

# 8

## Polynominterpolation

Dieses Kapitel ist der Approximation einer Funktion gewidmet, die durch ihre Knotenwerte gegeben ist.

Präziser gesagt, besteht das Problem darin, für  $m + 1$  Paare  $(x_i, y_i)$  eine Funktion  $\Phi = \Phi(x)$  derart zu bestimmen, dass  $\Phi(x_i) = y_i$  für  $i = 0, \dots, m$  gilt, wobei  $y_i$  gewisse gegebene Werte sind. Wir sagen, dass  $\Phi$  die Werte  $\{y_i\}$  in den Knoten  $\{x_i\}$  *interpoliert*. Wir sprechen von *Polynominterpolation*, wenn  $\Phi$  ein algebraisches Polynom ist, von *trigonometrischer Approximation*, wenn  $\Phi$  ein trigonometrisches Polynom ist, oder von *stückweiser Polynominterpolation* (oder *Spline-Interpolation*), wenn  $\Phi$  nur lokal ein Polynom ist.

Die Zahlen  $y_i$  könnten die Werte darstellen, die von einer Funktion  $f$  in den Knoten  $x_i$  angenommen werden, die in geschlossener Form oder auch durch experimentelle Daten bekannt ist. Im ersten Fall zielt der Approximationsprozess auf die Ersetzung von  $f$  durch eine einfacher zu handhabende Funktion, insbesondere im Hinblick auf ihre numerische Integration oder Differentiation. Im zweiten Fall ist das primäre Ziel der Approximation eine kompakte Darstellung der verfügbaren Daten zu liefern, deren Zahl oft sehr groß ist.

Polynominterpolation wird in den Abschnitten 8.1 und 8.2 behandelt, wohingegen stückweise Polynominterpolation in den Abschnitten 8.3, 8.4 und 8.5 eingeführt wird. Abschliessend betrachten wir univariate und parametrische Splines in den Abschnitten 8.6 und 8.7. Interpolationsprozesse, die auf trigonometrischen oder algebraischen orthogonalen Polynomen basieren, werden im Kapitel 10 studiert.

## 8.1 Polynominterpolation

Betrachten wir  $n+1$  Paare  $(x_i, y_i)$ . Das Problem besteht darin, ein Polynom  $\Pi_m \in \mathbb{P}_m$ , ein *sogenanntes Interpolationspolynom*, zu finden, so dass

$$\Pi_m(x_i) = a_m x_i^m + \dots + a_1 x_i + a_0 = y_i, \quad i = 0, \dots, n, \quad (8.1)$$

gilt. Die Punkte  $x_i$  heißen *Interpolationsknoten*. Wenn  $n \neq m$  gilt, ist das Problem über- oder unterbestimmt und wird in Abschnitt 10.7.1 besprochen. Ist  $n = m$ , so gilt das folgende Resultat.

**Theorem 8.1** *Seien  $n+1$  verschiedene Punkte  $x_0, \dots, x_n$  und  $n+1$  entsprechende Werte  $y_0, \dots, y_n$  gegeben. Dann existiert ein eindeutig bestimmtes Polynom  $\Pi_n \in \mathbb{P}_n$ , so dass  $\Pi_n(x_i) = y_i$  für  $i = 0, \dots, n$  gilt.*

**Beweis.** Um die Existenz zu zeigen, benutzen wir einen konstruktiven Ansatz, der einen Ausdruck für  $\Pi_n$  liefert. Bezeichnet  $\{l_i\}_{i=0}^n$  eine Basis in  $\mathbb{P}_n$ , so kann  $\Pi_n$  in der Form  $\Pi_n(x) = \sum_{i=0}^n b_i l_i(x)$  dargestellt werden, wobei

$$\Pi_n(x_i) = \sum_{j=0}^n b_j l_j(x_i) = y_i, \quad i = 0, \dots, n, \quad (8.2)$$

gilt. Definieren wir

$$l_i \in \mathbb{P}_n : \quad l_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x - x_j}{x_i - x_j} \quad i = 0, \dots, n, \quad (8.3)$$

so gilt  $l_i(x_j) = \delta_{ij}$  und wir erhalten aus (8.2) unmittelbar  $b_i = y_i$ .

Die Polynome  $\{l_i, i = 0, \dots, n\}$  bilden eine Basis in  $\mathbb{P}_n$  (siehe Übung 1). Somit gibt es ein Interpolationspolynom und es hat die (*Lagrangesche*) Form

$$\Pi_n(x) = \sum_{i=0}^n y_i l_i(x). \quad (8.4)$$

Um die Eindeutigkeit zu beweisen, nehmen wir an, dass ein weiteres Interpolationspolynom  $\Psi_m$  vom Grade  $m \leq n$  existiert, so dass  $\Psi_m(x_i) = y_i$  für  $i = 0, \dots, n$  gilt. Dann verschwindet das Polynom der Differenz  $\Pi_n - \Psi_m$  in  $n+1$  verschiedenen Punkten  $x_i$  und stimmt folglich mit dem Nullpolynom überein. Daher gilt  $\Psi_m = \Pi_n$ .

Ein anderer Zugang zum Beweis der Existenz und Einzigkeit von  $\Pi_n$  wird in Übung 2 gezeigt.  $\diamond$

Es kann gezeigt werden (siehe Übung 3), dass

$$\Pi_n(x) = \sum_{i=0}^n \frac{\omega_{n+1}(x)}{(x - x_i)\omega'_{n+1}(x_i)} y_i \quad (8.5)$$

gilt, wobei  $\omega_{n+1}$  die *Knotenpolynome* vom Grade  $n + 1$  sind, die durch

$$\omega_{n+1}(x) = \prod_{i=0}^n (x - x_i) \quad (8.6)$$

definiert sind. Formel (8.4) heißt die *Lagrangesche-Darstellung* des Interpolationspolynoms und die Polynome  $l_i(x)$  die *Basis-Polynome*. In Abbildung 8.1 sind die Basis-Polynome  $l_2(x)$ ,  $l_3(x)$  und  $l_4(x)$ , im Fall  $n = 6$  auf dem Intervall  $[-1, 1]$  dargestellt, wobei äquidistante Knoten, die Endpunkte eingeschlossen, genommen wurden.

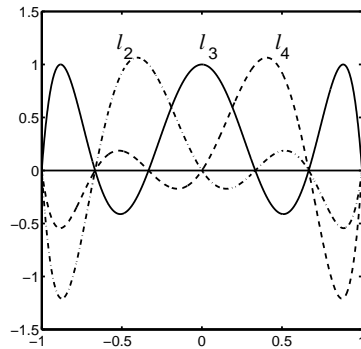


Abbildung 8.1. Basis-Polynome der Lagrange-Darstellung.

Beachte, dass  $|l_i(x)|$  größer als 1 innerhalb des Interpolationsintervalles werden kann.

Gelten für eine gegebene Funktion  $f$  die Beziehungen  $y_i = f(x_i)$ ,  $i = 0, \dots, n$ , so wird das zugeordnete Interpolationspolynom  $\Pi_n(x)$  durch  $\Pi_n f(x)$  bezeichnet.

### 8.1.1 Der Interpolationsfehler

In diesem Abschnitt schätzen wir den Interpolationsfehler ab, der bei Ersetzung einer gegebenen Funktion  $f$  durch ihr Interpolationspolynom  $\Pi_n f$  mit den Knoten  $x_0, x_1, \dots, x_n$  entsteht. Hinsichtlich weiterer Resultate verweisen wir auf [Wen66], [Dav63].

**Theorem 8.2** *Seien  $x_0, x_1, \dots, x_n$  verschiedene Knoten und  $x$  ein Punkt aus dem Definitionsbereich einer gegebenen Funktion  $f$ . Wir nehmen an, dass  $f \in C^{n+1}(I_x)$  gilt, wobei  $I_x$  das kleinste die Knoten  $x_0, x_1, \dots, x_n$  und  $x$  enthaltene Intervall ist. Dann ist der Interpolationsfehler im Punkt  $x$  durch*

$$E_n(x) = f(x) - \Pi_n f(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \omega_{n+1}(x), \quad (8.7)$$

gegeben, wobei  $\xi \in I_x$  und  $\omega_{n+1}$  das Knotenpolynom vom Grade  $n+1$  ist.

**Beweis.** Die Behauptung ist offensichtlich, wenn  $x$  mit irgendeinem Interpolationsknoten zusammenfällt. Andernfalls definieren wir für jedes  $t \in I_x$  die Funktion  $G(t) = E_n(t) - \omega_{n+1}(t)E_n(x)/\omega_{n+1}(x)$ . Da  $f \in C^{(n+1)}(I_x)$  und  $\omega_{n+1}$  ein Polynom ist, ist  $G \in C^{(n+1)}(I_x)$  und besitzt  $n+2$  verschiedene Nullstellen in  $I_x$ , denn

$$G(x_i) = E_n(x_i) - \omega_{n+1}(x_i)E_n(x)/\omega_{n+1}(x) = 0, \quad i = 0, \dots, n$$

$$G(x) = E_n(x) - \omega_{n+1}(x)E_n(x)/\omega_{n+1}(x) = 0.$$

Nach dem Mittelwertsatz besitzt  $G'$   $n+1$  verschiedene Nullstellen und mittels Rekursion  $G^{(j)}$   $n+2-j$  verschiedene Nullstellen. Folglich hat  $G^{(n+1)}$  genau eine Nullstelle, die wir mit  $\xi$  bezeichnen wollen. Andererseits bekommen wir wegen  $E_n^{(n+1)}(t) = f^{(n+1)}(t)$  und  $\omega_{n+1}^{(n+1)}(x) = (n+1)!$

$$G^{(n+1)}(t) = f^{(n+1)}(t) - \frac{(n+1)!}{\omega_{n+1}(x)} E_n(x),$$

was, ausgewertet in  $t = \xi$ , den gewünschten Ausdruck für  $E_n(x)$  ergibt.  $\diamond$

### 8.1.2 Nachteile der polynomialen Interpolation auf äquidistanten Knoten und Runge's Gegenbeispiel

In diesem Abschnitt analysieren wir das Verhalten des Interpolationsfehlers (8.7) wenn  $n$  gegen Unendlich strebt. Dazu definieren wir für jede Funktion  $f \in C^0([a, b])$  ihre *Maximumnorm*

$$\|f\|_\infty = \max_{x \in [a, b]} |f(x)|. \quad (8.8)$$

Ferner führen wir eine untere Dreiecksmatrix  $X$  endlicher Dimension – die *Interpolationsmatrix* auf  $[a, b]$  – ein, deren Einträge  $x_{ij}$ , für  $i, j = 0, 1, \dots$ , die Punkte auf  $[a, b]$  darstellen, wobei wir annehmen, dass in jeder Zeile alle Einträge voneinander verschieden sind.

Somit enthält für jedes  $n \geq 0$ , die  $n+1$ -te Zeile von  $X$   $n+1$  unterschiedliche Werte, die wir als Knoten identifizieren können, so dass wir für eine gegebene Funktion  $f$  ein Interpolationspolynom  $\Pi_n f$  vom Grade  $n$  eindeutig in diesen Knoten bestimmen können (jedes Polynom  $\Pi_n f$  hängt sowohl von  $X$ , als auch von  $f$  ab).

Nachdem wir  $f$  und eine Interpolationsmatrix  $X$  fest vorgegeben haben, definieren wir den Interpolationsfehler

$$E_{n,\infty}(X) = \|f - \Pi_n f\|_\infty, \quad n = 0, 1, \dots \quad (8.9)$$

Als nächstes bezeichnen wir durch  $p_n^* \in \mathbb{P}_n$  die *beste polynomiale Approximation*, für die

$$E_n^* = \|f - p_n^*\|_\infty \leq \|f - q_n\|_\infty \quad \forall q_n \in \mathbb{P}_n$$

gilt.

Es gilt das folgende Vergleichsresultat (zum Beweis siehe [Riv74]).

**Eigenschaft 8.1** Seien  $f \in C^0([a, b])$  und  $X$  eine Interpolationsmatrix auf  $[a, b]$ . Dann gilt

$$E_{n,\infty}(X) \leq E_n^*(1 + \Lambda_n(X)), \quad n = 0, 1, \dots, \quad (8.10)$$

wobei  $\Lambda_n(X)$  die Lebesguekonstante von  $X$  bezeichnet, die als

$$\Lambda_n(X) = \left\| \sum_{j=0}^n |l_j^{(n)}| \right\|_{\infty}, \quad (8.11)$$

definiert ist. Hierbei ist  $l_j^{(n)} \in \mathbb{P}_n$  das  $j$ -te charakteristische Polynom, das zur  $n+1$ -ten Zeile von  $X$  gehört, das also  $l_j^{(n)}(x_{nk}) = \delta_{jk}$ ,  $j, k = 0, 1, \dots$  genügt.

Da  $E_n^*$  nicht von  $X$  abhängt, müssen alle Informationen, die den Einfluss von  $X$  auf  $E_{n,\infty}(X)$  betreffen in  $\Lambda_n(X)$  gesucht werden. Obwohl eine Interpolationsmatrix  $X^*$  existiert, so dass  $\Lambda_n(X)$  minimiert wird, ist es im Allgemeinen nicht einfach ihre Einträge explizit zu bestimmen. Wir werden in Abschnitt 10.3 sehen, dass die Nullstellen der Tschebyschew-Polynome eine Interpolationsmatrix auf dem Intervall  $[-1, 1]$  mit einer sehr kleinen Lebesguekonstante liefern.

Andererseits gibt es für jede mögliche Wahl von  $X$  eine Konstante  $C > 0$ , so dass (siehe [Erd61])

$$\Lambda_n(X) > \frac{2}{\pi} \log(n+1) - C, \quad n = 0, 1, \dots$$

Diese Eigenschaft zeigt, dass  $\Lambda_n(X) \rightarrow \infty$  für  $n \rightarrow \infty$ . Diese Tatsache besitzt wichtige Folgerungen: insbesondere lässt sich beweisen (siehe [Fab14]), dass zu einer gegebenen Interpolationsmatrix  $X$  auf einem Intervall  $[a, b]$ , immer eine auf  $[a, b]$  stetige Funktion  $f$  existiert, so dass  $\Pi_n f$  nicht gleichmäßig (d.h. in der Maximumnorm) gegen  $f$  konvergiert. Folglich, eignet sich die polynomiale Interpolation nicht zur Approximation *jeder* stetigen Funktion, wie das folgende Beispiel zeigt.

**Beispiel 8.1 (Runge's Gegenbeispiel)** Angenommen wir approximieren die Funktion

$$f(x) = \frac{1}{1+x^2}, \quad -5 \leq x \leq 5 \quad (8.12)$$

mit Hilfe der Lagrange-Interpolation auf einem äquidistanten Gitter. Es kann gezeigt werden, dass es einige Punkte  $x$  innerhalb des Interpolationsintervalls gibt, so dass

$$\lim_{n \rightarrow \infty} |f(x) - \Pi_n f(x)| \neq 0.$$

Speziell divergiert die Lagrange-Interpolierende für  $|x| > 3.63 \dots$ . Dieses Phänomen ist, wie in Abbildung 8.2 ersichtlich, besonders deutlich in Umgebung der

Randpunkte des Interpolationsintervalls und ist auf die Verwendung äquidistanter Knoten zurückzuführen. Wir werden in Kapitel 10 sehen, dass durch geeignet gewählte Knoten die gleichmäßige Konvergenz der Interpolationspolynome gegen die Funktion  $f$  ermöglicht wird. •

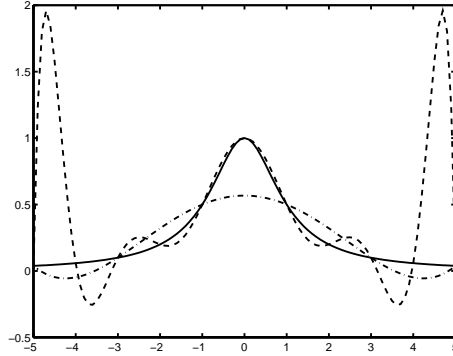


Abbildung 8.2. Lagrange-Interpolation der Funktion  $f(x) = 1/(1+x^2)$  auf äquidistanten Knoten: die Interpolationspolynome  $\Pi_5 f$  und  $\Pi_{10} f$  sind als gepunktete bzw. gestrichelte Kurve dargestellt.

### 8.1.3 Stabilität der Polynom-Interpolation

Betrachten wir eine Menge von Funktionswerten  $\{\tilde{f}(x_i)\}$ , die eine Störung der Daten  $f(x_i)$  in den Knoten  $x_i$ ,  $i = 0, \dots, n$ , im Intervall  $[a, b]$  sei. Die Störung kann beispielsweise durch Rundungsfehler oder durch Fehler bei der experimentellen Messung der Daten verursacht sein.

Indem wir durch  $\Pi_n \tilde{f}$  das Interpolationspolynom auf der Menge der Werte  $\tilde{f}(x_i)$  bezeichnen, erhalten wir

$$\begin{aligned} \|\Pi_n f - \Pi_n \tilde{f}\|_\infty &= \max_{a \leq x \leq b} \left| \sum_{j=0}^n (f(x_j) - \tilde{f}(x_j)) l_j(x) \right| \\ &\leq \Lambda_n(X) \max_{i=0, \dots, n} |f(x_i) - \tilde{f}(x_i)|. \end{aligned}$$

Folglich lassen kleine Änderungen der Daten nur dann kleine Änderungen des Interpolationspolynoms zu, wenn die Lebesguekonstante klein ist. Diese Konstante spielt die Rolle der *Konditionszahl* des Interpolationsproblems.

