WIKIPEDIA

Spline interpolation

In the mathematical field of numerical analysis, **spline interpolation** is a form of interpolation where the interpolant is a special type of piecewise polynomial called a spline. Spline interpolation is often preferred over polynomial interpolation because the interpolation error can be made small even when using low degree polynomials for the spline.[1] Spline interpolation avoids the problem of Runge's phenomenon, in which oscillation can occur between points when interpolating using high degree polynomials.

Contents

Introduction

Originally, *spline* was a term for elastic rulers that were bent to pass through a number of predefined points ("knots"). These were used to make technical drawings for shipbuilding and construction by hand, as illustrated by Figure 1.

The approach to mathematically model the shape of such elastic rulers fixed by $n + 1$ knots $\{(x_i, y_i) : i = 0, 1, \cdots, n\}$ is to interpolate between all the pairs of knots (x_{i-1}, y_{i-1}) and (x_i, y_i) with polynomials $y = q_i(x), i = 1, 2, \cdots, n$.

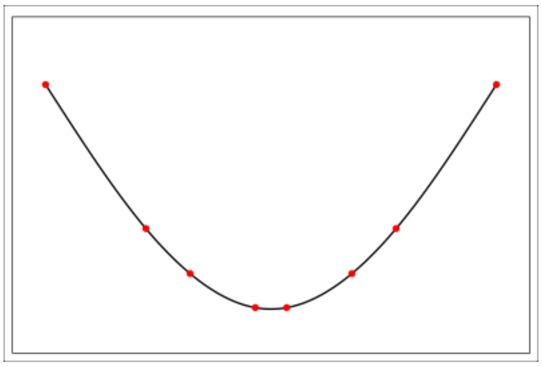
Figure 1: Interpolation with cubic splines between eight points. Hand-drawn technical drawings were made for shipbuilding etc. using flexible rulers that were bent to follow pre-defined points

The curvature of a curve $y = f(x)$ is given by:

$$\kappa = \frac{y''}{(1 + y'^2)^{3/2}}$$

As the spline will take a shape that minimizes the bending (under the constraint of passing through all knots) both y' and y'' will be continuous everywhere and at the knots. To achieve this one must have that

$$\begin{cases} q'_i(x_i) = q'_{i+1}(x_i) \\ q''_i(x_i) = q''_{i+1}(x_i) \end{cases} \qquad 1 \le i \le n-1$$

This can only be achieved if polynomials of degree 3 or higher are used. The classical approach is to use polynomials of degree 3 — the case of cubic splines.

Algorithm to find the interpolating cubic spline

A third-order polynomial $q(x)$ for which

$$\begin{aligned} q(x_1) &= y_1 \\ q(x_2) &= y_2 \\ q'(x_1) &= k_1 \\ q'(x_2) &= k_2 \end{aligned}$$

can be written in the symmetrical form

$$q(x) = (1 - t(x))y_1 + t(x)y_2 + t(x)(1 - t(x))(a(1 - t(x)) + bt(x)) \qquad (1)$$

where

$$t(x) = \frac{x - x_1}{x_2 - x_1}, \qquad (2)$$

$$a = k_1(x_2 - x_1) - (y_2 - y_1), \qquad (3)$$

$$b = -k_2(x_2 - x_1) + (y_2 - y_1). \qquad (4)$$

As

$$q' = \frac{dq}{dx} = \frac{dq}{dt}\frac{dt}{dx} = \frac{dq}{dt}\frac{1}{x_2 - x_1}$$

one gets that:

$$q' = \frac{y_2 - y_1}{x_2 - x_1} + (1-2t)\frac{a(1-t)+bt}{x_2 - x_1} + t(1-t)\frac{b-a}{x_2 - x_1}, \tag{5}$$

$$q'' = 2\frac{b - 2a + (a-b)3t}{(x_2 - x_1)^2}. \tag{6}$$

Setting $x = x_1$ and $x = x_2$ respectively in equations (5) and (6) one gets from (2) that indeed first derivatives $q'(x_1) = k_1$ and $q'(x_2) = k_2$ and also second derivatives

