Kapitel 4

Interpolation

4.1 Aufgabenstellung

Gegeben seien $(n+1)$ Paare (x_i, y_i), $i = 0, \ldots n$, wobei die x_i paarweise verschieden sind. Die Interpolationsaufgabe besteht darin, eine (einfache) Funktion Φ zu finden, so dass $\Phi(x_i) = y_i$, $i = 0, \ldots n$, ist.

Man bezeichnet $\{x_i\}$ als die Menge der Stützstellen und die Menge $\{y_i\}$ als die Menge der Stützwerte. Man sagt, dass die Funktion $\Phi(x)$ die Stützwerte $\{y_i\}$ an den Stützstellen $\{x_i\}$ interpoliert.

Natürlich gibt es unendlich viele Funktionen, die die Interpolationsaufgabe erfüllen. Deshalb muss man die Klasse der Funktionen festlegen, in der man die Interpolierende $\Phi(x)$ sucht. Man unterscheidet zum Beispiel:

- $\Phi(x)$ ist ein Polynom – Polynominterpolation,
- $\Phi(x) = a_0 + a_1 e^{ix} + \ldots + a_n e^{inx}$ – trigonometrische Interpolation ($i = \sqrt{-1}$),
- $\Phi(x)$ ist stückweise ein Polynom – Spline–Interpolation,
- $\Phi(x)$ ist eine rationale Funktion – rationale Interpolation.

4.2 Polynominterpolation

Wir bezeichnen mit

$$P_n := \left\{p(x) \ : \ p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0, \ a_i \in \mathbb{R}, i = 0, \ldots, n\right\}$$

den Raum aller Polynome vom Höchstgrad n. Es werden $(n+1)$ Paare (x_i, y_i), $i = 0, \ldots, n$, $x_i \neq x_j$ für $i \neq j$, betrachtet. Die Aufgabe der Polynominterpolation besteht nun darin, ein $p \in P_n$ zu finden, so dass gilt

$$p(x_i) = y_i, \quad i = 0, \ldots, n.$$

Als erstes stellt sich die Frage nach der Existenz und Eindeutigkeit eines solchen Polynoms. Diese wird mit dem folgenden Satz beantwortet.

Satz 4.1 *Zu gegebenen paarweise verschiedenen Stützstellen $x_0, \ldots, x_n \in \mathbb{R}$ und zugehörigen Werten $y_0, \ldots, y_n \in \mathbb{R}$ existiert genau ein Polynom $p \in P_n$ mit $p(x_i) = y_i$, $i = 0, \ldots, n$.*

Beweis: Literatur, zum Beispiel [QSS04]. ∎

Man kann zeigen, dass dieses Polynom sich wie folgt darstellen lässt:

$$p_n(x) = \sum_{k=0}^{n} \frac{\omega_{n+1}(x)}{(x - x_k)\omega'_{n+1}(x_k)} y_k, \tag{4.1}$$

wobei $\omega_{n+1}(x)$ das sogenannte Knotenpolynom vom Grad $n+1$ ist

$$\omega_{n+1}(x) = \prod_{j=0}^{n} (x - x_j)\,. \tag{4.2}$$

Mit der Produktregel erhält man *Übungsaufgabe*

$$\frac{\omega_{n+1}(x)}{(x - x_k)\omega'_{n+1}(x_k)} = \prod_{j=0, j\neq k}^{n} \frac{x - x_j}{x_k - x_j}. \tag{4.3}$$

Damit erhält man die zu (4.1) äquivalente Darstellung

$$p_n(x) = \sum_{k=0}^{n} \left(\prod_{j=0, j\neq k}^{n} \frac{x - x_j}{x_k - x_j} \right) y_k. \tag{4.4}$$

Mit dieser Darstellung erkennt man auch leicht, dass $p_n(x_i) = y_i$ für alle Stützstellen. Setzt man $x = x_i$ in (4.4), dann hat man im Zähler des Produkts für $k \neq i$ den Faktor $x_i - x_i = 0$. Es bleibt somit nur der Summand mit dem Summationsindex $k = i$ übrig

$$p_n(x_i) = \left(\prod_{j=0, j\neq i}^{n} \frac{x_i - x_j}{x_i - x_j} \right) y_i = y_i.$$

Die Formeln (4.1) oder (4.4) bezeichnet man als Langrange–Form des Interpolationspolynoms.

