Algorithmic Geometry WS 2017/2018

Prof. Dr. Hans Hagen
Benjamin Karer M.Sc.

http://gfx.uni-kl.de/~alggeom

Interpolation

Point Data Interpolation

Given: Set of $n+1$ data points (t_i, f_i); $i = 0, \ldots, n$;
with pairwise disjoint nodes t_i. $(t_i \neq t_j)$

Figure: Given Points

Point Data Interpolation

Find: Polynomial with degree $\leq n$ with

$$P(t) = \sum_{i=0}^{n} c_i t^i \tag{1}$$

Point Data Interpolation

$P(t)$ is called **Interpolation Polynomial** to (t_i, f_i), if

$$p(t_i) = f_i; \quad i = 0, \ldots, n \tag{2}$$

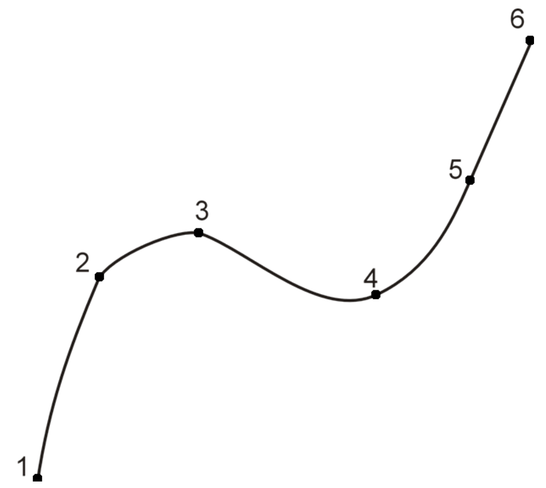


Figure: Interpolation Polynomial

Key Problems

(a) Existence of solution
(b) Uniqueness of the solution
(c) Quality of solution
(d) Algorithm

Theorem

There is a unique solution to the Point Data Interpolation Problem

Vandermonde-Matrix

Proof: Combining (1) and (2) leads to

$$\begin{bmatrix} 1 & t_0 & t_0^2 & \cdot & \cdot & t_0^n \\ 1 & t_1 & t_1^2 & \cdot & \cdot & t_1^n \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & t_n & t_n^2 & \cdot & \cdot & t_n^n \end{bmatrix} \cdot \begin{bmatrix} c_0 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ c_n \end{bmatrix} = \begin{bmatrix} f_0 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ f_n \end{bmatrix} \qquad (3)$$

or $Ac = f$.

A is called *Vandermonde-Matrix* with $\det A = \prod_{i,j=0;\ i>j}^{n} (t_i - t_j)$.

Because $t_i \neq t_j$ for $i \neq j$

$\Longrightarrow$ A is non-singular and a unique solution exists. □

Point Data Interpolation in 3D space

Given: $n+1$ data points $f_i = (x_i, y_i, z_i) \in \mathbb{R}^3$ with disjoint nodes t_i.
Find: Interpolating polynomial space curve p with $p(t_i) = f_i$.

Solution:

- Analogous to 2D with $Ac = f$
- The coefficients are 3D points, i.e. $c_i \in \mathbb{R}^3$
- For each coordinate there is an equation system with the matrix A.