$$q''(x_1) = 2\frac{b - 2a}{(x_2 - x_1)^2} \tag{7}$$

$$q''(x_2) = 2\frac{a - 2b}{(x_2 - x_1)^2} \tag{8}$$

If now (x_i, y_i), $i = 0, 1, ..., n$ are $n + 1$ points and

$$q_i = (1-t)y_{i-1} + ty_i + t(1-t)(a_i(1-t) + b_i t) \tag{9}$$

where i = 1, 2, ..., n and $t = \frac{x - x_{i-1}}{x_i - x_{i-1}}$ are n third degree polynomials interpolating y in the interval $x_{i-1} \le x \le x_i$ for i = 1, ..., n such that $q'_i\ (x_i) = q'_{i+1}(x_i)$ for i = 1, ..., n−1 then the n polynomials together define a differentiable function in the interval $x_0 \le x \le x_n$ and

$$a_i = k_{i-1}(x_i - x_{i-1}) - (y_i - y_{i-1}) \tag{10}$$

$$b_i = -k_i(x_i - x_{i-1}) + (y_i - y_{i-1}) \tag{11}$$

for i = 1, ..., n where

$$k_0 = q'_1(x_0) \tag{12}$$

$$k_i = q'_i(x_i) = q'_{i+1}(x_i) \qquad i = 1, \ldots, n-1 \tag{13}$$

$$k_n = q'_n(x_n) \tag{14}$$

If the sequence $k_0, k_1, ..., k_n$ is such that, in addition, $q''_i(x_i) = q''_{i+1}(x_i)$ holds for i = 1, ..., n-1, then the resulting function will even have a continuous second derivative.

From (7), (8), (10) and (11) follows that this is the case if and only if

$$\frac{k_{i-1}}{x_i - x_{i-1}} + \left(\frac{1}{x_i - x_{i-1}} + \frac{1}{x_{i+1} - x_i}\right)2k_i + \frac{k_{i+1}}{x_{i+1} - x_i} = 3\left(\frac{y_i - y_{i-1}}{(x_i - x_{i-1})^2} + \frac{y_{i+1} - y_i}{(x_{i+1} - x_i)^2}\right)$$

for i = 1, ..., n-1. The relations (15) are $n - 1$ linear equations for the $n + 1$ values $k_0, k_1, ..., k_n$.

For the elastic rulers being the model for the spline interpolation one has that to the left of the left-most "knot" and to the right of the right-most "knot" the ruler can move freely and will therefore take the form of a straight line with $q'' = 0$. As q'' should be a continuous function of x one gets that for "Natural Splines" one in addition to the $n - 1$ linear equations (15) should have that

$$q''_1(x_0) = 2\frac{3(y_1 - y_0) - (k_1 + 2k_0)(x_1 - x_0)}{(x_1 - x_0)^2} = 0,$$
$$q''_n(x_n) = -2\frac{3(y_n - y_{n-1}) - (2k_n + k_{n-1})(x_n - x_{n-1})}{(x_n - x_{n-1})^2} = 0,$$

i.e. that

$$\frac{2}{x_1 - x_0}k_0 + \frac{1}{x_1 - x_0}k_1 = 3\frac{y_1 - y_0}{(x_1 - x_0)^2}, \tag{16}$$

$$\frac{1}{x_n - x_{n-1}}k_{n-1} + \frac{2}{x_n - x_{n-1}}k_n = 3\frac{y_n - y_{n-1}}{(x_n - x_{n-1})^2}. \tag{17}$$

Eventually, (15) together with (16) and (17) constitute $n + 1$ linear equations that uniquely define the $n + 1$ parameters $k_0, k_1, ..., k_n$.

