UNIVERSITÄT BAYREUTH
MATHEMATISCHES INSTITUT

Adaptive Spline-Approximation in der optimalen Steuerung

Diplomarbeit

von

Florian Bauer

Datum: 30. September 2004

Aufgabenstellung / Betreuung:

Prof. Dr. L. Grüne

Zweiter Gutachter:

Prof. Dr. F. Lempio

Zusammenfassung:
Die vorliegende Arbeit untersucht ein Iterationsverfahren, das zur numerischen Lösung optimaler Steuerungsprobleme angewendet werden soll. Dieses Verfahren verwendet interpolierende Splinefunktionen, um die optimale Wertefunktion numerisch zu approximieren.
Zunächst werden einige theoretische Grundlagen interpolierender Splinefunktionen sowie der diskontierten optimalen Steuerung vorgestellt.
Es folgt die ausführliche Beschreibung und theoretische Betrachtung des Iterationsverfahrens selbst. Außerdem wird die Vorgehensweise der adaptiven Gittererzeugung erklärt und angewendet. Den Abschluss bilden die Darstellung eines nach diesem Iterationsverfahren implementierten Algorithmus' sowie die Diskussion der bei der Anwendung auf verschiedene Beispielprobleme erhaltenen Ergebnisse.

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

Diese Arbeit beschäftigt sich mit der Approximation durch interpolierende Splinefunktionen und ihrer Anwendungsmöglichkeit bei der numerischen Lösung eindimensionaler optimaler Steuerungsprobleme.

Die ersten Werke über Splinefunktionen, im Folgenden Splines genannt, wurden 1946 von Issac J. Schoenberg veröffentlicht, wobei schon Leonhard Euler im 18.Jahrhundert „exponential Euler splines“ (natürlich nicht unter diesem Namen) untersucht hat.
In den 60er, 70er und 80er Jahren erschienen unzählige Arbeiten, die sich mit der Theorie der Splines beschäftigten und diese weiterentwickelten.

In den meisten klassischen Lehrbüchern der numerischen Mathematik tauchen speziell kubische Splines als Lösung eines Interpolationsproblems auf. Ungeachtet der weitreichenden Einsatzgebiete von Splines soll diese klassische Anwendung betrachtet werden.

Einsatz finden die Interpolationssplines in einem Verfahren zur Lösung optimaler Steuerungsprobleme. Aufgaben und Problemstellungen aus vielen verschiedenen Bereichen können als optimale Steuerungsprobleme formuliert werden. Szenarien beispielsweise aus der Ökonomie, der Mechanik, der Biologie oder der Physik, um nur einige zu nennen, lassen sich derart beschreiben.

Grundlage des betrachteten numerischen Lösungsansatzes bildet ein in Grüne und Semmler [13] sowie Grüne [11] dargestelltes Iterationsverfahren, das sich linearer Interpolationsmethoden bedient, um die optimale Wertefunktion eines optimalen Steuerungsproblems auf einem Gitter Γ approximativ zu berechnen. Objekt der Untersuchungen der vorliegenden Arbeit ist die Anwendung der Spline-Interpolation anstelle der linearen Interpolation. Die Frage ist, ob sich die Eigenschaften der Splines für das Lösen von optimalen Steuerungsproblemen ausnutzen lassen. Carl de Boor spricht von „einer großen praktischen Nützlichkeit der

Splines“ auf Grund ihrer lokalen Einfachheit und globalen Flexibilität (siehe de Boor [7], S.2). Zudem approximieren (polynomiale) Interpolationssplines eine durch gegebene Werte an den Interpolationspunkten definierte Funktion im Allgemeinen wesentlich genauer als lineare Interpolationsfunktionen. In Trick und Zin [23] werden kubische Splines verwendet, um die Größe, d.h. die Anzahl der Variablen, von linearen Optimierungsproblemen zu reduzieren. Nun sollen die Splines bei der Lösung nichtlinearer Probleme herangezogen werden.
Bezüglich des Verfahrens aus Grüne [11] muss zunächst untersucht werden, ob sich die Eigenschaften des Verfahrens übertragen lassen und somit die Approximation der optimalen Wertefunktion erfolgreich bleibt, wenn dabei Spline-Interpolation angwendet wird. Dieser Gesichtspunkt soll im Folgenden betrachtet werden.

Die Arbeit gliedert sich in drei größere Teilbereiche. Der Erste (Kapitel 2) befasst sich mit der Spline-Interpolation an sich und bietet ein kurze Einführung. Splines werden definiert und eine Möglichkeit der Berechnung eines kubischen Interpolationssplines zu vorgegeben Werten aufgezeigt. Außerdem wird auf das Konvergenzverhalten kubischer Interpolationssplines eingegangen. Kernpunkt dabei ist die Untersuchung eines Beweises aus Hall [15], der sich mit der Konvergenz eines kubischen Splines gegen die von ihm interpolierte Funktion befasst.

Der zweite Abschnitt umfasst die Kapitel 3,4 und 5. Er stellt die betrachtete Form der optimalen Steuerungsprobleme vor und beschreibt die theoretischen Grundlagen des Verfahrens aus [11] und [13]. Die Vorgehensweise wird dargestellt und die durch die Einbeziehung der Spline-Interpolation auftretenden Änderungen aufgezeigt. Zusätzlich beschrieben wird die Vorgehensweise der adaptiven Gittererzeugung. Durch die sukzessive Berechnung der Approximationen auf geeignet veränderten Gittern wird die Bestimmung der optimalen Wertefunktion deutlich effektiver. Die Auswirkungen dieses Verfahrens werden anhand von Beispielrechnungen erläutert.

In Kapitel 6, dem dritten und letzten Abschnitt, wird zu Beginn das schrittweise Vorgehen bei der Implementierung eines nach dem vorgestellten Verfahren ablaufenden Algorithmus' beschrieben. Anschließend werden die erhaltenen Ergebnisse diskutiert. Dabei steht der Vergleich der durch die unterschiedlichen Interpolationsarten erzielten Ergebnisse im Mittelpunkt. Betrachtet werden vier Beispiele aus dem Bereich der Ökonomie aus Grüne und Semmler[13].

Es sei angemerkt, dass die anglo-amerikanische Notation für Kommazahlen (z.B. 0.1) sowie für Intervalle (z.B. $[0, 1]$) verwendet wird.

An dieser Stelle möchte ich mich bei Herrn Prof. Dr. Lars Grüne für die zahlreichen Anregungen und die hervorragende Betreuung bedanken.

Kapitel 2

Einführung in die Spline-Interpolation

Bevor die Splines bei der Lösung von optimalen Steuerungsproblemen Anwendung finden, werden in diesem Kapitel einige Grundlagen der Spline-Theorie vorgestellt. Nach der Definition und der Darstellung einiger grundlegender Eigenschaften sowie den notwendigen Randbedingungen von Splines folgt die Ausarbeitung einer Variante zur Berechnung kubischer Interpolationssplines. Diese ist wiederum abhängig von der Art der vorliegenden Randbedingungen. Der Schwerpunkt liegt in diesem Kapitel auf dem Beweis einer Aussage über die Konvergenzeigenschaften kubischer Interpolationssplines.

Aus der umfassenden Literatur über Splines erschienen mir die Einführungen in Böhmer [5], Kapitel 1, Grüne [12], Kapitel 4.2, sowie Stoer [22], Kapitel 2.4, am verständlichsten strukturiert. Aus diesem Grund orientiere ich mich im Folgenden weitgehend an diesen.

2.1 Definition und Eigenschaften von Splines

Splines werden durch die Werte an bestimmten Punkten gegeben. Diese Knotenpunkte sind durch ein Gitter gemäß folgender Definition festgelegt:

Definition 2.1 *Sei eine kompakte Menge $\Omega = [a, b] \in \mathbb{R}$ mit $a < b$ gegeben. Ein eindimensionales Gitter Γ auf Ω ist eine Menge von Intervallen $I_i,\ i = 1, ..., l$, mit*

$$\bigcup_{i=1}^{l} I_i = \Omega \quad und \quad intI_i \cap intI_j = \emptyset \quad für\ alle \quad i, j = 0, ..., l,\ i \neq j$$

Die Menge der Intervallendpunkte oder auch Knotenpunkte des Gitters ist durch $\Delta = \{x_i\},\ i = 0, ..., l$, mit $x_0 = a$ und $x_l = b$ gegeben.

Auf dieser Knotenmenge Δ bzw. diesem Gitter Γ definiert sich ein Spline wie folgt:

Definition 2.2 *Sei ein Gitter Γ auf $[x_0, x_n]$ mit Knotenmenge $\Delta = \{x_0, ..., x_n\}$ und $x_0 < x_1 < ... < x_n$ gegeben . Eine Funktion $s_\Delta : [x_0, x_n] \to \mathbb{R}$ heißt zu $\Delta -$ gehöriger Spline vom Grad $k \in \mathbb{N}$, wenn gilt:*

- $s_\Delta \in C^{k-1}[x_0, x_n]$
- *Auf jedem Teilintervall $I_i = [x_{i-1}, x_i],\ i = 1, 2, ..., n$ stimmt s_Δ mit einem Polynom p_i vom Grad k überein, d.h. für $x \in I_i$ kann man schreiben*

$$s_\Delta(x) = a_{i0} + a_{i1}(x - x_{i-1}) + ... + a_{ik}(x - x_{i-1})^k = p_i(x). \tag{2.1}$$

Ein Spline s_Δ ist also derart aus n Polynomen k-ten Grades zusammengesetzt, dass er selbst und die ersten $(k-1)$ Ableitungen an den Knoten x_i, $i = 1, ..., n-1$, keine Sprungstellen besitzen.

Bemerkung 2.3 *Offensichtlich löst der Spline s_Δ das Interpolationsproblem*

$$F(x_i) = y_i,\ i = 0, ..., n,$$

wenn gilt:

$$s_\Delta(x_i) = y_i,\ i = 0, ..., n.$$

Im Folgenden wird generell der Spline auf $\Delta = \{x_0, ..., x_n\}$ mit $x_0 < x_1 < ... < x_n$ betrachtet und s statt s_Δ geschrieben.

2.1.1 Eindeutigkeit

Zunächst stellt sich die Frage, ob der Interpolationsspline s mit $s(x_i) = y_i$ für gegebene Daten $(x_i, y_i),\ i = 0, ..., n$ eindeutig ist. Durch

$$y_{i-1} = S(x_{i-1}) = a_{i0} \quad f\ddot{u}r \quad i = 1, ..., n$$

sind gerade n Koeffizienten aus (2.1) bestimmt und durch

$$y_n = s(x_n) = a_{n0} + a_{n1}(x_n - x_{n-1}) + ... + a_{nk}(x_n - x_{n-1})^k \tag{2.2}$$

noch ein weiterer, da man (2.2) umformen kann zu

$$a_{n1} = \frac{y_n - a_{n0} - a_{n2}(x_n - x_{n-1})^2 - ... - a_{nk}(x_n - x_{n-1})^k}{x_n - x_{n-1}}.$$

Somit sind genau $(n + 1)$ Koeffizienten festgelegt.
Durch diese ist der interpolierende Spline allerdings noch nicht eindeutig bestimmt, da folgende Aussage gilt (siehe auch Grüne [12], Kapitel 4.2):

Satz 2.4 *Für $\Delta = \{x_0, ..., x_n\}$ mit $x_0 < x_1 < ... < x_n$ und $k \in \mathbb{N}$ ist der Raum der zur Knotenmenge Δ gehörigen Splines vom Grad k (bezeichnet mit $\mathcal{S}_{\Delta,k}$) ein $(k+n)-$dimensionaler Vektorraum über $\mathbb{R}$.*
Anders ausgedrückt: Durch $(k+n)$ gegebene Koeffizienten a_{ij} ist der Spline s eindeutig festgelegt.

Beweis: Die Gültigkeit der Vektorraumaxiome für $\mathcal{S}_{\Delta,k}$ ist auf Grund der Definition der Splines sofort zu erkennen.
Die Dimension ist durch die Anzahl der freien Parameter a_{ij} bestimmbar. Das Polynom p_1 auf dem ersten Intervall $[x_0, x_1]$ ist frei wählbar. Somit ergeben sich $(k+1)$ freie Parameter. Da der Spline s nach Definition 2.2 auf der ganzen Menge $[x_0, x_n]$ und demnach auch in den Stützstellen x_i stetig und $(k-1)$-mal stetig differenzierbar ist, sind die Werte der $j-ten$ Ableitung jedes folgenden Teilpolynoms am Intervallanfang durch das jeweils „vorherige“ Polynom bestimmt, d.h.

$$p_i^{(j)}(x_{i-1}) = p_{i-1}^{(j)}(x_{i-1}) \tag{2.3}$$

für $i = 2, ..., n$ und $j = 0, ..., k-1$.
Es folgt damit zwangsläufig aus $p_i^{(j)}(x_{i-1}) = j! \cdot a_{ij}$ und (2.3)

$$a_{ij} = p_{i-1}^{(j)}(x_{i-1})/j! \quad , \; j = 0, ..., k-1, \; i = 2, ..., n.$$

Daher ist von den $(n-1)$ restlichen Polynomen p_i , $i = 2, ..., n$, nur noch jeweils ein Koeffizient, nämlich a_{ik}, frei wählbar.
Insgesamt hat man folglich $(k+1)+(n-1) = (k+n)$ freie Parameter.

□

Folgerung 2.5 *Um einen Interpolationsspline s der Ordnung k eindeutig zu bestimmen, sind außer den Werten an den Knoten noch zusätzliche Bedingungen notwendig, durch die weitere $(k-1)$ Koeffizienten festgelegt sind.*
Diese Bedingungen werden in Form von Randbedingungen, d.h. Vorschriften an s oder an Ableitungen von s in den Punkten x_0 und x_n formuliert.

2.1.2 Randbedingungen am Beispiel kubischer Splines

Eine besondere Rolle spielen besonders in der Praxis die Splines der Ordnung $k = 3$, die kubischen Splines. Grund dafür ist, dass der kubische Spline von allen Funktionen, die gegebene Werte interpolieren, diejenige ist, die die kleinste Krümmung aufweist.
Bevor diese Extremaleigenschaft der Splines bewiesen wird, soll anhand von kubischen Splines darstellt werden, wie die oben genannten Randbedingungen aussehen können:

Lemma 2.6 *Seien die Werte (x_i, y_i), $i = 0, ..., n$ und ein zu diesen Werten gehöriger Interpolationsspline s gegeben. Dann gilt:*
Erfüllt s zusätzlich eine der drei folgenden Bedingungen

$$\begin{array}{lllll} \textit{(i)} & s''(x_0) = s''(x_n) = 0 & & & \textit{(„natürliche“ Randbedingungen),} \\ \textit{(ii)} & s'(x_0) = s'(x_n) & \textit{und} & s''(x_0) = s''(x_n) & \textit{(„periodische“ Randbedingungen),} \\ \textit{(iii)} & s'(x_0) = f'(x_0) & \textit{und} & s'(x_n) = f'(x_n) & \textit{(„hermitsche“ Randbedingungen),} \end{array}$$

so ist s eindeutig bestimmt.

Beweis: Nach Satz 2.4 ist ein kubischer Spline durch (n+3) Koeffizienten a_{ij} aus (2.1) eindeutig bestimmt. Oben wurde gezeigt, dass durch die Werte (x_i, y_i) genau (n+1) Koeffizienten festgelegt sind. Nun kann man jede der obigen Bedingungen als Vorschrift für zwei weitere Koeffizienten ausdrücken, wenn man s in der Form (2.1) für $k = 3$ (also $S(x) = a_{i0} + a_{i1}(x - x_{i-1}) + a_{i2}(x - x_{i-1})^2 + a_{i3}(x - x_{i-1})^3$) betrachtet:

$$\begin{array}{lll} \text{(i)} & s''(x_0) = 0 & \Leftrightarrow a_{12} = 0 \qquad \textit{und} \\ & s''(x_n) = 0 & \Leftrightarrow 2a_{n2} + 6a_{n3}(x_n - x_{n-1}) = 0 \\ & & \Leftrightarrow a_{n2} = -3a_{n3}(x_n - x_{n-1}) \end{array}$$

$$\begin{array}{lll} \text{(ii)} & s'(x_0) = s'(x_n) & \Leftrightarrow a_{11} = a_{n1} + 2a_{n2}(x_n - x_{n-1}) + 3a_{n3}(x_n - x_{n-1})^2 \quad \textit{und} \\ & s''(x_0) = s''(x_n) & \Leftrightarrow a_{12} = 2a_{n2} + 6a_{n3}(x_n - x_{n-1}) \end{array}$$

$$\begin{array}{lll} \text{(iii)} & s'(x_0) = f'(x_0) & \Leftrightarrow a_{11} = f'(x_0) \text{ und} \\ & s'(x_n) = f'(x_n) & \Leftrightarrow a_{n1} = f'(x_n) - 2a_{n2}(x_n - x_{n-1}) - 3a_{n3}(x_n - x_{n-1})^2 \end{array}$$

Somit sind für jeden Interpolationsspline s, der eine der obigen Bedingungen erfüllt, (n+3) Koeffizienten festgelegt. Nach Satz 2.4 folgt daraus die Behauptung.

□

Bemerkung 2.7 *(i) Man kann auch andere Randbedingungen definieren. Dabei ist wichtig, dass durch sie ebenfalls zwei zusätzliche Koeffizienten besimmt sind. Die drei genannten Bedingungen sind allerdings diejenigen, die in der Literatur hauptsächlich zu finden sind.*
(ii) Die natürlichen Randbedingungen lassen sich geometrisch gerade so interpretieren, dass der Spline über x_0 und x_n hinaus linear mit der Steigung $s'(x_0)$ bzw. $s'(x_n)$ fortgesetzt wird.
(iii) Der Algorithmus, der in dieser Arbeit entwickelt wird, wird in MATLAB implementiert und benutzt die MATLAB-eigenen Routinen zur Berechnung der kubischen Splines. MATLAB legt dabei natürliche Randbedingungen fest.

Nun zurück zur Minimalität der Krümmung:
Die Krümmung $\kappa_g(x)$ des Graphen einer zweimal stetig differenzierbaren Funktion $g : [x_o, x_n] \to \mathbb{R}$ am Punkt x ist nach Herrmann [18] definiert durch

$$\kappa_g(x) := \frac{g''(x)}{(1 + g'(x)^2)^{\frac{3}{2}}}.$$

Ist nun $|g'(x)| << 1$ für $x \in [x_0, x_n]$, so ist die Krümmung näherungsweise gleich der zweiten Ableitung:

$$\kappa_g(x) := g''(x)$$

Die Gesamtkrümmung auf $[x_0, x_n]$ kann nun durch $g''(x)$ in der L_2-Norm ausgedrückt werden:

$$\tilde{\kappa}_g := \|g''\|_2 = \left(\int_{x_0}^{x_n} g''(x)^2 dx \right)^{\frac{1}{2}}$$

Die Extremaleigenschaft des kubischen Interpolationssplines zeigt folgender Satz aus Grüne [12], Kapitel 4.2:

Satz 2.8 *Sei $s : [x_0, x_n] \to \mathbb{R}$ ein kubischer Interpolationsspline zu den Daten (x_i, y_i), $i = 0, ..., n$, der eine der Randbedingungen aus Lemma 2.6 erfüllt. Sei zudem $p : [x_0, x_n] \to \mathbb{R}$ eine zweimal stetig differenzierbare Funktion, die ebenfalls das Interpolationsproblem löst und die gleichen Randbedingungen wie s erfüllt. Dann gilt*

$$\tilde{\kappa}_s \leq \tilde{\kappa}_p.$$

Beweis: Zunächst quadriert man die L_2-Norm von p'' und ersetzt p'' durch $s'' + (p'' - s'')$. Es ergibt sich

$$\begin{aligned}
\|p''\|_2^2 &= \int_{x_0}^{x_n} (p''(x))^2 dx = \int_{x_0}^{x_n} (s'' + (p''(x) - s''(x)))^2 dx \\
&= \underbrace{\int_{x_0}^{x_n} (s''(x))^2 dx}_{=\tilde{\kappa}_s^2} + 2 \underbrace{\int_{x_0}^{x_n} s''(x)(p''(x) - s''(x)) dx}_{=:J} + \underbrace{\int_{x_0}^{x_n} (p''(x) - s''(x))^2 dx}_{\geq 0} \\
&\geq \tilde{\kappa}_s^2 + 2J.
\end{aligned}$$

Durch partielle Integration erhält man für J

$$\begin{aligned}
J &= \int_{x_0}^{x_n} s''(x)(p''(x) - s''(x)) dx \qquad (2.4) \\
&= \underbrace{[s''(x)(p'(x) - s'(x))]_{x=x_0}^{x_n}}_{=:a} - \underbrace{\int_{x_0}^{x_n} s'''(x)(p'(x) - s'(x)) dx}_{=:b}.
\end{aligned}$$

Mit jeder der drei Randbedingungen aus Lemma 2.6 gilt für a

$$\begin{aligned}[s''(x)(p'(x)-s'(x))]_{x=x_0}^{x_n} &= s''(x_n)(p'(x_n)-s'(x_n)) \\ &\quad -s''(x_0)(p'(x_0)-s'(x_0)) = 0,\end{aligned}$$

da für natürliche Randbedingungen $s''(x_n) = s''(x_0) = 0$ erfüllt ist, für periodische Randbedingungen der rechte und der linke Term übereinstimmen und für hermitsche Randbedingungen sowohl $s'(x_n) = p'(x_n) = f'(x_n)$ als auch $s'(x_0) = p'(x_0) = f'(x_0)$ gelten.

Folglich ist a in (2.4) gleich Null. Da $s(x)$ auf jedem Intervall $I_i = [x_{i-1}, x_i]$ nach Definition 2.2 ein kubisches Polynom ist, ist $s'''(x) \equiv d_i$ konstant für $x \in I_i$. Deshalb folgt für b

$$\begin{aligned} b &= \int_{x_0}^{x_n} s'''(x)(p'(x)-s'(x))dx = \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} d_i(p'(x)-s'(x))dx \\ &= \sum_{i=1}^{n} d_i \int_{x_{i-1}}^{x_i} (p'(x)-s'(x))dx = \sum_{i=1}^{n} d_i \underbrace{[(p(x_i)-p(x_{i-1})-s(x_i)+s(x_{i-1})]}_{=:c_i}. \end{aligned}$$

Nach Definition stimmen die Werte von s und p an allen Knoten x_j überein, also ist $s(x_i) = p(x_i)$ und $s(x_{i-1}) = p(x_{i-1})$. Damit ist $c_i = 0$ für alle i und daraus resultierend auch $b = 0$.
Demzufolge ist

$$J = a - b = 0$$

und die Behauptung ist bewiesen.

□

Bemerkung 2.9 *Durch diese Extremaleigenschaft lässt sich auch die Bezeichnung Spline erklären. Im Englischen ist ein „spline“ eine dünne Holzlatte. Ist diese Latte nun an gewissen Interpolationspunkten x_i festgehalten, so wird die Biegeenergie, die näherungsweise durch die Krümmung beschrieben ist, minimal. Falls nun die erste Ableitung an den Interpolationspunkten sehr klein ist, also $g'(x_i) << 1$, was einer kleinen Auslenkung entspricht, so stimmt die Splinefunktion mit der Lage der Holzlatte überein.*

2.2 Berechnung von kubischen Interpolationssplines

Es gibt verschiedene Varianten, die Koeffizienten $a_{i0}, ..., a_{i3}$ aus (2.1) eines kubischen Interpolationssplines zu den gegebenen Werten (x_i, y_i) zu bestimmen.

Dieser Abschnitt zeigt eine Möglichkeit, die Koeffizienten direkt aus den ersten Ableitungen der zu interpolierenden Funktion f an den Knoten x_i zu berechnen. Eine ähnliche Vorgehensweise zeigt beispielsweise Böhmer [5], S.20f, wobei dort nur zwischen periodischen und nicht-periodischen Splines unterschieden wird. Alternativ können auch die zweiten Ableitungen gewählt werden (siehe Grüne [12], Kapitel 4.2, oder Stoer [22], S.105f).

Zunächst stellt man die Einzelpolynome dritten Grades in Abhängigkeit von $Ss(x_{i-1})$ und $s(x_i)$ (den Funktionswerten an den Knotenpunkten) und $s'(x_{i-1})$ und $s'(x_i)$ (den Werten der Ableitung an den Knotenpunkten) für $i = 1, ..., n$ dar und zeigt, dass sie eindeutig sind.

Lemma 2.10 *Für ein Intervall $[a, b], a \neq b$ und gegebene Werte $p(a)$, $p(b)$, $p'(a)$, $p'(b)$ existiert genau ein Polynom $p(x)$ dritten Grades, das an den Endpunkten diese Werte annimmt.*
$p(x)$ lässt sich schreiben als

$$\begin{aligned} p(x) &= p(a) + p'(a)(x-a) + \left[3\frac{p(b)-p(a)}{(b-a)^2} - \frac{p'(b)+2p'(a)}{b-a}\right](x-a)^2 \\ &\quad + \left[-2\frac{p(b)-p(a)}{(b-a)^3} + \frac{p'(b)+p'(a)}{(b-a)^2}\right](x-a)^3. \end{aligned}$$

Beweis: Dass $p(x)$ die vorgegebenen Werte an den Endpunkten annimmt, ist leicht durch Einsetzen von a und b, sowie durch Differentiation zu erkennen.

Gäbe es neben p ein weiteres Polynom $\bar{p}$ vom Grad 3 mit den gewünschten Eigenschaften, so würde in a und in b sowohl das Polynom $\tilde{p} := p - \bar{p}$ als auch die erste Ableitung $\tilde{p}'$ verschwinden. Das heißt das Polynom $\tilde{p}$ dritten Grades hätte die beiden doppelten Nullstellen a und b. Somit muss $p = \bar{p}$ sein.

□

Im Folgenden gelte stets die Notation:

$$s_i := s(x_i),\ s'_i := s'(x_i),\ h_i := (x_i - x_{i-1})$$

Nach Lemma 2.10 kann man für $x \in I_i = [x_{i-1}, x_i],\ i = 1, ..., n$, schreiben:

$$\begin{aligned} s(x) &= s_{i-1} + s'_{i-1}(x - x_{i-1}) \\ &\quad + \left[3\frac{s_i - s_{i-1}}{h_i^2} - \frac{s'_i + 2s'_{i-1}}{h_i}\right](x - x_{i-1})^2 \\ &\quad + \left[-2\frac{s_i - s_{i-1}}{h_i^3} + \frac{s'_i + s'_{i-1}}{h_i^2}\right](x - x_{i-1})^3. \end{aligned} \tag{2.5}$$

Dabei stellen die Vorfaktoren der Terme $(x_i - x_{i-1})^i$, $i = 0, ..., 3$, gerade die Koefizienten $a_{i0}, ..., a_{i3}$ aus 2.1 dar. Man benötigt zu deren Berechnung damit lediglich die Werte s_i' für $i = 0, ..., n$, da h_i und s_i durch die Knotenmenge Δ und die Werte y_i an den Knoten gegeben sind. Es wird sich für alle Randbedingungen ein höchstens $(n + 1)$-dimensionales Gleichungssystem mit diagonaldominanter Matrix A ergeben, aus dem sich die s_i' bestimmen lassen. Dieses Gleichungssystem ist nach folgendem Satz eindeutig lösbar (siehe auch Böhmer [5], S.19).

Satz 2.11 *Sei die Matrix A diagonaldominant, d.h.*

$$|a_{ii}| > \sum_{j=1, j\neq i}^{n} |a_{ij}| \quad , i = 1, ..., n.$$

Dann ist A nichtsingulär und das Gleichungssystem $Ax = b$ somit eindeutig lösbar.

Beweis: Es sei $Ax = b$ mit $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $x \in \mathbb{R}^n$ und x_k der betragsgrößte Eintrag von x, also $\|x\|_\infty = |x_k| \neq 0$. Offensichtlich gilt

$$b_i = a_{ii}x_i + \sum_{j=1, j\neq i}^{n} a_{ij}x_j \quad , i = 1, ..., m.$$

Daraus folgt

$$\begin{aligned}
\|b\|_\infty &\geq |a_{kk}x_k + \sum_{j=1, j\neq k}^{n} a_{kj}x_j| \\
&= |a_{kk} + \sum_{j=1, j\neq k}^{n} a_{kj}\frac{x_j}{x_k}| \cdot |x_k| \\
&= |-a_{kk} - \sum_{j=1, j\neq k}^{n} a_{kj}\frac{x_j}{x_k}| \cdot \|x\|_\infty \\
&\geq \underbrace{(|a_{kk}| - \sum_{j=1, j\neq k}^{n} |a_{kj}|)}_{>0} \cdot \|x\|_\infty.
\end{aligned}$$

Die letzte Ungleichung ergibt sich aus Anwendung der Dreiecksungleichung und der Tatsache, dass $\left|\frac{x_j}{x_k}\right| \leq 1$.
Somit gibt es für $b = 0$ nur die triviale Lösung $x = 0$. Daraus folgt die Behauptung.

□

Zur Bestimmung der s_i' wird (2.5) durch Ausmultiplizieren umgeformt zu

$$s(x) = K_{i1}(x)s_{i-1} + K_{i2}(x)s_i + K_{i3}(x)s'_{i-1} + K_{i4}(x)s'_i,\ x \in I_i. \qquad (2.6)$$

Dabei sind

$$\begin{aligned}
K_{i1}(x) &= \frac{2(x-x_{i-1})^3 - 3(x-x_{i-1})^2 h_i + h_i^3}{h_i^3} \\
K_{i2}(x) &= -\frac{2(x-x_{i-1})^3 - 3(x-x_{i-1})^2 h_i}{h_i^3}, \\
K_{i3}(x) &= \frac{(x-x_{i-1})^3 - 2(x-x_{i-1})^2 h_i + (x-x_{i-1})h_i^2}{h_i^2} \\
K_{i4}(x) &= \frac{(x-x_{i-1})^3 - (x-x_{i-1})^2 h_i}{h_i^2}.
\end{aligned}$$

Die Idee ist nun, die Glattheitseigenschaft des Splines in den Stützstellen auszunutzen, um auf das gewünschte Gleichungssystem zu kommen.

Dazu leitet man (2.6) zweimal ab und erhält

$$\begin{aligned}
s'(x) &= \left[\frac{1}{h_i^3}(6(x-x_{i-1})^2 - 6(x-x_{i-1})h_i)\right] s_{i-1} \\
&\quad - \left[\frac{1}{h_i^3}(6(x-x_{i-1})^2 - 6(x-x_{i-1})h_i)\right] s_i \\
&\quad + \left[\frac{1}{h_i^2}(3(x-x_{i-1})^2 - 4(x-x_{i-1})h_i + h_i^2)\right] s'_{i-1} \\
&\quad + \left[\frac{1}{h_i^2}(3(x-x_{i-1})^2 - 2(x-x_{i-1})h_i)\right] s'_i
\end{aligned}$$

sowie

$$\begin{aligned}
s''(x) &= \left[\frac{1}{h_i^3}(12(x-x_{i-1}) - 6h_i)\right] s_{i-1} \\
&\quad - \left[\frac{1}{h_i^3}(12(x-x_{i-1}) - 6h_i)\right] s_i \\
&\quad + \left[\frac{1}{h_i^2}(6(x-x_{i-1}) - 4h_i)\right] s'_{i-1} \\
&\quad + \left[\frac{1}{h_i^2}(6(x-x_{i-1}) - 2h_i)\right] s_{i-1} \qquad \textit{für} \quad x \in I_i.
\end{aligned}$$

Nach Definition 2.2 ist s in den Stützstellen zweimal stetig differenzierbar, also ist der Graph der zweiten Ableitungen in $x_i,\ i = 1, ..., n-1$ stetig.

Somit müssen der linksseitige Grenzwert ($s''(x_i-)$) und der rechtsseitige Grenzwert ($s''(x_i+)$) von $s(x)$ für $x \to x_i,\ i = 1, ..., n-1$, identisch sein.

Es gelten

$$s''(x_i-) = s_{i-1}\frac{6}{h_i^2} - s_i\frac{6}{h_i^2} + s'_{i-1}\frac{2}{h_i} + s'_i\frac{4}{h_i}$$

und

$$s''(x_i+) = -s_i\frac{6}{h_{i+1}^2} + s_{i+1}\frac{6}{h_{i+1}^2} - s'_i\frac{4}{h_{i+1}} - s'_{i+1}\frac{2}{h_{i+1}}.$$

Aus $s''(x_i+) = s''(x_i-)$ folgt

$$\frac{2}{h_i}s'_{i-1} + 4\left(\frac{1}{h_i} + \frac{1}{h_{i+1}}\right)s'_i + \frac{2}{h_{i+1}}s'_{i+1} = -\frac{6}{h_i^2}s_{i-1} + \frac{6}{h_i^2}s_i - \frac{6}{h_{i+1}^2}s_i + \frac{6}{h_{i+1}^2}s_{i+1}.$$

Durch Umformung und Multiplikation mit $\frac{h_i h_{i+1}}{2}$ ergibt sich für $i = 1, ..., n-1$:

$$h_{i+1}s'_{i-1} + 2(h_{i+1} + h_i)s'_i + h_i s'_{i+1} = \underbrace{3\left[h_{i+1}\frac{s_i - s_{i-1}}{h_i} + h_i\frac{s_{i+1} - s_i}{h_{i+1}}\right]}_{=:\delta_i} \tag{2.7}$$

Man erhält auf diese Art $(n-1)$ Gleichungen für die $(n+1)$ Unbekannten s'_0 bis s'_n. Das Gleichungssystem ist also nicht eindeutig lösbar.

Aus den Randbedingungen ergeben sich jedoch zusätzliche Angaben:

Zunächst der periodische Fall, d.h. s wird über $[x_0, x_n]$ hinaus mit der Periode $(x_n - x_0)$ fortgesetzt. Dann gelten

$$s'_0 = s'_n \quad \text{und} \quad s'_1 = s'_{n+1} \quad \text{sowie} \quad h_1 = h_{n+1}. \tag{2.8}$$

Aus (2.7) und (2.8) ergibt sich in Matrixschreibweise das folgende Gleichungssystem der Dimension n:

$$\begin{pmatrix} 2(h_2+h_1) & h_1 & 0 & \cdots & 0 & h_2 \\ h_3 & 2(h_3+h_2) & h_2 & \ddots & & 0 \\ 0 & h_4 & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & & & \ddots & \ddots & h_{n-1} \\ h_n & 0 & \dots & 0 & h_1 & 2(h_1+h_n) \end{pmatrix} \begin{pmatrix} s'_1 \\ s'_2 \\ \vdots \\ \vdots \\ s'_n \end{pmatrix} = \begin{pmatrix} \delta_1 \\ \delta_2 \\ \vdots \\ \vdots \\ \delta_n \end{pmatrix}$$

mit den Werten δ_i aus (2.7) für $i = 1, ..., n-1$ und $\delta_n = 3[h_1\frac{s_n - s_{n-1}}{h_n} + h_n\frac{s_{n+1} - s_n}{h_1}]$. s'_0 ist durch s'_n bestimmt.

Nun der Fall der hermitschen Randbedingungen:
Sind die Werte s'_0 und s'_n durch die Ableitungen von f an den Stellen x_0 und x_n bekannt, also

$$s'_0 = f'(x_0), s'_n = f'(x_n), \tag{2.9}$$

werden nur noch die Werte s'_i für $i = 1, ..., n-1$ benötigt.

In Matrixform erhält man aus (2.7) und (2.9) das Gleichungssystem:

$$\begin{pmatrix} 2(h_2+h_1) & h_1 & 0 & \cdots & 0 & 0 \\ h_3 & 2(h_3+h_2) & h_2 & \ddots & & 0 \\ 0 & h_4 & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & & & \ddots & \ddots & h_{n-2} \\ 0 & 0 & \dots & 0 & h_n & 2(h_n+h_{n-1}) \end{pmatrix} \begin{pmatrix} s'_1 \\ s'_2 \\ \vdots \\ \vdots \\ s'_{n-1} \end{pmatrix} = \begin{pmatrix} \tilde{\delta}_1 \\ \delta_2 \\ \vdots \\ \vdots \\ \tilde{\delta}_{n-1} \end{pmatrix}$$

mit den Werten δ_i aus 2.7 und $\tilde{\delta}_1 = \delta_1 - h_2 s'_0$ sowie $\tilde{\delta}_{n-1} = \delta_{n-1} - h_{n-1} s'_n$.

Sind die Ableitungen von f unbekannt, benötigt man vorgeschriebene Werte $\tilde{s}''_0$ und $\tilde{s}''_n$, die näherungsweise den Werten der zweiten Ableitung des gesuchten Splines in den Randpunkten x_0 und x_n entsprechen.

Für $i = 0$ lautet (2.7)

$$h_1 s'_{-1} + 2(h_1+h_0)s'_0 + h_0 s'_1 = 3\left[h_1 \frac{s_0 - s_{-1}}{h_0} + h_0 \frac{s_1 - s_0}{h_1}\right].$$

Durch Umformen ergibt sich

$$2h_1 s'_0 + 2h_0 s'_0 + h_0 s'_1 = 3h_0 \frac{s_1 - s_0}{h_1} + 2h_1 \frac{s_0 - s_{-1}}{h_0} + h_1 \frac{s_0 - s_{-1}}{h_0} - h_1 s'_{-1}.$$

Ersetzt man nun approximativ den Differenzenquotient $\frac{S_0 - S_{-1}}{h_0}$ durch den Wert $\tilde{S}'_0$ für die erste Ableitung im Punkt x_0, so folgt weiter

$$\begin{aligned} 2h_0 s'_0 + h_0 s'_1 &= 3h_0 \frac{s_1 - s_0}{h_1} + h_1 s'_0 - h_1 s'_{-1} \\ &= 3h_0 \frac{s_1 - s_0}{h_1} + h_1(s'_0 - s'_{-1}), \end{aligned}$$

und nach Division durch h_0

$$2s'_0 + S'_1 = 3\frac{s_1 - s_0}{h_1} + h_1 \frac{(s'_0 - s'_{-1})}{h_0}.$$

Verwendet man nun erneut den Differenzenquotienten $\frac{(s_0'-s_{-1}')}{h_0}$ als Approximation für $\tilde{s}_0''$, so ergibt sich

$$2s_0' + s_1' = 3\frac{s_1 - s_0}{h_1} + h_1\tilde{s}_0''. \tag{2.10}$$

Analog erhält man für (2.7) mit $i = n$

$$S_{n-1}' + 2S_n' = 3\frac{S_n - S_{n-1}}{h_n} - h_n\tilde{S}_n''. \tag{2.11}$$

(2.7) und die beiden zusätzlichen Gleichungen (2.10) und (2.11) lassen sich folgendermaßen in Matrixform schreiben:

$$\begin{pmatrix} 2 & 1 & 0 & \cdots & 0 & 0 \\ h_2 & 2(h_2+h_1) & h_1 & \ddots & & 0 \\ 0 & h_3 & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & & & h_n & 2(h_n+h_{n-1}) & h_{n-1} \\ 0 & 0 & \dots & 0 & 1 & 2 \end{pmatrix} \begin{pmatrix} s_0' \\ s_1' \\ \vdots \\ s_{n-1}' \\ s_n' \end{pmatrix} = \begin{pmatrix} \tilde{\delta}_0 \\ \delta_1 \\ \vdots \\ \delta_{n-1} \\ \tilde{\delta}_n \end{pmatrix}$$

mit den δ_i aus (2.7) für $i = 1, ..., n-1$ und

$$\tilde{\delta}_0 = 3\frac{s_1 - s_0}{h_1} + h_1\tilde{s}_0'' \quad \text{sowie} \quad \tilde{\delta}_n = 3\frac{s_n - s_{n-1}}{h_n} - h_n\tilde{s}_n''.$$

Im Fall der natürlichen Randbedingungen gelten $\tilde{s}_0'' = s_0'' = 0$ und $\tilde{s}_n'' = s_n'' = 0$. Außerdem sind hier nach Bemerkung 2.7 die ersten und die zweiten Ableitungen bei x_0 und $x_n$exakt gleich dem Differenzenquotienten.

Man erhält in Matrixform:

$$\begin{pmatrix} 2 & 1 & 0 & \cdots & 0 & 0 \\ h_2 & 2(h_2+h_1) & h_1 & \ddots & & 0 \\ 0 & h_3 & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & & & h_n & 2(h_n+h_{n-1}) & h_{n-1} \\ 0 & 0 & \dots & 0 & 1 & 2 \end{pmatrix} \begin{pmatrix} s_0' \\ s_1' \\ \vdots \\ s_{n-1}' \\ s_n' \end{pmatrix} = \begin{pmatrix} \hat{\delta}_0 \\ \delta_1 \\ \vdots \\ \delta_{n-1} \\ \hat{\delta}_n \end{pmatrix}$$

mit den δ_i aus (2.7) für $i = 1, ..., n-1$ und

$$\hat{\delta}_0 = 3\frac{s_1 - s_0}{h_1} \quad \text{sowie} \quad \hat{\delta}_n = 3\frac{s_n - s_{n-1}}{h_n}.$$

Somit entsteht wie gefordert für alle aufgeführten Randbedingungen ein Gleichungssystem mit klar ersichtlich diagonaldominanter Matrix A, das nach Satz 2.11 eindeutig lösbar ist.
Mit den durch Auflösen dieses Gleichungssystems erhaltenen Werten s_i', $i = 0, ..., n$, lassen sich die Koeffizienten $K_{ij}(x)$ aus (2.6) und somit der kubische Interpolationsspline s für alle $x \in [x_0, x_n]$ berechnen.

2.3 Einige Konvergenzeigenschaften kubischer Interpolationssplines

Interpolationspolynome haben den Nachteil, dass sie im Allgemeinen nicht gegen die Funktion f, die sie interpolieren, konvergieren. Demgegenüber lässt sich für Interpolationssplines zeigen, dass sie unter schwachen Voraussetzungen an f und das Gitter Γ gegen f konvergieren.

2.3.1 Konvergenz bei Interpolation einer Funktion $f \in C^4[x_0, x_n]$

Dieser Abschnitt stellt ausführlich einen Beweis von C.A.Hall aus [15] dar, der den Interpolationsfehler bei der Spline-Interpolation einer zumindest 4-mal stetig differenzierbaren Funktion abschätzt und zeigt, dass dieser verschwindet, wenn die maximale Intervalllänge des Gitters Γ gegen Null konvergiert.

Es sei (wie teilweise bereits definiert):

$$s_i := s(x_i),\ s_i' := s'(x_i),\ f_i := f(x_i),\ f_i' := f'(x_i),\ h_i := (x_i - x_{i-1})$$

Satz 2.12 *Sei s der kubische Interpolationsspline mit hermitschen Randbedingungen von $f \in C^4[x_0, x_n]$ auf der Stützstellenmenge $\Delta = \{x_0, ..., x_n\}$ mit $x_0 < ... < x_n$. Dann gilt*

$$\|s^{(r)} - f^{(r)}\|_\infty \leq C_r \|f^{(4)}\| h_{max}^{4-r} \quad , r = 0, 1, 2, 3$$

mit

$$C_0 = \frac{5}{384}, \quad C_1 = \frac{9+\sqrt{3}}{216} \quad , \quad C_2 = \frac{3\varrho + 1}{12}, \quad C_3 = \frac{\varrho^2 + 1}{2}$$

und

$$\varrho = \frac{h_{max}}{h_{min}} \quad , \quad h_{max} = \max_{i=1,...,n} \{x_i - x_{i-1}\} \quad , \quad h_{min} = \min_{i=1,...,n} \{x_i - x_{i-1}\} \quad \textit{sowie}$$
$$\|f^{(4)}\| = \sup_{x \in [x_0, x_n]} f^{(4)}(x).$$

Der Beweis erfordert drei vorbereitende Lemmata. Zunächst wird die Differenz zwischen den ersten Ableitungen von f und denen von s in den Stützstellen abgeschätzt. Diese Abschätzung bestimmt später eine Schranke für den Fehler auf dem gesamten Intervall.

Lemma 2.13 *Für s, f, und Δ mit obigen Eigenschaften gilt*

$$|s_i' - f_i'| \leq \frac{1}{24} \|f^{(4)}\| h_{max}^3 \quad , i = 0, ..., n.$$

Beweis: Wie schon gezeigt, gilt für s das Gleichungssystem (2.7)

$$h_{i+1}s'_{i-1} + 2(h_{i+1}+h_i)s'_i + h_i s'_{i+1} = 3\left[h_{i+1}\frac{s_i - s_{i-1}}{h_i} + h_i\frac{s_{i+1}-s_i}{h_{i+1}}\right]$$

für $i = 1, ..., n-1$.
Durch Taylor-Entwicklung lässt sich nach Hall [15], S.8, herleiten, dass folgende Gleichung erfüllt ist:

$$\begin{aligned} h_{i+1}f'_{i-1} + 2(h_{i+1}+h_i)f'_i + h_i f'_{i+1} &= 3\left[h_{i+1}\frac{f_i - f_{i-1}}{h_i} + h_i\frac{f_{i+1}-f_i}{h_{i+1}}\right] \\ &\quad + \frac{1}{24}f^{(4)}(\xi_i)[h_{i+1}(h_i)^3 + h_i(h_{i+1})^3] \end{aligned} \tag{2.12}$$

für $i = 1, ..., n-1$ und $x_{i-1} \le \xi_i \le x_{i+1}$.

Aus (2.7) und (2.12) ergibt sich

$$\begin{aligned} h_{i+1}s'_{i-1} + 2(h_{i+1}+h_i)s'_i + h_i s'_{i+1} &= h_{i+1}f'_{i-1} + 2(h_{i+1}+h_i)f'_i + h_i f'_{i+1} \\ &\quad - \frac{1}{24}f^{(4)}(\xi_i)[h_{i+1}(h_i)^3 + h_i(h_{i+1})^3], \end{aligned}$$

da $f_i = s_i$ für $i = 0, ..., n$ gilt.
Und daraus erhält man durch Umformen

$$\begin{aligned} h_{i+1}(s'_{i-1} - f'_{i-1}) + 2(h_{i+1}+h_i)(s'_i - f'_i) + h_i(s'_{i+1} - f'_{i+1}) = \\ = -\frac{1}{24}f^{(4)}(\xi_i)[h_{i+1}(h_i)^3 + h_i(h_{i+1})^3]. \end{aligned}$$

Weil aus den hermitschen Randbedingungen folgt, dass $s'_i = f'_i$ für $i = 0$ und $i = n$ gilt, kann man in Matrixform schreiben

$$ME = Z \tag{2.13}$$

wobei $[E]_i = s_i - f_i$, $[Z]_i = \frac{-1}{24}f^{(4)}(\xi_i)[h_{i+1}(h_i^3) + h_i(h_{i+1})^3] \quad für\ i = 1, ..., n-1$ und

$$M = \begin{pmatrix} 2(h_2+h_1) & h_1 & 0 & \cdots & 0 & 0 \\ h_3 & 2(h_3+h_2) & h_2 & \ddots & & 0 \\ 0 & h_4 & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & & & \ddots & \ddots & h_{n-2} \\ 0 & 0 & \dots & 0 & h_n & 2(h_n+h_{n-1}) \end{pmatrix} \in \mathbb{R}^{n-1} \times \mathbb{R}^{n-1}$$

Nun werden beide Seiten von (2.13) mit der Diagonalmatrix $D \in \mathbb{R}^{n-1} \times \mathbb{R}^{n-1}$ mit den Diagonalelementen

$$[D]_{ii} = \frac{1}{2(h_{i+1} + h_i)}$$

multipliziert, so dass sich $DME = DZ$ ergibt.
Dabei hat DM die Form:

$$DM = \begin{pmatrix} 1 & \frac{1}{2}\frac{h_1}{h_2+h_1} & 0 & \cdots & 0 & 0 \\ \frac{1}{2}\frac{h_3}{h_3+h_2} & 1 & \frac{1}{2}\frac{h_2}{h_3+h_2} & \ddots & & 0 \\ 0 & \frac{1}{2}\frac{h_4}{h_4+h_3} & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & & & \ddots & \ddots & \frac{1}{2}\frac{h_{n-2}}{h_{n-1}+h_{n-2}} \\ 0 & 0 & \dots & 0 & \frac{1}{2}\frac{h_n}{h_n+h_{n-1}} & 1 \end{pmatrix}$$

Offensichtlich ist

$$DM = I^{(n-1)} + B \tag{2.14}$$

mit $I^{(n-1)}$ der Einheitsmatrix der Dimension $(n-1)$ und

$$B = \begin{pmatrix} 0 & \frac{1}{2}\frac{h_1}{h_2+h_1} & 0 & \cdots & 0 & 0 \\ \frac{1}{2}\frac{h_3}{h_3+h_2} & 0 & \frac{1}{2}\frac{h_2}{h_3+h_2} & \ddots & & 0 \\ 0 & \frac{1}{2}\frac{h_4}{h_4+h_3} & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & & & \ddots & \ddots & \frac{1}{2}\frac{h_{n-2}}{h_{n-1}+h_{n-2}} \\ 0 & 0 & \dots & 0 & \frac{1}{2}\frac{h_n}{h_n+h_{n-1}} & 0 \end{pmatrix}$$

Für B gilt

$$\|B\|_\infty = \frac{1}{2},$$

da für die Zeilensummen

$$\sum_{k=1}^{n} |b_{ik}| = \frac{1}{2}\frac{h_{i+1}}{h_{i+1}+h_i} + \frac{1}{2}\frac{h_i}{h_{i+1}+h_i} = \frac{1}{2}, \quad \textit{für } i = 2, ..., n-2 \quad \textit{und}$$

$$\sum_{k=1}^{n} |b_{ik}| = \frac{1}{2}\frac{h_{i+1}}{h_{i+1}+h_i} < \frac{1}{2} \quad \textit{für } i = 1 \quad \textit{bzw.}$$

$$\sum_{k=1}^{n} |b_{ik}| = \frac{1}{2}\frac{h_{i+1}}{h_{i+1}+h_i} < \frac{1}{2} \quad \textit{für } i = n-1$$

erfüllt ist.

Als Hilfsmittel dient folgender Satz (siehe Wilkinson [24], S.61):

Hilfssatz 2.14 *I sei die Einheitsmatrix. Für die Matrix A gelte* $\|A\|_\infty < 1$ *und* $(I + A)$ *sei invertierbar. Dann ist*

$$\|(I + A)^{-1}\|_\infty \leq (1 - \|A\|_\infty)^{-1}.$$

Beweis:

$$\begin{aligned}
I &= (I + A)^{-1}(I + A) = \\
&= (I + A)^{-1} + (I + A)^{-1}A \\
\Rightarrow \|I\|_\infty &= \|(I + A)^{-1} + (I + A)^{-1}A\|_\infty = \\
&= \|(I + A)^{-1} - (I + A)^{-1}(-A)\|_\infty \geq \\
&\geq \|(I + A)^{-1}\|_\infty - \|(I + A)^{-1}(-A)\|_\infty \geq \\
&\geq \|(I + A)^{-1}\|_\infty - \|(I + A)^{-1}\|_\infty \cdot \|A\|_\infty = \\
&= \|(I + A)^{-1}\|_\infty (1 - \|A\|_\infty)
\end{aligned}$$

Da $\|I\|_\infty = 1$ gilt, folgt die Behauptung und der Hilfssatz ist bewiesen.

□

Hilfssatz 2.14 liefert für (2.14)

$$\|(DM)^{-1}\|_\infty = \|(I + B)^{-1}\|_\infty \leq (1 - \|B\|_\infty)^{-1} = 2,$$

da $\|B\|_\infty = \frac{1}{2}$.