Interpolation in 3D

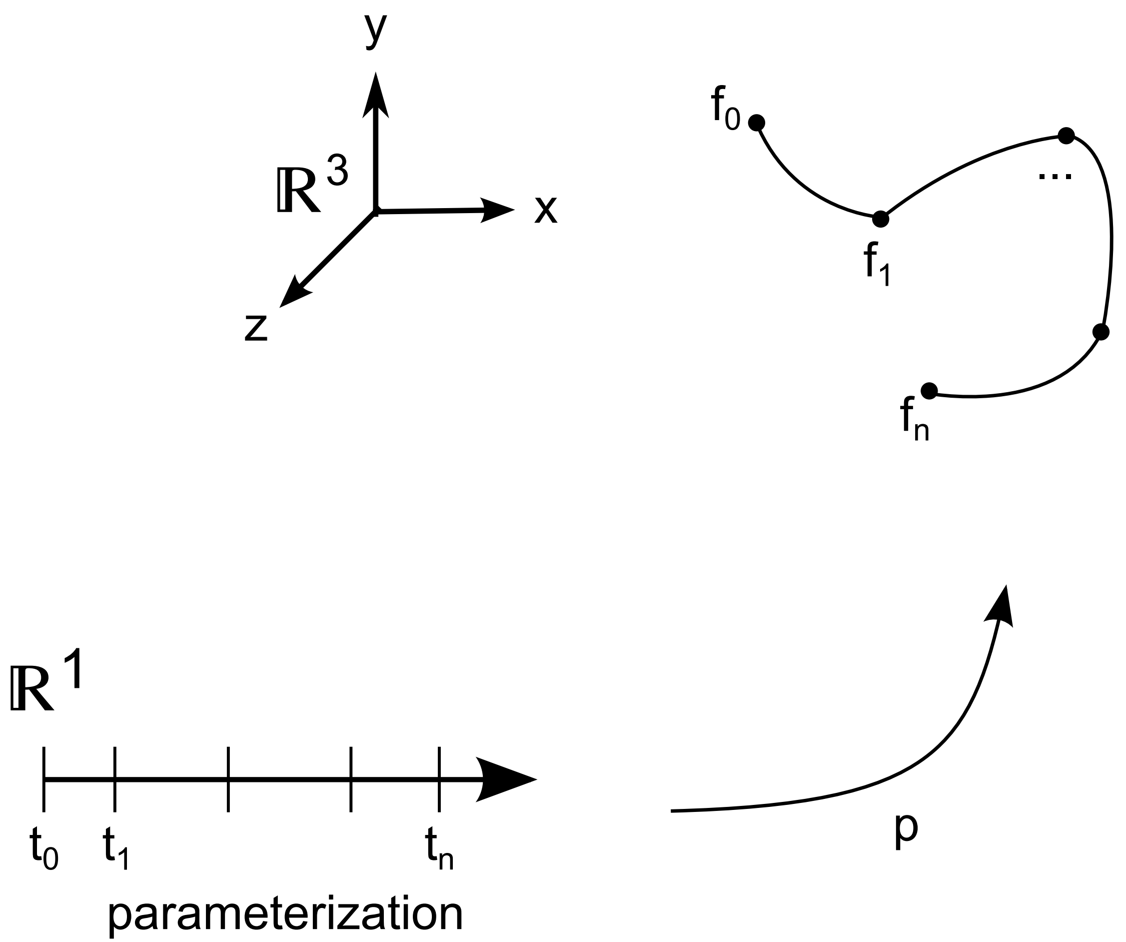


Figure: Interpolation in 3D

Lagrange-Interpolation

- In general, the runtime for solving a system of equations is $\mathcal{O}(n^3)$.
- Find an optimal polynomial basis $\{L_i(t)\}_{i=0}^{n}$, with

$$L_i(t_j) = \delta_{ij} =: \begin{cases} 0, i \neq j \\ 1, i = j \end{cases} \tag{4}$$

- Because of that basis, the matrix of the system (3) becomes an identity matrix, i.e. $c_i = f_i,\ i = 0, \ldots, n$ and the interpolation polynomial can be written as

$$p(t) = \sum_{i=0}^{n} f_i L_i(t). \tag{5}$$

Theorem

The ***Lagrange polynomials***

$$L_i(t) = \prod_{k=0, k \neq i}^{n} \frac{t - t_k}{t_i - t_k} \tag{6}$$

satisfy the property (4): $L_i(t_j) = \delta_{ij}$.

Proof:

$$L_i(t_i) = \prod_{k=0, k \neq i}^{n} \frac{t_i - t_k}{t_i - t_k} = 1$$

and with $j \neq i$:

$$L_i(t_j) = \cdots \frac{t_j - t_j}{t_i - t_j} \cdots = 0 \quad \square$$

Lagrange polynomials

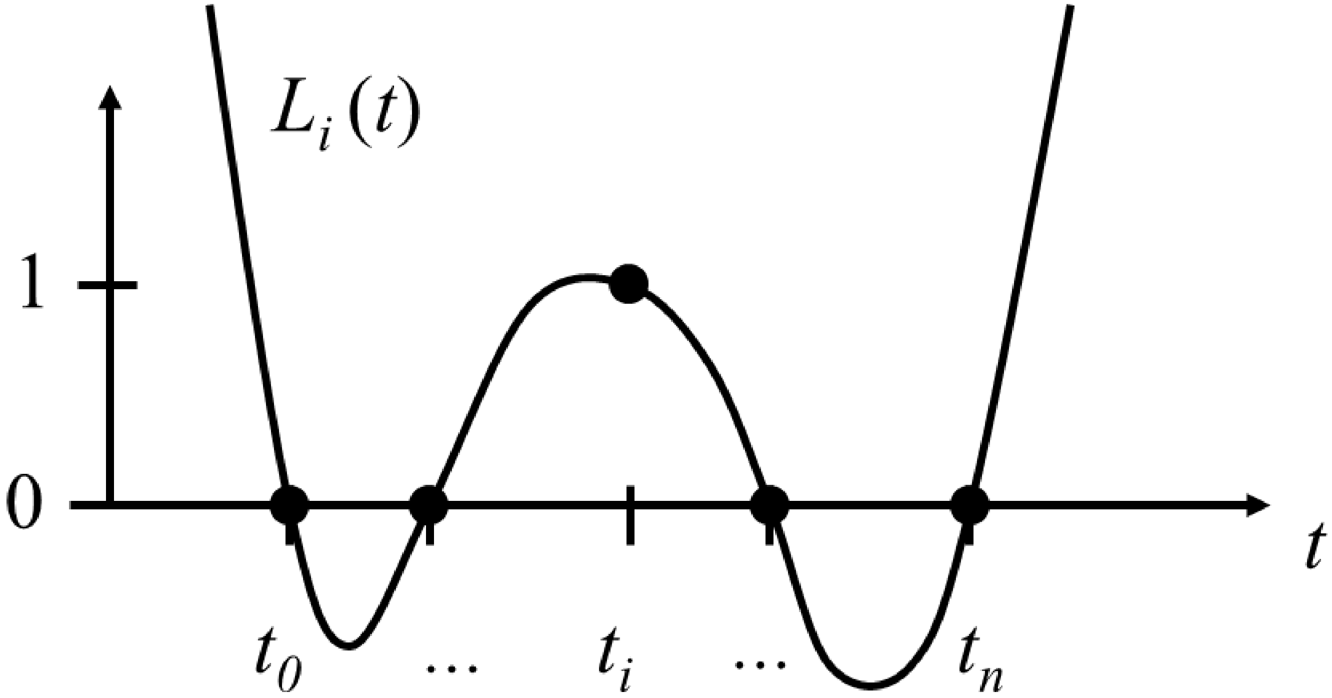


Figure: Lagrange polynomials

Algorithm: Lagrange-Interpolation

Input: $(x_i, y_i); i = 0, \ldots, n; x_i \neq x_j; i \neq j$

1. step:

$$L_i(x) = \frac{(x - x_0)\cdots(x - x_{i-1})(x - x_{i+1})\cdots(x - x_n)}{(x_i - x_0)\cdots(x_i - x_{i-1})(x_i - x_{i+1})\cdots(x_i - x_n)} = \prod_{j=0; j \neq i}^{n} \frac{(x - x_j)}{(x_i - x_j)}$$

