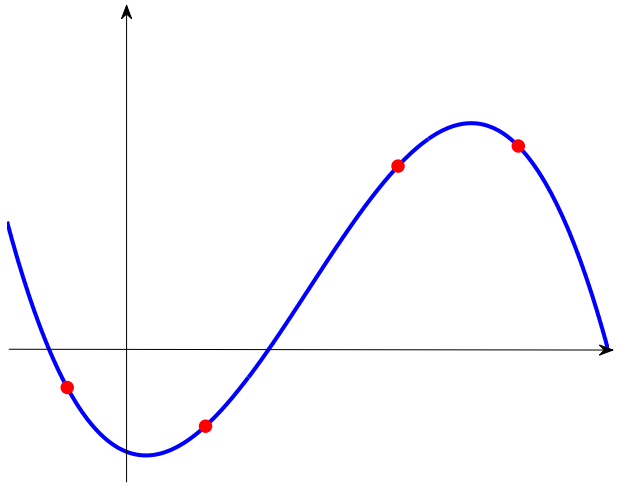
8 Interpolation

8.1 Problemstellung

Gegeben: Diskrete Werte einer Funktion $f : \mathbb{R} \to \mathbb{R}$ an $n+1$ Stützstellen

$$x_0 < x_1 < \ldots < x_n.$$

Eingabedaten: $(x_0, f_0), (x_1, f_1), \cdots, (x_n, f_n)$.

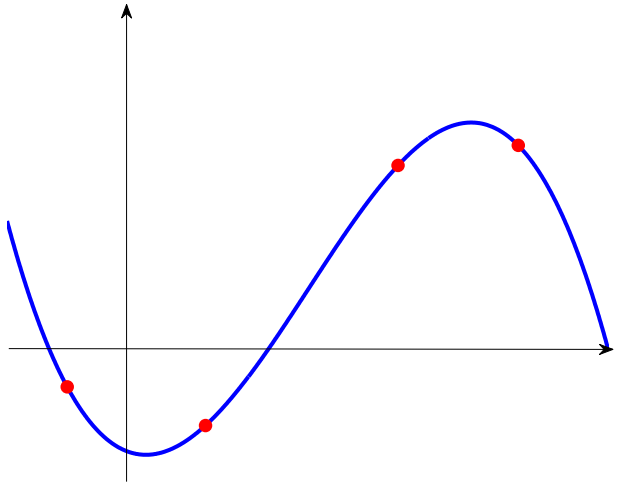


Gegebene Daten (x_j, f_j).

Gesucht: *Einfache* Funktion $p : \mathbb{R} \to \mathbb{R}$, die die Daten `interpoliert`, d.h.

$$p(x_i) = f_i \qquad \text{für alle } i = 0, 1, \ldots, n,$$

z.B.: p Polynom, trigonometrisches Polynom, rationale Funktion.



Gegebene Daten (x_j, f_j)**.**

Fragen:

- Gibt es so ein p? Falls ja, ist p eindeutig?
- Wie sieht die Lösung p aus und wie berechnet man p?

Klassische Polynom-Interpolation.

Bestimme ein Polynom (höchstens) n-ten Grades

$$p_n(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n,$$

das die gegebenen Daten interpoliert, so dass $p_n(x_i) = f_i$, $0 \leq i \leq n$.

Erster Lösungsansatz: Die Interpolationsbedingungen ergeben lineares System

$$\begin{aligned} a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_n x_0^n &= f_0 \\ a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_n x_1^n &= f_1 \\ &\vdots \quad \vdots \\ a_0 + a_1 x_n + a_2 x_n^2 + \ldots + a_n x_n^n &= f_n \end{aligned}$$

Setze:
$X = \{x_0, \ldots, x_n\}$, $f|_X = (f_0, \ldots, f_n)^T \in \mathbb{R}^{n+1}$ und $a = (a_0, \ldots, a_n)^T \in \mathbb{R}^{n+1}$.

