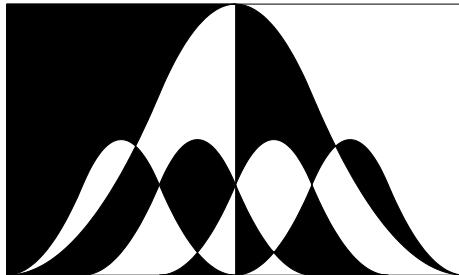


**Effiziente Lösung polynomialer und
nichtpolynomialer Gleichungssysteme mit Hilfe
von Subdivisionsalgorithmen**



Vom Fachbereich Mathematik
der Technischen Universität Darmstadt
zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte **Dissertation**

von
Dipl.-Math. Joachim Gaukel
geboren in Leonberg

Referent:	Prof. Dr. U. Reif
Koreferent:	Prof. Dr. K. Höllig
Tag der Einreichung:	5. Juni 2003
Tag der mündlichen Prüfung:	4. Juli 2003

Darmstadt 2003

D 17

Danksagung

Ganz herzlich möchte ich mich bei Herrn Prof. Dr. Ulrich Reif für die kompetente Betreuung meiner Arbeit bedanken. Außerdem bei den Mitgliedern der Arbeitsgruppe „Differenzialgeometrie und Geometrische Datenverarbeitung“ der Fakultät Mathematik an der Universität Darmstadt, die da wären Prof. Dr. Ulrich Reif, Prof. Dr. Karsten Große-Brauckmann, Prof. Dr. Erich Hartmann, Dipl. Math. Bernhard Mößner und Dr. Steffen Fröhlich und natürlich bei Frau Sybille Drexler. Sie alle haben zu einer wunderbaren menschlichen Atmosphäre beigetragen, die den Raum schafft, sich auch fachlich hervorragend entwickeln zu können. Des weiteren bedanke ich mich herzlich bei Herrn Prof. Dr. Klaus Höllig aus Stuttgart, der mir während meiner offiziellen Zeit in Darmstadt Unterschlupf an seinem Lehrstuhl gewährt hat und mir ein Arbeitszimmer in Stuttgart zur Verfügung gestellt hat. Außerdem bei Herrn Dipl. Math. Jörg Hörner, Herrn Dipl. Math. Joachim Wipper und ganz besonders bei Herrn Dipl. Math. Bernhard Mößner, die stets geholfen haben, bei fachlichen oder praktischen Fragen und Problemen weiterzuhelfen.

Vielen Dank auch an Herrn Herbert Fickel, einem meiner Mathe-Lehrer am Gymnasium Korntal, unter welchem ich die Freude an der Mathematik entdeckt habe.

Natürlich danke ich auch meinen Eltern, Elisabeth und Ewald Gaukel, die mir meine Ausbildung nicht zuletzt auch finanziell ermöglicht haben, außerdem bei meiner Frau Heike Gaukel, meiner Tochter Naomi und meinem Sohn Joel die mein Leben in wunderbarer Weise bereichern und wertvoll machen.

Schorndorf, den 5. Juni 2003

Joachim Gaukel

Inhaltsverzeichnis

1	Einführung	5
2	Polynomiale Gleichungssysteme	8
2.1	Univariate polynomiale Béziertheorie	8
2.2	Multivariate polynomiale Béziertheorie	21
3	Formulierung von effizienten Algorithmen	30
3.1	Vorgehen ohne Einschlusschätzer	30
3.1.1	Existenz und Eindeutigkeit von Nullstellen, Konvergenzgeschwindigkeit	43
3.1.2	Eine hocheffiziente Variante für einen Spezialfall	46
3.1.3	Speicherplatzbedarf und Rechenaufwand des Nullstellen-Algorithmus Roots	50
3.2	Vorgehen mit Einschlusschätzer	53
3.3	Vergleich der Ansätze mit bzw. ohne Einschlusschätzer	56
3.4	Lösen von polynomialen Gleichungssystemen auf ganz \mathbb{R}^n	58
3.5	Anwendungsbeispiele	58
3.5.1	Schnitt von zwei ebenen impliziten Kurven	59
3.5.2	Schnitt dreier parametrisierter Flächen im \mathbb{R}^3	59
3.5.3	Wilkinson-Polynom	59
3.5.4	Kritische Punkte einer algebraischen Kurve	60
3.5.5	Quadratisches System mit variabler Dimension	61
3.5.6	Konvexe Hülle eines polynomial berandeten Gebiets in \mathbb{R}^2	61
4	Nicht-polynomiale Gleichungssysteme	63
4.1	Theoretische Grundlagen	64
4.1.1	Allgemeine Algorithmen	79
4.2	Exponentialräume	82
4.3	Uniforme Splines in Exponentialräumen	94
4.4	Vergleich der Ansätze mit uniformen Splines bzw. mit der Bézierdarstellung zur Lösung von exponentiellen Gleichungssystemen	115
4.5	Transformation des Gleichungssystems	116
4.6	Besselfunktionen	119
5	Wertung und Ausblick	123
6	Bezeichnungen	125
7	Literaturverzeichnis	126

1 Einführung

Ein fundamentales Problem der Mathematik ist das Lösen von linearen und nichtlinearen Gleichungssystemen $Fx = 0$. Während lineare Systeme bereits 200 Jahre vor Christus in der Abhandlung *Chin chang suan shu* von Chang Ts'ang studiert und gelöst wurden, bereitete die Klasse von nichtlinearen Systemen weit größere Probleme. Die Newton-Iteration und ähnliche Fixpunktiterationen waren dafür lange Zeit nahezu die einzigen praktikablen Verfahren. Diese Iterationen zeichnen sich dadurch aus, dass sie überaus einfach sind und mit ihnen sehr effizient Nullstellen bestimmt werden können, sofern bereits geeignete Approximationen an die Lösungen vorliegen. Allerdings ist man mit ihnen im Allgemeinen nicht in der Lage, alle Lösungen zu finden. Man ging deshalb davon aus, dass es prinzipiell nicht möglich sei, nichtlineare Gleichungssysteme vollständig zu lösen. Spätestens durch die Arbeiten von Bucherger über Gröbner-Basen ([B85] und [C90]) zur Behandlung polynomialer Systeme und die Arbeiten von Moore über Intervall-Arithmetik für nichtpolynomiale Systeme ([M69], [MJ77], [M77], [M78a], [M78b], [M79], [MQ82], [M88]) war diese Ansicht widerlegt. Inzwischen existiert eine ganze Vielzahl von Algorithmen. Eine wunderbare Übersicht darüber mit ausführlichster Literaturliste findet sich in [R98]. Die Ansätze zur Lösung lassen sich grob in vier Klassen aufteilen, nämlich in algebraische Techniken, Homotopie-Ansätze, Fixpunktiterationen und Subdivision. Mit algebraischen Techniken wie den Gröbner-Basen oder Resultanten, lassen sich alle komplexen Nullstellen polynomialer Systeme bestimmen. Der große Vorteil dieser Methoden ist, dass sie theoretisch sehr elegant und sehr gut geeignet zur Implementation in symbolischen Systemen wie Maple sind. Dem gegenüber steht ihre numerische Instabilität bei Implementation in Floating-Point-Arithmetik. Desweiteren wird mit Speicher- und Zeitressourcen nicht effizient umgegangen, sodass sie nur anwendbar bei begrenztem Grad und begrenzter Dimensionalität sind. Die zweite Klasse der Homotopie-Ansätze ([ZG81]) neigt ebenfalls zu numerischer Instabilität. Will man dieses Problem umgehen durch Implementation in exakter rationaler Arithmetik, so werden erneut enorme Speicherkapazitäten benötigt. Fixpunktiterationen wie die Newton-Raphson Iteration ([P40]) oder Numerische Optimierung ([DB74]) sind im Allgemeinen einfach zu programmieren und dabei sehr effizient. Sie besitzen jedoch typischerweise den Nachteil, dass bereits gute Startnäherungen an die Lösungen vorliegen müssen, welche durch andere Ansätze bestimmt werden. Fixpunktiterationen besitzen somit keinen vollständig eigenständigen Charakter. In die vierte Klasse der Subdivisionsansätze fällt unser Ansatz. Lane und Riesenfeld ([LR81]) untersuchten erstmals binäre Subdivision von uniformen Splines. Dabei kommt deren variationsmindernde Eigenschaft zum Tragen. Mit dieser Technik lassen sich die reellen Nullstellen eines polynomialen Systems innerhalb eines kompakten Intervalls bestimmen. Boehm ([Bo80]) und Cohen ([CLR80]) übertrugen diese Idee auf allgemeine nichtuniforme Subdivision von B-Splines. Später entwickelte Geisow ([G83]) Subdivivions-Algorithmen um ebene algebraische Kurven

zu schneiden, die in baryzentrischen Tensorproduktbasen ausgedrückt sind. Koch gelang es in [K96] durch Aufstellung eines zum ursprünglichen Problem äquivalenten Gleichungssystems polynomiale Systeme in Monomform effizient zu lösen. Unser Ansatz schließlich ähnelt dem von Sherbrooke und Patrikalakis ([SP93]), die polynomiale Systeme betrachteten. Dort [SP93] werden zwei Varianten besprochen, von denen sich die erste als besonders leistungsfähig erwies. Leider verliert sie die zunächst erwartete quadratische Konvergenz. Mit dem in dieser Arbeit verfolgten Ansatz wird die quadratische Konvergenz hingegen erhalten.

Wir arbeiten mit der Darstellung der Gleichungen in Bézierform. Diese zeichnet sich dadurch aus, dass die Koeffizienten für die Basisfunktionen geometrische Information über die dargestellte Funktion F enthält. Durch geeignete Betrachtung kann man die Koeffizienten, in diesem Kontext Kontrollpunkte genannt, nämlich als Approximation von F auffassen und den Approximationsfehler mit sehr einfachen Mitteln abschätzen. Somit lässt sich der Bereich möglicher Lösungen von $Fx = 0$ einschränken. Durch Einschränkung des betrachteten Definitionsgebiets, in diesem Zusammenhang Subdivision genannt, konvergieren die Kontrollpunkte aber quadratisch gegen F und somit erhalten wir einen quadratisch konvergenten Algorithmus. Dieser ist in der Lage numerisch stabil alle Nullstellen in einer kompakten Box $[a_1, b_1] \times \dots \times [a_n, b_n]$ zu finden. Dabei wird mit Speicherplatzressourcen effizient umgegangen und auch die Länge des Codes bleibt mit wenigen Hundert Zeilen äußerst moderat.

In dieser Arbeit verfolgen wir das Ziel, konkrete Algorithmen zu formulieren und die notwendige Theorie hierzu vollständig bereit zu stellen. Sie ist dabei wie folgt aufgebaut. In Kapitel 2 stellen wir die Theorie polynomialer Bézierpolynome vor, wobei wir uns auf den Teil beschränken, der für die weitere Arbeit notwendig ist. In Abschnitt 2.1 behandeln wir zunächst den univariaten Fall, in Abschnitt 2.2 verallgemeinern wir die Theorie auf den multivariaten Fall durch Tensorprodukt-Bildung. Im Wesentlichen geben wir im Kapitel 2 Standardtheorie wieder, die in den CAGD-Büchern wie [F00] üblicherweise behandelt wird. Nicht Standard sind in Kapitel 2 die Einschlussschätzer für die Bézierpolynome. Diese gehen zurück auf [NPL99], [LP01] und [Reif00]. Deren mögliche Verallgemeinerung für den multivariaten Fall wurde ebenfalls in den genannten Papers besprochen, während wir eine praktikable Alternative hierzu vorschlagen, die ausgezeichnete Ergebnisse liefert.

In Kapitel 3 wenden wir die Béziertheorie dann an und formulieren zwei Algorithmen zur Lösung polynomialer Gleichungssysteme. Der erste der beiden arbeitet komplett ohne die Einschlussschätzer und wird in Abschnitt 3.1 vorgestellt. Der zweite Algorithmus bezieht die Einschlussschätzer mit ein, wir behandeln ihn in Abschnitt 3.2. Beide Algorithmen konvergieren quadratisch. In Abschnitt 3.1.2 behandeln wir spezielle Klassen von polynomialen Gleichungssystemen, für die sich der Aufwand drastisch reduzieren lässt.

Anschließend legen wir in Kapitel 4 die theoretischen Grundlagen für die Behandlung nicht-polynomialer Systeme, welche in Abschnitt 4.1 im Wesentlichen eine Aufarbei-

tung der Veröffentlichungen [Ma96], [ML96], [ML99], [Ma99] und [DR88] für unsere Zwecke darstellt. Aus diesen Grundlagen leiten wir explizite Algorithmen in MATLAB-Code für die notwendigen Erweiterungen des Programms aus Kapitel 3 an. In Abschnitt 4.2 gehen wir ausführlich auf Systeme ein, die aufgebaut sind aus trigonometrischen, polynomialen und exponentiellen Funktionen und machen so die Algorithmen aus Abschnitt 4.1 numerisch stabil unter Subdivision. Als Alternative zum in Abschnitt 4.1 gegebenen Vorgehen bieten wir in Abschnitt 4.3 einen Algorithmus an, der mit uniformen nicht-polynomialen Splines arbeitet. In Abschnitt 4.6 gehen wir noch explizit auf die Behandlung von Besselfunktionen ein.

2 Polynomiale Gleichungssysteme

In diesem Kapitel wollen wir den Fall polynomialer Gleichungssysteme behandeln. Zunächst wenden wir uns der univariaten Theorie polynomialer Bézierkurven zu. Die dort gewonnen Erkenntnisse lassen sich anschließend problemlos auf den multivariaten Fall verallgemeinern.

2.1 Univariete polynomiale Béziertheorie

Definition 2.1 Sei das k -te Bernsteinpolynom $b_{m,k}$ der Ordnung m definiert durch (siehe Abbildung 1)

$$b_{m,k}(x) := \binom{m-1}{k-1} x^{k-1} (1-x)^{m-k}, \quad x \in [0, 1], \quad k \in 1 : m.$$

Alle Bernsteinpolynome der Ordnung fassen wir zusammen in einem liegenden Vektor

$$B_m := [b_{m,1}, \dots, b_{m,m}].$$

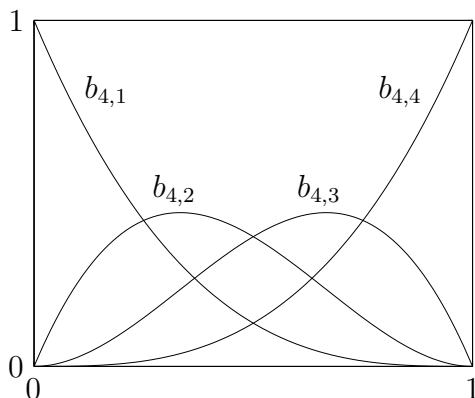


Abbildung 1: Bernsteinpolynome für $m = 4$

Die $b_{m,1}, \dots, b_{m,m}$ bilden eine Basis des Raums $\mathbb{P}_m = \langle 1, x, \dots, x^{m-1} \rangle$ aller Polynome, deren Grad kleiner als m ist. Dies ist unmittelbar klar, wenn man sukzessive zu $b_{m,1}, \dots, b_{m,k}$ noch $b_{m,k+1}$ hinzunimmt. Jede Linearkombination der $b_{m,1}, \dots, b_{m,k}$ besitzt in 1 mindestens eine $(m-k)$ -fache Nullstelle, während $b_{m,k+1}$ lediglich eine $(m-k-1)$ -fache Nullstelle besitzt. $b_{m,k+1}$ kann also nicht linear aus den $b_{m,1}, \dots, b_{m,k}$ kombiniert werden.

Definition 2.2 Für $p_1, \dots, p_m \in \mathbb{R}^d$ nennt man die rechte Seite von

$$p = \sum_{k=1}^m b_{m,k} p_k$$

Bézierdarstellung von p bzw. allgemein eine Bézierkurve der Ordnung m zu den sogenannten Kontrollpunkten $p_k \in \mathbb{R}^d$. Die Kontrollpunkte werden wir im Folgenden in einer Matrix $P = [p_1; \dots; p_m]$ zusammenfassen, jeweils ein Kontrollpunkt in einer Zeile. Somit können wir kurz mit der üblichen Matrixmultiplikation schreiben

$$p = B_m P.$$

Der Vorwärtsdifferenzen-Operator Δ sei definiert durch

$$\Delta P := [p_2 - p_1; \dots; p_m - p_{m-1}] \in \mathbb{R}^{(m-1) \times d};$$

Δ^n sei die n -malige Anwendung von Δ .

Wir sehen unmittelbar, dass z.B. $\Delta^2 P = [p_3 - 2p_2 + p_1; \dots; p_m - 2p_{m-1} + p_{m-2}] \in \mathbb{R}^{(m-2) \times d}$. Die Ableitung einer Bézierkurve der Ordnung m mit den Kontrollpunkten P ist sicherlich eine Bézierkurve der Ordnung $(m-1)$. Die Kontrollpunkte für die Ableitung berechnen sich mittels des Vorwärtsdifferenzen-Operators als $(m-1)\Delta P$, wie uns das folgende Lemma lehrt. Ein Beweis hierfür findet sich in jedem CAGD-Lehrbuch, wie z.B. in [F00].

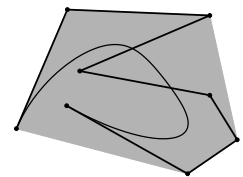
Lemma 2.3 Für $p = B_m P$ gilt $p' = (m-1)B_{m-1}\Delta P$. Folglich ist eine Bézierkurve genau dann ein Polynom der Ordnung n , wenn $\Delta^n P = [0; \dots; 0]$.

Wenn man mit Polynomen arbeitet, so werden diese meist in Monomform angegeben. Nicht so in der geometrischen Datenverarbeitung. Vielmehr wird mit Basen gearbeitet, die eine positive Partition der Eins bilden. Der Grund hierfür ist, dass dann intuitive Zusammenhänge zwischen den Koeffizienten -hier Kontrollpunkte genannt- und der dargestellten Funktion bestehen. Die Basis der Bernsteinfunktionen bildet eine solche Basis, wie wir im folgenden Satz sehen.

Satz 2.4 (Konvexe-Hülle-Eigenschaft) Es gilt $b_{m,k} \geq 0$ und $\sum_k b_{m,k} \equiv 1$ und damit gilt für beliebige Kontrollpunkte $p_k \in \mathbb{R}^d$ und $x \in [0, 1]$:

$$p(x) = B_m(x)P \in \text{ConvexHull}(p_1, \dots, p_m).$$

Außerdem ist $p(0) = p_1$ und $p(1) = p_m$.



Beweis Entwickelt man $1 = (x + (1-x))^m$ binomial, so sieht man, dass

$$1 = 1^m = (x + (1-x))^m = \sum_{k=0}^m \binom{m}{k} x^k (1-x)^{m-k} = \sum_{k=1}^{m+1} b_{m+1,k}(x).$$

□

Als nützliche Approximation einer Bézierkurve erweist sich das Kontrollpolygon, welches wir nun definieren wollen.

Definition 2.5 Sei für $x \in [0, 1]$

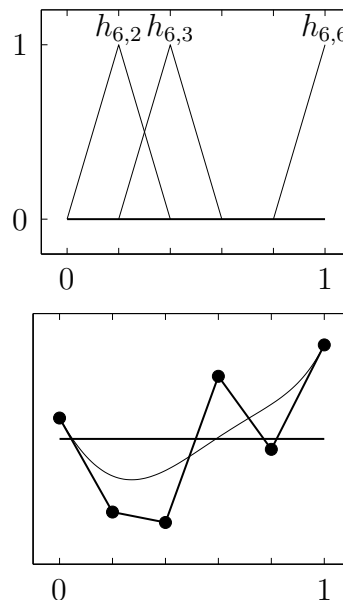
$$h_{m,i}(x) := \begin{cases} 0 & , \quad x \notin \left[\frac{i-2}{m-1}, \frac{i}{m-1} \right] \\ (m-1)x - (i-2) & , \quad x \in \left[\frac{i-2}{m-1}, \frac{i-1}{m-1} \right] \\ i - (m-1)x & , \quad x \in \left[\frac{i-1}{m-1}, \frac{i}{m-1} \right] \end{cases}$$

und $H_m = [h_{m,1}, \dots, h_{m,m}]$. Wir nennen $h_{m,i}$ i -te Hutfunktion.

Wir setzen dann als lineare Interpolation der Kontrollpunkte

$$\bar{p} = \sum_{i=1}^m h_{m,i}(x) p_i = H_m P$$

und nennen \bar{p} das Kontrollpolygon zu p .



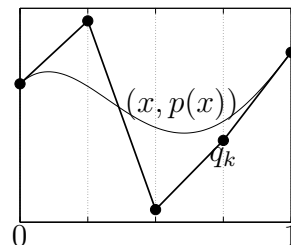
Wie man im zweiten Bild der obigen Definition sieht, besteht ein intuitiver Zusammenhang zwischen dem Kontrollpolygon und der dargestellten Kurve. Die Kurve versucht gewissermaßen, das Kontrollpolygon nachzubilden, ohne dabei aber die Glattheit zu verlieren.

Der zweite Teil des folgenden Satzes erlaubt es uns, Bézierkurven und Kontrollpolygone ohne Angabe eines Koordinatensystems graphisch darzustellen. Der erste Teil ist für unseren Ansatz zur Lösung von Gleichungssystemen von fundamentaler Bedeutung.

Satz 2.6 Mit $p_k := \frac{k-1}{m-1}$ ist $\sum_{k=1}^m b_{m,k}(x) p_k = x$. D.h. für $q_k = (q_{1,k}, \dots, q_{d,k})$ ist der Graph der Bézierkurve gegeben durch $\sum_{k=1}^m b_{m,k}(x) \left(\frac{k-1}{m-1}, q_{k,1}, \dots, q_{k,d} \right)$. Außerdem sind Bézierkurven affin invariant, d.h. mit der affinen Abbildung $a(q) = qM + q_0$, wobei $M \in \mathbb{R}^{d \times d}$ und $q_0 \in \mathbb{R}^{1 \times d}$, gilt:

$$a \left(\sum_{k=1}^m b_{m,k}(x) q_k \right) = \sum_{k=1}^m b_{m,k}(x) a(q_k).$$

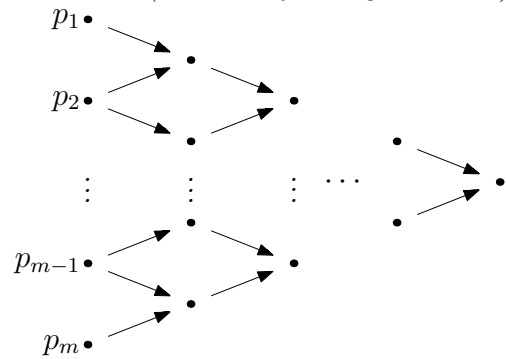
Die Koeffizienten $p_k = \frac{k-1}{m-1}$ nennt man Greville-Abszissen. Für die Greville-Abszissen P gilt $B_m P = H_m P$, d.h. in diesem Fall stimmen das Kontrollpolygon und die dargestellte Bézierkurve überein.



Beweis Mit den Bezeichnungen von oben gilt $p_1 = 0$ und mit den Differenzierungsformeln für Bézierkurven auch $D \sum_{k=1}^m b_{m,k}(x)p_k = (m-1) \sum_{m-1,k} b_{m-1,k}(x)(p_{k+1} - p_k) = (m-1) \frac{1}{m-1} = 1$, also $p(0) = 0$ und $p'(x) \equiv 1$. Die affine Invarianz folgt unmittelbar aus Satz 2.4. \square

Wenn p eine polynomiale Kurve der Ordnung m ist, dann auch $p(a \cdot + b)$. Man kann deshalb sicherlich zwei Kontrollpunktsätze P^l und P^r finden, sodass die zwei Abschnitte für $x \in [0, x_0]$ und $x \in [x_0, 1]$ paramterisiert werden, dass also $p^l(x) = p(xx_0) = \sum b_{m,k}(x)p_k^l$ bzw. $p^r(x) = p((1-x)x_0 + x) = \sum b_{m,k}(x)p_k^r$. Man beachte, dass $\{xx_0 \mid x \in [0, 1]\} = [0, x_0]$ und $\{(1-x)x_0 + x \mid x \in [0, 1]\} = [x_0, 1]$. Wie diese Kontrollpunktsätze für den linken bzw. für den rechten Teil zu berechnen sind, sagt uns der folgende Satz.

Satz 2.7 (DeCasteljau-Algorithmus)



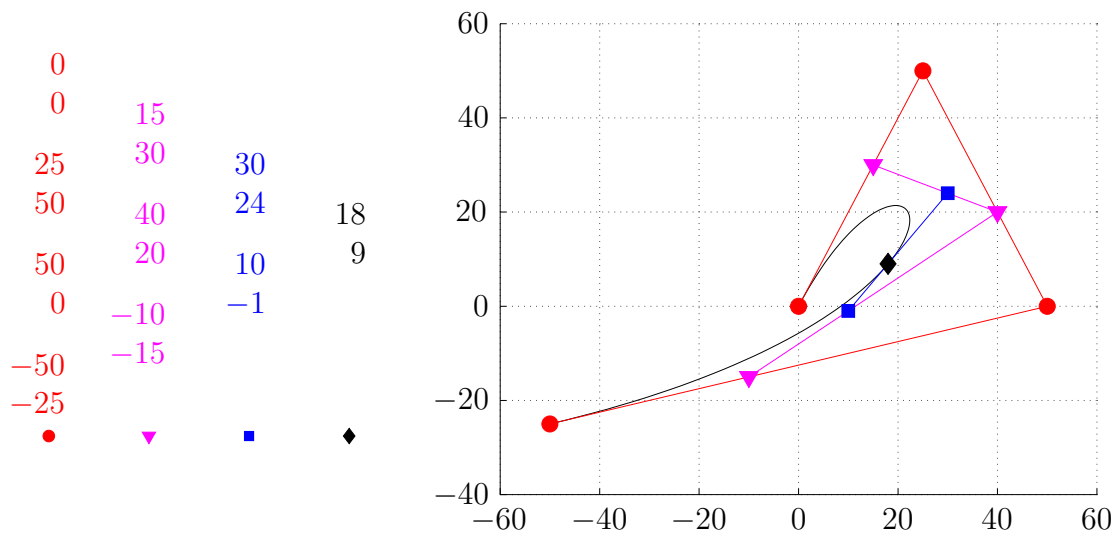
Gegeben seien die m Kontrollpunkte p_1, \dots, p_m zur polynomialen Kurve p und die Unterteilungsstelle x_0 . Man berechne das nebenstehende Schema. Die linke Spalte ist gegeben, alle anderen Kontrollpunkte werden berechnet als Konvexkombination der Punkte, die auf ihn zeigen, wobei ein Aufwärtspfeil für das Gewicht x_0 , ein Abwärtspfeil für das Gewicht $(1-x_0)$ steht.

Die oberen Einträge in den Spalten von links nach rechts sind die Kontrollpunkte P^l für die Kurve $p^l(x) := p(xx_0) = B_m P^l$, bzw. die unteren Einträge in den Spalten von rechts nach links sind die Kontrollpunkte P^r für die Kurve $p^r(x) := p((1-x)x_0 + x) = B_m P^r$. Insbesondere gibt der Kontrollpunkt ganz rechts im Schema den Wert der Bézierkurve an der Stelle x_0 an.

Dieser Satz ist ein Spezialfall von Satz 4.24 und wird dort bewiesen. Den Übergang von Kontrollpunkten P zu P^l bzw. P^r nennt man Subdivision. Wie wir sehen werden, liegen die subdividierten Kontrollpunkte bzw. das zugehörige Kontrollpolygon im Allgemeinen wesentlich dichter bei der dargestellten Kurve als das ursprüngliche Kontrollpolygon. Deshalb spielt die Subdivision von Bézierkurven und deren nicht-polynomialen Verallgemeinerungen in dieser Arbeit eine entscheidende Rolle.

Definition 2.8 Sei $p(x) = B_m(x)P$ und für $0 < a < b < 1$ sei \tilde{P} so, dass $\tilde{p}(x) := p((1-x)a + xb) = B_m(x)\tilde{P}$. Wir setzen den Subdivisionsoperator s als

$$s(P, a, b) := \tilde{P}.$$

Abbildung 2: DeCasteljau-Schema zum Unterteilungspunkt $x_0 = 3/5$

Lemma 2.9 *Der Subdivisionsoperator ist ein Automorphismus in \mathbb{R}^m und es gilt*

$$s(P, a, b) = s(s(P, 0, b), a/b, 1).$$

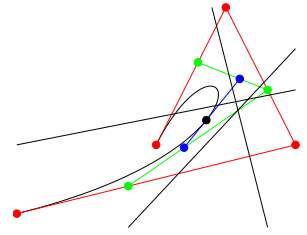
Soll also auf ein Intervall $[a, b] \subset [0, 1]$ subdividiert werden, so können wir folglich diesen Prozess aufspalten in zwei Schritte. In beiden Schritten ist das DeCasteljau-Schema durchzurechnen, zuerst mit $x_0 = b$, dann mit $x_0 = a/b$.

Betrachten wir ein ausführliches Beispiel:

Beispiel 2.10 *Gegeben seien die Kontrollpunkte $P = \begin{bmatrix} 0 & 25 & 50 & -50 \\ 0 & 50 & 0 & -25 \end{bmatrix}^T$. Wegen Satz 2.4 verläuft die Bézierkurve in der konvexen Hülle der Punkte $[0, 0]$, $[25, 50]$, $[50, 0]$ und $[-50, -25]$. Das DeCasteljau-Schema zum Unterteilungspunkt $x_0 = 3/5$ ist in Abbildung 2 zu sehen. Die Kontrollpunkte für den linken Teil $[0, 3/5]$ sind $P^l = \begin{bmatrix} 0 & 15 & 30 & 18 \\ 0 & 30 & 24 & 9 \end{bmatrix}^T$ bzw. für den rechten Teil $[3/5, 1]$ sind die Kontrollpunkte $P^r = \begin{bmatrix} 18 & 10 & -10 & -50 \\ 9 & -1 & -15 & -25 \end{bmatrix}^T$. Wie man sieht, liegt das zusammengesetzte Kontrollpolygon aus dem $[0, 3/5]$ - und dem $[3/5, 1]$ -Teil wesentlich näher an der dargestellten Kurve, als das ursprüngliche Kontrollpolygon.*

Das folgende Lemma besagt, dass die dargestellte Bézierkurve niemals „welliger“ ist als das zugehörige Kontrollpolyon, wobei das Maß der Welligkeit gegeben sei durch die potenzielle Zahl von Schnitten mit einer Hyperebene.

Lemma 2.11 Seien $p_k \in \mathbb{R}^d$ und $p = \sum_k b_{m,k} p_k$, $v \in \mathbb{R}^d$ und $d \in \mathbb{R}$. Wenn wir $V : \mathbb{R}^d \rightarrow \mathbb{R}, x \mapsto \langle v, x \rangle - d$ setzen, so besitzt die Funktion $V \circ p$ nicht mehr Vorzeichenwechsel als die Funktion $V \circ \bar{p}$. D.h. bezüglich einer durch v und d gegebenen Hyperebene $\{x : \langle v, x \rangle = d\}$ wechselt eine Bézierkurve nicht öfter die Seite als das zugehörige Kontrollpolygon.



Beweis Nach Satz 2.13 konvergiert bei Subdivision das Kontrollpolygon gegen die zugehörige Bézierkurve. Subdivision ist aber ein variationsmindernder Prozess, weil er als fortgesetztes „Eckenabschneiden“ aufgefasst werden kann, siehe dazu Abbildung 3. Geht man über von einem Polygonzug wie im linken Bild zu dem des rechten Bildes,

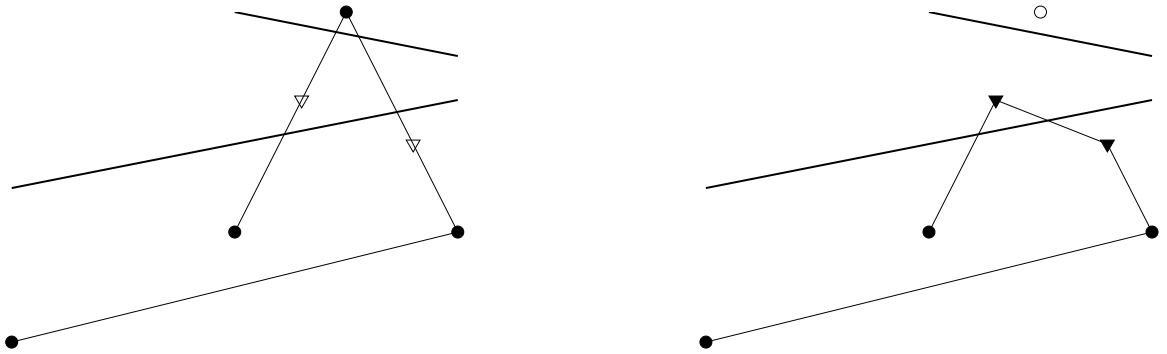


Abbildung 3: Durch Eckenabschneiden werden keine weiteren Schnittpunkte mit Geraden gewonnen

so muss jede Hyperebene, die den rechten Polygonzug schneidet, auch den linken Polygonzug schneiden. \square

Die Bemerkung in Beispiel 2.10 bezüglich des Abstandes des Kontrollpolygons zur Kurve im ursprünglichen bzw. subdividierten Fall werden wir in Satz 2.13 präzise formulieren und beweisen. Das geht sehr elegant mit dem folgenden Lemma. Natürlich sollten wir es nicht nur als Hilfsmittel betrachten, stellt es doch ein sehr effizientes Verfahren zur Beurteilung des Kontrollpolygons in Bezug zur dargestellten Kurve zur Verfügung. Auch später werden wir das Lemma noch gewinnbringend einsetzen. Zunächst definieren wir für einen Vektor $V \in \mathbb{R}^m$ bzw. für eine stetige Funktion $f : [0, 1] \rightarrow \mathbb{R}$ die Maximumnorm wie üblich,

$$\|V\|_\infty := \max_{k \in \{1:m\}} |v_k|, \quad \|f\|_\infty := \max_{x \in [0,1]} |f(x)|.$$

Der folgende Satz über den Abstand einer Bézierkurve zu ihrem Kontrollpolygon findet sich erstmals in [NPL99]. In [Reif00] wird er auf Splines mit beliebigen Knotenfolgen verallgemeinert. Satz 2.12 dient zwei Zwecken. Erstens verwenden wir es als zentralen Baustein zum Beweis der quadratischen Konvergenz bei Subdivision. Zweitens werden wir aus ihm Satz 2.14 ableiten, in dem wir Bézierkurven stückweise linear von oben und unten einschließen.

Satz 2.12 *Für die Differenz einer Bézierkurve $p : [0, 1] \rightarrow \mathbb{R}$ der Ordnung m und ihrem Kontrollpolygon $\bar{p}(x)$ gilt*

$$|p(x) - \bar{p}(x)| \leq \frac{m-1}{2} x(1-x) \|\Delta^2 P\|_\infty,$$

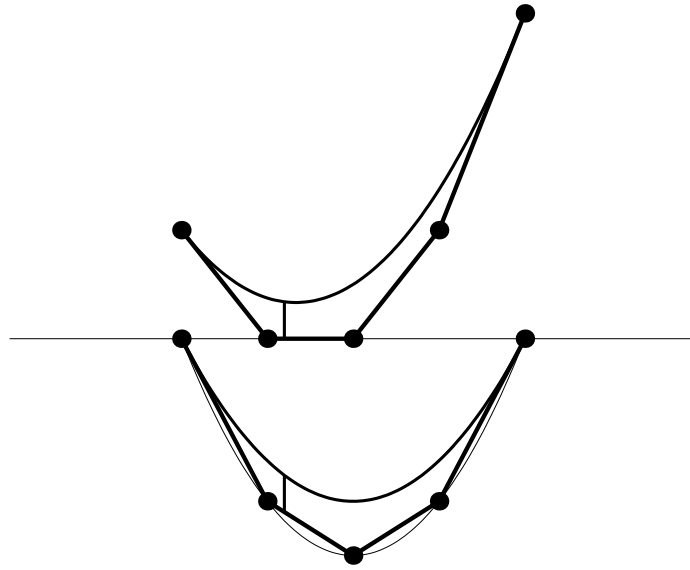
also insbesondere

$$\|p - \bar{p}\|_\infty \leq \frac{m-1}{8} \|\Delta^2 P\|_\infty.$$

Beweis Den konstruktiven Beweis wollen wir wie folgt angehen. Auf der rechten Seite der Abschätzung kommt zu gegebenem x außer Konstanten lediglich die maximale zweite Differenz der Kontrollpunkte ins Spiel. Also geben wir uns diese Größe vor und suchen Kontrollpunkte so, dass die linke Seite so groß wie möglich wird, um so die Konstante $(m-1)x(1-x)/2$ zu erhalten.

Falls $\|\Delta^2 P\|_\infty = 0$, so fällt nach Satz 2.6 das Kontrollpolygon mit der Funktion zusammen. In diesem Fall steht also sowohl auf der linken, als auch auf der rechten Seite Null.

Geben wir uns nun $\|\Delta^2 P\|_\infty = 1$ vor. Dies bedeutet keine Einschränkung, weil Skalierung von P die linke sowie die rechte Seite um denselben Faktor ändert. Wir suchen nun eine Bézierkurve p mit $\|\Delta^2 P\|_\infty = 1$, die maximalen Abstand zu ihrem Kontrollpolygon hat. Da für zwei Bézierkurven $B_m P$ und $B_m \tilde{P}$, die sich um einen linearen Term unterscheiden, sowohl $\|p - \bar{p}\|_\infty = \|\tilde{p} - \tilde{\bar{p}}\|_\infty$, als auch $\Delta^2 P = \Delta^2 \tilde{P}$ ist, kann o. B. d. A. verlangt werden, dass für ein k dann $p_k = 0 = p_{k+1}$ gilt. Nun fragen wir uns, wie der Abstand des Kontrollpolygons von der dargestellten Kurve an der Stelle $x \in [(k-1)/(m-1), k/(m-1)]$ maximiert werden kann, wenn $p_{k-1} = 0 = p_k$ und $\|\Delta^2 P\|_\infty = 1$ vorgegeben ist. Wie man zeigen kann, ist dies der Fall, wenn alle Kontrollpunkte unter Einhaltung der Vorgaben so groß wie möglich sind, was wiederum dann der Fall ist, wenn $\Delta^2 P = [1; \dots; 1]$. Nach Lemma 2.20 also genau dann, wenn bis auf einen linearen Term $p(x) = \frac{1}{2}(m-1)(m-2)x^2$. Zu diesem p gehören die Kontrollpunkte $p_k = \frac{1}{2}(k-1)(k-m)$. Betrachtet man das Kontrollpolygon zu diesen Punkten, so werden sie interpoliert durch das Polynom $\hat{p}(x) = \frac{1}{2}(m-1)^2 x(x-1)$. Dieses ist konvex, sodass das Kontrollpolygon sicher stets größer als \hat{p} ist. Es ergibt sich also $p(x) - \bar{p}(x) \leq p(x) - \hat{p}(x) = \frac{m-1}{2} x(1-x)$. \square

Abbildung 4: Zwei Bézierkurven mit $\Delta^2 P \equiv 1$

Nun können wir sehr elegant die quadratische Konvergenz des Kontrollpolygons gegen die dargestellte Kurve bei Subdivision beweisen.

Satz 2.13 Sei $p = B_m P$, $0 < a < b < 1$ und $p_s(x) = B_m P_s = p((1-x)a + xb)$ die Bézierkurve, die man durch Subdivision erhält. Dann gilt

$$\|p_s - \bar{p}_s\|_\infty \leq (b-a)^2 \frac{m-1}{8} \|\Delta^2 P\|_\infty.$$

Folglich konvergiert das Kontrollpolygon bei Subdivision quadratisch mit der Intervalllänge $b-a$ gegen die dargestellte Kurve.

Beweis Es gilt

$$\begin{aligned} & B_{m-2}(x) \Delta^2 P_s(m-1)(m-2) \\ &= D^2 p_s(x) = D^2 p((1-x)a + xb) \\ &= (b-a)^2 (D^2 p)((1-x)a + xb) = (b-a)^2 B_{m-2}((1-x)a + xb) \Delta^2 P(m-1)(m-2) \\ &= (b-a)^2 B_{m-2}(x) (\Delta^2 P)_s(m-1)(m-2), \end{aligned}$$

folglich also $\Delta^2 P_s = (b-a)^2 (\Delta^2 P)_s$. Wenn wir jetzt zeigen können, dass $\|\Delta^2 P_s\|_\infty \leq \|\Delta^2 P\|_\infty$, so folgt daraus sofort die Behauptung. Zeigen wir also noch diese Ungleichung. Anstatt die zweite Vorwärtsdifferenz von P_s zu berechnen, können wir auch $\Delta^2 P$

auf $[a, b]$ subdividieren und erhalten so ebenfalls $\Delta^2 P_s$. Bei subdividierten Kontrollpunkten kann aber der maximale Kontrollpunkt nicht größer sein, als der ursprünglich maximale Kontrollpunkt, schließlich werden bei der Durchführung des DeCasteljau-Algorithmus fortwährend Konvexkombinationen mit Gewichten aus $[0, 1]$ gebildet. \square

Die Abschätzungen aus Satz 2.12 für den Abstand einer Bézierkurve zu ihrem Kontrollpolygon können für das weitere Vorgehen in eine noch günstigere Form gebracht werden. Was wir nämlich wollen, sind zu vorgegebenen Kontrollpunkten gewissermaßen zwei neue Kontrollpunktsätze, sodass die Bézierkurve zwischen den entsprechenden Kontrollpolygone verläuft und der eingeschlossene Bereich dabei möglichst schmal ist. Wir suchen also zu Kontrollpunkten $P \in \mathbb{R}^m$ zwei Kontrollpunktsätze $U, O \in \mathbb{R}^m$, sodass $H_m U \leq B_m P \leq H_m O$.

Betrachten wir zunächst eine Bézierkurve p mit $p''(x) \equiv 1$, z.B. also $p(x) = x(x-1)/2$. Die Kontrollpunkte von p sind, wie man leicht durch Differenziation einsehen kann, $p_k = (k-1)(k-m)/(2(m-1)(m-2))$. p ist konvex, deshalb ist sicherlich

$$\begin{aligned}
p(x) &\leq \sum_{k=1}^m h_{m,k}(x) p\left(\frac{k-1}{m-1}\right) \\
&= \bar{p}(x) + \left(\sum_{k=1}^m p\left(\frac{k-1}{m-1}\right) h_{m,k}(x) - \bar{p}(x) \right) \\
&= \bar{p}(x) + \left(\sum_{k=1}^m \left(\frac{\frac{k-1}{m-1} \left(\frac{k-1}{m-1} - 1 \right)}{2} - \frac{(k-1)(k-m)}{2(m-1)(m-2)} \right) h_{m,k}(x) \right) \\
&= \bar{p}(x) + \sum_{k=1}^m \left(\frac{1}{2} \frac{(k-1)(m-k)}{(m-1)^2(m-2)} \right) h_{m,k}(x) \\
&= \bar{p}(x) + \|\Delta^2 P\|_\infty \sum_{k=1}^m \frac{1}{2} \frac{(k-1)(m-k)}{m-1} h_{m,k}(x) \\
&= \sum_{k=1}^m \left(p_k + \frac{\|\Delta^2 P\|_\infty (k-1)(m-k)}{2(m-1)} \right) h_{m,k}(x).
\end{aligned}$$

Der Übergang von p zu $ap + b$ für $a \in \mathbb{R}_{>0}$ und $b \in \mathbb{R}$ skaliert bzw. hebt die linke und rechte Seite gleichermaßen; obige Ungleichung gilt also für allgemeines p mit $p'' \equiv c > 0$. Aber sie gilt sogar für beliebiges p mit $\Delta^2 P > 0$, denn der Abstand einer Bézierkurve von seinem Kontrollpolygon ($p - \bar{p}$) in Abhängigkeit von $\|\Delta^2 P\|_\infty$ wird genau dann maximal, wenn $\Delta^2 P \equiv \|\Delta^2 P\|_\infty$, wie wir uns zu Beginn des Beweises von Satz 2.12 klar gemacht haben. Wir haben also einen oberen linearen Einschluss gefunden, der umso besser ist, umso weniger sich $(\Delta^2 P)/\|\Delta^2 P\|_\infty$ von $[1; \dots; 1]$ unterscheidet.

Wenden wir uns nun dem unteren Einschluss zu. Nehmen wir dazu wieder an, dass $p'' \equiv 1$. Wie wir oben gesehen haben, kann o.B.d.A. $p(x) = x^2/2$ gewählt werden. Gesucht ist nun eine Funktion $\sum_k h_{m,k} q_k$, die niemals größer als p ist, gleichzeitig aber möglichst groß ist, also ein guter unterer stückweise linearer Einschluss. Betrachten wir einmal nur das erste Segment $[0, 1/(m-1)]$. Wir suchen eine lineare Funktion q so, dass $\int_0^{1/(m-1)} (x^2/2 - q(x)) dx$ minimal ist, gleichzeitig aber $q(x) \leq p(x)$. In Frage kommen folglich alle Tangenten an p , also $q(x) = x_0^2/2 + x_0(x - x_0)$. Mit $x_0 = 1/(2(m-1))$ wird das Integral zwischen p und q minimiert. Da x_0 genau der Mittelpunkt ist, folgt, dass die Differenzen am Segmentanfang und Segmentende zwischen p und q gleich sind. Deshalb passen nun alle Tangenten in den Segmentmittelpunkten an den Segmentgrenzen stetig zusammen. Die Differenz von p zu den Tangenten ist an den Segmentgrenzen gerade $1/(8(m-1)^2)$. Es gilt also

$$p(x) \geq \sum_{k=1}^m h_{m,k}(x) \left(p_k + \min_k |(\Delta^2 P)_k| \left(\frac{(k-1)(m-k)}{2(m-1)} - \frac{(m-2)}{8(m-1)} \right) \right).$$

Auch diese Abschätzung bleibt für beliebige $p = B_m P$ mit $\Delta^2 P > 0$ richtig. Um dies einzusehen, betrachten wir ein $p \in \mathbb{P}_m$ auf einem Segment $S = [(l-1)/(m-1), l/(m-1)]$. O.B.d.A. sei $p_l = p_{l+1} = 0$. Sei $c := \min_k (\Delta^2 P)_k$ und $\tilde{p} = B_m \tilde{P}$ mit $\tilde{p}_l = \tilde{p}_{l+1} = 0$ und $\Delta^2 \tilde{P} \equiv c$. Es gilt dann $\tilde{p}_k \leq p_k$ und somit sicher $\tilde{p} < p$. Obige Abschätzung ist für \tilde{p} richtig, also insbesondere auch für p . Weil bei Skalierung mit -1 obere Einschlüsse zu unteren werden, können wir zusammenfassen:

Satz 2.14 *Sei $p = B_m P$. Falls $\min_k (\Delta^2 P)_k \geq 0$, dann gilt*

$$\begin{aligned} p(x) &\leq \sum_{k=1}^m h_{m,k}(x) \left(p_k + \max_k (\Delta^2 P)_k \frac{(k-1)(m-k)}{2(m-1)} \right) \\ p(x) &\geq \sum_{k=1}^m h_{m,k}(x) \left(p_k + \min_k (\Delta^2 P)_k \left(\frac{(k-1)(m-k)}{2(m-1)} - \frac{(m-2)}{8(m-1)} \right) \right). \end{aligned}$$

Falls $\max_k (\Delta^2 P)_k \leq 0$, dann gilt

$$\begin{aligned} p(x) &\geq \sum_{k=1}^m h_{m,k}(x) \left(p_k + \min_k (\Delta^2 P)_k \frac{(k-1)(m-k)}{2(m-1)} \right) \\ p(x) &\leq \sum_{k=1}^m h_{m,k}(x) \left(p_k + \max_k (\Delta^2 P)_k \left(\frac{(k-1)(m-k)}{2(m-1)} - \frac{(m-2)}{8(m-1)} \right) \right). \end{aligned}$$

Bei quadratischen Polynomen p ist die jeweils obere Ungleichung für $x = k/(m-1)$, bzw. die untere Ungleichung für $x = k/(m-1) + 1/(2(m-1))$ scharf.

