Technische Universität Dresden
Fakultät Elektrotechnik und Informationstechnik
Institut für Regelungs- und Steuerungstheorie

STUDIENARBEIT

Thema: Untersuchungen zur Trajektorienplanung durch Lösung eines Randwertproblems

Vorgelegt von: Oliver Schnabel

geboren am: 26.03.1988 in Meerane

Betreuer: Dipl.-Ing. C. Knoll

verantwortlicher Hochschullehrer: Prof. Dr.-Ing. habil. Dipl.-Math. K. Röbenack

Tag der Einreichung: 1. Februar 2013

Technische Universität Dresden
Fakultät Elektrotechnik und Informationstechnik
Institut für Regelungs- und Steuerungstheorie
Prof. Dr.-Ing. habil. Dipl.-Math. K. Röbenack

RST
Institut für
Regelungs- und Steuerungstheorie

Aufgabenstellung zur Studienarbeit
für Herrn cand. ing. Oliver Schnabel

Untersuchungen zur Trajektorienplanung durch Lösung eines Randwertproblems

Am Institut für Regelungs- und Steuerungstheorie wird an einer Toolbox für die Behandlung nichtlinearer regelungstechnischer Probleme gearbeitet. Aufgrund der funktionellen Eignung und der freien Verfügbarkeit des Interpreters und der relevanten Bibliotheken (Sympy und Numpy) wird diese Toolbox in Python implementiert. Neben Entwurfsverfahren für Regler und Beobachter soll dieses Programmpaket auch Routinen zur Trajektorienplanung (z. B. zwischen Ruhelagen) beinhalten.
Ein möglicher Zugang hierfür ist die Betrachtung als Randwertproblem mit freien Parametern. Als numerischer Lösungsalgorithmus kommt unter anderem ein sogenanntes Kollokationsverfahren in Frage. Dabei werden die Systemgleichungen zeitlich diskretisiert um dann das resultierende nichlineare Gleichungssystem zu lösen. Bisher verfügbare Implementierungen zielen auf allgemeine Randwertprobleme ab und können folglich nicht die spezielle Struktur berücksichtigen, die sich bei der Trajektorienplanung ergibt (z. B. durch eine partielle Linearisierung).
Ziel der Studienarbeit ist es, ein angepasstes Lösungsverfahren für Randwertprobleme zu entwickeln, dass sich in die oben genannte Toolbox einbettet und zur Trajektorienplanung für eine möglichst große Systemklasse genutzt werden kann.

Im Einzelnen sind dabei folgende Teilaufgaben zu bearbeiten:

- Einarbeitung in mathematische Grundlagen (Randwertprobleme, Lösungsverfahren)
- Einarbeitung in Python und den bisher existierenden relevante Quellcode
- Untersuchungen zur symbolischen Vorbehandlung der Systemgleichungen (z. B. Abspaltung des linearen Teils, Diskretisierungs- und Parametrisierungsvarianten)
- Untersuchungen zur (nummerischen) Lösung der nichtlinearen Gleichungen (Lösbarkeit, Eindeutigkeit, Konvergenzeigenschaften)
- Überprüfung des Verfahrens anhand geeigneter Beispiele
- Dokumentation der Ergebnisse

Betreuer:	Dipl.-Ing. C. Knoll
Verantwortlicher Hochschullehrer:	Prof. Dr.-Ing. habil. Dipl.-Math. K. Röbenack
Bearbeitungszeitraum:	01. 11. 2012 – 01. 02. 2013

Dresden, den 23. November 2012 Prof. Dr.-Ing. habil. Dipl.-Math. K. Röbenack

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die von mir am heutigen Tage eingereichte Studienarbeit zum Thema

„Untersuchungen zur Trajektorienplanung durch Lösung eines Randwertproblems“

vollkommen selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Dresden, den _____

Unterschrift _____
(Oliver Schnabel)

Inhaltsverzeichnis

Verzeichnis der wichtigsten verwendeten Symbole

Notation	Bedeutung
$\dot{x}$	erste Ableitung nach der Zeit
$\ddot{x}$	zweite Ableitung nach der Zeit
$x^{(p)}$	p-te Ableitung nach der Zeit
$F'(x)$	partielle Ableitung von F nach x
${}^*\mathbf{x}^d$	Sollwert für $\mathbf{x}$ zum Startzeitpunkt ${}^*t$
${}^\dagger\mathbf{x}^d$	Sollwert für $\mathbf{x}$ zum Endzeitpunkt ${}^\dagger t$
$\tilde{x}$	transformierte Größe x
x^*	gesuchte Lösung

Symbole	Bedeutung	Bezug
$\mathbf{A}$	Teilmatrix von $\mathbf{M}$	Abschnitt 3.2
$\mathbf{a}$	Teilvektor von $\mathbf{c}$	Abschnitt 3.2
α	Randwerte am linken Rand	Abschnitt 3
$\mathbf{B}$	Teilmatrix von $\mathbf{M}$	Abschnitt 3.2
$\mathbf{b}$	Teilvektor von $\mathbf{c}$	Abschnitt 3.2
β	Randwerte am rechten Rand	Abschnitt 3
$\mathbf{c}$	Koeffizienten des Spline-Ansatzes	Abschnitt 3.2
c_i	reelle Koeffizienten	Abschnitt 3
γ	Randbedingungen des Eingangs	Abschnitt 3.5
δ	Parameter für die Anzahl der Kollokationspunkte	Abschnitt 3.3 und 5
ε	Fehler des Endzustandes	Abschnitt 5
ε_{tol}	Fehlertoleranz	Abschnitt 5
$\mathbf{f}$	Systemdynamik	gesamte Arbeit
Φ	Funktion des flachen Ausgangs	Abschnitt 2.3
$\mathbf{G}$	Gleichungssystem	gesamte Arbeit
$\mathbf{h}$	Ausgangsabbildung	gesamte Arbeit
h	Abstand der Splineknotenpunkte	Abschnitt 3.2
$H_i(t)$	Fehlerfunktion von $\dot{x}_i$	Abschnitt 3 und 6
i, j, k	Indexvariablen mit wechselnder Bedeutung	
κ	Parameter für die Vervielfachung der Splineabschnitte	Abschnitt 5
λ	Anzahl der Splineabschnitte	Abschnitt 3
m	Eingangsdimension	gesamte Arbeit
$\mathbf{M}$	Matrix des Gleichungssystem	Abschnitt 3.2
N	Anzahl der Diskretisierungszeitpunkte	Abschnitt 3
N_1, N_2	Matrixdimensionen	Abschnitt 3.2 und 4
n	Zustandsdimension	gesamte Arbeit

p	Ausgangsdimension	Abschnitt 2
$\mathbf{r}$	Komponente des Gleichungssystems	Abschnitt 3.2
t	Zeit	gesamte Arbeit
${}^{*}t$	Startzeitpunkt der Trajektorie	Abschnitt 2 und 3.2
${}^{\dagger}t$	Endzeitpunkt der Trajektorie	Abschnitt 2 und 3.2
T	Manöverdauer	gesamte Arbeit
τ_i	Splineknotenounkte	Abschnitt 3
$\mathbf{u}$	Eingang	gesamte Arbeit
ν	Grad der Randbedingungen	gesamte Arbeit
v	Zählvariable für die Iterationschritte	Abschnitt 5
w	polynomialer Grad	Abschnitt 3
$\mathbf{x}$	Zustandsvektor	gesamte Arbeit
ξ	Anzahl der freien Parameter des Eingangs	gesamte Arbeit
$\mathbf{y}$	Ausgang	Abschnitt 2
Ψ	Hilfsfunktion	Abschnitt 2.3

Bemerkung zur Notation: In der Arbeit wird entgegen der im deutschen Sprachraum üblichen Schreibweise der Punkt („ . “) als Dezimaltrennzeichen verwendet. Hintergrund ist die Konsistenz zu automatisch generierten Achsenbeschriftungen in Abbildungen und die Eindeutigkeit bei der Angabe von Vektoren und Mengen, bei denen das Komma das Trennzeichen zwischen Komponenten bzw. Elementen darstellt.

1 Einleitung

Das Planen und Entwerfen von Trajektorien stellt eine wichtige Aufgabe bei der Steuerung technischer Prozesse dar. Bei der Überführung von Arbeitspunkten oder bei An- und Abfahrvorgängen in der Industrie ist der vorherige Entwurf und die Analyse dieser Vorgänge unabdingbar um das Einhalten von technischen Grenzen und die Durchführbarkeit zu gewährleisten.
In der Regelungstechnik hat sich das Prinzip der Zwei-Freiheitsgrade-Struktur als vorteilhaft erwiesen, dass die unabhängige Realisierung von Stör- und Führungsverhalten erlaubt. Dieses Entwurfsverfahren ermöglicht es den Stellgrößenverlauf vorzugeben und dennoch eine Regelung gegen Störeinflüsse vorzusehen.
In der vorliegenden Arbeit wird das Problem der Trajektorienplanung auf eine mehrdimensionales Randwertproblem mit freien Parametern zurück geführt. Im Allgemeinen kann dieses Problem nicht analytisch gelöst werden. Es wird daher auf das Verfahren der Kollokation zurück gegriffen um eine numerische Näherungslösung zu erzielen. Die Arbeit beschäftigt sich explizit mit den Besonderheiten, die aus der Modellierung des Systems als Differentialgleichungen resultieren und sich für den Entwurf des Lösungsverfahrens ausnutzen lassen. Zudem werden verschiedene Verfahren zur Lösung von nichtlinearen Gleichungssystem, die im Zusammenhang mit dem Kollokationsverfahren auftreten, diskutiert. Abschließend wird die Qualität des vorgestellten Verfahrens an ausgewählten Beispielen, vornehmlich aus der Mechanik, überprüft.
Bisherige Lösungsansätzen und verfügbare Implementierungen bauen auf die proprietäre Softwareumgebung von `Matlab` auf. Als Anregung der hier dargelegten Untersuchungen diente die Veröffentlichung von Knut Graichen [Gra06] in der eine Trajektorie für das Aufschwingen eines Doppelpendels mittels der `Matalb`-Funktion `bvp4c` ermittelt wurde. Die dort aufgezeigte Modellierung der Eingangsansatzfunktion als parameterbehafteter Polynomansatz, war Ausgangspunkt für die eigenen Überlegungen und Verallgemeinerung des Lösungsansatzes.
Im Zuge der Arbeit ist eine Umsetzung des Verfahrens als `Python`-Bibliothek entstanden, die auf die frei verfügbaren Bibliotheken `numpy`, `sympy` [Sym13] und `scipy` [JOP$^+$] sowie `algopy` [WL11] aufbaut.

2 Trajektorienplanung in der Regelungstechnik

Betrachtet werden sollen nichtlineare dynamische Systeme mit konzentrierten Parameter in Zustandsraumdarstellung und den dazugehörigen gewöhnlichen Differentialgleichungen in $\mathbf{x} = (x_1, ..., x_s)$. Die Größen $x_i, i = 1, ..., s$ werden dabei als Systemgrößen [1] bezeichnet und beinhalten alle zur Modellbildung benötigten Variablen. Die aus den Modellierung gewonnenen Zusammenhänge stellen Differentialgleichungen dar, die sich in der Zustandsraumdarstellung beschreiben lassen

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \mathbf{x}(t_0) = \mathbf{x}_0 \tag{2.1a}$$
$$\mathbf{y}(t) = \mathbf{h}(\mathbf{x}(t), \mathbf{u}(t)) \ . \tag{2.1b}$$

Die Größe $\mathbf{x}$ ist dabei der n-dimensionale Zustandsvektor, $\mathbf{u}$ der m-dimensionale Stell- bzw. Eingangsvektor und $\mathbf{y}$ der p-dimensionale Ausgang. Des Weiteren sind $\mathbf{f}$ und $\mathbf{h}$ zwei glatte Abbildungen mit

$$\begin{aligned} \Omega &\subseteq \mathbb{R}^n \\ \mathbf{f} : \Omega \times \mathbb{R}^m &\to \mathbb{R}^n \\ \mathbf{h} : \Omega \times \mathbb{R}^m &\to \mathbb{R}^p \ . \end{aligned} \tag{2.2}$$

Das System von (2.1a) bildet ein System von n expliziten gewöhnlichen Differentialgleichungen [2]

$$\begin{aligned} \dot{x}_1 &= f_1((x_1, x_2, ..., x_{\sigma_1}), (u_1, ..., u_{\nu_1})) \\ \dot{x}_2 &= f_2((x_1, x_2, ..., x_{\sigma_2}), (u_1, ..., u_{\nu_2})) \\ &\vdots \\ \dot{x}_n &= f_n((x_1, x_2, ..., x_{\sigma_n}), (u_1, ..., u_{\nu_n})) \ . \end{aligned} \tag{2.3}$$

2.1 Stellgrößen

Für die gestellte Aufgabe der gezielten Beeinflussung des Trajektorienverlaufs ist eine Wahl von Eingangsgrößen unabdingbar. Dabei muss die Wahl nicht endgültig bei der Analyse des Systems getroffen werden, vielmehr kann es sinnvoll sein, andere Größen als die Stellgrößen als Eingangsgrößen zu interpretieren. Dies ist z. B. bei unterlagerten Regelungen oder Eingangstransformationen der Fall [Ada09, Rud09]. Die Komponenten des Eingangs $\mathbf{u} = (u_1, u_2, \ldots, u_m)$ sollen die folgenden Voraussetzungen erfüllen [Rud09, Fli90].

[1] Unter Systemgrößen werden alle Größen verstanden, die der Beschreibung des Systems dienen. Das sind sowohl Zustandsgrößen als auch Eingangsgrößen.

[2] Im folgenden wird für $x_i(t)$ und $u_i(t)$ x_i bzw. u_i geschrieben.

- Die Zeitverläufe für u_i sind von einander unabhängig und können frei gewählt werden.
- Jede Systemgröße genügt einer Differentialgleichung

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

Gehen wir davon aus, dass die Elemente des Eingangs $\mathbf{u} = (u_1, u_2, \ldots, u_m)$ in den Zustandsgrößen $\tilde{\mathbf{x}} = (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_s)$ enthalten ist

$$\dot{\tilde{\mathbf{x}}}(t) = \mathbf{f}(\tilde{\mathbf{x}}(t)) \ , \tag{2.4}$$

dann können wir $\tilde{\mathbf{x}}$ so aufteilen, dass $\tilde{\mathbf{x}} = (\mathbf{x}, \mathbf{u})$ mit $\mathbf{x} = (x_1, \ldots, x_p)$ und $p = s - m$. Außerdem müssen die gewählten Komponenten von $\mathbf{u}$ differentiell unabhängig sein, d. h. es existiert keine Differentialgleichung aus (2.4), in den Komponenten von $\mathbf{u}$ allein. Damit ergibt sich die Zustandsraumdarstellung aus (2.1a) mit den dazugehörigen Differentialgleichungen (2.3).
Die Eingangsgrößen werden nun so gewählt, dass die Gleichung (2.4) zu einem Differentialgleichungssystem für $\mathbf{x}$ wird, das für vorgegebene zeitliche Verläufe von $\mathbf{u}$ bestimmt ist. Die Dimension m des Eingangsvektors ist also ein Maß für die Unterbestimmtheit des Systems. Ist $m = 0$ und das System bestimmt, so heißt es autonom. In der Mathematik werden Randwertaufgaben fast ausschließlich für diese Art von Systemen betrachtet, auch das hier zugrunde liegende Problem kann als Randwertaufgabe für autonome Systeme mit freien Parametern verstanden werden.

2.2 Trajektorienplanung

Bei dynamischen Systemen stellt eine Zustandstrajektorie die Verbindung zwischen zwei Zuständen eines Systems dar. Um einen bestimmten gewünschten Verlauf vorzugeben wird eine bestimmte Eingangstrajektorie benötigt. Dabei wird das System vom Startzustand $\mathbf{x}({}^*t)$ in der Zeit $T = {}^\dagger t - {}^* t$ in den Endzustand $\mathbf{x}({}^\dagger t)$ überführt. Bei technischen Systemen handelt es sich meist um Überführungen zwischen Arbeitspunkten oder Anfahrvorgänge.
Hierbei müssen sowohl Randbedingungen der Aufgabenstellung erfüllt sein als auch Beschränkungen für die Zustandsgrößen sowie für die Stellgrößen beachtet werden, wobei die Beschränkungen meist durch technische Möglichkeiten vorgegeben sind. Der Verlauf der Trajektorie wird durch die Dynamik des Systems bestimmt und kann im Allgemeinen nicht beliebig vorgegeben werden.
Das Ziel der Trajektorienplanung ist es einen zeitlichen Verlauf $t \mapsto \mathbf{u}^*(t)$ zu bestimmen für den das System in Zeit T ausgehend vom Zustand ${}^*\mathbf{x}^d$ den Zustand ${}^\dagger\mathbf{x}^d$ erreicht. Da die Menge der Lösungen für $\mathbf{u}^*$ unendlich groß ist, soll das Ziel

der Arbeit sein, (irgend)eine taugliche d. h. technisch und physikalisch sinnvolle Lösung zu finden.
Sind die Verläufe für $t \mapsto \mathbf{u}(t)$ und der Zustand des Systems zum Zeitpunkt ${}^*t$ bekannt, stellt der zeitliche Verlauf für die Systemgrößen eine Anfangswertaufgabe dar. Ein analytisches Lösen dieses Problems ist meist nicht möglich [SW95] und es muss auf numerische Lösungsverfahren zurück gegriffen werden, die die Vorwärtssimulation des Systems erlauben. Für verfügbare mathematische Software ist dies jedoch eine Standardaufgabe. In der vorliegenden Arbeit wurde auf entsprechende Routinen aus der Python Bibliothek `Scipy` [JOP$^+$] zurück gegriffen, um die aus der Lösung der Randwertaufgabe ermittelten Stellgrößenverläufe zu überprüfen.