Vandermonde-Matrix.

Die Koeffizientenmatrix des linearen Systems

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{bmatrix},$$

kurz

$$V \cdot a = f|_X,$$

heißt Vandermonde-Matrix.

Satz: *Für die Determinante der Vandermonde-Matrix* $V \equiv V(x_0, \dots, x_n)$ *gilt*

$$\det(V) = \prod_{0 \le i < j \le n} (x_j - x_i).$$

Beweis: Durch vollständige Induktion über n.

Induktionsanfang: $n = 1$: $\det(V(x_0, x_1)) = x_1 - x_0$.

Induktionsschritt: $n - 1 \to n$.

$$\det\left(V(x_0, \ldots, x_n)\right)$$

$$= \det \begin{bmatrix} 1 & x_0 & \cdots & x_0^{n-1} & x_0^n \\ 1 & x_1 & \cdots & x_1^{n-1} & x_1^n \\ \vdots & & & \vdots & \\ 1 & x_n & \cdots & x_n^{n-1} & x_n^n \end{bmatrix} = \det \begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 0 & x_1 - x_0 & \cdots & x_1^n - x_0^n \\ \vdots & & & \vdots \\ 0 & x_n - x_0 & \cdots & x_n^n - x_0^n \end{bmatrix}$$

$$= \det \begin{bmatrix} x_1 - x_0 & \cdots & x_1^n - x_0^n \\ \vdots & & \vdots \\ x_n - x_0 & \cdots & x_n^n - x_0^n \end{bmatrix} = \det \begin{bmatrix} x_1 - x_0 & x_1^2 - x_0 x_1 & \cdots & x_1^n - x_0 x_1^{n-1} \\ \vdots & & & \vdots \\ x_n - x_0 & x_n^2 - x_0 x_n & \cdots & x_n^n - x_0 x_n^{n-1} \end{bmatrix}$$

$$= (x_1 - x_0) \cdots (x_n - x_0) \cdot \det\left(V(x_1, \ldots, x_n)\right) = \prod_{0 \le i < j \le n} (x_j - x_i). \qquad \blacksquare$$

Existenz und Eindeutigkeit der Interpolation.

Folgerung: Falls Stützstellen $x_0, \ldots, x_n$ paarweise verschieden, so ist V regulär.

Satz: *Zu paarweise verschiedenen Stützstellen*

$$X = \{x_0, x_1, \ldots, x_n\} \subset \mathbb{R}$$

und Funktionswerten

$$f_0, f_1 \ldots, f_n \in \mathbb{R}$$

gibt es genau ein interpolierendes Polynom p_n vom Höchstgrad n mit

$$p_n(x_i) = f_i \qquad \text{für alle } 0 \leq i \leq n.$$

■

ABER: Wir berechnen die Lösung **nicht** über das lineare System $V \cdot a = f\big|_X$.

DENN: Dies ist zu **teuer** und **instabil**. □

8.2 Interpolationsformeln nach Lagrange und Newton

Lagrange-Darstellung.

Definieren Lagrange-Polynome

$$L_j(x) := \frac{(x-x_0)\cdot\ldots\cdot(x-x_{j-1})\cdot(x-x_{j+1})\cdot\ldots\cdot(x-x_n)}{(x_j-x_0)\cdot\ldots\cdot(x_j-x_{j-1})\cdot(x_j-x_{j+1})\cdot\ldots\cdot(x_j-x_n)}$$

$$= \prod_{\substack{i=0\\ i\neq j}}^{n} \frac{x-x_i}{x_j-x_i} \quad \text{für } 0 \leq j \leq n.$$

Dann ist L_j ein Polynom vom Grad n, und es gilt

$$L_j(x_i) = \begin{cases} 1 & \text{für } i = j \\ 0 & \text{für } i \neq j \end{cases} \qquad \text{für } 0 \leq i,j \leq n.$$

Lösung mit der Lagrange-Darstellung.

