Einführung in die Numerik WS2018/19

Dozent: Prof. Dr. Andreas Fischer

16. Oktober 2018

Inhaltsverzeichnis

Vorwort

Kapitel I

Interpolation

1. Grundlagen

Aufgabe:
Gegeben sind $n+1$ Datenpaare $(x_0, f_0), \ldots, (x_n, f_n)$, alles reelle Zahlen und paarweise verschieden. Gesucht ist eine Funktion $F : \mathbb{R} \to \mathbb{R}$, die die Interpolationsbedingungen

$$F(x_0) = f_0, \; \ldots, \; F(x_n) = f_n \tag{1}$$

genügt.

Definition (Stützstellen, Stützwerte)
Die x_0 bis x_n werden Stützstellen genannt.

Die f_0 bis f_n werden Stützwerte genannt.

Die oben gestellte Aufgabe wird zum Beispiel durch

$$F(x) = \begin{cases} 0 & x \notin \{x_0, \ldots, x_n\} \\ f_i & x = x_i \end{cases}$$

gelöst. Weitere Möglichkeiten sind: Polygonzug, Treppenfunktion, Polynom, ...

- In welcher Menge von Funktionen soll F liegen?
- Gibt es im gewählten Funktionenraum für beliebige Datenpaare eine Funktion F, die den Interpolationsbedingungen genügt (eine solche Funktion heißt Interpolierende)?
- Ist die Interpolierende in diesem Raum eindeutig bestimmt?
- Welche weiteren Eigenschaften besitzt die Interpolierende, zum Beispiel hinsichtlich ihrer Krümmung oder der Approximation einer Funktion $f : \mathbb{R} \to \mathbb{R}$ mit $f_k = f(x_k)$ für $k = 0, \ldots, n$
- Wie sollte man die Stützstellen wählen, falls nicht vorgegeben?
- Wie lässt sich die Interpolierende effizient bestimmen, gegebenenfalls auch unter der Berücksichtigung, dass neue Datenpaare hinzukommen oder dass sich nur die Stützwerte ändern?

■ **Beispiel 1.1**

k	0	1	2	3	4	5
x_k in s	0	1	2	3	4	5
f_k in °C	80	85,8	86,4	93,6	98,3	99,1

Interpolation im

- Raum der stetigen stückweise affinen Funktionen
- Raum der Polynome höchstens 5. Grades
- Raum der Polynome höchstens 4. Grades (Interpolation im Allgemeinen nicht lösbar, Regression nötig)

2. Interpolation durch Polynome

Π_n bezeichne den Vektorraum der Polynome von Höchstgrad n mit der üblichen Addition und Skalarmultiplikation. Für jedes $p \in \Pi_n$ gibt es $a_0, \dots, a_n \in \mathbb{R}$, sodass

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 \tag{2}$$

und umgekehrt.

2.1. Existenz und Eindeutigkeit

Satz 2.1
Zu $n+1$ Datenpaaren $(x_0, f_0), \dots, (x_n, f_n)$ mit paarweise verschiedenen Stützstellen existiert genau ein Polynom $p \in \Pi_n$, dass die Interpolationsbedingung Gleichung (1) erfüllt.

Beweis.
- Existenz: Sei $j \in \{0, \dots, n\}$ und $L_j : \mathbb{R} \to \mathbb{R}$ mit

$$L_j(x) := \prod_{\substack{i=0\\ i\neq j}}^{n} \frac{x - x_i}{x_j - x_i} = \frac{(x - x_0) \cdot \dots \cdot (x - x_{j-1})(x - x_{j+1}) \cdot \dots \cdot (x - x_n)}{(x_j - x_0) \cdot \dots \cdot (x_j - x_{j-1})(x_j - x_{j+1}) \cdot \dots \cdot (x_j - x_n)}$$

das Lagrange-Basispolynom vom Grad n. Offenbar gilt $L_j \in \Pi_n$ und

$$L_j(x_k) = \begin{cases} 1 & k = j \\ 0 & k \neq j \end{cases} = \delta_{jk} \tag{3}$$