Nun schreibt man E als $E = (DM)^{-1}DZ$ und schätzt es in der Norm ab durch

$$\|E\|_\infty = \|(DM)^{-1}DZ\|_\infty \leq \|(DM)^{-1}\|_\infty \cdot \|DZ\|_\infty \leq 2\|DZ\|_\infty. \tag{2.15}$$

Es bleibt daher noch DZ abzuschätzen. Die Zeilen von DZ haben die Form

$$\begin{aligned}
[DZ]_i &= \frac{1}{2(h_{i+1} + h_i)} \left[(\frac{-1}{24}) f^{(4)}(\xi_i)(h_{i+1}(h_i)^3 + h_i(h_{i+1})^3) \right] \\
&= \frac{-1}{48} f^{(4)}(\xi_i) \frac{h_{i+1}(h_i)^3 + h_i(h_{i+1})^3}{h_{i+1} + h_i}
\end{aligned}$$

für $i = 1, ..., n - 1$ und $\xi_i \in [x_{i-1}, x_{i+1}]$.

Demnach gilt

$$\begin{aligned}
\|DZ\|_\infty &\leq \frac{1}{48}\|f^{(4)}\| \frac{(h_{max})^3(h_i + h_{i+1})}{h_{i+1} + h_i} \\
&\leq \frac{1}{48}\|f^{(4)}\|(h_{max})^3 \qquad mit \quad \|f^{(4)}\| = \max_{x \in [x_0, x_n]} f^{(4)}(x).
\end{aligned} \tag{2.16}$$

Aus (2.15) und (2.16) folgt also

$$\|E\|_\infty \leq \frac{1}{24}\|f^{(4)}\|(h_{max})^3,$$

das heißt für $i = 1, ..., n-1$ ist

$$\|s_i' - f_i'\|_\infty \leq \frac{1}{24}\|f^{(4)}\|(h_{max})^3.$$

Für $i = 0$ und $i = n$ ist auf Grund der vorausgesetzten hermitschen Randbedingungen $s_i' = f_i'$. Somit ist das erste Lemma bewiesen.

□

Nun wird eine Schranke für den Betrag der Differenz zwischen f und dem stückweise kubischen Polynom q bestimmt, dessen Werte der 0-ten und der ersten Ableitung in den Stützstellen mit denen von f übereinstimmen. Sei $q \in C^1[x_0, x_n]$ das (nach Lemma 2.10 eindeutige) stückweise Polynom dritten Grades für das

$$q(x_i) = f(x_i) \quad \text{sowie} \quad q'(x_i) = f'(x_i) \quad f\ddot{u}r\ i = 0, ..., n \tag{2.17}$$

gilt. Analog zu (2.6) kann man für $x \in [x_{i-1}, x_i]$ schreiben:

$$q(x) = K_{i1}(x)q(x_{i-1}) + K_{i2}(x)q(x_i) + K_{i3}(x)q'(x_{i-1}) + K_{i4}(x)q'(x_i)$$

und damit

$$q(x) = K_{i1}(x)f_{i-1} + K_{i2}(x)f_i + K_{i3}(x)f_{i-1}' + K_{i4}(x)f_i', \tag{2.18}$$

wobei wiederum

$$\begin{aligned}
K_{i1}(x) &= \frac{2(x-x_{i-1})^3 - 3(x-x_{i-1})^2 h_i + h_i^3}{h_i^3} \\
K_{i2}(x) &= -\frac{2(x-x_{i-1})^3 - 3(x-x_{i-1})^2 h_i}{h_i^3}, \\
K_{i3}(x) &= \frac{(x-x_{i-1})^3 - 2(x-x_{i-1})^2 h_i + (x-x_{i-1})h_i^2}{h_i^2} \\
K_{i4}(x) &= \frac{(x-x_{i-1})^3 - (x-x_{i-1})^2 h_i}{h_i^2}.
\end{aligned}$$

Die Herleitung dieser Schranke beruht auf der Vorgehensweise in Birkhoff und Priver [4].
Der Beweis nutzt die Darstellung von $d(x) := q(x) - f(x)$ als Lösung einer Randwertaufgabe mit Hilfe einer Greenschen Funktion G. Deshalb erfolgt zunächst eine kurze Ausführung über die Greensche Funktion zu einem Randwertproblem und die für den Beweis nötigen Eigenschaften. Zu detaillierteren Betrachtungen, den Beweisen der folgenden Aussagen sowie der expliziten Bestimmung von G sei auf Spezialliteratur über gewöhnliche Differentialgleichungen, beispielsweise Arndt und Werner [2] oder Myint-U [19], verwiesen.

Einschub: Greensche Funktionen

Definition 2.15 *Sei L der auf* $C^n[a,b]$ *durch*

$$Lz := \mu_n(x)z^{(n)} + ... + \mu_1(x)z' + \mu_0(x)z \tag{2.19}$$

definierte lineare Differentialoperator der Ordnung n mit stetigen Koeffizienten $\mu_i,\ i = 0, ..., n$ *sowie* $R : C^{n-1}[a,b] \to \mathbb{R}^n$ *der Randoperator mit Bedingungen an* $z(a), ..., z^{(n-1)}(a)$ *und* $z(b), ..., z^{(n)}(b)$.
Dann heißt

$$Lz = f\ ,\ Rz = g$$

Randwertproblem für eine lineare Differentialgleichung der Ordnung n mit stetigem f und $g \in \mathbb{R}^n$.
Das Randwertproblem heißt homogen, falls $g = 0$ *und* $f = 0$ *gilt.*

Definition 2.16 *Die Greensche Funktion zur Randwertaufgabe* $Lz = f, Rz = 0$ *mit dem Differentialoperator L der Ordnung* $n > 1$ *ist diejenige Funktion* $G(x,t)$, *die die folgenden Bedingungen erfüllt:*

- *G ist stetig auf* $[a,b]\times[a,b]$ *und für jedes* $t \in [a,b]$ *ist* $G(.,t)$ *(n-2)-mal stetig differenzierbar in* $[a,b]$.
- $G(.,t)$ *besitzt in* $[a,b] \setminus \{t\}$ *stetige Ableitungen der Ordnung n-1 und n. Zusätzlich besteht die Sprungrelation*

  $$\frac{\partial^{n-1}G}{\partial x^{n-1}}(t+,t) - \frac{\partial^{n-1}G}{\partial x^{n-1}}(t-,t) = \frac{1}{\mu_n(x)},$$

  mit $\mu_n(x)$ *aus (2.19).*
- *Für jedes feste* $t \in [a,b]$ *ist* $G(.,t)$ *Lösung der homogenen Differentialgleichung in* $[a,b] \setminus \{t\}$ *(also* $LG(.,t) = 0$*) und für jedes feste* $t \in [a,b]$ *erfüllt* $G(.,t)$ *die homogenen Randbedingungen (also* $RG(.,t) = 0$*).*

Bemerkung 2.17 *Durch die oben aufgeführten Eigenschaften ist die Funktion G zur Randwertaufgabe* $Lz = f, Rz = 0$ *eindeutig definiert, falls der Koeffizient* $\mu_n(x)$ *aus (2.19) keine Nullstelle besitzt.*

Satz 2.18 *Sei f stetig auf dem Intervall* $[a,b]$. *Dann löst die Funktion*

$$\psi(x) = \int_a^b G(x,t)f(t)dt$$

das Randwertproblem $Lz = -f, Rz = 0$.

Mit Hilfe einer Funktion G mit diesen Eigenschaften ist es jetzt möglich, das folgende Lemma zu beweisen.

Lemma 2.19 *Sei $f \in C^4[x_0, x_n]$ und q wie in (2.17) definiert. Dann gilt*

$$\|q^{(r)} - f^{(r)}\|_\infty \leq \alpha_r \|f^{(4)}\| h_{max}^{4-r} \quad , r = 0, 1, 2, 3.$$

Dabei sind

$$\alpha_0 = \frac{1}{384}, \quad \alpha_1 = \frac{\sqrt{3}}{216}, \quad \alpha_2 = \frac{1}{12}, \quad \alpha_3 = \frac{1}{2}.$$

Beweis: Man leitet in dem Beweis eine Schranke für die Differenz auf einem beliebigen Teilintervall $[x_{i-1}, x_i]$ in Abhängigkeit von $\|f^{(4)}\|$ her. Aufgrund der besseren Übersichtlichkeit und zur algebraischen Vereinfachung seien hier zunächst die Punkte $x_{i-1} = 0$ und $x_i = 1$ gewählt. Anschließend wird das Resultat auf allgemeine Intervallendpunkte übertragen.
Durch die Definition von q als Polynom dritten Grades ist sofort ersichtlich, dass $d(x) = q(x) - f(x)$ eine Lösung des Randwertproblems

$$z^{(4)} = -f^{(4)}, \; z(0) = z(1) = z'(0) = z'(1) = 0 \tag{2.20}$$

ist.
Somit lässt sich d nach Satz 2.18 schreiben als

$$d(x) = \int_0^1 \hat{G}(x,t) f^{(4)}(t) dt \tag{2.21}$$

mit $\hat{G}(x,t)$ der zum Randwertproblem (2.20) gehörigen Greenschen Funktion.

Behauptung:
$\hat{G}(x,t)$ hat die Form

$$\hat{G}(x,t) = \frac{1}{12}[|x-t|^3 - (x+t)^3 + 6xt(x+t)(1+xt) - 4x^2t^2(3+xt)].$$

Um diese Behauptung zu beweisen, formt man zuerst $\hat{G}(x,t)$ um. Dadurch vereinfachen sich die notwendigen Differentiationen.

$$\hat{G}(x,t) = \begin{cases} \frac{1}{12}[(x-t)^3 - (x+t)^3 + 6x^2t + 6xt^2 + 6x^3t^2 + 6x^2t^3 - 12x^2t^2 + 4x^3t^3] \\ = \frac{1}{6}[(3t^2 - 2t^3)x^3 + (3t^3 - 6t^2)x^2 + 3t^2x - t^3] & für \quad t \leq x \\ \frac{1}{12}[(t-x)^3 - (x+t)^3 + 6x^2t + 6xt^2 + 6x^3t^2 + 6x^2t^3 - 12x^2t^2 + 4x^3t^3] \\ = \frac{1}{6}[(3t^2 - 2t^3 - 1)x^3 + (3t^3 - 6t^2 + 3t)x^2] & für \quad t \geq x \end{cases} \tag{2.22}$$

Nun bestimmt man die partiellen Ableitungen nach x

$$\frac{\partial^r \hat{G}}{\partial x^r}(x,t) \quad , r = 1, 2, 3.$$

Es ergeben sich:

$$\frac{\partial \hat{G}}{\partial x}(x,t) = \begin{cases} \frac{1}{2}[(3t^2-2t^3)x^2+(2t^3-4t^2)x+t^2] & für \quad t \le x \\ \frac{1}{2}[(3t^2-2t^3-1)x^2+(2t^3-4t^2+2t)x] & für \quad t \ge x \end{cases}$$

$$\frac{\partial^2 \hat{G}}{\partial x^2}(x,t) = \begin{cases} (3t^2-2t^3)x+(t^3-2t^2) & für \quad t \le x \\ (3t^2-2t^3-1)x+(t^3-2t^2+t) & für \quad t \ge x \end{cases}$$

$$\frac{\partial^3 \hat{G}}{\partial x^3}(x,t) = \begin{cases} 3t^2-2t^3 & für \quad t \le x \\ 3t^2-2t^3-1 & für \quad t > x \end{cases} \tag{2.23}$$

Nun lassen sich die verlangten Eigenschaften aus Definition 2.16 zeigen:

- $\hat{G}(x,t)$ ist stetig in $[0,1]\times[0,1]$, da der linksseitige Grenzwert $\hat{G}(t-,t)$ und der rechtsseitige Grenzwert $\hat{G}(t+,t)$ von $\hat{G}(x,t)$ für $x \to t$ übereinstimmen.

  $$\hat{G}(t+,t) = \frac{1}{6}[3t^5-2t^6+3t^5-6t^4+3t^3-t^3] = \frac{1}{6}[-2t^6+6t^5-6t^4+2t^3] = \hat{G}(t-,t)]$$

  $\frac{\partial \hat{G}}{\partial x}(x,t)$ und $\frac{\partial^2 \hat{G}}{\partial x^2}(x,t)$ existieren für jedes feste $t \in [0,1]$ und sind stetig in $[0,1]$, da

  $$\frac{\partial \hat{G}}{\partial x}(t+,t) = \frac{1}{2}[-2t^5+5t^4-4t^3+t^2] = \frac{\partial \hat{G}}{\partial x}(t-,t)$$

  sowie

  $$\frac{\partial^2 \hat{G}}{\partial x^2}(t+,t) = -2t^4+4t^3-2t^2 = \frac{\partial^2 \hat{G}}{\partial x^2}(t-,t).$$

- Für jedes feste $t \in [0,1]$ existiert auch die dritte partielle Ableitung $\frac{\partial^3 \hat{G}}{\partial x^3}(x,t)$ nach x. Sie ist stetig in $[0,1] \setminus \{t\}$ und es gilt

  $$\frac{\partial^3 \hat{G}}{\partial x^3}(t+,t) - \frac{\partial^3 \hat{G}}{\partial x^3}(t-,t) = 1 = \frac{1}{\mu_4(t)}$$

  mit $\mu_4(t) = 1$ aus (2.20).
  Klarerweise existiert auch $\frac{\partial^4 \hat{G}}{\partial x^4}(x,t) = 0$ und ist stetig.

- Für jedes feste $t \in [0,1]$ ist $\hat{G}(x,t)$ offensichtlich Lösung der homogenen Differentialgleichung $L\hat{G}(x,t) = 0$ (also $\frac{\partial^4 \hat{G}}{\partial x^4}(x,t) = 0$) und erfüllt die Randbedingungen $R\hat{G}(x,t) = 0$, da

  $$\hat{G}(0,t) = 0, \quad \hat{G}(1,t) = 0, \quad \frac{\partial \hat{G}}{\partial x}(0,t) = 0, \quad \frac{\partial \hat{G}}{\partial x}(1,t) = 0$$

  erfüllt sind.

Bemerkung 2.20 *(i) $\mu_4(x) = 1$ hat keine Nullstellen und $\hat{G}(x,t)$ ist somit nach Bemerkung 2.17 eindeutig.*
(ii) Die eben nachgewiesenen Eigenschaften werden auch sofort deutlich, wenn man betrachtet, dass $\hat{G}(x,t)$ auch in der Form

$$\hat{G}(x,t) = \begin{cases} \frac{1}{6}(x-t)^3 - P(x,t) & \text{für} \quad t \leq x \\ P(x,t) & \text{für} \quad t \geq x \end{cases}$$

mit $P(x,t) = \frac{x^2(1-t)^2(x+2xt-3t)}{6}$ genau dem kubischen Polynom, für das mit festem t

$$\hat{G}(0,t) = \hat{G}(1,t) = \frac{\partial \hat{G}}{\partial x}(0,t) = \frac{\partial \hat{G}}{\partial x}(1,t) = 0$$

gilt, geschrieben werden kann.

Da wie oben gezeigt die partiellen Ableitungen $\frac{\partial^r \hat{G}}{\partial x^r}(x,t)$ für $r = 1,2,3$ existieren, kann man (2.21) nach x ableiten zu

$$d^{(r)}(x) = \frac{\partial^r}{x^r}\left(\int_0^1 \hat{G}(x,t) f^{(4)}(t)dt\right) = \int_0^1 \left(\frac{\partial^r \hat{G}}{\partial x^r}(x,t)\right) f^{(4)}(t)dt \quad , \; r = 0,1,2,3.$$

Bemerkung 2.21 *Für $r = 0$ ist $\hat{G}$ auf dem gesamten Intervall ≥ 0, d.h. der Betrag der Differenz d ist offensichtlich dann maximal, wenn $f^{(4)}(t) = \max_x |f^{(4)}(x)| = \|f^{(4)}\|$ gilt.*
Für $r = 1,2,3$ trifft dies nicht mehr zu, da die $\frac{\partial^r \hat{G}}{\partial x^r}(x,t)$ variables Vorzeichen haben. Hier tritt die betragsmaximale Differenz dann auf, wenn $f^{(4)}(t) = \max_x |f^{(4)}(x)| = \|f^{(4)}\|$ und zusätzlich das Vorzeichen von $f^{(4)}(t)$ immer dem von $\frac{\partial^r G}{\partial x^r}(x,t)$ entspricht - oder immer das Gegenteilige aufweist.

Somit ist es möglich, $d^{(r)}$ abzuschätzen durch

$$\begin{aligned} d^{(r)}(x) &= \int_0^1 \left(\frac{\partial^r \hat{G}}{\partial x^r}(x,t)\right) f^{(4)}(t)dt \\ &\leq \int_0^1 \left(\frac{\partial^r \hat{G}}{\partial x^r}(x,t)\right) \|f^{(4)}\|dt \\ &= \|f^{(4)}\| \int_0^1 \left(\frac{\partial^r \hat{G}}{\partial x^r}(x,t)\right) dt \qquad \text{für } r = 0,1,2,3. \end{aligned}$$

Es bleibt also noch, obere Schranken für die Integrale

$$\int_0^1 \left(\frac{\partial^r \hat{G}}{\partial x^r}(x,t)\right) dt$$

zu bestimmen. Dafür müssen die jeweiligen Nullstellen der Ableitungspolynome von $\hat{G}$ aus (2.23) bestimmt werden und anschließend die entsprechenden Integrale errechnet werden. Da es sich ausschließlich um elementare Integrationsrechnungen handelt, werden nur die mittels Maple bestimmmten Integralwerte und nicht die ausführlichen Rechenwege angeben.

(i) Sei zunächst $r = 0$:

$$\int_0^1 \hat{G}(x,t)dt = \int_0^1 \frac{1}{12}[|x-t|^3-(x+t)^3+6xt(x+t)(1+xt)-4x^2t^2(3+xt)]dt \quad (2.24)$$

Da $\hat{G}(x,t) \geq 0$ nach Bemerkung 2.21 auf ganz $[0,1]$ gilt, sind keine weiteren Nullstellen zu beachten, und (2.24) lässt sich in der Form (2.22) schreiben:

$$\begin{aligned}\int_0^1 \hat{G}(x,t)dt &= \int_0^x \frac{1}{6}[(3t^2-2t^3)x^3+(3t^3-6t^2)x^2+3t^2x-t^3]dt \\ &\quad + \int_x^1 \frac{1}{6}[(3t^2-2t^3-1)x^3+(3t^3-6t^2+3t)x^2]dt \\ &= \underbrace{\frac{1}{24}x^4-\frac{1}{12}x^3+\frac{1}{24}x^2}_{=:\tilde{d}_0(x)}\end{aligned}$$

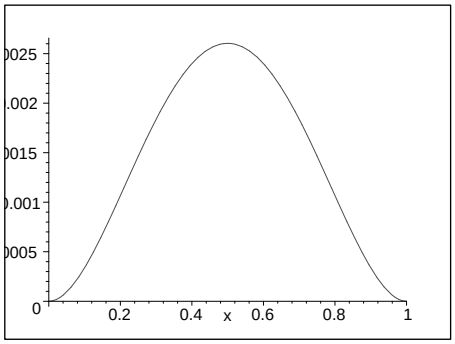


Abbildung 2.1: $\tilde{d}_0(x)$ für $x \in [0,1]$

$\tilde{d}_0(x)$ hat auf $[0,1]$ ein betragsmäßiges Maximum bei $x = \frac{1}{2}$ und es gilt:

$$\left|\tilde{d}_0\left(\frac{1}{2}\right)\right| = \frac{1}{384}$$

Somit ergibt sich für $d(x)$ die obere Schranke

$$|d(x)| \leq \frac{1}{384}\|f^{(4)}\|.$$

(ii) Nun sei $r = 1$, d.h. es ist

$$\int_0^1 \left(\frac{\partial \hat{G}}{\partial x}(x,t) \right) dt$$

zu betrachten.

Zur Erinnerung:

$$\frac{\partial \hat{G}}{\partial x}(x,t) = \begin{cases} \frac{1}{2}[(3t^2 - 2t^3)x^2 + (2t^3 - 4t^2)x + t^2] & f\ddot{u}r \quad t \leq x \\ \frac{1}{2}[(3t^2 - 2t^3 - 1)x^2 + (2t^3 - 4t^2 + 2t)x] & f\ddot{u}r \quad t \geq x \end{cases}$$

Hier sind die zusätzlichen Nullstellen von $\frac{\partial \hat{G}}{\partial x}(x,t)$ im Innern des Intervalls $[0,1]$ zu beachten. Wichtig ist, dass für die Nullstellen jeweils auch entsprechend $t \leq x$ bzw. $t \geq x$ erfüllt ist.

$$\frac{1}{2}[(3t^2 - 2t^3)x^2 + (2t^3 - 4t^2)x + t^2]$$

hat eine Nullstelle bei $t = \frac{3x-1}{2x}$. Diese liegt allerdings für $x \in [0, \frac{1}{3}[$ links von 0. Für $x \in]\frac{1}{2}, 1]$ gilt für die Nullstelle $t > x$. Also ist diese Nullstelle nur auf dem Intervall $x \in [\frac{1}{3}, \frac{1}{2}]$ relevant.

$$\frac{1}{2}[(3t^2 - 2t^3 - 1)x^2 + (2t^3 - 4t^2 + 2t)x]$$

hat eine Nullstelle bei $t = \frac{-x}{2(x-1)}$. Diese Nullstelle liegt für $x \in]\frac{2}{3}, 1]$ rechts von 1. Für $x \in [0, \frac{1}{2}[$ gilt für die Nullstelle $t < x$. Entsprechend ist diese Nullstelle nur auf dem Intervall $x \in [\frac{1}{2}, \frac{2}{3}]$ zu berücksichtigen.

Zusammenfassend gibt es somit im Inneren von $[0,1]$:

$\frac{\partial \hat{G}}{\partial x}(x,t)$ hat	für $x \in [0, \frac{1}{3}[$	keine Nullstelle,	
	für $x \in [\frac{1}{3}, \frac{1}{2}]$	die Nullstelle $t = \frac{3x-1}{2x}$ für $t \leq x$,	
	für $x \in [\frac{1}{2}, \frac{2}{3}]$	die Nullstelle $t = \frac{-x}{2(x-1)}$ für $t \geq x$	und
	für $x \in]\frac{2}{3}, 1]$	wieder keine Nullstelle.	

Es gilt also:

$$\int_0^1 \left(\frac{\partial \hat{G}}{\partial x}(x,t)\right) dt =$$

$$= \begin{cases} \int_0^x \frac{1}{2}[(3t^2-2t^3)x^2+(2t^3-4t^2)x+t^2]dt & \\ \quad + \int_x^1 \frac{1}{2}[(3t^2-2t^3-1)x^2+(2t^3-4t^2+2t)x]dt & für \quad 0 \le x < \frac{1}{3} \\ & \\ \int_0^{\frac{3x-1}{2x}} \frac{1}{2}[(3t^2-2t^3)x^2+(2t^3-4t^2)x+t^2]dt & \\ \quad - \int_{\frac{3x-1}{2x}}^x \frac{1}{2}[(3t^2-2t^3)x^2+(2t^3-4t^2)x+t^2]dt & \\ \quad - \int_x^1 \frac{1}{2}[(3t^2-2t^3-1)x^2+(2t^3-4t^2+2t)x]dt & für \quad \frac{1}{3} \le x \le \frac{1}{2} \\ & \\ -\int_0^x \frac{1}{2}[(3t^2-2t^3)x^2+(2t^3-4t^2)x+t^2]dt & \\ \quad - \int_x^{\frac{-x}{2(x-1)}} \frac{1}{2}[(3t^2-2t^3-1)x^2+(2t^3-4t^2+2t)x]dt & \\ \quad + \int_{\frac{-x}{2(x-1)}}^1 \frac{1}{2}[(3t^2-2t^3-1)x^2+(2t^3-4t^2+2t)x]dt & für \quad \frac{1}{2} \le x \le \frac{2}{3} \\ & \\ -\int_0^x \frac{1}{2}[(3t^2-2t^3)x^2+(2t^3-4t^2)x+t^2]dt & \\ \quad - \int_x^1 \frac{1}{2}[(3t^2-2t^3-1)x^2+(2t^3-4t^2+2t)x]dt & für \quad \frac{2}{3} < x \le 1 \end{cases}$$

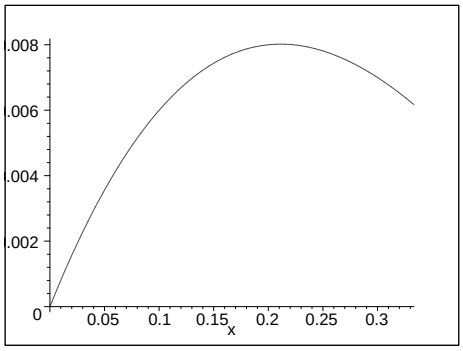
Die Vorzeichen vor den Integralen entsprechen dem Vorzeichen von $\frac{\partial \hat{G}}{\partial x}(x,t)$ im jeweiligen Intervall.

Ausgewertet ergibt sich:

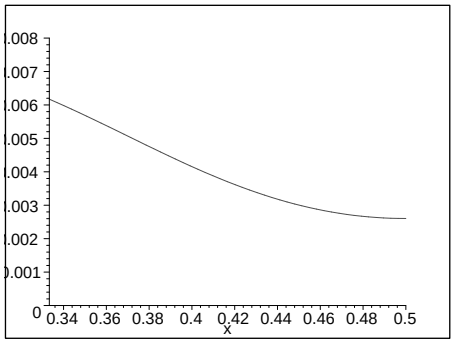
$$\int_0^1 \left(\frac{\partial \hat{G}}{\partial x}(x,t)\right) dt =$$

$$= \begin{cases} \underbrace{\frac{1}{6}x^3 - \frac{1}{4}x^2 + \frac{1}{12}x}_{=:\tilde{d}_{11}(x)} & für \quad 0 \le x < \frac{1}{3} \\ \underbrace{\frac{1}{96}\frac{(16x^6 - 105x^5 + 197x^4 - 162x^3 + 66x^2 - 13x + 1)}{x^3}}_{=:\tilde{d}_{12}(x)} & für \quad \frac{1}{3} \le x \le \frac{1}{2} \\ \underbrace{-\frac{1}{96}\frac{x(16x^5 + 9x^4 - 88x^3 + 104x^2 - 48x + 8)}{(-1+x)^3}}_{=:\tilde{d}_{13}(x)} & für \quad \frac{1}{2} \le x \le \frac{2}{3} \\ \underbrace{-\frac{1}{6}x^3 + \frac{1}{4}x^2 - \frac{1}{12}x}_{=:\tilde{d}_{14}(x)} & für \quad \frac{2}{3} < x \le 1 \end{cases}$$

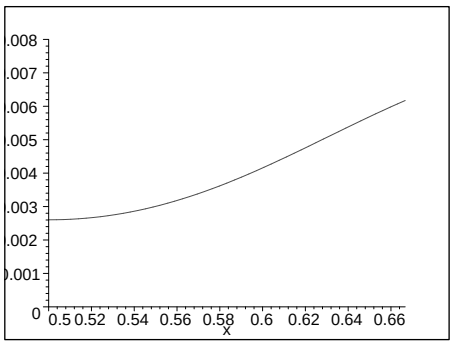
Die Funktionen $\tilde{d}_{1i}(x)$ haben in den entsprechenden Intervallen folgende Form:



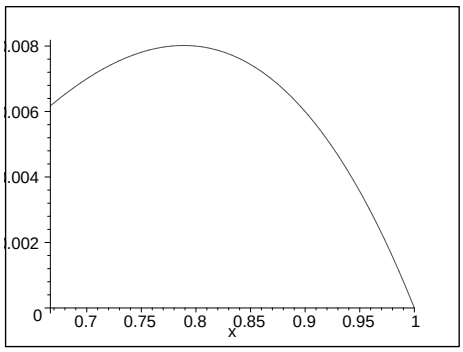
(a) $\tilde{d}_{11}(x)$ für $0 \le x < \frac{1}{3}$



(b) $\tilde{d}_{12}(x)$ für $\frac{1}{3} \le x \le \frac{1}{2}$



(c) $\tilde{d}_{13}(x)$ für $\frac{1}{2} \leq x \leq \frac{2}{3}$



(d) $\tilde{d}_{14}(x)$ für $\frac{2}{3} < x \leq 1$

Abbildung 2.2: $\tilde{d}_{1i}(x)$ für $x \in [0, 1]$

Die Betrags-Maxima auf den entsprechenden Intervallen betragen

$$\begin{aligned}
&\tilde{d}_{11}(x) \quad \text{bei } x = \tfrac{1}{2} - \tfrac{1}{6}\sqrt{3} \quad \text{mit } |\tilde{d}_{11}(\tfrac{1}{2} - \tfrac{1}{6}\sqrt{3})| = \tfrac{\sqrt{3}}{216},\\
&\tilde{d}_{12}(x) \quad \text{bei } x = \tfrac{1}{3} \qquad\quad \text{mit } |\tilde{d}_{12}(\tfrac{1}{3})| \approx 6.17 \cdot 10^{-3},\\
&\tilde{d}_{13}(x) \quad \text{bei } x = \tfrac{2}{3} \qquad\quad \text{mit } |\tilde{d}_{13}(\tfrac{2}{3})| \approx 6.17 \cdot 10^{-3},\\
&\tilde{d}_{14}(x) \quad \text{bei } x = \tfrac{1}{2} + \tfrac{1}{6}\sqrt{3} \quad \text{mit } |\tilde{d}_{14}(\tfrac{1}{2} + \tfrac{1}{6}\sqrt{3})| = \tfrac{\sqrt{3}}{216}.
\end{aligned}$$

Es ergibt sich also $\frac{\sqrt{3}}{216}$ als obere Schranke von $|\int_0^1 (\frac{\partial \hat{G}}{\partial x}(x,t))dt|$ auf dem gesamten Intervall $[0, 1]$ für $x = \frac{1}{2} \pm \frac{1}{6}\sqrt{3}$. Somit gilt:

$$|d'(x)| \leq \frac{\sqrt{3}}{216} \|f^{(4)}\|$$

(iii) Auf dieselbe Weise ergeben sich für $r = 2$ (also $\frac{\partial^2 \hat{G}}{\partial x^2}(x,t)$) mit

$$\frac{\partial^2 \hat{G}}{\partial x^2}(x,t) = \begin{cases} (3t^2 - 2t^3)x + (t^3 - 2t^2) & f\ddot{u}r \quad t \leq x \\ (3t^2 - 2t^3 - 1)x + (t^3 - 2t^2 + t) & f\ddot{u}r \quad t \geq x \end{cases}$$

folgende Nullstellen:

$$\begin{aligned}
\tfrac{\partial^2 \hat{G}}{\partial x^2}(x,t) \text{ hat} \quad &\text{für } x \in [0, \tfrac{1}{3}] \quad \text{die Nullstelle } t = \tfrac{-x}{2x-1} \text{ für } t \geq x,\\
&\text{für } x \in]\tfrac{1}{3}, \tfrac{2}{3}[\quad \text{keine Nullstelle und}\\
&\text{für } x \in [\tfrac{2}{3}, 1] \quad \text{die Nullstelle } t = \tfrac{3x-2}{2(x-1)} \text{ für } t \leq x.
\end{aligned}$$

Man muss daher folgende Integrale berechnen:

$$\int_0^1 \left(\frac{\partial^2 \hat{G}}{\partial x^2}(x,t)\right) dt =$$

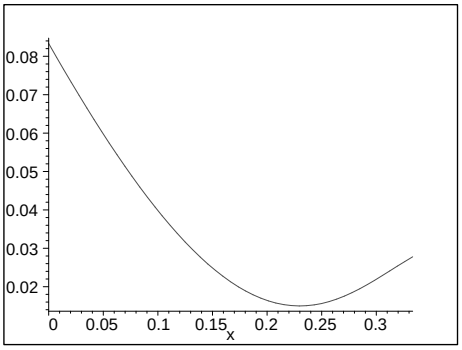
$$= \begin{cases} -\int_0^x (3t^2-2t^3)x + (t^3-2t^2)dt & \\ \quad -\int_x^{\frac{-x}{2x-1}} (3t^2-2t^3-1)x + (t^3-2t^2+t)dt & \\ \quad +\int_{\frac{-x}{2x-1}}^1 (3t^2-2t^3-1)x + (t^3-2t^2+t)dt & für \quad 0 \le x \le \frac{1}{3} \\ -\int_0^x (3t^2-2t^3)x + (t^3-2t^2)dt & \\ \quad -\int_x^1 (3t^2-2t^3-1)x + (t^3-2t^2+t)dt & für \quad \frac{1}{3} < x < \frac{2}{3} \\ \int_0^{\frac{3x-2}{2x-1}} (3t^2-2t^3)x + (t^3-2t^2)dt & \\ \quad -\int_{\frac{3x-2}{2x-1}}^x (3t^2-2t^3)x + (t^3-2t^2)dt & \\ \quad -\int_x^1 (3t^2-2t^3-1)x + (t^3-2t^2+t)dt & für \quad \frac{2}{3} \le x \le 1 \end{cases}$$

Für diese ergibt sich

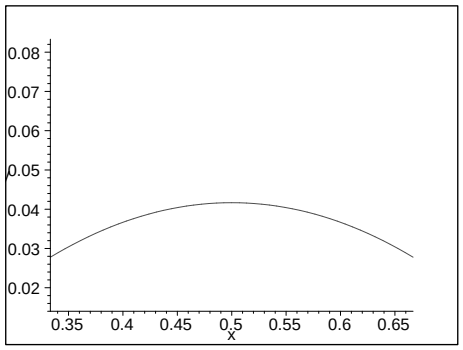
$$\int_0^1 \left(\frac{\partial^2 \hat{G}}{\partial x^2}(x,t)\right) dt =$$

$$= \begin{cases} \underbrace{\dfrac{48x^5+42x^4-100x^3+54x^2-12x+1}{12(1-2x)^3}}_{=:\tilde{d}_{21}(x)} & für \quad 0 \le x \le \frac{1}{3} \\ \underbrace{\dfrac{1}{12}(-6x^2+6x-1)}_{=:\tilde{d}_{22}(x)} & für \quad \frac{1}{3} < x < \frac{2}{3} \\ \underbrace{-\dfrac{(48x^5-282x^4+548x^3-486x^2+204x-33)}{12(2x-1)^3}}_{=:\tilde{d}_{23}(x)} & für \quad \frac{2}{3} \le x \le 1 \end{cases}$$

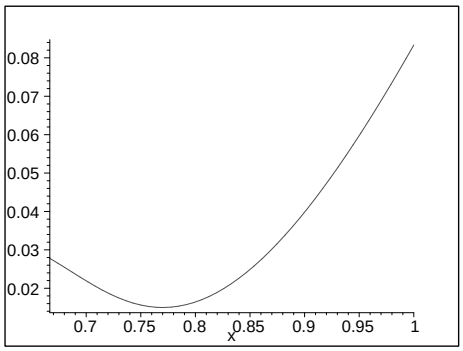
Dabei haben die $\tilde{d}_{2i}$ in den entsprechenden Intervallen die folgenden Graphen:



(a) $\tilde{d}_{21}(x),\ 0 \leq x \leq \frac{1}{3}$



(b) $\tilde{d}_{22}(x),\ \frac{1}{3} < x < \frac{2}{3}$



(c) $\tilde{d}_{23}(x),\ \frac{2}{3} \leq x \leq 1$

Abbildung 2.3: $\tilde{d}_{2i}(x)$ für $x \in [0, 1]$

Nun wird wiederum das Betrags-Maximum der Funktionen auf den entsprechenden Intervallen bestimmt.

$$\begin{array}{lll} \tilde{d}_{21}(x) & \text{bei } x = 0 & \text{mit } |\tilde{d}_{21}(0)| = \frac{1}{12}, \\ \tilde{d}_{22}(x) & \text{bei } x = \frac{1}{2} & \text{mit } |\tilde{d}_{22}(\frac{1}{2})| = \frac{1}{24}, \\ \tilde{d}_{23}(x) & \text{bei } x = 1 & \text{mit } |\tilde{d}_{23}(1)| = \frac{1}{12}. \end{array}$$

Man erhält $\frac{1}{12}$ als obere Schranke von $|\int_0^1 \frac{\partial^2 \hat{G}}{\partial x^2}(x,t)dt|$ auf $[0,1]$ für $x = 0$ und $x = 1$.

Somit gilt

$$|d''(x)| \leq \frac{1}{12}\|f^{(4)}\|.$$

(iv) Bei $r = 3$ (also $\frac{\partial^3 \hat{G}}{\partial x^3}(x,t)$)) mit

$$\frac{\partial^3 \hat{G}}{\partial x^3}(x,t) = \begin{cases} (3t^2 - 2t^3) & für \quad t \leq x \\ (3t^2 - 2t^3 - 1) & für \quad t > x \end{cases}$$

gibt es dagegen wieder keine weiteren Nullstellen im Inneren von $[0,1]$. Folglich gilt

$$\int_0^1 \frac{\partial^3 \hat{G}}{\partial x^3}(x,t)dt = \int_0^x (3t^2 - 2t^3)dt - \int_x^1 (3t^2 - 2t^3 - 1)dt \quad für\ 0 \leq x \leq 1.$$

Daraus ergibt sich

$$\int_0^1 \frac{\partial^3 \hat{G}}{\partial x^3}(x,t)dt = \underbrace{-x^4 + 2x^3 - x + \frac{1}{2}}_{=:\tilde{d}_3(x)} \quad für \quad 0 \leq x \leq 1.$$

$\tilde{d}_3(x)$ hat die Form

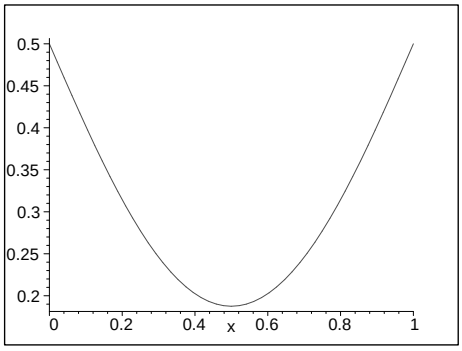


Abbildung 2.4: $\tilde{d}_3(x)$ für $x \in [0,1]$

Somit ist $\tilde{d}_3(x)$ betragsmaximal für $x \in [0,1]$ bei $x = 0$ und $x = 1$ mit $\tilde{d}_3(0) = \tilde{d}_3(1) = \frac{1}{2}$. Dementsprechend gilt:

$$|d'''(x)| \leq \frac{1}{2}\|f^{(4)}\|.$$

Damit sind die Konstanten $\tilde{\alpha_r}$ auf dem speziellen Intervall $[0,1]$ bewiesen.

Durch exakt dieselbe Vorgehensweise erhält man auf dem allgemeinen Intervall $[x_{i-1}, x_i]$ folgende obere Schranken für den Betrag von $d^{(r)}(x)$, $r = 0,1,2,3,$:

$$\begin{aligned} |d(x)| &\leq \frac{1}{384}\|f^{(4)}\|(x_i - x_{i-1})^4, \\ |d'(x)| &\leq \frac{\sqrt{3}}{216}\|f^{(4)}\|(x_i - x_{i-1})^3, \\ |d''(x)| &\leq \frac{1}{12}\|f^{(4)}\|(x_i - x_{i-1})^2, \\ |d'''(x)| &\leq \frac{1}{2}\|f^{(4)}\|(x_i - x_{i-1}). \end{aligned} \tag{2.25}$$

Die hierbei benötigte Greensche Funktion $\tilde{G}$ ist analog zu $\hat{G}$

$$\tilde{G}(x,t) = \begin{cases} \frac{1}{6}(x-t)^3 - \tilde{P}(x,t) & für \quad t \leq x \\ \tilde{P}(x,t) & für \quad t \geq x \end{cases}$$

mit $\tilde{P}(x,t)$ genau dem kubischen Polynom, für das mit festem t die Gleichungen $\tilde{G}(x_{i-1},t) = \tilde{G}(x_i,t) = \frac{\partial \tilde{G}}{\partial x}(x_{i-1},t) = \frac{\partial \tilde{G}}{\partial x}(x_i,t) = 0$ gelten.

Die Ausdrücke in (2.25) sind auf dem längsten Teilintervall am größten, d.h. auf $[x_0, x_n]$ wird $(x_i - x_{i-1})^{4-r} = h_i^{4-r}$ in (2.25) ersetzt durch h_{max}^{4-r} und Lemma 2.19 ist bewiesen.

□

Als dritte Vorbereitung wird eine Schranke für die Differenz zwischen diesem q und s sowie zwischen deren Ableitungen bestimmt, die, wie man später sehen wird, von der Schranke aus Lemma 2.13 abhängt.

Lemma 2.22 *Sei $f \in C^4[x_0, x_n]$, s der hermitsche Interpolationsspline zu f auf Δ und q sei wiederum wie in (2.17) definiert. Dann gilt:*

$$\|s^{(r)} - u^{(r)}\|_\infty \leq \gamma_r \|f^{(4)}\| h_{max}^{4-r} \quad , r = 0, 1, 2, 3.$$

Dabei sind $\gamma_0 = \frac{1}{96}$, $\gamma_1 = \frac{1}{24}$, $\gamma_2 = \frac{\varrho}{4}$, $\gamma_3 = \frac{\varrho^2}{2}$ und wie oben $\varrho = \frac{h_{max}}{h_{min}}$.

Beweis: Aus (2.6) und (2.18) ergibt sich für $x \in [x_{i-1}, x_i]$, $i = 1, .., n$

$$\begin{aligned} s(x) - q(x) &= K_{i1}(x)s_{i-1} + K_{i2}(x)s_i + K_{i3}(x)s'_{i-1} + K_{i4}(x)s'_i \\ &\quad -(K_{i1}(x)f_{i-1} + K_{i2}(x)f_i + K_{i3}(x)f'_{i-1} + K_{i4}(x)f'_i) \\ &= K_{i3}(x)(s'_{i-1} - f'_{i-1}) + K_{i4}(x)(s'_i - f'_i), \end{aligned}$$

da $s_i = f_i$ für $i = 0, ..., n$. Dabei sind die $K_{ij}(x)$ wie in (2.6) definiert.

Nach Lemma 2.13 ist $s_i' - f_i'$ für $i = 0, ..., n$ beschränkt durch $\frac{1}{24}\|f^{(4)}\|(h_{max})^3$. Also gilt

$$s(x) - q(x) \leq \frac{1}{24}\|f^{(4)}\|(h_{max})^3(K_{i3}(x) + K_{i4}(x)).$$

Durch r-maliges Ableiten ergibt sich in der ∞-Norm

$$\|s(x)^{(r)} - q(x)^{(r)}\|_\infty \leq \frac{1}{24}\|f^{(4)}\|(h_{max})^3(\|K_{i3}(x)^{(r)} + K_{i4}(x)^{(r)}\|_\infty). \tag{2.26}$$

Es bleibt also noch

$$\max_{x\in[x_{i-1},x_i]}\{|K_{i3}(x)^{(r)} + K_{i4}(x)^{(r)}|\} \tag{2.27}$$

zu bestimmen. Die Berechnung erleichtert sich etwas, wenn man (2.27) mit der Dreiecksungleichung abschätzt durch

$$\max_{x\in[x_{i-1},x_i]}\{|K_{i3}(x)^{(r)} + K_{i4}(x)^{(r)}|\} \leq \max_{x\in[x_{i-1},x_i]}\{\underbrace{|K_{i3}(x)^{(r)}| + |K_{i4}(x)^{(r)}|}_{=:K_r(x)}\}.$$

Für $r = 0$ zeigt der folgende Abschnitt eine ausführliche Herleitung der Bestimmung von K_r. Analog erhält man die Ergebnisse für die Fälle $r = 1, 2, 3$.

Sei $r = 0$, dann gilt

$$\begin{aligned} K_0(x) &= |K_{i3}(x)| + |K_{i4}(x)| \\ &= \left|\frac{(x - x_{i-1})^3 - 2(x - x_{i-1})^2 h_i + (x - x_{i-1})h_i^2}{h_i^2}\right| \\ &\quad + \left|\frac{(x - x_{i-1})^3 - (x - x_{i-1})^2 h_i}{h_i^2}\right|. \end{aligned}$$

Es ist leicht nachzurechnen, dass sowohl K_{i3} als auch K_{i4} auf dem betrachteten Intervall einzig zwei Nullstellen in den Intervallendpunkten x_{i-1} und x_i haben. K_{i3} ist im Innern des angegebenen Intervalls stets > 0, K_{i4} stets < 0. Somit gilt

$$\begin{aligned} K_0(x) &= |K_{i3}(x)| + |K_{i4}(x)| \\ &= \frac{(x - x_{i-1})^3 - 2(x - x_{i-1})^2 h_i + (x - x_{i-1})h_i^2}{h_i^2} \\ &\quad - \frac{(x - x_{i-1})^3 + (x - x_{i-1})^2 h_i}{h_i^2} \\ &= \frac{-(x - x_{i-1})^2 h_i + (x - x_{i-1})h_i^2}{h_i^2}. \end{aligned}$$

Eine Extremstelle liegt vor, wenn

$$K_0'(x) = \frac{-2(x - x_{i-1})h_i + h_i^2}{h_i^2} = 0 \quad \Leftrightarrow \quad x = \frac{-h_i^2}{-2h_i} + x_{i-1} = \frac{h_i}{2} + x_{i-1}.$$

Da die zweite Ableitung $K_0''(x) = -2h_i$ kleiner als Null ist, hat K_0 bei $x = \frac{h_i}{2} + x_{i-1}$ ein Maximum.

Damit ist also die obere Schranke von $K_0(x)$ auf $[x_{i-1}, x_i]$

$$\begin{aligned} K_0\left(\frac{h_i}{2} + x_{i-1}\right) &= \frac{-(\frac{h_i}{2} + x_{i-1} - x_{i-1})^2 h_i + (\frac{h_i}{2} + x_{i-1} - x_{i-1}) h_i^2}{h_i^2} \\ &= \frac{-\frac{h_i 3}{4} + \frac{h_i^3}{2}}{h_i^2} = \frac{h_i}{4}. \end{aligned}$$

Auf dieselbe Weise berechnet, ergeben sich auf dem Intervall $[x_{i-1}, x_i]$ die Maximalwerte

$$1 \text{ von } K_1(x), \quad \frac{6}{h_i} \text{ von } K_2(x) \quad \text{und} \quad \frac{12}{h_i^2} \text{ von } K_3(x).$$

Auf der gesamten Menge $[x_0, x_n]$ gelten somit offensichtlich die oberen Schranken

$$\begin{array}{llll} \hat{K}_0 = \frac{h_{max}}{4} & \text{für } K_0(x) & , \quad \hat{K}_1 = 1 & \text{für } K_1(x), \\ \hat{K}_2 = \frac{6}{h_{min}} & \text{für } K_2(x) & , \quad \hat{K}_3 = \frac{12}{h_{min}^2} & \text{für } K_3(x). \end{array}$$

(2.26) mit den oben festgelegten $\hat{K}_i$ impliziert

$$\begin{array}{lcll} \|s(x) - q(x)\|_\infty & \leq & \frac{1}{24}\|f^{(4)}\|(h_{max})^3 \frac{h_{max}}{4} & = \frac{1}{96}\|f^{(4)}\|(h_{max})^4, \\ \|s(x)' - q(x)'\|_\infty & \leq & \frac{1}{24}\|f^{(4)}\|(h_{max})^3, & \\ \|s(x)'' - q(x)''\|_\infty & \leq & \frac{1}{24}\|f^{(4)}\|(h_{max})^3 \frac{6}{h_{min}} & = \frac{\varrho}{4}\|f^{(4)}\|(h_{max})^2, \\ \|s(x)''' - q(x)'''\|_\infty & \leq & \frac{1}{24}\|f^{(4)}\|(h_{max})^3 \frac{12}{h_{min}^2} & = \frac{\varrho^2}{2}\|f^{(4)}\|(h_{max}), \end{array}$$

und damit die Behauptung.

□

Mit den Aussagen aus Lemma 2.19 und Lemma 2.22 lässt sich nun Satz 2.12 beweisen:

Beweis: Aus Lemma 2.19 und Lemma 2.22 folgt nach der Dreiecksungleichung für $r = 0, 1, 2, 3$

$$\begin{aligned} \|s^{(r)} - f^{(r)}\| &= \|q^{(r)} - f^{(r)} - (q^{(r)} - s^{(r)})\| \\ &\leq \|q^{(r)} - f^{(r)}\| - \|q^{(r)} - s^{(r)}\| \\ &\leq \|q^{(r)} - f^{(r)}\| + \|q^{(r)} - s^{(r)}\| \\ &\leq \gamma_r \|f^{(4)}\| h_{max}^{4-r} + \alpha_r \|f^{(4)}\| h_{max}^{4-r} \\ &= C_r \|f^{(4)}\| h_{max}^{4-r} \end{aligned}$$

mit den oben bestimmten α_r und γ_r, also

$$\begin{aligned} C_0 &= \alpha_0 + \gamma_0 &&= \tfrac{5}{384}, \\ C_1 &= \alpha_1 + \gamma_1 &&= \tfrac{9+\sqrt{3}}{216}, \\ C_2 &= \alpha_2 + \gamma_2 &&= \tfrac{3\varrho+1}{12}, \\ C_3 &= \alpha_3 + \gamma_3 &&= \tfrac{\varrho^2+1}{2}, \end{aligned}$$

womit der Satz bewiesen ist.

□

Bemerkung 2.23 *C_0 ist die optimale, d.h. minimale obere Schranke. Für $r = 1, 2, 3$ können noch bessere Schranken $\tilde{C}_1, \tilde{C}_2, \tilde{C}_3$ gefunden werden. In Hall und Meyer [16] werden die Schranken $\tilde{C}_1 = \frac{1}{24}$, $\tilde{C}_2 = \frac{3}{8}$ und $\tilde{C}_3 = \frac{\beta+\beta^{-1}}{2}$ bewiesen. Dann ist auch $\tilde{C}_1$ optimal. Für diese Arbeit ist jedoch der Fall $r = 0$ entscheidend.*

Bemerkung 2.24 *Dieser Satz lässt sich nicht auf natürliche Splines übertragen.*

2.3.2 Konvergenz bei Interpolation einer Funktion $f \in C[x_0, x_n]$

Da in der Praxis im Allgemeinen nicht angenommen werden kann, dass die interpolierte Funktion viermal stetig differenzierbar ist, ist es interessant, welche Konvergenzaussage man unter schwächeren Bedingungen an f machen kann.
Es lässt sich zeigen, dass auch der Abstand zwischen einer nicht zwingend differenzierbaren Funktion f und dem periodischen Interpolationsspline s gegen Null konvergiert, wenn die Größe der Gitterzellen gegen Null konvergiert. Für den Beweis sei auf Ahlberg u.a. [1], S.22f, oder Böhmer [5], S.26f, verwiesen.

Satz 2.25 *Sei f stetig auf $[x_0, x_n]$ und sei $\{\Gamma_k\}$ eine Folge von Gittern auf $[x_0, x_n]$ mit $\lim_{k\to\infty} h_{max,k} = 0$ und $\frac{h_{max,k}}{h_{min,k}} \leq \sigma < \infty$.*

Ist s_k der periodische Interpolationsspline zu f auf der zu dem Gitter Γ_k gehörenden Knotenmenge Δ_k, dann gilt

$$\lim_{k\to\infty} \|f - s_k\|_\infty = 0. \tag{2.28}$$

Ist f zusätzlich Hölder-stetig von der Ordnung $\alpha \in]0, 1]$, so ist schärfer:

$$\|f - s_k\|_\infty \leq C \cdot (h_{max,k})^\alpha \tag{2.29}$$

mit einer geeigneten Konstante C.