2.3 Flachheit

Eine nützliche Systemeigenschaft im Bezug auf die Planung von Trajektorien ist die sogenannte differentielle Flachheit [FLR95, MMR03]. Diese Merkmal trifft jedoch nur auf wenige Systeme zu oder muss durch geschickte Wahl des Ausgangs herbei geführt werden. Ein System (2.1) heißt differentiell [3] flach, wenn es einen fiktiven Ausgang

$$\mathbf{y} = \Phi(\mathbf{x}, u_1, \dots, u_1^{(\zeta_1)}, \dots, u_m, \dots, u_m^{(\zeta_m)}) \tag{2.5}$$

gibt, mit

$$\dim \mathbf{y} = \dim \mathbf{u} \tag{2.6}$$

der die Bedingungen

$$\mathbf{x} = \psi_1(\mathbf{y}, \dot{\mathbf{y}}, \dots, \mathbf{y}^{(\varsigma)}) \tag{2.7a}$$
$$\mathbf{u} = \psi_2(\mathbf{y}, \dot{\mathbf{y}}, \dots, \mathbf{y}^{(\varsigma+1)}) \tag{2.7b}$$

erfüllt. Sind diese Bedingungen zumindest lokal gültig, heißt der Ausgang (2.5) flacher Ausgang [RR97]. Zudem erfüllen die Komponenten von $\mathbf{y}$ wegen (2.7a) und (2.7b) keine Differentialgleichung der Form

$$\varphi(\mathbf{y}, \dot{\mathbf{y}}, \dots, \mathbf{y}^{(\sigma)}) = 0 \ . \tag{2.8}$$

Das bedeutet für die Trajektorienplanung solcher Systeme, dass eine genügend oft stetig differenzierbare Verlauf für den flachen Ausgang $\mathbf{y}$ vorgegeben werden kann [Rot97], der auch die geforderten Randbedingungen enthält. Eine numerische Lösung des Trajektorienproblems ist damit nicht nötig, die Verläufe der Stellgrößen

[3] Der Begriff *differentiell* bedeutet, dass die aus den Komponenten von $\mathbf{y}$ und deren zeitlichen Ableitungen und ohne Integration von Differentialgleichungen, alle Zustände und deren Ableitungen bestimmt werden können.

können aus (2.7b) berechnet werden.
Dennoch kann das in dieser Arbeit vorgestellte Verfahren auch bei diesen System Anwendung finden und von Vorteil sein, da die symbolische Berechnung der Funktionen in (2.6) und (2.7b) mitunter sehr aufwendig ist [Rud00] oder über andere Verfahren berechnet werden muss [Rö05].

2.4 Trajektorienfolgeregelung

Eine weitere Aufgabenstellung die oft aus der Trajektorienplanung folgt, ist das Realisieren einer Folgeregelung bei der versucht wird, das System entlang der vorgegebenen Bahn zu stabilisieren. Eine solche Regelung wird nötig, wenn auf Grund von äußeren Einflüssen z. B. durch Störgrößen oder durch Modellungenauigkeiten, eine reine Steuerung mittels der Stellgrößen nicht ausreicht, um eine befriedigende Folge der gewünschten Trajektorie zu gewährleisten. Für die Klasse der flachen Systeme wie sie im Abschnitt 2.3 diskutiert wurde gibt es hierfür eine Reihe von Ansätzen [Rud03, Rud09] mit deren Hilfe sich stabile Folgeregelungen realisieren lassen.
Mit Hilfe der Zwei-Freiheitsgrade-Struktur [Sch04, Deu12] ist es möglich , auch bei

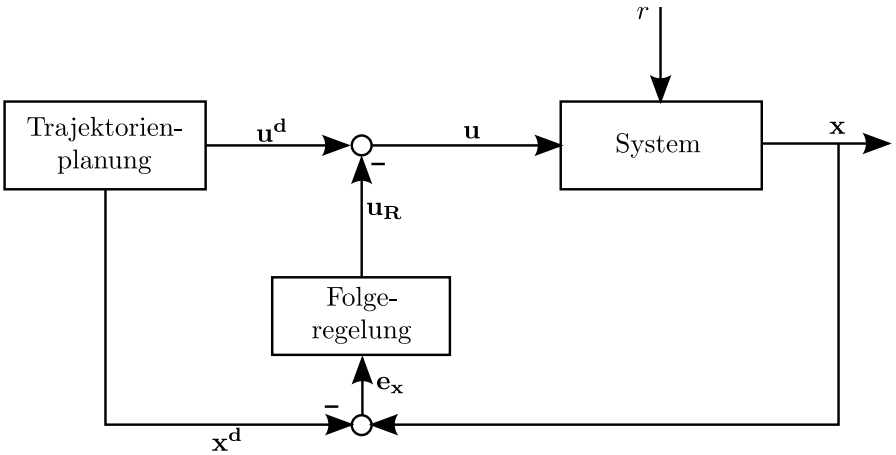


Abbildung 1: Schematische Darstellung einer Trajektorienfolgeregelung.

nicht flachen Systemen, Führungsverhalten unabhängig vom Störverhalten vorzugeben und die Steuerung des Systems durch die Eingangsreferenztrajektorie, um eine Regelung zu erweitern. Durch z. B. eine zeitvarianten Ausgangsrückführung [Fö78, AD10, RP12] lässt sich eine entsprechende Regelung entwerfen.

2.5 Bisherige Verfahren

Eine bekannte Methode um umd das Problem Trajektorienplanung numerischen zu lösen bietet das `Matlab`-Modul `bvp4c` [SK00]. Dabei wird mit Hilfe eines Kollokationsverfahrens, als dreistufigen Lobatto-IIIa-Methode [4], ein Randwertproblem gelöst. Durch Erweiterung des Verfahrens können auch freie Parameter bestimmt werden. Dies kann genutzt werden um die freien Parameter einer Eingangsansatzfunktion zu berechnen [GZ06, KAA12]. Hierbei sind jedoch viele Vorüberlegung nötig und die Lösung des Problems hängt stark vom gewählten Startwert ab. Für viele Probleme ist es außerdem nötig eine analytische Lösung der partiellen Ableitungen des Systems bereit zustellen, da die interne Schätzung dieser ungenügend ist. Zudem ist das Modul an die proprietäre Softwareumgebung von `Matlab` gebunden. Aus der beschränkten Verfügbarkeit und der aufwendigen Implementierung neuer Problemstellung motiviert sich die Aufgabenstellung der Arbeit.

[4] Das Labatto-IIIa-Verfahren kann aus den Runge-Kutta-Verfahren abgeleitet werden, stellt jedoch auch eine Kollokationsverfahren dar [Her04].

3 Numerischen Lösung der Randwetaufgabe

Zunächst sollen autonome Systeme gewöhnlicher Differentialgleichungen betrachtet werden

$$\begin{aligned} \dot{x}_1 &= f_1(x_1(t), ..., x_{\sigma_1(t)}) \\ \dot{x}_2 &= f_2(x_1(t), ..., x_{\sigma_2(t)}) \\ \vdots &= \quad \vdots \\ \dot{x}_s &= f_n(x_1(t), ..., x_{\sigma_s(t)}). \end{aligned} \tag{3.1}$$

Dabei stellt $\dot{x}_i(t)$ die Ableitung der Funktion $x_i(t)$ nach t dar mit $i = 1, ..., n$. Fasst man $\mathbf{x}(t)$ in der Form $\mathbf{x}(t) = (x_1(t), x_2(t), ..., x_n(t))^T$ zusammen, so kann das System in Vektorschreibweise wie folgt dargestellt werden

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) \ . \tag{3.2}$$

Hier bei ist t aus dem Intervall $[{}^*t, {}^\dagger t] := \{t \in \mathbb{R} | {}^*t \leq t \leq {}^\dagger t\}$ und $\mathbf{x}(t) \in \mathbf{\Omega} \subset \mathbb{R}^n$. Die Randbedingungen sind als Dirichlet-Randbedingung gegeben, dass heißt es sind Funktionswerte der Funktionen $x_i(t)$ an den beiden Rändern des Definitionsbereiches $[{}^*t, {}^\dagger t]$, für die Auswertepunkte $t = {}^*t$ und $t = {}^\dagger t$ mit

$${}^*x_i^d = \alpha_i, \quad {}^\dagger x_i^d = \beta_i \tag{3.3}$$

festgelegt [Ise08].

3.1 Kollokationsverfahren

Zur übersichtlicheren Darstellung der Überlegungen wird zunächst eine System (3.1) mit $n = 1$

$$\dot{x}_1 = f_1(t, x_1) \tag{3.4}$$

auf dem Intervall $t = [{}^*t, {}^\dagger t]$ betrachtet. Die Erkenntnisse können später ich gleicher Weise auf mehrdimensionale Probleme angewandt werden. Im Intervall $[{}^*t, {}^\dagger t]$,werden $N + 1$ (Zeit)-Punkte gewählt in denen das System diskretisiert wird. Die Kollokationspunkte t_i für $i = 0, ..., N$ mit $t_0 = {}^*t$, $t_N = {}^\dagger t$ bezeichnen die Punkte, in den die Gleichung (3.4) ausgewertet wird. Eine mögliche Wahl sind äquidistante Punkte mit

$$t_i = \frac{i}{N}({}^\dagger t - {}^*t) \ . \tag{3.5}$$

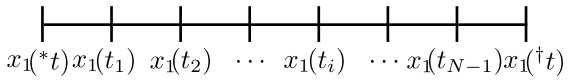


Abbildung 2: Diskretisierung der Auswertezeitpunkte

Die Auswertezeitpunkte sind in Abbildung 2 veranschaulicht. Des weiteren seien Randbedingungen wie in (3.3) gegeben. Gesucht ist die Funktion der Zeit $t \mapsto P_{x_1}(t)$ des Zustandes $x_1(t)$, die die Randbedingungen erfüllt und der Systemdynamik in den Kollokationspunkten genügt

$$P_{x_1}(^*t) = \alpha_1, \quad P_{x_1}(^\dagger t) = \beta_1 \tag{3.6a}$$

$$\frac{d}{dt} P_{x_1}(t_i) = f_1(t_i, P_{x_1}(t_i)), \quad i = 0, \ldots, N\ . \tag{3.6b}$$

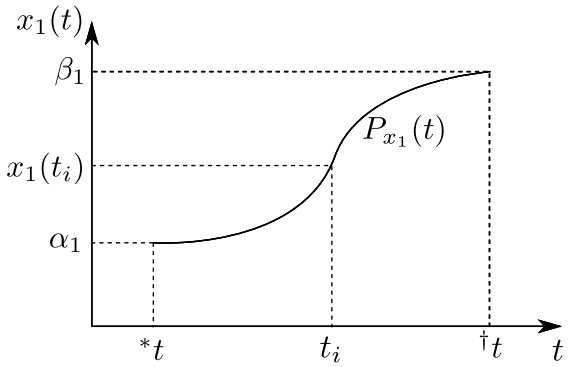


Abbildung 3: Ansatzfunktion $P_{x_1}(t)$

Durch die Forderungen in (3.6) wird der gesamte Verlauf der exakten Lösung approximiert. Als einfache Ansatzfunktion eignen sich Polynome vom Grad w mit den frei wählbaren Koeffizienten $c_k^{x_1}$, $k = 0, \ldots, w$.

$$P_{x_1}(t) = \sum_{k=0}^{w} c_k^{x_1} \left(\frac{t - {}^*t}{{}^\dagger t - {}^*t} \right)^k \tag{3.7}$$

Der Grad w muss nun so gewählt werden, dass genügend Freiheitsgrade $c_k^{x_1}$ zur Verfügung stehen um die Bedingungen (3.6) zu erfüllen. Das Polynom besitzt dabei $w + 1$ Koeffizienten. Gemäß der Gleichung (3.6a) müssen zwei Randbedingungen sowie $N + 1$ Kollokationsbedingungen (3.6b) erfüllt werden daraus folgt, dass P_{x_1} mindestens den Grad $N + 2$ besitzen muss. Hierbei ergibt sich aus den Gleichungen (3.6a), (3.6b) sowie (3.7) ein Gleichungssystem das nach den Koeffizienten $c_k^{x_1}$ aufgelöst werden kann. Das zu lösende System ist gemäß der Eigenschaften von f_1 linear oder nichtlinear.

Wurde eine Lösung für $P_{x_1}(t)$ gefunden, lässt sich die Qualität der Lösung überprüfen in dem man die Gleichung für den Fehler der Differentialgleichung f_1

$$H_{x_1}(t) = \frac{d}{dt}P_{x_1}(t) - f_1(t, P_{x_1}(t)) \tag{3.8}$$

kontinuierlich im gesamten Intervall $[{}^*t, {}^\dagger t]$ auswertet.
Die Fläche unter dem Graphen von $H_{x_1}(t)$ kann als Gesamtfehler der Näherungs-

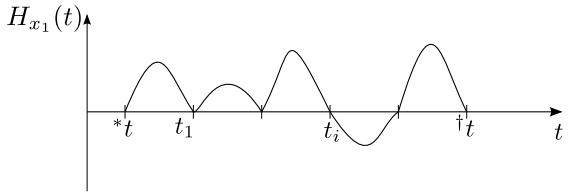


Abbildung 4: Schematischer Verlauf der Fehlerfunktion $H_{x_1}(t)$.

lösung verstanden werden. Für die exakte Lösung ergibt sich für $H_{x_1}(t)$ die Nullfunktion. Im Allgemeinen gilt: Die Lösung $P_{x_1}(t)$ ist umso besser, je kleiner die Fläche zwischen Graph und der Zeitachse ist. Die Differenz in den Kollokationspunkten ist für ein exakt gelöstes Gleichungssystem Null, dies entspricht den Nullstellen der Abbildung 4. Dennoch stellt die gefundene Lösung nur eine Näherung dar und wird außerhalb der Kollokationspunkte die Systemdynamik nicht exakt beschreiben. Um die Qualität der Lösung zu verbessern, kann die Anzahl der Auswertepunkte erhöht werden, allerdings verbunden damit muss auch der Grad der Ansatzfunktion im selben Maße erhöht werden. Die damit verbundenen Probleme sowie eine andere Möglichkeit für Ansatzfunktionen soll im Abschnitt 3.2 diskutiert werden.
Für ein Differentialgleichungssystem wie in (3.1), d. h. für $n \geq 1$, wird für jede Größe x_j eine Ansatzfunktion $P_{x_j}(t)$ definiert. Alle Differentialgleichungen werden in den gleichen Kollokationspunkten ausgewertet, woraus sich folgendes Gleichungssystem ergibt

$$P_{x_j}({}^*t_j) = \alpha_j, \quad P_{x_j}({}^\dagger t_j) = \beta_j \tag{3.9a}$$

$$\frac{d}{dt}P_{x_j}(t_i) = f_j(t_i, P_{x_1}(t_i), P_{x_2}(t_i), \ldots, P_{x_n}(t_i)) \quad j = 1, \ldots, n\ . \tag{3.9b}$$

3.1.1 Das Gleichungssystem

Um das Lösen des Gleichungssystems später zu vereinfachen sollen in den Gleichungen nur die Bedingungen aus (3.6b) vorkommen. Die Forderungen an die Randwerte lassen sich bereits durch geeignete Konstruktion der Ansatzfunktionen sicher stellen. Damit entspricht das Gleichungssystem, für den Fall $n = 1$ der

Fehlerfunktionen $t \mapsto H_{x_1}(t)$ an den Auswertezeitpunkten t_i

$$\begin{aligned}
\frac{d}{dt}P_{x_1}(t_0) - f(t_0, P_{x_1}(t_0)) &:= G_0(c_0^{x_1}, c_1^{x_1}, \ldots, c_w^{x_1}) \stackrel{!}{=} 0 \\
\frac{d}{dt}P_{x_1}(t_1) - f(t_1, P_{x_1}(t_1)) &:= G_1(c_0^{x_1}, c_1^{x_1}, \ldots, c_w^{x_1}) \stackrel{!}{=} 0 \\
&\vdots \\
\frac{d}{dt}P_{x_1}(t_N) - f(t_N, P_{x_1}(t_N)) &:= G_N(c_0^{x_1}, c_1^{x_1}, \ldots, c_w^{x_1}) \stackrel{!}{=} 0.
\end{aligned} \tag{3.10}$$

In gleicher Weise lässt sich dies für den mehrdimensionalen Fall $n \geq 1$ aufstellen, dabei werden die jeweiligen Fehlerfunktionen $H_{x_j}(t)$ mit $j = 1, \ldots, n$ ausgewertet. Das Gleichungssystem besteht somit aus den $N + 1$ Kollokationsbedingungen mit dem Lösungsvektor $\mathbf{c}^* = (c_0^*, c_1^*, \ldots, c_w^*)$. Die Gesuchte $\mathbf{c}^*$ ist genau dann die Lösung des Systems $\mathbf{G} = (G_0, G_1, \ldots, G_N)^T$, wenn gilt:

$$\mathbf{G}(c^*) \stackrel{!}{=} 0 \tag{3.11}$$

Das Lösen des Systems kann daher auf eine Nullstellensuche von $\mathbf{G}$ zurück geführt werden. Mit der gefundenen Lösung $\mathbf{c}^*$ ergibt sich der in Abbildung 4 skizzierte Verlauf für $t \mapsto H_{x_1}(t)$, in dem der Fehler in Kollokationspunkten gerade Null ist. Mögliche Lösungsverfahren für das Problem (3.11) werden im Abschnitt 4 diskutiert.

3.1.2 Anzahl der Kollokationspunkte

Aus den Betrachtung des Abschnitts ist bekannt, dass es sich bei den Ausdrücken im Gleichungssystem um Auswertezeitpunkte der Fehlerfunktionen $t \mapsto \mathbf{H}(t)$ mit $\mathbf{H}(t) = [H_{x_1}(t), H_{x_2}(t), \ldots, H_{x_n}(t)]^T$ handelt. Im Allgemeinen kann gesagt werden, dass die durch das Kollokationsverfahren ermittelte Lösung umso besser ist, je kleiner die Fläche unter den Graphen von $t \mapsto \mathbf{H}(t)$ ist. Da eine analytische Berechnung der Flächen nicht zielführend ist, erfolgt eine Näherung durch die Diskretisierung in den $N + 1$ Kollokationspunkten.
Um die eindeutige Lösbarkeit des Gleichungssystems (3.10) sicherzustellen, müssen genau so viele Gleichungen wie Unbekannte enthalten sein. Zunächst wieder im eindimensionalen Fall $n = 1$

$$N + 1 \stackrel{!}{=} \dim \mathbf{c}^{x_1} \ . \tag{3.12}$$

Damit folgt die Anzahl der Kollokationspunkte aus der Wahl der Ansatzfunktion. Zudem ergeben sich Besonderheiten bei der Wahl des Lösungsverfahrens (siehe Abschnitt 4).

Um das Integral der Funktion $H_{x_1}(t)$ nahe Null zu machen, ist es erstrebenswert eine möglichst große Anzahl an Diskretisierungszeitpunkten zu wählen. Dabei soll jedoch der Grad der Ansatzfunktion nicht im gleichen Maße erhöht werden. Denn dies führt zu einem Gleichungssystem mit viel mehr Gleichungen als Unbekannten.

$$N + 1 \gg \dim \mathbf{c}^{x_1} \tag{3.13}$$

Das Lösen dieses Systems führt nicht mehr zu einer Nullstellensuche sondern zu einem Optimierungsproblem für den Lösungsvektor $\mathbf{c}^*$, der das System bestmöglich löst. Entsprechende Verfahren werden im Abschnitt 4 diskutiert.