Die Interpolationsaufgabe

$$p_n(x_i) = f_i \qquad \text{für alle } 0 \le i \le n$$

wird gelöst durch das (eindeutige) Polynom

$$p_n(x) = f_0 L_0(x) + \ldots + f_n L_n(x) = \sum_{i=0}^{n} f_i L_i(x).$$

Die obige Darstellung von p_n heißt Lagrange-Darstellung. □

Beispiel. Betrachte die Daten

x_j	0	1	2	3
f_j	0	0	4	18

Dann sieht die zugehörige Lagrange-Basis wie folgt aus.

$$L_0(x) = \frac{(x-1)(x-2)(x-3)}{(0-1)(0-2)(0-3)} \qquad L_1(x) = \frac{(x-0)(x-2)(x-3)}{(1-0)(1-2)(1-3)}$$

$$L_2(x) = \frac{(x-0)(x-1)(x-3)}{(2-0)(2-1)(2-3)} \qquad L_3(x) = \frac{(x-0)(x-1)(x-2)}{(3-0)(3-1)(3-2)}$$

Das interpolierende kubische Polynom p_3 besitzt die Darstellungen

$$\begin{aligned} p_3(x) &= 4 \cdot L_2(x) + 18 \cdot L_3(x) \\ &= -4 \cdot \frac{x(x-1)(x-3)}{2} + 18 \cdot \frac{x(x-1)(x-2)}{6} \\ &= x^3 - x^2. \end{aligned}$$

Auswertung von Interpolationspolynomen.

Für $0 \le j \le k \le n$ bezeichne p_{kj} das eindeutige Interpolationspolynom vom Höchstgrad j zu den Daten

$$(x_{k-j}, f_{k-j}), \ldots, (x_k, f_k),$$

d.h. es gilt

$$p_{kj}(x_\ell) = f_\ell, \qquad \text{für alle } k-j \le \ell \le k.$$

Dann lassen sich für ein festes $x \in \mathbb{R}$ die Werte $p_{kj}(x)$ rekursiv berechnen.

Lemma (Aitken): *Es gilt die Rekursion*

$$\begin{aligned} p_{k0}(x) &= f_k \\ p_{kj}(x) &= p_{k,j-1}(x) + \frac{x - x_k}{x_{k-j} - x_k}(p_{k-1,j-1}(x) - p_{k,j-1}(x)) \quad \textit{für } j \ge 0. \end{aligned}$$

Beweis: Vollständige Induktion über j.

Induktionsanfang: Für $j = 0$ ist p_{k0} konstant mit $p_{k0} \equiv f_k$.

Induktionsschritt: $j - 1 \to j$: Die rechte Seite der Rekursion,

$$q(x) = p_{k,j-1}(x) + \frac{x - x_k}{x_{k-j} - x_k}(p_{k-1,j-1}(x) - p_{k,j-1}(x))$$

ist ein Polynom vom Höchstgrad j.

Weiterhin interpoliert $p_{k,j-1}$ nach Induktionsvoraussetzung die Daten

$$(x_{k-j+1}, f_{k-j+1}), \ldots, (x_k, f_k),$$

und $p_{k-1,j-1}$ interpoliert die Daten

$$(x_{k-j}, f_{k-j}), \ldots, (x_{k-1}, f_{k-1}).$$

Daraus folgt mit der Rekursion, dass das Polynom q die Daten

$$(x_{k-j}, f_{k-j}), \ldots, (x_k, f_k)$$

interpoliert, genauso wie p_{kj}. Wegen der Eindeutigkeit gilt $q \equiv p_{kj}$. ■

Algorithmus von Neville-Aitken.