Korollar 2.26 *Aussage (2.28) trifft nach Böhmer [5] ebenfalls zu, wenn die zweiten Ableitungen von s_k folgenden Randbedingungen genügen:*

$$2s_k''(x_0) + \lambda_{k,0} s_k''(x_1) = \delta_{k,0}$$

und

$$\rho_{k,n_k} s_k''(x_{n_k-1}) + 2s_k''(x_{n_k}) = \delta_{k,n_k},$$

mit

$$\sup_k \{|\lambda_{k,0}|, |\rho_{k,n_k}|\} < 2$$

und

$$h_{max,k}^2 (|\delta_{k,0}| + |\delta_{k,n_k}|) \to 0 \quad für\ k \to \infty.$$

Ist wiederum f zusätzlich Hölder-stetig von der Ordnung $\alpha \in]0,1]$ und es gilt

$$h_{max,k}^{2-\alpha} (|\delta_{k,0}| + |\delta_{k,n_k}|) \to 0\ für\ k \to \infty,$$

so ist (2.29) ebenfalls erfüllt.

Folgerung 2.27 *Nach Korollar 2.26 gilt für einen natürlichen Spline $\hat{s}_k$ auf der Knotenmenge Δ_k zu dem Gitter Γ_k:*

Sind

$$\lambda_{k,0} \hat{s}_k''(x_1) = \delta_{k,0} \quad mit\ \delta_{k,0} > 0$$

und

$$\rho_{k,n_k} \hat{s}_k''(x_{n_k-1}) = \delta_{k,n_k} \quad mit\ \delta_{k,n_k} > 0$$

sowie die anderen Voraussetzungen aus Korollar 2.26 erfüllt, so gilt auch für $\hat{s}_k$ Gleichung (2.28) sowie Gleichung (2.29) bei zusätzlicher Hölder-Stetigkeit von f.

Kapitel 3

Optimale Steuerungsprobleme

Dieses Kapitel stellt in Anlehnung an Grüne [11] die Probleme dar, die durch das später folgende Verfahren gelöst werden sollen. Es wird die typische Form der Problemstellung gezeigt und die für diese Arbeit wichtigen Eigenschaften diskutiert. Obwohl nur eindimensionale Probleme gelöst werden sollen, werden die folgenden Abschnitte allgemein in der Dimension $d \in \mathbb{N}$ formuliert.
Da für die Praxis, also die nachfolgende Anwendung des Lösungsalgorithmus', nur die Eigenschaften an sich von Bedeutung sind (nicht aber deren analytische Herleitung) wird in manchen Fällen auf die langwierigen Beweise verzichtet. Verweise auf tiefergehende Literatur sind beigefügt.

Grundlage der optimalen Steuerung bildet die Betrachtung eines Kontrollsystems.

3.1 Kontrollsysteme

Kontrollsysteme sind folgendermaßen definiert:

Definition 3.1 (Kontrollsystem) *(i) Ein Kontrollsystem in kontinuierlicher Zeit $\mathbb{T} = \mathbb{R}$ im $\mathbb{R}^d, d \in \mathbb{N}$, ist gegeben durch die gewöhnliche Differentialgleichung*

$$\frac{d}{dt}x(t) = f(x(t), u(t)), \tag{3.1}$$

wobei $f : \mathbb{R}^d \times U \to \mathbb{R}^d$ ein parameterabhängiges stetiges Vektorfeld ist.

(ii) Ein Kontrollsystem in diskreter Zeit $\mathbb{T} = h\mathbb{Z} = \{hk | k \in \mathbb{Z}\}$ im $\mathbb{R}^d, d \in \mathbb{N}$, ist für ein $h > 0$ gegeben durch die Differenzengleichung

$$x(t + h) = f_h(x(t), u(t)), \tag{3.2}$$

wobei $f_h : \mathbb{R}^d \times U \to \mathbb{R}^d$ eine stetige Abbildung ist. h ist die zeitliche Schrittweite.

Die Menge $U \subset \mathbb{R}^m$ heißt Kontrollwertebereich und definiert die Werte, die $u(t)$ für $t \in \mathbb{R}$ annehmen darf.
$\mathcal{U}$ bzw. $\mathcal{U}_h$ sei der Raum der zulässigen Kontrollfunktionen, d.h.

$$\mathcal{U} := \{u : \mathbb{R} \to U | u \text{ zulässig}\}$$

bzw.

$$\mathcal{U}_h := \{u_h : h\mathbb{Z} \to U | u_h \text{ zulässig}\}.$$

Bemerkung 3.2 *(i) Im Folgenden sei bei der numerischen Lösung optimaler Steuerungsprobleme U immer als kompakt vorausgesetzt. Diese Annahme führt im Allgemeinen zu keinerlei Einschränkungen, da in der Praxis ebenfalls keine unendlich großen oder kleinen Kontrollwerte möglich sind.*

(ii) Aus Gründen, die später erläutert werden, erweist es sich als günstig, die Werte $u_h(t)$ der diskreten Kontrollfunktion u_h aus einer endlichen Menge $\tilde{U} \subset U$ zu wählen. In diesem Fall werden sich ähnlich gute Resultate wie bei U ergeben, falls $\tilde{U}$ hinreichend „dicht“ in U liegt.

Nun stellt sich die Frage, was zulässig in diesem Zusammenhang bedeutet. Für die Bestimmung von $\mathcal{U}$ bzw. $\mathcal{U}_h$ spielen zwei Kriterien eine Rolle:
Erstens sollte eine möglichst große Menge an Funktionen zugelassen werden, und zweitens sollte eine eindeutige Lösung existieren.
Im zeitdiskreten Fall ist es kein Problem, diese beiden Kriterien gut zu erfüllen. Es werden alle möglichen Funktionen von $h\mathbb{Z}$ nach U zugelassen, also

$$\mathcal{U}_h := \{u_h : h\mathbb{Z} \to U\}.$$

Somit sind keine u_h ausgeschlossen und per Induktion lässt sich zeigen, dass für jeden Anfangswert x_0 und jede Kontrollfunktion $u_h \in \mathcal{U}_h$ eine eindeutige Lösung $\Phi(t, x_0, u_h)$ in positiver Zeitrichtung von (3.2) existiert.

D.h. für $\Phi : h\mathbb{N}_0 \times \mathbb{R}^d \times \mathcal{U}_h \to \mathbb{R}^d$ gilt

$$\Phi_h(0, x_0, u_h) = x_0 \quad \textit{und} \quad \Phi_h(t + h, x_0, u_h) = f_h(\Phi_h(t, x_0, u_h), u_h(t)). \tag{3.3}$$

Im kontinuierlichen Fall ist dies nicht ganz so einfach. Eine naheliegende Idee ist, für $\mathcal{U}$ die Menge der stetigen Funktionen mit Werten in U zu wählen. Bekanntlich lässt sich hierbei für Lösungen gewöhnlicher Differentialgleichungen eine Existenz- und Eindeutigkeitsaussage machen, wenn f zusätzlich Lipschitz-stetig in x ist.
Es zeigt sich allerdings, dass dieses Kriterium in der Praxis der optimalen Steuerung zu streng ist. Auch für viele einfache Beispiele sind die optimalen Steuerungsstrategien unstetig in t.
Als eine bessere Alternative erweist sich der Raum der messbaren Funktionen.

Definition 3.3 *Sei $I = [a, b] \subset \mathbb{R}$ ein abgeschlossenes Intervall.*

(i) Eine Funktion $g : I \to \mathbb{R}^m$ heißt stückweise konstant, falls eine Zerlegung von I in endlich viele Teilintervalle I_j, $j = 1, ..., n$, existiert, so dass g auf I_j konstant ist für alle j.

(ii) Eine Funktion $g : I \to \mathbb{R}^m$ heißt (Lebesgue-) messbar, falls eine Folge von stückweise konstanten Funktionen $g_i : I \to \mathbb{R}^m, i \in \mathbb{N}$, existiert mit
$\lim_{i\to\infty} g_i(x) = g(x)$ für fast alle $x \in I$.
(Fast alle $x \in I$ heißt in diesem Fall alle $x \in J \subseteq I$ mit $I \setminus J$ ist Lebesgue-Nullmenge.)

(iii) Eine Funktion $g : \mathbb{R} \to \mathbb{R}^m$ heißt (Lebesgue-) messbar, falls für jedes abgeschlossene Teilintervall $I = [a, b] \subset \mathbb{R}$ die Einschränkung $g\,|_I$ messbar im Sinne von (ii) ist.

Dass die Wahl $\mathcal{U}$ gleich der Menge der messbaren Kontrollfunktionen im Hinblick auf Aussagen über die Lösbarkeit sinnvoll ist, zeigt folgender Satz:

Satz 3.4 (Satz von Caratheodory) *Betrachte ein Kontrollsystem mit folgenden Eigenschaften:*
(i) Der Raum der Kontrollfunktionen ist gegeben durch

$$\begin{aligned}\mathcal{U} \;:=\; &\{u : \mathbb{R} \to U \mid \; u \text{ ist messbar und}\\ &\text{außerhalb einer Lebesgue-Nullmenge beschränkt.}\}\end{aligned}$$

(ii) Das Vektorfeld $f : \mathbb{R}^d \times U \to \mathbb{R}^d$ ist stetig.

(iii) Für jedes $R > 0$ existiert eine Konstante $L_R > 0$, so dass die Abschätzung

$$\|f(x_1, u) - f(x_2, u)\| \le L_R \|x_1 - x_2\|$$

für alle $x_1, x_2 \in \mathbb{R}^d$ und alle $u \in U$ mit $\|x_1\|, \|x_2\|, \|u\| \le R$ erfüllt ist.

Dann gibt es für jeden Punkt $x_0 \in R^d$ und jede Kontrollfunktion $u \in \mathcal{U}$ ein (maximales) offenes Intervall I mit $0 \in I$ und genau eine absolut stetige Funktion $x(t)$, die die Integralgleichung

$$x(t) = x_0 + \int_0^t f((x(\tau), u(\tau))d\tau$$

für alle $x \in I$ erfüllt.

Der Beweis ist z.B. in Sontag, *Mathematical Control Theory*, Springer Verlag, New York, 1998, Anhang C, nachzulesen.

Definition 3.5 *Man nennt die eindeutige Funktion $x(t)$ aus Satz 3.4 die Lösung von (3.1) zum Anfangswert $x_0 \in \mathbb{R}^d$ und zur Kontrollfunktion $u \in \mathcal{U}$. Im Folgenden sei sie mit $\Phi(t, x_0, u)$ bezeichnet.*

Nun wird ein sogenanntes Einschrittverfahren definiert, durch das ein kontinuierliches Kontrollsystem durch ein diskretes System approximativ dargestellt werden kann. Einschrittverfahren sind aus der Theorie der gewöhnlichen Differentialgleichungen bekannt und dienen der numerischen Lösung.
Die kontinuierliche Lösungsfunktion $\Phi(t, x_0, u)$ wird durch eine diskrete Funktion auf einem Gitter Γ mit der Schrittweite $h > 0$ approximiert. Diese Gitterfunktion hat die Form (3.2). Die Iterationsvorschrift f_h kann dann mit einem Computer ausgewertet werden.

Das folgende einfache Einschrittverfahren für Kontrollsysteme wird im Algorithmus zur Lösung optimaler Steuerungsprobleme verwendet.

Definition 3.6 (Euler-Verfahren für Kontrollsysteme) *Für einen Zeitschritt $h > 0$ und einen Kontrollwert $u \in U$ definiert man das Euler-Verfahren als das durch die Abbildung*

$$f_h(x, u) := x + hf(x, u) \tag{3.4}$$

beschriebene zeitdiskrete Kontrollsystem (3.2). Die Lösung sei mit $\tilde{\Phi}_h(t, x_0, u_h)$ bezeichnet.

Über die Konvergenzeigenschaften dieses Verfahrens lässt sich folgende Aussage machen. Für den langwierigen Beweis sei auf Gonzalez, Tidball, *On a discrete time approximation of the Hamilton-Jacobi equation of dynamic programming*, INRIA Rapports de Recherche Nr. 1375, 1991, verwiesen. Für die enifachere Klasse der konvexen Kontrollsysteme ist der Beweis einfacher. Siehe dazu Grüne [11], Kapitel 1.

Satz 3.7 *Sei ein Kontrollsystem mit den Eigenschaften (i)-(iii) aus Satz 3.4 gegeben und sei $\overline{B}_R(0)$ die abgeschlossene Kugel mit Radius R um 0 im $\mathbb{R}^d$.*
Dann gilt für das Verfahren aus Definition 3.6 und jede Konstante $R > 0$:

(i) Es existiert eine Konstante $K > 0$, so dass für jede Kontrollfunktion $u \in \mathcal{U}$ mit $\|u\|_\infty \leq R$ und jeden Anfangswert $x_0 \in \overline{B}_R(0)$ eine diskrete Kontrollfunktion $u_h \in \mathcal{U}_h$ existiert, mit der die Abschätzung

$$\|\tilde{\Phi}_h(t, x_0, u_h) - \Phi(t, x_0, u)\| \leq K\sqrt{h}e^{L_R t}$$

gilt für all die $t \in h\mathbb{N}_0$, deren zugehörige Lösungen in $\overline{B}_R(0)$ liegen.

Umgekehrt gilt auch:
(ii) Es gibt eine Konstante $K > 0$, so dass für jedes $x_0 \in \overline{B}_R(0)$, jede diskrete Kontrollfunktion $u_h \in \mathcal{U}_h$ mit $\|u\|_\infty \leq R$ und die durch

$$u(\tau) := u_h(t), \quad \tau \in [t, t+h[, \quad t \in h\mathbb{N}_0$$

definierte stückweise konstante (und damit messbare) Kontrollfunktion die Abschätzung

$$\|\tilde{\Phi}_h(t, x_0, u_h) - \Phi(t, x_0, u)\| \leq Kh(e^{L_R t} - 1)$$

gilt für all die $t \in h\mathbb{N}_0$, deren Lösungen $\tilde{\Phi}_h(t, x_0, u_h)$ in $\overline{B}_R(0)$ liegen.

Dabei ist K jeweils unabhängig von R. L_R ist die Lipschitz-Konstante für f aus Satz 3.4.

Bemerkung 3.8 *Zur Diskretisierung kann die Iterationsvorschrift (3.4) auch durch andere Einschrittverfahren ersetzt werden. Es zeigt sich jedoch, dass der Aufwand für die Konstruktion von Verfahren mit höherer Konvergenzordnung sehr hoch ist.*

3.2 Diskontierte Optimale Steuerung

Im Folgenden wird eine typische Form von optimalen Steuerungsproblemen vorgestellt.

3.2.1 Modellproblem

Man betrachte eine Funktion

$$g : \mathbb{R}^d \times U \to \mathbb{R}.$$

Ziel der optimalen Steuerung ist es, das Integral (im Kontinuierlichen) bzw. die Summe (im Diskreten) dieser Funktion entlang einer Trajektorie $\Phi(t, x_0, u)$ bzw. $\Phi_h(t, x_0, u)$ und der dazugehörigen Kontrollfunktion $u \in \mathcal{U}$ bzw. $u_h \in \mathcal{U}_h$ zu optimieren, d.h. zu maximieren oder zu minimieren. Integral und Summe hängen beide von x_0 und von u ab. Man sucht also zu einem Anfangswert x_0 diejenige Kontrollfunktion $u \in \mathcal{U}$ bzw. $u_h \in \mathcal{U}_h$, die das Integral bzw. die Summe optimiert.

Definition 3.9 (Optimales Steuerungsproblem) *Gegeben sei ein Kontrollsystem (3.1) bzw. (3.2). Für eine Funktion $g : \mathbb{R}^d \times U \to \mathbb{R}$ und einen Parameter $\delta > 0$ sei das diskontierte Funktional $J(x, u)$ auf unendlichem Zeithorizont in kontinuierlicher Zeit definiert durch*

$$J(x, u) := \int_0^\infty e^{-\delta t} g(\Phi(t, x, u), u(t)) dt \tag{3.5}$$

sowie in diskreter Zeit durch

$$J_h(x, u_h) := h \sum_{j=0}^{\infty} (1 - \delta h)^j g(\Phi_h(jh, x, u_h), u_h(jh)). \tag{3.6}$$

Das optimale Steuerungsproblem dazu lautet: Bestimme

$$v(x) := \max_{u \in \mathcal{U}} J(x, u) \quad bzw. \quad v_h(x) := \max_{u_h \in \mathcal{U}_h} J_h(x, u_h). \tag{3.7}$$

$v(x)$ bzw. $v_h(x)$ heißt dann optimale Wertefunktion.

Dabei gelte
(i) U ist kompakt.
(ii) g ist stetig und erfüllt

$$|g(x, u)| \leq M_g \quad und \quad |g(x_1, u) - g(x_2, u)| \leq L_g \|x_1 - x_2\| \tag{3.8}$$

für alle $x, x_1, x_2 \in \mathbb{R}^d$, alle $u \in \mathcal{U}$ und geeignete Konstanten $M_g, L_g > 0$.

Für das Kontrollsystem gelte außerdem:
In kontinuierlicher Zeit sind die Voraussetzungen aus Satz 3.4 erfüllt und in diskreter Zeit existiert eine Konstante $L > 0$, so dass gilt:

$$\|f_h(x_1, u) - f_h(x_2, u)\| \leq (1 + Lh)\|x_1 - x_2\|$$

für alle $x_1, x_2 \in \mathbb{R}^d$ und alle $u \in U$.

Bemerkung 3.10 *(i) Bei Minimierung lautet das optimale Steuerungsproblem analog*

$$v(x) := \min_{u \in \mathcal{U}} J(x, u) \quad bzw. \quad v_h(x) := \min_{u_h \in \mathcal{U}_h} J_h(x, u_h).$$

(ii) Es sei hier angenommen, dass eine optimale Kontrollfunktion in $\mathcal{U}$ bzw. in $\mathcal{U}_h$ existiert. Ansonsten schreibe „sup“ statt „max“ bzw. „inf“ statt „min“.

(iii) Der sogenannte Diskontfaktor $e^{-\delta t}$ bzw. $1-\delta h$ mit Diskontrate $\delta > 0$ stammt ursprünglich aus der Ökonomie. Er sorgt dafür, dass der Ertrag in naher Zukunft stärker gewichtet wird als derjenige in ferner Zukunft (beide konvergieren gegen 0 für $t \to \infty$).

(iv) Im Folgenden gelte stets die Annahme $\delta h < 1$. Im Kontinuierlichen ist dies immer erfüllt, da hier $h = 0$ ist.

Wichtig für den Algorithmus zur Lösung optimaler Steuerungsprobleme ist folgende Eigenschaft des diskontierten Funktionals.

Lemma 3.11 *Das diskontierte Funktional ist endlich. Es gilt*

$$|J(x,u)| \leq \frac{M_g}{\delta} \quad und \quad |J_h(x,u_h)| \leq \frac{M_g}{\delta}$$

mit M_g aus (3.8). Damit gilt offensichtlich auch $|v(x)|, |v_h(x)| \leq \frac{M_g}{\delta}$.

Beweis: Zunächst der Beweis in kontinuierlicher Zeit:

$$\begin{aligned}
|J(x,u)| &= \left| \int_0^\infty e^{-\delta t} g(\Phi(t,x,u),u(t))dt \right| \\
&\leq \int_0^\infty |e^{-\delta t} g(\Phi(t,x,u),u(t))|dt \\
&= \int_0^\infty e^{-\delta t} |g(\Phi(t,x,u),u(t))|dt \\
&\leq \int_0^\infty e^{-\delta t} M_g dt \\
&\leq M_g \int_0^\infty e^{-\delta t} dt \\
&= M_g \left[-\frac{1}{\delta} e^{-\delta t} \right]_0^\infty = \frac{M_g}{\delta}
\end{aligned}$$

In diskreter Zeit gilt

$$\begin{aligned}
|J_h(x,u_h)| &= \left| h \sum_{j=0}^\infty (1-\delta h)^j g(\Phi_h(jh,x,u_h),u_h(jh)) \right| \\
&\leq h \sum_{j=0}^\infty |(1-\delta h)^j g(\Phi_h(jh,x,u_h),u_h(jh))| \\
&= h \sum_{j=0}^\infty (1-\delta h)^j |g(\Phi_h(jh,x,u_h),u_h(jh))| \\
&\leq h \sum_{j=0}^\infty (1-\delta h)^j M_g \\
&= h M_g \sum_{j=0}^\infty (1-\delta h)^j \\
&= h M_g \frac{1}{\delta h} = \frac{M_g}{\delta}.
\end{aligned}$$

Da nach Voraussetzung $\delta h < 1$ gilt, konvergiert die geometrische Reihe $\sum_{j=0}^\infty (1-\delta h)^j$ gegen $\frac{1}{1-(1-\delta h)} = \frac{1}{\delta h}$, woraus die vorletzte Gleichung folgt.

□

3.2.2 Eigenschaften der optimalen Wertefunktion

Nun werden zwei Charakteristika der optimalen Wertefunktion $v(x)$ vorgestellt, die grundlegend für die numerische Approximation sind. Zum einen die Stetigkeitseigenschaft, zum anderen das sogenannte Bellmannsche Optimalitätsprinzip oder auch Prinzip der Dynamischen Programmierung, auf das der Algorithmus aufbauen wird.

Im Allgemeinen ist nicht davon auszugehen, dass die optimale Wertefunktion $v(x)$ differenzierbar ist. Es kann aber Hölder-Stetigkeit und unter gewissen Voraussetzungen auch Lipschitz-Stetigkeit angenommen werden.

Satz 3.12 *Es sei ein optimales Steuerungsproblem der Form (3.7) gegeben. Das Vektorfeld $f : \mathbb{R}^d \times U \to \mathbb{R}^d$ sei global Lipschitz-stetig, d.h. die Lipschitz-Konstante L_R für f sei unabhängig von R, also $L_R = L$.*

Dann gilt:
Ist die Diskontrate $\delta > L$, so ist $v(x)$ Lipschitz-stetig mit Konstante $\frac{L_g}{\delta - L}$.
Ist dagegen $\delta \leq L$, so ist v(x) „nur“ Hölder-stetig, d.h. es existieren Konstanten K, $\gamma > 0$, so dass für alle $x, y \in \mathbb{R}^d$ die Abschätzung

$$|v(x) - v(y)| \leq K\|x - y\|^\gamma$$

erfüllt ist. Für $\delta = L$ ist $\gamma \in]0, 1[$ beliebig, für $\delta < L$ ist $\gamma = \frac{\delta}{L}$.

Diese Aussage gilt auch im diskreten Fall für v_h.

Für den Beweis schätzt man die Integrale bzw. die Summen durch Konstanten ab, die von den Lipschitzkonstanten L von f und M_g von g abhängen. Dies wird in Grüne [11], Kapitel 2.3, ausführlich gezeigt.

Die zweite Eigenschaft, das Bellmannsche Optimalitätsprinzip, bildet wie bereits erwähnt die Grundlage für die numerische Lösung eines optimalen Steuerungsproblems bzw. die Approximation der optimalen Wertefunktion.
Es besagt, dass man den optimalen Wert $v(x)$ in einem Punkt erhält, wenn man für eine (beliebig kurze oder lange) endliche Zeit optimal steuert und dabei den Wert von v in dem erreichten Punkt berücksichtigt. Anders ausgedrückt: Die Endstücke optimaler Trajektorien sind wieder optimale Trajektorien.

Satz 3.13 (Bellmansches Optimalitätsprinzip) *Betrachte ein optimales Steuerungsproblem der Form (3.7). Dann erfüllt die optimale Wertefunktion $v(x)$ für jedes $x \in \mathbb{R}^d$ und jedes $T > 0$ die Gleichung*

$$v(x) = \sup_{u \in \mathcal{U}} \left\{ \int_0^T e^{-\delta t} g(\Phi(t, x, u), u(t)) dt + e^{-\delta T} v(\Phi(T, x, u)) \right\}, \tag{3.9}$$

bzw. im Diskreten für jedes $k \in \mathbb{N}$ die Gleichung

$$\begin{aligned} v_h(x) &= \sup_{u_h \in \mathcal{U}_h} \Big\{ h \sum_{j=0}^{k} (1-\delta h)^j g(\Phi_h(jh, x, u_h), u_h(jh)) \\ &\quad + (1-\delta h)^{k+1} v_h(\Phi_h((k+1)h, x, u_h)) \Big\}. \end{aligned} \tag{3.10}$$

Beweis: Die Beweise für die beiden Fälle laufen vollständig analog. Deshalb wird hier nur derjenige für den kontinuierlichen Fall dargestellt. Statt der Integrale sind im Diskreten entsprechend die Summen zu betrachten.

„ $\leq$ “: Seien $x \in \mathbb{R}^d$, $T > 0$ und $u \in \mathbb{U}$ beliebig. Dann gilt

$$\begin{aligned} J(x,u) &= \int_0^\infty e^{-\delta t} g(\Phi(t,x,u), u(t)) dt \\ &= \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t)) dt + \int_T^\infty e^{-\delta t} g(\Phi(t,x,u), u(t)) dt \\ &= \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t)) dt \\ &\quad + \int_0^\infty e^{-\delta(t+T)} g(\Phi(t, \Phi(T,x,u), u), u(t+T)) dt \\ &= \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t)) dt \\ &\quad + e^{-\delta T} \int_0^\infty e^{-\delta t} g(\Phi(t, \Phi(T,x,u), u), u(T+.)) dt \\ &= \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t)) dt + e^{-\delta T} J(\Phi(T,x,u), u(T+.)) \\ &\leq \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t)) dt + e^{-\delta T} v(\Phi(T,x,u)). \end{aligned}$$

Dies ist nach Annahme für jedes beliebige $u \in \mathbb{U}$ erfüllt, also auch für

$$v(x) = \max_{u \in \mathcal{U}} J(x,u).$$

„ $\geq$ “: Seien wiederum $x \in \mathbb{R}^d$ und $T > 0$. Sei $\varepsilon > 0$ beliebig. Man wähle $\tilde{u} \in \mathcal{U}$ so, dass gilt:

$$\begin{aligned} &\sup_{u \in \mathcal{U}} \left\{ \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t)) dt + e^{-\delta T} v(\Phi(T,x,u)) \right\} \\ &\quad \leq \int_0^T e^{-\delta t} g(\Phi(t,x,u), \tilde{u}(t)) dt + e^{-\delta T} v(\Phi(T,x,\tilde{u})) + \varepsilon \end{aligned} \tag{3.11}$$

Nachdem $\tilde{u}$ somit auf $[0,T]$ festgelegt ist, sei $\tilde{u}\,|_{]T,\infty[}$ so, dass

$$J(\Phi(T,x,\tilde{u}),\tilde{u}(T+.)) \geq v(\Phi(T,x,\tilde{u})) - \varepsilon \tag{3.12}$$

erfüllt ist. (3.12) in (3.11) eingesetzt ergibt

$$\begin{aligned}
&\sup_{u\in\mathcal{U}}\left\{\int_0^T e^{-\delta t} g(\Phi(t,x,u),u(t))dt + e^{-\delta T} v(\Phi(T,x,u))\right\} \qquad (3.13)\\
&\quad\leq \int_0^T e^{-\delta t} g(\Phi(t,x,\tilde{u}),\tilde{u}(t))dt + e^{-\delta T}(J(\Phi(T,x,\tilde{u}),\tilde{u}(T+.)) + \varepsilon) + \varepsilon\\
&\quad= \int_0^T e^{-\delta t} g(\Phi(t,x,\tilde{u}),\tilde{u}(t))dt + e^{-\delta T} J(\Phi(T,x,\tilde{u}),\tilde{u}(T+.)) + (1+e^{-\delta T})\varepsilon.
\end{aligned}$$

Nun geht man wie im Beweisteil „ $\leq$ “ vor und verschiebt bei

$$J(\Phi(T,x,\tilde{u}),\tilde{u}(T+.)) = \int_0^\infty e^{-\delta t} g(\Phi(t,\Phi(T,x,\tilde{u}),u),\tilde{u}(T+.))dt$$

die untere Integralgrenze; diesmal umgekehrt von 0 nach T. Dadurch ergibt sich wieder

$$J(\Phi(T,x,\tilde{u}),\tilde{u}(T+.)) = \frac{1}{e^{-\delta T}}\int_T^\infty e^{-\delta t} g(\Phi(t,x,\tilde{u}),\tilde{u}(t))dt.$$

Dies eingesetzt in (3.13) liefert

$$\begin{aligned}
&\sup_{u\in\mathcal{U}}\left\{\int_0^T e^{-\delta t} g(\Phi(t,x,u),u(t))dt + e^{-\delta T} v(\Phi(T,x,u))\right\}\\
&\quad= \int_0^T e^{-\delta t} g(\Phi(t,x,\tilde{u}),\tilde{u}(t))dt + e^{-\delta T} J(\Phi(T,x,\tilde{u}),\tilde{u}(T+.)) + (1+e^{-\delta T})\varepsilon\\
&\quad= \int_0^T e^{-\delta t} g(\Phi(t,x,\tilde{u}),\tilde{u}(t))dt + \int_T^\infty e^{-\delta t} g(\Phi(t,x,\tilde{u}),\tilde{u}(t))dt + (1+e^{-\delta T})\varepsilon\\
&\quad= J(x,\tilde{u}) + (1+e^{-\delta T})\varepsilon \quad \leq v(x) + (1+e^{-\delta T})\varepsilon.
\end{aligned}$$

$\varepsilon > 0$ ist beliebig wählbar, somit ist die Behauptung bewiesen.

□

Es lässt sich außerdem zeigen, dass die optimale Wertefunktion v(x) durch das Optimalitätsprinzip eindeutig bestimmt ist. Dafür ist zunächst folgendes Lemma notwendig:

Lemma 3.14 *Betrachte zwei Abbildungen $f_1, f_2 : A \to \mathbb{R}$ aus einer beliebigen Menge A nach $\mathbb{R}$. Dann gilt:*

$$\left|\sup_{a_1\in A} f_1(a_1) - \sup_{a_2\in A} f_2(a_2)\right| \leq \sup_{a\in A} |f_1(a) - f_2(a)|$$

Beweis: Ohne Beschränkung der Allgemeinheit gelte

$$\left| \sup_{a_1 \in A} f_1(a_1) - \sup_{a_2 \in A} f_2(a_2) \right| = \sup_{a_1 \in A} f_1(a_1) - \sup_{a_2 \in A} f_2(a_2)$$

Sei nun $\varepsilon > 0$ beliebig. Wähle ein $a_\varepsilon \in A$ so, dass gilt:

$$f_1(a_\varepsilon) \geq \sup_{a_1 \in A} f_1(a_1) - \varepsilon$$

Wegen $a_\varepsilon \in A$ gilt offensichtlich

$$\sup_{a_2 \in A} f_2(a_2) \geq f_2(a_\varepsilon)$$

und folglich

$$\begin{aligned} \sup_{a_1 \in A} f_1(a_1) - \sup_{a_2 \in A} f_2(a_2) &\leq f_1(a_\varepsilon) - f_2(a_\varepsilon) + \varepsilon \\ &\leq |f_1(a_\varepsilon) - f_2(a_\varepsilon)| + \varepsilon \\ &\leq \sup_{b \in B} |f_1(a_\varepsilon) - f_2(a_\varepsilon)| + \varepsilon. \end{aligned}$$

Da $\varepsilon > 0$ beliebig wählbar war, folgt die Behauptung.

□

Nun lässt sich folgende Aussage beweisen:

Satz 3.15 *Sei das optimale Steuerungsproblem (3.7) mit optimaler Wertefunktion v sowie ein $T > 0$ bzw. ein $k \geq 0$ gegeben.*
Sei $w : \mathbb{R}^d \to \mathbb{R}$ eine beschränkte Funktion, die das Optimalitätsprinzip (3.9) bzw. (3.10) für alle $x \in \mathbb{R}^d$ erfüllt. Dann gilt:

$$w = v \quad bzw. \quad w = v_h$$

Beweis: Zur besseren Übersichtlichkeit sei hier nur der kontinuierliche Fall aufgeführt. Im Diskreten werden statt der Integrale entsprechend die Summen betrachtet. Die Vorgehensweise bleibt die gleiche. Sowohl für w als auch für v gilt:

$$\left\{ \begin{array}{c} w \\ v \end{array} \right\}(x) = \sup_{u \in \mathcal{U}} \left\{ \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t)) dt + e^{-\delta T} \left\{ \begin{array}{c} w \\ v \end{array} \right\}(\Phi(T,x,u)) \right\}.$$

Sei nun

$$a_1(u) = \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t)) dt + e^{-\delta T} w(\Phi(T,x,u))$$

und analog

$$a_2(u) = \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t))dt + e^{-\delta T} v(\Phi(T,x,u)).$$

Dann erhält man mit Lemma 3.14

$$\begin{aligned}
|w(x) - v(x)| &= \left|\sup_{u\in\mathcal{U}} a_1(u) - \sup_{u\in\mathcal{U}} a_2(u)\right| \leq \sup_{u\in\mathcal{U}} |a_1(u) - a_2(u)| \\
&= \sup_{u\in\mathcal{U}} \left| \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t))dt + e^{-\delta T} w(\Phi(T,x,u)) \right. \\
&\qquad \left. - \int_0^T e^{-\delta t} g(\Phi(t,x,u), u(t))dt + e^{-\delta T} v(\Phi(T,x,u)) \right| \\
&= \sup_{u\in\mathcal{U}} \{ e^{-\delta T} |w(\Phi(T,x,u)) - v(\Phi(T,x,u))| \} \\
&\leq e^{-\delta T} \sup_{y\in\mathbb{R}^d} |w(y) - v(y)|.
\end{aligned}$$

Dies gilt für alle $x \in \mathbb{R}^d$, also auch für

$$\sup_{y\in\mathbb{R}^d} |w(y) - v(y)| \leq e^{-\delta T} \sup_{y\in\mathbb{R}^d} |w(y) - v(y)|.$$

Umgeformt ergibt sich demnach

$$(1 - e^{-\delta T}) \sup_{y\in\mathbb{R}^d} |w(y) - v(y)| \leq 0.$$

Da $e^{-\delta T} < 1$ ist, muss das Supremum 0 sein. Also ist $w = v$.

□

Kapitel 4

Numerische Lösung optimaler Steuerungsprobleme

In diesem Kapitel folgt die Darstellung des numerischen Verfahrens aus Grüne [11], mit dessen Hilfe optimale Steuerungsprobleme gelöst werden sollen. Im Gegensatz zu den dort verwendeten linearen Interpolationsmethoden soll allerdings hier Spline-Interpolation angewendet werden. Die theoretische Betrachtung richtet sich jedoch weitgehend nach der Vorgehensweise von Lars Grüne.
Das Verfahren besteht im Wesentlichen aus zwei Schritten. Der erste Schritt ist die Diskretisierung in der Zeit. Er ist nur notwendig, wenn ein kontinuierliches Problem vorliegt. Dieses kontinuierliche Problem wird hierbei durch ein zeitdiskretes approximiert. Durch eine Iterationsformel soll dann die optimale Wertefunktion des zeitdiskreten bzw. zeitdiskretisierten optimalen Steuerungsproblems bestimmt werden.
Dazu ist der zweite Schritt notwendig, die Diskretisierung im Ort. Grund dafür ist, dass in jeder Iteration unendlich viele Punkte berechnet werden müssten, um die jeweilige nächste Funktion zu erhalten, was natürlich nicht möglich ist. Um die Diskretisierung im Ort zu ermöglichen, muss vorher der Definitionsbereich $\mathbb{R}^d$ eingeschränkt werden. Auf die Auswirkungen dieser Einschränkung wird ebenfalls kurz eingegangen.

4.1 Diskretisierung in der Zeit

Nach Definition 3.6 und Satz 3.7 lässt sich zu einem kontinuierlichen Kontrollsystem mittels des Euler-Verfahrens (3.4) eine zeitdiskrete Approximation finden, die zu jedem Anfangswert $x_0 \in \mathbb{R}^d$ und jeder diskreten Kontrollfunktion $u_h : h\mathbb{Z} \to U$ eine zeitdiskrete approximierte Lösung $\tilde{\Phi}_h(t, x_o, u_h)$ liefert.
Sei für ein gegebenes kontinuierliches optimales Steuerungsproblem das zu dessen Euler-Approximation gehörige diskrete optimale Steuerungsproblem gegeben

durch

$$\tilde{v}_h(x) := \max_{u_h \in \mathcal{U}_h} \tilde{J}_h(x, u_h) \tag{4.1}$$

mit

$$\tilde{J}_h(x,u) := h \sum_{j=0}^{\infty} (1-\delta h)^j g(\tilde{\Phi}_h(jh, x, u_h), u_h(jh)).$$

Wie folgender Satz zeigt, konvergiert die Approximation $\tilde{v}_h(x)$ gegen $v(x)$, wenn der Zeitschritt h gegen 0 geht:

Satz 4.1 *Sei ein optimales Steuerungsproblem der Form (3.1) sowie das zugehörige durch das oben beschriebene Euler-Verfahren diskretisierte Steuerungsproblem (4.1) gegeben. Das zu Grunde liegende Kontrollsystem erfülle die Voraussetzungen von Satz 3.4.*
Dann gelten für die optimalen Wertefunktionen $v(x)$ und $\tilde{v}_h(x)$ mit passender Konstante $K > 0$

(i) $v(x) \leq \tilde{v}_h(x) + K(h^{\frac{\gamma}{2}} + h)$ und

(ii) $\tilde{v}_h(x) \leq v(x) + K(h^{\gamma} + h)$

für alle $h \in [0, \frac{1}{\delta}[$ und alle $x \in \mathbb{R}^d$. $\gamma \in]0,1]$ sei die Hölder-Konstante aus Satz 3.12.

Man kann also für alle $x \in \mathbb{R}^d$ schreiben:

$$|v(x) - \tilde{v}_h(x)| \leq \tilde{K} h^{\frac{\gamma}{2}} \tag{4.2}$$

mit passendem $\tilde{K} > 0$.

Einen Beweis kann man in Bardi und Capuzzo-Dolcetta [3], Kapitel VI, nachlesen. Für konvexe optimale Steuerungsprobleme wird die Behauptung wiederum in Grüne [11] bewiesen.

Bemerkung 4.2 *$\tilde{v}_h$ erfüllt nach Satz 3.13 für alle $k \in \mathbb{N}$ das Optimalitätsprinzip*

$$\begin{aligned} \tilde{v}_h(x) \quad = \quad \sup_{u_h \in \mathcal{U}_h} \Big\{ & h \sum_{j=0}^{k} (1-\delta h)^j g(\tilde{\Phi}_h(jh, x, u_h), u_h(jh)) \\ & + (1-\delta h)^{k+1} \tilde{v}_h(\tilde{\Phi}_h((k+1)h, x, u_h)) \Big\}. \end{aligned} \tag{4.3}$$

Man kann hier anstelle des Supremums auch das Maximum bilden, da sowohl $\tilde{\Phi}$ und g in $(u_h(0), ..., u_h(k))$ als auch $\tilde{v}_h$ stetig sind.

Aufgrund der Stetigkeit ist auch sichergestellt, dass es für $k = 0$ zu jedem $x \in \mathbb{R}^d$ mindestens ein $u_x^* \in U$ gibt, so dass das Supremum in (4.3) für ein $u_h \in \mathcal{U}_h$ mit $u_h(0) = u_x^*$ angenommen wird.

4.2 Ein Iterationsverfahren

Ausgehend vom Optimalitätsprinzip (4.3) kann man für zeitdiskrete, und entsprechend auch für zeitdiskretisierte, optimale Steuerungsprobleme ein Iterationsverfahren definieren, dessen errechnete Funktionenfolge gegen die optimale Wertefunktion v_h konvergiert.

Definition 4.3 *Durch*

$$v_h^0(x) = 0 \quad \textit{und} \quad v_h^{j+1}(x) = T_h(v_h^j)(x) \textit{ für alle } x \in \mathbb{R}^d \tag{4.4}$$

werden iterativ Funktionen $v_h^j : \mathbb{R}^d \to \mathbb{R}, j = 0, 1, ...$ definiert.

Dabei hat $T_h : C(\mathbb{R}^d, \mathbb{R}) \to C(\mathbb{R}^d, \mathbb{R})$ die Form

$$T_h(w)(x) := \max_{u \in \mathcal{U}} \{hg(x,u) + \beta w(f_h(x,u))\} \tag{4.5}$$

mit $\beta = 1 - \delta h$ und $C(\mathbb{R}^d, \mathbb{R})$ der Menge der stetigen Funktionen von $\mathbb{R}^d$ nach $\mathbb{R}$.

Nun wird die gewünschte Konvergenzeigenschaft der v_h^j untersucht:

Satz 4.4 *Sei v_h die Wertefunktion des zeitdiskretisierten optimalen Steuerungsproblems (4.1). Dann gilt für die Funktionen v_h^j aus (4.4)*

$$\|v_h^j - v_h\|_\infty \leq \beta^j \frac{M_g}{\delta}.$$

Nach Bemerkung 3.10(iv) gilt $\beta = 1 - \delta h < 1$, also folgt insbesondere, dass v_h^j gleichmäßig gegen v_h konvergiert.

Beweis: Seien $w_1, w_2 : \mathbb{R}^d \to \mathbb{R}$ zwei beliebige Funktionen. Dann gilt

$$\begin{aligned}
&|T_h(w_1)(x) - T_h(w_2)(x)| = \\
&= \left| \max_{u \in \mathcal{U}} \{hg(x,u) + \beta w_1(f_h(x,u))\} - \max_{u \in \mathcal{U}} \{hg(x,u) + \beta w_2(f_h(x,u))\} \right| \\
&= \left| \sup_{u \in \mathcal{U}} \{hg(x,u) + \beta w_1(f_h(x,u))\} - \sup_{u \in \mathcal{U}} \{hg(x,u) + \beta w_2(f_h(x,u))\} \right| \\
&\leq \sup_{u \in \mathcal{U}} |\beta w_1(f_h(x,u)) - \beta w_2(f_h(x,u))| \\
&= \beta \sup_{u \in \mathcal{U}} |w_1(f_h(x,u)) - w_2(f_h(x,u))| \\
&\leq \beta \|w_1 - w_2\|_\infty.
\end{aligned}$$

Dabei folgt die erste Ungleichung aus Lemma 3.14. Daraus resultiert

$$\|T_h(w_1) - T_h(w_2)\|_\infty \leq \beta \|w_1 - w_2\|_\infty. \tag{4.6}$$

Nun betrachtet man das Optimalitätsprinzip (4.3) für $k = 0$. Man erhält

$$v_h(x) = \max_{u_h \in \mathcal{U}_h} \{hg(\Phi_h(0, x, u_h), u_h) + \beta v_h(\Phi_h(h, x, u_h))\}.$$

Nach der Definition von Φ_h als Lösung des Kontrollproblems (siehe (3.3)) folgt die Gleichung

$$v_h(x) = \max_{u_h \in \mathcal{U}_h} \{hg(x, u_h) + \beta v_h(f_h(x, u_h))\} = T_h(v_h)(x). \tag{4.7}$$

Mit der Defintion von v_h^{j+1} aus (4.4) und (4.6) ergibt sich

$$\|v_h - v_h^{j+1}\|_\infty = \|T_h(v_h) - T_h(v_h^j)\|_\infty \leq \beta \|v_h - v_h^j\|_\infty. \tag{4.8}$$

Dabei wird jeweils (4.7) ausgenutzt.

Für $j = 0$ kann man nach Lemma 3.11 zudem schreiben:

$$\|v_h - v_h^0\|_\infty = \|v_h\|_\infty \leq \frac{M_g}{\delta} = \beta^0 \frac{M_g}{\delta} \tag{4.9}$$

Mit (4.8) und (4.9) ist somit die Behauptung durch Induktion über j bewiesen.

□

Bemerkung 4.5 *Außerdem lässt sich zeigen, dass die erhaltenen Funktionen v_h^j für alle j Lipschitz-stetig sind.*

4.2.1 Zustandsraumbeschränkung

Wie bereits angedeutet, ist es numerisch nicht möglich, v_h im ganzen $\mathbb{R}^d$ zu berechnen. Aus diesem Grund muss man den Definitionsbereich von v_h bzw. der zeitlichen Diskretisierung $\tilde{v}_h$ auf eine kompakte Menge $\Omega \subset \mathbb{R}^d$ einschränken. Dann werden nur die Lösungen betrachtet, die für alle positiven Zeiten in Ω bleiben. Das Erstellen einer geeigneten Menge Ω soll hier nicht weiter ausgeführt werden. In der Praxis ergibt sich ein solches Ω oft aus der Aufgabenstellung, z.B. aus dem physikalisch interessanten Bereich.
Entscheidend ist, welche Konsequenzen die Einschränkung auf eine kompakte Teilmenge des $\mathbb{R}^d$ für v_h hat. Es existieren drei Möglichkeiten:

(i) Ω ist stark invariant, d.h. für alle $x \in \Omega$ und alle $u \in U$ ist $f_h(x, u) \in \Omega$.
Hier ergibt sich kein Problem. Durch die Einschränkung ändert sich v_h auf Ω nicht.

(ii) Ω ist schwach invariant, d.h. für alle $x \in \Omega$ gibt es mindestens ein $u \in U$ mit $f_h(x, u) \in \Omega$.

In diesem Fall optimiert man nur über diese $u \in U$, also über eine kleinere Menge. Folglich wird die Funktion v_h kleiner.

Oft existieren jedoch optimal invariante Teilmengen $\hat{\Omega} \subseteq \Omega$, d.h. für alle $x \in \hat{\Omega}$ und für das jeweils optimale, also maximierende u_x^* gilt $f_h(x, u_x^*) \in \hat{\Omega}$. In Worten heißt das, dass die optimalen Trajektorien die Menge $\hat{\Omega}$ nicht verlassen. Somit ändert sich v_h auf $\hat{\Omega}$ nicht.

(iii) Der dritte Fall erweist sich als ungünstiger: Ω ist nicht invariant, d.h. es gibt ein $x \in \Omega$, so dass für alle $u \in U$ gilt: $f_h(x, u) \notin \Omega$.
In diesem Fall bietet sich an, $f_h(x, u)$ durch den nächstgelegenen Punkt in Ω zu ersetzen.
Hier kann aber die erhaltene numerische Lösung sehr stark von der tatsächlichen optimalen Wertefunktion abweichen.

4.3 Diskretisierung im Ort

Durch den Operator T_h aus (4.5) ist ein Iterationsverfahren definiert, das, angewendet auf die Funktionen v_h^j, die optimale Wertefunktion v_h eines diskreten oder diskretisierten optimalen Steuerungsproblems erzeugt. Bei der Implementierung müsste der Operator, also die jeweiligen Funktionen v_h^j, an unendlich vielen Punkten ausgewertet werden. Dieses Problem bleibt selbstverständlich auch bestehen, wenn man sich, wie oben beschrieben, auf eine kompakte Menge $\Omega \in \mathbb{R}^d$ beschränkt.

Der folgende Abschnitt beschränkt sich auf die Betrachtung des eindimensionalen Falls $d = 1$.
Man löst das Problem, indem man zunächst ein eindimensionales Gitter Γ gemäß Definition 2.1 mit der Knotenmenge $\Delta = \{x_0, ..., x_n\}$ auf der kompakten Menge Ω bildet, auf der das optimale Steuerungsproblem gelöst werden soll. Ziel ist es, einen endlichdimensionalen Funktionenraum zu finden, auf dessen Funktionen man den Operator T_h anwendet.
Eine geeignete Wahl ist hier der Raum $\mathcal{W}$ der stetigen und stückweise affin linearen Funktionen auf Ω bezüglich Γ. In Grüne [11], Kapitel 3.2, wird gezeigt, dass es möglich ist, v_h^j durch eine Funktion $\hat{v}_h^j \in \mathcal{W}$ anzunähern, und es ausreicht, die Werte an den Knotenpunkten des Gitters zu berechnen. Die Knotenpunkte bestimmen $\hat{v}_h^j$ eindeutig auf Ω. Diese $\hat{v}_h^j$ konvergieren gegen ein $\hat{v}_h \in \mathcal{W}$, wobei $\hat{v}_h$ wiederum für eine gegen Null gehende maximale Intervalllänge h_{max} gegen die exakte optimale Wertefunktion v_h konvergiert.

An dieser Stelle kommen nun die Splines ins Spiel. Als Alternative zu $\mathcal{W}$ wird der Raum der kubischen Splines auf der Stützstellenmenge Δ, also $\mathcal{S}_{\Delta,3}$, betrachtet

und untersucht, ob hierbei ähnliche Ergebnisse erzielt werden. Zur Vereinfachung sei im Folgenden immer $\mathcal{S}_\Delta = \mathcal{S}_{\Delta,3}$

In Lemma 2.6 ist gezeigt worden, dass jeder Spline $s \in \mathcal{S}_\Delta$ durch die Werte an den Knotenpunkten des Gitters $s(x_0), ..., s(x_n)$ sowie eine der beschriebenen Randbedingungen eindeutig bestimmt ist.
$s(x)$ ist für $x \in [x_{i-1}, x_i]$ durch die Vorschrift (2.5) bzw. (2.6) definiert.
Somit genügt es auch im Fall der Splines, für vorgegebene Randbedingungen die Werte an den Knoten des Gitters zu berechnen. Untersucht werden muss, ob sich die Funktionen v_h^j durch Splines $\check{v}_h^j \in \mathcal{S}_\Delta$ annähern lassen und ob diese Splines $\check{v}_h^j$ gegen ein $\check{v}_h^j \in \mathcal{S}_\Delta$ konvergieren.
Der folgende Abschnitt zeigt die Iterationsvorschrift für Funktionen aus $\mathcal{S}_\Delta$.

4.4 Vollständige Diskretisierung

Es genügt also für Splines aus $\mathcal{S}_\Delta$, den Operator T_h in jeder Iteration nur an den Knotenpunkten $x_i,\ i = 0, ..., n$ auszuwerten. Man berechnet demnach eine Folge von Funktionen $\check{v}_h^j \in \mathcal{S}_\Delta$ durch

$$\check{v}_h^{j+1}(x_i) = \max_{u \in U} \{hg(x_i, u) + \beta \check{v}_h^j(f_h(x_i, u))\}, \tag{4.10}$$

wie oben mit $\beta = 1 - \delta h$.
Schreibt man die Werte $\check{v}_h^j(x_i)$ als Einträge $V_i^j = \check{v}_h^j(x_i)$ in einen Vektor $V^j = (V_1^j, ..., V_{n+1}^j)^T \in \mathbb{R}^{n+1}$, so wird zu einem eindimensionalen Gitter Γ in jeder Iteration der Vektor $V^{j+1} \in \mathbb{R}^{n+1}$ durch $V^{j+1} = \check{T}_h(V^j)$ gemäß folgender Definition berechnet:

Definition 4.6 *Es sei ein diskretes optimales Steuerungsproblem sowie ein eindimensionales Gitter Γ auf $\Omega = [x_0, x_n]$ gemäß Definition 2.1 gegeben. $G(i, u)$ und $P_S(i, u)$ seien wie folgt definiert:*
Zu jedem $i = 1, ..., n + 1$ und jedem $u \in U$ sei

(i) $G(i, u) = hg(x_{i-1}, u)$

(ii) $P_S(i, u) = \check{s}(f_h(x_{i-1}, u))$
mit dem durch den Vektor $S \in \mathbb{R}^{n+1}$ mit den Einträgen $s_0, ..., s_n$ als Werte an den Knoten $x_0, ..., x_n$ bestimmten Spline $\check{s}$.
$P_S(i, u)$ errechnet sich dabei durch Auswertung des Splines $\check{s}$ am Punkt $f_h(x_{i-1}, u)$.

Dann werden die Komponenten der Vektoren V^j iterativ berechnet durch

$$V^{j+1} := V^j, \quad V_i^{j+1} := \max_{u \in U} \{G(i, u) + \beta P_{V^{j+1}}(i, u)\} \tag{4.11}$$

für $i = 1, ..., n + 1$ und mit $V^0 := (0, ..., 0)^T$.

Bemerkung 4.7 *(i) (4.11) beschreibt das Einzelschrittverfahren. Möglich ist auch das Gesamtschrittverfahren mit der Vorschrift*

$$V_i^{j+1} := \max_{u \in U}\{G(i,u) + \beta P_{V^j}(i,u)\} \quad für \quad i = 1, ..., n+1.$$

Dadurch, dass beim Einzelschrittverfahren für jedes $i > 1$ bereits die neuen Werte V_k^{i+1} für $1 \leq k < i$ berücksichtigt werden, kann allerdings hier eine leicht schnellere Konvergenz erwartet werden.