3.2 Splines als Ansatzfunktion

Wie bereits im letzten Abschnitt erwähnt, muss um den zeitlichen Verlauf der Zustandsgrößen $x_i(t)$ möglichst exakt wieder zugeben, die Anzahl der Kollokationspunkte groß gewählt werden. Aus der Überlegung das mit jeder weiteren Bedingung ein weiterer freier Parameter nötig ist, resultiert auch eine große Gesamtzahl an Freiheitsgraden . Eine Möglichkeit zur Konstruktion solcher Funktionen wurde bereits in Gleichung (3.7) diskutiert. Die Freiheitsgrade $c_k^{x_i}$ sind frei wählbare Konstanten mit $c_k^{x_i} \in \mathbb{R}$. Um eine hohe Anzahl an Freiheitsgraden zu erzeugen, muss der Grad w des Polynomes sehr groß gewählt werden. Polynome höheren Grades neigen jedoch zu starker Oszillation an den Intervallgrenzen und approximieren den tatsächlichen Verlauf der exakten Lösung sehr schlecht. Dies ist aus der Literatur auch als Runge-Phänomen bekannt [Her01].
Für physikalische Größen ist es sinnvoll, bis zu einem gewissen Grad stetige Differenzierbarkeit zu fordern. In der numerischen Mathematik haben sich hierbei Splines als geeignet erwiesen [Her01, Nur89].
Splines sind stückweise definierte Polynome mit einer globalen Differenzierbarkeit. Die Verbindungspunkte zwischen den polynomialen Abschnitten sind äquidistant und werden als Knoten τ_i bezeichnet

$$\begin{aligned} {}^*t = \tau_0 < \tau_1 < \cdots < \tau_{\lambda-1} < \tau_\lambda &= {}^\dagger t \\ h &= \frac{{}^\dagger t - {}^* t}{\lambda} \\ \tau_{i+1} &= \tau_i + h \quad \text{für } i = 0 \ldots (\lambda - 1) \end{aligned} \tag{3.14}$$

h ... Breite eines Splineabschnittes
λ ... Anzahl der Splineabschnitte .

Für die hier verwendeten Ansatzfunktionen sollen Splines der Polynomordnung $w = 3$, d. h. es werden sogenannte kubische Splines, verwendet. Die einzelnen

polynomialen Abschnitte lassen sich folgendermaßen konstruieren

$$\begin{aligned}&\tilde{P}_i(x) = c_0^i(x-ih)^3 + c_1^i(x-ih)^2 + c_2^i(x-ih) + c_3^i \\ &i = 1 \dots \lambda\end{aligned} \tag{3.15}$$

$$P(x) = \begin{cases} \tilde{P}_1(x), & {}^*t \le x \le h \\ \tilde{P}_2(x), & h < x \le 2h \\ \vdots & \\ \tilde{P}_i(x), & (i-1)h < x \le ih \\ \vdots & \\ \tilde{P}_\lambda(x), & (\lambda-1)h < x \le {}^\dagger t\ . \end{cases} \tag{3.16}$$

An den Knotenpunkten soll neben der Stetigkeit auch eine zweifache stetige Differenzierbarkeit gefordert werden.

Freiheitsgrade

Aus (3.7) ergibt sich, dass jeder Splineabschnitt mit dem Polynomgrad $w = 3$ vier freie Koeffizienten besitzt. Andererseits können in allen Knoten τ_i, $i = 1 \dots (\lambda - 1)$ drei Glattheitsbedingungen aufgestellt werden

$$\tilde{P}_i(ih) = \tilde{P}_{i+1}(ih) \tag{3.17a}$$

$$\frac{d}{dt}\tilde{P}_i(ih) = \frac{d}{dt}\tilde{P}_{i+1}(ih) \qquad i = 1...(\lambda-1) \tag{3.17b}$$

$$\frac{d^2}{dt^2}\tilde{P}_i(ih) = \frac{d^2}{dt^2}\tilde{P}_{i+1}(ih). \tag{3.17c}$$

Damit werden $3(\lambda - 1)$ Gleichungen formuliert und es verbleiben von den 4λ freien Koeffizienten noch $\lambda + 3$. Zusätzlich können Bedingungen an den Ränder, die sich aus den Randbedingungen des Differentialgleichungssystems ergeben, gestellt werden

$$\begin{aligned}&\frac{d^j}{dt^j}\tilde{P}_1(0) = \alpha_j \\ &\frac{d^j}{dt^j}\tilde{P}_\lambda(\lambda h) = \beta_j \\ &\text{für } j = 0...\nu\ .\end{aligned} \tag{3.18}$$

Der Grad ν der Randbedingungen richtet sich dabei nach der Struktur des Differentialgleichungssystems. Für Splines ohne Vorgaben an den Rändern ist $\nu = -1$.

Für die Zustandsgrößen sollen aber zumindest Randwerte der Funktionswerte an den Intervallgrenzen vorgeben sein also $\nu = 0$. Zu dem können sich durch Ausnutzung der Struktureigenschaften des Systems weitere Randbedingungen für die Ableitungen zum Start- und Endzeitpunkt ergeben $\nu \geq 1$ (siehe Abschnitt 3.4).
Um die Randbedingungen sowie Anforderung an die Stetigkeit und stetige Differenzierbarkeit schon bei der Konstruktion der Splines zu berücksichtigen, soll eine Gleichungssystem mit den nötigen Forderungen aufgestellt werden. Dadurch kann ein Teil der freien Parameter der Splineabschnitte durch diese Bedingungen ausgedrückt werden.
Aus den Gleichungen (3.17) und (3.18) kann ein unterbestimmtes Gleichungssystem in Matrixschreibweise formuliert werden. Der Vektor $\mathbf{c}$ beinhaltet dabei die Koeffizienten aus (3.15).

$$\underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 & -h^3 & h^2 & -h & 1 & \dots & 0 \\ 0 & 0 & 1 & 0 & 3h^2 & -2h & 1 & 0 & \dots & 0 \\ & \vdots & & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{bmatrix} c_0^1 \\ c_1^1 \\ c_2^1 \\ c_3^1 \\ c_0^2 \\ c_1^2 \\ c_2^2 \\ c_3^2 \\ \vdots \\ c_3^\lambda \end{bmatrix}}_{\mathbf{c}} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ \beta_0 \end{bmatrix}}_{\mathbf{r}} \tag{3.19}$$

Die Matrix $\mathbf{M}$ der Dimension $N_1 \times N_2$, $N_2 > N_1$ lässt sich in zwei Teilsysteme $\mathbf{A} \in \mathbb{R}^{N_1 \times (N_2 - N_1)}$ und $\mathbf{B} \in \mathbb{R}^{N_1 \times N_1}$ zerlegen. Zu den beiden Matrizen gehören die Vektoren $\mathbf{a}$ und $\mathbf{b}$ mit den jeweiligen Koeffizienten aus $\mathbf{c}$. Mit dieser Aufteilung lässt sich das Gleichungssystem nach $\mathbf{b}$ auflösen und es verbleiben die Parameter in $\mathbf{a}$ als Freiheitsgrade für den Spline.

$$\begin{aligned} \mathbf{A}\,\mathbf{a} + \mathbf{B}\,\mathbf{b} &= \mathbf{r} \\ \mathbf{b} &= \mathbf{B}^{-1}(\mathbf{r} - \mathbf{A}\,\mathbf{a}) \end{aligned} \tag{3.20}$$

Die Koeffizienten in (3.15) können nun mit den in (3.20) erhaltenen Beziehungen ersetzt werden.
Eine wichtige Entscheidung im Hinblick auf das numerische Lösen des Gleichungssystems stellt die Wahl der Koeffizienten in $\mathbf{a}$ dar. Die Dimension Von $\mathbf{M}$ setzt sich aus den

$$N_2 = 4\lambda \quad \lambda \dots \text{Anzahl der Splineabschnitte} \tag{3.21}$$

Freiheitsgraden aus der Gleichung (3.15), den Stetigkeitsbedingungen aus (3.17) und der Anzahl an Randbedingungen aus (3.18) zusammen

$$N_1 = 3(\lambda - 1) + 2(\nu + 1) \quad \nu \ldots \text{höchster Grad der Randbedingungen} . \tag{3.22}$$

Aus den Gleichungen (3.21) und (3.22) ergibt sich die Dimension von a mit

$$\dim \mathbf{a} = N_2 - N_1 = 4\lambda - 3(\lambda - 1) - 2(\nu + 1) = \lambda + 3 - 2(\nu + 1) \tag{3.23}$$

Dabei sind für die Zustandsgrößen mindestens die Funktionswerte an den beiden Rändern vorgegeben, d. h. $\nu = 0$, so dass sich maximal $\lambda+1$ frei wählbare Parameter ergeben. Sind zusätzlich noch Forderungen an die erste Ableitung, also $\nu = 1$, an den Grenzen gestellt verbleiben noch $\lambda - 1$ usw. Für λ Freiheitsgrade können die Koeffizienten vor einer Potenz von x gewählt werden und es kann je nach Randbedingungen ein weiterer Faktor einer anderen Potenz des ersten Splineabschnitts hinzukommen bzw. werden die Parameter der letzten Abschnitte nicht verwendet. Eine wichtige Eigenschaft für die Betrachtungen in (3.20) ist die Regularität der Matrix **B**. Aus der letzten Zeile der Matrix **M**, in der die Bedingung für den Funktionswert an der oberen Intervallgrenze gefordert wird, ist ersichtlich das **B** nur dann regulär ist, wenn **b** den Koeffizienten c_3^λ enthält. Ähnliche Überlegungen lassen sich für Randbedingungen der Ableitungen treffen aus denen sich ergibt, dass c_2^λ sowie c_1^λ ebenfalls nicht Teilmenge des Vektors **a** sein dürfen. Aus diesen Überlegungen und der heuristischen Forderung das **a** möglichst gleichartige Koeffizienten enthalten soll, werden für die ersten λ Komponenten in **a** die Faktoren c_0^i, $i = 1 \ldots \lambda$ gewählt. Für den Fall $\nu = 0$ soll der verbleibende Faktor c_3^1 sein. Damit ergibt sich für $\nu = 0$

$$\mathbf{a} = [c_0^1, c_0^1, c_0^2, \ldots, c_0^{\lambda-1}, c_0^\lambda, c_3^1]^T \tag{3.24}$$

für $\nu = 1$

$$\mathbf{a} = [c_0^1, c_0^1, c_0^2, \ldots, c_0^{\lambda-2}, c_0^{\lambda-1}]^T \tag{3.25}$$

und für $\nu = 2$

$$\mathbf{a} = [c_0^1, c_0^1, c_0^2, \ldots, c_0^{\lambda-4}, c_0^{\lambda-3}]^T . \tag{3.26}$$

Für die Ansatzfunktionen der Stellgrößenverläufe werden für gewöhnlich keine Randbedingungen gefordert, Stetigkeitsanforderungen sind jedoch durchaus sinnvoll. Analog zu den vorangegangenen Überlegungen sollen die Splines für den Eingang mit den freien Koeffizienten

$$\mathbf{a} = [c_0^1, c_0^1, c_0^2, \ldots, c_0^{\lambda-1}, c_0^\lambda, c_3^1, c_3^2, c_3^3]^T \tag{3.27}$$

konstruiert werden.

3.3 Konstruktion des Kollokationsverfahrens mit Splines

Durch das Verwenden der Splines als Ansatzfunktionen lassen sich durch geschickte Wahl der Kollokationspunkte die Gleichungen (3.10) vereinfachen. Werden die Splines in Knotenpunkten τ_i mit $i = 0, \ldots, \lambda$ ausgewertet, verbleibt als Funktionswert der Splines nur der Absolutwert c_3^j, $j = 1, \ldots, \lambda$. Einzige Ausnahme bildet der Rand bei $t = {}^*t$ für den $\tilde{P}_1(0)$ ausgewertet werden muss. Selbes gilt für die Ableitungen $\frac{d}{dt}\tilde{P}_j(t)$, für die für das Auswerten an den Splineknotenpunkten der freie Parameter c_2^j verbleibt. In selber Weise trifft dies auch für die zweiten Ableitungen zu. Die Gleichungen aus dem Gleichungssystem $\mathbf{G}$ aus (3.10) haben für eindimensionale Probleme damit die folgende Form

$$c_2^k - f(\tau_k, c_3^k) = 0 \quad k = 1, \ldots, \lambda \; . \tag{3.28}$$

Zu beachten ist hierbei, dass es sich bei den Koeffizienten um Linearkombinationen aus den Komponenten von $\mathbf{a}$ aus Abschnitt 3.2 handelt.
Ähnliche Überlegungen gelten auch, wenn wesentlich mehr Kollokationspunkte, wie in Abschnitt 3.1.2 beschrieben, konstruiert werden sollen als Splineabschnitte vorhanden sind. Neben Knotenpunkte sollen Splinefunktionen des weiteren noch in $(\delta-1)$-Punkten zwischen den Knoten ausgewertet werden. Die Auswertezeitpunkte t_i ergeben sich aus

$$t_i = \frac{i}{\delta\lambda} T \quad i = 0, \ldots, \delta\lambda \; . \tag{3.29}$$

In Abbildung 5 sind Kollokationspunkte für den Fall $\delta = 2$ schematisch dargestellt.

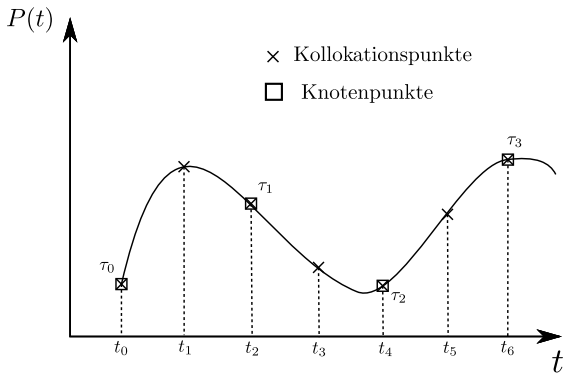


Abbildung 5: Skizze für Kollokationspunkte mit $\delta = 2$.

3.4 Ausnutzung der Systemstruktur

Häufig treten in physikalischen Modellen Differentialgleichungen der Form

$$\dot{x}_i = x_{i+1} \tag{3.30}$$

auf. Für diese Gleichungen ist es nicht notwendig eine Lösung durch das Kollokationsverfahrens zu ermitteln. Ohne Beeinträchtigung des Lösungsverfahrens reicht es aus, eine Ansatzfunktion für x_i zu definieren und x_{i+1} durch Differentiation zu gewinnen.

$$x_{i+1} = \frac{d}{dt} P_{x_i}(t) \tag{3.31}$$

Für die Randbedingungen von x_i gilt dann zusätzlich

$$\frac{d}{dt} P_{x_i}({}^{*}t) = \alpha_{x_{i+1}}, \qquad \frac{d}{dt} P_{x_i}({}^{\dagger}t) = \beta_{x_{i+1}} \ . \tag{3.32}$$

Ähnliche Vereinfachungen können getroffen werden, wenn Beziehungen der Form

$$\dot{x}_i = u_j \tag{3.33}$$

auftreten. Auch das Beispiel im nächsten Abschnitt illustriert diesen Sachverhalt.

3.5 Ansatzfunktion des Eingangs

Um das Differentialgleichungssystem (3.1) mit seinen Randbedingungen (3.3) lösen zu können, muss das System über Freiheitsgrade verfügen. Ersichtlich wird dies durch die Überlegung das ein Anfangswertproblem zunächst immer eindeutig lösbar ist. Jede weitere Forderung an das System muss durch einen Freiheitsgrad gewährleistet werden.
Für ein n-dimensionales Differentialgleichungssystem mit $2n$ Randbedingungen bedeutet dies, dass mindestens n freie Parameter benötigt werden, damit die Randwertaufgabe lösbar ist. Die Freiheitsgrade werden über die parameterabhängige Ansatzfunktion des Eingangs zur Verfügung gestellt. Da wir nur (irgend)eine mögliche Lösung der Randwertaufgabe suchen, ist es sinnvoll, die Anzahl der freien Parameter größer als n zu wählen

$$\xi \geq n \quad \xi \ldots \text{Anzahl der freien Parameter des Eingangs} \tag{3.34}$$

um so schneller zu einer geeigneten Lösung zu kommen.
Als Ansatzfunktionen für den Eingang werden wieder Splines genutzt. Die Anzahl

der Abschnitte ergibt sich aus der Forderung der Freiheitsgrade wie im Abschnitt 3.2 erläutert

$$\lambda_u = \xi - 3 + 2(\nu + 1) \quad \nu \ldots \text{höchster Grad der Randbedingungen} . \tag{3.35}$$

Zudem können weitere Forderungen an die Ansatzfunktion des Eingangs gestellt werden, wie z. B. die Funktionswerte zum Start- und Endzeitpunkt mit

$$u_j({}^*t) = {}^*\gamma_j \qquad u_j({}^\dagger t) = {}^\dagger\gamma_j \qquad j = 1, \ldots, m \qquad m = \dim \mathbf{u} \tag{3.36}$$

wobei für Überführungen von Ruhelagen z. B.

$${}^*\gamma_j = {}^\dagger\gamma_j = 0 \tag{3.37}$$

gelten kann.
Solche Forderungen entsprechen $\nu = 0$ in der Formel (3.35). Weitere Bedingungen an die Eingangsansatzfunktion können durch die Berücksichtigung der Integratorstrukturen, wie im Abschnitt 3.4 dargelegt, resultieren.

Beispiel

Um die Überlegungen aus Abschnitt 3.4 und 3.5 zu verdeutlichen soll ein System der Struktur

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u \\ \dot{x}_3 &= f_3(t, \mathbf{x}, u) \\ \dot{x}_4 &= f_4(t, \mathbf{x}, u) \end{aligned} \tag{3.38}$$

mit den Randbedingungen

$$x_i({}^*t) = \alpha_i, \quad x_i({}^\dagger t) = \beta_i, \quad i = 1, \ldots, 4 \tag{3.39}$$

betrachtet werden. Solche Systemstrukturen entstehen oft durch eine partielle Linearisierung mit der durch eine nichtlinearen Ausgangsrückführung ein linearer Zusammenhang zwischen einer neuen Einganggröße und dem Ausgang geschaffen wird [Ada09]. Löst man die Struktur für x_1 und x_2 auf, sieht man, dass nur eine Ansatzfunktion für x_1 benötigt wird. Die Größen x_1 und u werden als erste und zweite Ableitung der Ansatzfunktion definiert

$$\begin{aligned} x_1 &= P_{x_1}(t) \\ x_2 &= \frac{d}{dt} P_{x_1}(t) \\ u &= \frac{d^2}{dt^2} P_{x_1}(t) \ . \end{aligned} \tag{3.40}$$

Damit sind die ersten beiden Gleichungen des Systems (3.38) per Definition erfüllt und müssen nicht durch das Kollokationsverfahren gelöst werden. Der Spline für x_1 muss nun die vier Randbedingungen ($\nu = 1$) für die verbleibenden Zustandsgrößen erfüllen und mindestens

$$\xi \geq 4 - 2 = 2 \tag{3.41}$$

freie Parameter enthalten um die verbleibende Randwertaufgabe für x_3 und x_4 lösen zu können. Aus Abschnitt 3.2 ist bekannt, dass ein Spline mit wenigstens

$$\lambda_u = 2 - 3 + 2(1 + 1) = 3 \tag{3.42}$$

polynomialen Abschnitten diese Bedingungen erfüllt.

4 Numerische Lösung des Gleichungssystems

Das zuvor aufgestellte Gleichungssystem aus Abschnitt 3.1.1 soll nun mit Hilfe eines numerischen Verfahren gelöst werden. Entsprechend des betrachteten Systems aus (2.1) ist das Gleichungssystem linear bzw. nichtlinear. In diesem Abschnitt sollen speziell nichtlineare Systeme behandelt werden. Im Zuge der Arbeit wurden drei mögliche Lösungsverfahren untersucht.