Definiert man $p : \mathbb{R} \to \mathbb{R}$ durch

$$p(x) := \sum_{j=0}^{n} f_j \cdot L_j(x) \tag{4}$$

so ist $p \in \Pi_n$ und außerdem erfüllt p wegen Gleichung (3) die Interpolationsbedingung Gleichung (1)

- Eindeutigkeit: Angenommen es gibt Interpolierende $p, \tilde{p} \in \Pi_n$ mit $p \neq \tilde{p}$. Dann folgt $p - \tilde{p} \in \Pi_n$ und $(p - \tilde{p})(x_k) = p(x_k) - \tilde{p}(x_k) = 0$ für $k = 0, \dots, n$. Also hat $(p - \tilde{p})$ mindestens $n + 1$ Nullstellen, hat aber Grad n. Das heißt, dass $(p - \tilde{p})$ das Nullpolynom sein muss. □

Definition (Interpolationspolynom)
Das Polynom, dass die Interpolationsbedingung erfüllt, heißt Interpolationspolynom zu $(x_0, f_0), \dots, (x_n, f_n)$.

▶ Bemerkung 2.2

- Die Darstellung Gleichung (4) heißt Lagrange-Form des Interpolationspolynoms.
- Um mittels Gleichung (4) einen Funktionswert $p(x)$ zu berechnen, werden $\mathcal{O}(n^2)$ Operationen genötigt; bei gleichabständigen Stützstellen kann man diesen Aufwand auf $\mathcal{O}(n)$ verringern. Ändern sich die Stützwerte, kann man durch Wiederverwendung von den $L_j(x)$ das $p(x)$ in $\mathcal{O}(n)$ Operationen berechnen.
- Man kann zeigen, dass L_0 bis L_n eine Basis von Π_n bilden.

2.2. Newton-Form des Interpolationspolynoms

$$p(x) = c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \cdots + c_n(x - x_0)\dots(x - x_{n-1}) \tag{5}$$

mit Koeffizienten $c_0, \dots, c_n \in \mathbb{R}$. Die Berechnung des Koeffizienten c_j kann rekursiv durch Ausnutzen der Interpolationsbedingung Gleichung (1) erfolgen. Für c_0 erhält man

$$f_0 \overset{!}{=} p(x_0) = c_0$$

Seien c_0 bis c_{j-1} bereits ermittelt. Dann folgt:

$$f_j \overset{!}{=} p(x_j) = \underbrace{c_0 + \sum_{k=1}^{j-1} c_k(x_j - x_0)\dots(x_j - x_{k-1})}_{\text{bekannt}} + c_j \underbrace{(x_j - x_0)\dots(x_j - x_{j-1})}_{\text{bekannt}}$$

▶ **Bemerkung 2.3**

- Der Aufwand um die Koeffizienten $c_0, \dots, c_n$ zu ermitteln ist $\mathcal{O}(n^2)$. Kommt ein Datenpaar hinzu, kann man Gleichung (5) um einen Summanden erweitern und mit $\mathcal{O}(n)$ Operationen c_{n+1} bestimmen.
- Sind die Koeffizienten $c_0, \dots, c_n$ in Gleichung (5) bekannt, dann benötigt man zur Berechnung von $p(x)$ $\mathcal{O}(n)$ Operationen.
- Die Polynome $N_0, \dots, N_n : \mathbb{R} \to \mathbb{R}$ mit

  $$N_0 = 1 \quad \text{und} \quad N_j = (x - x_0)\dots(x - x_{j-1})$$

  heißen Newton-Basispolynome und bilden eine Basis von Π_n.

