



WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
العربية
Čeština
Deutsch
Français
Italiano
Русский
Svenska
Українська
 Edit links

Create account Log in

Article Talk

Read Edit View history

Search

Cubic Hermite spline

From Wikipedia, the free encyclopedia
(Redirected from [Cubic interpolation](#))

Not to be confused with [Hermite polynomial](#).

In [numerical analysis](#), a **cubic Hermite spline** or **cubic Hermite interpolator** is a [spline](#) where each piece is a third-degree [polynomial](#) specified in Hermite form:^[1] that is, by its values and first [derivatives](#) at the end points of the corresponding [domain](#) interval.

Cubic Hermite splines are typically used for [interpolation](#) of numeric data specified at given argument values x_1, x_2, \ldots, x_n , to obtain a smooth [continuous function](#). The data should consist of the desired function value and derivative at each x_k . (If only the values are provided, the derivatives must be estimated from them.) The Hermite formula is applied to each interval (x_k, x_{k+1}) separately. The resulting spline will be continuous and will have continuous first derivative.

Cubic polynomial splines can be specified in other ways, the [Bézier form](#) being the most common. However, these two methods provide the same set of splines, and data can be easily converted between the Bézier and Hermite forms; so the names are often used as if they were synonymous.

Cubic polynomial splines are extensively used in [computer graphics](#) and [geometric modeling](#) to obtain [curves](#) or motion [trajectories](#) that pass through specified points of the [plane](#) or three-dimensional [space](#). In these applications, each coordinate of the plane or space is separately interpolated by a cubic spline function of a separate parameter t .

Cubic splines can be extended to functions of two or more parameters, in several ways. Bicubic splines ([Bicubic interpolation](#)) are often used to interpolate data on a regular rectangular grid, such as [pixel](#) values in a [digital image](#) or [altitude](#) data on a terrain. [Bicubic surface patches](#), defined by three bicubic splines, are an essential tool in computer graphics.

Cubic splines are often called **csplines**, especially in computer graphics. Hermite splines are named after [Charles Hermite](#).

Contents [hide]

- 1 Interpolation on a single interval
 - 1.1 Unit interval (0, 1)
 - 1.2 Interpolation on an arbitrary interval
 - 1.3 Uniqueness
 - 1.4 Representations
- 2 Interpolating a data set
 - 2.1 Finite difference
 - 2.2 Cardinal spline
 - 2.3 Catmull–Rom spline
 - 2.4 Kochanek–Bartels spline
 - 2.5 Monotone cubic interpolation
- 3 Interpolation on the unit interval without exact derivatives
- 4 See also
- 5 References
- 6 External links

Interpolation on a single interval [\[edit\]](#)

Unit interval (0, 1) [\[edit\]](#)

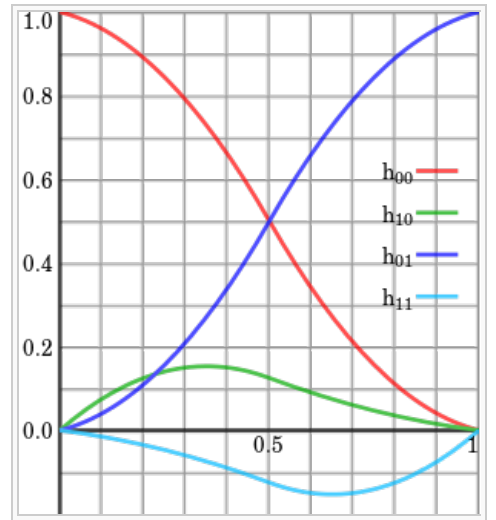
On the unit interval $(0, 1)$, given a starting point \mathbf{p}_0 at $t = 0$ and an ending point \mathbf{p}_1 at $t = 1$ with starting tangent \mathbf{m}_0 at $t = 0$ and ending tangent \mathbf{m}_1 at $t = 1$, the polynomial can be defined by

$$\mathbf{p}(t) = (2t^3 - 3t^2 + 1)\mathbf{p}_0 + (t^3 - 2t^2 + t)\mathbf{m}_0 + (-2t^3 + 3t^2)\mathbf{p}_1 + (t^3 - t^2)\mathbf{m}_1$$

where $t \in [0, 1]$.