(ii) Durch die Einschränkung von U auf eine endliche Menge $\{u_1, ..., u_l\}$, $l \in \mathbb{N}$ wird das Maximum in einer Iteration für $i = 1, ..., n+1$ durch Vergleichen der Werte

$$G(i,u_k) + \beta P_{V^{j+1}}(i,u_k) \quad , \; k = 1, ..., l$$

bestimmt.
Diese einfache Bestimmung des Maximums liefert bis zu einer gewissen Genauigkeit brauchbare Ergebnisse. Wird eine sehr hohe Genauigkeit gefordert, sind sehr viele Kontrollwerte u_i nötig, was eine sehr lange Rechenzeit zur Folge hat. In diesem Fall sollten bessere kontinuierliche Optimierungsverfahren angewendet werden, auf die hier allerdings nicht weiter eingegangen werden soll.

Es bleibt zu beweisen, dass diese Vektoren V^j gegen einen Vektor V konvergieren. Im Fall der Approximation durch Funktionen aus $\mathcal{W}$ wird dazu gezeigt (siehe wiederum Grüne [11], Kapitel 3.2), dass durch Anwendung des Operators $\check{T}$ auf zwei beliebige Vektoren V,W mit den Werten der zugehörigen Funktionen $v, w \in \mathcal{W}$ an den Knoten als Einträgen eine Kontraktion auf dem $\mathbb{R}^n$ bezüglich der $L_\infty - Norm$ im $\mathbb{R}^n$ definiert ist durch

$$\left|\check{T}_h(V) - \check{T}_h(W)\right| \leq \beta \|v - w\|_\infty.$$

Die Werte $P_W(i,u)$ bzw. $P_V(i,u)$ erhält man in diesem Fall durch lineare Interpolation, d.h. $P_W(i,u)$ wird als Linearkombination aus den Werten an denjenigen Knotenpunkten berechnet, zwischen denen $(f_h(x_{i-1}, u))$ liegt.

Für Funktionen aus $\mathcal{S}_\Delta$ lässt sich analog folgende Aussage machen:

Lemma 4.8 *Seien $S, R \in \mathbb{R}^{n+1}$ zwei Vektoren, deren Einträge die Werten an den Knoten $x_0, ..., x_n$ des Gitters Γ auf $\Omega = [x_0, x_n]$ sind. Für die zwei zu diesen Werten gehörigen kubischen Splines $\check{s}, \check{r} \in \mathcal{S}_\Delta$ sowie eine Norm $\|.\|_s$ gelte folgende Ungleichung:*

$$\|\check{s}(x) - \check{r}(x)\|_s \leq \|S - R\|_s \tag{4.12}$$

für alle $x \in \Omega$. Zusätzlich gelte die Aussage aus Lemma 3.14 ebenfalls, wenn man dort anstatt der Beträge die Norm $\|.\|_s$ untersucht.

Betrachte nun die Vektoren V^j aus Definition 4.6. Diese Vektoren V^j konvergieren dann für $j \to \infty$ komponentenweise gegen den Vektor V.

V ist bestimmt durch

$$V = \check{T}_h(V).$$

Für die Komponenten von V gilt also

$$V_i = \max_{u \in U}\{G(i,u) + \beta P_V(i,u)\} \quad , i = 0, ..., n. \tag{4.13}$$

Für die zu V^j und V gehörigen Funktionen $\check{v}_h^j$ und $\check{v}_h$ aus $\mathcal{S}_\Delta$ gilt außerdem: Ist

$$\|V_i^j - V_i^{j+1}\|_s \leq \varepsilon \tag{4.14}$$

für alle $i = 0, ..., n$ erfüllt, so folgt

$$\|\check{v}_h^j(x) - \check{v}_h(x)\|_s \leq \frac{\varepsilon}{h\delta}$$

für alle $x \in \Omega$.

Beweis: Lemma 3.14 und (4.12) implizieren für alle $i = 1, ..., n$

$$\begin{aligned} \|\max_{u \in U}\{G(i,u) + \beta P_S(i,u)\} &- \max_{u \in U}\{G(i,u) + \beta P_R(i,u)\}\|_s \\ &\leq \max_{u \in U}\{\|\beta(P_S(i,u) - P_R(i,u))\|_s\} \\ &\leq \beta\|S - R\|_s. \end{aligned} \tag{4.15}$$

Da $\beta = 1 - \delta h < 1$ vorausgesetzt ist, definiert (4.15) eine Kontraktion auf dem $\mathbb{R}^n$ bezüglich der Norm $\|.\|_s$. Das bedeutet, dass der Vektor V existiert. Die Differenz $V^j - V$ wird in der Norm $\|.\|_s$ für wachsendes j immer kleiner, was durch Einsetzen von V^j und V in (4.15) deutlich wird:

$$\|V^{j+1} - V\|_s \leq \beta\|V^j - V\|_s$$

Damit ist die Konvergenz gezeigt.

Für einen Vektor W mit denselben Eigenschaften wie V gilt

$$\|V - W\|_s \leq \beta\|V - W\|_s < \|V - W\|_s.$$

Es ergibt sich also ein Widerspruch. Demnach ist $W = V$, womit die Eindeutigkeit von V bewiesen wäre. Die erste Ungleichung folgt hierbei aus (4.13) sowie (4.15), die zweite aus $\beta < 1$.

Außerdem folgt mit der Dreiecksungleichung und (4.14)

$$\begin{aligned} \|V^j - V\|_s &= \|V^j - V^{j+1} + V^{j+1} - V\|_s \\ &\leq \|V^j - V^{j+1}\|_s + \|V^{j+1} - V\|_s \\ &\leq \varepsilon + \beta\|V^j - V\|_s. \end{aligned}$$

Umgeformt ergibt sich

$$\|V^j - V\|_s \leq \frac{\varepsilon}{1-\beta} = \frac{\varepsilon}{h\delta}$$

und nach (4.12) somit

$$\|\check{v}_h^j(x) - \check{v}_h(x)\|_s \leq \|V^j - V\|_s \leq \frac{\varepsilon}{h\delta}.$$

□

Bemerkung 4.9 *(i) (4.12) beschreibt, dass der Abstand der Werte zweier Splines auf demselben Gitter bzgl. der Norm $\|.\|_s$ unter der Bedingung, dass die Werte an den Knoten in dieser Norm nahe zusammenliegen, durch diese Entfernung beschränkt ist. Es ist nicht klar, ob eine Norm mit den oben beschriebenen Eigenschaften existiert. Es ist aber anzunehmen, dass in dieser Norm nicht nur die Funktionswerte an den Knoten, sondern auch die Werte der ersten und evtl. auch der zweiten Ableitung an den Knoten einbezogen werden müssen. Eine dies betreffende Untersuchung soll jedoch nicht Gegenstand dieser Arbeit sein.*

In der Praxis zeigt sich, dass die Anwendung der Spline-Interpolation zur Berechnung der Werte $P_{V^j}(i,u)$ nicht zwingend ähnlich gute Ergebnisse für die Funktionen $\check{v}_h^j$ liefert wie lineare Interpolationsmethoden. Aus diesem Grund ist anzunehmen, dass die Voraussetzungen für diesen Satz nicht generell gültig sind und somit auch keine Konvergenz der Vektoren V^j erreicht wird. Kapitel 6 geht anhand der praktischen Ergebnisse näher auf diese Problematik ein.

(ii) Durch Abschätzung (4.14) ist ein Abbruchkriterium für den Algorithmus definiert. Lemma 4.8 besagt, dass die Funktion $\check{v}_h^j$ zu den Werten V_i^j an den Knoten x_i bzgl. der Norm $\|.\|_s$ nahe an der Approximation $\check{v}_h$ der exakten optimalen Wertefunktion liegt, wenn die Differenz zwischen V_i^j und V_i^{j-1} in dieser Norm für alle i entsprechend klein ist.

Bei der Implementierung setzt man einen Wert ζ fest, der von der gewünschten Genauigkeit abhängt. Da nach (i) nicht genau klar ist, welche Form $\|.\|_s$ hat, betrachtet man lediglich das Betrags-Maximum des Differenzvektors mit den Einträgen $\zeta_i = V_i^j - V_i^{j-1}$. Ist dieses kleiner als ζ, so ist die Approximation genau genug und der Algorithmus bricht ab.

4.4.1 Diskretisierungsfehler

Der durch die Diskretisierung im Ort verursachte Diskretisierungsfehler ist bei Anwendung der Spline-Interpolation schwer abzuschätzen. Die Analyse soll hier auch nicht weiter vertieft werden.

Der Diskretisierungsfehler, also die Differenz

$$\|v_h - \check{v}_h\|_\infty,$$

ist sicherlich abhängig vom Interpolationsfehler bei Approximation einer Funktion auf einer Knotenmenge Δ durch den kubischen Spline auf Δ. Dieser Fehler wird in Satz 2.12 abgeschätzt. Da die Funktion $\check{v}_h \in \mathcal{S}_\Delta$ allerdings nicht der Interpolationsspline zu v_h auf Δ ist, ist die Fehleranalyse hiermit noch nicht abgeschlossen.

Wie in Bemerkung 4.9 angedeutet, ist die Abhängigkeit des Abstands zwischen zwei Interpolationssplines vom Abstand der interpolierten Funktionen nicht explizit klar.
Dadurch lässt sich der Abstand zwischen v_h^j und $\check{v}_h^j$ und damit zwischen v_h und $\check{v}_h$ schwer abschätzen.

Im Fall der linearen Interpolation wird in Grüne [11], Abschnitt 3.2.3, gezeigt, dass die Abschätzung

$$\|v_h - \hat{v}_h\| \leq K \left(\frac{h_{max}}{h}\right)^\gamma \tag{4.16}$$

gilt, mit einer geeigneten Konstante $K > 0$ und einem von der Diskontrate δ und der Lipschitzkonstanten L von f abhängigen $\gamma \in]0, 1]$. h_{max} ist wieder die maximale Intervalllänge des Gitters Γ mit Knotenmenge Δ.

Aus dem „räumlichen Diskretisierungsfehler“ (4.16) und dem „zeitlichen Diskretisierungsfehler“ (4.2) lässt sich somit für den gesamten Diskretisierungsfehler folgern:

$$\|v - \hat{v}_h\|_\infty \leq K h^{\frac{\gamma}{2}} + K \left(\frac{h_{max}}{h}\right)^\gamma,$$

mit einer geeigneten Konstante $K > 0$ und einem von der Diskontrate δ und der Lipschitzkonstanten L von f abhängigen $\gamma \in]0, 1]$.

Demnach erhält man Konvergenz dann, wenn h und h_{max} so gegen Null gehen, dass auch der Quotient gegen Null geht. Diese Abschätzung ist allerdings relativ schwach, da auch für etwa gleich große Werte h und h_{max} akzeptable Ergebnisse erzielt werden. Eine stärkere Abschätzung wird in Falcone und Giorgi [9] bewiesen.

4.5 Mögliche Verbesserungen des Verfahrens

Es gibt Möglichkeiten, die häufig langwierige Bestimmung von V weiter zu beschleunigen. Im Folgenden werden zwei Ansätze mit diesem Ziel vorgestellt. Obwohl beide vor allem bei der Verwendung linearer Interpolationsmethoden eine großen Beschleunigungseffekt erzielen und die Übertragung auf Spline
-Interpolation nicht unproblematisch ist, sollen die zu Grunde liegenden Ideen kurz dargelegt werden.
Der erste Variante verbindet den in dieser Arbeit vorgestellten Lösungsansatz mit einer schon länger bekannten Iterationsvariante, der sogenannten Strategie-Iteration.
Die zweite Vorgehensweise, eine Form des Gauß-Seidel-Verfahrens, wird in dieser Arbeit lediglich bei der Berechnung der zum Vergleich nötigen Ergebnisse mit linearer Interpolation angewendet.

4.5.1 Strategie-Iteration

Die Idee der Strategie-Iteration (vom englischen Wort policy-iteration) ist es, sich die Maximierung über die Werte u bei der Bestimmung des Vektors V^j zu sparen. Dies ist möglich, da (im Falle der linearen Interpolation) für einen Vektor $\tilde{u} = (\tilde{u}_0, ..., \tilde{u}_{n-1})^T \in U^n$ die Iteration

$$V^{\tilde{u},j+1} := V^{\tilde{u},j}, \quad V_i^{\tilde{u},j+1} := G(i, \tilde{u}_i) + \beta P_{V^{\tilde{u},j+1}}(i, \tilde{u}_i) \tag{4.17}$$

gegen den Vektor $V^{\tilde{u}}$ konvergiert, dessen Komponenten durch

$$V_i^{\tilde{u}} = G(i, \tilde{u}_i) + \beta P_{V^{\tilde{u},j+1}}(i, \tilde{u}_i)$$

eindeutig gegeben sind.

Wählt man nun in jeder Iteration, also zu jedem Vektor V^j gerade den Vektor $\tilde{u}^j$, dessen Komponenten die maximierenden Kontrollwerte aus (4.11) sind, um $V^{\tilde{u}^j} = V^{j+1}$ zu berechnen, so konvergieren die Vektoren V^j quadratisch gegen den Vektor V, dessen zugehörige Funktion $\hat{v}_h \in \mathcal{W}$ die optimale Wertefunktion approximiert. Der Beweis befindet sich in Puterman, Brumelle, *On the convergence of policy iteration in stationary dynamic programming*, Math. of Operations Research, 4 (1979).
Vorausgesetzt, man kann man diese Eigenschaften auf das Verfahren mit Spline-Interpolation übertragen, ist dies ist ein Vorteil gegenüber der Iteration (4.11), von der nur lineare Konvergenz erwarten werden kann.

Da dieses Verfahren am Anfang ziemlich langsam konvergiert und die optimalen Kontrollwerte im Allgemeinen unbekannt sind, verbindet man diese beiden Iterationsverfahren.

Zunächst erfolgt die iterative Berechnung der Vektoren V^j wie vorher auch durch

$$V^{j+1} := V^j, \quad V_i^{j+1} := \max_{u \in U} \{G(i,u) + \beta P_{V^{j+1}}(i,u)\}$$

für $i = 1, ..., n+1$ und mit $V^0 := (0, ..., 0)^T$.

Wichtig ist dabei, jeweils die optimalen (d.h. maximierenden) Kontrollwerte in einen Vektor $\tilde{u}^j \in U^n$ abzuspeichern. Wenn sich nun diese Vektoren $\tilde{u}^j$ nur noch wenig bzw. überhaupt nicht mehr ändern, wird zur Strategie-Iteration übergegangen. Eine leicht zu implementierende Möglichkeit dies festzustellen ist, zu überprüfen, wieviele Einträge der beiden Vektoren $\tilde{u}^j$ und $\tilde{u}^{j-1}$ übereinstimmen. Liegt diese Übereinstimmung über einer gewissen Grenze (sinnvoll ist eine Wert zwischen 80 und 100 Prozent) wird das Verfahren gewechselt. Diese Vorgehensweise stammt aus Seeck, *Iterative Lösungen der Hamilton-Jacobi-Bellman-Gleichung bei unendlichem Zeithorizont*, Diplomarbeit, Universität Kiel,1997.

Man lässt dann $\tilde{u}^j$ fest und berechnet so iterativ die Vektoren $V^{\tilde{u}^j,k}$ durch (4.17) mit $V^{\tilde{u}^j,0} = V^j$. Diese konvergieren gegen den Vektor $V^{\tilde{u}^j}$, der dann als Vektor V^{j+1} für die nächste Iteration des ursprünglichen Verfahrens verwendet wird.

Es zeigt sich, dass mit dieser Variante im Falle einer differenzierbaren optimalen Wertefunktion die Rechenzeit erheblich reduziert werden kann. Einzelheiten folgen in Abschnitt 6.2.1.

4.5.2 Kontrolliertes Gauß-Seidel-Verfahren

Das kontrollierte Gauß-Seidel-Verfahren (auch Koordinatenaufstiegsverfahren) fasst die Iterationsvorschrift

$$V^{j+1} := V^j, \quad V_i^{j+1} := \max_{u \in U} \{G(i,u) + \beta P_{V^{j+1}}(i,u)\}, \tag{4.18}$$

für $i = 1, ..., n+1$ und mit $V^0 := (0, ..., 0)^T$, nicht als Zuweisung, sondern als Gleichung auf.

Durch die Bestimmung des Wertes $P_{V^{j+1}}(i,u)$ mittels Interpolation (linear oder mit Splines) geht der Wert V_i^{j+1} auch in die rechten Seite ein. Betrachtet man (4.18) nun als Gleichung, kann man den von V_i^{j+1} abhängigen Term auf die linke Seite bringen und so die Gleichung nach V_i^{j+1} auflösen. Man erhält

$$V_i^{j+1} = \max_{u \in U} \left\{ \frac{G(i,u) + \beta \tilde{P}_{V^{j+1}}(i,u)}{1 - \beta\rho} \right\}, \tag{4.19}$$

wobei $\tilde{P}_{V^{j+1}}(i,u)$ dem Wert $P_{V^{j+1}}(i,u)$ ohne den Term mit V_i^{j+1} entspricht. $\rho \in [0,1]$ ist der Vorfaktor von V_i^{j+1} in der Interpolationsgleichung $P_{V^{j+1}}(i,u)$.

Ist der Faktor ρ gleich Null, so entspricht der durch das kontrollierte Gauß-Seidel-Verfahren ermittelte Wert für V_i^{j+1} genau dem, der mit dem ursprünglichen Verfahren ermittelt wird.

Ist dagegen $\rho > 0$, wird der Nenner in (4.19) kleiner als 1 und V_i^{j+1} ist hier größer als beim ursprünglichen Verfahren. Da man das Maximum sucht, wird somit eine Beschleunigung erzielt. Folglich ist die Beschleunigung umso größer, je größer ρ ist.
Da $P_{V^{j+1}}(i,u)$ den Wert der Funktion $\hat{v}_{j+1}$ bzw. $\check{v}_{j+1}$ an der Stelle $f_h(x_i,u)$ enthält, ist ρ genau dann groß, wenn $f_h(x_i,u)$ für das maximierende u sehr nahe bei x_i liegt.

Nach Grüne [11], Kapitel 4.2, konvergiert dieses Verfahren ebenso wie die ursprüngliche Iterationsvorschrift linear gegen den Fixpunkt $V \in \mathbb{R}^{n+1}$.

Für den Fall der linearen Interpolation ist die Implementierung gerade im Eindimensionalen relativ unkompliziert, da der Vorfaktor ρ leicht aus der Interpolationsgleichung $P_{V^{j+1}}(i,u)$ zu entnehmen, und die Gleichung

$$V_i^{j+1} = \max_{u \in U}\{G(i,u) + \beta P_{V^{j+1}}(i,u)\}$$

somit sehr einfach nach V_i^{j+1} aufzulösen ist.

Bei Anwendung der Spline-Interpolation gestaltet sich dieses Vorgehen als schwieriger, wenn man die MATLAB-eigenen Routinen zur Splineberechnung benutzt. MATLAB errechnet den Wert eines Splines am Punkt x durch Gleichungen der Form (2.5). Daraus ist der Vorfaktor ρ nicht direkt zu entnehmen.

Verwendet man eigene Routinen, ist es unkomplizierter, (2.5) in die Form (2.6) zu bringen, aus der der Faktor $\rho = K_{i1}$ bzw. $\rho = K_{i2}$ direkt abzulesen ist.

Kapitel 5

Adaptive Gitter

5.1 Grundidee der adaptiven Gittererzeugung

In Kapitel 4 wird die Approximation der optimalen Wertefunktion v_h eines zeitdiskreten optimalen Steuerungsproblems auf $\Omega = [x_0, x_n]$ durch den Spline $\check{v}_h$ auf einem gegebenen festen Gitter Γ mit Knotenmenge $\Delta = \{x_0, ..., x_n\}$ dargestellt. Eine ausreichend genaue Berechnung kann allerdings (auch bei Anwendung der beschriebenen Verbesserungsmöglichkeiten) immer noch sehr lange dauern. Ein Problem ist, dass die Implementierung des Verfahrens für sehr viele Knoten eine enorm lange Rechenzeit und sehr viel Speicherplatz in Anspruch nimmt. Für wenige Knoten kann es dagegen passieren, dass es Punkte $x \in \Omega \setminus \Delta$ gibt, deren approximierter Lösungswert $\check{v}_h(x)$ relativ stark vom tatsächlichen Wert $v_h(x)$ abweicht, die Approximation die exakte Lösung also nur unzureichend wiedergibt.

Dieses Problem löst die sogenannte adaptive Gittererzeugung, die bei der numerischen Lösung verschiedenster partieller Differentialgleichungen herangezogen werden kann.
Die Idee ist, die Approximation auf einem Startgitter Γ_0 zu berechnen, dann das Gitter geeignet zu verändern und anschließend die Lösung auf diesem neuen Gitter Γ_1 zu berechnen. Man setzt das so lange fort, bis man eine zufriedenstellende Approximation auf ganz $\Omega = [x_0, x_n]$ erhalten hat. Wie man die „Qualität“ der erhaltenen Approximation bestimmt und daraufhin das Gitter geeignet adaptiert, wird im Folgenden genauer betrachtet.

Bemerkung 5.1 *Wieder liegt das Hauptaugenmerk auf dem eindimensionalen Fall. Die entsprechende Vorgehensweise bei zweidimensionalen Problemen mit einem Rechteckgitter $\tilde{\Gamma}$ wird allerdings zur Verdeutlichung kurz aufgezeigt. Das Konzept der adaptiven Gittererzeugung kann auch auf beliebig höhere Dimensionen ausgeweitet werden.*

Eine Möglichkeit das Gitter zu verändern wäre es, mit einem groben Gitter zu starten, und nach jeder Lösungsberechnung äquidistant zu verfeinern, d.h. jedes

Intervall zu halbieren. Das führt zu einer rechenzeitlichen Verbesserung gegenüber der Berechnung auf einem sehr feinen Startgitter ohne Gitterveränderung, verschwendet aber immer noch Zeit und Speicherplatz. Der Grund dafür ist, dass es vorkommen kann, bzw. in den meisten Fällen vorkommen wird, dass die Approximation $\check{v}_h$ auf einem Gitter Γ_k nicht auf allen Intervallen $[x_{i-1}, x_i],\ i = 1, ..., n$, dieses Gitters gleich gut bzw. gleich schlecht ist. Das heißt auf manchen Teilintervallen I_j existieren Punkte x, in denen $\check{v}_h(x)$ noch relativ weit von $v_h(x)$ entfernt ist, während auf anderen Intervallen I_k die Differenz zwischen approximierter und tatsächlicher Lösung für alle $x \in I_k$ ausreichend klein ist. Bei der adaptiven Gittererzeugung wird nun dieser Fehler auf den einzelnen Teilintervallen abgeschätzt und das Gitter anschließend an den Intervallen I_j mit vergleichsweise großem Fehler verfeinert.
Bevor die genaue Vorgehensweise bei der Verfeinerung erläutert wird, soll vorweg die Bestimmung des lokalen Fehlers auf den einzelnen Teilintervallen beschrieben werden.

5.2 Lokale Fehlerschätzer

In Abschnitt 4.4.1 wird eine Abschätzung für den Fehler a-priori ausschließlich aus den Daten des Problems bestimmt. Nun gilt es unter Einbeziehung einer aus dem Lösungsalgorithmus zu einem Gitter Γ erhaltenen Approximation $\check{v}_h$, die maximale Differenz $\|v_h - \check{v}_h\|_\infty$ zu bestimmen, ohne die exakte Lösung v_h zu kennen. Die fehlende Kenntnis der tatsächlichen Lösung lässt keine exakte Bestimmung dieser Differenz, also des Fehlers der Approximation, zu.
Im Folgenden wird der Fehler durch einen sogenannten residualen Fehlerschätzer in der $\infty - Norm$ a-posteriori abgeschätzt. Die Definition und die Eigenschaften des Fehlerschätzers stammen aus Grüne [11], Kapitel 4.

Definition 5.2 *Für ein vollständig diskretes optimales Steuerungsproblem auf einem Gitter Γ mit $n \in \mathbb{N}$ Intervallen $I_1, ..., I_n$ ist ein lokaler a-posteriori Fehlerschätzer (in der $\infty - Norm$) eine Menge von Werten $\eta_1, ..., \eta_n$ mit folgenden Eigenschaften.*

(i) Der Wert η_i lässt sich aus den Daten des optimalen Steuerungsproblems und aus der Funktion $\check{v}_h$ in einer Umgebung $\mathcal{N}(I_i)$ berechnen.

(ii) Es existieren von Γ unabhängige Konstanten $C_1, C_2 > 0$, so dass für den Wert

$$\eta := \max_{i=1,...,n} \eta_i$$

die Abschätzungen

$$C_1\eta \leq \|v_h - \check{v}_h\|_\infty \leq C_2\eta$$

gelten. Man nennt den Fehlerschätzer effizient und zuverlässig.

Der Fehlerschätzer heißt lokal effizient, wenn die Abschätzung

$$C_1\eta_i \leq \sup_{x\in\mathcal{N}(I_i)} |v_h(x) - \hat{v}_h(x)|$$

gilt.

Effizient bedeutet hier, dass man aus einem großen Fehlerschätzer einen großen Fehler folgern kann, und zuverlässig steht dafür, dass ein kleiner Fehlerschätzer einen kleinen Fehler impliziert. Durch Effizienz wird also ein Überschätzen vermieden, während die Zuverlässigkeit ein Unterschätzen des Fehlers verhindert.

Durch diese Definition lässt sich schon erkennen, wie ein lokaler Fehlerschätzer zur adaptiven Gittererzeugung herangezogen wird. Die lokale Effizienz garantiert, dass ein großer Wert η_i auf einen großen Fehler in der Umgebung hinweist. Daraus ergibt sich für die Gitteradaption, dass es sinnvoll ist, das Gitter in den Bereichen mit großem Fehlerschätzer zu verfeinern, während auf den Bereichen mit kleinem Fehlerschätzer keine Verfeinerung notwendig ist.

Nun benötigt man einen geeigneten lokalen Fehlerschätzer für diesen Algorithmus:

Die exakte Lösung des betrachteten diskreten optimalen Steuerungsproblems erfüllt die Gleichung

$$v_h = T_h(v_h)$$

mit T_h aus (4.5). Durch die örtliche Diskretisierung löst der Algorithmus allerdings die Gleichung

$$V = \check{T}_h(V)$$

mit dem zu $\check{v}_h \in \mathcal{S}_\Delta$ gehörigen Vektor V. Auf ganz Ω wird die Gleichung

$$\check{v}_h = \pi_{\mathcal{S}_\Delta} T_h(\check{v}_h)$$

approximativ gelöst. $\pi_{\mathcal{S}_\Delta}$ ist dabei die Projektion auf den Raum der kubischen Splines auf Δ. Die Idee ist nun, den lokalen Fehlerschätzer durch das Residuum von T_h bzgl. $\check{v}_h$, also

$$\|\check{v}_h - T_h(\check{v}_h)\|_\infty$$

zu berechnen:

Definition 5.3 *Sei eine Funktion $\eta : \Omega \to \mathbb{R}_0^+$ gegeben durch*

$$\begin{aligned}\eta(x) &:= |\hat{v}_h(x) - T_h(\hat{v}_h)(x)| \\ &= \left|\hat{v}_h(x) - \left(\max_{u\in U}\{hg(x,u) + \beta\hat{v}_h(f_h(x,u))\}\right)\right|.\end{aligned}$$

Durch $\eta(x)$ sei dann der lokale Fehlerschätzer definiert gemäß

$$\eta_i := \max_{x \in I_i} \eta(x) \quad , i = 1, ..., n. \tag{5.1}$$

Eigenschaft (i) aus Definition 5.2 ist offensichtlich erfüllt. Die Werte η_i aus (5.1) hängen nur von den Werten des diskreten optimalen Steuerungsproblems (f_h, g, δ, h) und den Werten von $\check{v}_h$ in der Umgebung $\mathcal{N}(I_i)$ ab.
$\mathcal{N}(I_i)$ ist dabei definiert als

$$\mathcal{N}(I_i) := \{y \in \Omega | \text{ es existiert ein } x \in I_i \text{ mit } \|y - x\| \leq M_h\}, \tag{5.2}$$

wobei M_h eine obere Schranke für $\|x - f_h(x, u)\|$ für alle $x \in \Omega$ und alle $u \in U$ ist. Bei Anwendung der Euler-Diskretisierung gilt gerade $M_h = hM$, mit M einer oberen Schranke für $\|f(x, u)\|$. Folglich ist $\mathcal{N}(I_i)$ für kleines $h > 0$ auch eine kleine Umgebung.

Die zweite Eigenschaft aus Definition 5.2 sowie die lokale Effizienz beweist folgender Satz:

Satz 5.4 *Es sei $\delta h < 1$. Für*

$$\eta = \max_{i=1,...,n} \eta_i$$

mit den lokalen Fehlerschätzern η_i gemäß (5.1) gilt

$$\frac{1}{2}\eta \leq \|v_h - \hat{v}_h\|_\infty \leq \frac{1}{\delta h}\eta. \tag{5.3}$$

Außerdem gilt

$$\frac{1}{2}\eta_i \leq \sup_{x \in \mathcal{N}(I_i)} |v_h(x) - \hat{v}_h(x)| \tag{5.4}$$

für die Umgebung (5.2).

Beweis: Seien zwei stetige Funktionen $v_1, v_2 : \Omega \to \mathbb{R}$ gegeben. Dann gilt nach Lemma 3.14

$$\begin{aligned} |T_h(v_1)(x) - T_h(v_2)(x)| &= \Big| \max_{u \in U} \{hg(x, u) + \beta v_1(f_h(x, u))\} \\ &\quad - \max_{u \in U} \{hg(x, u) + \beta v_2(f_h(x, u))\} \Big| \\ &\leq \max_{u \in U} |\beta(v_1(f_h(x, u)) - v_2(f_h(x, u)))| \\ &\leq \beta \max_{y \in \bar{B}_{M_h}(x)} |v_1(y) - v_2(y)|. \end{aligned} \tag{5.5}$$

Aus der Tatsache, dass $v_h = T_h(v_h)$ gilt, folgt

$$\begin{aligned}
\eta(x) &= |\hat{v}_h(x) - T_h(\hat{v}_h)(x)| \\
&= |\hat{v}_h(x) - v_h(x) + T_h(v_h)(x) - T_h(\hat{v}_h)(x)| \\
&\leq |\hat{v}_h(x) - v_h(x)| + |T_h(\hat{v}_h)(x) - T_h(v_h)(x)| \\
&\leq |\hat{v}_h(x) - v_h(x)| + \beta \max_{y \in \bar{B}_{M_h}(x)} |\check{v}_h(y) - v_h(y)| \\
&\leq 2 \max_{y \in \bar{B}_{M_h}(x)} |\hat{v}_h(y) - v_h(y)|.
\end{aligned} \tag{5.6}$$

Dabei resultiert die vorletzte Ungleichung aus (5.5) und die letzte aus $\beta = 1 - \delta h < 1$. Maximiert man nun über alle $x \in I_i$, so erhält man (5.4).
Die erste Abschätzung aus (5.3) entsteht, wenn man nun das Maximum über $i = 1, ..., n$, also über alle Intervalle, bildet. Für die zweite Abschätzung verwendet man wieder $v_h = T_h(v_h)$ und schreibt:

$$\begin{aligned}
|v_h(x) - \hat{v}_h(x)| &= |T_h(v_h)(x) - \hat{v}_h(x)| \\
&= |T_h(v_h)(x) - \hat{v}_h(x) + T_h(\hat{v}_h)(x) - T_h(\hat{v}_h)(x)| \\
&\leq |T_h(\hat{v}_h)(x) - \hat{v}_h(x)| + |T_h(v_h)(x) - T_h(\hat{v}_h)(x)| \\
&\leq \eta(x) + \beta \max_{y \in \Omega} |v_h(y) - \hat{v}_h(y)|,
\end{aligned}$$

wobei die letzte Ungleichung wiederum aus (5.5) folgt.
Bildet man nun das Maximum über alle $x \in \Omega$, so ergibt sich

$$\|v_h - \hat{v}_h\|_\infty \leq \eta + \beta \|v_h - \hat{v}_h\|_\infty$$

Umgeformt erhält man

$$(1 - \beta)\|v_h - \hat{v}_h\|_\infty \leq \eta.$$

Da $1 - \beta = \delta h$ ist, ist auch die zweite Abschätzung aus (5.3) bewiesen.

□

Nachdem nun bewiesen ist, dass es sich bei den in (5.1) definierten η_i um lokale Fehlerschätzer handelt, gilt es, diese Werte zu berechnen. Wieder taucht das bekannte Problem auf, dass man $\eta(x)$ an unendlich vielen Punkten auswerten müsste, um

$$\eta_i = \max_{x \in I_i} \eta(x)$$

exakt zu bestimmen. Da aber die Funktion $\check{v}_h$ als kubischer Interpolationsspline nach Definition stetig ist, kann man das Maximum über alle $x \in I_i$ durch Auswertung von $\eta(x)$ in Testpunkten approximieren. Im eindimensionalen Fall zeigt die Praxis, dass es genügt, den Mittelpunkt des Intervalls I_i als Testpunkt $\tilde{x}_i$ zu wählen.

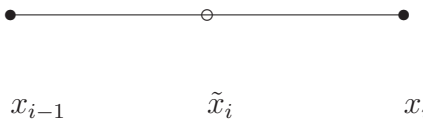


Abbildung 5.1: Testpunkt im eindimensionalen Fall

Somit ergibt sich

$$\eta_i = \eta(\tilde{x}_i) = |\hat{v}_h(\tilde{x}_i) - T_h(\hat{v}_h)(\tilde{x}_i)|.$$

Im zweidimensionalen Fall eignen sich folgende fünf Testpunkte $\tilde{x}_{ij},\ j = 1, ..., 5$, zur Auswertung über alle x im Rechteck R_i:

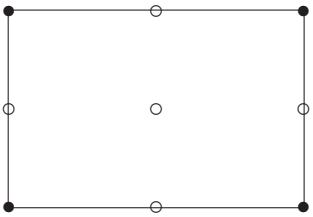


Abbildung 5.2: Testpunkte im zweidimensionalen Fall

Hier gilt also

$$\begin{aligned} \eta_i &= \max_{j=1,...,5}\{\eta(\tilde{x}_{ij})\} \\ &= \max_{j=1,...,5}\{|\hat{v}_h(\tilde{x}_{ij}) - T_h(\hat{v}_h)(\tilde{x}_{ij})|\}. \end{aligned}$$

Diese Vorgehensweise lässt sich leicht auf höhere Dimensionen verallgemeinern.

Bemerkung 5.5 *Die Wahl der Knotenpunkte, also im Eindimensionalen der Intervallendpunkte sowie im Zweidimensionalen der Rechteck-Eckpunkte, als Testpunkte ist natürlich wenig sinnvoll, da sich in diesen Punkten $\check{v}_h$ nach (4.13) nicht mehr ändert. Der Fehler in den Knotenpunkten ist also gleich Null.*

5.3 Durchführung der adaptiven Gittererzeugung

Die in den beiden vorangegangenen Abschnitten formulierten Ansätze werden jetzt kombiniert, um eine möglichst genaue Approximation der diskreten optimalen Wertefunktion v_h in akzeptabler Rechenzeit zu erhalten.
Für ein Gitter Γ_j läuft die Gitteradaption zusammengefasst folgendermaßen ab (siehe auch Grüne und Semmler [13]):

Sei $\check{v}_{h,j}$ die auf dem Gitter Γ_j errechnete Approximation von v_h.

- Man berechnet die lokalen Fehlerschätzer η_i gemäß (5.1) durch Anwendung des Operators T_h aus (4.5) auf die $\check{v}_{h,j}$ in den Testpunkten $\tilde{x}_i$ der Intervalle I_i , $i = 1, ..., n$.

- Wenn alle η_i , $i = 1, ..., n$ kleiner als eine angegebene Schranke sind, ist die Approximation $\check{v}_{h,j}$ schon gut genug. Es muss also kein neues Gitter mehr gebildet werden.

- Wenn das nicht der Fall ist, bestimmt man $\eta = \max_{i=1,...,n} \eta_i$.

- Man verfeinert alle Intervalle I_i, deren zugehöriges η_i relativ groß im Vergleich zu η ist, d.h. man verfeinert (also halbiert) alle Intervalle I_i mit $\eta_i \geq \theta \cdot \eta$. Die Größe des Parameters $\theta \in [0, 1]$ hängt von dem zu lösenden Problem ab. Oft erweist sich $\theta = 0.1$ als eine gute Wahl.

- Durch Verfeinerung werden also die Testpunkte $\tilde{x}_i$ mit großem Fehler zu Knotenpunkten im neuen Gitter Γ_{j+1}.

- Nun approximiert man die Lösung auf dem neuen Gitter Γ_{j+1}.

Im mehrdimensionalen Fall hat man verschiedene Möglichkeiten, die Zellen zu verfeinern. Bei der einfachsten Variante werden die Zellen mit großem Fehler in jeder möglichen Koordinatenrichtung verfeinert, im Zweidimensionalen ergibt sich:

Abbildung 5.3: Verfeinerung in beiden Koordinatenrichtungen

Bei bestimmten mehrdimensionalen Problemen kann es vorkommen, dass die Funktion $\hat{v}_h$ in manchen Richtungen eine starke Krümmung aufweist, in anderen Richtungen jedoch gar nicht bzw. nur schwach gekrümmt ist. Das führt dazu, dass der Fehler in manchen Koordinatenrichtungen im Vergleich zu anderen sehr groß ist. Hier ist es sinnvoll, die Zelle nur in den Richtungen mit großem Fehler zu verfeinern, im Zweidimensionalen also z.B.:

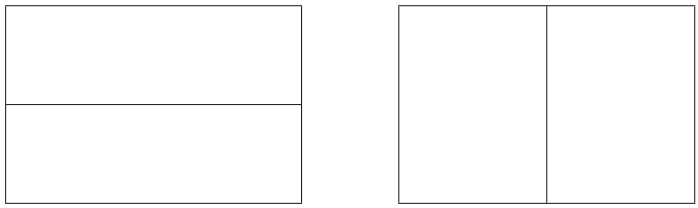


Abbildung 5.4: Verfeinerung in jeweils einer Koordinatenrichtung

Bemerkung 5.6 *Die Vorgehensweise der Gitteradaption funktioniert selbstverständlich auch in die andere Richtung. Stellt man zum Beispiel fest, dass ausgehend von einem groben Startgitter auch in Bereichen verfeinert worden ist, die ein sehr regelmäßiges Verhalten der Lösung zeigen, so können hier Knotenpunkte gestrichen werden. Das bewirkt eine „Weitung“ des Gitters an den entsprechenden Stellen. Diese Stellen sind auffindbar, indem man die Approximation $v_{h,j}$ auf dem Gitter Γ_j mit seiner Projektion $\tilde{v}_{h,j}$ auf dem Gitter $\tilde{\Gamma}_j$ vergleicht. Dabei entsteht $\tilde{\Gamma}_j$ aus Γ_j durch einmaliges “Ausweiten“ jeder Zelle.*
Anschließend kann man jede Zelle tatsächlich „ausweiten“, in der $\tilde{v}_{h,j}$ nicht weit von $v_{h,j}$ abweicht, oder anders ausgedrückt, in der die Approximation $v_{h,j}$ nicht viel besser ist als $\tilde{v}_{h,j}$. Diese Abweichung wird analog zur Fehlerbestimmung auf einer Zelle bei der Gitterverfeinerung bestimmt.
Dieses Verfahren macht allerdings erst bei Problemen höherer Dimension Sinn. Bei den in dieser Arbeit betrachteten eindimensionalen Problemen wird es nicht angewendet.

Die adaptive Gitterveränderung spart wie gesagt Speicherplatz und Rechenzeit. Zum einen werden durch die Konzentration auf die „komplizierten“ Regionen insgesamt weniger Gitterpunkte benötigt, um eine gewünschte Genauigkeit der Approximation zu erhalten. Zum anderen genügen dadurch auch weniger Iterationen, um diese Genauigkeit zu erreichen. Der Start auf einem sehr groben Gitter spart zusätzlich Rechenzeit, da die Lösungen auf dem alten Gitter immer als Startwerte auf dem neuen Gitter benutzt werden. Somit sind auf den letzten Gittern mit sehr vielen Knoten nur wenige Iterationen notwendig.

Beide Einsparungen hängen sehr stark von dem zu lösenden Problem ab und sind deshalb sehr schwer generell abzuschätzen. In Kapitel 6 wird bei Beispiel 1 genauer auf den Vergleich zwischen den auf äquidistanten Gittern erhaltenen Ergebnissen und denjenigen, die mit adaptiver Gitterveränderung erzeugt worden sind, eingegangen.

Eine Größe, über die allerdings allgemeingültige Aussagen gemacht werden können, ist die Reduzierung des Interpolationsfehlers bei Gitteradaption. Dieser der offensichtlich auch direkten Einfluss auf die Genauigkeit der Approximation im

Lösungsalgorithmus. In dieser Arbeit interessiert natürlich in erster Linie der Fall der Spline-Interpolation.
Wie in Satz 2.12 deutlich wird, hängt der Fehler bei der Interpolation mit kubischen Splines direkt von der Intervalllänge, also dem Abstand der einzelnen Knoten, ab. Der Fehler wird dort durch die maximale Intervalllänge h_{max} abgeschätzt. Wie sich der Fehler lokal verändert, wenn man einige Intervalle verfeinert, untersuchen die folgenden beiden Abschnitte. Zur besseren Veranschaulichung wird vorausgesetzt, dass mit Verfeinerung eine Halbierung gemeint ist. Dies entspricht auch der Vorgehensweise in der Praxis.

5.4 Theoretische Verbesserung des Ergebnisses durch Anwendung von adaptiven Gittern

In Satz 2.12 ist der Fehler zwischen einer Funktion f und dem zugehörigen kubischen Interpolationsspline s auf einer gegebenen Stützstellenmenge $\Delta = x_0, ..., x_n$ mit maximaler Intervalllänge h_{max} abgeschätzt worden. Zur Erinnerung:

$$\|s - f\|_\infty \leq \frac{5}{384}\|f^{(4)}\|h_{max}^4$$

Wie verändert sich nun aber der Fehler, wenn man das Gitter in einigen Intervallen verfeinert? An der Schranke wird sich nichts ändern, da die maximale Intervalllänge h_{max} im Allgemeinen gleich bleiben wird. Die Frage ist also eher, ob und wie sich der Fehler in den verfeinerten Bereichen verringert.
Dazu betrachtet man die drei einzelnen Teil-Lemmas, in die sich der Beweis aufgliedert.
Lemma 2.19 schätzt den Abstand zwischen f und dem stückweise kubischen Polynom q ab, dessen Werte der 0-ten und der ersten Ableitung in den Stützstellen mit denen von f übereinstimmen. Dieser Abstand ist auf jedem Intervall $I_i = [x_{i-1}, x_i]$ gerade kleiner als

$$\|f - q\|_\infty \leq \frac{1}{384}\|f^{(4)}\|h_i^4 \quad , h_i = x_i - x_{i-1}.$$

Somit ist sofort ersichtlich, dass sich bei Änderung der Intervalllänge durch Gitteradaption von h_i auf $\tilde{h}_i = \frac{1}{2}h_i$ dieser Abstand um den Faktor $\frac{1}{16}$ ändert.

Lemma 2.22 wiederum grenzt den Abstand zwischen q und dem Interpolationsspline s ab durch:

$$\|s - q\|_\infty \leq \frac{1}{24}\|f^{(4)}\|(h_{max})^3(K_{i3}(x) + K_{i4}(x)),$$

wobei für $(K_{i3}(x) + K_{i4}(x))$ auf dem Intervall $I_i = [x_{i-1}, x_i]$ gilt:

$$(K_{i3}(x) + K_{i4}(x)) \leq \frac{h_i}{4}.$$

Auch hier wirkt sich also die Veränderung durch Gitterverfeinerung direkt auf die obere Schranke aus. Die Schranke reduziert sich im selben Maße wie die Länge des betrachteten Intervalls.
Der Faktor $\frac{1}{24}\|f^{(4)}\|(h_{max})^3$ entsteht durch die Abschätzung der Differenz zwischen der ersten Ableitung von f und der ersten Ableitung von s an den Knoten x_i und wird in Lemma 2.13 bewiesen:

$$|s'(x_i) - f'(x_i)| \leq \frac{1}{24}\|f^{(4)}\|h_{max}^3 \quad , i = 0, ..., n$$

Hier lässt sich die genaue Auswirkung einer Verfeinerung nicht so leicht erkennen. Das soll durch das folgende einfache Rechenbeispiel veranschaulicht werden.
Die Werte des Vektors $E \in \mathbb{R}^{n-1}$ mit den Einträgen

$$E_i = s'(x_i) - f'(x_i) \quad , \ i = 1, ..., n-1,$$

errechnen sich aus dem Gleichungssystem

$$\begin{pmatrix} 2(h_2+h_1) & h_1 & 0 & \cdots & 0 & 0 \\ h_3 & 2(h_3+h_2) & h_2 & \ddots & & 0 \\ 0 & h_4 & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & & & \ddots & \ddots & h_{n-2} \\ 0 & 0 & \dots & 0 & h_n & 2(h_n+h_{n-1}) \end{pmatrix} E = Z$$

mit $[Z]_i = \frac{-1}{24} f^{(4)}(\xi_i)[h_{i+1}(h_i^3) + h_i(h_{i+1})^3]$ für $i = 1, ..., n-1$ und $\xi_i \in [x_{i-1}, x_{i+1}]$.

Nimmt man zur Vereinfachung an, ein sehr einfaches Ausgangsgitter habe sieben Knoten (also fünf innere Knoten) und sei äquidistant, d.h. alle h_i seien gleich groß ($h_i = h, \ i = 1, ..., 6$), so ergibt sich

$$\begin{pmatrix} 4h & h & 0 & 0 & 0 \\ h & 4h & h & 0 & 0 \\ 0 & h & 4h & h & 0 \\ 0 & 0 & h & 4h & h \\ 0 & 0 & 0 & h & 4h \end{pmatrix} \tilde{E} = \tilde{Z} \tag{5.7}$$

mit $[\tilde{Z}]_i = \frac{-1}{24} f^{(4)}(\xi_i) 2h^4 \ , \ i = 1, ..., 5.$

Bei der Lösung dieses Gleichungssystems (z.B. mit Maple) ergeben sich die exakten Werte:

$$\tilde{E}_1 = \frac{-11}{624}\hat{f}h^3, \quad \tilde{E}_2 = \frac{-1}{78}\hat{f}h^3, \quad \tilde{E}_3 = \frac{-3}{208}\hat{f}h^3, \quad \tilde{E}_4 = \frac{-1}{78}\hat{f}h^3, \quad \tilde{E}_5 = \frac{-11}{624}\hat{f}h^3$$

Dabei ist $\hat{f} = \|f^{(4)}\|$. Nun verfeinert man das Gitter zunächst nur einmal in einem Intervall, d.h. es wird ein neuer Knoten eingefügt und aus einem Intervall der Länge h entstehen zwei Intervalle der Länge $\frac{h}{2}$.
O.B.d.A. sei hier I_4 das verfeinerte Intervall. Aus $h_4 = h$ werden somit $\hat{h}_{41} = \frac{h}{2}$ und $\hat{h}_{42} = \frac{h}{2}$. Das obige Gleichungssystem (5.7) verändert sich zu

$$\begin{pmatrix} 4h & h & 0 & 0 & 0 & 0 \\ h & 4h & h & 0 & 0 & 0 \\ 0 & \frac{1}{2}h & 3h & h & 0 & 0 \\ 0 & 0 & \frac{1}{2}h & 2h & \frac{1}{2}h & 0 \\ 0 & 0 & 0 & h & 3h & \frac{1}{2}h \\ 0 & 0 & 0 & 0 & h & 4h \end{pmatrix} \hat{E} = \hat{Z}$$

mit

$$\left[\hat{Z}\right]_i = \frac{-1}{24} f^{(4)}(\xi_i) 2h^4$$

für $i = 1, 2, 6$, sowie

$$\begin{aligned} \left[\hat{Z}\right]_3 &= \frac{-1}{24} f^{(4)}(\xi_i) \left[\frac{h}{2}h^3 + h\left(\frac{h}{2}\right)^3\right] = \frac{-1}{24} f^{(4)}(\xi_i) \frac{5}{8} h^4, \\ \left[\hat{Z}\right]_4 &= \frac{-1}{24} f^{(4)}(\xi_i) 2\left(\frac{h}{2}\right)^4 = \frac{-1}{24} f^{(4)}(\xi_i) \frac{1}{8} h^4, \\ \left[\hat{Z}\right]_5 &= \frac{-1}{24} f^{(4)}(\xi_i) \left[h\left(\frac{h}{2}\right)^3 + \frac{h}{2}h^3\right] = \frac{-1}{24} f^{(4)}(\xi_i) \frac{5}{8} h^4. \end{aligned}$$

Dabei ist immer $\xi_i \in [x_{i-1}, x_{i+1}]$.

Löst man nun dieses Gleichungssystem, so ergeben sich für $\hat{E}$ folgende Werte:

$$\begin{aligned} \hat{E}_1 &= \frac{-5359}{313632}\hat{f}h^3, \quad \hat{E}_2 = \frac{-1175}{78408}\hat{f}h^3, \quad \hat{E}_3 = \frac{-659}{104544}\hat{f}h^3, \\ \hat{E}_4 &= \frac{-227}{627264}\hat{f}h^3, \quad \hat{E}_5 = \frac{-109}{19602}\hat{f}h^3, \quad \hat{E}_6 = \frac{-3049}{156816}\hat{f}h^3 \end{aligned}$$

Aus diesem Ergebnis lassen sich zwei Dinge ablesen. Zum einen ist deutlich, dass der Unterschied zwischen s' und f' an den Randpunkten des verfeinerten Intervalls abnimmt und am neuen Knotenpunkt in der Mitte des ursprünglichen Intervalls wesentlich kleiner ist als an den Randpunkten. Zum anderen ist schon bei diesem sehr einfachen äquidistanten Gitter die Verbesserung offensichtlich nicht allgemein auszudrücken. Noch deutlicher wird diese Tatsache, wenn man die Ergebnisse für $\tilde{E}$ auf einem Gitter mit sieben Knoten und unterschiedlichen Intervalllängen h_i betrachtet. Diese Werte sind in Abschnitt A.1 im Anhang der Arbeit angefügt.

Bedenkt man, dass in der Praxis selbst im Anfangsgitter von wesentlich mehr als sieben Knoten ausgegangen werden kann, so ist klar zu erkennen, dass man sich bei einer allgemeinen Aussage über die Fehlerveränderung auf die ersten beiden direkt aus dem Beweis abzulesenden Änderungen beschränken muss.