2.step:

$$P(x) = \sum_{i=0}^{n} y_i L_i(x)$$

Examples

1. $n = 1, \implies$ linear interpolation

   Point data: $(x_0, y_0), (x_1, y_1)$ (nodes in x)

   $$L_0(x) = \frac{x - x_1}{x_0 - x_1} \qquad L_1(x) = \frac{x - x_0}{x_1 - x_0}$$

   $$P(x) = y_0 \frac{x - x_1}{x_0 - x_1} + y_1 \frac{x - x_0}{x_1 - x_0} = \frac{y_0(x - x_1) + y_1(x_0 - x)}{x_0 - x_1}$$

2. Following points of a real function are known:

i	x_i	y_i
0	0	1
1	1	0.5
2	2	0.2

Examples

$$L_0(x) = \frac{(x - x_1)\cdot(x - x_2)}{(x_0 - x_1)\cdot(x_0 - x_2)} = \frac{(x-1)\cdot(x-2)}{(0-1)\cdot(0-2)} = \frac{1}{2}\cdot(x^2 - 3x + 2)$$

$$L_1(x) = \frac{(x - x_0)\cdot(x - x_2)}{(x_1 - x_0)\cdot(x_1 - x_2)} = \frac{(x-0)\cdot(x-2)}{(1-0)\cdot(1-2)} = -x^2 + 2x$$

$$L_2(x) = \frac{(x - x_0)\cdot(x - x_1)}{(x_1 - x_0)\cdot(x_2 - x_1)} = \frac{(x-0)\cdot(x-2)}{(1-0)\cdot(1-2)} = \frac{1}{2}\cdot(x^2 - x)$$

$$\begin{aligned} P(x) &= \frac{1}{2}(x^2 - 3x + 2)\cdot 1 + (2x - x^2)\cdot 0.5 + \frac{1}{2}(x^2 - x)\cdot 0.2 \\ &= 0.1x^2 - 0.6x + 1 \end{aligned}$$

Examples

Cubic Lagrange polynomials with equidistant Nodes $t_i = \frac{i}{3}$

$$L_0(t) = \left(\frac{t - t_1}{t_0 - t_1}\right)\left(\frac{t - t_2}{t_0 - t_2}\right)\left(\frac{t - t_3}{t_0 - t_3}\right) = \left(\frac{t - \frac{1}{3}}{-\frac{1}{3}}\right)\left(\frac{t - \frac{2}{3}}{-\frac{2}{3}}\right)\left(\frac{t - 1}{-1}\right)$$

$$= \frac{t^3 - 2t^2 + \frac{11}{9}t - \frac{2}{9}}{-\frac{2}{9}} = -\frac{9}{2}t^3 + 9t^2 - \frac{11}{2}t + 1$$

$$L_1(t) = \left(\frac{t - t_0}{t_1 - t_0}\right)\left(\frac{t - t_2}{t_1 - t_2}\right)\left(\frac{t - t_3}{t_1 - t_3}\right) = \left(\frac{t}{\frac{1}{3}}\right)\left(\frac{t - \frac{2}{3}}{-\frac{1}{3}}\right)\left(\frac{t - 1}{-\frac{2}{3}}\right)$$

$$= \frac{t^3 - \frac{5}{3}t^2 + \frac{2}{3}t}{\frac{2}{27}} = \frac{27}{2}t^3 - \frac{45}{2}t^2 + 9t$$

Examples

Cubic Lagrange polynomials with equidistant Nodes $t_i = \frac{i}{3}$

$$L_2(t) = L_1(1-t) = -\frac{27}{2}t^3 + 18t^2 - \frac{9}{2}t$$
$$L_3(t) = L_0(1-t) = \frac{9}{2}t^3 - \frac{9}{2}t^2 + t$$

Drawbacks

The Lagrange-Interpolation has severe drawbacks:

- The calculation of the Lagrange polynomials is not efficient (in this form).
- Adding additional points leads to recalculating everything.

Newton Interpolation

- The Newton form adresses the second aspect.
- The Newton base is given by

$$N_i(t) = \prod_{k=0}^{i-1} (t - t_k) \tag{7}$$

$$\text{with} \quad N_i(t_j) = 0; \quad i > j$$
$$\text{and} \quad N_i(t_i) \neq 0.$$

The coefficients a_i of

$$p(t) = \sum_{i=0}^{n} a_i N_i(t) \tag{8}$$

are determined recursively by the so called **divided differences** $f[t_j, \ldots, t_{j+k}]$:

$$f[t_j] := f_j \quad (j = 0, \ldots, n)$$

$$f[t_j, \ldots, t_{j+k}] := \frac{f\,[t_{j+1}, \ldots, t_{j+k}] - f\,[t_j, \ldots, t_{j+k-1}]}{t_{j+k} - t_j}$$

$$a_i = f\,[t_0, \ldots, t_i]$$

k	0	1	2	3
t_0	$f_0 = f[t_0]$			
	$= a_0$			
		$f[t_0, t_1] = a_1$		
t_1	$f_1 = f[t_1]$		$f[t_0, t_1, t_2] = a_2$	
		$f[t_1, t_2]$		$f[t_0, .., t_3] = a_3$
t_2	$f_2 = f[t_2]$		$f[t_1, t_2, t_3]$	$\dots$
		$f[t_2, t_3]$	$\dots$	
t_3	$f_3 = f[t_3]$	$\dots$		
$\vdots$	$\vdots$			

- The coefficients a_i can also be determined by the solution of a linear system of equations (see (3)).
- In this case, the matrix becomes a lower triangular matrix and the forward substition corresponds with the scheme of the divided differences.