4.1 Newton Verfahren

Ausgangspunkt für das Verfahren ist ein algebraisches Gleichungssystem der Form:

$$\mathbf{F}(\mathbf{x}) = \mathbf{0} \tag{4.1}$$

Mit dem Verfahren können Nullstellen der Funktion $\mathbf{F} : \mathbb{R}^{N_2} \to \mathbb{R}^{N_2}$ bestimmt werden. Zunächst muss eine Schätzung $\mathbf{x}_0$ der möglichen Nullstelle vorgegeben werden. Ausgehend von diesem Startwert wird die Funktion in dem Punkt linear approximiert. Der Schnittpunkt der Linearisierung mit der x-Achse dient wiederum als Ausgangspunkt für den nächsten Iterationsschritt. Anschaulich lässt sich dies im eindimensionalen Fall darstellen, dies ist in der Abbildung 6 veranschaulicht.

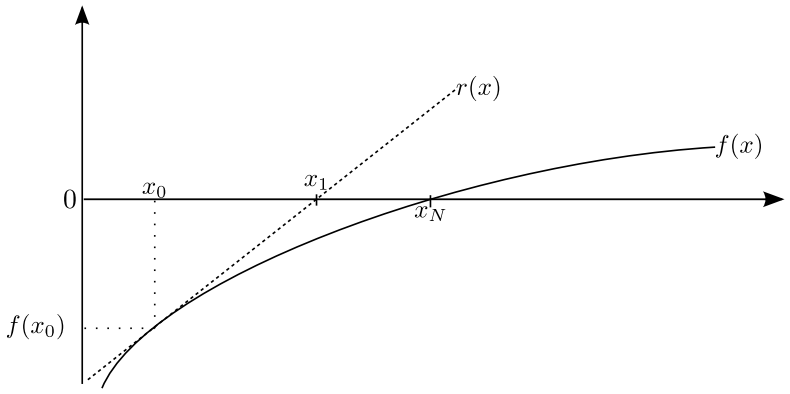


Abbildung 6: Skizze des Newton-Verfahrens im eindimensionalen Fall.

Mit der Ableitung $f'(x)$ an der Stelle x_0 ergibt sich die Tangente zu

$$r(x) = f(x_0) + f'(x_0)(x - x_0) \ . \tag{4.2}$$

Die Nullstelle der Geraden lässt sich durch

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{4.3}$$

angeben. Daraus kann eine iterative Rechenvorschrift abgeleitet werden, die auch für mehrdimensionale Funktionen $\mathbf{F}(x)$ gültig ist [TS92]

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{F}'(\mathbf{x}_k)^{-1}\mathbf{F}(\mathbf{x}_k) \qquad k = 0, 1, \ldots \qquad . \tag{4.4}$$

Die Ableitungen $\mathbf{F}'(\mathbf{x})$ mit $\mathbf{F}' : \mathbb{R}^{N_2} \to \mathbb{R}^{N_1}$ stellen hier bei die Jacobimatrix ($N_1 = N_2$)

$$\mathbf{F}'(\mathbf{x}) := \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \frac{\partial f_1}{\partial x_2}(\mathbf{x}) & \ldots & \frac{\partial f_1}{\partial x_{N_2}}(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{N_1}}{\partial x_1}(\mathbf{x}) & \frac{\partial f_{N_1}}{\partial x_2}(\mathbf{x}) & \ldots & \frac{\partial f_{N_1}}{\partial x_{N_2}}(\mathbf{x}) \end{pmatrix} \tag{4.5}$$

dar. Bei der Rechenvorschrift muss sichergestellt werden, dass diese Matrix invertierbar ist. Dies muss vor allem beim Erstellen des Gleichungssystems beachtet werden, d.h. es sollten gleich viele Gleichungen wie Unbekannte vorhanden sein. Durch das Newton-Verfahren wird demnach eine Lösung gesucht, für die der Fehler (3.8) in den Kollokationspunkten Null wird.
Die Konstruktion des Kollokationsverfahrens ist durch die Bedingung eines quadratischen Gleichungssystems sehr umständlich, da die Vorteile der Konstruktion mit Splines wie in Abschnitt 3.3 beschrieben nicht beachtet werden können.

4.2 Gauß-Newton-Verfahren

Betrachtet werden soll nun ein überbestimmtes Gleichungssystem mit mehr Variablen als Gleichungen und der abbildenden Funktion $F : \mathbb{R}^{N_1} \to \mathbb{R}^{N_2}$ mit $N_1 > N_2$. Solche Systeme entstehen durch die Erhöhung der Kollokationspunktanzahl wie in Abschnitt 3.1.2 beschrieben. Für diese Systeme soll die Lösung des nichtlinearen Ausgleichsproblems

$$\mathbf{g}(\mathbf{x}) := \|\mathbf{F}(\mathbf{x})\|_2^2 \to \min \tag{4.6}$$

gesucht werden. Zunächst wird die Funktion $\mathbf{F}(\mathbf{x})$, ähnlich wie beim Newton-Verfahren, um einen diskreten Punkt $\mathbf{x}_k$ linearisiert.

$$\mathbf{r}(\mathbf{x}) = \mathbf{F}(\mathbf{x}_k) + \mathbf{F}'(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) \tag{4.7}$$

Damit ergibt sich das Problem zu

$$\|\mathbf{F}(\mathbf{x}_k) + \mathbf{F}'(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k)\|_2^2 \to \min \quad . \tag{4.8}$$

Der Punkt $\mathbf{x}^*$ ist genau dann ein Minimum von $\mathbf{g}(\mathbf{x})$, wenn

$$\mathbf{g}'(\mathbf{x}^*) = \mathbf{0} \qquad \text{und} \qquad \mathbf{g}''(\mathbf{x}^*) \quad \text{positiv definit} \tag{4.9}$$

genügt [Deu08].
Die Ableitung von $\mathbf{g}(\mathbf{x})$ ergibt sich mit der Produktregel von

$$\mathbf{F}(\mathbf{x})^2 = \mathbf{F}(\mathbf{x})^T\mathbf{F}(\mathbf{x}) \tag{4.10}$$

zu

$$\mathbf{g}'(\mathbf{x}) = 2\mathbf{F}'(\mathbf{x})^T\mathbf{F}(\mathbf{x}) \ . \tag{4.11}$$

Setzt man nun die Taylorreihenentwicklung aus (4.7) in $\mathbf{g}'(\mathbf{x})$ ein, erhält man:

$$\begin{aligned} 2\mathbf{r}'(\mathbf{x})^T\mathbf{r}(\mathbf{x}) &= 0 \\ (\mathbf{F}'(\mathbf{x}_k) + \mathbf{F}''(\mathbf{x}_k)(\mathbf{x}-\mathbf{x}_k))^T(\mathbf{F}(\mathbf{x}_k) + \mathbf{F}'(\mathbf{x}_k)(\mathbf{x}-\mathbf{x}_k)) &= 0 \end{aligned} \tag{4.12}$$

Auf eine Berücksichtigung des Tensors $\mathbf{F}''(\mathbf{x}_k)$ kann verzichtet werden [DR08] und man erhält damit

$$\mathbf{F}'(\mathbf{x}_k)^T(\mathbf{F}(\mathbf{x}_k) + \mathbf{F}'(\mathbf{x}_k)(\mathbf{x}-\mathbf{x}_k)) = 0 \ . \tag{4.13}$$

Dies Vernachlässigung des Tensors führt zu einer Verschlechterung der Konvergenzordnung, vermindert jedoch den Gesamtrechenaufwand des Verfahrens erheblich. Durch das Lösen der Gleichung (4.13) nach $\mathbf{x}$ kann nun der nächste Iterationspunkt $\mathbf{x}_{k+1}$ ermittelt und eine Iterationsvorschrift abgeleitet werden

$$\mathbf{x}_{k+1} = \mathbf{x}_k - [\mathbf{F}'(\mathbf{x}_k)^T\mathbf{F}'(\mathbf{x}_k)]^{-1}\mathbf{F}'(\mathbf{x}_k)\mathbf{F}(\mathbf{x}_k) \ \ . \tag{4.14}$$

Wenn die Matrix $\mathbf{F}(\mathbf{x})$ Vollrang hat d.h. $\text{rang}(\mathbf{F}(\mathbf{x})) \geq \min(N_1, N_2)$, dann ist $[\mathbf{F}'(\mathbf{x}_k)^T\mathbf{F}'(\mathbf{x}_k)]$ immer regulär. Die lässt sich mit der Rangungleichung von Sylvester zeigen [SS94], welche lautet:

$$\begin{aligned} &\text{sei } \mathbf{A} \in \mathbb{R}^{N_1 \times N_2} \text{ und } \mathbf{B} \in \mathbb{R}^{N_2 \times R} \\ &\text{dann gilt:} \\ &\text{rang}(\mathbf{A}) + \text{rang}(\mathbf{B}) - N_2 \leq \text{rang}(\mathbf{AB}) \leq \min(\text{rang}(\mathbf{A}), \text{rang}(\mathbf{B})) \end{aligned} \tag{4.15}$$

Für die Jacobimatrizen von $\mathbf{F}(\mathbf{x})$ ergibt sich

$$\begin{aligned} &\mathbf{F}'(\mathbf{x})^{\mathbf{T}} \in \mathbb{R}^{N_2 \times N_1} \text{ und } \mathbf{F}'(\mathbf{x}) \in \mathbb{R}^{N_1 \times N_2} \text{ mit } N_1 \geq N_2 \\ &[\mathbf{F}'(\mathbf{x}_k)^T\mathbf{F}'(\mathbf{x}_k)] \in \mathbb{R}^{N_2 \times N_2} \\ &\text{rang}(\mathbf{F}'(\mathbf{x})^{\mathbf{T}}) = \text{rang}(\mathbf{F}'(\mathbf{x})) = N_2 \\ &\text{rang}(\mathbf{F}'(\mathbf{x})^{\mathbf{T}}) + \text{rang}(\mathbf{F}'(\mathbf{x})) - N_2 \leq \text{rang}(\mathbf{F}'(\mathbf{x})^{\mathbf{T}}\mathbf{F}'(\mathbf{x})) \leq \min(\text{rang}(\mathbf{F}'(\mathbf{x})^{\mathbf{T}}, \mathbf{F}'(\mathbf{x}))) \\ &N_2 + N_2 - N_2 \leq \text{rang}(\mathbf{F}'(\mathbf{x})\mathbf{F}'(\mathbf{x})^{\mathbf{T}}) \leq N_2 \\ &\text{rang}(\mathbf{F}'(\mathbf{x})\mathbf{F}'(\mathbf{x})^{\mathbf{T}}) = N_2 \ \ . \end{aligned} \tag{4.16}$$

4.3 Levenberg-Marquardt-Verfahren

Das Verfahren stellt eine Erweiterung zum Gauß-Newton-Verfahren dar. Dabei wird die Minimierungsaufgabe

$$\|\mathbf{F}'(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k) + \mathbf{F}(\mathbf{x}_k)\|_2^2 + \mu^2\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2^2 \to min \tag{4.17}$$

gelöst. Die reelle Zahl μ ist ein frei zu wählender Parameter, der zur Dämpfung der Differenz zwischen $\mathbf{x}_{k+1}$ und $\mathbf{x}_k$ dient. Damit wird vor allem das Erzeugen von zu großen Korrekturen vermieden, wie es oft beim Gauß-Newton-Verfahren der Fall ist und dort zu einem eventuellen Nichterreichen des lokalen Minimums führt. Das Gauß-Newton-Verfahren stellt mit einer verschwindenden Dämpfung $\mu = 0$ ein Spezialfall des Levenberg-Marquardt-Verfahrens dar. Durch ein Vorgehen wie in (4.2) kann eine Iterationsvorschrift in der folgenden Form angegeben werden:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - [\mathbf{F}'(\mathbf{x}_k)^T\mathbf{F}'(\mathbf{x}_k) + \mu^2\mathbf{I}]^{-1}\mathbf{F}'(\mathbf{x}_k)\mathbf{F}(\mathbf{x}_k) \tag{4.18}$$

Die Konvergenz kann nun mittels des Parameters μ beeinflusst werden. Nachteil ist, dass um die Konvergenz zu gewährleisten μ groß genug gewählt werden muss, zugleich führt dies jedoch auch zu einer sehr kleinen Korrektur. Damit besitzt das Levenberg-Marquardt-Verfahren eine niedrigere Konvergenzordnung als das Gauß-Newton-Verfahren, nähert sich jedoch in jedem Schritt der gesuchten Lösung an.

Steuerung des Parameters μ

Zunächst soll ein Merkmal definiert werden nachdem sich die Wahl von μ richtet. Hierzu wird die Änderung des tatsächlichen Residuums

$$\begin{aligned} &\Delta R(\mathbf{x}_k, \mathbf{s}_k) = \|\mathbf{F}(\mathbf{x}_k)\|_2^2 - \|\mathbf{F}(\mathbf{x}_k + \mathbf{s}_k)\|_2^2 \\ &\text{mit} \quad \mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k \end{aligned} \tag{4.19}$$

und die Änderung des Residuums der linearisierten Näherung

$$\Delta\tilde{R}(\mathbf{x}_k, \mathbf{s}_k) = \|\mathbf{F}(\mathbf{x}_k)\|_2^2 - \|\mathbf{F}(\mathbf{x}_k) + \mathbf{F}'(\mathbf{x}_k)\mathbf{x}_k\|_2^2 \tag{4.20}$$

verglichen [DR08]. Als Steuerungskriterium wird der Quotient

$$\rho_= \frac{\Delta R(x_k, s_k)}{\Delta\tilde{R}(x_k, s_k)} \tag{4.21}$$

eingeführt.
Aus (4.17) folgt, dass $\Delta\tilde{R}(\mathbf{x}_k, \mathbf{s}_k) \geq 0$ ist. Für eine sinnvolle Korrektur muss außerdem $\Delta R(\mathbf{x}_k, \mathbf{s}_k) \geq 0$ sein. Damit ist ρ ebenfalls größer 0 zudem ergibt sich für $\mu \to \infty$, $\rho \to 1$. ρ sollte dementsprechend zwischen 0 und 1 liegen. Zur Steuerung werden zwei neue Grenzen b_0 und b_1 eingeführt mit $0 < b_0 < b_1 < 1$ und die in [DR08] eingeführten Kriterien mit den Werten $b_0 = 0.2$, $b_1 = 0.8$:

- $\rho \leq b_0$: $\mathbf{s}_k$ wird neu berechnet, μ verdoppelt.
- $b_0 < \rho < b_1$: $\mathbf{s}_k$ wird für den nächsten Schritt verwendet, im nächstem Schritt wird μ beibehalten.
- $\rho \geq b_1$: $\mathbf{s}_k$ wird akzeptiert, μ wird beim der nächsten Iteration halbiert.

Für das später vorgestellte Verfahren hat sich das Lösungsverfahren Levenberg-Marquardt als geeignet erwiesen. Zum einen können Gleichungssysteme mit beliebig vielen Kollokationsbedingungen konstruiert werden, zum anderen werden die Probleme die im Zusammenhang mit der zu großen Korrektur des Gauß-Newton-Verfahrens entstehen vermieden werden. Die Implementierung erfolgte mit den hier angegebenen Kriterien und einer maximalen Iterationszahl für x_k mit $k = 20$ und einer Abbruchbedingungen falls die Norm $\|\mathbf{F}(\mathbf{x}_k)\|_2^2$ eine Toleranzgrenze unterschreitet oder keine signifikante Verbesserung $\|\mathbf{F}(\mathbf{x}_{k-1})\|_2^2 - \|\mathbf{F}(\mathbf{x}_k)\|_2^2$ statt findet. Für die genaue Umsetzung sei auf den Quelltext der Klasse `solver` verwiesen.

5 Das Verfahren zur Trajektorienberechnung

In den vorangegangenen Abschnitten wurden die Grundlagen diskutiert um eine numerische Berechnung von Trajektorien umzusetzen.
Das Abarbeiten der einzelnen Schritte und die im nächsten Abschnitt beschriebene iterative Erhöhung der Anzahl der Splineabschnitte und die damit verbunde Anzahl an Kollokationspunkte führt zu einem Verfahren, mit dem sich rechnergestützt Trajektorien von dynamischen Systemen der Form (2.1) numerisch berechnen lassen. Hierzu wurde das Verfahren als eine `Python` Bibliothek implementiert, die eine Wiederverwendbarkeit gewährleistet und die Umsetzung von Beispielen erleichtert.