Ziel: Rekursive Berechnung von $p_{nn}(x)$ für $x \in \mathbb{R}$ in Dreiecksschema:

$$\begin{array}{lllll} p_{00}(x) & & & & \\ p_{10}(x) & p_{11}(x) & & & \\ p_{20}(x) & p_{21}(x) & p_{22}(x) & & \\ \vdots & \vdots & \vdots & \ddots & \\ p_{n0}(x) & p_{n1}(x) & p_{n2}(x) & \cdots & p_{nn}(x) \end{array}$$

Besser: Effiziente Auswertung in Datenvektor $f = (f_0, f_1, \ldots, f_n)$:

$$\begin{array}{lllll} f_0 = f_0(x) & & & & \\ f_1 = f_1(x) & f_0(x) & & & \\ f_2 = f_2(x) & f_1(x) & f_0(x) & & \\ \vdots & \vdots & \vdots & \ddots & \\ f_n = f_n(x) & f_{n-1}(x) & f_{n-2}(x) & \cdots & f_0(x) \end{array}$$

Implementierung als Matlab-Funktion.

INPUT:

```
x = (x(1),x(2),...,x(n))  % Stuetzstellen
f = (f(1),f(2),...,f(n))  % Funktionswerte
y                         % Auswertungsstelle
```

```
function p = neville(x,f,y)
n = length(x);
for k=2:n
  z = y-x(k);
  for i=k-1:-1:1
    f(i) = f(i+1) + z/(x(i)-x(k))*(f(i)-f(i+1));
  end;
end;
p = f(1);
```

Newton-Darstellung. Betrachte die Newton-Basis

$$\omega_i(x) = \prod_{j=0}^{i-1}(x - x_j), \qquad \text{für } 0 \le i \le n.$$

Dann gibt es *eindeutige* Newton-Koeffizienten $c_0, c_1, \ldots, c_n \in \mathbb{R}$ mit

$$\begin{aligned} p_n(x) &= \sum_{i=0}^{n} c_i \omega_i(x) \\ &= c_0 + c_1(x - x_0) + \ldots + c_n(x - x_0)\cdots(x - x_{n-1}). \end{aligned}$$

Die obige Darstellung von p_n heißt Newton-Darstellung.

Beachte: Es gilt:

$$\begin{aligned} p_n(x_0) &= c_0 \\ p_n(x_1) &= c_0 + c_1(x_1 - x_0) \\ p_n(x_2) &= c_0 + c_1(x_2 - x_0) + c_2(x_2 - x_0)(x_2 - x_1) \\ &\vdots \qquad\qquad \vdots \end{aligned}$$

Berechnung der Newton-Koeffizienten.

Beachte: Aus den Interpolationsbedingungen folgt

$$
\begin{aligned}
p_n(x_0) &= c_0 \stackrel{!}{=} f_0 \quad \Longrightarrow \quad c_0 = f_0 \\
p_n(x_1) &= c_0 + c_1(x_1 - x_0) \stackrel{!}{=} f_1 \quad \Longrightarrow \quad c_1 = \frac{f_1 - f_0}{x_1 - x_0} \\
&\vdots \quad = \quad \vdots \\
p_n(x_n) &= \sum_{i=0}^{n} c_i \prod_{j=0}^{i-1} (x_n - x_j) \\
&= c_0 + c_1(x_n - x_0) + \ldots + c_n(x_n - x_0)(x_n - x_1) \cdots (x_n - x_{n-1}) \\
&\stackrel{!}{=} f_n
\end{aligned}
$$

mit

$$
c_n = \left(f_n - \sum_{j=0}^{n-1} c_j \prod_{i=0}^{j-1} (x_n - x_i) \right) \Big/ \prod_{j=0}^{n-1} (x_n - x_j).
$$

Beobachtungen.

- Zur Berechnung von c_j benötigt man nur die ersten $(j+1)$ Daten

$$(x_0, f_0), (x_1, f_1), \ldots, (x_j, f_j).$$

Notation:

$$c_j = f[x_0, x_1, \ldots, x_{j-1}, x_j] \qquad \text{für } j = 0, 1, \ldots, n$$

- Nimmt man ein Datum (x_{n+1}, f_{n+1}) hinzu, so gilt:

$$p_{n+1}(x) = p_n(x) + c_{n+1} \prod_{i=0}^{n} (x - x_i)$$

mit

$$c_{n+1} = (f_{n+1} - p_n(x_{n+1})) \Big/ \prod_{j=0}^{n} (x_{n+1} - x_j).$$

□

Dividierte Differenzen.