Wie zuvor bemerkt, wächst  $\Lambda_n$  mit  $n \rightarrow \infty$  und speziell im Fall der Lagrange-Interpolation auf äquidistanten Knoten kann

$$\Lambda_n(X) \simeq \frac{2^{n+1}}{en \log n}$$

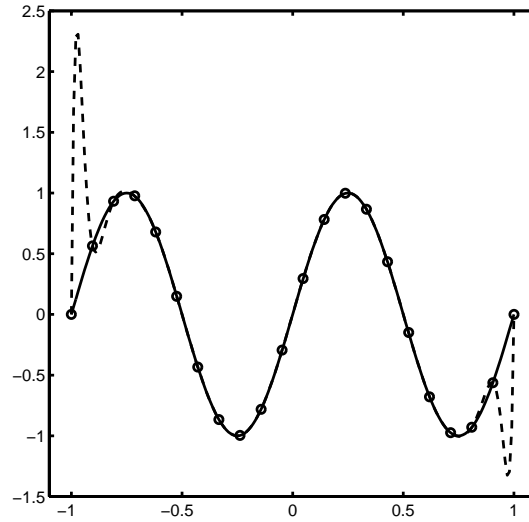


Abbildung 8.3. Instabilität der Lagrange-Interpolation. Durchgezogene Kurve  $\Pi_{21}f$  mit den ungestörten Daten, gestrichelte Kurve  $\Pi_{21}\tilde{f}$  mit den gestörten Daten für das Beispiel 8.2.

bewiesen werden (siehe [Nat65]), wobei  $e \simeq 2.7183$  die Nepersche Zahl ist. Dies zeigt, dass für große  $n$  diese Art der Interpolation instabil werden kann. Beachte, dass wir bis jetzt auch die Fehler vollständig vernachlässigt haben, die durch den Interpolationsprozess bei der Konstruktion der  $\Pi_n f$  erzeugt werden. Es lässt sich jedoch zeigen, dass die Auswirkungen solcher Fehler im Allgemeinen vernachlässigbar sind (siehe [Atk89]).

**Beispiel 8.2** Wir interpolieren auf dem Intervall  $[-1, 1]$  die Funktion  $f(x) = \sin(2\pi x)$  in 22 äquidistanten Knoten  $x_i$ . Danach erzeugen wir eine gestörte Menge von Werten  $\tilde{f}(x_i)$  der Funktionsauswertungen  $f(x_i) = \sin(2\pi x_i)$  für deren Abweichung  $\max_{i=0,\dots,21} |f(x_i) - \tilde{f}(x_i)| \simeq 9.5 \cdot 10^{-4}$  gilt. In Abbildung 8.3 vergleichen wir die Polynome  $\Pi_{21}f$  und  $\Pi_{21}\tilde{f}$ : beachte, dass die Differenz zwischen den beiden Interpolationspolynomen in Umgebung der Endpunkte des Interpolationsintervalls viel größer als die aufgeprägte Störung ist (tatsächlich gilt  $\|\Pi_{21}f - \Pi_{21}\tilde{f}\|_\infty \simeq 2.1635$  und  $\Lambda_{21} \simeq 24000$ ). •

## 8.2 Newtonsche Darstellung des Interpolationspolynoms

Vom praktischen Standpunkt aus ist die Lagrangesche Darstellung (8.4) des Interpolationspolynoms nicht die geeignetste. In diesem Abschnitt führen

wir eine andere Darstellung ein, die sich durch geringeren numerischen Aufwand auszeichnet. Unser Ziel ist das Folgende:

Zu gegebenen  $n + 1$  Paaren  $\{x_i, y_i\}$ ,  $i = 0, \dots, n$ , wollen wir  $\Pi_n$  (mit  $\Pi_n(x_i) = y_i$  für  $i = 0, \dots, n$ ) als Summe von  $\Pi_{n-1}$  (mit  $\Pi_{n-1}(x_i) = y_i$  für  $i = 0, \dots, n-1$ ) und einem Polynom vom Grade  $n$  darstellen, das von den Knoten  $x_i$  und von nur einem unbekanntem Koeffizienten abhängt. Wir setzen folglich

$$\Pi_n(x) = \Pi_{n-1}(x) + q_n(x), \quad (8.13)$$

wobei  $q_n \in \mathbb{P}_n$ . Da  $q_n(x_i) = \Pi_n(x_i) - \Pi_{n-1}(x_i) = 0$  für  $i = 0, \dots, n-1$  gilt, muss notwendigerweise

$$q_n(x) = a_n(x - x_0) \dots (x - x_{n-1}) = a_n \omega_n(x)$$

sein. Um den unbekannten Koeffizienten  $a_n$  zu bestimmen, nehmen wir an, dass  $y_i = f(x_i)$ ,  $i = 0, \dots, n$  gilt. Hierbei ist  $f$  eine geeignete Funktion, die nicht in expliziter Form gegeben sein muss. Wegen  $\Pi_n f(x_n) = f(x_n)$  folgt aus (8.13)

$$a_n = \frac{f(x_n) - \Pi_{n-1}f(x_n)}{\omega_n(x_n)}. \quad (8.14)$$

Der Koeffizient  $a_n$  heißt *n-te Newtonsche dividierte Differenz* und wird üblicherweise durch

$$a_n = f[x_0, x_1, \dots, x_n] \quad (8.15)$$

für  $n \geq 1$  bezeichnet. Damit kann (8.13) in der Form

$$\Pi_n f(x) = \Pi_{n-1} f(x) + \omega_n(x) f[x_0, x_1, \dots, x_n] \quad (8.16)$$

dargestellt werden. Setzen wir  $y_0 = f(x_0) = f[x_0]$  und  $\omega_0 = 1$ , so erhalten wir aus (8.16) durch Rekursion in  $n$  die Formel

$$\Pi_n f(x) = \sum_{k=0}^n \omega_k(x) f[x_0, \dots, x_k]. \quad (8.17)$$

Die Eindeutigkeit des Interpolationspolynoms sichert, dass der obige Ausdruck das gleiche Interpolationspolynom ergibt, das durch die Lagrangesche Darstellung erzeugt wird. Die Darstellung (8.17) ist gemeinhin als *Newtonsche dividierte Differenzenformel* des Interpolationspolynoms bekannt.

Das Programm 65 liefert eine Implementation der Newtonschen Formel. Die Eingabevektoren  $\mathbf{x}$  und  $\mathbf{y}$  enthalten die Interpolationsknoten bzw. die



dazugehörigen Funktionswerte von  $f$ , wohingegen der Vektor  $\mathbf{z}$  die Abszissen enthält an denen das Polynom  $\Pi_n f$  ausgewertet werden soll. Dieses Polynom ist im Ausgabevektor  $\mathbf{f}$  gespeichert.

**Program 65 - interpol** : Lagrangesches Polynom unter Verwendung der Newtonschen Darstellungsformel

```
function [f] = interpol (x,y,z)
[m n] = size(y);
for j = 1:m
    a(:,1) = y(j,:);
    for i = 2:n
        a(i:n,i) = (a(i:n,i-1)-a(i-1,i-1))./(x(i:n)-x(i-1));
    end
    f(j,:) = a(n,n).*(z-x(n-1)) + a(n-1,n-1);
    for i = 2:n-1
        f(j,:) = f(j,:).*(z-x(n-i))+a(n-i,n-i);
    end
end
end
```

### 8.2.1 Einige Eigenschaften der Newtonschen dividierten Differenzen

Die  $n$ -te dividierte Differenz  $f[x_0, \dots, x_n] = a_n$  kann auch dadurch charakterisiert werden, dass sie der Koeffizient von  $x^n$  in  $\Pi_n f$  ist. Selektieren wir diesen Koeffizienten aus (8.5) und setzen ihn gleich dem entsprechenden Koeffizienten in der Newtonschen Darstellungsformel (8.17), so gewinnen wir die explizite Darstellung

$$f[x_0, \dots, x_n] = \sum_{i=0}^n \frac{f(x_i)}{\omega'_{n+1}(x_i)}. \quad (8.18)$$

Diese Formel hat zwei bemerkenswerte Folgerungen:

1. der von der dividierten Differenz angenommene Wert ist invariant in Bezug auf Permutationen der Knotenindizes. Dieser Umstand kann vorteilhaft ausgenutzt werden, wenn Stabilitätsprobleme einen Wechsel der Indizes nahelegen (z.B. wenn  $x$  der Punkt ist, in dem das Polynom berechnet werden muss, ist es bequem eine Permutation der Indizes derart einzuführen, dass  $|x - x_k| \leq |x - x_{k-1}|$  mit  $k = 1, \dots, n$ );
2. gilt  $f = \alpha g + \beta h$  für gewisse  $\alpha, \beta \in \mathbb{R}$ , so folgt

$$f[x_0, \dots, x_n] = \alpha g[x_0, \dots, x_n] + \beta h[x_0, \dots, x_n];$$

3. ist  $f = gh$ , so gilt die folgende Formel (Leibniz-Formel genannt) (siehe [Die93])

$$f[x_0, \dots, x_n] = \sum_{j=0}^n g[x_0, \dots, x_j] h[x_j, \dots, x_n];$$

4. eine algebraische Umformung von (8.18) (siehe Übung 7) liefert die *Rekursionsbeziehung* zur Berechnung der dividierten Differenzen

$$f[x_0, \dots, x_n] = \frac{f[x_1, \dots, x_n] - f[x_0, \dots, x_{n-1}]}{x_n - x_0}, \quad n \geq 1. \quad (8.19)$$

Das Programm 66 implementiert die Rekursionsbeziehung (8.19). Die Funktionswerte von  $f$  in den Interpolationsknoten  $\mathbf{x}$  werden im Vektor  $\mathbf{y}$  gespeichert, die Ausgabematrix  $\mathbf{d}$  (untere Dreiecksmatrix) enthält die dividierten Differenzen, die in der Form

$$\begin{array}{c|cccc} x_0 & f[x_0] & & & \\ x_1 & f[x_1] & f[x_0, x_1] & & \\ x_2 & f[x_2] & f[x_1, x_2] & f[x_0, x_1, x_2] & \\ \vdots & \vdots & & \vdots & \ddots \\ x_n & f[x_n] & f[x_{n-1}, x_n] & f[x_{n-2}, x_{n-1}, x_n] & \dots & f[x_0, \dots, x_n] \end{array}$$

gespeichert sind. Die in der Newtonschen Formel involvierten Koeffizienten sind die Diagonaleinträge der Matrix.

**Program 66 - dividif** : Newtonsche dividierte Differenzen

```
function [d]=dividif(x,y)
[n,m]=size(y);
if n == 1, n = m; end
n = n-1; d = zeros (n+1,n+1); d(:,1) = y';
for j = 2:n+1
    for i = j:n+1
        d(i,j) = (d(i-1,j-1)-d(i,j-1))/(x(i-j+1)-x(i));
    end
end
end
```

Wenn man (8.19) nutzt, werden  $n(n+1)$  Summationen und  $n(n+1)/2$  Divisionen benötigt, um die Gesamtmatrix zu erzeugen. Wenn eine neue Auswertung von  $f$  in einem neuen Knoten  $x_{n+1}$  verfügbar wäre, würde nur die Berechnung einer neuen Zeile der Matrix erforderlich werden ( $f[x_n, x_{n+1}]$ ,  $\dots$ ,  $f[x_0, x_1, \dots, x_{n+1}]$ ). Für die Konstruktion von  $\Pi_{n+1}f$  aus  $\Pi_n f$  genügt es somit, den Term  $a_{n+1}\omega_{n+1}(x)$  mit einem Rechenaufwand von  $(n+1)$  Divisionen und  $2(n+1)$  Summationen zu  $\Pi_n f$  zuaddieren. Zur Vereinfachung der Schreibweise nutzen wir unten  $D^r f_i = f[x_i, x_{i+1}, \dots, x_r]$ .

**Beispiel 8.3** In Tabelle 8.1 sind die dividierten Differenzen der Funktion  $f(x) = 1 + \sin(3x)$  auf dem Intervall  $(0,2)$  angegeben. Die Funktionswerte  $f$  und die entsprechenden dividierten Differenzen wurden unter Verwendung von 16 signifikanten Stellen berechnet, obgleich nur die ersten 5 Ziffern angegeben wurden. Wenn der Wert von  $f$  im Knoten  $x = 0.2$  verfügbar wäre, würde eine Aktualisierung der Tabelle der dividierten Differenzen nur die Berechnung der in Tabelle 8.1 kursiv dargestellten Einträge erfordern. •

Tabelle 8.1. Dividierte Differenzen für die Funktion  $f(x) = 1 + \sin(3x)$  im Fall, in dem die Auswertung von  $f$  in  $x = 0.2$  auch verfügbar wird. Die neu berechneten Werte sind kursiv dargestellt.

$x_i$	$f(x_i)$	$f[x_i, x_{i-1}]$	$D^2 f_i$	$D^3 f_i$	$D^4 f_i$	$D^5 f_i$	$D^6 f_i$
0	1.0000						
0.2	1.5646	2.82					
0.4	1.9320	1.83	-2.46				
0.8	1.6755	-0.64	-4.13	-2.08			
1.2	0.5575	-2.79	-2.69	1.43	2.93		
1.6	0.0038	-1.38	1.76	3.71	1.62	-0.81	
2.0	0.7206	1.79	3.97	1.83	-1.17	-1.55	-0.36

Beachte, dass  $f[x_0, \dots, x_n] = 0$  für jedes  $f \in \mathbb{P}_{n-1}$  gilt. Diese Eigenschaft ist jedoch nicht immer numerisch erfüllt, da die Berechnung der dividierten Differenzen stark durch Rundungsfehler beeinflusst sein kann.

**Beispiel 8.4** Wir betrachten erneut die dividierten Differenzen für die Funktion  $f(x) = 1 + \sin(3x)$  nun aber im Intervall  $(0, 0.0002)$ . In einer hinreichend kleinen Nachbarschaft von 0 verhält sich die Funktion wie  $1 + 3x$ , so dass wir bei wachsender Ordnung der dividierten Differenzen kleinere Zahlen erwarten. Jedoch weisen die mit dem Programm 66 erhaltenen Ergebnisse, die in Tabelle 8.2 in Exponentenschreibweise auf die ersten 4 Ziffern (obgleich zur Berechnung 16 Stellen verwendet wurden) dargestellt sind, ein grundlegend anderes Bild auf. Die bei der Berechnung der dividierten Differenzen niederer Ordnung auftretenden Rundungsfehler wirken sich dramatisch auf die dividierten Differenzen höherer Ordnung aus. •

### 8.2.2 Der Interpolationsfehler bei der Verwendung dividierter Differenzen

Betrachten wir die Knoten  $x_0, \dots, x_n$  und sei  $\Pi_n f$  das Interpolationspolynom von  $f$  in diesen Knoten. Nun sei  $x$  ein von den bereits vorhandenen verschiedener Knoten. Wir setzen  $x_{n+1} = x$  und bezeichnen durch  $\Pi_{n+1} f$  das Interpolationspolynom von  $f$  in den Knoten  $x_k$ ,  $k = 0, \dots, n+1$ . Unter

Tabelle 8.2. Dividierte Differenzen der Funktion  $f(x) = 1 + \sin(3x)$  auf dem Intervall  $(0, 0.0002)$ . Beachte den vollständig falschen Wert in der letzten Spalte (er müsste näherungsweise Null sein), der auf die Ausbreitung von Rundungsfehlern durch den Algorithmus zurückzuführen ist.

$x_i$	$f(x_i)$	$f[x_i, x_{i-1}]$	$D^2 f_i$	$D^3 f_i$	$D^4 f_i$	$D^5 f_i$
0	1.0000					
4.0e-5	1.0001	3.000				
8.0e-5	1.0002	3.000	-5.39e-4			
1.2e-4	1.0004	3.000	-1.08e-3	-4.50		
1.6e-4	1.0005	3.000	-1.62e-3	-4.49	1.80e+1	
2.0e-4	1.0006	3.000	-2.15e-3	-4.49	-7.23	<span style="border: 1px solid black; padding: 2px;">-1.2e+5</span>

Verwendung der Newtonschen dividierten Differenzen erhalten wir

$$\Pi_{n+1}f(t) = \Pi_n f(t) + (t - x_0) \dots (t - x_n) f[x_0, \dots, x_n, t].$$

Da  $\Pi_{n+1}f(x) = f(x)$  gilt, ergibt sich die folgende Formel für den Interpolationsfehler im Punkt  $t = x$

$$\begin{aligned} E_n(x) &= f(x) - \Pi_n f(x) = \Pi_{n+1}f(x) - \Pi_n f(x) \\ &= (x - x_0) \dots (x - x_n) f[x_0, \dots, x_n, x] \\ &= \omega_{n+1}(x) f[x_0, \dots, x_n, x]. \end{aligned} \quad (8.20)$$

Die Annahme  $f \in C^{(n+1)}(I_x)$  und der Vergleich von (8.20) mit (8.7) erbringt

$$f[x_0, \dots, x_n, x] = \frac{f^{(n+1)}(\xi)}{(n+1)!} \quad (8.21)$$

für ein geeignet gewähltes  $\xi \in I_x$ . Da (8.21) dem Restglied der Taylorentwicklung von  $f$  ähnelt, wird die Newtonsche Formel (8.17) für das Interpolationspolynom oft als abgebrochene Entwicklung um  $x_0$  angesehen, vorausgesetzt, dass  $|x_n - x_0|$  nicht zu gross ist.

### 8.3 Stückweise Lagrange-Interpolation

In Abschnitt 8.1.2 haben wir den Fakt erwähnt, dass für äquidistante Interpolationsknoten die gleichmäßige Konvergenz  $\Pi_n f$  gegen  $f$  für  $n \rightarrow \infty$  nicht gesichert ist. Andererseits sind äquidistante Knoten numerisch bedeutend einfacher zu handhaben und die Lagrange-Interpolation niederen Grades ausreichend genau, wenn hinreichend kleine Interpolationsintervalle betrachtet werden.