5.1 Interative Erhöhung der Anzahl der Splineabschnitte

Eine wichtige Rolle, insbesondere bei Ansatzfunkionen mit vielen Splineabschnitten, spielt die Startschätzung der Splinekoeffizienten für die Lösung des Gleichungssystems. Nur durch eine konvergierende Lösung des numerischen Verfahrens kann ein sinnvolles Ergebnis für die Trajektorienverläufe erreicht werden. Da eine intuitive Schätzung für die Parameter kaum möglich ist, wird zunächst eine einfache Schätzung für eine Ansatzfunktion mit wenig freien Parametern vorgenommen. In der Arbeit wurden zunächst

$$\lambda_x^0 = 5 \quad \lambda_x ...\text{Anzahl der Splineabschnitte der Zustandsgrößen} \tag{5.1}$$

als Startwert für die Anzahl der Splineabschnitte der Zustandsgrößen gewählt. Die Ansatzfunktionen für den Eingang wurden nach den Kriterien aus Abschnitt 3.5 ausgewählt, für sie wird keine iterative Erhöhung vorgenommen. Für die freien Parameter des Eingangs wird zunächst die gleiche Startschätzung wie für die Zustandsgrößen verwendet, in jedem weiteren Iterationsschritt werden die Ergebnisse des letzten Schrittes als Startwert genutzt. Des weiteren wurde für die Anzahl der Kollokationspunkte die Überlegung aus Abschnitt 3.1.2 zu Grunde gelegt, hier bei wird als Verfahrensparameter $\delta = 2$ gewählt [5]. Als erste Startschätzung für den gesamten Lösungsvektor $\mathbf{c}$ des Gleichungssysteme $\mathbf{G}$ wurde

$$\mathbf{c}^0 = [0.1, 0.1, \ldots, 0.1]^T \tag{5.2}$$

gewählt. Es hat sich gezeigt, dass wenn die Startwerte in einem gewissen Intervall liegen, die Ergebnisse der numerischen Verfahren stets zu einer Lösung konvergieren. Die mit den Startschätzungen resultierenden Splineverläufe sind schematisch

[5] Eine Anpassung der verschiedene für das Verfahren relevanten Parameter entsprechend der gestellten Aufgabe ist durchaus sinnvoll. Die hier angegeben Werte wurden meist nur heuristisch ermittelt und sollen als Richtwert dienen

in Abbildung 7 dargestellt. Diese Ansätze erfüllen die Randbedingungen, genügen aber in keinem Punkt der Systemdynamik. Durch das Aufstellen der Kollokati-

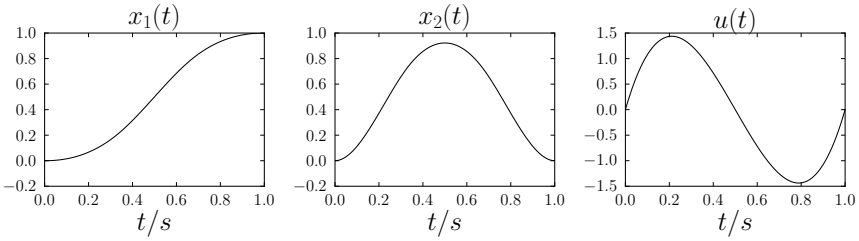


Abbildung 7: Schematischer Verlauf der Ansatzfunktionen mit der ersten Startschätzung $\mathbf{c}^0$ der Splinekoeffizienten.

onsbedingungen und Lösen des resultierenden Gleichungssystems, ergibt sich eine erste Lösung der freien Parameter der Splines. Mit den ermittelten Koeffizienten der Eingangstrajektorie kann die Anfangswertaufgabe mit dem Startwert ${}^*\mathbf{x}^d$ gelöst werden. Die resultierenden Verläufe stellen bereits eine erste, im Allgemeinen unbefriedigende, Näherungslösung dar. Im nächsten Schritt wird nun die Anzahl der Splineabschnitte λ_x erhöht

$$\begin{aligned} \lambda_x^{v+1} = \kappa\lambda_x^v \quad & \kappa\text{...Faktor für die folge Abschnittszahl} \\ & v\text{...Iterationsschritt .} \end{aligned} \tag{5.3}$$

Der Faktor κ stellt einen weiteren Verfahrensparameter dar, in der Arbeit wird $\kappa = 2$ vorgeschlagen. Die ermittelten Splineverläufe des letzten Iterationsschritts werden genutzt um die Startschätzung im aktuellen Schritt anzugeben. Hierbei wird der Verlauf der neuen Ansatzfunktion dem des bereits ermittelten angenähert. Erreicht werden kann dies durch ein Abtasten der bekannten Funktion. Dabei wird der Spline in genau so vielen Punkten diskretisiert, dass sich ein vollständig bestimmtes Gleichungssystem für die freien Parameter der neuen Ansatzfunktion ergibt

$$\begin{aligned} P_{x_i}^v(t_s) = P_{x_i}^{v+1}(t_s), \quad & t_s = \frac{T}{\omega_i + 1} s \\ & s = 1, ..., \omega_i \\ & \omega_i\text{...Anzahl der freien Parameter von } P_{x_i}^{v+1} . \end{aligned} \tag{5.4}$$

Durch die Lösung des daraus resultierenden quadratischen linearen Gleichungssystems ergeben sich gute Startschätzungen für den nächsten Iterationsschritt. Dies führt zu einer schnelleren Lösung des Kollokationsgleichungssystems und reduziert somit auch die Gesamtrechenzeit des Verfahrens. Als Lösungsverfahren für

das Gleichungssystem der Kollokoationsgleichungen hat sich der in der Arbeit der Levenberg-Marquardt-Algorithmus bewährt. Wie in Abschnitt 3.1.2 beschrieben, hängt die Anzahl der Kollokationspunkte mit der Anzahl der Splineabschnitte zusammen, eine Erhöhung der Abschnittsanzahl führt folglich auch zu mehr Kollokationspunkten. Nach jeder neu berechneten Lösung wir die Anfangswertaufgabe mit dem ermittelten Eingangsverlauf $\mathbf{u}^*$ gelöst. Das Ergebnis dieser Simulation ist der Endzustand des Systems $\mathbf{x}^\dagger$ zum Zeitpunkt ${}^\dagger t$. Die Iteration wird solange fortgesetzt bis eine vorher festgelegte maximale Iterationszahl v_{max} erreicht ist oder der Fehler zum Endzeitpunkt

$$\varepsilon = \max(|{}^\dagger x_1^d - x_1^\dagger|, |{}^\dagger x_2^d - x_2^\dagger|, \ldots, |{}^\dagger x_n^d - x_n^\dagger|) \tag{5.5}$$

einer Schranke

$$\varepsilon \leq \varepsilon_{tol} \tag{5.6}$$

genügt.
Der Ablauf des Programms ist in Abbildung 8 veranschaulicht.

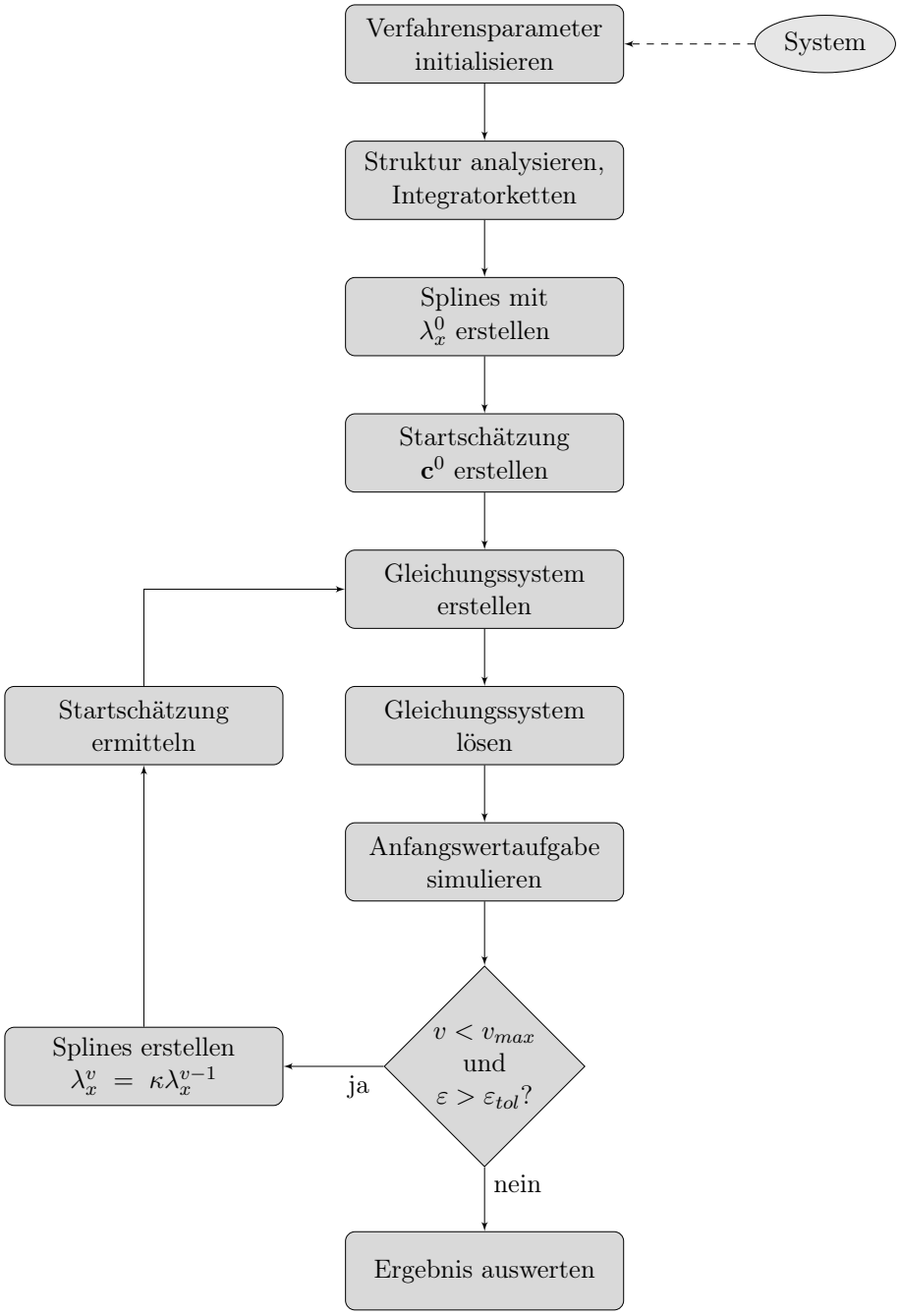


Abbildung 8: Programmablaufplan

5.2 Umsetzung

Im Zusammenhang mit der vorliegenden Arbeit wurde das Verfahren mit einer `Python`-Umgebung entwickelt und getestet. Hierbei wurden einzelne Klassen für die Splineansatzfunktionen und die numerischen Lösungsverfahren erstellt. Der iterative Vorgang ist in einer Klasse `Trajectory` abgelegt. Hierbei wurde Wert auf eine einfache Implementierung von neuen Problemen gelegt. Es genügt die Funktion $\mathbf{f}(\mathbf{x}, \mathbf{u})$ anzulegen und den Start-und Endzustand sowie die Dimension des Systems und des Eingangs anzugeben. Eine Auswahl an Standardparametern wurde getroffen, die sich jedoch entsprechend der Aufgabenstellung anpassen lassen. Die Bezeichnung der Parameter ist weitestgehend mit der vorliegenden Arbeit konsistent. Die Funktion `iteration()` startet dabei die Berechnung und endet wenn die Fehlergrenze ε_{tol} oder die maximale Iterationsanzahl v_{max} erreicht wurde. Mit dem Aufruf `plot()` werden automatisiert, Grafiken der simulierten Zustandsverläufe, Stellgrößenverläufe sowie die Fehlerfunktionen $H_{x_i}(t)$ erzeugt. Nach der Berechnung stehen alle relevanten Daten im Trajectory-Objekt zur Verfügung.
Durch symbolische Konstruktion der Kollokationsgleichungen mit der Bibliothek `sympy` kommt es auf Grund des großen Arbeitsspeicherbedarf der Ausdrücke, zu einer praktischen Begrenzung der Anzahl der Iterationsschritten und Komplexität der zu lösenden Probleme. Mit 8GB Arbeitsspeicher liegt die Grenze bei etwa 500 Kollokationsbedingungen mit den angegebenen Verfahrensparameter. Diese Anzahl reicht jedoch aus um eine große Vielfalt von Problemen ausreichend genau zu lösen. Bei den im nächsten Abschnitt vorgestellten Beispielen lag die Rechenzeit für eine Trajektorienbestimmung im Bereich von 5 - 30 Minuten.
In Abbildung 9 ist der Quelltext zum Beispiel aus Abschnitt 6.1.3 angegeben.

```
from sympy import cos, sin, pi
from trajectory import Trajectory

l=0.5
g=9.81

def f(x,u):
    x1,x2,x3,x4 = x
    u=u[0]
    ff = [  x2,
            u,
            x4,
           (1/l)*(g*sin(x3)+u*cos(x3))]
    return ff

xa=[ 0.0, 0.0, pi,  0.0 ]

xb=[ 0.0, 0.0, 0.0, 0.0 ]
```

```
T=Trajectory(f,xa,xb,n=4,m=1)

T.a=0.0
T.b=2.0
T.lambda_x=5
T.lambda_u=5
T.delta = 2
T.kappa = 3
T.v_max  = 4
T.find_integ = True
T.algo = 'leven'
T.gamma = [0,0]
T.epsilon_tol = 1e-2

T.iteration()
T.plot()
```

Abbildung 9: Quelltextbeispiel zum inversen Pendel aus Abschnitt 6.1.3.

5.3 Ermittlung der Jacobimatrizen

Für die verschiedenen Lösungsverfahren die im Abschnitt 4 beschrieben wurden, werden unabhängig vom Verfahren die Jocobimatrizen des Gleichungssystem $\mathbf{G}(\mathbf{c})$ benötigt.

$$\mathbf{G}'(\mathbf{c}) := \begin{pmatrix} \frac{\partial G_0}{\partial c_0}(\mathbf{c}) & \frac{\partial G_0}{\partial c_1}(\mathbf{c}) & \dots & \frac{\partial G_0}{\partial c_n}(\mathbf{c}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial G_m}{\partial c_0}(\mathbf{c}) & \frac{\partial G_m}{\partial c_1}(\mathbf{c}) & \dots & \frac{\partial G_m}{\partial c_n}(\mathbf{c}) \end{pmatrix} \tag{5.7}$$

Auf Grund der möglichen Nichtlinearität der Gleichungen in $\mathbf{G}$ und der großen Anzahl an Unbekannten ist ein symbolisches Berechnen der Ableitungen sehr zeitaufwendig und führt zu Problemen bei der Umsetzung des Verfahrens. In der vorgelegten Arbeit wurde auf die Methode des Algorithmischen Differenzierens zurück gegriffen um die partiellen Ableitungen zu berechnen. Die Details des Verfahrens gehen über den Umfang der Arbeit hinaus, daher sei auf die Literatur [GW08] verwiesen.
Um das Verfahren in der Arbeit zu nutzen wurden die zur Verfügung stehenden Funktionen der `Python`-Bibliothek `algopy` [WL11] verwendet.

6 Beispiele

In diesem Abschnitt werden verschiedene Beispielsysteme aus der Mechanik diskutiert. Auf das Herleiten der Modellgleichungen wurde verzichtet und stattdessen auf die entsprechenden Fachliteratur verwiesen.
Die dargestellten Verläufe zeigen die Ergebnisse der Lösung der Anfangswertaufgabe mit den ermittelten Stellgrößenverläufen. Zudem wird der zeitliche Verlauf der relevanten Fehlerfunktionen $t \mapsto H_i(t)$

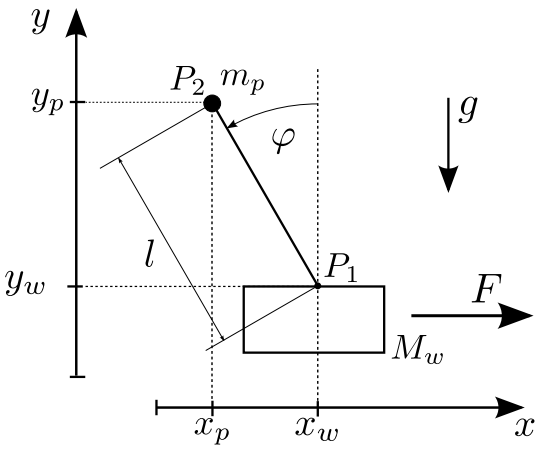
$$H_{x_i}(t) = \frac{d}{dt}P_{x_i}(t) - f_i(\mathbf{u}^*(t), P_{x_1}(t), P_{x_2}(t), \ldots, P_{x_n}(t)) \quad i = 1, \ldots, n$$
$$n \ldots \text{Anzahl der Zustandsgrößen} \tag{6.1}$$

angegeben [6]. Von Interesse sind dabei nur die Fehlerverläufe für die die Differentialgleichung mittels Kollokation gelöst wurde.
Um die Systemgrößenverläufe besser deuten zu können werden zudem Simulationen der Bewegungsabläufe gezeigt.

[6] Auf die Angabe von Einheiten wurde bei den Fehlerfunktionen verzichtet.

6.1 Das inverse Pendel

Ein bekanntes Beispiel aus der Literatur [CH95, CPJ02] ist das inverse Pendulum. Dabei greift an einem Wagen der Masse M_w eine Kraft F an. Außerdem ist der Wagen über einen masselosen Stab mit einer Pendelmasse m_p verbunden. Die Masse des Pendels ist im Punkt P_2 konzentriert, die des Wagens in P_1.



$m_p = 0.1\text{kg}$	...	Masse des Pendelkopfs
$l = 0.5\text{m}$	...	Länge des Pendels
$M_w = 1.0\text{kg}$	...	Masse des Wagens
$g = 9.81\text{m/s}^2$	...	Erdbeschleunigung
P_1	...	Schwerpunkt des Wagens
P_2	...	Schwerpunkt des Pendels
$y_p,\ x_p$	...	Koordinaten des Pendelkopfs
$y_w,\ x_w$	...	Koordinaten des Wagens
φ	...	Auslenkung des Pendels
F	...	Stellkraft

Abbildung 10: Schematische Darstellung des inversen Pendel.

Der Zustandsvektor des Systems lässt sich mithilfe der Wagenposition $x_w(t)$ und der Pendelauslenkung $\varphi(t)$ sowie deren Ableitungen angeben. Mit dem Lagrange-Formalismus lässt sich für das Modell eine Zustandsraumdarstellung mit $\mathbf{x} = [x_w, \dot{x}_w, \varphi, \dot{\varphi}]^T$ und der Stellgröße $u = F$ angeben

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{m_p \sin x_3(-l x_4^2 + g\cos x_3)}{M_w + m_p \sin^2 x_3} + \frac{1}{M_w + m_p \sin^2 x_3} u \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= \frac{\sin x_3(-m_p l x_4^2 \cos x_3 + g(M_w + m_p))}{M_w l + m_p l \sin^2 x_3} + \frac{\cos x_3}{M_w l + m_p l \sin^2 x_3} u \ .
\end{aligned} \tag{6.2}$$

6.1.1 Trajektorie für das Versetzen des Wagens

Eine mögliche geforderte Trajektorie stellt das Versetzen des Wagens in x-Richtung dar. Dabei soll der Wagen und das Pendel am Anfang und am Ende der Bewegung in Ruhe und das Pendel senkrecht nach oben ausgerichtet sein ($\varphi = 0$). Die Stellzeit

soll $T = 1\text{s}$ betragen.

$$ {}^{*}\mathbf{x}^{d} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \xrightarrow{\quad T \quad} {}^{\dagger}\mathbf{x}^{d} = \begin{bmatrix} 0.5 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{6.3} $$

Aus den Betrachtungen der vorherigen Kapiteln kann man erkennen das für das Kollokationsverfahren drei Ansatzfunktionen notwendig sind. Zwei für die Systemgrößen x_1 und x_3 sowie ein Spline für den Verlauf der Stellgröße. Die Größen x_2 und x_4 ergeben sich direkt als erste Ableitung von x_1 bzw. x_3. Als weitere Bedingung soll u zum Anfang um zum Ende stetig in die Ruhelage übergehen, d.h.

$$ u({}^{*}t) = u({}^{\dagger}t) = 0 \quad . \tag{6.4} $$

Für den Eingang wurde ein Spline mit drei Abschnitten gewählt da mindestens

$$ \xi = n = 4 \tag{6.5} $$

freie Parameter vorhanden sein müssen und sich mit den Forderungen an den Funktionswert zum Start- und Endzeitpunkt ($\nu = 0$)

$$ \lambda_u = \xi - 3 + 2(\nu + 1) = 3 \tag{6.6} $$

ergibt. Für das Verfahren wurden die folgenden Parameter gewählt

Parameter	T	λ_x^0	λ_u	δ	κ	ε_{tol}	v_{max}
Wert	1s	4	3	2	3	0.08	5

6.1.2 Ergebnisse für das Versetzen des Wagens

Nach vier Iterationsschritten ($v = 3$, $\lambda_x^3 = 108$) konnte ein Stellgrößenverlauf ermittelt werden für den der Fehler des Endzustands die geforderte Grenze ε_{tol} erfüllt. Die Verläufe der Zustandsgrößen, der Eingangsgröße sowie der Fehlerfunktionen sind in Abbildung 11 dargestellt. Die Abbildung 12 zeigt die Simulation des Vorgangs.