There exist other end conditions: "Clamped spline", that specifies the slope at the ends of the spline, and the popular "not-a-knot spline", that requires that the third derivative is also continuous at the x_1 and x_{N-1} points. For the "not-a-knot" spline, the additional equations will read:

$$q_1'''(x_1) = q_2'''(x_1) \Rightarrow \frac{1}{\Delta x_1^2} k_0 + \left(\frac{1}{\Delta x_1^2} - \frac{1}{\Delta x_2^2}\right) k_1 - \frac{1}{\Delta x_2^2} k_2 = 2\left(\frac{\Delta y_1}{\Delta x_1^3} - \frac{\Delta y_2}{\Delta x_2^3}\right)$$
$$q_{n-1}'''(x_{n-1}) = q_n'''(x_{n-1}) \Rightarrow \frac{1}{\Delta x_{n-1}^2} k_{n-2} + \left(\frac{1}{\Delta x_{n-1}^2} - \frac{1}{\Delta x_n^2}\right) k_{n-1} - \frac{1}{\Delta x_n^2} k_n = 2\left(\frac{\Delta y_{n-1}}{\Delta x_{n-1}^3} - \frac{\Delta y_n}{\Delta x_n^3}\right)$$

where $\Delta x_i = x_i - x_{i-1}, \Delta y_i = y_i - y_{i-1}$.

Example

In case of three points the values for k_0, k_1, k_2 are found by solving the tridiagonal linear equation system

$$\begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & a_{23} \\ 0 & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} k_0 \\ k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

with

$$\begin{aligned} a_{11} &= \frac{2}{x_1 - x_0} \\ a_{12} &= \frac{1}{x_1 - x_0} \\ a_{21} &= \frac{1}{x_1 - x_0} \\ a_{22} &= 2\left(\frac{1}{x_1 - x_0} + \frac{1}{x_2 - x_1}\right) \\ a_{23} &= \frac{1}{x_2 - x_1} \\ a_{32} &= \frac{1}{x_2 - x_1} \\ a_{33} &= \frac{2}{x_2 - x_1} \\ b_1 &= 3\,\frac{y_1 - y_0}{(x_1 - x_0)^2} \\ b_2 &= 3\left(\frac{y_1 - y_0}{(x_1 - x_0)^2} + \frac{y_2 - y_1}{(x_2 - x_1)^2}\right) \\ b_3 &= 3\,\frac{y_2 - y_1}{(x_2 - x_1)^2} \end{aligned}$$

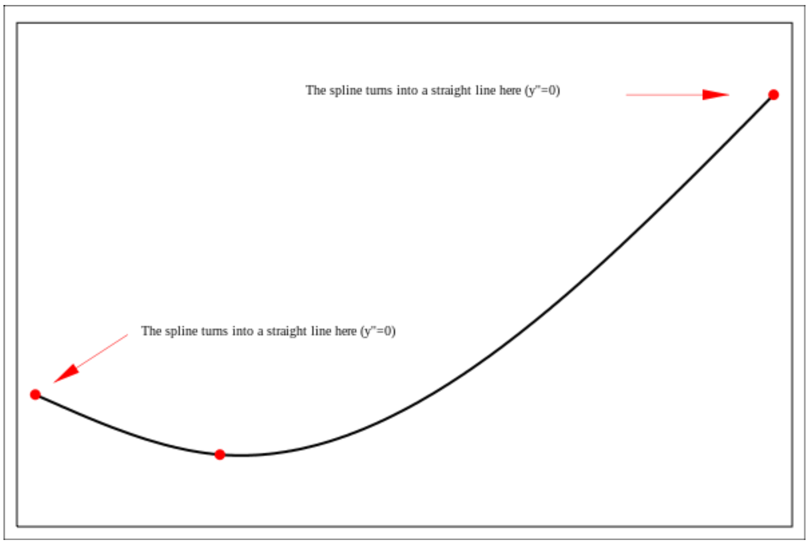

Figure 2: Interpolation with cubic "natural" splines between three points.

