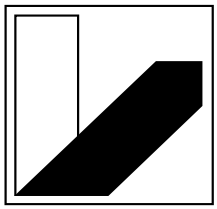

UNIVERSITÄT
BAYREUTH

Implizite Runge-Kutta-Verfahren und ihre Anwendung auf Steuerungsprobleme

Diplomarbeit

von

Eggert Rose

FAKULTÄT FÜR MATHEMATIK UND PHYSIK
MATHEMATISCHES INSTITUT

Datum: 31. Januar 2007

Aufgabenstellung / Betreuung:
Prof. Dr. F. Lempio / Dr. R. Baier

Danksagung

An dieser Stelle möchte ich mich bei allen Personen bedanken, die zur Entstehung dieser Arbeit beigetragen haben.
Bei Herrn Professor Lempio möchte ich mich für die Überlassung des vielschichtigen Themas und die Beaufsichtigung und Betreuung dieser Arbeit bedanken.
Bei Herrn Professor Zillober von der Universität Würzburg möchte ich mich für die Bereitstellung seines Optimierungsprogramms SCPIP 3.0 und Ratschläge zu dessen Gebrauch bedanken.
Besonderer Dank gebührt Herrn Dr. Baier. Als Betreuer dieser Arbeit hat er stets mit großem Interesse und Einsatz die Entwicklung der Ausarbeitungen verfolgt und begleitet. Seine Hilfestellungen bei der Ausrichtung dieser Arbeit, seine beständige Einarbeitung auch in Details und seine motivierende Anteilnahme haben ihn als Betreuer sehr ausgezeichnet.
Weiter gilt auch meinem Kommilitonen Heiko Schwartz großer Dank als anregender und kritischer Diskussionspartner.

Inhaltsverzeichnis

Kapitel 1

Einleitung

In dieser Diplomarbeit werden implizite Runge-Kutta-Verfahren hoher Konsistenzordnungen hergeleitet und die zugehörigen Diskretisierungen dann mittels eines Kollokationspolynomansatzes als ein direktes Lösungsverfahren für Optimalsteuerungsproblemen eingesetzt.

Themenübersicht. Diskretisierungsentscheidungen müssen für viele Aufgabenstellungen der numerischen Mathematik getroffen werden. Dazu gehören die beiden wichtigen Problemstellungen der Anfangswert- und Optimalsteuerungsprobleme. Eine grundlegende Position nehmen numerische Integrationsverfahren ein, denn die beiden erwähnten schwierigeren Problemstellungen besitzen Strukturen von Integrationsproblemen. Numerische Lösungsverfahren für Integrale lassen sich deshalb oft auf Differentialgleichungen und Optimalsteuerungsprobleme übertragen.
Für die numerische Integration einer Funktion f im Intervall $[t_0, t_0 + h]$ kann die Quadraturformel

$$\int_{t_0}^{t_0+h} f(t)\,dt \approx \sum_{i=1}^{s} b_i f(t_0 + c_i h)$$

angesetzt werden. Eine besondere Effektivität besitzt die Gauß-Quadratur, bei der die s Knoten $c = (c_1, \ldots, c_s)^T$ durch die Nullstellen von speziellen orthogonalen Polynomen ermittelt werden. Die Gauß-Quadratur wird in Numerik-Grundvorlesungen etwas durch die einfacheren Newton-Cotes-Regeln und die ausgeklügelte Romberg-Extrapolation überschattet, erreicht jedoch den maximalen Exaktheitsgrad $2s - 1$. Besonders beliebt ist die Abart der Gauß-Lobatto-Verfahren, bei der mit $c_1 = 0$ und $c_s = 1$ die Randpunkte des Intervalls als Knoten gewählt werden, und die Exaktheit nur auf $2s - 3$ sinkt. Der Stützstellenvektor $c = (c_1, \ldots, c_s)^T$ ist in dieser Arbeit ein elementarer Bestandteil für

Diskretisierungen aller drei Aufgabenstellungen. Eine erste Verwendung findet die Gauß-Quadratur bei der numerischen Lösung von Anfangswertproblemen. Für das Anfangswertproblem

$$\dot{x}(t) = f\big(t, x(t)\big), \quad x(t_0) = x_0, \quad t \in [t_0, t_0 + h],$$

daß auch als Integrationsproblem formulierbar ist, lassen sich die Runge-Kutta-Verfahren als eine Klasse von Einschrittverfahren entwickeln. Die Theorie der impliziten Runge-Kutta-Verfahren wurde mittels der graphentheoretischen Darstellung von John Charles Butcher gewonnen. Seine Werke, wie [12], sind aber aufgrund ihres Umfangs und ihrer Komplexität in dieser Diplomarbeit nicht rezeptierbar. Die impliziten Runge-Kutta-Verfahren, die durch die Butcher-Tableaus charakterisiert werden, sind aber im wesentlichen durch die Gauß-Quadratur erklärbar. Die Vektoren c und b des unten abgebildeten Butcher-Tableaus

$$\begin{array}{c|ccc} c_1 & a_{11} & \dots & a_{1s} \\ \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & \dots & a_{ss} \\ \hline & b_1 & \dots & b_s \end{array}$$

stimmen bei den Verfahren hoher Konsistenzordnungen mit den Knoten und Gewichten der Gauß-Quadratur überein. Man erreicht die maximale Konsistenzordnung $2s$, wenn die rechte Seite f der Differentialgleichung korrespondierende Differenzierbarkeitsvoraussetzungen aufweist. Bei Lobatto-Verfahren liegt die Konsistenzordnung wiederum um zwei niedriger bei $2s - 2$.
Ein alternativer numerischer Ansatz zur Lösung eines Anfangswertproblems besteht in einem Polynomansatz p mit $\operatorname{Grad} p = s$:

$$\dot{p}(t_0 + c_i h) = f\big(t_0 + c_i h, p(t_0 + c_i h)\big), \quad i = 1, \dots, s, \quad p(t_0) = x_0.$$

Überraschenderweise sind gewisse solche Ansätze äquivalent zu speziellen impliziten Runge-Kutta-Verfahren, wie in [28] oder [18] gezeigt wird. Diese Kollokationsverfahren besitzen dann nicht nur die selben Konsistenzordnungen, sondern liefern im Gegensatz zu Einschrittverfahren statt einer Gitterfunktion sogar eine kontinuierliche Lösung.
Geschickte Diskretisierungen bilden auch die Basis der direkten Lösungsverfahren von Optimalsteuerungsproblemen. Wie in [19] werden Gauß-Lobatto-Kollokationsverfahren verwendet, um den Zustand x und die Steuerung u polynomial zu approximieren. Hierbei spezialisiert sich die Arbeit weitgehend auf Linear-Quadratische Optimalsteuerungsprobleme der folgenden Grundstruktur:

Minimiere

$$\frac{1}{2}x(t_f)^T S\, x(t_f) + \frac{1}{2}\int_{t_0}^{t_f}\Big(x(t)^T Q(t)\, x(t) + u(t)^T R(t)\, u(t)\Big)dt$$

unter den Nebenbedingungen

$$\begin{aligned} \dot{x}(t) &= A(t)x(t) + B(t)u(t), \\ x(t_0) &= x_0. \end{aligned}$$

Ein Kollokationspolynomansatz für Optimalsteuerungsprobleme führt zu einem finiten nichtlinearen Optimierungsproblem, für Linear-Quadratische Optimalsteuerungsprobleme zu einem Quadratischen Programm. Für solche Aufgaben werden Beispielrechnungen mit dem Optimierungsprogramm SCPIP 3.0 durchgeführt, wobei dann teilweise auch schwierige Restriktionen wie Randwerte und Steuerbeschränkungen hinzugefügt werden.

Kapitelübersicht. In Kapitel 2 wird die Gauß-Quadratur erarbeitet. In den ersten Abschnitten 2.3 und 2.4 werden zuerst orthogonale Polynome, vor allem Legendre-Polynome, studiert. Numerische Integrationsverfahren folgen und führen im Abschnitt 2.6 zum Satz 2.16, der Verfahren des maximalen Exaktheitsgrades $2s-1$ charakterisiert. Danach werden modifizierte Verfahren wie die Gauß-Lobatto-Regeln entwickelt.
Das Kapitel 3 über Differentialgleichungen trägt in 3.2 zunächst allgemeine Sätze zu Anfangswertproblemen zusammen. In 3.3 werden numerische Einschrittverfahren definiert, während 3.4 die Konvergenzbegriffe schildert.
Das Kapitel 4 thematisiert Runge-Kutta-Verfahren, wobei vor allem [12] und [28] verwendet wurden. Im einführenden Abschnitt 4.2 werden bereits Zusammenhänge zur Gauß-Quadratur sichtbar. Abschnitt 4.3 definiert diese Verfahrensklasse allgemein und liefert den wichtigen Existenz- und Eindeutigkeitssatz 4.4. Im Satz 4.8 des Abschnitts 4.4 werden die Grenzen der expliziten Runge-Kutta-Verfahren ersichtlich. Bei s Stufen erreicht man höchstens die Ordnung s. Die Verfahren der maximalen Konsistenzordnung $2s$ werden in 4.5 hergeleitet. Hierbei liefern die Sätze 4.22 und 4.25 die entscheidenen Konstruktionsvorschriften. Die abgeleiteten Radau- und Lobatto-Verfahren werden im Abschnitt 4.6 vorgestellt.
Der Satz 5.3 des Kapitels 5 über Kollokationsverfahren verdeutlicht den Zusammenhang spezieller Kollokationsverfahren mit bestimmten impliziten Runge-Kutta-Verfahren. Im Satz 5.6 wird eine klare Bedingung gestellt, wann ein Kollokationsverfahren einem impliziten Runge-Kutta-Verfahren entspricht. Dieses Kapitel richtet sich vor allem nach [28], [18] und [58].
Das Kapitel 6 behandelt Optimalsteuerungsprobleme. In 6.3 erfolgt eine kurze Hinführung zu einem vereinfachten Minimumprinzip. Die Definition 6.13 im

Abschnitt 6.4 leitet dann zu Linear-Quadratischen Optimalsteuerungsproblemen über. Die in Bemerkung 6.16 resümierten Ergebnisse des Satzes 6.14 prädestinieren diese Problemstellung für einen stetig differenzierbaren numerischen Lösungsansatz.
Nachdem bereits in Abschnitt 6.5 direkte Lösungsverfahren für Optimalsteuerungsprobleme eingeführt wurden, wird in 7.2 ein Kollokationsansatz für den Zustand x und die Steuerung u vorgeschlagen. Hierbei muß auf theoretische Begründungen weitgehend verzichtet werden. [7], [46] und [63] dienen als Ideenreservoir. Die allgemeine Umsetzung mündet in Problem 7.4, der korrespondierende Ansatz für Linear-Quadratische Optimalsteuerungsprobleme hingegen in Problem 7.8 - wie in [19] - bzw. in Problem 7.12.
In Kapitel 8 werden Linear-Quadratische Optimalsteuerungsprobleme mit der Vorgehensweise des Kapitels 7 mithilfe des Fortran77-Codes SCPIP 3.0 numerisch gelöst. Die Abschnitte 8.2 und 8.3 beschreiben die Umsetzung und Implementierung der Problemstellungen 7.8 (bzw. 7.12) in SCPIP 3.0, wobei das Benutzermanual [64] unverzichtbar ist. Der Abschnitt 8.4 präsentiert Beispielrechnungen, die in Abschnitt 8.5 kommentiert werden.

Resümee. Die Runge-Kutta-Verfahren benötigen für die Konsistenzresultate ausreichende Differenzierbarkeitsvoraussetzungen der Funktion f. Davon abgesehen ist die Konvergenztheorie aber fundiert und abgeschlossen. Bei direkten Lösungsverfahren für Optimalsteuerungsprobleme ist die Konvergenztheorie viel schwieriger (vgl. [52]). Im Kontrast zu Anfangswertproblemen zerstören nicht fehlende Glattheitsgegebenheiten beteiligter Funktionen die Differenzierbarkeitseigenschaften der optimalen Lösungen, sondern die Aufgabenstruktur. Weisen die Linear-Quadratischen Beispiele in Abschnitt 8.4 relativ glatte optimale Trajektorien auf, so ist ein Polynomansatz ziemlich effektiv bei nur geringem Aufwand, siehe Beipiele 8.2 und 8.3. Jedoch führen bereits Steuerbeschränkungen und Randwerte bei Linear-Quadratischen Optimalsteuerungsproblemen zu Sprüngen in der optimalen Steuerung. Die Approximation dieser Umschaltpunkte ist schwierig, wie Beispiel 8.8 dokumentiert. Doch erweisen sich die Werte an den Lobatto-Knoten bereits bei geringem Abstand zu den Sprungstellen als sehr genau, es kommt sogar die polynomiale Interpolation der numerischen Lösungen zur Geltung. Eine zunehmende Gitterfeinheit verbessert die Lösungen weiter, aber noch mehr die Involvierung von Informationen des Lösungsverlaufs nach einer ersten Iteration.

Bayreuth, Januar 2007 Eggert Rose

Kapitel 2

Orthogonale Polynome und Gauß-Quadratur

2.1 Einleitung

Dieses Kapitel besteht aus zwei Teilen, um die sich verbindende und abrundende Abschnitte gruppieren: orthogonale Polynome zuerst und danach Gauß-Quadratur-Integrationsregeln. Diese beiden Themen repräsentieren Techniken aus dem Bereich der numerischen Interpolation und Integration. Obgleich diese Bereiche eigentlich bereits in Grundvorlesungen der numerischen Mathematik behandelt werden, unterbleibt doch meistens eine tiefgehendere Betrachtung der orthogonalen Polynome und der Gauß-Quadratur. Die orthonalen Polynome und die anknüpfende Gauß-Quadratur besitzen jedoch eine sehr umfassende, abgeschlossene Theorie, die nach gewissen Kriterien Höchstleistungen erzielt. Da sich Integrationsmerkmale auch auf andere Aufgabenstellungen wie Anfgangswertprobleme übertragen lassen, gelingt es nur mit Kenntnissen der Gauß-Quadratur auch für solche Problemstellungen exzellente numerische Verfahren herzuleiten. Die Ergebnisse dieses Kapitels durchdringen deshalb alle anderen Kapitel.

Die numerische Integration wird in Abschnitt 2.2 motiviert.
Im Abschnitt 2.3 werden orthogonale Polynome formal eingeführt.
Im anschließenden Abschnitt 2.4 werden Legendre-Polynome als ein Spezialfall orthogonaler Polynome behandelt.
Nun startet der zweite Block des Kapitels: Abschnitt 2.5 erörtert elementare Gedanken zur numerischen Integration.
Der Abschnitt 2.6 über die Gauß-Quadratur umfaßt das Kernstück der Darstellung der numerischen Integration. Die Gauß-Quadratur basiert dabei entscheidend auf den vorangehenden Abschnitten über orthogonale Polynome.
Abschnitt 2.7 modifiziert 2.6 um einige Erweiterungen.

Abschnitt 2.8 dient als Hilfs- und Ergänzungsabschnitt des Kapitels.

Der Inhalt dieses Kapitels ist größtenteils in Büchern und Skripten von Numerik-Grundvorlesungen zu finden. Verwendet wurden vor allem [26], [55], [56], [53], [30], [41], [42], [62], [47] und [27]. Funktionalanalytische Aspekte kann man in [60] nachlesen. [51] soll die Bedeutung der Thematik in der Physik untermauern.

2.2 Motivation

Die folgenden einleitenden Erläuterungen werden noch formlos geschrieben. Die auftretenden Aufgaben- und Fragestellungen sind durchaus geläufig; eine mathematisch exakte Ausformulierung erfolgt in den späteren Kapiteln.
Das Rechnen mit gewöhnlichen Differentialgleichungen gehört zu den Standardaufgaben der Mathematik. Ein Anfangswertproblem

$$\dot{x}(t) = f\big(t, x(t)\big), \quad x(t_0) = x_0, \tag{2.1}$$

läßt sich durch Integration in eine Integralgleichung umformen:

$$x(t) = x_0 + \int_{t_0}^{t} f(s, x(s))\, ds. \tag{2.2}$$

Das Lösen dieser Integralgleichung ist äquivalent zum Lösen der ursprünglichen Differentialgleichung. Die Schwierigkeit des Ausgangsproblems ist auf ein Integrationsproblem übergegangen. Eine offenbar verwandte, wenn auch nicht identische, Aufgabenstellung lautet:

$$I[g] = \int_a^b g(x)\, dx. \tag{2.3}$$

Möchte man den Wert des Integrals bei gegebenem Intervall $[a, b]$ ermitteln, sucht man ein geeignetes numerisches Integrationsverfahren. Der klassische Qudraturansatz lautet:

$$I[g] = I_n[g] + R_n[g] = \sum_{i=1}^{n} \alpha_i g(x_i) + R_n[g], \quad \alpha_i \in \mathbb{R}, \tag{2.4}$$

wobei $I_n[g]$ üblicherweise beliebige Polynome bis zu einem gewissem Grad exakt integrieren soll (der „Exaktheitsgrad“ einer Integrationsregel) und $R_n[g]$ die Fehlerdarstellung angibt. Die Wahl der x_i und α_i bestimmt den Exaktheitsgrad der Integrationsformel. Legt man die n Knoten x_i auf dem Intervall $[a, b]$ äquidistant fest, so erhält man die Newton-Cotes-Formeln. Diese gewährleisten in etwa den

Exaktheitsgrad n, wobei der Exaktheitsgrad aufgrund gewisser Feinheiten leicht variieren kann.
Bei einer äquidistanten Wahl der Knoten hat man jedoch den Spielraum der Integrationsregel stark eingeschränkt. Bei der Gauß-Quadratur werden dagegen die Knoten und Gewichte als $2n$ freie Parameter aufgefaßt. Die Knoten werden nun so positioniert, daß der Exaktheitsgrad möglichst maximal wird. Tatsächlich erreicht man so den (maximalen) Exaktheitsgrad $2n-1$. Die Knoten ergeben sich hierbei über die Nullstellen gewisser orthogonale Polynome.
Es wird sich - als Vorgriff auf Kapitel 4 - ergeben, daß die Eigenschaften der impliziten Runge-Kutta-Verfahren aus denjenigen der Gauß-Quadratur folgen. Die Resultate der Gauß-Quadratur werden aus dem Studium der orthogonalen Polynomen erkennbar. Dieses Kapitel startet logisch in der umgekehrten Reihenfolge zunächst mit den orthogonalen Polynomen, besonders den Legendre-Polynomen. Im Anschluß folgt dann die Anwendung dieser Theorie auf die Integration und führt zur Gauß-Quadratur.

2.3 Orthogonale Polynome

Der Abschnitt folgt zunächst [26, Abschnitt 4.3]. Es sei P der Raum der Polynome. Für die Charakterisierung von Orthogonalität benötigt man ein Skalarprodukt.

Definition 2.1. *Es sei durch $a, b \in \mathbb{R}$ das Intervall $[a, b]$ gegeben. $\omega : (a, b) \to \mathbb{R}^+$ sei eine nichtnegative, Lebesgue-integrierbare Funktion, die* `Gewichtsfunktion` *genannt wird. Für p, $q \in P$ sei*

$$\langle p, q \rangle_\omega = \int_a^b \omega(x) p(x) q(x)\, dx$$

das Skalarprodukt von p und q und

$$\|p\|_\omega = \sqrt{\langle p,p \rangle_\omega}$$

die zugehörige Norm.

Die Eigenschaften eines Skalarprodukt sind leicht nachzuweisen, siehe hierzu [60, Abschnitt V.1]. Zwei Polynome p und q sind zueinander orthogonal, wenn

$$\langle p, q \rangle_\omega = 0 \tag{2.5}$$

gilt. Im folgenden bezeichne $(p_i)_{i \in \mathbb{N}_0}$ eine Folge von Polynomen aus P, wobei i den Grad der Polynome angeben soll.

Satz 2.2 (Orthogonalsystem). *Die p_i bilden ein Orthogonalsystem bezüglich der Gewichtsfunktion ω, wenn*

$$\langle p_i, p_j \rangle_\omega = \begin{cases} \gamma_i, & i = j, \\ 0, & i \neq j \end{cases}$$

mit $\gamma_i > 0$ für alle i gilt.

Im weiteren Verlauf wird unter p_i, $i \in \mathbb{N}_0$ in diesem Abschnitt immer ein Orthogonalsystem verstanden, wohingegen ein p ohne Indizierung ein beliebiges Polynom bezeichnet.
Zuerst muß die Existenz und Eindeutigkeit einer solchen orthogonalen Polynomfolge zu einer beliebigen Gewichtsfunktion ω bewiesen werden. Auch eine Konstruktionsvorschrift ist wünschenswert. Dies liefert der nächste Satz.

Satz 2.3 (Rekursionsvorschrift). *Zu jeder Gewichtsfunktion ω existieren eindeutig bestimmte orthogonale Polynome $(p_i)_{i \in \mathbb{N}_0}$ mit führenden Koeffizienten 1. Sie erfüllen die Rekursionsgleichung*

$$p_i(x) = (x + b_i)p_{i-1}(x) + c_i p_{i-2}(x), \quad i = 1, 2, \ldots \quad \text{mit} \tag{2.6}$$

$$p_{-1} = p_0 \equiv 1 \quad \text{und} \tag{2.7}$$

$$b_i = -\frac{\langle x p_{i-1}, p_{i-1} \rangle_\omega}{\langle p_{i-1}, p_{i-1} \rangle_\omega}, \quad c_i = -\frac{\langle p_{i-1}, p_{i-1} \rangle_\omega}{\langle p_{i-2}, p_{i-2} \rangle_\omega}. \tag{2.8}$$

Beweis. Der Satz wird mit Induktion bewiesen und findet sich in zahlreichen Lehrbüchern und Skripten, etwa in [26, Satz 4.20] oder [55, Abschnitt 3.6]. □

Die obige Rekursion liefert eine Folge von Polynomen aufsteigenden Grades. Ferner läßt sich auch jedes Orthogonalsystem durch Normierung in ein Orthonormalsystem transformieren, so daß man sogar ein vollständiges Orthonormalsystem (VONS) von P gewinnt.
Es wird nun mit P_i der Raum der Polynome bis zum Grad i bezeichnet. Dann gilt mit Satz 2.3

$$\langle p, p_i \rangle_\omega = 0 \quad \forall p \in P_{i-1}, \tag{2.9}$$

denn das Polynom p kann als eine Linearkombination der Polynome $p_0, \ldots, p_{i-1}$ dargestellt werden. Die Linearität des Skalarprodukts führt mit der Orthogonalitätseigenschaft des Orthogonalsystems zum Ergebnis.
Eine besonders wichtige Rolle spielen die Nullstellen der orthogonalen Polynome. Dazu ergibt sich der folgende, bedeutende Satz, den man in [30, Abschnitt 5.4] und [55, Satz 3.6.10] findet:

Satz 2.4 (Nullstellensatz). *Bilden die Polynome $(p_i)_{i=0,\dots,n}$ ein Orthogonalsystem auf dem Intervall $[a,b]$ bezüglich der Gewichtsfunktion ω, so besitzt jedes dieser Polynome lauter einfache, reelle Nullstellen im offenen Intervall (a,b).*

Beweis des Nullstellensatzes. Es seien $x_{i,1},\dots,x_{i,i}$ die Nullstellen von p_i. Aufgrund der Orthogonalität gilt:

$$0 = \langle p_i, p_0 \rangle_\omega = \int_a^b (x - x_{i,1}) \cdot \ldots \cdot (x - x_{i,i}) \cdot \omega(x)\, dx, \quad i \geq 1.$$

Es existiert also mindestens eine reelle Nullstelle mit Vorzeichenwechsel in (a,b) mit ungerader Vielfachheit. Es seien mit $x_{i,k}$, $k = 1,\dots,l$ die reellen Nullstellen ungerader Vielfachheit in (a,b) bezeichnet. Dann gilt mit

$$\Pi(x) := \prod_{k=1}^{l} (x - x_{i,k}),$$

daß $p_i(x)\Pi(x)$ in (a,b) nicht das Vorzeichen wechselt:

$$p_i(x)\Pi(x) \leq 0 \quad \dot\vee \quad p_i(x)\Pi(x) \geq 0 \quad \forall x \in (a,b).$$

Daraus folgt:

$$\langle p_i, \Pi \rangle_\omega \neq 0, \quad \text{aber } \langle p_i, p \rangle_\omega = 0 \text{ für } p \in P_{i-1}.$$

Also muß Π ein Vielfaches von p_i sein, und somit sind alle Nullstellen reell, einfach und liegen im Intervall (a,b). □

Dieser Satz ist sehr bedeutsam. Die Eigenschaften der Nullstellen wirken sich prägend auf viele Bereiche der weiteren Diplomarbeit aus.
Orthogonalsysteme zu unterschiedlichen Gewichtsfunktionen sind nicht nur Spezialfälle der numerischen Integration. Sie sind vor allem von emminenter Bedeutung in der mathematischen Physik. Bekannte Beispiele sind die Tschebyscheff-Polynome für

$$\omega(x) := \frac{1}{\sqrt{(1-x^2)}} \quad \text{auf dem Intervall} \quad (a,b) := (-1,1) \tag{2.10}$$

und die Hermite-Polynome für

$$\omega(x) := 1 \quad \text{auf dem Intervall} \quad [a,b] := [-1,1]\,. \tag{2.11}$$

Diese und andere Fälle werden jedoch nicht weiter analysiert. Der wichtigste Spezialfall tritt bei

$$\omega(x) := 1 \quad \text{und} \quad [a,b] := [-1,1] \tag{2.12}$$

auf und wird im nachfolgenden Abschnitt behandelt: die Legendre-Polynome.

2.4 Legendre-Polynome

Eine bedeutende Rolle spielen die Legendre-Polynome in der mathematischen Physik, sie treten zum Beispiel bei der Analyse des Drehimpulses des Wasserstoffatoms auf, siehe etwa [51, Abschnitt 5.3]. Im Kontext der numerischen Integration bilden sie die Grundlage der Gauß-Quadratur im Abschnitt 2.6.

Satz 2.5 (Legendre-Polynome). *Die Legendre-Polynome*

$$L_n(x) = \frac{1}{2^n \cdot n!} \cdot \frac{d^n}{dx^n}\left[(x^2-1)^n\right], \quad n \in \mathbb{N}_0 \tag{2.13}$$

bilden auf dem Intervall $[-1,1]$ *bezüglich der Gewichtsfunktion* $\omega \equiv 1$ *mit dem Skalarprodukt* $\langle \cdot, \cdot \rangle_\omega$ *der Definition 2.1 ein Orthogonalsystem. Es gilt:*

$$\int_{-1}^{1} L_n(x) L_m(x)\, dx = \begin{cases} \frac{2}{2n+1}, & n = m, \\ 0, & n \neq m, \end{cases} \qquad n,\, m \in \mathbb{N}_0. \tag{2.14}$$

Beweis. Der Beweis verwendet den Nullstellensatz 2.4 und partielle Integration, siehe [53, Satz 3.38] oder [62, Lemma 3.10]. □

Das Skalarprodukt entspricht hier dem bekannten L_2-Skalarprodukt. Eine andere Darstellung der Legendre-Polynome ist

$$L_n(x) = \frac{1}{2^n} \cdot \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} (-1)^i \cdot \binom{n}{i} \cdot \binom{2n-2i}{n} \cdot x^{n-2i}, \quad n \in \mathbb{N}_0, \tag{2.15}$$

wobei $\lfloor \cdot \rfloor$ die Gauß-Klammer bezeichnet. Diese Formel ergibt sich aus der Anwendung des Binomischen Lehrsatzes auf $(x^2-1)^n$ und nachfolgender Differentiation. Zum Rechnen ist diese Formel häufig gut geeignet.
Mit dem Satz 2.3 berechnet man die Drei-Terme-Rekursion für $n \geq 1$:

$$L_0(x) = 1, \quad L_1(x) = x, \quad L_{n+1}(x) = \frac{2n+1}{n+1} L_n(x) - \frac{n}{n+1} L_{n-1}(x). \tag{2.16}$$

Die Legendre-Polynome der Grade 1 bis 5 lauten beispielsweise:

$$\begin{aligned} L_1(x) &= x, \\ L_2(x) &= 1{,}5x^2 - 0{,}5, \\ L_3(x) &= 2{,}5x^3 - 1{,}5x, \\ L_4(x) &= 4{,}375x^4 - 3{,}75x^2 + 0{,}375, \\ L_5(x) &= 7{,}875x^5 - 8{,}75x^3 + 1{,}875x. \end{aligned} \tag{2.17}$$

Mit dem Vorfaktor $\sqrt{\frac{2n+1}{2}}$ werden die Legendre-Polynome orthonormiert. Im übrigen erhält man die Legendre-Polynome auch durch die Anwendung des Orthonormalisierungsverfahrens von Gram-Schmidt auf die Monome x^n, $n \geq 0$ (zur Gram-Schmidt-Orthonormalisierung: [38, Kapitel 1, Abschnitt 13] oder [41, Abschnitt 4.4]).
Bei Betrachtung der Formel (2.13) erkennt man, daß jedes Legendre-Polynome entweder gerade oder ungerade ist. Es gilt damit die Gleichung

$$L_n(x) = (-1)^n \cdot L_n(-x). \tag{2.18}$$

Die L_n sind demnach punktsymmetrisch zu $(0,0)$ für ungerade n und achsensymmetrisch zur y-Achse für gerade n. Damit sind die Nullstellen der Legendre-Polynome symmetrisch um 0 verteilt sind. Nach dem Nullstellensatz 2.4 sind sie einfach, reell und in $(-1,1)$ liegend. Ungerade L_n haben somit insbesondere eine Nullstelle bei $x = 0$. Einen Eindruck vom Verlauf der Legendre-Polynome liefert die Abbildung 2.1.

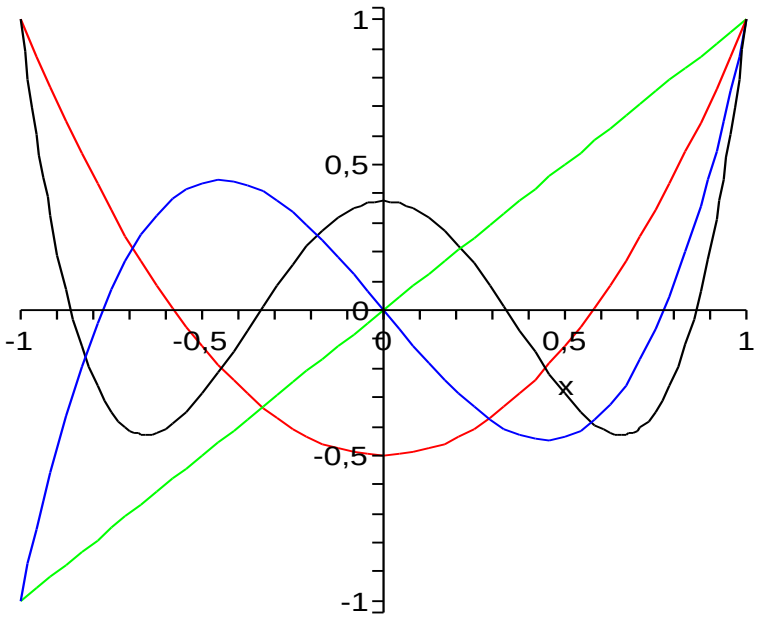

Abbildung 2.1: Legendre-Polynome L_1, L_2, L_3 und L_4

Es gibt keine explizite Formel zur Berechnung der Nullstellen. Sie müssen folglich numerisch berechnet werden. Dies kann direkt über die Lösung des nichtlinearen

Gleichungssystems $L_n(x) = 0$ geschehen. Stattdessen können die Nullstellen auch als Eigenwerte einer Tridiagonalmatrix ermittelt werden, was numerisch effizienter ist. Näheres dazu findet man in [55, Abschnitt 3.6] und [62, Abschnitt 4.3.2]. Die Nullstellen der Legendre-Polynome L_n sind überdies in zahlreichen Nachschlagewerken tabelliert. Eine klassische Adresse dafür ist das Buch [1, Kapitel 25]. Moderne Mathematikprogramme wie Maple und Matlab stellen die Nullstellen ohne größeren Arbeitsaufwand mit hinreichender Genauigkeit zur Verfügung, siehe hierzu auch Abschnitt A.3. Die Abbildung 2.2 vergleicht optisch die Lage der Nullstellen von Legendre-Polynomen verschiedenen Grades im Intervall $[-1, 1]$. Die Nullstellen liegen offenbar zu den Intervallenden hin dichter als im Zentrum.

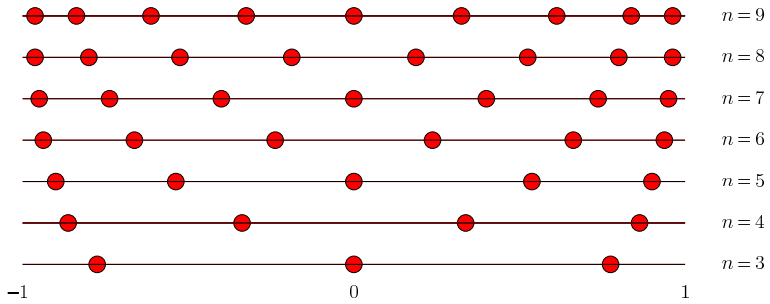

Abbildung 2.2: Nullstellen der Legendre-Polynome L_3 bis L_9

Oftmals wird das Intervall $[-1, 1]$ auf $[0, 1]$ transformiert, da bei einigen Anwendungen das Intervall $[0, 1]$ favorisiert wird, zum Beispiel bei Runge-Kutta-Verfahren (siehe Kapitel 4). Mit der Substitution $x := 2t - 1$ ergibt sich:

$$L_n(2t-1) = \frac{1}{n!} \cdot \frac{d^n}{dt^n} \left[t^n (t-1)^n \right], \quad t \in [0,1]. \tag{2.19}$$

Mit dem folgenden nützlichen Lemma wird die Thematik der orthogonalen Polynome abgeschlossen.

Lemma 2.6.

$$L_n(1) = 1, \tag{2.20}$$
$$L_n(-1) = (-1)^n. \tag{2.21}$$

Beweis. (2.20) und (2.21) lassen sich mit Induktion beweisen. Man rechnet (2.20) und (2.21) für die ersten n nach. Mit der Rekursionsformel (2.16) kann man dann leicht nachweisen, daß (2.20) und (2.21) allgemein gelten. □

(2.20) und (2.21) geben Aufschluß über die Funktionswerte am Rand des Intervalls des $[-1, 1]$. Das Lemma 2.6 wird für den Satz 2.30 verwendet, dem ab Abschnitt 7.4 eine wichtige Hilfsfunktion zukommt.

Man kann ferner eine Beschränktheit der Legendre-Polynome zeigen:

$$|L_n(x)| \leq 1, \quad x \in [-1, 1]\,. \tag{2.22}$$

Die letzte Eigenschaft und das Lemmas 2.6 werden auch in Abbildung 2.1 visualisiert.

2.5 Erste Integrationsmodelle

In diesem Abschnitt wird die numerische Integration eingeführt. Es werden erste Ideen geschildert, Begriffe festgelegt und die bekannten Newton-Cotes-Regeln vorgestellt. Die Aufgabe besteht in der numerischen Integration einer Funktion:

$$I[f] = \int_a^b f(x)\,dx. \tag{2.23}$$

Wie schon im Abschnitt 2.2 erwähnt, lautet eine Quadraturformel:

$$I[f] = I_n[f] + R_n[f] = \sum_{i=1}^{n} \alpha_i f(x_i) + R_n[f], \quad \alpha_i \in \mathbb{R}. \tag{2.24}$$

Der Term $I_n[f] = \sum_{i=1}^{n} \alpha_i f(x_i)$ soll das Integral möglichst exakt approximieren. $R_n[f]$ ist als Restglied letztendlich ein Fehlerterm. Ab jetzt gilt

$$x_i \in [a, b], \quad i = 1, \ldots, n. \tag{2.25}$$

Allgemeine Bezeichnungen sind:

Notation 2.7. *Die x_i in Formel (2.24) werden als* ***Knoten*** *oder* ***Stützstellen*** *bezeichnet, die α_i als* ***Gewichte****.*

Zu ersten numerischen Ansätzen gelangt man naiv und intuitiv. Die Abbildung 2.3 demonstriert zwei solche Modelle. Sie präsentiert die Mittelpunkts- und Trapezregel als Beispiele für äquidistant gewählte Stützstellen. Die graphischen Qualitätsunterschiede der Integrationsverfahren sind in Abbildung 2.3 allerdings besonders ausgeprägt. Tatsächlich besitzen beide Verfahren den selben Exaktheitsgrad. Später wird noch auf den charakteristischen Unterschied beider Verfahren eingegangen.

Notation 2.8. *Ein numerisches Integrationsverfahren besitzt den* ***Exaktheitsgrad*** *oder die* ***Genauigkeit*** *m, wenn sie alle Polyome p mit* $\operatorname{Grad} p \leq m$ *exakt integriert, und m die größtmöglichste Zahl mit dieser Eigenschaft ist.*

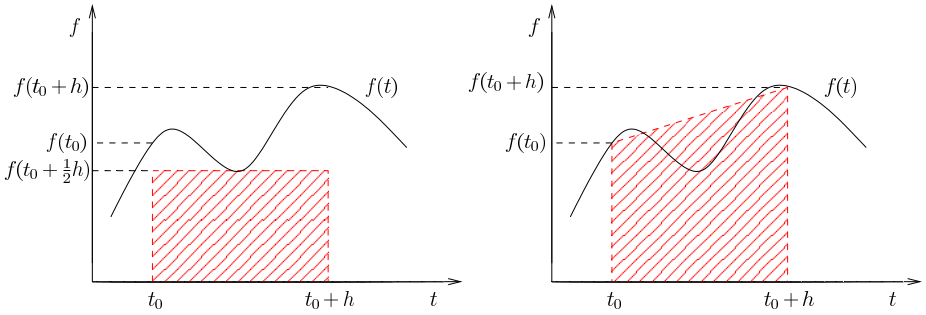

Abbildung 2.3: Mittelpunkts- (links) und Trapezregel (rechts)

Typischerweise wählt man nach ersten Überlegungen ganz unbefangen die Knoten x_i äquidistant und erhält durch diese Entscheidung die klassischen Newton-Cotes-Formeln. Newton-Cotes-Formeln sind in jedem Standardwerk der Numerik vorzufinden. Die resultierenden Ergebnisse sind weitgehend bekannt. Um nur eine konkrete Quelle zu nennen, sei für die folgenden Resultate auf [26, Abschnitte 5.1 und 5.2] verwiesen.
Die Gewichte α_i ergeben sich über die Lagrange-Polynome.

Definition 2.9 (Lagrange-Polynome). *Für paarweise verschiedene Stützstellen $x_1, \ldots, x_n$ sind die* `Lagrange-Polynome` *l_i definiert durch*

$$l_i(x) := \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}, \quad i = 1, \ldots, n.$$

mit $\operatorname{Grad} l_i = n - 1$. *Es gilt*

$$l_i(x_k) \quad = \quad \begin{cases} 1, & i = k, \\ 0, & i \neq k, \end{cases}$$

Der Hilfsabschnitt 2.8 enthält einige Aussagen über Lagrange-Polynomen.

Bemerkung 2.10. *Für die Lagrange-Polynome ist auch die Bezeichnung L_i gebräulich; dies kollidiert in dieser Arbeit aber mit der eingeführten Bezeichnung für die Legendre-Polynome.*

Oft wird zur Verallgemeinerung das Intervall $[a, b]$ auf eine bestimmte Länge normiert. Im Laufe des Abschnitts wird ferner meistens mit dem Intervall $[-1, 1]$ gearbeitet. Dies ist keine Einschränkung. Ein Integral läßt sich beliebig zwischen den Intervallen $[a, b]$ und $[c, d]$ umformen.

Bemerkung 2.11 (Integral-Transformation). *Mittels der affinen Abbildung*

$$x = \frac{b-a}{d-c} \cdot t + \frac{ad-bc}{d-c}, \quad c \neq d,$$

kann man folgende Transformation durchführen:

$$\int_a^b f(x)\,dx = \frac{b-a}{d-c} \int_c^d f\left(\frac{b-a}{d-c} \cdot t + \frac{ad-bc}{d-c}\right) dt,$$

wie man mit der Substitutionsregel der Integration berechnen kann. Für $[c,d] = [-1,1]$ *lautet die Transformation somit:*

$$x = \frac{b-a}{2} \cdot t + \frac{a+b}{2}.$$

Es wird zwischen offenen und geschlossenen Newton-Cotes-Formeln unterschieden. Mit den Bezeichnungen wird ausgedrückt, ob die Intervallendpunkte als Knoten genutzt werden. So ist die Mittelpunktsregel aus Abbilding 2.3 eine offene Newton-Cotes-Regel, die Trapezregel eine geschlossene Newton-Cotes-Regel. Bei fester Knotenzahl n unterscheiden sich beide Klassen in ihren Exaktheitsaussagen. Solche Unterschiede entstehen auch zwischen geraden und ungeraden Knotenzahlen n. Diese leichten Diskrepanzen sollen aber nicht erläutert werden, da sie quantitativ zweitrangig sind. Bei festgehaltenem n bewirken sie lediglich immer nur Unterschiede um einen Exaktheitsgrad. Bei der Gauß-Quadratur treten derartige Differenzen überhaupt nicht auf.
Der folgende Satz zu offenen Newton-Cotes-Formeln mit geradem n und äquidistanten Stützstellen reicht deshalb aus, die entscheidenen Resultate zusammenzufassen.

Satz 2.12 (Ergebnisse zur Newton-Cotes-Integration). *Es sei die Aufgabenstellung (2.23) gegeben. Die Stützstellen* x_i, $i = 1\ldots,n$ *seien auf dem Intervall* $(-1,1)$ *äquidistant verteilt.* n *sei gerade. Dann hat die* `Newton-Cotes-Formel` *mit den Gewichten*

$$\alpha_i = \int_{-1}^{1} l_i(x)\,dx, \quad i = 1,\ldots,n,$$

mindestens den Exaktheitsgrad $n-1$.

Beweis. Das Grundschema des Beweises verläuft ganz einfach, indem man den Integranden an den vorgegebenen Stützstellen x_i durch die Summe $\sum_{i=1}^{n} f(x_i)l_i(x)$ interpoliert. Damit folgen bereits die Genauigkeit und die Gewichte für gerades n. Genaueres findet man zum Beispiel in [26, Abschnitt 5.1]. □

Mit den Variationen zwischen offenen und abgeschlossenen sowie geraden und ungeraden Newton-Cotes-Regeln ergibt sich hier verallgemeinernd etwa der Exaktheitsgrad n. Es läßt sich beobachten:

Bemerkung 2.13. *Bei geschlossenen Newton-Cotes-Formeln treten ab $n = 9$ negative Gewichte auf, bei offenen Newton-Cotes-Formeln bereits ab $n = 3$, siehe [47, Tabelle 9.2 und Tabelle 9.6].*

Bei numerischen Anwendungen wirkt sich diese Erscheinung nachteilhaft aus, etwa durch Auslöschungen. Es empfielt sich, derartige Regeln zu vermeiden. Diese Eigenschaft verstärkt zusätzlich zur geringen Exaktheit die Unterlegenheit der Newton-Cotes-Regeln gegenüber der Gauß-Quadratur.
Auf die folgende interessante Bemerkung wird später noch zurückgegriffen.

Bemerkung 2.14. *Für den Exaktheitsgrad $n-1$ im Satz 2.12 wird die Äquidistanz der Stützstellen nicht gebraucht. Für die Lagrange-Polyome in Definition 2.9 benötigt man lediglich die paarweise Verschiedenheit der Knoten.*

Zum Abschluß des Abschnitts rufe man sich noch einmal die Exaktheit als Qualitätsmerkmal in Erinnerung. Hinter dieser Eigenschaft eines Integrationsverfahrens versteckt sich die Vorstellung, eine Funktion in eine Taylor-Reihe entwickeln zu können. Eine derartige polynomiale Approximation rechtfertigt dann dieses Qualitätsmerkmal, erzeugt aber auch eine Abhängigkeit von Differenzierbarkeitseigenschaften. Dies gilt auch für die Theorie der Gauß-Quadratur.
Integrationsverfahren werden aber normalerweise nicht auf spezielle Funktionenklassen zugeschnitten. Außerdem mangelt es an anderen Bewertungsmerkmalen. Die Nachteile - oder besser Unzulänglichkeiten - werden durchaus auch mit anderen Integrationsverfahren geteilt.

2.6 Gauß-Quadratur

Nun wird die entscheidende Verbesserung der bisherigen Ideen durchgeführt. Die Knoten x_i werden nicht mehr präventiv fest gewählt (insbesondere äquidistant), sondern vorerst frei gelassen. Damit gelangt man zur Gauß-Quadratur. In diesem Abschnitt wurden vor allem die Werke [53], [55], [62], [47] und [26] herangezogen. Gibt man die Knoten nicht vor, sind sowohl die n Gewichte als auch die n Knoten variabel wählbar. Man verfügt also über $2n$ freie Parameter. Intuitiv könnte man erwarten, daß damit der Exaktheitsgrad $2n-1$ erreichbar wäre. Das wird sich auch einstellen.
Die allgemeine Aufgabenstellung lautet wieder

$$I[f] = \int_{-1}^{1} \omega(x) f(x)\, dx. \tag{2.26}$$

Zunächst als Präludium:

Satz 2.15. *Der Exaktheitsgrad einer Quadraturregel zur Formel (2.26) beträgt höchstens* $2n-1$.

Beweis (Gegenbeispiel). Die Knoten der Quadraturformel seien $x_1, \ldots, x_n$ paarweise verschieden und in $[-1,1]$ gelegen. Die zu integrierende Funktion sei

$$f(x) := \prod_{i=1}^{n} (x - x_i)^2 \geq 0, \quad x \in [-1,1]$$

und hat den Grad $2n$. f ist nicht identisch null im Intervall $[-1,1]$. Damit gilt

$$I[f] = \int_{-1}^{1} \omega(x) f(x)\, dx > 0.$$

Die Quadraturformel liefert jedoch wegen $f(x_i) = 0$:

$$I_n[f] = \sum_{i=1}^{n} \alpha_i f(x_i) = 0, \quad \text{und somit} \quad R_n[f] = I[f] - I_n[f] \neq 0.$$

□

Als nächstes wird der Hauptsatz des Abschnitts vorgestellt. Für $\omega \equiv 1$ sind die orthogonalen Polynome die Legendre-Polynomen.

Satz 2.16. *Für das Integrationsproblem* $\int_{-1}^{1} \omega(x) f(x)\, dx$ *mit der Gewichtsfunktion* ω *existiert genau eine Quadraturformel*

$$I_n[f] = \sum_{i=1}^{n} \alpha_i f(x_i), \quad x_i \in [-1,1]$$

mit n *Integrationsstützstellen* x_i*, die den maximalen Genauigkeitsgrad* $2n-1$ *besitzt. Die Stützstellen* x_i *sind die Nullstellen des zur Gewichtsfunktion* ω *gehörenden orthogonalen Polynoms* p_n*. Als Gewichte ergeben sich*

$$\alpha_i \;=\; \int_{-1}^{1} \omega(x)\, l_i(x)\, dx = \int_{-1}^{1} \omega(x)\, l_i^2(x)\, dx > 0, \quad i = 1, \ldots, n,$$

die damit alle echt positiv sind.

Beweis (Existenz). $x_1, \ldots, x_n \in (-1,1)$ sind paarweise verschieden. Zu diesen Stützstellen existiert eine Newton-Cotes-Formel mit Genauigkeit $n-1$ nach der Bemerkung 2.14 zu Satz 2.12. Sei p ein beliebiges Polynom mit $\operatorname{Grad} p \leq 2n-1$.

p_n ist ein orthogonales Polynom im Sinne von Abschnitt 2.3. Wird nun p durch p_n dividiert, erhält man

$$p(x) = q(x)p_n(x) + r(x),$$

wobei r und q Polynome darstellen (r ist ein Restterm), mit

$$\operatorname{Grad} q \leq n-1, \quad \operatorname{Grad} r \leq n-1.$$

Wegen der Orthogonalitätseigenschaft von p_n, siehe (2.9), folgt:

$$\int_{-1}^{1} \omega(x)p(x)\,dx = \int_{-1}^{1} \omega(x)q(x)\,p_n(x)\,dx + \int_{-1}^{1} \omega(x)r(x)\,dx = \int_{-1}^{1} \omega(x)r(x)\,dx.$$

Aufgrund der Exaktheit $n-1$ der Newton-Cotes-Regel gilt mit deren Gewichten:

$$\begin{aligned} \sum_{i=1}^{n} \alpha_i p(x_i) \quad &= \quad \sum_{i=1}^{n} \overbrace{\alpha_i q(x_i) p_n(x_i)}^{=0\ \forall i} + \sum_{i=1}^{n} \alpha_i r(x_i) \\ = \sum_{i=1}^{n} \alpha_i r(x_i) \quad &\overset{\text{Satz 2.12}}{=} \quad \int_{-1}^{1} \omega(x)r(x)\,dx = \int_{-1}^{1} \omega(x)p(x)\,dx. \end{aligned}$$

Somit ist die obige Quadraturformel exakt für Polynome des Grades $\leq 2n-1$ und nach Satzes 2.15 folglich von maximaler Exaktheit. Da l_i^2 vom Grade $2n-2 \leq 2n-1$ ist, liefert die Exaktheit der Gauß-Quadratur:

$$0 < \int_{-1}^{1} \omega(x) l_i^2(x)\,dx = \sum_{j=1}^{n} \alpha_j l_i^2(x_j) = \alpha_i, \quad i = 1, \ldots, n.$$

Somit sind die Gewichte alle echt positiv. Auf den Beweis der Eindeutigkeit wird verzichtet, siehe auch [53, Satz 7.6]. □

Die Positivität aller Gewichte unterscheidet die Gauß-Quadratur von den Newton-Cotes-Formeln, siehe Bemerkung 2.13. Das verhindert numerische Auslöschungen bei hohen n.
Besonders wichtig ist für diese Diplomarbeit der Fall $\omega \equiv 1$. Integriert man die Funktion $f \equiv 1$ für diese Gewichtsfunktion, folgt die Summe der Gewichte:

$$\sum_{i=1}^{n} \alpha_i = \int_{-1}^{1} 1\,dx = 2. \tag{2.27}$$

Auf das Einheitsintervall normiert ergäbe die Summe Eins. Man kann sogar eine explizite Formel für die einzelnen Gewichte herleiten:

Satz 2.17. *Die Gewichte der Gauß-Quadratur mit der Gewichtsfunktion $\omega \equiv 1$ auf dem Intervall $[-1,1]$ sind gegeben durch*

$$\alpha_i = \frac{2}{(1-x_i^2)(\dot{L}_n(x_i))^2}, \quad i = 1, \ldots, n.$$

Dabei ist $\dot{L}_n$ ist Ableitung des Legendre-Polynoms von Formel (2.13). Als Folgerung des Satzes 2.4 kann der Nenner niemals null werden.

Beweis. Siehe [21, Abschnitt 7.3] oder [9, Proposition 4.2]. □

Man kann die Gewichte einer Gauß-Legendre-Regel also problemlos für jedes n berechnen und tabellieren. Zusammen mit den Nullstellen der Legendre-Polynome sind sie in [1, Kapitel 25] aufgelistet.
Ein bedeutsames Qualitätsmerkmal einer Integrationsvorschrift ist die Integrationsfehlerdarstellung. Für die Gauß-Quadratur gilt:

Satz 2.18 (Fehlerterm für Gauß-Quadratur). *Für $f \in C^{2n}([a,b])$ gilt mit einem $\xi \in (a,b)$:*

$$\int_a^b \omega(x) f(x)\, dx - \sum_{i=1}^{n} \alpha_i f(x_i) = \frac{f^{(2n)}(\xi)}{(2n)!} \langle p_n, p_n \rangle_\omega .$$

Beweis. In [42, Abschnitt 4.3] wird der Beweis mit dem Mittelwertsatz der Integralrechnung geführt, während [55, Abschnitt 3.6] Resultate der Hermite-Interpolation verwendet. □

Die Fehlerabschätzung des Satzes 2.18 ist zunächst abstrakt und verrät ohne angegebene Funktion f wenig. Die Differenzierbarkeitsvoraussetzungen sind zudem stark. Jedoch läßt sich der Fehlerterm trotz des unbekannten ξ bei genügend glatten Ableitungen im Intervall $[a,b]$ oftmals nach oben abschätzen. Auch ohne Berechnung der Ableitungen kann man bei ausreichender Glattheit der Funktion hoffen, daß der Fehler mit n schnell abnimmt. Jedenfalls weist der Term $(2n)!$ im Nenner auf ein günstiges Verhalten hin. Bei gegebener Gewichtsfunktion läßt sich das Skalarprodukt $\langle p_n, p_n \rangle_\omega$ im voraus berechnen. Man kann beispielsweise zeigen:

Bemerkung 2.19. *Für $\omega \equiv 1$ gilt:*

$$\langle p_n, p_n \rangle_\omega = \langle L_n, L_n \rangle_{\omega \equiv 1} = \frac{(n!)^4 2^{2n+1}}{((2n)!)^2 (2n+1)},$$

siehe [30, Kaptitel 7, §3] oder [62, S.241 zusammen mit Satz 3.7]. Wertvoll ist dieser Term für Fehlerabschätzungen umso mehr, da in dieser Diplomarbeit die konstante Gewichtsfunktion $\omega \equiv 1$ fokussiert wird.

Der Satz 2.18 und vorherige sollten nicht zu Fehlschlüssen verleiten: Die Analyse von Gauß-Quadratur darf sich nicht darauf beschränken, das Intervall $[a, b]$ fest zu wählen und dann n variieren zu lassen. Insbesondere wird n bei praktischen Anwendungen nicht beliebig vergrößert. Die Verkleinerung des Intervalls ist genauso wichtig. Die dazugehörige Analyse ist vergleichbar mit der Konsistenzanalyse bei numerischen Verfahren zur Lösung von Differentialgleichungen (vgl. Abschnitt 3.4). Das Intervall $[a, b]$ kann man insofern auch als Intervall $[a, a+h]$ interpretieren. Damit wird auch die lokale Aussage der Fehlertermschätzung betont, denn man muß davon ausgehen, daß das Intervall $[a, a + h]$ nur einen Teilbereich des zu integrierenden Intervalles darstellt. Die Intervalllänge h ist dann implizit im Fehlerterm des Satzes 2.18 enthalten. Der nächste Satz ist deswegen im Grunde genommen eine Folgerung. Er deckt nicht nur den in diesem Abschnitt fokussierten Fall der maximalen Exaktheit $2n-1$ ab, sondern auch Abschwächungen. Jene werden aber erst in Abschnitt 2.7 eingehender besprochen und analysiert.