Satz: *Die Koeffizienten*

$$c_j = f[x_0, x_1, \ldots, x_j], \qquad 0 \le j \le n,$$

des interpolierenden Newton-Polynoms

$$p_n(x) = \sum_{i=0}^{n} c_j \omega_j(x)$$

sind gegeben durch die dividierten Differenzen

$$\begin{aligned} f[x_j] &= f_j \\ f[x_i, x_{i+1}, \ldots, x_{i+k}] &= \frac{f[x_{i+1}, \ldots, x_{i+k}] - f[x_i, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}. \end{aligned}$$

Beweis: Mit Aitken-Lemma. □

Effiziente Berechnung der dividierten Differenzen.

Beispiel: Rekursives Berechnungsschema der dividierten Differenzen für $n = 3$.

x	$f[x_i]$	$f[x_i, x_{i+1}]$	$f[x_i, x_{i+1}, x_{i+2}]$	$f[x_i, x_{i+1}, x_{i+2}, x_{i+3}]$
x_0	f_0			
x_1	f_1	$f[x_0, x_1]$		
x_2	f_2	$f[x_1, x_2]$	$f[x_0, x_1, x_2]$	
x_3	f_3	$f[x_2, x_3]$	$f[x_1, x_2, x_3]$	$f[x_0, x_1, x_2, x_3]$

Zum Beispiel:

$$f[x_0, x_1] = \frac{f[x_1] - f[x_0]}{x_1 - x_0} = \frac{f_1 - f_0}{x_1 - x_0}$$

$$f[x_1, x_2, x_3] = \frac{f[x_2, x_3] - f[x_1, x_2]}{x_3 - x_1} = \frac{1}{x_3 - x_1}\left(\frac{f_3 - f_2}{x_3 - x_2} - \frac{f_2 - f_1}{x_2 - x_1}\right)$$

Der Interpolationsfehler. Für das Interpolationspolynom gilt

$$\begin{aligned}\varepsilon(x) &= f(x) - p_n(x)\\ &= f(x) - \left(p_{n+1}(x) - c_{n+1}\prod_{i=0}^{n}(x-x_i)\right)\\ &= f[x_0,\ldots,x_n,x]\prod_{i=0}^{n}(x-x_i).\end{aligned}$$

Satz: *Sei* $f \in C^{n+1}([a,b])$. *Dann gibt es ein* $\xi \in [a,b]$ *mit*

$$f[x_0,\ldots,x_{n+1}] = \frac{1}{(n+1)!}f^{(n+1)}(\xi).$$

Folgerung: *Für den* Interpolationsfehler *gilt die Abschätzung*

$$|f(x) - p_n(x)| \le \frac{1}{(n+1)!}\max_{\xi\in[a,b]}|f^{(n+1)}(\xi)| \cdot \left|\prod_{i=0}^{n}(x-x_i)\right|.$$

Tschebyscheff-Knoten.

Beachte: Ein Term des Interpolationsfehlers ist das Knotenpolynom

$$\omega_{n+1}(x) = \prod_{i=0}^{n}(x - x_i)$$

Optimierungsproblem: Bestimme die Knoten $x_0, x_1, \ldots, x_n$, so dass

$$\max_{x_0,\ldots,x_n \in [a,b]} \left| \prod_{i=0}^{n}(x - x_i) \right|$$

minimal auf $[a, b]$.

Lösung: Für das Intervall $[-1, 1]$ sind die Tschebyscheff-Knoten optimal.

$$x_j = \cos\left(\frac{2j+1}{2n+2}\,\pi\right) \qquad j = 0, 1, \ldots, n.$$