Zur Untersuchung der Genauigkeit der Polynominterpolation geht man wie folgt vor. Man nimmt sich eine Funktion $f(x)$, gibt sich $(n+1)$ Paare von Stützstellen und Stützwerten $(x_i, f(x_i))$ vor, berechnet das Interpolationspolynom $p_n f(x)$ durch diese Punkte und vergleicht dieses mit der vorgegebenen Funktion. *Bild* Man kann die folgende Abschätzung für den Interpolationsfehler beweisen.

Satz 4.2 *Seien $x_0, \ldots, x_1$ paarweise verschiedene Stützstellen und sei z Element das Definitionsbereichs von $f(x)$. Sei weiter $f \in C^{n+1}(I_x)$, wobei I_x das kleinste Intervall ist, mit $x_i \in I_x$, $i = 0, \ldots, n$. Dann ist der Interpolationsfehler im Punkt z durch*

$$E_n(z) := f(z) - p_n f(z) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \omega_{n+1}(z)$$

gegeben, wobei $\xi \in I_x$ ist und $\omega_{n+1}(z)$ in (4.2) definiert ist.

Beweis: Literatur, [QSS04]. ∎

Die Aussage impliziert nicht, dass für $n \to \infty$ gilt, dass $p_n f(x) \to f(x)$ für alle $x \in I_x$. Man kann Funktionen und zugehörige Mengen von Stützstellen so finden (zum Beispiel mit gleichem Abstand), dass es Teilintervalle von I_x gibt, in denen $p_n f(x) \not\to f(x)$ für alle Argumente x aus diesen Teilintervallen.

Zur Untersuchung der Stabilität der Polynominterpolation betrachtet man neben den Paaren $(x_i, f(x_i))$ Paare mit gestörten Werten $(x_i, \tilde{f}(x_i))$. Dann gilt

$$\begin{aligned} \left\| p_n f - p_n \tilde{f} \right\|_{C(I_x)} &:= \max_{x \in I_x} \left| \sum_{k=0}^{n} \left(\prod_{j=0, j\neq k}^{n} \frac{x - x_j}{x_k - x_j} \right) \left(f(x_k) - \tilde{f}(x_k) \right) \right| \\ &\leq \max_{k=1,\ldots,n} \left| f(x_k) - \tilde{f}(x_k) \right| \underbrace{\left\| \sum_{k=0}^{n} \left(\prod_{j=0, j\neq k}^{n} \frac{x - x_j}{x_k - x_j} \right) \right\|_{C(I_x)}}_{\Lambda_n(x)}. \end{aligned}$$

Man nennt $\Lambda_n(x)$ Lebesgue–Konstante und diese Konstante spielt die Rolle einer Konditionszahl für die Polynominterpolation. Das bedeutet, dass kleine Änderungen in den Daten nur dann zu kleinen Änderungen im Ergebnis führen, falls $\Lambda_n(x)$ für alle $x \in I_x$ klein ist. Man kann jedoch zeigen, dass $\Lambda_n \to \infty$ für $n \to \infty$. Für äquidistante Stützstellen hat man beispielsweise

$$\Lambda_n \approx \frac{2^{n+1}}{e\ n \log(n)}, \quad e \ - \text{ Eulersche Zahl.}$$

Daraus folgt, dass die Polynominterpolation für große n instabil werden kann. *MATLAB–Demo*

Im Prinzip ist mit den Darstellungen (4.1) oder (4.4) die Aufgabe der Polynominterpolation gelöst. Diese Darstellungen sind aber aus numerischer Sicht unvorteilhaft, weil:

- unnötig viele Flops zur Auswertung von $p_n f(x)$ an einer bestimmten Stelle x benötigt werden,
- es in diesen Darstellungen unmöglich ist, auf einfache Art und Weise zusätzliche Stützstellen zu integrieren.

4.3 Die Newton–Interpolation

In diesem Abschnitt wird ein Interpolation eingeführt, die die beiden Nachteile der Standard–Lagrange–Interpolation nicht besitzt, die sogenannte Newton–Interpolation.