Die Koeffizienten $c_0, \dots, c_n$ ergeben sich wegen Gleichung (2) auch als Lösung des folgenden linearen Gleichungssystems:

$$\begin{pmatrix} 1 & & & & \\ 1 & (x_1 - x_0) & & & \\ 1 & (x_2 - x_0) & (x_2 - x_0)(x_2 - x_1) & & \\ \vdots & \vdots & \vdots & \ddots & \\ 1 & (x_n - x_0) & (x_n - x_0)(x_n - x_1) & \dots & \prod_{i=0}^{n-1}(x_n - x_i) \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_n \end{pmatrix}$$

Die Systemmatrix dieses linearen Gleichungssystems ist eine reguläre untere Dreiecksmatrix.

Zu effizienten Berechnung eines Funktionswertes $p(x)$ nach Gleichung (5) mit gegebenen Koeffizienten

$c_0, \dots, c_n$ kann man das HORNER-Schema anwenden. Überlegung für $n = 3$.

$$\begin{aligned} p(x) &= c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + c_3(x - x_0)(x - x_1)(x - x_2) \\ &= c_0 + (x - x_0)\Big[c_1 + (x - x_1)\left[c_2 + (x - x_2)c_3\right]\Big] \end{aligned}$$

Für beliebiges n liefert das den folgenden Algorithmus:

■ **Algorithmus 2.4 (Horner-Schema für Newton-Form)**

Input: n, x, c_0,..., c_n, x_0,..., x_n

```
p = c_n
do j = n-1, 0, -1
 p = c_j + (x - x_j)p
end do
```

2.3. Interpolationsfehler

Definition (Maximum-Norm)

Die Norm

$$\|g\|_\infty := \max_{x \in [a,b]} |g(x)| \quad \text{für } g \in C[a,b]$$

definiert die Maximum-Norm in $C[a, b]$.

Satz 2.5

Sei $f \in C[a, b]$. Dann existiert zu jedem $\varepsilon > 0$ ein Polynom p_ε mit $\|f - p_\varepsilon\| \leq \varepsilon$.

Also liegt die Menge aller Polynome (beliebig hohen Grades) direkt in $C[a, b]$.

Definition 2.6 (Stützstellensystem)

Stützstellensystem : $a \leq x_0^{(n)} < \dots < x_n^{(n)} \leq b$. Weiterhin bezeichne $p_n \in \Pi_n$ das zu den Datenpaaren $(x_k^{(n)}, f(x_k^{(n)}))$ gehörende eindeutig bestimmte Interpolationspolynom.

Satz 2.7 (Satz von Faber 1914)

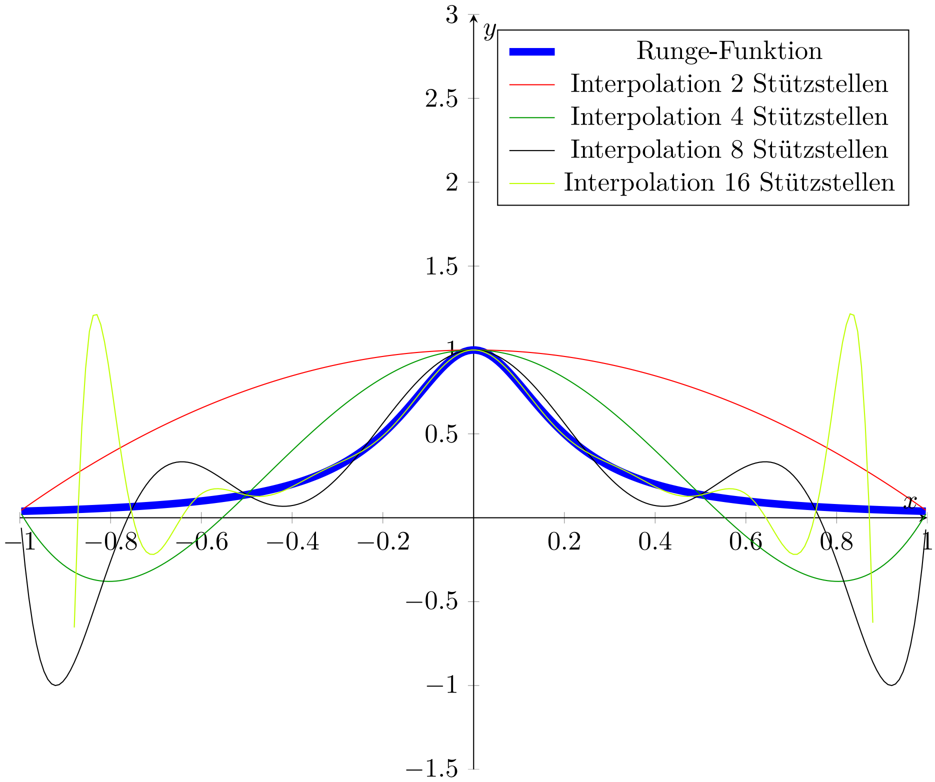
Zu jedem Stützstellensystem gibt es $f \in C[a, b]$, sodass (p_n) nicht gleichmäßig gegen f konvergiert.
$\|p_n - f\|_\infty \to 0$ bedeutet, dass (p_n) gleichmäßig gegen f konvergiert.