Example: $(t_i, f_i) \in \{(0,1);(2,3);(4,5)\}$

t_i	f_i		
0	$1 = a_0$		
		$1 = a_1$	
2	3		$0 = a_2$
		1	
4	5		

$$p(x) = a_0 + a_1(x - t_0) + a_2(x - t_0)(x - t_1) = 1 + x$$

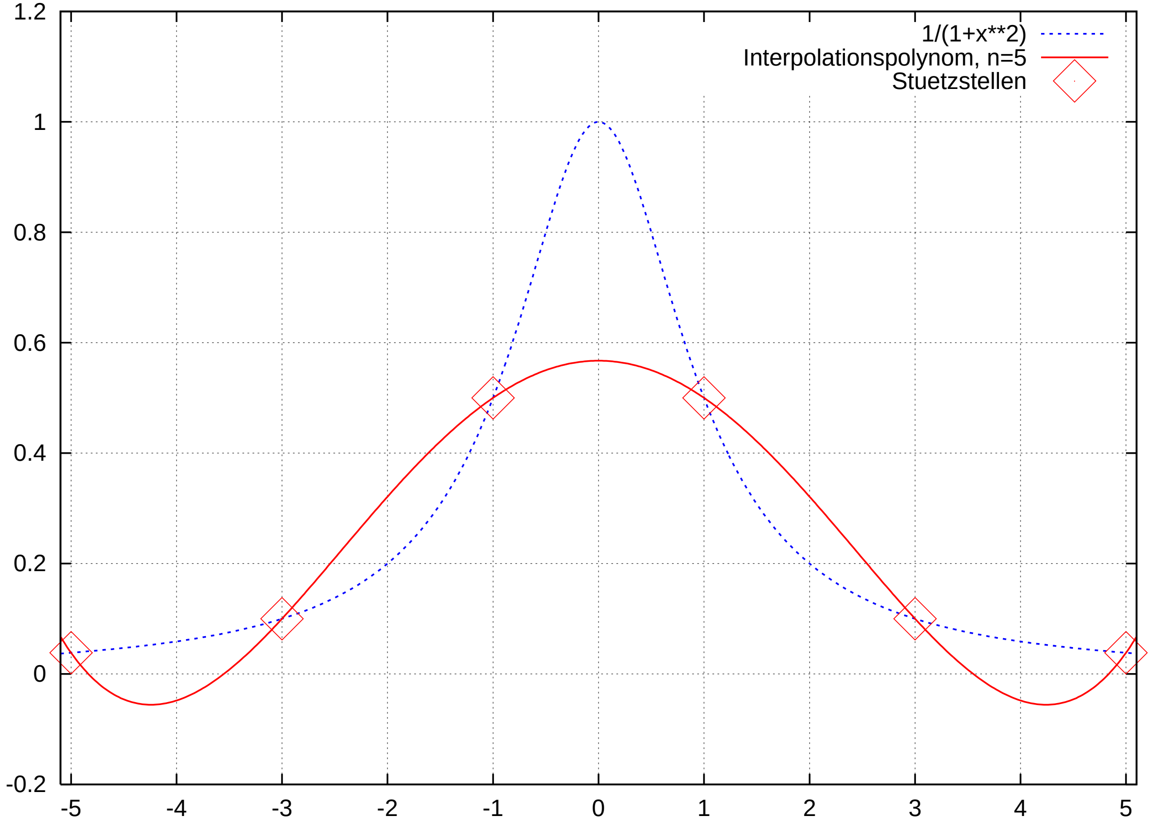
Caution! The interpolation polynomial for $n+1$ given data points has not necessarily a degree of n but **at most** a degree of n.

Remarks:

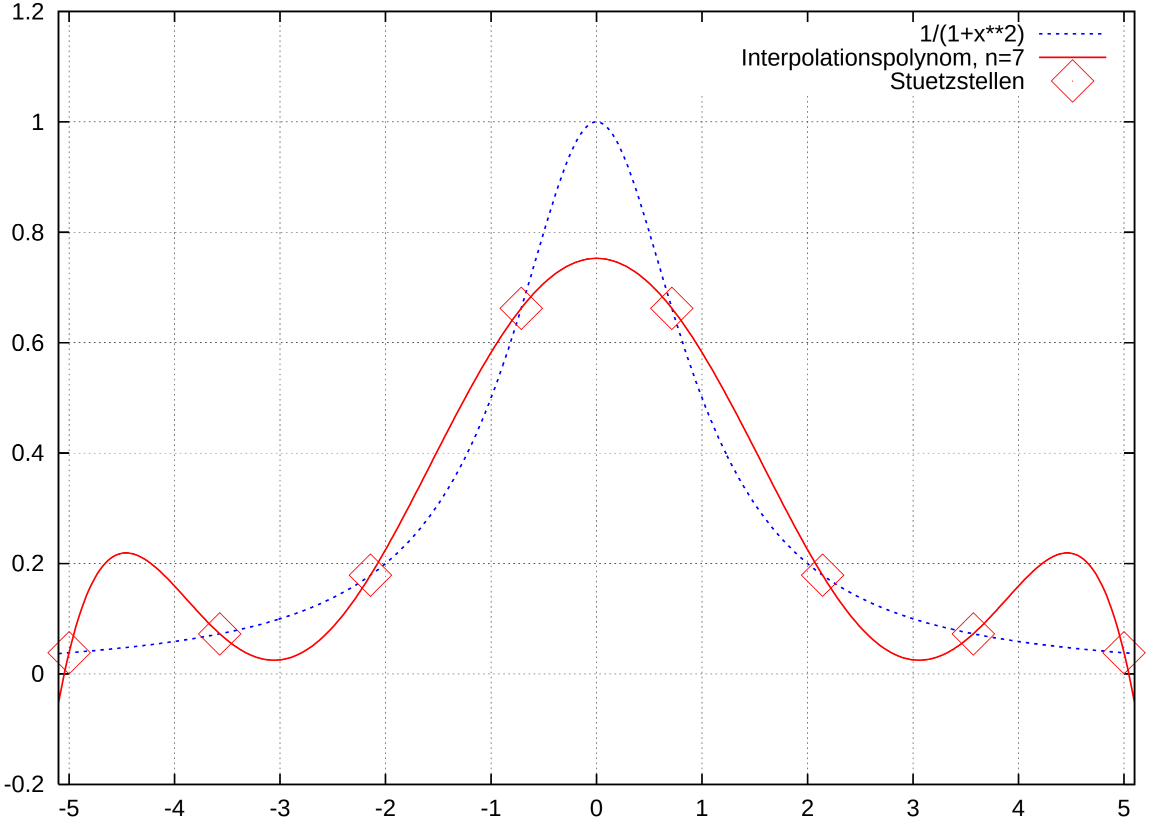
- For the Newton Interpolation, the input order of the nodes is irrelevant.
- Given a continuous function f on the interval $[a, b]$. When interpolating f in n data points, the resulting sequence (dt. Folge) of interpolation polynomials on $[a, b]$ does not necessarily converge to f for $n \to \infty$.

$\rightarrow$ Using more input data points does not necessarily result in better quality!

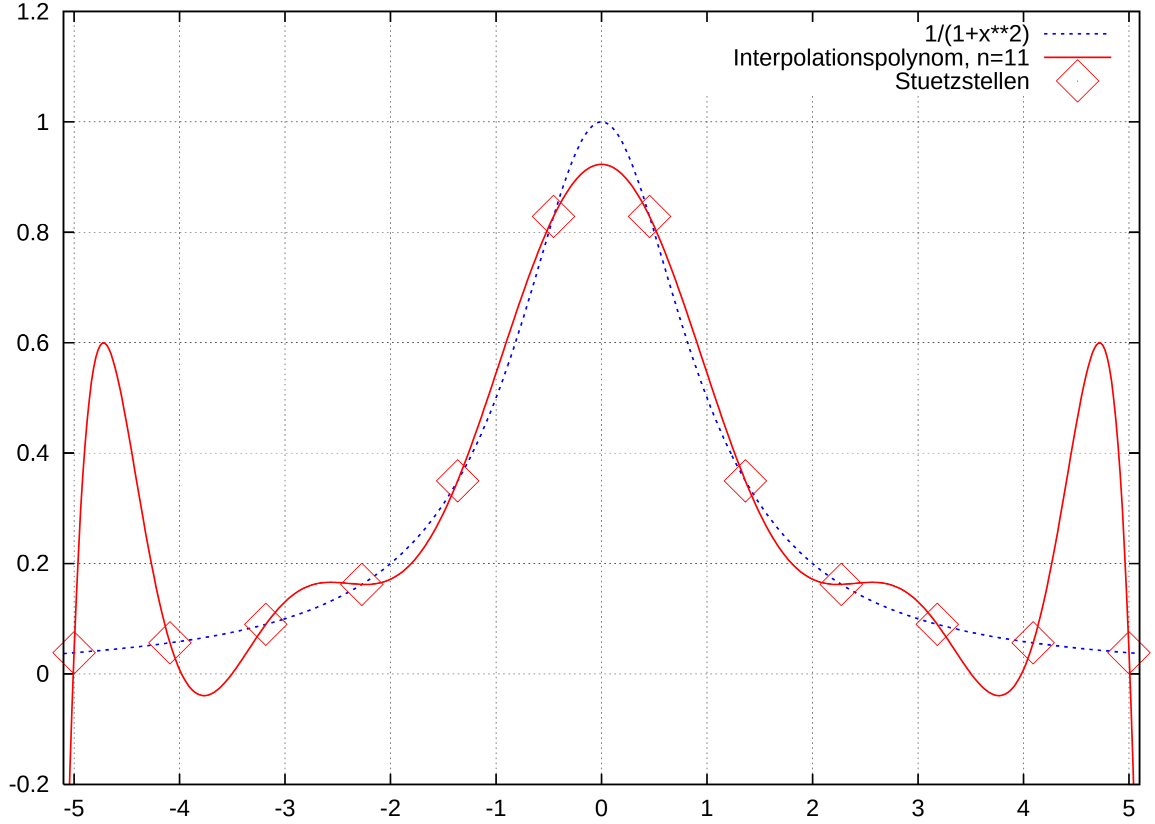
Example: Runge-Funktion: $\frac{1}{1+x^2}$ (degree $n = 5$)



