Dissertation

Faktorisierung in Schief-Polynomringen

zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften
(Dr. rer. nat.)
im Fachbereich Mathematik der Universität Kassel

Vorgelegt von: Peter Horn

Eingereicht bei:
Prof. Dr. Wolfram Koepf

Im April 2008

UNIKASSEL
VERSITÄT

Tag der mündlichen Prüfung: 7. Oktober 2008

Erstgutachter:
Prof. Dr Wolfram Koepf
Universität Kassel

Zweitgutachter:
Prof. Dr. Volker Strehl
Friedrich-Alexander-Universität Erlangen-Nürnberg

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Algorithmen

Verzeichnis der Sitzungen

Symbolverzeichnis

Einleitung

Für die Beschäftigung mit Orealgebren gibt es prinzipiell zwei Motivationen. Aus einer abstrakt algebraischen Perspektive kommt man zu ihnen, wenn man Polynome in X über einem Körper betrachten will, bei denen man auf „echte“ Kommutativität verzichtet, aber noch ein gewisses Maß an Struktur fordert, nämlich dass sich die Nicht-Kommutativität durch zwei Abbildungen σ, δ ausdrücken lässt:

$$X \cdot a = \sigma(a) \cdot X + \delta(a).$$

Andererseits stellt man fest, dass sich Operatoralgebren wie die Algebren der Differential- oder Differenzenoperatoren in einem gemeinsamen Setting bewegen und strukturell weitgehend ähnlich betrachten lassen. In seiner Arbeit [59] „Theory of Non-Commutative Polynomials“ von 1933 stellt Oystein Ore die „[...] principal results of a general non-commutative polynomial theory“ erstmals vor, wobei er den direkten algebraischen Zugang wählt. In Kapitel 1 dieser Arbeit wird die Theorie der Orealgebren kurz zusammengefasst und folgt dabei im Groben der Darstellung in [59], [62] und [21].

Aus der Sicht der Operatoren kommt auf natürliche Weise die Fragestellung der Lösungen auf, aus der algebraischen Sicht stellt sich die Frage nach Faktorisierbarkeit. Ähnlich wie im Polynomfall hängen diese Fragen auch im Ore-Fall eng zusammen, wobei bei jeder Verallgemeinerung zusätzliche Probleme auftreten.

Für den Differential-Fall beschrieb Emanuel Beke schon 1894 in [16] einen Algorithmus für das Finden vor Rechtsfaktoren erster Ordnung. Leider ist dieser Algorithmus allerdings nicht effizient und praktisch nicht zu gebrauchen. In der Folge finden sich etwa in [77] Faktorisierungsalgorithmen für Differentialoperatoren, die das Problem aber praktisch auch nicht wirklich lösen. In seinen Arbeiten [42], [43] und [44] stellte Mark van Hoeij einen neuen Algorithmus für Rechtsfaktoren erster Ordnung vor, der auf „verallgemeinerten Exponenten“ und der geschickten Benutzung der Fuchs-Relationen beruht. Hiermit wird die Faktorisierung erstmals wirklich effizient möglich.

Während das Thema der Faktorisierung von Differentialoperatoren als befriedigend behandelt betrachtet werden kann, sind im reinen Endomorphismenfall, also

$$X \cdot a = \sigma(a) \cdot X$$

noch einige Fragen offen, insbesondere für den q-Fall $\sigma(f(x)) = f(q \cdot x)$ sind derzeit keine effizienten Algorithmen verfügbar. Deswegen hat diese Arbeit ihren Fokus auf den beiden Endomorphismus-Fällen des Shifts und des q-Shifts.

Ein wichtiges Werkzeug für die Faktorisierungsalgorithmen ist das **Newtonpolygon**, das insbesondere in der Betrachtung der p-adischen Zahlen bekannt ist, und dessen Spezialfall Eisenstein-Kriterium jedem Drittsemester-Studenten bekannt sein dürfte. Da man aus dem Newtonpolygon einige Kenndaten von Lösungen (bzw. Rechtsfaktoren) a priori ablesen kann, ist die zugehörige nicht-kommutative Theorie (bzw. der Teil, der in dieser Arbeit benötigt wurde) in Kapitel 2 zusammengefasst. Die Darstellung stützt sich dabei auf die klassischen Arbeiten von C. Raymond Adams [8], [9] und [10] und George Birkhoff [18]. Außerdem wurden noch die eher bewertungstheoretischen Arbeiten [74], [71], [50], [61] sowie die Arbeiten [15], [13], und [7], die sich direkt mit der Lösung von Operatorgleichunngen beschäftigen für das Kapitel über das Newtonpolygon herangezogen.
Die Definition 2.37 der „charakteristischen Gleichung“ wurde für den q-Fall angepasst, und der für diese Arbeit zentrale Satz 2.40 konnte vollkommen elementar bewiesen werden.

Der „Petkovšek“-Algorithmus aus den Arbeiten [63], [64] und [6] findet Rechtsfaktoren erster Ordnung im Shift- bzw. q-Shift-Fall, ist aber leider nicht besonders effizient – seine Laufzeit ist 2^{2k}, wenn k der maximale polynomielle Grad der Koeffizienten ist. In [30] stellen Thomas Cluzeau und Mark van Hoeij einen effizienten Algorithmus vor, der von den Ideen aus dem Differential-Fall inspiriert ist. Der dort präsentierte Algorithmus wird in der vorliegenden Arbeit vereinfacht und auf den q-Fall übertragen.
Bei der gemeinsamen Betrachtung des van Hoeij- und des Petkovšek-Algorithmus ist in Abschnitt 4.5 eine Idee entstanden, die den Petkovšek-Algorithmus dramatisch beschleunigt und – zumindest im q-Fall – durchaus kompetitiv macht. Auch die Formulierung in Algorithmus 16 (vgl. Seite 74) weicht von der Original-Darstellung ab, und ist deutlich einfacher.

In den bisherigen Arbeiten wurde im Wesentlichen nur auf die Berechnung von Rechtsfaktoren erster Ordnung eingegangen. In Kapitel 5 wird gezeigt, wie mit Hilfe des adjungierten Operators auch **Linksfaktoren** erster Ordnung berechnet werden können. Mit Hilfe von zyklischen Vektoren werden

dann **Faktoren beliebiger Ordnung** berechnet. Die algorithmischen Ideen finden sich – wenngleich nicht so explizit – zumindest in [70].

Ziel der Arbeit war, effiziente Algorithmen zur Faktorisierung von Shift- und q-Shiftoperatoren zu finden. Dies gelingt erst durch das Auffinden problemangepasster und wesentlich verbesserter Algorithmen der linearen Algebra, denn viele der vorgestellten Algorithmen werden letztendlich auf das Lösen von homogenen **linearen Gleichungssystemen** über dem Körper der rationalen Funktionen zurüchgeführt. Die Einträge der Matrizen sind hier dicht besetzt, so dass bei der Arithmetik riesige Ausdrücke entstehen, die sich kaum noch effizient behandeln lassen. Insbesondere in MuPAD ließen sich einige auftretende Gleichungssysteme gar nicht mehr lösen. In Kapitel 6 werden Algorithmen hergeleitet, die hier Abhilfe schaffen und viele Systeme behandeln, die vorher deutlich außer Reichweite waren. Die hergeleiteten Algorithmen sind durchaus von allgemeinem Interesse, über den zentralen Fokus dieser Arbeit hinaus.

Es gibt bereits einige Pakete für das Rechnen in nichtkommutativen Algebren, wie etwa „NCPoly“ in Reduce [58], „Plural“ in Singular [57], und „Felix“ [11], die aber alle auf rein polynomielle Koeffizienten ausgelegt sind, und ihren Schwerpunkt vor allem in der Berechnung von nichtkommutativen Gröbnerbasen haben. In Mathematica gibt es verschiedene Pakete, etwa [49] oder [2]. Auch in Maple sind Pakete für (q-)Rekursiongleichungen und deren Lösungen enthalten.

Die entwickelten Algorithmen wurden in MuPAD implementiert. Für die Wahl von MuPAD sprachen eine Reihe von Gründen: Zunächst existieren in MuPAD bisher keine zufriedenstellenden Algorithmen zur Lösung von Shift- bzw. q-Shiftoperatoren, dann erlaubt das Domain-Konzept von MuPAD eine sehr natürliche Notation und komfortable Programmierung (vgl. Sitzung 2 auf Seite 22). Der letzte Grund für die Wahl von MuPAD war, dass der Autor mit MuPAD vertraut ist, und seine Arbeit sicher auch in einer der nächsten Versionen in die Distribution einfließen wird.
Ein Teil der Algorithmen wurde von Torsten Sprenger parallel in Maple implementiert, und insbesondere die Lösungstheorie wird in seiner Arbeit [80] noch detaillierter behandelt.

Ich bedanke mich bei Wolfram Koepf für die Betreuung, bei Torsten Sprenger für viele fruchtbare Diskussionen. Ganz herzlicher Dank geht auch an den Zweitgutachter Prof. Volker Strehl, der einige Ungenauigkeiten und Verbessungsmöglichkeiten aufzeigte. Zuletzt möchte ich noch bei meiner Familie für ihre Geduld und einen weitgehenden Vater-Verzicht meiner Töchter in den letzten Monaten bedanken: *Papa, Du bist so faul – Du gehst immer an die Arbeit*.