Satz 2.20. *Es sei n die Anzahl der Knoten und $\omega \equiv 1$. Die Qudraturformel besitze die Eigenschaft, Polynome vom Grade m exakt zu integrieren. Der Integrand f sei im Intervall $[a, a + h]$ wenigstens $(m + 1)$-mal stetig differenzierbar. Dann gibt es eine positive Konstante c und ein $\xi \in [a, a + h]$, so daß für den Restterm $R_n[f, h]$ des Integrals*

$$\int_a^{a+h} f(x)\,dx \tag{2.28}$$

die Abschätzung

$$|R_n[f, h]| \leq c \cdot h^{m+2} \max_{a \leq \xi \leq a+h} \left|f^{m+1}(\xi)\right| \tag{2.29}$$

gilt. Für den maximalen Exaktheitsgrad $2n - 1$ gilt dann

$$|R_n[f, h]| \leq c \cdot h^{2n+1} \max_{a \leq \xi \leq a+h} \left|f^{2n}(\xi)\right|. \tag{2.30}$$

Beweis. In [18, Lemma 6.39] findet man den komplette Beweis. Für die maximale Ordnung $2n - 1$ nutzt man als erstes die Transformationsregel für die Intervallgrenzen der Bemerkung 2.11:

$$\int_a^{a+h} f(x)\,dx = \frac{h}{2} \int_{-1}^{1} f\left(\frac{h}{2} \cdot t + \frac{2a + h}{2}\right) dt. \tag{2.31}$$

Jetzt wendet man den Satz 2.18 auf die rechte Seite von (2.31) an. Das Argument von f hängt im wesentlichen von $h \cdot t$ ab. Dann entsteht beim $2n$-fachen Differenzieren h^{2n} als Vorfaktor. Dann vergegenwärtigt man sich, daß nach der Bemerkung 2.19 das Skalarprodukt $\langle p_n, p_n \rangle_\omega$ eine Konstante ist. □

Ebenso wie die schon erwähnten Konsistenzanalysen beschreibt auch dieser Satz 2.20 keine globale Abschätzung, sondern eine lokale. Der Satz läßt sich auch direkt mit dem Satz 2.18 verknüpfen:

Bemerkung 2.21. *Im Term $\langle p_n, p_n \rangle_\omega$ des Satzes 2.18 steckt implizit die Schrittweite h der Satzes 2.20, da das Skalarprodukt ein Integral ist. Ebenso findet sich h im Argument des Ableitungsterms von Satz 2.18, so daß durch das $2n$-fache Differenzieren die Vorfaktoren des Satzes 2.20 enstehen.*

Es ist überdies einsichtig, daß die Fehlertermordnung den Exaktheitsgrad um zwei übersteigt, denn Exaktheit des Grades m bedeutet, daß für ein Polynom des Grades $\mu > m$ mit den reellen Koeffizienten c_i gilt:

$$\begin{aligned}
&\int_a^{a+h} \left(\sum_{i=0}^{\mu} c_i (t-a)^i \right) dt \\
= &\int_a^{a+h} \left(\sum_{i=0}^{m} c_i (t-a)^i \right) dt + \int_a^{a+h} \left(\sum_{i=m+1}^{\mu} c_i (t-a)^i \right) dt \\
= &\sum_{i=0}^{m} \frac{c_i}{i+1} h^{i+1} + \sum_{i=m+1}^{\mu} \frac{c_i}{i+1} h^{i+1} \\
= &\sum_{i=0}^{m} \frac{c_i}{i+1} h^{i+1} + h^{m+2} \sum_{i=0}^{\mu-m-1} \frac{c_{m+1+i}}{m+2+i} h^i. \qquad (2.32)
\end{aligned}$$

Die erste Summe in (2.32) wird numerisch exakt integriert, der zweite Term verhält sich wie $\mathcal{O}(h^{m+2})$, da die Summe beschränkt werden kann.
Der nächste Satz komplettiert die Fehlertermdiskussion. Anknüpfend an die nach Bemerkung 2.19 geäußerten Gedanken behandelt er jetzt genau die umgekehrte Situation: Auf einem festen Intervall wird die Anzahl der Stützstellen immer größer gewählt, und dann das Konvergenzverhalten für $n \to \infty$ untersucht. An f werden nur schwache Voraussetzungen gestellt.

Satz 2.22 (Konvergenz bei stetigen Funktionen). *x_i, $i = 1, \ldots, n$ seien die Knoten der Gauß-Quadratur. Es sei ω eine Gewichtsfunktion und $f \in C^0([-1,1])$, also stetig im Intervall $[-1,1]$. Dann gilt*

$$\lim_{n\to\infty} \left| \int_{-1}^{1} \omega(x) f(x)\, dx - \sum_{i=0}^{n} \alpha_i f(x_i) \right| = 0.$$

Beweis. Der Beweis verwendet den Weierstraßschen Approximationssatz (siehe [48, Satz 7.26] und kann in [42, Abschnitt 4.4] oder [22, Theorem 5.13] nachgelesen werden. □

Kurioserweise besitzen Newton-Cotes-Formeln nicht die Eigenschaft, für jede stetige Funktion im Intervall $[-1,1]$ für $n \to \infty$ zu konvergieren, wie in [21, Theorem 6.2.3] gezeigt wird. Der Satz 2.22 enthält keine quantitative Konvergenzangabe. Die Konvergenzordnung wird maßgeblich durch Zusatzeigenschaften der Funktion f bestimmt, vor allem durch zusätzliche Differenzierbarkeitsvoraussetzungen. In [47, Abschnitte 10.2-10.4] kann man Resultate dazu nachlesen.
In Tabelle 2.1 sind Nullstellen und Gewichte der ersten Gauß-Quadraturformeln aufgelistet.

Grad L_n	Nullstellen	Gewichte
$n=1$	0.0000000000000000	2.0000000000000000
$n=2$	−0.5773502691896258	1.0000000000000000
	0.5773502691896258	1.0000000000000000
$n=3$	−0.7745966692414834	0.5555555555555556
	0.0000000000000000	0.8888888888888889
	0.7745966692414834	0.5555555555555556
$n=4$	−0.8611363115940526	0.3478548451374539
	−0.3399810435848563	0.6521451548625461
	0.3399810435848563	0.6521451548625461
	0.8611363115940526	0.3478548451374539

Tabelle 2.1: Gauß-Quadraturformeln für das Intervall $[-1,1]$

Ein quantitatives Beispiel zur Gauß-Quadratur wird in [53, Abschnitt 7.4] betrachtet.
An sich übertrifft die Gauß-Quadratur für glatte Funktionen die Genauigkeit anderer, höchst effizienter Methoden, wie etwa der Romberg-Extrapolation. Die Gauß-Quadratur besitzt aber auch ernste Nachteile:

1. Für jede Anzahl n müssen die Knoten mit numerischen Verfahren berechnet und dann zusammen mit den Gewichten verwaltet werden.

2. Die schwierigen Fehlerabschätzungen verhindern, daß man einer Integrationsaufgabe eine Knotenanzahl n zuordnen kann.

3. Reicht die Genauigkeit einer numerischen Integrationsaufgabe für n nicht aus, kann man alte Werte für eine größere Knotenanzahl (etwa $n+1$) nicht wiederverwenden. Dies ist ein ein Vorteil von Verfahren wie der Romberg-Extrapolation.

Zum Abschluß des Abschnittes sei auf ergänzende Aspekte verwiesen, die aber später nicht aufgegriffen werden. Zum Teil beseitigen sie die aufgeführten Schwächen.

- Es existieren Algorithmen zur Bestimmung der Nullstellen der Legendre-Polynome. Hierbei kann statt einem Nullstellenproblem ein Eigenwertproblem einer symmetrischen, tridiagonalen Matrix gelöst werden, was stabil und effizient mit dem QR-Algorithmus geleistet werden kann. Ähnliches gilt für die Gewichte (vgl. [53, Satz 7.7], [55, Abschnitt 3.6], [50, Abschnitt 3.2]).
- Werte für die Gauß-Quadratur sind in [1, Kapitel 25] und [49, Anhang I] tabelliert.
- Man kann bei Gauß-Quadraturregeln adaptive Integration einführen, siehe [53, Abschnitt 7.5].
- Eingebettete Gauß-Regeln beheben die Nichtverwertbarkeit von alten Ergebnissen, siehe etwa [40] und [45]. Beispielregeln findet man in [44].
- Man kann die Gauß-Quadratur auch auf uneigentlichen Integralen abwandeln, siehe [30, Abschnitt 7.3.5].
- Integrale mit Singularitäten werden in [55, Abschnitt 3.7] untersucht.

2.7 Modifikationen der Gauß-Quadratur

In diesem Abschnitt werden gewisse Abwandlungen der Gauß-Quadraturregeln des letzten Abschnittes analysiert. Im Fokus steht der wichtige Spezialfall $\omega \equiv 1$, wobei allerdings auch mit anderen Gewichtsfunktionen argumentiert wird. Im letzten Abschnitt war p_n als orthogonales Polynom orthogonal zu allen Polynomen niedrigeren Grades. Angenommen, man fordert nicht wie bisher für p_n

$$\langle p_n, p \rangle_\omega = 0, \quad p \in P_{n-1}, \tag{2.33}$$

sondern nur die Abschwächung

$$\langle p_n^{mod}, p \rangle_\omega = 0, \quad p \in P_{n-m},\ m \geq 2, \tag{2.34}$$

so ist p_n^{mod} nur orthogonal zu Polynomen vom Grade $\leq n-m$. p_n^{mod} wäre also nicht mehr ein orthogonales Polynom im bisher verwendeten Sinne von Abschnitt 2.3, sondern ein modifiziertes, abgeschwächtes orthogonales Polynom. Auf solche Abarten der bisherigen orthogonalen Polynome wird in diesem Abschnitt 2.7 näher eingegangen. Man kann aber schon jetzt erkennen, daß solche „orthogonalen“ Polynome auch den Exaktheitsgrad zugehöriger Quadraturregeln beeinflussen. Man kann erahnen, daß die Genauigkeit dann $n - m$ beträgt (vgl. z.B. Satz 2.20).
Modifikationen der Gauß-Quadratur sind erforderlich, da man oft bestimmte Knoten frei von Exaktheitsdiskussionen festlegen möchte. Üblicherweise handelt es

sich dabei um die Endpunkte des Integrationsintervalls, für das Intervall $[-1,1]$ also um -1 und 1. Weiterhin soll zunächst $\omega \equiv 1$ gelten. Maßgeblich sind folgende drei Regeln:

- Gauß-Radau-I(links): $x_1 = -1$ ist ein Knoten,
- Gauß-Radau-II(rechts): $x_n = 1$ ist ein Knoten,
- Gauß-Lobatto: $x_1 = -1$ und $x_n = 1$ sind Knoten.

Zunächst wird das Intervall $[-1,1]$ temporär auf $[0,1]$ transformiert. Dies ist zwischenzeitlich etwas prakischer und weist auch bereits auf die gebräuchliche Schreibweise bei Runge-Kutta-Methoden hin. Die Transformation geschieht mittels der Substitution $x := 2t - 1$:

$$L_n(x) = \frac{1}{2^n \cdot n!} \cdot \frac{d^n}{dx^n}\left[(x^2-1)^n\right] \longrightarrow L_n(2t-1) = \frac{1}{n!} \cdot \frac{d^n}{dt^n}\left[t^n(t-1)^n\right],$$

wobei $t \in [0,1]$ ist. Zunächst soll eine Überlegung angestellt werden, die bei der anstehenden Erarbeitung viel Aufwand ersparen wird. Betrachtet wird der exemplarische Fall $t_1 = 0$, also ein transformiertes Radau-I-Verfahren. Bei insgesamt n Stützstellen kann man nur noch erwarten, daß

$$p_n^{mod}(t) = (t-t_1)\cdot(t-t_2)\cdot\ldots\cdot(t-t_n) = t \cdot \prod_{i=2}^{n}(t-t_i) \tag{2.35}$$

orthogonal zu Polynomen p mit $\operatorname{Grad} p \leq n-2$ ist:

$$\left\langle p_n^{mod}, p\right\rangle_\omega = \int_0^1 (t-t_1)\cdot\prod_{i=2}^{n}(t-t_i)\cdot p(t)\,dt = \int_0^1 t\cdot\prod_{i=2}^{n}(t-t_i)\cdot p(t)\,dt. \tag{2.36}$$

Man faßt nun $\omega(t) := t$ als Gewichtsfunktion auf. $\omega(t) := t$ ist nach der Definition 2.1 als Gewichtsfunktion zulässig. Damit ergeben sich sofort alle Resultate aus den Abschnitten 2.4 und 2.6. So kann man $\tilde{p}_{n-1}(t) = (t-t_2)\cdot\ldots\cdot(t-t_n)$ nach Satz 2.3 wieder als Polynom konstruieren, welches zu Polynomen p mit $\operatorname{Grad} p \leq n-2$ bezüglich $\omega(t) = t$ orthogonal ist. Auch liegen dann alle Nullstellen t_i, $i = 2,\ldots,n$ im Intervall $(0,1)$. Nach Satz 2.16 sind die zu t_i, $i = 2,\ldots,n$ gehörenden Gewichte α_i alle größer Null. Ganz analog argumentiert man (mit dann verändertem p_n^{mod}):

- Radau-II (Radau rechts):

$$\left\langle p_n^{mod}, p\right\rangle_\omega = \int_0^1 (t-1)\cdot\prod_{i=2}^{n}(t-t_i)\cdot p(t)\,dt, \quad \operatorname{Grad} p \leq n-2, \tag{2.37}$$

- Lobatto:

$$\left\langle p_n^{mod}, p \right\rangle_\omega = \int_0^1 t \cdot (t-1) \cdot \prod_{i=3}^{n} (t - t_i) \cdot p(t)\, dt, \quad \operatorname{Grad} p \leq n-3. \tag{2.38}$$

Interessant und studierenswert ist der Zusammenhang dieser drei Abarten mit den Legendre-Polynomen. Definiert man auf $t \in [0,1]$ mit λ, $\mu \in \mathbb{R}$

$$L_{n,\lambda,\mu}(2t-1) := L_n(2t-1) + \lambda L_{n-1}(2t-1) + \mu L_{n-2}(2t-1), \tag{2.39}$$

so kann man den Zusammenhang von $L_{n,\lambda,\mu}$ mit den soeben gefundenen Polynomen p_n^{mod} untersuchen. Aufgrund der Linearität des Skalarproduktes ist $L_{n,\lambda,\mu}$ je nach Wahl der Parameter λ und μ orthogonal mindestens zu Polynomen des Grades $n-3$. Ebenso wie eben soll der Zusammenhang vor allem anhand des Radau-I-Verfahrens analysiert werden. Hierbei wird indirekt vorgegangen, indem gezeigt wird, daß für gewisse $L_{n,\lambda,\mu}$ die anfangs beschriebenen Verfahren erzeugt werden.

Satz 2.23 (Radau- und Lobatto-Verfahren). *Die Knoten t_i, $i = 1, \ldots, n$ eines* `Radau-I-Verfahrens` *sind die n Nullstellen des Polynoms*

$$L_{n,1,0}(2t-1) = \frac{2}{(n-1)!} \cdot \frac{d^{n-1}}{dt^{n-1}} \left[t^n (t-1)^{n-1} \right], \quad t_1 = 0. \tag{2.40}$$

Bei einem `Radau-II-Verfahrens` *sind die Knoten die Nullstellen von*

$$L_{n,-1,0}(2t-1) = \frac{2}{(n-1)!} \cdot \frac{d^{n-1}}{dt^{n-1}} \left[t^{n-1} (t-1)^{n} \right], \quad t_n = 1. \tag{2.41}$$

Bei einem `Lobatto-Verfahrens` *sind die Knoten die Nullstellen von*

$$L_{n,0,-1}(2t-1) = \frac{2}{(n-2)!} \cdot \frac{d^{n-2}}{dt^{n-2}} \left[t^{n-1} (t-1)^{n-1} \right], \quad t_1 = 0,\ t_n = 1. \tag{2.42}$$

Alle Nullstellen der Polynome sind jeweils paarweise verschieden (vgl. Satz 2.4).

Beweis. Für das Radau-I-Verfahrens berechnet man:

$$\begin{aligned}
&= L_{n,1,0}(2t-1) = L_n(2t-1) + L_{n-1}(2t-1) \\
&= \frac{1}{n!} \cdot \frac{d^n}{dt^n} \left[t^n (t-1)^n \right] + \frac{1}{(n-1)!} \cdot \frac{d^{n-1}}{dt^{n-1}} \left[t^{n-1} (t-1)^{n-1} \right] \\
&= \frac{1}{n!} \cdot \frac{d^{n-1}}{dt^{n-1}} \left[n t^{n-1} (t-1)^n + n t^n (t-1)^{n-1} \right] \\
&+ \frac{1}{(n-1)!} \cdot \frac{d^{n-1}}{dt^{n-1}} \left[t^{n-1} (t-1)^{n-1} \right] \\
&= \frac{2}{(n-1)!} \cdot \frac{d^{n-1}}{dt^{n-1}} \left[t^n (t-1)^{n-1} \right].
\end{aligned}$$

Man sieht sofort, daß $L_{n,1,0}$ eine Nullstelle bei $t_1 = 0$ haben muß. Man wendet sukzessive den Satz von Rolle auf $\left[t^n (t-1)^{n-1}\right]$ an. Dabei erkennt man, daß die restlichen Nullstellen im Intervall $(0,1)$ liegen und alle verschieden sind. Aufgrund der Linearität des Skalarprodukts ist $L_{n,1,0}$ ferner orthogonal zu Polynomen p mit $\operatorname{Grad} p \leq n-2$. Aufgrund der Existenz und Eindeutigkeit aus Satz 2.3 sind die Knoten identisch mit denen der Polynome p_n^{mod}. Damit sind die Polynome $L_{n,1,0}$ und p_n^{mod} bis auf einen Faktor identisch. Der Nullstellensatz 2.4 liefert dann die restlichen Eigenschaften über die Nullstellen. Die Aussagen für die anderen Verfahren ermittelt man analog. □

Notation 2.24. *Das Polynom $L_{n,0,-1}$ wird im Laufe der Arbeit auch als* `Lobatto-Polynom` *bezeichnet.*

Die Tabelle 2.2 faßt die Gauß-Legendre-Regeln und die drei wichtigsten Sonderfälle zusammen.

Wichtige Gauß-Quadraturregeln			
Name	Polynom	Nullstellen durch:	Nullstellen
Gauß	L_n	$\frac{d^n}{dt^n}[t^n(t-1)^n] = 0$	$t_i \in (0,1)$
Radau-I	$L_{n,1,0}$	$\frac{d^{n-1}}{dt^{n-1}}[t^n(t-1)^{n-1}] = 0$	$t_i \in [0,1)$, $t_1 = 0$
Radau-II	$L_{n,-1,0}$	$\frac{d^{n-1}}{dt^{n-1}}[t^{n-1}(t-1)^n] = 0$	$t_i \in (0,1]$, $t_n = 1$
Lobatto	$L_{n,0,-1}$	$\frac{d^{n-2}}{dt^{n-2}}[t^{n-1}(t-1)^{n-1}] = 0$	$t_i \in [0,1]$, $t_1 = 0, t_n = 1$

Tabelle 2.2: Legendre-, Radau- und Lobatto-Gauß-Quadraturregeln

Die Abbildung 2.4 visualisiert die Nullstellen der Lobatto-Polynome im Intervall $[-1,1]$. Abgesehen von den Nullstellen -1 und 1 ähnelt die Positionierung der Nullstellen derjenigen von Abbildung 2.2 der Legendre-Polynome.

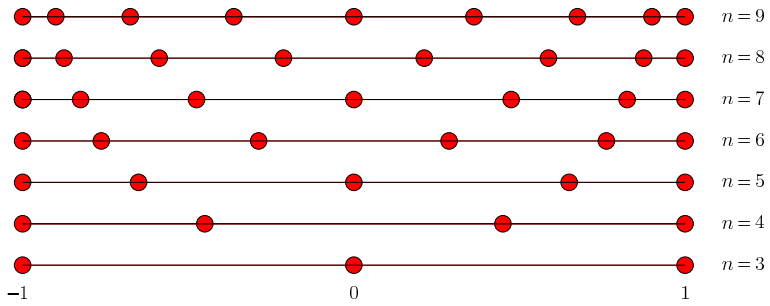

Abbildung 2.4: Nullstellen von $L_{n,0,-1}$ (Gauß-Lobatto) für $n = 3, \ldots, 9$

Für die Gewichte ändert sich der grundsätzliche Rechenansatz nicht, da im Satz 2.16 die Lage der Nullstellen nicht verwendet wurde. Der Ansatz

$$\alpha_i^{mod} = \int_{-1}^{1} \omega(x) l_i(x)\, dx, \quad i = 1, \ldots, n \tag{2.43}$$

für das Intervall $[-1, 1]$ bleibt bestehen, nur daß die Lagrange-Polynome l_i nun andere Knoten verwenden. Es lassen sich sogar wieder explizite Formeln herleiten. So gilt, vergleichbar mit Satz 2.17:

Satz 2.25 (Gewichte der Gauß-Lobatto-Integration). *Die x_i seien die Nullstellen der Lobatto-Polynome $L_{n,0,-1}$ im Intervall $[-1, 1]$. Dann ergeben sich die Integrationsgewichte*

$$\alpha_i^{mod} = \frac{2}{n(n-1)} \cdot \frac{1}{(L_{n-1}(x_i))^2}, \quad i = 1, \ldots, n.$$

Insbesondere sind alle Gewichte echt positiv.

Beweis. Siehe [9, Lemma 4.7]. □

Durch die beiden Knoten am Rand des Intervalls $[-1, 1]$ besitzen die Lobatto-Verfahren besondere Vorteile. Dies wird bei numerischen Verfahren zur Lösung von Anfangswert- und Randwertproblemen deutlich. Auf die Lobatto-Gewichte wird ab Kapitel 7 wieder zurückgegriffen. Für Fehlertermangaben sei auf den Satz 2.20 in Abschnitt 2.6 verwiesen.
Die Tabelle 2.3 listet die Daten der ersten drei Lobatto-Quadraturregeln für das Intervall $[-1, 1]$ auf. Die einzelnen Werte könnten hier noch mit Wurzelausdrücken und Brüchen dargestellt werden. Beispiel 8.2 demonstriert, daß viele Nachkommastellen für ausreichend genaue numerische Lösungen benötigt werden.
Im Abschnitt 4.6 werden Strukturen der Radau- und Lobatto-Quadraturverfahren bei impliziten Runge-Kutta-Verfahren auftreten.

2.8 Hilfsresultate

Dieser Abschnitt enthält verschiedene Resultate aus dem Bereich der orthogonalen Polynome und der Gauß-Quadratur. Die spezielle Struktur dieser Theorie führt zu oft verblüffenden mathematischen Beziehungen, die ermöglichen, daß Größen für Implementierungen nicht numerisch berechnet werden müssen. Die Herleitung dieser Zusammenhänge macht sich deshalb in Aufwandserleichterungen und Ungenauigkeitsreduzierungen bezahlt. Die Resultate des Abschnitts sind als Hilfssätze zu verstehen, auf die erst später zurückgegriffen wird, vor allem in den Kapiteln 4, 5 und 7. Mangels Quellen mußten die Beweise der Sätze 2.29 und 2.30 eigenständig geführt werden.
Es ist günstig, folgende Bezeichnung einzuführen:

Grad $L_{n,0,-1}$	Nullstellen	zugehörige Gewichte
$n = 3$	-1.0000000000000000000	0.3333333333333333
	0.0000000000000000000	1.3333333333333333
	1.0000000000000000000	0.3333333333333333
$n = 4$	-1.0000000000000000000	0.16666666666666667
	-0.44721359549995793928	0.8333333333333333
	0.44721359549995793928	0.8333333333333333
	1.0000000000000000000	0.16666666666666667
$n = 5$	-1.0000000000000000000	0.10000000000000000
	-0.65465367070797714379	0.54444444444444444
	0.0000000000000000000	0.71111111111111111
	0.65465367070797714379	0.54444444444444444
	1.0000000000000000000	0.10000000000000000

Tabelle 2.3: Lobatto-Quadraturregeln für das Intervall $[-1, 1]$

Notation 2.26.

$$g_n^{(k)}(x) := \frac{d^k}{dx^k}\Big(\frac{1}{2^n n!}(x^2-1)^n\Big) = \Big(\frac{1}{2^n n!}(x^2-1)^n\Big)^{(k)} \tag{2.44}$$

mit k, $n \in \mathbb{N}$ *und* $x \in [-1, 1]$.

Es gilt dann:

$$g_n^{(n)}(x) = L_n(x). \tag{2.45}$$

Im folgenden Satz findet diese Notation ihre erste Verwendung.

Satz 2.27. *Es gilt*

$$\frac{(x^2-1)}{n(n+1)} \cdot g_n^{(n+1)}(x) = g_n^{(n-1)}(x), \quad n \in \mathbb{N} \text{ und } n > 1, \quad x \in [-1, 1]. \tag{2.46}$$

Beweis. Man wendet zunächst den Binomischen Lehrsatz auf $(x^2-1)^n$ an, siehe (2.15). Dann leitet man auf beiden Seiten von (2.46) entsprechend oft ab. Der Koeffizientenvergleich der entstehenden Polynome schließt den Beweis ab. Einen analytischen Beweis findet man in [9, Proposition 3.4]. □

Dieser Satz wird bei den Gauß-Lobatto-Regeln verwendet. Der Term $g_n^{(n-1)}(x)$ erinnert stark an den Term $\frac{d^{n-2}}{dt^{n-2}}\left[t^{n-1}\,(t-1)^{n-1}\right]$ in der Tabelle 2.2. Andererseits gilt aber auch

$$g_{n-1}^{(n)}(x) = L'_{n-1}(x). \tag{2.47}$$

Dann kann man folgendes Schema zur Berechnung der Nullstellen für die Gauß-Lobatto-Knoten erstellen:

Folgerung 2.28. *Die Nullstellen eines Lobatto-Polynoms $L_{n,0,-1}$ lassen sich auch folgendermaßen berechnen:*

1. $x_1 := -1$,
2. $x_n := 1$,
3. *die restlichen x_i durch die Nullstellenmenge $\{L'_{n-1}(x) = 0\}$.*

Diese zunächst - im Vergleich zur Herleitung der Lobatto-Regeln im Abschnitt 2.7 - etwas eigentümliche Methode wird oft in Arbeiten erwähnt, zum Beispiel in [7, Abschnitt 2.1.2], [19, Abschnitt 2], [46, Beispiel 3.3] und [63, Abschnitt 2.1.3]. Die nächsten Sätze beschäftigen sich mit den Polynomen aus Satzes 2.27. Die Folgerung 2.28 wird in der erwähnten Literatur oft mit dem nächsten Satz verknüpft.

Satz 2.29. *Es seien y_i, $i = 1, \ldots, n$ beliebige reelle Zahlen. x_i, $i = 1, \ldots, n$ seien die Nullstellen des Lobatto-Polynoms $L_{n,0,-1}$. Die Punkte (x_i, y_i), $i = 1, \ldots, n$ sollen durch ein Polynom p interpoliert werden. Dann gilt*

$$p(x) = \sum_{l=1}^{n} y_l \cdot l_l(x) \quad \text{mit} \tag{2.48}$$

$$l_l(x) = \frac{1}{n(n-1)L_{n-1}(x_l)} \cdot \frac{(x^2-1)L'_{n-1}(x)}{(x-x_l)}, \quad l = 1, \ldots, n, \tag{2.49}$$

wobei die l_i Lagrange-Polynome sind.

Beweis. Der Zähler und $\frac{1}{n(n-1)}$ von (2.49) ist nach Satz 2.27 identisch mit dem Polynom $g_{n-1}^{(n-2)}$. Dieses Polynom kann man als Produkt schreiben, wie man aus (2.44) berechnen kann:

$$g_{n-1}^{(n-2)}(x) = \frac{(2(n-1))!}{2^{n-1}(n-1)!n!} \cdot \prod_{i=1}^{n}(x - x_i).$$

Der Nenneranteil L_{n-1} bildet sich aus

$$L_{n-1}(x) = g_{n-1}^{(n-1)}(x) = \frac{d}{dx}\left(g_{n-1}^{(n-2)}(x)\right) = \frac{(2(n-1))!}{2^{n-1}(n-1)!n!} \cdot \sum_{j=1}^{n} \prod_{\substack{i=1 \\ i \neq j}}^{n}(x - x_i)$$

und anschließender Auswertung bei x_l. Kürzt man die Vorfaktoren und $(x - x_l)$ des Nenners und setzt x_k ein, folgt:

$$l_l(x_k) = \frac{\prod_{i=1, i\neq l}^{n}(x_k - x_i)}{\sum_{j=1}^{n}\prod_{i=1, i\neq j}^{n}(x_l - x_i)} = \frac{\prod_{i=1, i\neq l}^{n}(x_k - x_i)}{\prod_{i=1, i\neq l}^{n}(x_l - x_i)} = \begin{cases} 1, & l = k, \\ 0, & l \neq k. \end{cases}$$

□

Der nächste Satz erlaubt eine Aussage über die Werte der Ableitungen der Funktionen l_l an den Knoten x_k.

Satz 2.30. *x_k, $k = 1, \ldots, n$ seien die Nullstellen des Lobatto-Polynoms $L_{n,0,-1}$. Dann gilt:*

$$D_{k,l} = l_l'(x_k) = \begin{cases} -\frac{n(n-1)}{4}, & l = k = 1, \\ \frac{n(n-1)}{4}, & l = k = n, \\ 0, & l = k,\ k = 2, \ldots, n-1, \\ \frac{L_{n-1}(x_k)}{L_{n-1}(x_l)\cdot(x_k - x_l)}, & \text{sonst.} \end{cases}$$

In Matrixschreibweise:

$$\begin{pmatrix} l_1'(x_1) & \ldots & l_n'(x_1) \\ \vdots & \ddots & \vdots \\ l_1'(x_n) & \ldots & l_n'(x_n) \end{pmatrix} = \begin{pmatrix} -\frac{n(n-1)}{4} & & \ldots & & \frac{L_{n-1}(x_1)}{L_{n-1}(x_n)(x_1 - x_n)} \\ & 0 & & & \\ \vdots & & \ddots & & \vdots \\ & & & 0 & \\ \frac{L_{n-1}(x_n)}{L_{n-1}(x_1)(x_n - x_1)} & & \ldots & & \frac{n(n-1)}{4} \end{pmatrix}$$

Die Matrixbezeichnung D wird ab Abschnitt 7.4 verwendet.

Beweis. Man muß offenbar zwischen zwei Fällen unterscheiden.
(I) $k \neq l$ ($\Longrightarrow x_k \neq x_l$): Nach Satz 2.29 kann man die Lagrange-Polynome folgendermaßen schreiben:

$$l_l(x) = \frac{1}{L_{n-1}(x_l)} \cdot \frac{(2(n-1))!}{2^{n-1}(n-1)!n!} \cdot \prod_{\substack{i=1 \\ i \neq l}}^{n} (x - x_i).$$

Abgeleitet ergibt sich:

$$\begin{aligned} l_l'(x) &= \frac{1}{L_{n-1}(x_l)} \cdot \frac{(2(n-1))!}{2^{n-1}(n-1)!n!} \cdot \sum_{\substack{j=1 \\ j \neq l}}^{n} \prod_{\substack{i=1 \\ i \neq l, i \neq j}}^{n} (x - x_i) \\ \Longrightarrow\ l_l'(x_k) &= \frac{1}{L_{n-1}(x_l)} \cdot \frac{(2(n-1))!}{2^{n-1}(n-1)!n!} \cdot \frac{1}{(x_k - x_l)} \cdot \sum_{\substack{j=1 \\ j \neq l}}^{n} \prod_{\substack{i=1 \\ i \neq j}}^{n} (x_k - x_i) \\ &= \frac{1}{L_{n-1}(x_l)} \cdot \frac{(2(n-1))!}{2^{n-1}(n-1)!n!} \cdot \frac{1}{(x_k - x_l)} \cdot \prod_{\substack{i=1 \\ i \neq k}}^{n} (x_k - x_i) \\ &\overset{\text{Satz 2.27}}{=} \frac{L_{n-1}(x_k)}{L_{n-1}(x_l) \cdot (x_k - x_l)}. \end{aligned}$$

Bei letzterer Gleichung nutzt man die Aussage des Satzes 2.27, wonach ein abgeleitetes Lobatto-Polynom $L_{n,0,-1}$ ein Legendre-Polynom L_{n-1} ist.
(II) $k = l$ ($\Longrightarrow x_k = x_l$): Beim Ableiten der Lagrange-Polynome ergibt sich nach der Quotientenregel:

$$l_l'(x) = \left(\frac{1}{L_{n-1}(x_l)} \cdot \frac{g_{n-1}^{(n-2)}(x)}{(x - x_l)} \right)' = \frac{1}{L_{n-1}(x_l)} \cdot \frac{g_{n-1}^{(n-1)}(x)(x - x_l) - g_{n-1}^{(n-2)}(x)}{(x - x_l)^2}.$$

Nach wie vor sind Zähler und Nenner jeweils null für $x = x_l$. Es sind die Voraussetzungen für die Regel von L'Hospital gegeben (vgl. zu L'Hospital [23, §16]).

$$\begin{aligned}
& \lim_{x \to x_l} \left(\frac{1}{L_{n-1}(x_l)} \cdot \frac{g_{n-1}^{(n-1)}(x)(x - x_l) - g_{n-1}^{(n-2)}(x)}{(x - x_l)^2} \right) \\
= & \lim_{x \to x_l} \left(\frac{1}{L_{n-1}(x_l)} \cdot \frac{g_{n-1}^{(n)}(x)(x - x_l) + g_{n-1}^{(n-1)}(x) - g_{n-1}^{(n-1)}(x)}{2(x - x_l)} \right) \\
= & \lim_{x \to x_l} \left(\frac{1}{L_{n-1}(x_l)} \cdot \frac{g_{n-1}^{(n)}(x)(x - x_l)}{2(x - x_l)} \right) \\
= & \lim_{x \to x_l} \left(\frac{g_{n-1}^{(n)}(x)}{2 \cdot L_{n-1}(x_l)} \right) = \lim_{x \to x_l} \left(\frac{L_{n-1}'(x)}{2 \cdot L_{n-1}(x_l)} \right)
\end{aligned}$$

Die x_l sind für $l = 2, \ldots, n-1$ nach Folgerung 2.28 genau die Nullstellen von L_{n-1}'. Für $l = 1$ und $l = n$ nutzt man die Rekursion (2.16) und das Lemma 2.6. Die Rekursionsformel (2.16) läßt sich ableiten. Dann kann ähnlich einfach wie (2.20) und (2.21) gezeigt werden, daß

$$L_{n-1}'(1) = \frac{n(n-1)}{2} \quad \text{und} \quad L_{n-1}'(-1) = (-1)^n \cdot \frac{n(n-1)}{2}$$

gelten, womit die Ableitungswerte für x_1 und x_n folgen. □

In [19, Abschnitt 2] und [20, Abschnitt 2] haben sich Schreibfehler in die Matrix D eingeschlichen.

Bemerkung 2.31. *Der Satz 2.30 hat zur Folge, daß die Ableitung eines an den Lobatto-Polynom-Nullstellen interpolierten Polynoms an genau diesen Stellen x_k, $k = 1, \ldots, n$ direkt ausgewertet werden kann:*

$$p(x) = \sum_{i=1}^{n} y_i \cdot l_i(x) \quad \Longrightarrow \quad p'(x_k) = \sum_{i=1}^{n} y_i \cdot l_i'(x_k) = \sum_{i=1}^{n} y_i \cdot D_{k,i}.$$

Diese Eigenschaft wird ab Abschnitt 7.4 verwendet.

Zu den Lagrange-Polynomen l_i sind folgende Aussagen bekannt:

Satz 2.32. *Es seien x_i, $i = 1, \ldots, n$ paarweise verschieden. Dann bilden die Lagrange-Polynome aus Definition 2.9 eine Basis des linearen Raums aller Polynome vom Grade $\leq n-1$.*

Beweis. Folgt unmittelbar. □

Die Lagrange-Polynome werden zur Interpolation verwendet. Alle Polynome bis zum Grad $n-1$ lassen sich durch eine Linearkombination der l_i darstellen.

Folgerung 2.33. *Ein Polynom P des Grades kleiner n läßt sich durch die obigen Lagrange-Polynome darstellen:*

$$P(x) = \sum_{i=1}^{n} P(x_i) l_i(x).$$

Insbesondere gilt für $P(x) = x^{n-1}$ dann

$$x^{n-1} = \sum_{i=1}^{n} x_i^{n-1} l_i(x).$$

Kapitel 3

Gewöhnliche Differentialgleichungen

3.1 Einleitung

Dieses Kapitel besteht aus zwei Komplexen. Als erstes wird die allgemeine Theorie der Differentialgleichungen behandelt, als zweites Begriffe der numerischen Methoden zur Lösung von Differentialgleichungen. Beide Theorien dienen als Grundlage für die Runge-Kutta-Verfahren des Kapitels 4.

Im Abschnitt 3.2 wird die analytische Theorie der gewöhnlichen Differentialgleichungen wiederholt.
Der Abschnitt 3.3 präsentiert die Ideen und Grundbegriffe der Einschrittverfahren zur numerischen Bearbeitung von Anfangswertproblemen.
Der Abschnitt 3.4 legt die Konvergenzbegriffe der Einschrittverfahren dar.
Der Abschnitt 3.5 enthält Hilfsresultate und sonstige Hinweise.

Die Begriffe dieses Kapitels werden im wesentlichen in den Einführungsveranstaltungen der numerischen Mathematik, der Theorie der gewöhnlichen Differentialgleichungen und der Analysis gelehrt. Die grundlegende Quellen sind dementsprechend [59], [24], [3] und [27], also Lehrbücher und Skripte dieser Themenbereiche. Die jeweiligen Thematiken kann man dort nachlesen und vertiefen.

3.2 Allgemeine Theorie

In diesem Abschnitt werden grundlegende Sätze, Eigenschaften und Bezeichnungen aus der Theorie der gewöhnlichen Differentialgleichungen vorgestellt. Sie sind als Fundament für die nachfolgenden numerischen Untersuchungen unabdingbar. Zuerst wird eine gewöhnliche Differentialgleichung formal definiert.

Definition 3.1 (Gewöhnliche Differentialgleichung). *Es sei $n \in \mathbb{N}$ und $[t_0, t_f]$ ein Intervall. Eine* `gewöhnliche Differentialgleichung (DGL)` *im $\mathbb{R}^n$ ist durch die Gleichung*

$$\frac{d}{dt}x(t) = f(t, x(t)), \quad t \in [t_0, t_f] \tag{3.1}$$

gegeben, wobei $f : D \to \mathbb{R}^n$ eine stetige Funktion ist und der Definitionsbereich D eine offene Teilmenge des $\mathbb{R} \times \mathbb{R}^n$. Eine Lösung ist eine stetig differenzierbare Funktion $x : [t_0, t_f] \to \mathbb{R}^n$, die die Gleichung (3.1) erfüllt und bei der der Graph von x in D liegt.

Für ein Optimalsteuerungsproblem reicht der obige Lösungsbegriff nicht aus, wie später in (3.3) deutlich wird.
In der Praxis haben viele anwendungsbezogene Differentialgleichungen die Gestalt eines Anfangswertproblems:

Definition 3.2 (Anfangswertproblem (AWP)). *Gegeben sei die Differentialgleichung aus Definition 3.1. Gegeben sei ferner der* `Anfangswert` *$x_0 \in \mathbb{R}^n$. (t_0, x_0) liege in D. Ein* `Anfangswertproblem (AWP)` *ist durch*

$$\begin{aligned} \frac{d}{dt}x(t) &= f(t, x(t)), \\ x(t_0) &= x_0 \end{aligned}$$

gegeben. (t_0, x_0) heißt auch `Anfangsbedingung`*.*

Während die Differentialgleichung (3.1) im allgemeinen unendlich viele Lösungen besitzt, wird später in den Existenz- und Eindeutigkeitssätzen 3.8 und 3.9 bewiesen, daß ein Anfangswertproblem mithilfe geeigneter Voraussetzungen eine eindeutig bestimmte Lösung besitzt.
Die Anfangsbedingung legt die gewöhnliche Differentialgleichung an einem Punkt (t_0, x_0) fest. Die Lösungsfunktion x wird auch `(Lösungs-)Trajektorie` genannt. Die nächste Notationsvereinbarung soll Mißverständnisse verhindern.

Notation 3.3.

$$\frac{d}{dt}x(t) = \dot{x}(t) = x^{(1)}(t) = x'(t).$$

Ein Schlüssel zur numerischen Bearbeitung von gewöhnlichen Differentialgleichungen ist die einfache Umformung in eine Integralgleichung.

Satz 3.4 (Integraldarstellung). *Gegeben sei eine stetige Funktion $f : D \to \mathbb{R}^n$, eine Anfangsbedingung $(t_0, x_0) \in D$ und ein Intervall $[t_0, t_f]$. Die stetig differenzierbare Funktion $x : [t_0, t_f] \to \mathbb{R}^n$ löst das Anfangswertproblem*

$$\frac{d}{dt}x(t) = f(t, x(t)), \quad x(t_0) = x_0$$

genau dann, wenn sie für alle $t \in [t_0, t_f]$

$$x(t) = x_0 + \int_{t_0}^{t} f(s, x(s))\, ds \tag{3.2}$$

löst.

Beweis. Siehe [3, Satz 1.4.4] oder [26, Bemerkung 2.3]. □

Die Übertragung von Methoden der numerischen Integration auf Anfangswertprobleme ist der wichtigste numerische Lösungsansatz.
Bedeutsam für die Optimalsteuerungsprobleme ab Kapitel 6 ist die in [35, Kapitel IX, §1, Definition 1] definierte Funktionenklasse der absolut stetigen Funktionen. Motivation für diesen Begriff ist eine allgemeinere Differentialgleichung

$$\dot{x}(t) = f(t, x(t)) \quad \text{für fast alle } t \in [t_0, t_f]. \tag{3.3}$$

Dann muß die Lösung x der Integraldarstellung

$$x(t) = x_0 + \int_{t_0}^{t} f(s, x(s))\, ds, \quad t \in [t_0, t_f]$$

nur noch absolut stetig sein. Dies wird in [35, Kapitel IX, §4] gezeigt. Die absolut stetigen Funktionen werden ab jetzt mit AC bezeichnet.
Von den vielen Bedingungen, die man an eine Differentialgleichung stellen kann, ist die Lipschitz-Bedingung wohl die wichtigste. Sie wird besonders für die Existenz- und Eindeutigkeitsätze von Anfangswertproblemen benötigt.

Definition 3.5 (Lipschitz-Bedingung). *Die Abbildung* $f : D \longrightarrow \mathbb{R}^n$ *genügt in* D *einer* `Lipschitz-Bedingung`*, wenn eine Konstante* $L \geq 0$ *existiert, so daß für alle Punkte* $(t, x_1), (t, x_2) \in D$ *gilt:*

$$\|f(t, x_1) - f(t, x_2)\| \leq L \cdot \|x_1 - x_2\|. \tag{3.4}$$

f *ist also* `Lipschitz-stetig` *bezüglich des zweiten Arguments.*

Je nach verwendeter Norm variiert die Konstante L, nicht aber die Lipschitz-Bedingung an sich, da alle Normen im Endlichdimensionalen äquivalent sind.
Eine Abschwächung dieser globalen Lipschitzbedingung ist eine lokale Lipschitz-Bedingung:

Definition 3.6 (Lokale Lipschitz-Bedingung). *Existiert zu jedem Punkt* $(t, x) \in D$ *eine Umgebung* $U_{(t,x)}$*, so daß* f *in* $U_{(t,x)} \cap D$ *einer Lipschitz-Bedingung mit einer von* t *und* x *abhängigen Konstante* $L(t, x)$ *genügt, spricht man von einer* `lokalen Lipschitz-Bedingung`*.*

Die Ungleichung (3.4) legt nahe, daß die Differenzierbarkeit von f die Gültigkeit der Lipschitz-Bedingung beeinflußt.

Satz 3.7. *Es sei wie bisher D eine offene Teilmenge des $\mathbb{R} \times \mathbb{R}^n$. Die Funktion $f : D \longrightarrow \mathbb{R}^n$ sei bezüglich $x = (x_1, \ldots, x_n)$ in D stetig partiell differenzierbar. Dann genügt f in D lokal einer Lipschitz-Bedingung.*

Beweis. Der Beweis kann mit dem Mittelwertsatz geführt werden, siehe [24, §10, Satz 1] oder [59, §10, Folgerung zu Hilfssatz V]. □

Die Lipschitz-Bedingung (3.4) ist oft wenigstens lokal erfüllt, da f meistens gewisse Differenzierbarkeitsvoraussetzungen besitzt.
Nachdem die Aufgabenstellung von Anfangswertproblemen nun ausreichend beschrieben wurde und die wichtigsten technischen Hilfsmittel bereit stehen, gelangt man nun zum Existenz- und Eindeutigkeitssatz für Anfangswertprobleme. Je nach Voraussetzungen kann man diesen schwächer oder stärker formulieren.

Satz 3.8 (Existenz- und Eindeutigkeitssatz I). *Es sei das Anfangswertproblem aus Definition 3.2 gegeben. Es sei f stetig in D und genüge lokal einer Lipschitz-Bedingung in D. Dann exstiert ein $\epsilon > 0$ und genau eine Lösung x des Anfangswertproblems 3.2, die auf dem offenen Intervall $(t_0 - \epsilon, t_0 + \epsilon)$ definiert ist.*

Beweis. Der Satz wird in Einführungsveranstaltungen der Analysis, der Differentialgleichungen und der numerischen Mathematik bewiesen. Man kann ihn in [59, §10, Satz VI] oder [24, §10, Satz 2+3] nachlesen. □

Satz 3.9 (Existenz- und Eindeutigkeitssatz II). *Es sei das Anfangswertproblem aus Definition 3.2 gegeben. Es sei f stetig in D und genüge einer Lipschitz-Bedingung (3.4) in D. Dann exstiert genau eine Lösung x des Anfangswertproblems 3.2 im Intervall $[t_0, t_f]$.*

Beweis. Siehe [59, §10, Satz VII]. □

Bis jetzt wurden ausschließlich gewöhnliche Differentialgleichungen 1. Ordnung, also der Form

$$\dot{x}(t) = f(t, x(t)),$$

betrachtet. Häufig treten auch höhere Ableitungsordnungen auf, also Differentialgleichungen der Gestalt

$$x^{(m)}(t) \quad = \quad f\big(t, x(t), x^{(1)}(t), x^{(2)}(t), \ldots, x^{(m-1)}(t)\big). \tag{3.5}$$

Man kann zeigen, daß sich Differentialgleichungen beliebig hoher Ordnung zu Systemen 1. Ordnung umformen lassen. Im Gegenzug ergibt sich eine Erhöhung

der Dimension. Auch die Anfangsbedingungen lassen sich geeignet anpassen. Die `Universalität von Systemen` 1. `Ordnung` kann man in [3, Satz 1.4.1], [24, S.99ff] oder [56, 7.0] nachlesen. Damit folgert man die nächste Bemerkung.

Bemerkung 3.10. *Aus der Universalität von Systemen 1. Ordnung resultiert, daß sämtliche theoretischen Erkenntnisse nur für Differentialgleichungen 1. Ordnung hergeleitet werden müssen.*

Allgemein hängt die Funktion f der Differentialgleichung von der Zeit t und dem Ort x ab. Der folgende Satz zeigt eine Umwandlung eines beliebigen Anfangswertproblems in ein zeitunabhängiges Anfangswertproblem.

Satz 3.11 (Autonomisierung). *Man kann ein Anfangswertproblem*

$$\dot{x}(t) = f\big(t, x(t)\big), \quad x(t_0) = x_0$$

mittels

$$y(t) := \begin{pmatrix} x(t) \\ s(t) \end{pmatrix}, \quad \widetilde{f}(y(t)) := \begin{pmatrix} f(s(t), x(t)) \\ 1 \end{pmatrix}$$

zu einem `autonomen` *System*

$$\dot{y}(t) = \widetilde{f}\big(y(t)\big), \quad y(t_0) = \begin{pmatrix} x_0 \\ t_0 \end{pmatrix}$$

umformen. Es gilt $s(t) = t$.

Bemerkung 3.12. *Die Autonomisierung der Differentialgleichung erhöht die Dimension der Differentialgleichung um 1. Die Lipschitz-Bedingung (3.4) muß für den Existenz- und Eindeutikeitssatz 3.9 nun bezüglich x und zusätzlich t gelten.*

Aufgrund der Voraussetzungen an f ist diese Beobachtung jedoch keine echte Einschränkung.

3.3 Einschrittverfahren

In Abschnitt 3.3 werden grundsätzliche Überlegungen und Begriffe zum numerischen Lösen von Anfangswertproblemen dargelegt. Hierbei werden nur die sogenannten Einschrittverfahren behandelt. Die Gültigkeit der Existenz- und Eindeutigkeitsaussagen für Anfangswertprobleme aus dem letzten Abschnitt wird vorausgesetzt. In diesem Abschnitt wurde vor allem [27] und [42] verwendet.
Die meisten Anfangswertprobleme sind tatsächlich nicht analytisch lösbar. Auch sonst ist nur in Glücksfällen eine Lösung erkennbar oder mittels Standardmethoden berechenbar. Wird also die Lösung des Anfangswertproblems

$$\dot{x}(t) = f(t, x(t)), \quad x(t_0) = x_0$$

mit den Eigenschaften der Definition 3.2 gesucht, so sucht man in der Regel eine Punktmenge, welche zu bestimmten Zeitpunkten t_i innerhalb eines Intervalls $[t_0, t_f]$ die eigentliche Lösungstrajektorie x möglichst gut approximiert. Eine wichtige Entscheidung für eine numerische Lösungsmethode besteht in der Vorgabe des diskreten Gitters mit bestimmten Zeitpunkten t_i. Hierbei ist nun wesentlich, zwischen der eigentlichen Lösung und der numerischen Lösung zu unterscheiden, da diese sich normalerweise unterscheiden. Im weiteren soll

$$x(t_i) = x_i \tag{3.6}$$

die analytische Lösung zum Zeitpunkt t_i bezeichnen. Die numerischen Werte zum Zeitpunkt t_i werden ab jetzt mit

$$\eta_i \text{ oder } \eta_i^N \tag{3.7}$$

bezeichnet, wobei N auf ein Gitter verweist. Die erste Definition des Abschnitts führt den Begriff des Gitters ein.

Definition 3.13 (Gitter). *Es seien $t_0, t_1, \ldots, t_N = t_f \in \mathbb{R}$ mit $t_0 < t_1 < \ldots < t_{N-1} < t_N = t_f$. Dann bezeichnet*

$$\mathbb{G}_N := \{t_0, \ldots, t_N |\ t_0 < t_1 < \ldots < t_N = t_f\} \tag{3.8}$$

ein **Gitter** *auf dem Intervall $[t_0, t_f]$. Es sei die* **Schrittweite** *h_i definiert durch*

$$h_{i+1} := t_{i+1} - t_i, \quad i = 0, \ldots, N-1. \tag{3.9}$$

Sind die Zeiten $t_i \in \mathbb{G}_N$ äquidistant auf dem Gitter verteilt, so gilt

$$h := h_i = t_{i+1} - t_i, \quad i = 0, \ldots, N-1 \quad \text{mit} \tag{3.10}$$

$$t_i = t_0 + ih = t_0 + i\frac{t_f - t_0}{N}, \quad i = 0, \ldots, N. \tag{3.11}$$

Die numerische Lösung eines Anfangswertproblems liefert nun auf den Gitterpunkten t_i, $i = 0, \ldots, N$ Werte.

Definition 3.14 (Gitterfunktion). *Es gelte $(t_i, \eta_i) \in D$, $i = 0, \ldots, N$. Eine Funktion $\eta^N : \mathbb{G}_N \to \mathbb{R}^n$ heißt* **Gitterfunktion**. *Die Gitterfunktion läßt sich als Vektor darstellen:*

$$\eta^N = (\eta_0^N, \eta_1^N, \ldots, \eta_N^N) \text{ mit } \eta^N(t_i) = \eta_i^N \in \mathbb{R}^n,$$

wobei n die Dimension der Lösung der Differentialgleichung ist.

Die numerische Lösung eines Anfangswertproblems ist eine Gitterfunktion. Die Gitterfunktion η^N sollte die Lösungstrajektorie x an den Gitterpunkten t_i möglichst gut approximieren.
Als nächstes wird das Konzept der Einschrittverfahren formalisiert.

Definition 3.15 (Einschrittverfahren). *Gegeben sei eine Funktionenklasse $\mathcal{F}$ und eine Abbildung Φ, auch* `Verfahrensfunktion` *genannt:*

$$\begin{aligned} \Phi : D \times \mathbb{R} \times \mathcal{F} &\longrightarrow \mathbb{R}^n, \\ (t, x, h, f) &\longmapsto \Phi(t, x, h, f). \end{aligned}$$

Weiter sei das Gitter aus der Definition 3.13 gegeben. Dann ist zu dem Anfangswertproblem der Definition 3.2 ein `Einschrittverfahren` *definiert durch*

$$\eta_0^N = x_0, \tag{3.12}$$
$$\eta_{i+1}^N = \eta_i^N + h_i \Phi(t_i, \eta_i^N, h_i, f), \quad i = 0, \ldots, N-1. \tag{3.13}$$

Wenn möglich werden die letzten beiden Argumente in der Verfahrensfunktion gerne weggelassen.
Namensgebend für die Bezeichnung Einschrittverfahren ist die Abhängigkeit der Verfahrensfunktion Φ von η_i^N für die Berechnung des Wertes η_{i+1}^N, nicht aber von Elementen aus

$$\left\{\eta_k^N \,|\, k = 0, \ldots, i-1\right\}, \tag{3.14}$$

im Unterschied zu Mehrschrittverfahren.
Es wird zwischen expliziten und impliziten Einschrittverfahren unterschieden. Diese Trennung beider Klassen ist etwas subtil. Die Formeln (3.12) und (3.13) garantieren noch nicht, daß man alle η_{i+1}^N sukzessive von η_0^N aus berechnen kann. Ist dies möglich, so nennt man ein Einschrittverfahren `explizit`. Hängt dagegen die Verfahrensfunktion Φ in (3.13) indirekt von der linken Seite η_{i+1}^N ab, so muß man stets erst ein Gleichungssystem lösen. Ein derartiges Einschrittverfahren wird `implizit` genannt, siehe auch [47, Definitionen 11.2 und 11.3]. Der Unterschied wird besonders in der Definition 4.1 deutlich.
Das bekannteste, einfachste und dennoch illustrative numerische Verfahren ist die Eulersche-Polygonzug-Methode, auch einfach Euler-Verfahren genannt. Anhand dieses Verfahrens werden die bisherigen Ideen veranschaulicht. Gleichzeitig werden weitere Ideen motiviert.

Definition 3.16 (Euler-Verfahren). *Gegeben sei ein Gitter $\mathbb{G}_N$ der Definition 3.13 und ein Anfangswertproblem der Definition 3.2. Die rechte Seite der Differentialgleichung liefert für die Anfangsbedingung (t_0, x_0) die Steigung der Lösung x bei t_0. Die Steigung am jeden Gitterpunkt wird als affine Approximation genutzt:*

$$\begin{aligned} \eta_0^N &= x_0, \\ \eta_{i+1}^N &= \eta_i^N + h_i f(t_i, \eta_i^N), \quad i = 0, \ldots, N-1. \end{aligned}$$

Das folgende Beispiel demonstriert das Euler-Verfahren auch graphisch.

Beispiel 3.17. *Für das Anfangswertproblem*

$$\begin{aligned} \frac{d}{dt}x(t) &= \frac{x(t)}{t^3}, \\ x(1) &= 1 \end{aligned}$$

sei die Lösung im Intervall $[1,2]$ *gesucht. Die analytische Lösung ist*

$$x(t) = \frac{e^{-\frac{1}{2t^2}}}{e^{-\frac{1}{2}}}.$$

Die Abbildung 3.1 veranschaulicht den tatsächlichen Verlauf der Lösung und die Wirkung des Eulerverfahrens, der als Polygonzug illustriert ist, bei der Schrittweite $h = 0.25$.