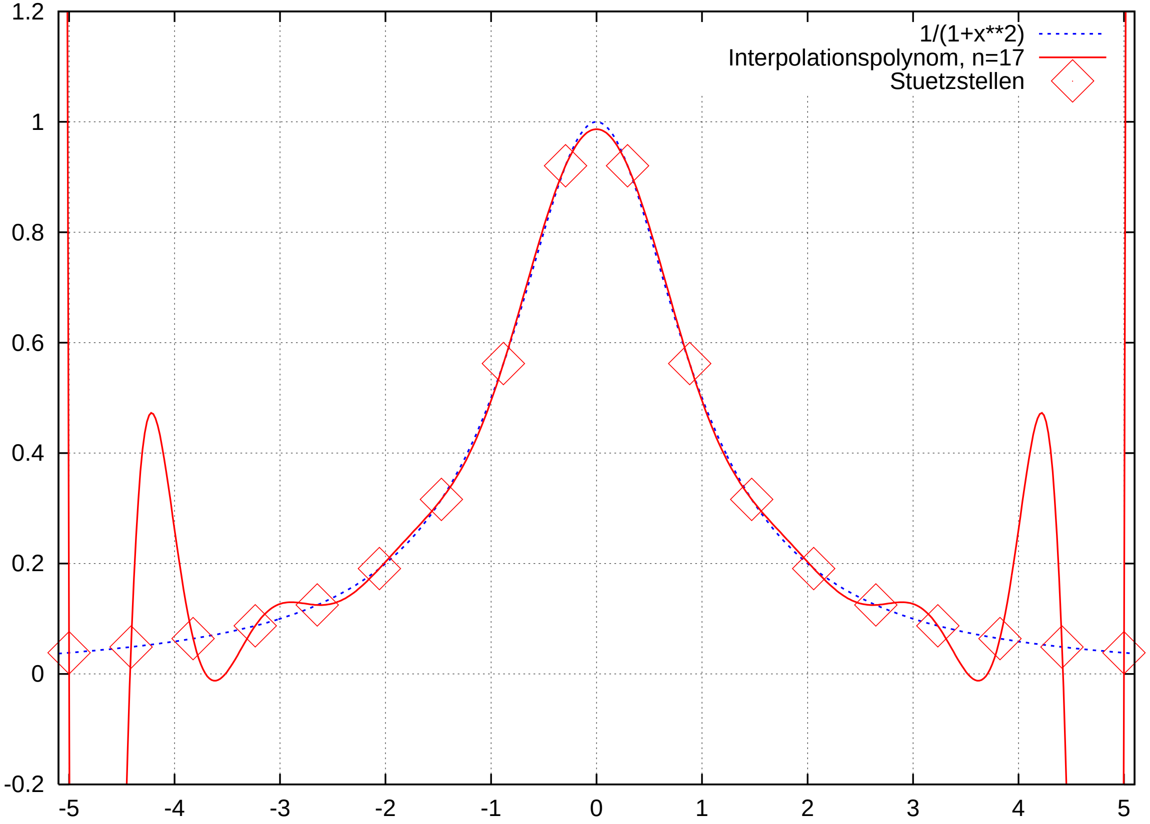
Example: Runge-Funktion: $\frac{1}{1+x^2}$ (degree $n = 7$)



Example: Runge-Funktion: $\frac{1}{1+x^2}$ (degree $n = 11$)

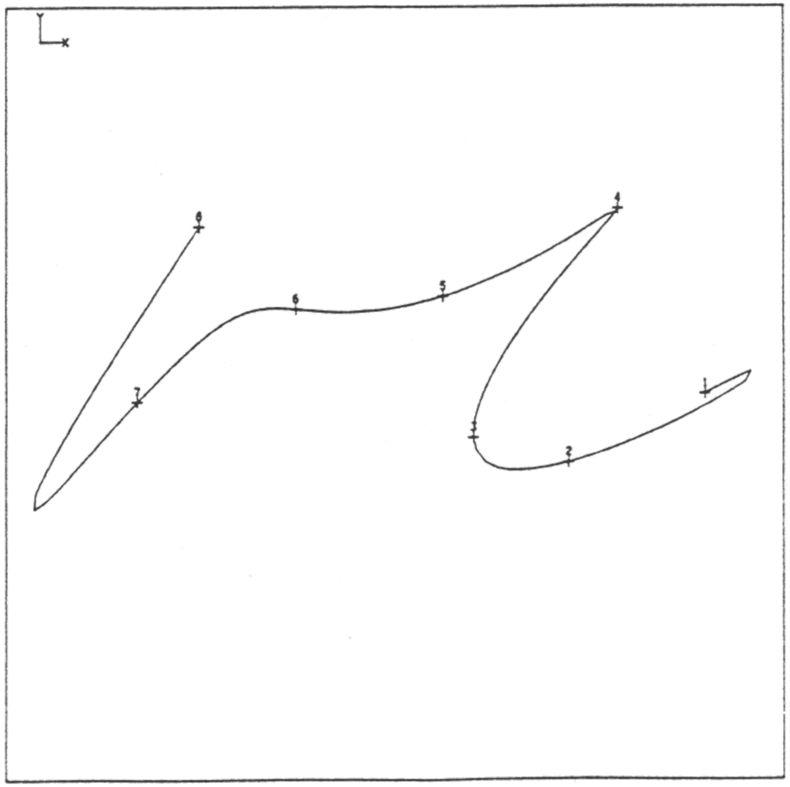


Example: Runge-Funktion: $\frac{1}{1+x^2}$ (degree $n = 17$)



Disadvantages of polynomial interpolation with respect to CAD/CAM-technology

1. Each data point influences the curve **globally**.
   $\Longrightarrow$ Rather use basis functions with local support (dt. Träger)
2. The parametrization (choice of nodes) is of great importance for the quality of the curve.
3. Interpolation polynomials with a degree of $n \geq 5$ produce uneven results.
   $\Longrightarrow$ Introduce additional constraints like "minimal bending energy"