Zusammengefasst lässt sich aus diesen Resultaten und Satz 2.12 folgern:

Folgerung 5.7 *Durch Verfeinerung eines Intervalles mit der Zellenlänge h aus einem Gitter Γ mit maximaler Intervalllänge h_{max} in zwei Teilintervalle mit den Intervalllängen $\tilde{h} = \frac{1}{2}h$ reduziert sich die obere Schranke für den Abstand zwischen f und s auf dem betrachteten Intervall von*

$$\begin{aligned} \|s - f\| &\leq \frac{1}{384}\|f^{(4)}\|h^4 + \frac{1}{96}\|f^{(4)}\|h_{max}^3 h \\ &= \frac{1}{384}\|f^{(4)}\|h(h^3 + 4h_{max}^3) \end{aligned}$$

auf

$$\begin{aligned} \|s - f\| &\leq \frac{1}{384}\|f^{(4)}\|\tilde{h}^4 + \frac{1}{96}\|f^{(4)}\|h_{max}^3\tilde{h} \\ &= \frac{1}{384}\|f^{(4)}\|\frac{1}{16}h^4 + \frac{1}{96}\|f^{(4)}\|h_{max}^3\frac{1}{2}h \\ &= \frac{1}{384}\|f^{(4)}\|h(\frac{1}{16}h^3 + 2h_{max}^3). \end{aligned}$$

Anders ausgedrückt: Eine Halbierung der Intervalllänge hat mindestens eine Verkleinerung der Schranke für den Interpolationsfehler mit dem Faktor $\frac{1}{16} \cdot \frac{h^3+32h_{max}^3}{h^3+4h_{max}^3}$ zur Folge.
Sei nun o.B.d.A. $h_{max} = 0.4$. Dann hat eine Halbierung des Intervalls der Länge h folgende Auswirkung auf die Fehlerschranke:

h	Veränderung der Schranke	Verhältnis
0.4	0.4125	1.28
0.2	0.4867	1.04
0.1	0.4983	1.005
0.05	0.4998	1.0006

Tabelle 5.1: Fehlerschrankenänderung bei verschiedenen Intervalllängen

Die Veränderung der Schranke ist der Faktor $\frac{1}{16} \cdot \frac{h^3+32h_{max}^3}{h^3+4h_{max}^3}$, mit dem sich die Fehlerschranke reduziert. Die letzte Spalte zeigt das Verhältnis zwischen der Fehlerschrankenänderung und der Änderung der Intervalllänge mit dem Faktor $\frac{1}{2}$ in der Form

$$Intervalll\ddot{a}ngen\ddot{a}nderung^{Verh\ddot{a}ltnis} = Fehlerschranken\ddot{a}nderung. \qquad (5.8)$$

Bei im Verhältnis zu h_{max} kleinem h ist demnach nur eine lineare Auswirkung der Verfeinerung auf die Fehlerschranke festzustellen.

Der größte Teil der Reduzierung lässt sich durch die oben gezeigte komplizierte Form der Veränderung beim Abstand der Ableitungen von f und s an den Knotenpunkten leider nicht allgemein ausdrücken. Tatsächlich ist die Reduzierung der oberen Schranke aber deutlich größer.
Aus diesem Grund stellt das nächste Kapitel die tatsächliche Änderung des Interpolationsfehlers bei Gitterverfeinerung an einem einfachen numerischen Beispiel dar.

5.5 Praktische Verbesserung des Ergebnisses durch Anwendung von adaptiven Gittern anhand eines einfachen Beispiels

Zur Illustration der Veränderung des Interpolationsfehlers durch Gitteradaption wird hier die sogenannte Runge-Funktion f_R (bis auf einen Streckungsfaktor) verwendet. Diese vergleichsweise einfache Funktion wird häufig für Tests von numerischen Interpolationsverfahren angewendet. Sei f_R definiert durch

$$f_R(x) = \frac{1}{1+x^2}\ , \quad x \in [-4, 4].$$

Man betrachte das grobe Ausgangsgitter mit den Knoten
$\{x_0, x_1, ..., x_{20}\} = \{-4, -3.6, ..., 4\}$. Aus diesen Daten ergibt sich für den maximalen Interpolationsfehler, berechnet auf 800 Testknoten, der Wert 0.0017 im Intervall $[-0.8, -0.4]$. Auf Grund der Symmetrie von f_R natürlich auch entsprechend im Intervall $[0.4, 0.8]$. O.B.d.A werde aber nur die Region $[-4, 0]$ betrachtet. Nun wird das Intervall mit dem größten Interpolationsfehler verfeinert. Es entsteht also ein neuer Knoten bei $x = -0.6$. Auf den beiden neuen kleineren Intervallen beträgt der maximale Interpolationsfehler nun $1.994 \cdot 10^{-4}$ im Intervall $[-0.6, -0.4]$. Dieses wird nun weiter verfeinert, so dass ein neuer Knoten bei $x = -0.5$ entsteht. Die Analyse der beiden neuen kleinsten Intervalle zeigt einen maximalen Interpolationsfehler von $3.988 \cdot 10^{-5}$ im Intervall $[-0.5, -0.4]$. Verfährt man nach diesem Muster weiter, ergeben sich die Resultate aus Tabelle 5.2.
Die Fehleränderung ist dabei der Faktor, mit dem sich der maximale Fehler auf den durch die letzte Verfeinerung entstandenen Intervallen durch diese Verfeinerung verkleinert hat, also

$$\frac{max.\ Fehler\ auf\ kleinsten\ Intervallen}{max.\ Fehler\ auf\ diesen\ Intervallen\ vor\ Verfeinerung} = Fehleränderung.$$

Das Verhältnis gibt analog zu (5.8) das Verhältnis der Fehleränderung zur Änderung der Intervalllänge mit dem Faktor $\frac{1}{2}$ an.

Intervalllänge vor Verfeinerung	max. Fehler auf kleinsten Intervallen	Fehleränderung	Verhältnis
0.4	$1.99 \cdot 10^{-4}$	0.117	3.0912
0.2	$3.99 \cdot 10^{-5}$	0.200	2.3225
0.1	$1.00 \cdot 10^{-5}$	0.251	1.9957
0.05	$2.59 \cdot 10^{-6}$	0.259	1.9490

Tabelle 5.2: Vergleich Intervalllängenänderung - Fehleränderung pro Verfeinerung

Im Vergleich zu Tabelle 5.1 fällt auf, dass durch Verfeinerung auch bei kleiner Intervalllänge noch eine relativ große (im Vergleich zur Intervalllängenänderung annähernd quadratische) Reduzierung des Fehlers erzielt wird.

Interessant ist auch das Verhältnis der Fehleränderung zur Intervalllängenänderung im Vergleich zum Ausgangsgitter. Hier erhält man diese Ergebnisse:

min. Intervalllänge	max. Fehler auf kleinsten Intervallen	Intervalllängenänderung	Fehleränderung	Verhältnis
0.4	0.0017			
0.2	$1.99 \cdot 10^{-4}$	0.5	0.117	3.0912
0.1	$3.99 \cdot 10^{-5}$	0.25	0.023	2.7069
0.05	$1.00 \cdot 10^{-5}$	0.125	0.007	2.4698
0.025	$2.59 \cdot 10^{-6}$	0.0625	0.002	2.3396

Tabelle 5.3: Vergleich Intervalllängenänderung - Fehleränderung gesamt

Die Intervalllängenänderung ist dabei die Größe der beiden kleinsten (also der betrachteten) Intervalle im Vergleich zur Ausgangsgröße 0.4. Also

$$\frac{aktuelle\ minimale\ Intervalllänge}{Ausgangsintervalllänge\ (= 0.4)} = Intervalllängenänderung.$$

Die Fehleränderung ist entsprechend der Faktor, mit dem sich der Fehler im Vergleich zum Ausgangsgitter verkleinert hat, also

$$\frac{max.\ Fehler\ auf\ kleinsten\ Intervallen}{max.\ Fehler\ auf\ Ausgangsintervall\ (= 0.0017)} = Fehleränderung.$$

Das Verhältnis ist wiederum das logarithmische Verhältnis der beiden Größen gemäß (5.8).

Der Vergleich der beiden Änderungen ist in Abbildung 5.5 als Diagramm dargestellt.

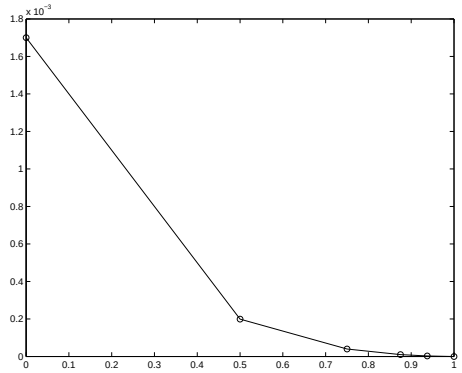


Abbildung 5.5: Vergleich Intervalllängenänderung - Fehleränderung

Dabei zeigt die x-Achse die Reduzierung der Intervalllänge
(1 - Intervalllängenänderung) und die y-Achse den Fehler auf dem betrachteten Intervall.

Natürlich sind diese Zahlen abhängig vom entsprechenden Problem, dem jeweiligen Intervall, sowie der Vorgehensweise beim Verfeinern. Bei diesem Beispiel wird nur das jeweils kleinste Intervall wieder verfeinert. Die zusätzliche Verfeinerung anderer Intervalle in der Umgebung führt selbstverständlich zu anderen Werten.

Die Tabellen 5.2 und 5.3 spiegeln aber wider, dass in der Praxis eine Gitterverfeinerung eine große Auswirkung auf den Fehler im verfeinerten Intervall haben kann bzw. hat.
Dieses kleine Rechenbeispiel verdeutlicht, dass Gitteradaption bei der Spline-Interpolation durchaus geeignet ist, um den Interpolationsfehler in kritischen Regionen entscheidend zu verkleinern.

Kapitel 6

Diskussion der praktischen Ergebnisse

Nach der theoretischen Betrachtung des Lösungsalgorithmus' und der verwendeten Spline-Interpolation werden nun die Ergebnisse bei der Lösung eindimensionaler optimaler Steuerungsprobleme diskutiert. Zunächst wird der Algorithmus im Detail vorgestellt sowie einige Hinweise zur Implementierung gegeben. Es folgt die Anwendung auf vier verschiedene Probleme aus unterschiedlichen Bereichen der Ökonomie und die Untersuchung der Brauchbarkeit der Ergebnisse. Als Grundlage dieser Aussage dienen die Ergebnisse, die durch Anwendung der Variante mit linearer Interpolation (mit dem in Kapitel 4.5.2 beschriebenen kontrollierten Gauß-Seidel-Verfahren, kurz GSV) erzielt worden sind. Wie schon die Spline-Theorie vermuten lässt, spielt das Differentiationsverhalten der exakten optimalen Wertefunktion eine große Rolle. So begründet sich die Unterteilung in Probleme mit differenzierbarer und Probleme mit nicht global differenzierbarer optimaler Wertefunktion. Besonderes Augenmerk richtet sich im zweiten Fall auf das Verhalten der errechneten Approximation der optimalen Wertefunktion in der Umgebung der nicht differenzierbaren Stellen.

6.1 Aufbau des Algorithmus'

Die Herleitung und die theoretische Fundierung des Algorithmus' ist bereits in Kapitel (4) dargestellt worden. Es folgt eine Aufstellung der exakten Vorgehensweise bei der Implementierung.
Der Algorithmus besteht aus zwei großen Teilen. Zum einen die Bestimmung der Lösung auf dem aktuellen Gitter und zum anderen die Gitteradaption.

Zunächst sind vom Benutzer die Daten des Problems sowie einige Parameter festzulegen. Zur Erinnerung (siehe auch (3.7)), das Problem ist in der Regel folgendermaßen gegeben:

(i) in diskreter Form $t \in \mathbb{N}_0$:

$$v_h(x) = \max_{u_h \in \mathcal{U}_h} \left\{ h \sum_{j=0}^{\infty} (1-\delta h)^j g(\Phi_h(jh, x, u_h), u_h(jh)) \right\}, \tag{6.1}$$

wobei

$$\Phi_h(0, x, u_h) = x \quad , \Phi_h((j+1)h, x, u_h) = f_h(\Phi_h(jh, x, u_h), u_h(jh)), \tag{6.2}$$

oder

(ii) in kontinuierlicher Form $t \in \mathbb{R}_0^+$:

$$v(x) = \max_{u \in \mathcal{U}} \left\{ \int_0^{\infty} e^{-\delta t} g(\Phi(t, x, u), u(t)) dt \right\}, \tag{6.3}$$

wobei

$$\Phi(0, x, u) = x \quad , \quad \frac{d\Phi(t, x, u)}{dt} = f(\Phi(t, x, u), u(t)). \tag{6.4}$$

Im zweiten Fall wird das Problem (wie in Kapitel 4.1 beschrieben) durch das zeitdiskretisierte Problem

$$\tilde{v}_h(x) = \max_{u \in \mathcal{U}} \{ h \sum_{j=0}^{\infty} (1-\delta h)^j g(\tilde{\Phi}_h(jh, x, u_h), u_h(jh)) \}$$

mit Zeitschritt h und

$$\begin{aligned} \tilde{\Phi}_h(0, x, u_h) &= x, \\ \tilde{\Phi}_h((j+1)h, x, u_h) &= f_h(\tilde{\Phi}_h(jh, x, u_h), u_h(jh)) \\ &= \tilde{\Phi}_h(jh, x, u_h) + hf(\tilde{\Phi}(jh, x, u_h), u_h(jh)) \end{aligned}$$

ersetzt.

Es müssen demnach die Kosten- oder Ertragsfunktion g sowie die die Trajektorie Φ_h bzw. $\tilde{\Phi}_h$ bestimmende Funktion f_h eingegeben werden, durch die das optimale Steuerungsproblem beschrieben ist. Ebenfalls festzulegen sind die Knoten x_i^0, $i = 1, ..., n$, des Ausgangsgitters Γ^0 im Vektor x sowie die Kontrollwerte u_k im Vektor u. Durch den ersten und letzten Eintrag von x ist damit auch der Lösungsbereich Ω festgelegt. Der Vektor v mit den Werten v_i^0 der optimalen Wertefunktion an den Knoten x_i^0 wird mit $v_i^0 = 0$ für $i = 1, ..., n$ vorbelegt. Um bei einer einheitlichen Bezeichnung für Vektoren im Algorithmus zu bleiben, wird der Vektor v mit einem Kleinbuchstaben bezeichnet und nicht mit V wie in den Kapiteln 4 und 5. Der zu v gehörige Spline aus $\mathcal{S}_\Delta$ wird wie vorher auch $\check{v}_h$ genannt.

Festzusetzende Parameter sind

Bezeichnung	Funktion
h	Schrittweite
δ	Diskontrate
θ	Parameter zur Bestimmung der Intervalle, die verfeinert werden
ζ	Abbruchparameter; gewünschte Genauigkeit der Approximation

Tabelle 6.1: Parameter

Der Algorithmus läuft in der Iteration $j+1$ folgendermaßen ab:

1. Berechne zu jedem Knoten $x_i^j \in x$ und jedem Kontrollwert $u_k \in u$ die Werte $g(x_i^j, u_k)$ und $f_h(x_i^j, u_k)$.

2. Bestimme den zu den Werten v_i^j an den Knoten x_i^j gehörigen Spline $\check{v}_h^j$.

3. Bestimme für jedes x_i^j den neuen Wert v_i^{j+1} gemäß der Iterationsvorschrift (4.11) durch Maximierung über alle u_k mit $f_h(x_i^j, u_k) \in \Omega$.

4. Bestimme die Differenz $\zeta_i = v_i^{j+1} - v_i^j$ für alle i

5. Wenn $\max_i\{|\zeta_i|\} < \zeta$ gehe zu 6. Die Lösung auf dem Gitter Γ^j mit den Knoten x_i^j ist durch $\check{v}_h^{j+1}$ nach (4.14) genau genug approximiert. Es folgt eine Gitteradaption.

   Ansonsten setze $j = j+1$ und gehe zu 2. Das Gitter bleibt in der nächsten Iteration gleich.

   Gitteradaption:

6. Bestimme den zu den Werten v_i^{j+1} an den Knoten x_i^j gehörigen Spline v_h^{j+1}.

7. Bestimme Vektor xx^j mit den Testpunkten xx_l^j in der Mitte der Intervalle $[x_l^j, x_{l+1}^j]$ sowie den zugehörigen Wert vv_l^j durch Auswertung von $\check{v}_h^{j+1}$ an der Stelle xx_l^j.

8. Bestimme für alle xx_l^j und alle u_k mit $f_h(xx_l^j, u_k) \in \Omega$ den Wert vv_l^{j+1} gemäß der Iterationsvorschrift (4.11).

9. Bestimme die Differenz $\eta_l = vv_l^{j+1} - vv_l^j$ für alle l.

10. Ist $\delta h \cdot \max_l \eta_l < \zeta$, breche ab. Die approximierte Lösung ist auf ganz Ω genau genug.

11. Mache alle xx_l^j mit $\eta_l > \theta \cdot \max_l \eta_l$ zu neuen Knoten. Dazu werden die Werte xx_l^j so in x eingefügt, dass alle Einträge der Größe nach sortiert bleiben. Die Werte vv_l^j werden neue Einträge in v an den entsprechenden Stellen. Der neue Vektor x enthält somit die Knoten x_i^{j+1}, $i = 1, ..., n$, und der neue Vektor v die enstprechenden Werte v_i^{j+1}, $i = 1, ..., n$, an den Knoten. Der Index n hat sich um die Anzahl der neuen Knoten vergrößert.

12. Setze $j = j + 1$. Gehe zu 1. Nun wird die Approximation auf dem neuen Gitter berechnet.

Zur Implementierung dieses Algorithmus' sind einige Anmerkungen zu machen:

Bemerkung 6.1 *(i) Der Algorithmus wird hier in MATLAB Version 6.0 Release 12 implementiert. So können die MATLAB-eigene Routine „spline" zur Errechnung der Splines verwendet werden. Die Auswertung des Splines an Nicht-Knotenpunkten erfolgt durch die Routine „ppval". Dafür müssen im Vergleich zu Implementierungen beispielsweise in C starke Einbußen bei der Rechenzeit gemacht werden.*
Die im Folgenden dargestellten Berechnungen sind auf einem 1700MHz Prozessor mit 512MB RAM unter Windows ME durchgeführt worden.

(ii) Je mehr Kontrollwerte $u \in U$ verwendet werden, desto besser ist das Ergebnis. Abhängig vom betrachteten Problem wird eine gewisse Mindestanzahl an Kontrollwerten benötigt. Ab etwa 50 Kontrollwerten lassen sich bei den betrachteten Problemen brauchbare Ergebnisse erzielen.

(iii) Die Wahl des Ausgangsgitters hat ebenfalls Einfluss auf das Ergebnis; bei Spline-Interpolation noch mehr als bei linearer Interpolation. Genaue Anmerkungen dazu folgen bei der Betrachtung der Beispiele im nächsten Abschnitt.

(iv) Zusätzlich kann noch ein weiteres Abbruchkriterium herangezogen werden. Beim Überschreiten einer vorgegebenen Anzahl an Gitterknoten wird kein neues Gitter mehr gebildet, sondern der Algorithmus beendet. Diese Vorgehensweise wird auch bei den Beispielrechnungen angewendet.

(v) Für den Abbruchparameter ζ ist - wenn nicht anders angegeben - der Wert 10^{-5} festgesetzt.

(vi) Der MATLAB-Quelltext befindet sich in Abschnitt B.1 im Anhang.

Die folgenden Abschnitte zeigen die Ergebnisse, die durch Anwendung des oben beschriebenen Algorithmus' auf verschiedene optimale Steuerungsprobleme erzielt worden sind. Die Auswahl der Beispiele stammt wie bereits erwähnt aus Grüne und Semmler [13].

Zunächst ein Problem, dessen exakte optimale Wertefunktion bekannt ist. An diesem Beispiel lassen sich somit die exakten Fehler bestimmen. Hier wird auch ausführlich auf den Unterschied zwischen den Implementierungen mit und ohne Gitteradaption eingegangen.

6.2 Beispiel mit differenzierbarer Optimalwertfunktion

6.2.1 Beispiel 1: Wachstumsmodell

Das folgende zeitdiskrete Wachstumsmodell, nach seinen „Entdeckern“ auch Brock-Mirman-Modell genannt, wird häufig als Testproblem für numerische Algorithmen verwendet.

Das Problem der Form (6.3)-(6.4) ist gegeben durch die Ertragsfunktion

$$g(x, u) = \ln u$$

und die Dynamik

$$x(t+1) = Ax^{\alpha} - u(t)$$

mit Konstanten $A > 0$ und $0 < \alpha < 1$.

Die exakte Lösung ist bestimmt durch

$$V(x) = B + C \ln x, \tag{6.5}$$

wobei

$$C = \frac{\alpha}{1 - \alpha\beta} \quad \text{und} \quad B = \frac{\ln((1 - \alpha\beta)A) + \frac{\beta\alpha}{1-\beta\alpha} \ln(\alpha\beta A)}{1 - \beta}$$

ist (vgl. Santos, Vigo-Aguiar [20]).

Die Berechnungen wurden durchgeführt mit $A = 5$, $\alpha = 0.34$ und Diskontrate $\delta = 0.05$. Da das Problem bereits in diskreter Zeit vorliegt, ist die Schrittweite $h = 1$. Der Diskontfaktor beträgt folglich $\beta = 1 - \delta = 0.95$.
Der betrachtete Bereich ist $\Omega = [0.1, 10]$ mit einer Diskretisierung von
$U = [0.1, 11]$ mit 500 Kontrollwerten.

Abbildung 6.1 zeigt die exakte optimale Wertefunktion auf Ω.

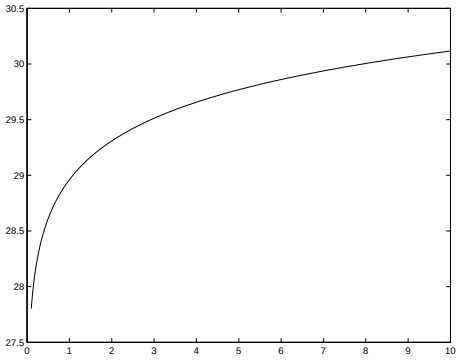


Abbildung 6.1: Optimale Wertefunktion zu Beispiel 1

Zunächst wird das Problem auf äquidistanten Gittern gelöst. Tabelle 6.2 beinhaltet die Differenz in der ∞ -Norm zwischen der exakten Lösung und der numerischen Approximation in Abhängigkeit von der Knotenzahl, also den maximalen Fehler der jeweiligen Approximation. Die zweite Spalte zeigt den Fehler bei Lösung mit linearer Interpolation, die dritte Spalte entsprechend mit Spline-Interpolation.

Der Fehler wird hier bstimmt durch den Vergleich der exakten optimalen Wertefunktion (vgl. (6.5)) mit der Auswertung der jeweils erhaltenen Approximation an 1000 Testpunkten.

Knotenzahl	Fehler mit linearer Interpolation	Fehler mit Spline-Interpolation	Rechenzeit (mit Spline-Int.)
17	$2.50 \cdot 10^{-1}$	$9.99 \cdot 10^{-2}$	$306s$
50	$7.99 \cdot 10^{-2}$	$2.53 \cdot 10^{-2}$	$940s$
100	$3.17 \cdot 10^{-2}$	$6.47 \cdot 10^{-3}$	$1993s$
130	$2.12 \cdot 10^{-2}$	$3.54 \cdot 10^{-3}$	$2536s$
150	$1.72 \cdot 10^{-2}$	$2.65 \cdot 10^{-3}$	$3125s$
200	$1.06 \cdot 10^{-2}$	$1.32 \cdot 10^{-3}$	$4236s$

Tabelle 6.2: Fehler der numerischen Lösung auf äquidistanten Gittern

Es zeigt sich also, dass bei der Lösung auf äquidistanten Gittern die Variante mit Spline-Interpolation genauere Ergebnisse liefert als diejenige mit linearer Interpolation. Zu den Rechenzeiten muss erwähnt werden, dass bei den hier verwendeten Implementierungen nicht primär auf Minimierung der Rechendauer geachtet worden ist. Es wird allerdings deutlich, dass die MATLAB-Berechnungen sehr zeitaufwendig sind. Die Rechenzeit kann bei diesem Beispiel durch die in Ab-

schnitt 4.5.1 beschrieben Verknüpfung mit dem Verfahren der Strategie-Iteration erheblich verkürzt werden. Einzelheiten dazu folgen bei der nun betrachteten Berechnung auf adaptiven Gittern.

Bei der Lösung mit Gitteradaption wird der Verfeinerungsparameter auf $\theta = 0.1$ gesetzt. Das Anfangsgitter besteht aus 17 Knoten und als zusätzliches Abbruchkriterium wird hier die Grenze von 300 Gitterknoten gewählt. Es ergeben sich mit Spline-Interpolation die Ergebnisse in Tabelle 6.3.

Knotenzahl	Fehler	Fehlerschranke	Rechenzeit
17	$9.99 \cdot 10^{-2}$	1.91	303s
19	$4.82 \cdot 10^{-2}$	0.74	439s
21	$1.57 \cdot 10^{-2}$	0.25	515s
23	$3.60 \cdot 10^{-3}$	$5.79 \cdot 10^{-2}$	538s
26	$6.43 \cdot 10^{-4}$	$9.15 \cdot 10^{-3}$	614s
33	$2.94 \cdot 10^{-4}$	$1.54 \cdot 10^{-3}$	655s
57	$1.73 \cdot 10^{-4}$	$6.70 \cdot 10^{-4}$	688s
109	$1.11 \cdot 10^{-4}$	$1.07 \cdot 10^{-3}$	746s
202	$7.79 \cdot 10^{-5}$	$1.16 \cdot 10^{-3}$	860s
279	$5.90 \cdot 10^{-5}$	$1.26 \cdot 10^{-3}$	1041s
345	$5.19 \cdot 10^{-5}$	$7.84 \cdot 10^{-4}$	1186s

Tabelle 6.3: Fehler der numerischen Lösung mit Gitteradaption

Dabei wird der (maximale) Fehler wie in Tabelle 6.2 berechnet, und die Fehlerschranke ist die obere Schranke, die durch die lokalen Fehlerschätzer η_i gemäß (5.3) bestimmt wird, also $\frac{1}{\delta h} \max_i \eta_i$. Durch diesen Wert ist nach Satz 5.4 der Abstand zwischen der auf dem jeweiligen Gitter errechneten Approximation und der exakten optimalen Wertefunktion nach oben beschränkt. In diesem Fall durch $20 \cdot \max_i \eta_i$. Die Werte η_i werden wie in Kapitel 5.2 bestimmt. Die vierte Spalte ist die gesamte bis zu der Approximation auf dem entsprechenden Gitter benötigte Rechenzeit (inklusive der Gitteradaptionen und Fehlerbestimmmungen).

Im Vergleich dazu erhält man mit dem kontrollierten Gauß-Seidel-Verfahren die Testergebnisse in Tabelle 6.4.

Knotenzahl	Fehler	Fehlerschranke
17	$2.50 \cdot 10^{-1}$	4.19
18	$1.41 \cdot 10^{-1}$	1.92
22	$6.84 \cdot 10^{-2}$	0.95
27	$3.24 \cdot 10^{-2}$	0.40
38	$1.72 \cdot 10^{-2}$	0.11
63	$4.21 \cdot 10^{-3}$	$3.38 \cdot 10^{-2}$
116	$1.48 \cdot 10^{-3}$	$1.26 \cdot 10^{-2}$
203	$3.73 \cdot 10^{-4}$	$3.92 \cdot 10^{-3}$
352	$2.13 \cdot 10^{-4}$	$1.34 \cdot 10^{-3}$

Tabelle 6.4: Fehler der numerischen Lösung mit GSV und Gitteradaption

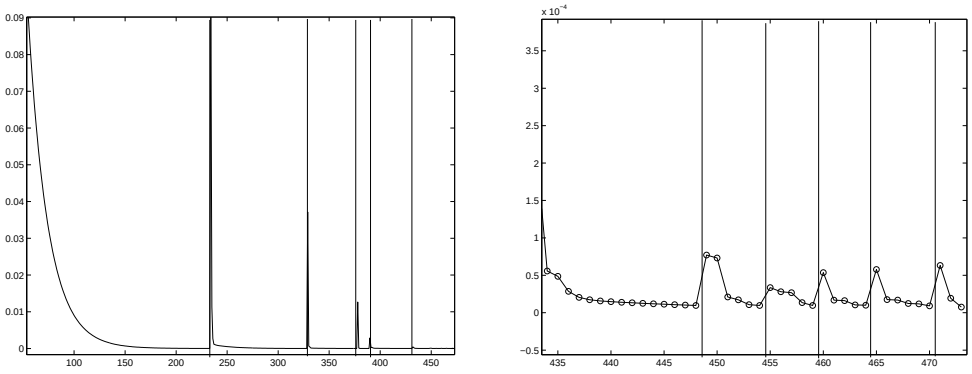
Es wird deutlich, dass durch den beschriebenen Algorithmus mit
Spline-Interpolation hier gute Ergebnisse erzielt werden. Sie erweist sich bei diesem Beispiel als eine geeignete Alternative zu linearen Interpolationsmethoden. Die mit der Iterationsvorschrift (4.11) und Spline-Interpolation ermittelten Splines $\check{v}_h^j$ konvergieren in diesem Beispiel gegen ein $\check{v}_h$, oder besser gesagt, die diese Splines bestimmenden Vektoren v^j konvergieren (monoton) gegen einen Vektor v. $\check{v}_h$ approximiert die exakte optimale Wertefunktion. Die Konvergenz folgt aus einer Kontraktion der Form

$$\|\check{v}_h^{j+1} - \check{v}_h\|_\infty < \|\check{v}_h^j - \check{v}_h\|_\infty$$

Die Gültigkeit dieser Kontraktion lässt sich aus den Werten für $\max_i\{|\zeta_i|\}$ im Laufe der Berechnungen ablesen (siehe Abbildung 6.2). Der Wert $\max_i\{|\zeta_i|\}$ nach der Berechnung von v^j in Iteration j sei von nun an mit ζ_{max}^j bezeichnet.

Die x-Achse zeigt die Anzahl der Iterationen, die y-Achse den Wert ζ_{max}^j nach der entsprechenden Iteration. In Abbildung 6.2 a) auf einem größeren Ausschnitt und lediglich als durchgezogene Graph; in Abbildung 6.2 b) nur die letzten Iterationen und mit zusätzlicher Markierung der Werte. Die senkrechten Striche bei etwa $x = 240$, 330, 380, usw. symbolisieren die nach dieser Iteration stattfindende Gitteradaption. Der in dieser Iteration berechnete Wert ζ_{max}^j war also kleiner als das Abbruchkriterium $\zeta = 1 \cdot 10^{-5}$. Direkt nach der Bildung eines neuen Gitters sind die „Sprünge“ zwischen den Funktionen $\check{v}_h^j$ und $\check{v}_h^{j+1}$ und damit der Wert ζ_{max}^j vergleichsweise groß und verringern sich anschließend wieder.

Zur Erinnerung: Der Wert ζ_i beschreibt den Abstand zwischen der aktuellen Approximation der optimalen Wertefunktion und derjenigen aus der letzten Iteration am Knoten x_i (im Algorithmus in der j-ten Iteration berechnet durch die Differenz der i-ten Einträge der Vektoren v^j und v^{j-1} , also $\zeta_i = v_i^j - v_i^{j-1}$). Durch das Betrags-Maximum ζ_{max}^j dieser Differenzen über alle Knoten x_i ist nach (4.14)



(a) Ausschnitt 1 (b) Ausschnitt 2

Abbildung 6.2: Entwicklung von ζ_{max}^j zu Beispiel 1

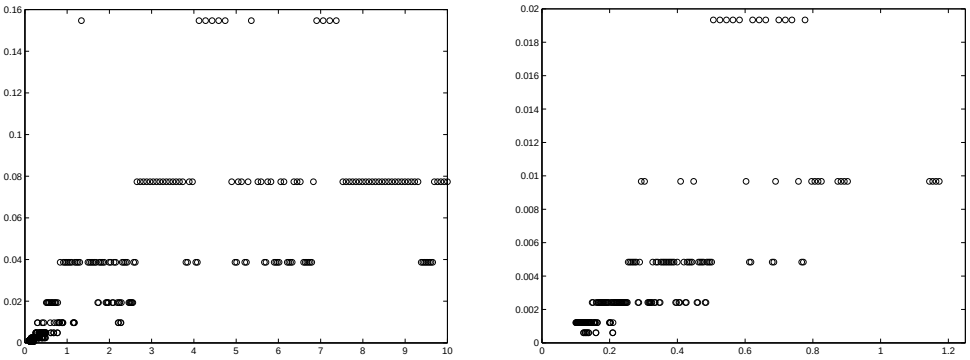
in Lemma 4.8 eben der Abstand der Approximation $\check{v}_h^j$ in der Iteration j von der Funktion $\check{v}_h$, gegen die die $\check{v}_h^j$ für $j \to \infty$ konvergieren, beschränkt.

In den bei diesen Berechnungen angewendeten Implementierungen zeigt sich außerdem, dass durch Anwendung der Spline-Interpolation Speicherplatz gespart werden kann. An den Tabellen 6.3 und 6.4 sieht man, dass eine gewünschte Genauigkeit schon auf einem Gitter mit weniger Knoten erzielt wird. So wird der maximale Fehler von $2.22 \cdot 10^{-4}$ auf dem Endgitter des kontrollierten Gauß-Seidel-Verfahrens mit 352 Knoten bei Anwendung der Spline-Interpolation schon auf einem Gitter mit 57 Knoten deutlich erreicht ($1.73 \cdot 10^{-4}$).

Zusätzlich verdeutlicht der Vergleich der Tabellen 6.2 und 6.3, dass die Anwendung von adaptiver Gittererzeugung bei der Spline-Interpolation eine große Auswirkung auf den benötigten Speicherplatz hat. So wird hier beispielsweise der Wert des maximalen Fehlers bei der Lösung mit Spline-Interpolation auf einem äquidistanten Gitter mit 130 Knoten ($3.54 \cdot 10^{-3}$) bei der Lösung mit Gitteradaption bereits auf dem noch sehr groben Gitter mit 23 Knoten erreicht. Oder von der anderen Seite betrachtet: Auf dem äquidistanten Gitter mit 200 Knoten hat die errechnete Approximation den maximalen Fehler $1.32 \cdot 10^{-3}$, während bei der adaptiven Variante auf dem annähernd „gleich feinen“ Gitter mit 202 Knoten ein Wert von $7.79 \cdot 10^{-5}$ errechnet wird. Der Fehler wird also bei dieser Knotenanzahl um knapp 94 Prozent reduziert. Zusätzlich wird die Rechenzeit dabei um ca. 80 Prozent verringert. Die Zeit zur Bestimmung der Approximation auf einem Gitter mit (etwa) 200 Knoten reduziert sich von 4236s (auf dem festen Gitter) auf 860s (mit Gitteradaption).

Diese Zahlen sind nur Beispielwerte und variieren abhängig von den dem Vergleich zugrundeliegenden Kennzahlen. Vergleicht man die Genauigkeit beispielsweise bei anderen Knotenzahlen, ergeben sich andere Werte. Eine eklatante Einsparung ist jedoch bei Anwendung des Verfahrens der Gitteradaption in jedem Fall auszumachen.

Die Gitteradaptionen sind in Abbildung 6.3 sichtbar. Sie zeigt die Intervalllängen des letzten Gitters in Abhängigkeit von x; links auf ganz Ω, rechts den Ausschnitt der kleinen Intervalle. Die y-Achse zeigt die Länge des Intervalls, das bei x beginnt. Das bedeutet: Je kleiner der y-Wert ist, desto feiner ist an dieser Stelle das Gitter.

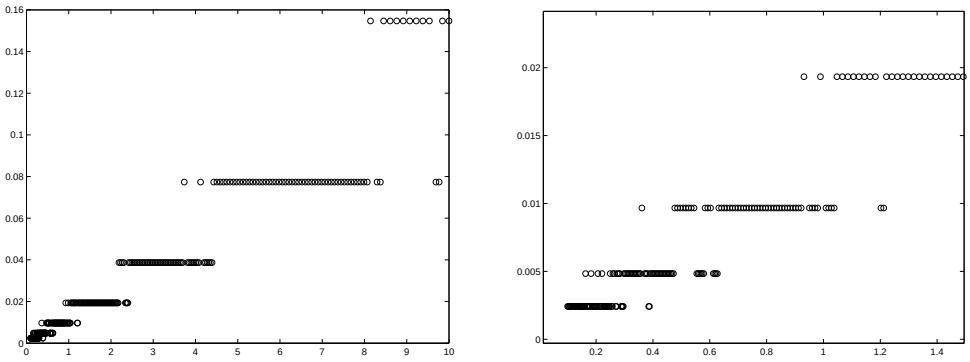


(a) auf ganz Ω (b) Ausschnitt

Abbildung 6.3: Intervalllängen des Endgitters zu Beispiel 1

Vergleicht man die Intervalllängen mit der optimalen Wertefunktion, kann man erkennen, dass eine größere Steigung ein feineres Gitter verursacht.

Dieser Zusammenhang ist bei der Lösung mit linearer Interpolation noch deutlicher (siehe Abbildung 6.4).



(a) auf ganz Ω (b) Ausschnitt

Abbildung 6.4: Intervalllängen des Endgitters zu Beispiel 1 mit GSV

Zum Abschluss der Betrachtungen zu Beispiel 1 folgt noch die Auswirkung bei Anwendung der in Abschnitt 4.17 beschriebenen Verbesserung des Verfahrens. Durch die Verknüpfung des beschriebenen Iterationsverfahrens mit Spline-Interpolation und der Idee der Strategie-Iteration, ist die Bestimmung der Approximation der Optimalwertfunktion wesentlich effizienter. Bei der hier angewendeten Implementierung ergeben sich die folgenden Fehlerwerte auf den entsprechenden Gittern:

Knotenzahl	Fehler	Fehlerschranke	Rechenzeit
17	$9.99 \cdot 10^{-2}$	1.91	16s
19	$4.82 \cdot 10^{-2}$	0.74	27s
21	$1.57 \cdot 10^{-2}$	0.25	36s
23	$3.61 \cdot 10^{-3}$	$5.79 \cdot 10^{-2}$	46s
26	$6.80 \cdot 10^{-4}$	$9.11 \cdot 10^{-3}$	59s
33	$2.81 \cdot 10^{-4}$	$1.53 \cdot 10^{-3}$	72s
55	$1.55 \cdot 10^{-4}$	$6.85 \cdot 10^{-4}$	93s
105	$8.95 \cdot 10^{-5}$	$1.04 \cdot 10^{-3}$	134s
188	$5.95 \cdot 10^{-5}$	$1.11 \cdot 10^{-3}$	214s
256	$5.06 \cdot 10^{-5}$	$1.56 \cdot 10^{-3}$	330s

Tabelle 6.5: Fehler der numerischen Lösung mit Spline-Strategie-Iteration

Die Genauigkeit auf feineren Gittern nimmt also noch einmal zu. Die größere Einsparung ist allerdings bei der Rechenzeit zu erkennen. Die Zeit um die Approximation auf dem Endgitter zu berechnen (die annähernd gleich genau ist)

reduziert sich von 1186s auf 330s. Das entspricht in diesem Fall einer Einsparung von etwa 72 Prozent.

Diese Beschleunigung der Berechnungen ist allerdings nur bei diesem Beispiel mit differenzierbarer Optimalwertfunktion anwendbar. Bei den nachfolgenden Beispielen schlägt sie fehl. Die Strategie-Iteration konvergiert in diesen Fällen nicht.
Die verwendete Implementierung 'verf_mit_splinestrategie.m' ist auf der beiliegenden CD zu finden.

6.3 Beispiele mit nicht global differenzierbarer Optimalwertfunktion

Bei den folgenden kontinuierlichen Beispielen ist die exakte optimale Wertefunktion nicht bekannt. Zu jedem Beispiel werden zunächst die Resultate vorgestellt, die unter Anwendung linearer Interpolationsmethoden errechnet worden sind. Im Vergleich dazu werden im Anschluss die jeweiligen Berechnungen dargelegt, bei denen Spline-Interpolation verwendet worden ist.

Ein weiteres Merkmal dieser Beispiele ist, dass die optimale Wertefunktion einen Knick besitzt, also an einer Stelle nicht differenzierbar ist. Wie man bei den einzelnen Beispielen sehen wird, führt dies zu Problemen bei der Anwendung der Spline-Interpolation.

Der folgende Abschnitt beinhaltet bei jeder Berechnung die adaptive Gittererzeugung. Aufgrund der langen Rechenzeiten werden die optimalen Werte mit lediglich 50 Kontrollwerten berechnet. Dadurch sind kleine Abstriche bei der Genauigkeit zu machen. Es ändert sich allerdings nichts am Verhalten der approximierten optimalen Wertefunktion.

6.3.1 Beispiel 2: Ökologie-Problem

Bei diesem Beispiel aus dem Bereich der Ökologie aus Brock und Starret [6] beschreibt die Ertragsfunktion $g(x, u)$ eine Nutzenfunktion mit zwei Komponenten. Auf der einen Seite der Nutzen einer Partei, beispielsweise eines Unternehmens, der durch die Leitung von Phosphor in einen See erzielt wird. Auf der anderen Seite der Schaden, den eine andere Partei, z.B. Badegäste, durch den Gehalt an Phosphor in dem See davonträgt. Die Menge des momentan in den See geleiteten Phosphors ist durch u dargestellt; die gesamte Menge an Phosphor in dem See durch x. Die Nutzenfunktion hat die Form

$$g(x, u) = au^{\sigma} - k\gamma x^2.$$

Die Zustandsfunktion zeigt die dynamische Entwicklung der Phosphormenge in dem See, gegeben durch

$$\frac{d}{dt}\Phi(t,x,u) = u(t) - \mu\Phi(t,x,u) + \frac{m\Phi(t,x,u)^\rho}{n^\rho + \Phi(t,x,u)^\rho}.$$

Durch die im Abschnitt4.1 beschriebene Eulerdiskretisierung erhält man

$$\begin{aligned}\tilde{\Phi}_h((j+1)h,x,u_h) &= \tilde{\Phi}_h(jh,x,u_h) + u_h(jh) - \mu\tilde{\Phi}_h(jh,x,u_h) \\ &\quad + \frac{m\tilde{\Phi}_h(jh,x,u_h)^\rho}{n^\rho + \tilde{\Phi}_h(jh,x,u_h)^\rho} \\ &= f_h(\tilde{\Phi}_h(jh,x,u_h), u_h(jh))\end{aligned}$$

Die Parameter werden besetzt mit

$$a = 2\ , k = \sigma = \gamma = 0.5\ , \mu = 0.55\ , m = n = 1\ , \rho = 2$$

Die Diskontrate beträgt $\delta = 0.1$. Der Zeitschritt für die notwendige zeitliche Diskretisierung ist $h = 0.05$ und die Kontrollwerte kommen aus $U = [0, 0.4]$. Berechnet wird die optimale Wertefunktion im Bereich $\Omega = [0, 3]$

Bei der Anwendung linearer Interpolation und einem Ausgangsgitter mit 50 Knoten ergibt sich auf Ω folgende optimale Wertefunktion:

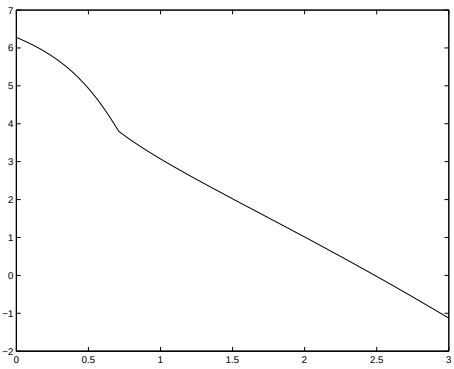


Abbildung 6.5: Optimale Wertefunktion zu Beispiel 2 mit GSV

Dabei beträgt die obere Fehlerschranke aus (5.3) $1.76 \cdot 10^{-2}$ auf einem Endgitter mit 225 Knoten.

Die Knotenzahl der Gitter sowie die entsprechende obere Fehlerschranke entwickelt sich im Laufe der Berechnungen wie folgt:

Knotenzahl	Fehlerschranke
50	$3.52 \cdot 10^{-1}$
71	$1.37 \cdot 10^{-1}$
117	$2.68 \cdot 10^{-1}$
128	$3.51 \cdot 10^{-1}$
129	$2.60 \cdot 10^{-1}$
131	$2.33 \cdot 10^{-1}$
134	$8.54 \cdot 10^{-2}$
220	$3.57 \cdot 10^{-1}$
221	$1.63 \cdot 10^{-1}$
223	$1.60 \cdot 10^{-1}$
225	$1.76 \cdot 10^{-2}$

Tabelle 6.6: Entwicklung der Fehlerschranke zu Beispiel 2 mit GSV

Bei genauer Betrachtung erkennt man einen Knick in der optimalen Wertefunktion bei etwa $x = 0.71$. Diese Art Schwellenpunkt (auch Skiba-Punkt) „trennt“ zwei Gleichgewichte bei $x = 0.396$ und bei $x = 1.861$. Rechts vom Skiba-Punkt konvergieren die Werte gegen das eine Gleichgewicht, links davon gegen das andere. Ausführungen zur Gleichgewichtstheorie sowie zur Thematik „Skiba-Punkte“ (siehe dazu Deissenberg u.a. [8]) sollen allerdings nicht Teil dieser Arbeit sein.

Deutlicher kommt der Skiba-Punkt bei der Betrachtung der optimalen Kontrollwerte, also derjenigen Werte u^* zu jedem $x \in \Omega$, die den Ausdruck auf der rechten Seite der Iterationsvorschrift (4.11) jeweils maximieren, zur Geltung. Hier ist in Abbildung 6.6 bei $x = 0.71$ ein Sprung zu erkennen.

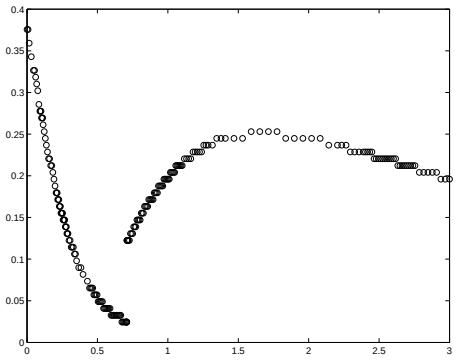
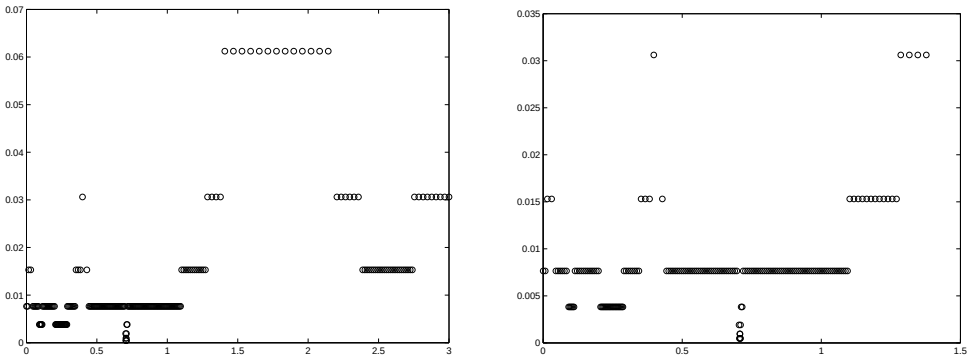


Abbildung 6.6: Optimale Kontrollwerte zu Beispiel 2 mit GSV

Ebenfalls sehr gut sichtbar ist der Skiba-Punkt, wenn man die Intervalllängen des Endgitters betrachtet. Hier fällt auf, dass das Gitter um den Skiba-Punkt herum wesentlich feiner ist als im Rest von Ω.



(a) auf ganz Ω (b) Ausschnitt

Abbildung 6.7: Intervalllängen des Endgitters zu Beispiel 2 mit GSV

Dies ist die Folge davon, dass der Knick in der optimalen Wertefunktion zu einem großen lokalen Fehler in dieser Region führt. Dies ist sehr gut an Abbildung 6.8 sichtbar. Sie zeigt den Abstand zwischen der auf einem Gitter mit 128 Knoten approximierten optimalen Wertefunktion an den Testpunkten xx_l (angetragen auf der x-Achse) und dem Wert an der selben Stelle nach einer Iteration gemäß (4.11). Durch diesen Abstand lässt sich nach Kapitel 5.2 der lokale Fehler zwischen der aktuellen Approximation und der exakten optimalen Wertefunktion abschätzen.

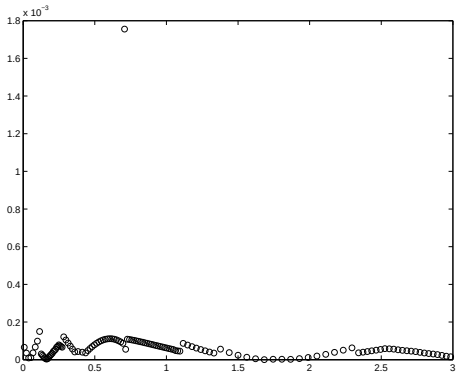


Abbildung 6.8: Beispiel zur Fehlerschätzung

Entsprechend der Vorgehensweise aus Abschnitt 5.3 wird nur ein Intervall verfeinert. Das ist hier das Intervall, das den Skiba-Punkt enthält. Eine ähnliche Tendenz ist auch auf den anderen Gittern, auf denen die optimale Wertefunktion berechnet wird, zu erkennen. Dadurch entsteht ein Endgitter, das in der Umgebung des Knicks wesentlich feiner ist als im Rest von Ω.

Diese ermittelte Stelle, an der die (approximierte) optimale Wertefunktion nicht differenzierbar ist, führt zu Problemen bei der Anwendung der Spline-Interpolation:

Zunächst wird der Algorithmus mit denselben Vorgaben wie bei der linearen Interpolation gestartet, d.h. auf demselben Ausgangsgitter mit 50 Knoten und 50 Kontrollwerten sowie den identischen Parameterwerten.

Man muss feststellen, dass der Algorithmus nie abbricht, d.h. es wird weder eine maximale Knotenzahl erreicht, noch eine geforderte obere Fehlerschranke unterschritten. Es kommt tatsächlich nicht einmal zu einer einzigen Gitteradaption. Das bedeutet, dass auf dem Ausgangsgitter im Laufe der Berechnungen der Wert ζ_{max}^j die gesetzte Grenze $\zeta = 10^{-5}$ nie erreicht.

Betrachtet man die Entwicklung der Werte ζ_{max}^j in jeder Iteration (siehe Abbildung 6.9), so zeigt sich, dass sie nicht wie bei Beispiel 1 gegen Null konvergieren. Die Funktionenfolge $\check{v}_h^j$ konvergiert hier folglich auch nicht gegen ein die exakte optimale Wertefunktion approximierendes $\check{v}_h$ für $j \to \infty$.

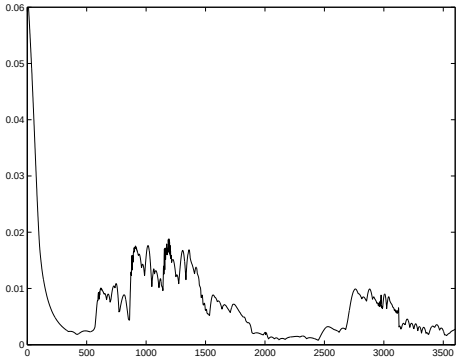


Abbildung 6.9: Entwicklung von ζ_{max}^j zu Beispiel 2

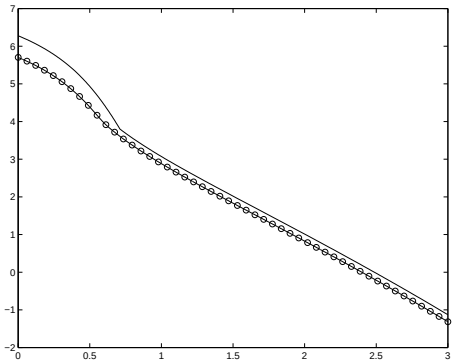
Die x-Achse zeigt analog zu Abbildung 6.2 die Anzahl der Iterationen, die y-Achse den Wert ζ_{max}^j nach der entsprechenden Iteration.

Der Algorithmus wird hier nach 3600 Iterationen abgebrochen. Zum Vergleich: Bei linearer Interpolation ist das Ergebnis auf dem Endgitter nach 230 Iteratio-

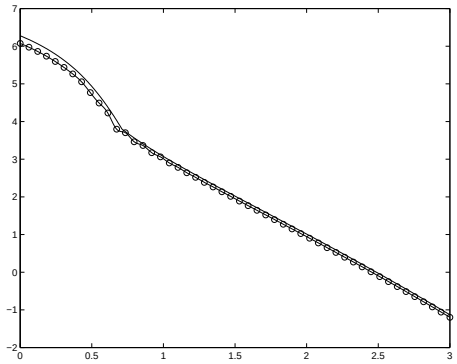
nen berechnet.