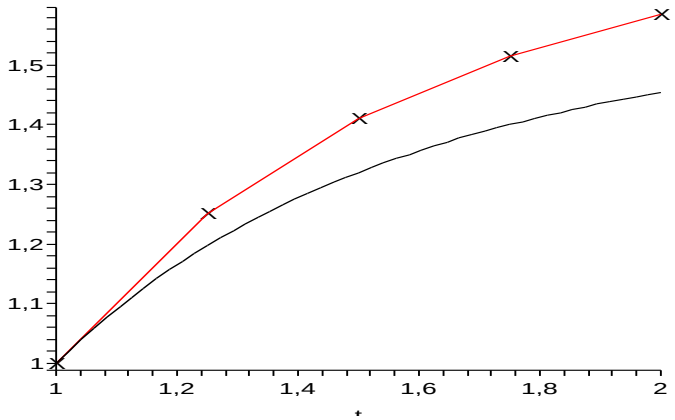

Abbildung 3.1: Euler-Polygonzugverfahren zu Beispiel 3.17

Eine andere Darstellung des Euler-Verfahrenes führt über die Integraldarstellung des Satzes 3.4:

$$\eta_i^N + \int_{t_i}^{t_{i+1}} f(s, x(s))\, ds \approx \eta_i^N + \overbrace{(t_{i+1} - t_i)}^{=h_i} f(t_i, \eta_i^N). \tag{3.15}$$

Das Integral und wird mit einer einfachen Riemann-Summe approximiert. In dieser Interpretation ist eine Riemann-Summe äquivalent zum Euler-Verfahren. Eine Rimann-Summe - und damit das Euler-Verfahren - besitzt nur eine niedrige Konsistenzordnung. Es besteht die begründete Hoffnung, daß man auch zu anderen, effizienteren Integrationsmethoden entsprechende numerische Verfahren findet, vor allem zur Gauß-Quadratur.

Die Abbildung 3.2 veranschaulicht die Riemann-Summe und die Mittelpunktsregel. Während eine Riemann-Summe nur den Exaktheitsgrad 0 besitzt, weist die Mittelpunktsregel immerhin trotz gleicher Knotenanzahl den Exaktheitsgrad 1 auf. Bei einem Einschrittverfahren liegt jedoch an der Stelle $t_0 + \frac{1}{2}h$ kein bekannter Wert vor. Ein auf der Mittelpunktsregel basierendes Einschrittverfahren wäre somit implizit. Hier offenbart sich bereits ein Qualitätsunterschied zwischen expliziten und impliziten Verfahren. Im Beispiel 4.2 werden die beiden angesprochenen Regeln nochmalig aufgegriffen.

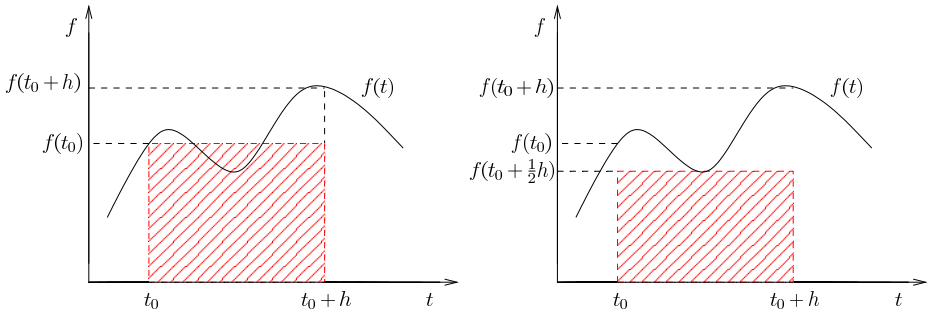

Abbildung 3.2: Links Riemann-Summe, rechts Mittelpunktsregel

3.4 Konvergenzanalyse

In diesem Abschnitt werden die bekannten Konvergenzbegriffe für Anfangswertprobleme erläutert. Die verwendeten Begriffe sind für die gesamte Theorie der numerischen Verfahren zur Lösung von Differentialgleichungen gebräuchlich. Mit ihnen bewertet und vergleicht man unterschiedliche Verfahren. Quellen des Abschnitts sind unter anderem [27] und [42].
Entscheidend für die Effektivität eines numerischen Verfahrens ist die Annäherung der Gitterfunktion an die echte Lösung. Dabei kommt der Schrittweite, also die Feinheit des Gitters, eine zentrale Bedeutung zu. Wie im vorherigen Abschnitt bezeichnet auch hier $\eta_i^N = \eta_i$ den numerischen Wert der Näherung für $x(t_i) = x_i$.

Definition 3.18 (Globaler Diskretisierungsfehler). *Das Gitter sei wie in Definition 3.13 gegeben. Ein Einschrittverfahren liefere die Gitterfunktion η^N. Die exakte Lösung sei x. Dann heißt*

$$e^N(t_i) := \eta_i^N - x_i, \quad i = 0, \ldots, N$$

`globaler Diskretisierungsfehler`. Der ganze Vektor wird mit e^N abgekürzt.

Man beachte, daß im Gitter die Schrittweiten h_i enthalten sind. Ab jetzt soll gelten:

$$h := \sup_{i=0,\ldots,N-1} h_i. \tag{3.16}$$

Die Konvergenzordnung eines Einschrittverfahrens wird über den globalen Diskretisierungsfehler definiert. Eine wesentliche Bedeutung kommt der Maximumsnorm zu:

$$\left\|e^N\right\|_\infty := \max_{i=0,\ldots,N} \left\|e^N(t_i)\right\|_2, \tag{3.17}$$

wobei $\|\cdot\|_2$ die euklische Norm bezeichnet.

Definition 3.19 (Konvergenz eines Einschrittverfahrens). *Ein Einschrittverfahren zu dem Anfangswertproblem der Definition 3.2 mit dem Gitter der Definition 3.13 heißt* **konvergent**, *falls bezüglich der Funktionenklasse $\mathcal{F}$*

$$\lim_{h\to 0} \left\|e^N\right\|_\infty = 0$$

ist. Es besitzt die **Konvergenzordnung** *p bezüglich der Funktionenklasse $\mathcal{F}$, wenn*

$$\left\|e^N\right\|_\infty = \mathcal{O}(h^p)$$

ist für alle hinreichend kleinen Gitter $\mathbb{G}_N$.

Bei der Konvergenzanalyse unterteilt man die numerischen Abweichungen in zwei unterschiedliche Kategorien. Dies kann durch folgende Rechnung begründet werden, bei der der Hilfsterm

$$\big(x_i + h_i\Phi(t_i, x_i, h_i, f)) - (x_i + h_i\Phi(t_i, x_i, h_i, f)\big) \tag{3.18}$$

eingefügt und dann die Dreiecksungleichung angewendet wird:

$$\begin{aligned}
& \left\|\eta_{i+1}^N - x_{i+1}\right\| \\
= \; & \left\|\eta_i^N + h_i\Phi(t_i, \eta_i^N, h_i, f) - x_{i+1}\right\| \\
\leq \; & \left\|\big(\eta_i^N + h_i\Phi(t_i, \eta_i^N, h_i, f)\big) - \big(x_i + h_i\Phi(t_i, x_i, h_i, f)\big)\right\| && (3.19)\\
+ \; & \left\|\big(x_i + h_i\Phi(t_i, x_i, h_i, f)\big) - x_{i+1}\right\|. && (3.20)
\end{aligned}$$

Man kann die zwei Fehlerarten erkennen und interpretieren:

- Der Term (3.19) beinhaltet alle Fehler, die bis zur Zeit t_i gemacht wurden. Hier findet man den globalen Diskretisierungsfehler aus der Definition 3.18 wieder.

- Im Term (3.20) erscheint der Fehler, der im aktuellen Schritt von t_i nach t_{i+1} gemacht wird. Über (3.20) definiert man den lokalen Diskretisierungsfehler.

Als erstes wird der lokale Diskretisierungsfehler untersucht.

Definition 3.20 (Lokaler Diskretisierungsfehler). *Es sei das Anfangswertproblem der Definition 3.2 und das Gitter aus Definition 3.13 gegeben. Dann definiert man für den Punkt $(t_i, x(t_i))$ den* `lokalen Diskretisierungsfehler`

$$\tau(t_i, x(t_i), h_i, f) := \frac{x(t_i + h_i) - x(t_i)}{h_i} - \Phi(t_i, x(t_i), h_i, f).$$

τ gibt die durch die Schrittweite dividierte Abweichung der numerischen Lösung von der exakten Lösung bei Ausführung eines Schrittes des Einschrittverfahrens ausgehend von der Anfangsbedingung (t_i, x_i) an. Der lokale Diskretisierungsfehler muß für kleine Intervalle gegen die Null gehen, denn ansonsten verkleinert sich zwar in jedem Rechenschritt die Abweichung, aber nicht relativ zur Schrittweite.

Definition 3.21 (Konsistenz). *Ein Einschrittverfahren der Definition 3.15 heißt* `konsistent`*, falls*

$$\lim_{h \to 0} \tau(t, x, h, f) = 0 \tag{3.21}$$

gleichmäßig für alle $(t, x) \in D$ gilt. Es besitzt die Konsistenzordnung p, wenn

$$\tau(t, x, h, f) = \mathcal{O}(h^p) \tag{3.22}$$

ist für $(t, x) \in D$ und für alle hinreichend kleinen Gitter $\mathbb{G}_N$.

Der folgende Satz legt einen einfachen Sachverhalt dar:

Satz 3.22 (Konsistenzbedingung). *Gegeben sei ein Einschrittverfahren aus Definition 3.15. Ein solches Verfahren ist genau dann konsistent, wenn für alle $(t, x) \in D$ die folgende Bedingung gilt:*

$$\Phi(t, x, 0, f) = f(t, x).$$

Beweis. Siehe [27, Lemma 2.10]. □

Dieses Resultat überrascht nicht, denn f läßt sich als Vektorfeld interpretieren. Die Lösung eines Anfangswertproblems führt durch das Vektorfeld und Φ muß die Richtungen approximieren (vgl. Euler-Verfahren in Definition 3.16).
Nun muß noch der globale Diskretisierungsfehler kontrolliert werden. Dazu muß die `Stabilität` eines Einschrittverfahrens untersucht werden, worauf aber weitgehend verzichtet werden soll. Die folgende Bedingung aus [27, Definition 2.9] reicht aus, um die Stabilität eines Einschrittverfahrens zu gewährleisten.

Definition 3.23 (Stabilitätsbedingung). *Ein Einschrittverfahren der Definition 3.15 erfüllt die* `Stabilitäts-` *oder* `Lipschitzbedingung`*, falls für jede kompakte Menge $K \subset D$ des Definitionsbereichs der Differentialgleichung ein $L > 0$ existiert, so daß für alle Paare $(t, y_1), (t, y_2) \in K$ und alle hinreichend kleinen $h > 0$ die folgende Abschätzung gilt:*

$$\|y_1 + h\Phi(t, y_1, h, f) - y_2 - h\Phi(t, y_2, h, f)\| \leq (1 + Lh) \cdot \|y_1 - y_2\| . \qquad (3.23)$$

(3.23) ist eine Lipschitz-Bedingung an die Verfahrensfunktion Φ eines Einschrittverfahrens.
Jetzt vereinen sich die Aussagen über lokale und globale Diskretisierungsfehler zu dem entscheidenen Satz des Abschnitts:

Satz 3.24. *Es sei ein Einschrittverfahren der Definition 3.15 gegeben. Das Einschrittverfahren erfülle die Stabilitätsbedingung der Definition 3.23 und sei ferner konsistent, siehe Definition 3.21. Dann ist das Einschrittverfahren mit der Verfahrensfunktion Φ konvergent im Sinne von Definition 3.19. Besitzt das Verfahren die Konsistenzordnung p, so ist die Konvergenzordnung ebenfalls p.*

Beweis. Siehe [42, Korollar 6.2.5], [27, Satz 2.11] oder [61, Satz 4.2.5]. □

Bei Gültigkeit der Stabilitätsbedingung wird die Konvergenzordnung eines Einschrittverfahrens also durch seine Konsistenzordnung bestimmt.
Bis jetzt wurden noch keine Aussagen zu den Eigenschaften der rechten Seiten f der Differentialgleichungen gemacht. Es wurde lediglich von einer Funktionenklasse $\mathcal{F}$ gesprochen. Aus der Definition 3.20 wird ersichlich, wie man die Konsistenzordnung berechnet. Man muß die Funktion x und die Verfahrensfunktion Φ im lokalen Diskretisierungsfehler

$$\tau(t, x, h, f) = \frac{x(t+h) - x(t)}{h} - \Phi(t, x, h, f)$$

mit der Taylor-Regel entwickeln. Hierbei enthält die Funktion Φ auch die rechte Seite f, wobei vereinfachend $\dot{x}(t) = f(t, x(t))$ gilt. Beispielhaft kann man zeigen:

Satz 3.25. *Es sei das Euler-Verfahren der Definition 3.16 gegeben. Die Funktionenklasse $\mathcal{F}$ bestehe aus allen Funktionen f, die einschließlich aller partiellen Ableitungen bis zur 1. Ordnung stetig und beschränkt in D sind. Dann besitzt das Eulerverfahren die Konsistenzordnung 1.*

Beweis.

$$\begin{aligned} \tau(t, x, h, f) &= \frac{x(t+h) - x(t)}{h} - \Phi(t, x, h, f) \\ &= \frac{x(t) + \dot{x}(t)h + \mathcal{O}(h^2) - x(t)}{h} - f(t, x(t)) = \mathcal{O}(h). \end{aligned}$$

□

In [42, Beispiel 6.1.16] wird gezeigt, daß das Euler-Verfahren der Definition 3.16 stabil ist, wenn die Lipschitz-Bedingung (3.4) gilt. Damit besitzt das Euler-Verfahren die Konvergenzordnung 1.
Man kann aber allgemeiner beobachten:

Bemerkung 3.26. *Für die Konsistenzordnung p eines Einschrittverfahrens wird benötigt, daß die partiellen Ableitungen aller Funktionen f der rechten Seite einer Differentialgleichung bis zur p-ten Ordnung stetig und beschränkt in D sind.*

Die folgende Bemerkung versucht eine erste quantitative Verknüpfung zwischen den Integrationsmethoden und den in diesem Kapitel erläuterten numerischen Verfahren herzustellen. Sie beseitigt außerdem eine leicht auftretende Irritation.

Bemerkung 3.27. *Es besteht ein enger Zusammenhang zwischen der Konsistenzordnung (Definition 3.21), dem Exaktsheitgrad eines Integrationsverfahrens (Notation 2.8) und deren Fehlertermordnung (Satz 2.20). Dem Exaktheitsgrad m einer Quadraturmethode entspricht die Fehlertermordnung $m+2$. Bei Konsistenzordnungen wird noch durch die Schrittweite h dividiert (Definition 3.20). Überträgt sich die Ordnung der Integrationsmethode auch auf die rechte Seite des Integrals, so ergibt sich die Konsistenzordnung $m+1$:*

Exaktheitsgrad $m \longleftrightarrow$ Fehlertermordnung $m+2 \longleftrightarrow$ Konsistenzordnung $m+1$.

Besitzt eine Integrationsmethode n Knoten, dann kann ein auf dieser Methode beruhendes Einschrittverfahren die Konsistenzordnung $2n$ nicht überschreiten.

In den Kapiteln 4 und 5 werden diese Zusammenhänge herausgearbeitet.

3.5 Weitere Eigenschaften

In diesem Abschnitt werden ein paar Literaturhinweise zu ausgewählten Fragestellungen gegeben. Außerdem wird vorbereitend ein Hilfssatz für den Abschnitt 5.3 zur Verfügung gestellt.
In diesem Abschnitt soll hauptsächlich kurz auf einige Felder verwiesen werden, die emminent wichtig sind, aber hier nicht behandelt werden können. Tatsächlich fehlen noch einige gewichtige Argumente, warum im nächsten Kapitel besonders mit impliziten Einschrittverfahren weitergearbeitet wird.
Mit die wichtigste Motivation für die später vor allem im Abschnitt 4.5 behandelten impliziten Runge-Kutta-Verfahren ist das Auftreten sogenannter `steifer` Differentialgleichungen. Beispiele für steife Differentialgleichungen enthält zum Beispiel [29, Abschnitt IV.1]. Erhellend ist auch die Darstellung in [32, Abschnitt 4.2]. In [27, Abschnitt 2.6] findet sich eine Einleitung. Diese Darstellungen enthalten in der Regel Beispiele, die zusätzlich untermauern, warum implizite Verfahren

Vorzüge gegenüber expliziten aufweisen. Steife Differentialgleichungen werden in [32, Kapitel 4] oder [28, Abschnitt I.13] erläutert. Auf die Kondition von Anfangswertproblemen geht [27, Abschnitt 2.2.4] ein, vielfältige Diskussionen finden sich in [18, Kapitel 3]. Auf Steifheitsuntersuchungen von Runge-Kutta-Verfahren wird im Abschnitt 4.7 verwiesen.
Für den wichtigen Satz 5.8 wird ein Resultat aus der Sensitivitätstheorie für Anfangswertprobleme benötigt. Der Satz analysiert, wie sich eine Störung der rechten Seite f auf die Lösung x eines Anfangswertproblems auswirkt.

Satz 3.28 (Aleksejew (1961), Gröbner (1960)). *Die Abbildungen f und δf seien im Phasenraum $D \subset \mathbb{R} \times \mathbb{R}^n$ stetig und dort nach den Zustandsvariablen stetig differenzierbar. Das Anfangswertproblem*

$$x'(t) = f(t, x(t)), \quad x(t_0) = x_0$$

besitze für den Punkt (t_0, x_0) des Phasenraums die Lösung x. Es sei

$$y'(t) = f(t, y(t)) + \delta f(t, y(t)), \quad y(t_0) = x_0$$

das gestörte Anfangswertproblem, welches die Lösung $y = x + \delta x$ besitze.
Dann existiert für ein hinreichend nahe bei t_0 liegendes t_1 eine auf

$$U = \left\{(t, s) \in \mathbb{R}^2 : t \in [t_0, t_1], s \in [t_0, t]\right\}$$

stetige matrixwertige Abbildung $M : U \longrightarrow Mat_n(\mathbb{R})$, so daß die Störung δx für alle $t \in [t_0, t_1]$ die Darstellung

$$\delta x(t) = \int_{t_0}^{t} M(t, s)\delta f(s, y(s))\, ds$$

besitzt.

Beweis. Den Beweis findet man in [18, Satz 3.4]. □

Aus dem Beweis kann man ersehen, daß die Abbildung M beliebig glatt wird, wenn die Funktion f beliebig glatt ist. Dies wird auch in Satz 5.8 benutzt.

Kapitel 4

Runge-Kutta-Verfahren

4.1 Einleitung

Ein essentielles Fundament dieses Kapitels 4 ist das vorangegangene Kapitel 3. Es werden die Existenz- und Eindeutigkeitssätze für Anfangswertprobleme aus Abschnitt 3.2 genauso als gegeben vorausgesetzt wie die Eigenschaften der Differentialgleichungen und Einschrittverfahren, die nach Abschnitt 3.4 die Konsistenzordnungen gestatten. Die in diesem Kapitel geschilderten Runge-Kutta-Verfahren lassen sich in die Klasse der Einschrittverfahren des Abschnitts 3.3 einordnen. Die Runge-Kutta-Verfahren erweisen sich nicht nur aus numerischen Gründen als besonders effektiv, sie lassen sich auch sehr schön formalisieren.
Die Runge-Kutta-Verfahren lassen sich in zwei große Klassen einteilen: explizite und implizite Runge-Kutta-Verfahren. Während explizite Runge-Kutta-Verfahren zunächst anschaulicher und einfacher als implizite Runge-Kutta-Verfahren sind, erweisen sich implizite Runge-Kutta-Verfahren letztendlich als effizienter. Sie sind es, die die im Kapitel 2 erarbeitete Gauß-Quadratur ausnutzen und deren Struktur übernehmen.
Entsprechend zielt der Verlauf des Kapitels auf die impliziten Runge-Kutta-Verfahren ab, auch wenn aus technischen und didaktischen Gründen vorerst beide Klassen eingeführt werden. Das allgemeine Vorgehen läßt auch Vorteile, Nachteile und Vergleiche der Varianten transparenter werden. Quellen sind am Anfang [32], [27] und [56], dann später die tiefgehenderen Werke [12], [18], [28] und [29].

In Abschnitt 4.2 wird die Idee der Runge-Kutta-Verfahren erläutert.
In Abschnitt 4.3 erfolgen Definitionen, Existenz- und Eindeutigkeitssätze, Darstellungsformen und Beispiele.
Der kurze Abschnitt 4.4 soll mit einigen wenigen Resultaten die Möglichkeiten expliziter Runge-Kutta-Verfahren bilanzieren, um einen Vergleich mit den impliziten Verfahren des folgenden Abschnitts vorzubereiten.

Der Abschnitt 4.5 bildet mit der Herleitung von impliziten Runge-Kutta-Verfahren maximaler Konsistenzordnung den Schwerpunkt des Kapitels.
In Abschnitt 4.6 wird nach nützlichen Derivaten der optimalen Verfahren des Abschnitts 4.5 geforscht.
Der Abschnitt 4.7 listet einige Literaturhinweise zu Stabilitätseigenschaften von impliziten Runge-Kutta-Verfahren auf.

4.2 Motivation und Idee

Explizite und implizite Runge-Kutta-Verfahren lassen sich formal gemeinsam einführen. Meistens werden der Einfachheit halber anfangs jedoch hauptsächlich die expliziten Runge-Kutta-Verfahren betrachtet, selbst wenn das eigentliche Ziel die impliziten Verfahren sind. Im folgenden wird eine Ausgangsidee der Runge-Kutta-Verfahren geschildert.
Gegeben sei die übliche Aufgabenstellung eines Einschrittverfahrens, also die Berechnung eines Wertes η_{i+1}^N aus einer gegebenen Differentialgleichung und dem Wert η_i^N, wie es sich etwa aus den Definitionen 3.2 und 3.15 ergeben könnte. Zur Notationsvereinfachung wird o.b.d.A. als Ausgangspaar (t_0, x_0) verwendet, und berechnet wird demnach die Näherung der Lösungstrajektorie bei $t_0 + h$.

$$x(t_0 + h) = x_0 + \int_{t_0}^{t_0+h} f(s, x(s))\, ds. \tag{4.1}$$

Man führt nun innerhalb des Intervalls $[t_0, t_0 + h]$ Stützstellen ein mit

$$t_0 + c_i h, \quad i = 1, \ldots, s, \quad 0 \leq c_i \leq 1. \tag{4.2}$$

Nehme man einmal an, daß die Werte $x(t_0 + c_i h)$, $i = 1, \ldots, s$ der Lösung x an den Stellen $t_0 + c_i h$ bekannt seien. Dann kann man ein Lagrange-Polynom vom Grade $\leq s - 1$ konstruieren, welches durch die Punkte $\big(t_0 + c_i h, f(t_0 + c_i h, x(t_0 + c_i h))\big)$, $i = 1, \ldots, s$ verläuft. Dieses Polynom ersetzt näherungsweise f in (4.1). Durch numerische Integration ergibt sich

$$x(t_0 + h) \approx x_0 + h \sum_{i=1}^{s} b_i f\big(t_0 + c_i h, x(t_0 + c_i h)\big), \tag{4.3}$$

wobei b_i, $i = 1, \ldots, s$ gewisse reelle Gewichte angibt.
Die Werte $x(t_0 + c_i h)$ sind jedoch unbekannt. Sie müssen ebenso näherungsweise berechnet werden. Bei analoger Vorgehensweise folgt

$$x(t_0 + c_i h) = x_0 + \int_{t_0}^{t_0+c_i h} f(s, x(s))\, ds, \quad i = 1, \ldots, s. \tag{4.4}$$

Temporär soll der Wert von $x(t_0 + c_i h)$ einfach nur von den $\big(t_0 + c_j h, x(t_0 + c_j h)\big)$ mit $j < i$ abhängen. Es ergibt sich analog zu eben für $i = 1, \ldots, s$:

$$x(t_0 + c_i h) = \int_{t_0}^{t_0 + c_i h} f(s, x(s))\, ds \;\approx\; x_0 + h c_i \sum_{j=1}^{i-1} d_j f(t_0 + c_j h, x(t_0 + c_j h))$$
$$= x_0 + h \sum_{j=1}^{i-1} a_{ij} f(t_0 + c_j h, x(t_0 + c_j h)). \quad (4.5)$$

Die Einführung der a_{ij} durch Zusammenfassen der Vorfaktoren c_i und d_j erzeugt eine zu (4.3) konforme Schreibweise. Dies ist bereits die Vorschrift für ein explizites Runge-Kutta-Verfahren. Ein implizites Runge-Kutta-Verfahren erhält man nun, indem man in (4.5) die Summation bis zur Stützstelle s erstreckt:

$$x(t_0 + c_i h) \approx x_0 + h \sum_{j=1}^{s} a_{ij} f\big(t_0 + c_j h, x(t_0 + c_j h)\big), \quad i = 1, \ldots, s. \quad (4.6)$$

Der Zusammenhang zwischen Integration und Interpolation eröffnet bereits Überlegungen in Richtung Gauß-Quadratur.
Der folgende Abschnitt formalisiert die Vorgehensweise und führt die gängigen Bezeichnungen ein. Die Bestimmung der bislang unbekannten Daten c_i, b_i, a_{ij} nach zweckmäßigen Kriterien erfolgt später.

4.3 Eigenschaften von Runge-Kutta-Verfahren

Es werden nun wieder die numerischen Werte η_i^N verwendet, aber ohne die für die weiteren Zwecke unnötige Gitterfeinheit N. Wiederum zur Vereinfachung werden nur η_0 und η_1 verwendet. Ausgangspunkt des Anfangswertproblems ist damit der Punkt (t_0, η_0). Die Schrittweite sei h.

Definition 4.1 (Runge-Kutta-Verfahren I). *Ein* ***s-stufiges Runge-Kutta-Verfahren*** *zur Lösung eines Anfangswertproblems aus Definition 3.2 ist definiert durch*

$$k_i = f(t_0 + c_i h, \eta_0 + h \sum_{j=1}^{s} a_{ij} k_j), \quad i = 1, \ldots, s, \quad (4.7)$$

$$\eta_1 = \eta_0 + h\Phi(t_0, \eta_0, h, f) = \eta_0 + h \sum_{i=1}^{s} b_i k_i. \quad (4.8)$$

Die Zeiten $t_0 + c_i h$ heißen ***Stützstellen****, die Vektoren k_i* ***Steigungen****. Die Anzahl s der Stützstellen wird* ***Stufenzahl*** *genannt.*

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \dots & a_{1s} \\ c_2 & a_{21} & a_{22} & \dots & a_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & a_{s2} & \dots & a_{ss} \\ \hline & b_1 & b_2 & \dots & b_s \end{array} \qquad \begin{array}{c|c} c & A \\ \hline & b^T \end{array}$$

Abbildung 4.1: Schema eines Butcher-Tableaus

Für die Veranschaulichung eines Runge-Kutta-Verfahrens hat sich das sogenannte Butcher-Tableau der Abbildung 4.1 bewährt.
Zu verschiedenen Vorgaben an die Matrix A der Runge-Kutta-Verfahren haben sich eigene Bezeichnungen etabliert. Man bezeichnet ein Runge-Kutta-Verfahren als

- explizit (ERK), falls für alle i und j mit $i \leq j$ gilt, daß $a_{ij} = 0$ ist,
- implizit (IRK), falls ein $a_{ij} \neq 0$ für $i \leq j$ existiert.

Bei expliziten Runge-Kutta-Verfahren ist A also eine strikte untere Dreiecksmatrix. Eine verfeinerte Unterteilung der Runge-Kutta-Verfahren wird in [28, Definition 7.1] vorgenommen. Die erwähnten Einschrittverfahren in Abschnitt 3.3 sind auch Runge-Kutta-Verfahren:

Beispiel 4.2. *Die Abbildung 4.2 stellt das Euler-Verfahren, die implizite Mittelpunktsregel, die Trapezregel und das sogenannte klassische Runge-Kutta-Verfahren dar. Bei expliziten Verfahren werden die Einträge auf und oberhalb der Diagonalen oft leer gelassen.*

$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array} \qquad \begin{array}{c|c} \frac{1}{2} & \frac{1}{2} \\ \hline & 1 \end{array} \qquad \begin{array}{c|cc} 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \qquad \begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ \hline & \frac{1}{6} & \frac{2}{6} & \frac{2}{6} & \frac{1}{6} \end{array}$$

Abbildung 4.2: Bekannte Runge-Kutta-Verfahren

Auch wenn noch keine Kriterien für die Tableau-Einträge A, b und c eines Runge-Kutta-Verfahren aufgestellt wurden, kann zu den Verfahren des Beispiels 4.2 mit

der üblichen Taylor-Entwicklung die Konsistenzordnung berechnet werden. Wie man in [32, Abschnitt 2.2] nachvollziehen kann, wird das bereits beim klassischen Runge-Kutta-Verfahren ziemlich aufwendig. Die Theorie der impliziten Runge-Kutta-Verfahren wird zur erheblichen Erleichterung beitragen. Es existiert eine weitere Definition von Runge-Kutta-Verfahren, die oft verwendet wird.

Definition 4.3 (Runge-Kutta-Verfahren II). *Ein `s-stufiges Runge-Kutta-Verfahren` zur Lösung eines Anfangswertproblems der Definition 3.2 ist definiert durch*

$$\begin{aligned} Y_i &= \eta_0 + h\sum_{j=1}^{s} a_{ij} f(Y_j), \quad i = 1, \ldots, s, \\ \eta_1 &= \eta_0 + h\sum_{i=1}^{s} b_i f(t_0 + c_i h, Y_i). \end{aligned}$$

Man sieht schnell ein, daß die Definitionen 4.1 und 4.3 äquivalent sind.
Ein Runge-Kutta-Verfahren läßt sich graphisch veranschaulichen, siehe Abbildung 4.3. Innerhalb des Intervalls $[t_0, t_0 + h]$ werden an den Stützstellen $t_0 + c_i h$ Näherungen der Steigungen der Lösungsfunktion x berechnet. Diese Steigungen werden anschließend mit einer Gewichtung gemittelt. Der gemittelte Wert dient dann als Richtung zum nächsten Punkt η_1.

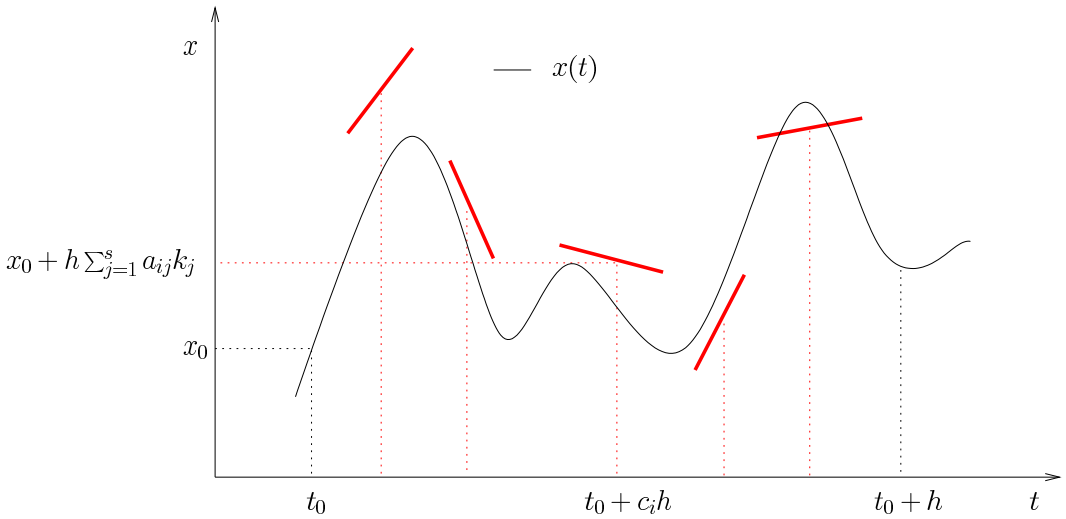

Abbildung 4.3: Veranschaulichung eines Runge-Kutta-Verfahrens. Die Steigungsnäherungen liegen im Vektorfeld der Differentialgleichung

Ein Blick auf die Gleichungen (4.7) verdeutlicht, daß bei einem expliziten Runge-Kutta-Verfahren jedes k_i lediglich von den k_j mit $j < i$ abhängt. Also können

sämtliche Steigungen k_i sukzessive berechnet werden, und damit auch η_1. Bei einem impliziten Runge-Kutta-Verfahren finden sich auch k_j mit $j \geq i$ als Argumente der Funktion f bei der Berechnung von k_i. Die k_i sind folglich nicht mehr direkt (explizit) berechenbar. Stattdessen muß ein nichtlineares Gleichungssystem mit allen k_i, $i = 1, \ldots, s$ als Unbekannten gelöst werden. Damit ist ein implizites Runge-Kutta-Verfahren nicht nur aufwendiger, es muß auch die Existenz und Eindeutigkeit des Gleichungssystems verifiziert werden.

Satz 4.4. *Es sei ein Anfangswertproblem der Definition 3.2 gegeben. Es gelte die Lipschitz-Bedingung (3.4) mit Konstante L in einer Umgebung von (t_0, η_0). Es sei*

$$h \cdot L \cdot \max_{i=1,\ldots,s} \sum_{j=1}^{s} |a_{ij}| < 1. \tag{4.9}$$

Dann existiert eine eindeutige Lösung des nichtlinearen Gleichungssystem (4.7), welche durch Fixpunktiteration gewonnen werden kann. Ist f p-mal stetig differenzierbar, so sind die $k_i = k_i(h)$ als Funktionen von h in einer Umgebung von (t_0, η_0) ebenfalls p-mal stetig differenzierbar, und damit auch η_1.

Beweis. Der Beweis wird in zwei Teilen geführt.
(I) Zunächst wird die Existenz und Eindeutigkeit gezeigt. Dabei wird das Gleichungssystem (4.7) als Fixpunktgleichung aufgefaßt:

$$k_i^{m+1} = f(t_0 + c_i h, \eta_0 + h \sum_{j=1}^{s} a_{ij} k_j^m), \quad i = 1, \ldots, s.$$

Hierbei bezeichnet m den Iterationsindex. Man beachte auch, daß jedes k_i die Dimension n von f besitzt. Nun setzt man

$$k = (k_1, \ldots, k_s)^T \in \mathbb{R}^{s \cdot n} \quad \text{mit der Norm} \quad \|k\| = \max_{i=1,\ldots,s} \|k_i\|_2 .$$

Jetzt konstruiert man eine Fixpunktgleichung $k = F(k)$:

$$\begin{aligned} F_i(k) &:= f(t_0 + c_i h, \eta_0 + h \sum_{j=1}^{s} a_{ij} k_j^m), \quad i = 1, \ldots, s \text{ und} \\ F(k) &:= \big(F_1(k), \ldots, F_s(k)\big)^T. \end{aligned}$$

Dann folgt mit der Lipschitz-Bedingung und wiederholter Anwendung der Dreiecksungleichung:

$$\left\| F(k^{m+1}) - F(k^m) \right\| \leq h \cdot L \cdot \max_{i=1,\ldots,s} \sum_{j=1}^{s} |a_{ij}| \cdot \left\| k^{m+1} - k^m \right\| .$$

(4.9) sichert, daß die Abbildung F eine Kontraktion ist. Der Banachsche Fixpunktsatz garantiert dann die Existenz und Eindeutigkeit der Lösung.
(II) Nun wird die Differenzierbarkeit der k_i bezüglich h bewiesen. Man wendet den Satz über implizite Funktionen an, den man in [24, Kapitel 8] oder [39, Abschnitt 3.4] findet. Es soll nach einer Funktion $k(h)$ aufgelöst werden, was im allgemeinen nur lokal möglich ist. Indem man die Abhängigkeit von h betont, setzt man

$$\Psi(h,k) := k - F(h,k) = k - F\big(h,k(h)\big) \stackrel{!}{=} 0.$$

Nach dem ersten Teil des Beweises existiert eine Lösung. Für die Anwendung des Satzes über implizite Funktionen ist die Regularität von $\frac{\partial \Psi}{\partial k}$ an einer Nullstelle (h^*,k^*) von Ψ erforderlich. Für $h=0$ ist die Matrix $\frac{\partial \Psi}{\partial k}$ jedoch die Einheitsmatrix, und damit, und aufgrund der Differenzierbarkeitsvoraussetzungen, folgt in einer Umgebung von $h=0$ die Regularität der Matrix. Dann kann man für ein Paar (h^*,k^*), welches das Gleichungssysten löst, formulieren:

$$\begin{aligned}
\frac{\partial k}{\partial h}(h^*) &= -\left(\frac{\partial \Psi}{\partial k}(h^*,k^*)\right)^{-1} \cdot \frac{\partial \Psi}{\partial h}(h^*,k^*) \\
&= -\left(Id - \frac{\partial F}{\partial k}(h^*,k^*)\right)^{-1} \cdot \left(-\frac{\partial F}{\partial h}(h^*,k^*)\right) \\
&= \left(Id - \frac{\partial F}{\partial k}(h^*,k^*)\right)^{-1} \cdot \frac{\partial F}{\partial h}(h^*,k^*).
\end{aligned}$$

Weiter ist $k_i = f(t_0,\eta_0)$ für $h=0$. Insgesamt existiert also die Inverse und k_i ist in einer Umgebung von $h=0$ stetig differenzierbar, wobei sich die Eigenschaften von f auf $k=k(h)$ übertragen.
Den Beweis findet man in [28, Theorem 7.2] und ausführlicher in [18, Satz 6.28]. □

Bemerkung 4.5. *Die Fixpunktiteration wird nur im Beweis verwendet. Gerade bei steifen Differentialgleichungen mit einer großen Lipschitz-Konstante L wäre die Fixpunktiteration nicht durchführbar. Praktisch wird man auf andere Verfahren zur Lösung des Gleichungssystems zurückgreifen. In [29, Abschnitt IV.8], [18, Abschnitt 6.2.2] und [58, Abschnitt 2.8] wird mit Newton-Iterationen gearbeitet.*

Es lassen sich relativ leicht die Konsistenzordnungen der Verfahren aus Beispiele 4.2 bestimmen, indem die entsprechenden Taylor-Entwicklungen gebildet werden. Dies ist aber nur eine Verifizierung, und keine Herleitung eines Runge-Kutta-Verfahrens. Nun werden erste Bedingungen an die Einträge eines Butcher-Tableaus gestellt.

Satz 4.6. *Das s-stufige Runge-Kutta-Verfahren der Definition 4.1 ist genau dann konsistent, wenn*

$$\sum_{i=1}^{s} b_i = b_1 + b_2 + \ldots + b_s = 1$$

gilt.

Beweis. Es werden die Bezeichnungen der Definition 3.20 verwendet. Zuerst bildet man die Taylor-Entwicklung von k_i:

$$k_i = f(t_0 + c_i h, \eta_0 + h \sum_{j=1}^{s} a_{ij} k_j) = f(t_0, \eta_0) + \mathcal{O}(h).$$

Dieser Term wird in die Verfahrensfunktion Φ eingesetzt:

$$\Phi(t_0, \eta_0, h, f) = \sum_{i=1}^{s} b_i k_i = f(t_0, \eta_0) \sum_{i=1}^{s} b_i + \mathcal{O}(h).$$

Der lokale Diskretisierungsfehler folgt dann mit erneuter Taylor-Entwicklung und $\dot{x}(t_0) = f(t_0, \eta_0)$:

$$\begin{aligned} \tau(t_0, \eta_0, h, f) &= \frac{1}{h}\Big(x(t_0 + h) - x(t_0) - h\Phi(t_0, \eta_0, h, f)\Big) \\ &= \frac{1}{h}\Big(hf(t_0, \eta_0) - hf(t_0, \eta_0) \sum_{i=1}^{s} b_i + \mathcal{O}(h^2)\Big) \\ &= f(t_0, \eta_0) \cdot \Big(1 - \sum_{i=1}^{s} b_i\Big) + \mathcal{O}(h). \end{aligned}$$

Damit gilt $\tau(t_0, \eta_0, h, f) = \mathcal{O}(h)$ genau dann, wenn $\sum b_i = 1$ erfüllt ist. □

Diese Forderung ist plausibel, handelt es sich bei den Koeffizienten b_i doch um die Gewichte, die die Steigungen beim Schritt von t_0 nach t_0+h mitteln. Die Summierung zu 1 stimmt ferner mit der Summierung der Gewichte bei der numerischen Integration in Kapitel 2 überein.
Der folgende Satz schildert, wann ein Runge-Kutta-Verfahren für beide Schreibweisen eines Anfangswertproblems aus Satz 3.11 die gleichenn Resultate erzeugt.

Satz 4.7 (Invarianz von Runge-Kutta-Verfahren). *Ein Runge-Kutta-Verfahren ist genau dann invariant gegen Autonomisierung, wenn es konsistent ist und des weiteren gilt:*

$$\sum_{j=1}^{s} a_{ij} = c_j, \quad i = 1, \ldots, s.$$

Beweis. Bei expliziten Runge-Kutta-Verfahren ist der Beweis ziemlich kurz, siehe [27, Lemma 2.24] oder [32, Satz 2.2]. Bei impliziten Runge-Kutta-Verfahren bedarf es einer Zusatzargumentation, siehe [18, Anwendung des Satzes 6.28]. □

Dieser Satz entspringt retrospektiv auch (4.5) aus dem Abschnitt 4.2. Die a_{ij} entstammten Integrationsgewichten d_j, die zur Berechnung des Wertes bei c_i benötigt wurden. Die d_j summieren sich aber zu Eins auf.

4.4 Explizite Runge-Kutta-Verfahren

Explizite Runge-Kutta-Verfahren sind kein Themenkomplex dieser Diplomarbeit. Dennoch sind einige Informationen zu ihnen unerläßlich. Im Abschnitt 4.3 wurden in Beispiel 4.2 verschiedene Runge-Kutta-Verfahren abgebildet. Ferner wurde darauf verwiesen, daß man die Konsistenzordnungen mit den Taylorentwicklungen berechnen kann. Es ist hier jedoch grundsätzlich nicht die Verifizierung der Konsistenzordnung von Runge-Kutta-Verfahren von Interesse, sondern die Berechnung der Einträge eines Butcher-Tableaus, die das Runge-Kutta-Verfahren bei vorgebener Stufenzahl s definieren, oder die Ermittlung eines Butcher-Tableaus einschließlich der Stufenzahl s bei Vorgabe einer Konsistenzordnung. Die Forderungen an die Koeffizienten c_i, b_i, a_{ij} können mit einer Taylor-Entwicklung gebildet werden. Dies kann man beispielhaft in [27, Abschnitt 2.4] oder [32, Abschnitt 2.3] nachvollziehen. Der Aufwand ist erheblich und die Rechnungen werden schnell unübersichtlich.
Explizite Runge-Kutta-Verfahren erscheinen wegen ihrer geringeren Anzahl von unbekannten a_{ij} im Vergleich zu impliziten Runge-Kutta-Verfahren prädestiniert, geringeren Aufwand nicht nur bei der Durchführung, sondern auch bei der Ermittlung der Koeffizienten, aufzuweisen. Dies trifft aber in entscheidenen Aspekten nicht zu. Es werden hier einige Resultate zu expliziten Runge-Kutta-Verfahren vorgestellt, die nicht bewiesen werden. Mit vorgreifenden Bemerkungen auf die Ergebnisse des Abschnitts 4.5 kann dann der erhebliche Vorzug von impliziten Runge-Kutta-Verfahren begründet werden. Genauere Betrachtungen finden sich im Standardwerk [12, Abschnitt 32].

Theorem 4.8 (Maximale Konsistenzordnung eines ERK). *Ein s-stufiges explizites Runge-Kutta-Verfahren hat maximal die Konsistenzordnung s. Es gilt also die folgende Ungleichung für die Konsistenzordnung p:*

$$p \leq s.$$

Beweis. Siehe [12, Theorem 321A] oder [27, Lemma 2.23]. □

Es kann auch gezeigt werden, daß ab der Stufenzahl $s = 5$ die Konsistenzordnung stets echt kleiner als s ist, siehe [12, Theorem 323B]. Der Zusammenhang zwischen Stufenzahl und Konsistenzordnung wird oft in einer Tabelle zusammengefaßt. Die Tabelle 4.1 entstammt [27, Abschnitt 2.4].

p	1	2	3	4	5	6	7	8	...	≥ 9
m_p	1	2	3	4	6	7	9	11	...	$p+3$

Tabelle 4.1: Mindeststufenzahl m_p eines ERK für die Ordnung p

Die Schranken m_p werden auch als `Butcher-Schranken` bezeichnet. Das rasche Anwachsen der Anzahl der Bedingungen bei einer Taylor-Entwicklung vermittelt Tabelle 4.2, die erneut in [27, Abschnitt 2.4] zu finden ist.

p	1	2	3	4	5	6	7	8	9	10	...	20
N_p	1	2	4	8	17	37	85	200	486	1205	...	20247374

Tabelle 4.2: Anzahl der Bedingungsgleichungen N_p für ein ERK

Das Wachsen der Ordnungsbedingungen ist gewaltig. Gleichzeitig entsteht aber keine erhellende Systematik. Damit ist es sehr schwer, ein Verfahren höherer Ordnung zu ermitteln. Für die impliziten Runge-Kutta-Verfahren gilt dagegen:

- Es können deutlich höhere Ordnungen erreicht werden,
- Verfahren höherer Ordnung lassen sich leicht finden und erstellen.

4.5 Implizite Runge-Kutta-Verfahren

Dieser Abschnitt wird von der Konstruktion impliziter Runge-Kutta-Verfahren beherrscht. Hierbei wird zunächst nach Verfahren maximaler Konsistenzordnung gestrebt. Diese Fixierung wird in Abschnitt 4.6 gelöst. Die Vorgehensweise lehnt sich damit an die Abschnitte 2.6 und 2.7 an.
Die höhere Anzahl von verfügbaren Einträgen in der Matrix A lassen erweiterte Möglichkeiten vermuten. Gründe für die Suche nach effektiven impliziten Runge-Kutta-Verfahren sind in [12, Abschnitt 340] angegeben. Wichtig sind hier

- eine höhere Konsistenzordnung im Vergleich zu expliziten Verfahren,
- eine schönere algebraische Form und Struktur,

- die Effektivität bei der Anwendung auf steife Differentialgleichungen.

Besonders der dritte Aspekt ist eine der Hauptursachen für den Erfolg impliziter Runge-Kutta-Verfahren, muß in dieser Arbeit aber weitgehend ausgespart bleiben. Außerdem lassen sich lineare Differentialgleichungen stets von einem impliziten Ansatz in eine explizite Ausführung umformen.
In diesem Abschnitt wird sich das Manko nicht vermeiden lassen, daß nicht die komplette Theorie impliziter Runge-Kutta-Verfahren hergeleitet werden kann. Die Ursache dafür ist die schon geschilderte Schwierigkeit, die Konsistenzbedingungen verständlich und zugleich kurz und bündig niederzuschreiben. Es ist der Verdienst Butchers, aber auch Hairers und Wanners, die Bedingungen für die Konsistenzordnung überschaubar geordnet zu haben. Es wurde dazu eine algebraische Theorie der Runge-Kutta-Verfahren entwickelt, die auf der Technik der sogenannten „monoton indizierten Wurzel-Bäume“ basiert. Dieser Technik kann hier nicht wiedergegeben werden. Dies geschieht ausführlich in [12, Abschnitt 14], [28, Abschnitt II.2] sowie einführend in [32, Abschnitt 2.4]. Stattdessen soll gleich mit den sogenannten „Vereinfachenden Bedingungen“ gearbeitet werden, die die Butcher-Darstellung erzeugt. Die Konstruktion verschiedener Typen von impliziten Runge-Kutta-Verfahren wird dadurch ausreichend deutlich, auch wenn die Herkunft der Gleichungen so etwas nebulös bleibt.
Die Vorgehensweise der monoton indizierten Wurzel-Bäume beruht auf einer geschickten Ordnung der durch Taylor-Entwicklung entstandenen Konsistenzbedingungen sowie einer Reformulierung mit graphentheoretischen Hilfsmitteln. Dazu müssen einleitend ein paar nicht weiter erklärte Bemerkungen und Definitionen statuiert werden.

Bemerkung 4.9. *Im Zusammenhang mit Graphen ist mit t nicht die Zeit, sondern ein Baum gemeint, siehe [12, Abschnitt 14].*

Es lassen sich Funktionen auf Bäumen definieren.

Definition 4.10. *T sei eine Menge von Bäumen. Die Funktion*

$$\begin{aligned} r : T &\longrightarrow \mathbb{N}, \\ t &\longmapsto r(t), \quad t \in T \end{aligned}$$

gibt die Anzahl der Knoten im Baum t an.

Eine zentrale Bedeutung hat die folgende Definition:

Definition 4.11 (Dichte eines Baumes (density)). *T sei eine Menge von Bäumen. Dann kann man eine Funktion*

$$\begin{aligned} \gamma : T &\longrightarrow \mathbb{R}, \\ t &\longmapsto \gamma(t), \quad t \in T \end{aligned}$$

definieren gemäß [12, Abschnitt 144], die die `Dichte` eines Baumes angibt.

Die zweite wesentliche Komponente in der graphentheoretischen Interpretation eines Runge-Kutta-Verfahrens ist die Zuordnung eines Polynoms zu einem Baum:

Definition 4.12 (Elementargewicht (elementary weight)). *Es sei t ein Wurzel-Baum. Gemäß [12, Abschnitt 304] kann man einem Wurzel-Baum ein Polynom zuordnen. Dieses wird mit*

$$\Phi(t)$$

bezeichnet und heißt das `Elementargewicht` *des Wurzel-Baumes t.*

Φ ist ein Term, der polynomial von den Einträgen A, b und c der Butcher-Tableaus abhängt.
Die beiden Definitionen 4.11 und 4.12 beinhalten die Quintessenz der von Butcher eingeführten Schreibweisen. In ihrem Zusammenwirken sind sie ausschlaggebend für die Konsistenzordnung eines Runge-Kutta-Verfahrens. Mit ihnen läßt sich das Hauptresultat über die Konsistenzordnung eines Runge-Kutta-Verfahrens finden:

Theorem 4.13 (Butcher). *Ein Runge-Kutta-Verfahren hat die Konsistenzordnung p genau dann, wenn für alle $t \in T$ mit $r(t) \leq p$ gilt:*

$$\Phi(t) = \frac{1}{\gamma(t)}. \tag{4.10}$$

Die Gültigkeit dieser Gleichungen wird später mit $G(p)$ bezeichnet.

Beweis. Siehe [12, Theorem 307B]. □

Formal hat man eigentlich die Aufgabe des gesamten Kapitels gelöst. Schließlich muß man nur die Gleichungen des Satzes 4.13 mit den dazugehörigen Definitionen für Φ und γ lösen. Tatsächlich ist dies aber nicht opportun:

- Die Bedingungen sind nach wie vor zahlreich, auch wenn das durch die nur angedeuteten Schreibweisen verschleiert wird.
- Die Polynome Φ sind sehr kompliziert. Sie bilden riesige Summen mit aufwendigen Verknüpfungen der Butcher-Tableau-Einträge A, b und c, siehe etwa [12, Abschnitt 304].
- Es wird keine übersichtliche Struktur erkennbar, die zu allgemeinen Lösungen verhilft.

Damit scheidet das Theorem 4.13 als technisches Verfahren zur Gewinnung eines impliziten Runge-Kutta-Verfahrens aus. Stattdessen gelingt es aber, die aufwendigen Gleichungen (4.10) in mehrere äquivalente Bedingungen aufzuspalten. Es verbleibt kein Ausweg, als den vielbeschworenen Sprung in das kalte Wasser zu

wagen, und mit diesen vorerst völlig unbegründeten und unvertrauten Gleichungen zu starten. Die jetzt folgenden Gleichungen werden den ganzen Abschnitt und darüberhinaus begleiten. Die Bemerkung 4.15 ergänzt die folgende Definition 4.14 wesentlich.

Definition 4.14 (Vereinfachende Bedingungen (simplifying assumptions)). *Die Koeffizienten eines Runge-Kutta-Verfahrens genügen den folgenden Bedingungen, wenn*

$$A(n_1) \quad : \quad \sum_{j=1}^{s} a_{ij} c_j^{k-1} = \frac{1}{k} c_i^k, \quad i = 1, \ldots, s, \quad k = 1, \ldots, n_1,$$

$$B(n_2) \quad : \quad \sum_{i=1}^{s} b_i c_i^{k-1} = \frac{1}{k}, \quad k = 1, \ldots, n_2,$$

$$C(s) \quad : \quad c_i,\ i = 1, \ldots, s \text{ sind die Nullstellen von } L_s(2t-1),$$

$$D(n_3) \quad : \quad \sum_{i=1}^{s} b_i c_i^{k-1} a_{ij} = \frac{1}{k} b_j (1 - c_j^k), \quad j = 1, \ldots, s, \quad k = 1, \ldots, n_3,$$

$$E(n_4, n_5) \quad : \quad \sum_{i,j=1}^{s} b_i c_i^{k-1} a_{ij} c_j^{l-1} = \frac{1}{l(k+l)}, \quad k = 1, \ldots, n_4, \quad l = 1, \ldots, n_5,$$

$G(n_6)$: Für jeden Baum t mit nicht mehr als n_6 Knoten gilt die Ordnungsbedingung

$$\Phi(t) = \frac{1}{\gamma(t)}$$

gelten. Diese Bedingungen heißen auch `Vereinfachende Bedingungen`*.*

Es werden gebräuchlicherweise nur die obigen Großbuchstaben wie $A(n_1)$ verwendet, wenn auf die Gültigkeit der dazugehörigen Gleichungen verwiesen wird. Die Gültigkeit einiger dieser Vereinfachenden Bedingungen wird der Schlüssel zu den Konsistenzordnungen sein.
Es ist an dieser Stelle wichtig, auf eine Notation hinzuweisen, die ansonsten geeignet ist, Verwirrung zu stiften:

Bemerkung 4.15. *In vielen Büchern wird die Bedingung $A(n_1)$ auch als $C(n_1)$ bezeichnet. Die hier mit $C(s)$ titulierte Bedingung erhält dann keinen eigenen Buchstaben. Die hier verwendete Notation richtet sich nach [12]. Diese Festlegung gründet sich auf die Bedeutung von Butcher und seiner Standardwerke im Bereich der Runge-Kutta-Verfahren (wobei er selber in [13] und [14] eine andere Bezeichnungsweise wählt).*

Die Vereinfachenden Bedingungen wurden hier zwar einfach per Definition eingeführt, sie lassen sich aber zum Teil relativ leicht erläutern und begründen. Für den Spezialfall eines Anfangswertproblems mit

$$f(t, x(t)) = f(t) = (t - t_0)^{k-1}, \quad k = 1, \ldots, n_2$$

erhält man bei exakter Rechnung für das Intervall $[t_0, t_0 + h]$:

$$x(t_0 + h) = \eta_0 + \int_{t_0}^{t_0+h} f(t)\, dt = \ldots = \eta_0 + \frac{1}{k} h^k.$$

Bei Anwendung des Runge-Kutta-Verfahrens ergibt sich

$$\eta_1 = \eta_0 + h \sum_{i=1}^{s} b_i f(t_0 + c_i h) = \eta_0 + h \sum_{i=1}^{s} b_i (c_i h)^{k-1}.$$

Fordert man die Gleichheit von $x(t_0+h)$ und η_1, so ensteht die Bedingung $B(n_2)$. Das Runge-Kutta-Verfahren liefert dann also exakte Ergebnisse für Polynome bis zum Grad $n_2 - 1$. Analog impliziert die Bedingung

$$A(n_1): \quad \sum_{j=1}^{s} a_{ij} c_j^{k-1} = \frac{1}{k} c_i^k, \quad i = 1, \ldots, s, \quad k = 1, \ldots, n_1,$$

daß im Intervall $[0, c_i]$ Polynome bis zum Grad $n_1 - 1$ exakt integriert werden. Eine derartige Bedingung wurde schon aus den Formeln (4.4) bis (4.6) im Abschnitt 4.2 ersichtlich.
Einen Überblick über diese Interpretationen der Vereinfachenden Bedingungen genehmigt die folgende Bemerkung.