Interpolation on an arbitrary interval [\[edit\]](#)

Interpolating x in an arbitrary interval (x_k, x_{k+1}) is done by mapping the latter to $[0, 1]$ through an [affine](#) (degree 1) change of variable. The formula is



The four Hermite basis functions. The interpolant in each subinterval is a linear combination of these four functions. ↗

$$\mathbf{p}(x) = h_{00}(t)\mathbf{p}_k + h_{10}(t)(x_{k+1} - x_k)\mathbf{m}_k + h_{01}(t)\mathbf{p}_{k+1} + h_{11}(t)(x_{k+1} - x_k)\mathbf{m}_{k+1}.$$

with $t = (x - x_k)/(x_{k+1} - x_k)$ and \mathbf{h} refers to the basis functions, defined below. Note that the tangent values have been scaled by $x_{k+1} - x_k$ compared to the equation on the unit interval.

Uniqueness [\[edit\]](#)

The formulae specified above provide the unique third-degree polynomial path between the two points with the given tangents.

Proof:

Let $Q(x)$ be another third degree polynomial satisfying the given boundary conditions. Define

$R(x) = Q(x) - P(x)$. Since both Q and P are third degree polynomials, R is at most a third degree polynomial. Furthermore:

$$\begin{aligned} R(0) &= Q(0) - P(0) = 0 \text{ (We assume both } P \text{ and } Q \text{ satisfy the boundary conditions)} \\ R(1) &= 0 \end{aligned}$$

So R must be of the form:

$$\begin{aligned} R(x) &= ax(x-1)(x-r) \\ R'(x) &= ax(x-1) + ax(x-r) + a(x-1)(x-r) \end{aligned}$$

We know furthermore that:

$$\begin{aligned} R'(0) &= Q'(0) - P'(0) = 0 \\ R'(0) &= 0 = ar \end{aligned} \tag{1}$$

$$\begin{aligned} R'(1) &= Q'(1) - P'(1) = 0 \\ R'(1) &= 0 = a(1-r) \end{aligned} \tag{2}$$

Putting (1) and (2) together, we deduce that $a = 0$ and therefore $R = 0$, thus $P(x) = Q(x)$

Representations [\[edit\]](#)

We can write the interpolation polynomial as

$$\mathbf{p}(t) = h_{00}(t)\mathbf{p}_0 + h_{10}(t)\mathbf{m}_0 + h_{01}(t)\mathbf{p}_1 + h_{11}(t)\mathbf{m}_1$$

where $h_{00}, h_{10}, h_{01}, h_{11}$ are Hermite basis functions. These can be written in different ways, each way revealing different properties.

| | expanded | factorized | Bernstein |
|-------------|-------------------|-----------------|----------------------------|
| $h_{00}(t)$ | $2t^3 - 3t^2 + 1$ | $(1+2t)(1-t)^2$ | $B_0(t) + B_1(t)$ |
| $h_{10}(t)$ | $t^3 - 2t^2 + t$ | $t(1-t)^2$ | $\frac{1}{3} \cdot B_1(t)$ |
| $h_{01}(t)$ | $-2t^3 + 3t^2$ | $t^2(3-2t)$ | $B_3(t) + B_2(t)$ |
| | | | |

| | | | |
|-------------|-------------|------------|-----------------------------|
| $h_{11}(t)$ | $t^3 - t^2$ | $t^2(t-1)$ | $-\frac{1}{3} \cdot B_2(t)$ |
|-------------|-------------|------------|-----------------------------|

The "expanded" column shows the representation used in the definition above. The "factorized" column shows immediately, that h_{10} and h_{11} are zero at the boundaries. You can further conclude that h_{01} and h_{11} have a **zero of multiplicity 2** at 0 and h_{00} and h_{10} have such a zero at 1, thus they have slope 0 at those boundaries. The "Bernstein" column shows the decomposition of the Hermite basis functions into **Bernstein polynomials** of order 3:

$$B_k(t) = \binom{3}{k} \cdot t^k \cdot (1-t)^{3-k}$$

Using this connection you can express cubic Hermite interpolation in terms of cubic **Bézier curves** with respect to the four values $\mathbf{p}_0, \mathbf{p}_0 + \frac{\mathbf{m}_0}{3}, \mathbf{p}_1 - \frac{\mathbf{m}_1}{3}, \mathbf{p}_1$ and do Hermite interpolation using the **de Casteljau algorithm**. It shows that in a cubic Bézier patch the two control points in the middle determine the tangents of the interpolation curve at the respective outer points.