Seien $(n+1)$ Paare von Stützstellen und Stützwerten $(x_i, f(x_i))$, $i = 0, \ldots, n$, gegeben und $p_n f(x)$ sei das zugehörige Interpolationspolynom. Nun soll $p_n f(x)$ als eine Summe des Interpolationspolynoms $p_{n-1} f(x)$ zu den Stützstellen $(x_i, f(x_i))$, $i = 0, \ldots, n-1$, und eines Polynom n–ten Grades $q_n(x)$ dargestellt werden

$$p_n f(x) = p_{n-1} f(x) + q_n(x). \tag{4.5}$$

Das Polynom $q_n(x)$ soll nur von den Stützstellen x_i abhängen und nur einen unbekannten Koeffizienten besitzen. Falls es eine solche Darstellung (4.5) gibt, ist die Hinzunahme zusätzlicher Stützstellen kein Problem.

Aus (4.5) folgt

$$q_n(x_i) = p_n f(x_i) - p_{n-1} f(x_i) = 0, \quad i = 0, \ldots, n-1.$$

Damit sind n Nullstellen von $q_n(x)$ bekannt. Mehr kann ein Polynom n–ten Grades nicht besitzen und somit hat $q_n(x)$ die Darstellung

$$q_n(x) = a_n(x - x_0)(x - x_1)\ldots(x - x_{n-1}) = a_n \omega_n(x).$$

Damit hat $q_n(x)$ den einzigen unbekannten Koeffizienten a_n. Zur Bestimmung dieses Koeffizienten verwendet man die Stützstelle $(x_n, f(x_n))$ und erhält

$$q_n(x_n) = a_n \omega_n(x_n) = p_n f(x_n) - p_{n-1} f(x_n) = f(x_n) - p_{n-1} f(x_n).$$

Daraus folgt

$$a_n = \frac{f(x_n) - p_{n-1} f(x_n)}{\omega_n(x_n)}. \tag{4.6}$$

Definition 4.3 Seien $(x_i, f(x_i)) \in \mathbb{R} \times \mathbb{R}$, $i = 0, \ldots, n$, paarweise verschiedene Stützstellen. Die k–te dividierte Differenz $f[x_i, x_{i+1}, \ldots, x_{i+k}]$ wird rekursiv definiert durch

$$\begin{aligned} f[x_i] &= f(x_i) \quad \text{für } i = 0, \ldots, n, \\ f[x_i, x_{i+1}, \ldots, x_{i+k}] &= \frac{f[x_{i+1}, x_{i+2}, \ldots, x_{i+k}] - f[x_i, x_{i+1}, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}. \end{aligned}$$

□

Um den Zusammenhang zwischen den dividierten Differenzen und der Interpolationsaufgabe herzustellen, betrachten wir zunächst den Fall $n = 1$. Dann sind $p_0 f(x) = f(x_0)$ für alle x und

$$a_1 = \frac{f(x_1) - p_0 f(x_1)}{\omega_1(x_1)} = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{f[x_1] - f[x_0]}{x_1 - x_0} = f[x_0, x_1].$$

Damit ergibt sich mit (4.5)

$$p_1 f(x) = f[x_0] + f[x_0, x_1](x - x_0).$$

Für $n = 2$ erhält man

$$\begin{aligned} a_2 &= \frac{f(x_2) - p_1 f(x_2)}{\omega_2(x_2)} = \frac{f[x_2] - f[x_0] - f[x_0, x_1](x_2 - x_0)}{(x_2 - x_0)(x_2 - x_1)} \\ &= \frac{1}{x_2 - x_0} \left(\frac{(f(x_2) - f(x_1)) + (f(x_1) - f(x_0)) - f[x_0, x_1](x_2 - x_0)}{x_2 - x_1} \right) \\ &= \frac{1}{x_2 - x_0} \left(\frac{f[x_1, x_2](x_2 - x_1) + f[x_0, x_1](x_1 - x_0) - f[x_0, x_1](x_2 - x_0)}{x_2 - x_1} \right) \\ &= \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} = f[x_0, x_1, x_2], \end{aligned}$$

also

$$p_2 f(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1).$$

Durch Induktion erhält man für a_n in (4.6)

$$a_n = f[x_0, x_1, \ldots, x_n]$$

und für das Interpolationspolynom

$$p_n f(x) = \sum_{k=0}^{n} \omega_k(x) f[x_0, x_1, \ldots, x_k]. \tag{4.7}$$

Diese Darstellung wird Newtonsche Interpolationsformel genannt. Die Eindeutigkeit der Polynominterpolation sichert, dass man das gleiche Polynom wie bei der Lagrange–Interpolation erhält.