Bemerkung 4.16. *Wenn P und Q Polynome mit* $\operatorname{Grad} P < m$ *und* $\operatorname{Grad} Q < n$ *sind, dann sind die die Bedingungen $A(m)$, $B(m)$, $D(m)$ und $E(m, n)$ äquivalent zu folgenden Aussagen:*

$$\begin{aligned}
A(m) \quad &\Longleftrightarrow \quad \sum_{j=1}^{s} a_{ij} P(c_j) = \int_0^{c_i} P(x)\, dx, \quad i = 1, \ldots, s, \\
B(m) \quad &\Longleftrightarrow \quad \sum_{j=1}^{s} b_j P(c_j) = \int_0^1 P(x)\, dx, \\
D(m) \quad &\Longleftrightarrow \quad \sum_{i=1}^{s} b_i P(c_i) a_{ij} = b_j \int_{c_j}^1 P(x)\, dx, \quad j = 1, \ldots, s, \\
E(m, n) \quad &\Longleftrightarrow \quad \sum_{j=1}^{s} \sum_{i=1}^{s} b_i P(c_i) a_{ij} Q(c_j) = \int_0^1 P(x) \left(\int_0^x Q(t)\, dt \right) dx,
\end{aligned}$$

wie in [13, Theorem 343B] erklärt wird, oder direkt nachgerechnet werden kann.

Die Vereinfachenden Bedingungen sind vielfältig miteinander verknüpft. Als nächstes werden drei exemplarische Implikationen vorgeführt, die einen Eindruck von den hierbei typischen Beweistechniken vermitteln.
Die erste Beziehung nutzt einfache algebraische Umformungen.

Satz 4.17. *Mit der „und"-Verknüpfung $\wedge$ gilt*

$$A(n) \wedge B(m+n) \Longrightarrow E(m,n).$$

Beweis. Es gilt

$$A(n) \quad : \quad \sum_{j=1}^{s} a_{ij} c_j^{k-1} = \frac{1}{k} c_i^k, \quad i = 1, \ldots, s, \quad k = 1, \ldots, n, \tag{4.11}$$

$$B(m+n) \quad : \quad \sum_{i=1}^{s} b_i c_i^{k-1} = \frac{1}{k}, \quad k = 1, \ldots, m+n. \tag{4.12}$$

Zuerst bringt man die Terme von (4.11) auf eine Seite. Dann multipliziert man mit $b_i c_i^{l-1}$ (für $l = 1, \ldots, m$) und summiert über $i = 1, \ldots, s$. Bei anschließender Verwendung von (4.12) ergibt sich für $l = 1, \ldots, m$ und $k = 1, \ldots, n$:

$$0 = \sum_{i,j=1}^{s} b_i c_i^{l-1} a_{ij} c_j^{k-1} - \frac{1}{k} \sum_{i=1}^{s} b_i c_i^{k+l-1} = \sum_{i,j=1}^{s} b_i c_i^{l-1} a_{ij} c_j^{k-1} - \frac{1}{k(l+k)}. \tag{4.13}$$

Die rechte Seite von (4.13) ist die Vereinfachende Bedingung $E(m,n)$. □

Die zweite Beziehung nutzt Techniken aus der Theorie der Gauß-Quadratur.

Satz 4.18.

$$B(2s) \Longrightarrow C(s).$$

Beweis. Es sei ψ ein Polynom des Grades l. Es gilt

$$B(2s): \quad \sum_{i=1}^{s} b_i c_i^{k-1} = \frac{1}{k}, \quad k = 1, \ldots, 2s.$$

Die rechte Seite von $B(2s)$ kann man als

$$\int_0^1 t^{k-1}\, dt, \quad k = 1, \ldots, 2s$$

interpretieren, den Term c_i^{k-1} der linken hingegen als

$$c_i^{k-1} = t^{k-1}\Big|_{t=c_i}, \quad k = 1, \ldots, 2s.$$

Diese Interpretationen lassen sich per Summation und Skalarmultiplikation auf ein Polynom ψ mit $\operatorname{Grad}\psi \leq 2s-1$ ausdehnen, so daß

$$\sum_{i=1}^{s} b_i \psi(c_i) = \int_0^1 \psi(t)\,dt$$

gilt. Dann bilden die Vektoren c und b jedoch eine Qudraturregel mit dem maximal möglichen Exaktheitsgrad $2s-1$. Nach Satz 2.16 müssen die c_i die Nullstellen des Legendre-Polynoms L_s sein. □

Die dritte Beziehung nutzt die Regularität von Matrizen aus.

Satz 4.19. *Es gelte $c_i \neq c_j$ für $i \neq j$ und $b_i \neq 0$ für alle i. Dann gilt:*

$$B(s+m) \wedge E(s,m) \Longrightarrow A(m).$$

Beweis. Es gelten:

$$\begin{aligned} B(s+m) \quad &: \quad \sum_{i=1}^{s} b_i c_i^{k-1} = \frac{1}{k}, \quad k = 1,\ldots,s+m, \\ E(s,m) \quad &: \quad \sum_{i,j=1}^{s} b_i c_i^{k-1} a_{ij} c_j^{l-1} = \frac{1}{l(k+l)}, \quad l = 1,\ldots,s, \quad k = 1,\ldots,m. \end{aligned}$$

Aus $E(s,m)$ folgt für $k = 1,\ldots,s$ und $l = 1,\ldots,m$:

$$\begin{aligned} 0 &= \sum_{i,j=1}^{s} b_i c_i^{k-1} a_{ij} c_j^{l-1} - \frac{1}{l(k+l)}, \\ \Longrightarrow\ 0 &= \sum_{i,j=1}^{s} b_i c_i^{k-1} a_{ij} c_j^{l-1} - \frac{1}{l}\sum_{i=1}^{s} b_i c_i^{k+l-1}, \\ \Longrightarrow\ 0 &= \sum_{i=1}^{s} b_i c_i^{k-1}\Big(\sum_{j=1}^{s} a_{ij} c_j^{l-1} - \frac{1}{l} c_i^{l}\Big). \end{aligned}$$

Für fixiertes l kann die letzte Gleichung als lineares Gleichungssystem aufgefaßt werden. Hierbei wird der Term in der Klammer als Variable gesehen. Die Voraussetzungen an c_i und b_i garantieren die Regularität der Koeffizientenmatrix, wie man mit der Vandermonde-Matrix verifizieren kann, siehe [31, Abschnitt 13.6]. Der Term in der Klammer muß verschwinden, und folglich $A(m)$ gelten. □

Weitere Beziehungen werden jetzt ohne Beweise in einem Satz zusammengefaßt. Auch die vorangegangenen drei Sätze werden nochmals aufgeführt.

Satz 4.20.

Satz 4.17:	1)	$A(n) \wedge B(m+n) \Longrightarrow E(m,n)$,
Satz 4.18:	2)	$B(2s) \Longrightarrow C(s)$,
Satz 4.19:	3)	$b_i \neq 0 \wedge c_i \neq c_j \wedge B(s+n) \wedge E(s,n) \Longrightarrow A(n)$,
	4)	$B(m+n) \wedge D(m) \Longrightarrow E(m,n)$,
	5)	$b_i \neq 0 \wedge c_i \neq c_j \wedge B(m+s) \wedge E(m,s) \Longrightarrow D(m)$,
	6)	$B(s) \wedge C(s) \Longrightarrow B(2s)$,
	7)	$B(2s) \wedge A(s) \Longrightarrow D(s)$,
	8)	$B(2s) \wedge D(s) \Longrightarrow A(s)$.

Beweis. Siehe [12, Abschnitt 342 und 343], [58, Abschnitt 2.5] und die angegebenen Sätze. □

Es ist nun der Zeitpunkt gekommen, wieder die „Blackbox“ des Abschnitts zu verwenden - die graphentheoretischen Voraussetzungen.

Satz 4.21. *Folgende Implikationen gelten:*

$$G(2s) \Longrightarrow E(s,s), \tag{4.14}$$

$$G(2s) \Longrightarrow B(2s), \tag{4.15}$$

$$A(s) \wedge B(2s) \wedge D(s) \Longrightarrow G(2s). \tag{4.16}$$

Beweis. Siehe [13, Theorem 342C]. Diese Beziehungen sind eine Folgerung der graphischen Veranschaulichung der Konsistenzbedingungen. Graphentheoretisch stellen die rechten Seiten von (4.14) und (4.15) gewisse Bäume dar, die bestimmte Ordnungsbedingungen für die Konsistenzordnung $2s$ repräsentieren. Die linke Seite von (4.16) umfaßt alle Bäume ab, die für die Konsistenzbedingungen gebildet werden. □

Damit gelangt man zu dem vorerst wichtigsten Satz des Abschnitts:

Satz 4.22. *Es existiert genau ein s-stufiges Runge-Kutta-Verfahren der Ordnung $p = 2s$. Es gilt*

$$A(s) \wedge B(s) \wedge C(s) \Longleftrightarrow G(2s).$$

Verfahren dieses Typs werden ***Gauß-Legendre-Methoden*** *genannt.*

Beweis. Mit Implikationsketten ergibt sich die Hinrichtung durch

$$B(s) \wedge C(s) \overset{\text{Satz 4.20, 6)}}{\Longrightarrow} B(2s),$$

$$B(2s) \wedge A(s) \overset{\text{Satz 4.20, 7)}}{\Longrightarrow} D(s),$$

$$A(s) \wedge B(2s) \wedge D(s) \overset{\text{Satz 4.21}}{\Longrightarrow} G(2s).$$

Die Rückrichtung folgt mit

$$\begin{aligned} G(2s) &\overset{\text{Satz 4.21}}{\Longrightarrow} B(2s) \text{ und } E(s,s), \\ B(2s) &\overset{\text{Satz 4.20, 2)}}{\Longrightarrow} C(s), \\ B(2s) \wedge C(s) \wedge E(s,s) &\overset{\text{Satz 4.20, 3)}}{\Longrightarrow} A(s). \end{aligned}$$

□

An dieser Stelle ist es lohnend, zur Bemerkung 3.27 zurückzukehren. Es ist durchaus beachtlich, daß es tatsächlich gelingt, die Gauß-Quadratur-Ordnung komplett auf die Runge-Kutta-Verfahren zu übertragen. Schließlich ist ein Anfangswertproblem in der Darstellung (3.4) doch deutlich komplexer als ein Integrationsproblem (2.23).
Ein Gauß-Legendre-Verfahren der Ordnung $2s$ läßt sich nun vergleichsweise einfach konstruieren.

Konstruktionsvorschrift 4.23 (Gauß-Legendre-Verfahren). *Ein s-stufiges* `Gauß-Legendre-Verfahren` *der Ordnung $2s$ erstellt man folgendermaßen:*

1. Wahl einer Stufenzahl s
2. Berechnung der c_i aus $C(s)$ (siehe A.3 oder [1, Kapitel 25])
3. Berechnung der b_i aus $B(s) \longrightarrow$ lineares Gleichungssystem:

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ c_1 & c_2 & \dots & c_s \\ \vdots & \vdots & \ddots & \vdots \\ c_1^{s-1} & c_2^{s-1} & \dots & c_s^{s-1} \end{pmatrix} \begin{pmatrix} b_1 \\ \vdots \\ \vdots \\ b_s \end{pmatrix} = \begin{pmatrix} \frac{1}{1} \\ \vdots \\ \vdots \\ \frac{1}{s} \end{pmatrix}$$

4. Berechnung der a_{ij} aus $A(s) \longrightarrow$ lineares Gleichungssystem:

$$\begin{pmatrix} a_{1,1} & \cdots & a_{1,s} \\ \vdots & \ddots & \vdots \\ a_{s,1} & \dots & a_{s,s} \end{pmatrix} \begin{pmatrix} 1 & c_1 & \cdots & c_1^{s-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & c_s & \dots & c_s^{s-1} \end{pmatrix} = \begin{pmatrix} c_1 & \cdots & c_1^s \\ \vdots & \ddots & \vdots \\ c_s & \dots & c_s^s \end{pmatrix} \begin{pmatrix} \frac{1}{1} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{s} \end{pmatrix}$$

Die Gleichungssysteme haben eindeutige Lösungen, da die Regularität der Matrizen gesichert ist, wie man erneut mit der Vandermonde-Matrix verifizieren kann.

Beispiel 4.24. *Für $s = 1$ erhält man die implizite Mittelpunktsregel, die bereits in Beispiel 4.2 visualisiert wurde. Die Gauß-Legendre-Verfahren für die Stufenzahlen $s = 2$ und $s = 3$ tabelliert Abbildung 4.4.*

$$\begin{array}{c|cc} \frac{3-\sqrt{3}}{6} & \frac{1}{4} & \frac{3-2\sqrt{3}}{12} \\ \frac{3+\sqrt{3}}{6} & \frac{3+2\sqrt{3}}{12} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \qquad \begin{array}{c|ccc} \frac{5-\sqrt{15}}{10} & \frac{5}{36} & \frac{10-3\sqrt{15}}{45} & \frac{25-6\sqrt{15}}{180} \\ \frac{1}{2} & \frac{10+3\sqrt{15}}{72} & \frac{2}{9} & \frac{10-3\sqrt{15}}{72} \\ \frac{5+\sqrt{15}}{10} & \frac{25+6\sqrt{15}}{180} & \frac{10+3\sqrt{15}}{45} & \frac{5}{36} \\ \hline & \frac{5}{18} & \frac{4}{9} & \frac{5}{18} \end{array}$$

Abbildung 4.4: Gauß-Legendre-Verfahren für $s = 2$ und $s = 3$

Der bedeutsame nächste Satz verallgemeinert den Satz 4.22.

Satz 4.25. *Ein s-stufiges Runge-Kutta-Verfahren hat die Konsistenzordnung p, wenn*

$$B(p) \wedge A(l) \wedge D(m) \quad \text{mit} \quad p \leq \min\left(1 + m + l, 2l + 2\right)$$

erfüllt ist.

Beweis. Siehe [14, Theorem 7] oder [28, Theorem 7.4]. □

Erfüllt ein Runge-Kutta-Verfahren gewisse Bedingungen, lassen sich nun leicht die zugehörigen Konsistenzordnungen angeben.
Schon am Anfang des Abschnitts wurde beschrieben, daß die Bedingung $B(p)$ dem Exaktheitsgrad $p - 1$ der zugrundeliegenden Quadraturformel entspricht. Satz 4.25 bestätigt, daß $B(p)$ für die Konsistenzordnung p stets erfüllt sein muß. Man erkennt rückblickend, daß die Konsistenzordnung höchstens gleich der Ordnung der zugehörigen Integrationsmethode sein kann. In Abschnitt 5.3 wird dieser Zusammenhang noch einmal hergeleitet und führt zum Satz 5.8.
Die Gauß-Legendre-Verfahren erfüllen trotz ihrer maximalen Konsistenzordnung $2s$ nicht alle Wünsche. Leichte Veränderungen schwächen die Konsistenzordnung ein wenig ab, bewirken aber ander Vorzüge. Der nächste Abschnitt wird einige maßgebliche Veränderungen thematisieren. Er lehnt sich an den Abschnitt 2.7 an.

4.6 Varianten impliziter Runge-Kutta-Verfahren

Wie auch in Abschnitt 2.7 besteht häufig der Wunsch, die Knoten c_i an gewissen Stellen zu fixieren. Für ein Einschrittverfahren zur Lösung von Anfangswertproblemen ist es vorteilhaft, wenn numerische Lösungen am Anfang und Ende eines Intervalls vorliegen, siehe Definition 3.15. Auch bei den numerischen Verfahren zur Lösung von Optimalsteuerungsproblemen in den Abschnitten 7.3 und 7.4 wird auf solche Diskretisierungen zurückgegriffen.

Aber auch andere Gesichtspunkte sind emminent. Bereits leicht veränderte Verfahren weisen bessere Stabilitätseigenschaften auf, wie Abschnitt 4.7 erwähnt. Allerdings gibt es hier verschiedenste Begriffe und Abstufungen, und dementsprechend auch viele Varianten, ein Verfahren nach geeigneten Maßstäben zu modifizieren.
In diesem Abschnitt werden die drei geläufigsten Varianten präsentiert, bei denen zudem die Verringerung der Konsistenzordnungen als gering einzustufen ist. Die drei fokussierten Spezialfälle korrespondieren mit den Quadraturvarianten des Abschnitts 2.7. Die Radau- und Lobatto-Methoden werden parallel entwickelt.

Satz 4.26. *Ein Runge-Kutta-Verfahren habe die Konsistenzordnung* $2s-1$ *bzw.* $2s-2$*. Dann sind die Knoten* c_i *die Nullstellen der folgenden Polynome:*

$$\begin{aligned} L_{s,\lambda}(2t-1) &= L_s(2t-1) + \lambda L_{s-1}(2t-1) \\ \text{bzw. } L_{s,\lambda,\mu}(2t-1) &= L_s(2t-1) + \lambda L_{s-1}(2t-1) + \mu L_{s-2}(2t-1). \end{aligned}$$

Beweis. Siehe [12, Theorem 345A]. Die Aussage des Satzes wird schon aus der Kombination der Sätze 2.20 und 4.25 verständlich, die die Bedeutung der Bedingung $B(2s-1)$ bzw. $B(2s-2)$ begründen. □

Im Satz 2.23 des Abschnitts 2.7 wurde gezeigt, wie die Koeffizienten λ und μ gewählt werden müssen, damit die gewünschten Knoten bei c_1 und c_s entstehen:

$$\text{Nullstellen von } L_{s,1,0} \quad \text{(Radau-I)}, \tag{4.17}$$

$$\text{Nullstellen von } L_{s,-1,0} \quad \text{(Radau-II)}, \tag{4.18}$$

$$\text{Nullstellen von } L_{s,0,-1} \quad \text{(Lobatto)}. \tag{4.19}$$

Der Tabelle 2.2 in Abschnitt 2.7 kann man alle wesentlichen Informationen zu den drei Verfahren entnehmen.
Die Gewichte b_i berechnet man mithilfe der Bedingungen $B(2s-1)$ beziehungsweise $B(2s-2)$. In [12, Theorem 345D] wird allgemein untersucht, wann die Gewichte b_i zu den Polynomen $L_{s,\lambda,\mu}$ des Satzes 4.26 positiv sind. Für die Radau- und Lobatto-Verfahren ist dies stets der Fall.

Satz 4.27. *Die Gewichte* b_i, $i = 1, \ldots, s$ *der verschiedenen Radau- und Lobatto-Regeln sind echt positiv.*

Beweis. Das Resultat folgert man aus [12, Theorem 345D]. [13, Theorem 344A] beinhaltet die Aussage direkt. □

Im Gegensatz zu den Einträgen c und b der Butcher-Tableaus sind die Matrixeinträge A für die drei obigen Verfahren nicht eindeutig bestimmt. Aus den Sätzen

4.20 und 4.25 wird erkennbar, daß gewisse alternative Auswahlmöglichkeiten zwischen den Vereinfachenden Bedingungen A und D bestehen. Hier gabeln sich die drei Methoden in verschiedene Unterfälle auf. Die gängigen Unterteilungen und Bezeichnungen lauten:

- Radau-I: Wahl der c_i durch (4.17) und der b_i durch $B(2s-1)$,
  1. Radau-I: Wahl der a_{ij} durch $A(s)$,
  2. Radau-IA: Wahl der a_{ij} durch $D(s)$,
- Radau-II: Wahl der c_i durch (4.18) und der b_i durch $B(2s-1)$,
  1. Radau-II: Wahl der a_{ij} durch $D(s)$,
  2. Radau-IIA: Wahl der a_{ij} durch $A(s)$,
- Lobatto: Wahl der c_i durch (4.19) und der b_i durch $B(2s-2)$,
  1. Lobatto-III: Wahl der a_{ij} durch $A(s-1)$ und $a_{is} = 0$, $i = 1, \ldots, s$,
  2. Lobatto-IIIA: Wahl der a_{ij} durch $A(s)$,
  3. Lobatto-IIIB: Wahl der a_{ij} durch $D(s)$,
  4. Lobatto-IIIC: Wahl der a_{ij} durch $A(s-1)$ und $a_{i1} = b_1$, $i = 1, \ldots, s$.

Mit dem Satz 4.25 kann man die Konsistenzordnungen $2s-1$ respektive $2s-2$ der Verfahren nachweisen, wobei die mannigfaltigen Beziehungen der Vereinfachenden Bedingungen des Satzes 4.20 als Hilfsmittel dienen. Eine beispielhafte Beweisführung demonstriert der nächste Satz.

Satz 4.28. *Die Verfahren Radau-IA und Radau-IIA besitzen die Konsistenzordnungen* $2s-1$.

Beweis. Die zugrundeliegenden Gauß-Qudraturmethoden haben den Exaktheitsgrad $2s-1$. Dann folgt für das Radau-IA-Verfahren:

$$B(2s-1) \wedge D(s) \overset{\text{Satz 4.20, 4)}}{\Longrightarrow} E(s, s-1),$$

$$b_i \neq 0 \wedge c_i \neq c_j \wedge B(2s-1) \wedge E(s, s-1) \overset{\text{Satz 4.20, 3)}}{\Longrightarrow} A(s-1),$$

$$B(2s-1) \wedge A(s-1) \wedge D(s) \overset{\text{Satz 4.25}}{\Longrightarrow} G(2s-1).$$

Für das Radau-IIA-Verfahren gilt:

$$B(2s-1) \wedge A(s) \overset{\text{Satz 4.20, 1)}}{\Longrightarrow} E(s-1, s),$$

$$b_i \neq 0 \wedge c_i \neq c_j \wedge B(2s-1) \wedge E(s-1, s) \overset{\text{Satz 4.20, 5)}}{\Longrightarrow} D(s-1),$$

$$B(2s-1) \wedge A(s) \wedge D(s-1) \overset{\text{Satz 4.25}}{\Longrightarrow} G(2s-1).$$

□

Die Konsistenzordnungen der anderen Verfahrensvarianten ergeben sich analog.

Beispiel 4.29. *Beispiele für $s = 2$ illustriert Abbildung 4.5.*

$$\begin{array}{c|cc} 0 & \frac{1}{4} & -\frac{1}{4} \\ \frac{2}{3} & \frac{1}{4} & \frac{5}{12} \\ \hline & \frac{1}{4} & \frac{3}{4} \end{array} \qquad \begin{array}{c|cc} \frac{1}{3} & \frac{1}{3} & 0 \\ 1 & 1 & 0 \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array} \qquad \begin{array}{c|cc} 0 & \frac{1}{2} & 0 \\ 1 & \frac{1}{2} & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Abbildung 4.5: v.l.n.r.: Radau-IA, Radau-II und Lobatto-IIIC

Eine Auflistung sämtlicher Varianten für $s = 2$ und $s = 3$ sowie weitere Beispiele für höhere Stufenzahlen s findet man in [13, Abschnitt 343 und 344].
Es lassen sich bei dem einen oder anderen Verfahren gewisse numerische Vorteile erkennen. So sind beim Lobatto-III-Verfahren die erste Zeile und letzte Spalte der Matrix A ausschließlich mit Nullen besetzt. Beim Lösen des im allgemeinen nichtlinearen Gleichungssystems (4.7) sind die erste und letzte Stufe deswegen explizit. Dann stehen $s-2$ implizite Stufen der Konsistenzordnung $2s-2$ gegenüber, womit man sogar effektiver als beim Gauß-Legendre-Verfahren ist! Zusätzlich tritt bei Lobatto-Verfahren der Vorzug auf, daß bei einer iterativen Lösung einer Differentialgleichung jeweils der letzte Knoten eines Gitterabschnitts der erste im nächsten Intervall ist, so daß die zugehörigen numerischen Werte zweimal verwendet werden können.
Gleichwohl können auch andere Gründe die Auswahl eines Verfahrens beeinflussen, zum Beispiel Stabilitätseigenschaften. Bei höherer Stufenzahl s verliert eine geringe Reduzierung der Ordnung zudem an Relevanz.
Abschließend sei noch einmal an die Bemerkung 3.26 erinnert. Die Konsistenzordnungen können nur eintreten, wenn die rechten Seiten eines Anfangswertproblems - die Funktion f - ausreichende Differenzierbarkeitseigenschaften besitzen.

4.7 Stabilität von Runge-Kutta-Verfahren

In den bisherigen Abschnitten des aktuellen Kapitels wurde stets auf eine möglichst hohe Konsistenzordnung der Runge-Kutta-Verfahren hingearbeitet, verbunden mit einer verfügbaren Konstruktionsanleitung. Gegen Ende des letzten Abschnitts wurden noch einfache numerische Betrachtungen angestellt. Ein weiteres wesentliches Kriterium der Verfahren sind deren Stabilitätseigenschaften. Die Wirkung der Runge-Kutta-Verfahren auf steife Differentialgleichungen ist besonders wichtig. Ausgezeichnete Stabilitätseigenschaften sind ein Aushängeschild von

impliziten Runge-Kutta-Verfahren. Gleichwohl existieren zwischen den Varianten teilweise veritable Unterschiede. In diesem Abschnitt werden hierzu ein paar Literaturhinweise gegeben.
Betrachtungen zu diesem Themenkomplex finden sich unter anderem in [12, Abschnitt 35], [18, Kapitel 6], [29, Kapitel IV] und [58, Abschnitt 2.7].
Man kann tendenziell sagen, daß sich mit den Varianten des Abschnitts 4.6 durch ihre Auswahlfreiheiten - ausgehend von der Lage der Knoten c_i - mehr Stabilitätskriterien erfüllen lassen als mit dem Gauß-Legendre-Verfahren. So sind beispielsweise die Gauß-Legendre-, Radau-IA-, Radau-IIA- und Lobatto-IIIC-Verfahren A-stabil, siehe [12, Korollar 352D]. Die drei letzteren Verfahren sind aber noch zusätzlich L-stabil, siehe [12, Theorem 352E]. Einen kleinen Überblick liefern die Tabellen [29, Tabelle 5.13] und [58, Tabelle 6.2.3].

Kapitel 5

Kollokationsverfahren

5.1 Einleitung

In diesem Kapitel wird ein neuer Rechenansatz eingeführt, der (auch) zur Lösung von Anfangswertproblemen eingesetzt wird: die Kollokationsverfahren. Die theoretische Idee des Ansatzes ist intuitiv, alt und auch recht einfach. Die Herleitung der Kollokationsverfahren erfordert deshalb keine wesentlichen neuen Techniken. Es wird sich aber herausstellen, daß ein Teil dieser einfachen Ansätze äquivalent zu bestimmten impliziten Runge-Kutta-Verfahren ist. Deren hervorragende Eigenschaften können damit auf die Kollokationsverfahren übertragen werden. Weitere Modifikationen sind ebenso leicht erarbeitbar. Der einfache Grundgedanke der Kollokationsverfahren macht sie leichter erklärbar und anschaulicher als ihre Gegenstücke, die impliziten Runge-Kutta-Verfahren. Nach gewissen Gesichtspunkten sind sie auch weniger aufwendig. Zusätzlich ermöglichen die Kollokationsverfahren weitere Einblicke in das Lösungsverhalten, da die numerischen Lösungen nun kontinuierlich sind.

Der Abschnitt 5.2 stellt den allgemeinen Grundgedanken der Kollokationsverfahren vor.
Der zentrale Abschnitt 5.3 präsentiert dann die Anwendung der Kollokationsverfahren auf Anfangswertprobleme. Der Zusammenhang mit impliziten Runge-Kutta-Verfahren wird hergestellt und die entscheidenen Merkmale charakterisiert.
Der Abschnitt 5.4 enthält einige Beispiele.
Der Abschnitt 5.5 führt kurz in eine Erweiterung der Aufgabenstellung ein.

Es wurde vor allem [28], [18], [12], [32] und [58] verwendet.

5.2 Problemstellung

Ausgangspunkt eines Kollokationsverfahrens ist die Aufgabe, zu einer gegebenen Funktionalgleichung eine Lösung zu finden:

$$T[x(t)] = T[x](t) = 0, \quad t \in I = [t_0, t_0 + h]. \tag{5.1}$$

T ist ein Operator auf einem geeigneten Funktionenraum, die reellen Zahlen t_0 und t_0+h begrenzen das Intervall. Ein Kollokationsverfahren beruht auf der Idee, die Lösung x durch ein Element p eines endlich-dimensionalen linearen Funktionenraumes zu approximieren. Die Funktionalgleichung (5.1) wird notwendig dahingehend angepaßt, daß ihr nur noch vorgeschrieben wird, auf einer endlichen Teilmenge I_e des Intervalls I erfüllt zu sein:

$$T[p](t) = T[p](t) = 0, \quad t \in I_e \subset I. \tag{5.2}$$

Die endliche Teilmenge I_e ist die Menge der Kollokationspunkte. Hinter diesem allgemeinen Ansatz mit seinen funktionalanalytischen Gedanken und Gebilden steht letztendlich die einfache, alte und häufige Idee, den endlich-dimensionalen Raum der Polynome endlichen Grades zur Approximation zu verwenden. Diese Idee erweist sich trotz ihrer Schlichtheit als fruchtbar.
Bei Anfangswertproblemstellungen ist der Operator T der Gleichung (5.1) durch

$$T[x](\cdot) = \begin{pmatrix} x'(\cdot) - f(\cdot, x(\cdot)) \\ x(t_0) - x_0 \end{pmatrix} \tag{5.3}$$

gegeben. Die Operatorgleichung für das Anfangswertproblem schreibt man dann folgendermaßen zu einem Kollokationsproblem um:

$$T[x](t) = \begin{pmatrix} x'(t) - f(t, x(t)) \\ x(t_0) - x_0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad t \in [t_0, t_0 + h], \tag{5.4}$$

$$\Longrightarrow$$

$$T[p](t) = \begin{pmatrix} p'(t) - f(t, p(t)) \\ p(t_0) - x_0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad t \in \{t_0 + c_i h | \, i = 1, \ldots, s\}. \tag{5.5}$$

Die Zeitpunkte $t_0 + c_i h$ sollten innerhalb des Intervalls $[t_0, t_0 + h]$ liegen. Es gilt also

$$0 \leq c_i \leq 1, \quad i = 1, \ldots, s. \tag{5.6}$$

Diese Voraussetzung ist auch bei Runge-Kutta-Verfahren üblich. Im folgenden Abschnitt wird der Polynomansatz auf ein Anfangswertproblem ausgeführt und entwickelt sowie die Verwandtschaft mit impliziten Runge-Kutta-Verfahren begründet.

5.3 Kollokationsverfahren und implizite Runge-Kutta-Verfahren

In diesem Abschnitt soll durchgehend gelten, daß die c_i nicht nur (5.6) erfüllen, sondern zusätzlich jeweils paarweise verschieden sind. Andernfalls lassen sich die Sätze dieses Abschnittes nicht mehr in der vorliegenden Form schreiben. Im Abschnitt 5.5 wird eine Abweichung von dieser Regel angesprochen. Erneut dient das Anfangswertproblem 3.2 als Aufgabenstellung. Dann erhält man durch die Anwendung der Idee des Abschnitts 5.2:

Definition 5.1 (Kollokationsverfahren). *Es seien $0 \leq c_i \leq 1$, $i = 1, \ldots, s$ reelle Zahlen. Diese seien paarweise verschieden. Das Kollokationsverfahren zum Anfangswertproblem 3.2 im Intervall $[t_0, t_0 + h]$ wird mit einem Polynom p vom Grade s definiert durch*

$$p(t_0) = x_0, \tag{5.7}$$

$$p'(t_0 + c_i h) = f\big(t_0 + c_i h, p(t_0 + c_i h)\big), \quad i = 1, \ldots, s. \tag{5.8}$$

Man erhält den Wert $p(t_0 + h)$ durch Auswertung des Polynoms p bei $t_0 + h$.

Offenbar fixiert das Kollokationsverfahren das Kollokationspolynom p am Intervallbeginn t_0 an den exakten Anfangswert x_0. Im Intervall $[t_0, t_0 + h]$ wird ergänzend versucht, die Steigung des Polynoms p an gewissen Stellen $t_0 + c_i h$ in möglichst exakter Nähe zur tatsächlichen Steigung der Lösung x zu halten. Die Abbildung 5.1 veranschaulicht die Idee graphisch.

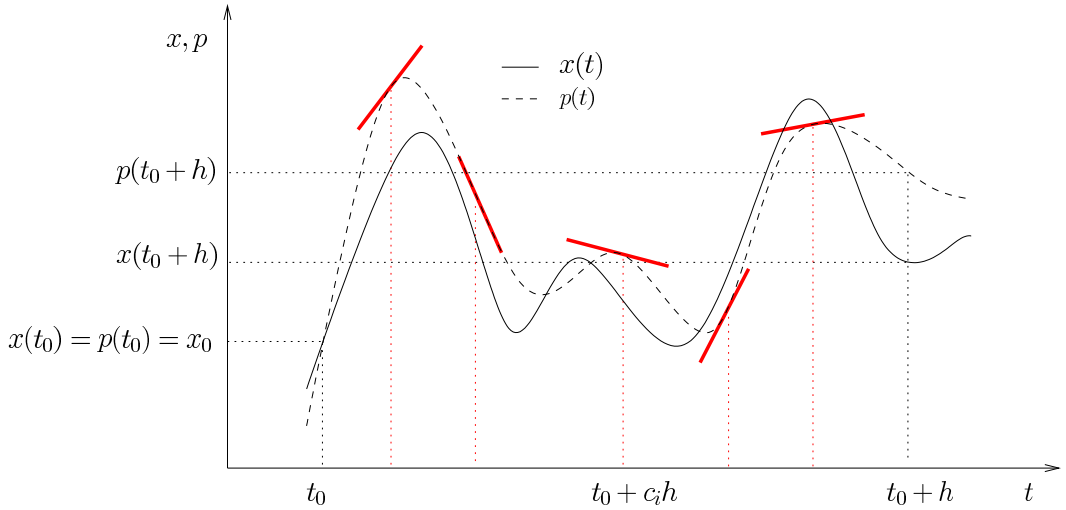

Abbildung 5.1: Idee des Kollokationsverfahrens der Definition 5.1

Die Verwandtschaft mit einem Runge-Kutta-Verfahren der Definition 4.1 schimmert durch. Die Bezeichnung „Steigungen“ der Vektoren k_i wird verständlicher. Mit dem Kollokationspolynom p gewinnt man allerdings eine auf ganz $[t_0, t_0 + h]$ kontinuierliche, sogar stetig differenzierbare, Funktion. Das unterscheidet die Kollokationsverfahren vorerst von den numerischen Einschrittverfahren des Kapitels 3. Einschrittverfahren erzeugen Gitterfunktionen. Das Verhalten des Polynoms p im Intervall $[t_0, t_0 + h]$, und zwar auch jenseits der Stützstellen $t_0 + c_i h$, wird später analysiert.
Ein Polynom p des Grades s ist durch genau $s + 1$ Koeffizienten bestimmt. Die Gleichungen (5.7) und (5.8) summieren sich zu $s + 1$ Bedingungen. Die Existenz und Eindeutigkeit einer Lösung eines Kollokationsverfahrens muß bewiesen werden, denn durch (5.8) entsteht im allgemeinen ein nichtlineares Gleichungssystem.

Satz 5.2. *Die rechte Seite f eines Anfangswertproblems genüge einer Lipschitz-Bedingung L. Die Schrittweite h sei ausreichend klein. Dann existiert eine eindeutige Lösung des Gleichungssystems aus Definition 5.1.*

Beweis. Siehe [12, Theorem 346A]. □

Der Beweis ist völlig deckungsgleich mit dem Existenz- und Eindeutigkeitssatz für implizite Runge-Kutta-Verfahren 4.4. Dafür ist eine Äquivalenz beider Ansätze verantwortlich.
Der folgende Satz eröffnet eine Reihe von Sätzen, an derem Ende die Äquivalenz von gewissen Kollokationsverfahren zu bestimmten impliziten Runge-Kutta-Verfahren bewiesen sein wird. Der Satz 5.3 legt dar, was für ein implizites Runge-Kutta-Verfahren durch ein Kollokationsverfahren mit vorgegebenen Vektor $c = (c_1, \ldots, c_s)^T$ impliziert wird.

Satz 5.3. *Das Kollokationsverfahren aus Definition 5.1 entspricht einem s-stufigen impliziten Runge-Kutta-Verfahren mit den durch folgende Operationen definierten Koeffizienten:*

$$a_{ij} = \int_0^{c_i} l_j(t)\,dt, \quad i, j = 1, \ldots, s, \tag{5.9}$$

$$b_j = \int_0^1 l_j(t)\,dt, \quad j = 1, \ldots, s. \tag{5.10}$$

Die l_j sind die Lagrange-Polynome der Definition 2.9:

$$l_j(t) := \prod_{\substack{k=1 \\ k \neq j}}^{s} \frac{t - c_k}{c_j - c_k}, \quad j = 1, \ldots, s. \tag{5.11}$$

Beweis. Mit einer Interpolationsdarstellung von p' im Sinne der Folgerung 2.33 ergibt sich zunächst

$$\begin{aligned} p'(t_0 + c_j h) &:= k_j \\ \Longrightarrow \quad p'(t_0 + th) &= \sum_{j=1}^{s} k_j \cdot l_j(t). \end{aligned}$$

Nach Integration über das Intervall $[0, c_i]$ erhält man:

$$\frac{1}{h}\Big(p(t_0 + c_i h) - \overbrace{p(t_0)}^{=x_0}\Big) = \int_0^{c_i} \sum_{j=1}^{s} k_j \cdot l_j(t)\, dt = \sum_{j=1}^{s} k_j \cdot \int_0^{c_i} l_j(t)\, dt = \sum_{j=1}^{s} a_{ij} k_j.$$

Nun löst man nach $p(t_0 + c_i h)$ auf und setzt den entstehenden Term in Gleichung (5.8) ein. Es ergibt sich das Runge-Kutta-Gleichungssystem der Definition 4.1. Integriert man über das Intervall $[0, 1]$, ergibt sich:

$$\frac{1}{h}\Big(p(t_0 + h) - \overbrace{p(t_0)}^{=x_0}\Big) = \int_0^{1} \sum_{j=1}^{s} k_j \cdot l_j(t)\, dt = \sum_{j=1}^{s} k_j \cdot \int_0^{1} l_j(t)\, dt = \sum_{j=1}^{s} b_j k_j.$$

Aufgelöst nach $p(t_0+h)$ erhält man die Einschrittverfahrensrekursionsformel (4.8) eines Runge-Kutta-Verfahrens. □

Angemerkt sei im nachhinein, daß die paarweise Verschiedenheit der c_i für die Formulierung der Lagrange-Polynome unabdingbar ist. Außerdem ist strenggenommen noch nicht sicher, ob das entstandene Runge-Kutta-Verfahren wirklich implizit ist.
Ein Kollokationsverfahren für ein Anfangswertproblem ist allein durch den Vektor $c = (c_1, \ldots, c_s)^T$ festgelegt und in seinen Eigenschaften determiniert. Bei einem impliziten Runge-Kutta-Verfahren werden neben dem Vektor c auch die Matrix A und der Vektor b durch die Vereinfachenden Bedingungen der Definition 4.14 bestimmt. Es gilt also:

Bemerkung 5.4.

$$\begin{aligned} \textit{impl. Runge-Kutta-Verfahren} \quad &\Longrightarrow \quad s^2 + 2s \ \textit{Parameter sind zu wählen,} \\ \textit{Kollokationsverfahren} \quad &\Longrightarrow \quad s \ \textit{Parameter sind zu wählen.} \end{aligned}$$

Offenbar hat ein Kollokationsverfahren viel weniger Freiheitsgrade als ein implizites Runge-Kutta-Verfahren. Es sollte deswegen so sein, daß ein durch Kollokation mittels Satz 5.3 erzeugtes implizites Runge-Kutta-Verfahren bereits gewisse Bedingungen automatisch erfüllt, die Konsistenzordnungen sicherstellen. Der nächste Satz beschäftigt sich mit dieser Frage.

Satz 5.5. *Die Koeffizienten c_i, b_i und a_{ij} eines impliziten Runge-Kutta-Verfahrens, welches durch ein Kollokationsverfahren der Definition 5.1 mittels Satz 5.3 erzeugt wurde, erfüllen*

$$\begin{aligned} A(s): \quad & \sum_{j=1}^{s} a_{ij} c_j^{k-1} &&= \frac{1}{k} c_i^k, \quad i,k = 1,\dots,s, \\ B(s): \quad & \sum_{i=1}^{s} b_i c_i^{k-1} &&= \frac{1}{k}, \quad k = 1,\dots,s. \end{aligned}$$

Beweis. Unter Zuhilfenahme der Folgerung 2.33 ergibt sich $A(s)$:

$$\begin{aligned} \sum_{j=1}^{s} a_{ij} c_j^{k-1} &\overset{(5.9)}{=} \sum_{j=1}^{s} c_j^{k-1} \int_0^{c_i} l_j(t)\,dt = \int_0^{c_i} \Big(\sum_{j=1}^{s} c_j^{k-1} l_j(t) \Big) dt \\ &\overset{Folg.2.33}{=} \int_0^{c_i} t^{k-1}\,dt = \frac{c_i^k}{k}, \quad k = 1,\dots,s. \end{aligned}$$

Mit der gleichen Beweisführung erhält man $B(s)$:

$$\begin{aligned} \sum_{j=1}^{s} b_j c_j^{k-1} &\overset{(5.10)}{=} \sum_{j=1}^{s} c_j^{k-1} \int_0^{1} l_j(t)\,dt = \int_0^{1} \Big(\sum_{j=1}^{s} c_j^{k-1} l_j(t) \Big) dt \\ &\overset{Folg.2.33}{=} \int_0^{1} t^{k-1}\,dt = \frac{1}{k}, \quad k = 1,\dots,s. \end{aligned}$$

Siehe auch [18, Lemma 6.37]. □

Für $k = 1$ erhält man $\sum_{j=1}^{s} b_j = 1$ und $\sum_{j=1}^{s} a_{ij} = c_i$, also die Bedingungen für die Konsistenz und Invarianz bezüglich Autonomisierung einer Differentialgleichung aus den Sätzen 4.6 und 4.7.
Zwei zentrale Bedingungen der impliziten Runge-Kutta-Verfahren sind damit erfüllt, die später Konsistenzordnungen sicherstellen werden (vgl. Abschnitt 4.5). Blättert man zurück zum Satz 4.22, so kann man bereits die Bildung eines zum Gauß-Legendre-Verfahren äquivalenten Kollokationsverfahren erkennen. Doch zunächst wird allgemeiner fortgefahren:

Satz 5.6. *Ein s-stufiges implizites Runge-Kutta-Verfahren mit paarweise verschiedenen Knoten $c_1,\dots,c_s$ und einer Ordnung $p \geq s$ ist genau dann ein Kollokationsverfahren, wenn die Bedingung $A(s)$ gilt.*

Beweis. Der Beweis verwendet wiederum die Techniken des Satzes 5.5. Es sei P ein Polynom mit $\operatorname{Grad} P \leq s-1$. Es sei zuerst ein implizites Runge-Kutta-Verfahren mit der Bedingung $A(s)$ gegeben. Diese Bedingung läßt sich wie in

Bemerkung 4.16 erweitern zu

$$\sum_{j=1}^{s} a_{ij} P(c_j) = \int_0^{c_i} P(t)\,dt.$$

Mit der Folgerung 2.33 gilt dann

$$\int_0^{c_i} P(t)\,dt = \int_0^{c_i} \sum_{j=1}^{s} P(c_j) l_j(t)\,dt = \sum_{j=1}^{s} P(c_j) \int_0^{c_i} l_j(t)\,dt.$$

Da die a_{ij} durch die Bedingung $A(s)$ eindeutig bestimmt sind, folgt nun die Formel (5.9) für ein Kollokationsverfahren. Die Formel (5.10) erhält man analog, wobei man beim Integrieren lediglich $[0, c_i]$ durch $[0, 1]$ zu ersetzen hat.
Es sei nun umgekehrt ein Kollokationsverfahren gegeben. Dann sind nach Satz 5.3 die a_{ij} des impliziten Runge-Kutta-Verfahren gegeben durch

$$a_{ij} = \int_0^{c_i} l_j(t)\,dt, \quad i, j = 1, \ldots, s.$$

Nun multipliziert man beide Seiten mit $P(c_j)$ und summiert dann über j. Es folgt

$$\sum_{j=1}^{s} a_{ij} P(c_j) = \int_0^{c_i} \Big(\sum_{j=1}^{s} l_j(t) P(c_j) \Big) dt.$$

Auf die rechts Seite wendet man erneut die Folgerung 2.33 an. Beim Vergleich der linken und rechten Seite für verschiedene Polynome P ergibt sich die Bedingung $A(s)$, siehe auch [57, Theorem 2.5.12]. □

Der folgende Satz 5.7 wird als Hilfsresultat für den Satz 5.8 gebraucht, ist aber auch sonst interessant. Er beschreibt die Approximation des Polynoms p an die eigentliche Lösung x des Anfangswertproblems, wobei auch Aussagen über die Approximation von Ableitungen gemacht werden. Im Unterschied zu Einschrittverfahren werden so nicht nur punktweise Vergleiche möglich, sondern über das ganze Intervall $[t_0, t_0 + h]$.

Satz 5.7 (Hulme 1972). *Gegeben sei das Kollokationsverfahren der Definition 5.1. Die rechte Seite f der Differentialgleichung sei hinreichend glatt, die Schrittweite h hinreichend klein. Dann gibt es eine positive Konstante C, so daß gilt:*

$$\left\| x^{(k)}(t) - p^{(k)}(t) \right\| \leq C \cdot h^{s+1-k}, \quad t \in [t_0, t_0 + h], \quad k = 0, \ldots, s. \tag{5.12}$$

Beweis. Es sei $t \in [0,1]$. Für die exakte Lösung x und das Kollokationspolynom p gelten mit Taylor-Entwicklung und der Interpolationsformel der Folgerung 2.33:

$$\begin{aligned} x'(t_0+th) &= \sum_{i=1}^{s} f\big(t_0+c_ih, x(t_0+c_ih)\big) l_i(t) + h^s R(t,h), \\ p'(t_0+th) &= \sum_{i=1}^{s} f\big(t_0+c_ih, p(t_0+c_ih)\big) l_i(t). \end{aligned}$$

Hierbei ist der Restterm $R(t,h)$ beschränkt. Später werden noch zusätzliche Differenzierbarkeitseigenschaften vorausgesetzt. Für p' ist die Interpolation exakt und ein etwaiger Restterm entfällt. Nun wird integriert:

$$\begin{aligned} \frac{1}{h}\big(x(t_0+th)-x_0\big) &= \sum_{i=1}^{s} f(t_0+c_ih, x(t_0+c_ih)) \int_0^t l_i(\tau)\,d\tau + h^s \int_0^t R(\tau,h)\,d\tau, \\ \frac{1}{h}\big(p(t_0+th)-p_0\big) &= \sum_{i=1}^{s} f(t_0+c_ih, p(t_0+c_ih)) \int_0^t l_i(\tau)\,d\tau. \end{aligned}$$

Wegen $x_0 = p_0$ erhält man bei Subtraktion der beiden obigen Gleichungen und Multiplikation mit h:

$$\begin{aligned} & x(t_0+th) - p(t_0+th) && (5.13) \\ = \; & h \sum_{i=1}^{s} \Big(f\big(t_0+c_ih, x(t_0+c_ih)\big) - f\big(t_0+c_ih, p(t_0+c_ih)\big) \Big) \int_0^t l_i(\tau)\,d\tau \\ + \; & h^{s+1} \int_0^t R(\tau,h)\,d\tau. && (5.14) \end{aligned}$$

In [28, Lemma 7.5] wird gezeigt, daß

$$f\big(t_0+c_ih, x(t_0+c_ih)\big) - f\big(t_0+c_ih, p(t_0+c_ih)\big) = \mathcal{O}(h^{s+1})$$

ist. Damit hat man (5.12) für $k=0$ gezeigt, wenn man über die Terme (5.13) und (5.14) die Maximumsnorm legt. Für die höheren Ableitungen müssen die Gleichungsseiten (5.13) und (5.14) entsprechend oft differenziert werden. Dabei werden mehr Voraussetzungen an $R(t,h)$ gestellt, die aber bei ausreichender Differenzierbarkeit von f erfüllt sind.
In [18, Lemma 6.41] wird [28, Lemma 7.5] nicht verwendet. Stattdessen werden für das weitere Vorgehen die folgenden Abkürzungen eingeführt:

$$C_0 := \max_{0\le t\le 1} \sum_{i=1}^{s} \left| \int_0^t l_i(\tau)\,d\tau \right|, \tag{5.15}$$

$$C_k := \max_{0\le t\le 1} \sum_{i=1}^{s} \left| l_i^{(k-1)}(t) \right|, \quad k = 1,\dots,s, \tag{5.16}$$

$$M := \max_{t_0\le t\le t_0+h} \|x(t)-p(t)\|. \tag{5.17}$$

Mit der Lipschitzkonstante L, der Dreiecksungleichung und der Maximumsnorm ergibt sich dann aus den beiden Gleichungsseiten (5.13) und (5.14):

$$M \leq h \cdot L \cdot C_0 \cdot M + h^{s+1} \max_{t_0 \leq t \leq t_0+h} \int_0^t R(\tau, h)\, d\tau \implies M \leq C \cdot \mathcal{O}(h^{s+1}).$$

Für die letzte Umformung ist hierbei ein genügend kleines h erforderlich. Für die Ableitungen werden auch hier die zusätzlichen Differenzierbarkeitseigenschaften von $R(t, h)$ benutzt. Nach Differenzieren werden die Terme (5.16) zur Abschätzung verwendet. Auf der linken Seite (5.13) ensteht mit jeder Ableitung ein h als weiterer Vorfaktor, was die Ordnung jeweils um Eins reduziert. Siehe [28, Theorem 7.10] und [18, Lemma 6.41]. □

Entscheidend für die Konsistenzordnung eines impliziten Runge-Kutta-Verfahrens ist die Lage des Stützstellenvektors c. Dieser wird günstigerweise durch die Nullstellen der Legendre-Polynomen oder Abbarten bestimmt. Drückt man dies mit Darstellung

$$L_{s,\lambda_1,\lambda_2,\ldots,\lambda_m}(2t-1) = L_s(2t-1) + \sum_{i=1}^{m} \lambda_i L_{s-i}(2t-1) \tag{5.18}$$

aus, so nimmt die Konsistenzordnung ab, je größer m ist (bei $\lambda_m \neq 0$). Dann ist $L_{s,\lambda_1,\lambda_2,\ldots,\lambda_m}$ wegen der Linearität des Skalarprodukts nur noch orthogonal zu Polynomen des Grades $s-m-1$. Die grundlegenden Zusammenhänge zwischen impliziten Runge-Kutta-Verfahren und Kollokationsverfahren sind ausgeführt. Nun beweist man den Hauptsatz des Kapitels. Die Verknüpfung zur Gauß-Quadratur des Abschnitts 2.6 wird wieder sehr deutlich. Den Satz findet man in [18, Satz 6.40] und [28, Theorem 7.9].

Satz 5.8. *Ein durch Kollokation erzeugtes s-stufiges Runge-Kutta-Verfahren besitzt die Konsistenzordnung p für p-mal stetig differenzierbare rechte Seiten f genau dann, wenn die durch die Stützstellen $c = (c_1, \ldots, c_s)^T$ und Gewichte $b = (b_1, \ldots, b_s)^T$ gegebene Quadraturformel für p-fach stetig differenzierbare Funktionen die (Fehlerterm-)Ordnung $p+1$ gemäß Satz 2.20 besitzt (vergleiche auch Bemerkung 3.27).*

Beweis. Quadraturprobleme sind spezielle Anfangswertprobleme und somit folgt die Ordnung der Quadraturformel aus der des Runge-Kutta-Verfahrens.
Für die kompliziertere Beweisrichtung wird die ausreichende Differenzierbarkeit von f wie immer vorausgesetzt. Für die entscheidende Beweisidee sorgt Satz 3.28. Das Anfangswertproblem

$$x'(t) = f(t, x(t)), \quad x(t_0) = x_0$$

besitzt die Lösung x. Das Polynom p sei die Lösung des Anfangswertproblems mit einer Störungsfunktion δf:

$$p'(t) = f(t, p(t)) + \delta f(t, p(t)), \quad p(t_0) = x_0.$$

Als Störung wird passenderweise

$$\delta f(t, p(t)) := p'(t) - f(t, p(t))$$

definiert. Für den Fehler $x - p$ gilt nach Satz 3.28 nun:

$$x(t_0 + h) - p(t_0 + h) = \int_{t_0}^{t_0+h} M(t_0 + h, s)\delta f(s, p(s))\, ds.$$

Die rechte Seite wird nun mit der Quadraturregel abgeschätzt:

$$\begin{aligned} & \int_{t_0}^{t_0+h} M(t_0 + h, s)\delta f(s, p(s))\, ds \\ = \; & h \sum_{i=1}^{s} M(t_0 + h, t_0 + c_i h)\delta f\big(t_0 + c_i h, p(t_0 + c_i h)\big) + \mathcal{O}(h^{p+1}). \end{aligned}$$

Da bei einem Kollokationsverfahren stets $\delta f(t_0 + c_i h, p(t_0 + c_i h)) = 0$ ist, verschwindet die Summe. Es bleibt zu zeigen, daß der Restterm das Verhalten $\mathcal{O}(h^{p+1})$ besitzt. Dazu muß man die Ableitungen von $M(t_0 + h, s)\delta f(s, p(s))$ studieren. Die Matrixabbildung M ist unproblematisch, da die Differenzierbarkeit von f als ausreichend vorausgesetzt wurde. Nach Satz 5.7 bleiben die Ableitungen von p für hinreichend kleine h gleichmäßig beschränkt. Siehe auch [18, Satz 6.40] und [28, Theorem 7.9]. □

Daß die Kollokationsverfahren aus Satz 5.3 letztendlich wirklich auf implizite Runge-Kutta-Verfahren führen, stellt die folgende Bemerkung fest.

Bemerkung 5.9. *Der Satz 4.8 begrenzt die maximale Konsistenzordnung von expliziten s-stufigen Runge-Kutta-Verfahren auf s. Der Satz 5.8 fokussiert die Konsistenzordnungen $\geq s$. Damit entsprechen die wesentlichen Kollokationsverfahren des Satzes 5.3 zwingend impliziten Runge-Kutta-Verfahren.*

Vor der nächsten Bemerkung sei auch hier auf die einprägsame Bemerkung 3.27 hingewiesen, in der die Bezeichnungen Exaktheitsgrad und Fehlertermabschätzung von Integrationsregeln mit dem Begriff der Konsistenzordnungen von Anfangswertproblemen in Verbindung gesetzt wurde.

Bemerkung 5.10. *Da aus Satz 2.15 bekannt ist, daß eine Quadraturformel den Exaktheitsgrad $2s - 1$ nicht überschreitet und Satz 2.16 aussagt, daß diese Exaktheit auch erreicht werden kann, ist der Satz 5.8 ein Beweis dafür, daß ein implizites Runge-Kutta-Verfahren maximal die Konsistenzordnung $2s$ besitzt.*

Aus Satz 5.7 schlußfolgert man ferner:

Bemerkung 5.11. *Bei der Konsistenzordnung $p > s$ ist die Approximation des Kollokationspolynoms zum Zeitpunkt $t_0 + h$ besser als im Rest des Intervalls $[t_0, t_0 + h]$.*

5.4 Beispiele

Mit dem Satz 5.6 konstruiert man Kollokationsverfahren hoher Konsistenzordnung. Die Abschnitte 4.5 und 4.6 behandeln implizite Runge-Kutta-Verfahren, die per Konstruktion die Bedingung $A(s)$ erfüllen.

Folgerung 5.12. *Neben der klassischen Gauß-Legendre-Methode sind die Radau-I-, Radau-IIA- und Lobatto-IIIA-Verfahren Kollokationsverfahren.*

Das Beispiel 5.13 enthält Butcher-Tableaus für $s = 2$.

Beispiel 5.13. *Bei $s = 2$ entspricht ein Kollokationsverfahren der Approximation der Lösung der Differentialgleichung durch ein quadratisches Polynom. Die in*

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \qquad \begin{array}{c|cc} 0 & 0 & 0 \\ \frac{2}{3} & \frac{1}{3} & \frac{1}{3} \\ \hline & \frac{1}{4} & \frac{3}{4} \end{array} \qquad \begin{array}{c|cc} \frac{1}{3} & \frac{5}{12} & -\frac{1}{12} \\ 1 & \frac{3}{4} & \frac{1}{4} \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array} \qquad \begin{array}{c|cc} \frac{3-\sqrt{3}}{6} & \frac{1}{4} & \frac{3-2\sqrt{3}}{12} \\ \frac{3+\sqrt{3}}{6} & \frac{3+2\sqrt{3}}{12} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Abbildung 5.2: v.l.n.r.: Lobatto-IIIA, Radau-I, Radau-IIA und Gauß

der Abbildung 5.2 angegebenen Verfahren sind gemäß der Folgerung 5.12 Kollokationsverfahren. Die Verfahren, die durch die Vektoren $c^T = (c_1, c_2)$ der ersten Spalte der Tableaus definiert sind, erreichen Konsistenzordnungen zwei, drei und vier.