Nach einem Resultat von ERDÖS/VERTESI (1980) gilt sogar, dass $(p_n(x))$ fast überall divergiert.

■ **Beispiel 2.8 (Runge)**

$f : \mathbb{R} \to \mathbb{R}$, $f(x) = \frac{1}{1+25x^2}$
äquidistante Stützstellen $x_0, ..., x_n$, $p \in \Pi_n$ als Interpolationspolynom

Stützstellen	**interpoliertes Polynom**
2	$1 - \frac{25x^2}{26}$
4	$3,31565x^4 - 4,27719x^2 + 1$
8	$53,6893x^8 - 102,815x^6 + 61,3672x^4 - 13,203x^2 + 1$
16	$15403,1x^{16} - 49713,5x^{14} + 63743,8x^{12} - 41870x^{10} + 15206x^8 - 3100,35x^6 + 351,984x^4 - 22,7759x^2 + 1$



Anmerkung
Wer mit Mathematica selber diese Polynome berechnen will, muss folgende Befehle benutzen:

- Funktion definieren: `f[x_]:=1/(1+25x^2)`
- Interpolationspolynome ausrechnen: `Expand[InterpolatingPolynomial[Table[{i,f[i]},{i,-1,-1,Schrittweite}],{x}]]`
- plotten: `Plot[f[x],InterpolatingPolynomial[Table[{i,f[i]},{i,-1,-1,Schrittweite}],{x}],{x,-1,1}]`

Satz 2.9
Sei $f \in C^{n+1}[a,b]$ und gelte $a \leq x_0 < ... < x_n \leq b$. Mit $p_n \in \Pi_n$ werde das zu den Datenpaaren $(x_0, f(x_0)), ..., (x_n, f(x_n))$ gehörende Interpolationspolynom bezeichnet. Dann existiert zu jedem $x \in [a,b]$ eine Zahl $\xi \in (a,b)$, so dass

$$f(x) - p_n(x) = \frac{f^{n+1}(\xi(x))}{(n+1)!} w(x) \quad \text{für alle } x \in [a,b]$$
$$\text{wobei } w(x) = (x - x_0) \cdot ... \cdot (x - x_n)$$

Beweis. Für $x = x_k$ mit $k = 0, ..., n$ ist nicht zu zeigen, da p_n die Interpolationsbedingung erfüllt. Sei nun $x \in [a,b]$ fest gewählt mit $x \notin \{x_0, ..., x_n\}$. Weiter seien

$$K = \frac{f(x) - p_n(x)}{w(x)} \quad \text{und} \quad F : \begin{cases} [a,b] \to \mathbb{R} \\ t \mapsto f(t) - p_n(t) - Kw(t) \end{cases}$$