Dieses „unerwünschte“ Resultat ist klarerweise auch in der Entwicklung der optimalen Wertefunktion zu sehen. Es zeigt sich, dass sich die Funktionen $\check{v}_h^j$ zunächst an die mit dem kontrollierten Gauß-Seidel-Verfahren bestimmten optimalen Wertefunktion $\hat{v}_h$ annähern, dann aber (ab etwa der 500. Iteration) ein oszillierendes Verhalten ausgehend vom Skiba-Punkt aufweisen. Diese Grenze ist auch in Abbildung 6.9 durch den dortigen Anstieg der Werte ζ_{max}^j zu erkennen.

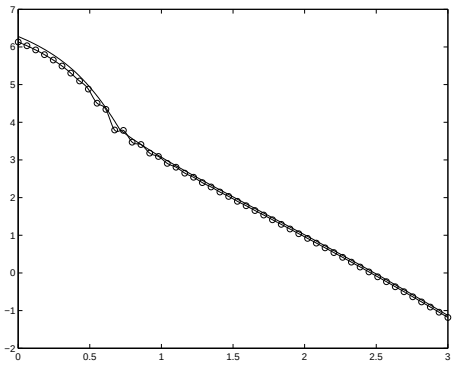
Die folgenden Graphen zeigen die Form von $\check{v}_h^j$ nach je 50 Iterationen. Als Vergleich ist in jeder Abbildung die „Gauß-Seidel-Lösung“ $\hat{v}_h$ zu sehen. Die Kreise zeigen die Werte von $\check{v}_h^j$ an den Knoten. Der „komplette“ Graph von $\check{v}_h^j$ wurde durch Auswertung an insgesamt 3000 Zwischenknoten erzeugt. Die genaue Entwicklung der $\check{v}_h^j$ anhand weiterer Graphen befindet sich in Abschnitt A.2 im Anhang.



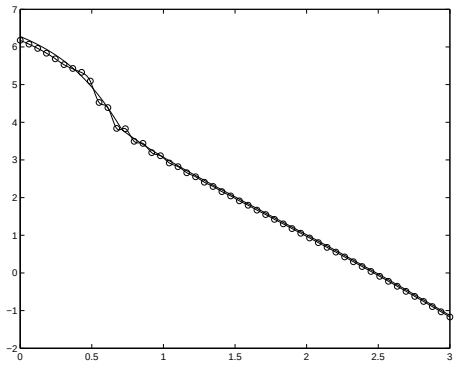
(a) nach 300 Iterationen



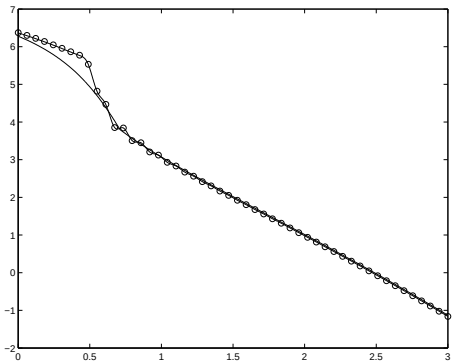
(b) nach 500 Iterationen



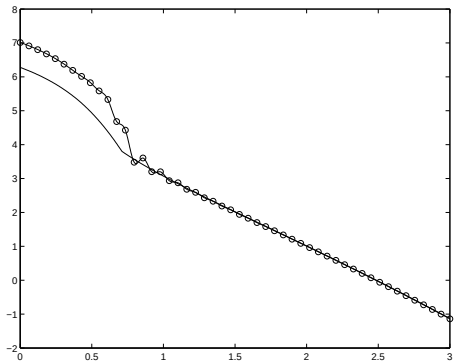
(c) nach 550 Iterationen



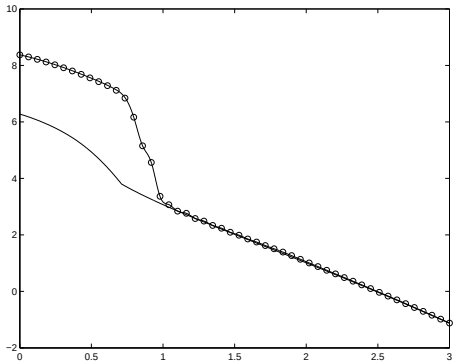
(d) nach 600 Iterationen



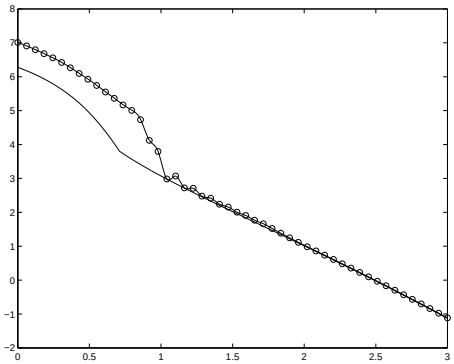
(e) nach 650 Iterationen



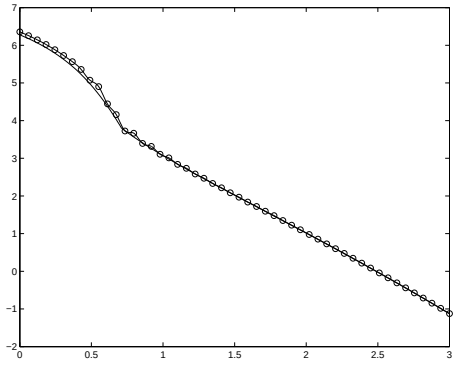
(f) nach 850 Iterationen



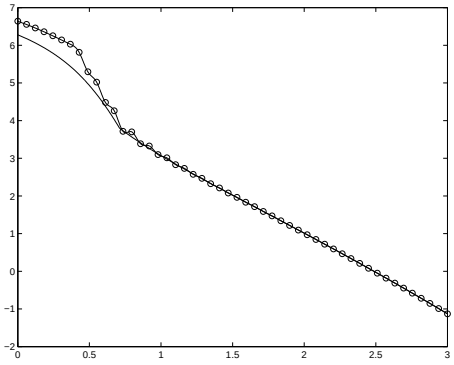
(g) nach 1350 Iterationen



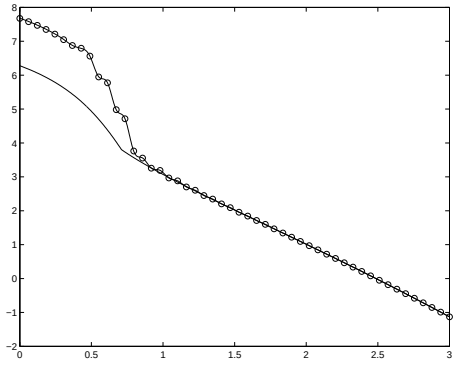
(h) nach 1600 Iterationen



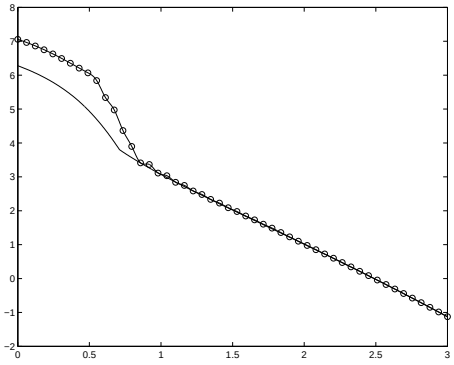
(i) nach 2100 Iterationen



(j) nach 2550 Iterationen



(k) nach 2950 Iterationen



(l) nach 3600 Iterationen

Abbildung 6.10: Entwicklung der optimalen Wertefunktion zu Beispiel 2

Es lässt sich sehr gut erkennen, dass die Funktionen $\check{v}_h^j$, ausgehend vom Skiba-Punkt, ein unregelmäßiges Verhalten zeigen und die erhoffte Konvergenz nicht eintritt.

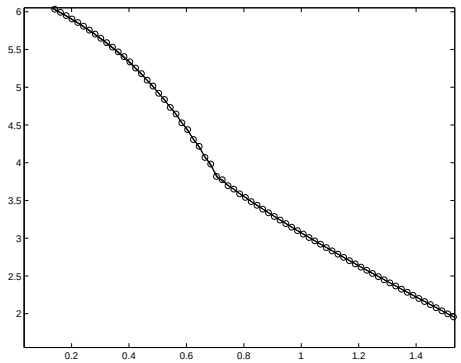
Bei weiteren Untersuchungen stellt man fest, dass sich durch Änderung der Parameter auch das Verhalten der $\check{v}_h^j$ ändert.

Eine Möglichkeit, kleinere Werte für ζ_{max}^j zu erreichen, ist, ein feineres Ausgangsgitter zu wählen. Durch die kürzeren Intervalle liegen die Funktionen $\check{v}_h^j$ auf dem Ausgangsgitter näher an $\check{v}_h$ und somit ist auch der Abstand zwischen $\check{v}_h^j$ und $\check{v}_h^{j-1}$, durch den die Werte ζ_{max}^j bestimmt sind, geringer. Auf diesen Zusammenhang zwischen der Feinheit des Ausgangsgitters und den Werten ζ_{max}^j wird bei Beispiel 4 noch einmal eingegangen.

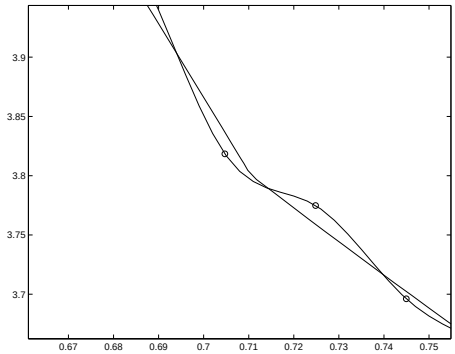
Dementsprechend wird nun ein Ausgangsgitter mit 150 Knoten gewählt. Die anderen Parameterwerte bleiben unverändert.

Wieder muss der Algorithmus „von Hand“ abgebrochen werden. Auch mit dem feinen Ausgangsgitter wird keine Gitteradaption erreicht. Es wird allerdings die Abweichung von der mit linearer Interpolation erhaltenen optimalen Wertefunktion erheblich reduziert. Die errechneten Funktionen $\check{v}_h^j$ oszillieren mit geringer Auslenkung rund um den Skiba-Punkt, während die Approximation außerhalb einer Umgebung dieser Stelle recht gut ist.

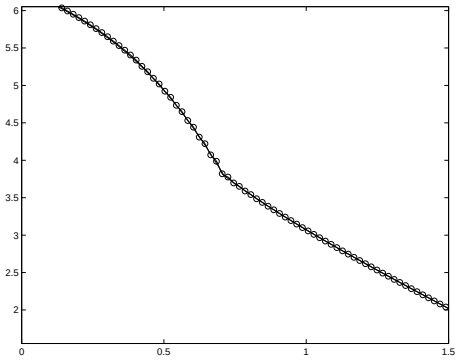
Dies verdeutlichen die folgenden Beispielgraphen:



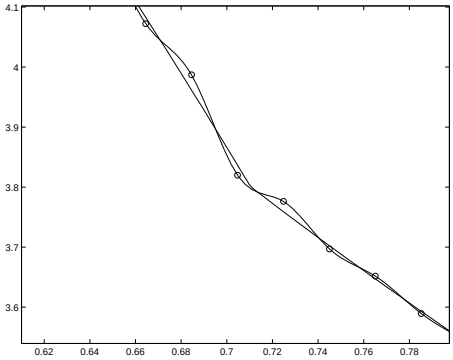
(a) nach 1400 Iterationen



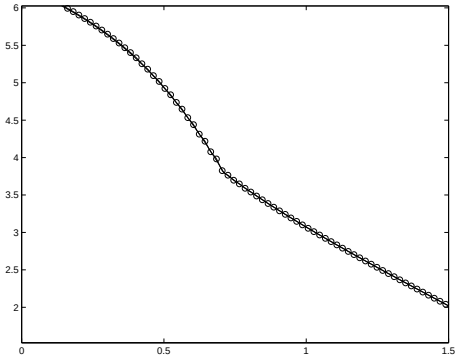
(b) nach 1400 Iterationen (Ausschnitt)



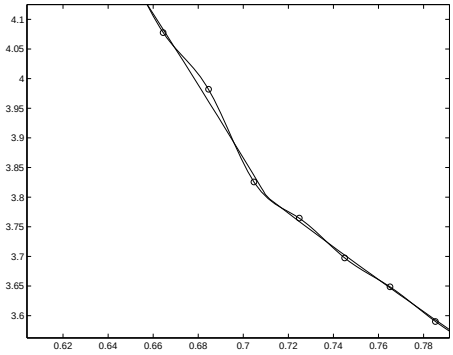
(c) nach 4000 Iterationen



(d) nach 4000 Iterationen (Ausschnitt)



(e) nach 5000 Iterationen



(f) nach 5000 Iterationen (Ausschnitt)

Abbildung 6.11: Entwicklung der optimalen Wertefunktion zu Beispiel 2 auf feinem Startgitter

Die gewünschte Genauigkeit in Form des Wertes ζ wird global nicht erzielt. Stattdessen verhalten sich die Werte ζ^j_{max}, durch die der Abstand von $\check{v}^j_h$ zu $\check{v}_h$ beschränkt ist, periodisch (siehe Abbildung 6.12).

Der minimal erreichte Wert ζ^j_{max} beträgt ca. $1.2 \cdot 10^{-5}$. In der betreffenden Iteration beträgt die obere Schranke für den Fehler gemäß (5.3) etwa 0.88.

Also kann durch das feinere Ausgangsgitter auch keine ausreichend genaue Approximation bestimmt werden; die Abweichung zur Approximation des kontrollierten Gauß-Seidel-Verfahrens ist aber erheblich kleiner geworden.

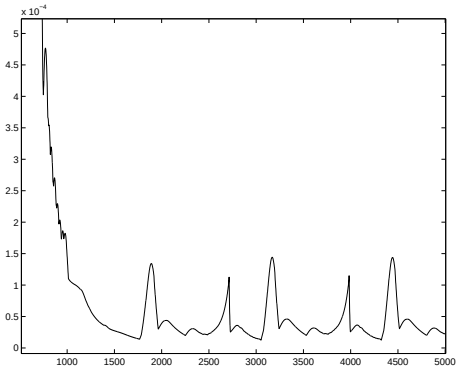


Abbildung 6.12: Entwicklung von ζ^j_{max} zu Beispiel 2 auf feinem Startgitter

Der Algorithmus wird jetzt abermals auf diesem feinen Ausgangsgitter gestartet, die gewünschte Genauigkeit allerdings in Form der Abbruchvariablen ζ auf 10^{-4} reduziert. Bei dieser Parametereinstellung läuft der Algorithmus durch und liefert folgende optimale Wertefunktion auf einem Endgitter mit 240 Knoten:

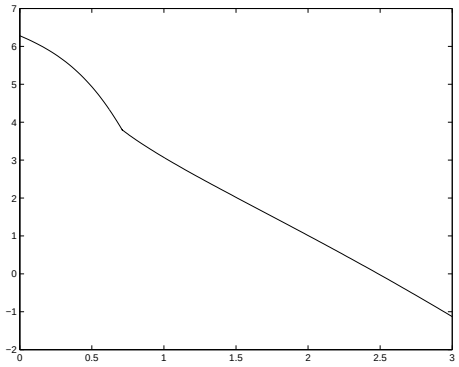


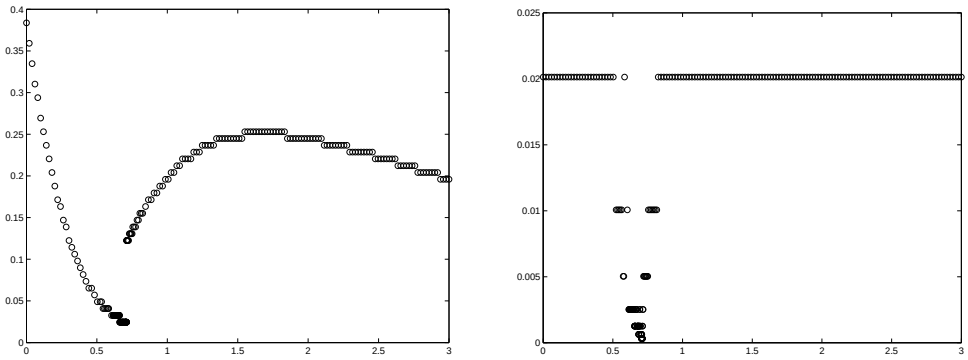
Abbildung 6.13: Optimale Wertefunktion zu Beispiel 2 auf feinem Startgitter mit $\zeta = 10^{-4}$

Die obere Schranke für den Fehler gemäß (5.3) beträgt $2.18 \cdot 10^{-2}$. Die Entwicklung der Knotenzahl der Gitter sowie die zugehörige obere Fehlerschranke zeigt Tabelle 6.7.

Knotenzahl	Fehlerschranke
150	2.19
163	$3.81 \cdot 10^{-1}$
172	$3.88 \cdot 10^{-1}$
179	$2.23 \cdot 10^{-1}$
183	$5.81 \cdot 10^{-2}$
201	$4.70 \cdot 10^{-2}$
240	$2.19 \cdot 10^{-2}$

Tabelle 6.7: Entwicklung der Fehlerschranke zu Beispiel 2 auf feinem Ausgangsgitter mit $\zeta = 10^{-4}$

Wie bei der Lösung mit linearer Interpolation lässt sich der Skiba-Punkt als Sprung bei den optimalen Kontrollwerten (siehe Abbildung 6.14 a)) sowie anhand der starken Verfeinerung bei den Intervalllängen erkennen (siehe Abbildung 6.14 b)).



(a) Optimale Kontrollwerte (b) Intervalllängen des Endgitters

Abbildung 6.14: Optimale Kontrollwerte und Intervalllängen des Endgitters zu Beispiel 2 mit $\zeta = 10^{-4}$

In diesem Fall wird demnach eine gute Approximation der exakten optimalen Wertefunktion erreicht. Der geschätzte Fehler (also die obere Schranke für die Abweichung) ist fast genauso klein wie bei der Lösung des kontrollierten Gauß-Seidel-Verfahrens. Allerdings wird schon durch die beiden beschriebenen Fehlversuche deutlich, dass die Konvergenzeigenschaft des Verfahrens mit linearer Interpolation nicht direkt auf das Spline-Verfahren übertragbar ist. Die in Lemma 4.8 beschriebene Kontraktionseigenschaft ist hier nicht erfüllt. Somit kann

im Fall der Spline-Interpolation auch keine äquivalente Konvergenzaussage der Vektoren v^j bzw. V^j gemacht und keine generelle Abschätzung des Abstands der momentan errechneten optimalen Wertefunktion $\check{v}_h^j$ von der evtl. existierenden Approximation $\check{v}_h$ der exakten optimalen Wertefunktion aufgestellt werden. Der Vergleich der Abbildungen 6.10 und 6.9 lässt aber darauf schließen, dass ein großer Wert ζ_{max}^j auch einen großen Abstand der Approximation $\check{v}_h^j$ von der mit linearer Interpolation bestimmten Optimalwertfunktion impliziert.

Nur durch einen geeignet gewählten Zeitpunkt für den Abbruch der Berechnungen auf den einzelnen Gittern und die damit erfolgende Gitteradaption (hier beim Erreichen des Wertes $\zeta_{max}^j < \zeta = 10^{-4}$) wird die Oszillation verhindert. Würde man nicht so früh zur Gitteradaption übergehen, würden die Werte im Laufe der Berechnungen wieder steigen (vgl. Abbildung 6.9), und die erhaltenen Resultate wären nicht zufriedenstellend . Im Falle eines zu groben Ausgangsgitters, in diesem Beispiel mit 50 Knoten, geht die Berechnung völlig schief und die tatsächliche optimale Wertefunktion wird nicht annähernd erreicht.
Somit lässt schon dieses erste Beispiel den Schluss zu, dass die Anwendung von Spline-Interpolation offenbar nicht sinnvoll ist, wenn die optimale Wertefunktion nicht im gesamten betrachteten Bereich differenzierbar ist. Bei passender Wahl des Ausgangsgitters sowie der Parameter kann in manchen Fällen eine gute Approximation errreicht werden. Diese Approximation ist allerdings nur eine „glückliche Momentaufnahme“ und nicht das Resultat aus der Konvergenz der Vektoren v^j gegen einen Vektor v wie in Beispiel 1.
Die folgenden Beispiele verdeutlichen ebenfalls den Zusammenhang zwischen dem Skiba-Punkt und dem oszillierenden Verhalten der mit Hilfe von Spline-Interpolation berrechneten Approximationen.

6.3.2 Beispiel 3: Investitionsmodell

Diese Problemstellung beschreibt das Ziel eines Unternehmens, den Cash-flow durch eine optimale Investitionsstrategie zu maximieren. Dabei sind bestimmte technologische Beschränkungen zu berücksichtigen. Ein Problem dieser Art wird in Hartl u.a. [17] vorgestellt
Die Ertragsfunktion, hier der Cash-flow, ist gegeben durch

$$g(x,u) = ax - bx^2 - ux - cu - du^2$$

Die Höhe des Kapitals ist beschrieben durch die Dynamik

$$\frac{d}{dt}\Phi(t,x,u) = u(t)\Phi(t,x,u) - \sigma\Phi(t,x,u)$$

Dabei steht x für das Kapital und $u \cdot x$ für die Investition.

Wiederum mittels Euler-Diskretisierung ergibt sich das zeitdiskrete Problem

$$\begin{aligned}\tilde{\Phi}_h((j+1)h, x, u_h) &= \tilde{\Phi}_h(jh, x, u_h) + u_h(jh)\tilde{\Phi}_h(jh, x, u_h) - \sigma\tilde{\Phi}_h(jh, x, u_h) \\ &= f_h(\tilde{\Phi}_h(jh, x, u_h), u_h(jh))\end{aligned}$$

Als Werte für die Parameter werden gewählt

$$a = 1\ , b = 0.6\ , c = 0\ , d = 0.3\ , \sigma = 0.1,$$

sowie die Diskontrate $\delta = 0.2$ und der Zeitschritt $h = 0.05$. Der relevante Bereich der optimalen Wertefunktion ist hier $\Omega = [0, 0.7]$, während Kontrollwerte aus $U = [0, 1]$ herangezogen werden.

Zunächst werden wieder die Ergebnisse des kontrollierten Gauß-Seidel-Verfahrens betrachtet. Das Ausgangsgitter besteht aus 131 Knoten und der Verfeinerungsparameter wird hier auf $\theta = 0.3$ verändert.
Abbildung 6.15 zeigt die errechnete optimale Wertefunktion.

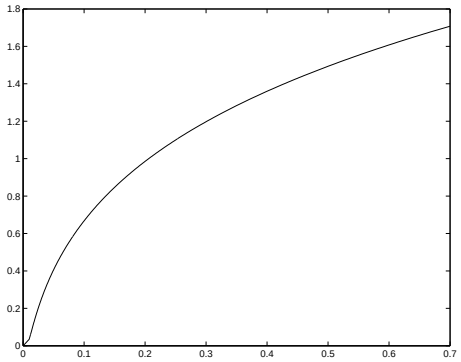
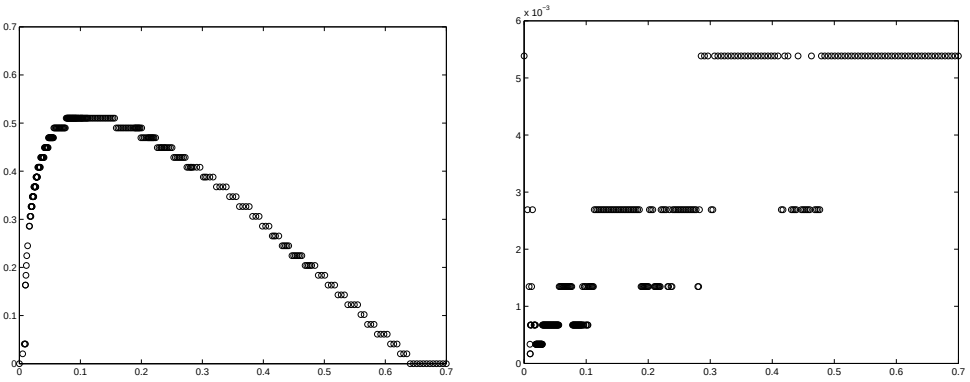


Abbildung 6.15: Optimale Wertefunktion zu Beispiel 3 mit GSV

Die durch die Fehlerschätzer aus (5.1) bestimmte obere Schranke des Fehlers beträgt auf dem Endgitter mit 307 Knoten $2.2 \cdot 10^{-3}$. Auch bei diesem Beispiel lässt sich ein Skiba-Punkt als Knick in der optimalen Wertefunktion ausfindig machen, der sich ebenfalls in den Graphen der optimalen Kontrollwerte sowie dem der Intervalllängen des Endgitters widerspiegelt (siehe Abbildung 6.16).

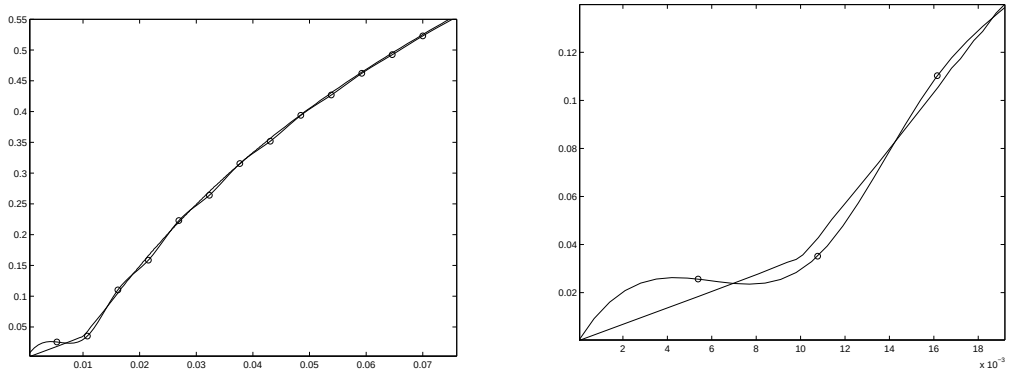
Der Knick bei $x = 0.01$ ist in Abbildung 6.15 noch relativ schwer zu erkennen. In Abbildung 6.16 (a) ist der Sprung der optimalen Wertefunktion bei $x = 0.01$ jedoch gut zu sehen und auch Graphik 6.16 (b) macht die starke Verfeinerung rund um den Skiba-Punkt deutlich.



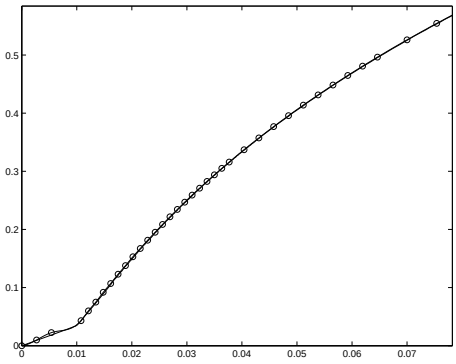
(a) Optimale Kontrollwerte (b) Intervalllängen

Abbildung 6.16: Optimale Kontrollwerte und Intervalllängen des Endgitters zu Beispiel 3 mit GSV

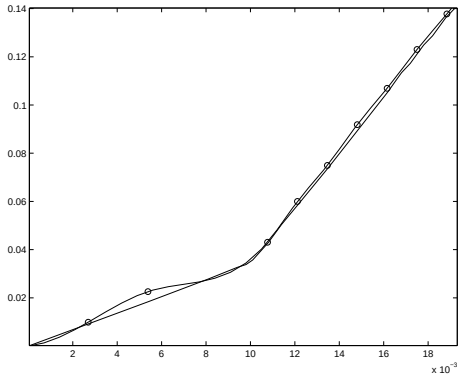
Bei Anwendung der Spline-Interpolation auf dem selben Ausgangsgitter mit 131 Knoten und den selben Parameterwerten ($\theta = 0.3,\ \zeta = 1 \cdot 10^{-5}$) muss der Algorithmus wieder abgebrochen werden. Das geschieht hier nach 3000 Iterationen. Offenbar wird wieder keines der Abbruchkriterien erfüllt. Es finden zwar vier Gitteradaptionen statt; auf dem fünften Gitter wird jedoch keine zufriedenstellende Lösung berechnet. Die errechnete optimale Wertefunktion oszilliert leicht um den Knick bei $x = 0.01$, was an den folgenden Graphen zu erkennen ist:



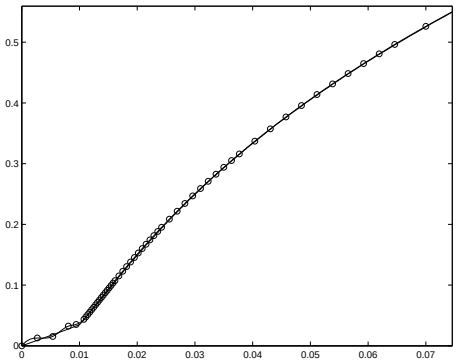
(a) nach 600 Iterationen (b) nach 600 Iterationen (Ausschnitt)



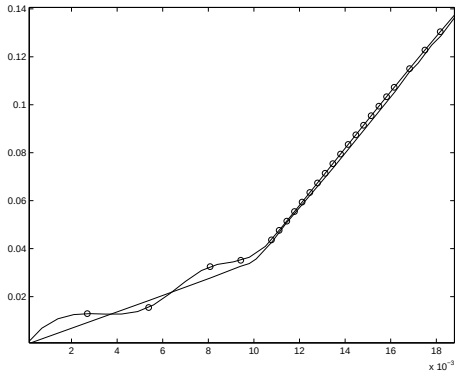
(c) nach 1200 Iterationen



(d) nach 1200 Iterationen (Ausschnitt)



(e) nach 2600 Iterationen



(f) nach 2600 Iterationen (Ausschnitt)

Abbildung 6.17: Entwicklung der optimalen Wertefunktion zu Beispiel 3

Betrachtet man die Entwicklung der Werte ζ^j_{max}, fällt nach der vierten Gitteradaption wie schon in Abbildung 6.12 bei Beispiel 2 ein periodisches Verhalten auf:

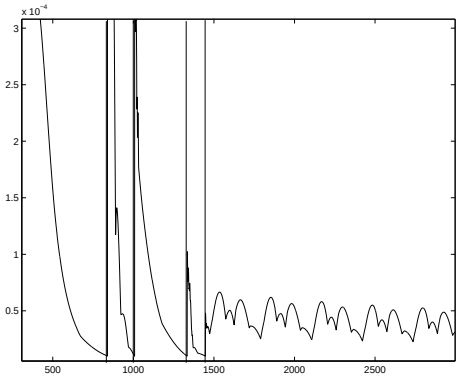


Abbildung 6.18: Entwicklung von ζ^j_{max} zu Beispiel 3

Die Gitteradaptionen bei etwa 800, 1000, 1300 und 1400 Iterationen sind wie schon in Abbildung 6.2 durch senkrechte Striche markiert und durch den dortigen deutlichen Anstieg der Werte zu erkennen.

Durch Veränderung der Parameter wird nun erneut versucht, bessere Ergebnisse zu erzielen. Der Start auf einem feineren Gitter verspricht wenig Erfolg, da auf den ersten Gittern die Werte ζ^j_{max} im Laufe der Berechnungen bereits die Schranke $\zeta = 10^{-5}$ unterschreiten.
Um auch auf feineren Gittern weitere Adaptionen zu erreichen, wird ζ vergrößert. Als eine geeignete Wahl erweist sich $\zeta = 6 \cdot 10^{-5}$. Hier beendet der Algorithmus seine Berechnungen auf einem Gitter mit 367 Knoten mit der optimalen Wertefunktion in Abbildung 6.19.
Zwar beträgt die obere Schranke für den Fehler $5.8 \cdot 10^{-3}$ (die Approximation ist also relativ nahe an der exakten optimalen Wertefunktion), bei genauer Betrachtung fällt jedoch auf, dass der Knick in der optimalen Wertefunktion nicht erkannt worden ist. Die Vergrößerung in Abbildung 6.20 macht dies deutlich.

Die untere Funktion ist dabei die mit dem kontrollierten Gauß-Seidel-Verfahren ermittelte optimale Wertefunktion, die obere diejenige, die mit Spline-Interpolation bestimmt worden ist.

Diese Problematik ist auch bei der Betrachtung der optimalen Kontrollwerte in Abbildung 6.21 zu erkennen. Der Sprung ist weit weniger deutlich ausgeprägt als bei der Lösung mit linearer Interpolation (vgl. Abbildung 6.16 a)).

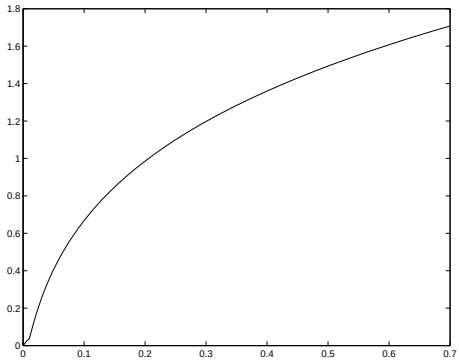


Abbildung 6.19: Optimale Wertefunktion zu Beispiel 3 mit $\zeta = 6 \cdot 10^{-5}$

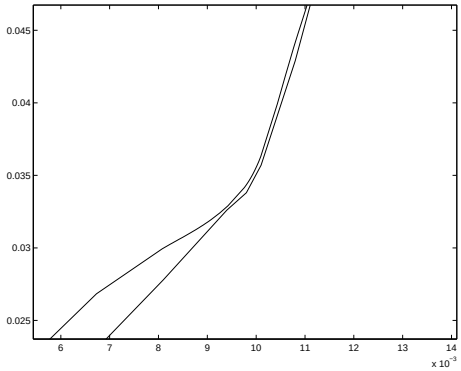


Abbildung 6.20: Optimale Wertefunktion zu Beispiel 3 mit $\zeta = 6 \cdot 10^{-5}$ (Ausschnitt)

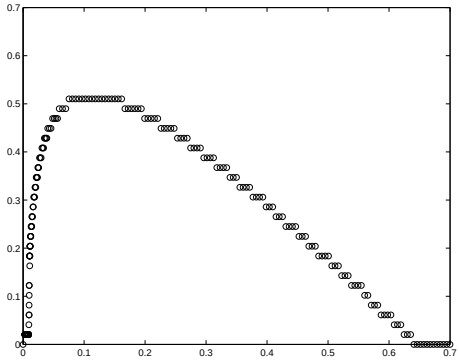
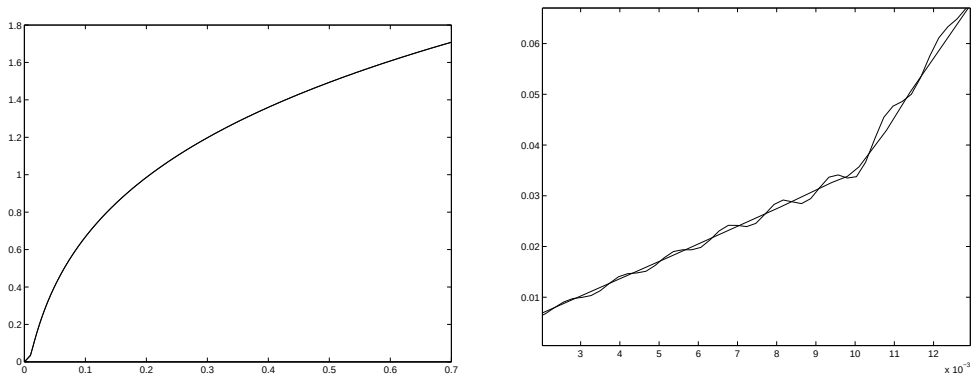


Abbildung 6.21: Optimale Kontrollwerte zu Beispiel 3 mit $\zeta = 6 \cdot 10^{-5}$

Außerhalb einer kleinen Umgebung des Knickes stimmt die berechnete optimale Wertefunktion fast genau mit der durch das kontrollierte Gauß-Seidel-Verfahren berechneten überein.

Dass dieses unregelmäßige Verhalten in der Umgebung des Skiba-Punktes nicht durch eine äquidistante Verfeinerung im ganzen betrachteten Bereich aufgefangen werden kann, zeigen diese Ergebnisse:
Verfeinert man bei der Gitteradaption nicht nur die Intervalle mit vergleichsweise großem Fehlerschätzer, sondern alle (das entspricht dem Verfeinerungsparameter $\theta = 0$), so zeigt die erhaltene optimale Wertefunktion ein ähnliches Verhalten rund um $x = 0.01$.



(a) auf ganz Ω (b) Ausschnitt

Abbildung 6.22: Optimale Wertefunktion zu Beispiel 3 bei äquidistanter Verfeinerung

Das Endgitter besteht aus 1041 Knoten, es finden also 4 Verfeinerungen statt. Wieder wird anhand des Vergleichs der Graphen der berechneten optimalen Wertefunktionen mit denen der „linearen Lösung“ deutlich, dass es nur um den Skiba-Punkt herum Probleme mit der Spline-Interpolation gibt.

Bei Beispiel 3 wird im Gegensatz zu Beispiel 2 in keinem der berechneten Fälle eine gute Approximation der optimalen Wertefunktion durch Spline-Interpolation erzielt.

6.3.3 Beispiel 4: Wachstumsmodell mit Kreditaufnahme

Als viertes und letztes Beispiel wird ein Wachstumsmodell aus Grüne u.a. [14] betrachtet. Es erlaubt z.B. einem Staat (unter Annahme eines uneingeschränkten

Zugangs zum Kapitalmarkt) die Aufnahme von Krediten, um zu investieren. Die maximale Kredithöhe ist allerdings durch das momentane Einkommen aus der Geschäftstätigkeit gegeben. Bestimmt werden soll eine Beschränkung der Kreditaufnahme abhängig vom Zeitpunkt. Diese Angaben führen zu dem Problem, das definiert ist durch die Ertragsfunktion

$$g(x,u) = Ax^{\alpha} - u - u^{\gamma}x^{-\mu}$$

sowie die Dynamik

$$\frac{d}{dt}\Phi(t,x,u) = u(t) - \sigma\Phi(t,x,u).$$

x beschreibt die Höhe des Kapitals, u die Höhe der Investition. In zeitdiskreter Form erhält man

$$\begin{aligned}\tilde{\Phi}_h((j+1)h,x,u_h) &= \tilde{\Phi}_h(jh,x,u_h) + u_h(jh) - \sigma\tilde{\Phi}_h(jh,x,u_h)\\ &= f_h(\tilde{\Phi}_h(jh,x,u_h),u_h(jh)).\end{aligned}$$

Die optimale Wertefunktion wird berechnet auf $\Omega =]0,2]$ mit den Parametern

$$A = 0.29\ ,\alpha = 1.1\ ,\gamma = 2\ ,\mu = 0.3c\ ,\sigma = 0.15,$$

dem Kontrollwertebereich $U = [0, 0.25]$ und einer Schrittweite $h = 0.05$. Die Diskontrate $\delta = 0.1$ entspricht hier dem konstanten Zinssatz auf den Kredit.

Bei Lösung mit linearer Interpolation auf einem Startgitter mit 39 Knoten und dem Verfeinerungsparameter $\theta = 0.5$ entsteht die optimale Wertefunktion in Abbildung 6.23.

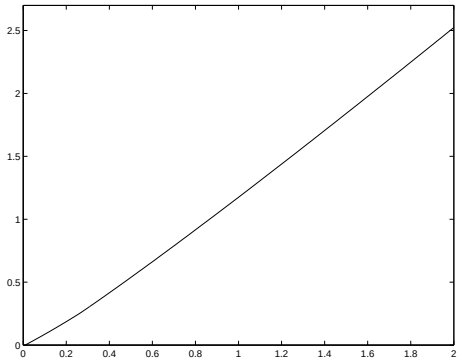
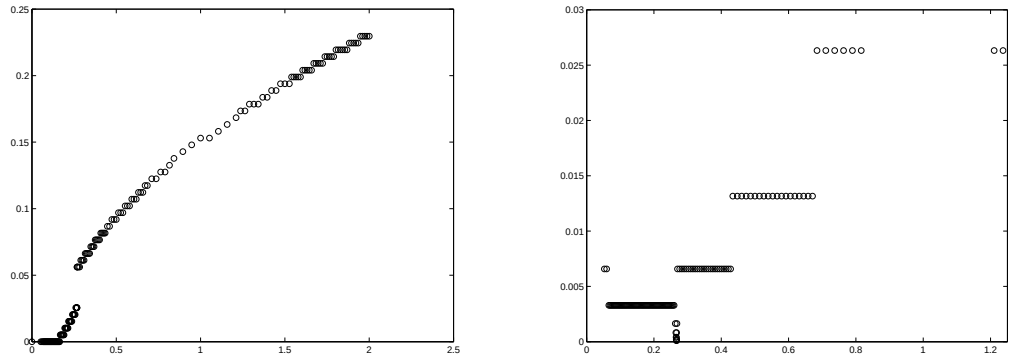


Abbildung 6.23: Optimale Wertefunktion zu Beispiel 4 mit GSV

Der Fehler auf dem Endgitter mit 180 Knoten ist durch die Schranke $9.2 \cdot 10^{-4}$ begrenzt.

Bei diesem Beispiel ist der Skiba-Punkt als Knick in der optimalen Wertefunktion nicht zu erkennen. Erst bei Betrachtung der optimalen Kontrollwerte (Abbildung 6.24 a)) sowie den Intervalllängen des Endgitters (Abbildung 6.24 b)) wird dieser durch den Sprung bzw. die hohe Gitterfeinheit bei $x = 0.267$ sichtbar. In diesem Fall stellt der Skipa-Punkt eine Grenze dar, unterhalb der es optimal ist, die Kapitalmenge zu vermindern. Oberhalb der Grenze steigt die Kapitalmenge und damit das Pro-Kopf Einkommen.

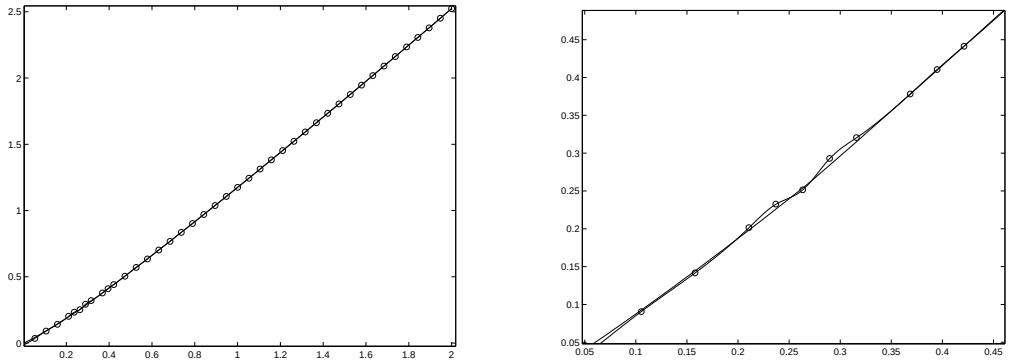


(a) Optimale Kontrollwerte

(b) Intervalllängen des Endgitters

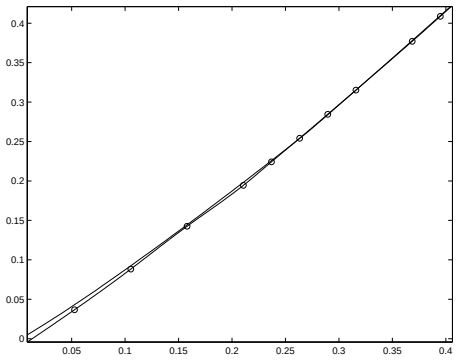
Abbildung 6.24: Optimale Kontrollwerte und Intervalllängen des Endgitters zu Beispiel 4 mit GSV

Bei der Berechnung mit Spline-Interpolation ergibt sich das inzwischen bekannte Ergebnis. Die Approximation oszilliert um den Skiba-Punkt:

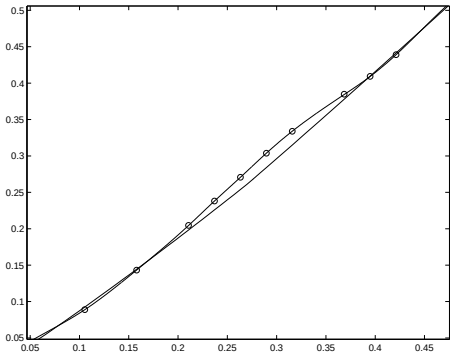


(a) nach 1750 Iterationen

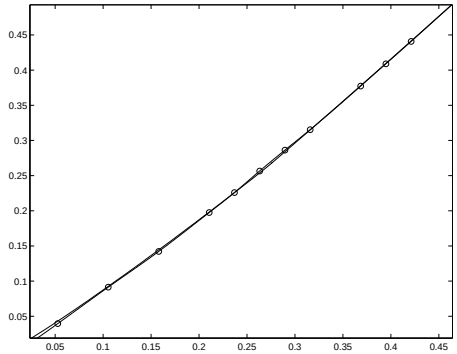
(b) nach 1750 Iterationen (Ausschnitt)



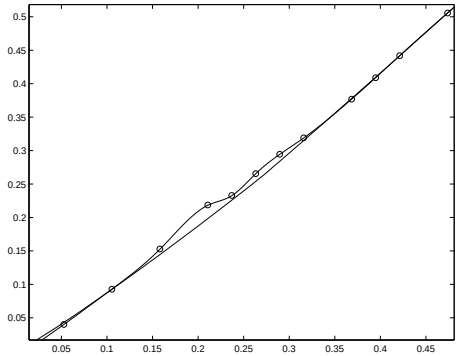
(c) nach 2250 Iterationen (Ausschnitt)



(d) nach 2750 Iterationen (Ausschnitt)



(e) nach 3250 Iterationen (Ausschnitt)



(f) nach 4000 Iterationen (Ausschnitt)

Abbildung 6.25: Entwicklung der optimalen Wertefunktion zu Beispiel 4

Es kommt zu einer Gitteradaption; anschließend muss der Algorithmus allerdings abgebrochen werden. Das geschieht hier nach 4000 Iterationen.

Die Betrachtung der Werte ζ_{max}^{j} in jeder Operation zeigt wiederum sehr deutlich die fehlende Konvergenzeigenschaft:

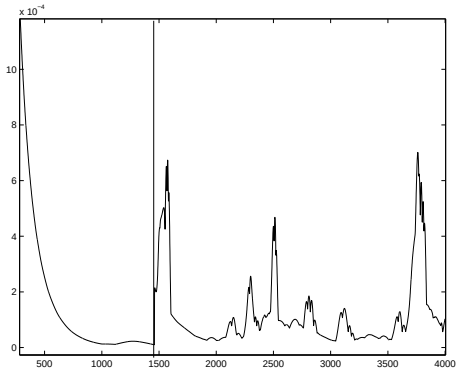


Abbildung 6.26: Entwicklung von ζ^j_{max} zu Beispiel 4

Um mehrere Adaptionen zu erreichen, wird nun wie schon bei den vorherigen Beispielen die geforderte Genauigkeit reduziert, indem der Parameter ζ auf $2 \cdot 10^{-5}$ erhöht wird. Durch diese Änderung kommt es zu weiteren Gitterverfeinerungen und auf einem Endgitter mit 119 Knoten ergibt sich die optimale Wertefunktion $\check{v}_h$ in Abbildung 6.27.

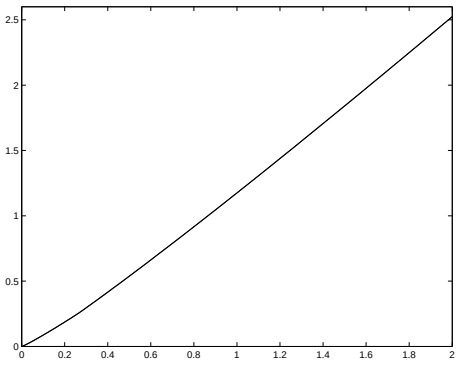


Abbildung 6.27: Optimale Wertefunktion zu Beispiel 4 mit $\zeta = 2 \cdot 10^{-5}$

Die obere Schranke für den Fehler beträgt $3.1 \cdot 10^{-3}$.

Hier ist das Ergebnis für die optimale Wertefunktion zwar was die obere Schranke des Fehlers betrifft zufriedenstellend, allerdings ist die Existenz des Skiba-Punktes bei $x = 0.267$ bei Betrachtung der optimalen Kontrollwerten nicht so deutlich zu erkennen wie in Abbildung 6.24:

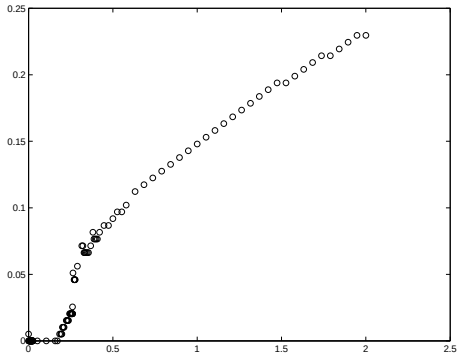


Abbildung 6.28: Optimale Kontrollwerte zu Beispiel 4 mit $\zeta = 2 \cdot 10^{-5}$

Aus der Betrachtung der Intervalllängen des Endgitters ergibt sich, dass zwar eine stärkere Verfeinerung in der Umgebung des Skiba-Punktes stattgefunden hat, allerdings nicht nur an diesem direkt.

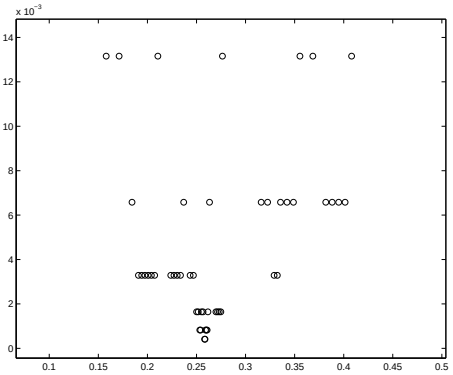
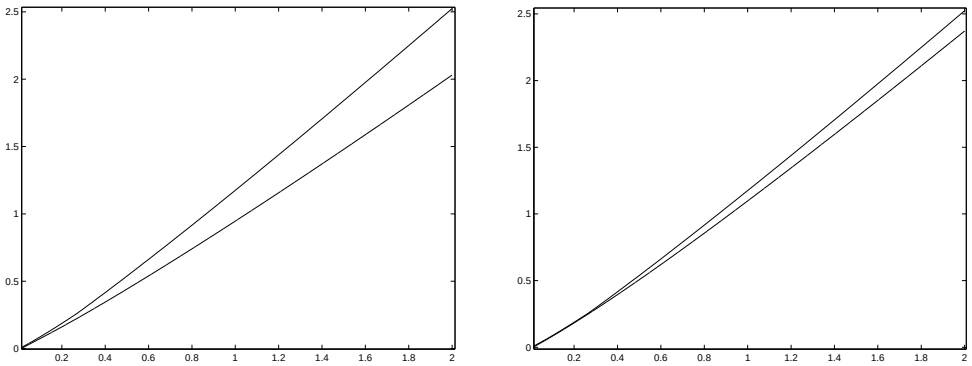


Abbildung 6.29: Intervalllängen des Endgitters in der Umgebung des Skiba-Punktes zu Beispiel 4 mit $\zeta = 2 \cdot 10^{-5}$

Abermals zeigt also die Approximation in dieser Umgebung ein unregelmäßiges Verhalten.