Kapitel 1

Einführung

In diesem Kapitel werden die grundlegenden Objekte für diese Arbeit mit ihren wesentlichen Eigenschaften und den elementaren Algorithmen vorgestellt.

Die Darstellung in diesem Kapitel orientiert sich an [59], [70], [69], [83], [62], [27] und [21]. Für die allgemeine Theorie nichtkommutativer Ringe und Schiefkörper siehe etwa [40], [48], [32] und [31].

1.1 Grundlegende Konstruktion

Sei $\mathbf{F}$ ein kommutativer Körper der Charakteristik Null und ∂ eine Unbestimmte über $\mathbf{F}$. Wir wollen nun das nichtkommutative Analogon zum üblichen Polynomring in ∂ betrachten, suchen also eine Konstruktion, die

1. ein Linksmodul über $\mathbf{F}$ ist,
2. von $1, \partial, \partial^2, \partial^3, \ldots$ erzeugt wird, also ein freier Modul ist,
3. eine verträgliche Multiplikation zulässt.

Aus diesen Bedingungen folgt bereits, dass jedes Element ungleich Null eine Darstellung der Form

$$\sum_{i=0}^{n} a_i \partial^i \quad \text{mit} \quad n \in \mathbf{N},\ a_i \in \mathbf{F},\ a_n \neq 0 \tag{1.1}$$

hat. Um ∂a in der Form (1.1) schreiben zu können, müssen demnach für die Multiplikation Abbildungen $\sigma, \delta : \mathbf{F} \to \mathbf{F}$ existieren, so dass für jedes $a \in \mathbf{F}$ gilt

$$\partial\, a = \sigma(a)\, \partial + \delta(a).$$

Aus den Moduleigenschaften folgt weiterhin:

$$\begin{aligned}\sigma(a+b)\partial + \delta(a+b) &= \partial \cdot (a+b) = \partial\, a + \partial\, b \\ &= (\sigma(a) + \sigma(b))\,\partial + (\delta(a) + \delta(b)),\end{aligned}$$

also müssen σ und δ mit der Addition verträglich sein, außerdem:

$$\begin{aligned}\sigma(a\,b)\partial + \delta(a\,b) &= \partial \cdot (a\,b) = (\partial\, a)\,b = \sigma(a)\,\partial\, b + \delta(a)b \\ &= \sigma(a)(\sigma(b)\,\partial + \delta(b)) + \delta(a)b \\ &= \sigma(a)\,\sigma(b)\,\partial + \sigma(a)\,\delta(b) + \delta(a)b,\end{aligned}$$

also müssen $\sigma(a\,b) = \sigma(a)\,\sigma(b)$ und $\delta(a\,b) = \sigma(a)\,\delta(b) + \delta(a)b$ gelten.
Mit anderen Worten: σ muss ein Endomorphismus von $\mathbf{F}$ sein, und δ eine σ-Pseudoderivation, die unten in Definition 1.1 definiert wird.

1.2 Pseudoderivationen

Sei im Folgenden stets $\sigma : \mathbf{F} \to \mathbf{F}$ ein Endomorphismus.

Pseudoderivation

1.1 Definition. Eine *Pseudoderivation* bezüglich σ (oder σ-Pseudoderivation) ist eine Abbildung $\delta : \mathbf{F} \to \mathbf{F}$ mit

$$\delta(a+b) = \delta a + \delta b, \quad \delta(a \cdot b) = \sigma(a) \cdot \delta(b) + \delta(a) \cdot b \tag{1.2}$$

für alle $a, b \in \mathbf{F}$.

1.2 Beispiel.

1. Ist $\sigma = \mathrm{id}$, so ist δ gerade eine gewöhnliche Derivation auf $\mathbf{F}$.
2. Die Abbildung $\delta : a \mapsto 0$ ist eine Pseudoderivation für jeden Endomorphismus σ.

Innere Derivation δ_a

1.3 Satz und Definition. Sei $a \in \mathbf{F}$. Die Abbildung $\delta_a := a \cdot (\sigma - \mathrm{id})$ mit $\delta_a(b) = a\,(\sigma(b) - b)$ ist eine Pseudoderivation bezüglich σ. Sie heisst *innere Derivation*.

Beweis. Seien $b, c \in \mathbf{F}$. Dann ist

$$\begin{aligned}\delta_a(b\,c) &= a(\sigma(b\,c) - b\,c) = a\,\sigma(b)\,\sigma(c) - a\,b\,c \\ &= a\,\sigma(b)\,\sigma(c) - a\,\sigma(b)c + a\,\sigma(b)c - a\,b\,c \\ &= \sigma(b)\,a\,(\sigma(c) - c) + a(\sigma(b) - b)\,c = \sigma(b)\,\delta_a(c) + \delta_a(b)\,c.\end{aligned}$$

□

1.4 Lemma. *Ist $\mathbf{F}$ ein kommutativer Körper, σ ein Endomorphismus und δ eine σ-Pseudoderivation auf $\mathbf{F}$. Dann gilt:*

1. *Ist $\sigma \neq \mathrm{id}$, dann gibt es ein $\tilde{a} \in \mathbf{F}$ so dass $\delta = \delta_{\tilde{a}} = \tilde{a}(\sigma - \mathrm{id})$.*

2. *Ist* $\delta \neq 0$, *so gibt es ein* $\tilde{b} \in \mathbf{F}$ *mit* $\sigma = \tilde{b}\delta + \mathrm{id}$.

Beweis. Da $\mathbf{F}$ kommutativ ist, gilt $\delta(a\,b) = \delta(b\,a)$. Mit (1.2) erhält man daraus:

$$\delta(a\,b) = \sigma(a)\delta(b) + \delta(a)\,b = \sigma(b)\delta(a) + \delta(b)\,a = \delta(b\,a) \tag{1.3}$$
$$\iff (\sigma(a) - a)\delta(b) = (\sigma(b) - b)\delta(a). \tag{1.4}$$

1. Ist $\sigma \neq \mathrm{id}$, so existiert ein $a \in \mathbf{F}$ mit $\sigma(a) \neq a$. Setze $\tilde{a} := \frac{\delta(a)}{\sigma(a)-a}$. Wegen (1.4) ist dann $\delta(b) = \tilde{a}(\sigma(b) - b)$ für alle $b \in \mathbf{F}$. Also ist $\delta = \tilde{a}(\sigma - \mathrm{id})$.

2. Ist $\delta \neq 0$, so existiert ein $a \in \mathbf{F}$ mit $\delta(a) \neq 0$. Setze $\tilde{b} := \frac{\sigma(a)-a}{\delta(a)}$. Dann ist – wiederum wegen (1.4) – für alle $b \in \mathbf{F}$: $\delta(b) = \tilde{b}\,\delta(b) + b$, also $\delta = \tilde{b}\,\delta + \mathrm{id}$.

□

1.5 Definition. 1. Die Teilmenge von $\mathbf{F}$

Fixkörper $\mathrm{Fix}(\sigma)$

$$\mathrm{Fix}(\sigma) := \{a \in \mathbf{F} \mid \sigma(a) = a\}$$

ist ein Körper und heißt der *Fixkörper zu* σ.

Konstantenkörper $\mathrm{Const}_{\sigma,\delta}(\mathbf{F})$

2. Die Teilmenge von $\mathbf{F}$

$$\mathrm{Const}_{\sigma,\delta}(\mathbf{F}) := \{a \in \mathbf{F} \mid \sigma(a) = a \wedge \delta a = 0\} = \mathrm{Fix}(\sigma) \cap \mathrm{Kern}(\delta)$$

ist ein Unterkörper von $\mathbf{F}$ und heißt der Unterkörper der *Konstanten* von $\mathbf{F}$ bezüglich σ und δ.

Beachte, dass auch $\mathrm{Kern}(\delta)$ ein Körper ist.

1.3 Univariate Orealgebren

Sei im Folgenden $\mathbf{F}$ ein Körper, $\sigma : \mathbf{F} \to \mathbf{F}$ ein Endomorphismus und δ eine σ-Pseudoderivation.

Orealgebra $\mathbf{F}[\partial; \sigma, \delta]$

1.6 Definition. Die *Links-Orealgebra* $\mathbf{F}[\partial; \sigma, \delta]$ gegeben durch σ und δ ist der Ring $(\mathbf{F}[\partial; \sigma, \delta], +, \cdot)$ mit der üblichen Addition und der Multiplikation, die durch

$$\partial \cdot a = \sigma(a) \cdot \partial + \delta(a), \quad \text{für alle } a \in \mathbf{F} \tag{1.5}$$

gegeben ist. Die Multiplikation (1.5) erweitert sich auf ganz $\mathbf{F}[\partial; \sigma, \delta]$. Die Elemente von $\mathbf{F}[\partial; \sigma, \delta]$ heißen *Orepolynome*.

Schief- und Differentialpolynomring

Ist $\delta = 0$, so heißt $\mathbf{F}[\partial; \sigma]$ ein *Schiefpolynomring*. Ist $\sigma = \mathrm{id}$, so heißt $\mathbf{F}[\partial; \delta]$ ein *Differentialpolynomring*.