Wenn es zweite Differenzen mit verschiedenen Vorzeichen gibt, dann gilt:

$$p(x) \leq \sum_{k=1}^m h_{m,k}(x) \left(p_k + \max_k(\Delta^2 P)_k \frac{(k-1)(m-k)}{2(m-1)} \right)$$

$$p(x) \geq \sum_{k=1}^m h_{m,k}(x) \left(p_k + \min_k(\Delta^2 P)_k \frac{(k-1)(m-k)}{2(m-1)} \right).$$

Beispiel 2.15 Betrachten wir in Abbildung 5 einige Beispiele. Das Kontrollpolygon ist gestrichelt. Die Polynome sind gegeben durch die Kontrollpunkte: (1) $[0, 1, 1, 0]$, (2) $[0, -13, -20, -20, -140]$, (3) $[0, 1, 2, 3, 2, 1]$, (4) $[0, 1, -1, 0]$, (5) $[0, -7, 2, -5, 4]$, (6) $[0, 0, 0, 0, 0, 0, 1]$. Als $\Delta^2 P$ ergibt sich: (1) $[-1, -1]$, (2) $[6, 7, 6, 8]$, (3) $[0, 0, -2, 0]$, (4) $[-3, 3]$, (5) $[16, -16, 16]$, (6) $[0, 0, 0, 0, 1]$ und damit die Einschlüsse (1) $[0, 2/3, 2/3, 0; 1/12, 3/4, 3/4, 1/12]$, (2) $[0, 1.4, 3.6, 6.6, 10.4, 14; 0, 1, 3, 6, 10, 14]$, (3) $[0, 1/5, 4/5, 9/5, 6/5, 1; 0, 1, 2, 3, 2, 1]$, (4) $[0, 2, 0, 0; 0, 0, -2, 0]$, (5) $[0, -1, 10, 1, 4; 0, -13, -6, -11, 4]$, (6) $[0, 5/18, 6/9, 3/4, 6/9, 5/18, 1; 0, 0, 0, 0, 0, 0, 1]$. Zu beachten ist, dass im 3. Beispiel der

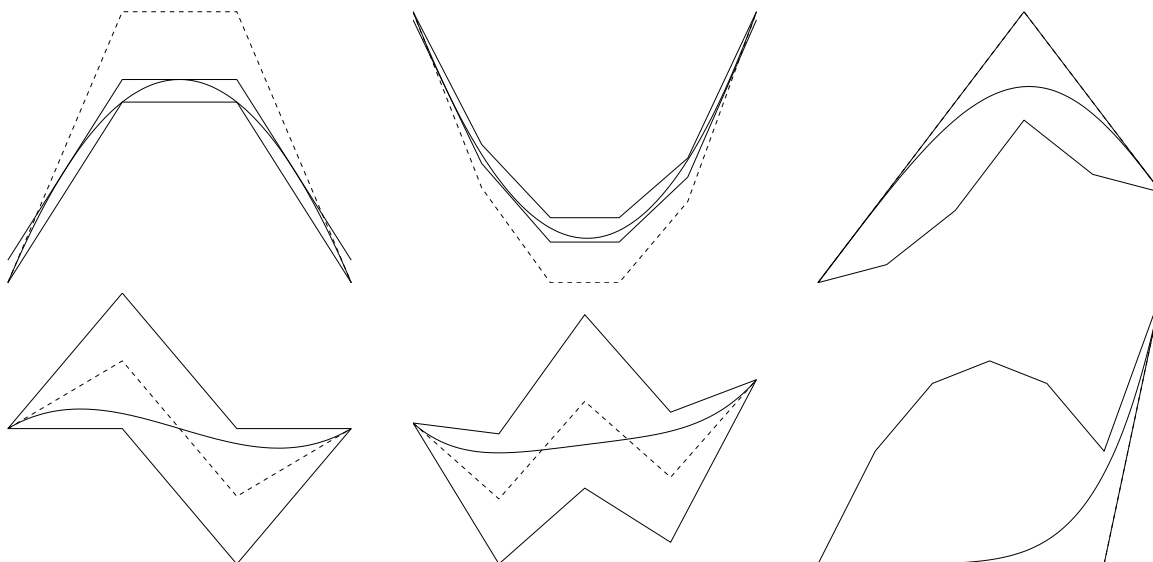


Abbildung 5: Bézierkurven und ihre linearen Einschlüsse nach Satz 2.14. Die zugehörigen Kontrollpolygone sind gestrichelt.

obere Einschluss mit dem Kontrollpolygon zusammenfällt. Ebenso verhält es sich im 6. Beispiel mit dem unteren Einschluss. Die linearen Einschlüsse sind in den ersten drei Fällen wesentlich besser, in den anderen Fällen wesentlich schlechter als die konvexe Hülle. Eine Faustregel ist die folgende: „Wenn alle zweiten Differenzen einerlei Vorzeichen besitzen, dann liefert Satz 2.14 gute Einschlüsse. Falls sowohl positive als auch negative zweite Differenzen auftreten, so wird die konvexe Hülle mit Satz 2.14 in der Regel nicht verbessert.“

Die Einschlussschätzer aus Satz 2.14 arbeiten ausschließlich mit der Größe $\max_k (\Delta^2 P)_k$ für den oberen bzw. mit $\min_k (\Delta^2 P)_k$ für den unteren Einschluss. Nach der Berechnung von $\Delta^2 P$ bleiben also alle Zahlen von $\Delta^2 P$ bis auf eine unberücksichtigt, was natürlich das Maß der eingehenden Information stark verringert. Mathematisch gesprochen schalten wir zwischen den Einschlussschätzer a und den Differenzen-Berechner $d := \Delta^2 P$ noch eine Abbildung $m : \mathbb{R}^{m-2} \rightarrow \mathbb{R}$, sodass dann $a \circ m \circ d$ uns insgesamt einen oberen bzw. unteren Einschluss liefert. Nachdem wir uns dies einmal klar gemacht haben, ist die Idee von Peters/Lutterkort in [LP01] naheliegend, alle Einträge auszunutzen, die Δ^2 liefert. Die in [LP01] verfolgte Idee macht sich folgendes zunutze. Sei U eine Funktion, die Kontrollpunkten P neue Kontrollpunkte $U(P)$ zuordnet, sodass $H_m U(P)$ ein unterer Einschluss für $B_m P$ ist. Dann ist für $a, b \geq 0$ und zwei Kontrollpunktsätze P_1, P_2 sicherlich $H_m(aU(P_1) + bU(P_2))$ ein unterer Einschluss für $B_m(aP_1 + bP_2)$. Außerdem ist $-H_m U(P)$ ein oberer Einschluss von $-B_m P$. Wenn nun für eine Basis von \mathbb{R}^m jeweils untere und obere Einschlässe bestimmt werden, so können diese dazu verwendet werden, untere und obere Einschlässe für beliebige Kontrollpunktsätze linear zu kombinieren. Denkt man in dieser Richtung kurz weiter, so wird klar, dass man, anstatt die Kontrollpunkte direkt zu betrachten, wieder auf $\Delta^2 P$ und eine Basis von \mathbb{R}^{m-2} zurückgreifen muss. Der folgende Satz ist nun eine Zusammenfassung des hier Gesagten.

Satz 2.16 *Eine Bézierkurve $B_m P$ ist durch $\Delta^2 P \in \mathbb{R}^{(m-2) \times 1}$ bis auf einen linearen Term festgelegt. Dieser Term besitzt auf den Abstand der Bézierkurve zu ihrem Kontrollpolygon keinen Einfluss. Seien nun zu Kontrollpunktvektoren $P_j \in \mathbb{R}^m$ ($j \in 1 : m - 2$) mit $P_j(1) = P_j(m) = 0$ und $(\Delta^2 P_j)(k) = \delta_{j,k}$ Einschlässe $L_j, U_j \in \mathbb{R}^m$ gegeben, sodass*

$$H_m(P_j + L_j) \leq B_m P_j \leq H_m(P_j + U_j).$$

Fassen wir nun die L_j, U_j , $j \in (1 : m - 2)$, in jeweils einer Matrix $L, U \in \mathbb{R}^{m \times (m-2)}$ zusammen. Setzen wir also $L(k, j) = L_j(k)$ und $U(k, j) = U_j(k)$. Dann gilt mit

$$\Delta^2 P_+(k) := \left\{ \begin{array}{ll} \Delta^2 P(k) & , \text{ falls } \geq 0 \\ 0 & , \text{ sonst} \end{array} \right\}, \quad \Delta^2 P_-(k) := \left\{ \begin{array}{ll} -\Delta^2 P(k) & , \text{ falls } \leq 0 \\ 0 & , \text{ sonst} \end{array} \right.$$

schließlich

$$H_m(P + L\Delta^2 P_+ - U\Delta^2 P_-) \leq B_m P \leq H_m(P + U\Delta^2 P_+ - L\Delta^2 P_-).$$

Zu beachten ist, dass die P_j durch die Forderungen $P_j(1) = P_j(m) = 0$ und $(\Delta^2 P_j)(k) = \delta_{j,k}$ eindeutig bestimmt sind.

In [LP01] wird angegeben, wie solche Vektoren L_j, U_j berechnet werden können. Sinnvollerweise sollte der durch L_j und U_j eingeschlossene Bereich möglichst kleinen Flächeninhalt besitzen und P_j enthalten. Dies läuft auf ein lineares Programm hinaus,

welches für jeden Vektor P_j gemäß Satz 2.16 gelöst werden muss. Der Aufwand hierfür ist relativ hoch, jedoch muss für eine vorgegebene Ordnung m bzw. Index j dieses lineare Programm genau einmal gelöst werden um dann in einer Tabelle protokolliert zu werden. Für die Einschlüsse, die wir anschließend gemäß Satz 2.16 berechnen, kann dann auf diese Tabellen zurückgegriffen werden. So können zu vorgegebenen Kontrollpunkten äußerst effizient gute Einschlüsse berechnet werden.

Beispiel 2.17 *Betrachten wir nun in Abbildung 6 dieselben Kontrollpunktsätze wie in Beispiel 2.15. Wie wir sehen, sind die Einschlüsse außer im ersten Beispiel durchweg wesentlich besser. Die Verschlechterung im ersten Beispiel ist jedoch minimal, hier ist der obere Einschluss für $t = 1/2$ nicht mehr tangential. Und gerade dieses Beispiel ist dasjenige, bei dem der erste Schätzer optimal ist. Wir sehen also, dass der erste Schätzer in seinem optimalen Fall nur geringfügig besser ist als der zweite. Der zweite Schätzer ist dem ersten jedoch in seinem optimalen Fall klar voraus. Wir kommen insgesamt also zum Schluss, dass der zweite Einschluss schätzer aus Satz 2.16 dem ersten aus Satz 2.14 für unsere Zwecke klar vorzuziehen ist. Der erste Schätzer hingegen hat gegenüber dem zweiten den deutlichen Vorteil, dass alle Größen klar quantifiziert sind und somit für die Theorie einfach zu überschauen sind. Außerdem lässt sich der erste Schätzer leicht übertragen auf Splines mit beliebigen Knotenfolgen (siehe dazu [Reif00]). Für den zweiten Schätzer müsste hierfür zu jeder neuen Knotenfolge ein lineares Programm zur Berechnung der Tabellen gelöst werden, was die Anwendbarkeit sehr stark einschränkt.*

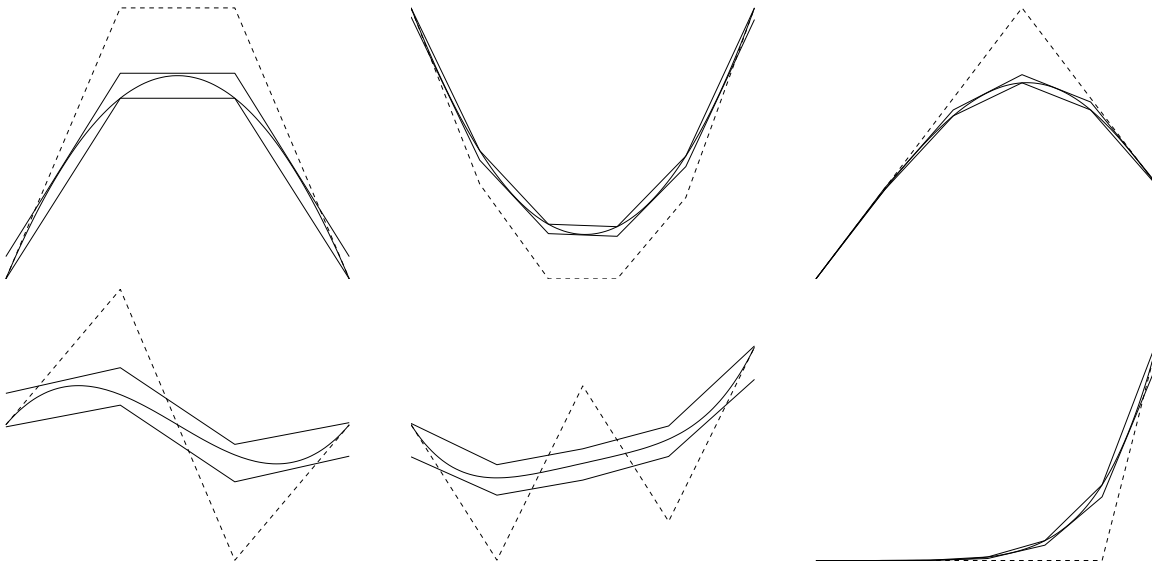


Abbildung 6: Bézierkurven und ihre linearen Einschlüsse nach Satz 2.14

Nun haben wir im Wesentlichen alle Hilfsmittel für den univariaten Fall $n = 1$ beisammen. Natürlich sind wir nicht damit zufrieden polynomiale Gleichungen zu lösen, gibt es doch schon eine Vielzahl von effizienten Algorithmen dafür. Eine sehr schöne Übersicht darüber findet sich in [Spe]. Die Stärke unserer Algorithmen wird sein, dass alle Nullstellen eines polynomialen Gleichungssystems effizient bestimmt werden können. Dazu brauchen wir noch die Theorie für den multivariaten Fall.

2.2 Multivariate polynomiale Béziertheorie

Definition 2.18 Sei $M = (m_1, \dots, m_n)$ ein Vektor von natürlichen Zahlen und $I = (i_1, \dots, i_n)$ mit $i_k \in 1 : m_k$. Wir nennen

$$b_{M,I}(x_1, \dots, x_n) := b_{m_1, i_1}(x_1) \cdots b_{m_n, i_n}(x_n) \quad , x_i \in [0, 1]$$

multivariates Bernsteinpolynom bzw.

$$h_{M,I}(x_1, \dots, x_n) := h_{m_1, i_1}(x_1) \cdots h_{m_n, i_n}(x_n)$$

multivariate Hutfunktion der Ordnung M . Die rechte Seite von

$$p = \sum_I b_{M,I} p_I$$

für $p_I \in \mathbb{R}^d$ heißt multivariate Bézierdarstellung von p bzw. Bézierpolynom. Die Funktion

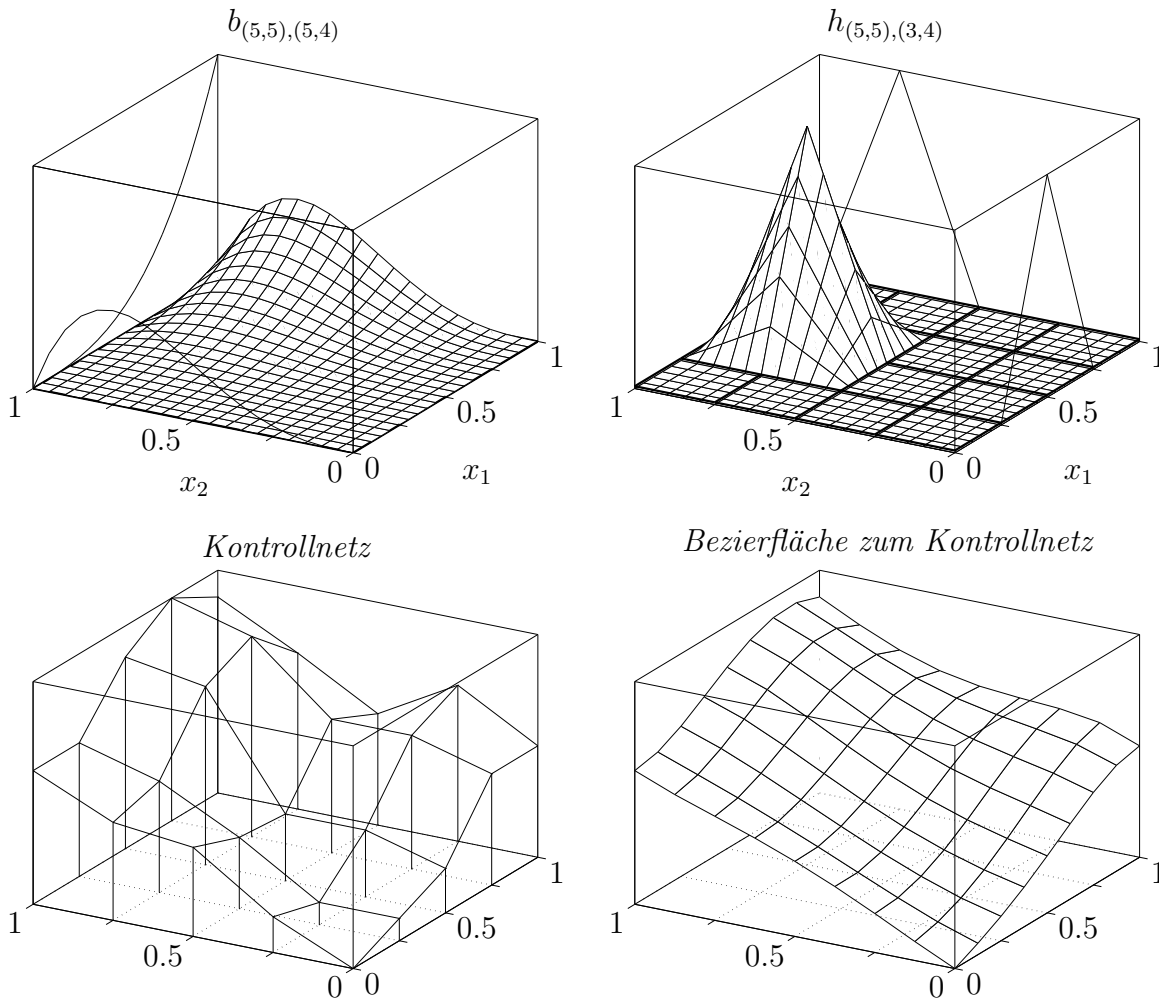
$$\bar{p} = \sum_I h_{M,I} p_I$$

nennen wir Kontrollnetz zu p . Die Kontrollpunkte p_I fassen wir in der multidimensionalen Matrix $P_{i_1, \dots, i_n} := p_I$ zusammen. Der Operator Δ_k sei der Vorwärtsdifferenzenoperator in der k -ten Dimension welchen wir wie folgt definieren:

$$(\Delta_k P)_{i_1, \dots, i_n} := p_{i_1, \dots, i_{k-1}, i_k+1, i_{k+1}, \dots, i_n} - p_{i_1, \dots, i_n}$$

$(\Delta_k P)$ ist also eine $(m_1, \dots, m_k - 1, \dots, m_n)$ -Matrix.

Beispiel 2.19 Für $n = 2$ können wir Bézierpolynome bzw. Kontrollnetze plotten.



Aufgrund der Tensorproduktstruktur der multivariaten Bernsteinfunktionen bzw. der Hutfunktionen lassen sich nun die bisherigen Ergebnisse leicht auf den multivariaten Fall verallgemeinern.

Lemma 2.20 *Die partielle Ableitung eines Bézierpolynom mit den Kontrollpunkten P nach der k -ten Variablen ist ein Bézierpolynom mit den Kontrollpunkten $(m_k - 1)\Delta_k P$.*

Beweis Wir zeigen die Behauptung für die Ableitung nach x_1 . Sei $M' = (m_1 - 1, m_2, \dots, m_n)$. Dann gilt

$$\begin{aligned} & D_1 \sum_I b_{M,I}(x_1, \dots, x_n) p_I \\ &= D_1 \sum_{i_1=1}^{m_1} b_{m_1, i_1}(x_1) \left(\sum_{i_2=1}^{m_2} \cdots \sum_{i_n=1}^{m_n} b_{m_2, i_2}(x_2) \cdots b_{m_n, i_n}(x_n) p_{M, (i_1, \dots, i_n)} \right) \\ &= \sum_I b_{M', I}(x_1, \dots, x_n) (m_1 - 1) (\Delta_1 P)_I. \end{aligned}$$

□

Satz 2.21 Es gilt $b_{M,I} \geq 0$ und $\sum_I b_{M,I} \equiv 1$. Damit gilt für beliebige Kontrollpunkte $p_I \in \mathbb{R}^d$

$$\sum_I b_{M,I}(x) p_I \in \text{ConvexHull}(p_I) \quad , x \in [0, 1]^n .$$

Beweis Der Beweis der Positivität folgt direkt aus der Definition. Die Partition der Eins ist unmittelbar klar mit Satz 2.4 wegen

$$\sum_I b_{M,I}(x) = \underbrace{\sum_{i_1=1}^{m_1} b_{m_1, i_1}(x_1)}_{=1} \cdots \underbrace{\sum_{i_{n-1}=1}^{m_{n-1}} b_{m_{n-1}, i_{n-1}}(x_{n-1})}_{=1} \underbrace{\sum_{i_n=1}^{m_n} b_{m_n, i_n}(x_n)}_{=1} .$$

□

Satz 2.22 Mit $p_I = (\frac{i_1-1}{m_1-1}, \dots, \frac{i_n-1}{m_n-1})$ ist $p(x_1, \dots, x_n) = (x_1, \dots, x_n)$. D.h. für $q_I \in \mathbb{R}^d$ ist der Graph von q , also $(x, q(x))$, durch $\sum_I b_{M_I}(x) (\frac{i_1-1}{m_1-1}, \dots, \frac{i_n-1}{m_n-1}, q_I)$ gegeben. Außerdem sind Bézierpolynome affin invariant.

Beweis Der Beweis für die erste Behauptung ergibt sich wieder durch richtige Klammerung mit Satz 2.6 aus

$$\pi_1 \left(\sum_I b_{M,I} p_I(x) \right) = \sum_{i_1=1}^{m_1} b_{m_1, i_1}(x_1) \frac{i_1 - 1}{m_1 - 1} \underbrace{\left(\sum_{i_2=1}^{m_2} \cdots \sum_{i_n=1}^{m_n} b_{m_2, i_2}(x_2) \cdots b_{m_n, i_n}(x_n) \right)}_{\equiv 1} = x_1 .$$

□

Um uns im Folgenden bequem ausdrücken zu können, benötigen wir die folgende Definition.

Definition 2.23 Sei die Kontrollpunktmatrix P wie in Definition 2.18 gegeben. Wir setzen dann für feste $I = (i_1, \dots, i_n)$ und $k \in (1 : n)$ die Spalte

$$P_{I|k} := [p_{i_1, \dots, i_{k-1}, 1, i_{k+1}, \dots, i_n}; \dots; p_{i_1, \dots, i_{k-1}, m_k, i_{k+1}, \dots, i_n}].$$

Ebenso sei mit $P_{I,K}$ die entsprechende Teilmatrix gemeint, wenn K ein Indexvektor ist.

Satz 2.24 Gegeben seien die Kontrollpunkte P zum Bézierpolynom p und eine Box $[a_1, b_1] \times \dots \times [a_n, b_n] \subset [0, 1]^n$. Dann gibt es Kontrollpunkte \tilde{P} zum Bézierpolynom $\tilde{p}(x_1, \dots, x_n) = p((1-x_1)a_1 + x_1b_1, \dots, (1-x_n)a_n + x_nb_n)$. Diese neuen Kontrollpunkte \tilde{P} berechnen sich wie folgt. Der Operator s sei der aus Definition 2.8.

$$\forall I, k : \begin{aligned} P^{[0]} &:= P \\ P_{I|k}^{[k]} &:= s(P_{I|k}^{[k-1]}, a_k, b_k) \\ \tilde{P} &:= P^{[n]} \end{aligned}$$

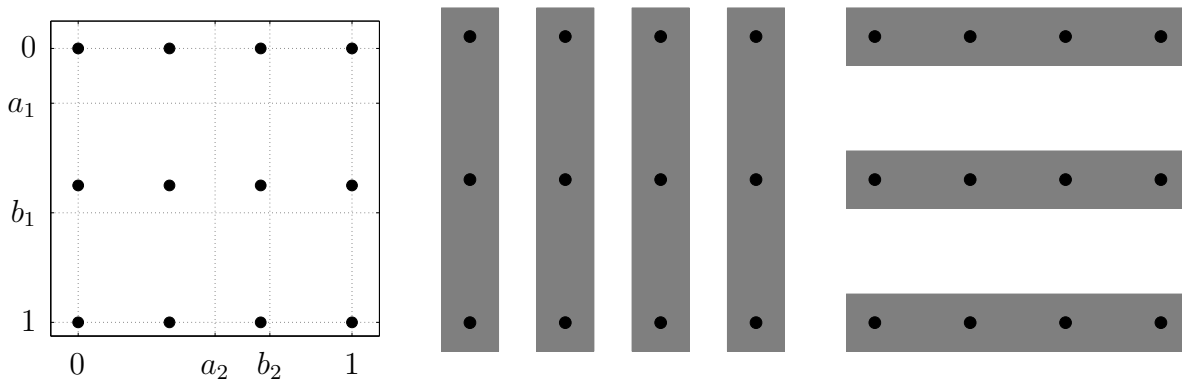


Abbildung 7: Illustration der Subdivision im Mehrdimensionalen

Die Aussage des letzten Satzes lautet kurz zusammengefasst folgendermaßen: Subdivision im multivariaten Fall ist nichts anderes als iterierte parallele univariate Subdivision. Abbildung 7 illustriert den bivariaten Fall für die Ordnung $M = (3, 4)$. Angenommen, wir wollen von $[0, 1]^2$ auf $[a_1, b_1] \times [a_2, b_2]$ subdividieren. Dazu fassen wir im ersten Schritt die 12 Kontrollpunkte wie im mittleren Bild auf als vier Vektoren von je drei Kontrollpunkten. Jeden dieser vier Vektoren subdividieren wir nach den Regeln

der univariaten Subdivision von $[0, 1]$ auf $[a_1, b_1]$ (siehe Satz 2.7 und Lemma 2.9). So erhalten wir vier neue Vektoren mit je drei Kontrollpunkten. Diese werden jeweils wieder „so abgelegt, wie vor der Subdivision aufgenommen“. Im zweiten Schritt fassen wir diese neuen 12 Kontrollpunkte nun wie im rechten Bild auf als drei Vektoren von je vier Kontrollpunkten. Jeden dieser drei Vektoren subdividieren wir nun nach den Regeln der univariaten Subdivision von $[0, 1]$ auf $[a_2, b_2]$. Wir erhalten also wieder drei neue Vektoren mit je vier Kontrollpunkten. Diese neuen zwölf Kontrollpunkte sind nun die gesuchten für den Bereich $[a_1, b_1] \times [a_2, b_2]$.

Auch der Beweis dieses Satzes besteht lediglich in der richtigen Klammerung. Anstatt ihn anzugeben, geben wir lieber ein kurzes Beispiel.

Beispiel 2.25 Seien die Kontrollpunkte P gegeben und die Kontrollpunkte zur Box $[1/3, 1/2] \times [1/3, 1/2]$ gesucht. Als neue Kontrollpunkte ergeben sich aus P nach Subdivision dann \tilde{P} :

$$P = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 8 & -8 & 8 & 16 \\ 16 & -16 & 16 & 32 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \tilde{P} = \begin{bmatrix} \frac{256}{243} & \frac{32}{27} & \frac{16}{9} & \frac{8}{3} \\ \frac{32}{27} & \frac{4}{3} & 2 & 3 \\ \frac{104}{81} & \frac{13}{9} & \frac{13}{6} & \frac{13}{4} \\ \frac{4}{3} & \frac{3}{2} & \frac{9}{4} & \frac{27}{8} \end{bmatrix}$$

Bezüglich der Einschlusssschätzer geben wir eine Möglichkeit an, diese vom univariaten auf den multivariaten Fall zu verallgemeinern. Das Vorgehen ähnelt sehr stark dem der multivariaten Subdivision (Satz 2.24).

Satz 2.26 Sei das Bézierpolynom $p : \mathbb{R}^n \rightarrow \mathbb{R}$ mit den Kontrollpunkten P gegeben. Sei a ein Funktional, das (für jeweils entsprechendes m) Bézierpunkten P Einschluss-Kontrollpunkte U zuordnet, sodass $B_m P \geq H_m U$. Sei

$$\begin{aligned} U^{[0]} &:= P \\ U_{I|k}^{[k]} &:= a(U_{I|k}^{[k-1]}) \quad , \forall I \quad , k \in (1 : n) \\ U &:= U^{[n]} \end{aligned}$$

Dann gilt

$$p(x_1, \dots, x_n) \geq \sum_I h_{M,I} U_I.$$

Entsprechendes gilt natürlich auch, wenn alle \geq durch \leq und a gleichzeitig durch ein Funktional für den oberen Einschluss ersetzt wird. Insbesondere ist durch Satz 2.14 und durch Satz 2.16 ein solches Funktional gegeben.

Beweis Der Beweis ist denkbar einfach, wieder muss nur richtig geklammert werden.

$$\begin{aligned}
& \sum_I b_{M,I}(x_1, \dots, x_n) p_I \\
&= \sum_{i_2=1}^{m_2} b_{m_2, i_2}(x_2) \cdots \sum_{i_n=1}^{m_n} b_{m_n, i_n}(x_n) \sum_{i_1=1}^{m_1} b_{m_1, i_1}(x_1) U_I^{[0]} \\
&\geq \sum_{i_2=1}^{m_2} b_{m_2, i_2}(x_2) \cdots \sum_{i_n=1}^{m_n} b_{m_n, i_n}(x_n) \sum_{i_1=1}^{m_1} h_{m_1, i_1}(x_1) U_I^{[1]} \\
&= \sum_{i_1=1}^{m_1} h_{m_1, i_1}(x_1) \sum_{i_3=1}^{m_3} b_{m_3, i_3}(x_3) \cdots \sum_{i_n=1}^{m_n} b_{m_n, i_n}(x_n) \sum_{i_2=1}^{m_2} b_{m_2, i_2}(x_2) U_I^{[1]} \\
&\dots \\
&\geq \sum_I h_{M,I}(x_1, \dots, x_n) U_I^{[n]} = \sum_I h_{M,I}(x_1, \dots, x_n) U_I
\end{aligned}$$

□

Die Idee des vorigen Satzes bestand darin, dass die Auswertung eines multivariaten Bézierpolynoms wie folgt aufgefasst werden kann. Angenommen, n ist gleich 2, und wir wollen an der Stelle (x_1, x_2) auswerten. Wir werten dann zunächst an der Stelle x_1 die m_2 Bézierkurven mit den Kontrollpunkten $P_{I|1}$ aus. Das ergibt m_2 Ergebnisse. Diese fassen wir als Kontrollpunkte einer Bézierkurve auf, die an der Stelle x_2 auszuwerten ist.

Satz 2.26 sagt uns, wie wir Einschlusschätzer für den multivariaten Fall aus univariaten Formeln gewinnen können. Eine ganz andere Möglichkeit, multivariate Einschlusschätzer zu gewinnen, wird in [LP01] besprochen. Dort wird beobachtet, dass der Einschlusschätzer aus Satz 2.16 wie folgt aufgebaut ist. Zunächst wird auf vorgegebene Kontrollpunkte der zweite Vorwärtsdifferenzen-Operator angewendet. Dieser bildet alle linearen Polynome in den Kern ab. Auf dem Raum der Polynome modulo von linearen Termen ist er bijektiv. Nun kann für multivariate Polynome ebenso ein „zweiter-Vorwärtsdifferenzen-Operator“ mit den selben Eigenschaften definiert werden. Dann werden zu Urbildern von „Einheitsvektoren“ wieder Einschluss-Kontrollnetze berechnet und in Tabellen abgelegt. Nun können zu konkret vorgelegten Kontrollpunkten wieder wie in Satz 2.16 Einschlüsse berechnet werden.

Diesen Vorschlag verwerfen wir aber, weil schon für die Behandlung von bivariaten Bézierpolynomen die Bereitstellung von sehr vielen Tabellen erforderlich ist. Für Probleme mit mehr als zwei Veränderlichen, ist dieser Ansatz im Grunde nicht mehr praktikabel.

Bevor wir ein Beispiel geben, wie der Einschlusschätzer arbeitet, möchten wir noch kurz erläutern, wie denn eine polynomiale Funktion in Bézierdarstellung konvertiert

werden kann. Die vorgestellte Methode besitzt dabei den Vorzug, dass sie sehr einfach ist, gleichzeitig aber den Nachteil, dass sie für große Ordnungen nicht mehr numerisch stabil ist. Wir weisen darauf hin, dass die Transformation in Bézierdarstellung je nach Vorgaben stabiler durchgeführt werden kann. So existieren z.B. für die Monomform rationale Umwandlungsmatrizen. Allen multivariaten Transformationen gemeinsam ist, dass sie univariat iteriert durchgeführt werden können.

Wir gehen aus von einer auswertbaren Funktion, von der wir wissen, dass sie polynomial ist und deren Ordnung wir kennen. Nun suchen wir die Kontrollpunkte, die diese Funktion darstellen.

Die Idee besteht darin, parallele iterierte univariate Interpolation zu betreiben. Im univariaten Fall können wir wie folgt vorgehen. Sei $f \in \mathbb{P}_m$ gegeben. Wir wählen für ein $l \geq m$ und für $i \in (1 : l)$ die Stützstellen $t_i = (i - 1)/(m - 1)$. Nun werten wir

$$b_{m,1}, \dots, b_{m,m}, f$$

an t_1, \dots, t_l aus und lösen das lineare (im Fall $l > m$ das überbestimmte) Gleichungssystem

$$\sum_{j=1}^m b_{m,j}(t_i)\alpha_j = f(t_i), \quad i \in 1 : l$$

nach $\alpha_1, \dots, \alpha_m$.

Betrachten wir nun den multivariaten Fall. Gegeben sei die Funktion

$$F : [0, 1]^n \longrightarrow \mathbb{R}$$

der Ordnung (m_1, \dots, m_n) . Gesucht sind die Koeffizienten für die entsprechenden multivariaten Bernsteinpolynome, sodass F dargestellt wird. Die erste Idee, die man in Anlehnung an den univariaten Fall hat, ist die folgende. Man wählt für $j \in 1 : n$ und $l_j \geq m_j$

$$t_i^j = \frac{i - 1}{m_j - 1}, \quad i \in 1 : l_j$$

und setzt $T_j := \{t_1^j, \dots, t_{l_j}^j\}$ und wertet alle multivariaten Bernsteinpolynome sowie die Funktion F an allen

$$(t_1, \dots, t_n) \in T_1 \times \dots \times T_n$$

aus. Somit erhält man wieder ein (ggf. überbestimmtes) lineares Gleichungssystem und löst dieses. Der entscheidende Nachteil aber ist, dass das Gleichungssystem unnötig groß und schlecht konditioniert ist. Geschickter ist es, die spezielle Tensorprodukt-Struktur auszunutzen. Seien die t_i^j wie oben. Wir setzen

$$k_0(i_1, \dots, i_n) := f(t_{i_1}^1, \dots, t_{i_n}^n)$$

und verfahren nun wie folgt. Sei zunächst $j = 1$. Wir bestimmen für alle $(i_2, \dots, i_n) \in T_2 \times \dots \times T_n$ Koeffizienten $k_j(1, i_2, \dots, i_n), k_j(2, i_2, \dots, i_n), \dots, k_j(m_j, i_2, \dots, i_n)$ so, dass

$$\sum_{i=1}^{m_j} k_j(i, i_2, \dots, i_n) b_{m_j, i}(t_r^j) = k_{j-1}(i_r, i_2, \dots, i_n), \quad r \in 1 : l_j.$$

Sukzessive setzen wir dieses Verfahren für $j = 2, \dots, n$ fort. Die $k_n(i_1, i_2, \dots, i_n)$ sind dann die Kontrollpunkte für die Bézierpolynome. Dies ist aufgrund der Tensorprodukt-Struktur unmittelbar einsehbar.

Beispiel 2.27 *Illustrieren wir nun Satz 2.26. Seien die Kontrollpunkte P gegeben. Wir fassen zunächst diese 12 Kontrollpunkte wie im mittleren Bild von Abbildung 7 auf als vier Vektoren von je drei Kontrollpunkten. Für jeden dieser Vektoren finden wir mit Satz 2.14 und Satz 2.26 einen unteren und oberen Einschluss, der gegeben wird durch die Kontrollpunkte U_1 bzw. O_1 . Nun fassen wir jeweils diese 12 Kontrollpunkte auf wie im rechten Bild von Abbildung 7 und finden zu U_1 den unteren Einschluss U bzw. zu O_1 den oberen Einschluss O (siehe auch Abbildung 8 links).*

$$P = \begin{bmatrix} 192 & 0 & 0 & 384 \\ -192 & -576 & -768 & 0 \\ 192 & 192 & 192 & -192 \end{bmatrix}$$

$$U_1 = \begin{bmatrix} 144 & -84 & -108 & 372 \\ -48 & -324 & -444 & 36 \\ 144 & 108 & 84 & -204 \end{bmatrix} \quad O_1 = \begin{bmatrix} 192 & 0 & 0 & 384 \\ 0 & -240 & -336 & 48 \\ 192 & 192 & 192 & -192 \end{bmatrix}$$

$$U = \begin{bmatrix} 127 & -33 & -57 & 355 \\ -61 & -285 & -405 & 23 \\ 144 & 20 & -4 & -204 \end{bmatrix} \quad O = \begin{bmatrix} 192 & 128 & 128 & 384 \\ 0 & -80 & -176 & 48 \\ 192 & 192 & 192 & -192 \end{bmatrix}$$

Benutzen wir zum Einschluss der univariaten Kontrollpunktsätze in Verbindung mit Satz 2.26 nicht wie oben Satz 2.14, sondern Satz 2.16, so erhalten wir als Einschluss

$$U = \begin{bmatrix} 123.7 & -34.7 & 6.9 & 331.2 \\ -65.1 & -284.6 & -307.5 & -12.0 \\ 141.1 & 97.9 & 3.0 & -204.5 \end{bmatrix} \quad O = \begin{bmatrix} 192.0 & 71.1 & 120.9 & 384.0 \\ 0 & -179.6 & -188.4 & 48.0 \\ 194.0 & 187.7 & 103.7 & -162.3 \end{bmatrix}.$$

Diese Einschlüsse erweisen sich wie erwartet als wesentlich besser (siehe auch Abbildung 8 (rechts)).

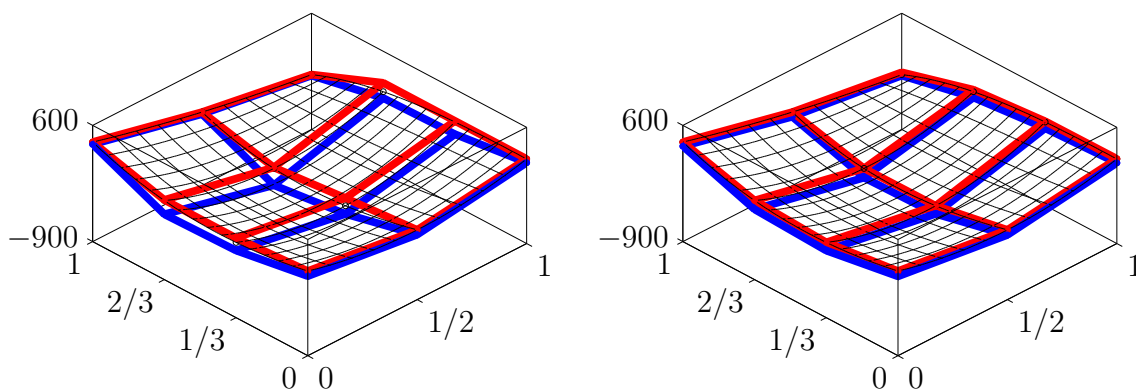
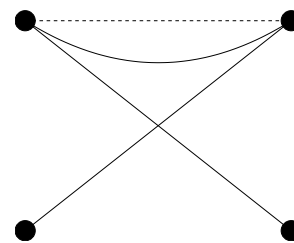


Abbildung 8: Einschlüsse von Bézierpolynomen mit Satz 2.14 sowie Satz 2.26 (links) bzw. mit Satz 2.16 sowie Satz 2.26 (rechts).

Sehr naheliegend, aber falsch ist die folgende Idee: „Wir berechnen mit zwei verschiedenen Verfahren zwei obere stückweise lineare Einschlüsse und bilden Kontrollpunktweise das Minimum, entsprechend verfahren wir mit den unteren Einschlüssen und bilden das Maximum.“ Das Bild rechts zeigt, warum diese Idee schief geht. Das kontrollpunktweise Bilden des Maximums von unteren Einschlüssen führt hier sogar zu einem oberen Einschluss. Möglich ist hingegen, das kontrollpunktweise Mittel von zwei Einschlüssen zu bilden.



Nun haben wir im Wesentlichen alle notwendige polynomiale Béziertheorie beisammen, wir können nun effiziente Algorithmen zur Lösung von polynomialen Gleichungssystemen formulieren.

3 Formulierung von effizienten Algorithmen

Wir können nun Algorithmen zur Auffindung aller Nullstellen eines polynomialen Gleichungssystems $p : \mathbb{R}^n \supset B \rightarrow \mathbb{R}^n$ mit kompaktem B formulieren. Wir haben im letzten Kapitel die dafür notwendige Theorie besprochen. Wir schlagen nun zwei Strategien vor, die sich dadurch unterscheiden, dass einmal mit den Einschlussschätzern gearbeitet wird (siehe Abschnitt 3.2) und einmal nicht (siehe Abschnitt 3.1 und 3.1.2). Wir werden in Abschnitt 3.1 anhand eines ausführlichen Beispiels die Grundzüge unseres Algorithmus vorstellen und dann die Idee des Programms in mehrere Einzelteile zerlegen und in Sätzen formulieren. Anschließend setzen wir dann die Bausteine zusammen und geben den Algorithmus an. Dieser wird als Output eine Menge von Intervallen zurückliefern, in denen Nullstellen möglich sind. Um sicherzustellen, dass in jedem dieser Intervalle sich tatsächlich genau eine Nullstelle befindet, genügt es in der Praxis, zu verlangen, dass diese Intervalle hinreichend kleinen Durchmesser besitzen. Mathematisch ist dies natürlich nicht befriedigend, deshalb gehen wir die Frage nach Existenz und Eindeutigkeit von Nullstellen in den zurück gelieferten Intervallen und die Konvergenzgeschwindigkeit des Algorithmus in Abschnitt 3.1.1 an. Es wird sich herausstellen, dass die Konvergenzrate quadratisch in allen Dimensionen gleichzeitig ist. In Abschnitt 3.1.2 besprechen wir die Behandlung eines Spezialfalls, der insbesondere für höherdimensionale Probleme entscheidende Reduktionen im Speicherplatzbedarf und der Rechenzeit erlaubt. Dieser Spezialfall dürfte im höherdimensionalen Fall eher die Regel als die Ausnahme sein. In Abschnitt 3.2 beziehen wir dann die Einschlussschätzer mit ein um in Abschnitt 3.3 festzustellen, dass der einfachere Algorithmus aus Abschnitt 3.1 meist überlegen ist. Nachdem wir dann in Abschnitt 3.4 noch darauf eingehen, wie ein polynomiales Gleichungssystem zu transformieren ist, um Lösungen auf ganz \mathbb{R}^n bestimmen zu können, betrachten wir in Abschnitt 3.5 noch einige Beispiele.

3.1 Vorgehen ohne Einschlussschätzer

Die Idee unseres Algorithmus besteht darin, dass die Kontrollstruktur in engem Zusammenhang mit dem dargestellten Bézierpolynom steht. Sei für die weiteren Betrachtungen zunächst $n = 1$. Nehmen wir als Beispiel das Bézierpolynom $p : [0, 1] \rightarrow \mathbb{R}$ mit den Kontrollpunkten $P = [-2; 1; 2; 4]$. Identifizieren wir nun dieses Bézierpolynom mit seinem Graphen. Dies entspricht der Erweiterung der Kontrollpunkte um ihre Greville-Abszissen, also $Q = [0, -2; 1/3, 1; 2/3, 2; 1, 4]$. Nullstellen von p sind Schnittpunkte von $q(x) = (x, p(x))$ und der x -Achse (siehe Abbildung 9, erstes Bild). Wegen der Konvexen-Hülle-Eigenschaft können sich Nullstellen von p nur dort befinden, wo die konvexe Hülle von Q die x -Achse schneidet (siehe Abbildung 9, zweites Bild). Die Berechnung von konvexen Hüllen ist sehr aufwändig, insbesondere für höhere Dimensionen $n > 1$. In noch stärkerem Maße trifft dies für den Schnitt von konvexen Mengen

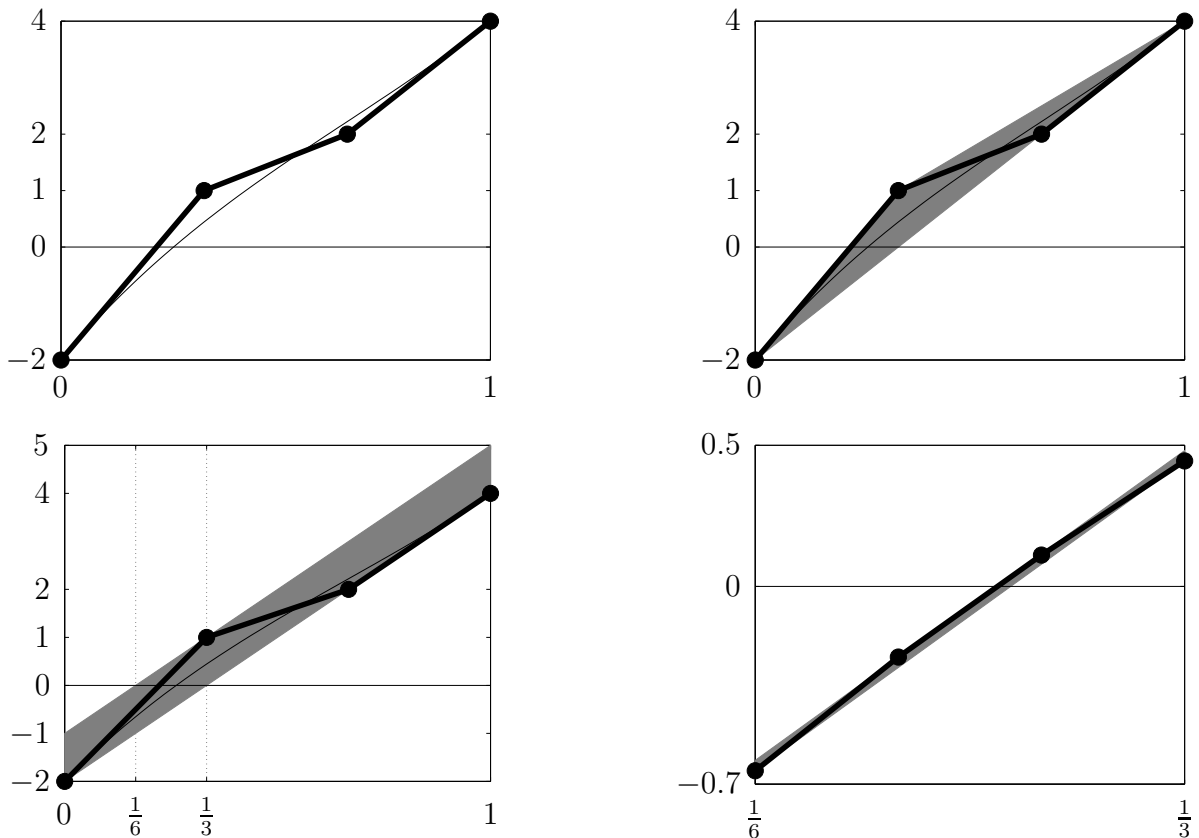


Abbildung 9: Eingrenzung des Bereichs möglicher Nullstellen für eine polynomiale Gleichung in einer Variablen

zu, welcher für $n > 1$ gebildet werden muss. Deshalb berechnen wir eine die konvexe Hülle umfassende Menge, die möglichst einfache Gestalt besitzt. Wir wählen für $n = 1$ zwei parallele Geraden, deren Steigung an die Kontrollpunkte P angepasst ist. Diese verschieben wir so, dass sie alle Kontrollpunkte einschließen (siehe Abbildung 9, drittes Bild). Nun berechnen wir die Schnittpunkte s_1, s_2 dieser beiden Geraden mit der x -Achse. Nullstellen von p sind nur im Intervall $[s_1, s_2]$ möglich. Dann subdividieren wir die Kontrollpunkte P auf das Intervall $[s_1, s_2]$. Nun liegen diese Kontrollpunkte noch dichter an dem dargestellten Bézierpolynom, sodass die Bereiche möglicher Nullstellen sukzessive immer kleiner werden (siehe Abbildung 9, viertes Bild). Wie wir sehen werden, kontrahieren die Bereiche möglicher Nullstellen quadratisch.

Das hier Gesagte lässt sich völlig analog für $n > 1$ übertragen. Dann muss für die einzelnen Komponenten $p^{\{k\}} = \pi_k(p)$ von p so wie oben erläutert verfahren werden und die erhaltenen Bereiche möglicher Nullstellen miteinander geschnitten werden. Sei z.B. $n = 2$ und $p : [0, 1]^2 \rightarrow \mathbb{R}^2$, wobei $P^{\{1\}} \in \mathbb{R}^{5 \times 4}$ mit $P^{\{1\}} = [1, 0.8, 0.5, 0.4; 0.4, 0.2, 0.2, 0.1; 0.1, -0.1, -0.2, -0.2; -0.2, -0.2, -0.4, -0.7; -0.5, -0.6, -0.8, -1.1]$ die Kontrollpunkte von $p^{\{1\}}(x, y) = \pi_1(p(x, y))$ seien. Wieder

erweitern wir zunächst die Kontrollpunkte um ihre Greville-Abszissen (siehe Abbildung 10, erstes Bild). Als Normalenvektor der einschließenden Ebenen wählen wir dann $N = [1.5, 0.6, 1]$. Mit $d_{\min} = 0.775$ und $d_{\max} = 1.15$ schließen die beiden Ebenen $\{x \in \mathbb{R}^3 : \langle N, x \rangle = d_{\min}\}$ und $\{x \in \mathbb{R}^3 : \langle N, x \rangle = d_{\max}\}$ alle Kontrollpunkte der um die Greville-Abszissen erweiterten Kontrollpunkte ein (siehe Abbildung 10, erstes Bild). Die beiden Ebenen schneiden die z -Ebene in den beiden Geraden, die im zweiten Bild von Abbildung 10 das grau gefärbte Gebiet beranden. Nur in diesem Gebiet sind Nullstellen der ersten Gleichung möglich. Verfährt man mit der zweiten Gleichung $p^{(2)}(x, y) = \pi_2(p(x, y))$ entsprechend, so erhält man für deren Nullstellen ein gleichartiges Gebiet (siehe Abbildung 10, drittes Bild). Gemeinsame Nullstellen sind nur im Schnitt dieser beiden Gebiete möglich. Wir rechnen also die 4 Schnittpunkte der vier Geraden aus und schließen diese in ein Intervall ein (siehe Abbildung 10, viertes Bild). Dann subdividieren wir auf diesen Bereich. So fahren wir sukzessive fort. Dabei sind zweckmäßigerweise die Kontrollpunkte stets als zum Intervall $[0, 1]^n$ gehörend zu betrachten. Zusätzlich protokollieren wir, in welcher Box wir uns tatsächlich befinden.

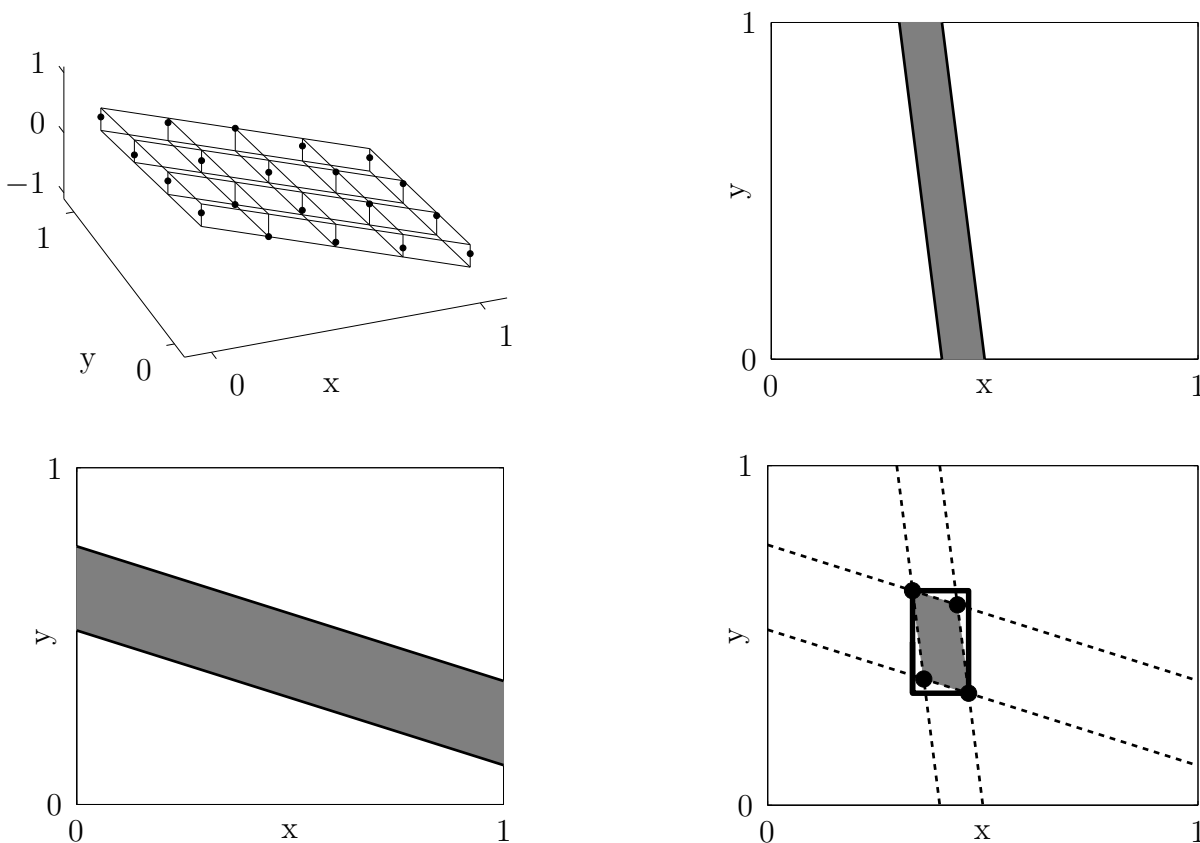


Abbildung 10: Eingrenzung des Bereichs möglicher Nullstellen für ein polynomiales Gleichungssystem in zwei Variablen

Denkt man kurz über das oben gesagt nach, so stellt man fest, dass auf diese Art und Weise nicht mehr als eine Nullstelle gefunden werden kann, schließlich arbeitet man nur mit einer Box, die sukzessive kleiner wird. Auch scheint keineswegs gewährleistet, dass diese eine Box beliebig klein wird. Kontrahiert aber in einem Schritt unsere Box nicht wesentlich, so spalten wir durch Subdivision diese auf in zwei neue, jeweils halb so große Boxen. Mit dieser Strategie separieren wir die verschiedenen Nullstellen und grenzen sie beliebig fein ein.