Man stellt unter Beachtung der Interpolationsbedingung fest, dass $F(x_0) = F(x_1) = ... = F(x_n) = 0$ und $F(x) = 0$. Also besitzt F mindestens $n+2$ paarweise verschiedene Nullstellen in $[a,b]$. Da $F \in C^{n+1}[a,b]$ erhält man durch $n+1$-fache Anwendung des Satzes von Rolle, dass $F^{(n+1)}$ mindestens eine Nullstelle $\xi(x)$ in (a,b) besitzt. Also folgt

$$0 = F^{(n+1)}(\xi(x)) = f^{(n+1)}(\xi(x)) - \underbrace{p_n^{(n+1)}(\xi(x))}_{=0} - K \underbrace{w^{(n+1)}(\xi(x))}_{\text{Konstante}}$$

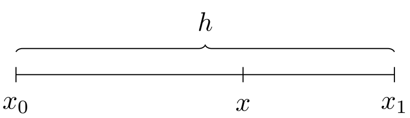
Da $w^{(n+1)} = (n+1)!$, erhält man

$$K = \frac{f^{(n+1)}(\xi(x))}{(n+1)!}$$

Da $x \in [a,b]$ beliebig gewählt war, ist die Behauptung bewiesen. □

■ **Beispiel 2.10**
Sei $f \in C^2[a,b]$ mit $\|f\|_\infty \leq M$. Weiter sei $a = x_0 < x_1 = x_0 + h = b$. Mit Satz 2.9 folgt:

$$\begin{aligned} |f(x) - p_2(x)| &= \left| \frac{f''(\xi(x))}{2} (x - x_0)(x - x_1) \right| \\ &\leq \frac{1}{2} M \cdot \lambda(x) h \cdot (1 - \lambda(x)) h \\ &\leq \frac{1}{2} M \cdot h^2 \underbrace{\lambda(x)(1 - \lambda(x))}_{\leq 1/4} \\ &\leq \frac{1}{8} M \cdot h^2 \end{aligned}$$



$\Rightarrow x = x_0 + \lambda \cdot (x_1 - x_0) = \lambda x_1 + (1 - \lambda) x_0$

3. Interpolation durch Polynomsplines

3.1. Polynomsplines

Zur Abkürzung bezeichne Δ eine Zerlegung des Intervall $[a, b]$ durch die Stützstellen $a =: x_0 < ... < x_n := b$.

Definition 3.1 (Polynomspline)
Ein Polynomspline vom Grad $m \in \mathbb{N}$ und Glattheit $l \in \mathbb{N}$ zur Zerlegung Δ ist eine Funktion $s \in C^l[a, b]$ mit

$$s_k := s|_{[x_k, x_{k+1}]} \in \Pi_n \quad \text{für } k = 0, ..., n-1$$

Dabei bezeichnet $s|_{[x_k, x_{k+1}]}$ die Einschränkung von s auf das Intervall $[x_k, x_{k+1}]$. Die Menge aller Splines wird mit $\mathcal{S}_m^l(\Delta)$ bezeichnet.

Folglich ist ein Polynomspline $s \in \mathcal{S}_m^l(\Delta)$ auf jedem der Teilintervall $[x_k, x_{k+1}]$ ein Polynom vom Höchstgrad m. Außerdem ist $s \in \mathcal{S}_m^l(\Delta)$ in allen Punkten $x \in [a, b]$ (also auch in den Stützstellen) l-mal stetig differenzierbar. $\mathcal{S}_m^l(\Delta)$ ist mit der üblichen Addition und Multiplikation ein Vektorraum. Speziell ist $\mathcal{S}_1^0(\Delta)$ die Menge aller stetigen stückweise affin linearen Funktionen.