Bemerkenswert ist außerdem folgendes Ergebnis:
Erhöht man ζ weiter, beispielsweise auf $5 \cdot 10^{-5}$, stoppt der Algorithmus mit einer Approximation auf einem Gitter mit 119 Knoten und der Fehlerschranke $9.8 \cdot 10^{-3}$. Die durchgeführten Gitteradaptionen finden aber in folgendem Sinn zu früh statt:

Die Funktionen nähern sich der Gauß-Seidel-Approximation von unten an.



(a) nach 300 Iterationen (b) nach 600 Iterationen

Abbildung 6.30: Entwicklung der optimalen Wertefunktion zu Beispiel 4 mit $\zeta = 5 \cdot 10^{-5}$

In der Iteration, in der die maximale Differenz zwischen den Vektoren v^j und v^{j-1} kleiner als $\zeta = 5 \cdot 10^{-5}$ ist, das Gitter also verfeinert wird, ist diese Annäherung noch nicht abgeschlossen. Im „rechten Teil“ von $\check{v}_h^j$ ist der Abstand zur exakten optimalen Wertefunktion noch relativ groß. Das Gitter wird daher fast nur in den rechts liegenden Intervallen verfeinert. Daraus resultiert ein Gitter, das rechts sehr fein ist, links aber beinahe unverändert bleibt:

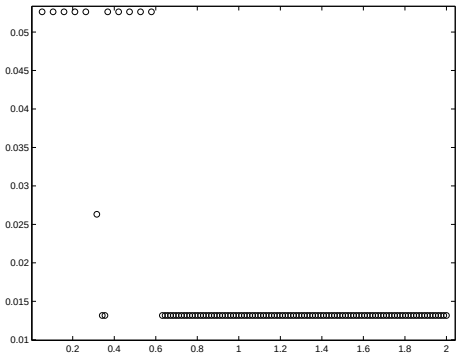
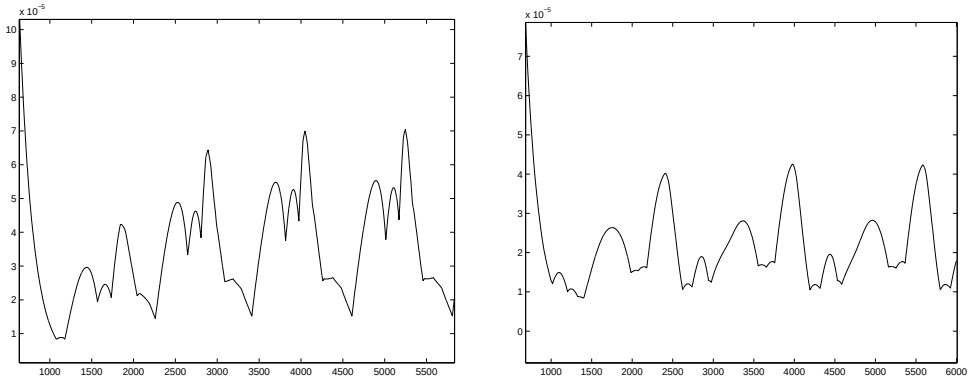


Abbildung 6.31: Intervalllängen des Endgitters zu Beispiel 4 mit $\zeta = 5 \cdot 10^{-5}$

Die Verfeinerung erfasst also die Struktur der errechneten optimalen Wertefunktion nicht; der Knick wird nicht erkannt. Dieses Resultat belegt, dass es nicht

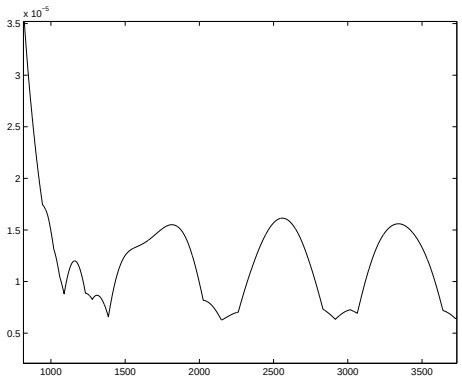
immer sinnvoll ist, den Parameter ζ zu erhöhen. Zumindest kann eine Art Grenze für ζ existieren, bei deren Überschreitung keine brauchbaren Ergebnisse mehr erzielt werden.

Bei den beiden vorhergehenden Beispielen fällt auf, dass es bei einigen Brechnungen zu einer periodischen oder zumindest annähernd periodischen Entwicklung der Werte ζ^j_{max} auf (vgl. Abbildungen 6.12 und 6.18) kommt. Lässt man den Algorithmus auf dem Ausgangsgitter weiterlaufen und geht nicht bei einer bestimmten Grenze auf ein neues Gitter über, so ist dieses Phänomen auch bei diesem Beispiel auf verschiedenen Gittern zu betrachten:



(a) 20 Knoten

(b) 49 Knoten



(c) 77 Knoten

Abbildung 6.32: Entwicklung von ζ^j_{max} zu Beispiel 4 auf verschiedenen äquidistanten Gittern

Dies belegt erneut, wie schon in den abschließenden Überlegungen zu Beispiel 2 auf Seite 102 bemerkt, dass keine Kontraktion und somit keine Konvergenz der Vektoren v^j gegen einen Vektor v für $j \to \infty$ vorliegt. An Abbildung 6.32 ist abzulesen, dass die Schwankungen des maximalen Abstands zwischen v^j und v^{j-1} auf feineren Gittern geringer werden. Die Äuslenkung der Oszillation hängt demnach von der Feinheit des Gitters ab. Einen derartigen Schluss lässt auch der Vergleich der Berechnungen zu Beispiel 2 auf den Gittern mit 50 bzw. 150 Startknoten zu (vgl. Abbildungen 6.9 und 6.12).

Zum Abschluss der praktischen Tests folgt noch eine weitere Beobachtung:
Die erhaltenen Ergebnisse sind auch von den Anfangswerten v_i^0 an den Knoten des zu Grunde liegenden Gitters abhängig. Zu Beginn der Berechnungen sind diese auf dem Ausgangsgitter mit Null vorbelegt. Startet man nun bei Beispiel 4 auf einem äquidistanten Gitter mit 77 Knoten, so erkennt man an Abbildung 6.32 c), dass die Oszillation erst sehr spät einsetzt (spät heißt hier bei einem kleinen Wert für ζ_{max}^j) und im weiteren Verlauf maximal ein Wert von etwa $\zeta_{max}^j \approx 1.6 \cdot 10^{-5}$ berchnet wird. Startet man im Gegensatz dazu allerdings auf einem Gitter mit 39 Knoten und verfeinert bei Erreichen der Grenze $\zeta = 10^{-5}$ jede Zelle, d.h. man erhält das identische äquidistante Gitter mit 77 Knoten wie oben, entwickeln sich die ζ_{max}^j völlig anders. Schon sehr bald nach der Verfeinerung steigen die Werte ζ_{max}^j an und pendeln zwischen wesentlich höheren Werten (bis zu $\zeta_{max}^j = 1 \cdot 10^{-3}$). Die leichte Auslenkung, die der Graph $\check{v}_h^j$ auf dem Ausgangsgitter um den Skiba-Punkt bei der Verfeinerung bereits aufweist, wird auf dem neuen Gitter verstärkt und die Abweichungen werden größer. Die folgende Abbildung zeigt die Entwicklung der Werte ζ_{max}^j nach der Verfeinerung auf das Gitter mit 77 Knoten (im Vergleich dazu die Entwicklung auf dem Ausgangsgitter mit den identischen 77 Knoten in Abbildung 6.32 c)):

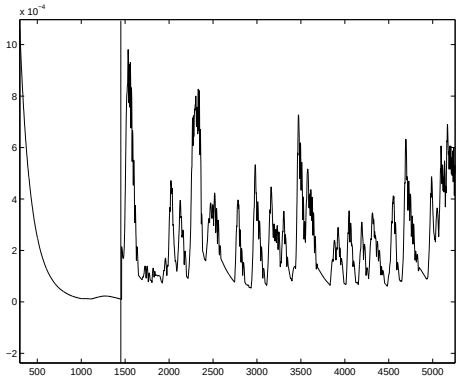


Abbildung 6.33: Entwicklung von ζ_{max}^j zu Beispiel 4 bei äquidistanter Verfeinerung

6.4 Fazit der praktischen Ergebnisse

Stellen nun interpolierende Splines bei der numerischen Bestimmung der optimalen Wertefunktion eindimensionaler optimaler Steuerungsprobleme mit dem vorgestellten Iterationsverfahren eine geeignete Alternative zur linearen Interpolation dar?

Als Antwort sollen die aus den Beispielen gewonnenen Erkenntnisse noch einmal kurz zusammengefasst werden.

Bei Problemen mit differenzierbarer optimaler Wertefunktion wie Beispiel 1 kann man diese Frage bejahen. Durch die Splines wird die optimale Wertefunktion genauer approximiert als durch die Verwendung stückweise affin linearer Funktionen.
Da wie bereits erwähnt bei der in dieser Arbeit verwendeten Implementierung nicht in erster Linie auf Minimierung der Rechenzeit Wert gelegt wurde, bleibt zu untersuchen, ob sich das Verfahren auch zeitsparender implementieren lässt. Das vergleichsweise langsame MATLAB ist dazu sicherlich nicht geeignet, auch wenn die Berechnung und Auswertung der Splinefunktionen hier sehr bequem ist. Generell lässt sich sagen, dass das Verfahren der adaptiven Gittererzeugung große Einsparungen bringt.

Bei den Beispielen 2 bis 4, bei denen die optimale Wertefunktion einen Knick aufweist, sind die Ergebnisse nicht so positiv. Die durch das Verfahren mit Spline-Interpolation errechneten optimalen Wertefunktionen $\check{v}_h^j$ entwickeln sich abhängig vom Ausgangsgitter, den Startwerten an den Knoten und der Wahl der Parameter im Algorithmus. Sie nähern sich zwar zunächst der mit Hilfe des Verfahrens mit linearer Interpolation bestimmten Approximation an, oszillieren dann aber früher oder später in einer Umgebung der nicht differenzierbaren Stelle mit einer von den eben genannten Bedingungen abhängigen Auslenkung. Es ist keine Konvergenz gegen eine die exakte optimale Wertefunktion approximierende Funktion $\check{v}_h$ für $j \to \infty$ zu erkennen. Bestenfalls durch geschicktes Eingreifen in den Algorithmus kann bei manchen Beispielen eine gute Approximation gefunden werden.

Der Algorithmus scheint folglich numerisch instabil zu sein.

Die vorliegende Arbeit zeigt einen ersten Einblick in die Betrachtung der Anwendungsmöglichkeiten der Spline-Interpolation bei der Lösung optimaler Steuerungsprobleme. Die verschiedene Berechnungen führen zu den dargestellten Rückschlüssen.
Diese bieten Anknüpfungspunkte, die in weiterführenden Arbeiten betrachtet werden können. Beispielsweise ausführliche Stabilitätsbetrachtungen des beschriebenen Lösungsverfahrens mit Spline-Interpolation oder auch die genauen Aus-

wirkungen der Gitteradaption, bzw. der lokalen Intervallverfeinerung, auf den Interpolationsfehler von interpolierenden Splines.

Anhang A

Ergänzungen

In diesem Abschnitt werden die genaueren Ergebnisse aufgeführt, auf die in der Arbeit verwiesen worden ist. Zum einen die Ergebnisse zu Kapitel 5.4 bei der Betrachtung eines Gitters mit unterschiedlichen Intervalllängen sowie zum anderen die genaue Entwicklung der errechneten optimalen Wertefunktion zu Beispiel 2, an der man die Enstehung der Oszillation um den Skiba-Punkt deutlich erkennen kann.

A.1 Ergänzungen zu Abschnitt 5.4

In Kapitel 5.4 wird gezeigt, dass die Auswirkung einer Intervallverfeinerung auf den Betrag des Abstandes zwischen der ersten Ableitung des Splines s und der ersten Ableitung der interpolierten Funktion f an den Knotenpunkten sehr schwer allgemein auszudrücken ist.
Schon für identische Intervalllängen ergeben sich sehr komplizierte Werte für $E_i = |s'(x_i) - f'(x_i)|$.
Stellt man nun das Gleichungssystem 5.7 für allgemeine unterschiedliche Intervalllängen $h_i,\ i = 1, ..., 6$, auf, so ergibt sich

$$\begin{pmatrix} 4(h_1+h_2) & h_1 & 0 & 0 & 0 \\ h_3 & 4(h_2+h_3) & h_2 & 0 & 0 \\ 0 & h_4 & 4(h_3+h_4) & h_3 & 0 \\ 0 & 0 & h_5 & 4(h_4+h_5) & h_4 \\ 0 & 0 & 0 & h_6 & 4(h_5+h_6) \end{pmatrix} \tilde{E} = \tilde{Z} \quad \text{(A.1)}$$

mit $[\tilde{Z}]_i = \frac{-1}{24} f^{(4)}(\xi_i)(h_i h_{i+1}^3 + h_i^3 h_{i+1}),\ i = 1, ..., 5.$

Löst man dieses System wiederum mit Maple, ergeben sich folgende kaum überschaubaren Werte für $\tilde{E}_i$:

$$\tilde{E}_1 = \frac{1}{C}\left(-\frac{1}{48}(6h3^2h2^2h5^2 + 12h2^3h_4^2h_5 + 9h_2^3h_4^2h_6 + 12h_2^3h_3h_5^2 + 12h_2^3h_4h_5^2\right.$$

$$
\begin{aligned}
&+2h_5^2h_3h_4^3 - 2h_5^4h_3h_4 - 4h_5^2h_4h_3^3 - 4h_4^2h_5h_3^3 + 4h_4^4h_5h_3 - 3h_6h_4^2h_3^3\\
&-6h_3^4h_5h_6 - 6h_3^4h_6h_4 - 8h_3^4h_4h_5 + 3h_6h_4^4h_3 + 12h_2^3h_4h_5h_6 + 12h_2^3h_5h_3h_6\\
&+16h_2^3h_3h_4h_5 + 12h_2^3h_3h_6h_4 + h_5h_6^3h_3h_4 - 4h_5h_6h_4h_3^3 - h_5^3h_6h_3h_4\\
&+2h_5h_6h_3h_4^3 + 8h_3h_2^2h_4h_5h_6 + 8h_3h_2^2h_4^2h_5 + 8h_3^2h_2^2h_4h_5\\
&+6h_3h_2^2h_4^2h_6 + 6h_3^2h_2^2h_6h_4 + 12h_1^2h_2h_4h_5^2 - 6h_3^4h_5^2 + 6h_3^2h_2^2h_5h_6\\
&+9h_1^2h_2h_4^2h_6 + 12h_1^2h_5h_6h_4h_2 + 12h_1^2h_5h_6h_3h_2 + 16h_1^2h_3h_4^2h_5\\
&+16h_1^2h_3h_4h_5h_6 + 12h_1^2h_3^2h_5^2 + 12h_1^2h_3h_4^2h_6 + 12h_1^2h_3^2h_6h_4\\
&+16h_1^2h_3h_4h_5^2 + 12h_1^2h_3^2h_5h_6 + 16h_1^2h_3^2h_4h_5 + 16h_1^2h_2h_3h_4h_5\\
&+12h_1^2h_2h_3h_6h_4 + 12h_1^2h_2h_3h_5^2 + 12h_1^2h_2h_4^2h_5 + 8h_2^2h_3h_4h_5^2)h_2h_1f\Big)
\end{aligned}
$$

$$
\begin{aligned}
\tilde{E}_2 \;=\; \frac{1}{C}\Big(&\frac{1}{24}h_2fh_3(-6h_3^3h_2h_5^2 - 8h_2^3h_4^2h_5 - 6h_2^3h_4^2h_6 - 6h_2^3h_3h_5^2\\
&-8h_2^3h_4h_5^2 - 8h_2^3h_4h_5h_6 - 6h_2^3h_5h_3h_6 - 8h_2^3h_3h_4h_5 - 6h_2^3h_3h_6h_4\\
&-4h_3^2h_5h_6h_4h_2 - 3h_3^2h_2h_4^2h_6 - 6h_3^3h_2h_6h_4 - 8h_3^3h_2h_4h_5 - 6h_3^3h_2h_5h_6\\
&-4h_3^2h_2h_4^2h_5 - 8h_4h_5h_1h_3^3 - 6h_6h_4h_1h_3^3 - 6h_5^2h_1h_3^3 - 6h_5h_6h_1h_3^3\\
&+4h_4^4h_5h_2 + 3h_6h_4^4h_2 + 3h_1^3h_3h_5h_6 + 4h_1^3h_3h_4h_5 - 3h_1h_3h_2^2h_5^2 + 4h_1^3h_4^2h_5\\
&+3h_1^3h_4^2h_6 + 3h_1^3h_3h_6h_4 + 4h_1^3h_4h_5^2 + 4h_1^3h_4h_5h_6 + 3h_1h_6h_4^4 - 4h_1h_4^2h_5h_3^2\\
&+2h_1h_5^2h_4^3 - 2h_1h_5^4h_4 + 2h_1h_5h_6h_4^3 - 3h_1h_6h_4^2h_3^2 - 3h_1h_3h_2^2h_6h_4\\
&+h_1h_5h_6^3h_4 - 4h_1h_5h_6h_4h_3^2 - h_1h_5^3h_6h_4 + 3h_1^3h_3h_5^2 - 4h_1h_2^2h_4h_5h_6\\
&-4h_1h_2^2h_4^2h_5 - 4h_1h_3h_2^2h_4h_5 - 3h_1h_2^2h_4^2h_6 + h_5h_6^3h_2h_4 - 4h_1h_2^2h_4h_5^2\\
&-3h_1h_3h_2^2h_5h_6 + 4h_1h_4^4h_5 + 2h_5^2h_2h_4^3 - h_5^3h_6h_2h_4 + 2h_5h_6h_2h_4^3\\
&-2h_5^4h_2h_4 - 4h_1h_3^2h_4h_5^2 - 4h_3^2h_2h_4h_5^2)\Big)
\end{aligned}
$$

$$
\begin{aligned}
\tilde{E}_3 \;=\; \frac{1}{C}\Big(&-\frac{1}{48}fh_3h_4(16h_3^3h_2h_5^2 + 8h_3^2h_2^2h_5^2 + 8h_3^2h_2^2h_4h_5 + 6h_3^2h_2^2h_6h_4\\
&+12h_3^3h_2h_6h_4 + 16h_3^3h_2h_4h_5 + 16h_3^3h_2h_5h_6 + 8h_3^2h_2^2h_5h_6 + 12h_4h_5h_1h_3^3\\
&+16h_4^3h_5h_3h_2 + 12h_4^3h_5h_1h_3 + 6h_6h_4h_2h_1h_3^2 + 9h_6h_4h_1h_3^3 + 9h_6h_4^3h_1h_3\\
&+12h_6h_4^3h_3h_2 - 6h_6h_4h_2^4 + 12h_6h_4^3h_2^2 + 3h_6h_4h_1^3h_2 - 3h_6h_4h_1h_2^3\\
&-6h_5^4h_1h_3 - 8h_4h_5h_2^4 + 16h_4^3h_5h_2^2 + 12h_5^2h_1h_3^3 + 8h_4h_5h_2h_1h_3^2\\
&+4h_4h_5h_1^3h_2 - 4h_4h_5h_1h_2^3 + 4h_5h_6^3h_2h_1 + 16h_4^3h_5h_2h_1 + 8h_5h_6h_2h_1h_3^2\\
&-8h_5^2h_2^4 - 8h_5^4h_3h_2 - 4h_5h_6h_1h_2^3 + 12h_6h_4^3h_2h_1 - 8h_5^4h_2^2 - 3h_5^3h_6h_1h_3\\
&+4h_5h_6h_1^3h_2 - 4h_5^3h_6h_3h_2 + 4h_5h_6^3h_3h_2 + 3h_5h_6^3h_1h_3 + 4h_5^2h_1^3h_2\\
&+8h_5h_6h_2^2h_4^2 - 8h_5h_6h_2^4 + 8h_5^2h_2h_1h_3^2 - 4h_5^2h_1h_2^3 + 4h_5h_6^3h_2^2\\
&+8h_5h_6h_3h_2h_4^2 - 4h_5^3h_6h_2^2 + 8h_5h_6h_2h_1h_4^2 + 12h_5h_6h_1h_3^3
\end{aligned}
$$

$$
\begin{aligned}
&+6h_5h_6h_1h_3h_4^2 - 8h_5^4h_2h_1 - 4h_5^3h_6h_2h_1 + 6h_1h_3h_4^2h_5^2 + 8h_3h_2h_4^2h_5^2 \\
&+8h_2^2h_4^2h_5^2 + 8h_2h_1h_4^2h_5^2)\Big)
\end{aligned}
$$

$$
\begin{aligned}
\tilde{E}_4 \;=\; \frac{1}{C}\Big(&-\frac{1}{48}h_6fh_5(-4h_2h_1h_3h_4^3 + 8h_3h_2h_4h_1h_5^2 + 9h_1h_3^2h_5^3 + 9h_1h_3^2h_5h_6^2 \\
&-2h_3h_2^4h_4 + 8h_2^2h_3h_4h_5^2 - h_1h_3h_4h_2^3 - 4h_3^2h_2h_4^3 + 16h_2^2h_3h_4h_6^2 \\
&-3h_1h_3^2h_4^3 + 3h_1h_3^4h_4 + h_1^3h_3h_4h_2 + 12h_1h_3h_4^2h_6^2 + 6h_1h_3h_4^2h_5^2 \\
&-6h_1h_3h_4^4 + 12h_1h_3h_4h_5^3 + 12h_1h_3h_4h_5h_6^2 + 12h_1h_3^2h_4h_6^2 + 6h_1h_3^2h_4h_5^2 \\
&+12h_3^2h_2h_5^3 + 12h_3^2h_2h_5h_6^2 + 4h_3^4h_2h_4 + 16h_3h_2h_4^2h_6^2 + 8h_3h_2h_4^2h_5^2 \\
&-8h_3h_2h_4^4 + 16h_3h_2h_4h_5^3 + 16h_3h_2h_4h_5h_6^2 + 16h_3^2h_2h_4h_6^2 + 8h_3^2h_2h_4h_5^2 \\
&+12h_2^2h_3h_5^3 + 12h_2^2h_3h_5h_6^2 - 4h_2^2h_3h_4^3 + 2h_2^2h_4h_3^3 + 12h_2^2h_4^2h_6^2 \\
&+6h_2^2h_4^2h_5^2 - 6h_2^2h_4^4 + 12h_2^2h_4h_5^3 + 12h_2^2h_4h_5h_6^2 + 12h_2h_1h_3h_5^3 \\
&+12h_2h_1h_3h_5h_6^2 + 2h_2h_1h_4h_3^3 + 12h_2h_1h_4^2h_6^2 + 6h_2h_1h_4^2h_5^2 - 6h_2h_1h_4^4 \\
&+12h_2h_1h_4h_5^3 + 12h_2h_1h_4h_5h_6^2 + 16h_2h_1h_3h_4h_6^2)\Big)
\end{aligned}
$$

$$
\begin{aligned}
\tilde{E}_5 \;=\; \frac{1}{C}\Big(&\frac{1}{24}fh_5h_4(-4h_3h_2^2h_4^2h_5 - 4h_3h_2^2h_4^2h_6 - 4h_3^2h_2h_4^2h_6 - 4h_3^2h_2h_4^2h_5 \\
&-8h_4^3h_5h_3h_2 - 6h_4^3h_5h_1h_3 - 6h_6h_4^3h_1h_3 - 8h_6h_4^3h_3h_2 - 6h_6h_4^3h_2^2 \\
&-6h_4^3h_5h_2^2 - 6h_4^3h_5h_2h_1 - 6h_6h_4^3h_2h_1 - 3h_1h_4^2h_5h_3^2 - 3h_1h_6h_4^2h_3^2 \\
&+3h_2^2h_4h_6^3 - 3h_6h_2^2h_4h_5^2 - 4h_5h_2h_1h_3h_4^2 + 3h_1h_3h_4h_6^3 + 3h_1h_3^2h_6^3 \\
&+4h_3h_2h_4h_6^3 + 4h_2^2h_3h_6^3 + h_1^3h_2h_3h_6 + h_1^3h_2h_3h_5 + 2h_5h_2h_1h_3^3 \\
&-4h_6h_3h_2h_4h_5^2 - 4h_6h_3h_2h_1h_5^2 - 4h_6h_2^2h_3h_5^2 - 4h_6h_2h_1h_3h_4^2 \\
&+4h_2h_1h_3h_6^3 - h_1h_2^3h_3h_6 + 3h_1h_3^4h_6 + 3h_1h_3^4h_5 + 2h_2^2h_3^3h_5 - h_1h_2^3h_3h_5 \\
&+4h_3^2h_2h_6^3 + 3h_2h_1h_4h_6^3 - 2h_2^4h_3h_6 - 2h_2^4h_3h_5 + 4h_3^4h_2h_5 + 4h_3^4h_2h_6 \\
&+2h_2^2h_3^3h_6 + 2h_6h_2h_1h_3^3 - 3h_6h_2h_1h_4h_5^2 - 4h_6h_3^2h_2h_5^2 - 3h_6h_1h_3h_4h_5^2 \\
&-3h_6h_1h_3^2h_5^2 - 6h_1h_3^2h_5^3 - 6h_1h_3h_4h_5^3 - 8h_3^2h_2h_5^3 - 8h_3h_2h_4h_5^3 \\
&-8h_2^2h_3h_5^3 - 6h_2^2h_4h_5^3 - 8h_2h_1h_3h_5^3 - 6h_2h_1h_4h_5^3)\Big)
\end{aligned}
$$

mit

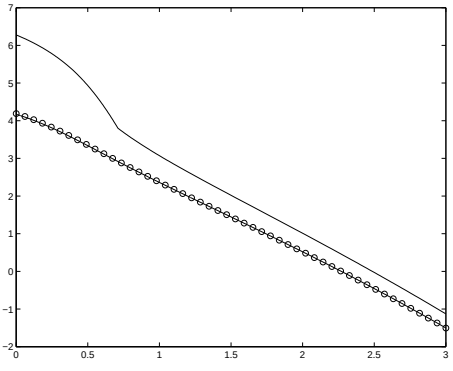
$$
\begin{aligned}
C \;=\; &12h_2^2h_4^2h_5 + 12h_3^2h_2h_5^2 + 12h_2^2h_3h_5^2 + 9h_2^2h_4^2h_6 + 12h_2^2h_4h_5^2 + 9h_1h_3^2h_5^2 \\
&+16h_3h_5h_6h_4h_2 + 12h_2^2h_3h_6h_4 + 12h_3^2h_2h_6h_4 + 12h_3h_2h_4^2h_6 + 16h_3h_2h_4h_5^2 \\
&+16h_3^2h_2h_4h_5 + 12h_2^2h_5h_3h_6 + 16h_3h_2h_4^2h_5 + 12h_3^2h_2h_5h_6 + 12h_2^2h_4h_5h_6 \\
&+16h_2^2h_3h_4h_5 + 9h_1h_3h_4^2h_6 + 9h_1h_3^2h_6h_4 + 12h_1h_3h_4^2h_5 + 9h_1h_3^2h_5h_6
\end{aligned}
$$

$$+12h_1h_3h_4h_5^2 + 12h_1h_3^2h_4h_5 + 12h_2h_1h_3h_5^2 + 9h_2h_1h_4^2h_6 + 12h_2h_1h_4h_5^2$$
$$+12h_2h_1h_4^2h_5 + 12h_1h_3h_4h_5h_6 + 12h_2h_1h_5h_3h_6 + 12h_2h_1h_4h_5h_6$$
$$+12h_2h_1h_3h_6h_4 + 16h_2h_1h_3h_4h_5)$$

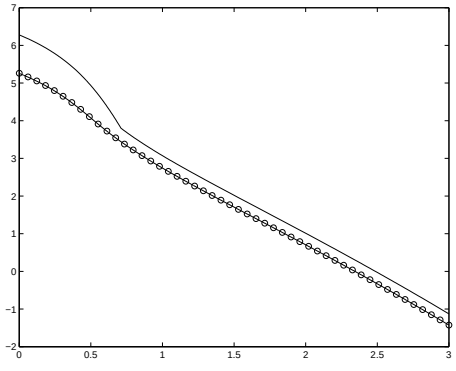
Wird nun eine Intervalllänge halbiert, so entstehen ähnliche Ausdrücke für die neuen Werte $\hat{E}_i$, deren Zusammenhang mit den alten Werten $\tilde{E}_i$ nicht direkt auszumachen ist.
Es lässt sich somit auf diesem Weg keine allgemeine Aussage über die Auswirkung einer Intervallverfeinerung auf die Werte $\tilde{E}_i = s'(x_i) - f'(x_i)$ machen.
Hier soll nicht weiter auf eine mögliche Abschätzung des Verhältnisses zwischen $\tilde{E}_i$ und $\hat{E}_i$ eingegangen werden. Dies würde den Rahmen sprengen.

A.2 Entwicklung der Optimalwertfunktion zu Beispiel 2

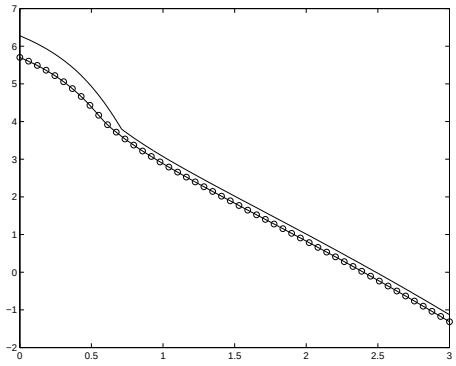
Im Laufe der Betrachtungen zu Beispiel 2 (Ökologieproblem) wird anhand von Beispielgraphen die Entwicklung der errechneten Approximation der Optimalwertfunktion (ausgehend von einem Startgitter mit 50 Knoten) aufgezeigt. Um die Entstehung der Oszillation deutlicher erkennbar zu machen, folgt die genaue Entwicklung der Approximation. In der Phase, in der sich die berechnete Funktion noch der Gauß-Seidel-Lösung annähert wird das Ergebnis nach jeweils 100 Iterationen gezeigt, anschließend nach jeweils 50 Iterationen.
Die Graphen sind wieder identisch aufgebaut wie schon in Kapitel 6, d.h als Vergleich ist die Lösung des kontrollierten Gauß-Seidel-Verfahrens eingezeichnet. Die Werte der Approximation an den Knoten sind wiederum durch Kreise gekennzeichnet.



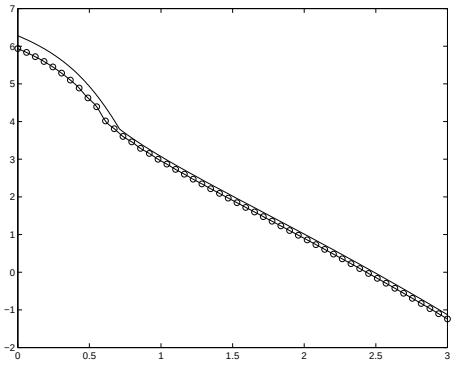
(a) nach 100 Iterationen



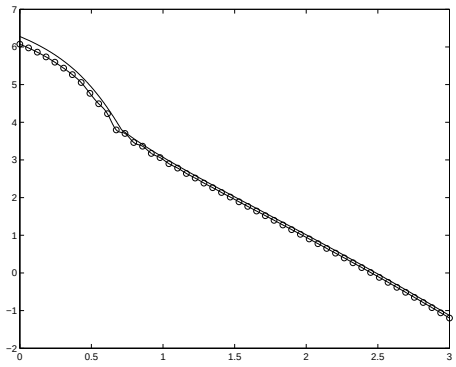
(b) nach 200 Iterationen



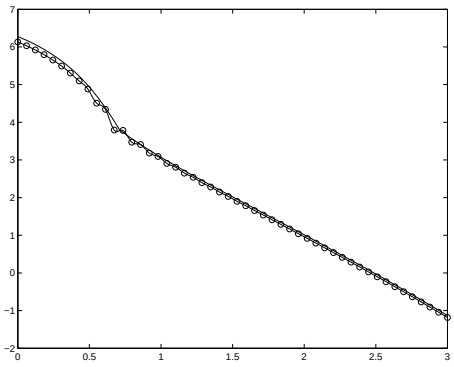
(c) nach 300 Iterationen



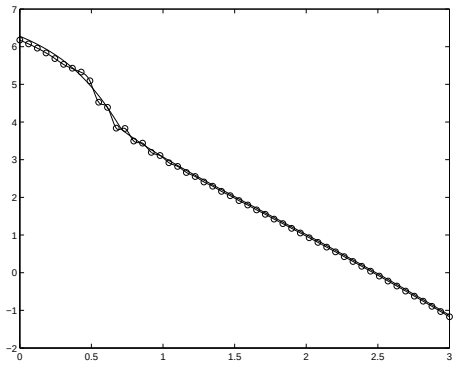
(d) nach 400 Iterationen



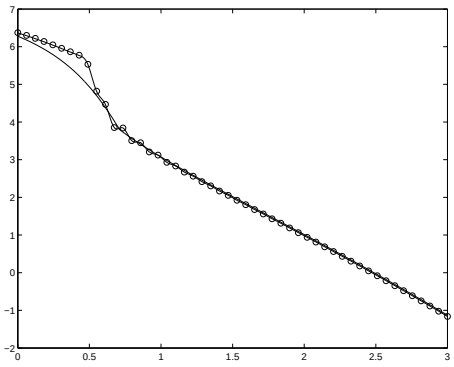
(e) nach 500 Iterationen



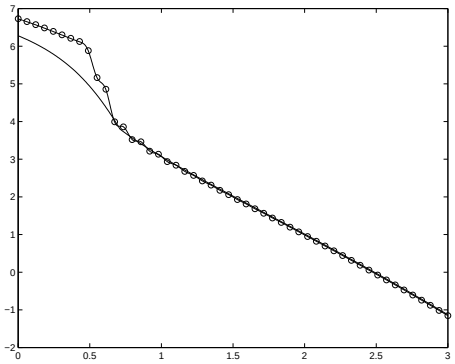
(f) nach 550 Iterationen



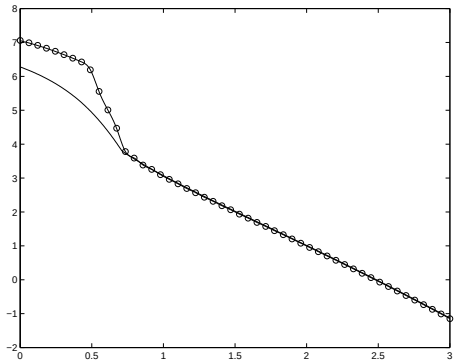
(g) nach 600 Iterationen



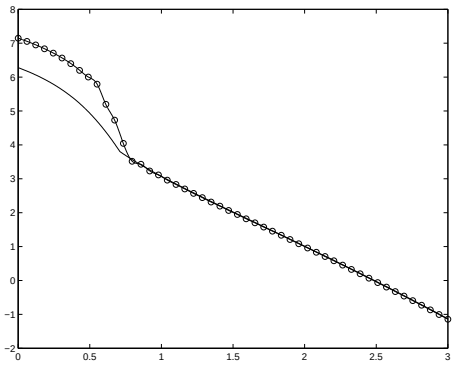
(h) nach 650 Iterationen



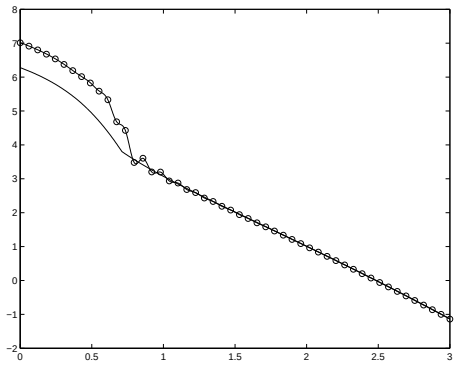
(i) nach 700 Iterationen



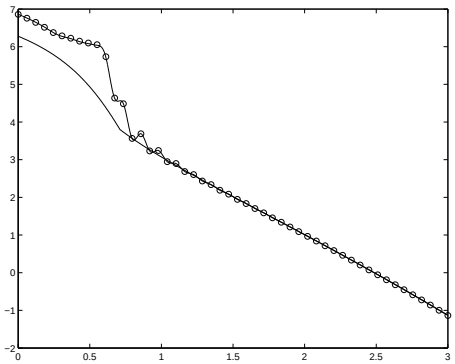
(j) nach 750 Iterationen



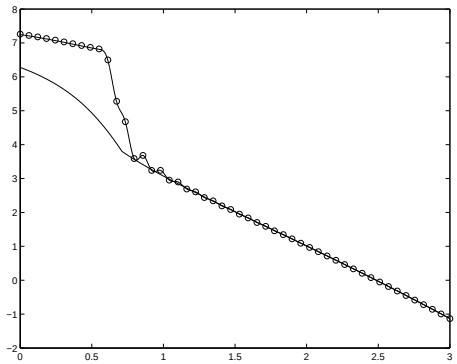
(k) nach 800 Iterationen



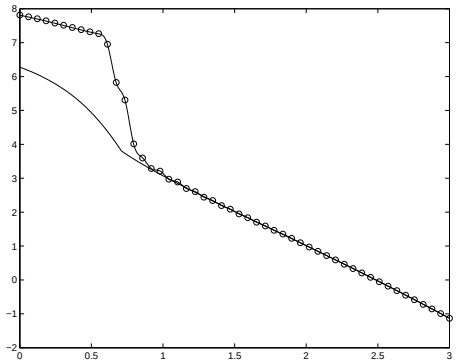
(l) nach 850 Iterationen



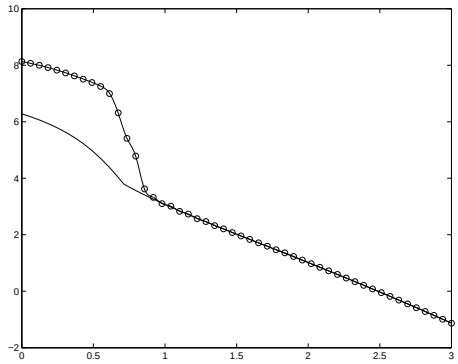
(m) nach 900 Iterationen



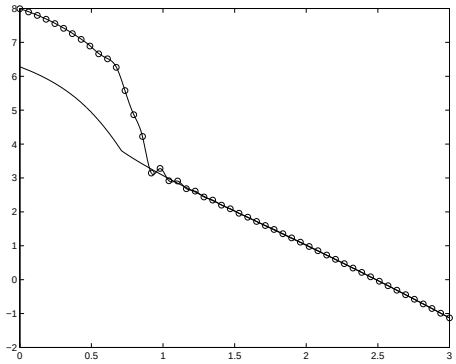
(n) nach 950 Iterationen



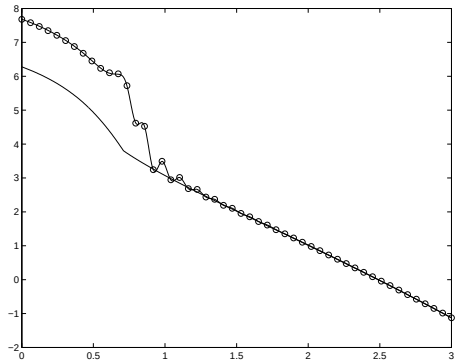
(o) nach 1000 Iterationen



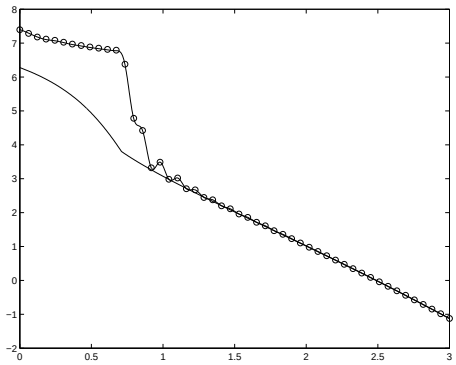
(p) nach 1050 Iterationen



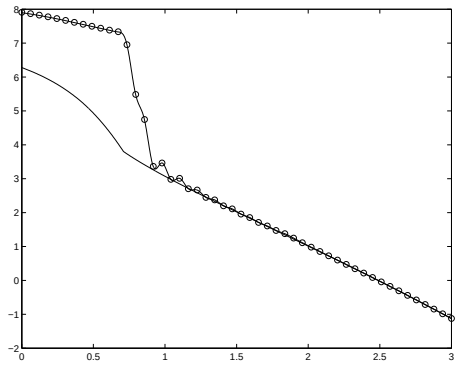
(q) nach 1100 Iterationen



(r) nach 1150 Iterationen

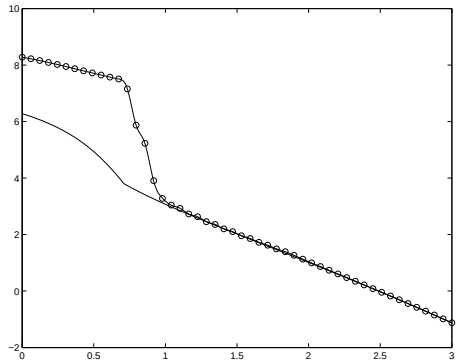


(s) nach 1200 Iterationen

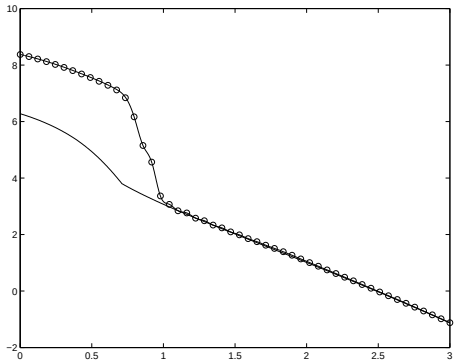


(t) nach 1250 Iterationen

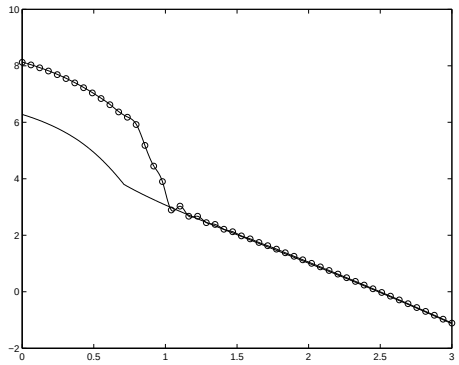
Abbildung A.1: Genaue Entwicklung der optimalen Wertefunktion zu Beispiel 2, Iterationen 0 bis 1250



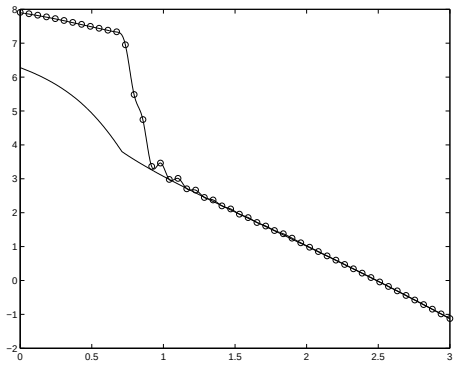
(a) nach 1300 Iterationen



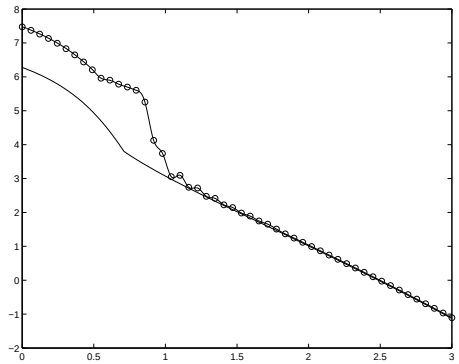
(b) nach 1350 Iterationen



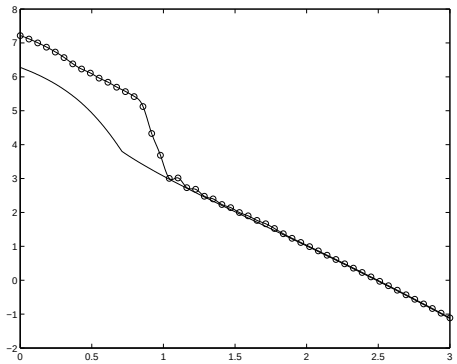
(c) nach 1400 Iterationen



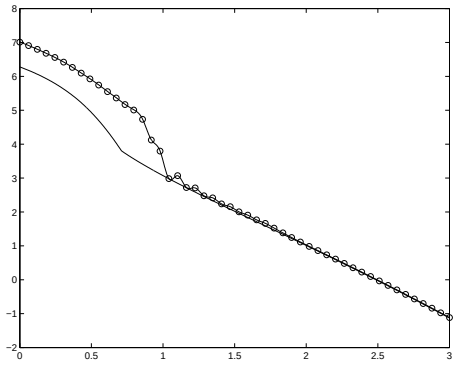
(d) nach 1450 Iterationen



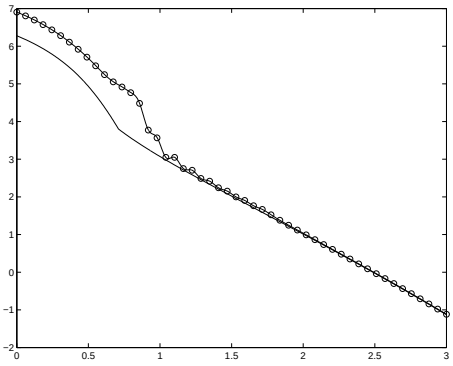
(e) nach 1500 Iterationen



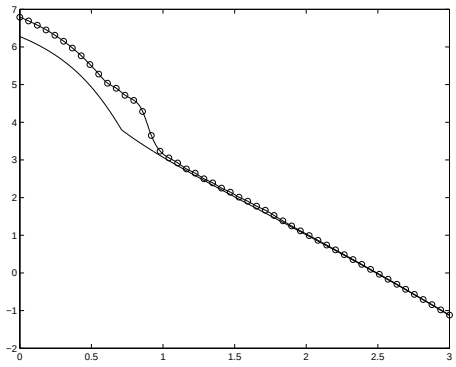
(f) nach 1550 Iterationen



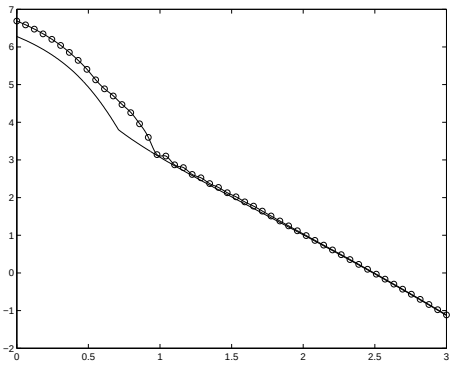
(g) nach 1600 Iterationen



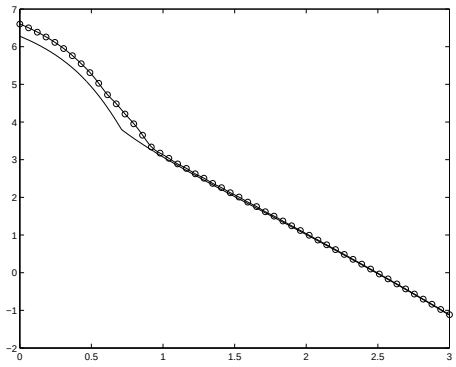
(h) nach 1650 Iterationen



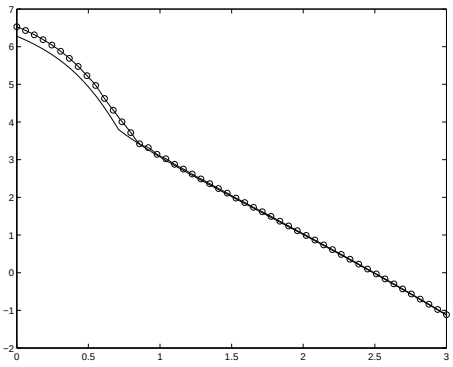
(i) nach 1700 Iterationen



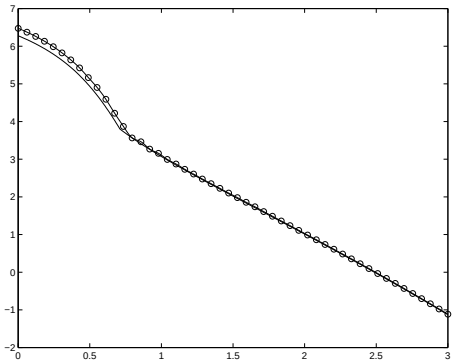
(j) nach 1750 Iterationen



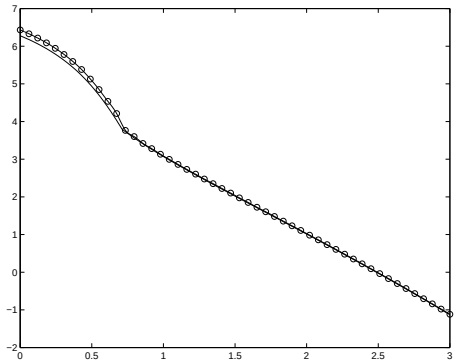
(k) nach 1800 Iterationen



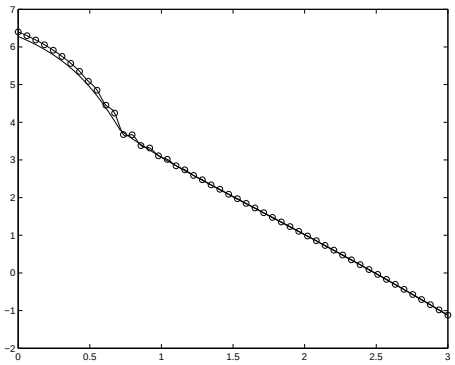
(l) nach 1850 Iterationen



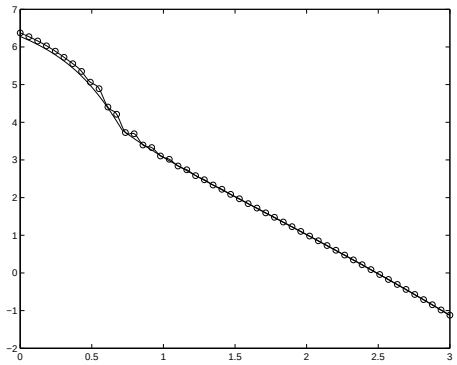
(m) nach 1900 Iterationen



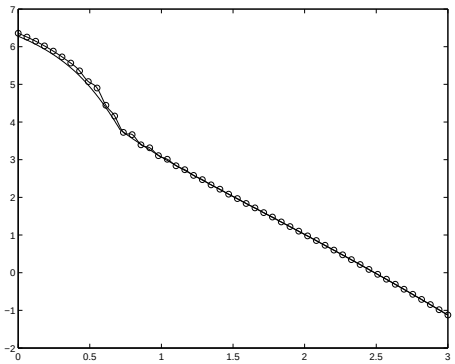
(n) nach 1950 Iterationen



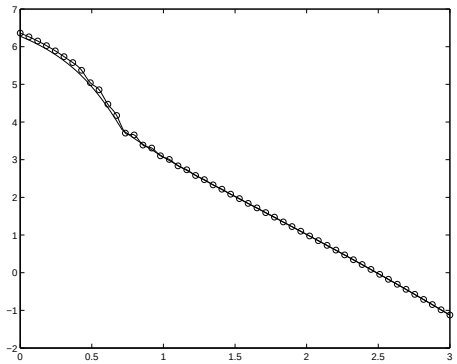
(o) nach 2000 Iterationen



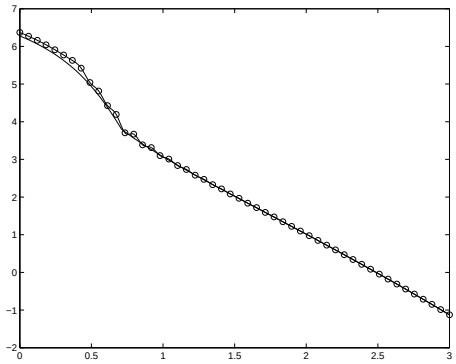
(p) nach 2050 Iterationen



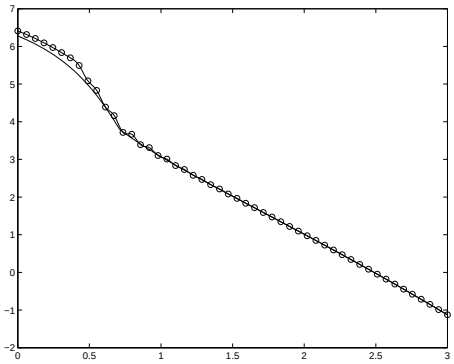
(q) nach 2100 Iterationen



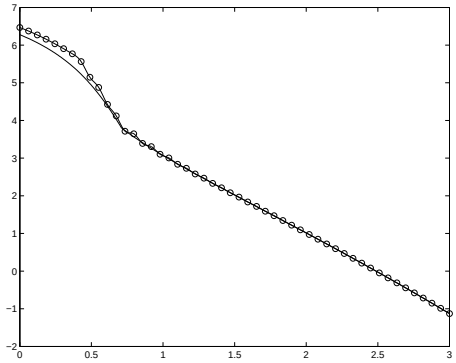
(r) nach 2150 Iterationen



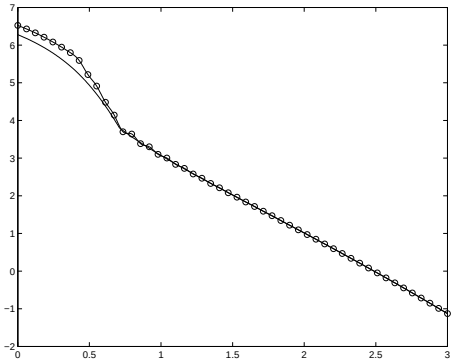
(s) nach 2200 Iterationen



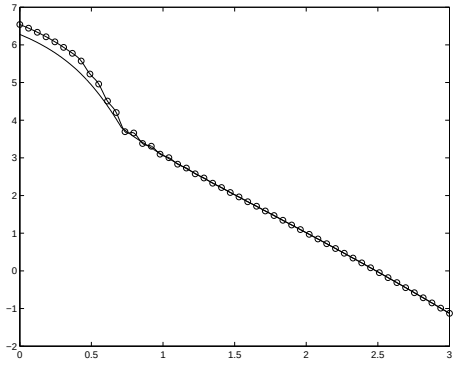
(t) nach 2250 Iterationen



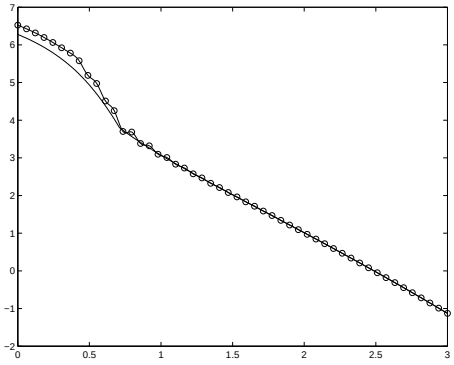
(u) nach 2300 Iterationen



(v) nach 2350 Iterationen

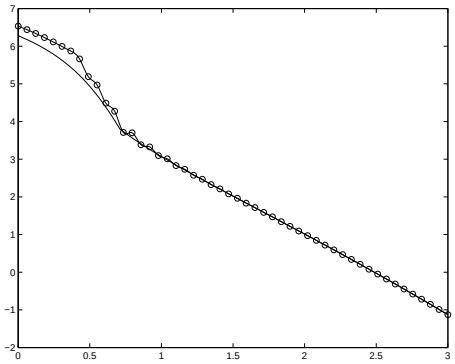


(w) nach 2400 Iterationen

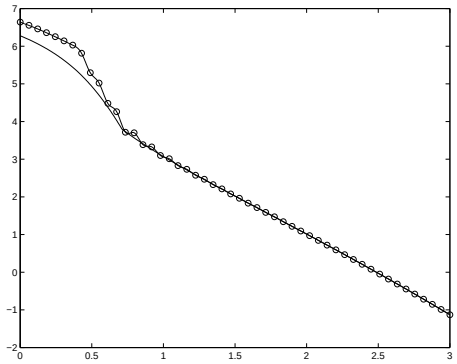


(x) nach 2450 Iterationen

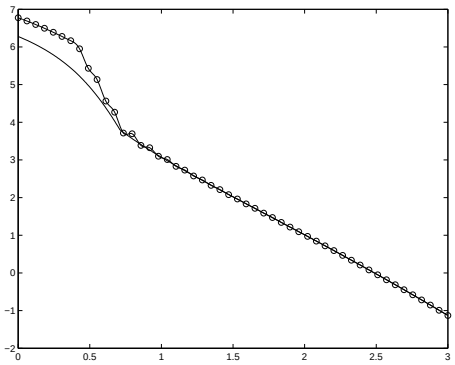
Abbildung A.2: Genaue Entwicklung der optimalen Wertefunktion zu Beispiel 2, Iterationen 1250 bis 2450