Die Zielrichtung dieses Abschnitts ist somit klar. Fassen wir unsere Ideen nun in den folgenden Sätzen zusammen. Der erste Satz 3.1 belegt, dass in den Kontrollpunkten eines multivariaten Polynoms geometrische Information für die Lage von Nullstellen liegt.

Satz 3.1 Sei $M = (m_1, \dots, m_n)$ die Ordnung des Bézierpolynoms $p(x) = \sum_I b_{M,I} p_I$ mit $p_I \in \mathbb{R}$, wobei $I = (i_1, \dots, i_n)$.

Erweitern wir nun die Kontrollpunkte p_I um ihre Greville-Abszissen, also

$$q_I := \left(\frac{i_1 - 1}{m_1 - 1}, \dots, \frac{i_n - 1}{m_n - 1}, p_I \right),$$

so gilt für $x \in [0, 1]^n$

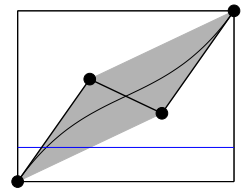
$$p(x) = 0 \quad \implies \quad (x, 0) \in \text{ConvexHull}_I(q_I).$$

Insbesondere gilt auch für jedes $S \subset \mathbb{R}^{n+1}$ mit $S \supset \text{ConvexHull}_I(q_I)$

$$p(x) = 0 \quad \implies \quad (x, 0) \in S.$$

Beweis Zunächst gilt mit $q = \sum_I b_{M,I} q_I$ nach Satz 2.22 $q(x) = (x, p(x))$, also mit Satz 2.21: $q(x) = (x, p(x)) \in \text{ConvexHull}_I(q_I)$. Wenn also $p(x) = 0$, dann muss gelten $(x, 0) \in \text{ConvexHull}_I(q_I)$. \square

Satz 3.1 bildet die Grundidee unseres Algorithmus zur Auffindung von Nullstellen und findet sich schon in [SP93]. Wenn man diese Idee weiter verfolgt, so stößt man unmittelbar auf das Problem, dass konvexe Hüllen im \mathbb{R}^{n+1} berechnet und miteinander geschnitten werden müssen. Dies ist aber, wie bereits erwähnt, ein sehr rechenintensives Problem, welches umgangen werden sollte. Unser Ansatz und der aus [SP93] unterscheiden sich genau darin, wie das Bilden von konvexen Hüllen im \mathbb{R}^{n+1} vermieden wird. Die in [SP93] vorgeschlagene Strategie der Projektion verliert die quadratische Konvergenz, während wir diese Konvergenzordnung theoretisch und praktisch erhalten können. Anstatt die konvexe Hülle zu berechnen, arbeiten wir mit einer geeigneten Menge S , die die konvexe Hülle umfasst.

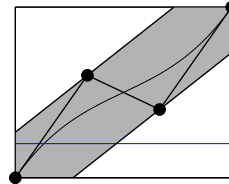


Satz 3.2 Seien die p_I und q_I wie im letzten Satz. Sei außerdem $N \in \mathbb{R}^{n+1}$ ein Normalenvektor. Wir setzen

$$d_{\min} := \min_I \langle N, q_I \rangle \quad \text{und} \quad d_{\max} := \max_I \langle N, q_I \rangle.$$

Dann gilt mit $S := \{(x_1, \dots, x_{n+1}) : \langle N, x \rangle \in [d_{\min}, d_{\max}]\}$

$$S \supset \text{ConvexHull}_I(q_I).$$



Beweis S ist konvex und enthält alle q_I . □

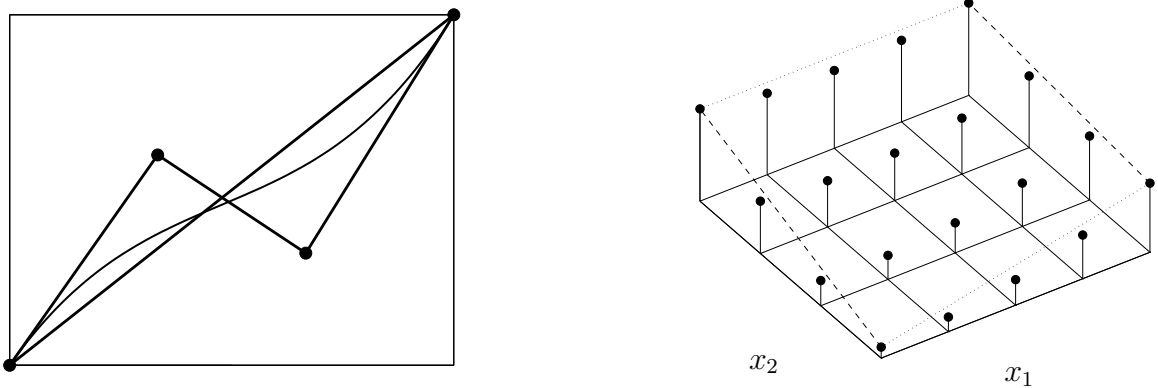
Wenn wir nun zu einem $p : \mathbb{R}^n \rightarrow \mathbb{R}$ den möglichen Nullstellenbereich abschätzen möchten, so können wir mit einem geeigneten $N \in \mathbb{R}^{n+1}$ die $d_{\min}/_{\max}$ berechnen. Es ist dann mit $S = \{(x_1, \dots, x_{n+1}) : \langle N, x \rangle \in [d_{\min}, d_{\max}]\}$ aus dem vorigen Satz der Bereich möglicher Nullstellen gegeben durch

$$S \cap \{(x_1, \dots, x_{n+1}) : x_{n+1} = 0\} \cong \{(x_1, \dots, x_n) : \langle N(1:n), x \rangle \in [d_{\min}, d_{\max}]\}.$$

Doch wie ist der Normalenvektor N zu wählen? N definiert die parallelen Hyperebenen, die durch Verschiebung die Kontrollpunkte möglichst dicht einschließen sollen. Dieser Einschluss sollte möglichst gut an das Kontrollnetz angepasst sein, sodass insbesondere im linearen Fall, wenn alle Kontrollpunkte in einer gemeinsamen Hyperebene liegen, N als der zu dieser Hyperebene gehörende Normalenvektor bestimmt wird. Außerdem sollte das Verfahren affin invariant sein, d.h. zu affin abgebildeten Kontrollpunkten, sollten die affin abgebildeten Einschluss-Ebenen berechnet werden. Unter Beachtung dieser Vorgaben sollte die Berechnung von N natürlich möglichst einfach sein. Wir gehen nun folgendermaßen vor. Sei zunächst $n = 1$. Als Steigung der Einschluss-Geraden wählen wir die mittlere Steigung des Bézierpolynoms, also $p(1) - p(0) = p_m - p_1$. Sei nun $n = 2$. Wir wählen die Ebene so (siehe Abbildung 11), dass deren Steigung r^1 in x_1 -Richtung gleich der mittleren Steigung der äußeren (gepunkteten) Geraden in x_1 -Richtung ist, bzw. deren Steigung r^2 in x_2 -Richtung gleich der mittleren Steigung der äußeren (gestrichelten) Geraden in x_2 -Richtung ist. Der Normalenvektor zur Ebene, die von den Vektoren $[1, 0, r^1]$ und $[0, 1, r^2]$ aufgespannt wird, ist $N = [-r^1, -r^2, 1]$. Verallgemeinern wir unser Vorgehen im folgenden Satz.

Satz 3.3 Sei das Bézierpolynom $p : [0, 1]^n \rightarrow \mathbb{R}$ gegeben und für $i \in 1 : n$ sei $r^i := \text{mean}[p_{(j_1, \dots, m_i, \dots, j_n)} - p_{(j_1, \dots, 1, \dots, j_n)} \mid j \in \{1, m_1\} \times \dots \times \{1, m_n\}]$ und $R^i \in \mathbb{R}^{n+1}$ der Vektor in x_i -Richtung mit Steigung r^i , also

$$R^i(j) = \begin{cases} 1 & , \quad j = i \\ r^i & , \quad j = n + 1 \\ 0 & , \quad \text{sonst} \end{cases}$$

Abbildung 11: Bestimmung der Hyperebene bzw. dessen Normalenvektor N

und $N \in \mathbb{R}^{n+1}$ mit

$$N(j) = \begin{cases} -r^i & , j \in 1 : n \\ 1 & , j = n + 1 . \end{cases}$$

Dann gilt für $i \in 1 : n$

$$\langle N, R^i \rangle = 0$$

und somit ist N ein Normalenvektor zu den beiden Einschuss-Hyperebenen, die durch die Vektoren R^i aufgespannt werden.

Alle drei Forderungen der affinen Invarianz, der Reproduktion von Hyperebenen und der Einfachheit haben wir mit dem obigen Vorgehen erfüllt. Alternativ kann man z.B. die Ebenen auch als Verschiebungen der Ausgleichsebene wählen. Dies bringt für die Größe der Boxen möglicher Nullstellen in der Praxis nach unserer Erfahrung aber weder Vor- noch Nachteile, auch nicht für die Zahl der untersuchten Boxen. Wir verwerfen deshalb diesen Ansatz, weil er für die Implementation und die Anzahl der Rechenoperationen pro Schritt aufwändiger ist.

Sei nun ein Polynom $p : [0, 1]^n \rightarrow \mathbb{R}^n$ in Bézierdarstellung gegeben und die Nullstellen gesucht. Für die Funktionen $p^{\{k\}} := \pi_k(p)$ können wir, wie oben erläutert, den Bereich möglicher Nullstellen abschätzen. Eine gemeinsame Nullstelle aller $p^{\{k\}}$ kann nur im Schnitt der Nullstellenbereiche liegen. Wie dieser Schnitt berechnet werden kann, zeigt uns der folgende Satz, welcher im Grunde eine nützliche Zusammenstellung von einfachen Schlussfolgerungen darstellt. Wir geben ihn ohne Beweis an. Für die Formulierung ist es notwendig, die bisher verwendeten Bezeichnungen $M, p, q, N, d_{\min}, d_{\max}$ um den zusätzlichen Index $\{k\}$ für die einzelnen $p^{\{k\}}$ zu erweitern, sodass wir nun $M^{\{k\}}, p^{\{k\}}, q^{\{k\}}, N^{\{k\}}, d_{\min}^{\{k\}}, d_{\max}^{\{k\}}$ schreiben.

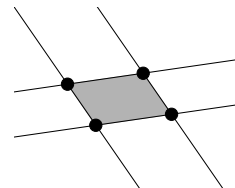
Satz 3.4 Seien $N^{\{k\}}, d_{\min}^{\{k\}}, d_{\max}^{\{k\}}$ für $k \in 1 : n$ wie in Satz 3.3 zum Polynom $p : [0, 1]^n \rightarrow \mathbb{R}^n$ gegeben. Setzen wir nun die Menge S als den Schnitt der möglichen

Nullstellenbereiche

$$S := \{x \in \mathbb{R}^n : \langle N^{\{k\}}(1:n), x \rangle \in [d_{\min}^{\{k\}}, d_{\max}^{\{k\}}] \quad \forall k \in (1:n)\}$$

und

$$E := \{x \in \mathbb{R}^n : \langle N^{\{k\}}(1:n), x \rangle \in \{d_{\min}^{\{k\}}, d_{\max}^{\{k\}}\} \quad \forall k \in (1:n)\},$$



dann ist E die Menge der Ecken der konvexen Menge S , es gilt also

$$\text{ConvexHull}(E) = S.$$

Die Menge E berechnet sich dabei wie folgt. Sei dazu die Matrix $N \in \mathbb{R}^{n \times n}$ definiert durch

$$N(k, j) = N^{\{k\}}(j)$$

und $D \in \mathbb{R}^{n \times 2^n}$ eine Matrix mit paarweise verschiedenen Spalten und

$$D(k, l) \in \{d_{\min}^{\{k\}}, d_{\max}^{\{k\}}\}.$$

D ist durch die obigen Eigenschaften bis auf Permutation der Spalten eindeutig bestimmt. Falls N invertierbar ist, ist die Menge E gegeben durch die Spalten von $N^{-1}D$, wir schreiben

$$E = N^{-1}D. \quad (3.1)$$

Mit $A := \min E$ und $B := \max E$, wobei das Minimum und Maximum zeilenweise zu bilden ist, sind Nullstellen von p nur in der Box $[A(1), B(1)] \times \dots \times [A(n), B(n)]$ möglich. Alternativ lässt sich die Box möglicher Nullstellen berechnen als

$$\hat{E} = N^{-1}\hat{D}. \quad (3.2)$$

Dabei sei \hat{D} der Intervall-Vektor $\hat{D} := ([d_{\min}^{\{1\}}, d_{\max}^{\{1\}}], \dots, [d_{\min}^{\{n\}}, d_{\max}^{\{n\}}])^T$. Der Intervall-Vektor \hat{E} ergibt sich dann gemäß den untenstehenden Rechenregeln der Intervall-Arithmetik als $\hat{E} = ([A(1), B(1)], \dots, [A(n), B(n)])^T$.

Die Rechenregeln der Intervall-Arithmetik sind

$$[a, b] + [c, d] = [a + c, b + d] \quad (3.3)$$

$$c[a, b] = \{cy : y \in [a, b]\}. \quad (3.4)$$

Die Variante (3.2) ist in obigem Satz ganz klar der Variante (3.1) vorzuziehen. Insbesondere für große n ist Variante (3.1) nicht mehr praktikabel. Der Grund, warum wir auch Variante (3.1) angegeben haben, liegt darin, dass (3.1) mehr zum Verständnis beiträgt als (3.2).

1. Wir bestimmen die $N^{\{k\}}$ zu den $p^{\{k\}}$ so wie in Satz 3.3 angegeben.
2. Wir bestimmen zu den $N^{\{k\}}$ die $d_{\min/\max}^{\{k\}}$ wie in Satz 3.2 angegeben.
3. Gemäß Satz 3.4 stellen wir die Matrizen N und \hat{D} auf. Wenn N invertierbar ist erhalten wir zwei Vektoren A und B sodass Nullstellen von p nur in der Box $[A(1), B(1)] \times \cdots \times [A(n), B(n)] \cap [0, 1]^n$ möglich sind.
4. Wir subdividieren gemäß Satz 2.24 auf den neuen Bereich, falls dieser auch tatsächlich deutlich kleiner ist als der alte. Ansonsten teilen wir die alte Box in zwei Teile und untersuchen beide Teile getrennt voneinander weiter. Ebenso teilen wir in zwei Teile auf, falls die Matrix N nicht invertierbar ist. So fahren wir rekursiv fort.

Die obigen Bezeichnungen haben wir in Abbildung 12 visualisiert. Außerdem geben wir mit Abbildung 13 ein Ablaufdiagramm an.

Wir können also den folgenden Algorithmus formulieren. Dabei erledige die Routine `FindHyperplaneEnclosure` die Schritte 1 und 2, bzw. die Routine `FindNewBox` den Schritt 3 und `Subdivide` den Schritt 4.

```
function RO =Roots(Controlpoints,Box,tol)
% ROOTS berechnet alle Nullstellen des multivariaten Polynoms p gegeben
% durch die Bezierpunkte P fuer die Box B=[a_1,b_1]x...x[a_n,b_n].
% Input P Cell mit n Elementen, also P={P_1,...,P_n}, wobei P_i die
% Bezierpunkte zu \pi_i(p) sind.
% Box zweispaltige Matrix [a_1,...a_n;b_1,...b_n]'
% tol Toleranz
% Output RO Matrix, wobei jede Spalte von N eine Box mit einem moeglichen
% Nullstellenbereich von p enthaelt.
% N(:,i)=[a_1,...a_n,b_1,...,b_n]'
```

```
ShrinkFactor=0.8;
```

```
Pv={Controlpoints};
```

```
Bo={Box};
```

```
k =1;
```

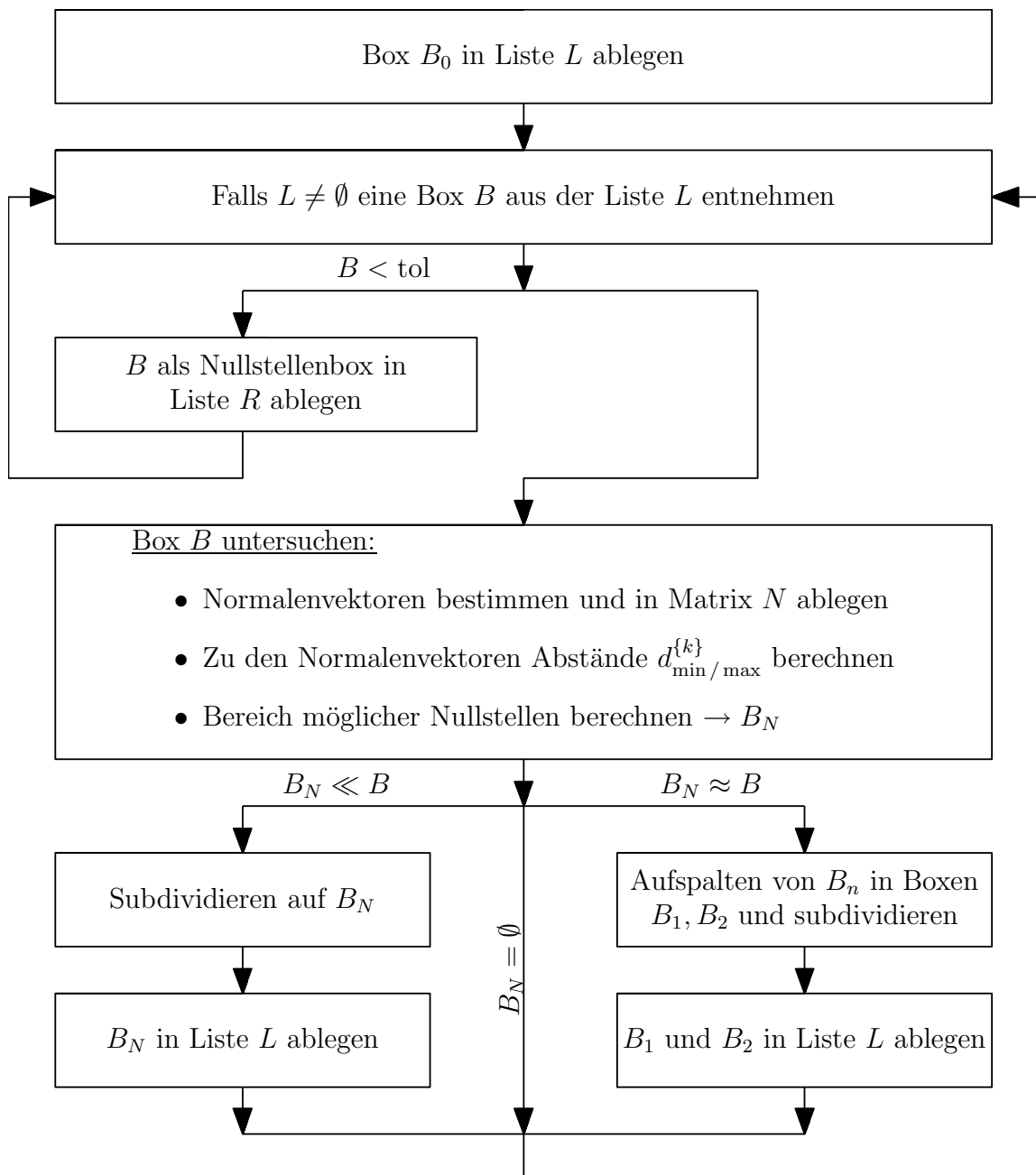
```
RO =[];
```

```
while k>0
```

```
    P=Pv{k};
```

```
    B=Bo{k};
```

```
    k=k-1;
```

Abbildung 13: Ablaufdiagramm des Nullstellenalgorithmus für Suche in der Box B_0

```

if all(B(2,:)-B(1,:)<tol) % Nullstelle gefunden
    RO=[RO B(:)];
    continue;
end
[N,D]=FindHyperplaneEnclosure(P);
NewBox=FindNewBox(N,D);
if isempty(NewBox)
    continue;
elseif all(NewBox(2,:)-NewBox(1,:)<ShrinkFactor)
    k=k+1;
    Pv{k}=Subdivide(P,NewBox);
    B{k}=[(1-NewBox(1,:)).*B(1,:)+NewBox(1,:).*B(2,:);
          (1-NewBox(2,:)).*B(1,:)+NewBox(2,:).*B(2,:)];
else
    %-> find dimension with longest boxedge of nonshrinking direction
    I=find(NewBox(2,:)-NewBox(1,:)>=ShrinkFactor);
    [Length,Dimension]=max(B(2,I)-B(1,I));
    MaxDim=I(Dimension);
    %<-
    NewBox(:,MaxDim)=[NewBox(1,MaxDim); ...
                      (NewBox(1,MaxDim)+NewBox(2,MaxDim))/2];
    k=k+1;
    Pv{k}=Subdivide(P,NewBox);
    Bo{k}=[(1-NewBox(1,:)).*B(1,:)+NewBox(1,:).*B(2,:);
           (1-NewBox(2,:)).*B(1,:)+NewBox(2,:).*B(2,:)];
    NewBox(:,MaxDim)=[(NewBox(1,MaxDim)+NewBox(2,MaxDim))/2; ...
                      NewBox(1,MaxDim)];
    k=k+1;
    Pv{k}=Subdivide(P,NewBox);
    Bo{k}=[(1-NewBox(1,:)).*B(1,:)+NewBox(1,:).*B(2,:);
           (1-NewBox(2,:)).*B(1,:)+NewBox(2,:).*B(2,:)];
end
end
end

```

Wir wollen noch einige Anmerkungen machen. Bevor man die relativ teure Routine `FindHyperplaneEnclosure` aufrufen, empfiehlt es sich zu testen, ob für ein $\pi_k(p)$ alle Kontrollpunkte dasselbe Vorzeichen besitzen. In diesem Fall sind Nullstellen unmöglich. Dieser Test ist vergleichsweise billig und erspart im günstigen Fall verhältnismäßig viel Arbeit.

Im obigen Algorithmus wurde getestet, ob die Box in allen Dimensionen um mindestens den `ShrinkFactor` schrumpft. Natürlich ist die Größe dieses Faktors diskutabel,

`ShrinkFactor=0.8` hat sich aber in der Praxis als günstig erwiesen.

Das Schrumpfen in allen Dimensionen gleichzeitig zu verlangen, wird durch praktische Tests nahegelegt. Es ist ansonsten ein typisches Verhalten des Algorithmus, dass die Box nur in einer Dimensionen kontrahiert, in den anderen Dimensionen jedoch nicht. Verbessern lässt sich dieser Ansatz wie folgt: Man kann sich zu einer Box merken, wieviele Iterationen je Dimension bereits vergangen sind, ohne dass die Box in dieser Dimension um mindestens `ShrinkFactor` kleiner wurde. Wenn weder in den letzten i Iterationen (Vorschlag: $i = 1$) noch in der aktuellen Iteration eine Kontraktion eintrat, so wird in derjenigen Dimension aufgespalten, in der am längsten nichts passierte. So ist sichergestellt, dass die Box in allen Dimensionen im Lauf der Zeit kontrahiert. Betrachten wir hierzu ein Beispiel ohne konkrete Kontrollpunkte mit $i = 2$. Sei das Polynom $p : [0, 1]^2 \rightarrow \mathbb{R}^2$ gegeben und die Nullstellen von p gesucht. Im ersten Schritt stellen wir fest, dass Nullstellen nur in $[0.3, 0.9] \times [0, 1]$ möglich sind. Wir subdividieren auf diesen Bereich und stellen dann fest, dass Nullstellen nur in $[0.4, 0.8] \times [0, 1]$ auftreten können und subdividieren nun auf diesen Bereich. Nun sind bereits $i = 2$ Iterationen ohne Kontraktion in der zweiten Dimension verstrichen. Stellen wir nun im nächsten Schritt fest, dass wieder nichts in der zweiten Dimension passiert, dass also z.B. nur in $[0.6, 0.7] \times [0, 1]$ Nullstellen liegen können, so spalten wir auf in die beiden Boxen $[0.6, 0.7] \times [0, 1/2]$ und $[0.6, 0.7] \times [1/2, 1]$.

Aus numerischen Gründen sollte man dafür sorgen, dass die Boxen nicht zu stark kontrahieren. Untersucht man beispielsweise ein fast-lineares System, so wird als Box möglicher Nullstellen $B = [a_1, b_1] \times \dots \times [a_n, b_n]$ mit $b_j - a_j \approx \varepsilon$ bestimmt. Subdividiert man nun auf dieses Intervall, so besitzen in der Regel durch Rundungsfehler anschließend alle Kontrollpunkte das gleiche Vorzeichen, sodass im nächsten Schritt festgestellt wird, dass Nullstellen ausgeschlossen werden können. Wir haben deshalb grundsätzlich bei der Subdivision die Kantenlänge der Boxen um den Faktor $(1 + 10^{-6})$ aufgebläht. Dies ist auch deshalb vorteilhaft, weil dann Nullstellen vom Rand ins Innere wandern. Der größte Teil des Aufwandes unseres Verfahrens liegt in der Subdivision, die nach der Berechnung des Bereichs der möglichen Nullstellen durchgeführt werden muss. Um sich „unnötige“ Arbeit zu sparen, kann man grundsätzlich nur in denjenigen Dimensionen subdividieren, in denen eine Kontraktion um mindestens einen festen Faktor $\alpha < 1$ stattfindet. Bei der Subdivision gewinnt man nämlich durch Einschränkung auf einen kleineren Bereich genauere Information über denselben. Verkleinert man jedoch den Bereich nur minimal, so gewinnt man auch nur minimale Zusatzinformation bei vollem Aufwand für die Subdivision. Wir haben deshalb in unseren Algorithmen mit $\alpha = \text{ShrinkFactor} = 0.8$ gearbeitet.

Wenn man die Algorithmen in MATLAB realisiert, dann ergibt sich ein großes Potenzial der Optimierung. Schleifen sind in MATLAB stets sehr teuer. Die Berechnung des (nicht schleifenfrei berechenbaren) DeCasteljau-Schemas kann umgangen werden und die Subdivision völlig schleifenfrei durchgeführt werden. Dies wird ganz am Ende des Kapitels 4.2 in Satz 4.44 besprochen. Der schleifenfreie Ansatz ist für große Ordnungen

allerdings numerisch instabil und sollte deshalb nur für kleine n ($n < 10$) benutzt werden. Für kleine Ordnungen ergibt sich in MATLAB, eine wesentliche Zeitersparnis. In anderen Programmiersprachen bringt die schleifenfreie Subdivision keine Vorteile. In unserem Algorithmus wurden alle diese Empfehlungen berücksichtigt und die Tests mit diesen Einstellungen vorgenommen.

Betrachten wir nun ein Beispiel für den Algorithmus `Roots` mit konkreten Kontrollpunkten.

Beispiel 3.6 Sei $n = 2$ und die Nullstellen des biquartischen Polynoms p gesucht, das durch die Kontrollpunkte P gegeben ist. Seien die Kontrollpunkte für die erste Gleichung

$$P^{\{1\}} = \begin{bmatrix} -0.3 & 0.5 & 0.9 & 1.1 & 0.9 \\ -0.2 & 0.6 & 0.9 & 1.1 & 0.9 \\ -0.1 & 0.7 & 0.9 & 1.1 & 0.9 \\ 0.1 & 0.8 & 1.1 & 1.2 & 1.1 \\ 0.1 & 0.9 & 1.2 & 1.3 & 1.1 \end{bmatrix}.$$

Die Kontrollpunkte für die zweite Gleichung seien das Doppelte der Transponierten von $P^{\{1\}}$. Nun berechnen wir zwei parallele Ebenen, die die Kontrollpunkte des Graphen zu $P^{\{1\}}$ einschließen. Die Ebenen spannen wir auf durch die beiden Vektoren $[1, 0, 1/2(0.1 - (-0.3)) + 1/2(1.1 - 0.9)] = [1, 0, 0.3]$ und $[0, 1, 1/2(0.9 - (-0.3)) + 1/2(1.1 - 0.1)] = [0, 1, 1.1]$. Das ergibt für die Hessesche Normalform mit der Normierung $N^{\{1\}}(3) = 1$ den Normalenvektor $N^{\{1\}} = [-0.3, -1.1, 1]$. Als Abstand der unteren bzw. der oberen Einschließenden ergibt sich $d_{\min}^{\{1\}} = -0.35$ bzw. $d_{\max}^{\{1\}} = 0.35$. Die Kontrollstruktur (gepunktet) und die einschließenden Ebenen sind im linken Bild in Abbildung 14 zu sehen. Für die zweite Gleichung erhalten wir $N^{\{2\}} = [-2.2, -0.6, 1]$ und $d_{\min}^{\{2\}} = -0.7$ bzw. $d_{\max}^{\{2\}} = 0.7$. Berechnen wir zunächst die Box möglicher Nullstellen gemäß (3.1). Nullstellen des Gleichungssystems können folglich nur in dem 2-Spat mit den Ecken $E = N \setminus D$ liegen, wobei

$$N = \begin{bmatrix} -0.3 & -1.1 \\ -2.2 & -0.6 \end{bmatrix} \quad D = \begin{bmatrix} -0.35 & 0.35 & -0.35 & 0.35 \\ -0.7 & -0.7 & 0.7 & 0.7 \end{bmatrix}.$$

Nun ergibt sich

$$E = N \setminus D = \begin{bmatrix} 0.25 & 0.4375 & -0.4375 & -0.25 \\ 0.25 & -0.4375 & 0.4375 & -0.25 \end{bmatrix}.$$

Also nur in $[-0.4375, 0.4375] \times [-0.4375, 0.4375] \cap [0, 1]^2 = [0, 0.4375]^2$ sind Nullstellen möglich.

In diesem Beispiel könnte man den Bereich möglicher Nullstellen sogar auf $[0, 0.3182]$ eingrenzen (Siehe Abbildung 14, unteres Bild). Die Diskrepanz ergibt sich, weil Schnittpunkte der Hyperebenen außerhalb von $[0, 1]^2$ die Box durch die Bildung des zeilenweisen Minimums und Maximums unnötig vergrößern. Diese Sonderfälle im Algorithmus

tatsächlich abzufangen ist aber nicht vorteilhaft, weil dies einen erheblichen Mehraufwand erfordert, aber nur minimale Vorteile bietet.

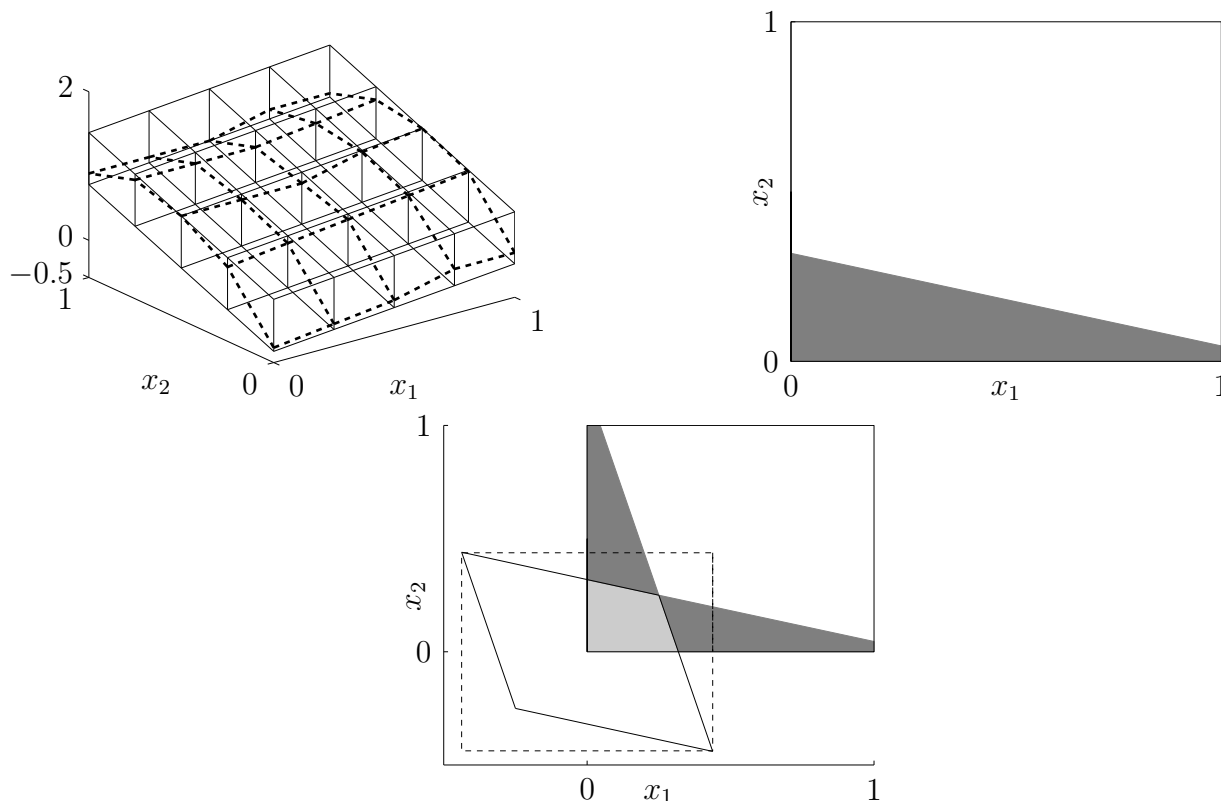


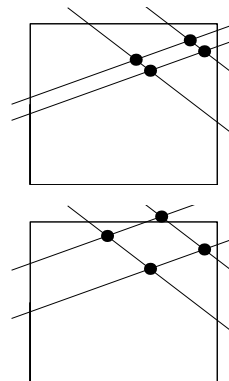
Abbildung 14: Links: Kontrollstruktur (gestrichelt) und oberer/unterer Einschuss für die erste Gleichung. Rechts: Sich daraus ergebender Bereich möglicher Nullstellen. Unten: Bereich möglicher gemeinsamer Nullstellen der beiden Gleichungen

3.1.1 Existenz und Eindeutigkeit von Nullstellen, Konvergenzgeschwindigkeit

Im vorangegangenen Abschnitt haben wir die Implementierung geklärt. Der Output von `Roots` ist eine Menge von Boxen. Nullstellen von p können ausschließlich in diesen Boxen liegen. Was wir gerne wüßten ist, ob denn tatsächlich genau eine Nullstelle in jeder Box liegt. Außerdem ist die quadratische Konvergenz unseres Algorithmus zu beweisen.

In Analogie zu Intervall-Newton-Methoden gibt es ein sehr einfaches Existenz-Kriterium, das keinerlei Zusatzaufwand erfordert. Kurz gesagt lautet es: „Ist die Box möglicher Nullstellen ganz in der betrachteten Box enthalten, so ist eine Nullstelle garantiert.“

Satz 3.7 Seien die Bezeichnungen allesamt wie bisher in diesem Kapitel (siehe Satz 3.4), also B der Definitionsbereich zu dem die Kontrollpunkte $P^{\{1\}}, \dots, P^{\{n\}}$ gehören (welche das Bézierpolynom $p : B \rightarrow \mathbb{R}^n$ definieren), N die Matrix der Normalenvektoren, die mit D die Einschlüsse der Bézierpolynome bestimmen. Liegen nun alle Spalten von $E = N^{-1}D$ (siehe Satz 3.4) im Inneren von B , so ist eine Nullstelle von p in B garantiert (siehe oberes Bild im Vergleich zum unteren).



Beweis Zum Beweis benötigen wir den Fixpunktsatz von Brouwer: „Jede stetige Abbildung $c : [-1, 1]^n \rightarrow [-1, 1]^n$ besitzt mindestens einen Fixpunkt.“

Zunächst existiert eine affine Abbildung $a : [-1, 1]^n \rightarrow E$ so, dass für beliebige $x \in [-1, 1]^n$ mit $d := p \circ a$ gilt:

$$d_k(x_1, \dots, -1, \dots, x_n) > 0, \quad d_k(x_1, \dots, 1, \dots, x_n) < 0.$$

Wenn d Nullstellen besitzt, so auch p . Dass aber d Nullstellen besitzt folgt nun aus dem Fixpunktsatz von Brouwer angewendet auf die Funktion $c(x) = x + \alpha d(x)$ für ein hinreichend kleines $\alpha > 0$. Dann ist c nämlich eine stetige Selbstabbildung auf $[-1, 1]^n$ und besitzt einen Fixpunkt. Ein Fixpunkt von c ist aber gleichbedeutend mit einer Nullstelle von d und somit ist alles gezeigt. \square

Als nächstes wenden wir uns der Eindeutigkeit von Lösungen zu. Zunächst klären wir die notwendige Terminologie.

Definition 3.8 Wir nennen eine Matrix M eine Intervall-Matrix, wenn die Einträge $M_{k,j}$ von M nicht reelle Zahlen, sondern Intervalle $I_{k,j} \subset \mathbb{R}$ sind. Wir nennen eine reelle Matrix A ein Element einer Intervall-Matrix M , falls für alle Einträge gilt: $A_{k,j} \in I_{k,j}$. Eine Intervall-Matrix M heißt invertierbar falls alle Elemente A von M invertierbar sind.

Sei B nun eine kompakte, konvexe Menge im \mathbb{R}^n und $p : B \rightarrow \mathbb{R}^n$ eine stetig definierbare Funktion auf B . Sei zusätzlich D eine Intervall-Matrix mit

$$D_{k,j} \supset \left[\min_{x \in B} \frac{\partial f_k}{\partial x_j}, \max_{x \in B} \frac{\partial f_k}{\partial x_j} \right].$$

Angenommen es gibt zur Funktion $p : B \rightarrow \mathbb{R}^n$ in der konvexen Menge $B \subset \mathbb{R}^n$ zwei Lösungen $p(x_1) = 0 = p(x_2)$ zur Funktion $p : B \rightarrow \mathbb{R}^n$, dann gibt es zu jedem

$k \in (1 : n)$ ein \tilde{x}_k auf der Verbindungsstrecke von x_1 und x_2 sodass

$$\left[\frac{\partial f_k}{\partial x_1}(\tilde{x}_k), \dots, \frac{\partial f_k}{\partial x_n}(\tilde{x}_k) \right] (x_2 - x_1) = 0.$$

Dies impliziert aber, dass $0 \in D_{k,:}(x_2 - x_1)$. Da dies aber für alle k gilt muss $0 \in D(x_2 - x_1)$ sein und somit kann D nicht im Intervall-Sinn invertierbar sein.

Die Intervall-Arithmetik stellt Algorithmen zur Verfügung, mit denen Intervall-Matrizen auf Invertierbarkeit untersucht werden können. Ein Standardproblem der Intervall-Arithmetik ist beispielsweise die Lösung von linearen Gleichungssystemen $Ax = b$, wobei A eine Intervall-Matrix und b ein Intervall-Vektor ist. Setzt man nun b als den Intervall-Vektor $[[1, 1], \dots, [1, 1]]^T$ und $A = D$ und löst $Ax = b$, so ist x genau dann ein Intervall-Vektor, falls A invertierbar ist. Im Fall, dass A nicht invertierbar ist und somit $Ax = b$ nicht gelöst werden kann, so wird dies festgestellt.

Die Intervall-Einträge von D können aber zu Polynomen p sehr leicht bestimmt werden. Wir leiten p gemäß Lemma 2.20 ab und schätzen die minimale bzw. maximale partielle Ableitung durch den minimalen/maximalen Kontrollpunkt ab.

In der Praxis empfiehlt es sich wie folgt vorzugehen. Man überprüft zunächst, ob die Bedingungen von Satz 3.7 erfüllt sind. Im positiven Fall ist die Existenz von Nullstellen garantiert. In diesem Fall überprüft man sofort das obige Eindeutigkeits-Kriterium und markiert die Box, falls auch die Eindeutigkeit sichergestellt ist.

Machen wir uns nun daran, die quadratische Konvergenz unseres Verfahrens zu beweisen. Der Beweis beruht letztendlich auf der Tatsache der quadratischen Konvergenz des Kontrollpolygons gegen die dargestellte Kurve unter Subdivision. Dies hat dann zur Folge, dass auch die Abstände der einschließenden Hyperebenen quadratisch gegen Null gehen.

Zunächst entsteht die Matrix N durch Mittelbildung von Differenzen, die allesamt bei Kontraktion der Box B in x_0 gegen die Ableitung in x_0 konvergieren. Konvergiert aber N gegen $p'(x_0)$ so konvergiert auch N^{-1} gegen $(p'(x_0))^{-1}$.

Nun können wir durch geschicktes Aufspalten die lokal quadratische Konvergenz unseres Algorithmus beweisen. Global erhalten wir offensichtlich stets lineare Konvergenz.

Satz 3.9 Sei $p : [0, 1]^n \rightarrow \mathbb{R}^n$ eine polynomiale Funktion und $x^0 \in (0, 1)^n$ eine Nullstelle von p . Dann konvergieren die Differenz der Spalten der gemäß Satz 3.4 definierten Matrix D quadratisch gegen Null.

Beweis Zu zeigen ist die quadratische Konvergenz von $\bar{p}_k(x) - A \cdot (x^0 - x)$ gegen Null. Dabei steht A für den Linearteil der einschließenden Ebenen.

$$\bar{p}_k(x) - A(x^0 - x) = (\bar{p}_k(x) - p_k(x)) + (p_k(x) - p'_k(x^0)(x^0 - x)) + (p'_k(x^0) - A) \cdot (x^0 - x).$$

Nun geht $\bar{p}_k(x) - p_k(x)$ quadratisch gegen Null. Durch Taylorentwicklung von $p_k(x)$ sieht man, dass auch $p_k(x) - p'_k(x^0)(x^0 - x)$ quadratisch gegen Null konvergiert und

der letzte Term $(p'_k(x^0) - N)(x^0 - x)$ setzt sich nach den obigen Überlegungen aus zwei linear konvergenten Faktoren mit Grenzwert Null zusammen, was schließlich die Behauptung beweist. \square

Fassen wir unsere Erkenntnisse aus den obigen Überlegungen zusammen und formulieren den folgenden Satz.

Satz 3.10 *Der in Abschnitt 3.1 vorgestellte Algorithmus grenzt die Nullstellen x_0 von $p : [0, 1]^n \rightarrow \mathbb{R}^n$ mit lokal quadratischer Konvergenz ein, sofern die Jacobi-Matrix $p'(x_0)$ invertierbar ist. Global ist die Konvergenz linear.*

Beweis Zu zeigen ist die quadratische Konvergenz der Spalten der Matrix E gemäß Satz 3.4 gegeneinander. Nun ergibt sich E nach Satz 3.4 als $N^{-1}D$. Wie schon erwähnt, konvergieren die N^{-1} . Die Spalten der (2×2^n) -Matrix D konvergieren aber nach Satz 3.9 quadratisch gegeneinander, sodass $N^{-1}(D(:, j) - D(:, l))$ schließlich quadratisch gegen Null geht, was aber zu zeigen war. \square

Legen wir uns nun zum Abschluss dieses Abschnitts noch die Frage vor, ob unser Verfahren unabhängig von der Wahl der Koordinaten ist, wie es z.B. das Newtonverfahren ist. Sofort klar ist, dass dies nicht der Fall sein kann, weil einerseits die Berechnung der Vektoren $N^{\{k\}}$ von der Wahl der Koordinaten abhängen und andererseits nach Berechnung der Menge E aus Satz 3.4 auf das einschließende Intervall von E subdividiert wird. Und welches Intervall nun das einschließende ist, hängt natürlich elementar von den Koordinaten ab. Fatal wäre nun aber, wenn der Konvergenzfaktor durch die Koordinatenwahl beliebig groß werden kann. Dies ist aber nicht der Fall. Die Konvergenz ist tatsächlich quadratisch z.B. in der Summe $S = \sum_k (b_k - a_k)$ des Intervalls $I = [a_1, b_1] \times [a_n, b_n]$ um die Nullstelle $x^0 \in I$. Dies sehen wir unmittelbar ein, wenn wir noch einen genaueren Blick auf den Beweis von Satz 3.9 werfen. Auf der rechten Seite von $\bar{p}_k(x) - A(x^0 - x)$ addieren sich drei Größen. Die quadratische Konvergenz beiden letzten Terme $(p_k(x) - p'_k(x^0)(x^0 - x)) + (p'_k(x^0) - A) \cdot (x^0 - x)$ hängt offenbar nur von S ab. Die Differenz $(\bar{p}_k(x) - p_k(x))$ ist aber ebenfalls eine Größe, die nur von den zweiten Ableitungen eines Bézierpolynoms abhängen (siehe dazu auch 2.12).

3.1.2 Eine hocheffiziente Variante für einen Spezialfall

In diesem Abschnitt wollen wir die Behandlung spezieller polynomialer Gleichungssysteme $p : [0, 1]^n \rightarrow \mathbb{R}^n$ besprechen, bei denen sich der Rechenaufwand wesentlich reduzieren lässt. Die gewonnenen Erkenntnisse der weiteren Betrachtungen fassen wir dann

in Satz 3.11 zusammen. Wir begleiten unser Vorgehen mit einem einfachen Beispiel. Anschließend verallgemeinern wir unsere Überlegungen noch etwas und formulieren Satz 3.12.

Falls in Monomdarstellung keine Produkte verschiedener Variablen auftreten, so lässt sich die spezielle Struktur ausnutzen. Produktfreiheit zu verlangen entspricht einer Dimensionsreduzierung des Vektorraums der möglichen Gleichungen. Entsprechend der Reduzierung der Freiheitsgrade bei der Formulierung einer Gleichung können wir die selbe Reduktion bei der Behandlung dieser Gleichung erzielen.

Wir nennen nun eine Gleichung $p^{\{k\}} = \pi_k(p)$ für ein $k \in (1 : n)$ produktfrei, falls gilt

$$p^{\{k\}}(x_1, \dots, x_n) = \sum_{j=1}^n p_j^{\{k\}}(x_j). \quad (3.5)$$

Beispielsweise ist die Gleichung

$$p^{\{1\}}(x_1, x_2) = \underbrace{x_1^2}_{=: p_1^{\{1\}}(x_1)} + \underbrace{4x_2^2 - 4}_{=: p_2^{\{1\}}(x_2)}, \quad x_1, x_2 \in [0, 1].$$

produktfrei. Wir bemerken, dass der zusätzliche Term $-2x_1x_2$ die Eigenschaft (3.5) zerstören würde.

Wie wir sehen werden, genügt es für unseren Algorithmus, die univariaten Kontrollpunkte der Bezierkurven $p_j^{\{k\}}$ zu betrachten. Seien also für entsprechendes $m_j^{\{k\}}$ die Kontrollpunkte $P_j^{\{k\}}$ so, dass $p_j^{\{k\}}(x_j) = B_{m_j^{\{k\}}}(x_j)P_j^{\{k\}}$. Nun sieht man direkt, dass die Kontrollpunkte $P_I^{\{k\}}$ für $p^{\{k\}}$ sich als Summe entsprechender Kontrollpunkte der $P_j^{\{k\}}$ ergeben, dass nämlich gilt:

$$P_I^{\{k\}} = \sum_j P_j^{\{k\}}(i_j).$$

Für unsere obige Gleichung $p_1^{\{k\}}(x_1, x_2)$ sind mit $P_1^{\{k\}} = [0; 0; 1]$ und $P_2^{\{k\}} = [-4; -4; 0]$ die Kontrollpunkte $P^{\{1\}}$ gegeben durch

$$P^{\{1\}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} -4 & -4 & 0 \\ -4 & -4 & 0 \\ -4 & -4 & 0 \end{bmatrix} = \begin{bmatrix} -4 & -4 & 0 \\ -4 & -4 & 0 \\ -3 & -3 & 1 \end{bmatrix}$$

Eine direkte Folge hieraus ist, dass sich der Normalenvektor für die Einschluss-Hyperebene gemäß Satz 3.3 als

$$N^{\{k\}} = [P_1^{\{k\}}(1) - P_1^{\{k\}}(m_1^{\{k\}}); \dots; P_n^{\{k\}}(1) - P_n^{\{k\}}(m_n^{\{k\}})]$$

ergibt, in unserem Beispiel erhalten wir also $N^{\{1\}} = [0-1; -4-0; 1] = [-1; -4; 1]$. Um die Größen $d_{\min}^{\{k\}}$ und $d_{\max}^{\{k\}}$ zu erhalten, muss nun das Minimum und Maximum des Skalarprodukts des Normalenvektors mit den Kontrollpunkten zum Graphen berechnet werden. Die Kontrollpunkte zum Graphen sind nun

$$\mathcal{P}^{\{k\}}(I) = \begin{bmatrix} (i_1 - 1)/(m_1^{\{k\}} - 1) \\ \vdots \\ (i_n - 1)/(m_n^{\{k\}} - 1) \\ P^{\{k\}}(I) \end{bmatrix} = \begin{bmatrix} (i_1 - 1)/(m_1^{\{k\}} - 1) \\ \vdots \\ (i_n - 1)/(m_n^{\{k\}} - 1) \\ P_1^{\{k\}}(i_1) + \dots + P_n^{\{k\}}(i_n) \end{bmatrix}.$$

Folglich gilt für das Minimum

$$\begin{aligned} d_{\min}^{\{k\}} &= \min_{I=(i_1, \dots, i_n) \in (1:m_1^{\{k\}}) \times \dots \times (1:m_n^{\{k\}})} \langle N^{\{k\}}, \mathcal{P}^{\{k\}}(I) \rangle \\ &= \min_I \left(N^{\{k\}}(1) \frac{i_1 - 1}{m_1^{\{k\}} - 1} + \dots + N^{\{k\}}(n) \frac{i_n - 1}{m_n^{\{k\}} - 1} + 1(P_1^{\{k\}}(i_1) + \dots + P_n^{\{k\}}(i_n)) \right) \\ &= \min_I \left(N^{\{k\}}(1) \frac{i_1 - 1}{m_1^{\{k\}} - 1} + P_1^{\{k\}}(i_1) \right) + \dots + \min_I \left(N^{\{k\}}(n) \frac{i_n - 1}{m_n^{\{k\}} - 1} + P_n^{\{k\}}(i_n) \right) \\ &= \min_{i_1 \in (1:m_1^{\{k\}})} \left(N^{\{k\}}(1) \frac{i_1 - 1}{m_1^{\{k\}} - 1} + P_1^{\{k\}}(i_1) \right) + \dots \\ &\quad + \min_{i_n \in (1:m_n^{\{k\}})} \left(N^{\{k\}}(n) \frac{i_n - 1}{m_n^{\{k\}} - 1} + P_n^{\{k\}}(i_n) \right). \end{aligned}$$

Entsprechendes gilt natürlich auch für das Maximum, wenn man überall min durch max ersetzt. Dass das Auseinanderziehen des Minimums und Maximums möglich ist, haben wir letztlich der Tatsache zu verdanken, dass die Identität ebenfalls die Eigenschaft (3.5) besitzt. In unserem Beispiel erhalten wir

$$\begin{aligned} d_{\min}^{\{1\}} &= \min_{I=(i_1, i_2) \in (1:3) \times \dots \times (1:3)} \langle [-1; -4; 1], [(i_1 - 1)/2; (i_2 - 1)/2; P_1^{\{1\}}(i_1) + P_2^{\{1\}}(i_2)] \rangle \\ &= \min_{i_1 \in (1:3)} \left((-1)(i_1 - 1)/2 + P_1^{\{1\}}(i_1) \right) + \min_{i_2 \in (1:3)} \left((-4)(i_2 - 1)/2 + P_2^{\{1\}}(i_2) \right) \\ &= \frac{-1}{2} + (-6) = -6\frac{1}{2} \\ d_{\max}^{\{1\}} &= -4. \end{aligned}$$

Die Berechnung des Bereichs möglicher Nullstellen läuft nun nach dem bekannten Schema ab, sie kann nicht abgekürzt werden. Wir erhalten dann gemäß Satz 3.4 die beiden Vektoren A und B , sodass Nullstellen nur in $[A(1), B(1)] \times [A(n), B(n)]$ möglich sind. Sehr wohl abgekürzt werden kann aber die sich dann anschließende Subdivision. Es müssen nämlich nur die Kontrollpunktvektoren $P_j^{\{k\}}$ auf $[A(j), B(j)]$ subdividiert werden.