Daher ist es naheliegend, eine Partition  $\mathcal{T}_h$  von  $[a, b]$  in  $K$  Teilintervalle  $I_j = [x_j, x_{j+1}]$  der Länge  $h_j$ , mit  $h = \max_{0 \leq j \leq K-1} h_j$ , einzuführen, so

Tabelle 8.3. Interpolationsfehler der stückweisen Lagrange-Interpolation vom Grade  $k = 1$  und  $k = 2$  im Fall der Runge-Funktion (8.12);  $p$  bezeichnet den Trend des Exponenten von  $h$ . Beachte, dass für  $h \rightarrow 0$ , wie durch (8.23) vorausgesetzt,  $p \rightarrow k + 1$  geht.

$h$	$\ f - \Pi_1^h\ _\infty$	$p$	$\ f - \Pi_2^h\ _\infty$	$p$
5	0.4153		0.0835	
2.5	0.1787	1.216	0.0971	-0.217
1.25	0.0631	1.501	0.0477	1.024
0.625	0.0535	0.237	0.0082	2.537
0.3125	0.0206	1.374	0.0010	3.038
0.15625	0.0058	1.819	1.3828e-04	2.856
0.078125	0.0015	1.954	1.7715e-05	2.964

dass  $[a, b] = \cup_{j=0}^{K-1} I_j$  gilt und die Lagrange-Interpolation auf jedem  $I_j$  unter Verwendung von  $n + 1$  äquidistanten Knoten  $\{x_j^{(i)}, 0 \leq i \leq k\}$  bei kleinem  $k$  anzuwenden.

Für  $k \geq 1$  führen wir auf  $\mathcal{T}_h$  den stückweise polynomialen Raum

$$X_h^k = \{v \in C^0([a, b]) : v|_{I_j} \in \mathbb{P}_k(I_j) \forall I_j \in \mathcal{T}_h\} \quad (8.22)$$

als Raum der auf  $[a, b]$  stetigen Funktionen, deren Einschränkungen auf jedem  $I_j$  Polynome vom Grade  $\leq k$  sind, ein.

Für eine beliebige, auf  $[a, b]$  stetige Funktion  $f$  stimmt dann die *stückweise polynomiale Interpolation*  $\Pi_h^k f$  auf jedem  $I_j$  mit dem Interpolationspolynom von  $f|_{I_j}$  in den  $k + 1$  Knoten  $\{x_j^{(i)}, 0 \leq i \leq k\}$  überein. Folglich erhalten wir für  $f \in C^{k+1}([a, b])$  unter Verwendung von (8.7) in jedem Teilintervall die Fehlerabschätzung

$$\|f - \Pi_h^k f\|_\infty \leq Ch^{k+1} \|f^{(k+1)}\|_\infty. \quad (8.23)$$

Beachte, dass ein kleiner Interpolationsfehler auch für kleine  $k$  erzielt werden kann, wenn nur  $h$  genügend “klein” ist.

**Beispiel 8.5** Kehren wir zu der Funktion des Gegenbeispiels von Runge zurück. Jetzt werden stückweise Polynome vom Grade  $k = 1$  und  $k = 2$  verwendet. Wir testen experimentell das Verhalten des Fehlers bei fallendem  $h$ . In Tabelle 8.3 sind die absoluten Fehler gemessen in der Maximumnorm über dem Intervall  $[-5, 5]$  und die entsprechenden Abschätzungen der Konvergenzordnung  $p$  in Bezug auf  $h$  dargestellt. Abgesehen vom Fall der Verwendung einer extrem kleinen Zahl von Teilintervallen entsprechen die Resultate der theoretischen Abschätzung (8.23), d.h.  $p = k + 1$ . •

Neben der Abschätzung (8.23) gibt es Konvergenzresultate in Integralnormen (siehe [QV94], [EEHJ96]). Hierzu führen wir den Raum

$$L^2(a, b) = \left\{ f : (a, b) \rightarrow \mathbb{R}, \int_a^b |f(x)|^2 dx < +\infty \right\}, \quad (8.24)$$

mit

$$\|f\|_{L^2(a, b)} = \left( \int_a^b |f(x)|^2 dx \right)^{1/2} \quad (8.25)$$

ein. Die Formel (8.25) definiert eine Norm auf  $L^2(a, b)$ . (Wir erinnern daran, dass Normen und Halbnormen von Funktionen auf gleiche Weise definiert werden können, wie dies in Definition 1.17 in Band 1 für Vektoren gemacht wurde.) Wir machen den Leser darauf aufmerksam, dass das Integral der Funktion  $|f|^2$  in (8.24) im Lebesgue Sinne aufzufassen ist (siehe z.B. [Rud83]). Insbesondere muss  $f$  keineswegs überall stetig sein.

**Theorem 8.3** *Sei  $0 \leq m \leq k+1$  mit  $k \geq 1$  und nehmen wir an, dass  $f^{(m)} \in L^2(a, b)$  für  $0 \leq m \leq k+1$ . Dann gibt es eine positive Konstante  $C$ , unabhängig von  $h$ , so dass*

$$\|(f - \Pi_h^k f)^{(m)}\|_{L^2(a, b)} \leq C h^{k+1-m} \|f^{(k+1)}\|_{L^2(a, b)}. \quad (8.26)$$

Für  $k=1$  und  $m=0$  oder  $m=1$  erhalten wir insbesondere

$$\begin{aligned} \|f - \Pi_h^1 f\|_{L^2(a, b)} &\leq C_1 h^2 \|f''\|_{L^2(a, b)}, \\ \|(f - \Pi_h^1 f)'\|_{L^2(a, b)} &\leq C_2 h \|f''\|_{L^2(a, b)}, \end{aligned} \quad (8.27)$$

mit positiven Konstanten  $C_1$  und  $C_2$ .

**Beweis.** Wir beweisen nur (8.27) und verweisen auf [QV94], Kapitel 3 für den Beweis von (8.26) im allgemeinen Fall.

Setze  $e = f - \Pi_h^1 f$ . Da  $e(x_j) = 0$  für alle  $j = 0, \dots, K$ , folgt aus dem Satz von Rolle die Existenz von  $\xi_j \in (x_j, x_{j+1})$  für  $j = 0, \dots, K-1$ , so dass  $e'(\xi_j) = 0$ .

$\Pi_h^1 f$  ist eine lineare Funktion auf jedem  $I_j$ , somit erhalten wir für  $x \in I_j$

$$e'(x) = \int_{\xi_j}^x e''(s) ds = \int_{\xi_j}^x f''(s) ds,$$

woraus

$$|e'(x)| \leq \int_{x_j}^{x_{j+1}} |f''(s)| ds, \quad \text{für } x \in [x_j, x_{j+1}]. \quad (8.28)$$

Wir erinnern an die *Cauchy-Schwarzsche Ungleichung*

$$\left| \int_{\alpha}^{\beta} u(x)v(x) dx \right| \leq \left( \int_{\alpha}^{\beta} u^2(x) dx \right)^{1/2} \left( \int_{\alpha}^{\beta} v^2(x) dx \right)^{1/2}, \quad (8.29)$$

die für  $u, v \in L^2(\alpha, \beta)$  gilt. Wenden wir diese Ungleichung auf (8.28) an, bekommen wir

$$\begin{aligned} |e'(x)| &\leq \left( \int_{x_j}^{x_{j+1}} 1^2 dx \right)^{1/2} \left( \int_{x_j}^{x_{j+1}} |f''(s)|^2 ds \right)^{1/2} \\ &\leq h^{1/2} \left( \int_{x_j}^{x_{j+1}} |f''(s)|^2 ds \right)^{1/2}. \end{aligned} \quad (8.30)$$

Um eine Schranke für  $|e(x)|$  zu finden, erwähnen wir, dass

$$e(x) = \int_{x_j}^x e'(s) ds$$

gilt, woraus mittels (8.30)

$$|e(x)| \leq \int_{x_j}^{x_{j+1}} |e'(s)| ds \leq h^{3/2} \left( \int_{x_j}^{x_{j+1}} |f''(s)|^2 ds \right)^{1/2} \quad (8.31)$$

folgt. Weiter gilt

$$\int_{x_j}^{x_{j+1}} |e'(x)|^2 dx \leq h^2 \int_{x_j}^{x_{j+1}} |f''(s)|^2 ds \quad \text{und} \quad \int_{x_j}^{x_{j+1}} |e(x)|^2 dx \leq h^4 \int_{x_j}^{x_{j+1}} |f''(s)|^2 ds,$$

woraus wir durch Summation über den Index  $j$  von 0 bis  $K-1$  und durch Ziehen der Quadratwurzel auf beiden Seiten

$$\left( \int_a^b |e'(x)|^2 dx \right)^{1/2} \leq h \left( \int_a^b |f''(x)|^2 dx \right)^{1/2},$$

und

$$\left( \int_a^b |e(x)|^2 dx \right)^{1/2} \leq h^2 \left( \int_a^b |f''(x)|^2 dx \right)^{1/2}$$

erhalten, die die gewünschte Abschätzung (8.27) mit  $C_1 = C_2 = 1$  ist.  $\diamond$

## 8.4 Hermite-Birkoff-Interpolation

Die polynomiale Lagrange-Interpolation kann auf den Fall verallgemeinert werden, in welchem auch die Werte der Ableitungen einer Funktion  $f$  in gewissen (oder allen) Knoten  $x_i$  bekannt sind.

Wir wollen nun annehmen, dass  $(x_i, f^{(k)}(x_i))$ , gegebene Daten sind, wobei  $i = 0, \dots, n$ ,  $k = 0, \dots, m_i$  und  $m_i \in \mathbb{N}$ . Sei ferner  $N = \sum_{i=0}^n (m_i + 1)$ . Es kann bewiesen werden (siehe [Dav63]), dass wenn alle Knoten  $\{x_i\}$  verschieden sind, ein eindeutig bestimmtes Polynom, das *Hermite'sche Interpolationspolynom*,  $H_{N-1} \in \mathbb{P}_{N-1}$  existiert, so dass

$$H_{N-1}^{(k)}(x_i) = y_i^{(k)}, \quad i = 0, \dots, n \quad k = 0, \dots, m_i$$

gilt. Das Hermitesche Interpolationspolynom ist von der Form

$$H_{N-1}(x) = \sum_{i=0}^n \sum_{k=0}^{m_i} y_i^{(k)} L_{ik}(x) \quad (8.32)$$

mit  $y_i^{(k)} = f^{(k)}(x_i)$ ,  $i = 0, \dots, n$ ,  $k = 0, \dots, m_i$ .

Die Funktionen  $L_{ik} \in \mathbb{P}_{N-1}$  werden die *Hermiteschen charakteristischen Polynome* genannt und sind durch die Beziehungen

$$\frac{d^p}{dx^p}(L_{ik})(x_j) = \begin{cases} 1 & \text{if } i = j \text{ und } k = p, \\ 0 & \text{andernfalls} \end{cases}$$

definiert. Führt man die Polynome

$$l_{ij}(x) = \frac{(x - x_i)^j}{j!} \prod_{\substack{k=0 \\ k \neq i}}^n \left( \frac{x - x_k}{x_i - x_k} \right)^{m_k+1}, \quad i = 0, \dots, n, \quad j = 0, \dots, m_i,$$

ein und setzt  $L_{im_i}(x) = l_{im_i}(x)$  für  $i = 0, \dots, n$ , so erhalten wir für die Polynome  $L_{ij}$  die Rekursionsbeziehung

$$L_{ij}(x) = l_{ij}(x) - \sum_{k=j+1}^{m_i} l_{ij}^{(k)}(x_i) L_{ik}(x) \quad j = m_i - 1, m_i - 2, \dots, 0.$$

Was den Interpolationsfehler anbetrifft, gilt die folgende Abschätzung

$$f(x) - H_{N-1}(x) = \frac{f^{(N)}(\xi)}{N!} \Omega_N(x) \quad \forall x \in \mathbb{R},$$

wobei  $\xi \in I(x; x_0, \dots, x_n)$  und  $\Omega_N$  das durch

$$\Omega_N(x) = (x - x_0)^{m_0+1} (x - x_1)^{m_1+1} \dots (x - x_n)^{m_n+1} \quad (8.33)$$

definierte Polynom vom Grade  $N$  sind.

**Beispiel 8.6 (Oskulierende Interpolation)** Sei  $m_i = 1$  für  $i = 0, \dots, n$  gesetzt. In diesem Fall ist  $N = 2n + 2$  und das interpolierende Hermite-Polynom wird *oskulierendes Polynom*, genannt. Es ist durch

$$H_{N-1}(x) = \sum_{i=0}^n \left( y_i A_i(x) + y_i^{(1)} B_i(x) \right)$$

gegeben, wobei  $A_i(x) = (1 - 2(x - x_i)l'_i(x_i))l_i(x)^2$  und  $B_i(x) = (x - x_i)l_i(x)^2$ , für  $i = 0, \dots, n$ , mit

$$l'_i(x_i) = \sum_{k=0, k \neq i}^n \frac{1}{x_i - x_k}, \quad i = 0, \dots, n$$



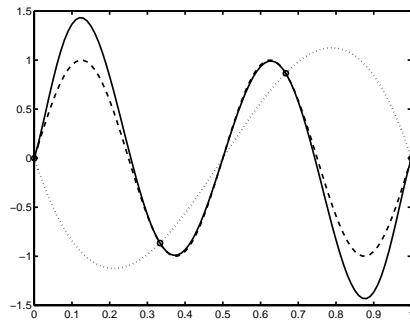


Abbildung 8.4. Lagrangesche und Hermiteische Interpolation der Funktion  $f(x) = \sin(4\pi x)$  auf dem Intervall  $[0, 1]$ .

sind. Zum Vergleich verwenden wir die Programme 65 und 67, um das Lagrange- bzw. das Hermiteische Interpolationspolynom der Funktion  $f(x) = \sin(4\pi x)$  auf dem Intervall  $[0, 1]$  mit vier äquidistanten Knoten ( $n = 3$ ) zu berechnen. Abbildung 8.4 zeigt die Graphen der Funktion  $f$  (gestrichelte Kurve) und der beiden Polynome  $\Pi_n f$  (gepunktete Kurve) und  $H_{N-1}$  (durchgezogene Kurve). •

Das Programm 67 berechnet die Werte des oskulierenden Polynoms in den Abszissen, die im Vektor  $\mathbf{z}$  enthalten sind. Die Eingabevektoren  $\mathbf{x}$ ,  $\mathbf{y}$  und  $\mathbf{dy}$  enthalten die Interpolationsknoten und die entsprechenden Funktionswerte von  $f$  bzw.  $f'$ .

**Program 67 - hermpol** : Oskulierendes Polynom

```
function [herm] = hermite(x,y,dy,z)
n = max(size(x)); m = max(size(z)); herm = [];
for j = 1:m
    xx = z(j); hxv = 0;
    for i = 1:n,
        den = 1; num = 1; xn = x(i); derLi = 0;
        for k = 1:n,
            if k ~= i, num = num*(xx-x(k)); arg = xn-x(k);
                den = den*arg; derLi = derLi+1/arg;
            end
        end
        Lix2 = (num/den)^2; p = (1-2*(xx-xn)*derLi)*Lix2;
        q = (xx-xn)*Lix2; hxv = hxv+(y(i)*p+dy(i)*q);
    end
    herm = [herm, hxv];
end
```

## 8.5 Erweiterung auf den zweidimensionalen Fall

In diesem Abschnitt widmen wir uns kurz der Erweiterung der vorangegangenen Konzepte auf den zweidimensionalen Fall und verweisen auf [SL89], [CHQZ88], [QV94] für weitere Details. Wir bezeichnen durch  $\Omega$  ein beschränktes Gebiet in  $\mathbb{R}^2$  und durch  $\mathbf{x} = (x, y)$  den Koordinatenvektor eines Punktes in  $\Omega$ .

### 8.5.1 Polynominterpolation

Eine besonders einfache Situation tritt auf, wenn  $\Omega = [a, b] \times [c, d]$ , d.h. wenn das Interpolationsgebiet  $\Omega$  das Tensorprodukt zweier Intervalle ist. In diesem Fall führen wir die Knoten  $a = x_0 < x_1 < \dots < x_n = b$  und  $c = y_0 < y_1 < \dots < y_m = d$  ein und das Interpolationspolynom  $\Pi_{n,m}f$  kann in der Form  $\Pi_{n,m}f(x, y) = \sum_{i=0}^n \sum_{j=0}^m \alpha_{ij} l_i(x) l_j(y)$  geschrieben werden, wobei  $l_i \in \mathbb{P}_n$ ,  $i = 0, \dots, n$ , und  $l_j \in \mathbb{P}_m$ ,  $j = 0, \dots, m$ , die charakteristischen eindimensionalen Lagrangeschen Polynome in Bezug auf die  $x$  bzw.  $y$  Variable und  $\alpha_{ij} = f(x_i, y_j)$  sind.

Die Nachteile der eindimensionalen Lagrange-Interpolation sind auch dem zweidimensionalen Fall eigen, wie durch das Beispiel in Abbildung 8.5 bestätigt wird.

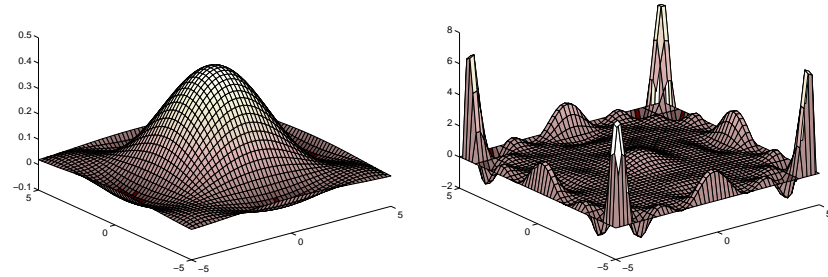


Abbildung 8.5. Runges Gegenbeispiel erweitert auf den zweidimensionalen Fall: Interpolationspolynom auf einem Gitter von  $6 \times 6$  Knoten (links) und  $11 \times 11$  Knoten (rechts). Beachte den Wechsel der senkrechten Skalierung in beiden Darstellungen.