Normalform

1.7 Bemerkung. Jedes von Null verschiedene Element $A \in \mathbf{F}[\partial; \sigma, \delta]$ lässt sich in eindeutiger Weise schreiben als

$$A = a_n \partial^n + a_{n-1} \partial^{n-1} + \ldots + a_1 \partial + a_0.$$

mit $a_0, \ldots, a_n \in \mathbf{F}$ und $a_n \neq 0$.

Ordnung

Wir nennen n die *Ordnung* von A.

1.8 Bemerkung. Damit bekommt man für das Produkt zweier Orepolynome

$$\sum_{i=0}^{m} a_i \partial^i \cdot \sum_{j=0}^{n} b_j \partial^j = \sum_{i=0}^{m} \sum_{j=0}^{n} (a_i \partial^i \cdot b_j \partial^j)$$

Für $\partial^n a$ kann – analog zu (1.5) – eine Rekursionsformel angegeben werden. Sei z eine Unbestimmte, die mit σ und δ kommutiert und definiere den Operator $A_{n,j}$ durch

$$\sum_{j=0}^{n} A_{n,j} z^j = (\sigma + \delta z)^n.$$

Dann ist

$$\begin{aligned}
\sum_{i=0}^{m} a_i \partial^i \cdot \sum_{j=0}^{n} b_j \partial^j &= \sum_{i=0}^{m} \sum_{j=0}^{n} (a_i \partial^i \cdot b_j \partial^j) \\
&= \sum_{i=0}^{m} \sum_{j=0}^{n} a_i \Big(\sum_{l=0}^{i} A_{i,l}(b_j) \partial^l \Big) \partial^j \\
&= \sum_{k=0}^{m+n} \Big(\sum_{i=0}^{m} \sum_{l=0}^{i} a_i A_{i,l}(b_{k-l}) \Big) \partial^k
\end{aligned}$$

und das ist die gewünschte Normalform.

1.9 Bemerkung. Sind $A = a \partial^m + \ldots$ und $B = b \partial^n + \ldots$ zwei Orepolynome, so ist der Leitkoeffizient des Produktes

$$\mathrm{lc}(A \cdot B) = a \sigma^m(b) \partial^{m+n}.$$

1.10 Lemma. *Es gilt:* $\mathbf{F}[\partial; \sigma, \delta]$ *ist genau dann ein Integritätsbereich, wenn* σ *injektiv ist.*

Beweis. Seien $A, B \in \mathbf{F}[\partial; \sigma, \delta] \setminus \{0\}$ mit $A \cdot B = 0$. Es genügt ohne Einschränkung, $A = a\partial^m$ und $B = b\partial^n$ mit $a, b \in \mathbf{F}^\times$ zu betrachten (ggf. als Einschränkung auf die Leitterme), dann muss der Leitterm von $A \cdot B$ verschwinden, also $0 = a\sigma^m(b)$, und da $\mathbf{F}$ keine Nullteiler hat, muss $\sigma^m(b) = 0$ sein, damit kann σ nicht injektiv sein.
Die andere Richtung ist offensichtlich. □

Im Folgenden sei deswegen σ stets ein Monomorphismus, also injektiv.

Die letzten beiden Aussagen liefern den folgenden Algorithmus zur Rechtsdivision bei Schiefpolynomen:

Algorithmus 1: rightdivide(A, B)

Input: $A, B \in \mathbf{F}[\partial; \sigma, \delta],\ n = \deg A \geq m = \deg B,\ A = a\,\partial^n + \ldots,\ B = b\,\partial^m + \ldots$
Output: Ein Paar $(Q, R) \in \mathbf{F}[\partial; \sigma, \delta]^2$ mit $A = QB + R$ und $\deg R < \deg B$ oder $R = 0$.

```
1  i ← 0, A_0 ← A, B_0 ← B, Q ← 0
2  while deg A_i ≥ deg B_i und A_i ≠ 0
3  |  n_i ← deg A_i, a_i ← lc A_i
4  |  Q_i ← a_i / σ^{n_i − m}(b) ∂^{n_i − m}
5  |  Q ← Q + Q_i
6  |  A_{i+1} ← A_i − Q_i · B
7  |  i ← i + 1
8  return (A_{i−1}, Q)
```

1.11 Bemerkung. Für $A, B \in \mathbf{F}[\partial; \sigma, \delta]$, $\deg A \geq \deg B$, das ist das Paar (Q, R) mit den Eigenschaften $A = QB + R$ und $\deg R < \deg B$ oder $R = 0$ eindeutig bestimmt.

Beweis. Seien $A = QB + R$ und $A = Q'B + R'$ zwei solche Darstellungen, dann ist $R - R' = (Q' - Q)B$. Wegen der Gradbedingung an R muss also $Q' - Q = 0$ gelten und damit auch $R = R'$. □

1.12 Lemma. *1. Für $A, B \in \mathbf{F}[\partial; \sigma, \delta]$ gilt $\deg(AB) = \deg A + \deg B \geq \max\{\deg A, \deg B\}$.*

2. Die Funktion

$$A \longmapsto \begin{cases} 0 & A = 0 \\ 2^{\deg(A)} & \textit{sonst} \end{cases}$$

ist eine euklidische Norm auf $\mathbf{F}[\partial; \sigma, \delta]$, es gibt also einen rechts-euklidischen Algorithmus.

Linksteiler, Rechtsteiler, $|_l$ $|_r$

1.13 Definition. Für $A \in \mathbf{F}[\partial; \sigma, \delta]$ nennen wir jedes $B \in \mathbf{F}[\partial; \sigma, \delta]$ einen *Linksteiler von* A, wenn es ein $C \in \mathbf{F}[\partial; \sigma, \delta]$ gibt mit $A = B \cdot C$. Dies notieren wir mit $B \,|_l\, A$.
Analog ist $B \in \mathbf{F}[\partial; \sigma, \delta]$ ein *Rechtsteiler von* A, wenn es ein $C \in \mathbf{F}[\partial; \sigma, \delta]$ gibt mit $A = C \cdot B$. Dies notieren wir mit $B \,|_r\, A$.

größter gemeinsamer Rechtsteiler, $\operatorname{gcrd}(A, B)$

1.14 Definition. Seien $A, B \in \mathbf{F}[\partial; \sigma, \delta]$. Das eindeutig bestimmte normierte Orepolynom $G \in \mathbf{F}[\partial; \sigma, \delta]$ mit

$$C \,|_r\, A \wedge C \,|_r\, B \Longrightarrow C \,|_r\, G$$

bezeichnen wir als *größten gemeinsamen Rechtsteiler* $\operatorname{gcrd}(A, B)$ von A und B.

Algorithmus 2: $\mathtt{gcrd}(A, B)$

Input: $A, B \in \mathbf{F}[\partial; \sigma, \delta]$ normiert, $\deg A \geq \deg B > 0$
Output: $G \in \mathbf{F}[\partial; \sigma, \delta]$ wie in Definition 1.14

```
1 repeat
2 |  (Q, R) ← rightdivide(A, B)
3 |  break if R = 0
4 |  (A, B) ← (B, Q)
5 return 1/lc B · B
```

Die folgenden zwei Aussagen werden wir nicht explizit benutzen, wollen sie aber der Vollständigkeit halber erwähnen:

1.15 Korollar. *Es gelten:*

1. $\mathbf{F}[\partial; \sigma, \delta]$ *ist ein Links-Hauptidealring.*
2. $\mathbf{F}[\partial; \sigma, \delta]$ *ist links-noethersch.*

1.16 Lemma. *Es gilt:* $\mathbf{F}[\partial; \sigma, \delta]$ *ist genau dann auch Rechts-Hauptidealring, wenn* σ *bijektiv ist. Insbesondere gibt es dann auch einen links-euklidischen Algorithmus.*

Beweis. Ist σ ein Automorphismus, so ist $a\partial = \partial\sigma^{-1}(a) - \delta(\sigma^{-1}(a))$, und man bekommt für $A \in \mathbf{F}[\partial; \sigma, \delta]$ eine eindeutige Darstellung $A = \partial^n a_n + \ldots + \partial a_1 + a_0$. Damit gibt es einen Algorithmus zur Linksdivision analog zu Algorithmus 1 rightdivide (vgl. Seite 15). Damit ist $\mathbf{F}[\partial; \sigma, \delta]$ auch Rechts-Hauptidealring. □

kleinstes gemeinsames Linksvielfaches, $\mathrm{lclm}(A, B)$

1.17 Satz und Definition. Seien $A, B \in \mathbf{F}[\partial; \sigma, \delta]$. Dann gibt es ein eindeutig bestimmtes normiertes Orepolynom $G \in \mathbf{F}[\partial; \sigma, \delta]$ mit

$$A \mid_r C \wedge B \mid_r C \Longrightarrow G \mid_r C. \tag{1.6}$$

Dieses bezeichnen wir als *kleinstes gemeinsames Linksvielfaches* $\mathrm{lclm}(A, B)$ von A und B.