Fassen wir das bis hierher Gesagte nun im folgenden Satz zusammen.

Satz 3.11 Sei $p : [0, 1]^n \rightarrow \mathbb{R}^n$ ein Bézierpolynom. Sei ferner für ein $k \in (1 : n)$ die Ordnung $M^{\{k\}} = (m_1^{\{k\}}, \dots, m_n^{\{k\}})$ für $\pi_k(p)$ und gelte

$$p^{\{k\}}(x_1, \dots, x_n) = \sum_{j=1}^n p_j^{\{k\}}(x_j). \quad (3.5)$$

Seien die Kontrollpunkte $P_j^{\{k\}} \in \mathbb{R}^{m_j}$ so, dass $p_j^{\{k\}}(x_j) = B_{m_j^{\{k\}}}(x_j)P_l^{\{k\}}$. Dann ist gemäß Satz 3.3

$$N^{\{k\}} = [P_1^{\{k\}}(1) - P_1^{\{k\}}(m_1^{\{k\}}); \dots; P_n^{\{k\}}(1) - P_n^{\{k\}}(m_n^{\{k\}}); 1]$$

und gemäß Satz 3.2

$$d_{\min}^{\{k\}} = \sum_{j=1}^n \min_{l \in 1:m_j^{\{k\}}} \left(N^{\{k\}}(j) \frac{l-1}{m_j^{\{k\}}-1} + P_j^{\{k\}}(l) \right),$$

bzw.

$$d_{\max}^{\{k\}} = \sum_{j=1}^n \max_{l \in 1:m_j^{\{k\}}} \left(N^{\{k\}}(j) \frac{l-1}{m_j^{\{k\}}-1} + P_j^{\{k\}}(l) \right).$$

Die Subdivision der Kontrollpunkte für die Gleichung $p^{\{k\}}(x)$ auf die Box $[A(1), B(1)] \times \dots \times [A(n), B(n)]$ kann komponentenweise durchgeführt werden durch Subdivision von $P_j^{\{k\}}$ auf $[A(j), B(j)]$.

Ändern wir unseren Blickwinkel ein klein wenig und verallgemeinern wir die in Satz 3.11 gewonnenen Erkenntnisse, so erhalten wir unmittelbar den folgenden Satz 3.12. Setzt man in ihm $r_j = j$, dann stellt er eine alternative Formulierung von Satz 3.11 dar.

Satz 3.12 Lässt sich die linke Seite einer polynomialen Gleichung

$$p^{\{k\}}(x_1, \dots, x_n) = 0$$

aufspalten in eine Summe von polynomialen Termen, welche von jeweils unterschiedlichen Variablen abhängen, falls also für $0 = r_0 \leq r_1 \leq \dots \leq r_l \leq n$ gilt:

$$p^{\{k\}}(x_1, \dots, x_n) = \sum_{i=1}^l p_i^{\{k\}}(x_{r_{i-1}+1}, \dots, x_{r_i}) \quad (3.6)$$

so können die einzelnen Terme $p_1^{\{k\}}, \dots, p_l^{\{k\}}$ getrennt voneinander behandelt werden. Es sind also lediglich die Kontrollpunkte für die Terme $p_i^{\{k\}}$ zu speichern, und gemäß

dem Standardvorgehen sind zu diesen Normalenvektoren $N^{\{k\},j}$ samt den Größen $d_{\min}^{\{k\},j}$ und $d_{\max}^{\{k\},j}$ zu bestimmen. Der Normalenvektor $N^{\{k\}}$ zur Gleichung $p^{\{k\}}$ ergibt sich dann als

$$N^{\{k\}} = [N^{\{k\},0}; \dots; N^{\{k\},l}; \text{zeros}(n - r_l, 1)],$$

und die Größen $d_{\min/\max}^{\{k\}}$ ergeben sich als

$$d_{\min}^{\{k\}} = \sum_{j=1}^l d_{\min}^{\{k\},j} \quad \text{bzw.} \quad d_{\max}^{\{k\}} = \sum_{j=1}^l d_{\max}^{\{k\},j}.$$

Die Bestimmung der Box möglicher Nullstellen \hat{E} kann nicht abgekürzt werden. Die anschließende Subdivision hingegen muss lediglich blockweise durchgeführt werden. Das Gesagte bleibt natürlich auch gültig nach Permutation der Variablen, jedoch ist die Möglichkeit, sich dann noch verständlich auszudrücken, nicht mehr gegeben.

Der Beweis birgt nach den Ausführungen zu Satz 3.11 keine weiteren mathematischen Schwierigkeiten und macht lediglich Mühe bei den Bezeichnungen, wir wollen in deshalb nicht angeben.

Wir wollen noch darauf hinweisen, dass Lemma 3.5 letztlich ein Spezialfall von Satz 3.12 darstellt.

3.1.3 Speicherplatzbedarf und Rechenaufwand des Nullstellen-Algorithmus `Roots`

Bevor man die Routine `Roots` implementiert bzw. aufruft, wäre es wünschenswert, den benötigten Speicherplatzbedarf und die Rechenzeit zu kennen bzw. abzuschätzen. Im Voraus zu wissen, wieviele Boxen untersucht werden müssen, bis die Nullstellen mit der gewünschten Toleranz eingegrenzt sind, ist natürlich nicht möglich. Sehr einfach hingegen ist es, den Aufwand für die Untersuchung einer Box anzugeben. Dies wollen wir in diesem Abschnitt tun. Wie wir sehen werden ist der Speicherplatzbedarf und der Rechenaufwand im Wesentlichen proportional zur Zahl der zur Darstellung benötigten Kontrollpunkte. Die Herleitung der angegebenen Formeln ist eher lästig als schwierig. Deshalb werden wir uns auf die Angabe der Formeln beschränken. Begleiten wir die weiteren Überlegungen mit einem Beispiel, mit dem wir das Gesagte illustrieren können. Sei

$$f(x_1, x_2, x_3, x_4) = \begin{bmatrix} x_1^4 - 3x_1x_2^5 + 2x_2x_3^3 - 2x_3^3x_4^2 \\ x_1^4 - 3x_1x_2^5 + 2x_2x_3^3 \\ x_1^4 \\ x_1^2 + x_2^2 + x_3^2 + x_4^2 - 1 \end{bmatrix} = \begin{bmatrix} p^{\{1\}}(x_1, \dots, x_4) \\ p^{\{2\}}(x_1, \dots, x_4) \\ p^{\{3\}}(x_1, \dots, x_4) \\ p^{\{4\}}(x_1, \dots, x_4) \end{bmatrix}$$

Zwischen Lemma 3.5 und dem Code der Routine `Roots` haben wir den Algorithmus in vier logische Einheiten aufgeteilt, nämlich in die

- (1) Berechnung der Normalenvektoren $N^{\{k\}}$.
- (2) Berechnung der Größen $d_{\min}^{\{k\}}$ und $d_{\max}^{\{k\}}$ zu den Normalenvektoren $N^{\{k\}}$.
- (3) Berechnung der Box \hat{E} bzw. der Vektoren A und B .
- (4) Subdivision auf die Box möglicher Nullstellen \hat{E} .

Die Größen $N^{\{k\}}$, $d_{\min}^{\{k\}}$, $d_{\max}^{\{k\}}$ und die Subdivision sind dabei für jede einzelne Gleichung zu berechnen, während die Box \hat{E} ein mal pro Gleichungssystem zu bestimmen ist. Nun gibt es neben der Einteilung in vier Einheiten noch verschiedene Gattungen von Gleichungssystemen bzw. von Gleichungen. Es sind drei Fälle zu unterscheiden.

- (a) Eine Gleichung $p^{\{k\}}$ hängt von allen Variablen ab und ist kein Spezialfall gemäß Abschnitt 3.1.2.
- (b) Eine Gleichung $p^{\{k\}}$ erfüllt die Voraussetzungen von Lemma 3.5, hängt also nicht von allen Variablen ab. Somit ist eine Reduktion des Speicherplatzbedarfs und der Rechenzeit möglich.
- (c) Eine Gleichung $p^{\{k\}}$ stellt einen Spezialfall gemäß Abschnitt 3.1.2 dar und erlaubt somit eine erhebliche Reduktion des Speicherplatzbedarfs und der Rechenzeit.

In unserem Beispiel fällt $p^{\{1\}}$ in die Gattung (a), $p^{\{2\}}$ in Gattung (b) und $p^{\{3\}}$, $p^{\{4\}}$ fallen in die Gattung (c). Sei $M^{\{k\}}$ der Vektor der Ordnungen zur Gleichung $p^{\{k\}}$. Im Beispiel ist also

$$\begin{aligned}
 M^{\{1\}} &= (5, 6, 4, 3) \\
 M^{\{2\}} &= (5, 6, 4, 1) \\
 M^{\{3\}} &= (5, 1, 4, 3) \\
 M^{\{4\}} &= (3, 3, 3, 3).
 \end{aligned}$$

Nun geben wir den Aufwand an gezählt in „Floating point operations“ bzw. kurz flops. n ist dabei, wie bisher, die Zahl der Veränderlichen. Im Beispiel ist also $n = 4$.

- (a1) $n2^n$ flops.
- (a2) $2n \prod_{j=1}^n m_j^{\{k\}}$ flops.
- (a3) $2n^3 + 4n^2$ flops, da die Invertierung einer Matrix $O(2n^3)$ flops benötigt.
- (a4) $3 \prod_{j=1}^n m_j^{\{k\}} \sum_{i=1}^n (m_i^{\{k\}} - 1)$ flops.

In unserem Beispiel ergeben sich also für die erste Gleichung und für (a1) $4 \cdot 16 = 64$ flops, für (a2) $8 \cdot 360 = 2880$ flops, für (a3) $128 + 64 = 192$ flops und für (a4) $3 \cdot 360 \cdot 14 = 15120$ flops. Der Aufwand für Gleichungen der Gattung (b) ergibt sich wie folgt.

(b1) $s_k 2^{s_k}$ flops, wobei s_k die Zahl der Veränderlichen ist, von denen die Gleichung $p^{\{k\}}$ tatsächlich abhängt.

(b2) $2s_k \prod_{j=1}^n m_j^{\{k\}}$ flops.

(b3) $2n^3 + 4n^2$ flops.

(b4) $3 \prod_{j=1}^n m_j^{\{k\}} \sum_{i=1}^n (m_i^{\{k\}} - 1)$ flops, wobei für Variablen x_j , von denen die Gleichung $p^{\{k\}}$ nicht abhängt, $m_j^{\{k\}} = 1$ ist.

Für die zweite Gleichung ergibt sich als Aufwand für (b1) $3 \cdot 8 = 24$ flops, (b2) $6 \cdot 120 = 720$ flops und (b4) $3 \cdot 120 \cdot 12 = 4320$ flops. Für Gleichungen der Gattung (c) summiert sich offenbar der Aufwand über den Aufwand für die einzelnen Blöcke. Mit den Bezeichnungen r_0, \dots, r_l aus Satz 3.12 ist der Aufwand:

(c1) $\sum_{b=1}^l s_{k,b} 2^{s_{k,b}}$ flops, wobei $s_{k,b} = r_b - r_{b-1}$ die Zahl der Veränderlichen ist, von denen der Block $p_b^{\{k\}}$ abhängt

(c2) $\sum_{b=1}^l 2s_{k,b} \prod_{j=r_{b-1}+1}^{r_b} n m_j^{\{k\}}$ flops.

(c3) $2n^3 + 4n^2$ flops.

(c4) $\sum_{b=1}^l 3 \prod_{j=r_{b-1}+1}^{r_b} m_j^{\{k\}} \sum_{i=r_{b-1}+1}^{r_b} (m_i^{\{k\}} - 1)$ flops.

Für die dritte Gleichung ergibt sich als Aufwand für (c1) $2 + 8 = 10$ flops, (c2) $2 \cdot 5 + 4 \cdot 12 = 58$ flops und für (c4) $3 \cdot 5 \cdot 4 + 3 \cdot 12 \cdot 5 = 240$ flops. Für die vierte Gleichung ergibt sich als Aufwand für (c1) $4 \cdot 2 = 8$ flops, (c2) $4 \cdot 2 \cdot 3 = 24$ flops und für (c4) $4 \cdot 3 \cdot 3 \cdot 2 = 72$ flops. Offenbar ist der Speicherplatzbedarf und der Rechenaufwand im Wesentlichen proportional zur Zahl der zur Darstellung benötigten Kontrollpunkte, zumindest für kleine n . Für große n sind Gleichungen vom Typ (a) nicht mehr handhabbar, weil die Zahl der Kontrollpunkte astronomisch anwächst. Lediglich für Gleichungen vom Typ (b) und (c) sind größere n ($n > 15$) praktikabel. Dann gewinnt der Aufwand für die Invertierung von N in Schritt (3) immer stärker an Bedeutung und dominiert schließlich für große n . In diesem Fall empfiehlt es sich, die Normalenvektoren N nicht jedesmal neu zu berechnen, sondern die Normalenvektoren der letzten Iteration zu verwenden. Man verliert dann allerdings die quadratische Konvergenz. Ist der Konvergenzfaktor jedoch klein genug, wiegt dieser Nachteil nicht schwer.

3.2 Vorgehen mit Einschlussschätzer

Im letzten Abschnitte 3 haben wir komplett ohne die Einschlussschätzer gearbeitet. Wir haben die Kontrollpunkte eines Bézierpolynoms zwischen zwei parallele Ebenen eingeschlossen und nur benutzt, dass das Bézierpolynom sicher zwischen diesen Ebenen verläuft. Das hat eine Vergrößerung der konvexen Hülle bedeutet, die wir aber gerne in Kauf genommen haben, weil trotzdem die quadratische Konvergenz erhalten blieb. Mit den Einschlussschätzern schaffen wir es, anstatt mit einer Vergrößerung mit einer Verfeinerung zu arbeiten. Es ist zu erwarten, dass die neuen Boxen möglicher Nullstellen wesentlich kleiner werden, als sie das im letzten Abschnitt wurden. Allerdings wächst der Arbeitsaufwand pro Schritt und alles, was wir erwarten können, ist eine Verbesserung der quadratischen Konvergenz um einen linearen Faktor. Es ist deshalb durchaus offen, ob wir mit den Einschlussschätzern tatsächlich schneller sind.

Zunächst nehmen wir an, dass alle Gleichungen in den Ordnungen $M^{\{k\}}$ übereinstimmen. Dies kann notfalls durch Graderhöhung der Bézierdarstellung erreicht werden. Man kann zwar auch mit verschiedenen Ordnungen $M^{\{k\}}$ arbeiten, was die Sache aber technisch wesentlich aufwändiger macht, weil dann die Boxen der verschiedenen Gleichungen nicht mehr harmonieren.

Wir stellen zunächst unsere Überlegungen für $n = 2$ und ein konkretes Beispiel an. Alles Gesagte lässt sich dann völlig problemlos verallgemeinern, ohne dass dabei zusätzliche Schwierigkeiten auftauchen.

Sei das Bézierpolynom $p : [0, 1] \rightarrow \mathbb{R}^2$ gegeben wie in Beispiel 3.6 durch die beiden Kontrollpunktmatrizen

$$P^{\{1\}} = \begin{bmatrix} -0.3 & 0.5 & 0.9 & 1.1 & 0.9 \\ -0.2 & 0.6 & 0.9 & 1.1 & 0.9 \\ -0.1 & 0.7 & 0.9 & 1.1 & 0.9 \\ 0.1 & 0.8 & 1.1 & 1.2 & 1.1 \\ 0.1 & 0.9 & 1.2 & 1.3 & 1.1 \end{bmatrix}, \quad P^{\{2\}} = 2P^{\{1\}T}.$$

Wir bestimmen für $P^{\{1\}}$ und $P^{\{2\}}$ untere und obere bilineare Einschüsse und erhalten

als Kontrollpunktmatrizen für die Einschlüsse

$$\begin{aligned}
 U^{\{1\}} &= \begin{bmatrix} -0.30 & 0.36 & 0.77 & 0.96 & 0.90 \\ -0.20 & 0.44 & 0.81 & 0.97 & 0.90 \\ -0.08 & 0.53 & 0.87 & 1.01 & 0.95 \\ 0.04 & 0.64 & 0.97 & 1.10 & 1.04 \\ 0.10 & 0.73 & 1.07 & 1.19 & 1.10 \end{bmatrix} \\
 O^{\{1\}} &= \begin{bmatrix} -0.26 & 0.39 & 0.81 & 0.99 & 0.94 \\ -0.14 & 0.47 & 0.84 & 1.00 & 0.95 \\ -0.02 & 0.57 & 0.91 & 1.05 & 1.00 \\ 0.10 & 0.68 & 1.00 & 1.13 & 1.09 \\ 0.17 & 0.77 & 1.11 & 1.23 & 1.15 \end{bmatrix} \\
 U^{\{2\}} &= \begin{bmatrix} -0.60 & -0.40 & -0.16 & 0.08 & 0.20 \\ 0.72 & 0.87 & 1.06 & 1.27 & 1.46 \\ 1.55 & 1.61 & 1.75 & 1.94 & 2.15 \\ 1.92 & 1.94 & 2.03 & 2.20 & 2.38 \\ 1.80 & 1.80 & 1.91 & 2.08 & 2.20 \end{bmatrix} \\
 O^{\{2\}} &= \begin{bmatrix} -0.52 & -0.29 & -0.03 & 0.20 & 0.34 \\ 0.78 & 0.94 & 1.13 & 1.35 & 1.55 \\ 1.61 & 1.68 & 1.81 & 2.01 & 2.22 \\ 1.98 & 2.00 & 2.09 & 2.27 & 2.45 \\ 1.88 & 1.90 & 2.01 & 2.17 & 2.30 \end{bmatrix} .
 \end{aligned}$$

Zu beachten ist, dass sich für die zweite Gleichung nicht exakt das Doppelte der Transponierten ergibt. Im gegebenen Beispiel mag man dies auf die Rundung zurückführen, tatsächlich ergibt sich aber ein kleiner Unterschied, weil die Einschlusssschätzer gemäß Satz 2.26 nicht symmetrisch sind. Wir haben die Kontrollpunkte zur ersten Gleichung und deren Einschlüsse in Abbildung 15 geplottet.

Wir stellen diese Ergebnisse in Abbildung 16 übersichtlich dar. Vorzeichenwechsel des Bézierpolynoms $p^{\{1\}}$ sind offenbar nur in Bereichen möglich, in denen die angrenzenden acht Einschluss-Kontrollpunkte nicht allesamt das selbe Vorzeichen tragen. Solche Bereiche haben wir mit einem Kreuz markiert.

Gemeinsame Nullstellen von $p^{\{1\}}$ und $p^{\{2\}}$ sind nur in Bereichen möglich, die in beiden Gleichungen markiert wurden. Dies ist offenbar alleine der Bereich $[0, 1/4] \times [0, 1/4]$. Nun könnten wir uns damit zufrieden geben und auf diesen Bereich subdividieren. Dadurch würden wir aber unsere quadratische Konvergenzrate einbüßen. Deshalb versuchen wir, den Bereich möglicher Nullstellen noch genauer abzuschätzen. Wie in den beiden letzten Abschnitten bestimmen wir nun für die beiden Gleichungen getrennt voneinander zwei parallele Ebenen, die sowohl den unteren als auch den oberen Einschluss auf $[0, 1/4]^2$ beinhalten. Dazu mitteln wir $U^{\{i\}}(1 : 2, 1 : 2)$ und $O^{\{i\}}(1 : 2, 1 : 2)$ jeweils miteinander und spannen die Ebene gemäß diesem Mittel und Satz 3.3 auf. Die

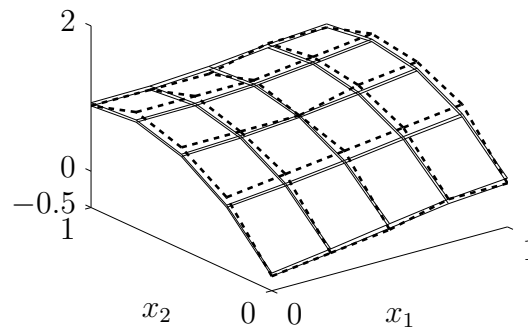


Abbildung 15: Kontrollstruktur und oberer/unterer Einschluss mit Einschlusschätzern, vgl. Abbildung 14 ohne Einschlusschätzer

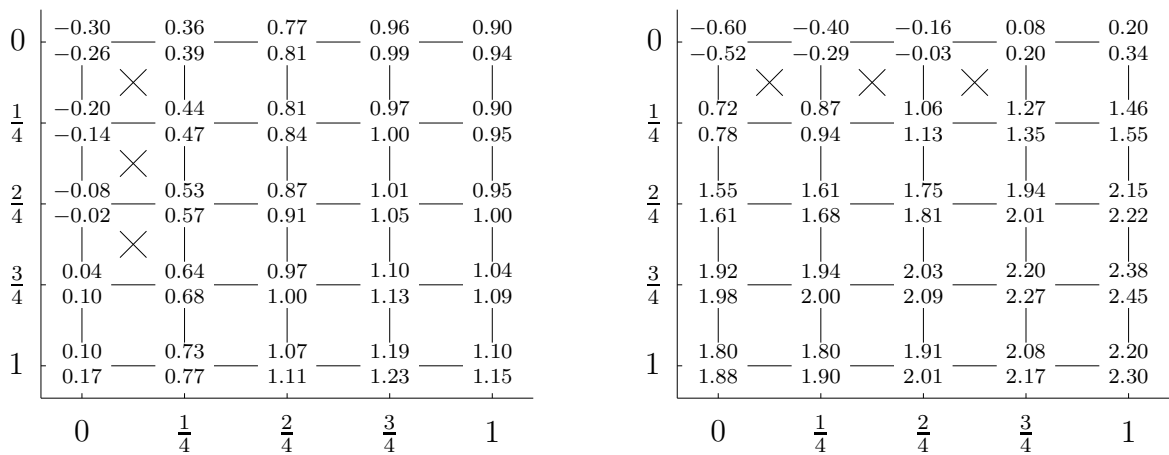


Abbildung 16: Einschlüsse des Bézierpolynoms

Abstände zum Normalenvektor bestimmen wir dann so, dass die unteren und oberen Einschlüsse zwischen den erhaltenen Ebenen liegen. Anschließend fahren wir genau so fort wie in den beiden letzten Abschnitten, d.h. wir schneiden die erhaltenen Bereiche miteinander und subdividieren auf das Gebiet möglicher Nullstellen. In unserem Beispiel erhalten wir als gemittelte Kontrollpunkte für den Bereich $[0, 0.25]^2$ dann

$$\begin{aligned} (U^{\{1\}}(1 : 2, 1 : 2) + O^{\{1\}}(1 : 2, 1 : 2)) / 2 &= \begin{bmatrix} -0.28 & 0.38 \\ -0.17 & 0.45 \end{bmatrix} \\ (U^{\{2\}}(1 : 2, 1 : 2) + O^{\{2\}}(1 : 2, 1 : 2)) / 2 &= \begin{bmatrix} -0.56 & -0.34 \\ 0.75 & 0.91 \end{bmatrix} \end{aligned}$$

und somit als Normalenvektoren $N^{\{1\}} = [-0.37, -2.56, 1]$ bzw. $N^{\{2\}} = [-5.12, -0.74, 1]$ und als zugehörige Abstände bezüglich dieser Normalenvektoren $d_{\min}^{\{1\}} = -0.30$, $d_{\max}^{\{1\}} =$

-0.24 , $d_{\min}^{\{2\}} = -0.60$, $d_{\max}^{\{1\}} = -0.47$. Berechnen wir nun die Box B gemäß Satz 3.4, so erhalten wir $B = [0.08, 0.11] \times [0.08, 0.11]$. Anschließend subdividieren wir auf diese Box B und fahren sukzessive wie beschrieben fort. Wenn wir die neue Box mit der in Beispiel 3.6 berechneten vergleichen, so sehen wir, dass sie wesentlich kleiner ist und somit der Bereich möglicher Nullstellen deutlich schärfer eingegrenzt werden konnte. Das allgemeine Vorgehen wird aus dem oben gegebenen Beispiel fast völlig klar. Nur einen Teilschritt wollen wir nochmals näher kommentieren. Wir haben festgestellt, dass gemeinsame Nullstellen der Gleichungen $\pi_1(p)$ und $\pi_2(p)$ nur auf einem einzigen Bereich möglich waren und haben dann unsere Betrachtungen alleine auf diesen Bereich konzentriert. Selbst asymptotisch können wir nicht davon ausgehen, dass lediglich ein einziger Bereich als Nullstellenbereich in Frage kommt. Wenn nämlich eine Nullstelle auf die Bereichsgrenzen fällt, so werden mindestens die angrenzenden Bereiche von allen Gleichungen markiert. Deshalb verfahren wir wie oben beschrieben, falls die markierten Bereiche allesamt im Trägergebiet einer einzigen multivariaten Hutfunktion liegen, falls also pro Dimension höchstens zwei Segmente als Bereich möglicher Nullstellen übrig bleiben (siehe Abbildung 17, linkes Bild). Ansonsten, wenn der Bereich möglicher Nullstellen größer ist, subdividieren wir sofort ohne weitere Betrachtungen auf den von diesen Bereichen aufgespannten Bereich (siehe Abbildung 17, rechtes Bild. Wir subdividieren hier auf den Bereich $[0, 0.5] \times [0.25, 1]$).

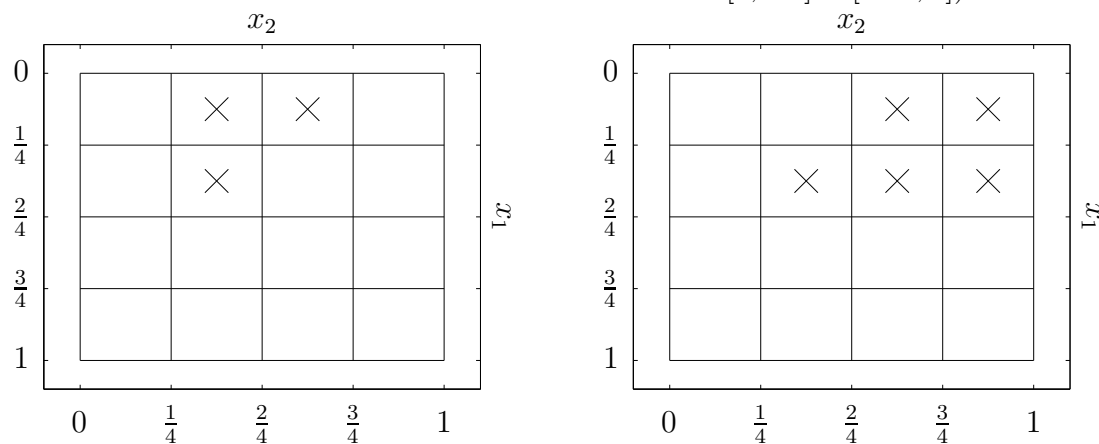


Abbildung 17: Von jeweils zwei Gleichungen gemeinsam markierte Bereiche möglicher Nullstellen, zwei Beispiele.

3.3 Vergleich der Ansätze mit bzw. ohne Einschlussschätzer

Zunächst ist zu vermuten, dass der Ansatz mit Einschlussschätzer zwar einen Mehraufwand bei der Implementation bedeutet, dass er aber die Rechenzeiten erheblich verkürzt. Diese intuitive Erwartung hat unsere Untersuchungen in Richtung der Einschlussschätzer motiviert. Bei vergleichenden Tests in MATLAB zeigt sich aber, dass keinem der beiden Ansätze der generelle Vorzug erteilt werden kann. Die Performance

hängt stark vom vorgelegten Problem ab. Durch Verwendung der Einschlussschätzer ergibt sich allerdings lediglich dann eine wesentliche Reduzierung der benötigten Floating-point-operations (kurz Flops), wenn mindestens trivariate Probleme mit stark oszillierenden Kontrollpunkten angegangen werden. Z.B. reduziert sich die Zahl der Flops für $n = 4$ bei zwischen -1 und 1 gleichverteilten Kontrollpunkten zur Ordnung $[12, 12, 12, 12]$ um etwa 70%.

Natürlich werden bei der Arbeit mit Einschlussschätzer insgesamt wesentlich weniger Boxen untersucht, schließlich sind ja auch unsere Einschlüsse deutlich schärfer. Aber die Berechnung dieser schärferen Einschlüsse bedeuten einen klaren Mehraufwand.

Warum ist nun der einfache Algorithmus so gut? Zunächst ist die Anzahl der Flops in einem Schritt bei der Arbeit mit Einschlussschätzer wesentlich höher als ohne. Wenn nun am Ende der Untersuchung einer Box mit den Einschlussschätzern sich herausstellt, dass Nullstellen überall möglich sind, so wäre dieses ungünstige Ergebnis auch ohne Einschlussschätzer billiger zu haben gewesen. Dies wird aber in der Regel der Fall sein, solange die Funktion entweder noch mehrere Nullstellen in der untersuchten Box besitzt oder wenn in der Nähe von Nullstellen noch zweite partielle Ableitungen mit wechselndem Vorzeichen auftreten. Gibt es keine Vorzeichenwechsel bei den zweiten Differenzen, so greift die quadratische Kontraktion der Boxen, sowohl bei den Einschlussschätzern als auch ohne diese. In diesem Fall verbessern wir die quadratische Konvergenz um einen linearen Faktor auf Kosten von linearem Mehraufwand in jedem Schritt. Dies aber führt insgesamt auch zu Mehrarbeit. Der einzige Fall, in dem die Einschlussschätzer gewinnen ist der, wenn die zweiten Differenzen das Vorzeichen wechseln, die zweiten Ableitungen aber tatsächlich nur ein Vorzeichen besitzen. Bei der Arbeit mit Einschlussschätzern wird dies nämlich im Wesentlichen erkannt und somit haben wir einmal quadratische Kontraktion und das andere mal gar keine Kontraktion.

Insgesamt jedoch erweist sich die Arbeit mit Einschlussschätzern als wenig vorteilhaft. Erstens verkürzen sie nur in wenigen Fällen die Rechenzeit, vielmehr verlängern sie diese meist sogar deutlich. Zweitens lässt sich der Algorithmus ohne Einschlussschätzer wesentlich knapper formulieren und drittens erfordert die Implementation nicht das theoretische Verständnis der Einschlussschätzer, was besonders Lesern zugute kommen dürfte, deren Spezialgebiet die Béziertheorie nicht umfassen. Unsere intuitive Erwartung der Überlegenheit der Einschlussschätzer hat sich also nicht bestätigt. Das Arbeiten mit Einschlussschätzern besitzt zwar den Charme, dass wir die Werte von polynomialen Funktionen sehr genau abschätzen können, was sich bei höherdimensionalen Problemen mit stark oszillierenden Funktionen tatsächlich als vorteilhaft erweist. Das Arbeiten ohne Einschlussschätzer hingegen hat den Vorteil, dass der gesamte Algorithmus relativ einfach ist und in wenigen hundert Zeilen formuliert werden kann und sich dabei als höchst effizient und robust erweist.

Man kann nun versuchen, mit den Einschlussschätzern zu beginnen und, sobald die quadratische Konvergenz greift, umzuschalten auf den Algorithmus ohne die Ein-

schlusschätzer. Diese Variante haben wir jedoch nicht verfolgt.

Zwei grundsätzliche Probleme tauchen bei der Arbeit mit Einschlussschätzern auf. Erstens sollten, wie bereits erwähnt, die Ordnungen $M^{\{k\}}$ in den n Gleichungen übereinstimmen. Ist dies nicht der Fall, wird die Implementierung nochmals komplizierter und die Performance geht deutlich zurück. Das zweite Problem ist das folgende. Liegen die Nullstellenmengen von zwei Gleichungen um eine gemeinsame Lösung fast aufeinander, was von einer fast-singulären Ableitungsmatrix an der Nullstelle impliziert wird, so werden bei der Markierung der Patches stets dieselben Patches in beiden Gleichungen als mögliche Nullstellenmengen erkannt. Dies führt dazu, dass eben nicht nur ein oder zwei Patches für die nächste Box übrig bleiben, sondern stets ein ganzer Streifen, was zu einer fortgesetzten Halbierung führt. In diesem Fall geht die quadratische Kontraktion der Boxen ganz verloren, was natürlich sehr problematisch ist.

3.4 Lösen von polynomialen Gleichungssystemen auf ganz \mathbb{R}^n

Im Fall von polynomialen Gleichungssystemen $p : \mathbb{R}^n \rightarrow \mathbb{R}^n$ ist es möglich, alle Nullstellen von p auf \mathbb{R}^n zu bestimmen. Sei zunächst $n = 1$. Wir können gemäß Abschnitt 3.1 alle Nullstellen von p auf $[-1, 1]$ bestimmen. Mit $p(x) = \sum_{i=0}^{m-1} \alpha_i x^i$ ist nun durch Multiplikation mit x^{m-1} :

$$p\left(\frac{1}{x}\right) = \sum_{i=0}^{m-1} \alpha_i \left(\frac{1}{x}\right)^i = 0 \quad \Leftrightarrow \quad \sum_{i=0}^{m-1} \alpha_i x^{m-1-i} = 0 =: q(x).$$

Wenn nun \tilde{x} eine Nullstelle des Polynoms q in $[-1, 1]$ ist, so ist offenbar $1/\tilde{x}$ eine Nullstelle von p in $\mathbb{R} \setminus (-1, 1)$. Durch Lösung der beiden Gleichungssysteme $p(x) = 0$ und $q(x) = 0$ für $x \in [-1, 1]$ lassen sich so also alle Nullstellen von p auf \mathbb{R} bestimmen. Für $n > 1$ kann nun entsprechend verfahren werden. Ersetzt man in $p(x_1, \dots, x_n) = 0$ nun x_{k_1}, \dots, x_{k_j} durch $1/x_{k_1}, \dots, 1/x_{k_j}$, so erhält man nach Multiplikation mit $x_{k_1}^{m_{k_1}-1}, \dots, x_{k_j}^{m_{k_j}-1}$ wieder ein polynomialen Gleichungssystem, das auf $[-1, 1]^n$ zu lösen ist. Substituiert man in den Lösungen $(\tilde{x}_1, \dots, \tilde{x}_n)$ wieder durch Kehrwertbildung zurück, so erhält man die Nullstellen von p . Insgesamt sind die Substitutionen mit anschließendem Lösen des erhaltenen Gleichungssystems durchzuführen für alle möglichen Teilmengen $\{k_1, \dots, k_j\}$ von $(1 : n)$. Insgesamt sind also 2^n Gleichungssysteme zu lösen.

3.5 Anwendungsbeispiele

Wir wollen im Folgenden einige Beispiele untersuchen, die ersten drei wurden ebenfalls in [SP93], das vierte in [K96] behandelt.

3.5.1 Schnitt von zwei ebenen implizierten Kurven

Untersuchen wir zunächst ein ganz einfaches Beispiel. Schneiden wir einen kreisförmigen Bogen mit einem ellipsenförmigen,

$$x^2 + y^2 - 1 = 0, \quad \frac{x^2}{4} + 4y^2 - 1 = 0, \quad 0 \leq x \leq 1, \quad 0 \leq y \leq 1.$$

Die beiden Bögen schneiden sich nur in $(2/\sqrt{5}, 1/\sqrt{5})$. Lassen wir unsere beiden Algorithmen mit einer Toleranz von 10^{-8} laufen, so erhalten wir ohne Einschlusschätzer nach 7 untersuchten Boxen und 3486 Flops das Ergebnis. Mit Einschlusschätzer sind wir nach 5 untersuchten Boxen und 4842 Flops am Ziel.

3.5.2 Schnitt dreier parametrisierter Flächen im \mathbb{R}^3

Wir bestimmen zu drei parametrisierten Flächen f_1, f_2, f_3 im \mathbb{R}^3 deren Schnittpunkte. Es sei für $x, y, s, t, u, v \in [0, 3]$

$$f_1(x, y) = \begin{bmatrix} x^2y - xy^2 + x^3 \\ 2x - 3xy^3 + 3y^2 \\ 2x^3 - 3xy^2 + y^2 - y \end{bmatrix}, \quad f_2(s, t) = \begin{bmatrix} s + t - s^2t - st^3 + st \\ s - t - s^3t - st^3 + s^2t + 3 \\ t + s^2t - s^3t^2 - 2 \end{bmatrix},$$

$$f_3(u, v) = \begin{bmatrix} 2u^2v^3 - uv^2 \\ uv^2 + u^3v - u^2v + 1 \\ uv^3 - u + v - u^3 + v^3 - 2 \end{bmatrix}.$$

Die Schnittpunkte erhalten wir, indem wir die gemeinsamen Lösungen von $f_1 = f_2$ und $f_1 = f_3$ suchen. Somit erhalten wir ein polynomiales Gleichungssystem in sechs Variablen. Dieses Gleichungssystem ist ein Spezialfall gemäß Satz 3.12 und kann dementsprechend behandelt werden. Nach der Untersuchung von 7512 Boxen ohne Einschlusschätzer mit insgesamt 58069937 Flops sind die Lösungen

$$\begin{aligned} (x, y, s, t, u, v) &= (0.99, 1.01, 1.08, 0.98, 1.00, 0.99) \\ (x, y, s, t, u, v) &= (1, 1, 1, 1, 1, 1) \\ (x, y, s, t, u, v) &= (1, 1, 0, 1, 1, 1) \end{aligned}$$

mit Genauigkeit 10^{-12} bestimmt.

3.5.3 Wilkinson-Polynom

Im nächsten Beispiel versuchen wir, das folgende univariate Polynom vom Grad 20 zu lösen. Wilkinson zeigte in [W63], dass es extrem schlecht konditioniert ist:

$$P_{20}(x) = (x - 1/20)(x - 2/20) \cdots (x - 20/20) \quad , x \in [0, 1]$$

Die Nullstellen dieses Polynoms können mit wesentlich kleinerem Fehler bestimmt werden, wenn man es in Bézierform anstatt der Monomform löst (siehe [FR87]). Allerdings ist die Transformation in Bézierform extrem schlecht konditioniert. Um diesem Problem aus dem Wege zu gehen, haben wir es in exakter rationaler Arithmetik transformiert, es dann in floating-point-Darstellung umgewandelt und mit unseren Algorithmus gelöst. Wir erhalten bei einer Toleranz von 10^{-5} ohne Einschlusschätzer nach 113 untersuchten Boxen und 146561 Flops die 20 Lösungen.

3.5.4 Kritische Punkte einer algebraischen Kurve

Ein Standard-Problem in der CAGD ist die Auswertung von implizit gegebenen Funktionen. Verfolgt man Lösungszweige, so ist es notwendig, die kritischen Punkte ($df/du = df/dv = 0$) zu kennen. Betrachten wir als Beispiel die Funktion

$$f(u, v) = -64v^4 + 128v^3 - 96u^2v^2 + 140uv^2 - 139v^2 + 96u^2v - 140uv + 75v - 96u^4 + 276u^3 - 313u^2 + 165u - 36 \quad u, v \in [0, 1].$$

Die beiden partiellen Ableitungen ergeben ein polynomiales Gleichungssystem. Lösen wir dieses mit unseren Algorithmen bei einer vorgegebenen Toleranz von 10^{-8} , so erhalten wir ohne Einschlusschätzer nach 186191 Flops und 140 untersuchten Boxen insgesamt alle 9 Lösungen. Mit Einschlusschätzern werden 120 Boxen untersucht und 497857 Flops sind erforderlich. Die Lösungen des Gleichungssystems und die Unterteilungsstruktur haben wir in Abbildung 18 geplottet.

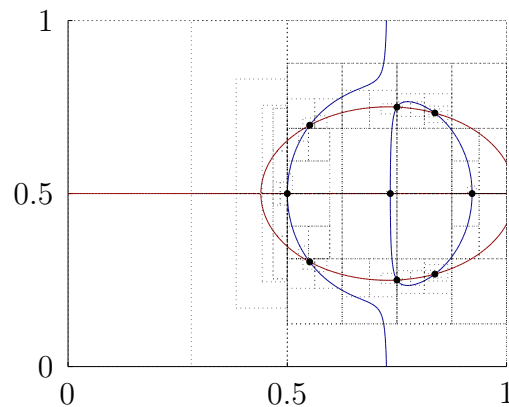


Abbildung 18: Unterteilungsstruktur des Nullstellen-Algorithmus bei einer polynomia- len Gleichung in zwei Variablen. Untersuchte Boxen gepunktet, Lösungen der ersten und Lösungen der zweiten Gleichung durchgezogen, gemeinsame Lösungen mit Kreis markiert.

3.5.5 Quadratisches System mit variabler Dimension

Wir lösen das zyklische, quadratische Gleichungssystem mit variabler Dimension

$$(2x_l - 1.1)^2 + 2x_{l+1} - 1.1 = 0, \quad l = 1 : d, \quad x_{d+1} = x_1, \quad x_l \in [0, 1].$$

Die beiden einzigen reellen Lösungen sind

$$x_1 = (0.05, \dots, 0.05), \quad x_2 = (0.55, \dots, 0.55).$$

Wir haben das System für $d \in (2 : 7)$ mit Toleranz 10^{-16} untersucht. Nach jeweils 25, 45, 82, 185, 386, 821 Schritten wurden ohne die Einschlusschätzer die beiden Nullstellen gefunden.

3.5.6 Konvexe Hülle eines polynomial berandeten Gebiets in \mathbb{R}^2

Gegeben sei ein Gebiet, das durch eine geschlossene glatte polynomiale Kurve p ohne Selbstdurchdringungen berandet sei. Also z.B.

$$p(t) = \left[\begin{array}{c} 56t^2 + 224t^3 - 980t^4 + 1064t^5 - 252t^6 - 224t^7 + 112t^8 \\ 8t - 112t^2 + 616t^3 - 1680t^4 + 2352t^5 - 1568t^6 + 328t^7 + 56t^8 \end{array} \right], \quad t \in [0, 1].$$

Gesucht sei der konvexe Abschluss dieser Menge (siehe Abbildung 19). Dazu müssen wir Punktpaare auf p mit jeweils der selben Tangente finden. Alternativ können wir auch sagen, dass wir Punktpaare aufspüren müssen, deren Ableitungen jeweils parallel zu ihrer Differenz sind. Dies können wir nochmals umformulieren. Gesucht sind Punktpaare, sodass deren Normale zur Ableitung jeweils senkrecht auf die Differenz steht. Somit haben wir eine Formulierung gefunden, sodass das Problem im Kontext dieser Arbeit gestellt und gelöst werden kann. Gesucht sind also (t, s) , sodass

$$\begin{aligned} \left\langle p'(t), \begin{pmatrix} p^{\{2\}}(t) - p^{\{2\}}(s) \\ p^{\{1\}}(s) - p^{\{1\}}(t) \end{pmatrix} \right\rangle &= 0 \\ \left\langle p'(s), \begin{pmatrix} p^{\{2\}}(t) - p^{\{2\}}(s) \\ p^{\{1\}}(s) - p^{\{1\}}(t) \end{pmatrix} \right\rangle &= 0 \end{aligned} \quad (3.7)$$

Wenn p nun eine polynomiale Kurve vom Grad m ist, so sind die beiden linken Seiten des Gleichungssystems polynomiale Funktionen in den zwei Veränderlichen (t, s) vom Grad $(2m - 1, m)$ bzw. $(m, 2m - 1)$. Für welche (t, s) muss dieses Gleichungssystem nun gelöst werden? Für $\{(t, s) \in [0, 1]^2 : t \neq s\}$. Somit haben wir nun aber ein Parametergebiet, das nicht zu unserem Algorithmus passt. Deshalb substituieren wir s durch $t + h$ und lösen dieses Gleichungssystem. Die erste Gleichung ist vom Grad $(2m - 1, m)$, die zweite Gleichung vom Grad $(2m - 1, 2m - 1)$. Wir erhalten nun das Parametergebiet $\{(t, h) \in [0, 1]^2 : h > 0 \text{ und } t + h \leq 1\}$. Scheinbar hat sich nicht

viel verbessert. Jedoch können wir das Parametergebiet auf $\{(t, h) \in [0, 1]^2 : h > 0\}$ ausdehnen und am Ende unserer Rechnung diejenigen Punktpaare verwerfen, für die $t + h > 1$. Doch noch immer genügt das Parametergebiet nicht unseren Vorgaben, schließlich sind nur kompakte Intervalle als Definitionsgebiet zulässig.

Wir bemerken nun, dass gemeinsame Tangenten nur an zwei Punkten möglich sind, zwischen denen die Krümmung mindestens zweimal das Vorzeichen wechselt. Suchen wir also zuerst alle $t_k \in [0, 1]$, sodass $p(t_k)$ dort verschwindende Krümmung besitzt. Dann bestimmen wir den minimalen Abstand zwischen zwei solchen t_k und setzen diesen gleich h_{\min} . Lösungen unseres Gleichungssystems sind nun für $(t, h) \in [0, 1] \times [h_{\min}, 1]$ zu suchen. Dass die Krümmung verschwindet ist äquivalent dazu, dass die zweite Ableitung parallel zur ersten ist oder alternativ: Die Krümmung verschwindet dort, wo die Normale zur ersten Ableitung senkrecht zur zweiten Ableitung steht, also dort, wo

$$\left\langle p'(t), \begin{pmatrix} \pi_2(p''(t)) \\ -\pi_1(p''(t)) \end{pmatrix} \right\rangle = 0. \quad (3.8)$$

Lösen wir also zunächst die Gleichung (3.8). Bei der Arbeit ohne Einschlusschätzer erhalten wir nach 31 untersuchten Boxen und 19502 Flops die in Abbildung 19 rot eingetragenen Krümmungswechsel bei $t = 0.0971, 0.2750, 0.7294, 0.9018$ (mit Genauigkeit 10^{-12}). Lösen wir diese Gleichung mit Einschlusschätzer, so erhalten wir das Ergebnis nach 22 untersuchten Boxen und 40446 Flops. Es ergibt sich also $h_{\min} = 0.1724$.

Nun können wir das Gleichungssystem (3.7) lösen. Ohne Einschlusschätzer erhalten wir dann nach 175 untersuchten Boxen und $4.9 \cdot 10^6$ flops das Ergebnis $(t, h) = (0.6460, 0.0631), (t, h) = (0.2934, 0.2995)$ (mit Genauigkeit 10^{-12}). Lösen wir dieses Gleichungssystem mit Einschlusschätzern, so erhalten wir das Ergebnis nach 30 untersuchten Boxen und $4.3 \cdot 10^6$ Flops.

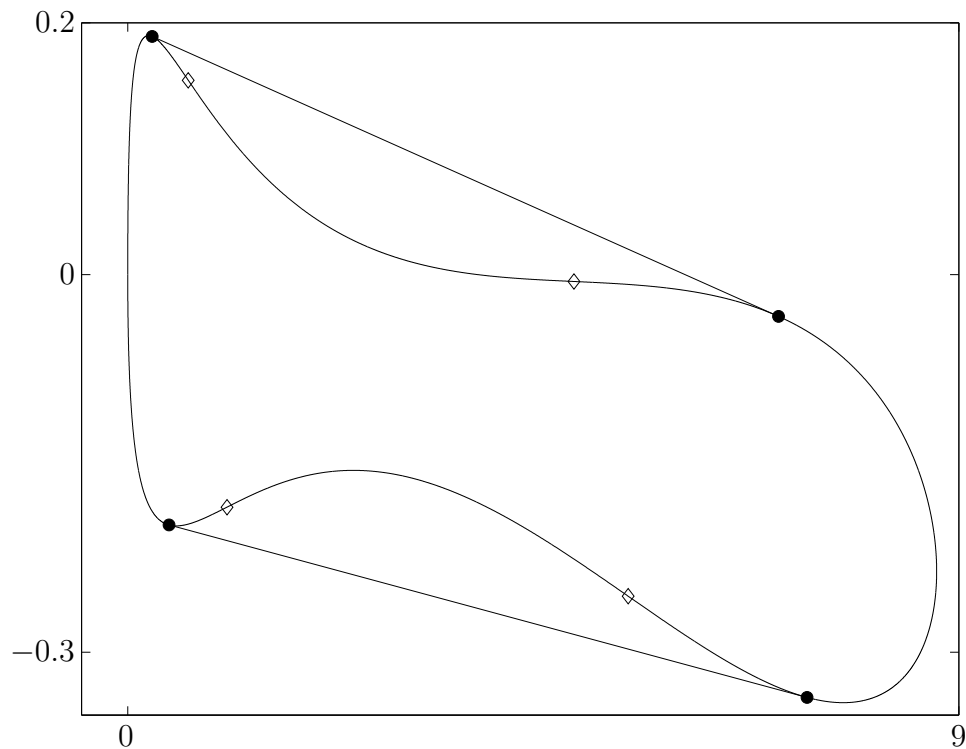


Abbildung 19: Konvexe Hülle eines polynomial berandeten Gebiets

4 Nicht-polynomiale Gleichungssysteme

Jetzt stellt sich die Frage, ob wir unser Vorgehen zum Lösen von Gleichungssystemen auch auf nicht-polynomiale Systeme übertragen können. Alles, was wir in Abschnitt 3.1 gebraucht haben, war die Darstellung der Gleichung in Bernstein-Basisfunktionen. Von diesen haben wir benutzt, dass sie eine positive Partition der Eins bilden und dass sich für die Identifikation einer Gleichung mit ihrem Graphen Kontrollpunkte für die Identität bestimmen lassen. Außerdem hatten wir effiziente Formeln für die Subdivision. Die quadratische Konvergenz unseres Nullstellenalgorithmus war dann eine Folge der quadratischen Konvergenz der Kontrollstruktur unter Subdivision. Nun kann man auch für nicht-polynomiale Funktionenräume Bernsteinfunktionen definieren. Die Frage ist nur, ob sie die selben Eigenschaften besitzen. Die Partition der Eins lässt sich gewöhnlich erhalten, die Positivität hingegen ist nur dann garantiert, falls der zugrunde liegende Funktionenraum ein sogenanntes EC-System bildet. Außerdem ist zu klären, wie die Subdivision durchgeführt werden kann. Diese offenen Fragen können am einfachsten beantwortet werden, indem man zuerst eine allgemeine Splinetheorie entwickelt und die Bernsteinfunktionen dann als Splines zu ganz bestimmten Knotenfolgen auffasst.