3.2. Interpolation durch kubische Polynomsplines

Gegeben sei eine Zerlegung Δ und die Stützwerte $f_0, ..., f_n$. Gesucht ist eine Funktion $s \in \mathcal{S}_3^l(\Delta)$ mit $l = 1, 2$ derart, dass

$$s(x_k) = f_k \quad \text{für } k = 0, ..., n \tag{6}$$

Jede derartige Funktion heißt kubischer Interpolationsspline .

Konstruktion eines solchen Splines:

$$\begin{aligned} h_k &:= x_{k-1} - x_k \quad \text{für } k = 0, ..., n-1 \\ m_k &:= s'(x_k) \quad \text{für } k = 0, ..., n-1 \end{aligned}$$

Wegen $l \in \{1, 2\}$ ist s zunächst stetig differenzierbar. Wegen $s_k = s|_{[x_k, x_{k+1}]}$ für $k = 0, ..., n-1$ und $m = 3$ kann man folgenden Ansatz für s_k benutzen:

$$s_k(x) = a_k(x - x_k)^3 + b_k(x - x_k)^2 + c_k(x - x_k) + d_k \tag{7}$$

Aus den Interpolationsbedingungen Gleichung (6) und der stetigen Differenzierbarkeit aller Funktionen in $s \in \mathcal{S}_m^l(\Delta)$ für $l \geq 1$ ergeben sich folgende Forderungen an s_k, $k = 0, ..., n-1$:

$$\begin{aligned} s_k(x_k) = f_k \quad &\text{und} \quad s_k(x_{k+1}) = f_{k+1} \\ s_k'(x_k) = m_k \quad &\text{und} \quad s_k'(x_{k+1}) = m_{k+1} \end{aligned} \tag{8}$$

Diese liefern:

$$\begin{aligned} d_k &= s_k(x_k) = f_k \\ c_k &= s_k'(x_k) = m_k \end{aligned} \tag{9}$$

und damit:

$$\begin{aligned} s_k(x_{k+1}) &= a_k h_k^3 + b_k h_k^2 + m_k h_k + f_k = f_{k+1} \\ s_k'(x_{k+1}) &= 3a_k h_k^2 + 2b_k h_k + m_k = m_{k+1} \end{aligned}$$

Damit ergeben sich a_k und b_k als eindeutige Lösung für das lineare Gleichungssystem

$$\begin{pmatrix} h_k^3 & h_k^2 \\ 3h_k^2 & 2h_k \end{pmatrix} \begin{pmatrix} a_k \\ b_k \end{pmatrix} = \begin{pmatrix} f_{k+1} - f_k - m_k f_k \\ m_{k+1} - m_k \end{pmatrix} \tag{10}$$

Die Determinante ist $-h_k^4 \neq 0$.

Kapitel II

numerische Quadratur und Integration

1. Integration von Interpolationspolynomen

2. Newton-Cotes-Formeln

3. spezielle Newton-Cotes-Formeln

4. Zusammengesetzte Newton-Cotes-Formeln

5. Gauss'sche Quadraturformeln

Kapitel III

direkte Verfahren für lineare Gleichungssysteme

1. Gauss'scher Algorithmus für quadratische Systeme

2. Lineare Quadratmittelprobleme

3. Kondition linearer Gleichungssysteme

Kapitel IV

Kondition von Aufgaben und Stabilität von Algorithmen

1. Maschinenzahlen und Rundungsfehler

2. Fehleranalyse

Kapitel V

Newton-*Verfahren zur Lösung nichtlinearer Gleichungssysteme*

1. Das Newton-Verfahren

2. Gedämpftes Newton-Verfahren

Kapitel VI

lineare Optimierung

1. Ecken und ihre Charakterisierung

2. Simplex-Verfahren

3. Die Tableauform des Simplex-Verfahrens

4. Revidiertes Simplex-Verfahren

5. Bestimmung einer ersten zulässigen Basislösung

Anhang

Anhang A: Listen

A.1. Liste der Theoreme

A.2. Liste der benannten Sätze, Lemmata und Folgerungen

Index