Für den Beweis benötigen wir ein kleines Lemma:

1.18 Lemma. *Wenn das kleinste gemeinsame Linksvielfache existiert, dann gilt: Stimmen die Reste von* $A, B \in \mathbf{F}[\partial; \sigma, \delta]$ *bei Rechtsdivision durch* $C \in \mathbf{F}[\partial; \sigma, \delta]$ *überein, so ist*

$$\mathrm{lclm}(A, C) = c \cdot \mathrm{lclm}(B, C) \cdot B^{-1} \cdot A \tag{1.7}$$

mit normierendem $c \in \mathbf{F}$. *(Beachte: Die Division durch* B *ist exakt.)*

Beweis. Existiert $L = \operatorname{lclm}(A, C)$, dann gibt es ein $C_1 \in \mathbf{F}[\partial; \sigma, \delta]$ minimalen Grades mit $L = C_1 A$. Nach Voraussetzung gilt $A = B + Q C$, und damit

$$L = C_1 A = C_1 B + C_1 Q C,$$

und da L rechtsteilbar durch C ist, muss auch $C_1 B$ durch C teilbar sein. Demnach wird der niedrigste Grad von C_1 angenommen, wenn $C_1 B = \operatorname{lclm}(B, C)$. □

Beweis von Satz 1.17. Wir folgen hier [59]. Um die Existenz des kleinsten gemeinsamen Linksvielfachen zu zeigen, leiten wir eine explizite Formel mit Hilfe des erweiterten euklidischen Algorithmus her. Dieser liefert uns mit $A_1 = A$ und $A_2 = B$ eine Folge

$$\begin{aligned}
A_1 &= Q_1 \cdot A_2 + A_3 \\
A_2 &= Q_2 \cdot A_3 + A_4 \\
&\vdots \quad \vdots \\
A_{m-2} &= Q_{m-2} \cdot A_{m-1} + A_m \\
A_{m-1} &= Q_{m-1} \cdot A_m.
\end{aligned}$$

Die wiederholte Anwendung von (1.7) auf $\operatorname{lclm}(A, B)$ liefert

$$\begin{aligned}
\operatorname{lclm}(A_1, A_2) &= a_1 \operatorname{lclm}(A_2, A_3) A_3^{-1} A_1 \\
&= a_2 \operatorname{lclm}(A_3, A_4) A_4^{-1} A_2 A_3^{-1} A_1 \\
&= \dots \\
&= a_{m-1} \operatorname{lclm}(A_m, A_{m-1}) A_m^{-1} A_{m-2} A_{m-1}^{-1} \cdots A_2 A_3^{-1} A_1, \\
&= a_m A_{m-1} A_m^{-1} A_{m-2} A_{m-1}^{-1} \cdots A_3 A_4^{-1} A_2 A_3^{-1} A_1,
\end{aligned}$$

also

$$\operatorname{lclm}(A, B) = a A_{m-1} A_m^{-1} A_{m-2} A_{m-1}^{-1} \cdots A_3 A_4^{-1} A_2 A_3^{-1} A_1 \qquad (1.8)$$

wobei $a, a_1, \dots, a_m \in \mathbf{F}^\times$ die rechten Seiten normieren.
Es bleibt zu zeigen, dass die rechte Seite von (1.8) tatsächlich immer ein Orepolynom ist, dass von A_1 und A_2 geteilt wird. Dies zeigen wir durch Induktion. Setze

$$\tilde{L}_i := A_{m-1} A_m^{-1} A_{m-2} A_{m-1}^{-1} \cdots A_{i+1} A_{i+2}^{-1} A_i,$$

dann ist $\tilde{L}_{m-1} = A_{m-1}$ durch A_m und A_{m-1} rechtsteilbar. Sei nun gezeigt, dass für $\tilde{L}_{i+1}$ durch A_{i+1} und A_{i+2} rechtsteilbar ist. Dann ist $\tilde{L}_i = \tilde{L}_{i+1} A_{i+2}^{-1} A_i$, und $\tilde{L}_i$ ist zumindest ein Orepolynom, das durch A_i rechtsteilbar ist.
Benutze nun die Identität $A_i = Q_i A_{i+1} + A_{i+2}$ aus dem erweiterten euklidischen Algorithmus, dann gilt

$$\tilde{L}_i = \tilde{L}_{i+1} A_{i+2}^{-1} Q_i A_{i+1} + \tilde{L}_{i+1},$$

und $\tilde{L}_i$ ist auch durch A_{i+1} rechtsteilbar. □

Nachdem wir nun gesehen haben, dass das kleinste gemeinsame Linksvielfache stets existiert und eindeutig ist, wollen wir es auch berechnen. Hierzu benutzen wir allerdings nicht die explizite Formel aus (1.8), sondern den Ansatz, einen allgemeinen Operator mit unbestimmten Koeffizienten c_i zu konstruieren und dann durch A und B eine Rechtsdivision mit Rest durchzuführen. Die Koeffizienten der Reste liefern dann nach Koeffizientenvergleich ein homogenes lineares Gleichungssystem, dessen Resultat dann in den Ansatz eingesetzt wird. Ist die Dimension des Lösungsraumes größer als Null, wird der Grad des Ansatzes reduziert.
Hierbei wird der Algorithmus (A1) rightdivide implizit über der transzendenten Erweiterung $\mathbf{F}(c_0, \ldots, c_{n-1})$ ausgeführt.

Algorithmus 3: lclm(A, B)

Input: $A, B \in \mathbf{F}[\partial; \sigma, \delta]$, $\deg A, \deg B > 0$
Output: $G \in \mathbf{F}[\partial; \sigma, \delta]$ wie in Gleichung (1.6).

```
1  n ← deg A + deg B,   d ← 1
2  while d ≠ 0
3  |  C ← ∂^n + Σ_{i=0}^{n-1} c_i ∂^i
4  |  (q_1, r_1) ← rightdivide(C, A)
5  |  (q_2, r_2) ← rightdivide(C, B)
6  |  S ← linsolve(r_1 = 0, r_2 = 0)
7  |  d ← dim S,   n ← n − d
8  return subs(C, S)
```

1.4 Beispiele

Es ist offenbar ausreichend anzugeben, wie σ und δ auf x operieren.

	Name	$\mathbf{F}$	$\sigma(x)$	$\delta(x)$	$\partial \cdot x$
[O1]	Differentialpolynome	$\mathbf{Q}(x)$	x	1	$x\,\partial + 1$
[O2]	Shiftpolynome	$\mathbf{Q}(x)$	$x+1$	0	$(x+1)\,\partial$
[O3]	q-Differenz	$\mathbf{Q}(x, q)$	$q\,x$	$q\,x - x$	$qx\,\partial + (q-1)x$
[O4]	Euler-Derivation	$\mathbf{Q}(x)$	x	x	$x\,\partial + x$

Diese Liste ist bei Weitem nicht vollständig, deckt aber die *typischen* Fälle ab. In [27], [26] findet sich ein sehr schöner und umfangreicher Überblick.

1.5 Klassifikation, Generatorsubstitution

1.19 Bemerkung. Wollen wir $R = \mathbf{F}[\partial; \sigma, \delta]$ durch einen anderen Generator $\tilde{\partial} = a\partial + b$, $a \in \mathbf{F}^\times$, $b \in \mathbf{F}$ erzeugen, so müssen wir auch einen neuen

Endomorphismus $\tilde{\sigma}$ und eine $\tilde{\sigma}$-Derivation $\tilde{\delta}$ einführen. Wir erhalten dann $\partial = a^{-1}\tilde{\partial} - a^{-1}b$ und wegen

$$\tilde{\partial}c = \begin{cases} \tilde{\sigma}(c)\tilde{\partial} + \tilde{\delta}(c) = \tilde{\sigma}(c)a\partial + \tilde{\sigma}(c)b + \tilde{\delta}(c) \\ (a\partial + b)c = a\sigma(c)\partial + a\,\delta(c) + bc \end{cases}$$

muss nach Koeffizientenvergleich gelten:

$$\tilde{\sigma}(c) = a\,\sigma(c)\,a^{-1} \tag{1.9}$$

$$\tilde{\delta}(c) = a\,\delta(c) + b\,c - a\,\sigma(c)\,a^{-1}\,b. \tag{1.10}$$

Für kommutative Körper heißt das, dass der Endomorphismus unverändert bleibt.

1.20 Lemma. *Ist δ eine innere Derivation $\delta = \delta_c = c(\sigma - \mathrm{id})$ (vgl. 1.3), so kann mit der Substitution $\tilde{\partial} = \partial + c$ erreicht werden, dass $\tilde{\delta} = 0$. Dann ist $\mathbf{F}[\tilde{\partial}; \tilde{\sigma}]$ ein Schiefpolynomring.*

Beweis. Nach der Substitution $\tilde{\partial} = a\partial + b$ ist wegen (1.10) $\tilde{\delta} = a\,\delta + b\,\mathrm{id} - b\,\sigma$, und mit $a = 1$ und $b = c$ ist $\tilde{\delta} = 0$. □

1.21 Definition. In Verallgemeinerung von Definition 1.6 wollen wir den Polynomring $\mathbf{F}[\partial; \sigma, \delta]$ auch dann einen Schief- oder Differentialpolynomring nennen, wenn er durch eine Generatorsubstitution $\partial \mapsto \tilde{\partial}$ in einen solchen überführt werden kann.