Interpolating a data set [\[edit\]](#)

A data set, (t_k, \mathbf{p}_k) for $k = 1, \dots, n$, can be interpolated by applying the above procedure on each interval, where the tangents are chosen in a sensible manner, meaning that the tangents for intervals sharing endpoints are equal. The interpolated curve then consists of piecewise cubic Hermite splines, and is globally continuously differentiable in (t_1, t_n) .

The choice of tangents is non-unique, and there are several options available.

Finite difference [\[edit\]](#)

The simplest choice is the three-point difference, not requiring constant interval lengths,

$$\mathbf{m}_k = \frac{\mathbf{p}_{k+1} - \mathbf{p}_k}{2(t_{k+1} - t_k)} + \frac{\mathbf{p}_k - \mathbf{p}_{k-1}}{2(t_k - t_{k-1})}$$

for internal points $k = 2, \dots, n-1$, and one-sided difference at the endpoints of the data set.

Cardinal spline [\[edit\]](#)

A **cardinal spline**, sometimes called a **canonical spline**,^[2] is obtained^[3] if

$$\mathbf{m}_k = (1-c) \frac{\mathbf{p}_{k+1} - \mathbf{p}_{k-1}}{t_{k+1} - t_{k-1}}$$

is used to calculate the tangents. The parameter c is a *tension* parameter that must be in the interval $(0, 1)$. In some sense, this can be interpreted as the "length" of the tangent. $c = 1$ will yield all zero tangents, and $c = 0$ yields a Catmull–Rom spline.

Catmull–Rom spline [\[edit\]](#)

See also: *Centripetal Catmull–Rom spline*

For tangents chosen to be

$$\mathbf{m}_k = \frac{\mathbf{p}_{k+1} - \mathbf{p}_{k-1}}{t_{k+1} - t_{k-1}}$$

a **Catmull–Rom spline** is obtained, being a special case of a cardinal spline.

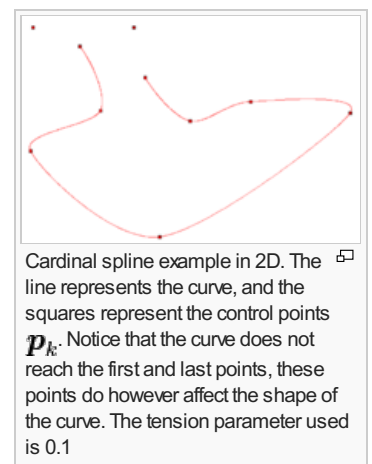
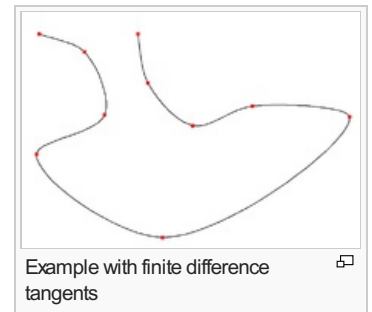
This assumes uniform parameter spacing.

The curve is named after **Edwin Catmull** and **Raphael Rom**. The principal advantage of this technique is that the points along the original set of points also make up the control points for the spline curve.^[4] Two additional points are required on either end of the curve. The default implementation of the Catmull–Rom algorithm can produce loops and self intersections. The chordal and **centripetal Catmull–Rom** implementations^[5] solve this problem, but use a slightly different calculation.^[6] In **computer graphics**, Catmull–Rom splines are frequently used to get smooth interpolated motion between **key frames**. For example, most camera path animations generated from discrete key-frames are handled using Catmull–Rom splines. They are popular mainly for being relatively easy to compute, guaranteeing that each key frame position will be hit exactly, and also guaranteeing that the tangents of the generated curve are continuous over multiple segments.