Zur Berechnung der dividierten Differenzen nutzt man ein Dreiecksschema:

	$k=0$		$k=1$		$k=2$		$k=n$
x_0	$\underline{f[x_0]} = f(x_0)$	$\searrow$					
x_1	$f[x_1] = f(x_1)$	$\rightarrow$	$\underline{f[x_0, x_1]}$				
		$\searrow$		$\searrow$			
x_2	$f[x_2] = f(x_2)$	$\rightarrow$	$f[x_1, x_2]$	$\rightarrow$	$\underline{f[x_0, x_1, x_2]}$		
$\vdots$							
		$\searrow$		$\searrow$			
x_n	$f[x_n] = f(x_n)$	$\rightarrow$	$f[x_{n-1}, x_n]$	$\rightarrow$	$f[x_{n-2}, x_{n-1}, x_n]$	$\ldots$	$\underline{f[x_0, \ldots, x_n]}$

Beispiel 4.4 Man berechne das Newton–Interpolationspolynom durch die in der Tabelle angegebenen Paare von Stützstellen und Stützwerten:

x_i	$f(x_i) = f[x_i]$			
-1	$\underline{2}$			
0	4	$\frac{4-2}{0-(-1)} = \underline{2}$		
2	6	$\frac{6-4}{2-0} = 1$	$\frac{1-2}{2-(-1)} = \underline{-\frac{1}{3}}$	
3	12	$\frac{12-6}{3-2} = 6$	$\frac{6-1}{3-0} = \frac{5}{3}$	$\frac{5/3-(-1/3)}{3-(-1)} = \underline{\frac{1}{2}}$

Es folgt

$$p_3 f(x) = 2 + 2(x+1) - \frac{1}{3}(x+1)x + \frac{1}{2}(x+1)x(x-2).$$

□

Zur Berechnung der Koeffizienten im Newton–Interpolationspolynom (4.7) benötigt man nur die unterstrichenen Werte. Deswegen braucht man zu ihrer Berechnung nur einen Vektor der Länge $n+1$. Man geht spaltenweise vor. Zunächst belegt man den Vektor mit $f[x_0], \ldots, f[x_n]$. Im zweiten Schritt überschreibt man $f[x_1], \ldots, f[x_n]$ mit $f[x_0, x_1], \ldots, f[x_{n-1}, x_n]$ und so weiter. Wenn man allerdings später noch Stützstellen hinzufügen will, braucht man das gesamte Dreieck.

Der Rechenaufwand pro dividierter Differenz ist 3 Flops. Zur Berechnung aller Koeffizienten des Newton–Interpolationspolynoms hat man

$$n + (n-1) + (n-2) + \ldots + 1 = \frac{n}{2}(n+1)$$

dividierte Differenzen zu berechnen, so dass der Gesamtaufwand

$$\frac{3}{2}n^2 + \frac{3}{2}n \text{ Flops}$$

beträgt.

Bezüglich der Approximationsgüte des Newton–Interpolationspolynoms gilt die Aussage von Satz 4.1.

4.4 Spline–Interpolation

spline (engl.) – längliches, dünnes Stück Holz oder Metall

Die Polynominterpolation besitzt zwei sehr negative Eigenschaften:

- für große n, das heißt viele Stützstellen kann sie instabil werden, insbesondere bei äquidistanten Stützstellen,
- für nichtglatte Funktionen kann der Interpolationsfehler zwischen den Stützstellen beliebig groß werden.

Matlab–Demo Diese beiden Situationen (viele äquidistante Stützstellen oder nichtglatte Funktionen) treten in den Anwendung jedoch häufig auf. In diesen Fällen verwendet man oft die Spline–Interpolation.

