1, \ldots, n-1\}$):

$$c_i = c_{i+1} - 3d_{i+1}h$$

4. Boundaries: $c_n = 0 \qquad c_1 - 3d_1 h = 0$.

Natural Cubic Splines

The equation system can be simplified as follows:

1. We know the coefficients $a_i = f(x_i)$.
2. From $c_i = c_{i+1} - 3d_{i+1}h$ we conclude $d_i = \frac{c_i - c_{i-1}}{3h}$ (define $c_0 = 0$).
3. From $a_i - b_i h + c_i h^2 - d_i h^3 = f(x_{i-1})$ we conclude

$$b_i = \frac{f(x_i) - f(x_{i-1})}{h} + \frac{2c_i + c_{i-1}}{3} h \, .$$

So, in summary, we can express the coefficients a_i, b_i, and d_i in dependence on the sequence $c_0, \dots, c_n$.

From the boundary conditions we know that $c_0 = c_n = 0$.

Natural Cubic Splines

The equation system can be simplified as follows:

1. We know the coefficients $a_i = f(x_i)$.
2. From $c_i = c_{i+1} - 3d_{i+1}h$ we conclude $d_i = \frac{c_i - c_{i-1}}{3h}$ (define $c_0 = 0$).
3. From $a_i - b_ih + c_ih^2 - d_ih^3 = f(x_{i-1})$ we conclude

$$b_i = \frac{f(x_i) - f(x_{i-1})}{h} + \frac{2c_i + c_{i-1}}{3}h\ .$$

So, in summary, we can express the coefficients a_i, b_i, and d_i in dependence on the sequence $c_0, \ldots, c_n$.

From the boundary conditions we know that $c_0 = c_n = 0$.

Natural Cubic Splines

The equation system can be simplified as follows:

1. We know the coefficients $a_i = f(x_i)$.
2. From $c_i = c_{i+1} - 3d_{i+1}h$ we conclude $d_i = \frac{c_i - c_{i-1}}{3h}$ (define $c_0 = 0$).
3. From $a_i - b_i h + c_i h^2 - d_i h^3 = f(x_{i-1})$ we conclude

$$b_i = \frac{f(x_i) - f(x_{i-1})}{h} + \frac{2c_i + c_{i-1}}{3}h \; .$$

So, in summary, we can express the coefficients a_i, b_i, and d_i in dependence on the sequence $c_0, \ldots, c_n$.

From the boundary conditions we know that $c_0 = c_n = 0$.

Natural Cubic Splines

The equation system can be simplified as follows:

1. We know the coefficients $a_i = f(x_i)$.
2. From $c_i = c_{i+1} - 3d_{i+1}h$ we conclude $d_i = \frac{c_i - c_{i-1}}{3h}$ (define $c_0 = 0$).
3. From $a_i - b_i h + c_i h^2 - d_i h^3 = f(x_{i-1})$ we conclude

$$b_i = \frac{f(x_i) - f(x_{i-1})}{h} + \frac{2c_i + c_{i-1}}{3} h \ .$$

So, in summary, we can express the coefficients a_i, b_i, and d_i in dependence on the sequence $c_0, \ldots, c_n$.

From the boundary conditions we know that $c_0 = c_n = 0$.

Natural Cubic Splines

The next step is a bit cumbersome: we have to substitute our expression for a_i, b_i, and d_i into $b_i = b_{i+1} - 2c_{i+1}h + 3d_{i+1}h^2$. This gives the recursion:

$$\begin{aligned} &\tfrac{f(x_i)-f(x_{i-1})}{h} + \tfrac{2c_i+c_{i-1}}{3}h \\ &= \tfrac{f(x_{i+1})-f(x_i)}{h} + \tfrac{2c_{i+1}+c_i}{3}h - 2c_{i+1}h + (c_{i+1}-c_i)h \\ &\Longrightarrow h(c_{i-1}+4c_i+c_{i+1}) = r_i \end{aligned}$$

for $i = 1, \ldots, n-1$ and $c_0 = c_n = 0$ as well as the short-hand

$$r_i = \frac{3}{h}\left(f(x_{i+1}) - 2f(x_i) + f(x_{i-1})\right) \ .$$

Natural Cubic Splines

The next step is a bit cumbersome: we have to substitute our expression for a_i, b_i, and d_i into $b_i = b_{i+1} - 2c_{i+1}h + 3d_{i+1}h^2$.This gives the recursion:

$$\begin{aligned}
&\tfrac{f(x_i)-f(x_{i-1})}{h} + \tfrac{2c_i+c_{i-1}}{3}h \\
&= \tfrac{f(x_{i+1})-f(x_i)}{h} + \tfrac{2c_{i+1}+c_i}{3}h - 2c_{i+1}h + (c_{i+1}-c_i)h \\
&\Longrightarrow h(c_{i-1} + 4c_i + c_{i+1}) = r_i
\end{aligned}$$

for $i = 1, \ldots, n-1$ and $c_0 = c_n = 0$ as well as the short-hand

$$r_i = \frac{3}{h}\left(f(x_{i+1}) - 2f(x_i) + f(x_{i-1})\right) \ .$$

Natural Cubic Splines

Thus, the equation system for $c_1, \ldots, c_{n-1}$ can be written in the form

$$\begin{pmatrix} 4h & h & & \ldots & 0 \\ h & 4h & h & & \\ & \ddots & \ddots & \ddots & \\ \vdots & & h & 4h & h \\ 0 & & & h & 4h \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-2} \\ c_{n-1} \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_{n-2} \\ r_{n-1} \end{pmatrix}$$

This equation system has a unique solution and can be solved using tridiagonal matrix inversion algorithms. (for non-equidistant points x_i a similar result can be obtained)

Properties of Natural Cubic Splines

The natural cubic splines satisfy the inequality

$$\int_{x_{\min}}^{x_{\max}} \left|p''(x)\right|^2 \mathrm{d}x \leq \int_{x_{\min}}^{x_{\max}} \left|f''(x)\right|^2 \mathrm{d}x \; .$$

"The natural cubic spline p never oscillates more than the function f."

Proof: Any twice continuously differentiable function w with $w(x_{\min}) = w(x_{\max}) = 0$ satisfies

$$\begin{aligned} \int_{x_{\min}}^{x_{\max}} p''(x) w''(x) \mathrm{d}x &= \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} p''(x) w''(x) \mathrm{d}x \\ &= \sum_{i=0}^{n-1} \left\{ p'' w' \Big|_{x_i}^{x_{i+1}} - p''' w \Big|_{x_i}^{x_{i+1}} \right\} = 0 \; . \end{aligned}$$

Here, we have used that $p''(x_{\min}) = p''(x_{\max})$ as well as $p'''' \equiv 0$, as p is piecewise cubic. For $w = f - p$ we find

$$\int_{x_{\min}}^{x_{\max}} \left|f''(x)\right|^2 \mathrm{d}x = \int_{x_{\min}}^{x_{\max}} \left|p''(x) + w''(x)\right|^2 \mathrm{d}x \geq \int_{x_{\min}}^{x_{\max}} \left|p''(x)\right|^2 \mathrm{d}x \; .$$

Properties of Natural Cubic Splines

The natural cubic splines satisfy the inequality

$$\int_{x_{\min}}^{x_{\max}} \left|p''(x)\right|^2 \mathrm{d}x \le \int_{x_{\min}}^{x_{\max}} \left|f''(x)\right|^2 \mathrm{d}x \, .$$

"The natural cubic spline p never oscillates more than the function f."

Proof: Any twice continuously differentiable function w with $w(x_{\min}) = w(x_{\max}) = 0$ satisfies

$$\begin{aligned} \textstyle\int_{x_{\min}}^{x_{\max}} p''(x) w''(x) \mathrm{d}x &= \textstyle\sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} p''(x) w''(x) \mathrm{d}x \\ &= \textstyle\sum_{i=0}^{n-1} \left\{ p'' w' \big|_{x_i}^{x_{i+1}} - p''' w \big|_{x_i}^{x_{i+1}} \right\} = 0 \, . \end{aligned}$$

Here, we have used that $p''(x_{\min}) = p''(x_{\max})$ as well as $p'''' \equiv 0$, as p is piecewise cubic. For $w = f - p$ we find

$$\int_{x_{\min}}^{x_{\max}} \left|f''(x)\right|^2 \mathrm{d}x = \int_{x_{\min}}^{x_{\max}} \left|p''(x) + w''(x)\right|^2 \mathrm{d}x \ge \int_{x_{\min}}^{x_{\max}} \left|p''(x)\right|^2 \mathrm{d}x \, .$$

Summary

- Extrapolation can be used to compute limit values of functions with high accuracy (only needed if L'Hospital's rule in combination with algorithmic differentiation is not applicable).
- We have discussed two ways to construct cubic splines:
  - Hermite interpolation (continuously differentiable interpolation)
  - Natural Splines (twice continuously differentiable interpolation)
- Natural splines do not require us to evaluate derivatives of f.
- The average of the square of the second derivative of a natural spline p is bounded by the average of the square of the second derivative of the original function f.