Zusammengefasst erhalten wir:

1.22 Satz. *Ist $\mathbf{F}$ kommutativ und σ injektiv, so ist $\mathbf{F}[\partial; \sigma, \delta]$ **entweder** ein differentieller **oder** ein Schiefpolyomring.*

Damit können wir uns in der weiteren Betrachtung auf die zwei Fälle der differentiellen oder Schiefpolynomringe beschränken.

1.23 Beispiel. In Abschnitt 1.4 haben wir als [O3] die Orealgebra der q-Differenzen kennengelernt. Der Endomorphismus ist gegeben durch

$$\sigma : \mathbf{F}(x) \longrightarrow \mathbf{F}(x), \qquad x \longmapsto q \cdot x$$

und die Derivation durch

$$\delta : \mathbf{F}(x) \longrightarrow \mathbf{F}(x), \qquad x \longmapsto q\,x - x.$$

Damit ist $\delta = \delta_1 = \sigma - \mathrm{id}$ ein innerer Endomorphismus und mit $\tilde{\partial} = \partial + 1$ ist

$$\tilde{\partial}\,x = (\partial + 1)\,x = qx(\partial + 1) - x + x = qx(\partial + 1) = qx\,\tilde{\partial}$$

und $\tilde{\delta} = 0$.

Da die differentiellen Polynomringe durch Mark van Hoeijs Arbeiten [29, 43, 42, 44] umfassend behandelt sind, betrachten wir in dieser Arbeit nur die Schiefpolynomringe.
Die Automorphismen von $\mathbf{F}(x)$ sind von der Form

$$\sigma(x) = \frac{a\,x + b}{c\,x + d} \quad \text{mit} \quad \det\begin{pmatrix} a & b \\ c & d \end{pmatrix} \neq 0,$$

damit haben wir $\mathrm{Aut}(\mathbf{F}(x)) \cong \mathrm{PGL}(\mathbf{F}, 2)$. $\mathrm{PGL}(\mathbf{F}, 2)$ ist dabei die Gruppe der invertierbaren 2×2-Matrizen über $\mathbf{F}$ bis auf skalare Vielfache. Jede Matrix $S \in \mathrm{PGL}(\mathbf{F}, 2)$ hat mit einer Matrix $U \in \mathrm{Gl}(\mathbf{F}, 2)$ eine Jordandarstellung

$$U\,S\,U^{-1} = \begin{cases} \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix} & \text{(Fall 1)}, \\ \begin{pmatrix} \alpha & 1 \\ 0 & \alpha \end{pmatrix} & \text{(Fall 2)}. \end{cases}$$

In (Fall 1) kann es nötig sein, zu einer quadratische Körpererweiterung überzugehen, da α, β nicht notwendigerweise in $\mathbf{F}$ liegen müssen.
Durch den Übergang von σ zu $\tilde{\sigma}$, dessen Matrixdarstellung $U\,S\,U^{-1}$ ist, können wir uns bei der Betrachtung der Schiefpolynomringe auf die zwei Fälle der *Translation* $x \mapsto x + q$ bzw. der *Dilatation* $x \mapsto q\,x$ beschränken.
Die Dilatation ist gerade der q-Shift-Fall [O3], für die allgemeine Translation lassen sich die Aussagen über den „klassischen“ Shift-Fall [O2] verallgemeinern, so dass diese beiden Fälle den kompletten Schiefpolynom-Fall abdecken.
Für den Fall positiver Charakteristik ist das Faktorisierungsproblem in [39] behandelt.

1.6 Operatorschreibweise

Wie wir in Abschnitt 1.4 (bzw. genauer in [27], [26]) bereits gesehen haben, sind alle relevanten Beispiele für Orealgebren Operatoralgebren. Es ist also naheliegend und komfortabel, die Orepolynome als Operatoren auszufassen. Dazu interpretieren wir das Symbol ∂ als lineare Abbildung $\mathbf{F} \to \mathbf{F}$. Die Elemente $a \in \mathbf{F}$ operieren kanonisch durch Links-Multiplikation auf $\mathbf{F}$. Wir erhalten dadurch eine Operatoralgebra $\mathbf{F}[\partial]$, deren Wirkung wir nun an den Beispielen aus Abschnitt 1.4 studieren wollen. Sei dazu $f(x) \in \mathbf{Q}[x]$ bzw. $f(x) \in \mathbf{Q}(q)[x]$ für [O3].

	Name	$\sigma(f(x))$	$\delta(f(x))$	$\partial\,x$	∂
[O1]	Differentialpolynome	$f(x)$	$f'(x)$	$x\,\partial + 1$	δ
[O2]	Shiftpolynome	$f(x+1)$	0	$(x+1)\,\partial$	σ
[O3]	q-Differenz	$f(q\,x)$	0	$qx\,\partial$	σ
[O4]	Euler-Derivation	$f(x)$	$x\,f'(x)$	$x\,\partial + x$	δ

Im Folgenden werden wir stets die Operatorschreibweise benutzen, wir identifizieren also

$$L = \sum_{i=0}^{n} a_i \, \partial^i \in \mathbf{F}(x)[\partial], \quad a_i \in \mathbf{F}(x),$$

mit der linearen Abbildung

$$L : \mathbf{F}(x) \to \mathbf{F}(x), \quad u(x) \mapsto \sum_{i=0}^{n} a_i \cdot \partial^i(u(x)),$$

Beachte: Wegen Satz 1.22 muss man den gemischten Fall nicht betrachten.

1.7 Notation

Für den Rest der Arbeit sei stets $\mathbf{F}$ ein berechenbarer Körper der Charakteristik null, x eine Unbestimmte über $\mathbf{F}$, $\mathbf{F}(x)$ der Körper der rationalen Funktionen in x und

$$\sigma : \mathbf{F}(x) \longrightarrow \mathbf{F}(x)$$

ein bijektiver Automorphismus mit $\operatorname{Fix} \sigma = \mathbf{F}$. An einigen Stellen wird es wichtig sein, welche Art von Automorphismus betrachtet wird, deshalb führen wir hier noch explizite Notationen ein.

Shift $\tau(x) = x + 1$

Für den **Shift-Fall** bezeichne mit

$$\tau : \mathbf{F}(x) \longrightarrow \mathbf{F}(x), \qquad x \longmapsto x + 1$$

den Shiftoperator.

Im q**-Shift-Fall** ist $\mathbf{K}$ ein berechenbarer Körper der Charakteristik 0, q eine Unbestimmte über $\mathbf{K}$ und $\mathbf{F} = \mathbf{K}(q)$ die transzendente Erweiterung. $\mathbf{F}(x)$ ist hier also der Körper der rationalen Funktionen in x, deren Koeffizienten wieder rationale Funktionen in q mit Koeffizienten aus $\mathbf{K}$ sind. Der Endomorphismus ist hier

q-Shift $\epsilon(x) = q\,x$

$$\epsilon : \mathbf{F}(x) \longrightarrow \mathbf{F}(x), \qquad x \longmapsto q\,x$$

der q-Shiftoperator.

Ringe $\mathcal{S}$, $\mathcal{S}_\tau$, $\mathcal{S}_\epsilon$

Zur Vereinfachung der Notation bezeichne mit $\mathcal{S} := \mathbf{F}(x)[\sigma]$ den allgemeinen Schiefpolynomring und mit $\mathcal{S}_\tau$ bzw. $\mathcal{S}_\epsilon$ die Ringe der Shift- bzw. q-Shiftoperatoren.

1.8 Implementierung

In MuPAD gibt es ein Domain-Konzept, vergleichbar mit Klassen in objektorientierten Programmiersprachen. Dieses erlaubt eine intuitive Notation und gewährt eine gewisse Typsicherheit.
Im Rahmen dieser Arbeit sind zwei Domains entstanden, die die beiden Schiefpolynomringe der Shiftoperatoren sowie der q-Shiftoperatoren repräsentieren. Ab einer der nächsten MuPAD-Versionen werden sie als zusätzliches Package sumtools zur Verfügung stehen. Durch die Befehle

MuPAD-Session 1: Initialisierung

```
>> package("sumtools")

>> R := Dom::ShiftOperators(x, S, Dom::Rational);
                    Dom::ShiftOperators(x, S, ℚ)
>> qR := Dom::qShiftOperators(q, x, Eq);
                    Dom::qShiftOperators(q, x, Eq)
```

wird das Paket geladen und die Domains R der Shiftoperatoren mit x als Unbestimmter, S als Shiftoperator und Dom::Rational als Koeffizientenkörper sowie qR der q-Shift Operatoren mit q, x als Unbestimmter und Eq als q-Shiftoperator erzeugt.
Im Folgenden nehmen wir stets an, dass die Domains so initialisiert wurden.