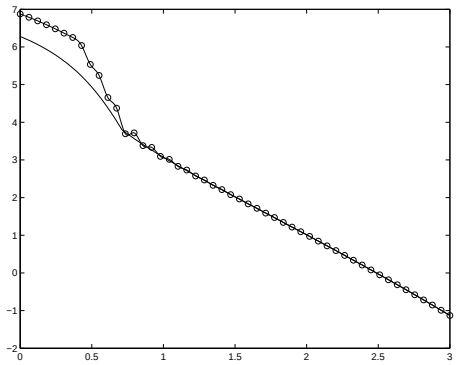
(a) nach 2500 Iterationen



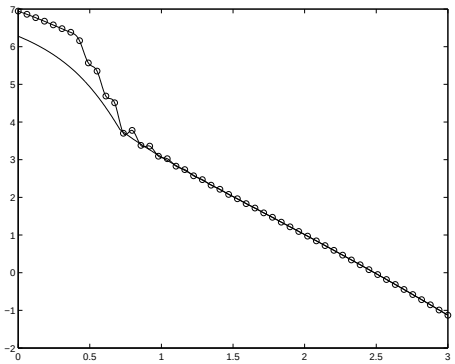
(b) nach 2550 Iterationen



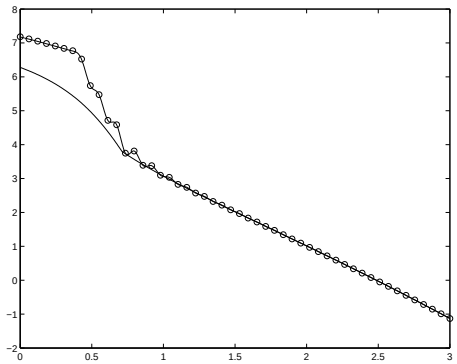
(c) nach 2600 Iterationen



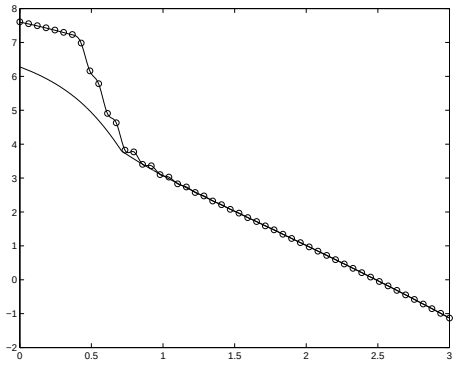
(d) nach 2650 Iterationen



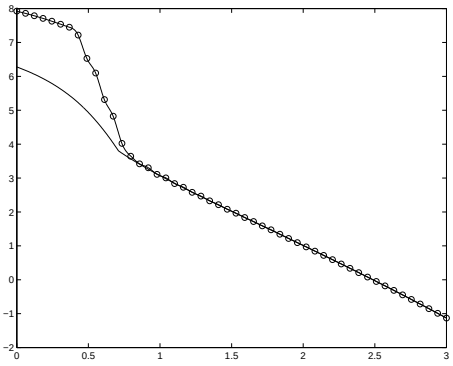
(e) nach 2700 Iterationen



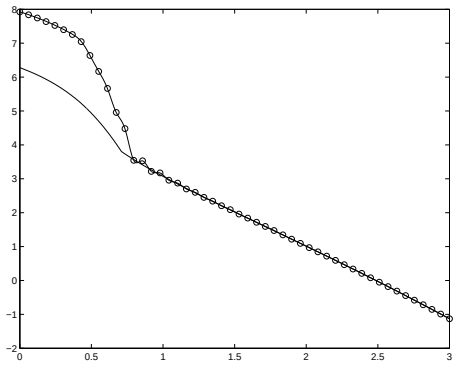
(f) nach 2750 Iterationen



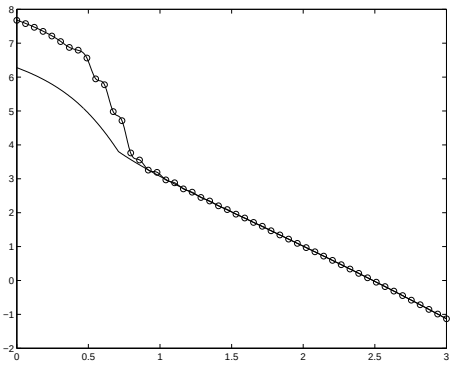
(g) nach 2800 Iterationen



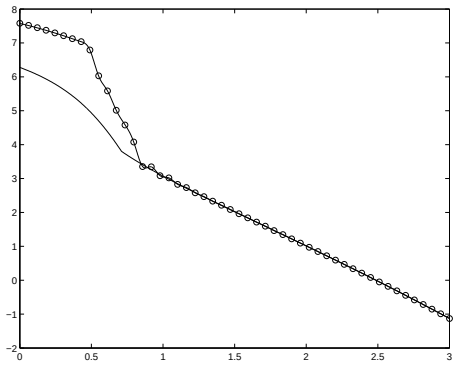
(h) nach 2850 Iterationen



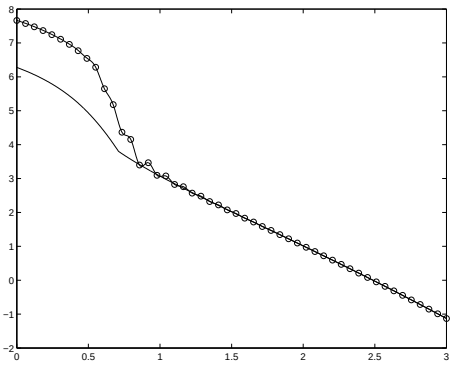
(i) nach 2900 Iterationen



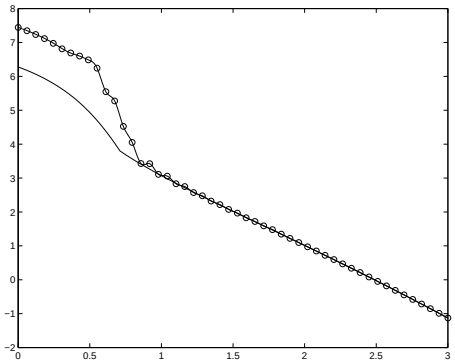
(j) nach 2950 Iterationen



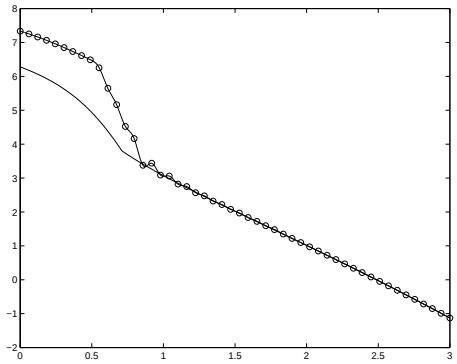
(k) nach 3000 Iterationen



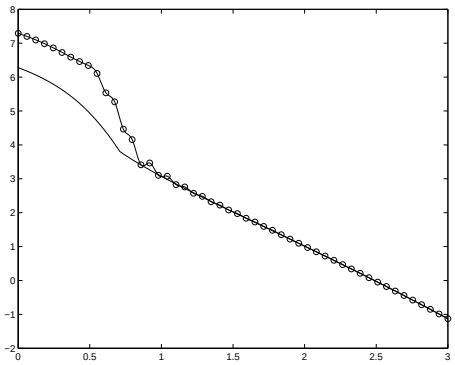
(l) nach 3050 Iterationen



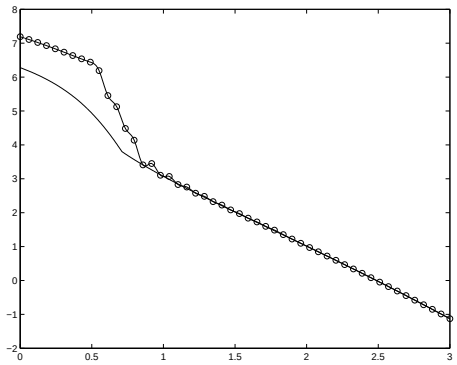
(m) nach 3100 Iterationen



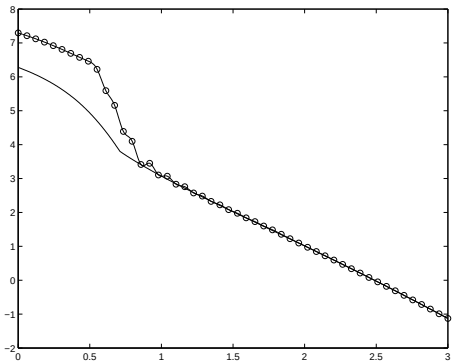
(n) nach 3150 Iterationen



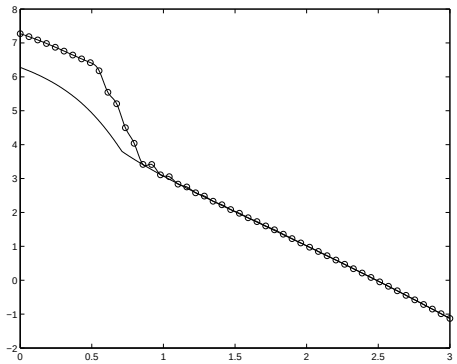
(o) nach 3200 Iterationen



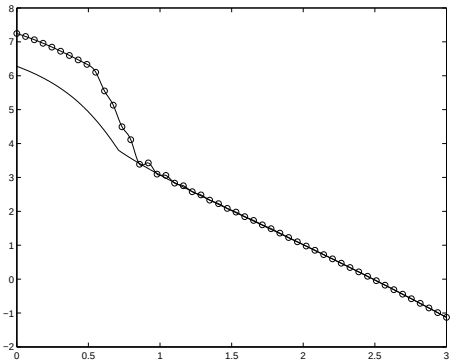
(p) nach 3250 Iterationen



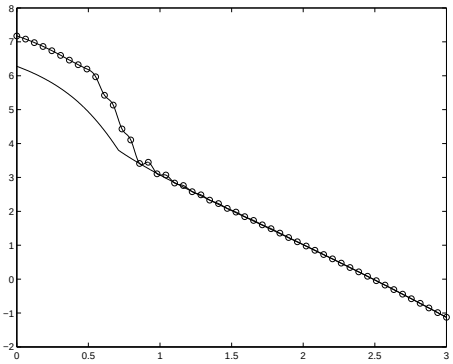
(q) nach 3300 Iterationen



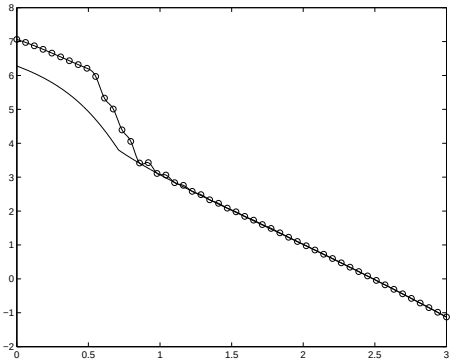
(r) nach 3350 Iterationen



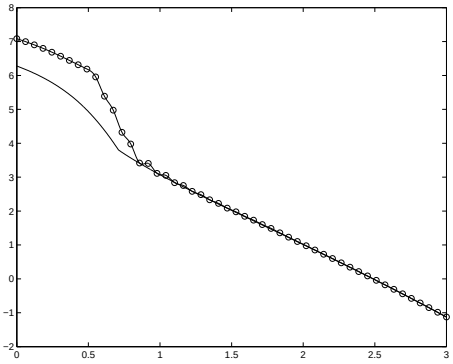
(s) nach 3400 Iterationen



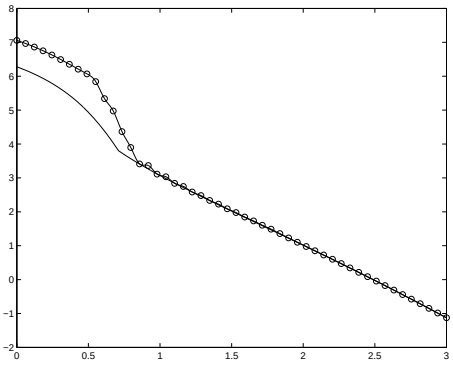
(t) nach 3450 Iterationen



(u) nach 3500 Iterationen



(v) nach 3550 Iterationen



(w) nach 3600 Iterationen

Abbildung A.3: Genaue Entwicklung der optimalen Wertefunktion zu Beispiel 2, Iterationen 2450 bis 3600

Anhang B

MATLAB-Quelltexte

Im Folgenden ist der Quelltext des verwendeten MATLAB-Programms aufgeführt, das zur Bestimmung der optimalen Wertefunktion eindimensionaler optimaler Steuerungsprobleme gemäß des oben beschriebenen Verfahrens mit Spline -Interpolation verwendet worden ist.

B.1 Quelltext des Hauptprogramms

Dieser Abschnitt enthält den vollständigen Quelltext der Funktion *'aufruf.m'*, welche die Parameter festlegt und das Iterationsverfahren aufruft, sowie die Implementierung des Iterationsverfahrens mit Spline-Interpolation in der Funktion *'verf_mit_spline.m'*.

Um die Implementierung für den Leser verständlicher zu machen, ist jeder Schritt ausführlich kommentiert. Umlaute sind in der Form 'ae' und 'ß' ist als 'ss' geschrieben.
Es werden weitestgehend die selben Bezeichnungen verwendet, die auch in den obigen Kapiteln benutzt worden sind.

Quelltext aufruf.m:

```
function aufruf(problem)
% aufruf(problem) legt die Variablen und Parameter fuer das
% entsprechende Problem fest und ruft das Iterationsverfahren
% fuer das entsprechende Problem auf.

% Die Variable problem ist entweder eines der Probleme
% 'wm'    =   Wachstumsmodell                    (Beispiel 1)
% 'oek'   =   Oekologieproblem                   (Beispiel 2)
% 'im'    =   Investitionsmodell                 (Beispiel 3)
% 'kred'  =   Wachstumsmodell mit Kreditaufnahme  (Beispiel 4)
```

```
% oder ein neues Problem 'neu', das durch die Funktionen fh_neu
% und g_neu definiert ist.


tic;              % Start der Messung der Rechenzeit

switch problem case 'wm'
    fh          =   @fh_wm;
    g           =   @g_wm;
    % Aufruf der das Problem bestimmenden Funktionen;
    % als M-file unter fh_. bzw g_. abgespeichert.

    startknoten =   17;
    % Anzahl der Knoten des Ausgangsgitters
    x_1         =   0.1;
    % linke Grenze des zu betrachtenden Bereichs [x_1,x_n]
    x_n         =   10;
    % rechte Grenze des zu betrachtenden Bereichs [x_1,x_n]
    x           =   x_1:((x_n-x_1)/(startknoten-1)):x_n;
    % Speicherung aller Knoten des Ausgangsgitters im Vektor x

    kontrollen  =   500;
    % Anzahl der Kontrollwerte aus der Diskretisierung von U
    u_1         =   0.1;
    % linke Grenze des Kontrollwertebereichs U
    u_m         =   11;
    % rechte Grenze des Kontrollwertebereichs U
    u           =   u_1:((u_m-u_1)/(kontrollen-1)):u_m;
    % Speicherung aller Kontrollwerte im Vektor u

    h           =   1;
    % Schrittweite der zeitlichen Diskretisierung
    delta       =   0.05;
    % Diskontrate

    zeta        =   1e-5;
    % Abbruchparamter; bestimmt, ab wann die Approximation auf dem
    % betrachteten Gitter genau genug ist

    theta       =   0.1;
    % Verfeinerungsparamter; bestimmt, welche Intervalle bei der
    % Gitteradaption verfeinert werden
```

```
    grenze_kn   =   300;
    % maximale Knotenzahl; hat das betrachtete Gitter mehr Knoten,
    % wird nicht mehr verfeinert

    x_vgl       =   x_1:((x_n-x_1)/(1000-1)):x_n;
    v_vgl       =   exakt_wm(x_vgl);
    % Die bekannten exakten Werte dienen als Vergleichswerte.

case 'oek'      % Variablen analog zu oben
    fh          =   @fh_oek;
    g           =   @g_oek;

    startknoten =   50;
    x_1         =   0;
    x_n         =   3;
    x           =   x_1:((x_n-x_1)/(startknoten-1)):x_n;

    kontrollen  =   50;
    u_1         =   0;
    u_m         =   0.4;
    u           =   u_1:((u_m-u_1)/(kontrollen-1)):u_m;

    h           =   0.05;
    delta       =   0.1;

    zeta        =   1e-5;

    theta       =   0.1;

    grenze_kn   =   225;

    [x_vgl, v_vgl]=loesung_linear('oek');
    % Die Loesungen, die mit kontrolliertem Gauss-Seidel-Verfahren
    % errechnet wurden, werden eingelesen. Sie dienen hier als
    % Vergleichswerte.

case 'im'           % Variablen analog zu oben
    fh          =   @fh_im;
    g           =   @g_im;

    startknoten =   131;
    x_1         =   0;
```

```
    x_n          =   0.7;
    x            =   x_1:((x_n-x_1)/(startknoten-1)):x_n;

    kontrollen   =   50;
    u_1          =   0;
    u_m          =   1;
    u            =   u_1:((u_m-u_1)/(kontrollen-1)):u_m;

    h            =   0.05;
    delta        =   0.2;

    zeta         =   1e-5;

    theta        =   0.3;

    grenze_kn    =   307;

    [x_vgl, v_vgl]=loesung_linear('im');

case 'kred'
    fh           =   @fh_kred;
    g            =   @g_kred;

    startknoten  =   39;
    x_1          =   0.0001;
    x_n          =   2.0001;
    x            =   x_1:((x_n-x_1)/(startknoten-1)):x_n;

    kontrollen   =   50;
    u_1          =   0;
    u_m          =   0.25;
    u            =   u_1:((u_m-u_1)/(kontrollen-1)):u_m;

    h            =   0.05;
    delta        =   0.1;

    zeta         =   1e-5;

    theta        =   0.5;

    grenze_kn    =   116;

    [x_vgl, v_vgl]=loesung_linear('kred');
```

```
case 'neu'  % Hier koennen die Daten fuer ein weiteres Problem
            % abgespeichert werden.
    fh          =   @fh_neu;
    g           =   @g_neu;

    startknoten =   1;
    x_1         =   0;
    x_n         =   1;
    x           =   x_1:((x_n-x_1)/(startknoten-1)):x_n;

    kontrollen  =   1;
    u_1         =   0;
    u_m         =   1;
    u           =   u_1:((u_m-u_1)/(kontrollen-1)):u_m;

    h           =   0.1;
    delta       =   0.1;

    zeta        =   1e-5;

    theta       =   0.1;

    grenze_kn   =   11;

    [x_vgl, v_vgl]=loesung_linear('neu');

otherwise
    fprintf('Unbekanntes Problem!');
    break;
end

% Aufruf des Iterationsverfahrens mit den festgelegten
% Parametern und Funktionen

beta    =   1-delta*h;          % Diskontfaktor wird berechnet

verf_mit_spline(fh,g,x,u,h,delta,beta,zeta,theta,grenze_kn,\\
x_vgl,v_vgl,problem);

toc                             % Ausgabe der gesamten Rechenzeit
```

Quelltext verf_mit_spline.m:

```
function verf_mit_spline(fh,g,x,u,h,delta,beta,zeta,theta,
grenze_kn,x_vgl,v_vgl,problem);

% Die Funktion 'verf_mit_spline.m' berechnet Optimalwertfunktion
% eines eindimensionalen optimalen Steuerungsproblems mit
% der MATLAB-Spline-Interpolationsroutine auf adaptiven Gittern.
%
% In jeder Iteration wird die maximale Differenz zwischen der
% aktuellen und der vorherigen Approximation sowie die Anzahl
% der momentanen Knoten ausgegeben. Ausserdem wird nach jeder
% Hauptiteration (d.h. auf dem aktuellen Gitter ist die optimale
% Wertefunktion bestimmt) eine obere Grenze fuer den Fehler auf
% dem gesamten betrachteten Bereich [x_1,x_n] angezeigt.

z_iter  =   0;
% Iterationszaehler wird auf Null gesetzt.


% FESTLEGUNG DER VOM BETRACHTETEN PROBLEM UNABHAENGIGEN PARAMETER

grenze_iter     =   4000;
% Sind grenze_iter Iterationen durchlaufen, wird abgebrochen
% (noetig bei periodischer Entwicklung der Optimalwertfunktion)

plotintervall   =   200;
% Gibt an, nach wie vielen Iterationen die momentane
% Optimalwertfunktion jeweils geplottet werden soll.


% VORBESETZUNG DER VEKTOREN

 ff   = zeros(length(x),length(u));
 gg   = zeros(length(x),length(u));
 uopt = ones(1,length(x));
 v    = zeros(length(x),1);

abbruch = 0;
% Wenn Variable abbruch=1, wird der Algorithmus beendet
```

```
% DURCHLAUFE SCHLEIFE SO LANGE, BIS EIN ABBRUCHKRITERIUM ERFUELLT IST

while(abbruch==0)
    for i=1:length(x)
        for k=1:length(u)
            ff(i,k) = feval(fh,x(i),u(k),h);
            gg(i,k) = feval(g,x(i),u(k));
        end
    end
% Berechnung von fh an allen Knotenpunkten fuer alle Kontrollwerte
% Berechnung von g an allen Knotenpunkten fuer alle Kontrollwerte

    zetamax = 2*zeta;

    while(zetamax>zeta)
    % Durchlaufe Schleife so lange, bis der maximale Schritt zetamax
    % in einer Iteration klein genug ist, d.h. bis
    % das Ergebnis auf dem betrachteten Gitter genau genug ist

       z_iter=z_iter+1;
       % Erhoehe Iterationszaehler

       ss=spline(x,v);
       % Berechnung des Splines zu den Werten v_i an den Knoten x_i

       v2=v;
       % Speicherung zur Bestimmung der Werte zeta

           for i=1:length(x)
              for k=1:length(u)
                  if ff(i,k)>=x(1) & ff(i,k)<=x(length(x))
                     % Es werden nur die Knoten x betrachtet, bei
                     % denen fh(x,u) noch im Bereich [x_1,x_n] liegt

                     ss2(k)=ppval(ss,ff(i,k));
                     % Auswertung der Splines an den Werten von ff

                     t(k)=h*gg(i,k)+beta*ss2(k);
                     % Berechnung der Vergleichswerte, aus denen
                     % anschliessend das Maximum und so der neue
                     % Wert v_i am Knoten x_i berechnet wird.
                     % Sie werden im Vektor t abgespeichert.
```

```
            else
                t(k)=-1e+5;
                % Fuer alle u_k, fuer die der Wert fh(x_i,u_k)
                % nicht im Bereich [x_1,x_n] liegt, wird der
                % Vergleichswert so  besetzt,dass er bei der
                % Maximierung keine Rolle spielt.
            end
        end

        v(i)=max(t);
        % Bestimmung des neuen Wertes v_i am Knoten x_i
        % durch Maximierung

        for j=1:length(t)
            if t(j)==max(t)
               uopt(i)=u(j);
            end
        end
        % Speicherung der optimalen Kontrollwerte zu
        % jedem Knoten x_i im Vektor uopt.
        clear t;
        clear ss2;
    end;

zeta2=v-v2;
% Berechnung der "Schritte" zeta_i in der letzten Iteration,
% abgespeichert im Vektor zeta2.

zetamax=max(abs(zeta2));
% Bestimmung des Betrags-Maximums von zeta2.
% Es zeigt an, ob die Approximation auf dem betrachteten
% Gitter genau genug ist.

fprintf('\nmax. Schritt: %f  %d  %d',zetamax,length(x),z_iter);
% Ausgabe maximaler Schritt, Knotenzahl, Iterationsanzahl

maxschritt(z_iter)=zetamax;
% Abspeichern des Wertes zetamax im Vektor maxschritt
% zur spaeteren Veranschaulichung der Entwicklung
% im Laufe der Berechnungen
```

```
    if z_iter>=grenze_iter
        grenze_kn=1;
        break;
    end
    % Sind grenze_iter Iterationen bereits durchlaufen wird der
    % Algorithmus beendet. Dazu wird die Begrenzung der Knotenzahl
    % grenze_knoten auf 1 gesetzt und die while-Schleife verlassen.

    if problem(1:2)=='wm'
         if mod(z_iter,plotintervall)==0
             xhilf2 = x(1):(x(length(x))-x(1))/3000:x(length(x));
             for i=1:length(xhilf2)
                     vhilf2(i)=ppval(ss,xhilf2(i));
             end
             figure;
             plot(xhilf2,vhilf2,x,v,'o',x_vgl,v_vgl);
             title('Entwicklung der Approximation');
         end
    else if mod(z_iter,plotintervall)==0
             xhilf2 = x(1):(x(length(x))-x(1))/3000:x(length(x));
             for i=1:length(xhilf2)
                     vhilf2(i)=ppval(ss,xhilf2(i));
             end
             figure;
             plot(xhilf2,vhilf2,x,v,'o',x_vgl, v_vgl,'r');
             title('Entwicklung der Approximation');
         end
    end
    % Nach jeweils "plotintervall" Iterationen wird
    % die momentane Approximation der optimalen Wertefunktion
    % geplottet. Um die Form des Splines deutlicher darzustellen,
    % werden 3000 Hilfspunkte erzeugt und der Spline an diesen
    % ausgewertet. Zusaetzlich wird zum Vergleich die mit linearer
    % Interpolation ermittelte Optimalwertfunktion ausgegeben.
    % Ist die exakte optimale Wertefunktion bekannt
    % (hier bei Beispiel 1) wird die exakte Funktion als
    % Vergleichsfunktion eingezeichnet. Der Wert v_i an den
    % Knoten x_i wird zusaetzlich als Kreis markiert.
    % Durch diese regelmaessige Ausgabe kann die Entwicklung der
    % Approximationen nachvollzogen werden.

end         % Ende der inneren While-Schleife
```

```
figure;
plot(x,v,x_vgl, v_vgl,'r');
title('Approximation vor Gitteradaption');
% Die auf dem betrachteten Gitter ermittelte Optimalwertfunktion
% wird ausgegeben. Wieder zum Vergleich zusammen mit der
% mit lin. Interpolation ermittelten Approximation.

abbruch=1;
% Den Abbruch bestimmende Variable wird auf eins (=Abbruch)
% gesetzt.

if problem(1:2)=='wm'
    % Ist die exakte Optimalwertfunktion bekannt (in dieser
    % Arbeit bei Beispiel 1), wird die Approximation mit dieser
    % verglichen. Das geschieht an 1000 Hilfspunkten des
    % Vektors x_vgl, der durch die Aufruffunktion mit
    % uebergeben wird.

    ss=spline(x,v);
    % Bestimmung des Splines mit den neuen Werten v_i.

    z2=1;            % Hilfszaehler

    for i=1:length(x_vgl)
        vhilf3(i)=ppval(ss,x_vgl(i));
    end
    % Auswertung des Splines an den 1000 Hilfspunkten
    % des Vektors x_vgl (aus 'aufruf.m').
    % Die Werte werden abgespeichert im Vektor vhilf3.

    maxfehl(z2)=max(abs(vhilf3-v_vgl));
    % Der maximale Fehler wird bestimmt durch Vergleich
    % der Werte der Approximation mit denen
    % der exakten Optimalwertfunktion an den Hilfspunkten.
    % Diese sind im Vektor v_vgl mit uebergeben worden.

    fprintf('     max. Fehler=    %d',maxfehl(z2));
    % Ausgabe des maximalen Fehlers der aktuellen Approximation

    z2=z2+1;
end
```

```
% GITTERVERAENDERUNG

    ss=spline(x,v);
    % Berechnung des Splines zu den neuen Werten v_i an
    % den Knoten x_i

    for i=1:length(x)-1
        xx(i)=((x(i+1)+x(i))/2);
        % Alle Zellen werden halbiert und die so gewonnenen
        % Testpunkte im Vektor xx abgespeichert. Durch die Werte
        % an den Testpunkten wird der loakle Fehler abgeschaetzt.

        vv(i)=ppval(ss,xx(i));
        % Berechnung des Wertes an den Testpunkten durch Auswertung
        % des zu x und v gehoerigen Splines an den Testpunkten
    end

    for i=1:length(xx)
        for k=1:length(u)
            ff2(i,k) = feval(fh,xx(i),u(k),h);
            % Berechnung von fh an den Testpunkten

            gg2(i,k) = feval(g,xx(i),u(k));
            % Berechnung von g an den Testpunkten
        end;
    end;

    for i=1:length(xx)

        for k=1:length(u)
                  if ff2(i,k)>=x(1) & ff2(i,k)<=x(length(x))
                      t2(k)=(h*gg2(i,k)+beta.*ppval(ss,ff2(i,k)));
            % Fuer jeden Testpunkt wird analog zu oben eine Iteration
            % durchgefuehrt. Die Vergleichswerte werden bestimmt und
            % im Vektor t2 abgespeichert.

                  else
                      t2(k)=-1e+5;
                  end
            % Die Werte fuer diejenigen Kontrollwerte u_k,
            % fuer die fh(xx_i, u_k) nicht in [x_1,x_n] liegt,
            % werden "ausgeschlossen".
        end
```

```
    vtest(i)=max(t2);
    % Bestimmung des neuen Wertes am Punkt xx_i durch Maximierung.
    % Die neuen Werte an den Testpunkten werden
    % im Vektor vtest abgespeichert.

    clear t2;
end

for i=1:length(xx)
     eta(i)=abs(vtest(i)-vv(i));
end
% Bestimmung der betragsmaessigen Groesse
% eines Schrittes in allen Testpunkten, durch Vergleich
% mit dem alten Wert. Abgespeichert im Vektor eta.
% Durch diesen Wert ist der Fehler im Intervall begrenzt.

maxeta=max(eta);
% Bestimmung des Residuums durch Maximum-Bestimmung
% des Vektors eta

fprintf('    eta: %f   ',maxeta);
% Ausgabe des Residdums zum betrachteten Gitter
fprintf('    Gesamtfehlerschranke: %f  ',maxeta*(1/(1-beta)));
% Ausgabe der oberen Schranke fuer den Gesamtfehler
% gemaess Satz 5.4

figure;
plot(xx,eta,'o');
title('lokaler Fehler in den einzelnen Intervallen');
% Ausgabe der Schritte und somit des Fehlers am
% jeweiligen Knoten.


% NEUE KNOTENPUNKTE:

if(length(x)<grenze_kn)
% Neues Gitter wird nur gebildet, wenn die maximale
% Knotenzahl noch nicht erreicht ist.

    z3=1;       % Hilfszaehler
```

```
        for i=1:length(xx)
             if eta(i)>theta*maxeta
               xneu(z3)=xx(i);
               vneu(z3)=vv(i);
               z3=z3+1;
            end
        end
  % Die Knoten, an denen der Fehler vergleichsweise gross ist,
  % werden im Vektor xneu abgespeichert. Die zugehoerigen
  % Werte im Vektor vneu. Die Variable 0 < theta < 1 wird
  % in 'aufruf.m' festgesetzt.

        xhilf=x;
        vhilf=v;
  % Hilfsvektoren zur Erstellung des Gitters fuer die
  % naechste Hauptiteration


        [x,v]=vektsort(xhilf,vhilf,xneu,vneu);
        % Erstellung der Vektoren fuer die naechste Hauptiteration
        % durch Hilfsfunktion vektsort.
        % vektsort gibt die alten Knoten aus x und die neuen Knoten
        % aus xneu sortiert im neuen Vektor x aus. Die zugehoerigen
        % Werte stehen an den entsprechenden Stellen im
        % neuen Vektor v.

        clear xneu;
        clear vneu;

        if maxeta>zeta/(1/(1-beta))
            abbruch = 0;
            clear uopt;
        end
        % Wenn der maximale Fehler nicht klein genug ist, wird die
        % Abbruchvariable auf Null gesetzt und man geht zur naechsten
        % Hauptiteration. Es wird also die Approximation auf
        % dem neuen Gitter mit den Knoten x_i bestimmt.
        % Ansonsten wird der Algorithmus beendet.
    end

end         % Ende der aeusseren While-Schleife
```

```
% VERANSCHAULICHUNG DER ERRECHNETEN ERGEBNISSE

intlaengen=diff(x);
% Berechnung der Intervalllaengen

intlaengen(length(x))=intlaengen(length(x)-1);
% Laenge des Vektors intlaengen wird an x angeglichen

erstelle_graphen(x,v,uopt,intlaengen,x_vgl,v_vgl,maxschritt);
% Aufruf des Unterprogramms erstelle_graphen.
% Die errechneten Ergebnisse werden als Plots ausgegeben.
```

Die Funktionen befinden sich auf der beiliegenden CD im Verzeichnis 'MATLAB-Funktionen'. Da durch die ausführlichen Kommentare die Übersichtlichkeit ziemlich nachlässt, ist auf der CD auch eine identische Version ohne Kommentare (*'verf_mit_spline_ohnekom.m'*) angefügt. Ebenfalls auf der CD befindet sich eine Implementierung des Verfahrens (*'verf_mit_spline_ohneplots.m'*), welche auf die Ausgabe der Zwischenergebnisse verzichtet und nur die Ergebnisse am Ende der Berechnungen ausgibt. Diese wird durch die der Funktion *'aufruf.m'* entsprechende Funktion *'aufruf_ohneplots.m'* aufgerufen.

Desweiteren enthält die CD die Implementierungen des Verfahrens mit Spline-Interpolation verknüpft mit der in Abschnitt 4.17 vorgestellten Strategie-Iteration (*'verf_mit_splinestrategie.m'*) sowie die verwendete Implementierung des kontrollierten Gauß-Seidel-Verfahrens mit linearer Interpolation (*'gsv.m'*). Diese werden angesprochen durch *'aufruf_strategie.m'* bzw. *'aufruf_gsv.m'*. Eine Anleitung befindet sich zusätzlich in der Datei *'ReadMe.txt'*.

B.2 Verwendete Hilfsfunktionen

Diese Hilfsfunktionen werden im Hauptprogramm aufgerufen:

In *'loesung_linear.m'* sind die Knoten und die entsprechenden Werte der optimalen Wertefunktion gespeichert, die mit dem kontrollierten Gauß-Seidel-Verfahren mit linearer Interpolation ermittelt worden sind. Entsprechend dem Aufruf werden die zum jeweiligen Problem gehörigen Vektoren x_vgl mit den Knoten des Endgitters und v_vgl mit den dazugehörigen Werten ausgegeben. Diese Werte dienen als Maßstab für die mit Spline-Interpolation ermittelten Approximationen.

Die Funktion *'exakt_wm.m'* berechnet zum Vektor x die exakte Lösung der Optimalwertfunktion zu Beispiel 1 gemäß (6.5). Diese dient bei der Berechnung der optimalen Wertefunktion des Wachstumsmodells (Beispiel 1) als Vergleichsmaßstab für die berechnete Approximation.

An die Funktion *'vektsort.m'* werden die Knoten des alten Gitters im Vektor x mit den entsprechenden Werten im Vektor v sowie die durch Gitteradaption neu hinzugekommen Knoten im Vektor xneu mit den dazugehörigen Werten im Vektor vneu übergeben. Die Funktion erzeugt den Vektor x für die Berechnungen auf dem neuen Gitter. Die Einträge sind in x der Größe nach geordnet ($x_1 < ... < x_n$). Die Werte v_i stehen an den entsprechenden Stellen im neuen Vektor v.

Die Ergebnisse werden nach Ablauf der Berechnungen durch die Funktion *'erstelle_graphen.m'* veranschaulicht. Zum einen wird die Approximation der optimalen Wertefunktion gezeichnet, zum anderen auch die optimalen Kontrollwerte sowie die Intervalllängen des Endgitters. Zum Schluss wird zusätzlich noch einmal die errechnete Approximation mit der durch lineare Interpolation bestimmten Approximation verglichen (bzw. bei Beispiel 1 mit der exakten Optimalwertfunktion). Bei Berechnung mit dem kontrollierten Gauß-Seidel-Verfahren wird die entsprechende Funktion *'erstelle_graphen_gsv.m'* aufgerufen. Der Vergleich mit der Approximation durch lineare Interpolation fällt dabei klarerweise weg.

Die beschriebenen Hilfsfunktionen finden sich ebenfalls auf der beiliegenden CD.

Eine nochmalige Zusammenfassung aller Dateien auf der CD befindet sich in Kapitel C.

Anhang C

Material auf der beiliegenden CD

Die dieser Arbeit beiliegende CD enthält alle angesprochenen MATLAB-Funktionen (im Verzeichnis MATLAB-Funktionen), die Abbildungen zu Kapitel 6 als eps-Dateien (im Verzeichnis Abbildungen) sowie die komplette Arbeit als pdf-Datei (im Verzeichnis Arbeit).

Verzeichnis Abbildungen:
Die eps-Dateien der Abbildungen haben die jeweilige Bildunterschrift als Dateinamen. Zur besseren Übersicht ist das Verzeichnis 'Abbildungen' in die Unterverzeichnisse 'Beispiel 1' bis 'Beispiel 4' untergliedert. Die Abbildungen befinden sich im jeweils zugehörigen Verzeichnis.

Verzeichnis MATLAB-Funktionen:
Tabelle C.1 beinhaltet noch einmal eine komplette Übersicht über alle verwendeten und alle angesprochenen MATLAB-Funktionen sowie eine kurze Beschreibung.

Zusätzlich enthält dieses Verzeichnis die Datei 'ReadMe.txt', die eine kurze Anleitung zur Ausführung der Funktionen gibt.

Funktion	**Aufgabe**
aufruf.m	legt Parameter fest und ruft Iterationsverfahren mit Spline-Interpolation auf
verf_mit_spline.m	berechnet Approximation der optimalen Wertefunktion mit Spline-Interpolation und gibt Ergebnisse aus
verf_mit_spline_ohnekom.m	wie 'verf_mit_spline.m'; allerdings ohne Kommentierung
aufruf_ohneplots.m	ruft Iterationsverfahren auf, bei em keine Zwischenergebnisse ausgegeben werden
verf_mit_spline_ohneplots.m	wie 'verf_mit_spline.m'; allerdings ohne Ausgabe der Zwischenergebnisse
aufruf_strategie.m	ruft Iterationsverfahren mit Spline-Strategie-Iteration auf
verf_mit_splinestrategie.m	berechnet Approximation unter Anwendung der Strategie-Iteration
aufruf_gsv.m	ruft kontrolliertes Gauß-Seidel-Verfahren mit linearer Interpolation auf
gsv.m	berechnet Approximation mit Hilfe des kontrolliertes Gauß-Seidel-Verfahrens mit linearer Interpolation
erstelle_graphen.m	stellt Endergebnisse graphisch dar
erstelle_graphen_gsv.m	stellt Endergebnisse bei Anwendung von 'gsv.m' graphisch dar
vektsort.m	sortiert Vektoren der Knotenpunkte und der zugehörigen Werte entsprechend
exakt_wm.m	berechnet exakte optimale Wertefunktion zu Beispiel 1
loesung_linear.m	enthlt die Lösungen bei Anwendung von 'gsv.m'
fh_..m	enthält die Funkion fh des Problems '.'
g_..m	enthält die Funkion g des Problems '.'

Tabelle C.1: Verwendete MATLAB-Funktionen

Notation

Diese Übersicht enthält einige wichtige Bezeichnungen und Notationen, die in der vorliegenden Arbeit regelmäßig erscheinen, in der Reihenfolge ihres ersten Auftretens. Die jeweilige Bedeutung der folgenden Ausdrücke ist in den entsprechenden Kapiteln erklärt.

Γ	Gitter
Δ	Menge der Knotenpunkte des Gitters Γ
x_i	Knotenpunkte des Gitters Γ, $i = 0, ..., n$
I_i	Intervalle des Gitters Γ, $I_i = [x_{i-1}, x_i]$
s_Δ	Spline zur Knotenmenge Δ
$\mathcal{S}_\Delta$	Raum der Splines zur Knotenmenge Δ
$C^k[a, b]$	Raum der k-mal stetig differenzierbaren Funktionen auf dem Intervall [a,b]
$\kappa_g(x)$	Krümmung des Graphen der Funktion g am Punkt x
$\tilde{\kappa}_g$	Gesamtkrümmung des Graphen g
f_i	$f(x_i)$
f_i'	$f'(x_i)$
h_i	$x_i - x_{i-1}$; Länge des Intervalls I_i
h_{max}	$\max_{i=1,...,n}\{x_i - x_{i-1}\}$; maximale Intervalllänge des Gitters Γ
h_{min}	$\min_{i=1,...,n}\{x_i - x_{i-1}\}$; minimale Intervalllänge des Gitters Γ
ϱ	$\frac{h_{max}}{h_{min}}$
$\|f^{(4)}\|$	$\sup_{x\in[x_0,x_n]} f^{(4)}(x)$
$G,\ \tilde{G}$	Greensche Funktionen
$C[a, b]$	Raum der stetigen Funktionen auf dem Intervall [a,b]

h	zeitliche Schrittweite; Ausnahme: Kapitel 5.2, hier steht h für eine einheitliche Intervalllänge
U	Kontrollwertebereich
$\mathcal{U}$	Raum der kontinuierlichen Kontrollfunktionen
$\mathcal{U}_h$	Raum der diskreten Kontrollfunktionen
u	kontinuierliche Kontrollfunktion; auch Kontrollwert
u_h	diskrete Kontrollfunktion
Φ	Trajektorie in kontinuierlicher Zeit
Φ_h	Trajektorie in diskreter Zeit
J	diskontiertes Funktional
v	optimale Wertefunktion in kontinuierlicher Zeit
v_h	optimale Wertefunktion in diskreter Zeit
δ	Diskontrate
β	Diskontfaktor in diskreter Zeit
$\tilde{v}_h$	zeitdiskretisierte optimale Wertefunktion
v_h^j	optimale Wertefunktion im Laufe des Iterationsverfahrens nach der j-ten Iteration
T_h	Operator zur Bestimmung der v_h^j
Ω	betrachteter Bereich der optimalen Wertefunktion
$\mathcal{W}$	Raum der stetigen, stückweise affin linearen Funktionen
$\hat{v}_h$	Approximation der opt. Wertefunktion aus $\mathcal{W}$
$\hat{v}_h^j$	Approximation der opt. Wertefunktion nach der j-ten Iteration aus $\mathcal{W}$
$\check{v}_h$	Approximation der opt. Wertefunktion aus $\mathcal{S}_\Delta$
$\check{v}_h^j$	Approximation der opt. Wertefunktion nach der j-ten Iteration aus $\mathcal{S}_\Delta$
V bzw. v	Vektor der Werte von $\check{v}_h$ an den Knoten
V^j bzw. v^j	Vektor der Werte von $\check{v}_h^j$ an den Knoten
η_i	lokaler Fehlerschätzer
η	Maximum der lokalen Fehlerschätzer
θ	Parameter zur Verfeinerung der Intervalle

Literaturverzeichnis

[1] Ahlberg, J.H., Nilson E.N., Walsh, J.L.: *The Theory of Splines and Their Applications*, Academic Press, New York, 1967.

[2] Arndt, H., Werner, H.: *Gewöhnliche Differentialgleichungen*, Springer-Verlag, Berlin, Heidelberg, 1986.

[3] Bardi, M., Capuzzo-Dolcetta, I.: *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman equations*, Birkhäuser, Boston, 1997

[4] Birkhoff, G., Priver, A.: *Hermite interpolation error for derivatives*, Journal of Mathematics and Physics 46 (1967), S.440-447.

[5] Böhmer, K.: *Spline-Funktionen*, B.G.Teubner, Stuttgart, 1974.

[6] Brock, W., Starrett, D.: *Nonconvexities in ecological management problems*, SSRI Working Paper 2026, Department of Economics, University of Wisconsin - Madison, 1999,
`http://www.ssc.wisc.edu/econ/archive/authorb.htm`.

[7] de Boor, C.: *Splinefunktionen*, Birkhäuser, Basel, 1990.

[8] Deissenberg, D., Feichtinger, G., Wemmler, W., Wirl, F.: *History Dependance and Global Dynamics in Models with Multiple Equilibria*, Working paper No. 12, Center of Empirical Macroeconomics (CEM), Universität Bielefeld, 2001, `http://www.wiwi.uni-bielefeld.de`.

[9] Falcone, M., Giorgi, T.: *An approximation scheme for evolutive Hamilton-Jacobi equations*, in: McEneaney, W.M., Yin, G.G., Zhang, Q. (Hrsg.): *Stochastic analysis, control, optimization and applications*, Birkhäuser, Boston, 1999, S.288-303.

[10] Grüne, L.: *An adaptive grid scheme for the discrete Hamilton-Jacobi-Bellman equation*, Numerische Mathematik 75 (1997), S.319-337.

[11] Grüne, L.: *Numerische Dynamik von Kontrollsystemen*, Ausarbeitung einer Vorlesung gehalten im Sommersemester 2004 an der Universität Bayreuth.

[12] Grüne, L.: *Numerische Mathematik I*, Ausarbeitung einer Vorlesung gehalten im Wintersemester 2002/2003 an der Universität Bayreuth.

[13] Grüne, L., Semmler, W.: *Using Dynamic Programming with Adaptive Grid Scheme for Optimal Control Problems in Economics*, Journal of Economic Dynamics and Control, noch nicht erschienen.

[14] Grüne, L., Semmler, W., Sieveking, M.: *Creditworthyness and Thresholds in a Credit Market Model with Multiple Equilibria*, Economic Theory 25 (2005), S.287-315.

[15] Hall, C.A.: *On Error Bounds for Spline Interpolation*, Journal of Approximation Theory 1 (1968), S.209-218.

[16] Hall, C.A., Meyer, W.W.: *Optimal Error Bounds for Cubic Spline Interpolation*, Journal of Approximation Theory 16 (1976), S.105-122.

[17] Hartl, R.F., Kort, P., Feichtinger, G., Wirl, F.: *Multiple equlibria and thresholds due to relative investment costs: non-concave-concave, focus-node, continuous-discontinuous*, Technische Universität Wien, 2000, `http://www.eos.tuwie.ac.at/OR/`.

[18] Herrmann, N.: *Spline-Funktionen*, Skript zur Vorlesung 'Mathematische Modellbildung', Universität Hannover, 2003, `http://www.ifam.uni-hannover.de/~herrmann/`.

[19] Myint-U, T.: *Ordinary Differential Equations*, North-Holland, New York, 1987.

[20] Santos, M.S., Vigo-Aguiar, J.: *Analysis of a numerical dynamic programming algorithm applied to economic models*, Econometrica 66(2) (1998), S.409-426.

[21] Sauer, T.: *Numerik 1*, Ausarbeitung einer Vorlesung gehalten im Wintersemester 2002/2003 an der Universität Gießen.

[22] Stoer, J.: *Numerische Mathematik 1*, Springer-Verlag. Berlin, Heidelberg, 1994.

[23] Trick, M.A., Zin, S.E.: *Spline approximations to value functions: a linear programming approach*, Macroeconomic Dynamics 1 (1997), S.255-277.

[24] Wilkinson, J.H.: *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1972.

ERKLÄRUNG

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen.

Bayreuth, den 30. September 2004 .....

Florian Bauer