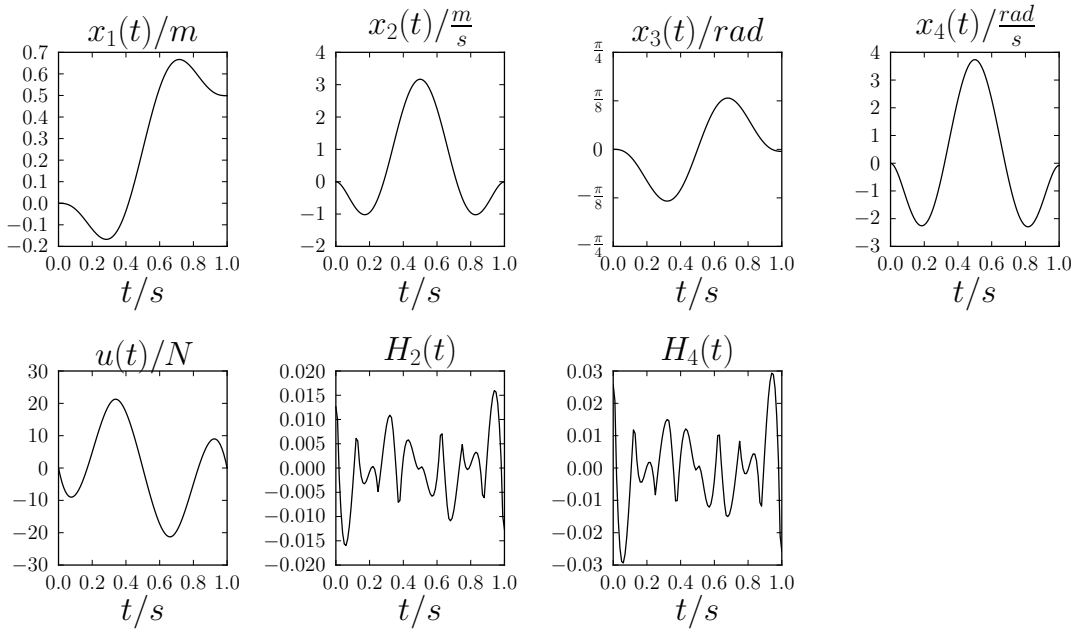


Abbildung 11: Zustands- Eingangs- und Fehlerverläufe für das Versetzen des inversen Pendels.

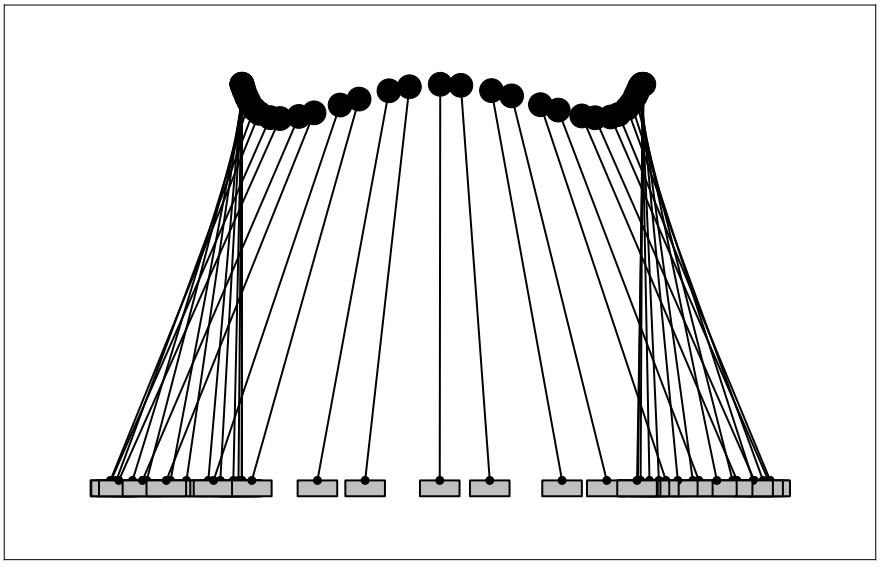


Abbildung 12: Ergebnisse der Simulation des Versetzens des inversen Pendels.

6.1.3 Trajektorie für das Aufschwingen des Pendels

Eine weitere Aufgabestellung stellt das Überführen zwischen den beiden Winkelruhelagen des Pendels dar. Dabei soll das Pendel zum Anfang der Betrachtungen mit $\varphi = \pi$ nach unten hängen und aufgestellt werden $\varphi = 0$. Der Wagen soll zum Ende des Vorgangs wieder an der gleichen Position stehen und sowohl das Pendel als auch der Wagen sollen in Ruhe sein.
Um den Entwurf zu vereinfachen soll das System des Pendels (6.2) durch eine Eingangs-/Ausgangslinearisierung mit dem Ausgang $y = x_1$ partiell linearisiert werden. Dabei erhält das System einen neuen Eingang $\tilde{u} = \ddot{y}$ was der Vorgabe der Wagenbeschleunigung entspricht. Das neue Zustandsraummodell hat die folgende Form

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \tilde{u} \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= \frac{1}{l}(g \sin x_3 + \tilde{u} \cos x_3) \ . \end{aligned} \tag{6.7}$$

Die beiden Zustandsraummodelle sind äquivalent, die partielle Linearisierung führt jedoch auf vereinfachte Gleichungen, die das Lösen des Problems vereinfachen. Der Stellgrößenverlauf für den ursprünglichen Eingang u kann durch das Einsetzen des ermittelten virtuellen Eingangs $\tilde{u}$ in das Regelgesetz berechnet werden.
Gesucht ist der Verlauf $t \mapsto \tilde{u}(t)^*$ der das System mit den folgenden Start-und Endzuständen überführt

$${}^*\mathbf{x}^d = \begin{bmatrix} 0 \\ 0 \\ \pi \\ 0 \end{bmatrix} \xrightarrow{\quad T \quad} {}^\dagger\mathbf{x}^d = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad . \tag{6.8}$$

Analog wie im Beispiel im Abschnitt 3.5 wird eine Spline für x_1 angesetzt der die Randbedingungen für x_1 und x_2 enthält sowie die zusätzlichen Forderungen

$$\tilde{u}({}^*t) = \tilde{u}({}^\dagger t) = 0 \tag{6.9}$$

erfüllt. Damit muss die Splineansatzfunktion für $\tilde{u}$ mindestens $\lambda_{\tilde{u}} = 5$ Abschnitte haben. Als Vorgangsdauer wurde $T = 2$s gewählt und folgende Parameter verwendet

Parameter	T	λ_x^0	λ_u	δ	κ	ε_{tol}	v_{max}
Wert	2s	5	5	2	3	0.05	5

.

6.1.4 Ergebnisse für das Aufschwingen des inversen Pendels

Nach dem dritten Iterationsschritt ($v = 3$, $\lambda_x^3 = 135$) wurde ein Stellgrößenverlauf ermittelt, der das System mit einem Fehler des Endzustandes kleiner ε_{max} überführt. Die entsprechenden Verläufe sind in Abbildung 13 dargestellt mit den dazugehörigen Simulationsergebnissen in Abbildung 14.

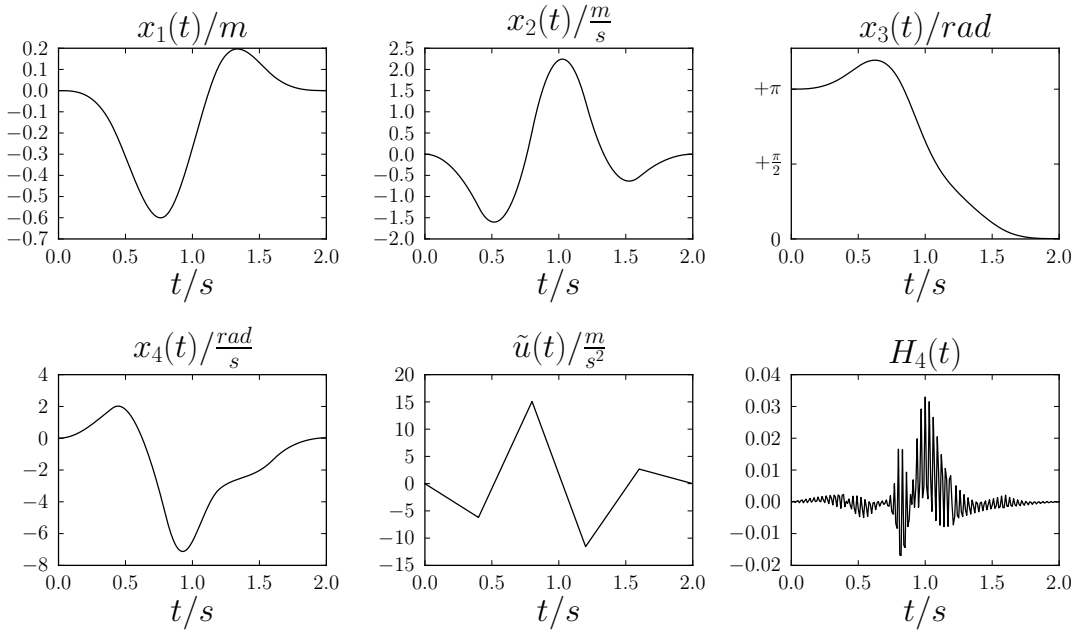


Abbildung 13: Verläufe für das Aufschwingen des inversen Pendels.

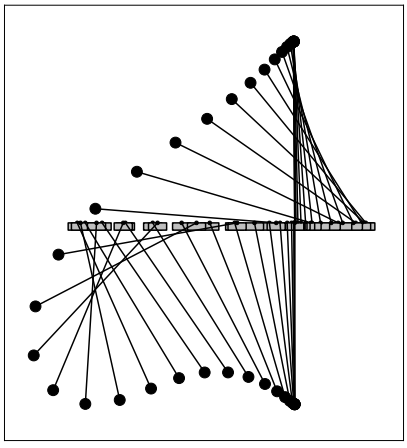
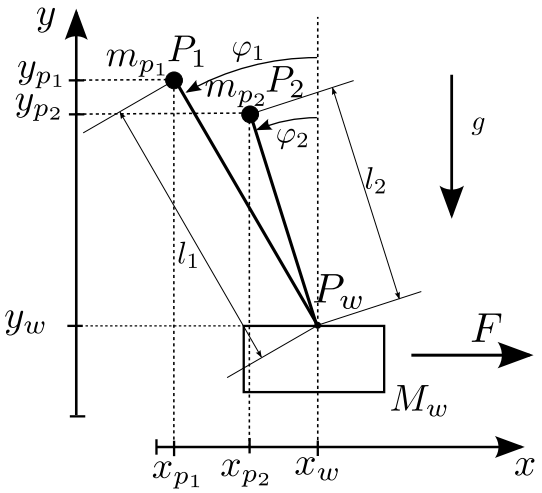


Abbildung 14: Ergebnisse der Simulation des Aufschwingens des inversen Pendels.

6.2 Das inverse Zweifach-Pendel

Das Modell aus Abschnitt 6.1 soll um ein weiteres Pendel am Wagen erweitert werden. Schematisch ist das Zweifach-Pendel in Abbildung 15 dargestellt.



m_{p_1}, m_{p_2}	...	Massen der Pendelköpfe
$l_1 = 0.7\text{m}$,	...	Längen der Pendel
$l_2 = 0.5\text{m}$		
M_w	...	Masse des Wagens
P_w	...	Schwerpunkt des Wagens
P_1, P_2	...	Schwerpunkte der Pendel
y_{p_1}, y_{p_2}, x_{p_1}, x_{p_2}	...	Koordinaten der Pendelköpfe
y_w, x_w	...	Koordinaten des Wagens
φ_1, φ_2	...	Auslenkungen der Pendel
F	...	Stellkraft
$g = 9.81\text{m/s}^2$	...	Erdbeschleunigung

Abbildung 15: Schematische Darstellung des inversen Zweifach-Pendels.

Das System hat den Zustandsvektor

$$\mathbf{x} = [x_1, \dot{x}_1, \varphi_1, \dot{\varphi}_1, \varphi_2, \dot{\varphi}_2]^T \ . \tag{6.10}$$

Durch eine partielle Linearisierung mit $y = x_1$ erhält man das System Zustandsraumdarstellung mit dem Eingang $\tilde{u} = \ddot{y}$

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \tilde{u} \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= \frac{1}{l_1}(g \sin x_3 + \tilde{u} \cos x_3) \\
\dot{x}_5 &= x_6 \\
\dot{x}_6 &= \frac{1}{l_2}(g \sin x_5 + \tilde{u} \cos x_5) \ .
\end{aligned} \tag{6.11}$$

6.2.1 Aufschwingen des Zweifach-Pendels

Geplant werden soll eine Trajektorie, die das System zwischen beiden Ruhelagen

$$ {}^{*}\mathbf{x}^{d} = \begin{bmatrix} 0 \\ 0 \\ \pi \\ 0 \\ \pi \\ 0 \end{bmatrix} \xrightarrow{\quad T \quad} {}^{\dagger}\mathbf{x}^{d} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{6.12} $$

überführt. Beide Pendel sind zu Beginn nach unten gerichtet $\varphi_1 = \varphi_2 = \pi$. Nach einer Überführungszeit von $T = 2$s soll der Wagen wieder an der gleichen Position stehen und die beiden Pendel in Ruhe bei $\varphi_1 = \varphi_2 = 0$ sein. Für die Ermittlung des Stellgrößenverlaufs wurde die folgenden Parameter verwendet

Parameter	T	λ_x^0	λ_u	δ	κ	ε_{tol}	v_{max}
Wert	2s	4	10	2	2	0.08	6

.

Die Anzahl der freien Parameter für die Ansatzfunktion $P_{\tilde{u}}(t)$ wurde mit $\lambda_u = 10$ größer gewählt als die minimale Anforderung von

$$ \begin{aligned} \xi &= 4 \\ \lambda_{\tilde{u}} &= \xi - 3 + 2(1+1) = 5 \ . \end{aligned} \tag{6.13} $$

Mit weniger Freiheitsgraden konnte mit der Abbruchbedingung v_{max} keine befriedigende Lösung ermittelt werden. Die vier Freiheitsgrade ergeben sich aus dem Teilsystem für $[x_3, x_4, x_5, x_6]^T$. Die Größen x_1 und x_2 ergeben sich aus der Integration von $\tilde{u}$, entsprechend genügt eine Ansatzfunktion für diese Systemgrößen mit den Randbedingungen von x_1 und x_2.

6.2.2 Ergebnisse für das Aufschwingen des Zweifach-Pendels

Nach vier Iterationsschritten ($v = 5$, $\lambda_x^5 = 128$) wurde eine Differenz des Endzustandes zur Sollvorgabe mit

$$ \mathbf{x}^{d}({}^{\dagger}t) - \mathbf{x}^{\dagger} = \begin{bmatrix} 0.0083 \\ 0.011 \\ -1.612 \\ -6.207 \\ -0.7929 \\ -3.840 \end{bmatrix} \times 10^{-2} \tag{6.14} $$

erreicht. Die Verläufe der Systemgrößen sowie der Fehlerfunktionen sind in Abbildung 16 dargestellt. Die Abbildung 17 zeigt die Simulation des Manövers.

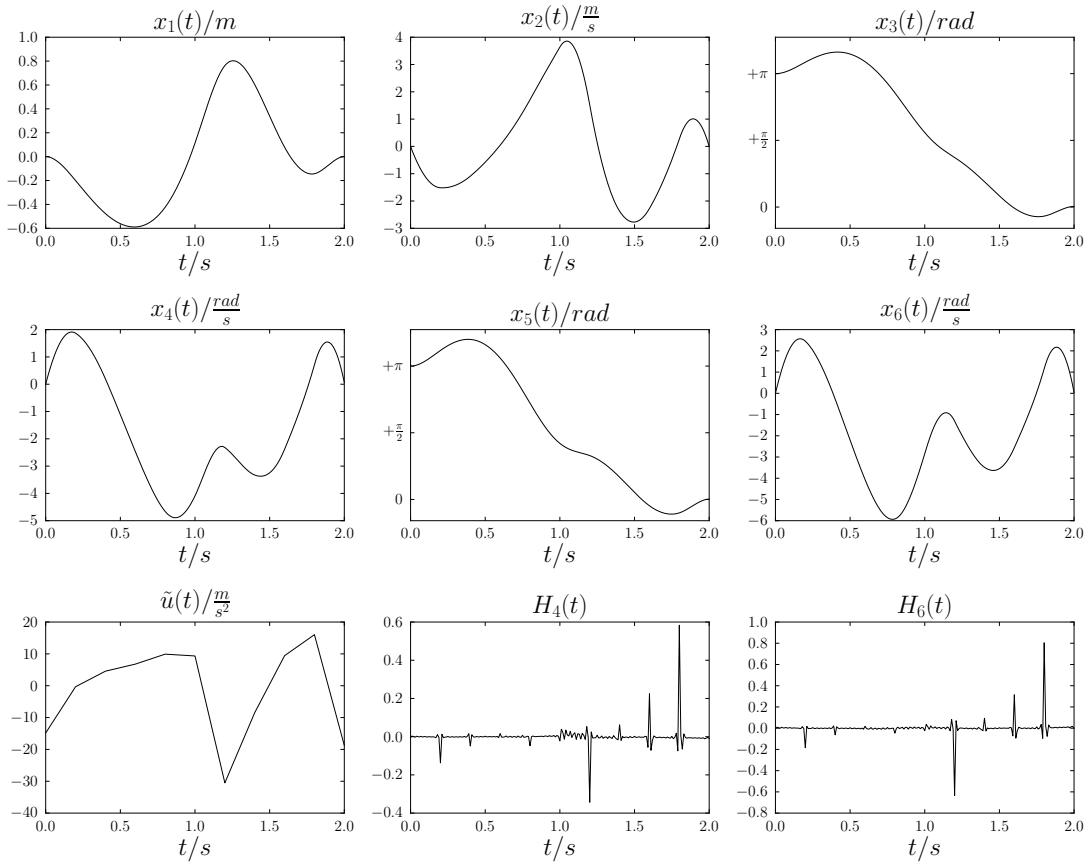


Abbildung 16: Systemgrößen- und Fehlerverläufe des Aufschwingens des inversen Zweifach-Pendels.