1.8.1 Dom::ShiftOperators

MuPAD-Session 2: Elementare Arithmetik mit Dom::ShiftOperators

Wir definieren einen Operator L1 und wenden ihn auf einige Ausdrücke an:

```
>> L1 := R(S+3)
                              S + 3
>> L1(x+1)
                             4 x + 5
>> L1(x^2+2)
                          4 x^2 + 2 x + 9
>> L1(f(x))
                        f(x + 1) + 3 f(x)
```

Jetzt definieren wir einen zweiten Operator und betreiben ein wenig elementare nichtkommutative Arithmetik:

```
>> L2 := R((x+1)*S+x^2-2)
```

$$(x+1)\,S + (x^2-2)$$

```
>> L1 * L2
```

$$(x+2)\,S^2 + (x^2+5\,x+2)\,S + (3\,x^2-6)$$

```
>> L2 * L1
```

$$(x+1)\,S^2 + (x^2+3\,x+1)\,S + (3\,x^2-6)$$

```
>> L1 + L2
```

$$(x+2)\,S + (x^2+1)$$

Hier benutzen wir die Algorithmen für den größten gemeinsamen Rechtsteiler bzw. das kleinste gemeinsame Linksvielfache:

```
>> R::gcrd(L1, L2)
```

$$1$$

```
>> L3 := R::lclm(L1, L2)
```

$$S^2 + \left(\frac{-x^4-2\,x^3+12\,x^2+31\,x+16}{-x^3+x^2+11\,x+10}\right) S - \frac{3\,x^4-3\,x^3-27\,x^2+6\,x+42}{-x^3+x^2+11\,x+10}$$

```
>> R::rightdivide(L3, L1)
```

$$\left[S + -\frac{x^4-x^3-9\,x^2+2\,x+14}{-x^3+x^2+11\,x+10}, 0\right]$$

```
>> [Q, RR] := R::rightdivide(L3, L2)
```

$$\left[\left(\frac{1}{x+2}\right) S + \frac{-3\,x^2+3\,x+21}{-x^3+x^2+11\,x+10}, 0\right]$$

```
>> (Q*L2+RR)-L3
```

$$0$$

```
>> (L2*Q+RR)-L3
```

$$\left(-\frac{2}{x+3}\right) S^2 + \left(-\frac{2\,x^6+2\,x^5-40\,x^4-56\,x^3+264\,x^2+626\,x+336}{x^6+x^5-23\,x^4-43\,x^3+111\,x^2+331\,x+210}\right) S$$

1.8.2 Dom::qShiftOperators

MuPAD-Session 3: Elementare Arithmetik mit Dom::qShiftOperators

Wir erstellen nun – wie oben – zwei Operatoren, mit denen wir dann etwas Arithmetik betreiben:

```
>> L1 := qR((x+1)*Eq^2+3*q*x*Eq+12*x^2+1)
```

$$(x+1)\,Eq^2 + (3\,q\,x)\,Eq + 12\,x^2 + 1$$

```
>> L1(f(x))
```

$$f(q^2 x) + f(x) + 12 x^2 f(x) + x f(q^2 x) + 3 q x f(q x)$$

```
>> L2 := qR((3*x+q*x^2+4)*Eq+q*x+1)
```

$$(q x^2 + 3 x + 4)\,Eq + q x + 1$$

```
>> L1 + L2
```

$$(x+1)\,Eq^2 + (q x^2 + (3q+3) x + 4)\,Eq + 12 x^2 + q x + 2$$

```
>> L1 * L2
```

$$\left(q^5 x^3 + \left(q^5 + 3 q^2\right) x^2 + \left(3 q^2 + 4\right) x + 4\right) Eq^3 +$$

$$\left(\left(3 q^4\right) x^3 + \left(q^3 + 9 q^2\right) x^2 + \left(q^3 + 12 q + 1\right) x + 1\right) Eq^2 +$$

$$\left((12 q)\, x^4 + 36 x^3 + \left(3 q^3 + q + 48\right) x^2 + (3 q + 3)\, x + 4\right) Eq +$$

$$(12 q)\, x^3 + 12 x^2 + q x + 1$$

```
>> L2 * L1
```

$$\left(q^2 x^3 + (4 q)\, x^2 + (4 q + 3)\, x + 4\right) Eq^3 +$$

$$\left(\left(3 q^3\right) x^3 + \left(9 q^2 + q\right) x^2 + \left(12 q^2 + q + 1\right) x + 1\right) Eq^2 +$$

$$\left(\left(12 q^3\right) x^4 + \left(36 q^2\right) x^3 + \left(51\ q^2 + q\right) x^2 + (3 q + 3)\, x + 4\right) Eq +$$

$$(12 q)\, x^3 + 12 x^2 + q x + 1$$

```
>> LL1 := L1 * qR(x*Eq+1):
>> LL2 := L2 * qR(x*Eq+1):
>> qR::gcrd(LL1, LL2)
```

$$Eq + \frac{1}{x}$$

```
>> L3 := qR::lclm(L1, L2)
```

$$Eq^3 + \left(\frac{(-36 q^{11})\, x^9 + \ldots + (-q^3 - 205 q^2 - 30 q - 17)\, x - 17}{(-12 q^{10})\, x^9 + \ldots + (-55 q^2 - 72 q - 52)\, x - 68}\right) Eq^2 +$$

$$\left(\frac{(-144 q^{11})\, x^{10} + \ldots + (-55 q^2 - 55 q - 52)\, x - 68}{(-12 q^{10})\, x^9 + \ldots + (-55 q^2 - 72 q - 52)\, x - 68}\right) Eq +$$

$$\frac{(-144 q^{10})\, x^8 + \ldots + (-q^3 - q^2 - 13 q)\, x - 17}{(-12 q^9)\, x^8 + \ldots + (-55 q^2 - 4 q - 52)\, x - 68}$$

```
>> [Q1, R1] := qR::rightdivide(L3, L1):
>> [Q2, R2] := qR::rightdivide(L3, L2):
```

```
>> R1, R2
                                    0, 0
>> Q1*L1 - L3, Q2*L2 - L3
                                    0, 0
```

Damit haben wir die elementaren Operationen zusammen, um höhere Algorithmen zu implementieren und – insbesondere – unkompliziert mit Shift- und q-Shiftoperatoren in MuPAD zu experimentieren.

Das MuPAD-Package sumtools finden Sie auf der beiliegenden CD im Verzeichnis „packages“.
Alle in dieser Arbeit abgedruckten Sessions finden Sie auf der CD im Verzeichnis „MuPAD-Sessions“ als MuPAD-Notebook.

Kapitel 2

Das Newtonpolygon

Das Newtonpolygon ist eine geometrische Figur, die einem Operator zugeordnet werden kann und aus der man sehr einfach einige überraschend detaillierte Informationen über mögliche Lösungen ablesen kann.

Dieses Kapitel stellt eine Übertragung vom differentiellen Fall [50], [74] und [71] dar, welche – vor allem im q-Shift-Fall – nicht trivial ist. Insbesondere die angepasste Definition der charakteristischen Gleichung (Definition 2.37) und der Beweis zu Satz 2.40 sind neu.

Im kommutativen Fall ist die Theorie des Newtonpolygons wohlbekannt und – implizit z. B. beim Eisensteinkriterium – weit verbreitet. Leider lässt sich die Theorie nicht an jeder Stelle wörtlich auf den Schiefpolynom-Fall übertragen.

2.1 Lösungen

Wir benutzen das Newtonpolygon, um Informationen über die Lösungen eines Operators abzulesen. Deswegen ist es notwendig, die wichtigsten Lösungstypen hier kurz zu definieren.

Lösung

2.1 Definition. Sei $L = \sum_{i=0}^{n} a_i \sigma^i \in \mathcal{S}$ und $u(x)$ ein Ausdruck in Elementen aus $\mathbf{F}(x)$, der möglicherweise weitere Funktionen aus einem geeigneten Erweiterungskörper beinhaltet. $u(x)$ nennen wir eine *Lösung* von L, wenn $L(u) = \sum_{i=0}^{n} a_i \sigma^i(u(x)) = 0$ gilt.
Wir nennen $u(x)$ eine *polynomielle Lösung*, wenn $u(x) \in \mathbf{F}[x]$ und eine *rationale Lösung*, wenn $u(x) \in \mathbf{F}(x)$ gilt.

σ-hypergeometrische Lösung

$u(x)$ heißt σ-*hypergeometrisch* (oder einfach *hypergeometrisch*), wenn

$$\operatorname{cert}_\sigma(u) := \frac{\sigma(u(x))}{u(x)} \in \mathbf{F}(x)$$

gilt. Auf hypergeometrische Lösungen wird in Kapitel 4 noch näher eingegangen, Definition 4.1 führt noch einige weitere Begriffe ein.

2.2 Bewertung

Im Folgenden sei R ein Ring, wir werden später stets den Fall $R = \mathbf{F}[x]$ des univariaten, kommutativen Polynomrings über $\mathbf{F}$ betrachten.

Bewertung ν

2.2 Definition. Eine Abbildung $\nu : R \to \mathbf{R} \cup \{\infty\}$ heißt eine *Bewertung auf* R, wenn für alle $a, b \in R$ gelten:

(i) $\nu(0) = +\infty$ und $\nu(a) = \infty \Leftrightarrow a = 0$,

(ii) $\nu(a \cdot b) = \nu(a) + \nu(b)$

(iii) $\nu(a + b) \geq \min\{\nu(a), \nu(b)\}$.

(iv) $\nu(a) \neq \nu(b) \Longrightarrow \nu(a + b) = \min\{\nu(a), \nu(b)\}$.

trailing degree $\operatorname{tdeg}(f)$

2.3 Definition. Zu einem Polynom $f = \sum_{i=0}^{m} f_i x^i$ definiere den *trailing degree* tdeg als die niedrigste auftretende x-Potenz:

$$\operatorname{tdeg}(f) = \min\{i \mid f_i \neq 0\}.$$

2.4 Beispiel.

(i) Auf jedem Ring ist $\nu_{\text{triv}}(a) = \begin{cases} \infty & \text{für } a = 0 \\ 0 & \text{für } a \neq 0 \end{cases}$
eine Bewertung, die die *triviale Bewertung* genannt wird.