4. With higher degrees, the interpolation polynomial becomes less smooth, especially at the borders of the parameter interval.
$\Longrightarrow$ Introduce border constraints

Interpolation of derivatives

Given: Disjoint nodes t_i, $i = 0, \ldots, n$ and for each i one interpolation value and $n_i - 1$ derivative values ($n_i \geq 1$):

$$f_i =: f_i^{(0)}, f_i^{(1)}, \ldots, f_i^{(n_i-1)}; \quad (i = 0, \ldots, n)$$

Find: Polynomial p with a degree $\leq m := \left(\sum_{i=0}^{n} n_i\right) - 1$,

$$p(t) = \sum_{j=0}^{m} c_j t^j \tag{9}$$

so that

$$p^{(j)}(t_i) = f_i^{(j)} \quad (i = 0, \ldots, n;\ j = 0, \ldots, n_i - 1). \tag{10}$$

Applying (9) to (10) delivers a linear system of equations.

Theorem

The System (10) has a unique solution.

Proof:

- There are $m + 1$ interpolation constraints and $m + 1$ coefficients.
- The System is non-singular, if the homogeneous system $f_i^{(j)} = 0$ has only the trivial solution.
- Because p has exactly $m + 1$ roots (dt. Nullstellen) including multiplicities and has a degree $\leq m$, p must be the zero polynomial. $\square$

Example: Find the cubic polynomial, which interpolates $f(0), f'(0), f(1)$ and $f'(1)$

$$\begin{aligned} \text{With} \quad p(t) &= c_3 t^3 + c_2 t^2 + c_1 t + c_0 \\ \text{and} \quad p'(t) &= 3c_3 t^2 + 2c_2 t + c_1 \end{aligned}$$

$$\begin{aligned} \text{follows} \quad f(0) &= c_0 \\ f'(0) &= c_1 \\ f(1) &= c_3 + c_2 + c_1 + c_0 \\ f'(1) &= 3c_3 + 2c_2 + c_1 \end{aligned}$$

In matrix form:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} f(0) \\ f'(0) \\ f(1) \\ f'(1) \end{pmatrix} \tag{11}$$

Hermite Interpolation

Analogous to the Lagrange Interpolation, one can create basis polynomials that are well suited for the interpolation of the derivatives.

These are called **Hermite Polynomials**.

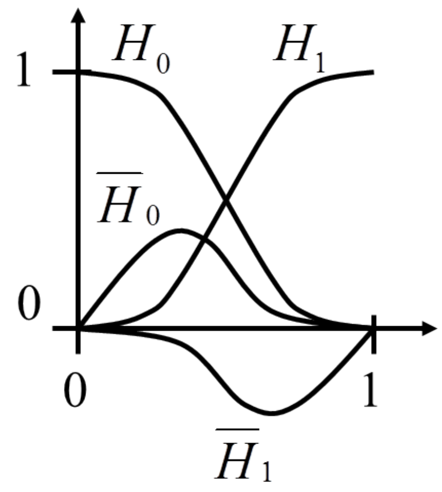
Example

Find: Cubic Hermite polynomials for the system (11)

Solution: The coefficients are the column vectors of the inverse matrix:

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -3 & -2 & 3 & -1 \\ 2 & 1 & -2 & 1 \end{pmatrix} \begin{pmatrix} f(0) \\ f'(0) \\ f(1) \\ f'(1) \end{pmatrix}$$

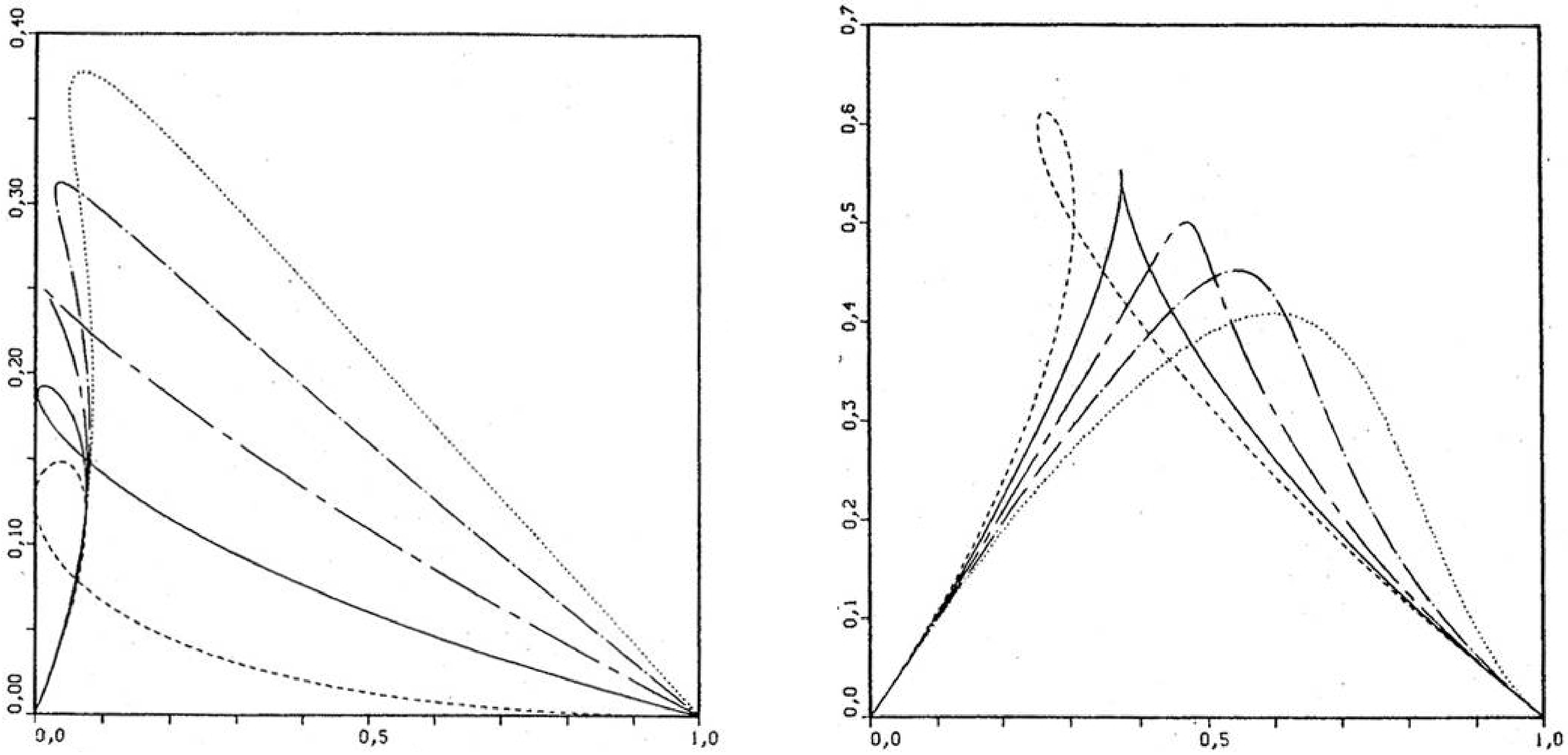
$$\Longrightarrow$$
$$H_0(t) = 2t^3 - 3t^2 + 1$$
$$\bar{H}_0(t) = t^3 - 2t^2 + t$$
$$H_1(t) = -2t^3 + 3t^2$$
$$\bar{H}_1(t) = t^3 - t^2$$