Für $s = 3$ kann man berechnen:

Beispiel 5.14. *Bei $s = 3$ entspricht ein Kollokationsverfahren der Approximation der Lösung der Differentialgleichung durch ein kubisches Polynom. Die Verfahren der Abbildung 5.3 besitzen die Konsistenzordnungen vier und sechs.*

Weitere Beispiele finden sich in [13, Abschnitt 344]. Andererseits gilt:

Beispiel 5.15. *Die Verfahren des Beispiels 4.29 sind keine Kollokationsverfahren, da sie die Bedingung $A(s)$ nicht erfüllen.*

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{5}{24} & \frac{1}{3} & \frac{-1}{24} \\ 1 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array} \qquad \begin{array}{c|ccc} \frac{5-\sqrt{15}}{10} & \frac{5}{36} & \frac{10-3\sqrt{15}}{45} & \frac{25-6\sqrt{15}}{180} \\ \frac{1}{2} & \frac{10+3\sqrt{15}}{72} & \frac{2}{9} & \frac{10-3\sqrt{15}}{72} \\ \frac{5+\sqrt{15}}{10} & \frac{25+6\sqrt{15}}{180} & \frac{10+3\sqrt{15}}{45} & \frac{5}{36} \\ \hline & \frac{5}{18} & \frac{4}{9} & \frac{5}{18} \end{array}$$

Abbildung 5.3: v.l.n.r.: Lobatto-IIIA und Gauß für $s = 3$

$$\begin{array}{c|cc} \frac{1}{3} & \frac{1}{3} & 0 \\ 1 & 1 & 0 \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array}$$

Abbildung 5.4: Radau-II für $s = 2$

Beweis. Die Behauptungen lassen sich wie im Satz 4.28 mit Implikationsketten vorführen, basierend auf den Beziehungen des Satzes 4.20. Beispielhaft kann man für das Radau-II-Verfahrens des Beispiels 4.29 mit $s = 2$ verifizieren:

$$A(2): \quad \sum_{j=1}^{2} a_{ij} c_j^{k-1} = \frac{1}{k} c_i^k, \quad i,\, k = 1, 2.$$

Für $k = 1$ gilt offenbar:

$$\begin{aligned} i = 1: \qquad & \frac{1}{3} \cdot 1 + 0 \cdot 1 = \frac{1}{3} \stackrel{!}{=} \frac{1}{1} \cdot (\frac{1}{3})^1 \ \surd, \\ i = 2: \qquad & 1 \cdot 1 + 0 \cdot 1 = 1 \stackrel{!}{=} \frac{1}{1} \cdot 1^1 \ \surd. \end{aligned}$$

Dagegen erhält man bei $k = 2$:

$$\begin{aligned} i = 1: \qquad & \frac{1}{3} \cdot \frac{1}{3} + 0 \cdot 1 = \frac{1}{9} \neq \frac{1}{2} \cdot (\frac{1}{3})^1, \\ i = 2: \qquad & 1 \cdot \frac{1}{3} + 0 \cdot 1 = \frac{1}{3} \neq \frac{1}{2} \cdot 1^1. \end{aligned}$$

$A(s) \stackrel{hier}{=} A(2)$ ist also nicht erfüllt. Dies entspricht den Erwartungen. Im Satz 4.28 wurde gezeigt, daß für Radau-II-Verfahren im allgemeinen nur $A(s-1) \stackrel{hier}{=} A(1)$ gültig ist. □

5.5 Erweiterungen

Bisher wurde die paarweise Verschiedenheit der c_i betont. Dies ist - wie schon angedeutet - nicht notwendig. Allerdings erfordert eine Abweichung eine Anpassung des Abschnitts 5.3. Diese Veränderungen werden in diesem Abschnitt angerissen. Die Hauptschwierigkeit besteht nicht in der Formulierung des neuen Kollokationsverfahrens, sondern im Auffinden eines entsprechenden impliziten Runge-Kutta-Verfahrens. Hierbei wird wie in [28, Abschnitt II.13] vorgegangen. Offenbar genügt die Definition 5.1 nicht mehr den Anforderungen, da bei mehrfach identischen c_i die Anzahl der Gleichungen nicht mehr ausreicht, um das Polynom p eindeutig zu bestimmen. Die Lösung besteht im Aufstellen von Forderungen an höhere Ableitungen von p bei derartigen mehrfachen c_i. Dies ist technisch aufwendiger, da damit auch die rechte Seite f mehrfach abgeleitet werden muß. Die mehrdimensionale Kettenregel wird als Hilfsmittel benötigt. In [28, Abschnitt II.13] wird die Schreibweise hierfür spezialisiert. In diesem Abschnitt soll $\frac{d}{dt}$ die mehrdimensionale Kettenregel andeuten.

Definition 5.16 (Modifiziertes Kollokationsverfahren). *Es seien c_i, $i = 1, \ldots, s$ paarweise verschiedene Knoten. Die c_i haben jeweils die Vielfachheit q_i. Dann erhält man ein Kollokationsverfahren mit mehrfachen Knoten durch*

$$\begin{aligned} p(t_0) &= x_0, \\ p^{(l_i)}(t_0 + c_i h) &= \frac{d^{l_i-1}}{dt^{l_i-1}}\Big[f\big(t_0 + c_i h, p(t_0 + c_i h)\big)\Big], \\ & \qquad i = 1, \ldots, s, \quad l_i = 1, \ldots, q_i. \end{aligned}$$

Man erhält den Wert $p(t_0 + h)$ durch Auswertung des Polynoms p bei $t_0 + h$.

Das Polynom p hat demnach den Grad $\sum_{j=1}^{s} q_j$.

Bemerkung 5.17. *Für $q_i = 1$ und $i = 1, \ldots, s$ entsteht das Kollokationsverfahrens der Definition 5.1.*

Die natürliche Erweiterung der Lagrange-Polynomen sind die `Hermite-Polynome`. Sie sollen hier nicht weiter erläutert werden, auch in [28, Abschnitt II.13] werden sie nur angedeutet. Hermite-Polynome erfüllen die folgende Eigenschaft:

Definition 5.18. *Die Hermite-Polynome $l_{jr}^{(k)}$ erfüllen*

$$l_{jr}^{(k)}(c_i) = \begin{cases} r!, & i = j \text{ und } k = r - 1, \\ 0, & \text{sonst.} \end{cases}$$

Man erkennt, daß sich ein Polynom g des Grades $\sum_{i=1}^{s} q_i - 1$ dann durch

$$g(t) = \sum_{j=1}^{s} \sum_{r=1}^{q_j} \frac{1}{r!} l_{jr}(t) g^{(r-1)}(c_j) \tag{5.19}$$

schreiben läßt. Dies verifiziert man, indem man links und rechts bis zu q_j-mal ableitet und dann beidseitig c_i einsetzt. Man kann die Polynome $l_{jr}^{(k)}$ also durchaus als Basis des Polynomraumes ansprechen.
Es ist nicht ganz leicht, die Äquivalenz der neuen Kollokationsverfahren zu gewissen Runge-Kutta-Verfahren zu zeigen, da die entsprechenden Runge-Kutta-Verfahren völlig neu Struktur strukturiert sind. Deswegen werden zunächst diese sogenannten „q-derivative Runge-Kutta methods“ definiert, wobei auch hier die vereinfachten Ableitungssymbole ausreichen sollen.

Definition 5.19 (q-derivative Runge-Kutta methods). *Es seien $a_{ij}^{(r)}$ und $b_j^{(r)}$ mit $i, j = 1, \ldots, s$ und $r = 1, \ldots, q$ reelle Koeffizienten. Das Verfahren*

$$k_i^{(l)} = \frac{h^l}{l!} \cdot \frac{d^{l-1}}{dt^{l-1}} \Big[f\big(t_0 + c_i h, \eta_0 + \sum_{r=1}^{q} \sum_{j=1}^{s} a_{ij}^{(r)} k_j^{(r)}\big) \Big], \tag{5.20}$$

$$\eta_1 = \eta_0 + \sum_{j=1}^{s} \sum_{r=1}^{q} b_j^{(r)} k_j^{(r)}, \tag{5.21}$$

$$l = 1, \ldots, q, \quad i = 1, \ldots, s$$

ist ein s-stufiges Runge-Kutta-Verfahren mit Ableitungen bis zur Ordnung q.

Auffällig ist zuerst die Absenz der Schrittweite h vor den Summen in (5.20) und (5.21). Um ein Runge-Kutta-Verfahren der Definition 4.1 in diese Definition 5.19 einzuordnen, bedarf es deshalb einer kleinen Detailänderung. Man setzt

$$q := 1 \quad \text{und} \quad \tilde{k_i} := \frac{k_i^{(1)}}{h} \tag{5.22}$$

und ersetzt damit k_i in (5.20) und (5.21). Schreibt man dann wieder k_i für $\tilde{k_i}$, ergibt sich das Runge-Kutta-Verfahren der Definition 4.1.
Die Verfahren der Definition 5.19 lassen sich nicht mehr mit einem Butcher-Tableaus aufschreiben. Es müssen mehrere separate Matrizen verwendet werden.

Beispiel 5.20. *Das Beispiel stammt aus [28, Abschnitt II.13]. Es sei $s = 2$ und $q = 3$. Das gemäß Definition 5.19 gebildete Verfahren der Abbildung 5.5 hat die Ordnung* 8.

0.1854	0.2019	−0.0165		0.1854	−0.0223	0.0087
0.8146	0.5165	0.2981		0.8146	0.0568	−0.0705
	0.5000	0.5000			0.0241	−0.0241

0.1854	0.0117	−0.0022
0.8146	0.0241	0.0103
	0.0037	0.0037

Abbildung 5.5: Alle Zahlen auf vier Nachkommastellen gerundet

Offenbar lassen sich nur bestimmte Kollokationsverfahren der Definition 5.16 mit den Methoden der Definition 5.19 verbinden (für $q_1 = \ldots = q_s = q$). Eine Äquivalenzaussage der eingeführten Kollokationsverfahren mit den q-derivative Runge-Kutta methods liefert der nächste Satz.

Satz 5.21. *Ein Kollokationsverfahren der Definition 5.16 entspricht einem Runge-Kutta-Verfahren der Definition 5.19 mit den folgenden Koeffizienten:*

$$a_{ij}^{(r)} = \int_0^{c_i} l_{jr}(t)\,dt, \quad i,j = 1,\ldots,s, \;\; r = 1,\ldots,q, \tag{5.23}$$

$$b_j^{(r)} = \int_0^1 l_{jr}(t)\,dt, \quad j = 1,\ldots,s, \;\; r = 1,\ldots,q. \tag{5.24}$$

Beweis. Wie bei Satz 5.3, siehe auch [28, Theorem 13.2]. □

Kapitel 6

Optimalsteuerungsprobleme

6.1 Einleitung

Bis jetzt standen in der Diplomarbeit Anfangswertprobleme im Vordergrund. Es wurden implizite Runge-Kutta-Verfahren und Kollokationsverfahren in den Kapiteln 4 und 5 als numerische Lösungsvarianten vorgeschlagen.
In diesem Kapitel beginnt mit Optimalsteuerungsproblemen ein neuer Abschnitt der Diplomarbeit. Kapitel 6 führt in die Theorie dieser Problemklasse ein, während im Kapitel 7 versucht wird, die impliziten Runge-Kutta- und Kollokationsverfahren für numerische Lösungsmethoden nutzbar zu machen.

Optimalsteuerungsprobleme unterscheiden sich emminent. Der Abschnitt 6.2 stellt mehrere Aufgabentypen vor und vergleicht diese miteinander.
Der Abschnitt 6.3 liefert einen kurzen Einblick in das Minimumprinzip.
Der Abschnitt 6.4 behandelt Linear-Quadratische Optimalsteuerungsprobleme. Diese Problemklasse steht ab Kapitel 7 im Mittelpunkt des Interesses.
Im Abschnitt 6.5 erfolgt eine vorbereitende Einführung in direkte Lösungsverfahren für Optimalsteuerungsprobleme.

Die Schreibweise richtet sich unter anderem nach [25]. Für theoretische Erklärungen sind zum Beispiel [2] und [16] nützlich. Beispiele kann man in [54] und [15] finden.

6.2 Allgemeine Problemstellung

Kontinuierliche oder auch diskrete Optimalsteuerungsprobleme gehören zu den typischen Aufgabenstellungen der Optimierungstheorie und damit auch der numerischen Mathematik. Das folgende einführende Beispiel verdeutlicht die An-

wendungsmöglichkeiten von Optimalsteuerungsproblemen in der Ökonomie.

Beispiel 6.1 (Ressourcenverteilungsproblem). *Die folgenden Größen haben die Bedeutung:*

$$\begin{aligned}
x(t) &: \text{ Produktion eines Gutes}: \ x(t) \geq 0,\\
x_0 &: \text{ Produktion zu Beginn (vorgegeben)}: \ x(0) = x_0 > 0,\\
u(t) &: \text{ Prozentsatz von } x(t), \text{ der für Investitionen verwendet wird}:\\
&\quad 0 \leq u(t) \leq 1,\\
1 - u(t) &: \text{ Konsumrate von } x(t): \ 0 \leq u(t) \leq 1,\\
r &: \text{ Diskontrate (vorgegeben)}: \ 0 \leq r < 1,\\
t_0,\, t_f &: \text{ Zeitintervall (vorgegeben)}: \ t_0 < t_f.
\end{aligned}$$

Maximiere

$$\int_{t_0}^{t_f} e^{-rt}\big(1 - u(t)\big)x(t)\,dt$$

unter den Nebenbedingungen

$$\begin{aligned}
&\dot{x}(t) = x(t)u(t) \quad \text{für fast alle } t \in [t_0, t_f],\\
&x(t_0) = x_0,\\
&0 \leq u(t) \leq 1, \quad t \in [t_0, t_f].
\end{aligned}$$

Die Investitionssteuerung u beeinflußt über die Differentialgleichung die Produktion x des Gutes. Typisch ist des weiteren die Maximierung oder Minimierung eines Funktionals. Darüberhinaus unterliegen die unbekannten Funktionen x und u je nach Gegebenheit noch weiteren Restriktionen, so den obigen, weitverbreiteten „Box-Constraints“ für u.
Die erste Definition des Abschnitts formalisiert die Aufgabenstellung nach [25, 7.2.1]. Vorbereitend werden folgende Angaben gemacht:

$$\begin{aligned}
f_a : \mathbb{R}^n &\longrightarrow \mathbb{R},\\
f_b : \mathbb{R}^n &\longrightarrow \mathbb{R},\\
f : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m &\longrightarrow \mathbb{R},\\
g : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m &\longrightarrow \mathbb{R}^n,\\
\alpha : \mathbb{R} \times \mathbb{R}^n &\longrightarrow \mathbb{R}^{s_a},\\
\beta : \mathbb{R} \times \mathbb{R}^n &\longrightarrow \mathbb{R}^{s_b},\\
Z : \mathbb{R} \times \mathbb{R}^n &\longrightarrow \mathbb{R}^s,\\
U &\subset \mathbb{R}^m.
\end{aligned} \tag{6.1}$$

Problem 6.2 (Kontinuierliches Optimalsteuerungsproblem). *Es seien die Angaben (6.1) und das Intervall $[a,b]$ gegeben. Es seien ferner*

$$x(\cdot) \in AC([a,b])^n, \quad u(\cdot) \in L_\infty([a,b])^m,$$

wobei AC für den Raum der absolut stetigen Funktionen steht. Dann ist ein `kontinuierliches Optimalsteuerungsproblem` *gegeben durch das folgende Optimierungsproblem:*
Minimiere

$$f_a(x(a)) + f_b(x(b)) + \int_a^b f\big(t, x(t), u(t)\big)\, dt$$

unter den Nebenbedingungen

$$\dot{x}(t) = g\big(t, x(t), u(t)\big) \quad \text{für fast alle } t \in [a,b],$$

$$\alpha_i(a, x(a)) \begin{cases} \leq 0 & , \quad i = 1, \ldots, s'_a, \\ = 0 & , \quad i = s'_a + 1, \ldots, s_a, \end{cases}$$

$$\beta_i(b, x(b)) \begin{cases} \leq 0 & , \quad i = 1, \ldots, s'_b, \\ = 0 & , \quad i = s'_b + 1, \ldots, s_b, \end{cases}$$

$$u(t) \in U \quad \text{für fast alle } t \in [a,b],$$

$$S(t, x(t)) \leq 0_{\mathbb{R}^s} \quad \text{für alle } t \in [a,b].$$

Später wird von der Bezeichnung $[a,b]$ auf $[t_0, t_f]$ gewechselt. Üblich sind des weiteren Transformationen auf die Intervalle $[-1,1]$ oder $[0,1]$. Durch Negation der Zielfunktion läßt sich jedes Maximierungsproblem in ein Minimierungsproblem transformieren und umgekehrt.
Die Nebenbedingungen können offenkundig beliebig kompliziert sein. Die Problemstellung 6.2 ist sehr allgemein gehalten. Oftmals haben die Problemstellungen glücklicherweise eine einfachere Struktur. Gewisse Bedingungen liegen dann nicht vor oder lassen sich vereinfachen. Für einige Aufgabentypen existieren spezielle Bezeichnungen. Die besonders häufigen Fälle werden jetzt angegeben.

Problem 6.3 (Formulierungen optimaler Steuerungsprozesse). *Die Dimensionen und Eigenschaften der Funktionen seien die gleichen wie bei Problem 6.2. Dann ist ein Optimalsteuerungsproblem gegeben durch:*
Minimiere

$$S(t_f, x(t_f)) + \int_{t_0}^{t_f} f\big(t, x(t), u(t)\big)\, dt$$

unter den Nebenbedingungen

$$\begin{aligned} \dot{x}(t) &= g\big(t, x(t), u(t)\big) \quad \text{für fast alle } t \in [t_0, t_f], \\ x(t_0) &= x_0, \\ u(t) &\in U, \quad t \in [t_0, t_f]. \end{aligned}$$

Die obige Aufgabenstellung heißt

- *`Bolza-Problem`,*
- *`Lagrange-Problem`, wenn $S \equiv 0$,*
- *`Mayer-Problem`, wenn $f \equiv 0$,*
- *`Lineares Mayer-Problem`, wenn $f \equiv 0$ und S linear ist.*

Scheinbar sind diese Probleme jeweils Spezialfälle. Dies täuscht jedoch. Man kann die Äquivalenz der vier Fälle nachweisen.

Satz 6.4 (Äquivalente Formulierungen optimaler Steuerungsprozesse). *Die vier Aufgabenstellungen des Problems 6.3 sind äquivalent. Insbesondere lassen sich alle Formulierungen auf die lineare Mayer-Form bringen.*

Beweis. Nur beispielhaft wird die Transformation eines Bolza-Problems zu einem linearen Mayer-Problem demonstriert. Dazu führt man zuerst einen neuen Zustandsvektor $\tilde{x}$ ein:

$$\begin{aligned} \tilde{x}(t) &= \big(x_1(t), \ldots, x_n(t), x_{n+1}(t)\big)^T \text{ und} \\ \dot{x}_{n+1}(t) &= f(t, x(t), u(t)) + \frac{d}{dt}\big(S(t, x(t))\big), \\ x_{n+1}(t_0) &= 0. \end{aligned}$$

Als Zielfunktion wählt man $x_{n+1}(t_f)$. Es gilt nämlich:

$$\begin{aligned} x_{n+1}(t_f) &= x_{n+1}(t_f) - \overbrace{x_{n+1}(t_0)}^{=0} = \int_{t_0}^{t_f} \dot{x}_{n+1}(t)\, dt \\ &= \int_{t_0}^{t_f} f\big(t, x(t), u(t)\big)\, dt + S(t_f, x(t_f)) + \overbrace{S(t_0, x(t_0))}^{=S(t_0,x_0)=const} \end{aligned}$$

Die Konstante am Ende hat für eine Lösung nur kosmetische Bedeutung. Insgesamt gilt also:

Minimiere

$$x_{n+1}(t_f)$$

unter den Nebenbedingungen

$$\begin{aligned}
\dot{x}_i(t) &= g_i(t, x(t), u(t)), \quad i = 1, \ldots, n \text{ für fast alle } t \in [t_0, t_f], \\
\dot{x}_{n+1}(t) &= f(t, x(t), u(t)) + \frac{d}{dt}\big(S(t, x(t))\big), \quad t \in [t_0, t_f], \\
x_i(t_0) &= (x_0)_i, \quad i = 1, \ldots, n, \\
x_{n+1}(t_0) &= 0, \\
u(t) &\in U.
\end{aligned}$$

Die anderen Umformungen werden ähnlich durchgeführt, siehe etwa [8, Kapitel II, Abschnitt 4]. □

Für manche Überlegungen sind bestimmte Problemformulierungen besser geeignet als andere. Die eine oder andere Überlegung kann man sich dann erleichtern, so in Abschnitt 6.5.

6.3 Grundlagen zum Minimumprinzip

Wie bei allen Optimierungsproblemen sucht man auch bei Problemstellungen des Typs der Aufgabe 6.2 nach notwendigen Optimalitätskriterien. Die Gegebenheiten der aktuellen Aufgabenstellung führen aber im Gegensatz zur finiten Optimierungstheorie auf unendlichdimensionale Aufgaben. Entscheidenen Anteil hat das Optimalitätsprinzip von Bellman, mit dessen geschickter Anwendung man zu Optimalitätskriterien - dem sogenannten Minimumprinzip - gelangt. Dieser Abschnitt wird einen Teil dieser Argumentationskette wiedergeben. Es wird nur soviel dargestellt, wie zum flüssigen Lesen und allgemeinen Verständnis der nachfolgenden Abschnitte notwendig ist. Es wird ferner die vereinfachte Problemstellung 6.3 zugrundegelegt.
Um das Bellmansche Optimalitätsprinzip formulieren zu können, braucht man zuerst ein neues Hilfskonstrukt:

Definition 6.5 (Optimalwertfunktion). *Es sei das Problem 6.3 gegeben. Es sei $s \in [t_0, t_f]$. $x(s)$ sei ein zulässiger Anfangswert bei $t = s$. Man definiert nun die* `Optimalwertfunktion` *V durch*

$$V(s, x(s)) := \min_{\substack{u:[t_0,s]\to\mathbb{R}^m \\ u(t)\in U}} \left(S(t_f, x(t_f)) + \int_s^{t_f} f\big(t, x(t), u(t)\big)\, dt \right),$$

wobei alle Nebenbedingungen des Problems 6.3 zu beachten sind.

Die Optimalwertfunktion läßt sich als optimale Lösung der Optimalsteuerungsaufgabe ab dem Zeitpunkt s mit dem Startwert $x(s)$ interpretieren. Die nähere Betrachtung der Optimalwertfunktion führt zu dem berühmten Bellmanschen Optimalitätsprinzip:

Theorem 6.6 (Bellmansches Optimalitätsprinzip). *In Richard Bellmans Werk [6, Abschnitt 3.3] über Dynamische Optimierung heißt es dazu:*

> Principle of Optimality. *An optimal policy has the property that, whatever the initial state and initial decision are, the remaining decision must constitute an optimal policy with regard to the outcome resulting from the first decision.*

Die Quintessenz des Bellmanschen Optimalitätsprinzips ist die folgende Darstellung der Optimalwertfunktion $V(s, x(s))$:

$$V(s,x(s)) = \min_{\substack{u(t)\in U \\ t\in[s,s+\delta t]}} \left(\int_s^{s+\delta t} f\big(t,x(t),u(t)\big)\,dt + V\big(s+\delta t, x(s+\delta t)\big)\right). \tag{6.2}$$

Hierbei ist δt ein kleines Zeitintervall. Dieses Prinzip entpuppt sich als der Schlüssel zum weiteren Vorgehen. Die Darstellungsweise der Formel (6.2) legt nahe, den zweiten Term nach δt zu entwickleln und das Integral im Riemannschen Sinne näherungsweise mit $\delta t \cdot f(t, x(t), u(t))$ zu bewerten. Nach Umformungen und Grenzübergang von δt gelangt man dann zu der Hamilton-Jacobi-Bellman-Gleichung (HJB-Gleichung). Dabei handelt es sich um eine partielle Differentialgleichung.

Definition 6.7 (Hamilton-Jacobi-Bellman-Gleichung). *Die Gleichung*

$$0 = \min_{u\in U} \{f(t,x,u) + V_x(t,x)g(t,x,u) + V_t(t,x)\}$$

heißt `Hamilton-Jacobi-Bellman-Gleichung`*. Mit V_t und V_x sind die Ableitungen von V bezüglich t respektive x gemeint.*

Bemerkung 6.8. *Der Term $V_t(t,x)$ hängt offenbar nicht von der Steuerung u ab. Er ist damit von der Minimumbildung unabhängig und kann aus der obigen geschweiften Klammer herausgezogen werden.*

Bemerkung 6.9. *Aus der Definition 6.5 folgt außerdem die Randbedingung*

$$V(t_f, x(t_f)) = S(t_f, x(t_f)).$$

Folgende Bezeichnung ist ferner geläufig:

Definition 6.10 (Hamilton-Funktion). *Der Term*

$$H(t, x, u, \lambda) := f(t, x, u) + \lambda g(t, x, u)$$

wird als ***Hamilton-Funktion*** *bezeichnet.*

Mit der Formulierung der Hamilton-Jacobi-Bellman-Gleichung hat man einen wesentlichen Schritt für die Herleitung des Minimumprinzips für die Problemstellung 6.3 geleistet. Es fehlt im wesentlichen noch die Herleitung der sogenannten adjungierten Differentialgleichung, worauf aber verzichtet wird. Das Minimumprinzip lautet nun:

Satz 6.11 (Minimumprinzip für Problem 6.3). *Die notwendigen Bedingungen für die Optimalität von u^* und x^* für das Problem 6.3 lauten:*
Für alle $t \in [t_0, t_f]$ und zulässige u gilt

$$\begin{aligned}
\dot{x^*}(t) &= g(t, x^*(t), u^*(t)), \\
x^*(t_0) &= x_0, \\
\dot{\lambda}(t) &= -H_x(t, x^*(t), u^*(t), \lambda(t)), \\
\lambda(t_f) &= S_x(t_f, x^*(t_f)), \\
H(t, x^*(t), u^*(t), \lambda(t)) &\geq H(t, x^*(t), u(t), \lambda(t)).
\end{aligned}$$

Beweis. Der Beweis für diese Aufgabenstellung findet sich in [54, Abschnitt 2.2]. Ohne den Term $S(t_f, x(t_f))$ vor dem Integral, also für ein Lagrange-Problem, findet man ihn auch in [15, §5]. □

Einfache Anwendungen des Minimumprinzips auf Beispiele der Problemstellung 6.3 findet man in [54, Abschnitt 2.3].
Satz 6.11 ist nur Spezialfall des allgemeinen Minimumprinzips. Die Problemstellung 6.2 ist in [25, Abschnitt 7.6] formuliert worden. Dort wird auf einen Beweis des Minimumprinzips in [34] verwiesen.
Schon einfache Beispiele zum Minimumprinzip offenbaren eine bedeutsame Erkenntnis über die Struktur von optimalen Lösungen:

Beobachtung 6.12. *Die optimale Steuerung kann auch dann unstetig sein, wenn alle Funktionen in der Aufgabenstellung 6.3 hinreichend glatt sind.*

Auf numerische Lösungsansätze hat diese Beobachtung eine tiefgreifende Wirkung.
Das Minimumprinzip in Satz 6.11 führt auf ein Randwertproblem, also eine Differentialgleichung. Hierüber eine Lösung zu erlangen, wird als „indirektes Verfahren“ bezeichnet. Dieses wird mit „indirekt“ tituliert, da man den technischen „Umweg“ über die notwendigen Optimalitätskritieren einschlägt. Demgegenüber

investieren sogenannte „direkte Verfahren“ keine theoretischen Erkenntnisse in das Problem, sondern bearbeiten das Problem direkt mittels einer geschickten Diskretisierung der Aufgabe. In dieser Arbeit werden letztendlich nur direkte Methoden weiterverfolgt.

6.4 Linear-Quadratische Optimalsteuerungsprobleme

Eine spezielle Aufgabenklasse der finiten Optimierungstheorie sind linear-quadratische Problemstellungen; die Zielfunktion ist dabei quadratisch, die Nebenbedingungen sind linear. Viele praktischen Fragestellungen führen auf diese Form, außerdem existiert zu ihnen eine geschlossene Lösungstheorie. Zu diesem Typus existiert ein Analogon in der Theorie der Optimalsteuerungsprobleme. Wie ihre Pendants im finiten Raum besitzen diese eine vergleichsweise einfache Struktur. Der Abschnitt orientiert sich an [43], [2] und [54].

Definition 6.13 (Linear-Quadratisches Optimalsteuerungsproblem).
Es sei ein Intervall $[t_0, t_f]$ *sowie ein Anfangswert* $x_0 \in \mathbb{R}^n$ *gegeben. Die Matrizen*

$$A(t) \in \mathbb{R}^{n\times n},\ \ B(t) \in \mathbb{R}^{n\times m},\ \ Q(t) \in \mathbb{R}^{n\times n},\ \ R(t) \in \mathbb{R}^{m\times m},\ \ S \in \mathbb{R}^{n\times n}$$

seien bezüglich t *stetig differenzierbar im obigen Intervall. Ferner seien*

$$R(t) > 0,\ \ Q(t) \geq 0,\ \ S \geq 0, \quad t \in [t_0, t_f],$$

also symmetrisch und positiv definit bzw. semidefinit. Dann ist ein `Linear-Quadratisches Optimalsteuerungsproblem` *durch das folgende Optimierungsproblem gegeben:*
Minimiere

$$\frac{1}{2}x(t_f)^T S\, x(t_f) + \frac{1}{2}\int_{t_0}^{t_f}\Big(x(t)^T Q(t)\, x(t) + u(t)^T R(t)\, u(t)\Big)dt$$

unter den Nebenbedingungen

$$\begin{aligned} \dot{x}(t) &= A(t)x(t) + B(t)u(t), \\ x(t_0) &= x_0. \end{aligned}$$

Die Terme vor und unter dem Integral sind quadratische Funktionen. Die Systemdynamik ist durch eine lineare Differentialgleichung gegeben. Der Vorfaktor $\frac{1}{2}$ spiegelt eine allgemeine Konvention wider, die sich eingebürgert hat, damit bei Ableitungen quadratischer Funktionen keine Vorfaktoren entstehen. Die Bezeichung „linear-quadratisch“ ist also gerechtfertigt.
Für diese Linear-Quadratischen Optimalsteuerungsprobleme kann man einen wertvollen Existenz- und Eindeutigkeitssatz beweisen.

Satz 6.14. *Für jedes endliche Intervall $[t_0, t_f]$ und jeden Anfangswert x_0 existiert eine eindeutige Lösung des Linear-Quadratischen Optimalsteuerungsproblems aus Definition 6.13. Man erhält:*
Die Steuerung u genügt dem Kontrollgesetz

$$u_{opt}(t) = -R(t)^{-1}B(t)^T P(t)\, x(t), \tag{6.3}$$

wobei man $P(t)$ als Lösung der sogenannen `Riccati-Gleichung`

$$\begin{aligned} \dot{P}(t) &= -P(t)A(t) - A(t)^T P(t) + P(t)B(t)R(t)^{-1}B(t)^T P(t) - Q(t), && (6.4)\\ P(t_f) &= S && (6.5) \end{aligned}$$

erhält, und $P(t)$ symmetrisch ist. Der Minimalwert der Zielfunktion ist

$$\frac{1}{2}x_0^T P(t_0)\, x_0. \tag{6.6}$$

Beweis. Der Beweis verwendet die Theorie des Abschnitts 6.3, insbesondere die Hamilton-Jacobi-Bellman-Gleichung der Definition 6.7. Die HJB-Gleichung lautet hier

$$\begin{aligned} &-V_t(t,x) = \\ &\min_{u(t)\in\mathbb{R}^m} \left\{ \frac{1}{2}\big(x(t)^T Q(t)x(t) + u(t)^T R(t)u(t)\big) + V_x(t,x)\big(A(t)x(t) + B(t)u(t)\big) \right\}. \end{aligned}$$

Zuerst leitet man die HJB-Gleichung nach u ab, da ohne Restriktionen an u hier ein unrestringiertes Optimierungsproblem vorliegt. Aufgelöst nach der Steuerung ergibt sich

$$u(t) = -R(t)^{-1}B(t)^T V_x(t,x)^T. \tag{6.7}$$

Man setzt dieses Resultat wieder in die HJB-Gleichung ein. Dann weist man nach, daß die Hamilton-Jacobi-Bellman-Gleichung eine quadratische Form

$$V(t,x) = x^T P(t)\, x \tag{6.8}$$

mit symmetrischem P als Lösung besitzt. Nun fährt man fort, indem man auch dies wiederum in die HJB-Gleichung einsetzt. Dann erhält man direkt die Riccati-Gleichung (6.4) mit der Randbedingung (6.5). (6.8) eingesetzt in (6.7) führt dann zu der optimalen Steuerung (6.3). Für genauere Ausführungen siehe [2, Abschnitt 2.3] oder [43, Theorem 3.1]. □

Durchaus interessant und wissenswert ist der Vergleich zu Linear-Quadratischen Systemen bei Diskreter Dynamischer Programmierung, siehe hierzu [10, Abschnitt 4.1].
Bisher enthielt das Zielfunktional nur quadratische Terme. Eine Erweiterung erhält man durch das Hinzufügen von linearen Termen.

Definition 6.15. *Es seien die Daten und Eigenschaften aus Definition 6.13 gegeben. Die Vektoren*

$$h(t) \in \mathbb{R}^n, \;\; k(t) \in \mathbb{R}^m, \;\; l \in \mathbb{R}^n \tag{6.9}$$

seien bezüglich t stetig differenzierbar im gegebenen Intervall. Die Aufgabenstellung lautet nun:
Minimiere

$$\begin{aligned} & \frac{1}{2} x(t_f)^T S\, x(t_f) + l^T x(t_f) \\ + \;\; & \frac{1}{2} \int_{t_0}^{t_f} \Big(x(t)^T Q(t)\, x(t) + u(t)^T R(t)\, u(t) + 2h(t)^T x(t) + 2k(t)^T u(t) \Big)\, dt \end{aligned}$$

unter den Nebenbedingungen

$$\begin{aligned} \dot{x}(t) &= A(t)x(t) + B(t)u(t), \\ x(t_0) &= x_0. \end{aligned}$$

Die Vorfaktoren innerhalb des Integrals sind so angelegt, daß sie beim Differenzieren verschwinden. Für diese Aufgabe läßt sich erneut ein Existenz- und Eindeutigkeitssatz der Lösung formulieren. Die Lösung unterscheidet sich kaum von derjenigen in Satz 6.14. An verschiedenen Stellen schieben sich die linearen Terme in die dortige Lösungsstruktur, siehe [43, Bemerkung 3.3].
Eine weitere Problemerweiterung entsteht, wenn man mit $x(t)^T Z(t)\, u(t)$ auch gemischte Terme im Zielfunktional zuläßt, wobei Z eine geeignete Matrix ist. In [11, Kapitel 5] wird dieser Fall eingehender betrachtet, auch in [43, Bemerkung 3.4] wird hierauf eingegangen.
An dieser Stelle ist es gewinnbringend, sich die Aussagen des Satzes 6.14 noch einmal qualitativ anzuschauen:

Bemerkung 6.16. *Die Ergebnisse des Satzes 6.14 zu der Linear-Quadratischen Aufgabe in Definition 6.13 lassen bei geeigneten Ausgangsdaten erhebliche Glattheitseigenschaften des optimalen Zustandes und der optimalen Steuerung vermuten (ebenso für Definition 6.15).*

Diese Beobachtung dient im Kapitel 7 als Motivation (und Rechtfertigung) für den numerischen Ansatz, Polynome als Näherungen der unbekannten Trajektorien x und u anzusetzen. Für ein allgemeines kontinuierliches Optimalsteuerungsproblem 6.2 wäre dies von vorneherein viel fragwürdiger.

6.5 Direkte Lösungsverfahren

Es gibt im wesentlichen zwei Herangehensweisen zur Lösung von Optimalsteuerungsproblemen:

- Indirekte Verfahren nutzen die notwendigen Optimalitätskriterien des Minimumprinzips, die analytisch hergeleitet werden müssen.
- Direkte Verfahren diskretisieren die einzelnen Elemente der Aufgabenstellung sofort, so daß statt einer kontinuierlichen Problemstellung eine endlichdimensionale entsteht.

Die Abbildung 6.1 veranschaulicht beide Verfahrensklassen.

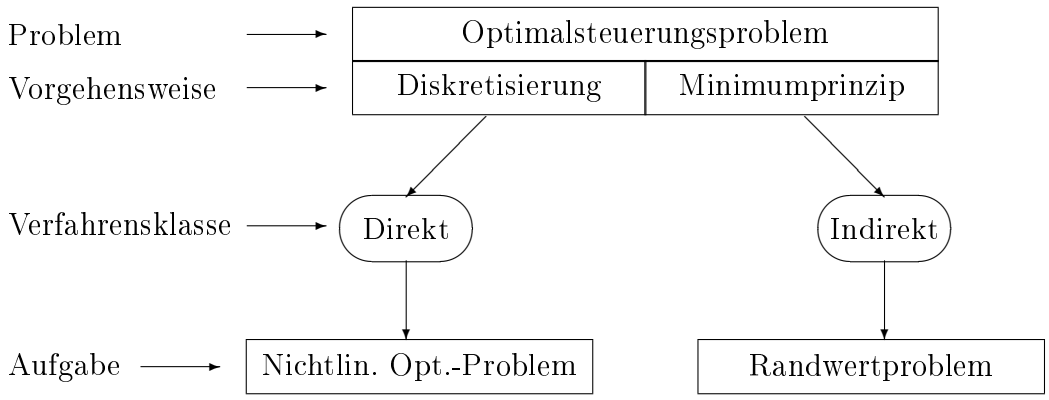

Abbildung 6.1: Lösungsstrategien von Optimalsteuerungsproblemen

In dieser Arbeit wird nur der direkte Ansatz verfolgt. Direkte Lösungsmethoden beruhen auf einer geeigneten Diskretisierung des Funktionals, der Differentialgleichung sowie der restlichen Zustands- und Steuerbeschränkungen. Die Wahl dieser Diskretisierung ist ausschlaggebend für die weiteren Rechnungen und die Güte der Ergebnisse. Für die allgemeine Aufgabenstellung 6.2 führt die naheliegenste und einfachste Wahl über die Festlegung eines Gitters $\mathbb{G}$

$$\mathbb{G} = \{t_i | \, i = 0, \ldots, N\} \quad \text{mit} \quad a = t_0 < \ldots < t_N = b \tag{6.10}$$

und der Näherungen

$$x(t_i) \approx x_i, \quad i = 0, \ldots, N, \tag{6.11}$$

$$u(t_i) \approx u_i, \quad i = 0, \ldots, N-1 \tag{6.12}$$

zu der Approximation des Integrals mittels einer Riemann-Summe und der Differentialgleichung durch ein Euler-Verfahren. Mit der Schrittweite

$$h_{i-1} := t_i - t_{i-1}, \quad i = 1, \ldots, N \tag{6.13}$$

entsteht ein neues Optimierungsproblem in der Formulierung aus [25, 7.2.1]:

Definition 6.17 (Diskretes Dynamisches Optimierungsproblem).
Minimiere

$$f_a(x_0) + f_b(x_N) + \sum_{i=1}^{N} h_{i-1} f(t_{i-1}, x_{i-1}, u_{i-1})$$

unter den Nebenbedingungen

$$x_i = x_{i-1} + h_{i-1} g(t_{i-1}, x_{i-1}, u_{i-1}), \quad i = 1, \ldots, N,$$

$$\alpha_i(t_0, x_0) \begin{cases} \leq 0 & , \quad i = 1, \ldots, s'_a, \\ = 0 & , \quad i = s'_a + 1, \ldots, s_a, \end{cases}$$

$$\beta_i(t_N, x_N) \begin{cases} \leq 0 & , \quad i = 1, \ldots, s'_b, \\ = 0 & , \quad i = s'_b + 1, \ldots, s_b, \end{cases}$$

$$u_{i-1} \in U, \quad i = 1, \ldots, N,$$

$$S(t_i, x_i) \leq 0_{\mathbb{R}^s}, \quad i = 0, \ldots, N,$$

$$x_i \in \mathbb{R}, \quad i = 0, \ldots, N,$$

$$u_i \in \mathbb{R}, \quad i = 0, \ldots, N-1.$$

Ein Diskretes Dynamisches Optimierungsproblem läßt sich in verschiedener Weise bearbeiten. Zum Beispiel existieren für Diskrete Dynamische Optimierungsprobleme spezielle Lösungsalgorithmen. Entscheidend und hier allein bedeutsam ist jedoch, daß durch die Diskretisierung ein endlichdimensionales Optimierungsproblem entsteht. Die allgemeine Problemstruktur läßt sich in die Aufgabenklasse der folgenden Definition aus [25, 5.1.1] einordnen.

Definition 6.18 (Nichtlineares Optimierungsproblem (NLP)). *K sei eine Teilmenge des $\mathbb{R}^n$. Die Funktionen $f, g_1, \ldots, g_m : K \to \mathbb{R}$ seien wenigstens auf K definiert. Ein* `Nichtlineares Optimierungsproblem (NLP)` *ist durch die folgende Aufgabenstellung gegeben:*
Minimiere

$$f(x)$$

unter den Nebenbedingungen

$$\begin{aligned} g_i(x) &\leq 0, \quad i = 1, \ldots, m', \\ g_i(x) &= 0, \quad i = m' + 1, \ldots, m, \\ x &\in K. \end{aligned}$$

Die Lösungstheorie derartiger Problemstellungen wird in dieser Arbeit nicht besprochen. Zur Lösung solcher nichtlinearen Optimierungsaufgaben gibt es gut entwickelte Konzepte und Algorithmen, siehe dazu [37] oder [25, Kapitel 6].
Die Definition 6.18 gibt die Struktur eines direkten Verfahrens bei der Anwendung auf allgemeine Optimalsteuerungsprobleme an. Einen Spezialfall nichtlinearer Optimierungsprobleme gewinnt man, wenn die Optimalsteuerungsprobleme die linear-quadratische Struktur der Definitionen 6.13 oder 6.15 besitzen:

Definition 6.19 (Quadratische Programm (QP)). *Es sei $Q \in \mathbb{R}^{n \times n}$ eine positiv definite Matrix. Ferner sei $h \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ und $b \in \mathbb{R}^m$. Die Aufgabe Minimiere*

$$x^T Q\, x + h^T x$$

unter den Nebenbedingungen

$$Ax = b$$

wird als `Quadratisches Programm` *bezeichnet.*

Zu diesen Optimierungsaufgaben sind die verfügbaren Algorithmen besonders effektiv.
Andere Diskretisierungen eines Optimalsteuerungsproblems sind unbedingt wünschenswert, da die Riemann-Summe und das Euler-Verfahren in ihren Aufgabenbereichen der numerischen Integration und der Lösung von Differentialgleichungen die primitivsten Ansätze repräsentieren. Die Gauß-Quadratur und Runge-Kutta-Verfahren sind hochwertiger, wenn auch aufwendiger. Unweigerlich stellt sich die Frage, wie nun die Aufgabenstellung dieser beiden Lösungsverfahren auf ein Optimalsteuerungsproblem übertragen werden kann. Der Satz 6.4 über die Äquivalenz verschiedener Optimalsteuerungsaufgaben ist nun hilfreich.
Die folgende - vereinfachte - Aufgabenstellung soll die Wahl der Diskretisierung erklären. Zur Vereinfachung werden das Intervall $t \in [t_0, t_0 + h]$ und die numeri-

schen Näherungen η_0 und η_1 verwendet. Es gilt nach Satz 6.4:

$$
\begin{array}{ccc}
\text{(a)} & \Longleftrightarrow & \text{(b)} \\
\min \displaystyle\int_{t_0}^{t_0+h} f\big(t, x_1(t), u(t)\big)\, dt & & \min\, x_2(t_0+h) \\
s.t. \quad \dot{x}_1(t) = g\big(t, x_1(t), u(t)\big), & & s.t. \quad \dot{x}_1(t) = g\big(t, x_1(t), u(t)\big), \\
\quad x_1(t_0) = \eta_0, & & \qquad \dot{x}_2(t) = f\big(t, x_1(t), u(t)\big), \\
& & \qquad x_1(t_0) = \eta_0, \\
& & \qquad x_2(t_0) = 0.
\end{array}
$$

Für das Problem (b) ist ein Runge-Kutta-Ansatz naheliegend:

1. Man wähle Stützstellen $t_0 + c_i h$, $i = 1, \ldots, s$.
2. Über die Diskretisierung der Steuerung u mache man zunächst keine Aussagen.
3. Man formuliere das Gleichungssystem eines Runge-Kutta-Verfahrens wie in Definition 4.1.

Es wird $\eta_0^1 := \eta_0$ gesetzt, wobei der obere Index wie bei anderen Variablen nun die Dimension bezeichnen soll.
Dann entsteht das folgende nichtlineare Optimierungsproblem:

Minimiere

$$\eta_1^2$$

unter den Nebenbedingungen

$$
\begin{pmatrix} k_i^1 \\ k_i^2 \end{pmatrix} = \begin{pmatrix} g(t_0 + c_i h, \eta_0^1 + h \sum_{j=1}^s a_{ij} k_j^1, u_i) \\ f(t_0 + c_i h, \eta_0^1 + h \sum_{j=1}^s a_{ij} k_j^1, u_i) \end{pmatrix}, \quad i = 1, \ldots, s, \tag{6.14}
$$

$$
\begin{pmatrix} \eta_1^1 \\ \eta_1^2 \end{pmatrix} = \begin{pmatrix} \eta_0^1 \\ 0 \end{pmatrix} + h \sum_{i=1}^s b_i \begin{pmatrix} k_i^1 \\ k_i^2 \end{pmatrix}. \tag{6.15}
$$

Die rechte Seite von (6.14) hängt nicht von k_i^2 ab. Für die Lösung des Gleichungssystems ist die zweite Gleichungskomponente in (6.14) deshalb irrelevant. Gleichzeitig ist die Zielfunktion gleich der zweiten Komponente von (6.15). Man ersetzt nun den Zielfunktionsterm η_1^2 durch die zweite Komponente aus (6.15) und damit auch der zweiten Komponente aus (6.14). Dann läßt man wieder die nun überflüssigen Dimensionsindizes weg mit $\eta_0 := \eta_0^1$ und $k_i := k_i^1$. Es ergibt sich:

Minimiere

$$h \sum_{i=1}^{s} b_i f\big(t_0 + c_i h, \eta_0 + h \sum_{j=1}^{s} a_{ij} k_j, u_i\big) \tag{6.16}$$

unter den Nebenbedingungen

$$k_i = g\big(t_0 + c_i h, \eta_0 + h \sum_{j=1}^{s} a_{ij} k_j, u_i\big), \quad i = 1, \ldots, s. \tag{6.17}$$

Man blicke nun zurück auf die Aufgabenstellung (a). Die Nebenbedingung (6.17) hat genau diejenige Gestalt, die man auch bei einem Runge-Kutta-Ansatz für das Problem (a) angesetzt hätte. Die Formel (6.16) eröffnet nun, wie das Integral für (a) approximiert zu werden hat. Wenn also der Lösungsansatz für die Differentialgleichung einem der impliziten Runge-Kutta-Verfahren aus den Abschnitten 4.5 und 4.6 entstammt, so muß das Integral mit der zugehörigen Gauß-Quadraturregel berechnet werden. Die Formel (6.16) entspricht im übrigen auch den Formeln (4.3) bis (4.6) aus dem Abschnitt 4.2, in dem die Runge-Kutta-Verfahren über die Integration motiviert wurden.

Bemerkung 6.20. *Die Diskretisierung der Differentialgleichung und die Approximation des Kostenfunktionals sollten nach zusammengehörenden Regeln erfolgen. Insbesondere kombinieren sich die Regeln der Gauß-Quadratur mit jenen der impliziten Runge-Kutta-Verfahren.*

Die anfänglichen Umformungen, die zu dem Diskreten Dynamischen Optimierungsproblem in Definition 6.17 geführt haben, waren unausgesprochen mit diesem Prinzip konsistent. Die Riemann-Summe und das Euler-Verfahren sind in diesem Sinne zusammengehörig, wie schon aus der Definition 3.16 und (3.15) hervorging.
In [52] werden Varianten für die Diskretisierung der Steuerung u diskutiert. Die Stützstellen für u werden durch einen Vektor $\sigma \in \mathbb{R}^s$ mit

$$t_0 + \sigma_i h, \quad 0 \leq \sigma_i \leq 1, \quad i = 1, \ldots, s \tag{6.18}$$

festgelegt, so daß

$$u_i \approx u(t_0 + \sigma_i h), \quad 0 \leq \sigma_i \leq 1, \quad i = 1, \ldots, s \tag{6.19}$$

ist.

Bemerkung 6.21. *Es sei das Problem 6.3 mit noch gewissen Zusatzvoraussetzungen und folgende Auswahlregeln gegeben:*

$$\sigma_i = c_i, \quad i = 1, \ldots, s.$$

Dann kann man zeigen, daß für ein Runge-Kutta-Verfahren, welches die Konvergenzordnung 2 besitzt, die Bedingung $D(1)$ erfüllt und bei dem alle Gewichte b_i echt größer null sind, die Diskretisierungen (6.14) und (6.15) zu der Konvergenzordnung 2 des nichtlinearen Optimierungsproblems führen, siehe [52].

Es lassen sich einige Phänomene generalisieren, die direkte und indirekte Methoden auszeichnen. Die Tabelle 6.1 listet die für diese Arbeit bedeutsamen Beobachtungen auf. Die gegensätzlichen Ansätze beider Methoden resultieren in zum Teil spiegelsymmetrischen Vorzügen und Nachteilen.

	Indirekte Methoden	Direkte Methoden
Vorteile	- Erfüllen die notwendigen Optimalitätskriterien - Liefern sehr präzise Lösungen	- Umgehen die notwendigen Optimalitätskriterien - NLPs sind relativ leicht lösbar
Nachteile	- Notwendige Optimalitätskriterien müssen analytisch hergeleitet werden - Guter Startwert für das Randwertproblem erforderlich	- Lösungen sind nicht so genau wie bei indirekten Methoden - Notwendigen Optimalitätskriterien werden im allgemeinen nicht erfüllt

Tabelle 6.1: Einige Vergleiche zwischen direkten und indirekten Verfahren zur Lösung von Optimalsteuerungsproblemen

Besonders das Umgehen der analytischen notwendigen Bedingungen zeichnet die direkten Verfahren aus. Die Lösung von Optimalsteuerungsproblemen mittels direkter Methoden ist deshalb in letzter Zeit intensiviert worden. Drei Beispiele dafür sind [7], [46] und [63]. Ein in diesen drei Arbeiten maßgeblicher Ansatz wird im nächsten Kapitel vorgestellt.

Kapitel 7

Kollokationsverfahren im Bereich optimaler Steuerungen

7.1 Einleitung

Im Abschnitt 6.5 erfolgte bereits eine kurze Einleitung in direkte Lösungsverfahren für Optimalsteuerungsprobleme. Zum Diskretisieren boten sich Runge-Kutta-Verfahren an. Man erhofft sich damit eine effektive Approximation der Differentialgleichung und des Zielfunktionals, ähnlich wie bei Anfangswertproblemen. Die Formeln (6.16) und (6.17) sind aber noch etwas unpraktisch für ein Optimalsteuerungsproblem. Außerdem erzeugt man so nur diskrete Werte, nicht aber kontinuierliche Lösungen für den Zustand x und die Steuerung u. Die in Kapitel 5 bewiesene Gleichwertigkeit bestimmter Polynomansätze kreiert jetzt eine geeignete Schreibweise und führt zu einem Kollokationsansatz.
Einige Aspekte des folgenden Ansatzes müssen allerdings eingehender erläutert werden. So wurde die Runge-Kutta-Diskretisierung in Abschnitt 6.5 für den Zustand x vorgenommen, während die Diskretisierung der Steuerung u noch etwas wage blieb. Es ist nicht leicht zu begründen, wie man u approximiert. Weiter ist auch eine generelle Diskrepanz zwischen dem in diesem Kapitel verwendeten Polynomansatz und den Kollokationsverfahren des Kapitels 5 zu attestieren. Auch darf nicht vergessen werden, daß ein Optimalsteuerungsproblem eine grundsätzlich andere Aufgabenstellung als ein Anfangswertproblem ist.

Im Abschnitt 7.2 wird versucht, die soeben angekündigten kritischen Gesichtspunkte näher zu beleuchten und zu rechtfertigen.
Die Anwendung der Diskretisierung auf eine relativ allgemeine Problemklasse wird im Abschnitt 7.3 geschildert.
Der Abschnitt 7.4 präzisiert den Ansatz für Linear-Qudratische Optimalsteuerungsprobleme des Abschnitts 6.4. Dabei wird die Definition 6.15 noch um Ne-

benbedingungen erweitert.

Es würde zuweit führen, die numerischen Ansätze dieses Kapitels theoretisch zu begründen. Insbesondere werden keine Konvergenzaussagen behauptet und Konvergenzuntersuchungen durchgeführt, wie es [52] leistet. Es wird sich mehr auf die Vorgehensweise anderer Arbeiten gestützt, bei denen die Analyse direkter Verfahren zur Lösung von Optimalsteuerungsproblemen den Schwerpunkt bilden. Im Kontext der Diplomarbeit soll diese Anwendung verdeutlichen, wie Strukturen der Gauß-Quadratur und der impliziten Runge-Kutta-Verfahren auch in einen Lösungsansatz für Optimalsteuerungsproblemen einfließen können. Im Hinblick auf die Beispiele des Kapitels 8 wird eine Methode angeboten, Linear-Quadratische Optimalsteuerungsprobleme numerisch zu bearbeiten. Die Quellen des Abschnitts sind deshalb als Ideenlieferanten zu verstehen. Ein Ausgangspunkt der Diplomarbeit war [19]. In [7], [46] und [63] werden direkte Methoden grundlegender untersucht.

7.2 Motivation und Rechtfertigung

Die Idee, Polynome mit zunächst ungekannten Koeffizienten für den Zustand x in ein allgemeines Optimalsteuerungsproblem einzusetzen, beruht auf den Erkenntnissen des Kapitels 5. Bei einer geeigneten Wahl von Stützstellen sind die Kollokationsverfahren äquivalent zu impliziten Runge-Kutta-Verfahren. Im Sinne der direkten Methoden des Abschnitts 6.5 scheint ein solcher Ansatz gerechtfertigt. Die Diskretisierung der Steuerung u ist dagegen zunächst eine völlig neue Fragestellung. In [52] werden hierzu Untersuchungen durchgeführt. Die Bemerkung 6.21 kann als Begründung für den hier gewählten Ansatz dienen. Aber auch andere Argumente können verwendet werden.