Definition 4.5 Seien $x_0, \ldots, x_n \in [a,b]$ paarweise verschiedene Punkte mit $a = x_0 < x_1 < \ldots < x_n = b$. Die Funktion $s_k(x) : [a,b] \to \mathbb{R}$ wird Spline vom Grad k bezüglich der Stützstellen $\{x_j\}$ genannt, falls

$$s_k(x)|_{[x_j, x_{j+1}]} \in P_k, \quad j = 0, \ldots, n-1, \tag{4.8}$$

$$s_k(x) \in C^{k-1}([a,b]). \tag{4.9}$$

□

Das bedeutet ein Spline vom Grad k ist

- eine stückweise polynomiale Funktion, in jedem Teilintervall ein Polynom vom Grad k,
- im Gesamtintervall noch $(k-1)$–mal differenzierbar.

In jedem Teilintervall kann man den Spline als

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

darstellen. Man hat also $n(k+1)$ unbekannte Koeffizienten s_{ij}. Aus (4.9) folgt

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

Das sind $k(n-1)$ Bedingungen an die Koeffizienten. Im allgemeinen soll der Spline eine Funktion f approximieren, deren Werte in den Stützstellen berechnet werden können. Damit hat man weitere $n+1$ Bedingungen: $s_k(x_j) = f(x_j)$, $j = 0, \ldots, n$. Es fehlen noch $k-1$ Bedingungen.

In der Praxis nutzt man oft kubische Splines, das heißt $k = 3$. Ein kubischer Spline ist zweimal stetig bis zum Rand differenzierbar, insbesondere ist seine Krümmung wohldefiniert. Für die zwei fehlenden Bedingungen setzt man zum Beispiel die Randwerte

$$s_3''(a) = s_3''(b) = 0. \tag{4.10}$$

Betrachten wir nun eine Funktion f, die durch einen kubischen Spline interpoliert werden soll. Wir verwenden die Bezeichnungen

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

Die zweite Ableitung $s_3''(x)$ ist eine stetige, stückweise lineare Funktion (Polygonzug). Mit den obigen Bezeichnungen gilt

$$s_3''(x) = M_{i-1}\frac{x_i - x}{x_i - x_{i-1}} + M_i\frac{x - x_{i-1}}{x_i - x_{i-1}}, \quad x \in [x_{i-1}, x_i].$$

Die Stetigkeit von $s_3''(x)$ überprüft man durch Einsetzen der Intervallgrenzen. Zweimaliges integrieren liefert eine Darstellung des Splines in $[x_{i-1}, x_i]$

$$s_3(x) = \frac{M_{i-1}}{6}\frac{(x_i - x)^3}{x_i - x_{i-1}} + \frac{M_i}{6}\frac{(x - x_{i-1})^3}{x_i - x_{i-1}} + c_{i-1}(x - x_{i-1}) + d_{i-1}. \tag{4.11}$$

Einsetzen der Endpunkte des Teilintervalls ergibt die Konstanten:

$$\begin{aligned} f_{i-1} &= s_3(x_{i-1}) = \frac{M_{i-1}}{6}(x_i - x_{i-1})^2 + d_{i-1}, \quad \Longrightarrow \\ d_{i-1} &= f_{i-1} - \frac{M_{i-1}}{6}(x_i - x_{i-1})^2, \end{aligned} \tag{4.12}$$

und

$$\begin{aligned} f_i &= s_3(x_i) = \frac{M_i}{6}(x_i - x_{i-1})^2 + c_{i-1}(x_i - x_{i-1}) + d_{i-1} \Longrightarrow \\ c_{i-1} &= \frac{f_i - f_{i-1}}{x_i - x_{i-1}} - \frac{x_i - x_{i-1}}{6}(M_i - M_{i-1}). \end{aligned} \tag{4.13}$$

Man rechnet schnell nach, dass mit diesen Konstanten die Stetigkeit von $s_3(x)$ in den Stützstellen gewährleistet ist.