Properties:

$$H_i(j) = \delta_{ij} \quad H_i'(j) = 0$$
$$\bar{H}_i(j) = 0 \quad \bar{H}_i'(j) = \delta_{ij}$$
$$i, j = \{0, 1\}$$

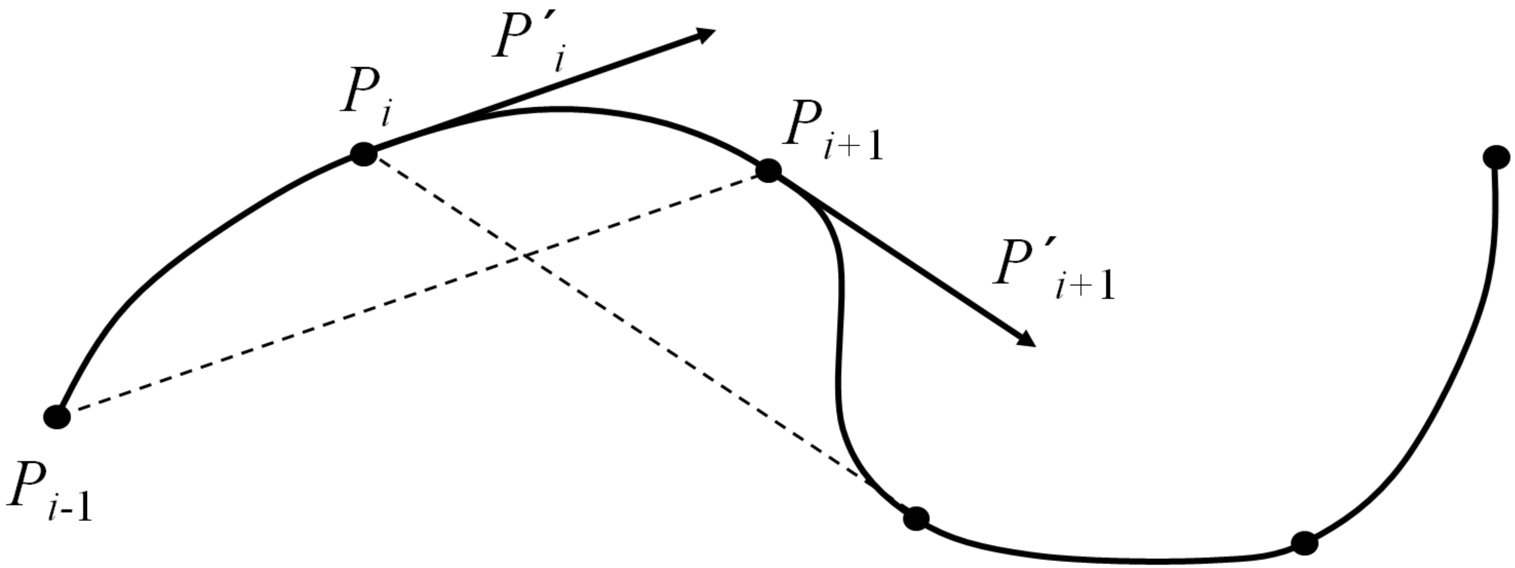
Interpolation



From: Hoschek/Lasser

Catmull-Rom Splines (1974)

- Combine Hermite segments to a C^1 continuous interpolation curve
- The derivatives in the interpolation points P_i are defined by $P_i' = T_i(P_{i+1} - P_{i-1})$ with **tension** T_i
- A common value for T_i is 0.5



Goals

- How to interpolate given points using polynomials?
- How to do Newton-Interpolation?
- How to interpolate given points with derivatives using polynomials?
- Drawbacks of polynomial interpolation
- Advantages / Disadvantages of the different forms (Lagrange, Newton, Hermite, Catmull-Rom)
- Problems using polynomial interpolation