**Bemerkung 8.1 (Der allgemeine Fall)** Ist  $\Omega$  kein Rechteck oder sind die Interpolationsknoten nicht gleichförmig über ein kartesisches Gitter verteilt, ist das Interpolationsproblem schwierig zu lösen und es ist im Allgemeinen vorteilhafter, zur kleinste Quadratellösung zu greifen (siehe Abschnitt 10.7). Wir verweisen auch darauf, dass in  $d$  Dimensionen (mit  $d \geq 2$ )

das Problem der Bestimmung eines Interpolationspolynoms vom Grade  $n$  in Bezug auf jede Raumvariable auf  $n + 1$  verschiedenen Knoten schlecht gestellt sein kann.

Betrachten wir beispielsweise ein Polynom vom Grade 1 in Bezug auf  $x$  und  $y$  der Gestalt  $p(x, y) = a_3xy + a_2x + a_1y + a_0$  um eine Funktion  $f$  in den Knoten  $(-1, 0)$ ,  $(0, -1)$ ,  $(1, 0)$  und  $(0, 1)$  zu interpolieren. Obwohl die Knoten verschieden sind, besitzt das Problem (das nichtlinear ist) im Allgemeinen keine eindeutig bestimmte Lösung; tatsächlich gelangen wir durch Auferlegung der Interpolationsbedingungen zu einem System, das für jeden Wert des Koeffizienten  $a_3$  erfüllt ist. ■

### 8.5.2 Stückweise polynomiale Interpolation

Im mehrdimensionalen Fall ermöglicht die grössere Flexibilität der stückweisen Interpolation Gebiete komplexer Struktur leichter zu handhaben. Nehmen wir an, dass  $\Omega$  ein Polynom im  $\mathbb{R}^2$  ist. Dann kann  $\Omega$  in  $K$  nichtüberlappende Dreiecke (oder *Elemente*)  $T$  zerlegt werden. Diese sogenannte *Triangulation* des Gebietes werden wir durch  $\mathcal{T}_h$  bezeichnen. Offensichtlich gilt  $\overline{\Omega} = \bigcup_{T \in \mathcal{T}_h} T$ . Nehmen wir an, dass die maximale Länge der Dreiecksanten kleiner als eine positive Zahl  $h$  ist. Wie in Abbildung 8.6 (links) dargestellt, sind nicht alle Zerlegungen erlaubt. Genauer gesagt, sind die zulässigen Zerlegungen jene, für die jedes Paar nicht disjunkter Dreiecke eine Ecke oder eine Kante gemeinsam haben.

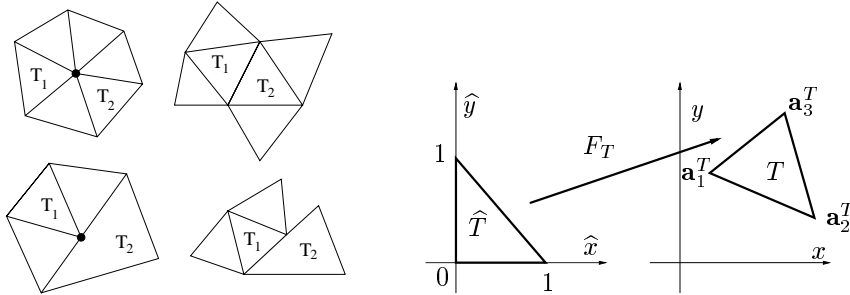


Abbildung 8.6. Die linke Seite des Bildes zeigt zulässige (oben) und nicht zulässige Triangulierungen, die rechte Seite die affine Abbildung des Referenzdreiecks  $\hat{T}$  auf das allgemeine Element  $T \in \mathcal{T}_h$ .

Jedes Element  $T \in \mathcal{T}_h$  der Fläche  $|T|$  ist das Bild der affinen Abbildung  $\mathbf{x} = F_T(\hat{\mathbf{x}}) = \mathbf{B}_T \hat{\mathbf{x}} + \mathbf{b}_T$  des *Referenzdreiecks*  $\hat{T}$  mit den Ecken  $(0,0)$ ,  $(1,0)$  und  $(0,1)$  in der  $\hat{\mathbf{x}} = (\hat{x}, \hat{y})$  Ebene (siehe Abbildung 8.6, rechts), wobei die invertierbare Matrix  $\mathbf{B}_T$  bzw. der rechte Seite Vektor  $\mathbf{b}_T$  durch

$$\mathbf{B}_T = \begin{bmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{bmatrix}, \quad \mathbf{b}_T = (x_1, y_1)^T, \quad (8.34)$$

gegeben sind und die Koordinaten der Ecken von  $T$  durch  $\mathbf{a}_T^{(l)} = (x_l, y_l)^T$ ,  $l = 1, 2, 3$ , bezeichnet wurden.

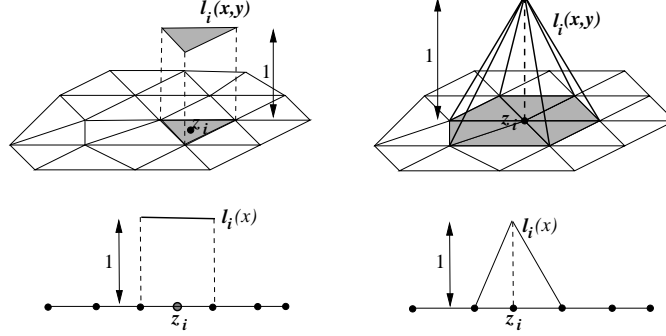


Abbildung 8.7. Charakteristisches stückweise Lagrangesche Polynom, in einer und zwei Raumdimensionen. Links  $k = 0$ ; rechts  $k = 1$ .

Die affine Abbildung (8.34) ist bei praktischen Berechnungen von immenser Bedeutung, da, wenn einmal eine Basis zur Darstellung der stückweise polynomialen Interpolation auf  $\hat{T}$  erzeugt wurde, es durch Koordinatenwechsel  $\mathbf{x} = F_T(\hat{\mathbf{x}})$  möglich ist, das Polynom auf jedem Element  $T$  von  $\mathcal{T}_h$  zu rekonstruieren. Wir sind daher interessiert, lokale Basisfunktionen zu finden, die auf jedem Dreieck vollständig ohne zusätzliche Informationen benachbarter Dreiecke beschrieben werden können.

Zu diesem Zweck führen wir auf  $\mathcal{T}_h$  die Menge  $\mathcal{Z}$  der *stückweisen Interpolationsknoten*  $\mathbf{z}_i = (x_i, y_i)^T$ ,  $i = 1, \dots, N$ , ein und bezeichnen durch  $\mathbb{P}_k(\Omega)$ ,  $k \geq 0$ , den Raum der algebraischen Polynome vom Grade  $\leq k$  in den räumlichen Variablen  $x, y$

$$\mathbb{P}_k(\Omega) = \left\{ p(x, y) = \sum_{\substack{i, j=0 \\ i+j \leq k}}^k a_{ij} x^i y^j, \quad x, y \in \Omega \right\}. \quad (8.35)$$

Für  $k \geq 0$  sei schließlich  $\mathbb{P}_k^c(\Omega)$  der Raum stückweiser Polynome vom Grade  $\leq k$ , so dass für jedes  $p \in \mathbb{P}_k^c(\Omega)$  die Einschränkung  $p|_T$  in  $\mathbb{P}_k(T)$  für jedes  $T \in \mathcal{T}_h$  liegt. Eine einfache Basis für  $\mathbb{P}_k^c(\Omega)$  besteht aus den *Lagrangeschen charakteristischen Polynomen*  $l_i = l_i(x, y)$ , so dass  $l_i \in \mathbb{P}_k^c(\Omega)$  und

$$l_i(\mathbf{z}_j) = \delta_{ij}, \quad i, j = 1, \dots, N, \quad (8.36)$$

mit dem Kroneckersymbol  $\delta_{ij}$  gilt. Wir zeigen in Abbildung 8.7 die Funktionen  $l_i$ ,  $k = 0, 1$ , gemeinsam mit den eindimensionalen Entsprechungen. Im Fall  $k = 0$  sind die Interpolationsknoten in den *Schwerpunkten* der Dreiecke gelegen, wogegen die Knoten im Fall  $k = 1$  mit den *Ecken* der Dreiecke übereinstimmen. Diese Wahl, die wir im Folgenden beibehalten

wollen, ist nicht die einzig mögliche. Die Mittelpunkte der Dreiecksseiten könnten ebenso verwendet werden, was zu einer unstetigen stückweise polynomialen Interpolation über  $\Omega$  führen würde.

Für  $k \geq 0$  ist das *Lagrangesche stückweise interpolierende Polynom*  $\Pi_h^k f \in \mathbb{P}_k^c(\Omega)$  von  $f$  durch

$$\Pi_h^k f(x, y) = \sum_{i=1}^N f(\mathbf{z}_i) l_i(x, y) \quad (8.37)$$

definiert. Beachte, dass  $\Pi_h^0 f$  eine stückweise konstante Funktion ist, wegen  $\Pi_h^1 f$  eine lineare Funktion auf jedem Dreieck darstellt, die in den Ecken stetig und damit auch global stetig ist.

Für jedes  $T \in \mathcal{T}_h$  bezeichnen wir durch  $\Pi_T^k f$  die Einschränkung des stückweise interpolierenden Polynoms von  $f$  auf dem Element  $T$ . Nach Definition ist  $\Pi_T^k f \in \mathbb{P}_k(T)$ ; wegen  $d_k = \dim \mathbb{P}_k(T) = (k+1)(k+2)/2$  können wir somit schreiben

$$\Pi_T^k f(x, y) = \sum_{m=0}^{d_k-1} f(\tilde{\mathbf{z}}_T^{(m)}) l_{m,T}(x, y), \quad \forall T \in \mathcal{T}_h. \quad (8.38)$$

In (8.38) haben wir durch  $\tilde{\mathbf{z}}_T^{(m)}$ ,  $m = 0, \dots, d_k - 1$ , die stückweisen Interpolationsknoten auf  $T$  und durch  $l_{m,T}(x, y)$  die Einschränkung des Lagrangeschen charakteristischen Polynoms auf  $T$  bezeichnet, das in (8.37) den Index  $i$  besitzt, der in der Liste der "globalen" Knoten  $\mathbf{z}_i$  dem "lokalen" Knoten  $\tilde{\mathbf{z}}_T^{(m)}$  entspricht.

Fahren wir mit dieser Notation fort, haben wir  $l_{j,T}(\mathbf{x}) = \hat{l}_j \circ F_T^{-1}(\mathbf{x})$ , wobei  $\hat{l}_j = \hat{l}_j(\hat{\mathbf{x}})$ ,  $j = 0, \dots, d_k - 1$ , die  $j$ -te Lagrangesche Basisfunktion für  $\mathbb{P}_k(\hat{T})$  ist, die auf dem Referenzelement  $\hat{T}$  erzeugt wurde. Wir vermerken, dass wenn  $k = 0$  ist,  $d_0 = 1$  gilt, d.h. es existiert nur ein lokaler Interpolationsknoten (der mit dem Schwerpunkt des Dreiecks  $T$  übereinstimmt). Wohingegen  $d_1 = 3$  für  $k = 1$  gilt, d.h. drei lokale Interpolationsknoten existieren, die mit den Ecken von  $T$  übereinstimmen. In Abbildung 8.8 sind die lokalen Interpolationsknoten auf  $\hat{T}$  für  $k = 0, 1$  und  $2$  dargestellt.

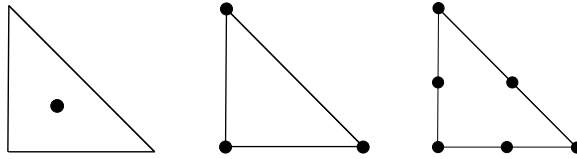


Abbildung 8.8. Lokale Interpolationsknoten auf  $\hat{T}$ ; links  $k = 0$ , Mitte  $k = 1$ , rechts  $k = 2$ .

Hinsichtlich der Interpolationsfehlerabschätzung gilt das folgende Resultat (zum Beweis siehe [CL91], Theorem 16.1, S. 125-126 und [QV94], Bemerkung 3.4.2, S. 89-90)

$$\|f - \Pi_T^k f\|_{\infty, T} \leq C h_T^{k+1} \|f^{(k+1)}\|_{\infty, T}, \quad k \geq 0. \quad (8.39)$$

Hierbei sei für jedes  $T \in \mathcal{T}_h$  durch  $h_T$  die maximale Kantenlänge von  $T$  bezeichnet und für jedes  $g \in C^0(T)$ ,  $\|g\|_{\infty, T} = \max_{\mathbf{x} \in T} |g(\mathbf{x})|$ . Ferner ist  $C$  in (8.39) eine positive Konstante unabhängig von  $h_T$  und  $f$ .

Wir wollen annehmen, dass die Triangulation  $\mathcal{T}_h$  *regulär* ist, d.h. dass es eine positive Konstante  $\sigma$  gibt, so dass

$$\max_{T \in \mathcal{T}_h} \frac{h_T}{\rho_T} \leq \sigma$$

gilt. Hier sei  $\rho_T$  für alle  $T \in \mathcal{T}_h$  der Durchmesser des  $T$  einbeschriebenen Kreises. Dann ist es möglich aus (8.39) die folgende Interpolationsfehlerabschätzung auf dem Gesamtgebiet  $\Omega$

$$\|f - \Pi_h^k f\|_{\infty, \Omega} \leq C h^{k+1} \|f^{(k+1)}\|_{\infty, \Omega}, \quad k \geq 0, \quad \forall f \in C^{k+1}(\Omega) \quad (8.40)$$

abzuleiten.

Die Theorie der stückweisen Interpolation ist ein grundlegendes Werkzeug der *finiten Elemente Methode*, einem numerischen Verfahren, das weit verbreitet bei der numerischen Approximation partieller Differentialgleichungen ist (siehe Kapitel 12 für den eindimensionalen Fall und [QV94] für eine vollständige Darstellung der Methode).

**Beispiel 8.7** Wir vergleichen die Konvergenz der stückweisen polynomialen Interpolation vom Grade 0, 1 und 2, der Funktion  $f(x, y) = e^{-(x^2+y^2)}$  auf  $\Omega = (-1, 1)^2$ . In Tabelle 8.4 sind der Fehler  $\mathcal{E}_k = \|f - \Pi_h^k f\|_{\infty, \Omega}$ , für  $k = 0, 1, 2$ , und die Konvergenzordnung  $p_k$  als Funktion der Netzweite  $h = 2/N$  für  $N = 2, \dots, 32$  dargestellt. Deutlich kann man lineare Konvergenz für die Interpolation vom Grade 0 erkennen, wohingegen die Konvergenz in Bezug auf  $h$  quadratisch für die Interpolation vom Grade 1 und kubisch für die Interpolation vom Grade 2 ist. •

Tabelle 8.4. Konvergenzraten und Ordnungen für stückweise Interpolation vom Grade 0, 1 und 2.

$h$	$\mathcal{E}_0$	$p_0$	$\mathcal{E}_1$	$p_1$	$\mathcal{E}_2$	$p_2$
1	0.4384		0.2387		0.016	
$\frac{1}{2}$	0.2931	0.5809	0.1037	1.2028	$1.6678 \cdot 10^{-3}$	3.2639
$\frac{1}{4}$	0.1579	0.8924	0.0298	1.7990	$2.8151 \cdot 10^{-4}$	2.5667
$\frac{1}{8}$	0.0795	0.9900	0.0077	1.9524	$3.5165 \cdot 10^{-5}$	3.001
$\frac{1}{16}$	0.0399	0.9946	0.0019	2.0189	$4.555 \cdot 10^{-6}$	2.9486

## 8.6 Splineapproximation

In diesem Abschnitt widmen wir uns der Approximation einer gegebenen Funktion unter Verwendung von *Splines*, die eine stückweise Interpolation bei globaler Glattheit ermöglicht.

**Definition 8.1** Seien  $n + 1$  verschiedene Knoten  $x_0, \dots, x_n$  in  $[a, b]$  mit  $a = x_0 < x_1 < \dots < x_n = b$  gegeben. Die auf dem Intervall  $[a, b]$  definierte Funktion  $s_k(x)$  ist ein *Spline* vom Grade  $k$  bezüglich der Knoten  $x_j$ , wenn

$$s_k|_{[x_j, x_{j+1}]} \in \mathbb{P}_k, \quad j = 0, 1, \dots, n-1, \quad (8.41)$$

$$s_k \in C^{k-1}[a, b]. \quad (8.42)$$

■

Bezeichnen wir durch  $\mathcal{S}_k$  den Raum der Splines  $s_k$  auf  $[a, b]$  bezüglich  $n + 1$  verschiedener Knoten, so ist  $\dim \mathcal{S}_k = n + k$ . Offensichtlich ist jedes Polynom vom Grade  $k$  auf  $[a, b]$  ein Spline; in der Praxis wird jedoch ein Spline durch verschiedene Polynome auf jedem Teilintervall dargestellt und kann daher unstetig in der  $k$ -ten Ableitung in einem inneren Knoten  $x_1, \dots, x_{n-1}$  sein. Die Knoten, in denen dies tatsächlich der Fall ist, heißen *aktive* Knoten.

Es ist einfach zu sehen, dass die Bedingungen (8.41) und (8.42) nicht ausreichen, um einen Spline vom Grade  $k$  zu charakterisieren. Die Restriktion  $s_{k,j} = s_k|_{[x_j, x_{j+1}]}$  kann in der Form

$$s_{k,j}(x) = \sum_{i=0}^k s_{ij}(x - x_j)^i, \quad \text{wenn } x \in [x_j, x_{j+1}] \quad (8.43)$$

dargestellt werden, so dass tatsächlich  $(k + 1)n$  Koeffizienten  $s_{ij}$  bestimmt werden müssen. Andererseits folgt aus (8.42)

$$s_{k,j-1}^{(m)}(x_j) = s_{k,j}^{(m)}(x_j), \quad j = 1, \dots, n-1, \quad m = 0, \dots, k-1$$

was das Stellen von  $k(n - 1)$  Bedingungen beinhaltet. Folglich verbleiben  $(k + 1)n - k(n - 1) = k + n$  Freiheitsgrade.