For the three points

$$(-1, 0.5)\ ,\ (0, 0)\ ,\ (3, 3),$$

one gets that

$$k_0 = -0.6875\ ,\ k_1 = -0.1250\ ,\ k_2 = 1.5625$$

and from (**10**) and (**11**) that

$$\begin{aligned} a_1 &= k_0(x_1 - x_0) - (y_1 - y_0) = -0.1875 \\ b_1 &= -k_1(x_1 - x_0) + (y_1 - y_0) = -0.3750 \\ a_2 &= k_1(x_2 - x_1) - (y_2 - y_1) = -3.3750 \\ b_2 &= -k_2(x_2 - x_1) + (y_2 - y_1) = -1.6875 \end{aligned}$$

In Figure 2, the spline function consisting of the two cubic polynomials $q_1(x)$ and $q_2(x)$ given by (**9**) is displayed.

See also

- Cubic Hermite spline
- Centripetal Catmull–Rom spline
- Discrete spline interpolation
- Monotone cubic interpolation
- NURBS
- Multivariate interpolation
- Polynomial interpolation
- Smoothing spline
- Spline wavelet
- Thin plate spline
- Polyharmonic spline

Computer code

TinySpline: Open source C-library for splines which implements cubic spline interpolation (https://github.com/msteinbeck/tinyspline)

SciPy Spline Interpolation: a Python package that implements interpolation (https://docs.scipy.org/doc/scipy/reference/tutorial/interpolate.html#id5)

Cubic Interpolation: Open source C#-library for cubic spline interpolation by Vadim A. Onuchin, Valex Corp. (https://github.com/ValexCorp/Cubic-Interpolation)

References

1. Hall, Charles A.; Meyer, Weston W. (1976). "Optimal Error Bounds for Cubic Spline Interpolation" (https://www.sciencedirect.com/science/article/pii/002190457690040X). *Journal of Approximation Theory*. **16** (2): 105–122.

- Schoenberg, Isaac J. (1946). "Contributions to the Problem of Approximation of Equidistant Data by Analytic Functions: Part A.—On the Problem of Smoothing or Graduation. A First Class of Analytic Approximation Formulae" (http://www.ams.org/journals/qam/1946-04-01/S0033-569X-1946-15914-5/S0033-569X-1946-15914-5.pdf) (PDF). *Quarterly of Applied Mathematics*. **4** (2): 45–99.
- Schoenberg, Isaac J. (1946). "Contributions to the Problem of Approximation of Equidistant Data by Analytic Functions: Part B.—On the Problem of Osculatory Interpolation. A Second Class of Analytic Approximation Formulae" (http://www.ams.org/journals/qam/1946-04-02/S0033-569X-1946-16705-2/S0033-569X-1946-16705-2.pdf) (PDF). *Quarterly of Applied Mathematics*. **4** (2): 112–141.

External links

- Cubic Spline Interpolation Online Calculation and Visualization Tool (with JavaScript source code) (http://tools.timodenk.com/?p=cubic-spline-interpolation)
- Hazewinkel, Michiel, ed. (2001) [1994], "Spline interpolation" (https://www.encyclopediaofmath.org/index.php?title=p/s086820), *Encyclopedia of Mathematics*, Springer Science+Business Media B.V. / Kluwer Academic Publishers, ISBN 978-1-55608-010-4
- Dynamic cubic splines with JSXGraph (http://jsxgraph.uni-bayreuth.de/wiki/index.php/Cubic_spline_interpolation)
- Lectures on the theory and practice of spline interpolation (https://www.youtube.com/view_play_list?p=DAB608CD1A9A0D55)
- Paper which explains step by step how cubic spline interpolation is done, but only for equidistant knots. (https://web.archive.org/web/20090408054627/http://online.redwoods.cc.ca.us/instruct/darnold/laproj/Fall98/SkyMeg/Proj.PDF)
- Numerical Recipes in C, Go to Chapter 3 Section 3-3 (http://apps.nrbook.com/c/index.html)
- A note on cubic splines (http://www.cs.tau.ac.il/~turkel/notes/numeng/spline_note.pdf)
- Information about spline interpolation (including code in Fortran 77) (https://websites.pmc.ucsc.edu/~fnimmo/eart290c_17/NumericalRecipesinF77.pdf)