Kochanek–Bartels spline [\[edit\]](#)

*Main article: **Kochanek–Bartels spline***

A Kochanek–Bartels spline is a further generalization on how to choose the tangents given the data points $\mathbf{p}_{k-1}, \mathbf{p}_k$



and \mathbf{p}_{k+1} , with three parameters possible, tension, bias and a continuity parameter.

Monotone cubic interpolation [\[edit\]](#)

Main article: [Monotone cubic interpolation](#)

If a cubic Hermite spline of any of the above listed types is used for [interpolation](#) of a [monotonic](#) data set, the interpolated function will not necessarily be monotonic, but monotonicity can be preserved by adjusting the tangents.

Interpolation on the unit interval without exact derivatives [\[edit\]](#)

Given p_{-1} , p_0 , p_1 and p_2 as the values that the function should take on at -1 , 0 , 1 and 2 , we can use centered differences instead of exact derivatives.^[7] Thus the Catmull–Rom spline is

$$\text{CINT}_{x(p_{-1}, p_0, p_1, p_2)} = \frac{1}{2} \begin{pmatrix} -x^3 + 2x^2 - x \\ 3x^3 - 5x^2 + 2 \\ -3x^3 + 4x^2 + x \\ x^3 - x^2 \end{pmatrix} \cdot \begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \\ p_2 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} x((2-x)x-1) \\ x^2(3x-5)+2 \\ x((4-3x)x+1) \\ (x-1)x^2 \end{pmatrix} \cdot \begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \\ p_2 \end{pmatrix}$$

for $x \in [0, 1]$, where the left-hand vector is independent of the p .

This writing is relevant for [tricubic interpolation](#), where one optimization requires you to compute CINT_x sixteen times with the same x and different p .







See also [\[edit\]](#)

- **Bicubic interpolation**, a generalization to two dimensions
- **Tricubic interpolation**, a generalization to three dimensions
- **Hermite interpolation**
- **Multivariate interpolation**
- **Spline interpolation**
- **Discrete spline interpolation**

References [\[edit\]](#)

1. [^] Erwin Kreyszig (2005). *Advanced Engineering Mathematics* (9 ed.). Wiley. p. 816. [ISBN 9780471488859](#).
2. [^] Charles Petzold. "Canonical Splines in WPF and Silverlight" [↗](#). 2009.
3. [^] [Cardinal Splines at Microsoft Developer Network](#) [↗](#)
4. [^] Catmull, Edwin; Rom, Raphael (1974), "A class of local interpolating splines", in Barnhill, R. E.; Riesenfeld, R. F., *Computer Aided Geometric Design*, New York: Academic Press, pp. 317–326
5. [^] N. Dyn, M. S. Floater, and K. Hormann. Four-point curve subdivision based on iterated chordal and centripetal parameterizations. *Computer Aided Geometric Design*, 26(3):279{286, 2009
6. [^] P. J. Barry and R. N. Goldman. A recursive evaluation algorithm for a class of Catmull-Rom splines. *SIGGRAPH Computer Graphics*, 22(4):199{204, 1988.
7. [^] [Two hierarchies of spline interpolations. Practical algorithms for multivariate higher order splines](#) [↗](#)

External links [[edit](#)]

- [Spline Curves](#) , Prof. Donald H. House [Clemson University](#)
- [Multi-dimensional Hermite Interpolation and Approximation](#) , Prof. Chandrajit Bajaj, [Purdue University](#)
- [Introduction to Catmull–Rom Splines](#) , MVPs.org
- [Interpolating Cardinal and Catmull–Rom splines](#) 
- [Interpolation methods: linear, cosine, cubic and hermite \(with C sources\)](#) 
- [Common Spline Equations](#) 

Categories: [Splines](#) | [Interpolation](#)

This page was last modified on 28 August 2015, at 22:14.

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. By using this site, you agree to the [Terms of Use](#) and [Privacy Policy](#). Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#) [Contact Wikipedia](#) [Developers](#) [Mobile view](#)