Auch wenn der Spline *interpolierend*, d.h. derart ist, dass  $s_k(x_j) = f_j$  für  $j = 0, \dots, n$  gilt, wobei  $f_0, \dots, f_n$  gegebene Werte sind, würden immer noch  $k - 1$  unbestimmte Freiheitsgrade auftreten. Aus diesem Grund werden weitere Bedingungen auferlegt, die auf

1. *periodische Splines* führen, wenn

$$s_k^{(m)}(a) = s_k^{(m)}(b), \quad m = 0, 1, \dots, k-1; \quad (8.44)$$

2. *natürliche Splines* führen, wenn für  $k = 2l - 1$  mit  $l \geq 2$

$$s_k^{(l+j)}(a) = s_k^{(l+j)}(b) = 0, \quad j = 0, 1, \dots, l-2, \quad (8.45)$$

gilt. Aus (8.43) ist ersichtlich, dass ein Spline bequem mit Hilfe von  $k + n$  Spline-Basisfunktionen dargestellt werden kann, so dass (8.42) automatisch erfüllt ist. Die einfachste Wahl, die aus der Verwendung einer geeignet angereicherten monomialen Basis (siehe Übung 10) besteht, ist vom numerischen Standpunkt ungeeignet, da sie schlecht konditioniert ist. In den Abschnitten 8.6.1 und 8.6.2 werden mögliche Beispiele von Spline-Basisfunktionen angegeben: Kardinal-Splines für den Spezialfall  $k = 3$  und B-Splines für ein allgemeines  $k$ .

### 8.6.1 Interpolierende kubische Splines

Interpolierende kubische Splines sind besonders wichtig, da: *i.* sie Splines minimalen Grades sind, die  $C^2$  Approximationen liefern; *ii.* sie hinreichend glatt beim Vorhandensein kleiner Krümmungen sind.

Betrachten wir also in  $[a, b]$  die  $n + 1$  geordneten Knoten  $a = x_0 < x_1 < \dots < x_n = b$  und die entsprechenden Auswertungen  $f_i$ ,  $i = 0, \dots, n$ . Unser Ziel ist es, ein effizientes Verfahren zur Konstruktion kubischer Splines, die jene Werte interpolieren, zu liefern. Da der Spline vom Grade 3 ist, müssen seine zweiten Ableitungen stetig sein. Wir wollen die folgende Notation einführen

$$f_i = s_3(x_i), \quad m_i = s'_3(x_i), \quad M_i = s''_3(x_i), \quad i = 0, \dots, n.$$

Da  $s_{3,i-1} \in \mathbb{P}_3$  ist  $s''_{3,i-1}$  linear und

$$s''_{3,i-1}(x) = M_{i-1} \frac{x_i - x}{h_i} + M_i \frac{x - x_{i-1}}{h_i} \quad \text{für } x \in [x_{i-1}, x_i], \quad (8.46)$$

wobei  $h_i = x_i - x_{i-1}$ ,  $i = 1, \dots, n$ . Zweifache Integration von (8.46) ergibt

$$s_{3,i-1}(x) = M_{i-1} \frac{(x_i - x)^3}{6h_i} + M_i \frac{(x - x_{i-1})^3}{6h_i} + C_{i-1}(x - x_{i-1}) + \tilde{C}_{i-1},$$

und die Konstanten  $C_{i-1}$  und  $\tilde{C}_{i-1}$  sind durch Aufprägen der Endpunktwerte  $s_3(x_{i-1}) = f_{i-1}$  und  $s_3(x_i) = f_i$  bestimmt. Dies ergibt für  $i = 1, \dots, n-1$

$$\tilde{C}_{i-1} = f_{i-1} - M_{i-1} \frac{h_i^2}{6}, \quad C_{i-1} = \frac{f_i - f_{i-1}}{h_i} - \frac{h_i}{6}(M_i - M_{i-1}).$$

Fordern wir nun die Stetigkeit der ersten Ableitungen in  $x_i$ , so erhalten wir

$$\begin{aligned} s'_3(x_i^-) &= \frac{h_i}{6} M_{i-1} + \frac{h_i}{3} M_i + \frac{f_i - f_{i-1}}{h_i} \\ &= -\frac{h_{i+1}}{3} M_i - \frac{h_{i+1}}{6} M_{i+1} + \frac{f_{i+1} - f_i}{h_{i+1}} = s'_3(x_i^+), \end{aligned}$$



wobei  $s'_3(x_i^\pm) = \lim_{t \rightarrow 0} s'_3(x_i \pm t)$ . Dies führt zu dem linearen Gleichungssystem (M-Stetigkeitssystem genannt)

$$\mu_i M_{i-1} + 2M_i + \lambda_i M_{i+1} = d_i \quad i = 1, \dots, n-1 \quad (8.47)$$

wobei wir

$$\mu_i = \frac{h_i}{h_i + h_{i+1}}, \quad \lambda_i = \frac{h_{i+1}}{h_i + h_{i+1}},$$

$$d_i = \frac{6}{h_i + h_{i+1}} \left( \frac{f_{i+1} - f_i}{h_{i+1}} - \frac{f_i - f_{i-1}}{h_i} \right), \quad i = 1, \dots, n-1,$$

gesetzt haben. Das System (8.47) hat  $n+1$  Unbekannte und  $n-1$  Gleichungen; somit sind noch  $2(=k-1)$  Bedingungen offen. Im Allgemeinen können diese Bedingungen von der Gestalt

$$2M_0 + \lambda_0 M_1 = d_0, \quad \mu_n M_{n-1} + 2M_n = d_n,$$

mit  $0 \leq \lambda_0, \mu_n \leq 1$  und  $d_0, d_n$  gegebene Werte sein. Um zum Beispiel die natürlichen Splines zu erhalten, (die  $s''_3(a) = s''_3(b) = 0$  genügen,) müssen wir die obigen Koeffizienten gleich Null setzen. Eine populäre Wahl ist  $\lambda_0 = \mu_n = 1$  und  $d_0 = d_1, d_n = d_{n-1}$ , die der Fortsetzung der Splines über die Endpunkte des Intervalls hinaus und der Behandlung von  $a$  und  $b$  als innere Punkte entspricht. Diese Strategie führt zu einem Spline mit "glattem" Verhalten. In allgemeinen ist das resultierende lineare System tridiagonal von der Form

$$\begin{bmatrix} 2 & \lambda_0 & 0 & \dots & 0 \\ \mu_1 & 2 & \lambda_1 & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & \mu_{n-1} & 2 & \lambda_{n-1} \\ 0 & \dots & 0 & \mu_n & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ \vdots \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix} \quad (8.48)$$

und kann effizient mit dem Thomas-Verfahren (3.53) gelöst werden (vgl. Band 1).

Eine Abschlussbedingung für das System (8.48), die nützlich sein kann, wenn die Ableitungen  $f'(a)$  und  $f'(b)$  nicht bekannt sind, besteht darin, die Stetigkeit von  $s'''_3(x)$  in  $x_1$  und  $x_{n-1}$  zu fordern. Da die Knoten  $x_1$  und  $x_{n-1}$  nicht wirklich zur Konstruktion des kubischen Splines beitragen, wird er als ein *nicht-ein-Knotenspline* bezeichnet. Dieser hat die "aktiven" Knoten  $\{x_0, x_2, \dots, x_{n-2}, x_n\}$  und interpoliert  $f$  in allen Knoten  $\{x_0, x_1, x_2, \dots, x_{n-2}, x_{n-1}, x_n\}$ .

**Bemerkung 8.2 (Spezielle Software)** Verschiedene Pakete existieren für die Splineinterpolation. Im Fall kubischer Splines erwähnen wir den

Befehl `spline`, der die oben eingeführte *nicht-ein-Knoten* Bedingung verwendet, oder im Allgemeinen, die `Spline-Toolbox` von MATLAB [dB90] und die Bibliothek FITPACK [Die87a], [Die87b]. ■

Ein völlig anderer Zugang zur Erzeugung von  $s_3$  besteht in der Angabe einer Basis  $\{\varphi_i\}$  für den Raum  $\mathcal{S}_3$  kubischer Splines, dessen Dimension  $n+3$  ist. Wir betrachten hier den Fall, in dem die  $n+3$  Basisfunktionen  $\varphi_i$  globalen Träger im Intervall  $[a, b]$  haben und verweisen auf Abschnitt 8.6.2 für den Fall einer Basis mit lokalem Träger.

Die Funktionen  $\varphi_i$ ,  $i, j = 0, \dots, n$ , werden durch die folgenden Interpolationsbedingungen festgelegt

$$\varphi_i(x_j) = \delta_{ij}, \quad \varphi'_i(x_0) = \varphi'_i(x_n) = 0,$$

und zwei weitere Splines,  $\varphi_{n+1}$  und  $\varphi_{n+2}$ , müssen hinzugefügt werden. Wenn zum Beispiel der Spline gewissen vorgeschriebenen Ableitungsbedingungen in den Endpunkten genügen soll, fordern wir, dass

$$\begin{aligned} \varphi_{n+1}(x_j) &= 0, & j = 0, \dots, n & \quad \varphi'_{n+1}(x_0) = 1, \quad \varphi'_{n+1}(x_n) = 0, \\ \varphi_{n+2}(x_j) &= 0, & j = 0, \dots, n & \quad \varphi'_{n+2}(x_0) = 0, \quad \varphi'_{n+2}(x_n) = 1. \end{aligned}$$

Auf diese Weise nimmt der Spline die Form

$$s_3(x) = \sum_{i=0}^n f_i \varphi_i(x) + f'_0 \varphi_{n+1}(x) + f'_n \varphi_{n+2}(x)$$

an, wobei  $f'_0$  und  $f'_n$  zwei gegebene Werte sind. Die resultierende Basis  $\{\varphi_i, i = 0, \dots, n+2\}$  wird *kardinale Splinebasis* genannt und häufig bei der numerischen Lösung von Differential- oder Integralgleichungen verwendet.

Abbildung 8.9 zeigt einen allgemeinen Kardinal-Spline, der über praktisch unbeschränktes Intervall berechnet wurde, wobei die Interpolationsknoten  $x_j$  ganze Zahlen sind. Der Spline wechselt das Vorzeichen in jeweils benachbarten Intervallen  $[x_{j-1}, x_j]$  und  $[x_j, x_{j+1}]$  und klingt rasch auf Null ab.

Beschränken wir uns auf die positive Achse, so kann gezeigt werden (siehe [SL89]), dass die Extremante der Funktion auf dem Intervall  $[x_j, x_{j+1}]$  gleich der Extremanten auf dem Intervall  $[x_{j+1}, x_{j+2}]$  multipliziert mit einem Abklingfaktor  $\lambda \in (0, 1)$  ist. Auf diese Weise werden mögliche auf einem Intervall auftretende Fehler schnell auf dem benachbarten gedämpft, was die Stabilität des Algorithmus sichert.

Wir wollen nun die Haupteigenschaften interpolierender kubischer Splines zusammenfassen und verweisen für die Beweise und allgemeinere Ergebnisse auf [Sch81] und [dB83].

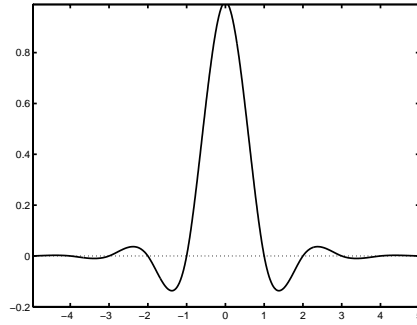


Abbildung 8.9. Kardinal-Spline.

**Eigenschaft 8.2** Seien  $f \in C^2([a, b])$  und  $s_3$  der  $f$  interpolierende natürliche kubische Spline. Dann ist

$$\int_a^b [s_3''(x)]^2 dx \leq \int_a^b [f''(x)]^2 dx, \quad (8.49)$$

wobei das Gleichheitszeichen genau für  $f = s_3$  gilt.

Das obige Resultat ist als die *minimale Normeigenschaft* bekannt und hat die Bedeutung des minimalen Energieprinzips in der Mechanik. Die Eigenschaft (8.49) gilt auch, wenn die Bedingungen für die ersten Ableitungen des Splines in den Endpunkten anstelle der natürlichen Bedingungen auferlegt werden (in solch einem Fall heißt der Spline *gebunden*, siehe Übung 11)

Der interpolierende kubische Spline  $s_f$  einer Funktion  $f \in C^2([a, b])$ , mit  $s_f'(a) = f'(a)$  und  $s_f'(b) = f'(b)$ , genügt auch der folgenden Eigenschaft

$$\int_a^b [f''(x) - s_f''(x)]^2 dx \leq \int_a^b [f''(x) - s''(x)]^2 dx, \quad \forall s \in S_3.$$

In Bezug auf die Fehlerabschätzung gilt folgendes Ergebnis.

**Eigenschaft 8.3** Sei  $f \in C^4([a, b])$  und fixiere eine Zerlegung von  $[a, b]$  in Teilintervalle der Breite  $h_i$  derart, dass  $h = \max_i h_i$  und  $\beta = h / \min_i h_i$ . Sei  $s_3$  der  $f$  interpolierende kubische Spline. Dann gilt

$$\|f^{(r)} - s_3^{(r)}\|_\infty \leq C_r h^{4-r} \|f^{(4)}\|_\infty, \quad r = 0, 1, 2, 3, \quad (8.50)$$

mit  $C_0 = 5/384$ ,  $C_1 = 1/24$ ,  $C_2 = 3/8$  und  $C_3 = (\beta + \beta^{-1})/2$ .

Somit konvergieren für  $h$  gegen Null der Spline  $s_3$  sowie seine ersten und zweiten Ableitungen gleichmäßig gegen  $f$  und deren Ableitungen. Die dritten Ableitungen konvergieren ebenso, vorausgesetzt, dass  $\beta$  gleichmäßig beschränkt ist.

**Beispiel 8.8** Abbildung 8.10 zeigt den kubischen Spline, der die Funktion im Runge-Beispiel approximiert, sowie seine Ableitungen erster, zweiter und dritter Ordnung auf einem Gitter von 11 äquidistanten Knoten. In Tabelle 8.5 wird der Fehler  $\|s_3 - f\|_\infty$  als Funktion von  $h$  zusammen mit der berechneten Konvergenzordnung angegeben. Die Ergebnisse zeigen deutlich, dass  $p$  gegen 4 (der theoretischen Ordnung) konvergiert, wenn  $h$  gegen Null geht. •

Tabelle 8.5. Experimentell bestimmter Interpolationsfehler für die Runge-Funktion unter Verwendung kubischer Splines.

$h$	1	0.5	0.25	0.125	0.0625
$\ s_3 - f\ _\infty$	0.022	0.0032	2.7741e-4	1.5983e-5	9.6343e-7
$p$	—	2.7881	3.5197	4.1175	4.0522

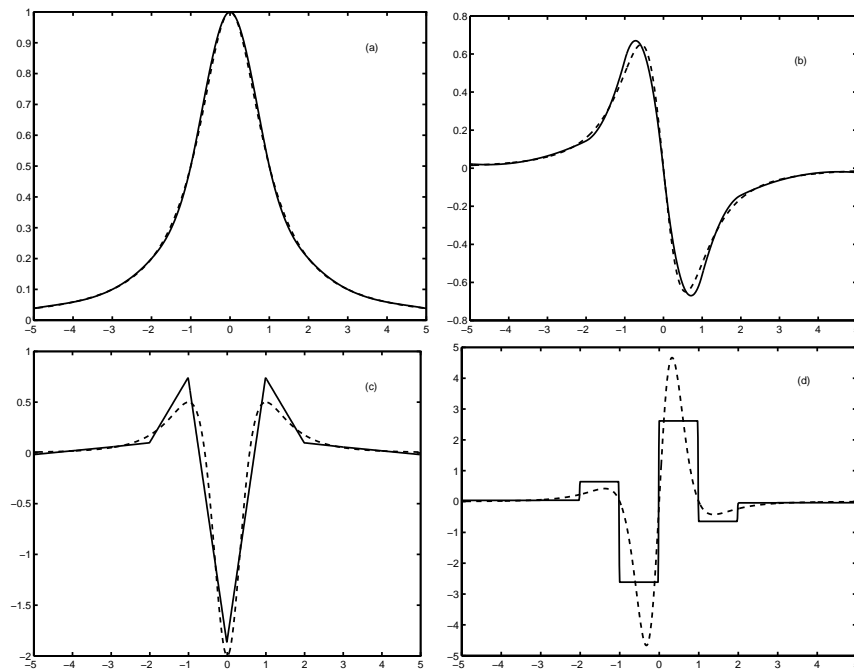


Abbildung 8.10. Interpolierender Spline (a) und seine Ableitung erster (b), zweiter (c) sowie dritter (d) Ordnung (als durchgezogene Kurve) für die Funktion des Beispiels von Runge (als gestrichelte Kurve).

### 8.6.2 B-Splines

Kehren wir zu Splines allgemeinen Grades  $k$  zurück und betrachten die B-Spline (oder engl.: *bell-spline*) Basis, die sich auf die in Abschnitt 8.2.1 eingeführten dividierten Differenzen bezieht.