Diese Theorie wollen wir im ersten Abschnitt 4.1 besprechen. Die für das weitere Vorgehen notwendigen Erkenntnisse sind in den Sätzen 4.24 und 4.26 zusammengefasst. In Abschnitt 4.1.1 werden wir dann aus der Theorie die benötigten Algorithmen für die Subdivision ableiten. Diese benötigen Input, welcher sehr einfach bereit gestellt werden kann. Allerdings kann die Berechnung des Inputs im Allgemeinen nicht ohne weiteres numerisch stabil durchgeführt werden. Deshalb werden wir für die sogenannten Exponentialräume in Abschnitt 4.2 detailliert besprechen, wie hier der Input stabil zu berechnen ist. Zu beachten ist, dass mit der Klasse der Exponentialräume durch Transformation des Gleichungssystems ein großes Feld von potenziellen Gleichungssystemen abgedeckt werden kann (siehe Abschnitt 4.5). In Abschnitt 4.3 besprechen wir dann eine Variante zur Darstellung der Gleichungen in Bernsteinfunktionen, welche sich in bestimmten Situationen als sehr nützlich erweist (siehe Abschnitt 4.4). In Abschnitt 4.6 besprechen wir schließlich exemplarisch die Behandlung von Funktionenräumen, die nicht in die Klasse der Exponentialräume fallen.

4.1 Theoretische Grundlagen

Dieses Kapitel setzt keine Kenntnisse über polynomiale Splinetheorie voraus, jedoch ist eine gute Kenntnis derselben von allergrößtem Nutzen für das Verständnis des Folgenden. Die in diesem Abschnitt vorgestellten Resultate stammen im Wesentlichen aus [Ma96], [ML96], [ML99], [Ma99] und [DR88].

Unser Anspruch ist zunächst, in Funktionenräumen $\mathcal{F} = \langle f_1, \dots, f_m \rangle$ Splines zu definieren. Dies ist nur möglich, falls \mathcal{F} ein sogenanntes EC-System bildet. Die Splines definieren wir dann so, wie man sie auch im polynomialen Fall definieren kann, nämlich über die Marsdenidentität. Sie lautet im polynomialen Fall

$$(x - a)^{m-1} = \sum_k b_k(x) \Psi_k^*(a) \quad , \text{supp}(b_k) = [\tau_k, \tau_{k+m}] .$$

In dieser tauchen drei verschiedene Arten von Ausdrücken auf. Links steht die Funktion $(x - a)^{m-1}$. Diese ist diejenige polynomiale Funktion vom Grad $(m - 1)$, die an der Stelle a eine $(m - 1)$ -fache Nullstelle besitzt, wir werden sie *glatte Potenz* nennen. Die ersten Terme $b_k(x)$ in der Summe auf der rechten Seite sind die durch eben jene Marsdenidentität definierten Splines. Die zweiten Terme in der Summe $\Psi_k^*(a)$ sind Polynome (Marsdenpolynome), genauso wie die Splines stückweise Polynome sind. Die Ψ_k^* besitzen Nullstellen an sogenannten Knoten und sind nach Skalierung durch diese Nullstellen eindeutig definiert. Nun können wir genau dieses Vorgehen auf nicht-polynomiale Systeme übertragen, bis auf den Unterschied, dass die Ψ_k^* nicht mehr im selben Funktionenraum wie die Segmente der Splines liegen. Die Ψ_k^* sind nun Elemente in einem sogenannten Dualsystem.

Definition 4.1 (*C-System F bzw. \mathcal{F} , EC-System, glatte Potenz $H(\mathcal{F}|x, a)$, Marsdenidentität, Dualsystem F^* bzw. \mathcal{F}^*)*

Sei f_1, \dots, f_m eine reellwertige Basis eines m -dimensionalen Funktionenraums, wobei die f_k ($m - 1$) mal stetig differenzierbar sind. Die f_k seien dabei auf einem Intervall $I \subset \mathbb{R}$ definiert. Wir nennen den liegenden Vektor $F := [f_1, \dots, f_m]$ bzw. auch den Funktionenraum $\mathcal{F} := \langle f_1, \dots, f_m \rangle$ ein C-System (extended Chebyshev system), falls jedes Herimite-Interpolationsproblem mit m Daten sich in \mathcal{F} eindeutig lösen lässt, also wenn für beliebige $x_1, \dots, x_r \in I$ und μ_1, \dots, μ_r mit $\sum_{k=1}^r (\mu_k + 1) = m$

$$\det \begin{vmatrix} f_1(x_1) & f_1'(x_1) & \cdots & f_1^{(\mu_1)}(x_1) & f_1(x_2) & \cdots & f_1^{(\mu_r)}(x_r) \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ f_m(x_1) & f_m'(x_1) & \cdots & f_m^{(\mu_1)}(x_1) & f_m(x_2) & \cdots & f_m^{(\mu_r)}(x_r) \end{vmatrix} \neq 0.$$

Wir nennen ein C-System $F = [f_1, \dots, f_m]$ ein EC-System, falls zusätzlich gilt: $f_1 \equiv 1$ und f_2, \dots, f_m ist ebenfalls ein C-System.

Für ein C-System \mathcal{F} nennen wir für ein $a \in I$ die Funktion $f_a \in \mathcal{F}$ mit $D^k f_a(a) = \delta_{k,m-1}$, $k \in 0 : m - 1$, glatte Potenz und definieren $H(\mathcal{F}|x, a) := f_a(x)$. Die glatte Potenz fassen wir als Funktion auf I^2 in den beiden Veränderlichen x und a auf.

Die im stehenden Vektor zusammengefassten Koeffizienten $F^*(a) := [f_1^*(a), \dots, f_m^*(a)]^T$, für die die sogenannte Marsdenidentität

$$H(\mathcal{F}|x, a) = F(x)F^*(a) \quad , (x, a) \in I^2, \quad (4.1)$$

erfüllt ist, fassen wir ebenfalls auf als Funktionen in a und nennen F^* bzw. $\mathcal{F}^* := \langle f_1^*, \dots, f_m^* \rangle$ das Dualsystem zu F .

Machen wir einige Anmerkungen zu dieser Definition. Zunächst liegt offenbar genau dann ein C-System vor, falls jedes $f \in \mathcal{F} \setminus \{0\}$ höchstens $(m - 1)$ Nullstellen (mit Vielfachheiten gezählt) besitzt. Somit ist in einem C-System die glatte Potenz eindeutig und das Dualsystem wohldefiniert. Bei der Schreibweise f_k^* ist zu beachten, dass f_k^* natürlich nicht nur von f_k abhängt, sondern natürlich von allen f_1, \dots, f_n . Jedoch hängt \mathcal{F}^* nur von \mathcal{F} , nicht aber von der gewählten Basis ab, denn sei $H(\mathcal{F}|x, a) = F(x)F^*(a)$ und $G = FM$ eine andere Basis (M invertierbare Matrix), dann gilt $G^* = M^{-1}F^*$.

Zur Entscheidung, ob es sich bei einem vorgegebenen Funktionenraum um ein C-System handelt, gibt es ein Kriterium, das wir in Satz 4.4 angeben werden. Um diesen Satz formulieren zu können benötigen wir die folgende Definition.

Definition 4.2 Seien f_1, \dots, f_n Funktionen und x ein Wert aus dem gemeinsamen

Definitionsbereich. Wir schreiben

$$W(x, f_1, \dots, f_n) := \det \begin{pmatrix} f_1(x) & \dots & f_n(x) \\ f_1'(x) & \dots & f_n'(x) \\ \vdots & & \vdots \\ f_1^{(n-1)}(x) & \dots & f_n^{(n-1)}(x) \end{pmatrix}$$

und nennen obigen Ausdruck die Wronski-Determinante von f_1, \dots, f_n an der Stelle x .

Uns wird im weiteren Verlauf nur interessieren, ob die Wronski-Determinante verschwindet oder nicht. Sofort bemerken wir das folgende Lemma.

Lemma 4.3 Falls $\langle f_1, \dots, f_n \rangle = \langle g_1, \dots, g_n \rangle$, so gilt

$$W(x, f_1, \dots, f_n) = 0 \quad \iff \quad W(x, g_1, \dots, g_n) = 0.$$

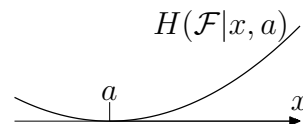
In [S74] findet sich das folgende Kriterium zur Entscheidung, ob es sich bei einem Funktionenraum um ein EC-System handelt.

Satz 4.4 $\mathcal{F} = \langle f_1, \dots, f_m \rangle$ ist genau dann ein C-System auf $I = [a, b]$, falls Funktionen $h_1, \dots, h_m \in \mathcal{F}$ existieren, sodass die Wronski-Determinanten $W(x, h_1), W(x, h_1, h_2), \dots, W(x, h_1, \dots, h_m)$ allesamt nirgendwo auf I verschwinden.

Betrachten wir kurz drei Beispiele.

Beispiel 4.5 Für $f_k(x) := x^k$ ist $\mathcal{F} = \langle f_1, \dots, f_m \rangle$ bekanntermaßen auf ganz \mathbb{R} ein C-System, schließlich ist mit den Lagrange-Polynomen jedes Interpolationsproblem eindeutig lösbar. Die glatte Potenz lautet im polynomialen Fall nun $(m-1)!H(\mathcal{F}|x, a) = (x-a)^{m-1}$. Da

$$(x-a)^{m-1} = \sum_{k=0}^{m-1} (-1)^{m-1-k} \binom{m-1}{k} x^k a^{m-1-k}$$



gilt, ist offenbar $\mathcal{F}^* = \mathcal{F}$.

Beispiel 4.6 Betrachten wir nun einen zweiten Funktionenraum \mathcal{F} . Für m paarweise verschiedene $\omega_k \in \mathbb{R}$ ist $\mathcal{F} = \langle \exp(\omega_1 x), \dots, \exp(\omega_m x) \rangle$ ebenfalls auf ganz \mathbb{R} ein C-System. Dies folgt unmittelbar mit Satz 4.4. Hier gilt nach (4.15)

$$H(\mathcal{F}|x, a) = \sum_{k=1}^m \exp(\omega_k x) \frac{\exp(-\omega_k a)}{\prod_{j=1, j \neq k}^m (\omega_j - \omega_k)},$$

woraus $\mathcal{F}^* = \langle \exp(-\omega_1 x), \dots, \exp(-\omega_m x) \rangle$ folgt.

Beispiel 4.7 Mit $\mathcal{F} = \langle 1, \sin(x), \cos(x) \rangle$ ist \mathcal{F} offenbar kein C-System auf \mathbb{R} . Denn die Funktion $1 - \cos(x)$ besitzt in 0 und 2π jeweils eine doppelte Nullstelle. Jedoch lässt sich mit Satz 4.4 zeigen, dass \mathcal{F} auf jedem offenen Intervall I mit einer Intervalllänge, die kleiner oder gleich 2π ist, ein C-System ist. Dies sieht man wie folgt. Für Satz 4.4 sei zunächst $h_1(x) = \cos(x) - 1$, $h_2(x) = \sin(x)$ und $h_3(x) = 1$. Mit $x \in (0, 2\pi)$ ist $h_1(x) < 0$, außerdem ist

$$\det \begin{bmatrix} \cos(x) - 1 & \sin(x) \\ -\sin(x) & \cos(x) \end{bmatrix} = 1 - \cos(x) > 0$$

und

$$\det \begin{bmatrix} \cos(x) - 1 & \sin(x) & 1 \\ -\sin(x) & \cos(x) & 0 \\ -\cos(x) & -\sin(x) & 0 \end{bmatrix} = 1.$$

Da \mathcal{F} translationsinvariant ist, kann nun geschlossen werden, dass auf beliebigen Intervallen (a, b) mit $b - a \leq 2\pi$ der Funktionenraum \mathcal{F} ein C-System ist.

Im Folgenden wollen wir C-Systeme \mathcal{F} und deren Dualsysteme diskutieren. Mit \mathcal{F} sei deshalb im Folgenden stets ein m -dimensionales C-System gemeint, $[f_1, \dots, f_m]$ sei eine Basis von \mathcal{F} und $[f_1^*, \dots, f_m^*]$ das zugehörige Dualsystem, ohne dies jedesmal zu sagen.

Im Weiteren wird sich herausstellen, dass wir nicht nur benötigen, dass \mathcal{F} ein C-System ist, sondern auch \mathcal{F}^* . Dies ist aber eine äquivalente Forderung, wie das folgende Lemma zusammen mit Lemma 4.10 lehrt.

Lemma 4.8 Wenn \mathcal{F} ein C-System ist, so ist auch \mathcal{F}^* ein C-System.

Ein Beweis findet sich beispielsweise in [ML96]. Man beachte, dass mit Lemma 4.8 die $(m - 1)$ -malige stetige Differenzierbarkeit der f_k^* impliziert wird. Die glatten Potenzen in den beiden C-Systemen \mathcal{F} und \mathcal{F}^* hängen nun elementar zusammen, wie wir im folgenden Lemma sehen. Diese Tatsache machen wir uns dann beim Beweis von Lemma 4.10 zunutze.

Lemma 4.9 Es gilt für k, l mit $k + l \in 0 : m - 1$

$$D_1^k D_2^l H(\mathcal{F}|a, a) = (-1)^l \delta_{k+l, m-1}.$$

Außerdem ist für festes x die Funktion $H(\mathcal{F}|x, \cdot) = F(x)F(\cdot)^* \in \mathcal{F}^*$ und es gilt für die glatte Potenz in \mathcal{F} nun $H(\mathcal{F}^*|a, x) = (-1)^{m-1} H(\mathcal{F}|x, a)$.

Beweis Da $H(\mathcal{F}|x, a) = F(x)F^*(a)$ gilt, ist $H(\mathcal{F}|\cdot, \cdot) \in C^{m-1}$. Sei $D_{(c,d)}H$ die Ableitung von H in Richtung (c, d) . Es gilt offenbar $D_{(1,0)}^k D_{(0,1)}^l H(\mathcal{F}|x, a) = D_{(1,0)}^k (D_{(1,1)} - D_{(1,0)})^l H(\mathcal{F}|x, a)$. Da $H(\mathcal{F}|x, x) \equiv 0$ ist, ist $(D_{(1,1)} - D_{(1,0)})^l H(\mathcal{F}|x, a) = (-D_{(1,0)})^l H(\mathcal{F}|x, a)$ und somit gilt $D_{(1,0)}^k D_{(0,1)}^l H(\mathcal{F}|x, a) = (-1)^l D_{(1,0)}^{k+l} H(\mathcal{F}|\cdot, \cdot)$. Die zweite Behauptung folgt direkt aus (4.1) und die dritte Behauptung ergibt sich aus der ersten mit $k = 0$ und $l = m - 1$. \square

Dass die Bezeichnung von \mathcal{F}^* als Dualsystem zu \mathcal{F} gerechtfertigt ist, zeigt das nächste Lemma. Es stellt ein wichtiges Hilfsmittel dar, um nachzuprüfen, ob ein vorgelegter Funktionenraum \mathcal{F} ein C-System bildet. Es erlaubt uns zusammen mit Lemma 4.8, dies alternativ für \mathcal{F}^* zu zeigen.

Lemma 4.10 *Es gilt $(\mathcal{F}^*)^* = \mathcal{F}$.*

Beweis

$H(\mathcal{F}^*|a, x) = (-1)^{m-1} H(\mathcal{F}|x, a) = (-1)^{m-1} \sum_k f_k(x) f_k^*(a) = \sum_k f_k^*(a) (-1)^{m-1} f_k(x)$. \square

Von größtem theoretischem Interesse ist die im Folgenden definierte Bilinearform. Für uns gibt sie in dieser Arbeit die Möglichkeit, die noch zu definierenden Marsdenfunktionen richtig zu normieren, was dann die Normierung der Splines als Partition der Eins zur Folge hat.

Definition 4.11 (*Bilinearform $[\cdot, \cdot]$*)

Auf $\mathcal{F} \times \mathcal{F}^*$ definieren wir mit $f = \sum_k \alpha_k f_k$ und $f^* = \sum_k \alpha_k^* f_k^*$ die Bilinearform

$$[f, f_k^*] := \sum_k \alpha_k \alpha_k^*.$$

Diese Definition scheint von der gewählten Basis abzuhängen. Dass dem nicht so ist, und dass sie demnach tatsächlich sinnvoll ist, sagt uns das folgende Lemma.

Lemma 4.12 *Obige Bilinearform ist unabhängig von der gewählten Basis F von \mathcal{F} .*

Beweis Sei \mathcal{F} ein C-System, $f \in \mathcal{F}$ und $f^* \in \mathcal{F}^*$. Gelte ferner mit $A := [\alpha_1, \dots, \alpha_m]^T$ und $A^* := [\alpha_1^*, \dots, \alpha_m^*]$ nun $f = FA$ bzw. $f^* = A^*F^*$, so ist für ein invertierbares M in der Basis $G = FM$ nun $G^* = M^{-1}F^*$ und somit $f = G(M^{-1}A)$ und $f^* = (A^*M)G^*$. Nun ist wegen $(A^*M)(M^{-1}A) = A^*A$ die Bilinearform unabhängig von der gewählten Basis. \square

Das folgende Lemma benötigen wir, um Lemma 4.14 zu beweisen, welches dann den letzten Schritt zur Definition der Splines darstellt.

Lemma 4.13 *Sei f_1, \dots, f_m EC-System mit Dualsystem f_1^*, \dots, f_m^* . Dann ist f_2^*, \dots, f_m^* Dualsystem zu f_2', \dots, f_m' und somit nach Lemma 4.8 ebenfalls C-System.*

Beweis Der Beweis birgt keine Schwierigkeiten, es gilt nämlich

$$\begin{bmatrix} 1 & f_2(a) & \cdots & f_m(a) \\ 0 & f_2'(a) & \cdots & f_m'(a) \\ \vdots & \vdots & & \vdots \\ 0 & f_2^{(m-1)}(a) & \cdots & f_m^{(m-1)}(a) \end{bmatrix} F^*(a) = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

und somit natürlich

$$\begin{bmatrix} f_2'(a) & \cdots & f_m'(a) \\ f_2''(a) & \cdots & f_m''(a) \\ \vdots & & \vdots \\ f_2^{(m-1)}(a) & \cdots & f_m^{(m-1)}(a) \end{bmatrix} \begin{bmatrix} f_2^*(a) \\ f_3^*(a) \\ \vdots \\ f_m^*(a) \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}.$$

□

Lemma 4.14 *Seien $(m-1)$ Stellen $\tau_{k+1}, \dots, \tau_{k+m-1} \in I$ beliebig vorgegeben und \mathcal{F} ein EC-System. Dann existiert eine eindeutige Funktion $\Psi_k^* \in \mathcal{F}^*$, welche in den $\tau_{k+1}, \dots, \tau_{k+m-1}$ verschwindet und geeignet normiert werden kann, für die also gilt:*

- $D^{\sharp j} \Psi_k^*(\tau_j) = 0, \quad \forall j \in (k+1 : k+m-1).$
- $[1, \Psi_k^*] = 1.$

Dabei sei $\sharp j := |\{k > j : \tau_k = \tau_j\}|$ die Zahl der identischen Nachfolger von τ_j .

Beweis Nehmen wir an, dass für alle j zunächst $\tau_j < \tau_{j+1}$ gilt, also dass für alle j nun $\sharp j = 0$ ist. Wir setzen

$$\mathcal{F} \ni \Psi_k^*(a) := \frac{\begin{vmatrix} f_1^*(a) & f_2^*(a) & \cdots & f_m^*(a) \\ f_2^*(\tau_{k+1}) & f_2^*(\tau_{k+1}) & \cdots & f_m^*(\tau_{k+1}) \\ \vdots & & & \vdots \\ f_1^*(\tau_{k+m-1}) & f_2^*(\tau_{k+m-1}) & \cdots & f_m^*(\tau_{k+m-1}) \end{vmatrix}}{\begin{vmatrix} f_2^*(\tau_{k+1}) & \cdots & f_m^*(\tau_{k+1}) \\ \vdots & & \vdots \\ f_2^*(\tau_{k+m-1}) & \cdots & f_m^*(\tau_{k+m-1}) \end{vmatrix}}.$$

Ψ_k^* hat die verlangten Nullstellen, außerdem ist $[1, \Psi_k^*] = [1, 0, \dots, 0][1, *, \dots, *]^T = 1$. Im Fall von mehrfachen τ_j sind in den obigen Matrizen die entsprechenden Zeilen durch die Ableitungen der f_j^* zu ersetzen. Die Determinante im Nenner ist ungleich Null, wenn nur f_2^*, \dots, f_m^* C-System ist, was aber per Voraussetzung der Fall ist. Nun ist $[1, \Psi_k^*] = [1, 0, \dots, 0][1, *, \dots, *]^T = 1$. \square

Nun können wir den Begriff der Marsdenpolynome verallgemeinern und mit diesem dann Chebyshev-B-Splines definieren.

Definition 4.15 (Knotenfolge T , Marsdenfunktionen Ψ_k^*)

Sei \mathcal{F} ein EC-System und $T = (\tau_k)_k$ eine Folge mit $\tau_k \leq \tau_{k+1}$ und $\tau_k < \tau_{k+n}$. Wir nennen die τ_k Knoten und T die Knotenfolge. Das Symbol $\sharp_T k$ sei bezogen auf eine Knotenfolge $T = (t_k)_k$ die Anzahl der identischen Nachfolger von τ_k in T , also $\sharp_T k := |\{j > k : t_j = t_k\}|$. Außerdem sei $*_T k$ die Vielfachheit eines Knotens τ_k in T , also $*_T k := |\{j : t_j = t_k\}|$. Die nach dem vorigen Lemma eindeutigen Funktion $\Psi_k^* \in \mathcal{F}^*$ mit $D^{\sharp_T j} \Psi_k^*(\tau_j) = 0$ und $[1, \Psi_k^*] = 1$ nennen wir k -te Marsdenfunktion.

Im Folgenden wollen wir stets annehmen, dass \mathcal{F} ein EC-System ist, ohne dies jeweils explizit zu erwähnen. Wir wollen also nur noch C-Systeme betrachten für die $f_1 \equiv 1$ und $\langle f'_2, \dots, f'_m \rangle$ ebenfalls C-System ist.

Für die Definition der Splines ist es notwendig, dass jeweils m aufeinanderfolgende Ψ_k^* linear unabhängig sind. Dass dem für die betrachteten Knotenfolgen so ist, sehen wir im folgenden Lemma. Der Beweis dieser Aussage ist eine Kopie des polynomialen Standardbeweis.

Lemma 4.16 Falls $\tau_j < \tau_{j+1}$, so bilden die n Marsdenfunktionen $\Psi_{j-m+1}^*, \dots, \Psi_j^*$ eine Basis von \mathcal{F}^* .

Beweis Wir markieren im folgenden Schema die Nullstellen der Marsdenfunktionen. Daraus ergibt sich dann zusammen mit der Tatsache, dass \mathcal{F} ein C-System ist, wie wir sehen werden, die lineare Unabhängigkeit.

	τ_{j-m+2}	τ_{j-m+3}	\dots	$\tau_j \neq \tau_{j+1}$	\dots	τ_{j+m-2}	τ_{j+m-1}
Ψ_{j-m+1}	×	×	×	×			
Ψ_{j-m+2}		×	×	×	×		
\vdots			×	×	×	×	
Ψ_{j-1}				×	×	×	×
Ψ_j					×	×	×

Angenommen $\sum_{k=j-m+1}^j \alpha_k \Psi_k^* \equiv 0$, dann muss $\alpha_j = 0$ sein, denn Ψ_j^* ist die einzige Funktion, die nicht in τ_j verschwindet. Nun muss α_{j-1} verschwinden, denn allein Ψ_{j-1}^* besitzt in τ_{j-1} keine Nullstelle bzw., falls $\tau_{j-1} = \tau_{j-2}$, keine doppelte Nullstelle. So ergibt sich sukzessive die lineare Unabhängigkeit der Ψ_k^* . \square

Nun sind wir endlich am Ziel und können Splines definieren. Diese sind, wie wir in den folgenden Sätzen sehen werden, stückweise Funktionen aus \mathcal{F} , besitzen kompakten Support und besitzen an den Knoten τ_k gewisse Differenzierbarkeitsordnungen. Außerdem bilden sie eine positive Partition der Eins.

Definition 4.17 (*Chebyshev-B-Splines*)

Sei \mathcal{F} ein EC-System, T und Ψ_k^* wie in Definition 4.15 vereinbart. Dann lassen sich Funktionen b_k durch die folgenden beiden Eigenschaften definieren.

- $b_k(x) \neq 0 \Rightarrow x \in [\tau_k, \tau_{k+m})$
- $H(\mathcal{F}|x, a) = \sum_k b_k(x) \Psi_k^*(a)$

Wir nennen die Funktionen $b_k : I \rightarrow \mathbb{R}$ Chebyshev-B-Splines.

Durch die Forderung an den Träger von b_k sind die Chebyshev-B-Splines durch die Marsden-Identität wohldefiniert. Dadurch dürfen auf einem Segment $[\tau_k, \tau_{k+1})$ nämlich nur n Chebyshev-B-Splines nicht verschwinden. Im Folgenden sei stets eine Knotenfolge T und ein EC-System \mathcal{F} gegeben, auch wenn dies nicht jedesmal explizit in den Lemmata, Sätzen bzw. Definitionen gefordert werden sollte.

Beispiel 4.18 *Wir wollen zwei Chebyshev-B-Splines plotten. Sei die Knotenfolge gegeben durch $\tau_k = k$ und $\mathcal{F}_1 = \langle 1, x, x^2 \rangle$ bzw. $\mathcal{F}_2 = \langle 1, \exp(2x), \exp(3x) \rangle$. Beidesmal handelt es sich nach den Beispielen 4.5 und 4.6 um EC-Systeme. In Abbildung 20 sehen wir jeweils den Chebyshev-B-Spline b_0 .*

Die grundlegenden Eigenschaften der Chebyshev-B-Splines sind nun in Satz 4.19 und Satz 4.20 aufgeführt. Wir fassen diese später in Satz 4.23 zusammen.

Satz 4.19 *Die Chebyshev-B-Splines spannen stückweise den ursprünglichen Funktionenraum \mathcal{F} auf, es gilt also $\langle b_{j-m+1}, \dots, b_j \rangle|_{[\tau_j, \tau_{j+1})} = \mathcal{F}|_{[\tau_j, \tau_{j+1})}$. Wir nennen dies die lokale-Basis-Eigenschaft. Es gilt außerdem $\sum_k b_k \equiv 1$.*

Wenn also eine Linearkombination von Chebyshev-B-Splines auf einem Segment identisch verschwindet, so müssen die Koeffizienten der dort von Null verschiedenen Chebyshev-B-Splines allesamt verschwinden.

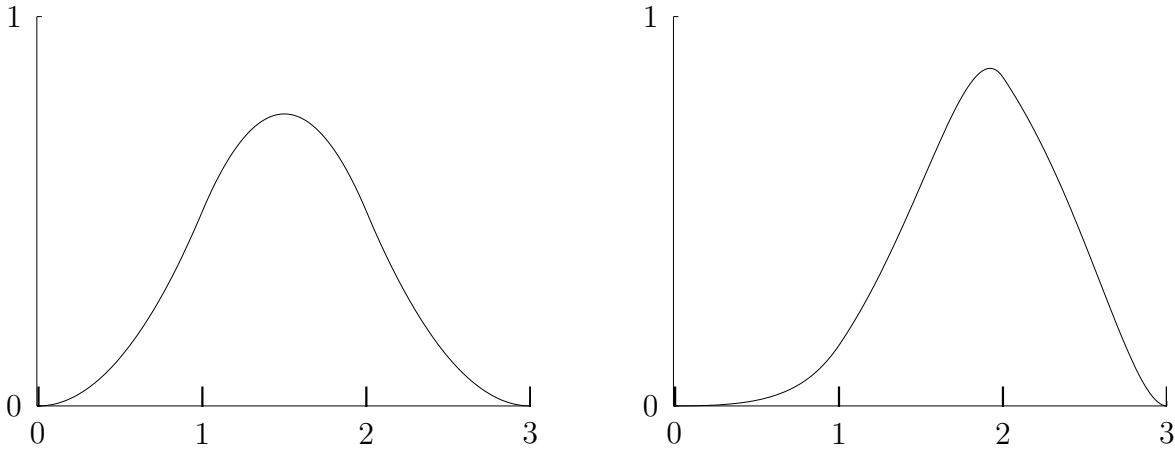


Abbildung 20: Chebyshev-B-Spline b_0 in $\mathcal{F}_1 = \langle 1, x, x^2 \rangle$ (links) bzw. in $\mathcal{F}_1 = \langle 1, \exp(2x), \exp(3x) \rangle$ (rechts) zur Knotenfolge $\tau_k = k$.

Beweis Die Funktionen $b_l|_{[\tau_j, \tau_{j+1}]}$ sind nach Lemma 4.9 und Lemma 4.10 offenbar Funktionen in $\mathcal{F}|_{[\tau_j, \tau_{j+1}]}$. Die lineare Unabhängigkeit der b_k ist nach Lemma 4.8 und Lemma 4.10 eine Folgerung der linearen Unabhängigkeit der Ψ_k^* . Wenden wir uns nun der Partition der Eins zu. Wir betrachten die Chebyshev-B-Splines auf dem Intervall $[\tau_j, \tau_{j+1}]$. Dort stimmt b_k mit einer Funktion $f_{k,j} \in \mathcal{F}$ überein und wegen der Dualität der Ψ_l^* zu den $f_{k,l}$ gilt $[\sum_k f_{k,j}, \Psi_l^*] = \sum_k [f_{k,j}, \Psi_l^*] = \sum_k \delta_{k,l} = 1$. Da die Ψ_l^* eine Basis von \mathcal{F}^* bilden, so muss wegen $[1, \Psi_l^*] = 1$ schließlich $\sum_k b_k = 1$ gelten. \square

Satz 4.20 *Der Chebyshev-B-Spline b_k ist*

- $(m - 1 - \sharp_T k)$ mal stetig differenzierbar in τ_k .
- $(m - 1 - \text{Anzahl der identischen Vorgängerknoten})$ mal stetig differenzierbar in τ_{k+m} .
- $(m - *_T(k + j))$ mal stetig differenzierbar in τ_{k+j} für $j \in 1 : m - 1$.

Beweis Sei wie im Beweis von Satz 4.19 $f_{k,j} \in \mathcal{F}$ so, dass $f_{k,j}$ auf dem j -ten Segment mit b_k übereinstimmt, dass also $f_{k,j}|_{[\tau_j, \tau_{j+1}]} \equiv b_k|_{[\tau_j, \tau_{j+1}]}$. Nehmen wir zunächst an, dass τ_j einfacher Knoten ist. Zu betrachten ist die Differenzierbarkeitsordnung von $(f_{k,j-1} - f_{k,j})$ an der Stelle τ_j . Sei zunächst $H := [H(\mathcal{F}|x, \tau_j), D_2 H(\mathcal{F}|x, \tau_j), \dots, D_2^{m-1} H(\mathcal{F}|x, \tau_j)]^T$,

$F_1 := [f_{j-m,j-1}(x), f_{j-m+1,j-1}(x), \dots, f_{j-1,j-1}(x)]^T$ und

$$M_1 := \begin{bmatrix} \Psi_{j-m}^*(\tau_j) & \Psi_{j-m+1}^*(\tau_j) & \cdots & \Psi_{j-1}^*(\tau_j) \\ D\Psi_{j-m}^*(\tau_j) & D\Psi_{j-m+1}^*(\tau_j) & \cdots & D\Psi_{j-1}^*(\tau_j) \\ D^{m-1}\Psi_{j-m}^*(\tau_j) & D^{m-1}\Psi_{j-m+1}^*(\tau_j) & \cdots & D^{m-1}\Psi_{j-1}^*(\tau_j) \end{bmatrix} = \left[\begin{array}{c|c} \Psi_{j-m}^*(\tau_j) & \mathbf{0} \\ \hline V_1 & |N \end{array} \right].$$

Dann ist

$$H = M_1 F_1. \quad (4.2)$$

Tauschen wir nun in F_1 den ersten Eintrag und in M_2 entsprechend die erste Spalte geeignet aus. Wir setzen $F_2 := [f_{j,j}(x), f_{j-m+1,j}(x), \dots, f_{j-1,j}(x)]^T$ und $V_2 := [D\Psi_j^*(\tau_j), \dots, D^{n-1}\Psi_j^*(\tau_j)]^T$ und

$$M_2 := \left[\begin{array}{c|c} \Psi_j^*(\tau_j) & 0 \\ \hline V_2 & |N \end{array} \right].$$

Dann ist

$$H = M_2 F_2. \quad (4.3)$$

Aus (4.2) folgt unmittelbar, dass $f_{j-m,j-1}(x) = H(\mathcal{F}|x, \tau_j)/\Psi_{j-m}^*(\tau_j)$, also die zweite Behauptung. Ebenso folgt aus (4.3) $f_{j,j}(x) = H(\mathcal{F}|x, \tau_j)/\Psi_j^*(\tau_j)$ und damit die erste Behauptung.

Die dritte Behauptung erhalten wir durch Subtraktion von (4.2) und (4.3) und Weglassen der ersten Zeile. Wir erhalten $f_{j,j}(x)V_2 - f_{j-m,j-1}(x)V_1 = ND$ und somit gilt wegen der Invertierbarkeit von N

$$D = f_{j-m,j-1}(x)N^{-1}V_1 - f_{j,j}(x)N^{-1}V_2,$$

wobei $D = [f_{j-m+1,j-1}(x) - f_{j-m+1,j}(x), \dots, f_{j-1,j-1}(x) - f_{j-1,j}(x)]^T$. Die Einträge der beiden Vektoren auf der rechten Seite sind nach den ersten beiden Beweisteilen ein Vielfaches von $H(\mathcal{F}|x, \tau_j)$, was schließlich die dritte Behauptung beweist.

Für mehrfache Knoten lässt sich völlig analog schließen. Sei j dabei so, dass wieder τ_j keine identischen Nachfolger besitzt, jedoch identische Vorgänger. Dann finden sich auch in der zweiten Zeile von M ab der dritten Spalte nur noch Nulleinträge. Nun folgt wieder aus (4.2), dass $f_{j-m+1,j-1}$ in τ_j eine $(m-1)$ -fache Nullstelle besitzt und $f_{j-m+2,j-1}$ eine $(m-2)$ -fache. Sukzessive folgt so die zweite und analog aus (4.3) auch die erste Behauptung. Nach Subtraktion der beiden Gleichungen (4.2) und (4.3) und Weglassen der entsprechenden ersten Zeilen bringt man die schon identifizierten Funktionen auf die linke Seite, sodass dort eine Funktion mit $(m - *Tj)$ -facher Nullstelle steht. Wieder findet sich auf der rechten Seite eine invertierbare Matrix, woraus die dritte Behauptung folgt. \square

Zum besseren Verständnis der Splines trägt nun Definition 4.21 und Satz 4.22 bei.

Definition 4.21 (Stutzfunktion a_k , Splineraum $\Lambda_{\mathcal{F},T}$)

Eine Linearkombination von Chebyshev-B-Splines $\sum_k \alpha_k b_k$ nennen wir Spline. Die Menge aller Splines bezeichnen wir mit $\Lambda_{\mathcal{F},T}$. Die Funktion

$$a_k(x) := \begin{cases} 0, & x < \tau_k \\ H(\mathcal{F}|x, \tau_k), & x \geq \tau_k \end{cases}$$

nennen wir k -te Stutzfunktion.

Die Stutzfunktionen a_k sind offenbar Splines.

Satz 4.22 Die Chebyshev-B-Splines bilden eine Basis des Splineraums $\Lambda_{\mathcal{F},T}$. Alternativ sind für endliche Knotenfolgen $\tau_1, \dots, \tau_{n+m}$ und das Definitionsgebiet $[\tau_m, \tau_{n+1}]$ die Stutzfunktionen $a_1(x), \dots, a_n(x)$ eine Basis. Die Dimension des Splineraums ist n .

Beweis Die lineare Unabhängigkeit von m aufeinanderfolgenden Chebyshev-B-Splines, die auf einem gemeinsamen Segment nicht verschwinden, ist gewährleistet durch die lineare Unabhängigkeit der Ψ_k^* und der Dualität der b_k zu diesen. Die Unabhängigkeit aller Chebyshev-B-Splines ist wegen der Anordnung der Träger der b_k trivial. Als Dimension des Splineraums ergibt sich n und, da die Chebyshev-B-Splines Linearkombinationen der Stutzfunktionen sind, folgt, dass auch die Stutzfunktionen linear unabhängig sind. \square

Warum bilden im Allgemeinen die Stutzfunktionen keine Basis des Splineraums, wenn die Knotenfolge unendlich ist? Sei z.B. für k der Indexbereich $-\infty : l$. Dann ist die Konstant-Eins-Funktion ein Element des Splineraums, es ist nämlich $1 = \sum_k b_k(x)$. Versuchen wir nun Koeffizienten α_k zu finden, sodass $1 \equiv \sum_k \alpha_k a_k(x)$. Wäre $\alpha_j \neq 0$, dann wäre $\sum_k \alpha_k a_k$ in τ_j nicht beliebig oft stetig differenzierbar, also muss α_j gleich 0 sein. Wenn aber alle $\alpha_j = 0$, dann ist auch $\sum_k \alpha_k a_k = 0 \neq 1$. Wenn der Indexbereich hingegen $1 : \infty$ ist, so bilden auch die Stutzfunktionen eine Basis. Im letzten Satz genügt es deshalb eigentlich vorauszusetzen, dass der Indexbereich nach unten beschränkt ist.

Satz 4.23 Die Chebyshev-B-Splines b_k besitzen die folgenden Eigenschaften und sind auch eindeutig bestimmt durch diese.

- $b_k|_{[\tau_j, \tau_{j+1})} \in \mathcal{F}|_{[\tau_j, \tau_{j+1})}$

- $\text{supp}(b_k) = [\tau_k, \tau_{k+m}]$
- b_k ist in τ_k $(m - 1 - \sharp_T k)$ mal stetig differenzierbar
- b_k ist in τ_{k+m} genau $(m - 1 - \text{Anzahl der identischen Vorgängerknoten})$ mal stetig differenzierbar
- b_k ist in $(\tau_{k+j} (m - *_T(k + j)))$ mal stetig differenzierbar, $j \in 1 : m - 1$
- $\sum_k b_k \equiv 1$
- $b_k \geq 0, \forall x$

Beweis Dass die b_k die geforderten Eigenschaften besitzen, haben wir bereits bis auf die Positivität bewiesen. Den Beweis der Positivität erbringen wir in Satz 4.24. Dem Nachweis der Eindeutigkeit wenden wir uns nun zu. Wir versuchen, eine Linearkombination von Stutzfunktionen zu bestimmen, sodass die geforderten Eigenschaften eingehalten werden. Wegen des Trägers von b_k muss $b_k = \sum_{j=k}^{m+k-1} \alpha_j a_j$ gelten. Wenn wir α_k vorgeben, so sind die $\alpha_{k+1}, \dots, \alpha_{k+m}$ durch $0 \equiv \sum_{j=k}^{k+m} \alpha_j a_j$ für $x > \tau_{k+m}$ festgelegt. α_{k-m} aber ist wegen $\sum_k b_k \equiv 1$ eindeutig bestimmt, weil eine Skalierung von α_{k-m} den Chebyshev-B-Spline skaliert. Da die Menge der Stutzfunktionen nach Satz 4.22 eine Basis des Splineraums bilden, ist alles gezeigt. \square

Sei nun $T = (\tau_k)_k$ eine Knotenfolge und \hat{T} die um einen Knoten $\hat{\tau} \in [\tau_j, \tau_{j+1})$ erweiterte Knotenfolge. Sei also

$$\hat{\tau}_k = \begin{cases} \tau_k, & k \leq j, \\ \hat{\tau}, & k = j + 1, \\ \tau_{k-1}, & k \geq j + 2. \end{cases}$$

Die Chebyshev-B-Splines b_k sind Splines zur Knotenfolge \hat{T} und lassen sich deshalb darstellen als Linearkombinationen der \hat{b}_k . Wegen der lokalen Basis-Eigenschaft und des Trägers von b_k ist $b_k = \alpha_k \hat{b}_k + \beta_{k+1} \hat{b}_{k+1}$. Da $1 \equiv \sum_k b_k = \sum_k (\alpha_k \hat{b}_k + \beta_{k+1} \hat{b}_{k+1}) = \sum_k (\alpha_k + \beta_k) \hat{b}_k$ ist, gilt $\beta_k = 1 - \alpha_k$. Es genügt also, die α_k zu bestimmen. Sei dazu τ_k $(l + 1)$ -facher Knoten in T mit $\tau_k < \hat{\tau}$ und $\tau_k < \tau_{k+1}$. Für $x \in [\tau_k, \tau_{k+1})$ gilt sowohl

$$\begin{bmatrix} H(\mathcal{F}|x, \tau_k) \\ D_2 H(\mathcal{F}|x, \tau_k) \\ \vdots \\ D_2^l H(\mathcal{F}|x, \tau_k) \end{bmatrix} = \underbrace{\begin{bmatrix} \Psi_k^*(\tau_k) & & & \mathbf{0} \\ D\Psi_k^*(\tau_k) & D\Psi_{k-1}^*(\tau_k) & & \\ \vdots & \vdots & \ddots & \\ D^l \Psi_k^*(\tau_k) & D^l \Psi_{k-1}^*(\tau_k) & \dots & D^l \Psi_{k-l}^*(\tau_k) \end{bmatrix}}_{=:M} \begin{bmatrix} b_k(x) \\ b_{k-1}(x) \\ \vdots \\ b_{k-l}(x) \end{bmatrix}$$

als auch

$$\begin{bmatrix} H(\mathcal{F}|x, \tau_k) \\ D_2 H(\mathcal{F}|x, \tau_k) \\ \vdots \\ D_2^l H(\mathcal{F}|x, \tau_k) \end{bmatrix} = \underbrace{\begin{bmatrix} \hat{\Psi}_k^*(\tau_k) & & & \mathbf{0} \\ D\hat{\Psi}_k^*(\tau_k) & D\hat{\Psi}_{k-1}^*(\tau_k) & & \\ \vdots & \vdots & \ddots & \\ D^l \hat{\Psi}_k^*(\tau_k) & D^l \hat{\Psi}_{k-1}^*(\tau_k) & \dots & D^l \hat{\Psi}_{k-l}^*(\tau_k) \end{bmatrix}}_{=: \hat{M}} \begin{bmatrix} \hat{b}_k(x) \\ \hat{b}_{k-1}(x) \\ \vdots \\ \hat{b}_{k-l}(x) \end{bmatrix}.$$

Wir erhalten durch Gleichsetzen

$$\begin{bmatrix} b_k(x) \\ b_{k-1}(x) \\ \vdots \\ b_{k-l}(x) \end{bmatrix} = M^{-1} \hat{M} \begin{bmatrix} \hat{b}_k(x) \\ \hat{b}_{k-1}(x) \\ \vdots \\ \hat{b}_{k-l}(x) \end{bmatrix}.$$

Dabei ist M^{-1} genau wie M und \hat{M} untere Dreiecksmatrix. Die Diagonaleinträge von M^{-1} sind die Kehrwerte der Diagonaleinträge von M . Somit ergeben sich die Diagonaleinträge α_k von $M^{-1}\hat{M}$ als $\hat{\Psi}_{k-i}^*(\tau_k)/\Psi_{k-i}^*(\tau_k)$. Es gilt also $\alpha_k = \frac{D^{\sharp k} \hat{\Psi}_k^*(\tau_k)}{D^{\sharp k} \Psi_k^*(\tau_k)}$, wobei wir aus Gründen der Lesbarkeit den Index T bei \sharp weggelassen haben.

Durch entsprechende Betrachtungen, sehen wir, dass obige Darstellung allgemein für beliebige α_k richtig ist. Solange k so klein ist, dass $\hat{\tau}$ weder Nullstelle von Ψ_k^* noch von $\hat{\Psi}_k^*$ ist, ist deshalb $\alpha_k = 1$. Sobald $\hat{\tau}$ Nullstelle von $\hat{\Psi}_k^*$ aber nicht von Ψ_k^* bzw. Nullstelle höherer Vielfachheit in $\hat{\Psi}_k^*$ ist, so ist $\alpha_k = 0$. Wir erhalten also insgesamt:

Satz 4.24 *Sei T eine Knotenfolge und $\sum_k b_k p_k$ ein Spline zur Knotenfolge T . Sei außerdem \hat{T} die um den Knoten $\hat{\tau} \in [\tau_j, \tau_{j+1})$ erweiterte Knotenfolge, also*

$$\hat{\tau}_k = \begin{cases} \tau_k, & k \leq j, \\ \hat{\tau}, & k = j + 1, \\ \tau_{k-1}, & k \geq j + 2. \end{cases}$$

Dann gilt:

$$\sum_k b_k p_k = \sum_k \hat{b}_k (\alpha_k p_k + (1 - \alpha_k) p_{k-1}),$$

wobei mit $v(\hat{\tau}) := |\{i : \tau_i = \hat{\tau}\}|$

$$\alpha_k = \begin{cases} 1, & k \leq j - n + 1, \\ \frac{D^{\sharp k} \hat{\Psi}_k^*(\tau_k)}{D^{\sharp k} \Psi_k^*(\tau_k)}, & k \in j - n + 2 : j - v(\hat{\tau}), \\ 0, & k \geq j - v(\hat{\tau}) + 1, \end{cases}$$

sei. Wir haben dabei aus Gründen der Lesbarkeit anstatt $\sharp_T k$ nur $\sharp k$ geschrieben. Die α_k lassen sich wie folgt angeben. Sei

$$q(t) := \frac{\begin{vmatrix} f_1^*(\tau_k) & f_2^*(\tau_k) & \cdots & f_n^*(\tau_k) \\ f_1^*(\tau_{k+1}) & f_2^*(\tau_{k+1}) & \cdots & f_n^*(\tau_{k+1}) \\ \vdots & \vdots & & \vdots \\ f_1^*(\tau_{k+n-2}) & f_2^*(\tau_{k+n-2}) & \cdots & f_n^*(\tau_{k+n-2}) \\ f_1^*(t) & f_2^*(t) & \cdots & f_n^*(t) \end{vmatrix}}{\begin{vmatrix} f_2^*(\tau_{k+1}) & \cdots & f_n^*(\tau_{k+1}) \\ \vdots & & \vdots \\ f_2^*(\tau_{k+n-2}) & \cdots & f_n^*(\tau_{k+n-2}) \\ f_2^*(t) & \cdots & f_n^*(t) \end{vmatrix}},$$

wobei die Matrix im Nenner offenbar durch Streichung der ersten Zeile und Spalte der Matrix des Zählers entsteht. Für mehrfache Knoten sind in den entsprechenden Zeilen die $f_i^*(\tau_l)$ durch die k -te Ableitung $f_i^{*(k)}(\tau_l)$ auszutauschen. Nun ist

$$\alpha_k = \frac{q(\hat{\tau})}{q(\tau_{k+n-1})}.$$

Für α_k gilt allgemein

$$\alpha_k \in [0, 1],$$

und für die Chebyshev-B-Splines gilt $b_k \geq 0$.

Beweis Zu beweisen ist lediglich noch die letzte Behauptung $\alpha_k \in [0, 1]$. Die α_k können wir als Funktion in $\hat{\tau}$ auffassen, also $\alpha_k(\tau) = q(\hat{\tau})/q(\tau_{k+n-1})$. Nun gilt

- α_k lässt sich stetig fortsetzen in $\tau_{k+1}, \dots, \tau_{k+n-2}$, denn

$$\alpha_k(\hat{\tau}) = \frac{g(\hat{\tau})}{h(\hat{\tau})}$$

und g und h haben außer in τ_k Nullstellen derselben Vielfachheit an den entsprechenden Stellen.

- $\alpha_k(\tau_{k+n-1}) = 1$ und $\alpha_k(\tau_k) = 0$.
- α_k besitzt außer in τ_k keine weitere Nullstelle, sonst wäre der Zähler dort Null, was nicht sein kann, da \mathcal{F}^* C-System ist.

Insgesamt muss deshalb $\alpha_k(\hat{\tau}) \geq 0$ sein. Nun können wir alles im Zusammenhang des Knoteneinfügens Gesagte analog für $\beta_k = 1 - \alpha_k$ übertragen, sodass wir $\beta_k \geq 0$ erhalten, was schließlich unsere Behauptung beweist. Durch m -maliges Knoteneinfügen an der Stelle τ in die Knotenfolge T mit den Kontrollpunkten $p_k = \delta_{j,k}$ erhält man den Wert des Chebyshev-B-Splines an der Stelle τ . Da $\alpha_k, \beta_k > 0$ und somit beim Knoteneinfügen nur Konvexkombinationen von nicht-negativen Kontrollpunkten gebildet werden, muss der Spline nicht-negativ sein. \square

Der folgende Begriff der Bernsteinfunktionen stellt eine Verallgemeinerung der Bernsteinpolynome dar, denn die Bernsteinfunktionen $b_{m,k}$ auf $[0, 1]$ im polynomialen Fall besitzen in 0 eine $(k - 1)$ -fache Nullstelle und in 1 eine $(m - k)$ -fache Nullstelle, außerdem gilt $\sum_k b_{m,k} \equiv 1$, wodurch die $b_{m,k} \equiv 1$ als Splines nach Theorem 4.23 eindeutig bestimmt sind. Wir definieren:

Definition 4.25 Sei $a < b$ und $T = [\tau_1, \dots, \tau_m, \tau_{m+1}, \dots, \tau_{2m}]$ mit

$$\tau_k = \begin{cases} a & , \quad k \leq m \\ b & , \quad k > m \end{cases}$$

und \mathcal{F} ein EC-System. Dann nennen wir die zugehörigen Chebyshev-B-Splines b_k Bernsteinfunktionen und eine beliebige Linearkombination von Bernsteinfunktionen Bézierfunktion.