Diskretisierung der Steuerung. Die folgenden Ausführungen richten sich nach [7], [46] und [63]. In [7, Abschnitt 2.2 und 3.3] findet man diese Darlegungen ausführlicher. Im weiteren werden die anglophonen Bezeichnungen übernommen. Die hier gewählte Diskretisierung der Steuerung u (und auch des Zustandes x) läßt sich in einen größeren Rahmen einordnen. Einen allgemeinen Ansatz zur Lösung von gewöhnlichen (und auch partiellen) Differentialgleichungen bieten die sogenannten „Spectral methods“. Im Unterschied zu anderen Lösungsverfahren - zum Beispiel Finite-Element-Methoden - steht bei Spectral methods zunächst nicht die Unterteilung des Gebietes im Vordergrund. Stattdessen wird die numerische Lösung auf dem gesamten Gebiet als eine Linearkombination von gewissen Basisfunktionen Φ_j angesetzt, die auf dem gesamten Gebiet (global) wirken. Für das Optimalsteuerungsproblem bildet man beispielsweise auf dem Intervall

$[t_0, t_f = t_0 + h]$:

$$x(t) \approx x^p(t) = \sum_{j=1}^{s} a_j \cdot \Phi_j(t), \tag{7.1}$$

$$u(t) \approx u^p(t) = \sum_{j=1}^{s} b_j \cdot \Phi_j(t), \tag{7.2}$$

wobei die Φ_j gewisse „Global basis functions“ repräsentieren, und die reellwertigen Koeffizienten a_j und b_j vorerst unbekannt sind. Typischerweise werden vor allem orthogonale Polynome oder trigonometrische Funktionen als Basisfunktionen gewählt.
Nach der Substitution solcher Näherungsfunktionen müssen in der Regel auch die Differentialgleichungen angepaßt werden, denn normalerweise können diese von den eingesetzten Linearkombinationen nicht mehr erfüllt werden. Hierbei existieren durchaus mehrere Alternativen, zum Beispiel „Tau methods“ oder „Galerkin methods“, siehe [7, Abschnitt 2.2.2]. Die hier verwendete Methode wird „Collocation method“ oder auch „Pseudospectral method“ genannt. Bei den pseudospectral methods wird die Gültigkeit der Differentialgleichungen an einer Menge von Punkten $\{t_0 + c_i h,\ i = 1, \ldots, s\}$ gefordert. Diese Stellen heißen Kollokationspunkte. Die Differentialgleichung eines Optimalsteuerungsproblems wird dann durch folgende Gleichungen ersetzt:

$$\begin{aligned}
\big(x^p(t)\big)'\Big|_{t=t_0+c_i h} &\stackrel{!}{=} g\big(t_0 + c_i h, x^p(t_0 + c_i h), u^p(t_0 + c_i h)\big), \\
&\quad i = 1, \ldots, s \quad \text{beziehungsweise} \\
\Big(\sum_{j=1}^{s} a_j \Phi_j(t)\Big)'\Big|_{t=t_0+c_i h} &\stackrel{!}{=} g\Big(t_0 + c_i h, \sum_{j=1}^{s} a_j \Phi_j(t_0 + c_i h), \sum_{j=1}^{s} b_j \Phi_j(t_0 + c_i h)\Big), \\
&\quad i = 1, \ldots, s.
\end{aligned} \tag{7.3}$$

Die pseudospectral methods sind entscheidend durch die Wahl der Kollokationspunkte $\{t_0 + c_i h,\ i = 1, \ldots, s\}$ geprägt. Werden als Basisfunktionen Polynome benutzt und als Kollokationspunkte die Nullstellen der Lobatto-Polynome, entstehen die „Legendre-Gauss-Lobatto collocation methods“. Auch die Bezeichnung „Legendre pseudospectral methods“ ist gebräuchlich. In Abschnitt 7.4 wird begründet, warum sich die Lagrange-Polynome l_j als Basisfunktionen besonders bewähren.
Im Abschnitt 2.8 sind bereits viele Strukturen für die Legendre pseudospectral methods erarbeitet, etwa der Satz 2.29. Die Formel (2.49) weist auf den Zusammenhang zwischen Lagrange-Polynomen und Lobatto-Polynomen hin. Der Satz 2.30 ermöglicht die Berechnung der Vorfaktoren der Variablen a_j der linken Seite in (7.3). So lohnt sich im nachhinein die geleistete Arbeit.

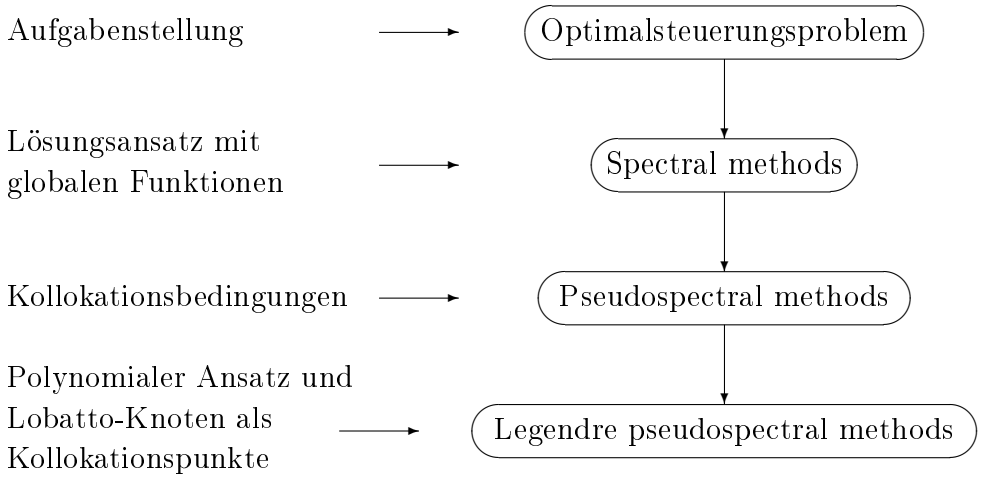

Abbildung 7.1: Konstruktion der Legendre pseudospectral methods

Es läßt sich resümieren, daß Legendre pseudospectral methods eine weitere Begründung für einen Polynomansatz für den Zustand, und eine erste für die Steuerung, liefern. Für mathematischere Erklärungen sei auf die erwähnten Quellen verwiesen. Hier bleibt die gewichtige Festlegung zu dokumentieren, daß ab nun die Polynome x^p für x und u^p für u angesetzt werden mit

$$\operatorname{Grad} x^p = \operatorname{Grad} u^p. \tag{7.4}$$

In einer gewissen Abweichung von den Legedre pseudospectral methods wird in den Abschnitten 7.3 und 7.4 aber doch noch zusätzlich das Intervalls $[t_0, t_f]$ in N Abschnitte zerlegt. x und u werden dann stückweise polynomial formuliert. Gerade für komplexere Aufgabenstellungen erscheint eine solche ergänzende Diskretisierung unverzichtbar.
Die Abbildung 7.1 kommentiert links den rechts zurückgelegten Weg zu den Legrende pseudospectral methods.

Polynomgrad versus Anzahl der Kollokationsstellen. Die Ausführungen zu Spectral methods sollten vor allem den Polynomansatz für die Steuerung rechtfertigen. Für den Zustand schien eine grundsätzliche Übereinstimmung mit den bekannten Kollokationsverfahren des Kapitels vorzuliegen. Das ist aber nicht ganz richtig, wie bei genauerer Betrachtung ersichtlich wird. Bei der Anwendung eines Kollokationsverfahren mit Polynom x^p auf ein Anfangswertproblem

$$\begin{aligned} \dot{x}(t) &= g(t, x(t)), \\ x(t_0) &= x_0 \end{aligned}$$

wurde in Definition 5.1 festgelegt, wie der Polynomgrad zu wählen ist:

$$\operatorname{Grad} x^p = \big|\{t_0 + c_i h, i = 1, \ldots, s\}\big| = s. \tag{7.5}$$

Den $s+1$ Koeffizienten des Polynoms x^p stehen s Kollokationsbedingungen und die Anfangswertbedingung gegenüber. Diese Festlegung erwies sich als zweckmäßig für die Existenz und Eindeutigkeit eines Kollokationsverfahrens. Genau diese Kollokationspolynome sind äquivalent zu gewissen impliziten Runge-Kutta-Verfahren. Setzt man aber für ein Optimalsteuerungsproblem die Legendre pseudospectral methods mit den Formeln (7.1) und (7.2) an, wobei

$$\Phi_j(t) = l_j(t) \tag{7.6}$$

gewählt werden kann, ergibt sich jedoch

$$\operatorname{Grad} x^p = \operatorname{Grad} u^p = s - 1 < s = \big|\{t_0 + c_i h, i = 1, \ldots, s\}\big|. \tag{7.7}$$

Das Zustandspolynom x^p besitzt nur s Koeffizienten, die Kollokationsnebenbedingungen und die Anfangsbedingung einer Differentialgleichung summieren sich jedoch zu $s+1$ Gleichungen. Für eine fest gewählte Steuerung wären die Nebenbedingungen damit allgemein nicht erfüllbar! In [46, Abschnitt 3.1] wird hierauf hingewiesen. Mit sinngemäßen Bezeichnungen heißt es dort:

> [...] *it is, in general, not possible to solve the* $n_y(N+2)$ *discretized state equations* [...] *for the* $n_y(N+1)$ *states* $y_0, \ldots, y_N$ *given controls* $u_0, \ldots, u_N$*, even if the infinite dimensional state equation* [...] *has a unique solution* y *for given control* u*. This is quite different from the collocation discretizations* [...]*, where the discretized state equation consists of* $n_y(N+1)$ *equations and where, under suitable assumptions, the discretized state equation has a unique solution* $y_0, \ldots, y_N$*, given controls* $u_0, \ldots, u_N$*. Hence the discretization* [...] *of the state equation* [...] *only makes sense in the context of optimal control, but not for simulations.*

Diese Legendre pseudospectral methods stimmen also auch für den Zustand x nicht ganz mit den Kollokationsverfahren des Kapitels 5 überein. Erst der Polynomansatz für die Steuerung u erhöht (verdoppelt) die Anzahl der Variablen und erzeugt Freiheitsgrade, da nun im allgemeinen weniger Nebenbedingungen als Variablen vorliegen. Die Optimierung legt die Koeffizienten der Polynome schließlich fest.

Wahl der Lobatto-Nullstellen. Nach der Beschreibung des Polynomansatzes fehlen noch ein paar Argumente, warum die Gauß-Lobatto-Knoten als Kollokationspunkte verwendet werden. Ein Vorzug besteht im Vorliegen von $c_1 = 0$ und $c_s = 1$. Für $c_1 = 0$ gilt:

$$x^p(t_0 + c_1 h) = \sum_{j=1}^{s} a_j \cdot l_j(t_0 + c_1 h) = a_1 = x_0. \tag{7.8}$$

Für eine Randbedingung kann man $a_s = x_f$ folgern. Damit sind bereits vor der Optimierung Variablenbelegungen bekannt.
In den Quellen [7], [46] und [63] werden weitere gewichtige Argumente angeführt. Diese können in dieser Arbeit nicht wiedergegeben werden. Es kann hier auch nicht überprüft werden, ob die dortigen Gegebenheiten identisch mit den hiesigen sind. Einige Zitate sollen dem neugierigen Leser als Anregung dienen:

> [63, Abschnitt 1.2]: *Elnagar et al* [...] *and Ross and Fahroo* [...] *employed the Legendre spectral collocation method for solving a variety of optimal control problems, and showed that highly accurate results can be obtained with a low degree of discretization.*

> [63, Abschnitt 1.2 und Abschnitt 2.1.3]: *The LGL [Legendre-Gauß-Lobatto] points offer the „best" discretization in the sense of minimal least-square error.*

> [46, Kapitel 1 (Einleitung)]: *Such methods have attracted attention* [...] *because of their alleged superior approximation properties and, in the case of Legendre pseudospectral method, the availability of a so-called adjoint map or estimate.*

Ausführlich untersucht [7] pseudospectral methods. Festzuhalten bleibt, daß die Lobatto-Polynom-Nullstellen als Kollokationspunkte festgelegt werden und die Nullstellen von Radau- oder Gauß-Legendre-Polynomen offenbar unüblich sind.
Als letztes sind einige allgemeine Gedanken zu den Polynomansätzen für Optimalsteuerungsprobleme notwendig.

Diskussion des Polynomansatzes. Es lassen sich gravierende Einwände gegen einen Polynomansatz erheben. Im wesentlichen beruhen diese auf mögliche kontrastierende Glattheitseigenschaften der optimalen Lösungen und derjenigen der numerischen Lösungen. Charakteristisch dafür ist insbesondere die Steuerung u, die im Optimum Sprünge aufweisen kann und damit gegebenfalls noch nicht einmal stetig ist, wie schon in Bemerkung 6.12 beobachtet wurde.

Die Lösung eines implementierten Optimierungsprogramms besitzt immer die Struktur, die der Programmierer involviert:

$$\begin{aligned} \text{Zustand } x \text{ unbekannter Funktionenklasse} &\longleftrightarrow \text{Polynom } x^p, \\ \text{Steuerung } u \text{ unbekannter Funktionenklasse} &\longleftrightarrow \text{Polynom } u^p. \end{aligned}$$

Damit ensteht ein unüberbrückbarer Gegensatz zwischen der tatsächlichen optimalen Lösung und den numerischen optimalen Ergebnissen:

$$\begin{aligned} x_{opt} \text{ liegt in } AC &\longleftrightarrow x^p_{opt} \text{ liegt in } C^\infty, \\ u_{opt} \text{ liegt in } L_\infty &\longleftrightarrow u^p_{opt} \text{ liegt in } C^\infty. \end{aligned}$$

Die numerische Lösung ist unendlich oft differenzierbar. Sie kann folglich Unstetigkeitsstellen oder nichtglatte Punkte nicht beschreiben.
Für die Linear-Quadratischen Optimalsteuerungsprobleme des Abschnitts 6.4 herrscht dagegen eine konträre Ausgangssituation vor. Nach Bemerkung 6.16 existieren hier günstige Bedingungen für einen glatten Lösungsansatz. Beispiele im Abschnitt 8.4 werden exzellente Approximationen der optimalen Lösung bei nur geringem Aufwand bestätigen. Die spezielle - aber geläufige - Linear-Quadratische Aufgabenstruktur bildet damit das Fundament des numerischen Ansatzes. Solche Aufgaben bilden auch den Kern der Beispielrechnungen in Abschnitt 8.4.
Doch gerade die numerische Struktur eines allgemeinen nichtlinearen Optimierungsproblems der Definition 6.18 reizt, die obigen Anfechtungen zurückzustellen. Zusätzliche Bedingungen im Optimalsteuerungsproblem bewirken keine strukturellen Änderungen im diskretisierten nichtlinearen Optimierungsproblem 6.18, sondern fügen lediglich weitere Gleichungen oder Ungleichungen hinzu. Man überläßt es dem Optimierungsprogramm, wie sich solche Modifikationen auf die numerische Lösung auswirken. Insofern besitzen die Abschnitte dieses Kapitels Spielräume in der Aufgabenstellung. Letztendlich ist es ein Experiment, ausgehend von der gesicherten Basis der Linear-Quadratischen Aufgaben in verschiedene Richtungen auszufächern. Gründe dafür sind beispielsweise:

- Auch bei Anfangswertproblemen gelten die Konsistenzordnungen der impliziten Runge-Kutta-Verfahren nur bei entsprechenden Glattheitsvoraussetzungen, siehe Bemerkung 3.26. Kaum ein numerisches Verfahren wird allen widrigen Gegebenheiten von praktischen Aufgaben gerecht.

- Die optimalen Lösungen sind prinzipiell unbekannt. Ein einfacher erster Ansatz kann Erkenntnisse zutage fördern, ohne viel Vorarbeit zu investieren.

- Eine Näherungslösung kann für genauere - und aufwendigere - Verfahren einen guten Startwert liefern.

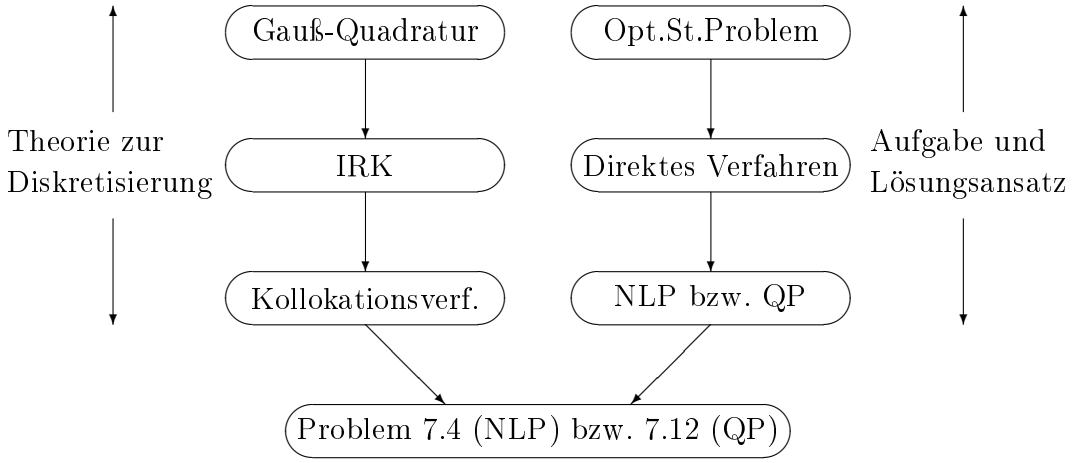

Abbildung 7.2: Synthese von numerischen Verfahren zur Lösung von Anfangswertproblemen und Optimalsteuerungsproblemen

Zwei abschließende Bemerkungen fassen die Schwierigkeiten des Kapitels zusammen.

Bemerkung 7.1. *Ein Optimalsteuerungsproblem unterscheidet sich grundsätzlich von einem Anfangswertproblem. Qualitative und quantitative Übertragungen von Konsistenzordnungen bei einer Runge-Kutta-Diskretierung können nicht erwartet werden.*

Bemerkung 7.2. *Der vorgeschlagende Kollokationsansatz weicht von den Kollokationsverfahren des Kapitels 5 ab.*

Die Abbildung 7.2 verdeutlicht, wie die Diskretisierungstheorie für Anfangswertprobleme der Kapitel 2 bis 5 mit der numerischen Lösungstheorie der direkten Verfahren für Optimalsteuerungsprobleme der Kapitel 6 und 7 zusammengeführt wird. Einerseits kulminieren die Anfänge der Gauß-Quadratur in den Kollokationsverfahren. Andererseits führt ein direktes Verfahren zu einem nichtlinearen Optimierungsproblem. Die Vereinigung vollzieht sich in den Aufgabenstellungen 7.4 und 7.12, die im laufenden Kapitel erarbeitet werden.

7.3 Allgemeiner numerischer Ansatz

Ausgangspunkt der Betrachtungen sei das Problem 6.3, unter Umständen noch ergänzt um eine Randbedingung bei t_f. Diese Beliebigkeit ist nicht untypisch für direkte Verfahren. Im Bewußtsein der Diskussion des Abschnitts 7.2 überläßt man es dem Optimierungsprogramm, wie sich zusätzlichen Restriktionen auswirken.

Problem 7.3. *Minimiere*

$$S(t_f, x(t_f)) + \int_{t_0}^{t_f} f\big(t, x(t), u(t)\big)\, dt$$

unter den Nebenbedingungen

$$\begin{aligned} \dot{x}(t) &= g\big(t, x(t), u(t)\big) \quad \text{für fast alle } t \in [t_0, t_f], \\ x(t_0) &= x_0, \\ x(t_f) &= x_f, \\ u(t) &\in U, \quad t \in [t_0, t_f]. \end{aligned}$$

Das Intervall $[t_0, t_f]$ wird zuerst in N Abschnitte zerlegt:

$$\mathbb{G}_N := \{t_0, \ldots, t_N |\ t_0 < t_1 < \ldots < t_N = t_f\}\,. \tag{7.9}$$

Die t_i können äquidistant oder auch unregelmäßig verteilt sein. Für die Intervallbreite ergibt sich je nach Wahl:

$$h_{i-1} := t_i - t_{i-1}, \quad i = 1, \ldots, N, \tag{7.10}$$

$$\text{bzw. } h_{i-1} = h := \frac{t_f - t_0}{N}, \quad i = 1, \ldots, N. \tag{7.11}$$

Bei einem äquidistanten Gitter läßt sich jedes Intervall $[t_{i-1}, t_i]$ angeben:

$$\left[t_0 + (i-1) \cdot \frac{t_f - t_0}{N}, t_0 + i \cdot \frac{t_f - t_0}{N}\right], \quad i = 1, \ldots, N. \tag{7.12}$$

Damit zerfallen die Zielfunktion und die Nebenbedingungen in N Abschnitte. Im weiteren Verlauf wird nun darauf hingearbeitet, Zustand und Steuerung in jedem Intervall polynomial zu approximieren. Für die Zielfunktion läßt sich die Intervallzerlegung folgendermaßen andeuten:

$$S(t_N, x(t_N)) + \sum_{i=1}^{N} \int_{t_{i-1}}^{t_i} f\big(t, x(t), u(t)\big)\, dt. \tag{7.13}$$

Ebenso verfährt man mit den Nebenbedingungen. Zu berücksichtigen ist hierbei, daß die Abschnitte dennoch zusammenhängen, da der Zustand als stetig vorausgesetzt wird. So ist ist bei der Formlierung

$$\dot{x}(t) = g\big(t, x(t), u(t)\big), \quad t \in [t_{i-1}, t_i], \quad i = 1, \ldots, N \tag{7.14}$$

darauf zu achten, daß der Wert des Zustandes am Ende jedes Intervalls $[t_{i-1}, t_i]$ mit demjenigen am Anfang des folgenden Intervalls $[t_i, t_{i+1}]$ übereinstimmt. Damit

sind die Intervalle auf natürliche Weise gekoppelt. Die Stetigkeit des Zustandes ist gewährleistet.
Es ist frei wählbar, ob man eine solche Kopplung für die Steuerung fordert. Eine Entkopplung kann eine Unstetigkeit simulieren. Unstetigkeitsstellen sind a priori jedoch unbekannt. Bei einer Vermutung über die Lage von Sprungstellen kann eine passende Intervallzerlegung mit entkoppelter Steuerung erfolgversprechend sein.
Jetzt erfolgen die Ansätze des Abschnitts 7.2. Zustand und Steuerung werden stückweise als Polynome des Grades $s-1$ mit s Koeffizienten angesetzt. Mit x^i und u^i werden jetzt die Polynome bezeichnet, wobei der Index i im folgenden die Intervallabschnitte markiert:

$$x(t) \approx x^i(t), \quad u(t) \approx u^i(t), \quad t \in [t_{i-1}, t_i], \quad i = 1, \ldots, N. \tag{7.15}$$

In jedem Intervall $[t_{i-1}, t_i]$ werden s Stützstellen ausgesucht. Die Lage der Stützstellen wird durch die auf $[0,1]$ normierten Werten c_j, $j = 1, \ldots, s$ gekennzeichnet:

$$t_{i-1} + c_j h_{i-1}, \quad j = 1, \ldots, s, \quad i = 1, \ldots, N. \tag{7.16}$$

Damit entstehen für jedes Intervall $[t_{i-1}, t_i]$ die folgenden Kollokationsbedingungen (vgl. Definition 5.1):

$$\begin{aligned} \dot{x^i}(t_{i-1} + c_j h_{i-1}) &= g\big(t_{i-1} + c_j h_{i-1}, x^i(t_{i-1} + c_j h_{i-1}), u^i(t_{i-1} + c_j h_{i-1})\big), \\ x^i(t_{i-1}) &= x^{i-1}(t_{i-1}), \\ & \qquad j = 1, \ldots, s, \quad i = 1, \ldots, N. \end{aligned}$$

Selbstverständlich müssen $x^1(t_0) = x_0$ und $x^N(t_N) = x_f$ gelten. Die Steuerung muß an allen Punkten $t_{i-1} + c_j h_{i-1}$ die Steuerbeschränkungen einhalten.
In Kapitel 6.5 wurde erläutert, wie mit der Zielfunktion zu verfahren ist, nachdem die Polynome x^i und u^i substituiert wurden. Das Integral muß mit der dem Kollokationsverfahren entsprechenden Quadraturregel approximiert werden. Insgesamt entsteht ein Optimierungsproblem der Definition 6.18.

Problem 7.4. *α_j, $j = 1, \ldots, s$ sind die Gewichte der Integrationsregel. Die $2 \cdot s \cdot N$ reellen Koeffizienten der Polynome*

$$\begin{aligned} & x^i, \quad i = 1, \ldots, N, \\ & u^i, \quad i = 1, \ldots, N \end{aligned}$$

bilden die Argumente der folgenden Aufgabe:
Minimiere

$$S(t_N, x^N(t_N)) + \sum_{i=1}^{N} \left(\sum_{j=1}^{s} \alpha_j f\big(t_{i-1} + c_j h_{i-1}, x^i(t_{i-1} + c_j h_{i-1}), u^i(t_{i-1} + c_j h_{i-1})\big) \right)$$

unter den Nebenbedingungen

$$\begin{aligned}
\dot{x^i}(t_{i-1}+c_jh_{i-1}) &= g\big(t_{i-1}+c_jh_{i-1}, x^i(t_{i-1}+c_jh_{i-1}), u^i(t_{i-1}+c_jh_{i-1})\big),\\
&\quad j=1,\ldots,s, \quad i=1,\ldots,N,\\
x^i(t_{i-1}) &= x^{i-1}(t_{i-1}), \quad i=2,\ldots,N,\\
x^1(t_0) &= x_0,\\
x^N(t_N) &= x_f,\\
u^i(t_{i-1}+c_jh_{i-1}) &\in U, \quad j=1,\ldots,s, \quad i=1,\ldots,N,\\
\Big(u^i(t_{i-1}) &= u^{i-1}(t_{i-1}), \quad i=2,\ldots,N \ \text{(nach Wahl)}\Big).
\end{aligned}$$

Im nächsten Abschnitt wird der Kollokationsansatz auf Linear-Quadratische Optimalsteuerungsprobleme spezialisiert.

7.4 Ansatz für Linear-Quadratische Optimalsteuerungsprobleme

Der Abschnitt 7.3 erläuterte den Kollokationspolynomansatz für eher allgemeine Problemstellungen. Jetzt wird dieser Ansatz auf die Linear-Quadratischen Aufgabenstellungen des Abschnitts 6.4 spezialisiert. Später werden auch zusätzliche Nebenbedingungen hinzugefügt. Es werden die Lobatto-Knoten als Stützstellen verwendet, wie in Abschnitt 7.2 erwähnt. Da Gauß-Lobatto-Nullstellen üblicherweise auf das Intervall $[-1,1]$ standardisiert sind, wird dieses Intervall jetzt auch verwendet. Die Aufgabenstellung lautet nochmals:

Problem 7.5. *Minimiere*

$$\begin{aligned}
J(x,u) = &\ \frac{1}{2}x(t_f)^T S\, x(t_f) + l^T x(t_f)\\
&+\frac{1}{2}\int_{t_0}^{t_f}\Big(x(t)^T Q(t)\, x(t) + u(t)^T R(t)\, u(t) + 2h(t)^T x(t) + 2k(t)^T u(t)\Big)\, dt
\end{aligned}$$

unter den Nebenbedingungen

$$\begin{aligned}
\dot{x}(t) &= A(t)x(t) + B(t)u(t),\\
x(t_0) &= x_0
\end{aligned}$$

mit den Matrizen und Vektoren aus der Definition 6.15.

Nachdrücklich wird hier nochmals auf die Bemerkungen 6.12, 6.16 und die Diskussion im Abschnitt 7.2 hingewiesen. Die Aufgabenstellung eignet sich für einen Polynomansatz. Das Hinzufügen von Steuerbeschränkungen oder einem Endzustand $x(t_f) = x_f$ ändert die Ausgangslage allerdings maßgeblich Doch sei vorerst von diesen Restriktionen abgesehen. Es wird ferner von einem äquidistanten Gitter $\mathbb{G}_N$ ausgegangen.
Um den Lobatto-Nullstellen des Intervalls $[-1, 1]$ zu genügen, muß jedes Intervall $[t_{i-1}, t_i]$ ebenfalls auf $[-1, 1]$ transformiert werden. Im letzten Abschnitt wurde dieses Intervall noch nicht verwendet. Insofern besitzen die mit c_j bezeichneten Lobatto-Knoten in diesem Abschnitt eine andere Lage als im letzten Abschnitt. Die allgemeine Transformationsformel der Bemerkung 2.11 lautet beim äquidistanten Gitter (7.12):

$$t(i) := \frac{t_f - t_0}{2N} \cdot \tau + t_0 + \frac{(2i-1)(t_f - t_0)}{2N}, \quad i = 1, \ldots, N. \tag{7.17}$$

Die Transformation unterscheidet sich für jedes Intervall $[t_{i-1}, t_i]$ durch die Abhängigkeit von dem Intervallindex i. Die Zeit τ läuft stets im Intervall $[-1, 1]$. Die Vorfaktoren $\frac{t_f - t_0}{2N}$ der Substitution sind dagegen für jedes Intervall identisch. Nun kann man für jedes Intervall die Polynome bilden.

Notation 7.6. *Der Zustand x habe die Dimension n, die Steuerung u die Dimension m. Dann werden in jedem Intervall Polynome angesetzt:*

$$x_k^i(\tau) = \sum_{j=1}^{s} a_{k,j}^i \cdot l_j(\tau), \quad k = 1, \ldots, n, \quad i = 1, \ldots, N, \tag{7.18}$$

$$u_k^i(\tau) = \sum_{j=1}^{s} b_{k,j}^i \cdot l_j(\tau), \quad k = 1, \ldots, m, \quad i = 1, \ldots, N, \tag{7.19}$$

$$l_j(\tau) = \prod_{\substack{l=1 \\ l \neq j}}^{s} \frac{\tau - c_l}{c_j - c_l}, \quad j = 1, \ldots, s. \tag{7.20}$$

Bei den Koeffizienten $a_{k,j}^i$ und $b_{k,j}^i$ weist der obere Index auf den Intervallabschnitt und der linke untere auf die Dimension hin, während der rechte untere die Polynomkoeffizienten markiert.

Man kann leicht nachrechnen, daß mehr Variablen als Nebenbedingungen vorliegen. Die Lagrange-Polynome erbringen vor allem zwei gewichtige Vorteile:

1. Die Variablen der Optimierungsaufgabe sind die Koeffizienten der Lagrange-Polynome. Da aber $x_k^i(c_l) = a_{k,l}^i$ und $u_k^i(c_l) = b_{k,l}^i$ gelten, sind die gesuchten Variablen dann bereits die Werte des Zustands und der Steuerung an den entsprechenden Stellen. Es muß keine weitere Umrechnung erfolgen.

2. Die Eigenschaft $x_k^i(c_l) = a_{k,l}^i$ bzw. $u_k^i(c_l) = b_{k,l}^i$ verringert die Anzahl der Variablen im entstehenden Quadratischen Programm.

Diese Vorzüge werden später noch deutlich, so in (7.23).

Approximation des Zielfunktionals. Man setzt die Vektoren x^i und u^i in das Kostenfunktional ein. Zum Einsparen von Platz wird gezielt die Abkürzung der Formel (7.17) verwendet.

$$\begin{aligned} & J(x,u) \approx J(x^i,u^i) \\ = \;& \frac{1}{2}x^N(t_N)^T S\, x^N(t_N) + l^T x^N(t_N) \\ + \;& \frac{1}{2}\cdot\frac{t_f-t_0}{2N}\sum_{i=1}^{N}\int_{t_{i-1}}^{t_i}\Big(x^i(t(i))^T Q(t(i))\, x^i(t(i)) + u^i(t(i))^T R(t(i))\, u^i(t(i))\Big) \\ + \;& \Big(2h(t(i))^T x^i(t(i)) + 2k(t(i))^T u^i(t(i))\Big)\, d\tau. \end{aligned}$$

Es wird die Schreibweise

$$Q^i(\tau) := Q(t(i)) = Q\left(\frac{t_f-t_0}{2N}\cdot\tau + t_0 + \frac{(2i-1)(t_f-t_0)}{2N}\right) \tag{7.21}$$

eingeführt und auch bei anderen Matrizen und Vektoren verwendet. Für die Matrizen und Vektoren der Aufgabenstellung werden im folgenden beispielsweise mit $Q_{j,k}^i$ die Matrixeinträge der Matrix Q^i bezeichnet, wobei wie üblich der linke untere Index für die Zeile und der rechte für die Spalte steht. Jetzt wird das Integral mit der Gauß-Lobatto-Regel approximiert. Die Matrixprodukte werden durch Summen ersetzt. Die Gewichte α_j gewinnt man mit dem Satz 2.25.

$$J(x^i,u^i) \approx J_s^N = \frac{1}{2}\sum_{\mu,\nu=1}^{n} S_{\mu,\nu}a_{\mu,s}^N a_{\nu,s}^N + \sum_{\mu=1}^{n} l_\mu a_{\mu,s}^N + \frac{t_f-t_0}{4N}\sum_{i=1}^{N}\sum_{j=1}^{s}\alpha_j$$
$$\left(\sum_{\mu,\nu=1}^{n} Q_{\mu,\nu}^i(c_j)a_{\mu,j}^i a_{\nu,j}^i + \sum_{\mu,\nu=1}^{m} R_{\mu,\nu}^i(c_j)b_{\mu,j}^i b_{\nu,j}^i + 2\sum_{\mu=1}^{n} h_\mu^i(c_j)a_{\mu,j}^i + 2\sum_{\mu=1}^{m} k_\mu^i(c_j)b_{\mu,j}^i\right)$$

Approximation der Differentialgleichung. Für die linke Seite $\dot{x}$ der Differentialgleichung gilt nach Satz 2.30:

$$\begin{aligned} \dot{x}_k^i(c_j) \;&=\; \frac{d}{d\tau}\Big(\sum_{\mu=1}^{s} a_{k,\mu}^i\cdot l_\mu(\tau)\Big)\Big|_{\tau=c_j} \;=\; \sum_{\mu=1}^{s} D_{j,\mu}a_{k,\mu}^i, \qquad (7.22)\\ & k=1,\ldots,n,\quad j=1,\ldots,s,\quad i=1,\ldots,N. \end{aligned}$$

Auf der rechten Seiten wirken sich die Lagrange-Polynome in der Kombination mit der linearen Differentialgleichung vereinfachend aus. Anstelle der Funktionen x

und u entstehen lediglich die Vorfaktoren der Polynome. Mit der Zeittransformation $\frac{t_f - t_0}{2N}$ entsteht rechts der folgende Term:

$$\frac{t_f - t_0}{2N} \cdot \left(\sum_{\mu=1}^{n} A^i_{k,\mu}(c_j) a^i_{\mu,j} + \sum_{\mu=1}^{m} B^i_{k,\mu}(c_j) b^i_{\mu,j} \right), \tag{7.23}$$
$$k = 1, \ldots, n, \quad j = 1, \ldots, s, \quad i = 1, \ldots, N.$$

Nun führt man beide Seiten der Differentialgleichung auf eine Seite zusammen. Dann lauten die Kollokationsbedingungen:

$$0 = \sum_{\mu=1}^{s} D_{j,\mu} a^i_{k,\mu} - \frac{t_f - t_0}{2N} \cdot \left(\sum_{\mu=1}^{n} A^i_{k,\mu}(c_j) a^i_{\mu,j} + \sum_{\mu=1}^{m} B^i_{k,\mu}(c_j) b^i_{\mu,j} \right), \tag{7.24}$$
$$k = 1, \ldots, n, \quad j = 1, \ldots, s, \quad i = 1, \ldots, N.$$

Ferner werden die Polynome intervallweise gekoppelt: für den Zustand x einerseits

$$a^i_{k,1} - a^{i-1}_{k,s} = 0, \quad k = 1, \ldots, n, \quad i = 2, \ldots, N, \tag{7.25}$$

für die Steuerung andererseits

$$b^i_{k,1} - b^{i-1}_{k,s} = 0, \quad k = 1, \ldots, m \quad i = 2, \ldots, N. \tag{7.26}$$

Dazu gilt natürlich die Anfangsbedingung:

$$a^1_{k,1} - (x_0)_k = 0, \quad k = 1, \ldots, n. \tag{7.27}$$

Letztere Gleichungen liefern bereits die Werte der Variablen $a^1_{k,1}$, $k = 1, \ldots, n$. Man könnte sie in die anderen Gleichungen substituieren und damit die Anzahl der Variablen verringern, womit jedoch keine bedeutende Problemverkleinerung erreicht wird. Folgende Beobachtung ist betonungswert:

Bemerkung 7.7. *Alle Nebenbedingungen (7.24), (7.25), (7.26) und (7.27) sind linear.*

Das Optimierungsproblem kann jetzt zusammengefaßt werden.

Problem 7.8. *Gegeben sei die Aufgabenstellung der Definition 6.15. Mit dem Lösungsansatz der Abschnitte 7.2 und 7.3 sowie den Bezeichnungen dieses Abschnitts erhält man dann das folgende Quadratische Programm:*
Minimiere

$$\frac{1}{2} \sum_{\mu,\nu=1}^{n} S_{\mu,\nu} a^N_{\mu,s} a^N_{\nu,s} + \sum_{\mu=1}^{n} l_\mu a^N_{\mu,s} + \frac{t_f - t_0}{4N} \sum_{i=1}^{N} \sum_{j=1}^{s} \alpha_j \Bigg(\sum_{\mu,\nu=1}^{n} Q^i_{\mu,\nu}(c_j) a^i_{\mu,j} a^i_{\nu,j}$$
$$+ \sum_{\mu,\nu=1}^{m} R^i_{\mu,\nu}(c_j) b^i_{\mu,j} b^i_{\nu,j} + 2 \sum_{\mu=1}^{n} h^i_\mu(c_j) a^i_{\mu,j} + 2 \sum_{\mu=1}^{m} k^i_\mu(c_j) b^i_{\mu,j} \Bigg)$$

unter den Nebenbedingungen

$$\begin{aligned}
0 &= \sum_{\mu=1}^{s} D_{j,\mu} a_{k,\mu}^i - \frac{t_f - t_0}{2N} \cdot \left(\sum_{\mu=1}^{n} A_{k,\mu}^i(c_j) a_{\mu,j}^i + \sum_{\mu=1}^{m} B_{k,\mu}^i(c_j) b_{\mu,j}^i \right), \\
&\quad k = 1, \ldots, n, \quad j = 1, \ldots, s, \quad i = 1, \ldots, N, \\
0 &= a_{k,1}^i - a_{k,s}^{i-1}, \quad k = 1, \ldots, n, \quad i = 2, \ldots, N, \\
0 &= b_{k,1}^i - b_{k,s}^{i-1}, \quad k = 1, \ldots, m, \quad i = 2, \ldots, N, \\
0 &= a_{k,1}^1 - (x_0)_k, \quad k = 1, \ldots, n, \\
a_{k,j}^i &\in \mathbb{R}, \quad k = 1, \ldots, n, \quad j = 1, \ldots, s, \quad i = 1, \ldots, N, \\
b_{k,j}^i &\in \mathbb{R}, \quad k = 1, \ldots, m, \quad j = 1, \ldots, s, \quad i = 1, \ldots, N.
\end{aligned}$$

Für die numerische Bearbeitung ist die folgende Beobachtung bedeutsam.

Bemerkung 7.9. *Formuliert man die Nebenbedingungen (7.24), (7.25), (7.26) und (7.27) in Matrixschreibweisen, entstehen dünn besetzte Matrizen.*

Die Summen in Problem 7.8 entstehen zum Teil durch Matrixprodukte. In [19] werden diese Summen durch Tensorprodukte ersetzt. Damit vermeidet einige Indizes. Für die Implementierung in Kapitel 8 ist die Indexschreibweise dagegen sinnvoller.
Nun wird die Problemstellung um Steuerbeschränkungen und Randwerte ergänzt. Ein Randwert ist das Gegenstück zur Anfangsbedingung.

Definition 7.10. *Eine Randbedingung zur Aufgabe 7.5 ist durch*

$$\begin{aligned}
x(t_f) &= x_f \\
\text{bzw.} \quad x_k(t_f) &= (x_f)_k, \quad k = 1, \ldots, n
\end{aligned}$$

gegeben.

Diese Randbedingung läßt sich sehr einfach in die Aufgabe 7.8 integrieren:

$$a_{k,s}^N - (x_f)_k = 0, \quad k = 1, \ldots, n. \tag{7.28}$$

Wie bei der Anfangsbedingung lassen sich die Lösungen dieser Gleichungen sofort ablesen.
Zwei verschiedene Typen von Steuerbeschränkungen werden für die Aufgabe wahlweise zugelassen.

Definition 7.11. *`Box-Constraints` begrenzen mit unteren und oberen Schranken jede Steuerungsdimension u_k:*

$$(u_l)_k \leq u_k \leq (u_u)_k, \quad (u_l)_k, (u_u)_k \in \mathbb{R}, \quad k = 1, \ldots, m.$$

`Kugel-Steuerbeschränkungen` stellen Forderungen an die euklidische Norm der Steuerbeschränkungen:

$$\begin{aligned} \|u\|_2 &\leq u_u, \\ \Longrightarrow \sum_{k=1}^{m} u_k^2 &\leq u_u^2. \end{aligned}$$

Die oberen und unteren Schranken sind oftmals dimensionsunabhängig. Die Box-Constraints lassen sich sehr einfach in das Optimierungsproblem einfügen. Man fordert einfach

$$(u_l)_k \leq b_{k,j}^i \leq (u_u)_k, \quad k = 1, \ldots, m, \quad j = 1, \ldots, s, \quad i = 1, \ldots, N. \tag{7.29}$$

Die Kugel-Steuerbeschränkungen müssen zu jedem Zeitpunkt der diskreten Zeit gelten:

$$\sum_{k=1}^{m} \left(b_{k,j}^i\right)^2 \leq u_u^2, \quad j = 1, \ldots, s, \quad i = 1, \ldots, N. \tag{7.30}$$

Diese Ungleichungen sind nicht mehr linear, was aber für die Bezeichnung eines Quadratischen Programms hier ignoriert wird. Zusammengefaßt ergibt sich:

Problem 7.12. *Ein erweitertes Quadratisches Programm entsteht durch das Problem 7.8 und die Nebenbedingungen der Definitionen 7.10 und 7.11.*

Die Anzahl der Variablen der Optimierungsproblems 7.8 und 7.12 wachsen besonders mit der Diskretisierung N. Nach Bemerkung 6.16 kann man jedoch fordern:

Bemerkung 7.13. *Für Probleme des Typs 7.8 wählt man nach Bemerkung 6.16 bevorzugt $N = 1$.*

Beispielrechnungen folgen im nächsten Kapitel 8. Ist man vorerst nicht an der Software-Beschreibung und Implementierung interessiert, kann man direkt in den Abschnitt 8.4 springen.

Kapitel 8

Numerische Beispiele

8.1 Einleitung

In diesem Kapitel 8 werden Beispiele zu der Aufgabenstellung 7.12 demonstriert. Dem Diskurs der Beispiele muß eingangs ein Einblick in die verwendete Software vorausgehen.

Im Abschnitt 8.2 wird die verwendete Software erläutert.
Im Abschnitt 8.3 wird die Anpassung der Software auf die Aufgabenstellung 7.12 beschrieben. Hierbei wird auch die Bedienung des Quellcodes erklärt.
Die Beispiele werden dann im Abschnitt 8.4 vorgestellt.
Im Abschnitt 8.5 werden Beobachtungen und Schlußfolgerungen zu den Beispielen des Abschnitts 8.4 zusammengefaßt.
Der Anhang A ergänzt das laufende Kapitel, beispielsweise mit ausgiebigeren Quellcode-Informationen. Hier werden auch ein paar Beispieldaten ausgegliedert.
Der Anhang B beschreibt die beiliegende CD.

8.2 Einblicke in SCPIP 3.0

Für die Beispiele des Abschnitts 8.4 wurde das Programm SCPIP 3.0 verwendet. SCPIP 3.0 ist ein Fortran77-Code zur Lösung nichtlinearer Optimierungsprobleme, der von Professor Christian Zillober geschrieben wurde. Ein Manual findet man unter [64]. Die Programmanwendung läßt sich in die allgemeine Definition 6.18 der nichtlinearen Optimierungsprobleme einordnen. Somit kann SCPIP 3.0 für die in Kapitel 7 konstruierte Aufgabenstellung zur Lösung von Optimalsteuerungsproblemen eingesetzt werden. Die Aufgabenstellung von SCPIP 3.0 ist in [64, Einleitung] formuliert. In diesem Abschnitt werden die Originalbezeichnungen von SCPIP 3.0 verwendet.

Problem 8.1 (Aufgabenstellung von SCPIP 3.0). *Es sei* $i = 1, \ldots, m_{mie}$, $j = 1, \ldots, m_{meq}$ *und* $k = 1, \ldots, n$. *Die Menge*

$$X := \{x |\ \underline{x}_k \leq x_k \leq \overline{x}_k,\ k = 1, \ldots, n\}$$

sei nichtleer. Die Funktionen f, h_i *und* g_j *seien auf* X *definiert und im Inneren von* X *mindestens zweimal stetig differenzierbar. Dann besteht die Aufgabenstellung von SCPIP 3.0 aus dem folgenden Optimierungsproblem:*
Minimiere

$$f(x)$$

unter den Nebenbedingungen

$$\begin{aligned} h_i(x) &\leq 0, \quad i = 1, \ldots, m_{mie}, \\ g_j(x) &= 0, \quad j = 1, \ldots, m_{meq}, \\ \underline{x}_k \leq\ x_k &\leq \overline{x}_k, \quad k = 1, \ldots, n, \\ x &\in \mathbb{R}^n. \end{aligned}$$

Die zulässige Menge sei nichtleer.

Der Quellcode von SCPIP 3.0 setzt sich aus vier Fortran77-Dateien zusammen:

- *v30main.f*
- *Scpip30.f*
- *scpblas.f*
- *Sparsecz.f*

Für den Anwender ist die kurze Datei *v30main.f* entscheidend, da sie die Bedienungselemente enthält. In den letzteren drei großen Dateien ist dagegen die mathematische Optimierungstechnik implementiert.
Der Nutzer hat in der Datei *v30.main* im wesentlichen nachstehende Elemente zu programmieren (Originalbezeichnungen aus dem Quellcode):

- die Zielfunktion f_org sowie den zugehörigen Gradienten,
- die Ungleichungsnebenbedingungen $h\_org(i)$, $i = 1, \ldots, m_{mie}$ sowie die zugehörigen Gradienten,
- die Gleichungsnebenbedingungen $g\_org(j)$, $j = 1, \ldots, m_{meq}$ sowie die zugehörigen Gradienten,
- Box-constraints für die Variablen,
- den Startwert $x0(k)$, $k = 1, \ldots, n$ für den Optimierungs-Algorithmus.

Zuzüglich müssen je nach Problemgröße gewisse Parameter justiert werden, zum Beispiel Obergrenzen für die Anzahl der Variablen oder Nebenbedingungen. Das Fehlerarray *IERR* dient bei Kompilierung als Kontrollinstanz. Neben diesen notwendigen Bedienungselementen können aber noch andere Einstellungen vorgenommen werden. So besteht die Auswahl zwischen mehreren Optimierungstechniken. Dazu gehört beispielsweise der Integer-Array-Eintrag *icntl(1)*, wobei *icntl(1)=1* für die Vorgehensweise der *„method of moving asymptotes“* und *icntl(1)=2* für *„sequential convex programming“* steht. Zusätzlich können weitere Feineinstellungen in *v30main.f* vom Nutzer manuell verstellt werden, um eine problemspezifische Anpassung herzustellen. Mit dem Double-Precision-Array *rcntl(1)* läßt sich etwa eine Obergrenze für die erlaubte Abweichung einer Verletzung einer Nebenbedingung festlegen. Die Existenz von *rcntl(1)* unterstreicht übrigens die Diskrepanz zwischen der mathematischen Formulierung eines Optimierungsproblems und Problematiken beim numerischen Lösen.
Bei den Beispielen des Abschnitts 8.4 konnten diese Feinheiten aber nicht berücksichtigt werden. Stattdessen wurde eine Einstellung nach erfolgreichen Erprobungen festgelegt. Damit wird zumindest auch eine gewisse Gleichwertigkeit der Beispielergebnisse garantiert. Die verwendeten Einstellungen sind im Anhang A.1 festgehalten. Das Manual [64] erläutert die Bedeutungen.
SCPIP 3.0 ist völlig allgemein gehalten. Es wird dem Nutzer überlassen, einen individuellen, problemspezifischen Quellcode zu schreiben, der die Funktionen und Gradienten der erwünschten Aufgabe generiert.
SCPIP 3.0 wird über die Datei *v30main.f* bearbeitet und bedient. Die *v30main.f*-Datei enthält als wesentliche Bedienungsroutinen:

1. Der *main*-Kopf enthält neben den Box-constraints und dem Startwert allgemeine Optimierungseinstellungen. Dieser Bereich ist erst nach Fertigstellung der beiden folgenden Subroutinen bedeutsam.

2. Die Subroutine *scpfct* enthält die Zielfunktion und die Ungleichungs- und Gleichungsnebenbedingungen.

3. Die Subroutine *scpgrd* enthält die Gradienten aller unter 2. definierten Funktionen.

Die folgenden Quellcode-Ausschnitte der beiden zentralen Subroutinen schematisiert die Arbeit, die man zu leisten hat. Die Subroutine *scpfct* hat prinzipiell die folgende Gestalt:

```
************************************************************************
      subroutine scpfct (n,mie,meq,x,f_org,g_org,h_org)
```

```
c     input:  n,mie,meq,x
c     output: f_org,g_org,h_org

      integer          n,mie,meq
      double precision f_org,g_org(*),h_org(*),x(*)

c     Zielfunktion
      f_org = ...

c     Ungleichungsnebenbedingungen
      h_org(i) =  ...

c     Gleichungsnebenbedingungen
      g_org(i) =  ...

      end
*****************************************************************
```

Die Subroutine *scpgrd* der Gradienten ist komplexer. Zum einen resultiert dies aus der unterschiedlichen Behandlung der Gradienten von Gleichungs- und Ungleichungsrestriktionen. Zum anderen erfordert die Abspeicherung der Gradienten der Funktionen der Subroutine *scpfct* große Sorgfalt wegen der notwendigen Beachtung der korrekten Anordnung. Der folgende Quellcode-Auszug ist erneut als Strukturabbild zu verstehen.

```
*****************************************************************
      subroutine scpgrd (n,mie,meq,x,f_org,g_org,h_org,active,df,
     +                   iern,iecn,iederv,ieleng,eqrn,eqcn,eqcoef,
     +                   eqleng)
      ...

c     Gradient der Zielfunktion f_org
      df(i) = ...

c     Zähler für Funktionsgradienten und Komponenten
      ieleng = 0
      eqleng = 0
      numbercon = 0
      numbercon2 = 0

c     Ableitung der Ungleichungsbedingung i nach Komponente j
```

```
      if (active(i) .eq. 1) then
         numbercon=numbercon+1
         ieleng = ieleng + 1
         iecn(ieleng) = numbercon          ! Funktionsnummer
         iern(ieleng) = j                  ! Komponentennummer
         iederv(ieleng) = ...              ! Ableitung
         .... (weitere Komponenten)
      endif

c     Ableitung einer Gleichungsbedingung nach Komponente j
         numbercon2=numbercon2+1
         eqleng = eqleng + 1
         eqcn(eqleng) = numbercon2         ! Funktionsnummer
         eqrn(eqleng) = j                  ! Komponentennummer
         eqcoef(eqleng) = ...              ! Ableitung
         .... (weitere Komponenten)

      end
*****************************************************************
```

Das Kompilieren des Quellcodes erzeugt die Ausgabedatei *SCPERG.txt*. Bei Fehlern oder anderen Unzulänglichkeiten des Programms verweist sie auf Ursachen. Bei erfolgreichem Ablauf werden hier unter anderem der Zielfunktionswert und die finalen Variablenbelegungen ausgegeben. Die Ausgabeinformationen der Datei *SCPERG.txt* sind in der Datei gründlich erklärt, so daß sich eine weitere Beschreibung erübrigt.
Erwähnenswert ist noch, daß die Variablen in einem Vektor abgelegt werden. Es ist deshalb besonders überlegenswert, wie man die Unbekannten eines Optimalsteuerungsproblems innerhalb dieses Vektors anordnet.

8.3 Software-Bedienung und -Analyse

Der Abschnitt 8.2 informierte über die wesentlichen Bedienungselemente von SCPIP 3.0. Das Ziel dieses Abschnitts ist eine Vermittlung der Anpassung des Quellcodes auf die Problemstellung 7.12.
Das Optimierungsaufgabe 7.12 ist letztendlich ein Quadratisches Programm der Definition 6.19. Die Übertragung dieser speziellen Aufgabe in den Quellcode von SCPIP 3.0 führt zur Formulierung sämtlicher Funktionen und Nebenbedingungen, aber auch zur notwendigen Implementierung von Hilfsdaten. Zu den Daten gehören die Nullstellen und Gewichte der Gauß-Lobatto-Regeln. Insgesamt gelangen zahlreiche Daten und Parameter in den Quellcode. Es ist deshalb ratsam,

weitere Fortran-Dateien zur Ergänzung der bisherigen anzulegen, um Teile der Aufgaben auszugliedern. So entlastet man die Datei *v30main.f*. Gleichzeitig kann man die verschiedenen Daten inhaltlich trennen.
Zwei neue Dateien wurden angelegt. Die Datei *Aufgabendaten.f* beinhaltet die Daten der Problemstellung, einschließlich der gewählten Optimierungsgrößen wie etwa den Quadraturgrad. Diese Datei ist die Bedienungskonsole des Programms. Die Datei *Hilfsfunktionen.f* enthält Daten und Hilfsfunktionen für die technische Durchführung der Optimierungsaufgabe. Die Tabelle 8.1 faßt die essentiellen Zwecke und Inhalte der Dateien zusammen.

Neu erstellte Dateien			
Aufgabendaten.f		*Hilfsfunktionen.f*	
Zweck:	Enthält Aufgabendaten	Zweck:	Enthält Hilfskonstrukte
Inhalt:	1) Dimensionen, Quadraturgrad, Intervallzerlegung 2) Matrizen und Vektoren des Problems 7.12 3) Steuerbeschränkungen und Randbedingungen	Inhalt:	1) Gewichte und Nullstellen der Lobatto-Regeln 2) Intervalltransformation mit Substitutionsfaktor 3) Hilfsfunktionen: Legendre-Polynome und ihre Ableitungen
$\longrightarrow$ Hauptbedienungselement		$\longrightarrow$ Arbeitet im Hintergrund	

Tabelle 8.1: Überblick über die neuerstellten Dateien

Der Subroutinen sollten effizient programmiert und zugleich bedienungsfreundlich sein. Es wurde versucht, zusammengehörige Elemente in Subroutinen zu vereinigen und unterschiedliche zu separieren. So sind die Angaben des Integrals in einer Subroutine vereint und getrennt von den Daten der Differentialgleichung. In A.2 ist eine genauere Beschreibung des Quellcodes tabelliert.
Weiter muß entschieden werden, ob gewisse Daten im Quellcode aufgelistet oder numerisch berechnet werden. Exemplarisches Beispiel dafür sind die Legendre-Polynome und die Lobatto-Gewichte. Für erstere könnte man die Formel (2.15) verwenden, für letztere den Satz 2.25. Es wurden folgende Entscheidungen gefällt:

- Die Nullstellen müssen implementiert werden.
- Die Gewichte wurden als konstante Vektoren gespeichert, und nicht im SCPIP 3.0-Code mit Satz 2.25 berechnet.

- Die Legendre-Polynome wurden in der Form (2.17) gespeichert. Mit Maple kann man die Koeffizienten leicht ermitteln, siehe A.3.
- Die Matrix D wird in SCPIP 3.0 berechnet.