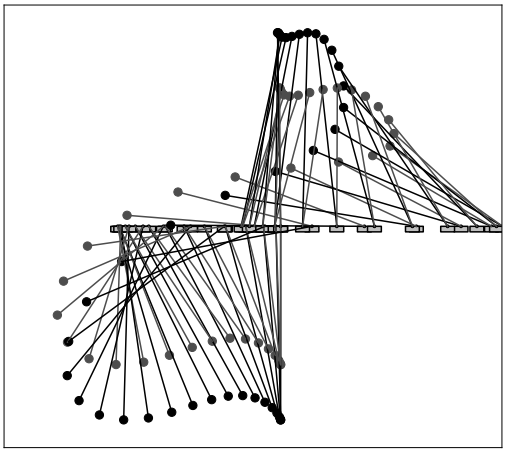
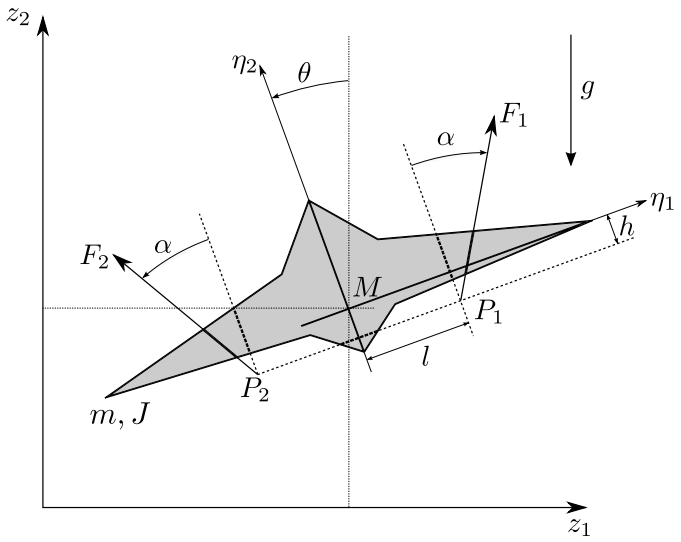


Abbildung 17: Ergebnisse der Simulation des Aufschwingens des inversen Zweifach-Pendels.

6.3 Das senkrecht startende Flugzeug

In diesem Abschnitt wird das Modell eines senkrecht startenden unbemannten Flugzeugs betrachtet. Das Fluggerät verfügt über zwei festmontierte Schubdüsen an den Flügeln, die unabhängig von einander die Schubkräfte F_1 und F_2 aufbringen können. Die beiden Triebwerke sind dabei um den Winkel α gegenüber der flugzeugfesten Achse η_2 geneigt und greifen in den Punkten $P_1 = (l, -h)$ und $P_2 = (-l, -h)$ an. Die Koordinaten des Massenschwerpunkt M des Flugzeugs im Intertial-System werden mit z_1 und z_2 bezeichnet. Zugleich bildet der Punkt den Ursprung des Flugzeug-Koordinatensystems. Die Flugzeug-Achsen sind um den Winkel θ gegenüber der z_2-Achse verdreht [Rud09, HSM92].



$m = 50\text{kg}$	...	Masse des Flugzeugs
$J = 25\text{kgm}^2$	...	Trägheitsmoment um M
M	...	Schwerpunkt des Flugzeugs
P_1, P_2	...	Angriffspunkte der Kräfte
z_1, z_2	...	Koordinaten des Inertial-Systems
η_1, η_2	...	Flugzeug-Koordinatensystem
$\alpha = 5°$	...	Auslenkung der Triebwerke
F_1, F_2	...	Schubkräfte
$g = 9.81\text{m/s}^2$	...	Erdbeschleunigung
$l = 1\text{m}$	...	Abstand der Triebwerke zur η_2-Achse
$h = 0.1\text{m}$	...	Abstand der Triebwerke zur η_1-Achse

Abbildung 18: Schematische Darstellung des Flugzeugs.

Durch das Aufstellen der Impulsbilanzen für das Modell erhält man die Glei-

chungen [Rud09]

$$\begin{aligned} m\ddot{z_1} &= -\sin\theta(F_1+F_2)\cos\alpha + \cos\theta(F_1-F_2)\sin\alpha \\ m\ddot{z_2} &= \cos\theta(F_1+F_2)\cos\alpha + \sin\theta(F_1-F_2)\sin\alpha - mg \\ J\ddot{\theta} &= (F_1-F_2)(l\cos\alpha + h\sin\alpha)\ . \end{aligned} \tag{6.15}$$

Mit dem Zustandsvektor $\mathbf{x} = [z_1, \dot{z}_1, z_2, \dot{z}_2, \theta, \dot{\theta}]^T$ erhält man das zugehörige Zustandsraummodell mit dem $\mathbf{u} = [F_1, F_2]^T$

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{m}(-\sin x_5(u_1+u_2)\cos\alpha + \cos x_5(u_1-u_2)\sin\alpha) \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= \frac{1}{m}(\cos x_5(u_1+u_2)\cos\alpha + \sin x_5(u_1-u_2)\sin\alpha) - g \\ \dot{x}_5 &= x_6 \\ \dot{x}_6 &= \frac{1}{J}(l\cos\alpha + h\sin\alpha)\ . \end{aligned} \tag{6.16}$$

6.3.1 Horizontale und Vertikale Bewegung des Flugzeugs

Für das Fluggerät soll eine Trajektorie geplant werden, die das horizontal ausgerichtete Flugobjekt aus einer Ruhelage *Schweben*, in z_1 und z_2 Richtung bewegt und das Flugzeug wieder in den Schwebezustand überführt.
Das Schweben soll über die Randbedingungen des Eingangs realisiert werden dazu sollen die Ableitungen der Zustandsgrößen zum Anfangs-und Endzeitpunkt verschwinden

$$\dot{z}_1 = \ddot{z}_1 = \dot{z}_2 = \ddot{z}_2 = \dot{\theta} = \ddot{\theta} = 0\ . \tag{6.17}$$

Für die horizontale Lage gilt $\theta = 0$. Aus den Forderungen und den Gleichungen (6.15) ergeben sich die Randbedingungen der Eingangsgrößen zu

$$F_1({}^*t) = F_1({}^\dagger t) = F_2({}^*t) = F_2({}^\dagger t) = \frac{mg}{2\cos\alpha}\ . \tag{6.18}$$

Die Sollvorgaben für das Manöver sind

$${}^*\mathbf{x}^d = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \xrightarrow{\quad T \quad} {}^\dagger\mathbf{x}^d = \begin{bmatrix} 10 \\ 0 \\ 5 \\ 0 \\ 0 \\ 0 \end{bmatrix}\ . \tag{6.19}$$

für eine Dauer von $T = 3s$. Die

$$\xi_{u_1} + \xi_{u_2} = \xi \geq n = 6 \tag{6.20}$$

freien Parameter für den Eingang wurden gleichmäßig auf die beiden Ansatzfunktionen P_{u_1} und P_{u_2} aufgeteilt $\xi_{u_1} = \xi_{u_2}$. Mit den Forderungen an die Randwerte der Funktionen, resultierend aus den Ruhelagen ergibt sich die Anzahl der Splineabschnitte zu

$$\lambda_{u_1} = \lambda_{u_2} \geq \xi_{u_1} - 3 + 2(0 + 1) = 3 \ . \tag{6.21}$$

Für die Berechnung wurde $\lambda_{u_1} = \lambda_{u_2} = 4$ und die folgenden Verfahrensparameter gewählt

Parameter	T	λ_x^0	$\lambda_{u_1}, \lambda_{u_2}$	δ	κ	ε_{tol}	v_{max}
Wert	3s	4	3	2	5	0.01	6

.

6.3.2 Ergebnisse für der Trajektorienberechnung des Flugzeugs

Nach drei Iterationen ($v = 2$, $\lambda_x^2 = 100$) konnten die Stellgrößenverläufe, die das System mit der geforderten Toleranz ε_{tol} in den Endzustand $\mathbf{x}^\dagger$ überführen, ermittelt werden. Die Abbildung 19 zeigt die dazu gehörigen Verläufe. Zudem ist der Vorgang in Abbildung 20 veranschaulicht.

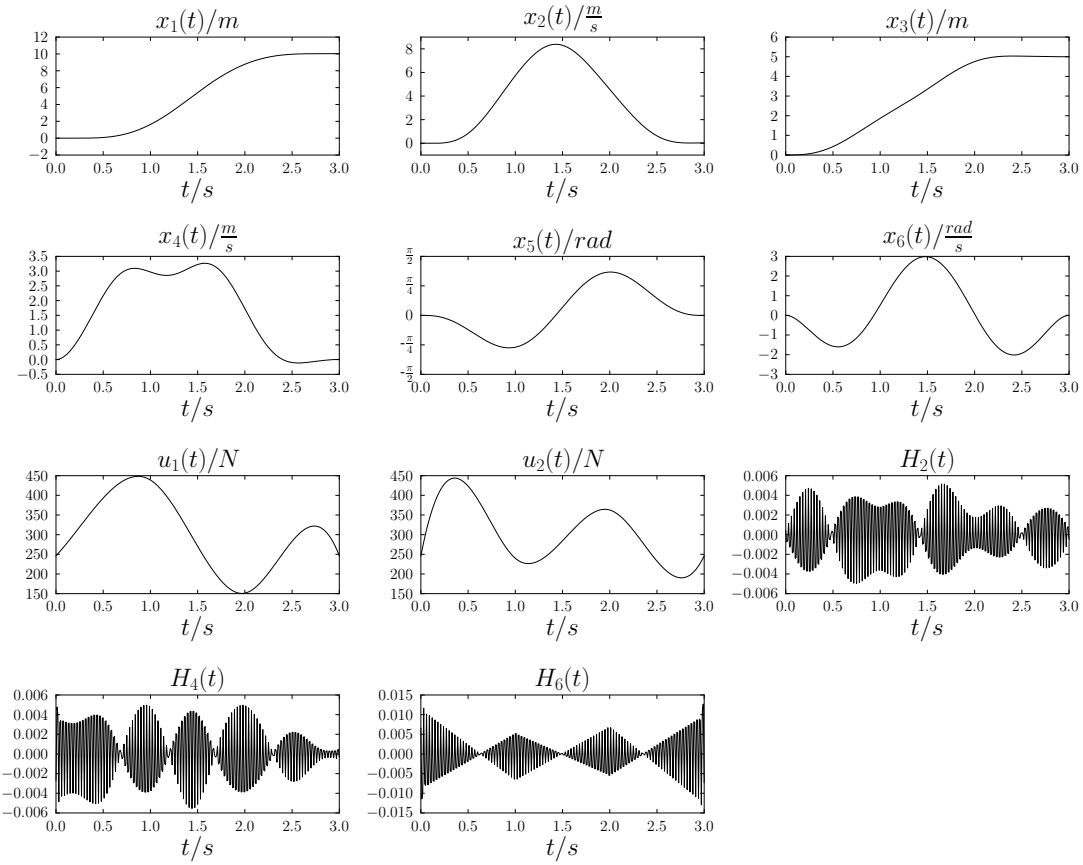


Abbildung 19: Verläufe für die gleichzeitige horizontale und vertikale Bewegung des Flugzeugs.

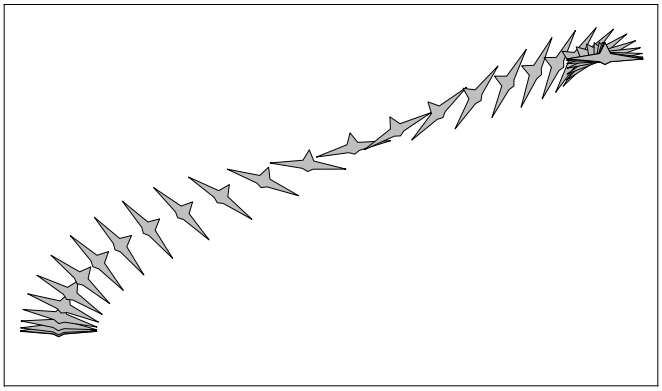
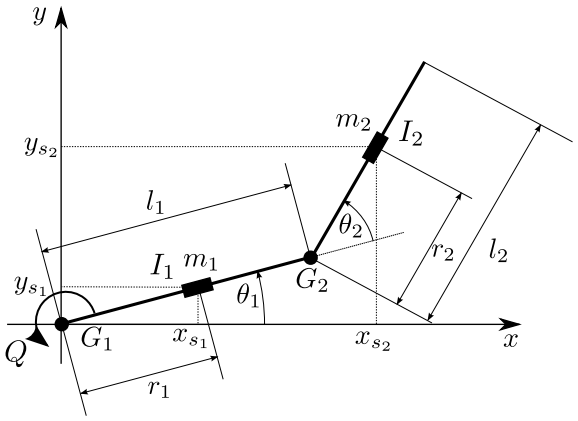


Abbildung 20: Ergebnisse der Simulation der horizontalen und vertikalen Bewegung des Flugzeugs.

6.4 Der Unteraktuirter Manipulator

In diesem Abschnitt wird das Modell eines unteraktuirten Manipulators, wie in Abbildung 21 dargestellt, behandelt. Das System besteht aus zwei Stäben der Masse m_1 und m_2 die über das Gelenk G_2 miteinander verbunden sind. Der Winkel zwischen ihnen wird mit θ_2 bezeichnet. Das Gelenk G_1 verbindet den ersten Stab mit dem Inertial-System, der Winkel zur x-Achse trägt die Bezeichnung θ_1. Im Gelenk



m_1,m_2	...	Stabmassen
l_1,l_2	...	Stablängen
r_1,r_2	...	Achsabstände der Schwerpunkte
I_1, I_2	...	Trägheitsmomente
x_{s_1}, y_{s_1}, x_{s_2}, y_{s_2},	...	Koordinaten der Massenschwerpunkte
G_1, G_2	...	Gelenke
θ_1, θ_2	...	Gelenkwinkel
Q	...	Drehmoment

Abbildung 21: Schematische Darstellung des unteraktuirten Manipulators.

G_1 wird das Stellmoment Q aufgebracht. Die Stäbe besitzen die Trägheitsmomente I_1 und I_2. Die Abstände der Massenschwerpunkte zu den Gelenken sind r_1 und r_2. Schematisch ist der Manipulator in Abbildung 21 dargestellt.
Die Modellbildung wurde aus der Quelle [Kno09] entnommen. In dieser wurde Trägheitsparameter η eingeführt

$$\eta = \frac{m_2 l_1 r_2}{I_2 + m_2 r_2^2} \quad . \tag{6.22}$$

Für das hier betrachtete Beispiel wurde *starke Trägheitskopplung* mit $\eta = 0.9$ angenommen.
Durch partielle Linearisierung mit dem Ausgang $y = \theta_1$ erhält man die Zustandsdarstellung mit dem Zuständen $\mathbf{x} = [\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2]^T$ und dem neuen Eingang $\tilde{u} = \ddot{\theta}_1$

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \tilde{u} \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= -\eta \sin x_3 x_2^2 - (1 + \eta \cos x_3)\tilde{u} \ . \end{aligned} \tag{6.23}$$

6.4.1 Ruhelagenüberführung des Manipulators

Für das System soll eine Trajektorie für das Überführen zwischen zwei Ruhelagen bestimmt werden.

$$ {}^{*}\mathbf{x}^{d} = \begin{bmatrix} 0 \\ 0 \\ 0.4\pi \\ 0 \end{bmatrix} \xrightarrow{\quad T \quad} {}^{\dagger}\mathbf{x}^{d} = \begin{bmatrix} 0.2\pi \\ 0 \\ 0.2\pi \\ 0 \end{bmatrix} \quad . \tag{6.24} $$

Der Eingangsverlauf soll beim Übergang in die Start- und Endruhelagen frei von Sprüngen sein

$$ \tilde{u}({}^{*}t) = \tilde{u}({}^{\dagger}t) = 0 \ . \tag{6.25} $$

Aus dem Teilsystem x_3,x_4 ergibt sich die Forderung an die freien Parameter für $\tilde{u}$

$$ \xi \geq n' = 2 \qquad n'...\text{Dimension des Teilsystem}(x_3, x_4) \quad . \tag{6.26} $$

Aus der Berücksichtigung der Systemstruktur und den Randbedingungen von x_1 und x_2 ergibt sich Anzahl der Splineabschnitte

$$ \lambda_{\tilde{u}} \geq \xi - 3 + 2(2+1) = 5 \ . \tag{6.27} $$

Gute Ergebnisse konnten jedoch erst mit einer Ansatzfunktion mit $\lambda_{\tilde{u}} = 20$ erreicht werden. Das Verfahren wurde mit den folgenden Parametern durchgeführt

Parameter	T	λ_x^0	λ_u	δ	κ	ε_{tol}	v_{max}
Wert	1.8s	5	20	2	3	0.01	5

.

6.4.2 Ergebnisse der Ruhelagenüberführung des Manipulators

Nach drei Schritten ($v = 2$, $\lambda_x = 135$) erfüllt der ermittelte Trajektorienverlauf die Toleranzvorgaben. Die Abbildung 22 zeigt die dazugehörigen Verläufe mit dem Veranschaulichung des Simulation in Abbildung 23.

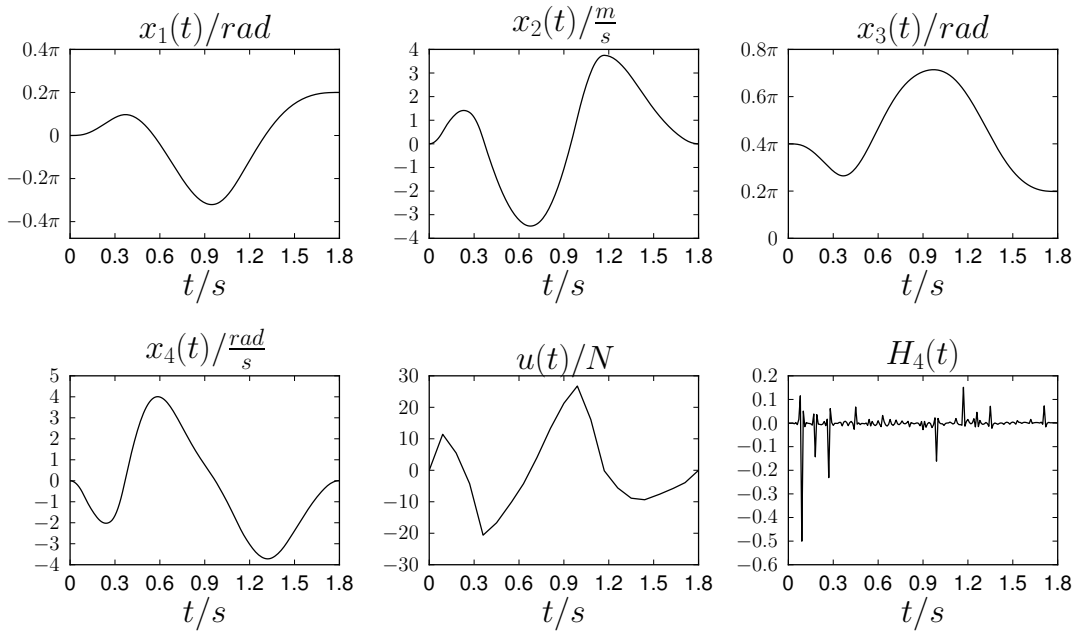


Abbildung 22: Verläufe für die Ruhelagenüberführung des Manipulators.