Fassen wir nun die Hauptaussagen dieses Abschnitts für die Bernsteinfunktionen nochmals zusammen.

Satz 4.26 Sei $a < b$ und $F = \langle 1, f_2, \dots, f_m \rangle$ ein EC-System auf $[a, b]$. Dann existieren eindeutig bestimmte Bernsteinfunktionen b_1, \dots, b_m auf $[a, b]$ mit den folgenden Eigenschaften:

- $b_k \geq 0$.
- $\sum_k b_k \equiv 1$.
- b_k besitzt in a eine $(k - 1)$ -fache Nullstelle.
- b_k besitzt in b eine $(m - k)$ -fache Nullstelle.

Außerdem sind die Bernsteinfunktionen eindeutig durch die oben genannten Eigenschaften bestimmt.

Nun haben wir die polynomiale Theorie auf nicht-polynomiale Bernsteinfunktionen verallgemeinert. Dabei haben wir aber die Einschlussschätzer außen vor gelassen. Dies hat zwei Gründe. Erstens hat sich der Ansatz ohne die Einschlussschätzer in den allermeisten Fällen als mindestens gleichwertig erwiesen. Zweitens aber gibt es ein weiteres Problem. Die Einschlussschätzer haben ausgenutzt, dass die polynomialen Bernsteinfunktionen unabhängig vom zugrunde liegenden Intervall sind. Somit war es möglich, die Bernsteinfunktionen einmal mit viel Aufwand genau zu untersuchen und scharfe stückweise lineare Einschlüsse zu bestimmen. Diese haben wir für verschiedene Ordnungen in Tabellen abgespeichert und dann bei der Arbeit mit den Einschlussschätzern auf diese Tabellen zurückgegriffen. Im Allgemeinen hängen die Bernsteinfunktionen aber vom zugrunde liegenden Intervall ab. Nun müsste man mit viel Aufwand obere und untere Einschlüsse in jedem Schritt berechnen, was aber weder theoretisch noch praktisch sinnvoll ist. Einen möglichen Ausweg bei der Verallgemeinerung der Einschlussschätzer wollen wir aber doch skizzieren. Nehmen wir im ersten Schritt an, dass die Kontrollpunkte zu polynomialen Bézierpolynomen gehören und berechnen untere und obere Einschlüsse. Nun ist eine möglichst scharfe Abschätzung zu finden, wie stark sich die nicht-polynomialen Bernsteinfunktionen von den polynomialen unterscheiden. Skaliert man nun den Maximalabstand mit dem absolut maximalen Kontrollpunkt und subtrahiert bzw. addiert man ihn zum unteren bzw. oberen Einschluss, so erhält man schließlich Einschlüsse für den nicht-polynomialen Fall. Das Problem hierbei ist, wirklich scharfe Abschätzungen für die Differenzen der polynomialen und nicht-polynomialen Bernsteinfunktionen zu finden.

4.1.1 Allgemeine Algorithmen

Kehren wir nun in den Kontext unserer Arbeit an einem Algorithmus zur Lösung von Gleichungssystemen zurück. Wie bereits in der Einleitung von Abschnitt 4 gesagt, können wir unser komplettes Vorgehen bei der Lösung von polynomialen Systemen übertragen, falls die zugrunde liegenden Funktionenräume EC-Systeme bilden. Wir arbeiten wieder mit der Darstellung der Gleichungen in Tensorprodukt-Bernsteinfunktionen. Für nicht-polynomiale Systeme muss der polynomiale Nullstellen-Algorithmus nur geringfügig modifiziert werden. Zunächst müssen wir die univariate Subdivision implementieren. Ihre multivariate Verallgemeinerung besteht dann offenbar wieder in der iterierten parallelen univariaten Subdivision. Außerdem benötigen wir einen Algorithmus zur Bestimmung von Kontrollpunkten für die univariate Identität.

Sprechen wir also zunächst über die Subdivision. Satz 4.24 sagt uns, wie wir bei der Implementation des univariaten Subdivisions-Algorithmus zur Subdivision vom Intervall $[a, b]$ auf $[a, y]$, bzw. auf $[y, b]$ vorgehen müssen. Als Input brauchen wir neben den Kontrollpunkten P die Dualfunktionen mit Ableitungen an den Stellen a, b und y . Zu beachten ist dabei, dass diese zu einer Basis f_1, \dots, f_m gehören mit den

Eigenschaften, dass $f_1 \equiv 1$ und f_2, \dots, f_m ebenfalls C-System ist.

```
function P=LSubdivide(P,A,B,Y)
% Input: P: Controllpoint-Columns for [a,b]
%        A: Dualfunctions at a, A(i,j)=D^(i-1) f^s_j(a) i,j \in 1:m
%        B: Dualfunctions at b, B(i,j)=D^(i-1) f^s_j(b) i,j \in 1:m
%        Y: Dualfunctions at y, Y(i,j)=D^(i-1) f^s_j(y) i,j \in 1:m
% Output: P: New Controlpoints for [a,y] (left part)

m=size(P,1);
d=size(P,2);
for j=1:m
    for k=j:m
        M      =[A(m-k+1,:);A(1:m-k,:);Y(1:j-1,:);B(1:k-j,:)];
        Psi(k,j)=det(M)/det(M(2:m,2:m));
    end
end

for j=2:m
    c      =(Psi(j:m,j)./Psi(j:m,j-1))*ones(1,d);
    P(j:m,:)=c.*P(j:m,:)+(1-c).*P(j-1:m-1,:);
end
```

Sollen die Kontrollpunkte für den rechten Teil $[y, b]$ berechnet werden, so ist die vorletzte Zeile durch

```
P(1:m-j+1,:)=a.*P(2:end-j+2,:)+(1-a).*P(1:m-j+1,:);
```

zu ersetzen. Wir erhalten so die Routine `RSubdivide`. Soll von $[a, b]$ auf $[\alpha, \beta] \subset [a, b]$ subdividiert werden, so spalten wir diesen Prozess in zwei Schritte auf. Zuerst subdividieren wir auf $[a, \beta]$ und dann auf $[\alpha, \beta]$.

Neben der Subdivision können wir obiges „DeCasteljau“-Schema natürlich auch zur Auswertung benutzen, schließlich gibt der erste bzw. letzte Kontrollpunkt den Wert an der Stelle a bzw. b an.

Alternativ schlagen wir neben dem obigen Algorithmus ein anderes Vorgehen zur Berechnung der Subdivisionsgewichte vor.

Satz 4.27 *Die Subdivision von $[a, b]$ auf $[\alpha, \beta]$ ist eine lineare Abbildung von \mathbb{R}^m nach \mathbb{R}^m . Sei $F = [f_1, \dots, f_m]$ eine Basis von \mathcal{F} und $K, \tilde{K} \in \mathbb{R}^{m \times m}$ so, dass*

$$FK = [b_1, \dots, b_m]$$

bzw.

$$F\tilde{K} = [\tilde{b}_1, \dots, \tilde{b}_m],$$

wobei b_k die Bernsteinfunktionen auf $[a, b]$ und \tilde{b}_k die Bernsteinfunktionen auf $[\alpha, \beta]$ seien. Dann ist die Subdivisionsmatrix gegeben durch

$$\tilde{K}^{-1}K.$$

Was wir brauchen, ist also eine Routine, mit der die Koeffizienten zu einer beliebigen Basis von \mathcal{F} für die Bernsteinfunktionen auf $[a, b]$ berechnet werden. Spezialisieren wir Satz 4.23 auf die Knotenfolge $[a, \dots, a, b, \dots, b]$, sodass wir im Bernsteinfall sind, dann erhalten wir:

Satz 4.28 Die Bernsteinfunktionen $b_k \in \mathcal{F}$ zur Knotenfolge $[a, \dots, a, b, \dots, b]$ sind eindeutig bestimmt durch die Eigenschaften

- $D^j b_k(a) = 0$, für $j \in 0 : k - 2$
- $D^j b_k(b) = 0$, für $j \in 0 : m - k - 1$,
- $\sum_k b_k \equiv 1$.

Wir können also wie folgt vorgehen: Zuerst bestimmen wir die Koeffizienten für b_1 bezüglich einer beliebigen Basis so, dass sich die entsprechenden Nullstellen einstellen und sich an der Stelle a als Wert 1 ergibt. Von b_1 merken wir uns alle Ableitungen an der Stelle a . Da die Summe aller b_k nun 1 ist, muss die Summe aller ersten Ableitungen der b_k folglich 0 sein. Dann bestimmen wir die Koeffizienten für b_2 so, dass wieder die entsprechenden Werte und Ableitungen verschwinden und sich an der Stelle a als erste Ableitung der negative Wert der ersten Ableitung von b_1 ergibt. Alle weiteren b_k besitzen in a mindestens eine doppelte Nullstelle. Nun addieren wir zu den Werten der Ableitungen von b_1 die von b_2 und fahren so sukzessive fort. Es ergibt sich der folgende Algorithmus.

```
function K=BernsteinCoefs(A,B)
% Input:  A: Basisfunctions at a, A(i,j)=D^(i-1) f_j(a)  i,j \in 1:m
%         B: Basisfunctions at b, B(i,j)=D^(i-1) f_j(b)  i,j \in 1:m
% Output: K: Coefficients such that
%         [f_1,...,f_n]*K=[b_1,...,b_n]

m=size(A,1);
K=zeros(m,m);           % zero-matrix of size mxm
D=[-1;zeros(m-1,1)];
for k=1:m
    M=[A(1:k,:);B(1:m-k,:)];
```

```

V=zeros(m,1);           % column of m zeros
V(k)=-D(k);
K(:,k)=M\V;             % M\V=R <=> M*R=V
D=D+A*K(:,k)
end;

```

Zu beachten ist, dass obige Strategie für höherdimensionale Funktionenräume \mathcal{F} ($m > 10$) nicht numerisch stabil ist. In der Praxis wird dadurch die Anwendbarkeit unseres Algorithmus limitiert. In obiger Routine wird eine Folge von Gleichungssystemen $M \setminus V$ gelöst. Dabei ist jeweils M eine $m \times m$ -Matrix. Alternativ lässt sich auch ein dünnbesetztes Gleichungssystem der Größe $m^2 \times m^2$ formulieren. Damit kann man versuchen, die gefundene Lösung nachzuiteilieren. Dies verbessert zwar die Ergebnisse, aber das Problem der Instabilität wird dadurch nicht wirklich gelöst.

Mit dem Algorithmus `BernsteinCoefs` ist nun auch die Bestimmung von Kontrollpunkten für die Identität kein Problem. Finden wir zunächst $K_{\text{Id}} \in \mathbb{R}^m$ so, dass $[f_1, \dots, f_m]K_{\text{Id}} = \text{Id}$, so sind mit $K = \text{BernsteinCoefs}(A, B)$ nun $(K^{-1}K_{\text{Id}})$ offenbar die Kontrollpunkte für die Identität, denn

$$\underbrace{[f_1, \dots, f_m]K}_{=[b_1, \dots, b_m]} (K^{-1}K_{\text{Id}}) = \text{Id}.$$

4.2 Exponentialräume

Für die Zwecke dieser Arbeit starten wir mit einem Funktionenraum \mathcal{F} auf einem Intervall I . Um unsere Ergebnisse anwenden zu können, müssen wir einerseits klären, ob es sich um ein EC-System handelt und, wenn wir nicht die obige Alternative zur Berechnung der Subdivisionsmatrix benutzen wollen, die Dualfunktionen stabil auswerten. Beides ist nicht einfach zu automatisieren. Dies limitiert unsere theoretisch sehr universellen Einsatzmöglichkeiten leider erheblich. Satz 4.4 stellt ein Kriterium zur Entscheidung bereit, ob ein Funktionenraum ein C-System bildet oder nicht. Dieses ist in der algorithmischen Praxis einsetzbar. Es ist aber nicht völlig zufriedenstellend, da mit ihm nur die Annahme eines C-Systems bestätigt, nicht aber widerlegt werden kann. Wünschenswert wären neue, bessere Kriterien. Die Suche nach solchen sprengt aber den Rahmen dieser Arbeit und ist mögliches Thema für weitere Forschung.

Im Fall von Exponentialräumen lässt sich unsere Theorie jedoch einwandfrei anwenden. Unser Vorgehen in diesem Fall ist als Modell zu verstehen, wie allgemein verfahren werden muss. Die Hauptfrage ist: Wie subdividiert man numerisch möglichst stabil und wann sind die zugrunde liegenden Funktionenräume EC-Systeme?

In diesem Kapitel befassen wir uns zunächst intensiv mit Exponentialräumen an sich und klären dann, wie wir zu numerisch anwendbaren Algorithmen kommen. Die Quintessenz dieses Abschnitts ist letztlich in den beiden (alternativen) Algorithmen `Subdivide` und im Algorithmus `LinearControlpoints` zusammengefasst.

Im weiteren Text setzen wir Grundwissen über lineare Differentialgleichungen voraus, welches z.B. in [Hi74] besprochen wird.

Um in Definition 4.31 zu klären, was wir unter einem Exponentialraum verstehen wollen, benötigen wir die folgende Definition 4.29 und das Lemma 4.30.

Definition 4.29 Sei $Q : K \rightarrow K$ ein Operator auf $K \subset \{f : \mathbb{R} \rightarrow \mathbb{C}\}$ und $p(x) = \sum_{i=0}^n \alpha_i x^i$ ein Polynom beliebigen Grades mit $\alpha_i \in \mathbb{C}$ und $\alpha_m = 1$. Wir definieren dann den Operator

$$p(Q) := \sum_{i=0}^m \alpha_i Q^i, \quad \text{wobei } Q^i := \underbrace{Q \circ \dots \circ Q}_i \text{ mal} \quad \text{und } Q^0 := \text{id}.$$

Ziehen wir einige einfache, aber nützliche Folgerungen:

Lemma 4.30 Im Fall, dass Q linear ist, folgt für zwei Polynome p und q :

1. $p(Q)$ ist linear. Insbesondere ist dann also $p(Q)0 = 0$,
2. $p(Q) \circ q(Q) = (pq)(Q) = q(Q) \circ p(Q)$,
3. $p(Q) = 0 \Rightarrow pq(Q) = 0$.

Definition 4.31 Wir nennen mit einem reellen Polynom p Funktionenräume der Form $\mathcal{F} = \{f : p(D)f = 0\}$ Exponentialräume.

Exponentialräume sind translationsinvariant, wie wir mit dem nächsten Lemma sehen.

Lemma 4.32 Es gilt für beliebige $\Delta \in \mathbb{R}$: $f \in \mathcal{F} \iff f(\cdot + \Delta) \in \mathcal{F}$.

Beweis $p(D)f(x) \equiv 0 \iff p(D)f(x + \Delta) \equiv 0$. □

Für Polynome p mit ausschließlich reellen Nullstellen ist damit für Funktionenräume nach Definition 4.31 klar, wie sie aufgebaut sind. Mit dem vorletzten Lemma können wir nämlich schließen, dass jedem Faktor $(D - \omega_k)^{m_k}$ von $p(D) = (D - \omega_1)^{m_1} \dots (D - \omega_{m_r})^{m_r}$ ein Teilraum von \mathcal{F} entspricht, nämlich der Kern des Operators $(D - \omega_k)^{m_k}$, also $\langle \exp(\omega_k x), x \exp(\omega_k x), \dots, x^{m_k-1} \exp(\omega_k x) \rangle$. Um Exponentialräume zu verstehen, für die das Polynom p auch komplexe Nullstellen besitzt, wollen wir noch präziser werden.

Lemma 4.33 Sei p ein reelles Polynom mit den Nullstellen ω_k . Die ω_k sind reell oder treten komplex konjugiert auf. Der zugehörige Exponentialraum \mathcal{F} besitzt eine Basis aus reellwertigen Funktionen. Setzen wir für reelle ω_k

$$\mathcal{F}_k = \langle \exp(\omega_k x), x \exp(\omega_k x), \dots, x^{m_k-1} \exp(\omega_k x) \rangle,$$

bzw. für komplexe $\omega_k = \nu_k + i\mu_k$

$$\begin{aligned} \mathcal{F}_k = & \langle \sin(\mu_k x) \exp(\nu_k x), x \sin(\mu_k x) \exp(\nu_k x), \dots, x^{m_k-1} \sin(\mu_k x) \exp(\nu_k x), \\ & \cos(\mu_k x) \exp(\nu_k x), x \cos(\mu_k x) \exp(\nu_k x), \dots, x^{m_k-1} \cos(\mu_k x) \exp(\nu_k x) \rangle, \end{aligned}$$

dann gilt

$$\mathcal{F} = \sum_k \mathcal{F}_k.$$

Alternativ gibt es eine andere reellwertige Basis, die sich für das weitere Vorgehen als vorteilhaft erweist. Diese werden wir im nächsten Lemma als \tilde{F} definieren.

Lemma 4.34 Sei $\mathcal{F} = \{f : p(D)f = 0\}$ mit $p(D) = \sum_{i=0}^m \alpha_i D^i f = (D - \omega_1) \cdots (D - \omega_m)f$. Dann können wir auch schreiben $\mathcal{F} = \{f : D^m f = \sum_{i=0}^{m-1} -\frac{\alpha_i}{\alpha_m} D^i f\}$. Es gilt also mit

$$G_W := \begin{bmatrix} 0 & 1 & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & 1 \\ \frac{-\alpha_0}{\alpha_m} & \frac{-\alpha_1}{\alpha_m} & \dots & \frac{-\alpha_{m-1}}{\alpha_m} \end{bmatrix} \quad \text{und} \quad \tilde{f} := [f, Df, \dots, D^{m-1}f]^T$$

$D\tilde{f} = G_W \tilde{f}$. Deshalb bildet die erste Zeile von $\exp(G_W \cdot)$ eine Basis von \mathcal{F} . Sei $\tilde{F} := [\exp(G_W \cdot)]_{1,:} = [\tilde{f}_1, \dots, \tilde{f}_m]$. \tilde{f}_k ist nun diejenige eindeutige Funktion in \mathcal{F} , für die $D^j \tilde{f}_k(0) = \delta_{j,k-1}$ für $j \in 0 : m-1$.

Formulieren wir noch den Algorithmus zum Aufstellen der Differenzialgleichungsmatrix G_W aus dem vorigen Lemma.

```
function G=DEQmatrix(W)
% Input: W: Vektor defining differential equation
% Output: G: Matrix of linear differential equation

m=length(W);
P=[-W(1) 1];
for k=2:m
    P=conv(P,[-W(k) 1]);
end
G=[zeros(m-1,1) eye(m-1);-P(1:m)];
```

Mit dieser zweiten Basis aus Lemma 4.34 werden wir im Folgenden stets arbeiten. Die erste Basis haben wir lediglich aus Gründen der Klarheit angegeben.

Definition 4.35 Wir setzen mit dem Verständnis des vorigen Lemmas

$$E(W, x) := \exp(G_W \cdot x) \quad \text{und} \quad \bar{E}(W, x) := [\exp(G_W \cdot x)]_{1,}$$

Zunächst wollen wir noch die folgende Tatsache bemerken.

Lemma 4.36 Für einen Exponentialraum \mathcal{F} gilt mit $f \in \mathcal{F}$ auch $Df \in \mathcal{F}$. Mit einer Funktion in \mathcal{F} befinden sich also auch alle ihre Ableitungen in \mathcal{F} .

Im Folgenden wollen wir annehmen, dass stets $1 \in \mathcal{F}$. Nun stellt sich die Frage, ob die hier betrachteten Funktionenräume auf Intervallen I C- bzw. EC-Systeme sind. Zunächst müssen wir feststellen, dass dies im Allgemeinen nicht der Fall ist (siehe Beispiel 4.7). Mit Satz 4.4 jedoch, ist für rein reelle W offenbar \mathcal{F}^W stets ein EC-System auf ganz \mathbb{R} . Für nicht rein reelle W genügt es, dass die Intervalllänge von I hinreichend klein ist. Für Funktionenräume $\mathcal{F} = \{p(D)f \equiv 0\}$, wobei p nur ein paar von komplex konjugierten Nullstellen $\alpha + i\beta$, $\alpha - i\beta$ besitzt, kann eine Schranke für die Intervalllängen angegeben werden. Falls nämlich die Intervalllänge von I kleiner als π/β ist, so handelt es sich bei \mathcal{F} auf I garantiert um ein EC-System.

Betrachten wir also im Folgenden ausschließlich EC-Systeme \mathcal{F} . Um uns bequem ausdrücken zu können, definieren wir

Definition 4.37 Sei $W = (\omega_1, \dots, \omega_m)$ und $h > 0$ und $a < b \in \mathbb{R}$. Wir setzen dann

$$\mathcal{F}^W := \{f : (D - \omega_1) \cdots (D - \omega_m)f = 0\}$$

und gemäß Definition 4.25

$$b_k^{W,[a,b]} := k\text{-te Bernsteinfunktion in } \mathcal{F}^W \text{ auf dem Intervall } [a, b].$$

Außerdem sei

$$b_k^{W,h} := b_k^{W,[0,h]}.$$

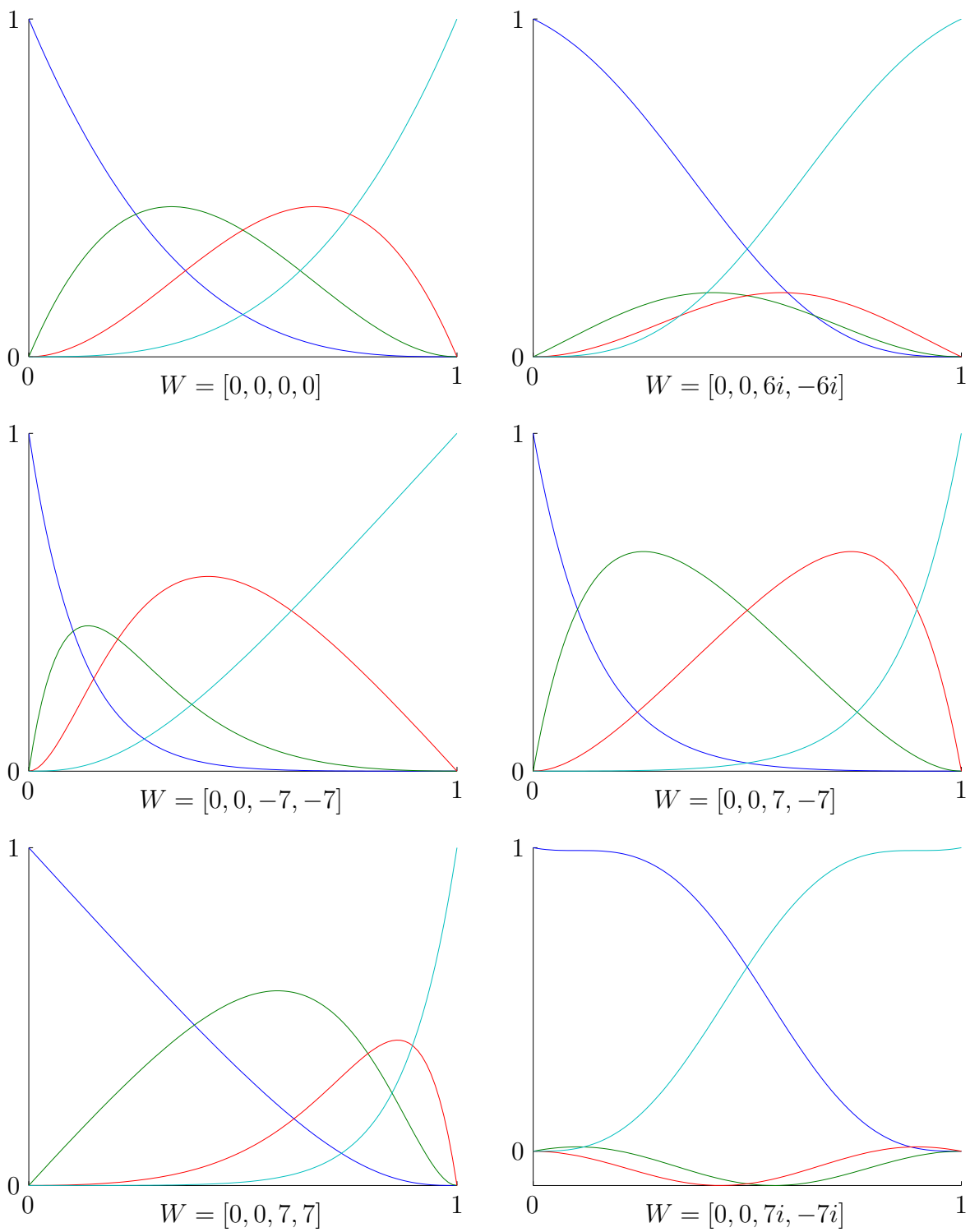
Entsprechend sei

$$B^{W,[a,b]} := [b_k^{W,[a,b]}, \dots, b_k^{W,[a,b]}] \quad \text{bzw.} \quad B^{W,h} := [b_k^{W,h}, \dots, b_k^{W,h}]$$

Zudem setzen wir $S_{[\alpha,\beta]}^{W,[a,b]}$ so, dass für beliebige Kontrollpunkte P gilt

$$B^{W,[a,b]}P = B^{W,[\alpha,\beta]} \left(S_{[\alpha,\beta]}^{W,[a,b]}P \right).$$

D.h. $S_{[\alpha,\beta]}^{W,[a,b]}$ sei diejenige lineare Abbildung, die Kontrollpunkten zu Bernsteinfunktionen auf $[a, b]$ neue Kontrollpunkte für Bernsteinfunktionen auf $[\alpha, \beta]$ zuordnet, sodass die zugehörigen Bézierfunktionen übereinstimmen.

Abbildung 21: Verschiedene $B^{W,1}$.

Zur Illustration der letzten Definition betrachten wir Abbildung 21. Zu beachten ist dass Für $W = [0, 0, 7i, -7i]$ der Funktionenraum $\mathcal{F}^{W,1}$ kein EC-System ist. Trotzdem existieren durch die in Satz 4.26 aufgeführten Eigenschaften eindeutig definierte Bernsteinfunktionen. Diese sind aber nicht mehr positiv.

Um auch stabil arbeiten zu können bei Intervalllängen, die gegen Null konvergieren, ist das folgende Lemma von fundamentaler Bedeutung.

Lemma 4.38 *Es gilt $f(\cdot) \in \mathcal{F}^W \iff f(h\cdot) \in \mathcal{F}^{hW}$.*

Beweis Sei $(D - \omega_1) \cdots (D - \omega_m)f = \sum_{i=0}^m \alpha_i D^i f = 0$. Dann ist

$$\begin{aligned} ((D - h\omega_1) \cdots (D - h\omega_m))f(hx) &= \sum_{i=0}^m \alpha_i h^{m-i} D^i(f(hx)) \\ &= \sum_{i=0}^m \alpha_i h^{m-i} (D^i f)(hx) h^i \\ &= 0. \end{aligned}$$

□

Aus Satz 4.28, Lemma 4.32 und Lemma 4.38 ergeben sich unmittelbar die beiden folgenden Sätze. Satz 4.39 erlaubt es uns, stets mit Bézierfunktionen auf $[0, 1]$ zu arbeiten, da sich eine Bézierfunktion auf $[a, b]$ durch Transformation des Definitionsgebiets und Skalierung von W stets ausdrücken lässt als eine Bézierfunktion auf $[0, 1]$.

Satz 4.39 *Es gilt*

$$B^{W,[a,b]}(x) = B^{W,b-a}(x-a) \quad \text{und} \quad B^{W,h}(x) = B^{hW,1}\left(\frac{x}{h}\right)$$

und somit

$$B^{W,[a,b]}(x) = B^{(b-a)W,1}\left(\frac{x-a}{b-a}\right).$$

Nach Satz 4.39 können wir nun in Satz 4.40 auch die Subdivision von $[a, b]$ auf $[\alpha, \beta]$ ausdrücken als Subdivision von $[0, 1]$ auf ein entsprechendes Teilintervall.

Satz 4.40 *Es gilt*

$$S_{[\alpha,\beta]}^{W,[a,b]} = S_{\left[\frac{\alpha-a}{b-a}, \frac{\beta-a}{b-a}\right]}^{(b-a)W,1}.$$

Gemäß Satz 4.40 können wir nun also die Routine `Subdivide` zur Subdivision von $[a, b]$ auf $[\alpha, \beta]$ formulieren. In Abschnitt 4.1.1 haben wir die Algorithmen `LSubdivide` und `RSubdivide` angegeben, mit deren Hilfe die Subdivision durchgeführt werden kann. Als Input erwarten diese das ausgewertete Dualsystem an verschiedenen Stellen. Klären wir deshalb nun in Lemma 4.41 und in Lemma 4.42 die Auswertung des Dualsystems.

Lemma 4.41 *Die Einträge des liegenden Vektors $F(x) := \bar{E}(W, x)$ sind eine Basis des EC-Systems \mathcal{F} im Sinne von Definition 4.1. Die hierzu gehörende Dualbasis ist*

$$F^*(a) = [\exp(-G_W \cdot a)]_{:,m}$$

Beweis Zu zeigen ist, dass $(D^k F)(a) \cdot F^*(a) = \delta_{k,m-1}$.
Es ist $(D^k F)(a) = [E(W, a)]_{k+1,:}$ also

$$\begin{aligned} (D^k F)(a) \cdot F^*(a) &= [E(W, a) \cdot \exp(-G_W \cdot a)]_{k+1,m} \\ &= [\text{Id}_m]_{k+1,m} = \delta_{k,m-1} \end{aligned}$$

□

Lemma 4.42 *Es gilt mit $\mathcal{F} := \mathcal{F}^W$:*

$$F^* = \{f : (D + \omega_1) \cdots (D + \omega_m) f = 0\} = \{f(x) : f(-x) \in F\}$$

Mit

$$[f_1^*, \dots, f_m^*] = [\exp(-G_W \cdot a)]_{:,m}$$

und

$$[g_1^*, \dots, g_m^*] = [\exp(G_{-W} \cdot a)]_{:,m}$$

gilt

$$g_k^* = (-1)^{m+k} f_k^*.$$

Beweis Man kann auch schreiben: $[f_1^*, \dots, f_m^*] = [\exp(G_W(-a))]_{:,m}$. Somit gilt mit Lemma 4.33: $f_1^* \in \{f(-x) : f(x) \in F\}$. Wegen Lemma 4.36 sind somit aber auch $f_k^*(x) = (D^{k-1} f_1^*)(-x) = (-1)^{k-1} (D^{k-1} f_1^*(x)) \in F^*$, was die erste Behauptung beweist. Außerdem gilt somit $D^j f_k^*(x) = (-1)^{k-1} \delta_{j+1,k}$. Für die g_k gilt $g_k^* = D^{k-1} g_1^*$ und $D^j g_1^* = \delta_{j+1,m}$, woraus $g_1^* = (-1)^{m-1} f_1^*$ folgt und damit die zweite Behauptung. □

Bis auf Skalierung ist das Dualsystem zu $\bar{E}(W, x)$ gegeben durch den Vektor $[E(-W, a)]_{:,m}$. Rufen wir uns Satz 4.24 in Erinnerung. Die α_k ergeben sich als Quotienten von Determinanten. Wird nun das Dualsystem bis auf f_1^* beliebig skaliert, so kürzt sich diese Skalierung in dieser Darstellung wieder heraus. Wenn wir also die α_k so berechnen wie dies mit Satz 4.24 vorgeschlagen wird, so können wir mit $[\exp(G_{-W} \cdot a)]_{:,m}$ als Dualsystem arbeiten, d.h. wir können nun mit Satz 4.40 den folgenden Algorithmus formulieren.

```
function P=Subdivide(P,W,a,b,alpha,beta)
% Input:  P: old controlpoints
%         W: vector defining differential equation
%         a,b: old intervall [a,b]
%         alpha,beta: new intervall [alpha,beta]
% Output: P: new controlpoints for intervall [alpha,beta]

m=size(P,1);
G=DEQmatrix(-W*(b-a));
A=zeros(m,m);
A(end,1)=1;
for k=2:m
    A(:,k)=G*A(:,k-1);
end;
P=LSubdivide(P,A,expm(DEQmatrix(-W*(b-a)))*A,...
             expm(DEQmatrix(-W*(b-a))*(beta-a)/(b-a))*A);
G=DEQmatrix(-W*(beta-a));
A=zeros(m,m);
A(end,1)=1;
for k=2:m
    A(:,k)=G*A(:,k-1);
end;
P=RSubdivide(P,A,expm(DEQmatrix(-W*(beta-a)))*A,...
             expm(DEQmatrix(-W*(beta-a))*(alpha-a)/(beta-a))*A);
```

Die wichtige Frage der Subdivision haben wir also zunächst geklärt. Mit Satz 4.27 haben wir aber eine Alternative zur Durchführung der Subdivision vorgeschlagen. Wir möchten auch diese Variante implementieren. In der Praxis zeichnet diese sich sogar noch gegenüber der obigen Variante aus.

Um uns im Folgenden bequem ausdrücken zu können, definieren wir wie folgt.

$$D(h) := \begin{bmatrix} 1 & & & 0 \\ & h^{-1} & & \\ & & \ddots & \\ 0 & & & h^{1-m} \end{bmatrix}.$$

Lemma 4.43 *Es gilt*

$$E(W \cdot (b-a), x) = E(W, x(b-a))D(1/(b-a)). \quad (4.4)$$

Beweis Seien $[f_1(x), \dots, f_m(x)] = E(W \cdot (b-a), x)$, $[g_1(x), \dots, g_m(x)] = E(W, x(b-a))$. Dann ist wegen Lemma 4.33 $g_k \in \langle f_1, \dots, f_m \rangle$. Außerdem ist $D^j g_k(x) = (b-a)^k \delta_{j+1,k}$. Gleichzeitig ist $D^j f_k(x) = \delta_{j+1,k}$, woraus $f_k = (b-a)^{-k} g_k$ folgt. \square

Wenn wir nun $S_{[\alpha, \beta]}^{W, [a, b]}$ bestimmen wollen, wo können wir unter Verwendung von Satz 4.27, Satz 4.39 und Lemma 4.43 wie folgt vorgehen. Zunächst gilt wegen Satz 4.40

$$S_{[\alpha, \beta]}^{W, [a, b]} = S_{\left[\frac{\alpha-a}{b-a}, \frac{\beta-a}{b-a}\right]}^{W(b-a), [0, 1]}.$$

Bestimmen wir K und \bar{K} so, dass

$$E(W \cdot (b-a), x) \cdot K = B^{W \cdot (b-a), 1}(x) \quad (4.5)$$

$$E(W \cdot (\beta - \alpha), x) \cdot \bar{K} = B^{W \cdot (\beta - \alpha), 1}(x). \quad (4.6)$$

Ersetzt man x durch $(\beta - \alpha)x$, dann gilt auch wegen Gleichheit der beiden Seiten in (4.4) mit der vorigen Gleichung

$$E\left(W \cdot (b-a), \frac{\beta - \alpha}{b-a}x\right) \cdot D\left(\frac{b-a}{\beta - \alpha}\right) \bar{K} = B^{W \cdot (b-a), \frac{\beta - \alpha}{b-a}}\left(\frac{\beta - \alpha}{b-a}x\right)$$

und somit

$$E(W \cdot (b-a), x) \cdot D\left(\frac{b-a}{\beta - \alpha}\right) \bar{K} = B^{W \cdot (b-a), \frac{\beta - \alpha}{b-a}}(x)$$

und damit

$$\begin{aligned} & E(W \cdot (b-a), x) \underbrace{\exp\left(G_{W \cdot (b-a)} \frac{a - \alpha}{b-a}\right) D\left(\frac{b-a}{\beta - \alpha}\right) \bar{K}}_{=: \tilde{K}} \\ &= B^{W \cdot (b-a), [0, \frac{\beta - \alpha}{b-a}]} \left(x - \frac{\alpha - a}{b-a}\right) \\ &= B^{W \cdot (b-a), [\frac{\alpha - a}{b-a}, \frac{\beta - a}{b-a}]}(x). \end{aligned}$$

Insgesamt ist also

$$S_{[\alpha,\beta]}^{W,[a,b]} = \tilde{K}^{-1}K = \bar{K}^{-1}D \left(\frac{\beta - \alpha}{b - a} \right) E \left(W \cdot (b - a), \frac{\alpha - a}{b - a} \right) K. \quad (4.7)$$

Wir können den Subdivisions-Algorithmus `Subdivide` also alternativ formulieren als:

```
function P=Subdivide(P,W,a,b,alpha,beta)

m=size(P,1);
K =BernsteinCoefs(eye(m),expm(DEQmatrix(W* (b-a))));
Kb=BernsteinCoefs(eye(m),expm(DEQmatrix(W* (beta-alpha))));
P=inv(Kb)...
    *diag([1 cumprod((beta-alpha)/(b-a)*ones(1,m-1))])...
    *expm(DEQmatrix(W*(b-a))*(alpha-a)/(b-a))...
    *K*P;
```

Für kleine m ($m < 10$) erzielen wir mit dieser Subdivisionsstrategie in der Praxis sehr gute Ergebnisse. Für große m erweist sich der Algorithmus dagegen als numerisch nicht stabil.

Bei der Verallgemeinerung des Vorgehens aus Abschnitt 3.1 zur Lösung von polynomialen Systemen auf nicht-polynomiale Gleichungssysteme gibt es nur zwei Unterschiede in den Algorithmen. Erstens haben wir einen neuen Subdivisionsalgorithmus zur univariaten Subdivision angegeben. Nun ist in der Routine `Roots` aus Abschnitt 3.1 zusätzlich zu protokollieren, in welchen Räumen die entsprechenden Funktionen liegen. Zweitens ist bei der Identifikation der Funktion mit ihrem Graphen darauf zu achten, dass die richtigen Kontrollpunkte gewählt werden, denn es gilt nun im Allgemeinen nicht mehr $\sum_k b_k(k-1)/(m-1) = id$. Die Kontrollpunkte für die Identität auf $[0, h]$ können aber leicht mit der folgenden Routine berechnet werden. Bei der Realisierung wird lediglich berücksichtigt, dass die Identität auf $[0, h]$ eindeutig festgelegt ist durch die Ableitung 1 in 0 sowie den Wert samt allen höheren Ableitungen gleich 0. Natürlich ist die Identität nur darstellbar, wenn sie auch ein Element von \mathcal{F}^W ist. Dies wiederum impliziert $\omega_i = \omega_j = 0$ für $i \neq j$.

Wir weisen an dieser Stelle darauf hin, dass wir

```
function LP=LinearControlpoints(W,h)
% Input:   W: vector defining differential equation
%          h: intervall-length
% Output: LP: controlpoints of function (x-a) for intervall [a,a+h]

m=size(P,1);
V=[0;1;zeros(m-2,1)];
LP=BernsteinCoefs(eye(m),expm(DEQmatrix(W*h)))\V;
```

Bevor wir diesen Abschnitt beschließen, wollen wir nun noch einmal die Gleichungen (4.5), (4.6),(4.7) im rein polynomialen Fall genauer betrachten. Nun ist $W = [0, \dots, 0]$ und somit gilt $\bar{K} = K$. Außerdem ist

$$E \left(W \cdot (b - a), \frac{\alpha - a}{b - a} \right) = \begin{bmatrix} 1 & h & \frac{h^2}{2} & \cdots & \frac{h^{m-1}}{(m-1)!} \\ & \ddots & \ddots & \ddots & \vdots \\ & & \ddots & \ddots & \frac{h^2}{2} \\ & & & \ddots & h \\ & & & & 1 \end{bmatrix}. \quad (4.8)$$

Satz 4.44 Die Subdivisionsmatrix $S_{\alpha,\beta}^{[0,1]} \in \mathbb{R}^{m \times m}$ zur Subdivision von $[0, 1]$ auf $[\alpha, \beta] \subset [0, 1]$ im polynomialen Fall lässt sich berechnen als Produkt von vier Matrizen. Es gilt mit

$$\begin{aligned} K_{i,j} &= \binom{i-1}{j-1} \frac{(m-i)!}{(m-1)!} \\ K_{i,j}^{-1} &= \binom{i-1}{j-1} \frac{(m-j)!}{(m-1)!} \\ E &= \begin{bmatrix} 1 & h & \frac{h^2}{2} & \cdots & \frac{h^{m-1}}{(m-1)!} \\ & \ddots & \ddots & \ddots & \vdots \\ & & \ddots & \ddots & \frac{h^2}{2} \\ & & & \ddots & h \\ & & & & 1 \end{bmatrix} \\ D_{i,j} &= \delta_{i,j} (\beta - \alpha)^{i-1} \end{aligned}$$

nun

$$S_{\alpha,\beta}^{[0,1]} = K^{-1} D E K.$$

Offenbar sind K^{-1} und K unabhängig von α, β und D, E von sehr einfacher Gestalt. In der Programmiersprache MATLAB sind Matrixoperationen billig und Schleifen teuer. Der klassische DeCasteljau muss in einer Schleife berechnet werden, mit obigem Ansatz kommen wir ohne Schleifen aus. Es ergibt sich der folgende Algorithmus. Die Routine `PolSubdivideInit` braucht für festes m nur ein einziges Mal ausgeführt werden. Anschließend leistet die Routine `PolSubdivide` die Subdivision. Dabei liefere `Binom` die Matrix $B_{i,j} = \binom{i-1}{j-1}$ und `cumprod(A)` das kumulative Produkt, also $[A(1), A(1) \cdot A(2), \dots, \prod_k A(k)]$.

```
function [K,Kinvers,Ind]=PolSubdivideInit(m)
% Input : m: Order m of bernstein-polynomials
```

```

% Output: K,Kinvers,Ind: input for function PolSubdivide not depending on
%          subdivision-intervals

B=Binom(0:m-1,0:m-1);
K=diag([1 cumprod(m-1:-1:1)])*B...
    *(cumprod(-ones(m,1))*cumprod(-ones(1,m)));
Kinvers=B*diag(1./[1 cumprod(m-1:-1:1)]);
Ind=eye(m)+tril((m+1)*ones(m,m),-1);
for k=2:m
    Ind=Ind+diag(k*ones(1,m+1-k),k-1);
end;

function P=PolSubdivide(P,alpha,beta,Ind,K,Kinvers)
% Input: P: controlpoints for interval [0,1]
%        alpha,beta: new interval [alpha,beta]
%        Ind,K,Kinvers: output of PolSubdivideInit for m=size(P,1)
% Output: P: controlpoints for interval [alpha,beta]

m=size(P,1);
H=[1 cumprod(alpha*ones(1,m-1))./cumprod(1:m-1) 0];
P=Kinvers*diag([1;cumprod((beta-alpha)*ones(m-1,1))])*H(Ind)*K*P;

```

Zum Schluss wollen wir noch darauf eingehen, wie die in diesem Abschnitt gegebenen Bausteine zu einem Algorithmus zur Lösung exponentieller Gleichungssysteme zusammengesetzt werden. Bei der Arbeit mit exponentiellen Bézierfunktionen kann der in Kapitel 3 formulierte Algorithmus eins zu eins übernommen werden. Zuerst muss allerdings festgestellt werden, ob die zugrunde liegenden Räume allesamt EC-Systeme sind (siehe dazu Satz 4.4). Andernfalls sind die Bernsteinfunktionen nicht notwendigerweise überall größer als Null und somit ist die Konvexe-Hülle-Eigenschaft der Funktion in Bezug zu ihren Kontrollpunkten nicht mehr gewährleistet. Dies wiederum war aber eine fundamentale Voraussetzung unseres Algorithmus. Ohne die Positivität ist der Algorithmus unrettbar verloren. Zweitens müssen natürlich verschiedene Bausteine ausgetauscht werden. Zunächst müssen die zugrunde liegenden Funktionenräume protokolliert werden. Dann muss in der Routine `FindHyperplaneEnclosure` die Bereitstellung der Kontrollpunkte für die Identität gemäß der Routine `LinearControlpoints` erfolgen. Außerdem ist die Subdivision mit der Routine `Subdivide` durchzuführen. Hiermit haben sich die Unterschiede schon erschöpft.

Satz 4.45 *Auch im nicht-polynomialen Fall ergibt sich quadratische Konvergenz des Nullstellen-Algorithmus sofern die Jacobi-Matrix in einer Lösung invertierbar ist.*

Beweis Auch im nicht-polynomialen Fall kann man einen Δ^2 -Operator definieren und zeigen dass die Größe $|\Delta^2|_{\max}$ quadratisch gegen Null geht. Daraus folgt dann die quadratische Konvergenz des Kontrollpolygons gegen die dargestellte Funktion. Somit überträgt sich der Beweis im polynomialen Fall. \square

Wir wollen noch kurz bemerken, dass auch hier dieselbe Reduktion des Aufwands erzielt werden kann, sofern die Voraussetzungen von Satz 3.11 erfüllt sind. Das gesamte Vorgehen lässt sich dann unmittelbar übertragen.

Beispiel 4.46 Sei das globale Minimum der Funktion

$$f(s, t) = 4 \left(s - \frac{1}{2} \right)^4 + 2 \left(t - \frac{1}{2} \right)^2 - \frac{t + s - \frac{1}{2}}{5} - \frac{\exp(s - t)}{10}$$

gesucht. Zur Bestimmung der kritischen Punkte ist f nach s bzw. t abzuleiten und diejenigen Paare (s, t) zu bestimmen, für die beide Ableitungen verschwinden. Man kann relativ einfach zeigen, dass das Minimum im Intervall $[-1.5, 2.5]^2$ liegen muss. Bereits nach 13 Schritten ist die einzige kritische Stelle $(0.7740, 0.5177)$ mit Genauigkeit 10^{-12} eingegrenzt. Siehe auch Abbildung 22.

4.3 Uniforme Splines in Exponentialräumen

Bei der Lösung exponentieller Gleichungssysteme können wir nun anstatt mit der Darstellung in Bézierform auch mit der in uniformen Splines arbeiten und den Knotenabstand sukzessive verkleinern. Dies scheint auf den ersten Blick keinerlei Vorteile zu bieten, sondern nur Nachteile. Es wird sich aber bei näherer Betrachtung herausstellen, dass dieser Ansatz an zwei Stellen Vorteile aufweist. Erstens müssen wir bisher entscheiden, ob zu einer vorgegebenen Intervalllänge ein EC-System vorliegt. Diese Frage haben wir positiv beantwortet, falls kein Sinus oder Cosinus auftaucht. Tauchen aber solche Funktionen auf, so ist die Frage, ob ein EC-System vorliegt, nicht einfach zu beantworten. Im Kapitel 4.6 werden wir anhand eines Beispiels zeigen, wie man sich einer Antwort nähert. Dies stellt sich aber als nicht einfach und als schwer zu automatisieren heraus. Bei der Arbeit mit uniformen exponentiellen Splines können wir die Frage nach dem EC-System durch die Frage der Entartung ersetzen und diese sehr elegant beantworten. Zweitens ist die Berechnung der Subdivisionsmatrizen recht aufwändig und für große Ordnungen problematisch. Im Fall uniformer Splines in Exponentialräumen lassen sich die Subdivisionsgewichte explizit angeben und sind stabil berechenbar.

Gegenstand unserer weiteren Untersuchungen werden nun uniforme Splines zur Knotenfolge $h\mathbb{Z}$ sein. Wir werden eine Fülle von Erkenntnissen bekommen, der Weg dorthin

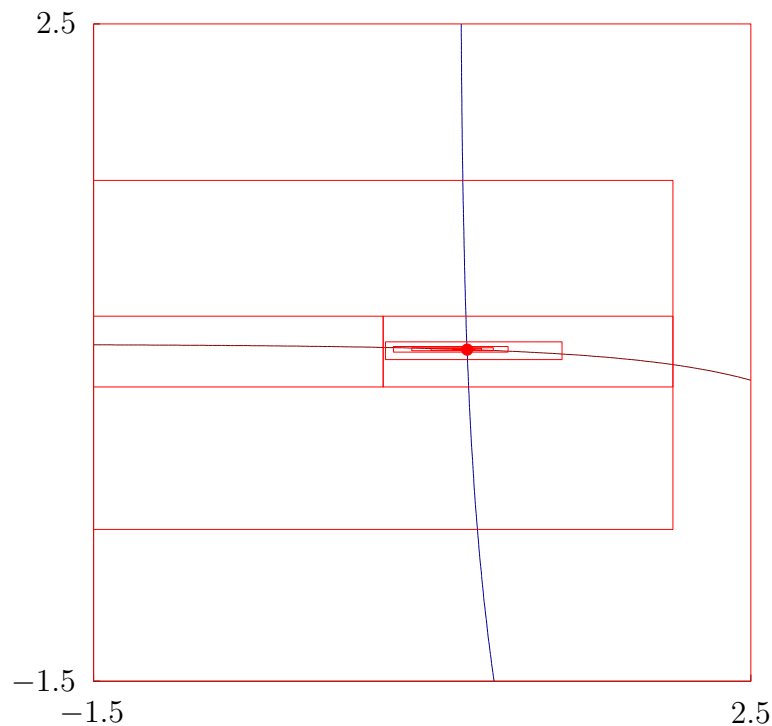


Abbildung 22: Unterteilungsstruktur im nicht-polynomialen Fall

ist aber sehr verschieden von unserem Vorgehen in Abschnitt 4 bzw. 4.2. Deshalb wollen wir die entsprechenden Definitionen neu formulieren, ohne dabei aber den alten zu widersprechen, sodass dieser Abschnitt keine Inhalte aus den vorherigen Abschnitten voraussetzt.

Die Inhalte dieses Abschnitts sind teilweise der Diplomarbeit [Gaukel99] entnommen.

Definition 4.47 (*glatte Potenz, Stutzfunktion, entartet, nicht-entartet, lokale Basis, Splines*) Sei \mathcal{F} ein Funktionenraum der Dimension m mit der Eigenschaft, dass zu jedem $a \in \mathbb{R}$ eine eindeutige Funktion $f \in \mathcal{F}$ existiert, sodass $D^k f(a) = \delta_{m-1,k}$ für $k \in 0 : m - 1$ gilt. Wir setzen nun

$$H(\mathcal{F}|x, a) := f(x)$$

und

$$T(\mathcal{F}|x, a) := \begin{cases} 0 & , x < a \\ H(\mathcal{F}|x, a) & , x \geq a \end{cases}$$

und nennen $H(\mathcal{F}|x, a)$ glatte Potenz in \mathcal{F} bzw. $T(\mathcal{F}|x, a)$ Stutzfunktion. Sei außerdem $h > 0$. Dann setzen wir $\Lambda^{\mathcal{F},h}$ als die Menge derjenigen Funktionen f , für die gilt:

$$\begin{aligned} f|_{[kh, (k+1)h)} &\in \mathcal{F}, & \forall k \in \mathbb{Z} \\ f &\in C^{m-2}(\mathbb{R}) \\ \text{supp}(f) &\subset [M, \infty) & \text{für ein } M \in \mathbb{R} \end{aligned}$$

und bezeichnen diese als Splines. Der Funktionenraum $\Lambda^{\mathcal{F},h}$ heißt nicht-entartet, falls die Folge $(H(\mathcal{F}|x, kh))_k$ eine lokale Basis bildet, falls also für beliebiges k gilt:

$$\langle H(\mathcal{F}|x, kh), \dots, H(\mathcal{F}|x, (k+m-1)h) \rangle = \mathcal{F}.$$

Im anderen Fall nennen wir $\Lambda^{\mathcal{F},h}$ entartet. Sei $W = (\omega_1, \dots, \omega_m)$ ein Vektor komplexer Zahlen, wobei komplexe Einträge stets als Paar komplex konjugierter Zahlen auftreten. Wir setzen dann

$$\mathcal{F}^W := \{f : (D - \omega_1) \cdots (D - \omega_m)f = 0\}$$

als den auf \mathbb{R} reellwertigen Lösungsraum zur entsprechenden Differenzialgleichung.