Nun müssen die Daten mit einer ausreichenden Genauigkeit ermittelt werden. Maschinengenauigkeit ist wünschenswert. Die Literatur (wie [1, Kapitel 25]) ist für die Lobatto-Verfahren nicht unbedingt gerüstet, da stets mehr die Gauß-Legendre-Verfahren betont werden. Eine Vorberechnung mittels mathematischer Programme ist erforderlich, siehe hierzu A.3. Für das Beispiel 8.2 wird getestet, wie sich unterschiedliche Nachkommastellenangaben auf eine numerische Lösung auswirken. Für den Quellcode wurde zur Verfügung gestellt:

Auswahlmöglichkeiten für den Polynomgrad $s-1$: $\quad 3 \leq s \leq 12$

Mit $s = 3$ wird in Aufgabenstellung 7.12 quadratisch interpoliert. $s = 12$ ermöglicht den maximalen Polynomgrad 11. Höhere Polynomgrade erscheinen nicht gerechtfertigt, da solchen Ansätzen entsprechende Glattheitseigenschaften gegenüberstehen müßten.
Es wurden zusätzlich zwei neue Ausgabedateien geschrieben: *SCPERG2.txt* und *SCPERG3.txt*. Erstere ermöglicht einen problemspezifischen Überblick, indem nunmehr eine Beschriftung der Variablen erfolgt. *SCPERG3.txt* ist auf das Erstellen von Graphiken spezialisiert.

Neu erstellte Ausgabedateien		
	SCPERG2.txt	*SCPERG3.txt*
Zweck und Darstellung	Problemgerechter Überblick: Kenntlichmachung der Variablen bezüglich Zustand und Steuerung, Dimension und Intervallsegment	Ausgabeformat für Mapleplots, siehe A.4

Tabelle 8.2: Charakterisierung der neuen Ausgabedateien

Die Graphiken im Abschnitt 8.4 wurden mit Maple 9.5 erstellt. In A.4 sind die entsprechenden Befehle für die Ausgabedatei *SCPERG3.txt* aufgeführt.
Interessant ist noch ein Blick auf die Dimensionen des Problems 7.12. Die Beispiele des Abschnitts 8.4 sind niedrigdimensional. Die Intervallanzahl N wird ebenfalls eher klein gehalten. Die Tabelle 8.3 verdeutlicht, daß neben der Variablenanzahl $N \cdot s \cdot (n+m)$ vor allem $N \cdot s \cdot n + N \cdot n$ Gleichungsnebenbedingungen entstehen. Insgesamt wird bei den Beispielen des Abschnitts 8.4 nur mit wenigen hundert Variablen gerechnet, meistens ist die Anzahl noch geringer. Damit treten auch keine längeren Rechenzeiten auf.

Größenordnungen für Variablen und Nebenbedingungen	
Bedeutung	Anzahl
Variablen	$N \cdot s \cdot (n+m)$
Gleichungs-NB	$N \cdot s \cdot n + N \cdot n + n \cdot Endbed + ((N-1) \cdot m) \cdot Koppu$, wobei $Endbed \in \{0,1\}$ einen Randwert der DGL bzw. $Koppu \in \{0,1\}$ eine unstetige Steuerung ermöglicht
Ungleichungs-NB	$N \cdot s \cdot Kugel$, wobei $Kugel \in \{0,1\}$ eine Kugel-Steuerbeschränkung anzeigt

Tabelle 8.3: Problemdimensionen für die Aufgabenstellung 7.12

8.4 Beispiele

In diesem Abschnitt werden Beispiele demonstriert. Zur Lösung der Aufgaben wird der numerische Lösungsvorschlag des Abschnitts 7.4 angesetzt. Die Umsetzung erfolgt mit dem vorgestellten Programm. Dabei entwickeln sich die Beispiele im Laufe des Abschnitts. Die anfänglichen Beispiele besitzen erfolgversprechende Eigenschaften, später werden immer kritischere Restriktionen wie Steuerbeschränkungen und Randbedingungen hinzugefügt. Die späteren Beispiele werden deshalb mehr qualitativ als quantitativ analysiert. Das erste Beispiel wird am Ausführlichsten bearbeitet.

Die graphischen Abbildungen der Beispiele zeigen jeweils die kontinuierlichen Trajektorien, wie sie ein Kollokationsverfahren hervorbringt. Die Implementierung der Problemstellung 7.12 in SCPIP 3.0 ist aber ein finites nichtlineares Optimierungsproblem. In den Abbildungen werden die numerischen Werte an den Kollokationspunkten deshalb zusätzlich mit Punkten markiert. Die Lage der Gauß-Lobatto-Knoten wurde schon in Abbbildung 2.4 am Ende des Abschnitts 2.7 detailliert illustriert.

In den Beispielen wird meistens die Abweichung der numerischen Lösungen von den tatsächlichen Lösungen betrachtet. Die Abweichung wird - wie allgemein üblich - überwiegend betragsmäßig angegeben. Mit „Gitterabweichung“ ist die die Differenz an den Kollokationspunkten gemeint.

Beispiel 8.2. *Das Beispiel stammt aus [19, Beispiel 1].*
Minimiere

$$J(x,u) := \frac{1}{2} \int_0^1 \left(2x^2(t) + u^2(t)\right) dt$$

unter den Nebenbedingungen

$$\begin{aligned} \dot{x}(t) &= -\frac{1}{2}x(t) + u(t), \quad t \in [0,1], \\ x(0) &= 1. \end{aligned}$$

Die analytischen Optimallösungen lauten:

$$\begin{aligned} \hat{x}(t) &= \frac{2e^{-1,5t} + e^{1,5t-3}}{2 + e^{-3}}, \quad t \in [0,1], \\ \hat{u}(t) &= \frac{2e^{1,5t-3} - 2e^{-1,5t}}{2 + e^{-3}}, \quad t \in [0,1], \\ \hat{J}(\hat{x}, \hat{u}) &= \frac{e^3 - 1}{2e^3 + 1} \approx 0.46356665348110519. \end{aligned}$$

Das Beispiel läßt sich der Definition 6.13 des klassischen Linear-Quadratischen Optimalsteuerungsproblems zuordnen:

$$A(t) = -0.5, \;\; B(t) = 1, \;\; Q(t) = 2, \;\; R(t) = 1, \;\; S = 0. \tag{8.1}$$

Es läßt sich leicht die Riccati-Gleichung aus dem Satz 6.14 bilden. Als ein exemplarisches Beispiel für Linear-Quadratische Optimalsteuerungsprobleme lassen sich optimale analytische Lösungen $\hat{x}$ und $\hat{u}$ angeben. Damit ist ein guter Vergleich mit den numerischen Ergebnissen möglich. Gleichwohl ist jetzt schon ersichtlich, daß man die optimalen Lösungen nicht erreichen wird, da der Polynomansatz eine Exponentialfunktion als numerisches Ergebnis ausschließt.
Die Aufgabe muß in das Problem 7.8 umgeformt werden. Dies geschieht auch in [19, Beispiel 1]. Das entstehende Quadratische Programm wird dort aber anders gelöst. Es wird sich zeigen, ob hier ebenbürtige Resultate gewonnen werden (in [19, Beispiel 1] befindet sich ein Schreibfehler in der Steuerung $\hat{u}$).
Der erste Schritt des Lösungsansatzes fordert die Bildung von Polynomen, wobei jedes Polynom in jedem Intervallabschnitt $[t_i, t_{i+1}]$ auf $[-1, 1]$ transformiert wird, siehe Transformation (7.17). Die Rechnungen werden durch die Eindimensionalität von Zustand und Steuerung vereinfacht. Mit den Schreibweisen des Abschnitts 7.4 folgt:

$$\begin{aligned} x_1^i(\tau) &= \sum_{j=1}^{s} a_{1,j}^i \cdot l_j(\tau), \quad i = 1, \ldots, N, \\ u_1^i(\tau) &= \sum_{j=1}^{s} b_{1,j}^i \cdot l_j(\tau), \quad i = 1, \ldots, N, \\ l_j(\tau) &= \prod_{\substack{l=1 \\ l \neq j}}^{s} \frac{\tau - c_l}{c_j - c_l}, \quad j = 1, \ldots, s. \end{aligned} \tag{8.2}$$

Die Knoten c_j sind die Nullstellen der Lobatto-Polynome im Intervall $[-1,1]$, die Zeit τ läuft damit ebenfalls im Intervall $[-1,1]$. Dann erfolgt die Substitution dieser Polynome in die Differentialgleichung und in das Integral. Statt der kontinuierlichen Differentialgleichung werden die Kollokationsbedingungen aufgestellt. Das Integral wird mit der äquivalenten Gauß-Lobatto-Quadraturregel approximiert, wie in Bemerkung 6.20 im Abschnitt 6.5 begründet wurde. Es ensteht folgende Aufgabe:

Minimiere

$$J_s^N := \frac{1}{4N} \sum_{i=1}^{N} \sum_{j=1}^{s} \alpha_j \Big(2(a_{1,j}^i)^2 + (b_{1,j}^i)^2 \Big)$$

unter den Nebenbedingungen

$$\begin{aligned}
\sum_{l=1}^{s} D_{j,l}\, a_{1,l}^i - \frac{1}{2N} \cdot \Big(-\frac{1}{2} a_{1,j}^i + b_{1,j}^i \Big) &= 0, \quad i = 1,\ldots,N, \quad j = 1,\ldots,s, \\
a_{1,1}^i - a_{1,s}^{i-1} &= 0, \quad i = 2,\ldots,N, \\
b_{1,1}^i - b_{1,s}^{i-1} &= 0, \quad i = 2,\ldots,N, \qquad (8.3) \\
a_{1,1}^i - 1 &= 0, \qquad (8.4) \\
a_{1,j}^i,\ b_{1,j}^i &\in \mathbb{R}, \quad i = 1,\ldots,N, \quad j = 1,\ldots,s.
\end{aligned}$$

Es wird zuerst $N = 1$ gesetzt. Es werden verschiedene Polynomgrade $s-1$ getestet. Es werden keine Überlegungen in einen möglichen Lösungsverlauf investiert. Der Optimierer SCPIP 3.0 verwendet darum den Nullvektor als Startwert für die Variablen. Die Tabelle 8.4 faßt die wichtigsten Ergebnisse zusammen. Die erste Spalte enthält mit s die Anzahl der Knoten. Die zweite Spalte gibt die betragsmäßigem Abweichungen der numerischen Zielfunktionswerte vom optimalen Zielfunktionswert an. Die dritte und vierte Spalte enthalten die betragsmäßigen Abweichungen der optimalen Trajektorien $\hat{x}$ und $\hat{u}$ von den numerischen Lösungen am Endzeitpunkt $t = 1$. In der fünften und sechsten Spalte wurde versucht, die maximale Abweichung der numerischen von der optimalen Lösung im gesamten Intervall $[0,1]$ zu bestimmen. Für die Steuerung u lauten bei $s = 3$ in Maple 9.5 die Befehlszeilen und Ausgaben hierfür:

```
>u := t -> (-2*exp(-1.5*t) + 2*exp(-3+1.5*t))/(2 + exp(-3));
>du_3 := evalf(maximize(abs( (0.7278479920e-1*(2.000000000*t-1.)^2+
        .8277072562*t-.8287622561) - (u(t)) ) , t=0..1, location));
 du_3 := 0.1711558500,{[{ t = 1.289439883*10^(-10)}, 0.1711558500]}
```

	Alle Werte als Beträge der Abweichungen				
		Abweichung bei $t = 1$		max. Abweichung in $[0, 1]$	
s	Zielfunktion	Zustand	Steuerung	Zustand	Steuerung
3	$4.876E-03$	$2.649E-04$	$7.173E-02$	$1.662E-02$	$1.712E-01$
4	$6.675E-05$	$1.643E-05$	$1.844E-02$	$1.377E-03$	$2.711E-02$
5	$5.970E-07$	$5.922E-08$	$1.483E-03$	$7.308E-05$	$2.368E-03$
6	$3.145E-09$	$4.166E-08$	$1.755E-04$	$4.589E-06$	$2.228E-04$
7	$1.257E-11$	$1.032E-07$	$8.926E-06$	$1.708E-07$	$1.213E-05$
8	$8.704E-14$	$3.544E-08$	$1.224E-07$	$3.550E-08$	$1.109E-06$
9	$1.499E-14$	$4.907E-08$	$1.427E-08$	$4.910E-08$	$1.122E-06$
10	$3.303E-14$	$8.481E-08$	$5.358E-08$	$8.460E-08$	$1.084E-07$
11	$1.299E-14$	$4.847E-10$	$6.577E-08$	$5.750E-08$	$3.397E-07$
12	$3.603E-14$	$6.904E-08$	$1.763E-07$	$6.920E-08$	$5.088E-07$

Tabelle 8.4: Bsp. 8.2. Ergebnisse für $N = 1$ und variables s

Für größere s nimmt die Genauigkeit offenbar schnell zu. Für den Zustand und den Zielfunktionswert kann man bereits ab etwa $s = 8$ ein exzellentes Ergebnis beobachten. Die Genauigkeit der Steuerung hinkt stets hinter derjenigen des Zustands hinterher. Diese Erscheinung ist auch für die späteren Beispiele kennzeichnend. Die maximale Abweichung im gesamten Intervall demonstriert die Fähigkeit der Polynome, die analytischen exponentiellen Funktionen anzunähern. Graphisch ist bereits ab $s = 5$ kein Unterschied zwischen exakten und numerischen Trajektorien zu erkennen. Das schwächere Verhalten der Steuerung ist ebenfalls deutlich beobachtbar. (Abbildung 8.1). Beim Zustand kann man die Fälle 4 und 5 nicht mehr unterscheiden.
In [19, Beispiel 1] wurden $s = 5$ und $s = 7$ getestet. Die hier ermittelten Werte stimmen mit den in [19, Beispiel 1] angegebenen Lösungen überein.
Nun wird s festgehalten und N variiert. Die numerischen Trajektorien sind dann an den Übergangsstellen wahrscheinlich nur noch stetig. Der Optimierer hat aber nun erstmals Schwierigkeiten und läßt bei $s = 5$ nur noch maximal $N = 7$ zu. Tabelle 8.5 demonstriert, daß die maximale Gitterabweichung für x bereits bei $N = 1$ sehr gut ist, sich aber für größeres N nicht weiter verbessert (zweite Spalte). Für die Steuerung hat man zwar wie zuvor zunächst schlechtere Näherungen. Die Verbesserungen fallen dann aber deutlich aus (dritte Spalte).
Die vierte Spalte von Tabelle 8.5 versucht, die Konvergenzordnung für die Steuerung zu schätzen. Dafür wird die Veränderung des Fehlers in Beziehung zur Veränderung von N und damit zur Schrittweite h gesetzt, so wie der Konsistenz- bzw. Konvergenzbegriff definiert ist. Bei dieser Untersuchung werden Werte um ca. 4 erreicht. Allerdings ist dieses Ergebnis bei dieser geringen Schrittweitener-

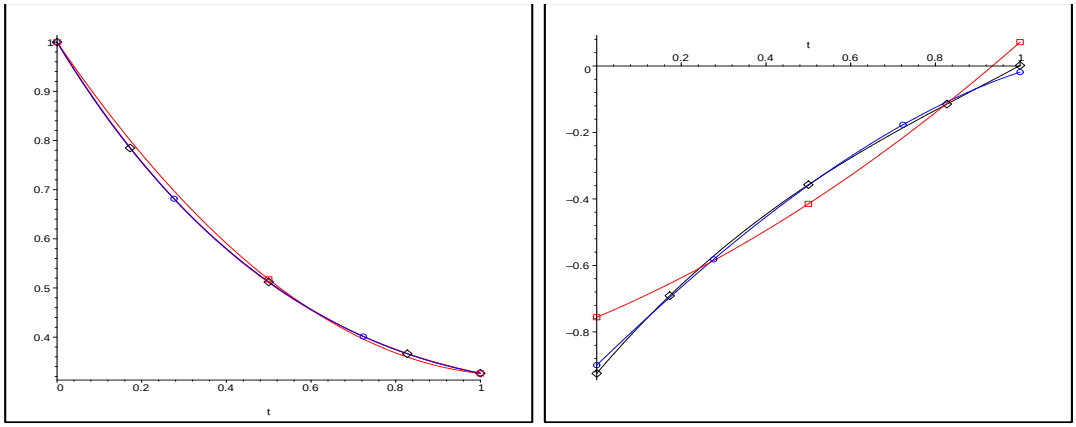

Abbildung 8.1: Bsp. 8.2. Zustand (links) und Steuerung (rechts) für $s = 3$ (rot), $s = 4$ (blau), $s = 5$ (schwarz)

Maximale Gitterabweichung und geschätzte Konvergenzordnung für Zustand und Steuerung bei $s = 5$			
	Zustand	Steuerung	
N	Abweichung	Abweichung	Ordnung
1	$5.910E-08$	$1.483E-03$	–
2	$1.286E-06$	$1.204E-04$	3.623
3	$8.710E-08$	$3.202E-05$	3.267
4	$1.683E-07$	$1.143E-05$	3.580
5	$3.710E-08$	$5.041E-06$	3.669
6	$4.780E-08$	$2.553E-06$	3.731
7	$1.581E-07$	$1.344E-06$	4.163

Tabelle 8.5: Bsp. 8.2. Ergebnisse für Zustand und Steuerung für $s = 5$ und variables N

höhung mit Vorsicht aufzunehmen. Es ist deshalb lohnend, diese Untersuchung auch auf weitere s auszudehnen.

An dieser Stelle muß man sich aber mit dem Startwert der numerischen Optimierung auseinandersetzen. Vorhin war geschildert worden, daß der Optimierer beim Nullvektor als Startwert und $s = 5$ nicht über $N = 7$ hinauskam. Bei der erhöhten Anzahl Unbekannter und Nebenbedingungen gewinnt der Startwert des Optimierungsalgorithmusses an Bedeutung. Bereits der Startwert 1 für alle Variablen erweist sich als viel geeigneter. So erhält man mit diesem Startwert und $N = 32$ ein Ergebnis. Ein Grund dafür könnte sein, daß wenigstens die Nebenbedingung (8.4), die den Anfangswert x_0 der Differentialgleichung enthält, erfüllt

ist. Bei einer glatten, vielleicht nicht zu stark oszillierenden Lösung kann man zudem hoffen, daß weitere Punkte des Zustandes in der Nähe von x_0 liegen.

Maximale Gitterabweichung und geschätzte Konvergenzordnung für Zustand und Steuerung bei $s = 4$				
	Zustand		Steuerung	
N	Abweichung	Ordnung	Abweichung	Ordnung
1	$1.638E-05$	–	$1.844E-02$	–
2	$1.570E-04$	-3.260	$2.870E-03$	2.684
3	$4.269E-05$	3.211	$8.375E-04$	3.037
4	$1.532E-05$	3.562	$3.399E-04$	3.134
5	$6.747E-06$	3.676	$1.698E-04$	3.110
6	$3.426E-06$	3.717	$9.663E-05$	3.092
7	$1.927E-06$	3.732	$5.993E-05$	3.099

Tabelle 8.6: Bsp. 8.2. Ergebnisse für den Zustand und die Steuerung bei $s = 4$ und variablem N

Maximale Gitterabweichung und geschätzte Konvergenzordnung für Zustand und Steuerung bei $s = 3$				
	Zustand		Steuerung	
N	Abweichung	Ordnung	Abweichung	Ordnung
1	$2.649E-04$	–	$7.173E-02$	–
2	$8.846E-04$	-1.740	$2.880E-02$	1.316
3	$1.888E-04$	3.809	$1.754E-02$	1.224
4	$6.742E-05$	3.580	$1.128E-02$	1.533
5	$2.890E-05$	3.797	$7.836E-03$	1.633
6	$1.447E-05$	3.795	$5.739E-03$	1.708
7	$8.018E-06$	3.828	$4.378E-03$	1.755

Tabelle 8.7: Bsp. 8.2. Ergebnisse für den Zustand und die Steuerung bei $s = 3$ und variablem N

Die Tabelle 8.6 listet Ergebnisse für $s = 4$ auf, Tabelle 8.7 für $s = 3$. Es lassen sich erstmals auch Konvergenzschätzungen für den Zustand x erstellen. Die beiden Tabellen sind wie die Tabelle 8.5 gegliedert, mit einer zusätzlichen Spalte für die geschätzte Ordnung des Zustandes. Die Ergebnisse sind doch ziemlich überraschend und interessant. Man erkennt zuerst einen Qualitätsverlust beim Übergang von $N = 1$ auf $N = 2$. Danach verbessern sich die Näherungen für den Zustand rasch, etwa mit der Konvergenzordnung 4. Eine mögliche Erklärung für

die anfängliche Verschlechterung ist, daß der nichtglatte Übergang des Zustandes bei $t = 0.5$ zunächst mehr Nachteile bewirkt, als die Verdopplung der Variablen Vorteile erzeugt. Schließlich liegt die optimale Lösung in $C^\infty([0,1])$.
Noch spannender entwickelt sich die Steuerung: hier bewirkt jede Intervallverkleinerung sofort eine Verbesserung. Die geschätzte Konvergenzordnung ist für $s = 3$ und $s = 4$ erstaunlich stabil. Für $s = 4$ scheint eine Ordnung 3 naheliegend. Nicht ganz so eindeutig ist die Ordnung für $s = 3$ einzuordnen. Aber sie nähert sich der Ordnung 2. Die geschätzte Konvergenzordnung der Steuerung erhöht sich scheinbar bei einer Polynomgraderhöhung um 1.
Eine zentrale Prämisse für die Optimierungsaufgabe ist die Genauigkeit der Nullstellen und Gewichte. Jetzt wird einfach die Anzahl der Nachkommastellen von Nullstellen und Gewichten sukzessive reduziert. Die Tabellen 8.8 und 8.9 dokumentieren diese Testreihe für $s = 5$ und $s = 8$ (bei $N = 1$). $s = 5$ repräsentiert hierbei eine noch eher unzureichende Polynomwahl, $s = 8$ hingegen bereits einen kaum mehr verbesserbaren Polynomgrad (vgl. Tabelle 8.4).

Zielfunktion und maximale Gitterabweichung für x und u bei $N = 1$ und $s = 5$ und variabler Datengenauigkeit der Knoten und Gewichte			
Stellengenauigkeit	Zielfunktion	Zustand	Steuerung
17	$5.970E-07$	$5.754E-08$	$1.483E-03$
15	$5.970E-07$	$5.754E-08$	$1.483E-03$
13	$5.970E-07$	$5.754E-08$	$1.483E-03$
11	$5.970E-07$	$5.754E-08$	$1.483E-03$
9	$5.967E-07$	$5.754E-08$	$1.483E-03$
7	$5.727E-07$	$5.727E-08$	$1.483E-03$
5	$1.812E-06$	$4.248E-08$	$1.455E-03$
3	$2.410E-04$	$1.570E-06$	$3.286E-03$

Tabelle 8.8: Bsp. 8.2. Die erste Spalte gibt die Nachkommastellen der Daten an

Bei $s = 5$ fällt auf, daß signifikante Unterschiede erst bei sehr ungenauen Daten auftreten. Anders bei $s = 8$: ab etwa 11 Nachkommastellen beginnen emminente Differenzen sichtbar zu werden. Offenbar dominiert bei $s = 5$ zunächst der schwächere Ansatz die Ungenauigkeit der Daten. Die Fehler der ungenauen Approximation und die geringe Datenqualität überlagern sich, so daß bei 3 und 5 Nachkommastellen sogar partiell besser als bei $s = 8$ approximiert wird. Beim präzisen Ansatz $s = 8$ können die Vorzüge exakter Daten aber gut beobachtet werden.
Das nächste Beispiel enthält zusätzliche Terme in der Zielfunktion.

Zielfunktion und maximale Gitterabweichung für x und u bei $N = 1$ und $s = 8$ und variabler Datengenauigkeit der Knoten und Gewichte			
Stellengenauigkeit	Zielfunktion	Zustand	Steuerung
17	$4.402E-14$	$4.712E-09$	$2.544E-07$
15	$4.402E-14$	$4.712E-09$	$2.544E-07$
13	$2.998E-14$	$4.712E-09$	$2.544E-07$
11	$1.013E-11$	$4.712E-09$	$2.536E-07$
9	$1.092E-09$	$4.694E-09$	$2.054E-07$
7	$1.588E-07$	$7.811E-10$	$1.173E-06$
5	$1.029E-05$	$7.948E-08$	$2.174E-04$
3	$1.088E-03$	$9.830E-05$	$4.500E-02$

Tabelle 8.9: Bsp. 8.2. Die erste Spalte gibt die Nachkommastellen der Daten an

Beispiel 8.3. *Diese Aufgabe stammt aus [43, Beispiel 3.3].*
Minimiere

$$J(x,u) := \frac{5}{2}x^2(1) + \int_0^1 \Big(\frac{1}{2}u^2(t) + 5x(t)\Big)dt$$

unter den Nebenbedingungen

$$\begin{aligned} \dot{x}(t) &= u(t), \quad t \in [0,1], \\ x(0) &= 1. \end{aligned}$$

Die analytischen Lösungen lauten:

$$\begin{aligned} \hat{x}(t) &= \frac{5}{2}t^2 - \frac{15}{4}t + 1, \quad t \in [0,1], \\ \hat{u}(t) &= 5t - \frac{15}{4}, \quad t \in [0,1], \\ \hat{J}(\hat{x},\hat{u}) &= \frac{85}{48} = 1.7708\bar{3}. \end{aligned}$$

In diesem Beispiel treten der lineare Term unter dem Integral und der quadratische Term davor neu auf. Die Ausgangssituation für den numerischen Ansatz ist damit nach wie vor günstig, wie auch die besonders einfachen analytischen Lösungen belegen. Diese können mit einem Polynomansatz sogar erreicht werden.
Zu Beginn erfolgt wieder eine Umwandlung des Beispiels in die Aufgabe 7.8. Polynome werden also für x und u substituiert, das Integral mit der Lobatto-Quadraturregel approximiert und die Differentialgleichung nur noch an ausgewählten Punkten als Gleichungsbedingung gefordert, wie in Beispiel 8.2 beschrieben. Ferner sollen alle Variablen des Startwertes des Optimierers gleich 1 sein,

um die numerische Rechnung zu erleichtern, siehe ebenfalls Beispiel 8.2. Eine Intervalldiskretisierung unterbleibt, es wird $N = 1$ gesetzt. Der Polynomgrad wird variiert. Die Ergebnisse sind in der Tabelle 8.10 schematisiert.

	Alle Werte als Beträge der Abweichungen		
s	Zielfunktion	Zustand und Steuerung bei $t = 1$	
3	0	$1.002E-09$	$1.468E-08$
4	0	$2.736E-09$	$2.233E-08$
5	0	$3.186E-09$	$1.026E-08$
6	0	$3.438E-09$	$4.789E-08$
8	$9.992E-15$	$1.983E-09$	$1.885E-08$
10	$1.201E-13$	$9.281E-09$	$1.230E-08$
12	$1.998E-14$	$2.602E-08$	$1.044E-07$

Tabelle 8.10: Bsp. 8.3. Ergebnisse für $N = 1$ und variables s

Bereits für $s = 3$ ist der Zielfunktionswert exakt auf Maschinengenauigkeit. Auch sind die Trajektorien bei $t = 1$ schon sehr präzise. Allerdings erreicht man mit höherem Polynomgrad keine Verbesserung mehr. Die Abweichungen vergrößern sich sogar. Gerade weil das Problem so effektiv gelöst werden kann, wirken sich vermutlich höhere Grade programm- und rechnerintern negativ aus (mehr Gleichungen, Schwierigkeiten bei der Matrix D, Rundungsfehler usw.). Die Abbildung 8.2 visualisiert Zustand und Steuerung, wobei die obigen numerischen Lösungen graphisch mit den optimalen übereinstimmen.

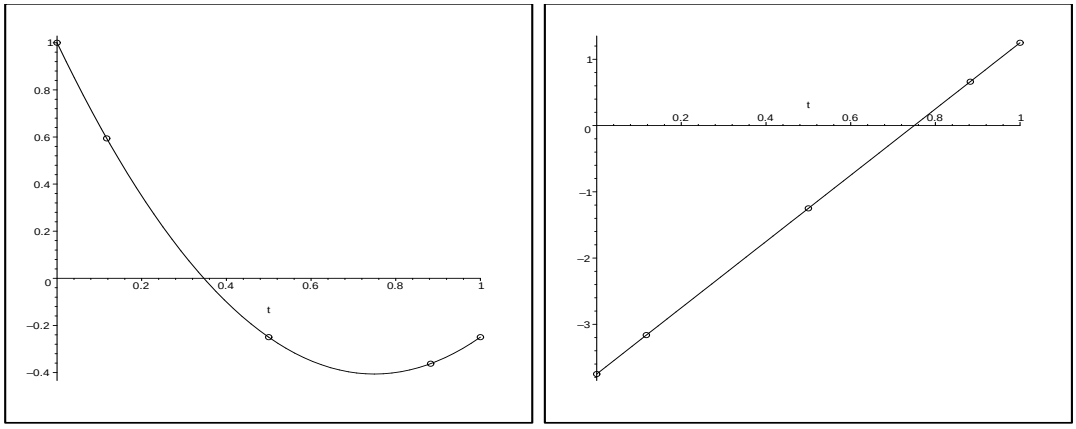

Abbildung 8.2: Bsp. 8.3. Optimaler Zustand (links) und Steuerung (rechts)

Die Abbildung 8.3 greift den Aspekt der maximalen Abweichung auf dem gan-

zen Intervall $[0,1]$ auf. Die Graphen stellen jeweils die Differenzfunktionen von numerischen und optimalen Lösung bei $s = 5$ dar. Interessanterweise treten die Differenzen offenbar losgelöst von den Kollokationstellen auf. Allerdings sind auch Rundungsfehler unvermeidbar. Für $t = 0$ ensteht beim Zustand x zwangsläufig keine Abweichung.

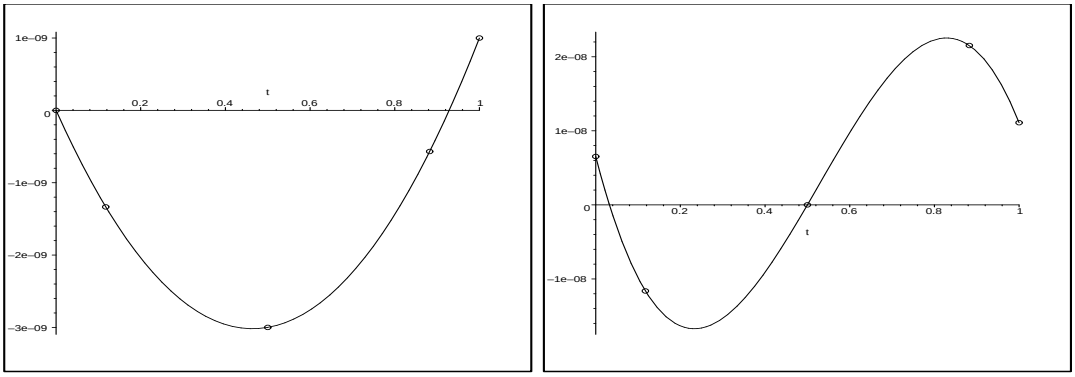

Abbildung 8.3: Bsp. 8.3. Differenzfunktionen für Zustand (links) und Steuerung (rechts)

Das dritte, kurze Beispiel soll den Einsatz des Programms auf Aufgaben höherer Dimensionalität testen.

Beispiel 8.4. *Dieses Beispiel stammt aus [36, Abschnitt 3.6].*
Minimiere

$$J(x_1, x_2, u) := \int_0^1 \big(x_1^2(t) + x_2^2(t) + 0.005u^2(t)\big)dt$$

unter den Nebenbedingungen

$$\begin{aligned}\begin{pmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\end{pmatrix} &= \begin{pmatrix}0 & 1\\ 0 & -1\end{pmatrix}\begin{pmatrix}x_1(t)\\ x_2(t)\end{pmatrix} + \begin{pmatrix}0\\ 1\end{pmatrix}u(t),\\ \begin{pmatrix}x_1(0)\\ x_2(0)\end{pmatrix} &= \begin{pmatrix}0\\ -1\end{pmatrix}.\end{aligned}$$

Es ist laut [36, Abschnitt 3.6]

$$\hat{J} = 0.06936094.$$

Ansonsten liegen außer Graphiken in [36, Abschnitt 3.6] keine weiteren Informationen über die optimalen Lösungen vor.

Das Beispiel ist nicht nur erstmals mehrdimensional, es besitzt in der Zielfunktion auch sehr unterschiedlich gewichtete Faktoren (0.005 einerseits und 1 andererseits).
Der Optimierer hat bei diesem Beispiel große Schwierigkeiten. Nur für gewisse Polynomgrade $s-1$ und eine Diskretisierung bis $N=2$ gelangt er zu Endergebnissen, ja überhaupt zu mehreren Iterationen. Eine Startwertanpassung führt zu keiner Verbesserung. Aber auch hier reichen die erfolgreichen Kombinationen von s und N aus, um das obige Ergebnis zu verifizieren:

$$J_{12}^1 = 0.06936103677824514 \quad \text{bzw.} \tag{8.5}$$
$$J_{12}^2 = 0.06936094372810254. \tag{8.6}$$

Die Abbildungen 8.4 und 8.5 veranschaulichen die numerischen Ergebnisse. Angesichts der Zielfunktionsgenauigkeit spiegeln sie wahrscheinlich ziemlich genau die tatsächlichen optimalen Lösungsverläufe wider. Die graphischen Verläufe der drei Trajektorien stimmen überdies mit den Graphiken aus [36, Abschnitt 3.6] überein, die allerdings auch numerische Lösungen repräsentieren.

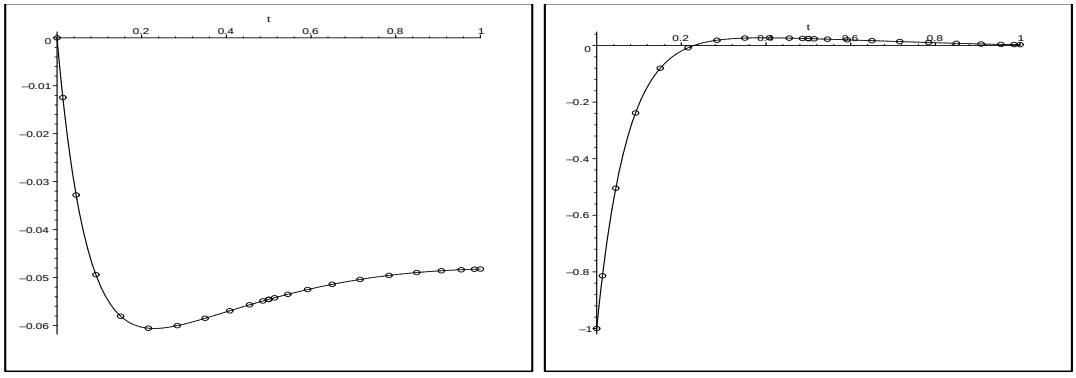

Abbildung 8.4: Bsp. 8.4. Lösungsverläufe von x_1 (links) und x_2 (rechts) für $s=12$ und $N=2$

Diese Beispielaufgabe hat zwar (möglicherweise) glatte Optimallösungen, gegenüber den Beispielen 8.2 und 8.3 sind die Trajektorien aber erheblich abwechslungsreicher. Immerhin bewegt sich die Steuerung innerhalb kürzester Zeit von ca. 14 zum Zeitpunkt $t=0$ mit steilem Abstieg auf 0 zum Zeitpunkt ca. $t=0.4$, um dort (anscheinend) zu verbleiben. Vielleicht tragen diese Schwankungen mit zu den Schwierigkeiten des Optimierers bei. Die Tabelle 8.11 bietet ein paar ungeordnete numerische Lösungen an, die verschiedene Ansätze geliefert haben.
Obgleich keine systematischen Ergebnisse vorliegen, kann die Tabelle 8.11 doch einige Erkenntnisse vermitteln. Der Gegensatz zu den Beispielen 8.2 und 8.3 verdeutlicht sich in den ungenügenden Werten für die Steuerung. Während bei den

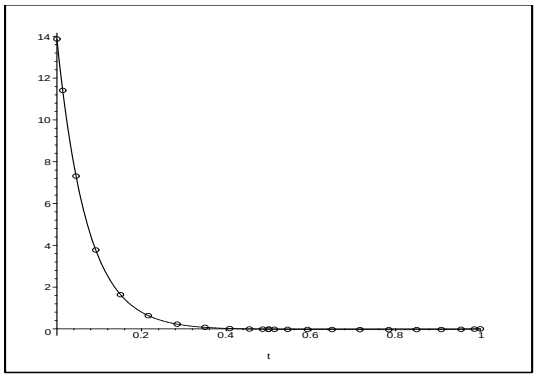

Abbildung 8.5: Bsp. 8.4. Steuerung für $s = 12$ und $N = 2$

$u(0)$, $u(1)$, $x_1(1)$ und $x_2(1)$ sind numerische Ergebnisse am Intervallende						
s	N	Zielfunktion	$u(0)$	$u(1)$	$x_1(1)$	$x_2(1)$
12	2	$6.936E-02$	13.87	$1.82E-05$	$-4.82E-02$	$3.18E-03$
12	1	$6.936E-02$	13.85	$-1.46E-02$	$-4.82E-02$	$3.18E-03$
10	1	$6.937E-02$	13.63	$-1.15E-01$	$-4.82E-02$	$3.10E-03$
8	1	$6.970E-02$	12.44	$-5.99E-01$	$-4.84E-02$	$2.42E-04$
6	1	$7.597E-02$	8.64	$-1.89E+00$	$-5.21E-02$	$-4.31E-02$

Tabelle 8.11: Bsp. 8.4. Informationen der Optimierung zu verschiedenen s und N

vorangegangenen Beispielen jeweils etwa $N = 1$ und $s = 6$ für brauchbare Approximationen ausreichten, beziehungsweise bei $s = 8$ exzellent waren, sind die Näherungen für u hier bei gleichen Einstellungen unbrauchbar und für $N = 1$ und $s = 8$ erst langsam aussagekräftig. Die um etwa 50% gesteigerte Anzahl der Variablen und Nebenbedingungen sind hierfür auch keine ausreichende Erklärung, wie Beispiel 8.7 bestätigen wird. Dort werden sehr gute Ergebnis für ein höherdimensionales Problem erzielt. Auch hier bestätigt sich im übrigen, daß die ungenügende Approximation von u nicht im Zustand oder der Zielfunktion erscheint. Offenbar ist das Konvergenzverhalten der Steuerung veritabel schwächer. Im folgenden Beispiel wird erstmals eine Steuerbeschränkung verwendet. Da die optimalen Trajektorien aber stetig sind, eignet es sich dennoch für einen Polynomansatz.

Beispiel 8.5. *Diese Aufgabe spezialisiert ein Beispiel aus [15, §4].*
Minimiere

$$J(x,u) := \int_0^2 \Big(\frac{1}{2}u^2(t) - x(t)\Big)\,dt$$

unter den Nebenbedingungen

$$\begin{aligned} \dot{x}(t) &= u(t) \quad \text{für fast alle } t \in [0,2], \\ x(0) &= 1, \\ u(t) &\in [0,1], \quad t \in [0,1]. \end{aligned}$$

Die analytischen Lösungen lauten:

$$\hat{x}(t) = \begin{cases} t+1, & 0 \le t \le 1, \\ -\frac{1}{2}t^2 + 2t + \frac{1}{2}, & 1 \le t \le 2, \end{cases}$$

$$\hat{u}(t) = \begin{cases} 1, & 0 \le t \le 1, \\ 2-t, & 1 \le t \le 2, \end{cases}$$

$$\hat{J}(\hat{x},\hat{u}) = -\frac{19}{6} = -3.1\bar{6}.$$

Das Optimierungsprogramm hat erneut gewisse Schwierigkeiten mit dieser Aufgabe. Öfters bricht es ergebnislos ab. In der Regel rechnet es aber sehr lange. Teilweise reichen ihm die zur Verfügung gestellten Rechneriterationen nicht aus (vgl. Einstellung *icntl(3)* in A.1). So reichen bei der Kombination $N = 1$ und $s = 6$ *icntl(3)*=10000 Iterationen nicht aus, um einen Optimalpunkt zu finden, obwohl bereits bei der zehnten Iteration der Zielfunktionswert -3.1666639 angenommen wird! Interessante Unterschiede treten auch zwischen geraden und ungeraden s auf. Bei $N = 1$ finden alle Aufgaben mit ungeradem s Minima, diejenigen mit geradem s aber nur zum Teil. Eine Erklärung für dieses Phänomen könnte die subsidiäre Lage eines Kollokationspunktes bei $t = 1$ für ungerade s sein, denn dort liegt die einzige nichtglatte Stelle von $\hat{x}$ und $\hat{u}$.
Trotz dieser Mängel liefert das Programm auch bei vorzeitigen Abbrüchen immer verwertbare Daten. Die Optimierer pendelt um die analytischen Optimallösungen. Die letzten Ergebnisse der Optimierung vor einem formal ergebnislosen Programmabbruch werden im folgenden deshalb stets als numerische Optimallösungen gewertet.
Die Abbildung 8.6 bildet die optimalen und numerischen Trajektorien für $N = 1$ und $s = 9$ ab. Typisch sind die qualitativen Abschläge der Näherung von u im Vergleich zu denjenigen von x.
Als nächstes wird N variiert und $s = 3$ respektive $s = 4$ gesetzt. Dazu werden jeweils die numerischen Werte am Ende jedes Intervalls mit den Werten der optimalen Lösung an diesen Zeitpunkten verglichen. Die Tabelle 8.12 gibt die Ergebnisse wieder. Die maximalen Abweichungen schwanken unkontrolliert, es treten teilweise erstaunliche Verschlechterungen für größere N auf. Folglich können keine Konvergenzschätzungen eruiert werden.

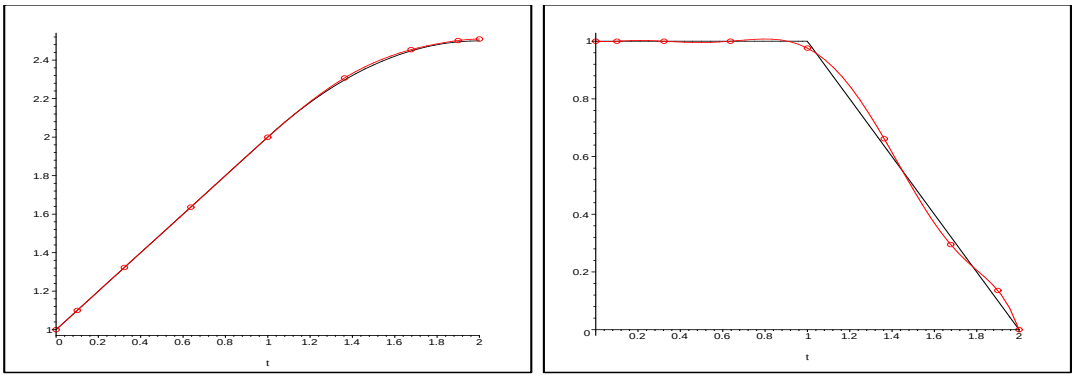

Abbildung 8.6: Bsp. 8.5. Optimale und numerische Lösungen für Zustand (links) und Steuerung (rechts) bei $N = 1$ und $s = 9$

Maximale Gitterabweichungen von Zustand und Steuerung für $s = 3$ bzw. $s = 4$ und verschiedene Intervalldiskretisierungen N					
$s = 3$			$s = 4$		
N	Zustand	Steuerung	N	Zustand	Steuerung
1	$1.400E-08$	$5.000E-01$	1	$3.929E-02$	$9.701E-05$
2	$4.000E-08$	$7.933E-08$	2	$3.348E-05$	$4.040E-04$
4	$6.893E-06$	$4.964E-05$	4	$4.341E-06$	$6.239E-05$
8	$1.560E-07$	$1.058E-06$	8	$1.148E-05$	$7.476E-04$
16	$6.759E-06$	$1.901E-04$	16	$1.361E-06$	$5.950E-05$

Tabelle 8.12: Bsp. 8.5. $s = 3$ und 4 für verschiedene N im Vergleich

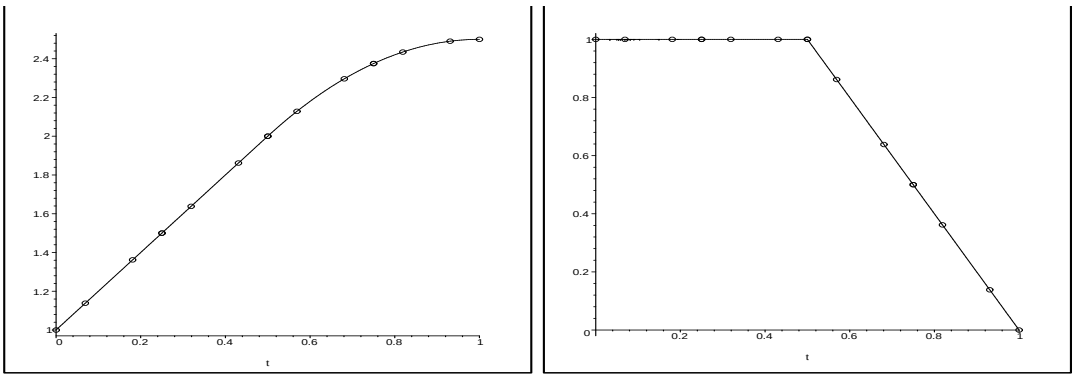

Abbildung 8.7: Bsp. 8.5. Zustand (links) und Steuerung (rechts) für $N = s = 4$

Jedoch lassen sich graphisch trotz der etwas unbefriedigenden Werte Verbesserungen feststellen. Abbildung 8.7 demonstriert die numerischen Ergebnisse für $N = 4$ und $s = 4$. Obwohl im Vergleich zur Abbildung 8.6 die Variablenanzahl jeder Trajektorie nicht einmal verdoppelt wurde ($4 \cdot 4 = 16$ gegenüber $1 \cdot 9 = 9$), verschwindet die graphische Diskrepanz zur Optimallösung. Übrigens ist auch dieses Ergebnis eine letzte Variablenbelegung vor einem unerwünschten Programmabbruch.
Das nächste Beispiel besitzt eine lineare Zielfunktion.

Beispiel 8.6. *Das Beispiel stammt aus [4, Beispiel 3.3.2]. Der Vektor* $(c_1, c_2)^T \in \mathbb{R}^2$ *sei fest. Man beachte Anhang A.5. Die Aufgabe lautet:*
Minimiere

$$J(x_1, x_2, u) := \langle c, x(t_f) \rangle = c_1 x_1(1) + c_2 x_2(1)$$

unter den Nebenbedingungen

$$\begin{aligned}
\begin{pmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{pmatrix} &= \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u(t) \quad \text{für fast alle } \ t \in [0,1], \\
\begin{pmatrix} x_1(0) \\ x_2(0) \end{pmatrix} &= \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \\
u(t) &\in [-1,1], \quad t \in [0,1].
\end{aligned}$$

Während dieses Beispiel 8.6 offenkundig ein Optimierungsproblem ist, wird in [4, Beispiel 3.3.2] eine sogenannte erreichbare Menge, die Menge der möglichen Endzustände $x_1(1)$ und $x_2(1)$ bei den gegebenen Nebenbedingunggen, ohne Aufstellen einer Zielfunktion, gesucht. In [4] kann man die Thematik erreichbarer Mengen studieren. Hier lenkt der Vektor $(c_1, c_2)^T$ die Zusammensetzung der Gewichtung des Endzustandes. Variiert man den Vektor $(c_1, c_2)^T$ in alle denkbaren Richtungen, etwa per

$$(c_1, c_2) = \big(sin(\varphi), cos(\varphi)\big), \quad \varphi \in [0, 2\pi], \tag{8.7}$$

läßt sich die Menge dieser Minimierungsprobleme auch als Versuch interpretieren, alle möglichen Endzustände einzukreisen.
Die lineare und zeitunabhängige Differentialgleichung ermöglicht die Berechnung eines Fundamentalsystems. Die Lösung x kann in Abhängigkeit von der Steuerung u konstruiert und in die Zielfunktion eingesetzt werden, wie man in [59, §16] oder [24, §12] nachlesen kann:

$$x(t) = \int_0^t \begin{pmatrix} 1 - \tau \\ 1 \end{pmatrix} u(\tau)\, d\tau. \tag{8.8}$$

Für die Zielfunktion ergibt sich bei Wiederverwendung des Zeitbuchstabens t:

$$J(u) = \int_0^1 u(t) \cdot (c_1 + c_2 - c_1 t)\, dt, \quad u(t) \in [-1, 1], \quad t \in [0, 1]. \tag{8.9}$$

Nach eingehender Betrachtung der Steuerbeschränkungen kann man folgern, daß die Steuerung fast überall -1 oder 1 ist. Für den mögliche Umschaltpunkt gilt

$$u(t) = \begin{cases} 1, & c_1 + c_2 - c_1 t < 0, \\ \gamma, & c_1 + c_2 - c_1 t = 0, \quad \gamma \in [-1, 1], \\ -1, & c_1 + c_2 - c_1 t > 0. \end{cases} \tag{8.10}$$

Als erstes wird die Richtung $(c_1, c_2) = (-3, 1)$ getestet. Es gilt

$$\hat{u}(t) = \begin{cases} 1, & 0 \le t < \frac{2}{3}, \\ \gamma, & t = \frac{2}{3}, \quad \gamma \in [-1, 1], \\ -1, & \frac{2}{3} < t \le 1. \end{cases} \tag{8.11}$$

Das Verhalten für $t = \frac{2}{3}$ ist hier irrelevant. Es ergibt sich

$$\begin{aligned} \hat{x}_1(1) &= \frac{7}{18}, \\ \hat{x}_2(1) &= \frac{1}{3}, \\ \hat{J}(\hat{x}_1, \hat{x}_2, \hat{u}) &= -\frac{5}{6}. \end{aligned} \tag{8.12}$$

Es wird $N = 1$ gesetzt. Die Tabelle 8.13 vergleicht Zielfunktionswerte und Endzustände für mehrere Polynomgrade. Auffällig sind die schwachen Näherungen für $s = 12$.

	Alle Werte als Beträge der Abweichungen		
s	Zielfunktion	Zustand bei $t = 1$	Steuerung bei $t = 1$
6	$8.984E-02$	$7.353E-02$	$1.308E-01$
8	$3.881E-02$	$2.971E-02$	$5.033E-02$
10	$3.035E-02$	$9.009E-03$	$3.325E-03$
12	$1.582E-02$	$1.245E-02$	$5.317E-02$

Tabelle 8.13: Bsp. 8.6. Ergebnisse für $N = 1$ und variables s

Die Abbildung 8.8 vergleicht die optimale Steuerung mit numerischen Resultaten aus der Tabelle 8.13. Die Unstetigkeitsstelle ist für ein Polynom schwer zu approximieren, bei $s = 10$ setzt der Optimierer immerhin 8 der 10 Werte ziemlich korrekt.

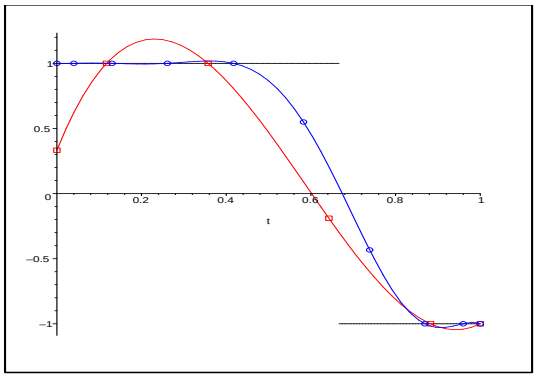

Abbildung 8.8: Bsp. 8.6. Optimale Steuerung (schwarz), Näherungen für $N = 1$ und $s = 10$ (blau), für $N = 1$ und $s = 6$ (rot)

Die Vergrößerung der Intervallanzahl bereitet dem Optimierer Probleme. So gelingt es zunächst nicht, für $N = 2$ und $s = 12$ ein Ergebnis zu erhalten. Stattdessen wird nun mit dem kleineren Polynomgrad 4 ($s = 5$) weitergearbeitet. Aber auch hier gelangt der Optimierer anfangs erneut nicht über $N = 4$ hinaus. Der numerische Verlauf der Steuerung ist in der Abbildung 8.9 abgebildet. Immerhin kann man auch ohne Vorwissen eine Umschaltstelle der Steuerung in der Nähe von $t = \frac{2}{3}$ vermuten.

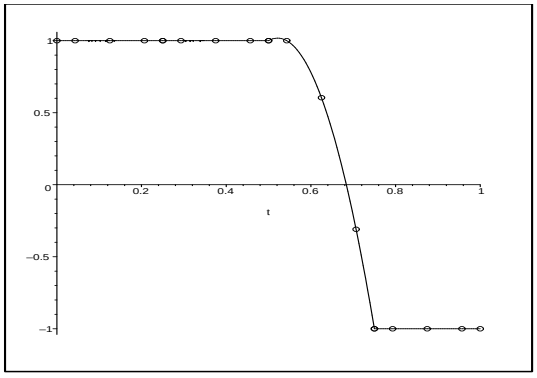

Abbildung 8.9: Bsp. 8.6. Approximierte Steuerung für $N = 4$ und $s = 5$

Es ist deshalb nun angebracht, die Stetigkeit der Steuerung beim Intervallübergang aufzuheben (siehe auch (8.3) im Beispiel 8.2). Hierbei soll vernachlässigt werden, daß man an den Übergängen formal je zwei verschiedene Werte für einen Zeitpunkt zuläßt. Bei $N = 3$ führt diese Enkopplung dazu, daß ein Intervallende mit der Sprungstelle übereinstimmt. Der Zielfunktionswert weicht nur noch um $3.70372 \cdot 10^{-9}$ von Optimum ab. Ähnliche Abweichungen ergeben sich auch für

den Zustand und die Steuerung. Die Graphen in Abbildung 8.10 visualisieren die numerischen Ergebnisse.

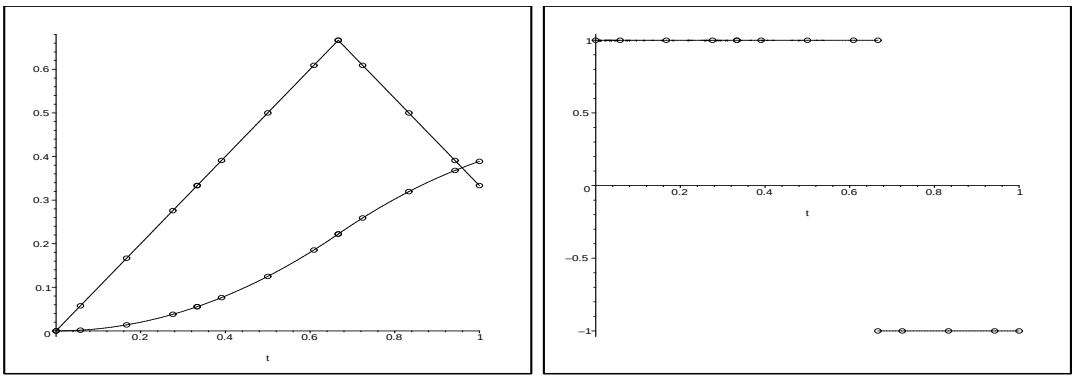

Abbildung 8.10: Bsp. 8.6. x_2 (mit Knick) und x_1 links. u (rechts) ist unstetig angesetzt ($N = 3$ und $s = 5$)

Trifft die Diskretisierung den Schaltpunkt der Steuerung nicht, wird der Vorteil des unstetigen Ansatzes wieder aufgehoben, da der Schaltpunkt dann wieder durch ein Polynom überbrückt werden muß.
Aber hier erweist sich der Optimierer wiederum als unberechenbar und trickreich zugleich. Er löst nun Probleme mit deutlich kleineren Intervallen, also größeren N. Damit kann man den Effekt einer Intervallverkleinerung doch noch beobachten. Für $s = 5$ und $N = 20$ liegt der Schaltpunkt $t = \frac{2}{3}$ im Intervall $[0.65, 0.70]$. Die Abbildung 8.11 veranschaulicht die numerische Steuerung.