Betrachte nun $R = \mathbf{F}[x]$:

(ii) Die Abbildung $\nu_{\text{tdeg}}(a) = \begin{cases} \operatorname{tdeg}(a) & a \neq 0 \\ \infty & a = 0 \end{cases}$
ist eine Bewertung.

(iii) Die Abbildung $\nu_{\deg}(a) = \begin{cases} -\deg a & a \neq 0 \\ \infty & a = 0 \end{cases}$
ist eine Bewertung.

2.5 Bemerkung. Will man allgemein eine Bewertung ν des Rings R zu einer Bewertung $\tilde{\nu}$ des Polynomrings $R[X]$ ausweiten, so ist sie durch den Wert $\tilde{\nu}(X)$ eindeutig bestimmt, denn aus den Bedingungen für eine Bewertung folgt für $a, a_i \in R$

$$\tilde{\nu}(a X^i) = \nu(a) + i\,\tilde{\nu}(X) \quad \text{und}$$
$$\tilde{\nu}\Big(\sum\nolimits_i a_i X^i\Big) = \min\nolimits_i \tilde{\nu}(a_i X^i).$$

Damit bestimmt $r := \tilde{\nu}(X) \in \mathbf{R}$ die Erweiterung der Bewertung eindeutig.

2.6 Definition. Zu einer Bewertung v auf dem Ring R und $r \in \mathbf{R}$ definiere auf $R[X]$

Steigungs-Bewertung v_r

$$v_r\big(\sum\nolimits_i a_i X^i\big) := \min_i(v(a_i) + i\,r).$$

Wir nennen v_r die *Steigungs-Bewertung*.

2.7 Bemerkung. Die Bewertungen $v_{\deg}$ und v_{tdeg} gehen aus der trivialen Bewertung v_{triv} genau so hervor:

$$v_{\mathrm{tdeg}}\Big(\sum_i a_i x^i\Big) = v_1\Big(\sum_i a_i x^i\Big) = \min_i\{v(a_i) + i\} = \min_{a_i \neq 0} i,$$
$$v_{\deg}\Big(\sum_i a_i x^i\Big) = v_{-1}\Big(\sum_i a_i x^i\Big) = \min_i\{v(a_i) - i\} = -\max_{a_i \neq 0} i.$$

2.8 Bemerkung.

(i) Eine Bewertung v auf einem Ring R kann eindeutig zu einer Bewertung auf dem Quotientenkörper von R fortgesetzt werden durch $v(\frac{a}{b}) = v(a) - v(b)$. Dies ist auch wohldefiniert.

(ii) Ist $\mathbf{F}$ ein Körper und v eine Bewertung auf $\mathbf{F}$, so kann diese auf den algebraischen Abschluss von $\mathbf{F}$ fortgesetzt werden.

2.3 Das Bewertungspolygon

2.3.1 Der kommutative Fall

In diesem Abschnitt sei stets $R = \mathbf{F}[Y]$ und $P = \sum_{i=0}^{n} a_i Y^i \in R$.

2.9 Definition. Zu $P \in R$ heißt die Funktion $\mathcal{V}_{v,P} : \mathbf{R} \to \mathbf{R},\ r \mapsto v_r(P)$ die *Bewertungsfunktion von* P.

Bewertungs-funktion $\mathcal{V}_{v,L}$

Der Graph der Bewertungsfunktion ist stetig, konkav und stückweise linear. Er heißt das *Bewertungspolygon von* P.

Bewertungs-polygon

Definiere

$$N_r(P) := \max\{i \mid v(a_i) + ir = v_r(P)\} \in \mathbf{N},$$
$$n_r(P) := \min\{i \mid v(a_i) + ir = v_r(P)\} \in \mathbf{N}.$$

2.10 Lemma. *Es gelten:*

(i) $N_r(P)$ *bzw.* $n_r(P)$ *sind die Ableitungen von* $v_r(P)$ *für sehr große* $r \gg 0$ *bzw. sehr kleine* $r \ll 0$.

(ii) Für alle $t < r \in \mathbf{R}$ *gilt:* $N_t(P) \geq n_t(P) \geq N_r(P) \geq n_r(P)$.

exzeptionelle Stellen

2.11 Definition. Die Menge $\{r \in \mathbf{R} \mid N_r(P) \neq n_r(P)\}$ heißt die Menge der *exzeptionellen Stellen*.

2.12 Bemerkung. Es gibt nur endlich viele exzeptionelle Stellen. Dieses sind die Stellen, an denen das Bewertungspolygon seine Steigung ändert.

2.13 Lemma. *Sei* $P \in R$ *ein kommutatives Polynom und* $a \in \mathbf{F}$*, dann ist*

$$\nu(P(a)) \geq \nu_{\nu(a)}(P)$$

und Gleichheit gilt genau dann, wenn a *keine exzeptionelle Stelle ist.*

Beweis. Sei $P = a_0 + a_1Y + \ldots + a_nY^n$. Dann gilt

$$\nu(P(a)) \geq \min\{\nu(a_0), \nu(a_1) + \nu(a), \ldots, \nu(a_n) + n\nu(a)\} = \nu_{\nu(a)}(P).$$

□

2.14 Lemma. *Seien* $P, Q \in \mathbf{F}[Y]$*. Dann gelten für alle* $r \in \mathbf{R}$

$$\begin{aligned} \nu_r(P \cdot Q) &= \nu_r(P) + \nu_r(Q), \\ N_r(P \cdot Q) &= N_r(P) + N_r(Q), \\ n_r(P \cdot Q) &= n_r(P) + n_r(Q). \end{aligned}$$

(Beweis siehe allgemeinere Aussage 2.20.)

2.15 Satz. *Seien* $\overline{\mathbf{F}}$ *der algebraische Abschluss von* $\mathbf{F}$, $P \in \mathbf{F}[Y]$ *und* $\Lambda = \{\lambda_1, \ldots, \lambda_n\} \subseteq \overline{\mathbf{F}}$ *die Nullstellen von* P*, dann gelten:*

(i) $N_r(P) = |\{\lambda \in \Lambda \mid \nu(\lambda) \geq r\}|$,

(ii) $n_r(P) = |\{\lambda \in \Lambda \mid \nu(\lambda) > r\}|$,

(iii) $N_r(P) - n_r(P) = |\{\lambda \in \Lambda \mid \nu(\lambda) = r\}|$.

Insbesondere ist die Bewertung einer Nullstelle stets eine exzeptionelle Stelle von P.

Beweis. Wir zeigen die Aussage für normierte Polynome durch Induktion nach dem Grad, dazu muss die Bewertung auf den algebraischen Abschluss $\overline{\mathbf{F}}$ ausgeweitet werden. Sei $P = Y - a$ ein Polynom vom Grad 1. Dann sind die beiden Kanten des Bewertungspolygons $0 + r$ und $\nu(a)$, diese schneiden sich genau an der Stelle $r = \nu(a)$.
Sei nun Q ein Polynom vom Grad n und $P = Y - a$. Dann ist $\nu_r(Q \cdot P) = \nu_r(Q) + \nu_r(P)$ als punktweise Summe stückweise linearer Funktionen wieder stückweise linear, wobei die Summe an jeder exzeptionellen Stelle eines der Summanden auch eine exzeptionelle Stelle hat. Die Steigungen addieren sich ebenfalls. (Zur Veranschaulichung siehe Abbildung 2.1) □

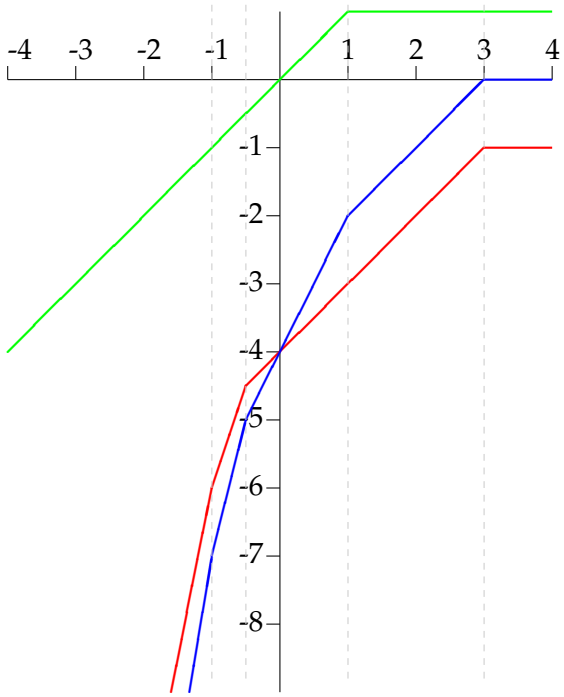

Abbildung 2.1: Summe von Polygonen zum Beweis von Satz 2.15

2.3.2 Der nichtkommutative Fall

Sei im Folgenden stets

$$L = \sum_{i=0}^{n} a_i \sigma^i \in \mathbf{F}[x][\sigma]$$

ein Schiefpolynom der Ordnung n mit polynomiellen Koeffizienten von maximalem Grad $m \in \mathbf{N}$.

verträgliche Bewertung

2.16 Definition. Wir nennen eine Bewertung ν *mit* σ *verträglich*, wenn für jedes Element $a \in \mathbf{F}[x]$ gilt $\nu(a) = \nu(\sigma(a))$.