Die Forderung, dass in \mathcal{F} eine eindeutige Funktion f existiert, sodass $D^k f(a) = \delta_{m-1,k}$ für $k \in 0 : m-1$ gilt, ist äquivalent dazu, dass die Wronski-Matrix $W_{i,j} = D^{i-1} f_j(a)$ für eine Basis f_1, \dots, f_m von \mathcal{F} für beliebiges a invertierbar ist.

Zunächst bemerken wir, dass für beliebige $\omega_1, \dots, \omega_m$ die Wronski-Matrix in Definition 4.47 invertierbar ist. Dies folgt unmittelbar aus der Tatsache, dass \mathcal{F}^W die Lösungsmenge einer linearen Differenzialgleichung mit konstanten Koeffizienten ist. Somit machen die in Definition 4.47 definierten Bezeichnungen $H(\mathcal{F}^W|x, a)$, $T(\mathcal{F}^W|x, a)$ und $\Lambda^{\mathcal{F}^W,h}$ für beliebige W Sinn.

Sofort können wir nun einen sehr wichtigen und interessanten Satz formulieren. Mit diesem kann sehr elegant geklärt werden, ob für einen Knotenabstand h der Spline-raum $\Lambda^{\mathcal{F},h}$ entartet ist oder nicht. Die Frage der Entartung ersetzt später im Algorithmus `Roots` mit uniformen Splines die Frage, ob ein EC-System vorliegt.

Satz 4.48 Sei $W = (\omega_1, \dots, \omega_m)$, $\mathcal{F} := \mathcal{F}^W$ und $h \in \mathbb{R}_{>0}$. Die folgenden Aussagen sind nun äquivalent:

$$\text{Der Raum } \Lambda^{\mathcal{F},h} \text{ ist nicht entartet.} \quad (4.9)$$

$$\iff \text{Die Matrix } B_{i,j} = (D^i H(\mathcal{F}|jh, 0))_{i,j \in 0:m-1} \text{ ist invertierbar.} \quad (4.10)$$

$$\iff \text{Die Matrix } V_{i,j} = (f_{i+1}(jh))_{i,j \in 0:m-1} \text{ ist invertierbar} \\ \text{für eine beliebige Basis } f_1(t), \dots, f_m(t) \text{ von } \mathcal{F}. \quad (4.11)$$

$$\iff \exp(\omega_i h) \neq \exp(\omega_j h) \text{ für } i, j \in 1 : m \text{ mit } \omega_i \neq \omega_j. \quad (4.12)$$

$$\iff h \notin \mathbb{Z} \left\{ \frac{2\pi I}{\omega_i - \omega_j} : 1 \leq i < j \leq n \text{ für } \omega_i \neq \omega_j \right\}. \quad (4.13)$$

Beweis Der Beweis ist wie folgt aufgebaut:

1. (4.9) \iff (4.10)
2. Im Fall paarweise verschiedener ω_k : (4.10) \iff (4.11) \iff (4.12)
3. Im Fall beliebiger ω_k : (4.10) \iff (4.12)

4. Im Fall beliebiger ω_k : (4.12) \iff (4.11)

5. (4.12) \iff (4.13).

1. Der erste Teil folgt unmittelbar aus der Definition der Entartung.

2. Seien die ω_k paarweise verschieden. Wählen wir dann β_1, \dots, β_m so, dass die Stutzfunktion $H(\mathcal{F}|x, 0) = [\exp(\omega_1 x), \dots, \exp(\omega_m x)][\beta_1, \dots, \beta_m]^T$. Die β_k sind dabei ungleich Null, denn wäre z.B. $\beta_m = 0$, dann wäre die Stutzfunktion $H(\mathcal{F}|x, 0)$ ein Element in $\langle \exp(\omega_1 t), \dots, \exp(\omega_{m-1} t) \rangle$ und somit eindeutig festgelegt durch den Funktionswert ($= 0$) und die erste bis $(m-2)$ -te Ableitung ($= 0$) an der Stelle $x = 0$ müsste also konstant gleich Null sein im Widerspruch zu $D^{m-1}H(\mathcal{F}|0, 0) = 1$.

Es gilt nun für $j \in \mathbb{Z}$ mit $a(x) := H(\mathcal{F}|x, 0)$:

$$a(jh) = [\exp(\omega_1 0), \dots, \exp(\omega_m 0)][\beta_1 \exp(\omega_1 jh), \dots, \beta_m \exp(\omega_m jh)]^T. \quad (4.14)$$

Also

$$B = (a^{(i)}(jh))_{i,j \in 0:m-1} = GKV,$$

wobei

$$G = \begin{bmatrix} 1 & \dots & 1 \\ \omega_1^1 & \dots & \omega_m^1 \\ \vdots & \ddots & \vdots \\ \omega_1^{m-1} & \dots & \omega_m^{m-1} \end{bmatrix} \quad \text{und} \quad K = \begin{bmatrix} \beta_1 & & & 0 \\ & \beta_2 & & \\ & & \ddots & \\ 0 & & & \beta_m \end{bmatrix}$$

und mit $\lambda_k = \exp(\omega_k h)$

$$V = \begin{bmatrix} 1 & \lambda_1^1 & \dots & \lambda_1^{m-1} \\ 1 & \lambda_2^1 & \dots & \lambda_2^{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_m^1 & \dots & \lambda_m^{m-1} \end{bmatrix}$$

ist. G und V sind Vandermonde-Matrizen, somit sind wir mit dem Beweisteil fertig, denn wenn Behauptung (4.11) für eine Basis gilt, dann gilt sie auch für beliebige Basen. Für den nächsten Beweisteil brauchen wir noch explizit die Determinante von B , $\det(B) = \det(G) \det(K) \det(V)$. Wir stellen fest, dass

$$\det(G) = \prod_{1 \leq i < j \leq m} (\omega_j - \omega_i).$$

Mit der Cramerschen Regel erhalten wir die β_k als Lösung von $G[\beta_1, \dots, \beta_m]^T = [0, \dots, 0, 1]^T$ und somit

$$\beta_k = \frac{\det(R^k)}{\det(G)}, \quad \text{wobei} \quad R_{i,j}^k := \begin{cases} G_{i,j} & : j \neq k \\ \delta_{i,m} & : j = k \end{cases}.$$

Somit folgt

$$\det(K) = \prod_{k=1}^n \frac{\det(R^k)}{\det(G)}.$$

Man beachte nun, dass in R^k die k -te Spalte und m -te Zeile gestrichen werden kann und die Determinante höchstens das Vorzeichen ändert. Es ergibt sich also

$$\det(R^k) = (-1)^{m+k} \prod_{\substack{1 \leq i < j \leq m \\ i, j \neq k}} (\omega_j - \omega_i),$$

also

$$\frac{\det(R^k)}{\det(G)} = \frac{1}{\prod_{i \in \{1, \dots, m\}, i \neq k} (\omega_k - \omega_i)}.$$

Außerdem ist

$$\det(V) = \prod_{1 \leq i < j \leq m} (\exp(\omega_j h) - \exp(\omega_i h)),$$

also

$$|\det(B)| = \prod_{1 \leq i < j \leq n} \frac{\exp(\omega_j h) - \exp(\omega_i h)}{\omega_j - \omega_i}.$$

Die Determinante von B wäre also berechnet. Ganz nebenbei haben wir übrigens eine explizite Darstellung der Stutzfunktion im Fall einfacher ω_i gefunden. Es ist nach (4.14)

$$H(\mathcal{F}|x, a) = \sum_{i=1}^m \frac{\exp(\omega_i x)}{\prod_{j=1, j \neq i}^m (\omega_i - \omega_j)} \exp(-\omega_i a). \quad (4.15)$$

3. Seien nun die Nullstellen ω_i beliebig und $p(s) = (s - \omega_1)^{m_1} \dots (s - \omega_k)^{m_k}$. Wir zeigen (4.10) \iff (4.12).

Die Determinante von B hängt stetig von den ω_i ab,

$$H(\mathcal{F}|x, 0) = (\exp(\tilde{A}x)[0, \dots, 0, 1]^T)_1.$$

Für einfache ω_i haben wir sie explizit berechnet. Für mehrfache Nullstellen können wir die Determinante als Grenzwert berechnen. Lassen wir also ω_i und ω_j

zusammenfallen, dann ist in obigem Produkt der entsprechende Faktor durch $h \exp(\omega_i h)$ (Ableitung von $\exp(\omega_i h)$ nach ω_i) zu ersetzen.

Es ist nun für mehrfache ω_i

$$\det(B) = \pm \prod_{1 \leq i < j \leq k} \frac{\exp(\omega_j h) - \exp(\omega_i h)}{\omega_j - \omega_i} \prod_{i=1:k} (h \exp(\omega_i h))^{m_i-1}.$$

Die Determinante verschwindet folglich auch für allgemeine ω_i genau für die behaupteten Werte von h .

4. Sei nun $h \in \mathbb{R}_{>0}$ und $p(s)$ wie im Beweisteil 3. Wir wählen die Basis

$$\begin{aligned} f_1(t) &= t^{m_1-1} \exp(\omega_1 t) \\ f_2(t) &= t^{m_1-2} \exp(\omega_1 t) \\ &\vdots \\ f_{m_1}(t) &= t^0 \exp(\omega_1 t) \\ f_{m_1+1}(t) &= t^{m_2-1} \exp(\omega_2 t) \\ &\vdots \end{aligned}$$

Wenn die Äquivalenz (4.12) \iff (4.11) für diese spezielle Basis gilt, dann auch für beliebige Basen. Die Matrix V hat die Form

$$V = \begin{bmatrix} V^1 \\ V^2 \\ \vdots \\ V^k \end{bmatrix}$$

mit Blöcken

$$V^k = \text{diag}(1, h, \dots, h^{m_k-1}) \begin{bmatrix} 1 & \lambda_k & \lambda_k^1 & \lambda_k^2 & \dots & \lambda_k^{m-1} \\ 0 & \lambda_k & 1^1 \lambda_k^1 & 2^1 \lambda_k^2 & \dots & m^1 \lambda_k^{m-1} \\ 0 & \lambda_k & 1^2 \lambda_k^1 & 2^2 \lambda_k^2 & \dots & m^2 \lambda_k^{m-1} \\ \vdots & & & & & \vdots \\ 0 & \lambda_k & 1^{m_k-1} \lambda_k^1 & 2^{m_k-1} \lambda_k^2 & \dots & m^{m_k-1} \lambda_k^{m-1} \end{bmatrix},$$

wobei $\lambda_k = \exp(\omega_k h)$ ist. O.B.d.A. für die Invertierbarkeit von V sei $h = 1$. Wenn man sukzessive für $i = 1 : n-2$ in den j -ten Zeilen ($j > i$) den i -ten Eintrag durch Addition eines Vielfachens der i -ten Zeile zur j -ten Zeile zu Null macht und die so erhaltenen Blöcke zusammensetzt, dann erhält man gerade dieselbe Matrix, wie beim Interpolationsproblem (bei Polynomen) mit mehrfachen Stützstellen λ_i und gesuchten Koeffizienten für die Monomform. Dieses Problem ist stets eindeutig lösbar. V muß also für paarweise verschiedene λ_i invertierbar sein.

5. Der letzte Teil ist trivial. □

Um in Definition 4.51 uniforme Splines erklären zu können, benötigen wir die folgende Definition.

Definition 4.49 (*Verschiebe-Operator*) Sei V_h der lineare Operator, der einer Funktion f die Funktion $f(\cdot - h)$ zuordnet. Wir nennen diesen Verschiebe-Operator.

V_h verschiebt also eine Funktion um h nach rechts. Das folgende Lemma 4.50 ist nun von fundamentaler Bedeutung für Definition 4.51. Erst im Licht von Lemma 4.50 macht Definition 4.51 nämlich Sinn, wie wir dann in Satz 4.52 sehen werden.

Lemma 4.50 Für ein Polynom $p \in \mathbb{P}_n$ gilt

$$(V_h - \exp(-\omega h))^n p(x) \exp(\omega x) \equiv 0.$$

Beweis Wegen $(p(x) - p(x - h)) \in \mathbb{P}_{n-1}$ sieht man induktiv, dass

$$\begin{aligned} & (V_h - \exp(-\omega h))^n p(x) \exp(\omega x) \\ &= \exp(-\omega h) (V_h - \exp(-\omega h))^{n-1} (p(x - h) - p(x)) \exp(\omega h) \equiv 0. \end{aligned}$$

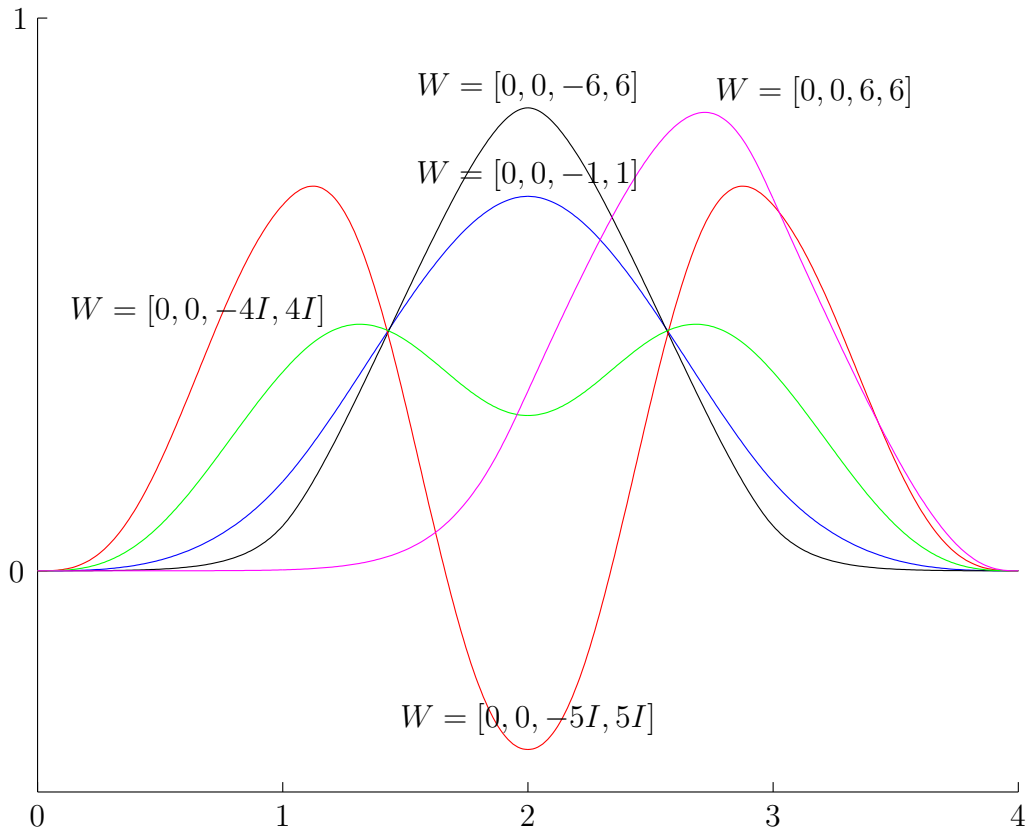
□

Definieren wir nun die Chebyshev-B-Splines in Splineräumen $\Lambda^{\mathcal{F}^W, h}$.

Definition 4.51 (*uniforme Chebyshev-B-Splines*) Sei $W \in \mathbb{C}^m$ ein Vektor von komplexen Zahlen mit $\omega_1 = 0$, sodass $\prod_k (x - \omega_k)$ ein reelles Polynom ist. Wir setzen mit dem Verständnis von Lemma 4.30 und von Lemma 4.50 für ein $h > 0$, für das $\Lambda^{\mathcal{F}^W, h}$ nicht entartet ist:

$$b_k^{W, h}(x) := \alpha_h \left(\prod_{j=1}^m (V_h - \exp(-\omega_j h)) \right) (T(\mathcal{F}^W | x, kh)), \quad (4.16)$$

wobei α_h unabhängig von k so gewählt ist, dass $\sum_k b_k^{W, h} \equiv 1$, siehe Abbildung 23. Wir nennen die Funktionen $b_0^{W, 1}$ uniforme exponentielle Chebyshev-B-Splines bzw. kurz uniforme Chebyshev-B-Splines.

Abbildung 23: Verschiedene $b_0^{W,1}$

Wir haben in Definition 4.51 gefordert, dass die Chebyshev-B-Splines so zu normieren seien, dass sie eine Partition der Eins bilden. Dass dies möglich ist, zeigen wir mit dem nächsten Satz. Die Bezeichnung $b^{W,h}$ für die Chebyshev-B-Splines kollidiert strenggenommen mit unserer Bezeichnung für die Bernsteinfunktionen in Abschnitt 4.2. Da wir aber in diesem Kapitel ausschließlich über Chebyshev-B-Splines reden werden, dürften Missverständnisse ausgeschlossen sein.

Dass Definition 4.51 sinnvoll ist, sehen wir nun in Satz 4.52. Alle wichtigen Eigenschaften der Chebyshev-B-Splines lassen sich dann recht schnell folgern. Die Positivität der Chebyshev-B-Splines ist nicht grundsätzlich garantiert, allerdings stets für hinreichend kleine h . Die Frage der Positivität werden wir aber erst in Satz 4.58 angehen.

Satz 4.52 *Es gilt mit den Bezeichnungen der letzten Definition, falls $\Lambda^{\mathcal{F}^W, h}$ nicht entartet ist:*

1. $b_k^{W,h}(x) = b_{k-1}^{W,h}(x-h)$ (Translationsinvarianz)
2. $\text{supp}(b_k^{W,h}) = [kh, (k+m)h]$ (kompakter Träger)

3. $b_k^{W,h} \in C^{m-2}$ (Differenzierbarkeitsordnung)
4. Die Folge $(b_k^{W,h})_k$ bildet eine lokale Basis von $\Lambda^{W,h}$ (lokale Basis)
5. $\sum_k b_k^{W,h} \equiv 1$ (Partition der Eins)

Beweis Solange die fünfte Behauptung nicht gezeigt ist, setzen wir der Einfachheit halber $\alpha_h = 1$, was die ersten vier Behauptungen nicht tangiert. Sei ferner $\mathcal{F} := \mathcal{F}^W$.

1. Die erste Behauptung ist unmittelbar klar wegen $T(\mathcal{F}|x, kh) = T(\mathcal{F}|x - h, (k - 1)h)$.
2. Es gilt wegen Lemma 4.30 und Lemma 4.50 für jede Linearkombination der $p_j(x) \exp(\omega_j x)$ und damit insbesondere für $H(\mathcal{F}|x, kh)$

$$\prod_{j=1}^m (V_h - \exp(-\omega_j h)) H(\mathcal{F}|x, kh) \equiv 0$$

und damit die Behauptung.

3. Da $T(\mathcal{F}|x, kh) \in C^{m-2}$, ist die Behauptung trivial.
4. Es ist

$$\left[b_{k-m+1}^{W,h}(x), \dots, b_k^{W,h}(x) \right] = [H(\mathcal{F}|x, (k - m + 1)h), \dots, H(\mathcal{F}|x, kh)] M,$$

wobei M eine untere Dreiecksmatrix ist, deren Diagonalen bzw. Nebendiagonalen aus jeweils identischen Einträgen bestehen. Die Hauptdiagonale ist ungleich Null, somit ist M invertierbar, was die Behauptung beweist.

5. Zu zeigen ist, dass ein α_h existiert, sodass $f := \sum_k b_k^{W,h} \equiv 1$. Wegen dem ersten Beweisteil ist f eine h -periodische Funktion in $\Lambda^{\mathcal{F}^W, h}$, die wegen (4.) sicherlich nicht komplett verschwindet. Wenn wir nun zeigen können, dass es in \mathcal{F} bis auf Skalierung nur eine einzige Funktion gibt, deren Wert und erste bis $(m - 2)$ -te Ableitung an den Stellen 0 und h übereinstimmen, so sind wir fertig, denn die Konstante ist eine Funktion mit den oben genannten Anforderungen. Da wir im nichtentarteten Fall sind, können wir mit $A_{i,j} := D^i H(\mathcal{F}|0, -jh) = D^i H(\mathcal{F}|h, (1 - j)h)$, wobei $i \in 0 : m - 2$ und $j \in 0 : m$, wie folgt ansetzen. Wir suchen Koeffizienten $B = [\beta_1, \dots, \beta_m]^T$, sodass

$$A(:, 0 : m - 1)B = A(:, 1 : m)B.$$

Die erste Spalte von A besteht nur aus Nulleinträgen, deshalb ist die letzte Gleichung äquivalent zu

$$A(:, 1 : m)[\beta_1 - \beta_2, \dots, \beta_{m-1} - \beta_m, \beta_m] = [0, \dots, 0]^T.$$

A besitzt wegen der Nicht-Entartung maximalen Rang, es gibt also bis auf Skalierung genau eine nichttriviale Lösung B von $A(:, 1 : m)B = [0, \dots, 0]^T$; also ist f konstant.

□

Nun wollen wir uns an die Subdivision von Chebyshev-B-Splines machen. Wir werden explizite Formeln hierfür angeben, die sich stabil auswerten lassen. Der Weg dorthin führt uns über die folgende Definition 4.53 und Lemma 4.54.

Definition 4.53 Wir definieren mit $\mathcal{F} := \mathcal{F}^W$ die beiden Abbildungen Θ und $\tilde{\Theta}$ von $\Lambda^{\mathcal{F},h}$ in den Raum der formalen Laurentreihen wie folgt:

$$\begin{aligned} \Theta^{\mathcal{F},h} \left(\sum_k b_k^{W,h} \beta_k \right) &:= \sum_k \beta_k x^k, \\ \tilde{\Theta}^{\mathcal{F},h} \left(\sum_k T(\mathcal{F}|x, kh) \tilde{\beta}_k \right) &:= \sum_k \tilde{\beta}_k x^k. \end{aligned}$$

Außerdem setzen wir für ein $n \geq 2$ den Subdivisionsoperator als n -Teilung des Knotenabstands wie folgt:

$$S_n^{\mathcal{F},h} : \Lambda^{\mathcal{F},h} \rightarrow \Lambda^{\mathcal{F},h/n}, s \mapsto s.$$

Durch Θ bzw. durch $\tilde{\Theta}$ ordnen wir einem Spline diejenige Laurentreihe zu, deren Koeffizienten für die Monome x^k für $k \in \mathbb{Z}$ gleich den Koeffizienten für die Kontrollpunkte sind bzw. gleich den Koeffizienten für die Stutzfunktionen sind. Wir tun das deshalb, weil sich später die Subdivision im Raum der Laurentreihen besonders elegant ausdrücken lässt.

Mit Definition 4.53 betrachten wir mit $\Theta^{\mathcal{F},h}$ Splines in der Basis der Chebyshev-B-Splines bzw. mit $\tilde{\Theta}^{\mathcal{F},h}$ in der Basis der Stutzfunktionen. Praktisch relevant ist zwar lediglich die Basis der Chebyshev-B-Splines, zur Herleitung der Subdivisionsgewichte für die Chebyshev-B-Spline-Basis ist der Umweg über die Stutzfunktionen jedoch sehr günstig, schließlich ist die Subdivision in der Stutzfunktion-Basis völlig trivial und der Übergang von einer Basis in die andere ist mit Definition 4.51 bereits vollständig geklärt.

Die Frage ist nun also, wie die Subdivision $S_n^{\mathcal{F},h}$ in der Basis der Chebyshev-B-Splines aussieht, d.h. wir interessieren uns für die Abbildung

$$\Theta^{\mathcal{F},h/n} S_n^{\mathcal{F},h} (\Theta^{\mathcal{F},h})^{-1}.$$

Der Trick ist, dass wir $\tilde{\Theta}^{\mathcal{F},h/n} S_n^{\mathcal{F},h} (\tilde{\Theta}^{\mathcal{F},h})^{-1}$ leicht beschreiben können, ebenso die Abbildungen $\Theta^{\mathcal{F},h/n} (\tilde{\Theta}^{\mathcal{F},h/n})^{-1}$ und $\tilde{\Theta}^{\mathcal{F},h} (\Theta^{\mathcal{F},h})^{-1}$.

Lemma 4.54 Sei $W = (\omega_1, \dots, \omega_m)$ und $\mathcal{F} = \mathcal{F}^W$. Die Subdivision in der Basis der Stutzfunktionen (nach Anwendung der Transformation $\tilde{\Theta}$) ist gegeben durch

$$\left(\tilde{\Theta}^{\mathcal{F},h/n} S_n^{\mathcal{F},h} (\tilde{\Theta}^{\mathcal{F},h})^{-1} \right) (p(x)) = p(x^n).$$

Der Wechsel von einer Basis in die andere ist gegeben durch

$$\left(\Theta^{\mathcal{F},h/n} (\tilde{\Theta}^{\mathcal{F},h/n})^{-1} \right) (p(x)) = \frac{p(x)}{\alpha_{h/n} (x - \exp(-\omega_1 h/n)) \cdots (x - \exp(-\omega_m h/n))},$$

bzw.

$$\tilde{\Theta}^{\mathcal{F},h} (\Theta^{\mathcal{F},h})^{-1} (p(x)) = p(x) \alpha_h (x - \exp(-\omega_1 h)) \cdots (x - \exp(-\omega_m h)).$$

Also ist

$$\left(\Theta^{\mathcal{F},h/n} S_n^{\mathcal{F},h} (\Theta^{\mathcal{F},h})^{-1} \right) (p(x)) = p(x^n) \frac{\alpha_h (x^n - \exp(-\omega_1 h)) \cdots (x^n - \exp(-\omega_m h))}{\alpha_{h/n} (x - \exp(-\omega_1 h/n)) \cdots (x - \exp(-\omega_m h/n))},$$

wobei $\alpha_h, \alpha_{h/n} \in \mathbb{R}$ gemäß Definition 4.51 gewählt seien.

Beweis Die erste Behauptung ist wegen

$$\sum_k T(\mathcal{F}|x, kh) \beta_k = \sum_k T(\mathcal{F}|x, kh/n) \gamma_k$$

mit

$$\gamma_k = \begin{cases} \beta_{k/n} & , \quad k/n \in \mathbb{Z} \\ 0 & , \quad \text{sonst} \end{cases}$$

trivial. Die zweite bzw. dritte Behauptung ist eine unmittelbare Folgerung von Definition 4.51, denn

$$\tilde{\Theta}^{\mathcal{F},h} \left(\alpha_h \prod_{k=1}^m (V_h - \exp(-\omega_k h)) T(\mathcal{F}|x, 0) \right) = \alpha_h \prod_{k=1}^m (x - \exp(-\omega_k h)).$$

□

Nun können wir das Subdivisionspolynom $s_n^{W,h}(x)$ definieren.

Definition 4.55 Für $W = (\omega_1, \dots, \omega_m)$ und $n \geq 2$ setzen wir

$$s_n^{W,h}(x) = r \frac{(x^n - \exp(-\omega_1 h)) \cdots (x^n - \exp(-\omega_m h))}{(x - \exp(-\omega_1 h/n)) \cdots (x - \exp(-\omega_m h/n))},$$

wobei $r \in \mathbb{R}$ so zu wählen ist, dass $s_n^{W,h}(1) = n$.

Dass der Quotient der Polynome in Definition 4.55 selbst wieder ein Polynom ist, zeigt uns der folgende Satz. Die Faktoren von $s_n^{W,h}(x)$ lassen sich schreiben als

$$\frac{x^n - \exp(-\omega_k h)}{x - \exp(-\omega_k h/n)} = \sum_{j=0}^{n-1} x^{n-1-j} \exp(-\omega_k h j/n).$$

Fassen wir unsere Erkenntnisse zusammen, so erhalten wir den folgenden Satz.

Satz 4.56 Sei $W = (\omega_1, \dots, \omega_m)$ und $\mathcal{F} = \mathcal{F}^W$ und $n \geq 2$. Für $f := \sum_k b_k^{W,h} \gamma_k$ gilt $f = \sum_k b_k^{W,h/n} \beta_k$, wobei sich die β_k mit

$$p(x) := \sum_k \gamma_k x^k$$

wie folgt berechnen lassen

$$\sum_k \beta_k x^k := p(x^n) s_n^{W,h}(x).$$

Beweis Zu begründen ist lediglich noch die Normierung $s_n^{W,h}(1) = n$. Sie bedeutet nichts anderes, als dass die Summe aller Koeffizienten von $s_n^{W,h}$ gerade n ergeben muss. Dies folgt aber unmittelbar aus der Tatsache, dass die Konstant-Eins-Funktion wie folgt subdividiert wird: $\sum_k b_k^{W,h} 1 = \sum_k b_k^{W,h/n} 1$. \square

Für $n = 2$ besitzt das Subdivisionspolynom die folgende Gestalt.

Satz 4.57 Es gilt

$$s_2^{W,h}(x) = 2 \frac{(x + \exp(-\omega_1 h/2)) \cdots (x + \exp(-\omega_m h/2))}{(1 + \exp(-\omega_1 h/2)) \cdots (1 + \exp(-\omega_m h/2))}.$$

Offenbar konvergiert für $h \rightarrow 0$ das Subdivisionspolynom $s_2^{W,h}$ linear gegen das polynomiale Subdivisionspolynom $2^{-n} \sum_{k=0}^n \binom{n}{k} x^k$.

Satz 4.58 Für $W = (\omega_1, \dots, \omega_m)$ gilt mit

$$h < \frac{\pi}{\max_k |\operatorname{Imag}(\omega_k)|}$$

stets

$$b_k^{W,h} \geq 0.$$

Falls alle ω_k reell sind, so ist $b_k^{W,h} \geq 0$ für beliebige $h > 0$. Insgesamt sind also für Knotenabstände unterhalb des kleinsten Entartungsfalls die Chebyshev-B-Splines $b_k^{W,h}$ nicht negativ.

Beweis Man kann einem Chebyshev-B-Spline sein Kontrollpolygon zuordnen. Nach [DL92] konvergiert dieses gleichmäßig gegen den dargestellten Spline. Gehen wir nun sukzessive von der Knotenfolge $h\mathbb{Z}$ zu der doppelt so feinen $(h/2)\mathbb{Z}$ über, so treten nur positive Subdivisionsgewichte auf, der Spline selbst kann also nicht negativ sein. Warum sind für obige h alle Subdivisionsgewichte positiv? Für reelle ω_k sind stets alle Gewichte der Faktoren $(x + \exp(-\omega_k h/2))$ positiv, für komplexe $\omega_k = (\alpha + \beta I)$ existiert auch ein komplex konjugiertes $\omega_{k+1} = (\alpha - \beta I)$. Fassen wir die entsprechenden Faktoren zusammen, dann ergibt sich

$$\begin{aligned} & (x + \exp(-(\alpha_k + \beta_k I)h/2))(x + \exp(-(\alpha_k - \beta_k I)h/2)) \\ &= x^2 + x(2 \exp(-\alpha_k h/2) \underbrace{\cos(h\beta_k/2)}_{>0}) + \exp(\alpha_k h). \end{aligned}$$

□

Nun wollen wir noch zwei interessante Aussagen formulieren und beweisen, bevor wir uns den Kontrollpunkten für die Identität zuwenden. Die erste macht eine Aussage, wie die Koeffizienten für die Splines gewählt werden müssen, um die Basisfunktionen des EC-Systems \mathcal{F}^W zu erhalten. Die zweite gibt eine einfache Vorschrift für die Berechnung der Kontrollpunkte der Ableitung aus den Kontrollpunkten des Splines an. Zunächst benötigen wir zwei Lemmata.

Lemma 4.59 Zu einer vorgegebenen Funktion p auf \mathbb{R} existiere ein $q \in \mathbb{P}_{d-1}$ sodass:

$$p(kh) - p((k-1)h) = q(kh), \quad \forall k \in \mathbb{Z}.$$

Dann existiert auch ein $\hat{q} \in \mathbb{P}_d$ sodass für alle $k \in \mathbb{Z}$ gilt:

$$\hat{q}(kh) = p(kh).$$

Beweis Für $k > l$ gilt

$$p(kh) = p(lh) + \sum_{j=l+1}^k (p(jh) - p((j-1)h)) = p(lh) + \sum_{j=l+1}^k q(jh) =: \hat{q}(kh).$$

Da $\sum_{j=l+1}^k j^{d-1}$ ein Polynom der Ordnung d in der Variablen k ist, ist alles gezeigt. \square

Nun formulieren wir gewissermaßen die Umkehrung des letzten Lemmas.

Lemma 4.60 Falls $p(x) \in \mathbb{P}_d$ ist, so gilt:

$$p(x) - p(x-h) \in \mathbb{P}_{d-1}$$

Der Beweis ist durch Betrachtung des Polynoms in Monomdarstellung von p trivial, weil sich dann bei der Subtraktion von $p(x)$ und $p(x-h)$ die führenden Monomkoeffizienten auslöschen.

Nun können wir das folgende Lemma formulieren, in dem wir die Darstellung der Basisfunktionen von \mathcal{F}^W in der Basis der Chebyshev-B-Splines klären.

Lemma 4.61 Sei $\mathcal{F} = \{f : (D - \omega_1)^{m_1} \cdots (D - \omega_r)^{m_r} f = 0\}$ und $h > 0$ so, dass $\Lambda^{\mathcal{F},h}$ nicht entartet ist. Sei nun $j \in 1 : r$ und $d \leq m_j$. Dann existiert zu einem vorgegebenen $p \in \mathbb{P}_d$ ein $q \in \mathbb{P}_d$, sodass

$$\sum_{k \in \mathbb{Z}} b_k^{W,h}(x) \exp(\omega_j kh) p(kh) = q(x) \exp(\omega_j x). \quad (4.17)$$

Umgekehrt existiert auch zu einem vorgegebenen $q \in \mathbb{P}_d$ ein $p \in \mathbb{P}_d$, sodass (4.17) gilt. Für $\omega_j = 0$ stimmen die führenden Koeffizienten von p und q überein.

Beweis Wir führen Induktion über d . Für $d = 0$ ist die Behauptung trivial. Sei $q \in \mathbb{P}_d$ vorgegeben. Wir wählen nun $p(k)$ so, dass $\sum_{k \in \mathbb{Z}} b_k^{W,h}(x) \exp(\omega_j kh) p(k) = q(x) \exp(\omega_j x)$, wobei wir nicht voraussetzen, dass p ein Polynom ist, dies müssen wir zeigen. Nun gilt

$$\begin{aligned} \exp(\omega_j x) \mathbb{P}_d &\ni q(x) \exp(\omega_j x) - \exp(\omega_j h) q(x-h) \exp(\omega_j (x-h)) \\ &= \sum_{k \in \mathbb{Z}} b_k^{W,h}(x) \exp(\omega_j kh) (p(k) - p(k-1)). \end{aligned}$$

Nach der Induktionsvoraussetzung ist aber $p(k) - p(k-1)$ ein Polynom in k der Ordnung $d-1$, also ist nach Lemma 4.59 p ein Polynom der Ordnung d . Damit ist dieser Beweisteil abgeschlossen.

Nun kann man eine Abbildung von \mathbb{P}_d in den Raum $\exp(\omega_j x)\mathbb{P}_d$ wie folgt definieren:

$$p \mapsto \sum_{k \in \mathbb{Z}} b_k^{W,h}(x) \exp(\omega_j kh) p(kh).$$

Diese ist linear, außerdem haben wir oben gezeigt, dass sie surjektiv ist, also ist sie auch bijektiv, was aber die Existenz eines $q \in \mathbb{P}_d$ zu einem vorgegebenen $p \in \mathbb{P}_d$ gemäß (4.17) beweist. Dass im Fall $\omega_j = 0$ die führenden Koeffizienten übereinstimmen, sieht man induktiv durch die Betrachtung von $q(x) - q(x-h)$ als Differenz von Splines wie weiter oben in diesem Beweis. \square

Sei $W = [0, 0, \omega_3, \dots, \omega_m]$ und $\mathcal{F} = \mathcal{F}^W$. Dann sind $1, x \in \mathcal{F}$. Wie sind die Kontrollpunkte zu wählen, sodass $\sum_k b_k^{W,h} p_k = 1$? Nach dem letzten Lemma müssen sie konstant Eins gewählt werden. Das ist nicht neu, wir wissen dies bereits aus Satz 4.52. Stellen wir uns nun die Frage, wie die Identität reproduziert wird, wie also die Kontrollpunkte gewählt werden müssen, sodass $\sum_k b_k^{W,h} p_k = x$. Diese Frage müssen wir in der Praxis bei der Nullstellensuche tatsächlich beantworten, schließlich wird eine Funktion $f(x)$ mit ihrem Graphen $(x, f(x))$ identifiziert und dessen Kontrollpunkte betrachtet. Zunächst sagt uns das letzte Lemma, dass $p_k = hk + c$. Wie ist nun aber c zu wählen? Dieser Frage werden wir im Folgenden nachgehen. Dabei werden wir zunächst nicht c selbst berechnen, sondern $\sum_k (kh) b_k^{W,h} - \sum_k (kh) b_k^h$, wobei b_k^h der rein polynomiale Chebyshev-B-Spline gleicher Ordnung sei, also die Verschiebung gegenüber dem rein polynomialen Fall.

Dazu stellen wir zunächst einige Vorüberlegungen an. Betrachten wir nochmals die Subdivision von h nach $h/2$, wobei $W = [0, 0, \omega_1, \dots, \omega_{m-2}]$ und $\mathcal{F} = \mathcal{F}^W$. Sei $f = \sum_k b_k^{W,h} p_k$ und $p(x) = \sum_k p_k x^k$. Dann gibt es q_k , sodass $f = \sum_k b_k^{W,h/2} q_k$ und mit $q(x) = \sum_k q_k x^k$ gilt nach Satz 4.57

$$q(x) = p(x^2)(x+1) \frac{x+1}{2} \frac{x + \exp(-\omega_1 h/2)}{1 + \exp(-\omega_1 h/2)} \dots \frac{x + \exp(-\omega_{m-2} h/2)}{1 + \exp(-\omega_{m-2} h/2)}.$$

Die Faktorisierung der rechten Seite liefert uns eine Zerlegung der an und für sich sehr technischen Subdivision in mehrere einfach zu überschauende Einzelschritte. Diese Beobachtung geht auf Lane-Riesenfeld in [LR81] zurück. Zunächst betrachten wir das Kontrollpolygon zu den $p_k \in \mathbb{R}^d$ als Punkte im \mathbb{R}^d , wobei aufeinanderfolgende Punkte verbunden werden (siehe Abbildung 24).

Nun ist $p(x^2)(x+1) = \dots + p_{-1}x^{-2} + p_{-1}x^{-1} + p_0x^0 + p_0x^1 + p_1x^2 + p_1x^3 + \dots$. Wir betrachten zu diesem Polynom das entsprechende Kontrollpolygon. Wir erhalten graphisch das alte, nur dass jeder Kontrollpunkt doppelt ist. Betrachten wir nun $p(x^2)(x+1) \frac{x+1}{2}$. Der Faktor $\frac{x+1}{2}$ lässt aus dem Kontrollpolygon mit den doppelten

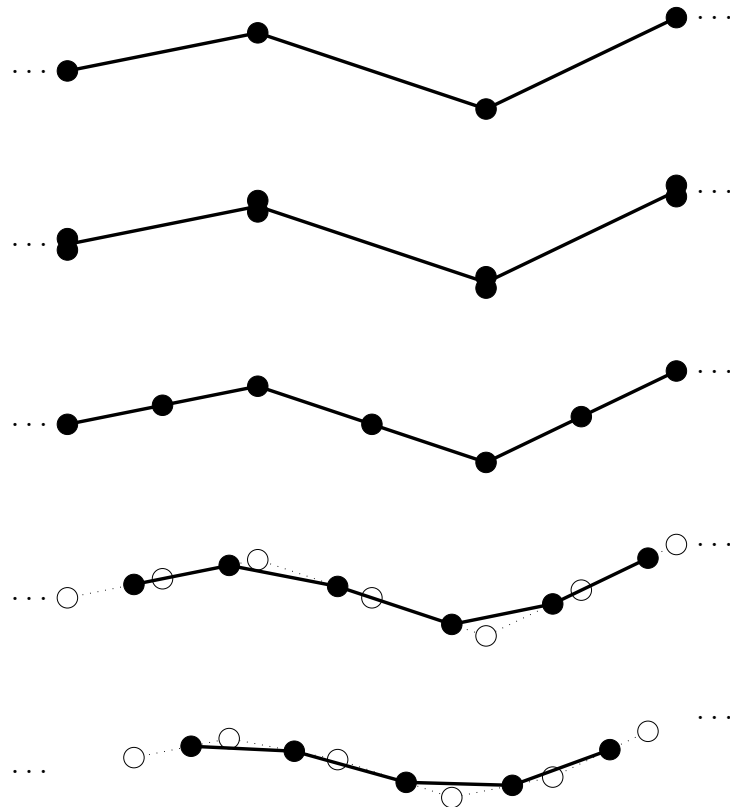


Abbildung 24: Subdivision in $\langle 1, x, \exp(\omega_1 x), \dots, \exp(\omega_{m-2} x) \rangle$ aufgefasst als Knotenverdoppelung gefolgt von Mittelungen benachbarter Kontrollpunkte

Punkten durch Mittelung (im Verhältnis 1 : 1) von „aufeinanderfolgenden“ Kontrollpunkten ein neues entstehen. Die vorher doppelten Kontrollpunkte bleiben als einfache erhalten und zwischen zwei Kontrollpunkten wird genau in die Mitte ein neuer eingefügt (siehe Abbildung 24).

Betrachten wir nun $p(x^2)(x+1)^{\frac{x+1}{2}}(\alpha x + \beta)$, wobei $\alpha = 1/(1 + \exp(-\omega_1 h/2))$ und $\beta = \exp(-\omega_1 h/2)/(1 + \exp(-\omega_1 h/2))$. Der Faktor $(\alpha x + \beta)$ bewirkt nun die Mittelung im Verhältnis $\alpha : \beta$. Der „Vorgänger“-Kontrollpunkt erhält dabei das zu x gehörende Gewicht α , sein „rechter Nachbar“ das Gewicht β usw. (siehe Abbildung 24).

Im rein polynomialen Fall ergibt diese Betrachtung den Algorithmus von Lane-Riesenfeld, nämlich eine fortgesetzte Mittelung jeweils im Verhältnis 1 : 1 (siehe [LR81]).

Im Fall von komplexen ω_k kommen komplexe Gewichte bei der Mittelung ins Spiel. In diesem Fall ist es für das Verständnis praktisch, die entsprechenden beiden Faktoren zu

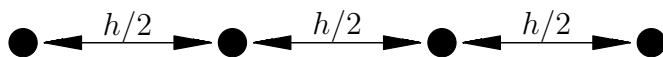
dem komplex konjugierten Paar zusammenzufassen. Es ergibt sich dann für $\omega_k = a + ib$

$$\frac{x + \exp(-\omega_k h)}{1 + \exp(-\omega_k h)} \frac{x + \exp(-\bar{\omega}_k h)}{1 + \exp(-\bar{\omega}_k h)} = \frac{x^2 + x(2 \exp(-ah/2) \cos(bh/2)) + \exp(-ah)}{1 + 2 \exp(-ah/2) \cos(bh/2) + \exp(-ah)}.$$

Es werden bei Multiplikation mit dem zusammengefassten Faktor nicht mehr zwei aufeinanderfolgende Kontrollpunkte gemittelt, sondern drei.

Anschaulich gar nicht klar, aber durch den Polynom-Ansatz trivial ergibt sich, dass die Reihenfolge der Faktoren keine Rolle spielt.

Machen wir uns nun daran, mit den Kontrollpunkten $p_k = kh$ die Verschiebung $\sum_k (kh) b_k^{W,h} - \sum_k (kh) b_k^h$ gegenüber dem rein polynomialen Fall zu berechnen. Dabei sei, wie gesagt b_k^h der polynomiale Chebyshev-B-Spline gleicher Ordnung. Sei zunächst $W = [0, 0, -\omega]$. Subdividieren wir also die Kontrollpunkte $(kh)_k$. Es ergibt sich als Zwischenschritt $p(x^2)(x+1)^{\frac{x+1}{2}}$. Die „Kontrollpunkte“ des Zwischenschritts stimmen noch mit dem polynomialen Fall überein. Jeweils zwei aufeinanderfolgende „Kontrollpunkte“ unterscheiden sich außerdem jeweils um $h/2$.



Nun werden aufeinanderfolgende Kontrollpunkte gemittelt mit den beiden Gewichten $\frac{1}{1+\exp(\omega h/2)}$ und $\frac{\exp(\omega h/2)}{1+\exp(\omega h/2)}$. Im polynomialen Fall $\omega = 0$ liegen die neuen Kontrollpunkte genau in der Mitte zwischen den alten, im nichtpolynomialen Fall $\omega \neq 0$ um $\frac{h}{2} \left(\frac{\exp(\omega h/2)}{1+\exp(\omega h/2)} - \frac{1}{2} \right)$ gegenüber dem polynomialen Fall nach „rechts“ verschoben. Die Kontrollpunkte behalten zu ihren Nachbarn den Abstand $h/2$. Also ergibt sich bei der Auswertung im Grenzübergang die Verschiebung gegenüber dem polynomialen Fall als

$$\sum_{k=1}^{\infty} \frac{h}{2^k} \left(\frac{\exp(\omega h/2^k)}{1 + \exp(\omega h/2^k)} - \frac{1}{2} \right) = \sum_{k=1}^{\infty} \frac{h}{2^k} \frac{\exp(\omega h/2^k)}{1 + \exp(\omega h/2^k)} - \frac{h}{2}.$$

Die fragliche Reihe ist für beliebiges ω durch 0 und h beschränkt und damit wegen der Positivität der Summanden konvergent (sogar gleichmäßige Konvergenz in ω). Fragt sich nur, was der Grenzwert ist. Betrachten wir die Verschiebung als Funktion in ω , also

$$f(\omega) := \sum_{k=1}^{\infty} \frac{h}{2^k} \frac{\exp(\omega h/2^k)}{1 + \exp(\omega h/2^k)}$$

bzw.

$$f_m(\omega) := \sum_{k=1}^m \frac{h}{2^k} \frac{\exp(\omega h/2^k)}{1 + \exp(\omega h/2^k)}.$$

Eine Stammfunktion von f_m ist

$$\begin{aligned} F_m(\omega) &= \ln\left(\frac{1}{2^m}\right) + \sum_{k=1}^m \ln(1 + \exp(\omega h/2^k)) \\ &= \ln\left(\frac{1}{2^m} \prod_{k=1}^m (1 + \exp(\omega h/2^k))\right) \\ &= \ln\left(\frac{1}{2^m} \sum_{l=0}^{2^m-1} \exp\left(\omega h \frac{l}{2^m}\right)\right) \end{aligned}$$

Der letzte Ausdruck konvergiert für wachsendes m wegen der Stetigkeit des \ln und der Definition des Riemann-Integrals gegen

$$\begin{aligned} F(\omega) &:= \lim_{m \rightarrow \infty} F_m(\omega) = \ln\left(\int_{x=0}^{x=1} \exp(\omega h x) dx\right) \\ &= \ln\left(\frac{\exp(\omega h) - 1}{\omega h}\right). \end{aligned}$$

Nun gilt

$$F'(\omega) = \frac{(\omega h - 1) \exp(\omega h) + 1}{\omega(\exp(\omega h) - 1)}.$$

Nun stellt sich die Frage, ob $F'(\omega) = f(\omega)$ gilt. Da aber f_m gegen f gleichmäßig konvergiert, ist dies garantiert.

In Übereinstimmung mit der Intuition gilt

$$\lim_{\omega \rightarrow 0} \frac{(\omega h - 1) \exp(\omega h) + 1}{\omega(\exp(\omega h) - 1)} = \frac{h}{2}$$

die Verschiebung verschwindet also für $\omega \rightarrow 0$. Unmittelbar einsichtig ist, dass sich für allgemeines W die Verschiebungen einfach addieren. Dies ergibt sich aus der Tatsache, dass Doppelsummen mit positiven Summanden in beliebiger Reihenfolge aufaddiert werden können. Ersetzt man noch $-\omega$ durch ω , so erhält man mit

$$v_h(\omega) := \frac{(\omega h + 1) \exp(-\omega h) - 1}{\omega(\exp(-\omega h) - 1)} - \frac{h}{2} = \frac{1}{\omega} - h \frac{\exp(-\omega h/2)}{2 \sinh(\omega h/2)} - \frac{h}{2}$$

schließlich für $W = [0, 0, \omega_3, \dots, \omega_m]$ als Gesamtverschiebung gegenüber dem rein polynomialen Fall

$$V_h(W) := \sum_{k=3}^m v_h(\omega_k).$$

Unsere Überlegungen haben wir bisher strenggenommen nur für reelle ω_k gemacht, doch bei komplexen ω_k (treten stets komplex konjugiert auf) sieht man durch Zusammenfassung der beiden „komplex konjugierten Faktoren“ bei der Subdivision, dass sich alle Formeln übertragen.

Betrachten wir nun $v_h(\omega)$ etwas genauer. Es gilt $v_h(\omega) + v_h(-\omega) = 0$. Das bedeutet, dass Paare von additiv inversen ω_k keinen Beitrag zur Verschiebung gegenüber dem rein polynomialen Fall leisten, es gilt also z.B. mit $W = [0, 0, 1, -1, 2, 1 + I, 1 - i, 2I, -2I]$: $V_h(W) = v_h(2) + v_h(1 + I) + v_h(1 - I)$.

Die Kontrollpunkte für die Identität im rein polynomialen Fall sind die Greville-Abszissen, also im uniformen Fall $p_k = h(k + (k + m))/2 = h(k + m/2)$.