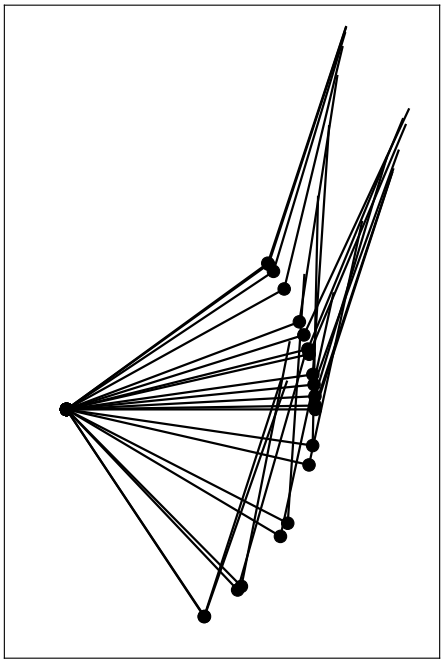
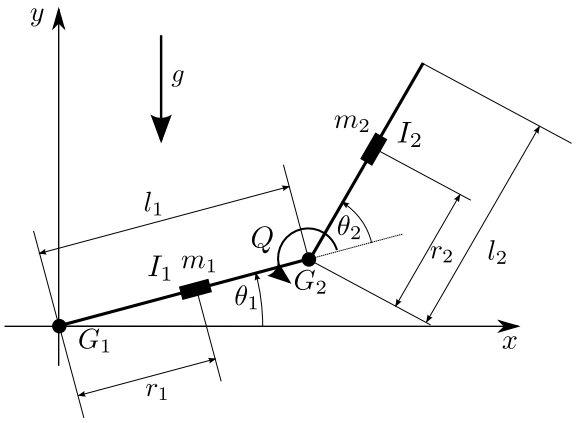


Abbildung 23: Ergebnisse der Simulation der Ruhelagenüberführung des unteraktuierten Manipulators.

6.5 Der Acrobot

Eines weiteres interessantes Beispiel, ist das des Acrobot. Das Modell lässt sich als vereinfachte Modellierung eines Turners verstehen der mit beiden Händen an einer Reckstange hängt. Die Bewegungen des gesamten Systems soll nur durch Bewegung der Hüfte gesteuert werden. Der Körper des Turners wird durch zwei im Gelenk G_2 mit einander verbundene Stäbe dargestellt. Der Stab 1 ist im Gelenk G_1 beweglich mit dem Inertial-System verbunden, dies entspricht dem Umgreifen der Reckstange mit den Händen. Für das Modell wurden zwei gleichlange Stäbe mit der Länge $l_1 = l_2 = l$ und mit einer homogenen Massenverteilung $m_1 = m_2 = m$ über die gesamte Stablänge angenommen. Dies entspricht nicht den Proportionen eines Menschen, zudem wurden keine Beschränkungen an die Beweglichkeit des Hüftgelenks gestellt. Die Abbildung 24 zeigt die schematische Darstellung des Modells.



$m_1 = m_2 = m = 1.0\text{kg}$	...	Stabmassen
$l_1 = l_2 = l = 0.5\text{m}$	...	Stablängen
I_1, I_2	...	Trägheitsmomente
$r_1 = r_2 = \frac{l}{2}$	...	Abstände der Massenschwerpunkte
G_1, G_2	...	Gelenke
θ_1, θ_2	...	Gelenkwinkel
Q	...	Drehmoment
$g = 9.81\text{m/s}^2$	...	Erdbeschleunigung

Abbildung 24: Schematische Darstellung des Acrobots.

Die Modellbildung wurde der Quelle [Spo94] entnommen.
Mit der zuvor angenommen Modellparametern und den Schreibabkürzungen

$$
\begin{aligned}
I &= I_1 = I_2 = \frac{1}{3}ml^2 \\
d_{11} &= m\frac{l^2}{4} + m(l^2 + \frac{l^2}{4} + l^2\cos\theta_2) + 2I \\
h_1 &= -\frac{1}{2}ml^2\sin\theta_2(\dot{\theta}_2(\dot{\theta}_2 + 2\dot{\theta}_1)) \\
d_{12} &= m(\frac{l^2}{4} + \frac{l^2}{2}\cos\theta_1) + I \\
\phi_1 &= \frac{3}{2}mlg\cos\theta_1 + \frac{1}{2}mlg\cos(\theta_1 + \theta_2)
\end{aligned}
\tag{6.28}
$$

sowie dem Zustandsvektor $\mathbf{x} = [\theta_2, \dot{\theta}_2, \theta_1, \dot{\theta}_1]$ ergibt sich die Zustandsraumdarstel-

lung mit dem virtuellen Eingang $\tilde{u} = \ddot{\theta}_2$ zu

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \tilde{u} \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= -d_{11}^{-1}(h_1 + \phi_1 + d_{12}\tilde{u}) \ . \end{aligned} \tag{6.29}$$

6.5.1 Aufschwingen des Acrobot

Für das Modell soll der Verlauf der Stellgröße für ein Aufschwingen des Turners ermittelt werden. Ausgangspunkt der Bewegung sind die beiden nach unten hängenden Stäbe die in einer Manöverdauer von $T = 2$s in eine weitere Ruhelagen in der beiden Stäbe vertikal nach oben zeigen überführt werden. Zu Anfang und zum Ende des Vorgangs sollen die Stellgrößen stetig in die Nulllage übergehen dies wird durch die Randbedingungen des Eingangs

$$\tilde{u}({}^{*}t) = \tilde{u}({}^{\dagger}t) = 0 \tag{6.30}$$

gefordert. Die Anzahl an freien Parametern für die Ansatzfunktion wurde mit $\xi = 13$ größer als die Minimalforderung von $\xi_{min} = 8$ gewählt. Die Anfangs- und Endzustände lauten

$${}^{*}\mathbf{x}^{d} = \begin{bmatrix} 0 \\ 0 \\ \frac{3}{2}\pi \\ 0 \end{bmatrix} \xrightarrow{\quad T \quad} {}^{\dagger}\mathbf{x}^{d} = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{2}\pi \\ 0 \end{bmatrix} \quad . \tag{6.31}$$

Die Berechnung wurde mit den folgenden Parametern durchgeführt

Parameter	T	λ_x^0	λ_u	δ	κ	ε_{tol}	v_{max}
Wert	2s	4	10	2	5	0.01	5

.

6.5.2 Ergebnisse des Aufschwingens des Acrobot

Nach drei Berechnungsschritten ($v = 2$, $\lambda_x = 100$) konnte ein Stellgrößenverlauf, der das System mit der Einhaltung der Fehlertoleranz ε_{tol} in den gewünschten Endzustand überführt. Die Abbildung 25 zeigt die ermitellten Verläufe und in Abbildung 26 sind die dazugehörigen Veranschaulichung der Simulation dargestellt.

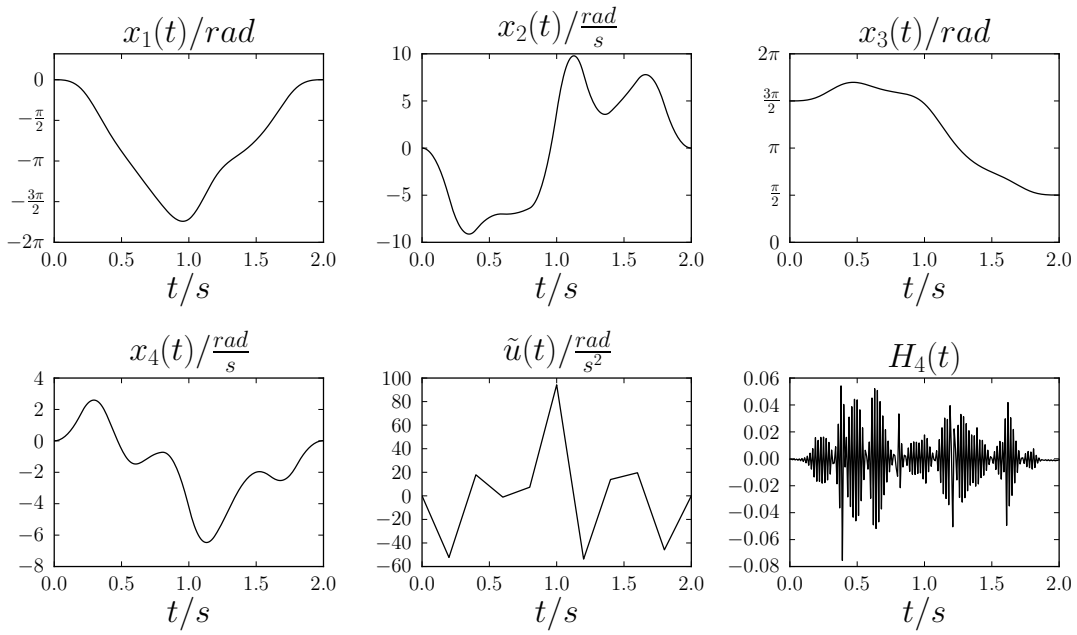


Abbildung 25: Zustandsverläufe mit den dazugehörigen Stellgrößen- und Fehlerverläufen für das Aufschwingen des Acrobots.

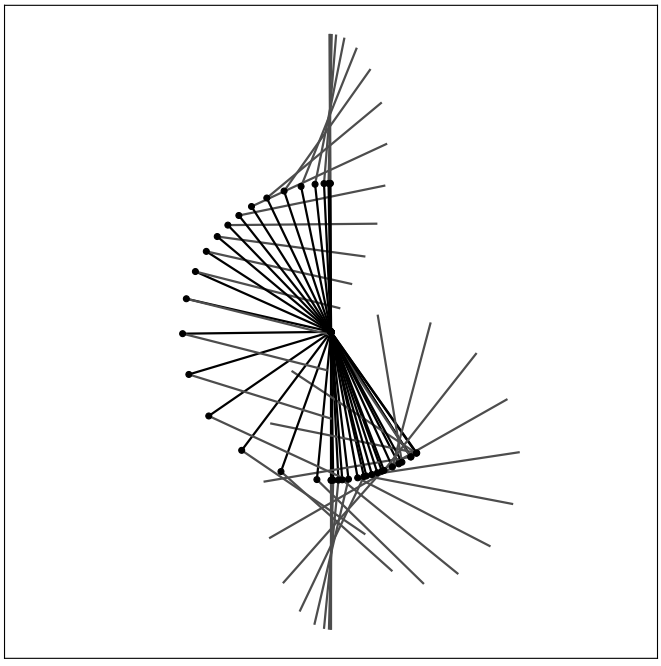


Abbildung 26: Ergebnisse der Simulation des Aufschwingens des Acrobots.

7 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde eine Verfahren zur numerischen Trajektorienbestimmung entwickelt, das eine flexible Implementierung von verschiedenen Aufgabenstellung erlaubt. Im ersten Abschnitt wurde die Notwendigkeit der Trajektorienplanung in der Regelungstechnik und die damit verbundene Klasse von Systemen diskutiert. In den darauf folgenden Kapiteln konnte mit den erörterten Grundlagen, des Kollokationsverfahrens sowie der numerischen Lösung des resultierenden Gleichungssystemen, im Abschnitt 5 ein komplexer, iterativer Algorithmus entworfen werden. Dieser berücksichtigt dabei die Struktur von typischen Modellen der Regelungstechnik und nutzt die Erkenntnisse aus Abschnitt 3 um den numerischen Aufwand des Verfahrens zu reduzieren. Die dargelegte Konstruktion des Kollokationsverfahrens mit Splinefunktionen hat sich dabei als sehr hilfreich erwiesen und ermöglichte eine gute Automatisierbarkeit beim Erstellen der Ansatzfunktionen.
Im letzten Abschnitt wurden verschiedene Entwurfsaufgaben für Systeme aus der Mechanik realisiert. Hier konnte eine Relevanz und Funktionstüchtigkeit des betrachteten Lösungsansatzes nachgewiesen werden. Zudem kann die erstellte Software, als Ausgangspunkt für weitere Betrachtungen dienen.

Für zukünftige Arbeiten, die das Verfahren aufgreifen, wäre eine Untersuchung von Trajektorien mit Eingangs- bzw. Zustandsbeschränkungen möglich. Zudem kann der Kollokationsansatz weiter entwickelt werden. Bei der Recherche zur Arbeit ergaben sich weitere Möglichkeiten, speziell bei Wahl der Kollokationspunkte bzw. bei der Konstruktion der Ansatzfunktionen, die die Effizienz und die Fehlergenauigkeit weiter steigern können. Des weiteren kann eine Verbesserung der Implementierung vorgenommen werden, um die Probleme, die durch symbolische Konstruktion mit `sympy` entstanden sind, zu umgehen. Hierbei sei speziell auf das Projekt `openopt` [Kro] hingewiesen, das besonders auf Optimierungsaufgaben und die damit verbundene Konstruktion von Gleichungssystemen abzielt.

Literatur

[AD10] Antritter, Felix und Joachim Deutscher: *Folgeregelung nichtlinearer Eingrößensysteme mittels linearer zeitvarianter Ausgangsrückführung (Tracking Control for Nonlinear Single Input Systems Using Linear Time-Varying Output Feedback).* Automatisierungstechnik, 58(7):383–393, 2010.

[Ada09] Adamy, Jürgen: *Nichtlineare Regelungen.* Springer, 2009.

[CH95] Chung, Chung Choo und John Hauser: *Nonlinear control of a swinging pendulum.* Automatica, 31(6):851 – 862, 1995.

[CPJ02] Chatterjee, Debasish, Amit Patra und Harish K. Joglekar: *Swing-up and stabilization of a cart–pendulum system under restricted cart track length.* Systems & Control Letters, 47(4):355 – 364, 2002.

[Deu08] Deuflhard, Peter: *Numerische Mathematik: Eine algorithmisch orientierte Einführung (de Gruyter Lehrbuch) (German Edition).* Walter de Gruyter, 2008.

[Deu12] Deutscher, Joachim: *Zustandsregelung verteilt-parametrischer Systeme (German Edition).* Springer, 2012.

[DR08] Dahmen, Wolfgang und Arnold Reusken: *Numerik für Ingenieure und Naturwissenschaftler (Springer-Lehrbuch) (German Edition).* Springer, 2008.

[Fli90] Fliess, M.: *Generalized controller canonical form for linear and nonlinear dynamics.* IEEE, 35(9):994–1001, 1990.

[FLR95] Fliess, Michel, Jean Lévine und Pierre Rouchon: *Flatness and defect of nonlinear systems: Introductory theory and examples.* 1995.

[Fö78] Föllinger, Otto: *Entwurf zeitvarianter Systeme durch Polvorgabe.* Regelungstechnik, 1978.

[Gra06] Graichen, Knut: *Feedforward Control Design for Finite-Time Transition Problems of Nonlinear Systems with Input and Output Constraints (Berichte Aus Dem Institut Fur Systemdynamik Universitat Stuttgart).* Shaker Verlag GmbH, Germany, 2006.

[GW08] Griewank, Andreas und Andrea Walther: *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, Second Edition.* Society for Industrial and Applied Mathematic, 2008.

[GZ06] Graichen, Knut und Michael Zeitz: *Inversionsbasierter Vorsteuerungsentwurf mit Ein- und Ausgangsbeschränkungen (Inversion-Based Feedforward Control Design under Input and Output Constraints).* Automatisierungstechnik, Seiten 187–202, 2006.

[Her01] Hermann, Martin: *Numerische Mathematik.* Oldenbourg, München [u.a.], 2001.

[Her04] Hermann, Martin: *Numerik gewöhnlicher Differentialgleichungen.* Oldenbourg Wissensch.Vlg, 2004.

[HSM92] Hauser, John, Shankar Sastry und George Meyer: *Nonlinear control design for slightly non-minimum phase systems: Application to V/STOL aircraft.* Automatica, 28(4):665 – 679, 1992.

[Ise08] Iserles, Arieh: *A First Course in the Numerical Analysis of Differential Equations (Cambridge Texts in Applied Mathematics).* Cambridge University Press, 2008.

[JOP+] Jones, Eric, Travis Oliphant, Pearu Peterson et al.: *SciPy: Open source scientific tools for Python*, 2001–.

[KAA12] Kefferpütz, Klaus, Carlo Ackermann und Jürgen Adamy: *Zwei-Freiheitsgrade-Regelung linearer Systeme mit Stellgrößen- und Stellratenbegrenzungen.* at-Automatisierungstechnik, 60(3):155–167, March 2012.

[Kno09] Knoll, Carsten: *Steuerung und Regelung eines nicht-holonomen Manipulatormodells.* TU Dresden, Diplomarbeit, April 2009. Diplomarbeit.

[Kro] Kroshko, Dmitrey: *OpenOpt: Free scientific-engineering software for mathematical modeling and optimization*, 2007–.

[MMR03] Martin, Ph., R. M. Murray und P. Rouchon: *Flat systems, equivalence and trajectory generation.* 2003.

[Nur89] Nurnberger, Gunther: *Approximation by Spline Functions.* Springer-Verlag Berlin and Heidelberg GmbH & Co. K, 1989.

[Rot97] Rothfuss, R.: *Anwendung der flachheitsbasierten Analyse und Regelung nichtlinearer Mehrgrößensysteme.* Fortschritt-Berichte VDI. Reihe 8, Mess-, Steuerungs- und Regelungstechnik. VDI-Verlag, 1997.

[RP12] Röbenack, K. und F Paschke: *Approximately linear tracking control of nonlinear systems.* Proc. Appl. Math. Mech., 2012.

[RR97] Ralf Rothfuss, Joachim Rudolph, Michael Zeitz: *Flachheit: Ein neuer Zugang zur Steuerung und Regelung nichtlinearer Systeme.* AT97, 1997.

[Rud00] Rudolph, J.: *Flatness-based control by quasi-static feedback illustrated on a cascade of two chemical reactors.* International Journal of Control 01/2000, 2000.

[Rud03] Rudolph, Joachim: *Beiträge zur flachheitsbasierten Folgeregelung linearer und nichtlinearer Systeme endlicher und unendlicher Dimension.* 2003.

[Rud09] Rudolph, Joachim: *Flachheitsbasierte Folgeregelung.* 2009.

[Rö05] Röbenack, Klaus: *Regler- und Beobachterentwurf für nichtlineare Systeme mit Hilfe des Automatischen Differenzierens.* Shaker Verlag, 2005.

[Sch04] Schulz, Gerd: *Regelungstechnik 1.* Oldenbourg Wissensch.Vlg, 2004.

[SK00] Shampine, L.F., M.W. Reichelt und J. Kierzenka: *Solving Boundary Value Problems for Ordinary Differential Equations in MATLAB with bvp4c.* 2000.

[Spo94] Spong, M.W.: *Swing up control of the Acrobot.* Seiten 2356 –2361 vol.3, may 1994.

[SS94] Scheja, Günter und Uwe Storch: *Lehrbuch der Algebra: Unter Einschluß der linearen Algebra Teil 1 (Mathematische Leitfäden) (German Edition).* Vieweg+Teubner Verlag, 1994.

[SW95] Strehmel, K. und R. Weiner: *Numerik gewöhnlicher Differentialgleichungen.* Teubner Studienbücher, 1995.

[Sym13] SymPy Development Team: *SymPy: Python library for symbolic mathematics*, 2013.

[TS92] Törnig, Willi und Peter Spellucci: *Numerische Mathematik für Ingenieure und Physiker: Band 2: Numerische Methoden der Analysis (German Edition).* Springer, 1992.

[WL11] Walter, Sebastian F. und Lutz Lehmann: *Algorithmic differentiation in Python with AlgoPy.* Journal of Computational Science, 2011.

Abbildungsverzeichnis