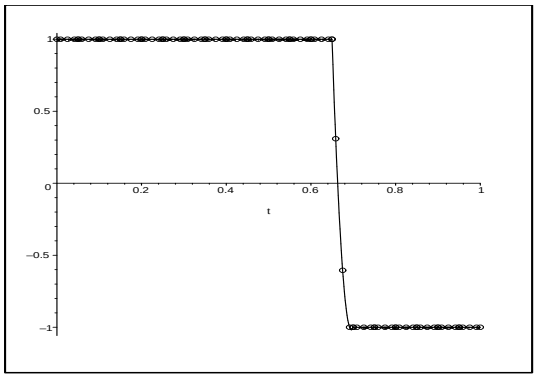

Abbildung 8.11: Bsp. 8.6. Steuerung für $N = 20$ und $s = 5$

Die Abbildung 8.12 greift die Variation des Zielfunktionvektors $(c_1, c_2)^T$ auf. Hier wurden mit $N = 1$ und $s = 12$ insgesamt 50 Vektoren getestet. Da der Zustand

zweidimensional ist, lassen sich die Endzustände bei $t = 1$ in einer zweidimensionale Ebene abbilden. Die Zeichnung 8.12 illustriert 50 Punkte, also 50 Optimierungsprobleme. Die 50 ausgewählten Vektoren sind in A.5 aufgelistet. Beim Vergleich mit [4, Abbildung 15] erkennt man insgesamt eine gute Approximation. Besonders die Spitzen des augenförmigen Gebietes sind in ihren Krümmungen fein herausgearbeitet, wenn man sich die Punkte verbunden vorstellt.

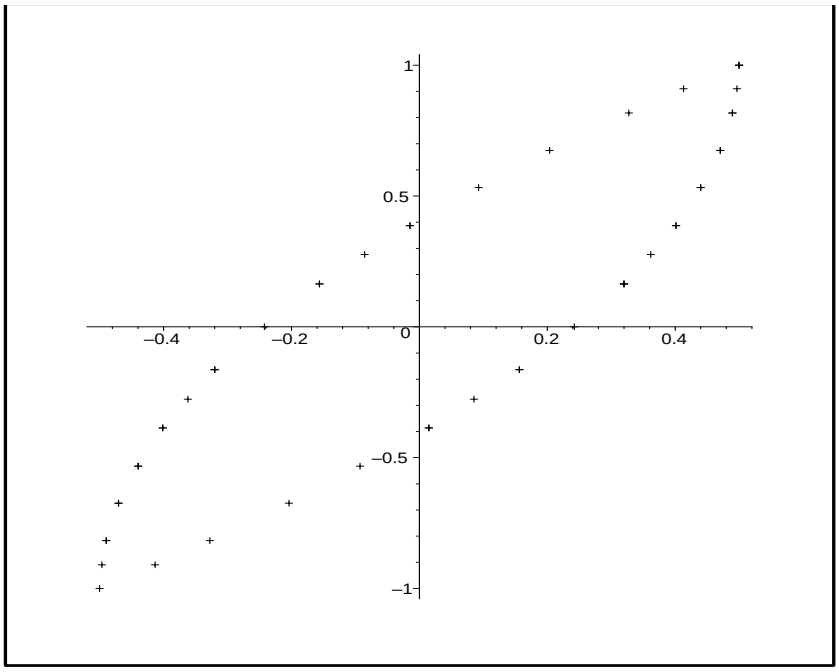

Abbildung 8.12: Bsp. 8.6. Bei $N = 1$ und $s = 12$ sind für ausgewählte Vektoren $(c_1, c_2)^T$ die Endzustände abgebildet (vgl. [4, Abbildung 15])

Das Beispiel 8.7 führt eine mehrdimensionale Kontrolle ein.

Beispiel 8.7. *Das Beispiel stammt aus [4, Beispiel 3.3.1]. Der Vektor* $(c_1, c_2)^T \in \mathbb{R}^2$ *sei fest. Man beachte Anhang A.6. Die Aufgabe lautet:*
Minimiere

$$J(x_1, x_2, u) := \langle c, x(t_f) \rangle = c_1 x_1(2) + c_2 x_2(2)$$

unter den Nebenbedingungen

$$\begin{pmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -2 & -3 \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix}$$

$$\text{für fast alle } \; t \in [0, 2],$$

$$\begin{pmatrix} x_1(0) \\ x_2(0) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$
$$u_1^2(t) + u_2^2(t) \leq 1, \quad t \in [0,2].$$

Es ist hier schwierig, die Optimallösung im voraus zu berechnen. Mit einigen Überlegungen gelangt man aber zu dem Schluß, daß die Steuerungen nicht springen werden.
Als erstes wird der Vektor $(c_1, c_2) = (-1, 0)$ gewählt. Nach A.6 lauten die exakten Lösungen

$$\begin{aligned} \hat{J} &= -1.3023318865252076, \\ \hat{x}_1(2) &= 1.3023318865252076, \\ \hat{x}_2(2) &= -0.7218334911613578. \end{aligned} \tag{8.13}$$

Für $N = 1$ und variablem Polynomgrad $s - 1$ erhält man die Ergebnisse der Tabelle 8.14.

	Alle Werte als Abweichungsbeträge		
s	Zielfunktion	Zustand bei $t = 2$	Steuerung bei $t = 2$
12	$2.764E-10$	$2.764E-10$	$8.474E-09$
11	$3.218E-09$	$3.218E-09$	$5.379E-08$
10	$3.030E-08$	$3.030E-08$	$3.018E-07$
9	$2.625E-07$	$2.625E-07$	$2.543E-06$
8	$1.670E-06$	$1.670E-06$	$1.643E-05$
7	$1.514E-05$	$1.514E-05$	$1.150E-04$
6	$5.634E-05$	$5.634E-05$	$4.074E-04$
5	$9.793E-04$	$9.793E-04$	$3.662E-03$
4	$3.015E-03$	$3.015E-03$	$2.581E-03$
3	$2.997E-02$	$2.997E-02$	$4.546E-02$

Tabelle 8.14: Bsp. 8.7. Ergebnisse für $N = 1$ und variables s

Die zweite Spalte gibt die Abweichung des numerischen Zielfunktionswertes vom exakten an. Die dritte und vierte Spalte enthalten die Abweichungen von Zustand und Steuerung am Intervallende. Wegen des spezifischen Vektors c sind die Werte der zweiten und dritten Spalte identisch. Bei steigendem Polynomgrad verbessern sich die numerischen Ergebnisse merklich. Die Güte der Ergebnisse spiegelt glatte Verläufe der optimalen Trajektorien $\hat{x}$ und $\hat{u}$ wider.
Wie in Beispiel 8.6 werden nun 50 Vektoren $(c_1, c_2)^T$ ausgewählt, siehe A.6. Die Abbildung 8.13 kann mit [4, Abbildung 14] verglichen werden. Auch hier wird das augenförmige Gebilde recht gut getroffen.

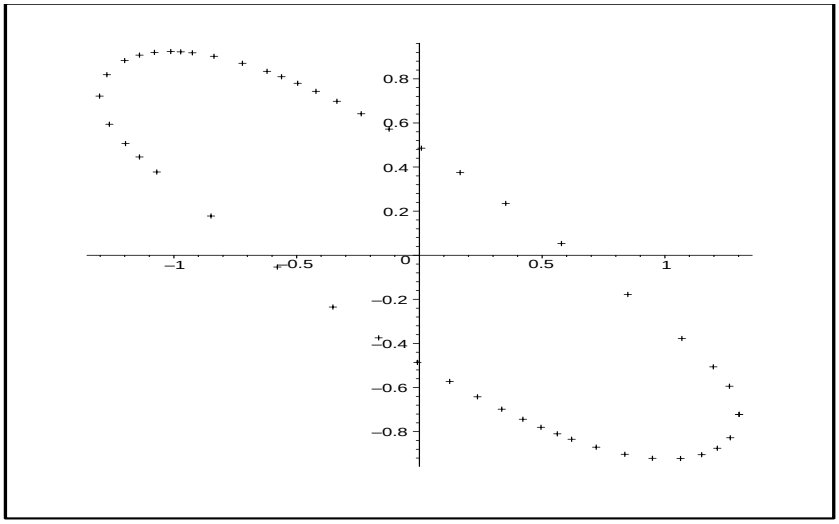

Abbildung 8.13: Bsp. 8.7. Bei $N = 1$ und $s = 12$ sind für ausgewählte Vektoren $(c_1, c_2)^T$ die Endzustände abgebildet (vgl. [4, Abbildung 14])

Im letzten Beispiel wird ein Optimalsteuerungsproblem mit einer Randwertbedingung und mit Steuerbeschränkungen untersucht.

Beispiel 8.8. *Diese Aufgabe spezialisiert ein Beispiel aus [15, §7].*
Minimiere

$$J(x,u) := \frac{1}{2}\int_0^3 x^2(t)\,dt$$

unter den Nebenbedingungen

$$\begin{aligned} \dot{x}(t) &= u(t) \text{ für fast alle } t \in [0,3], \\ x(0) &= 1, \\ x(3) &= 1, \\ u(t) &\in [-1,1], \quad t \in [0,3]. \end{aligned}$$

Die analytischen Lösungen lauten:

$$\hat{x}(t) = \begin{cases} 1-t, & 0 \le t \le 1, \\ 0, & 1 \le t \le 2, \\ t-2, & 2 \le t \le 3, \end{cases}$$

$$\hat{u}(t) = \begin{cases} -1, & 0 \le t \le 1, \\ 0, & 1 \le t \le 2, \\ 1, & 0 \le t \le 1, \end{cases}$$

Der Zielfunktionswert beträgt hier

$$\hat{J}(\hat{x}, \hat{u}) = \frac{1}{3}.$$

Bei diesem Beispiel liegen mit dem Randwert und den Steuerbeschränkungen gleich zwei „Verstöße“ gegen die ideale Aufgabenstellung eines Linear-Quadratischen Optimalsteuerungsproblems vor. Die optimale Steuerung springt zweimal. Ein Polynomansatz wird hier immer fraglicher, soll aber trotzdem versucht werden. Zunächst gilt $N = 1$. In Abbildung 8.14 werden die optimalen Lösungen den numerischen Lösungen mit den Polynomgraden 6 ($s = 7$) sowie 11 ($s = 12$) gegenübergestellt.

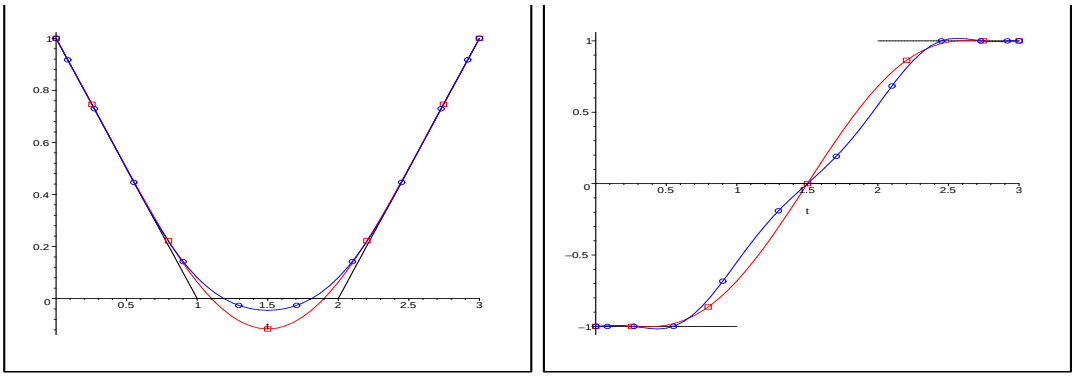

Abbildung 8.14: Bsp. 8.8. Zustand (links) und Steuerung (rechts). $s = 7$ (rot), $s = 12$ (blau) bei $N = 1$ und die optimalen Lösungen (schwarz)

Die Zustandsfunktion wird insgesamt ganz gut getroffen. Die Sprünge der Steuerung können jedoch kaum beschrieben werden. Bei $s = 12$ sind die 8 peripheren Punkte exakt gesetzt, die zentralen 4 müssen zwei Schaltpunkte überbrücken. Das Dilemma der Optimierung wird auch aus den Zielfunktionswerten ersichtlich:

$$\begin{aligned} J_7^1 &= 0.3396366982715388, \\ J_{12}^1 &= 0.3360237423510303, \\ \hat{J} &= \frac{1}{3} = 0.\bar{3}. \end{aligned} \tag{8.14}$$

Der Zielfunktionswert J_{12}^1 weicht um weniger als ein Prozent vom tatsächlichen Optimum ab. Dem Optimierer ist also kein Vorwurf zu machen.
Die Schaltstellen werden für jeden stetigen Polynomansatz unzureichend beschrieben. Es ist deshalb plausibel, den Polynomgrad klein zu halten und stattdessen die Intervallanzahl zu vergrößern. Von nun an wird $s = 3$ gesetzt. Zustand und Steuerung werden also in jedem Intervall quadratisch approximiert. Die Abbildung 8.15 zeigt die numerischen Lösungen für $N = 8$ und $N = 32$.

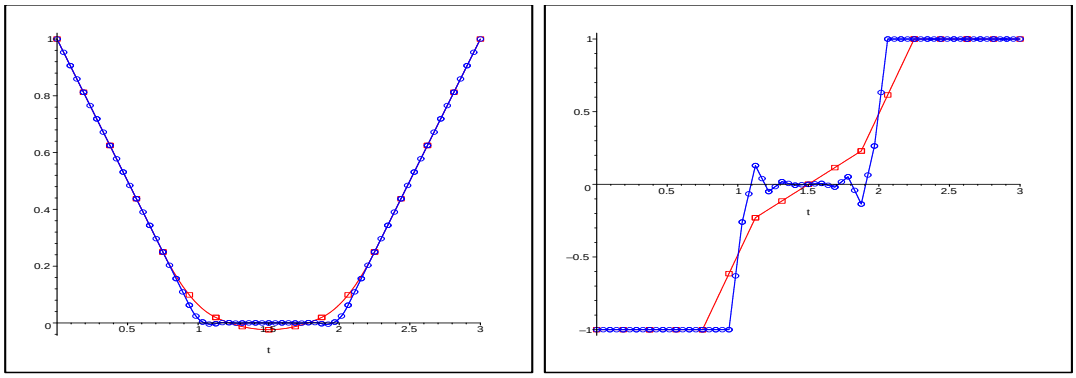

Abbildung 8.15: Bsp. 8.8. Zustand (links), Steuerung (rechts) bei $s = 3$ und $N = 8$ (rot) beziehungsweise $N = 32$ (blau)

Der Optimierer findet für $N = 32$ kein Optimum, aber seine letzte interne Variablenbelegung soll auch hier als Optimum gewertet werden. Der Zustand wird für $N = 32$ exakt approximiert. Die Steuerung ist nur noch in kleinen Intervallen um die Schaltpunkte unzutreffend approximiert. Die Verbesserung gegenüber $N = 8$ ist klar ersichtlich.
Dieses Beispiel ist erneut glänzend geeignet für eine Entkopplung der polynomialen Steuerungsintervalle. Die Abbildung 8.16 visualisiert die numerischen Lösungen für $N = 6$ und $s = 3$.

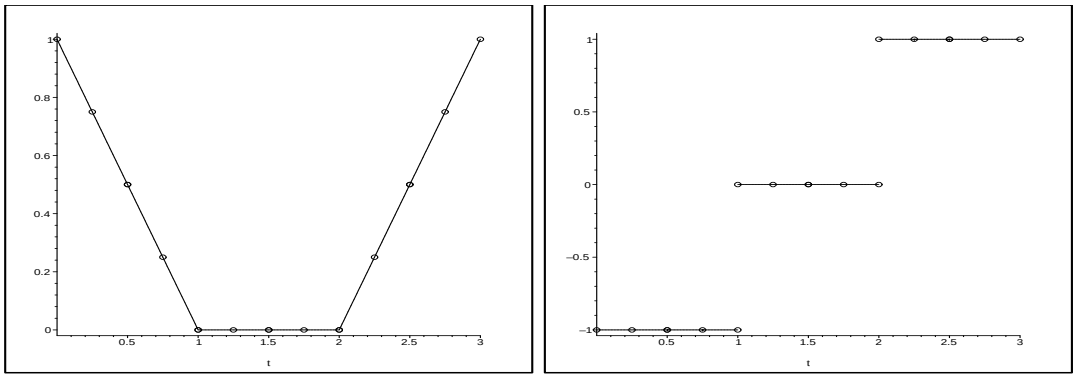

Abbildung 8.16: Bsp. 8.8. Zustand (links) und Steuerung (rechts) für $N = 6$ und $s = 3$ und unstetig angesetzte Steuerung

Offenkundig werden die optimalen Trajektorien exakt getroffen. Zusätzlich demonstrieren die Graphen in 8.16 bei $N = 6$ aber auch, daß die erlaubten Unstetigkeitsstellen bei $t = 0.5$, $t = 1.5$ und $t = 2.5$ nicht auftreten. Überdies sehen diese Übergänge graphisch glatt aus. Der Zielfunktionswert ist kaum verbesse-

rungsfähig:

$$J_3^6 = 0.333333333337136, \tag{8.15}$$

$$\Longrightarrow J_3^6 - \hat{J} = 3.803 \cdot 10^{-12}. \tag{8.16}$$

Die graphische Approximation bestätigt sich in der Tabelle 8.15.

	Zeitpunkt	Alle Werte als Abweichungsbeträge	
Variable	t	Zustand	Steuerung
1	0.00	$0.000E+00$	$4.820E-12$
2	0.25	$1.476E-12$	$6.996E-12$
3	0.50	$3.497E-12$	$9.171E-12$
4	0.50	$3.497E-12$	$2.389E-11$
5	0.75	$1.490E-11$	$6.733E-11$
6	1.00	$3.716E-11$	$1.108E-10$
7	1.00	$3.716E-11$	$2.498E-09$
8	1.25	$2.121E-10$	$1.099E-09$
9	1.50	$5.124E-10$	$4.697E-09$
10	1.50	$5.124E-10$	$1.955E-09$
11	1.75	$1.307E-10$	$1.099E-09$
12	2.00	$3.711E-11$	$2.435E-10$
13	2.00	$3.711E-11$	$1.106E-10$
14	2.25	$1.489E-11$	$6.723E-11$
15	2.50	$3.497E-12$	$2.388E-11$
16	2.50	$3.497E-12$	$9.169E-12$
17	2.75	$1.476E-12$	$6.995E-12$
18	3.00	$0.000E+00$	$4.820E-12$

Tabelle 8.15: Bsp. 8.8. $N = 6$ und $s = 3$ bei unstetiger Steuerung

8.5 Zusammenfassung der Beispiele

Dieser Abschnitt komplettiert das Kapitel. Es wird versucht, Erkenntnisse aus den Beispielen zu abstrahieren. Bei den einzelnen Beobachtungen wird auf ausgewählte Beispiele verwiesen. Oftmals sind die jeweiligen Merkmale auch bei anderen Beispielen anzutreffen. Viele Beobachtungen sind eindeutig und schlüssig. Andere Folgerungen sind nicht so naheliegend und bedürften einer tiefergehenden Analyse.
Resümierend lassen sich folgende Schlüsse ziehen:

- Für den Optimierer ist der Startwert des Algorithmus eine essentielle Information, siehe Beispiel 8.2.

- Genaue Knoten- und Gewichtsangaben sind wesentlich, wie die Tabelle 8.9 im Beispiel 8.2 demonstriert.

- Bei glatten optimalen Trajektorien ist die Approximation bereis bei geringen Polynomgraden effektiv, siehe Beispiele 8.2, 8.3 und 8.7. Es scheinen dann hohe Konvergenzordnungen aufzutreten, wie die Tabellen 8.5, 8.6 und 8.7 nahelegen.

- Die Näherung der Steuerung ist auch bei glatten optimalen Funktionen schlecher als diejenige für den Zustand, wie beispielsweise die Tabellen 8.5, 8.6 und 8.7 im Beispiel 8.2 verraten.

- Eine höhere Anzahl von Intervallen N ist bei glatten Problemen nicht ganz entscheidend und zudem nicht unproblematisch, da Rundungsfehler in der Matrix D, die erhöhte Anzahl der Variablen und Gleichungen sowie der nichtglatte Ansatz für die Trajektorien x und u Risiken beinhalten.

- Nach der Kollokationsidee in Definition 5.1 und der Bemerkung 5.11 könnte man vermuten, daß das numerische Lösungspolynom an den Kollokationspunkten und am Intervallende besonders gut approximiert. Die Abbildung 8.3 im Beispiel 8.3 bestätigt dies nicht. Gleichwohl müssen die geringen absoluten Unterschiede in Abbildung 8.3 mit großer Vorsicht beurteilt werden.

- Das Optimierungsprogramm rechnet mit diskreten Werten, nicht mit den kontinuierlichen Polynomen. Gerade im Sinne eines Runge-Kutta-Verfahrens stellt die polynomiale Interpolation der diskreten Werte nur eine Interpretation des Anwenders dar. Zwischen den diskreten Lösungspunkten könnte das Polynom erratisch schwingen, ohne Beachtung der kontinuierlichen optimalen Lösungen. Nun bestätigt sich aber der folgenreiche Satz 5.7, daß ein Kollokationsverfahren auch jenseits der Stützstellen approximiert. Ein Eindruck davon vermittelt die Abbildung 8.6 des Beispiels 8.5, wo das Polynom 8. Grades nicht nur die $s = 9$ diskreten Werte erfolgreich beschreibt, sondern sich auch an die stückweise affine Funktion anschmiegt.

- Eine höhere Anzahl von Intervallen N ist wünschenswert und erfolgversprechend, wenn Lösungen stärker oszillieren oder sogar Einschränkungen der Glattheit der Optimallösungen vorliegen.

- In Intervallbereichen ohne Steuersprünge arbeitet der Optimierer sehr effektiv, wie die Abbildungen 8.9, 8.11 und auch 8.8 des Beispiels 8.6 zeigen.

- Die optimale Zustandstrajektorie wird selbst bei Sprüngen der Steuerung relativ gut getroffen, wie man in der Abbildung 8.14 im Beispiel 8.8 erkennen kann. Auch der Zielfunktionswert wird relativ unabhängig von der Steuerfunktion erfolgreich erreicht. Dies wird im Beispiel 8.8 deutlich, noch stärker aber in der Tabelle 8.11 des Beispiels 8.4. Gleichzeitig erschwert vielleicht auch diese Tatsache dem Optimierer eine effektive Bearbeitung der Steuerungen in den Sprungumgebungen.

- Befindet sich in einem Intervall (7.12) eine Sprungstelle der Steuerung, so werden von den s Punkten dieses Intervalls stets die Punkte um die Sprungstelle falsch gesetzt, und zwar scheinbar so, daß beim Polynom keine abrupten Steigungsschwankungen auftreten. Die weiter entfernten Werte liegen durchaus richtig. Diese Erscheinung kann man bei allen derartigen Abbildungen beobachten, zum Beispiel in Abbildung 8.8 des Beispiels 8.6 oder in den Abbildungen 8.14 und 8.15 des Beispiels 8.8. Dies verwundert, da der Optimierer schließlich nicht mit den kontinuierlichen Polynomen, sondern mit diskreten Werten rechnet, und somit nicht zu solchen Polynomen verpflichtet ist. Diese Charakteristik weist darauf hin, daß die Gleichungen (7.24) der Kollokationsbedingungen der Simulation einer unstetigen Funktion entgegenwirken.

Anhang A

Ergänzungen zu Kapitel 8

Dieser Anhang ergänzt das Kapitel 8.

Die Abschnitte A.1 und A.2 liefern nähere Informationen zur Bedienung der Software SCPIP 3.0.
Die Abschnitte A.3 und A.4 enthalten wichtige Maple 9.5-Befehle.
Die Abschnitte A.5 und A.6 tabellieren ausgegliederte Beispieldaten.

A.1 Verwendete Einstellungen in SCPIP 3.0

Im Abschnitt 8.2 wurden Auswahlmöglichkeiten von Einstellungen im *main*-Kopf der Datei *v30main.f* des Programms SCPIP 3.0 erwähnt. Für die Beispiele des Abschnitts 8.4 wurden die untenstehenden Einstellungen dauerhaft festgelegt.

```
icntl(1) = 1
icntl(2) = 1
icntl(3) = 10000
icntl(4) = 3
icntl(5) = 10
icntl(6) = 0
icntl(11) = 0
icntl(12) = 0
icntl(13) = 0

rcntl(1) = 1.d-7
rcntl(2) = 1.d30
rcntl(3) = 1.d30
rcntl(4) = 0.d0
rcntl(5) = 0.d0
rcntl(6) = 0.d0

ierr = 0
nout = 7
rdim = rd
rsubdim = rd2
idim = id
isubdim = id2
spiwdim = id3
spdwdim = rd3

mode = 2

spstrat = 1
linsys = 1
```

Lediglich im Beispiel 8.5 wurde *icntl(3)*, das die Anzahl der Iterationen des Optimierungsalgorithmus von SCPIP 3.0 angibt, variiert. Gerade diese Einstellung verursacht aber keine einschneidenden Konsequenzen. *icntl(3)* dient als Obergrenze und bewirkt, daß das Programm abbricht, wenn es nach zahlreichen Iterationen immer noch keine Optimallösung gefunden hat (vgl. Beispiel 8.5).

A.2 Software-Beschreibung

Im Quellcode der Dateien *Aufgabendaten.f* und *Hilfsfunktionen.f* unterscheiden sich manche Bezeichnungen von denjenigen der Probleme 7.5 (bzw. 7.8 und 7.12). So ist in SCPIP 3.0 „n“ für die Variablenanzahl reserviert, womit n nicht als globale Variable für die Zustandsdimension gewählt werden kann, sondern lediglich lokal. Da Fortran77 nicht zwischen Groß- und Kleinschreibung unterscheidet, scheidet gleichzeitig „N“ als Parameter für die Anzahl der Intervalle aus. Auch wurde auf einige Vorfaktoren der Aufgaben der Definition 6.15 und des Problems 7.5 verzichtet.
Die folgende Aufgabenstellung hilft beim Verständnis der Tabellen A.1 und A.2. In der Notation A.1 sind die Bezeichnungen des Linear-Quadratischen Optimalsteuerungsproblems in SCPIP 3.0 niedergeschrieben.

Notation A.1 (Aufgabenstellung in SCPIP 3.0).
Minimiere

$$VorS \cdot \Big(x(tf)^T Stf\, x(tf) + mtf^T x(tf) \Big)$$
$$+ \quad VorI \cdot \int_{t0}^{tf} \big(x(t)^T Qt(t)\, x(t) + u(t)^T R\, t(t) u(t) + ht(t)^T x(t) + kt(t)^T u(t) \big)\, dt$$

unter den Nebenbedingungen

$$\begin{aligned}
\dot{x}(t) &= At(t)x(t) + Bt(t)u(t) \quad \text{für fast alle } t \in [t0, tf], \\
x(t0) &= xt0, \\
x(tf) &= xtf, \quad \text{falls } Endbed = 1, \\
lu \leq\ & u \ \leq uu, \quad \forall u, \\
\|u\|^2 &\leq 1, \quad \text{falls } Kugel = 1.
\end{aligned}$$

Weitere Unterschiede werden aus den Tabellen A.1 und A.2 ersichtlich. Nach Bedienung der Datei *Aufgabendaten.f* muß der Nutzer keine Eingaben mehr vornehmen. Zielfunktion, Nebenbedingungen und Gradienten werden selbstständig in den beiden Subroutinenen *scpfct* und *scpgrd* der Datei *v30main.f* generiert. Änderungen im *main*-Kopf von *v30main.f* sind unabhängig von der hiesigen Implementierung und lassen sich [64] entnehmen.

Aufgabendaten.f		
Subroutine	Variablen	Beschreibung
Dimensionen	s	Polynomgrad $s-1$
	Dis	Intervallanzahl N
	$Dimx$	Dimension von x (Zustand)
	$Dimu$	Dimension von u (Steuerung)
	$Endbed$	Randwert: (ja/nein)= (1/0)
	$Kugel$	Kugelsteuerbeschränkung: (ja/nein)= (1/0)
	$Koppu$	Stetige Steuerung: (ja/nein)= (1/0)
Intervall	$t0$	Anfangszeit
	tf	Endzeit
ZF1	n	Dimension von x (Zustand)
	$VorS$	Vorfaktor für Stf und mtf
	Stf	Matrix S
	mtf	Vektor l
ZF2	n	Dimension von x (Zustand)
	m	Dimension von u (Steuerung)
	$VorI$	Vorfaktor des Integrals
	Qt	Matrix Q
	Rt	Matrix R
	ht	Vektor h
	kt	Vektor k
DGL	n	Dimension von x (Zustand)
	m	Dimension von u (Steuerung)
	At	Matrix A
	Bt	Matrix B
	$xt0$	Anfangswert x_0
	xtf	Randwert x_f
	t	Zeit
	li	Aktuelles Intervall
BC	lu	Untere Steuerbeschränkungsschranken
	uu	Obere Steuerbeschränkungsschranken

Tabelle A.1: Details der Datei *Aufgabendaten.f*

Hilfsfunktionen.f		
Subroutine	Variablen	Beschreibung
LobattoNS	s	Polynomgrad $s-1$
	t	Nullstellen der Lobatto-Polynome
LobattoGew	s	Polynomgrad $s-1$
	w	Gewichte der Lobatto-Integration
Legendre	s	Polynomgrad $s-1$
	t	Zeit
	wert	Wert von $L_s(t)$
LobattoDiffMatD	m	Polynomgrad $m-1$
	ti	Zeit
	D	Matrix D
Subfaktor	*subst*	Substitutionsfaktor durch Transformation auf $[-1,1]$
Translation	*li*	Intervallnumerierung
	$t0$	Anfangszeit
	tf	Endzeit
	t	Aktuelle, standardisierte Zeit

Tabelle A.2: Details der Datei *Hilfsfunktionen.f*

A.3 Gewinnung von Gauß-Quadratur-Daten

In diesem Abschnitt wird die Gewinnung von Daten zur Gauß-Quadratur beschrieben, die für den numerischen Lösungsansatz in SCPIP 3.0 benötigt werden. Folgende Daten werden gebraucht:

- die Nullstellen der Lobatto-Polynome,
- die Gewichte der Lobatto-Integration,
- die Legendre-Polynome für die Matrix D.

Die Lobatto-Polynome entstehen aus den Legendre-Polynomen, die in verschiedenen Mathematikprogrammen implementiert sind. Die Formeln der Tabelle 2.2

gestatten einfache Befehle in Maple 9.5 mit dem Paket *orthopoly*. Die Koeffizienten der Legendre-Polynome ermittelt man mit

```
> with(orthopoly):
> for i from 1 to 12 do print(P(i,x)) end do;
```

Die Lobatto-Polynome lassen sich nach Satz 2.23 durch $L_s - L_{s-2}$ darstellen. Unten ist ein einfaches Beispiel angegeben, wie man in Maple 9.5 zum Polynomgrad 5 die Nullstellen des Lobatto-Polynoms erzeugt:

```
> with(orthopoly):
> evalf(solve(P(5,x)-P(5-2,x)=0,x),20);
  0., -1., 1., 0.65465367070797714379, -0.65465367070797714379
```

Die Genauigkeit (hier 20) läßt sich beliebig steuern. Über den Satz 2.25 konstruiert man dann unmittelbar die Gewichte für $s = 5$:

```
> with(orthopoly):
> s := 5:
> t := Vector([0., -1., 1., .65465367070797714379,
                -.65465367070797714379]):
> for j from 1 to s do print(evalf(2.0/(s*(s-1)*(P(s-1,t[j])^2)),
  20)) end do;
  .71111111111111112, .1, .1, .54444444444444450,.54444444444444450
```

Beim Computeralgebrasystem Maple hängt die Genauigkeit der Gewichte im wesentlichen von der Genauigkeit der Nullstellen ab.
In [7, Appendix A] findet man Matlab-Befehle für die Berechnung von Nullstellen und Gewichten zu Gauß-Legendre-Verfahren. Diese Befehle lassen sich leicht auf die Gauß-Lobatto-Verfahren übertragen.

A.4 Graphische Darstellung

Die Ausgabedatei *SCPERG3.txt* wurde speziell für Maple-Graphiken konstruiert. Die Ergebnisse lassen sich aus der Datei *SCPERG3.txt* direkt in die geschriebene Maple-Datei kopieren.
Die folgenden Maple 9.5-Befehle erstellen für ein beliebiges Intervall $[t_0, t_f]$, den Polynomgrad $s - 1$ und die Diskretisierung N eine Graphik mit Datenpunkten und stetigem Funktionsverlauf. Hier wurde beispielhaft $[t_0, t_f] = [0, 2]$, $s = 3$ und $N = 2$ gewählt sowie sechs Werte eingesetzt.

```
> with(CurveFitting):
> with(plots):
> Polynomgrad := [[0],[0,0],[-1.0,0,1.0],
  [-1,-0.44721359549995793928,0.44721359549995793928,1],
  [-1,-0.65465367070797714379,0,0.65465367070797714379,1],...]:
# Knoten verschiedener Gauß-Lobatto-Regeln
> tf := 2.0:
> t0 := 0.0:
> s := 3:
> Dis := 2:
> Num_Lös :=
  [
  [
           1.000000000000000    ,
          0.7066921311200736    ,
          0.5114282475531583
  ],
  [
          0.5114282475531583    ,
          0.3920877406939194    ,
          0.3265500019370219
  ]
  ]:
# Numerische Lösungen aus der Datei SCPERG3.txt
> for i from 1 to Dis do x_stand[i]:=
  PolynomialInterpolation(Polynomgrad[s],Num_Lös[i],t ) end do:
# Polynomiale Interpolation der Datenpunkte
> for i from 1 to Dis do x_orig[i] :=
  subs(t=2*Dis*t/(tf-t0)+1-2*i-2*t0*Dis/(tf-t0),x_stand[i])
  end do:
# Transformation auf die Intervalle der Formel (7.17)
> for i from 1 to Dis do PG[i] :=
  pointplot({seq([((tf-t0)/(2*Dis))*(Polynomgrad[s][j])+
  ((2*i-1+((2*Dis*t0)/(tf-t0)))*((tf-t0)/(2*Dis))),
  Num_Lös[i][j]],j=1..s)},color=blue) end do:
# Plot der Datenpunkte (aufgeschoben)
> display([seq(plot(x_orig[i](t),
  t= t0+(i-1)*(tf-t0)/Dis..t0+i*(tf-t0)/Dis),i=1..Dis),seq(PG[i],
  i=1..Dis)]);
# Plot der Polynome gemeinsam mit den Datenpunkten
```

A.5 Daten zu Beispiel 8.6

Für die Abbildung 8.12 wurden die 50 Vektoren $(c_1, c_2)^T$ der Tabelle A.3 verwendet. Die 50 Vektoren sind einer Auswahl von 201 Vektoren einer Datei entnommen, die Dr. Baier freundlicherweise zur Verfügung gestellt hat. Auf der beigelegten CD findet man sie unter *SCPIP 3.0\Diplomarbeits-Beispiele des Abschnitts 8.4\8.6\Korrekte Lösungen\BSP_11_ref.* Die Datei listet numerische Ergebnisse für die Endzustände $x_1(1)$ und $x_2(1)$ auf. Diese wurden mit einem mengenwertigen Kombinationsverfahren berechnet. Ein solches kombiniert eine mengenwertige Quadraturformel mit dem zugehörigen punktwertigen Einschrittverfahren, hier die iterierte Trapezregel mit dem Euler-Cauchy-Verfahren, siehe hierzu [4] zusammen mit [5]. Wegen der hohen numerischen Präzision werden diese Werte als exakte Lösungen angesehen. Die 50 ausgewählten Vektoren sind mit Bedacht ausgesucht worden, da bei ihnen starke Unterschiede in den Endzuständen erkennbar sind.

A.6 Daten zu Beispiel 8.7

Für die Abbildung 8.13 wurden die 50 Vektoren $(c_1, c_2)^T$ der Tabelle A.4 verwendet. Die 50 Vektoren sind einer Auswahl von 201 Vektoren einer Datei entnommen, die Dr. Baier freundlicherweise zur Verfügung gestellt hat. Auf der beigelegten CD findet man sie unter *SCPIP 3.0\Diplomarbeits-Beispiele des Abschnitts 8.4\8.7\Korrekte Lösungen\BSP_12_ref.* Die Datei listet numerische Ergebnisse für die Endzustände $x_1(2)$ und $x_2(2)$ auf. Diese wurden mit einem mengenwertigen Kombinationsverfahren berechnet. Ein solches kombiniert eine mengenwertige Quadraturformel mit dem zugehörigen punktwertigen Einschrittverfahren, hier die iterierte Simpsonregel mit dem klassischen Runge-Kutta-Verfahren, siehe hierzu [4] zusammen mit [5]. Wegen der hohen numerischen Präzision werden diese Werte als exakte Lösungen angesehen. Die 50 ausgewählten Vektoren sind mit Bedacht ausgesucht worden, da bei ihnen starke Unterschiede in den Endzuständen sichtbar sind.

	c_1	c_2
1	1.0000000000000000	0.0000000000000000
2	−0.7289686274214113	0.6845471059286888
3	−0.7501110696304596	0.6613118653236518
4	−0.7705132427757891	0.6374239897486899
5	−0.7901550123756904	0.6129070536529764
6	−0.8090169943749473	0.5877852522924732
7	−0.8270805742745617	0.5620833778521308
8	−0.8443279255020151	0.5358267949789967
9	−0.8607420270039435	0.5090414157503714
10	−0.8763066800438636	0.4817536741017152
11	−0.8910065241883678	0.4539904997395469
12	−0.9048270524660194	0.4257792915650729
13	−0.9177546256839811	0.3971478906347806
14	−0.9297764858882513	0.3681245526846781
15	−0.9408807689542255	0.3387379202452913
16	−0.9510565162951535	0.3090169943749475
17	−0.9602936856769430	0.2789911060392296
18	−0.9685831611286311	0.2486898871648548
19	−0.9759167619387473	0.2181432413965428
20	−0.9822872507286887	0.1873813145857246
21	−0.9876883405951377	0.1564344650402310
22	−0.9921147013144778	0.1253332335643045
23	−0.9955619646030800	0.0941083133185144
24	−0.9980267284282716	0.0627905195293136
25	−0.9995065603657316	0.0314107590781282
26	−1.0000000000000000	0.0000000000000001
27	0.7289686274214112	−0.6845471059286889
28	0.7501110696304591	−0.6613118653236523
29	0.7705132427757894	−0.6374239897486896
30	0.7901550123756903	−0.6129070536529765
31	0.8090169943749473	−0.5877852522924734
32	0.8270805742745616	−0.5620833778521309
33	0.8443279255020147	−0.5358267949789971
34	0.8607420270039438	−0.5090414157503712
35	0.8763066800438636	−0.4817536741017153
36	0.8910065241883678	−0.4539904997395470
37	0.9048270524660194	−0.4257792915650730
38	0.9177546256839809	−0.3971478906347812
39	0.9297764858882515	−0.3681245526846779
40	0.9408807689542255	−0.3387379202452914
41	0.9510565162951535	−0.3090169943749476
42	0.9602936856769430	−0.2789911060392297
43	0.9685831611286310	−0.2486898871648553
44	0.9759167619387474	−0.2181432413965424
45	0.9822872507286887	−0.1873813145857247
46	0.9876883405951377	−0.1564344650402311
47	0.9921147013144778	−0.1253332335643046
48	0.9955619646030800	−0.0941083133185149
49	0.9980267284282716	−0.0627905195293133
50	0.9995065603657316	−0.0314107590781284

Tabelle A.3: Für die Abbildung 8.12 verwendete Vektoren $(c_1, c_2)^T$

	c_1	c_2
1	1.0000000000000000	0.0000000000000000
2	0.8607420270039436	0.5090414157503713
3	0.7501110696304596	0.6613118653236518
4	0.6845471059286886	0.7289686274214116
5	0.6613118653236519	0.7501110696304595
6	0.6374239897486897	0.7705132427757893
7	0.6129070536529765	0.7901550123756904
8	0.5877852522924732	0.8090169943749473
9	0.5620833778521307	0.8270805742745618
10	0.5358267949789965	0.8443279255020151
11	0.5090414157503712	0.8607420270039436
12	0.4817536741017152	0.8763066800438637
13	0.4539904997395469	0.8910065241883678
14	0.4257792915650727	0.9048270524660196
15	0.3971478906347806	0.9177546256839811
16	0.3681245526846781	0.9297764858882513
17	0.3090169943749475	0.9510565162951535
18	0.2181432413965427	0.9759167619387473
19	0.1253332335643045	0.9921147013144778
20	0.0627905195293135	0.9980267284282716
21	0.0000000000000001	1.0000000000000000
22	−0.1253332335643041	0.9921147013144779
23	−0.2789911060392292	0.9602936856769431
24	−0.4817536741017154	0.8763066800438635
25	−0.8270805742745616	0.5620833778521308
26	−1.0000000000000000	0.0000000000000001
27	−0.8607420270039436	−0.5090414157503712
28	−0.7501110696304597	−0.6613118653236517
29	−0.7071067811865479	−0.7071067811865471
30	−0.6845471059286886	−0.7289686274214116
31	−0.6613118653236519	−0.7501110696304595
32	−0.6374239897486895	−0.7705132427757894
33	−0.6129070536529765	−0.7901550123756904
34	−0.5877852522924732	−0.8090169943749473
35	−0.5620833778521309	−0.8270805742745616
36	−0.5358267949789963	−0.8443279255020153
37	−0.5090414157503711	−0.8607420270039438
38	−0.4817536741017153	−0.8763066800438636
39	−0.4539904997395469	−0.8910065241883678
40	−0.4257792915650722	−0.9048270524660198
41	−0.3971478906347803	−0.9177546256839813
42	−0.3681245526846778	−0.9297764858882515
43	−0.3090169943749476	−0.9510565162951535
44	−0.2181432413965424	−0.9759167619387474
45	−0.0941083133185140	−0.9955619646030800
46	0.0941083133185145	−0.9955619646030800
47	0.3090169943749472	−0.9510565162951536
48	0.5358267949789968	−0.8443279255020150
49	0.7901550123756903	−0.6129070536529765
50	1.0000000000000000	−0.0000000000000002

Tabelle A.4: Für die Abbildung 8.13 verwendete Vektoren $(c_1, c_2)^T$

Anhang B

Material auf der beiliegenden CD

Die dieser Diplomarbeit beigefügte CD enthält neben diesen Ausführungen im pdf- und ps-Format das modifizierte SCPIP 3.0-Programm mit den Beispielrechnungen und Graphiken des Abschnitts 8.4. Zu den SCPIP-Unterlagen gehören auch Informationen und Beispiele zur Erstellung des Programms und der Beispiele, etwa Informationen zur Gewinnung von Gauß-Quadratur-Nullstellen und -Gewichten. Ferner werden auch verschiedenste pdf-Dokumente bereitgestellt, die in die Diplomarbeit eingeflossen sind. Als Ergänzungen und Hilfsapparate werden auch ein paar Skripten über Fortran und LaTeX zur Verfügung gestellt.
Die Ordner der CD tragen Titel, die in der Regel schon auf den jeweiligen Inhalt hinweisen. Zahlreiche Ordner informieren zusätzlich mit einer *Info.txt*-Datei. Die CD enthält folgende (Ober-)Verzeichnisse:

Diplomarbeit:
Graphiken und LaTeX-Datei der Diplomarbeit.
SCPIP 3.0:
Umfaßt alles, was mit SCPIP 3.0 und den Beispielen des Abschnitts 8.4 zusammenhängt. Wichtig ist vor allem die Ordner *ZIM*, der das fertige Programm beinhaltet. Der Ordner *Zeichnen von Graphiken* erklärt das Erstellen von zugehörigen Graphiken, während *Lobatto- und Legendre-Daten* die Herkunft der Nullstellen und Gewichte erläutert. *Diplomarbeits-Beispiele des Abschnitts 8.4* enthält die Daten, Ergebnisse, Rechnungen und Zeichnungen der Beispiele aus Abschnitt 8.4.
Verschiedene LaTeX-Skripten:
Verschiedene LaTeX-Skripten.
Verwendete Mathematik-Dokumente:
Zahlreiche pdf-Dokumente. Unterteilt in vier weitere Ordner. Besonders wichtig sind die Ordner *Skripte (Universität Bayreuth)* und *Arbeiten und Artikel im Internet*.

Literaturverzeichnis

[1] Abramowitz, Milton und Stegun, Irene A.: *Handbook of Mathematical Functions, with Formulas, Graphs and Mathematical Tables*, manufactured in the United States of America, Dover Publications, Inc. 180 Varick Street, New York, N.Y. 10014, 1972.

[2] Anderson, Brian D.O. und Moore, John B.: *Optimal Control: Linear Quadratic Methods*, 1990 by Prentice-Hall, Inc. A Division of Simon & Schuster. Englewood Cliffs, New Jersey 07632.

[3] Aulbach, Bernd: *Gewöhnliche Differentialgleichungen*, 2. Auflage, Elsevier GmbH, München 2004.

[4] Baier, Robert: *Mengenwertige Integration und die diskrete Approximation erreichbarer Mengen*, Bayreuther Mathematische Schriften, Heft 50, Universität Bayreuth 1995.

[5] Baier, Robert und Lempio, Frank: *Computing Aumann's integral*, Modeling Techniques for Uncertain Systems, Proceedings of a Conference held in Sopron, Hungary, July 1992, pages 71-92, Boston-Basel-Berlin, 1994, Birkhäuser.

[6] Bellman, Richard: *Dynamic Programming*, Princeton University Press, Princeton, New Yersey, 1957.

[7] Benson, David: *A Gauss Pseudospectral Transcription for Optimal Control*, M.S. Aerospace Engineering, University of Colorado, 2001, B.S. Aerospace Engineering, University of Colorado, 2000, Massachusetts Institute of Technology, Februar 2005, http://dspace.mit.edu/bitstream/1721.1/28919/1/60495690.pdf.

[8] Berkovitz, Leonard David: *Optimal Control Theory*, Applied Mathematical Sciences 12, Springer-Verlag New York, 1974.

[9] Bernardi, Christine und Maday, Yvon: *Approximations spectrales de problèmes aux limites elliptiques*, Mathématiques & Applications 10, Springer-Verlag France, Paris, 1992.

[10] Bertsekas, Dimitri P., *Dynamic Programming and Optimal Control, Volume I*, Second Edition, Athena Scientific, Belmont, Massachusetts, 2000.

[11] Bryson, Arthur Earl Jr. und Ho, Yu-Chi: *Applied optimal control*, Hemisphere Publishing Corporation, 1025 Vermont Ave., N.W., Washington, D.C. 20005, 1975.

[12] Butcher, John Charles: *The numerical analysis of ordinary differential equations: Runge-Kutta and general linear methods*, John Wiley & Sons Ltd. 1987.

[13] Butcher, John Charles: *Numerical Methods for Ordinary Differential Equations*, John Wiley & Sons, Ltd, The Atrium, Southern Gate, Chichester, West Sussex P019 8SQ, England.

[14] Butcher, John Charles: *Implicit Runge-Kutta Processes*, Mathematics of Computation, 18, 50-64, 1964.

[15] Chudej, Kurt: *Optimale Steuerung von gewöhnlichen und partiellen Differentialgleichungen*, Vorlesungsskript Wintersemester 2004/05, Bayreuth.

[16] Clarke, Frank H.: *Optimization and Nonsmooth Analysis*, 1983 by John Wiley & Sons, Inc..

[17] Deuflhard, Peter und Hohmann, Andreas: *Numerische Mathematik I*, 3. überarbeitete und erweiterte Auflage, Walter de Gruyter, Berlin, New York 2002.

[18] Deuflhard, Peter und Bornemann, Folkmar: *Numerische Mathematik II*, 2. Auflage, Walter de Gruyter, Berlin, New York 2002.

[19] Elnagar, Gamal N. und Razzaghi, Mohsen: *A collocation-type method for linear quadratic optimal control problems*, Optimal Control Applications & Methods, Vol. 18, 227-235, 1997, http://www3.interscience.wiley.com/cgi-bin/fulltext/7878/PDFSTART.

[20] Elnagar, Gamal N.: *Optimal Control Computation for Integro-Differential Aerodynamic Equations*, Mathematical Methods in the Applied Sciences Volume 21, Issue 7, 10 May 1998, Pages: 653-664. http://www3.interscience.wiley.com/cgi-bin/fulltext/92012932/7.pdf.

[21] Engels, Herrmann: *Numerical Quadrature and Cubature*, Academic Press Inc. (London) Ltd, 24-28 Oval Road, London NW1, 1980.

[22] Evans, Michael und Swartz, Tim: *Approximating Integrals via Monte Carlo and Deterministic Methods*, Oxford Statistical Science Series 20, Oxford University Press, Great Clarendon Street, Oxford OX2 6DP, 2000.

[23] Forster, Otto: *Analysis 1*, 6., verbesserte Auflage, Friedr. Vieweg & und Sohn Verlagsgesellschaft mbh, Braunschweig/Wiesbaden 2001.

[24] Forster, Otto: *Analysis 2*, 5., durchgesehene Auflage, Friedr. Vieweg & Sohn Verlagsgesellschaft mhH, Braunschweig/Wiesbaden, 1984.

[25] Gerdts, Matthias und Lempio, Frank: *Mathematische Methoden des Operations Research*, http://www.uni-bayreuth.de/departments/math/∼flempio/, 2006.

[26] Grüne, Lars: *Numerische Mathematik I*, Universität Bayreuth, Vorlesungsskript Wintersemester 2002/2003, http://www.uni-bayreuth.de/departments/math/∼lgruene/.

[27] Grüne, Lars: *Numerische Mathematik II*, Universität Bayreuth, Vorlesungsskript Sommersemester 2003, http://www.uni-bayreuth.de/departments/math/∼lgruene/.

[28] Hairer, Ernst und Wanner, Gerhard und Nørsett, Syvert Paul: *Solving Differential Equations I*, Springer Verlag Berlin Heidelberg 1993.

[29] Hairer, Ernst und Wanner, Gerhard: *Solving Differential Equations II*, Springer Verlag Berlin Heidelberg, 2., revidierte Auflage, 1996.

[30] Hämmerlin, Günther und Hoffmann, Karl-Heinz: *Numerische Mathematik*, 4. Auflage, Springer-Verlag 1994.

[31] Harville, David A.: *Matrix Algebra from a statistican's perspective*, Springer-Verlag New York Inc., 1997.

[32] Hermann, Martin: *Numerik gewöhnlicher Differentialgleichungen*, Oldenbourg Wissenschaftsverlag GmbH München 2004.

[33] Infeld, Samantha A.: *Optimization of Mission Design for Constrained Libration Point Space Missions*, Department of Aeronautics and Austronautics Stanford University, December 2005, http://www.stanford.edu/group/SOL/dissertations/samantha-thesis.pdf.

[34] Ioffe, Aleksandr D. und Tihomirov, Vladimir M.: *Theory of Extremal Problems, volume 6 of Studies in Mathematics and Applications*, North-Holland Publishing Company, Amsterdam-New York-Oxford, 1979.

[35] Natanson, Isidor P.: *Theorie der Funktionen einer reellen Veränderlichen*, Akademie-Verlag, Berlin, 1961.

[36] Jaddu, Hussein M., *Numerical Methods for Solving Optimal Control Problems Using Chebyshev Polynomials*, School of Information Science, Japan Advanced Institute of Science and Technology, September 1998, http://www.jaist.ac.jp/library/thesis/is-doctor-1999/paper/jaddu/paper.pdf.

[37] Jarre, Florian und Stoer, Josef: *Optimierung*, Springer-Verlag Berlin Heidelberg, 2004.

[38] Kaballo, Winfried: *Einführung in die Analysis II*, Spektrum Akademischer Verlag GmbH Heidelberg Berlin Oxford, 1997.

[39] Königsberger, Konrad: *Analysis 2*, 3., überarbeitete Auflage, Springer-Verlag Berlin Heidelberg 1993, 1997, 2000.

[40] Kronrod, Aleksandr S.: *Nodes and weights of quadrature formulas*, New York, Consultants Bureau, 1966.

[41] Lempio, Frank: *Numerische Mathematik I, Methoden der Linearen Algebra*, Bayreuther Mathematische Schriften, Heft 51, Universität Bayreuth 1997.

[42] Lempio, Frank: *Numerische Mathematik II, Methoden der Analysis*, Bayreuther Mathematische Schriften, Heft 55, Universität Bayreuth 1998.

[43] Locatelli, Arturo: *Optimal Control, An Introduction*, 2000 Mathematical Subject Classification 49-01, 2001 Birkhäuser Verlag, P.O. Box 133, CH-4010 Basel, Switzerland.

[44] Patterson, T.N.L.: *The optimum addition of points to quadrature formulae*, Mathematics of Computation 22, 1968, 847-856.

[45] Piessens, Robert und Doncker-Kapenga, Elise de und Überhuber, Christoph W. und Kahaner, David K.: *Quadpack, A Subroutine Package for Automatic Integration*, Springer-Verlag Berlin Heidelberg 1983.

[46] Pietz, Jesse A., *Pseudospectral Collocation Methods for the Direct Transcription of Optimal Control Problems*, Rice University, Houston, Texas, 2003, http://www.caam.rice.edu/caam/trs/2003/TR03-10.pdf.

[47] Quarteroni, Alfio und Sacco, Riccardo und Saleri, Fausto: *Numerische Mathematik 2*, Springer-Verlag Berlin Heidelberg 2002.

[48] Rudin, Walter: *Analysis*, R. Oldenbourg Verlag München Wien 1998.

[49] Schmeisser, Gerhard und Schirmeier, Horst: *Praktische Mathematik*, de Gruyter, Berlin, 1976.

[50] Schönhage, Arnold: *Approximationstheorie*, Walter de Gruyter & Co, Berlin, New York, 1971.

[51] Schwabl, Franz: *Quantenmechanik*, 4. Auflage, Springer-Verlag Berlin Heidelberg New York, 1993.

[52] Schwartz, Heiko: *Einschrittverfahren der Ordnung 2 für Optimale Steuerungsprobleme*, Universität Bayreuth, 2007.

[53] Schwarz, Hans-Rudolf und Köckler, Norbert: *Numerische Mathematik*, B.G. Teubner Verlag/GWV Fachverlage GmbH, Wiesbaden, 5. erweiterte Auflage, 2005.

[54] Sethi, Suresh P. und Thompson, Gerald L.: *Optimal Control Theory, Applications to Management Science and Economics*, Second Edition, Kluwer Academic Publischers, Boston/Dordrecht/London 2000. Fachverlage GmbH, Wiesbaden, 5. erweiterte Auflage, 2005.

[55] Stoer, Josef: *Numerische Mathematik 1*, Springer-Verlag Berlin Heidelberg, 9. Auflage, 2005.

[56] Stoer, Josef und Bulirsch, Roland: *Numerische Mathematik 2*, Springer-Verlag Berlin Heidelberg, 5. Auflage, 2005.

[57] Strehmel, Karl und Weiner, Rüdiger: *Linear-implizite Runge-Kutta-Methoden und ihre Anwendung*, B.G. Teubner Verlagsgesellschaft, Leipzig 1992.

[58] Strehmel, Karl und Weiner, Rüdiger: *Numerik gewöhnlicher Differentialgleichungen*, B.G. Teubner Stuttgart 1995.

[59] Walter, Wolfgang: *Gewöhnliche Differentialgleichungen*, Sechste, überarbeitete und erweiterte Auflage, Springer-Verlag Berlin Heidelberg New York, 1996.

[60] Werner, Dirk: *Funktionalanalysis*, 5. erweiterte Auflage, Springer-Verlag Berlin Heidelberg 2000.

[61] Werner, Helmut und Arndt, Herbert: *Gewöhnliche Differentialgleichungen: Eine Einführung in Theorie und Praxis*, Springer-Verlag Berlin Heidelberg New York, 1986.

[62] Werner, Jochen: *Numerische Mathematik 1*, Friedr. Vieweg & Sohn Verlagsgesellschaft mbH, Braunschweig/Wiesbaden 1992.

[63] Yan, Hui: *Dynamics and real-time Optimal Control of Satellite Attitude and Satellite Formation Systems*, Texas A&M University, August 2006.

[64] Zillober, Christian: *Software manual for SCPIP 3.0*, http://www.mathematik.uni-wuerzburg.de/∼zillober/pubs/manual30.pdf.

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 31. Januar 2007 .....

Eggert Rose