Es ergibt sich also der folgende Satz

Satz 4.62 Sei $W = [0, 0, \omega_3, \dots, \omega_m]$, $\mathcal{F} := \mathcal{F}^W$ und

$$v_h(\omega) := \begin{cases} \frac{(\omega h + 1) \exp(-\omega h) - 1}{\omega (\exp(-\omega h) - 1)} - \frac{h}{2} = \frac{1}{\omega} - h \frac{\exp(-\omega h/2)}{2 \sinh(\omega h/2)} - \frac{h}{2} & , \quad \omega \neq 0 \\ 0 & , \quad \omega = 0 \end{cases} .$$

Dann gilt mit $p_k := h(k + m/2) - \sum_{k=3}^m v_h(\omega_k)$

$$\sum_k b_k^{W,h}(x) p_k = x .$$

Es gilt $\lim_{\omega \rightarrow 0} v_h(\omega) = 0$, $v_h(\omega) + v_h(-\omega) = 0$, $\lim_{\omega \rightarrow \infty} v_h(\omega) = -h/2$ und $\lim_{\omega \rightarrow -\infty} v_h(\omega) = h/2$ und mit $\omega = \mu + I\nu$:

$$v_h(\omega) + v_h(\bar{\omega}) = \frac{2\mu}{\mu^2 + \nu^2} - 2h \frac{\cos(h\nu) - \exp(-h\mu)}{\exp(h\mu) - 2\cos(h\nu) + \exp(-h\mu)} - h$$

(siehe Abbildung 25). Bei der numerischen Berechnung von v_h ist zu beachten, dass

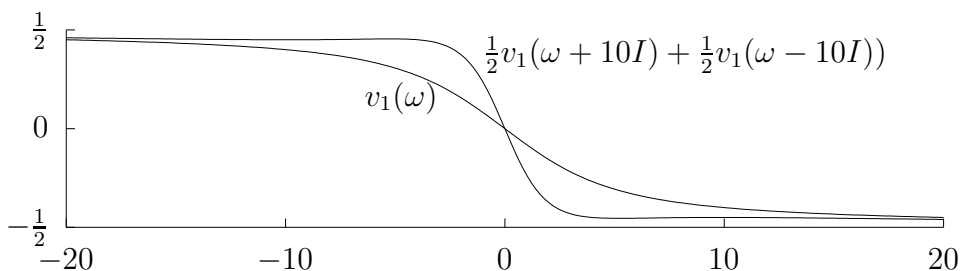


Abbildung 25: v_h für reelle und imaginäre ω und $h = 1$

sich für ω mit $|\omega| \ll 1$ die Auswertung der Taylorentwicklung anbietet, andernfalls ist die direkte Auswertung des obigen Ausdrucks vorzuziehen. Die Taylorentwicklung lautet:

$$v_h(\omega) = \frac{h^2}{12}\omega + \frac{h^4}{720}\omega^3 - \frac{h^6}{30240}\omega^5 + \frac{h^8}{1209600}\omega^7 + \dots$$

Untersuchen wir nun noch ganz am Ende dieses Abschnitts, wie uniforme Splines abgeleitet werden können. Erstaunlicherweise ergibt sich völlige Analogie zum polynomialen Fall. Dies zeigt das folgende Lemma.

Lemma 4.63 Sei $\mathcal{F} = \{f : (D - \omega_1) \cdots (D - \omega_{m-2}) D^2 f = 0\}$ und $\mathcal{G} = \{f : (D - \omega_1) \cdots (D - \omega_{m-2}) D f = 0\}$ und $h > 0$ so, dass $\Lambda^{\mathcal{F},h}$ nicht entartet ist. Dann gilt

$$D \sum_k b_k^{W,h} p_k = \sum_k b_k^{\mathcal{G},h} \left(\frac{p_k - p_{k-1}}{h} \right).$$

Beweis Zunächst folgt mittels Satz 4.48 aus der Nicht-Entartung von $\Lambda^{\mathcal{F},h}$ auch die von $\Lambda^{\mathcal{G},h}$. Die Ableitung eines Splines aus $\Lambda^{\mathcal{F},h}$ ist offensichtlich ein Spline in $\Lambda^{\mathcal{G},h}$. Wegen der lokalen-Basis-Eigenschaft muss nun $D b_k^{W,h} = \alpha b_k^{\mathcal{G},h} + \beta b_{k+1}^{\mathcal{G},h}$ sein, wobei α und β wegen der Translationsinvarianz unabhängig von k sind. Wegen $D \sum_k b_k^{W,h} = 0$ folgt $\alpha = -\beta$. Nach dem letzten Satz gilt $\sum_k b_k^{W,h}(x) k h = x + c$. Somit ist $1 = D \sum_k b_k^{W,h}(x) k h = \sum_k \alpha (k h - (k-1) h) b_k^{\mathcal{G},h} = \alpha h$ und deshalb $\alpha = 1/h$. \square

Nun haben wir die notwendige Theorie für uniforme Exponentialsplines zusammengestellt. Bei der Lösung exponentieller Gleichungssysteme sind nun die linearen Kontrollpunkte für die Identifikation der einzelnen Gleichungen mit ihren Graphen durch Satz 4.62 gegeben. Die Subdivision erfolgt gemäß Satz 4.56. Erklären wir unser Vorgehen für den univariaten Fall, die Verallgemeinerung ins Multivariate stellt dann kein Problem dar.

Sei $f : [a, b] \rightarrow \mathbb{R}$ gegeben und für $W = [0, 0, \omega_1, \dots, \omega_{m-2}]$ gelte $f \in \mathcal{F} := \mathcal{F}^W$. Wir setzen $h = b - a$. Nun könnten wir m Koeffizienten α_k bestimmen, sodass

$$\sum_{k=-m+1}^0 \alpha_k b_k^{W,h}(x) = f(a+x). \quad (4.18)$$

Diese Idee ist zwar sehr natürlich, führt uns aber, wie wir sehen werden, zu Problemen bei der Subdivision. Stattdessen bestimmen wir $(m+1)$ Kontrollpunkte α_k , sodass

$$\sum_{k=-m+1}^1 \alpha_k b_k^{W,h/2}(x) = f(a+x). \quad (4.19)$$

Was nun folgt, entspricht bis zur Subdivision exakt unserem Standardvorgehen. Wir identifizieren die Funktion f mit ihrem Graphen, was bei den Kontrollpunkten mit den p_k aus Satz 4.62 die Kontrollpunkte $q_k = [p_k, \alpha_k]$ entspricht. Zu diesen q_k bestimmen wir zwei einschließende Geraden mit Steigung $(\alpha_{-m+1} - \alpha_1)/(p_{-m+1} - p_1)$.

Nun schneiden wir diese beiden Geraden mit der Nulllinie $[x, 0]$, was die die beiden Schnittpunkte $(c, 0)$ und $(d, 0)$ ergibt. Nullstellen sind also nur für $x \in [c, d]$ möglich. Unser Wunsch ist nun, auf dieses Intervall $[c, d]$ zu subdividieren. Diese Subdivision können wir im Allgemeinen nicht mit unserer Theorie realisieren. Wir können jedoch auf Intervalle $[a + kh/(2n), a + jh/(2n)]$ für beliebige n subdividieren. Wenn wir nun $n := \lfloor h/(2(d - c)) \rfloor$ setzen (wobei $\lfloor \cdot \rfloor$ die nächst kleinere ganze Zahl sei), so ist nicht gewährleistet, dass mit $k = \lfloor (c - a)2n \rfloor$ nun $[c, d] \subset [a + kh/(2n), (a + (k + 1)h)/(2n)]$ ist. Ganz sicher aber ist $[c, d] \subset [a + kh/(2n), (a + (k + 2)h)/(2n)]$, was schließlich der Grund ist, warum wir die Darstellung (4.18) der von (4.19) vorziehen. Wir subdividieren also auf das Intervall $[a + kh/(2n), (a + (k + 2)h)/(2n)]$ und fahren so sukzessive fort. Sinnvollerweise sollte n nach oben z.B. durch 500 beschränkt werden.

Nun formulieren wir noch den Satz über die Konvergenzrate unseres Algorithmus. Sie ist theoretisch quadratisch, da das das Kontrollpolygon quadratisch gegen die dargestellte Funktion konvergiert. Dies kann man zeigen, wenn man analog zum polynomialen Bernsteinfall den Δ^2 -Operator definiert und bewiest, dass $|\Delta^2|_\infty$ quadratisch gegen Null geht.

Satz 4.64 *Der Algorithmus zur Einschließung von Nullstellen konvergiert theoretisch quadratisch sofern die Jacobi-Matrix in einer Lösung invertierbar ist. Durch obige Beschränkung von n durch z.B. 500 geht die quadratische Konvergenz in eine lineare mit Faktor $1/500$ über.*

Und erneut bemerken wir, dass dieselbe Reduktion des Aufwands erzielt werden kann, sofern die Voraussetzungen von Satz 3.11 erfüllt sind. Das gesamte Vorgehen lässt sich dann unmittelbar übertragen.

Betrachten wir noch ein Beispiel zur Illustration.

Beispiel 4.65 *Seien die Nullstellen von $f : [0, 1] \rightarrow \mathbb{R}, x \mapsto \exp(x) - 4 \exp(-x/2) + 5/4$ gesucht (siehe auch Abbildung 26). f liegt für $W = [0, 0, 1, 1/2]$ im Funktionenraum $\mathcal{F} = \mathcal{F}^W$ und es gilt für $x \in [0, 1]$:*

$$f(x) \approx -3.2190b_{-3}^{\mathcal{F},1/2} - 1.7190b_{-2}^{\mathcal{F},1/2} - 0.2190b_{-1}^{\mathcal{F},1/2} + 1.4963b_0^{\mathcal{F},1/2} + 3.7340b_1^{\mathcal{F},1/2}$$

Die Kontrollpunkte der Identität sind gerundet $[-0.4896, 0.0103, 0.5103, 1.0104, 1.5104]$. Als unterer Einschluss ergibt sich in Hesse-Normalform $\{x \in \mathbb{R}^2 : \langle x, (-3.4765; 1) \rangle = -1.9932\}$, bzw. als oberer Einschluss erhalten wir $\{x \in \mathbb{R}^2 : \langle x, (-3.4765; 1) \rangle = -1.5167\}$. Schnittpunkte dieser beiden Geraden mit der Geraden $(x, 0)$ sind: $x = 0.5733$ und $x = 0.4363$. Nullstellen sind also nur im Intervall $[c, d] = [0.4363, 0.5733]$ möglich (siehe Abbildung 26).

Nun ergibt sich $n = 3$ und $k = 2$. Wir subdividieren nun auf das Intervall $[2, 4]/6$. Es ergeben sich $[-1.2465, -0.7393, -0.2168, 0.3298, 0.9106]$ als neue Kontrollpunkte. Bereits nach insgesamt 13 Schritten ist die Nullstelle $x = 0.5669$ mit Genauigkeit $5 \cdot 10^{-13}$ eingegrenzt.

4.4 Vergleich der Ansätze mit uniformen Splines bzw. mit der Bézierdarstellung zur Lösung von

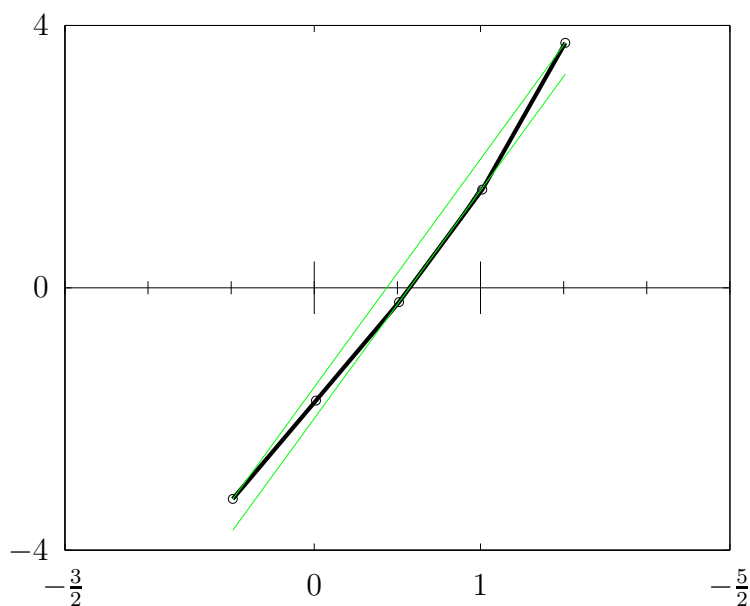


Abbildung 26: Nullstellensuche für $f(x) = \exp(x) - 4 \exp(-x/2) + 5/4$ mit uniformen Splines

4.4 Vergleich der Ansätze mit uniformen Splines bzw. mit der Bézierdarstellung zur Lösung von exponentiellen Gleichungssystemen

Eine Bilanz für die beiden Ansätze zu ziehen fällt nicht schwer. Ist für einen Funktionenraum $\mathcal{F} = \{f : p(D)f \equiv 0\}$ nicht entscheidbar, ob es sich auf einem gegebenen Intervall um ein EC-System handelt, so ist der Ansatz mit uniformen Splines der richtige. Weiß man hingegen, dass es sich bei \mathcal{F} um ein EC-System handelt, so ist der Ansatz mit der Bézierdarstellung vorzuziehen. Dies ist deshalb eindeutig klar, weil bei der Identifikation der Funktion mit ihrem Graphen die Kontrollpunkte weit außerhalb der betrachteten Box liegen und so die konvexe Hülle, eingeschränkt auf die betrachtete Box, wesentlich unschärfer wird. Ein typisches Beispiel ist das folgende. Betrachten wir in Abbildung 27 die Funktion $F(x) = -(x-1)^2 + 9/16$ für $x \in [0, 1]$, welche im Funktionenraum $\langle 1, x, \exp(x), \exp(-2x), \exp(3x) \rangle$ liegen.

Als möglicher Nullstellenbereich ergibt sich für uniforme Splines das Intervall $[0.121, 1]$ und im Bézierfall das Intervall $[0.0925, 0.4375]$.

Als weiteres Beispiel wollen wir die Unterteilungsstruktur für die beiden Ansätze in Abbildung 28 plotten, wobei das Gleichungssystem $F(x, y) = 0$ gegeben sei durch

$$F(x, y) = \begin{bmatrix} -3 \exp(y) - 4 \exp(3x) - 1 + 2 \exp(y + 3x) + 2 \exp(-4x) + 9 \\ -\exp(x) + 2 \exp(3y) - 1 + \exp(x + 3y) - 23 \end{bmatrix}.$$

Die beiden Algorithmen laufen so lange, bis sie die Nullstellen in $[0, 1]^2$ mit einer To-

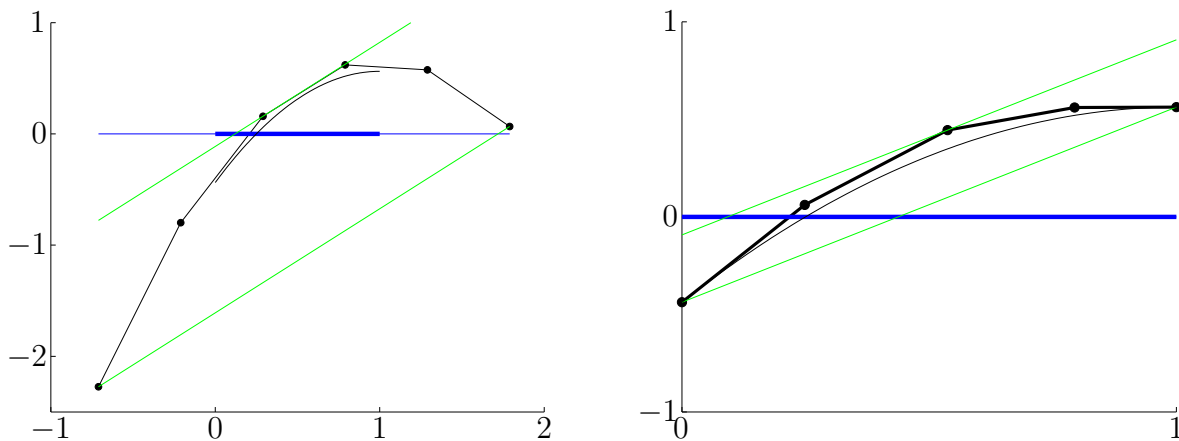


Abbildung 27: Eingrenzung der Nullstelle im Vergleich zwischen den Ansätzen mit uniformen Splines (links) und der Bézierdarstellung (rechts).

leranz von 10^{-12} eingegrenzt sind. Bei der Arbeit mit uniformen Splines ist hierfür die Untersuchung von insgesamt 81 Boxen notwendig, bei der Arbeit in Bézierdarstellung hingegen nur 18.

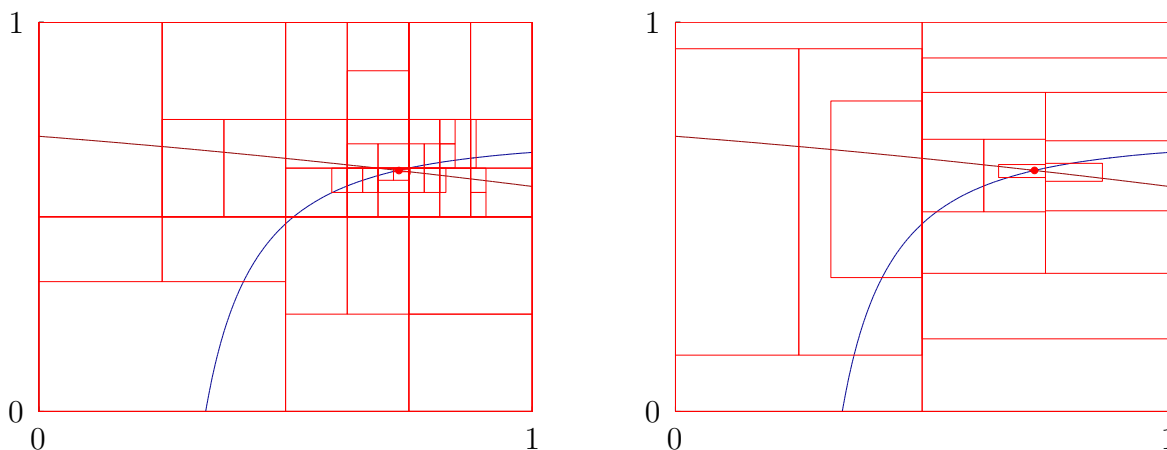


Abbildung 28: Eingrenzung der Nullstelle im Vergleich zwischen den Ansätzen mit uniformen Splines (links) und der Bézierdarstellung (rechts).

4.5 Transformation des Gleichungssystems

Durch Transformation des Gleichungssystems lässt sich unser Algorithmus auf eine weit größere Klasse von Problemen anwenden. So lassen sich Quotienten, Umkehrfunktionen und Hintereinanderausführungen auflösen. Dabei ist zu beachten, dass den neu eingeführten Variablen entsprechende Definitionsbereiche zuzuordnen sind.

- Die Gleichung

$$0 = f(g(x_1)) + h$$

ist äquivalent zum Gleichungssystem

$$\begin{aligned} 0 &= f(x_2) + h \\ 0 &= g(x_1) - x_2. \end{aligned}$$

- Die Gleichung

$$0 = f(x_1)/g(x_2) + h \quad , g(x_2) \neq 0$$

ist äquivalent zum Gleichungssystem

$$\begin{aligned} 0 &= x_3 + h \\ 0 &= f(x_1) - x_3 g(x_2). \end{aligned}$$

- Die Gleichung

$$0 = f^{-1}(x_1) + h$$

ist äquivalent zum Gleichungssystem

$$\begin{aligned} 0 &= x_2 + h \\ 0 &= f(x_2) - x_1. \end{aligned}$$

Somit können wir auch Quotienten, Umkehrfunktionen und Hintereinanderausführungen von Funktionen in EC-Systemen behandeln.

Beispiel 4.66 *Die Gleichung*

$$0 = \frac{t_1}{\sin(t_1)} + \arccos(t_1) + \exp(2 + \sin(t_1)) - 16$$

mit $t_1 \in [1/2, 1]$ ist äquivalent zum Gleichungssystem

$$\begin{aligned} 0 &= t_2 + t_3 + \exp(t_4) - 16 \\ 0 &= t_2 \sin(t_1) - t_1 \\ 0 &= \cos(t_3) - t_1 \\ 0 &= t_4 - 2 - \sin(t_1) \end{aligned}$$

mit $t_1 \in [1/2, 1]$, $t_2 \in [1, 2]$, $t_3 \in [0, \pi/2]$ und $t_4 \in [1, 3]$. Dieses zweite Gleichungssystem besitzt die einzige Lösung $t_1 = 0.7047$, welche bereits nach 5 Schritten mit Genauigkeit 10^{-3} eingegrenzt werden kann.

Im Folgenden wollen wir eine zweite Erweiterungsmöglichkeit besprechen. Tauchen in einem Gleichungssystem vereinzelt Ausdrücke auf, die sich nicht ins Schema der EC-Systeme einordnen lassen, so können diese auch substituiert werden und mit alternativen Methoden abgeschätzt werden. Betrachten wir hierzu als Modellproblem das Gleichungssystem

$$\begin{aligned}x_1^2 - x_2^3 &= 0 \\(1 - x_1)^\pi - 3x_2^2 &= 0\end{aligned}$$

für $x_1, x_2 \in [0, 1]$. Wir eliminieren den Ausdruck $(1 - x_1)^\pi$ und stellen das folgende äquivalente Gleichungssystem auf:

$$\begin{aligned}x_1^2 - x_2^3 &= 0 \\x_3 - 3x_2^2 &= 0 \\(1 - x_1)^\pi - x_3 &= 0,\end{aligned}$$

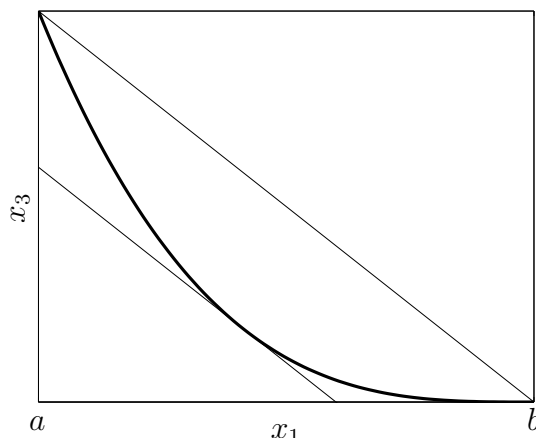
wobei $x_1, x_2, x_3 \in [0, 1]$.

Die beiden ersten Gleichungen des transformierten Systems sind Standard, wir können völlig problemlos den Einschluss der linken Seiten durch parallele Hyperebenen berechnen. Die dritte Gleichung muss nun gesondert betrachtet werden, bereitet aber keine Probleme. Wir bringen sie zunächst auf die Form: $x_3 = (1 - x_1)^\pi$. Die Funktion $f(x_1) = (1 - x_1)^\pi$ ist offenbar auf $[0, 1]$ monoton fallend und konvex. Betrachten wir nun den Graphen von f . f verläuft wegen der Konvexität sicherlich unterhalb der Verbindungsgerade von $(0, f(0))$ und $(1, f(1))$. Verschieben wir nun diese Gerade soweit nach unten bis f oberhalb dieser verläuft, also bis die verschobene Gerade Tangente an f ist. Mit $m = f(1) - f(0)$ muss also dasjenige x_t bestimmt werden, für das $m = f'(x_t) = -\pi(1 - x_t)^{\pi-1}$. Wir erhalten $x_t = 1 - (-m/\pi)^{1/(\pi-1)}$ (siehe Abbildung 29).

Mit

$$\begin{aligned}N^{\{3\}} &= [-m, 0, 1]^T, \\d_{\min}^{\{3\}} &= \langle N^{\{3\}}, [x_t, 0, f(x_t)]^T \rangle = -mx_t + f(x_t), \\d_{\max}^{\{3\}} &= \langle N^{\{3\}}, [0, 0, f(a)]^T \rangle = f(a).\end{aligned}$$

ist $N^{\{3\}}$ der gesuchte Normalenvektor und $d_{\min/\max}^{\{3\}}$ die Abstände bezüglich dieses Normalenvektors. Wir können also die neue Box möglicher Nullstellen gemäß Satz 3.4 berechnen und die ersten beiden Gleichungen des transformierten Systems subdividieren. Die dritte Gleichung bedarf keiner Subdivision, wir behandeln sie mit den oben aufgezeigten Techniken weiter. Wir erhalten so die Lösung $(x_1, x_2, x_3) = (0.234, 0.380, 0.433)$ nach 22 untersuchten Boxen mit Genauigkeit 10^{-3} .

Abbildung 29: Einschluss der Funktion $(1 - x_1)^\pi$ zwischen zwei parallele Geraden

4.6 Besselfunktionen

Betrachten wir hier nun noch exemplarisch ein Beispiel für ein nicht-polynomiales Gleichungssystem etwas allgemeinerer Art.

In den vorangegangenen Abschnitten haben wir Gleichungssysteme diskutiert, deren zugrunde liegende Funktionenräume F von der Form $F = \{f : \sum_{j=0}^n \alpha_j D^j f \equiv 0\}$ waren. Also Funktionenräume, die Lösungsmenge von linearen Differentialgleichungen mit konstanten Koeffizienten waren. Der Tatsache, dass wir es mit konstanten Koeffizienten zu tun hatten, war es zu verdanken, dass mit $f(x) \in F$ auch $f(x+d) \in F$ galt. Dies wiederum hatte zur Folge, dass die Splines selbst auch translationsinvariant waren. Wenden wir uns nun Gleichungssystemen zu, deren zugrunde liegende Funktionenräume Lösungsmenge von linearen Differentialgleichungen mit nicht-konstanten Koeffizienten sind, bzw. betrachten wir ein Beispiel eines solchen Gleichungssystems. Dabei wollen wir nicht der sonst üblichen Vorgehensweise folgen und gerade ein solches Beispiel herausgreifen, bei dem keine Probleme auftauchen, sondern ein solches, bei dem auch die Grenzen unseres Ansatzes deutlich werden.

Geben wir uns das Gleichungssystem

$$G(x, y) = \begin{bmatrix} Y_1(x) + 2J_1(y) + Y_1(x)J_1(y) \\ 3Y_1(x) - 2Y_1(y) \end{bmatrix} = 0$$

vor, wobei J_α bzw. Y_α die Besselfunktionen erster bzw. zweiter Art seien, die die Besselsche Differentialgleichung

$$x^2 f'' + x f' + (x^2 - \alpha^2) f = 0 \quad (4.20)$$

lösen. J_α und Y_α sind dabei linear unabhängig und spannen somit den zweidimensionalen Lösungsraum von (4.20) auf. Suchen wir nun alle Lösungen dieser Gleichung mit

$(x, y) \in B := [1, 5]^2$. Die vier zugrunde liegenden Funktionenräume (zwei Gleichungen mit je zwei Unbekannten) sind identisch und zwar zunächst $F = \langle J_1, Y_1 \rangle$.

Um Bernstein-Funktionen bauen zu können, die eine positive Partition der Eins bilden, muss natürlich auch $1 \in F$ sein. Um unser Verfahren zur Auffindung von Nullstellen durchführen zu können, muss zusätzlich die Identität Element von F sein. Betrachten wir nun also den Funktionenraum

$$F = \langle 1, Id, J_1, Y_1 \rangle.$$

Für diesen Raum ist zu klären, ob es sich um ein EC-System auf $[1, 5]$ handelt. Nur bei einer positiven Antwort können wir unseren Standard-Algorithmus durchführen. Im Fall einer negativen Antwort wäre zu hoffen, dass wenigstens stückweise EC-Systeme vorliegen (z.B. stückweise auf $[1, 2]$ und $[2, 5]$), sodass die Boxen aufgespalten werden können (z.B. in die vier Boxen $[1, 2] \times [1, 2]$, $[1, 2] \times [2, 5]$, $[2, 5] \times [1, 2]$ und $[2, 5] \times [2, 5]$). Zum Nachweis, dass es sich bei F um ein EC-System handelt ist zu zeigen, dass sowohl $F = \langle 1, Id, J_1, Y_1 \rangle$ als auch $F' = \langle 1, J'_1, Y'_1 \rangle$ C-Systeme sind.

Wenden wir uns zunächst der Frage zu, ob $F = \langle 1, Id, J_1, Y_1 \rangle$ auf $[1, 5]$ ein C-System ist. Dazu brauchen wir Satz 4.4.

Wir versuchen deshalb nun eine Sequenz $w_1, w_2, w_3, w_4 \in F$ zu finden, sodass $W(x, w_1)$, $W(x, w_1, w_2)$, $W(x, w_1, w_2, w_3)$, $W(x, w_1, w_2, w_3, w_4)$ allesamt für kein $x \in I = [1, 5]$ verschwinden. Wenn dies gelingt, so ist sichergestellt, dass es sich bei F um ein C-System auf I handelt.

Dies könnten wir auf numerischem Wege versuchen, wir wollen es aber streng analytisch tun. Fragen wir uns zunächst, ob es eine Lösung w_1 von (4.20) gibt, sodass w_1 auf $[1, 5]$ nirgendwo verschwindet.

Man kann zeigen, dass für beliebiges $w_1 \in F$ ein $x \in [1, 5]$ existiert, sodass $w_1(x) = 0$, folglich kann F kein C-System auf $[1, 5]$ sein. Jedoch gibt es sowohl für $I = [1, 2]$ als auch für $I = [2, 5]$ eine Linearkombination von Besselfunktionen, sodass diese auf I nirgendwo verschwindet.

Mit $w_1 = J_1$ und $w_2 = Y_1$ kann $W(x, w_1, w_2)$ nirgendwo verschwinden, weil J_1 und Y_1 den Lösungsraum einer linearen Differentialgleichung aufspannen und die Theorie der linearen Differentialgleichungen sagt uns, dass die Wronski-Determinante einer Basislösung stets von Null verschieden ist.

Wählen wir $w_3(x) = 1$. Nun kann man mit derselben Technik, wie wir sie im Anschluss vorführen werden, zeigen, dass $W(\cdot, w_1, w_2, w_3)$ in 1, jedoch nicht auf $(1, 5]$ verschwindet. Somit haben wir also schon wieder eine Problemstelle, nämlich bei $x = 1$, wir können also nicht garantieren, dass F auf $[1, 2]$ C-System ist.

Untersuchen wir nun, ob $W(x, w_1, w_2, w_3, w_4)$ auf $[1, 5]$ irgendwo verschwindet. Wie können wir diese Frage analytisch exakt beantworten? Zunächst sagt uns die Theorie der linearen Differentialgleichungen, dass die Wronski-Determinante einer Basislösung zu einer linearen Differentialgleichung stets von Null verschieden ist. Versuchen wir

also, eine lineare DGL zu finden, sodass F gerade die Lösungsmenge ist. Nun ist man versucht zu behaupten, dass diese DGL einfach $x^2 f'''' + x f''' + (x^2 - \alpha^2) f'' = 0$ sei. Diese Behauptung ist ebenso naheliegend wie falsch.

Zur Konstruktion der richtigen DGL leiten wir zunächst den Ausdruck

$$f'' = -\frac{x f' + (x^2 - \alpha^2) f}{x^2} \quad (4.21)$$

einmal ab und substituieren dann f'' auf der rechten Seite durch $-\frac{x f' + (x^2 - \alpha^2) f}{x^2}$. Dasselbe Spiel spielen wir mit der dann erhaltenen Gleichung für f''' nochmals. Somit erhalten wir die beiden Gleichungen

$$f''' = f(x) \left[\frac{x^2 - 3\alpha^2}{x^3} \right] + f'(x) \left[\frac{2 + \alpha^2 - x^2}{x^2} \right], \quad (4.22)$$

$$f'''' = f(x) \left[\frac{x^4 - 2x^2\alpha^2 + \alpha^4 + 11\alpha^2 - 3x^2}{x^4} \right] + f'(x) \left[\frac{2x^2 - 6\alpha^2 - 6}{x^3} \right]. \quad (4.23)$$

Nun fragen wir uns, wie die beiden rechten Seiten von (4.22) und (4.23) miteinander linear zu kombinieren sind, sodass das Ergebnis die rechte Seite von (4.21) ist. Mit den erhaltenen Koeffizienten $a(x), b(x)$ formulieren wir die DGL $f''(x) = a(x)f'''(x) + b(x)f''''(x)$. Diese DGL besitzt als Lösungen per Konstruktion nach wie vor J_α und Y_α und, da keine nullten und ersten Ableitungen auftauchen, auch 1 und die Identität. Es ergibt sich für $\alpha = 1$ nun $a(x) = \frac{x(9-x^2)}{(x^2-3)^2}$ und $b(x) = \frac{x^2}{3-x^2}$. Die Differentialgleichung ist für $x = \sqrt{3}$ folglich nicht definiert, es ist also zu befürchten, dass $W(\sqrt{3}, 1, Id, J_1, Y_1)$ verschwindet. Und tatsächlich besitzt die Funktion $W(\cdot, 1, Id, J_1, Y_1)$ bei $\sqrt{3}$ einen Vorzeichenwechsel, dessen Existenz numerisch leicht nachgewiesen werden kann durch Einsetzen von $\sqrt{3} - \delta$ und $\sqrt{3} + \delta$ mit z.B. $\delta = 0.1$.

Wir können die bisherigen Ergebnisse wie folgt zusammenfassen: F ist sicherlich auf $[2, 5]$ C-System. Definitiv nicht um ein C-System handelt es sich bei F auf einem Intervall I mit $\sqrt{3} \in I$. Falls $1 \in I$, so können wir nicht garantieren, dass F auf I C-System ist.

Schränken wir unsere weiteren Betrachtungen also ein auf das Intervall $[2, 5]$.

Zum Nachweis dass F EC-System auf $[2, 5]$ ist, bleibt noch zu zeigen, dass $F' = \langle 1, J'_1, Y'_1 \rangle$ C-System ist. Zunächst kann wieder gezeigt werden, dass es ein $w_1 \in \langle J'_1, Y'_1 \rangle$ gibt, das auf $[2, 5]$ nirgendwo verschwindet. Des weiteren ist $W(x, J'_1, Y'_1) = W(x, 1, J_1, Y_1)$ und $W(x, 1, J'_1, Y'_1) = W(1, Id, J_1, Y_1)$. Folglich ist nach unseren vorigen Überlegungen F' auf $[2, 5]$ C-System und somit endlich F als EC-System auf $[2, 5]$ erkannt.

Unser Algorithmus liefert die in Abbildung 30 visualisierten Boxen. Schon nach insgesamt 17 Schritten sind die beiden Lösungen in $[2, 5]^2$ mit Genauigkeit 10^{-3} eingegrenzt. Obige Probleme bedeuten natürlich eine starke Einschränkung bei der Anwendung unseres Algorithmus auf allgemeine Gleichungssysteme. Interessanterweise gibt es überhaupt keine Probleme, wenn man mit den modifizierten Besselfunktionen anstatt der

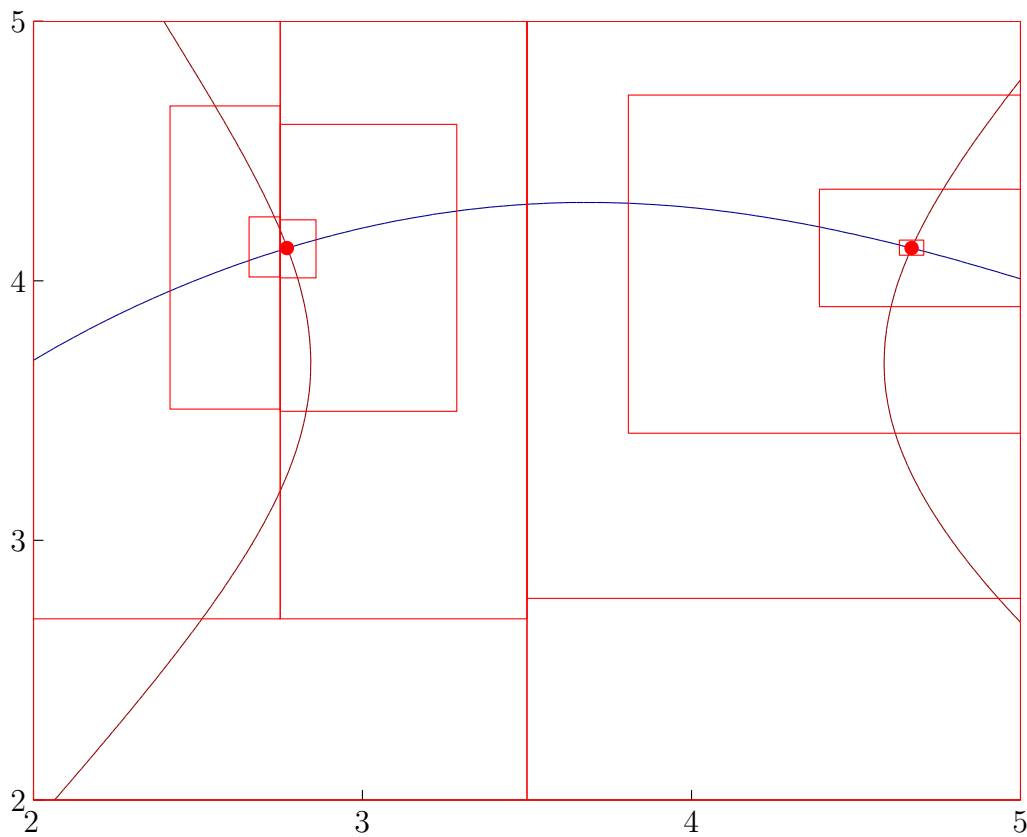


Abbildung 30: Eingrenzung der Nullstellen eines Gleichungssystems mit Besselfunktionen

Besselfunktionen arbeitet. Wenn also I_α und K_α Lösungen der DGL $x^2y'' + xy' - (x^2 + \alpha^2)y = 0$ sind, so handelt es sich, wie man zeigen kann, bei $\langle 1, Id, I_\alpha, K_\alpha \rangle$ für beliebiges α stets um ein EC-System auf jedem Intervall $I = [a, b]$ für $0 < a < b$.

5 Wertung und Ausblick

Wir haben in dieser Arbeit zunächst die Lösung von polynomialen Gleichungssystemen $p : [0, 1]^n \rightarrow \mathbb{R}^n$ untersucht. Durch die Betrachtung der Bézierdarstellung gelang es, die Bereiche möglicher Nullstellen effizient einzugrenzen und schließlich quadratisch konvergent in Intervalle einzuschließen. Dabei haben wir zwei Alternativen besprochen. Die erste Alternative baut vollständig auf der allgemeinen Béziertheorie auf, während die zweite sich neuer Erkenntnisse über Einschlussschätzer bedient. Erstaunlicherweise beweist sich der einfachere Algorithmus ohne Einschlussschätzer in den meisten Fällen als leistungsfähiger. Nur in wenigen Fällen zahlt sich die Arbeit mit Einschlussschätzern aus. Nun könnte man versuchen, mit adaptiven Strategien zu arbeiten. Z.B. kann man in einem Schritt zunächst ohne Einschlussschätzer arbeiten. Verkleinert sich dabei der Bereich möglicher Nullstellen nicht, so kann man erneut den Bereich möglicher Nullstellen mit den Einschlussschätzern berechnen.

Die Grenzen der Anwendbarkeit im polynomialen Fall liegen zum Einen in der Dimensionalität n des Problems, denn pro Verfahrensschritt muss eine $n \times n$ -Matrix invertiert werden, sodass $n > 100$ im Allgemeinen nicht realisierbar ist. Zum anderen muss die Zahl der zur Darstellung benötigten Kontrollpunkte beachtet werden (siehe dazu auch Abschnitt 3.1.2). Bleiben die Zahl der Kontrollpunkte und die Dimensionalität moderat, so erweisen sich unsere Algorithmen als äußerst leistungsfähig.

Das Vorgehen im polynomialen Fall übertragen wir anschließend auf nicht-polynomiale Systeme. Diese Übertragung stellt theoretisch kein Problem dar. In der Praxis hingegen zeigt sich, dass die Automatisierung der Behandlung völlig allgemeiner Gleichungssysteme sich als schwierig erweist (siehe dazu Abschnitt 4.6). Im Fall von exponentiell-trigonometrischen Gleichungssystemen gelang es uns, die Theorie völlig zufriedenstellend zu übertragen. Durch Transformation des Gleichungssystems (siehe dazu Abschnitt 4.5) lässt sich eine große Klasse von Systemen in exponentiell-trigonometrische Systeme überführen.

Während im polynomialen Fall keine Wünsche offenblieben, sind im exponentiell-trigonometrischen Fall noch leistungsfähigere Subdivisionsalgorithmen wünschenswert. Außerdem sind für allgemeine Gleichungssysteme effiziente Kriterien zur Entscheidung, ob ein EC-System vorliegt, bereitzustellen.

Alle vorgestellten Algorithmen lassen sich relativ problemlos in Intervall-Arithmetik implementieren, sodass tatsächlich verifizierte Lösungen berechnet werden können. Dabei ist zu beachten, dass nicht einfach sämtliche Rechenoperationen in Intervall-Arithmetik ausgeführt werden sollten. Vielmehr ist nach Berechnung der Box B möglicher Nullstellen in Intervall-Arithmetik auf das scharfe einschließende Intervall in Intervallarithmetik zu subdividieren. Das bedeutet insgesamt, dass mit scharfen Boxen und Intervall-Kontrollpunkten gearbeitet wird. Die Implementierung in Intervall-Arithmetik hat dann zur Folge, dass der Bereich möglicher Nullstellen nicht beliebig scharf eingegrenzbar ist. Spätestens, wenn für die Kontrollpunkt-Intervalle nicht mehr

das Vorzeichen festgestellt werden kann, kann keine Kontraktion mehr stattfinden. Nun wollen wir noch unsere Vorgehen gegenüber den Standard-Algorithmen werten. Wie in der Einleitung bereits aufgeführt lassen sich diese im Wesentlichen in algebraische Ansätze mittels Gröbner-Basen, in Homotopie-Ansätze, in Fixpunktiterationen wie das Newtonverfahren und allgemeine Subdivisionsalgorithmen wie allgemeine Intervall-Arithmetik unterteilen. Gegenüber den bereits erwähnten Problemen der Gröbner-Basen kommt noch die Übertragbarkeit auf nicht-polynomiale Systeme hinzu. Während Gröbner-Basen nur im polynomialen Fall einsetzbar sind, trägt das in dieser Arbeit vorgestellte Konzept weiter. Homotopie-Ansätze und Fixpunktiterationen benötigen im Allgemeinen gute Startnäherungen an die gesuchte Lösung. So ist es mit ihnen gewöhnlich nicht möglich zu entscheiden, ob neben den gefundenen Lösungen noch weitere Lösungen existieren. Wir haben neben der Gewähr, dass keine Lösung verloren wird sogar mit Satz 3.7 die Möglichkeit, die Existenz von Nullstellen ohne jeden Zusatzaufwand zu garantieren. Subdivisionsalgorithmen leiden im Gegensatz zu unserem Ansatz im Allgemeinen unter der lediglich linearen Konvergenzgeschwindigkeit, sodass diese gewöhnlich mit anderen Verfahren gekoppelt werden müssen, während wir mit einem Ansatz auskommen.

6 Bezeichnungen

D	Differenzialoperator
D_k	Differenzialoperator nach der k -ten Veränderlichen
\mathbb{P}_n	Menge der Polynome, deren Grad kleiner als n ist
$1 : n$	Menge der natürlichen Zahlen von 1 bis n
$\pi_i(V) := v_i$	Projektionsoperator für einen Vektor $V = (v_1, \dots, v_m)$
$\max_{i \in I} V_i$	für eine Menge $\{V_i : i \in I\} \subset \mathbb{R}^n$ sei definiert als das komponentenweise Maximum
$\text{mean } M$	Mittelwert der Elemente der Menge $M \subset \mathbb{R}$
$\#_T k$	sei bezogen auf eine Folge $T = (t_k)_k$ die Anzahl $ \{j > k : t_j = t_k\} $.
$*_T k$	sei bezogen auf eine Folge $T = (t_k)_k$ die Anzahl $ \{j : t_j = t_k\} $.
\square bzw. $()$	Vektorklammern

7 Literaturverzeichnis

Literatur

- [B85] B. Buchberger. Gröbner bases: An algorithmic method in polynomial ideal theory. In N. K. Bose, editor, *Multidimensional Systems Theory*, chapter 6, pages 184-232. D. Reidel Publishing, 1985.
- [Bo80] W. Boehm, Inserting new knots into B-spline curves, *Computer Aided Design* 12(4), 199-201, 1980.
- [C90] J. Canny. Generalized characteristic polynomials. *J. Symbolic Comput.* 9:241-250, 1990.
- [CLR80] E. Cohen, T. Lyche and R.F. Riesenfeld. Discrete B-splines and subdivision techniques in computer-aided geometric design and computer graphics, *Comput. Graphic Image Processing* 14, 87-111, 1980.
- [CMP96] Wonjoon Cho, Takashi Maekawa, Nicholas M. Patrikalakis, Topologically reliable approximation of composite Bézier curves, *Computer Aided Geometric Design* 13 (1996) pp. 497-520.
- [CS85] E. Cohen and L.L. Schumaker, Rates of convergence of control polygons, *Comp. Aided Geom. Design*, 1985, 2:229-235.
- [D91] Nira Dyn. *Subdivision Schemes in Computer aided geometric design. Advances in Numerical Analysis II, Wavelets, Subdivision Algorithms and Radial Functions*, W.A. Light (ed.), Oxford University Press, 1991, 36-104.
- [DB74] *Numerical Methods*, Prentice-Hall, Englewood Cliffs, NJ, 1974.
- [DL92] Nira Dyn and David Levin. Stationary and Non-Stationary Binary Subdivision Schemes. *Mathematical Methods in Computer Aided Geometric Design II*, Tom Lyche and Larry L. Schumaker(eds.). Academic Press, 1992, 209-216.
- [DR88] N. Dyn and A. Ron. Recurrence Relations for Tchebycheffian B-Splines, *J. Anal. Math.* 51:118-138, 1988.
- [F00] Gerald E. Farin. *The essentials of CAGD*, ISBN 1-56881-123-3, 2000
- [FR87] R.T. Farouki and V.T. Rajan. On the numerical condition of polynomials in Bernstein form, *Computer Aided Geometric Design* 4:191-216, 1987.
- [G83] A. Geisow. *Surface Interrogations*, Ph.D. thesis, School of Computing Studies and Accountancy, University of East Anglia, Norwich, UK, 1983.

- [Gaukel99] Joachim Gaukel. Lösung nichtalgebraischer Gleichungssysteme mit Hilfe von kardinalen L-Splines. Stuttgart, 1999.
- [GL80] C.B. Garcia and T.Y. Li, On the number of solutions to polynomial systems of equations, Siam J. Numer. Anal. Vol. 17, No4, August 1980, pp. 540-546.
- [H98] Klaus Höllig. Grundlagen der Numerik. MathText, Zavelstein, 1998.
- [Hi74] M.W. Hirsch and S. Smale. Differential Equations, Dynamical Systems and Linear Algebra. Academic Press (New York), 1974.
- [Ke99] R.B. Kearfott. Rigorous global search: Industrial applications. Dordrecht: Kluwer Academic Publishers. 1-16, 1999. ISBN 0-7923-6057-5.
- [K69] R. Krawczyk, Newton-Algorithmen zur Bestimmung von Nullstellen mit Fehlerschranken, Computing 4, 1969, pp. 187-201.
- [K96] Jürgen Koch. Parallele Algorithmen zur Lösung schwachbesetzter polynomialer Gleichungssysteme. Berlin, Köster, 1996.
- [LP01] D. Lutterkort, J. Peters, Optimized Refinable Enclosures of Multivariate Polynomial Pieces Computer Aided Geometric Design 18(9), 851-863, 2001.
- [LR81] J.M. Lane and R.F. Riesenfeld. Bounds on a polynomial, BIT: Nordisk Tidskrift for Informations-Behandling, 21(1): 112-117, 1981.
- [MJ77] R.E. Moore and S.T. Jones. Save starting regions for iterative methods. SIAM J. Numer. Anal., 14(6):1051-1065, 1977.
- [M69] R.E. Moore. Intervallanalyse. R. Oldenbourg Verlag, München und Wien, 1969.
- [M77] R.E. Moore. A test for existence of solutions to nonlinear systems. SIAM J. Numer. Anal., 14(4):611-615, 1977.
- [M78a] R.E. Moore. A computational test for convergence of iterative methods for nonlinear systems. SIAM J. Numer. Anal., 15(6):1194-1196, 1978.
- [M78b] R.E. Moore. Bounding sets in function spaces with applications to nonlinear systems. SIAM Review, 20(3):492-512, 1978.
- [M79] R.E. Moore. Methods and Applications of Interval Analysis. SIAM Studies in Applied Mathematics 2, 1979.
- [M88] Reliability in Computing. Academic Press, 1988.

- [Ma96] M.L. Mazure. Chebyshev spaces, RR 952M IMAG, University Joseph Fourier, Genoble, 1996.
- [Ma99] M.L. Mazure. Chebyshev-Bernstein bases, *Computer Aided Geometric Design* 16: 649-669, 1999.
- [ML96] M.L. Mazure and P.J. Laurent. Marsden Identity, Blossoming and de Boor-Fix Formula, *Advanced Topics in Multivariate Approximation* F. Fontanella, K. Jetter and P.J. Laurent (eds.) pp. 227-242.
- [ML99] M.L. Mazure and P.J. Laurent. Polynomial Chebyshev splines, *Computer Aided Geometric Design* 16:317-343, 1999.
- [MQ82] R.E. Moore and L. Qi. A successive interval test for nonlinear systems. *SIAM J. Numer. Anal.*, 19:845-850, 1982.
- [N90] A. Neumaier, *Interval Methods for Systems of Equations*, *Encyclopedia of Mathematics and its Applications*, Cambridge University press, 1990.
- [NPL99] D. Nairin, J. Peters, D. Lutterkort, Sharp quantitative bounds on the distance between a polynomial piece and its Bezier control polygon. *Computer Aided Geometric Design* 16:613-631, 1999.
- [P40] M.G. Pawley. New criteria for accuracy in approximating real roots by the Newton-Raphson method. *Nat. Math. Mag.* 15:111-120, 1940.
- [R98] W.C. Rheinboldt. *Methods for solving systems of nonlinear equations*. SIAM, 148 p., ISBN 0-89871-415-X, 1998.
- [Reif00] Ulrich Reif, Best bounds on the approximation of polynomials and splines by their control structure, *Computer Aided Geometric Design* 17:579-589, 2000.
- [S74] Larry L. Schumaker. *Spline Functions Basic Theory*. Academic Press (New York), 1974.
- [S79] Krzysztof Sikorski, A three-dimensional Analogue to the Method of Bisections for solving Nonlinear Equations, *Mathematics of Computation* Volume 33, Number 146, April 1979, pp. 722-738.
- [SP93] E.C. Sherbrooke and N.M. Patrikalakis, Computation of the solutions of nonlinear polynomial systems, *Computer Aided Geometric Design* 10, 379-405, 1993.
- [Spe] Melvin R. Spencer, *Polynomial Real Root Finding in Bernstein Form*, A Dissertation presented to the Department of Civil Engineering, Brigham Young University.

[W63] J.H. Wilkinson. Rounding Errors in Algebraic Processes, HMSO, London.

[ZG81] W.I. Zangwill and C.B. Garcia. Pathways to Solutions, Fixed Points, and Equilibria, Prentice-Hall, Englewood Cliffs, NJ, 1981.

Lebenslauf

Am 5. Oktober 1972 wurde ich, Joachim Gaukel, als Sohn von Ewald Gaukel und Elisabeth Gaukel, geb. Fischle, in Leonberg geboren.

- 8/79 - 7/83 Grundschule in Münchingen
- 8/83 - 6/92 Gymnasium Korntal
- 7/92 - 10/92 Wehrdienst in Mittenwald
- 11/92 - 9/93 Zivildienst im Karl-Olga-Krankenhaus in Stuttgart
- 10/93 - 9/99 Studium der Mathematik mit Nebenfach Informatik
an der Universität Stuttgart
- 8/97 - 12/97 Auslandssemester an der Universität Uppsala/Schweden
- 10/99 - 9/00 Hilfwissenschaftler an der Universität Stuttgart und
an der TU Darmstadt
- 10/00 - 7/03 Mitarbeiter an der TU Darmstadt, Fakultät Mathematik