**Definition 8.2** Der *normalisierte B-Spline*  $B_{i,k+1}$  vom Grade  $k$  bezogen auf die verschiedenen Knoten  $x_i, \dots, x_{i+k+1}$  ist definiert als

$$B_{i,k+1}(x) = (x_{i+k+1} - x_i)g[x_i, \dots, x_{i+k+1}], \quad (8.51)$$

wobei

$$g(t) = (t - x)_+^k = \begin{cases} (t - x)^k & \text{wenn } x \leq t, \\ 0 & \text{andernfalls.} \end{cases} \quad (8.52)$$

■

Substitution von (8.18) in (8.51) liefert die explizite Darstellung

$$B_{i,k+1}(x) = (x_{i+k+1} - x_i) \sum_{j=0}^{k+1} \frac{(x_{j+i} - x)_+^k}{\prod_{\substack{l=0 \\ l \neq j}}^{k+1} (x_{i+j} - x_{i+l})}. \quad (8.53)$$

Aus (8.53) folgt, dass die aktiven Knoten von  $B_{i,k+1}(x)$  gerade die Punkte  $x_i, \dots, x_{i+k+1}$  sind, und dass  $B_{i,k+1}(x)$  nur innerhalb des Intervalls  $[x_i, x_{i+k+1}]$  von Null verschieden ist.

Tatsächlich kann gezeigt werden, dass es der einzige nicht verschwindende Spline mit minimalem Träger bezogen auf die Knoten  $x_i, \dots, x_{i+k+1}$  ist [Sch67]. Es kann auch gezeigt werden, dass  $B_{i,k+1}(x) \geq 0$  [dB83] und  $|B_{i,k+1}^{(l)}(x_i)| = |B_{i,k+1}^{(l)}(x_{i+k+1})|$  für  $l = 0, \dots, k-1$  gilt [Sch81]. B-Splines erlauben die rekursive Darstellung ([dB72], [Cox72])

$$B_{i,1}(x) = \begin{cases} 1 & \text{wenn } x \in [x_i, x_{i+1}], \\ 0 & \text{andernfalls,} \end{cases} \quad (8.54)$$

$$B_{i,k+1}(x) = \frac{x - x_i}{x_{i+k} - x_i} B_{i,k}(x) + \frac{x_{i+k+1} - x}{x_{i+k+1} - x_{i+1}} B_{i+1,k}(x), \quad k \geq 1,$$

die gewöhnlich gegenüber (8.53) favorisiert wird, wenn ein B-Spline in einem gegebenen Punkt auszuwerten ist.

**Bemerkung 8.3** Es ist möglich, B-Splines sogar im Fall teilweise übereinstimmender Knoten zu definieren, wenn die Definition dividierten Differenzen geeignet erweitert wird. Dies führt auf eine neue rekursive Form

Newtonscher dividierter Differenzen, die gegeben ist durch (für weitere Details siehe [Die93])

$$f[x_0, \dots, x_n] = \begin{cases} \frac{f[x_1, \dots, x_n] - f[x_0, \dots, x_{n-1}]}{x_n - x_0} & \text{für } x_0 < x_1 < \dots < x_n \\ \frac{f^{(n+1)}(x_0)}{(n+1)!} & \text{für } x_0 = x_1 = \dots = x_n. \end{cases}$$

Angenommen  $m$  (mit  $1 < m < k+2$ ) der  $k+2$  Knoten  $x_i, \dots, x_{i+k+1}$  stimmen überein und sind gleich  $\lambda$ , dann wird (8.46) eine Linearkombination der Funktionen  $(\lambda - x)_+^{k+1-j}$ ,  $j = 1, \dots, m$ , enthalten. Folglich kann der B-Spline stetige Ableitungen in  $\lambda$  nur bis zur Ordnung  $k-m$  besitzen, und ist daher unstetig, wenn  $m = k+1$  gilt. Es läßt sich zeigen [Die93], dass für  $x_{i-1} < x_i = \dots = x_{i+k} < x_{i+k+1}$

$$B_{i,k+1}(x) = \begin{cases} \left( \frac{x_{i+k+1} - x}{x_{i+k+1} - x_i} \right)^k & \text{wenn } x \in [x_i, x_{i+k+1}], \\ 0 & \text{andernfalls,} \end{cases}$$

während für  $x_i < x_{i+1} = \dots = x_{i+k+1} < x_{i+k+2}$

$$B_{i,k+1}(x) = \begin{cases} \left( \frac{x - x_i}{x_{i+k+1} - x_i} \right)^k & \text{wenn } x \in [x_i, x_{i+k+1}], \\ 0 & \text{andernfalls} \end{cases}$$

gilt. Die Kombination dieser Formeln mit der rekursiven Beziehung (8.54) ermöglicht die Konstruktion von B-Splines mit zusammenfallenden Knoten. ■

**Beispiel 8.9** Untersuchen wir den Spezialfall kubischer Splines auf äquidistanten Knoten  $x_{i+1} = x_i + h$ ,  $i = 0, \dots, n-1$ . Gleichung (8.53) wird

$$6h^3 B_{i,4}(x) = \begin{cases} (x - x_i)^3, & \text{wenn } x \in [x_i, x_{i+1}], \\ h^3 + 3h^2(x - x_{i+1}) + 3h(x - x_{i+1})^2 - 3(x - x_{i+1})^3, & \text{wenn } x \in [x_{i+1}, x_{i+2}], \\ h^3 + 3h^2(x_{i+3} - x) + 3h(x_{i+3} - x)^2 - 3(x_{i+3} - x)^3, & \text{wenn } x \in [x_{i+2}, x_{i+3}], \\ (x_{i+4} - x)^3, & \text{wenn } x \in [x_{i+3}, x_{i+4}], \\ 0 & \text{andernfalls.} \end{cases}$$

In Abbildung 8.11 ist der Graph von  $B_{i,4}$  im Fall verschiedener und teilweise zusammenfallender Knoten dargestellt. •

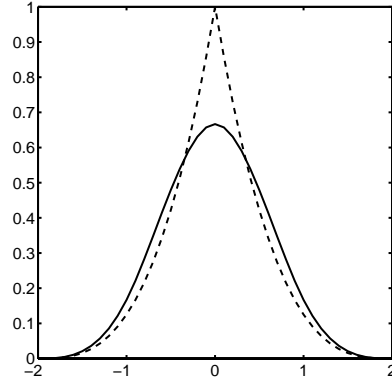


Abbildung 8.11. B-Spline mit verschiedenen Knoten (als durchgezogene Kurve) und mit drei im Ursprung zusammenfallenden Knoten (als gestrichelte Kurve). Beachte die Unstetigkeit der ersten Ableitung.

Sind  $n+1$  verschiedene Knoten  $x_j$ ,  $j = 0, \dots, n$  gegeben, so können  $n-k$  linear unabhängige B-Splines vom Grade  $k$  konstruiert werden, obwohl  $2k$  Freiheitsgrade noch verfügbar sind, um eine Basis für  $\mathcal{S}_k$  zu erzeugen. Ein Weg des Vorgehens besteht darin,  $2k$  fiktive Knoten

$$\begin{aligned} x_{-k} &\leq x_{-k+1} \leq \dots \leq x_{-1} \leq x_0 = a, \\ b &= x_n \leq x_{n+1} \leq \dots \leq x_{n+k} \end{aligned} \quad (8.55)$$

einzuführen, denen die B-Splines  $B_{i,k+1}$ , mit  $i = -k, \dots, -1$  und  $i = n-k, \dots, n-1$ , zugeordnet sind. Auf diese Weise kann jeder Spline  $s_k \in \mathcal{S}_k$  eindeutig in der Form

$$s_k(x) = \sum_{i=-k}^{n-1} c_i B_{i,k+1}(x) \quad (8.56)$$

dargestellt werden. Die reellen Zahlen  $c_i$  sind die *B-Splinekoeffizienten* von  $s_k$ . Die Knoten (8.55) werden gewöhnlich zusammenfallend oder periodisch gewählt.

1. *Zusammenfallend*: diese Wahl ist geeignet, um die Werte die ein Spline am Ende des Definitionsintervalls annimmt, zu erzwingen. In diesem Fall bekommen wir dank Bemerkung 8.3 über B-Splines mit zusammenfallenden Knoten in der Tat

$$s_k(a) = c_{-k}, \quad s_k(b) = c_{n-1}. \quad (8.57)$$

2. *Periodisch*, d.h.

$$x_{-i} = x_{n-i} - b + a, \quad x_{i+n} = x_i + b - a, \quad i = 1, \dots, k.$$

Diese Wahl ist nützlich, wenn Periodizitätsbedingungen (8.44) auferlegt werden müssen.

**Bemerkung 8.4 (Einfügen von Knoten)** B-Splines anstelle von Kardinal-Splines zu verwenden ist vorteilhaft, wenn mit reduziertem Aufwand eine gegebene Konfiguration von Knoten, für die ein Spline  $s_k$  bekannt ist, verändert werden soll. Nehmen wir an, dass die Koeffizienten  $c_i$  von  $s_k$  (in der Form (8.56)) über den Knoten  $x_{-k}, x_{-k+1}, \dots, x_{n+k}$  bekannt sind, und dass wir zu diesen einen neuen Knoten  $\tilde{x}$  hinzufügen wollen.

Der Spline  $\tilde{s}_k \in \mathcal{S}_k$ , der über der neuen Knotenmenge definiert ist, kann in Bezug auf eine neue B-Spline-Basis  $\{\tilde{B}_{i,k+1}\}$  in der Form

$$\tilde{s}_k(x) = \sum_{i=-k}^{n-1} d_i \tilde{B}_{i,k+1}(x)$$

dargestellt werden. Die neuen Koeffizienten  $d_i$  lassen sich aus den bekannten Koeffizienten  $c_i$  unter Verwendung des folgenden Algorithmus berechnen [Boe80]:

sei  $\tilde{x} \in [x_j, x_{j+1})$ ; dann konstruiere eine neue Knotenmenge  $\{y_i\}$ , so dass

$$\begin{aligned} y_i &= x_i \text{ für } i = -k, \dots, j, & y_{j+1} &= \tilde{x}, \\ y_i &= x_{i-1} \text{ für } i = j+2, \dots, n+k+1; \end{aligned}$$

definiere

$$\omega_i = \begin{cases} 1 & \text{für } i = -k, \dots, j-k, \\ \frac{y_{j+1} - y_i}{y_{i+k+1} - y_i} & \text{für } i = j-k+1, \dots, j, \\ 0 & \text{für } i = j+1, \dots, n; \end{cases}$$

berechne

$$d_i = \omega_i c_i + (1 - \omega_i) c_i \quad i = -k, \dots, n-1.$$

Dieser Algorithmus besitzt gute Stabilitätseigenschaften und kann auf den Fall verallgemeinert werden, in dem mehr als ein Knoten gleichzeitig eingefügt wird (siehe [Die93]). ■

## 8.7 Splines in parametrischer Form

Die Verwendung interpolierender Splines besitzt die beiden nachfolgend genannten Nachteile:



1. die resultierende Approximation ist nur dann von guter Qualität, wenn die Funktion  $f$  keine großen Ableitungen besitzt (insbesondere fordern wir, dass  $|f'(x)| < 1$  für jedes  $x$ ). Anderenfalls kann oszillatorisches Verhalten im Spline auftreten, wie durch das in Abbildung 8.12 betrachtete Beispiel demonstriert wird. Diese Beispiel zeigt als durchgezogene Kurve den interpolierenden kubischen Spline auf der folgenden Datenmenge (aus [SL89])

$x_i$	8.125	8.4	9	9.845	9.6	9.959	10.166	10.2
$f_i$	0.0774	0.099	0.28	0.6	0.708	1.3	1.8	2.177

2.  $s_k$  hängt von der Wahl des Koordinatensystems ab. Führt man in dem obigen Beispiel eine Rotation des Koordinatensystems um 36 Grad in Uhrzeigerrichtung aus, würde man tatsächlich zu dem Spline ohne fadenscheinige Oszillationen gelangen, der in der umrahmten Box in Abbildung 8.12 dargestellt ist.

Alle Interpolationsverfahren, die bislang betrachtet wurden, hängen vom gewählten kartesischen Referenzsystem ab, was ein negatives Kennzeichen ist, wenn der Spline zur grafischen Darstellung einer gegebenen Figur (z.B. einer Ellipse) benutzt werden soll. Wir würden gern eine solche Darstellung haben, die unabhängig vom Referenzsystem ist, d.h. die einer geometrischen Invarianzeigenschaft genügt.

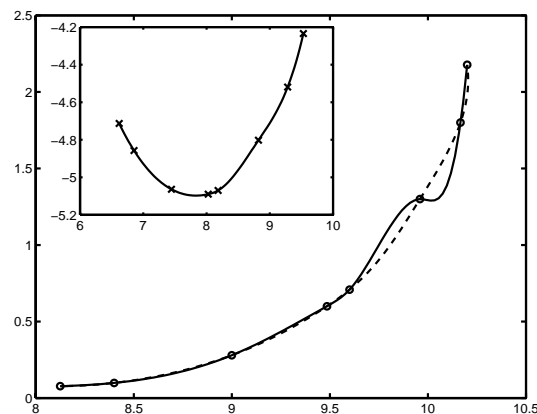


Abbildung 8.12. Geometrische Nichtinvarianz eines interpolierenden kubischen Splines  $s_3$ : die Datenmenge für  $s_3$  in der gerahmten Box ist die gleiche wie im Hauptbild, rotiert um 36 Grad. Die Rotation verringert die Steigung der interpolierten Kurve und eliminiert jede Oszillation von  $s_3$ . Beachte, dass die Verwendung eines parametrischen Splines (gestrichelte Kurve) die Oszillationen in  $s_3$  ohne jede Rotation des Referenzsystems entfernt.

Eine Lösung ist durch parametrische Splines gegeben, bei denen jede Komponente der in parametrischer Form geschriebenen Kurve durch eine Spline-Funktion approximiert wird. Betrachte eine ebene Kurve in parametrischer Form  $\mathbf{P}(t) = (x(t), y(t))$ ,  $t \in [0, T]$ , nimm dann die Menge von Punkten in der Ebene der Koordinaten  $\mathbf{P}_i = (x_i, y_i)$ ,  $i = 0, \dots, n$ , und führe eine Zerlegung auf  $[0, T]$ :  $0 = t_0 < t_1 < \dots < t_n = T$  ein.

Verwenden wir die beiden Mengen von Werten  $\{t_i, x_i\}$  und  $\{t_i, y_i\}$  als Interpolationsdaten, erhalten wir die beiden Splines  $s_{k,x}$  und  $s_{k,y}$ , in Abhängigkeit von der unabhängigen Variablen  $t$ , die  $x(t)$  bzw.  $y(t)$  interpolieren. Die parametrische Kurve  $\mathbf{S}_k(t) = (s_{k,x}(t), s_{k,y}(t))$  wird *parametrischer Spline* genannt. Offensichtlich liefern verschiedene Parametrisierungen des Intervalls  $[0, T]$  verschiedene Splines (siehe Abbildung 8.13).

Eine vernünftige Wahl der Parametrisierung verwendet die Länge jedes Segmentes  $\mathbf{P}_{i-1}\mathbf{P}_i$ ,

$$l_i = \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}, \quad i = 1, \dots, n.$$

Setzt man  $t_0 = 0$  und  $t_i = \sum_{k=1}^i l_k$  für  $i = 1, \dots, n$ , so stellt jedes  $t_i$  die aufsummierte Länge des Polygonzuges dar, der die Punkte von  $\mathbf{P}_0$  nach  $\mathbf{P}_i$  verbindet. Diese Funktion heißt *kumulativer Längenspline* und approximiert sogar Kurven mit großer Krümmung zufriedenstellend.

Darüber hinaus kann auch bewiesen werden (siehe [SL89]), dass er geometrisch invariant ist.

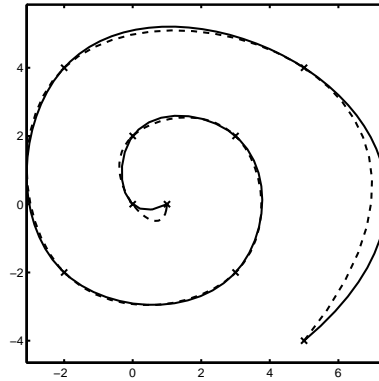


Abbildung 8.13. Parametrische Splines für eine spiralförmige Knotenverteilung. Der Spline kumulativer Länge ist als durchgezogene Kurve dargestellt.

Das Programm 68 implementiert die Konstruktion kumulativer parametrischer kubischer Splines in zwei Dimensionen (und kann leicht auf den dreidimensionalen Fall verallgemeinert werden). Zusammengesetzte parametrische Splines können ebenso durch Aufprägen geeigneter Stetigkeitsbedingungen erzeugt werden (siehe [SL89]).

**Program 68 - par\_spline** : Parametrische Splines

```

function [xi,yi] = par_spline (x, y)
t (1) = 0;
for i = 1:length (x)-1
    t (i+1) = t (i) + sqrt ( (x(i+1)-x(i))^2 + (y(i+1)-y(i))^2 );
end
z = [t(1):(t(length(t))-t(1))/100:t(length(t))];
xi = spline (t,x,z);
yi = spline (t,y,z);

```

**8.7.1 Bézier-Kurven und parametrische B-Splines**

Die Bézier-Kurven und parametrische B-Splines sind bei grafischen Anwendungen weit verbreitet, wobei die Lage der Knoten durch gewisse Unsicherheiten beeinflusst sein kann.

Seien  $\mathbf{P}_0, \mathbf{P}_1, \dots, \mathbf{P}_n$ ,  $n+1$  geordnete Punkte in der Ebene. Das durch sie gebildete orientierte Polygon heißt das *charakteristische Polygon* oder *Bézier-Polygon*. Wir wollen die Bernstein-Polynome auf dem Intervall  $[0, 1]$  einführen, die als

$$b_{n,k}(t) = \binom{n}{k} t^k (1-t)^{n-k} = \frac{n!}{k!(n-k)!} t^k (1-t)^{n-k},$$

für  $n = 0, 1, \dots$  und  $k = 0, \dots, n$  definiert sind. Sie können durch die Rekursionsformeln

$$\begin{cases} b_{n,0}(t) = (1-t)^n \\ b_{n,k}(t) = (1-t)b_{n-1,k}(t) + tb_{n-1,k-1}(t), \quad k = 1, \dots, n, \quad t \in [0, 1] \end{cases}$$

erhalten werden. Es ist leicht zu sehen, dass  $b_{n,k} \in \mathbb{P}_n$  für  $k = 0, \dots, n$  gilt. Ferner stellt  $\{b_{n,k}, k = 0, \dots, n\}$  eine Basis für  $\mathbb{P}_n$  dar. Die Bézier-Kurve ist wie folgt definiert

$$B_n(\mathbf{P}_0, \mathbf{P}_1, \dots, \mathbf{P}_n, t) = \sum_{k=0}^n \mathbf{P}_k b_{n,k}(t), \quad 0 \leq t \leq 1. \quad (8.58)$$

Dieser Ausdruck kann als gewichtetes Mittel der Punkte  $\mathbf{P}_k$  mit den Gewichten  $b_{n,k}(t)$  aufgefasst werden.