Aus der Stetigkeitsbedingung für die erste Ableitung erhält man $(n-1)$ Gleichungen für $M_0, \ldots, M_n$. Es gilt für $x \in [x_{i-1}, x_i]$

$$s_3'(x) = -\frac{M_{i-1}}{2}\frac{(x_i - x)^2}{x_i - x_{i-1}} + \frac{M_i}{2}\frac{(x - x_{i-1})^2}{x_i - x_{i-1}} + c_{i-1}.$$

Für den rechten Randpunkt erhält man

$$\begin{aligned} s_3'(x_{i-0}) &= \frac{M_i}{2}(x_i - x_{i-1}) + \frac{f_i - f_{i-1}}{x_i - x_{i-1}} - \frac{x_i - x_{i-1}}{6}(M_i - M_{i-1}) \\ &= \left(\frac{M_{i-1}}{6} + \frac{M_i}{3}\right)(x_i - x_{i-1}) + \frac{f_i - f_{i-1}}{x_i - x_{i-1}}. \end{aligned}$$

Betrachtet man nun den linken Endpunkt des Intervalls $[x_i, x_{i+1}]$, so erhält man auf die gleiche Weise

$$\begin{aligned} s_3'(x_{i+0}) &= -\frac{M_i}{2}(x_{i+1} - x_i) + \frac{f_{i+1} - f_i}{x_{i+1} - x_i} - \frac{x_{i+1} - x_i}{6}(M_{i+1} - M_i) \\ &= \left(-\frac{M_i}{3} - \frac{M_{i+1}}{6}\right)(x_{i+1} - x_i) + \frac{f_{i+1} - f_i}{x_{i+1} - x_i}. \end{aligned}$$

Setzt man diese Bedingungen gleich, so erhält man $(n-1)$ Gleichungen für $M_0, \ldots, M_n$. Für die zwei fehlenden Gleichungen setzt man

$$2M_0 + \lambda_0 M_1 = g_0, \quad \mu_n M_{n-1} + 2M_n = g_n$$

mit vorgegebenen Werten $\lambda_0, \mu_n \in [0, 1]$ und g_0, g_n. Zur Erfüllung von (4.10) setzt man alle diese Werte gleich Null.

Zur Berechnung der Koeffizienten $M_0, \ldots, M_n$ hat man damit folgendes lineares Gleichungssystem zu lösen

$$\begin{pmatrix} 2 & \lambda_0 & & & & & & \\ \mu_1 & 2 & \lambda_1 & & & & & \\ & \mu_2 & 2 & \lambda_2 & & & & \\ & & & \ddots & & & & \\ & & & & \mu_{n-1} & 2 & \lambda_{n-1} & \\ & & & & & \mu_n & 2 & \end{pmatrix} \begin{pmatrix} M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_{n-1} \\ M_n \end{pmatrix} = \begin{pmatrix} g_0 \\ g_1 \\ g_2 \\ \vdots \\ g_{n-1} \\ g_n \end{pmatrix}.$$

Die Koeffizienten sind

$$\begin{aligned} \mu_i &= \frac{x_i - x_{i-1}}{x_{i+1} - x_{i-1}}, \\ \lambda_i &= \frac{x_{i+1} - x_i}{x_{i+1} - x_{i-1}}, \\ g_i &= \frac{6}{x_{i+1} - x_{i-1}}\left(\frac{f_{i+1} - f_i}{x_{i+1} - x_i} - \frac{f_i - f_{i-1}}{x_i - x_{i-1}}\right), \quad i = 1, \ldots, n-1. \end{aligned}$$

Nach der Lösung dieses Systems erhält man mit (4.12), (4.13) durch Einsetzen in (4.11) die Darstellung des kubischen Splines. *MATLAB–Demo*

Satz 4.6 *Seien $f \in C^4([a, b])$,*

$$h_{\max} = \max_{i=0,\ldots,n-1}(x_{i+1} - x_i), \quad h_{\min} = \min_{i=0,\ldots,n-1}(x_{i+1} - x_i), \quad \beta = \frac{h_{\max}}{h_{\min}} \geq 1.$$

Dann gilt

$$\max_{x\in[a,b]} \left| f^{(r)}(x) - s_3^{(r)}(x) \right| \leq C_r h_{\max}^{4-r} \max_{x\in[a,b]} \left| f^{(4)}(x) \right|$$

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

Das bedeutet, dass für $h_{\max} \to 0$ der kubische Spline samt seinen ersten beiden Ableitungen punktweise gegen f beziehungsweise gegen die entsprechenden Ableitungen von f konvergiert. Innerhalb der Intervalle konvergiert auch noch die dritte Ableitung des kubischen Splines, an den Stützstellen konvergiert der Mittelwert der beiden einseitigen dritten Ableitungen.