Die Bézier-Kurven können auch durch einen rein geometrischen Zugang beginnend mit dem charakteristischen Polygon erhalten werden. Für jedes feste  $t \in [0, 1]$  definieren wir  $\mathbf{P}_{i,1}(t) = (1-t)\mathbf{P}_i + t\mathbf{P}_{i+1}$  für  $i = 0, \dots, n-1$  und, für festes  $t$ , bilden die stückweise, die neuen Knoten  $\mathbf{P}_{i,1}(t)$  verbindenden Linien eine Polygonzug von  $n-1$  Ecken. Wir können nun dieses

Verfahren durch die Erzeugung neuer Ecken  $\mathbf{P}_{i,2}(t)$  ( $i = 0, \dots, n-2$ ) wiederholen und brechen ab, sobald das Polygon nur aus den Ecken  $\mathbf{P}_{0,n-1}(t)$  und  $\mathbf{P}_{1,n-1}(t)$  besteht. Es lässt sich zeigen, dass

$$\mathbf{P}_{0,n}(t) = (1-t)\mathbf{P}_{0,n-1}(t) + t\mathbf{P}_{1,n-1}(t) = B_n(\mathbf{P}_0, \mathbf{P}_1, \dots, \mathbf{P}_n, t),$$

d.h.  $\mathbf{P}_{0,n}(t)$  ist gleich dem Funktionswert der Bézier-Kurve  $B_n$  in den Punkten, die festen Werten von  $t$  entsprechen. Wiederholung dieses Prozesses für verschiedene Werte des Parameters  $t$  ergibt die Konstruktion der Kurve in dem betrachteten Bereich der Ebene.

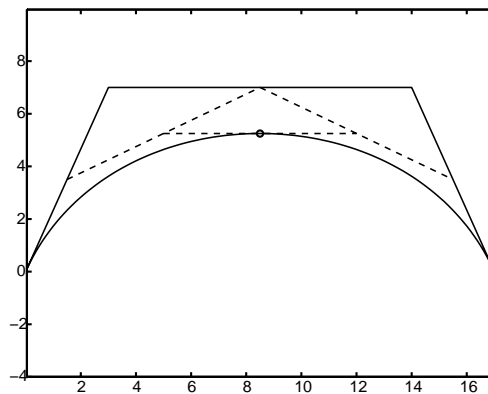


Abbildung 8.14. Berechnung des Wertes von  $B_3$  bezüglich der Punkte  $(0,0)$ ,  $(3,7)$ ,  $(14,7)$ ,  $(17,0)$  für  $t = 0.5$  unter Verwendung der im Text beschriebenen grafischen Methode.

Beachte, dass bei gegebener Knotenkonfiguration entsprechend der Reihenfolge der Punkte  $\mathbf{P}_i$  verschiedene Kurven konstruiert werden können. Darüber hinaus stimmt die Bézier-Kurve  $B_n(\mathbf{P}_0, \mathbf{P}_1, \dots, \mathbf{P}_n, t)$  abgesehen von der Orientierung mit  $B_n(\mathbf{P}_n, \mathbf{P}_{n-1}, \dots, \mathbf{P}_0, t)$  überein.

Das Programm 69 berechnet  $b_{n,k}$  im Punkt  $x$  für  $x \in [0, 1]$ .

#### Program 69 - bernstein : Bernstein-Polynome

```
function [bnk]=bernstein (n,k,x)
if k == 0,
    C = 1;
else,
    C = prod ([1:n]) / ( prod([1:k])*prod([1:n-k]));
end
bnk = C * x^k * (1-x)^(n-k);
```

Das Programm 70 zeichnet die Bézier-Kurve die sich auf die Menge von Punkten  $(x, y)$  bezieht.

**Program 70 - bezier** : Bézier-Kurven

```

function [bezx,bezy] = bezier (x, y, n)
i = 0; k = 0;
for t = 0:0.01:1,
    i = i + 1; bnk = bernstein (n,k,t); ber(i) = bnk;
end
bezx = ber * x (1); bezy = ber * y (1);
for k = 1:n
    i = 0;
    for t = 0:0.01:1
        i = i + 1; bnk = bernstein (n,k,t); ber(i) = bnk;
    end
    bezx = bezx + ber * x (k+1); bezy = bezy + ber * y (k+1);
end
plot(bezx,bezy)

```

In der Praxis werden die Bézier-Kurven selten verwendet, da sie keine genügend genaue Approximation des charakteristischen Polygons liefern. Aus diesem Grunde wurden in den 70er Jahren die *parametrischen B-Splines* eingeführt und in (8.58) anstelle der Bernstein-Polynome verwendet. Parametrische B-Splines sind in Paketen für Computer Grafiken weit verbreitet, da sie die folgenden Eigenschaften besitzen:

1. die Störung einer einzelnen Ecke des charakteristischen Polygons liefert nur eine lokale Störung der Kurve um die Ecke selbst;
2. der parametrische B-Spline approximiert das Kontrollpolygon besser als die entsprechende Bézier-Kurve es tut, und er ist stets in der konvexen Hülle des Polygon enthalten.

In Abbildung 8.15 werden Bézier-Kurven und parametrische B-Splines für die Approximation eines gegebenen charakteristischen Polygons verglichen.

Wir beenden diesen Abschnitt mit der Bemerkung, dass parametrische kubische B-Splines durch die geradlinige Ausrichtung von vier aufeinanderfolgenden Punkten lokal gerade Linien ermöglichen (siehe Abbildung 8.16), und dass ein parametrischer B-Spline durch einen speziellen Punkt des charakteristischen Polygons geht, wenn drei aufeinanderfolgende Punkte des Polygons mit dem gewünschten Punkt übereinstimmen.

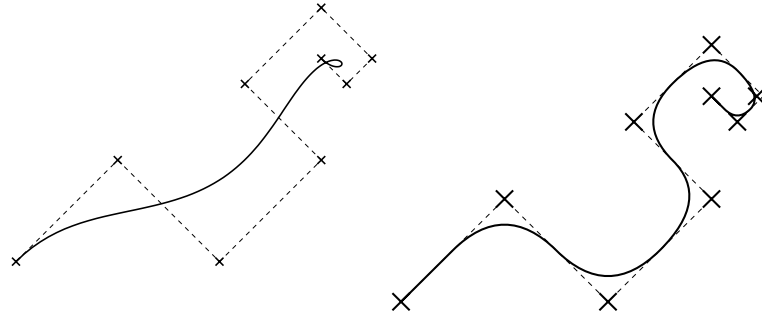


Abbildung 8.15. Vergleich einer Bézier-Kurve (links) mit einem parametrischen B-Spline (rechts). Die Ecken des charakteristischen Polygons sind durch  $\times$  markiert.

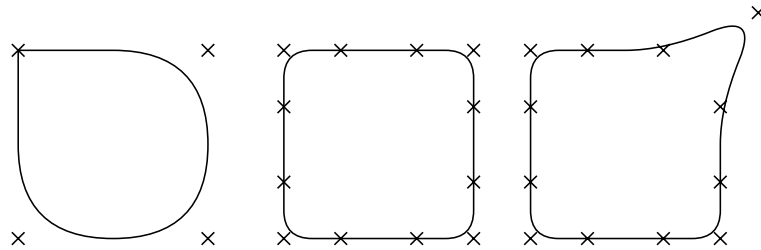


Abbildung 8.16. Einige parametrische B-splines als Funktion der Anzahl und der Lage der Ecken des charakteristischen Polygons. Beachte in der letzten Abbildung (rechts) die Lokalisierungseffekte, die durch die Bewegung eines einzelnen Knotens verursacht werden.

## 8.8 Anwendungen

In diesem Abschnitt betrachten wir zwei Probleme, die bei der Lösung von Differentialgleichungen vierter Ordnung und bei der Rekonstruktion von Bildern bei der axialen Tomografie auftreten.

### 8.8.1 Finite Elemente Analyse eines eingespannten Balkens

Wir wollen stückweise Hermitesche Polynome (siehe Abschnitt 8.4) für die numerische Approximation der Querbiegung eines eingespannten Balkens verwenden. Dieses Problem wurde bereits in Abschnitt 4.7.2 betrachtet, in dem finite Differenzen verwendet wurden.

Das mathematische Modell ist das Randwertproblem (4.74) (vgl. Band 1) vierter Ordnung, hier in der allgemeinen Formulierung

$$\begin{cases} (\alpha(x)u''(x))'' = f(x), & 0 < x < \mathcal{L} \\ u(0) = u(\mathcal{L}) = 0, & u'(0) = u'(\mathcal{L}) = 0 \end{cases} \quad (8.59)$$

präsentiert. Im Spezialfall (4.74) (vgl. Band 1) haben wir  $\alpha = EJ$  und  $f = P$ ; für das Folgende nehmen wir an, dass  $\alpha$  eine positive und beschränkte Funktion auf  $(0, \mathcal{L})$  ist, und dass  $f \in L^2(0, \mathcal{L})$ .

Wir multiplizieren (8.59) mit einer beliebigen, hinreichend glatten Funktion  $v$  und integrieren zweimal partiell, um

$$\int_0^{\mathcal{L}} \alpha u'' v'' dx - [\alpha u''' v]_0^{\mathcal{L}} + [\alpha u'' v']_0^{\mathcal{L}} = \int_0^{\mathcal{L}} f v dx$$

zu erhalten. Das Problem (8.59) wird nun durch das folgende Problem in Integralform ersetzt

$$\text{finde } u \in V, \text{ so dass } \int_0^{\mathcal{L}} \alpha u'' v'' dx = \int_0^{\mathcal{L}} f v dx, \quad \forall v \in V, \quad (8.60)$$

wobei

$$V = \left\{ v : v^{(k)} \in L^2(0, \mathcal{L}), k = 0, 1, 2, v^{(k)}(0) = v^{(k)}(\mathcal{L}) = 0, k = 0, 1 \right\}.$$

Das Problem (8.60) besitzt eine eindeutige Lösung, die die deformierte Konfiguration beschreibt, die die Gesamtenergie des Balkens über den Raum  $V$  minimiert (siehe beispielsweise [Red86], S. 156)

$$J(u) = \int_0^{\mathcal{L}} \left( \frac{1}{2} \alpha (u'')^2 - f u \right) dx.$$

Zur numerischen Lösung des Problems (8.60), führen wir eine Zerlegung  $\mathcal{T}_h$  von  $[0, \mathcal{L}]$  in  $K$  Teilintervalle  $T_k = [x_{k-1}, x_k]$ ,  $k = 1, \dots, K$ , von gleicher Länge  $h = \mathcal{L}/K$  mit  $x_k = kh$  durch und den endlich dimensionalen Raum

$$V_h = \left\{ v_h \in C^1([0, \mathcal{L}]), v_h|_T \in \mathbb{P}_3(T) \right. \\ \left. \forall T \in \mathcal{T}_h, v_h^{(k)}(0) = v_h^{(k)}(\mathcal{L}) = 0, k = 0, 1 \right\} \quad (8.61)$$

ein. Wir versehen  $V_h$  mit einer Basis. Hierzu ordnen wir jedem inneren Knoten  $x_i$  ( $i = 1, \dots, K-1$ ) einen Träger  $\sigma_i = T_i \cup T_{i+1}$  und *zwei* Funktionen  $\varphi_i, \psi_i$  zu, die wie folgt definiert sind: für jedes  $k$  gilt  $\varphi_i|_{T_k} \in \mathbb{P}_3(T_k)$ ,  $\psi_i|_{T_k} \in \mathbb{P}_3(T_k)$  und für jedes  $j = 0, \dots, K$ ,

$$\begin{cases} \varphi_i(x_j) = \delta_{ij}, & \varphi_i'(x_j) = 0, \\ \psi_i(x_j) = 0, & \psi_i'(x_j) = \delta_{ij}. \end{cases} \quad (8.62)$$

Beachte, dass die obigen Funktionen in  $V_h$  liegen und eine Basis

$$B_h = \{\varphi_i, \psi_i, i = 1, \dots, K-1\} \quad (8.63)$$

bilden. Diese Basisfunktionen können auf das Referenzintervall  $\hat{T} = [0, 1]$  für  $0 \leq \hat{x} \leq 1$  durch die affine Abbildung  $x = h\hat{x} + x_{k-1}$  zwischen  $\hat{T}$  und  $T_k$ , für  $k = 1, \dots, K$ , transformiert werden.

Wir können daher auf dem Intervall  $\hat{T}$  die Basisfunktionen  $\hat{\varphi}_0^{(0)}$  und  $\hat{\varphi}_0^{(1)}$  einführen, die dem Knoten  $\hat{x} = 0$  entsprechen, sowie  $\hat{\varphi}_1^{(0)}$  und  $\hat{\varphi}_1^{(1)}$ , die dem Knoten  $\hat{x} = 1$  zugeordnet sind. Jede dieser Basisfunktionen hat die Gestalt  $\hat{\varphi} = a_0 + a_1\hat{x} + a_2\hat{x}^2 + a_3\hat{x}^3$ ; insbesondere müssen die Funktionen mit der hochgestellten "0" den ersten beiden Bedingungen von (8.62) und die mit der hochgestellten "1" den restlichen beiden Bedingungen genügen. Die Lösung des dazugehörigen  $(4 \times 4)$  Systems liefert

$$\begin{aligned} \hat{\varphi}_0^{(0)}(\hat{x}) &= 1 - 3\hat{x}^2 + 2\hat{x}^3, & \hat{\varphi}_0^{(1)}(\hat{x}) &= \hat{x} - 2\hat{x}^2 + \hat{x}^3, \\ \hat{\varphi}_1^{(0)}(\hat{x}) &= 3\hat{x}^2 - 2\hat{x}^3, & \hat{\varphi}_1^{(1)}(\hat{x}) &= -\hat{x}^2 + \hat{x}^3. \end{aligned} \quad (8.64)$$

Die Graphen der Funktionen (8.64) sind in Abbildung 8.17 (links) dargestellt, wobei (0), (1), (2) und (3) die Funktionen  $\hat{\varphi}_0^{(0)}$ ,  $\hat{\varphi}_1^{(0)}$ ,  $\hat{\varphi}_0^{(1)}$  bzw.  $\hat{\varphi}_1^{(1)}$  bezeichnen.

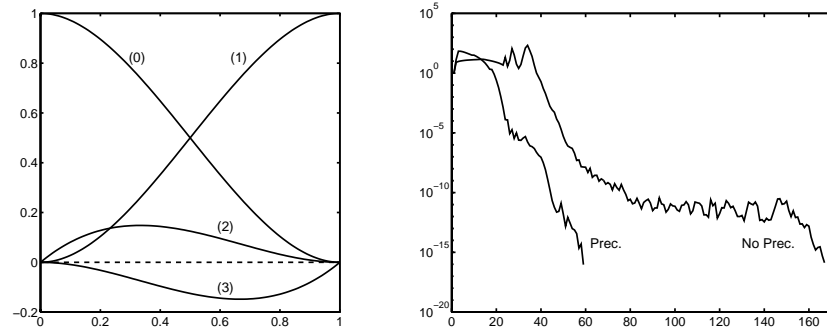


Abbildung 8.17. Natürliche Hermite-Basis auf dem Referenzintervall  $0 \leq \hat{x} \leq 1$  (links); Konvergenzverlauf für die Methode der konjugierten Gradienten bei der Lösung des Systems (8.69) (rechts). Auf der  $x$ -Achse ist die Iterationszahl  $k$  angegeben, auf der  $y$ -Achse die Größe  $\|\mathbf{r}^{(k)}\|_2 / \|\mathbf{b}_1\|_2$  dargestellt, wobei  $\mathbf{r}$  das Residuum des Systems (8.69) ist.

Die Funktion  $u_h \in V_h$  kann dargestellt werden als

$$u_h(x) = \sum_{i=1}^{K-1} u_i \varphi_i(x) + \sum_{i=1}^{K-1} u_i^{(1)} \psi_i(x). \quad (8.65)$$

Hierbei haben die Koeffizienten und die *Freiheitsgrade* von  $u_h$  die folgende Bedeutung:  $u_i = u_h(x_i)$ ,  $u_i^{(1)}(x_i) = u_h'(x_i)$  für  $i = 1, \dots, K-1$ . Beachte, dass (8.65) ein Spezialfall von (8.32) ist, nämlich  $m_i = 1$  in (8.32) gesetzt wurde.



Die Diskretisierung des Problem (8.60) lautet

$$\text{finde } u_h \in V_h, \text{ so dass } \int_0^{\mathcal{L}} \alpha u_h'' v_h'' dx = \int_0^{\mathcal{L}} f v_h dx, \quad \forall v_h \in B_h. \quad (8.66)$$

Sie wird *Galerkin finite Elemente* Approximation des Randwertproblems (8.59) genannt. Hinsichtlich einer ausführlicheren Diskussion und Analyse der Methode verweisen wir auf Kapitel 12, Abschnitte 12.4 und 12.4.5.

Unter Verwendung der Darstellung (8.65) gelangen wir zu folgendem System in den  $2K - 2$  Unbekannten  $u_1, u_2, \dots, u_{K-1}, u_1^{(1)}, u_2^{(1)}, \dots, u_{K-1}^{(1)}$

$$\begin{cases} \sum_{j=1}^{K-1} \left\{ u_j \int_0^{\mathcal{L}} \alpha \varphi_j'' \varphi_i'' dx + u_j^{(1)} \int_0^{\mathcal{L}} \alpha \psi_j'' \varphi_i'' dx \right\} = \int_0^{\mathcal{L}} f \varphi_i dx, \\ \sum_{j=1}^{K-1} \left\{ u_j \int_0^{\mathcal{L}} \alpha \varphi_j'' \psi_i'' dx + u_j^{(1)} \int_0^{\mathcal{L}} \alpha \psi_j'' \psi_i'' dx \right\} = \int_0^{\mathcal{L}} f \psi_i dx, \end{cases} \quad (8.67)$$

für  $i = 1, \dots, K - 1$ . Nehmen wir der Einfachheit halber an, dass der Balken die Einheitslänge  $\mathcal{L}$  habe, und dass  $\alpha$  sowie  $f$  zwei Konstanten sind. Berechnen wir die in (8.67) auftretenden Integrale, lautet das zu lösende System in Matrixform

$$\begin{cases} \mathbf{A} \mathbf{u} + \mathbf{B} \mathbf{p} = \mathbf{b}_1 \\ \mathbf{B}^T \mathbf{u} + \mathbf{C} \mathbf{p} = \mathbf{0}, \end{cases} \quad (8.68)$$

wobei die Vektoren  $\mathbf{u}, \mathbf{p} \in \mathbb{R}^{K-1}$  die unbekannten Knotenwerte  $u_i$  und  $u_i^{(1)}$  enthalten,  $\mathbf{b}_1 \in \mathbb{R}^{K-1}$  der Vektor mit den Komponenten gleich  $h^4 f / \alpha$  ist, und

$$\mathbf{A} = \text{tridiag}_{K-1}(-12, 24, -12),$$

$$\mathbf{B} = \text{tridiag}_{K-1}(-6, 0, 6),$$

$$\mathbf{C} = \text{tridiag}_{K-1}(2, 8, 2).$$

Das System (8.68) hat die Größe  $2(K - 1)$ ; durch Elimination der Unbekannten  $\mathbf{p}$  aus der zweiten Gleichung kann es auf das System (der Größe  $K - 1$ )

$$(\mathbf{A} - \mathbf{B} \mathbf{C}^{-1} \mathbf{B}^T) \mathbf{u} = \mathbf{b}_1. \quad (8.69)$$

reduziert werden. Da  $\mathbf{B}$  anti-symmetrisch und  $\mathbf{A}$  symmetrisch, positiv definit sind (s.p.d.), ist auch die Matrix  $\mathbf{M} = \mathbf{A} - \mathbf{B} \mathbf{C}^{-1} \mathbf{B}^T$  s.p.d.. Die Cholesky-Faktorisierung ist zur Lösung des Systems (8.69) ungeeignet, da  $\mathbf{C}^{-1}$  eine vollbesetzte Matrix ist. Alternativ bietet sich das Verfahren der konjugierten Gradienten (CG) in Kombination mit einen geeigneten Vorkonditionierer an, denn die spektrale Konditionszahl von  $\mathbf{M}$  ist von der Ordnung  $h^{-4} = K^4$ .

Wir bemerken, dass die Berechnung des Residuum in jedem Schritt  $k \geq 0$  die Lösung eines linearen Gleichungssystems erfordert, dessen rechte Seite der Vektor  $\mathbf{B}^T \mathbf{u}^{(k)}$  mit der aktuellen Iterierten  $\mathbf{u}^{(k)}$  ist und dessen Koeffizientenmatrix die Matrix  $\mathbf{C}$  ist. Dieses System kann mit dem Thomas-Algorithmus (3.53) (vgl. Band 1) mit einem Aufwand von  $K$  flops gelöst werden.

Das CG-Verfahren bricht mit dem kleinsten Wert von  $k$  ab, für den  $\|\mathbf{r}^{(k)}\|_2 \leq \mathbf{u} \|\mathbf{b}_1\|_2$  gilt, wobei  $\mathbf{r}^{(k)}$  das Residuum des System (8.69) und  $\mathbf{u}$  die *Rundungseinheit* bezeichnen.

Die mit dem CG-Verfahren im Fall einer gleichmäßigen Zerlegung von  $[0, 1]$  in  $K = 50$  Elemente und  $\alpha = f = 1$  erzielten Ergebnisse wurden in Abbildung 8.17 (rechts) zusammengefasst. Sie zeigt den Konvergenzverlauf sowohl für die nichtvorkonditionierte Form (durch "Non Prec." gekennzeichnet), als auch für den SSOR-Vorkonditionierer (durch "Prec." gekennzeichnet) bei einem gesetzten Relaxationsparameter von  $\omega = 1.95$ .

Wir bemerken, dass das CG-Verfahren aufgrund von Rundungsfehlern nicht innerhalb von  $K - 1$  Schritten konvergiert. Beachte auch die Effizienz des SSOR-Vorkonditionierers in Bezug auf die Reduktion der Iterationszahlen. Jedoch veranlassen uns die hohen numerischen Kosten dieses Vorkonditionierers einen anderen Vorkonditionierer zu wählen. Ausgehend von der Struktur der Matrix  $\mathbf{M}$  ist ein natürlicher Kandidat für einen Vorkonditionierer  $\mathcal{M} = \mathbf{A} - \mathbf{B}\tilde{\mathbf{C}}^{-1}\mathbf{B}^T$ , wobei  $\tilde{\mathbf{C}}$  die Diagonalmatrix mit den Einträgen  $\tilde{c}_{ii} = \sum_{j=1}^{K-1} |c_{ij}|$  ist. Die Matrix  $\mathcal{M}$  ist eine Bandmatrix, so dass Ihre Inversion wesentlich geringere Kosten verursacht als der SSOR-Vorkonditionierer. Darüber hinaus führt die Verwendung von  $\mathcal{M}$  zu einem dramatischen Absinken der Iterationszahl wie aus der Tabelle 8.6 ersichtlich ist.

Tabelle 8.6. Iterationszahl als Funktion von  $K$ .

$K$	Ohne Vorkond.	SSOR	$\mathcal{M}$
25	51	27	12
50	178	61	25
100	685	118	33
200	2849	237	34

### 8.8.2 Geometrische Rekonstruktion mittels Computertomographie

Eine typische Anwendung der in Abschnitt 8.7 vorgestellten Algorithmen ist die Rekonstruktion einer dreidimensionalen Struktur der inneren Organe des menschlichen Körpers mit Hilfe der *Computertomographie* (CT).

Die Computertomographie (CT) liefert üblicherweise eine Folge von Bildern, die die Bereiche eines Organs in verschiedenen horizontalen Ebenen

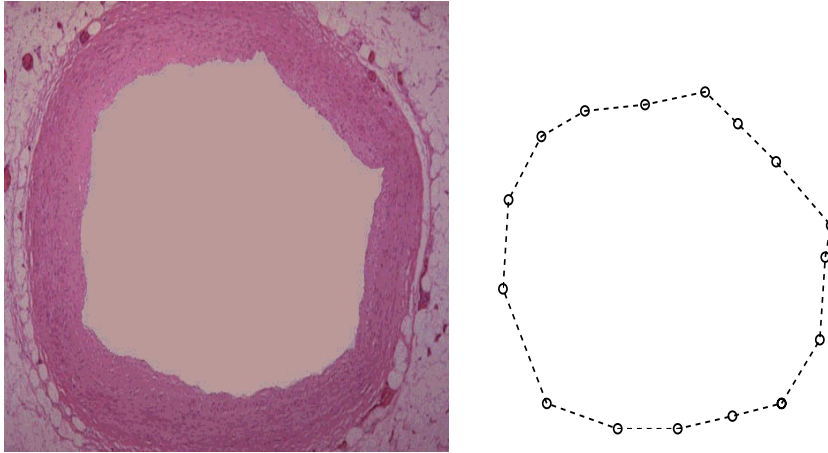


Abbildung 8.18. Querschnitt eines Blutgefäßes (links) und ein entsprechendes 16 Punkte  $P_i$  benutzendes charakteristisches Polygon (rechts).

darstellen; als Vereinbarung legen wir fest, dass eine CT Querschnitte in der  $x, y$  Ebene zu verschiedenen Werten von  $z$  liefert. Das Ergebnis ist analog zu dem, was man durch schichtenweises Zerlegen des Organs bei verschiedenen Werten von  $z$  und Fotografieren der Querschnitte bekommen würde. Offensichtlich besteht der große Vorteil bei der Anwendung der CT darin, dass das zu untersuchende Organ visualisiert werden kann ohne hinter benachbarten versteckt zu sein, wie es bei anderen Arten medizinischer Bilder, z.B. bei Angiogrammen, passieren kann.

Das für jeden Querschnitt erhaltene Bild wird in eine Matrix von *Pixeln* (abgekürzt: von *Bildelementen*) in der  $x, y$  Ebene kodiert; ein bestimmter Wert ist mit jedem Pixel verbunden und drückt die Graustufe des Bildes in diesem Punkt aus. Diese Stufe wird aus der Dichte der Röntgenstrahlung bestimmt, die von einem Detektor nach dem Durchgang durch den menschlichen Körper aufgefangen wird. In der Praxis wird die in einem CT enthaltene Information in einem gegebenen Wert  $z$  als Menge von Punkten  $(x_i, y_i)$ , die den Rand des Organs bei  $z$  identifizieren, ausgedrückt.

Zur Verbesserung der Diagnostik ist es oftmals nützlich die dreidimensionale Struktur des zu untersuchenden Organs ausgehend von den vom CT gelieferten Querschnitten zu rekonstruieren. Für dieses Ziel ist es erforderlich, die pixelkodierte Information in eine parametrische Darstellung zu überführen, die durch geeignete Funktionen ausgedrückt werden kann, die das Bild in einigen wichtigen Punkten seines Randes interpolieren. Diese Rekonstruktion kann unter Verwendung der im Abschnitt 8.7 beschriebenen Methoden ausgeführt werden, wie Abbildung 8.19 zeigt.

Eine Menge von Kurven, wie die in Abbildung 8.19 kann geeignet geschichtet werden, um eine Gesamtansicht des untersuchten Organs zu erzeugen.

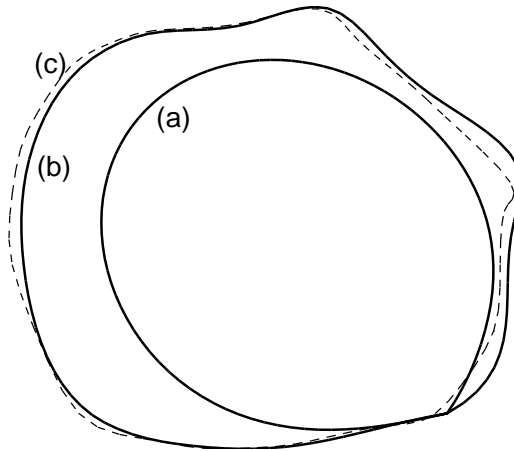


Abbildung 8.19. Rekonstruktion des inneren Gefäßes von Abbildung 8.18 unter Verwendung verschiedener interpolierender Splines mit dem gleichen charakteristischen Polynoms: (a) Bézier-Kurven, (b) parametrische Splines und (c) Parametrische B-Splines.

## 8.9 Übungen

1. Beweise, dass die in (8.3) definierten charakteristischen Polynome  $l_i \in \mathbb{P}_n$  eine Basis für  $\mathbb{P}_n$  bilden.
2. Ein anderer Zugang zu der Methode von Theorem 8.1, der Konstruktion des Interpolationspolynoms, besteht darin, direkt die  $n+1$  Interpolationsbedingungen auf  $\Pi_n$  aufzuprägen und die Koeffizienten  $a_i$  zu berechnen. Wenn wir so verfahren, erhalten wir ein lineares Gleichungssystem  $\mathbf{X}\mathbf{a} = \mathbf{y}$ , wobei  $\mathbf{a} = (a_0, \dots, a_n)^T$ ,  $\mathbf{y} = (y_0, \dots, y_n)^T$  und  $\mathbf{X} = [x_i^j]$ .  $\mathbf{X}$  wird *Vandermondesche Matrix* genannt. Beweise, dass  $\mathbf{X}$  nichtsingulär ist, wenn die Knoten  $x_i$  verschieden sind.

[Hinweis: Zeige durch Rekursion in  $n$ , dass  $\det(\mathbf{X}) = \prod_{0 \leq j < i \leq n} (x_i - x_j)$ .]

3. Beweise, dass  $\omega'_{n+1}(x_i) = \prod_{\substack{j=0 \\ j \neq i}}^n (x_i - x_j)$  wobei  $\omega_{n+1}$  das Knotenpolynom (8.6) ist. Überprüfe dann (8.5).

4. Gib eine Abschätzung von  $\|\omega_{n+1}\|_\infty$  in den Fällen  $n = 1$  und  $n = 2$  für äquidistante Knoten an.
5. Beweise, dass

$$\begin{aligned} (n-1)!h^{n-1}|(x-x_{n-1})(x-x_n)| &\leq |\omega_{n+1}(x)| \\ |\omega_{n+1}(x)| &\leq n!h^{n-1}|(x-x_{n-1})(x-x_n)|, \end{aligned}$$

wobei  $n$  gerade,  $-1 = x_0 < x_1 < \dots < x_{n-1} < x_n = 1$ ,  $x \in (x_{n-1}, x_n)$  und  $h = 2/n$  sind.

[*Hinweis* : Setze  $N = n/2$  und zeige zunächst, dass

$$\begin{aligned}\omega_{n+1}(x) &= (x + Nh)(x + (N-1)h) \dots (x + h)x \\ &\quad (x - h) \dots (x - (N-1)h)(x - Nh).\end{aligned}\tag{8.70}$$

Nimm dann  $x = rh$  mit  $N-1 < r < N$ .]

6. Zeige unter den Annahmen von Übung 5, dass  $|\omega_{n+1}|$  für  $x \in (x_{n-1}, x_n)$  maximal wird (beachte, dass  $|\omega_{n+1}|$  eine gerade Funktion ist).

[*Hinweis* : Verwende (8.70) um zu zeigen, dass  $|\omega_{n+1}(x+h)/\omega_{n+1}(x)| > 1$  für jedes  $x \in (0, x_{n-1})$  und  $x$  kein Interpolationsknoten ist.]

7. Beweise die Rekursionsbeziehung (8.19) für die Newtonschen dividierten Differenzen.

8. Bestimme ein Interpolationspolynom  $Hf \in \mathbb{P}_n$  derart, dass

$$(Hf)^{(k)}(x_0) = f^{(k)}(x_0), \quad k = 0, \dots, n,$$

und zeige, dass

$$Hf(x) = \sum_{j=0}^n \frac{f^{(j)}(x_0)}{j!} (x - x_0)^j$$

gilt. Dies bedeutet, dass das Hermitesche Interpolationspolynom auf einem Knoten mit dem *Taylorpolynom* übereinstimmt.

9. Beweise, dass für die gegebene Menge von Daten

$$\{f_0 = f(-1) = 1, f_1 = f'(-1) = 1, f_2 = f'(1) = 2, f_3 = f(2) = 1\}$$

das Hermite-Birkoff-Interpolationspolynom  $H_3$  nicht existiert.

[*Lösung* : Für den Ansatz  $H_3(x) = a_3x^3 + a_2x^2 + a_1x + a_0$  muss man zeigen, dass die Matrix des linearen Gleichungssystems  $H_3(x_i) = f_i$ ,  $i = 0, \dots, 3$ , singulär ist.]

10. Zeigen Sie, dass jedes  $s_k \in \mathcal{S}_k[a, b]$  eine Darstellung der Form

$$s_k(x) = \sum_{i=0}^k b_i x^i + \sum_{i=1}^g c_i (x - x_i)_+^k$$

besitzt, d.h.  $1, x, x^2, \dots, x^k, (x - x_1)_+^k, \dots, (x - x_g)_+^k$  bilden eine Basis für  $\mathcal{S}_k[a, b]$ .

11. Beweise die Eigenschaft 8.2 und zeige die Gültigkeit auch in dem Fall, wenn der Spline  $s$  Bedingungen der Form  $s'(a) = f'(a)$ ,  $s'(b) = f'(b)$  genügt.

[*Hinweis*: Beginne mit

$$\int_a^b [f''(x) - s''(x)] s''(x) dx = \sum_{i=1}^n \int_{x_{i-1}}^{x_i} [f''(x) - s''(x)] s''(x) dx$$

und integriere zweimal partiell.]

12. Sei  $f(x) = \cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$ . Betrachte die folgende rationale Approximation

$$r(x) = \frac{a_0 + a_2 x^2 + a_4 x^4}{1 + b_2 x^2}, \quad (8.71)$$

die *Padé Approximation* genannt wird. Bestimme die Koeffizienten von  $r$  derart, dass

$$f(x) - r(x) = \gamma_8 x^8 + \gamma_{10} x^{10} + \dots$$

[Lösung:  $a_0 = 1$ ,  $a_2 = -7/15$ ,  $a_4 = 1/40$ ,  $b_2 = 1/30$ .]

13. Angenommen, die Funktion  $f$  des vorigen Beispiels sei auf einer Menge von  $n$  äquidistanten Punkten  $x_i \in (-\pi/2, \pi/2)$ ,  $i = 0, \dots, n$ , bekannt. Wiederhole Übung 12, indem die Koeffizienten von  $r$  mittels MATLAB auf solche Weise bestimmt werden, dass die Größe  $\sum_{i=0}^n |f(x_i) - r(x_i)|^2$  minimiert wird. Betrachte die Fälle  $n = 5$  und  $n = 10$ .

Numerische Mathematik 2

Quarteroni, A.; Sacco, R.; Saleri, F.

2002, XIV, 330 S. 55 Abb., Softcover

ISBN: 978-3-540-43616-4