2.17 Beispiel. (i) Für $\sigma = \epsilon_q$ ist $\nu = \nu_{tdeg}$ verträglich.

(ii) Für $\sigma = \epsilon_q$ ist $\nu = \nu_{deg}$ verträglich.

(iii) Für $\sigma = \tau_x$ ist $\nu = \nu_{tdeg}$ nicht verträglich, denn für $0 < k \in \mathbf{N}$ ist $\nu_{tdeg}(x^k) = k \neq 0 = \nu_{tdeg}(\tau_x(x^k))$.

(iv) Für $\sigma = \tau_x$ ist $\nu = \nu_{deg}$ verträglich.

Im Folgenden wollen wir immer Bewertungen betrachten, die mit dem benutzten Automorphismus verträglich sind.

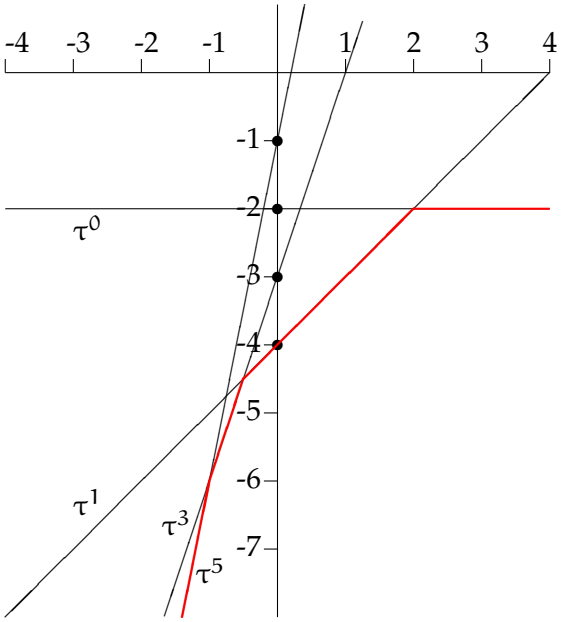

Abbildung 2.2: Bewertungspolygon zu (2.1) aus Beispiel 2.19

2.18 Definition. Die folgenden Definitionen übertragen sich mitsamt den gezeigten Eigenschaften wörtlich auf den nichtkommutativen Fall:

(i) Die *Steigungs-Bewertung* von L und $r \in \mathbf{R}$

$$\nu_r(L) = \nu_r\left(\sum_{i=0}^{n} a_i \sigma^i\right) := \min\{\nu(a_i) + r \cdot i \mid i = 0, \ldots, n\},$$

(ii) die *Bewertungsfunktion* von L: $\mathcal{V}_{\nu,L} : \mathbf{R} \to \mathbf{R},\ r \mapsto \nu_r(L)$

(iii) das *Bewertungspolygon* von L,

(iv) $N_r(L) := \max\{i \mid \nu(a_i) + ir = \nu_r(L)\}$ und
$n_r(L) := \min\{i \mid \nu(a_i) + ir = \nu_r(L)\}$

(v) die Menge der *exzeptionellen Stellen* $\{r \in \mathbf{R} \mid N_r(L) \neq n_r(L)\}$

2.19 Beispiel. Sei

$$L := (4x + 1)\tau^5 + (4x^3 + 2x - 2)\tau^3 + (x^4 + 2)\tau + 2x^2 + 1, \tag{2.1}$$

und benutze die Bewertung $\nu_{\deg}$. Dann ist das zugehörige Bewertungspolygon in Abbildung 2.2 zu sehen. Die exzeptionellen Stellen sind gerade die Abszissen der Schnittpunkte der Geraden, die das Bewertungspolygon bilden: $-1,\ -\frac{1}{2},\ 2$.

2.20 Satz. *Für* $r \in \mathbf{R}$ *und* $L, M \in \mathbf{F}[x;\sigma]$ *gilt*

$$\nu_r(L \cdot M) = \nu_r(L) + \nu_r(M).$$

Beweis. Seien $r \in \mathbf{R}$, $L = \sum_{i=0}^{n} a_i\sigma^i$ und $M = \sum_{j=0}^{m} b_j\sigma^j$ und i_0, j_0 minimal mit $\nu_r(L) = \nu_r(a_{i_0}\sigma^{i_0}) = \nu(a_{i_0}) + i_0 r$ bzw. $\nu_r(M) = \nu_r(b_{j_0}\sigma^{j_0}) = \nu(b_{j_0}) + j_0 r$. Dann ist

$$L \cdot M = \sum_{k=0}^{n+m} \Bigg(\sum_{\substack{0\le i\le n\\ 0\le j\le m\\ i+j=k}} a_i\sigma^i(b_j) \Bigg)\sigma^k$$

und damit

$$\begin{aligned}
\nu_r(L \cdot M) &= \min_{k=0,\dots,m+n} \left\{ \nu\Bigg(\sum_{i+j=k} a_i\sigma^i(b_j) \Bigg) + k r \right\} \\
&\ge \min_{k=0,\dots,m+n} \left\{ \min_{i+j=k} \{\nu(a_i\sigma^i(b_j))\} + k r \right\} \\
&= \min_{k=0,\dots,m+n} \left\{ \min_{i+j=k} \{\nu(a_i) + \nu(\sigma^i(b_j)) + (i+j) r\} \right\} \\
&= \nu(a_{i_0}) + \nu(b_{j_0}) + (i_0 + j_0) r \;=\; \nu_r(L) + \nu_r(M).
\end{aligned}$$

Sei nun $i_0 + j_0 = k_0$, dann ist für alle i, j mit $(i, j) \neq (i_0, j_0)$ und $i + j = k_0$

$$\nu(a_i\,\sigma^i(b_j)) = \nu(a_i) + \nu(b_j) > \nu(a_{i_0}) + \nu(b_{j_0}) = \nu(a_{i_0}\,\sigma^{i_0}(b_{j_0})).$$

Insbesondere ist

$$\nu\Bigg(\sum_{\substack{i+j=k_0\\ (i,j)\neq(i_0,j_0)}} a_i\,\sigma^i(b_j) \Bigg) \ge \min_{\substack{i+j=k_0\\ (i,j)\neq(i_0,j_0)}} \nu(a_i\,\sigma^i(b_j)) > \nu(a_{i_0}\,\sigma^{i_0}(b_{j_0}))$$

und damit

$$\begin{aligned}
\nu\Bigg(\sum_{i+j=k_0} a_i\,\sigma^i(b_j) \Bigg) &= \nu\Bigg(a_{i_0}\,\sigma^{i_0}(b_{j_0}) + \sum_{\substack{i+j=k_0\\ (i,j)\neq(i_0,j_0)}} a_i\,\sigma^i(b_j) \Bigg) \\
&= \nu(a_{i_0}\,\sigma^{i_0}(b_{j_0})),
\end{aligned}$$

also gilt Gleichheit. Der allen Ausdrücken gemeinsame Summand $r\,k_0$ wurde hier weggelassen. □

Damit ist dieser zentrale Satz auch für den nichtkommutativen Fall richtig.

2.21 Bemerkung. Sei $L := \sum_{i=0}^{n} a_i\,\sigma^i \in \mathcal{S}$ ein Operator und $u(x)$ eine hypergeometrische Lösung von L, also $L(u) = 0$ und $\lambda := \mathrm{cert}_\sigma(u) = \frac{\sigma(u(x))}{u(x)} \in \mathbf{F}(x)$. Dann ist $\sigma - \lambda$ ein Rechtsfaktor von L.

Die nichtkommutative Verallgemeinerung von Satz 2.15 ist damit:

2.22 Satz. *Sei* $\Lambda := \{\lambda \mid \sigma - \lambda \text{ ist Rechtsfaktor von } L\}$. *Dann gelten:*

(i) $N_r(L) = |\{\lambda \in \Lambda \mid \nu(\lambda) \geq r\}|$,

(ii) $n_r(L) = |\{\lambda \in \Lambda \mid \nu(\lambda) > r\}|$,

(iii) $N_r(L) - n_r(L) = |\{\lambda \in \Lambda \mid \nu(\lambda) = r\}|$.

Insbesondere ist die Bewertung des Zertifikats einer Lösung stets eine exzeptionelle Stelle von L.

Beweis. Analog zum Beweis von Satz 2.15. □

2.23 Korollar. *Hat* L *eine polynomielle oder eine rationale Lösung, so muss das Bewertungspolygon einen exzeptionelle Stelle in* 0 *haben.*

Beweis. Folgt aus $\nu(a) = \nu(\sigma(a))$. □

2.4 Das Newtonpolygon

2.4.1 Die Legendretransformation

Um den Zusammenhang zwischen Bewertungs- und Newtonpolygon herzustellen, benötigen wir die Legendretransformation. Diese soll hier kurz eingeführt werden. Für mehr Details siehe etwa [12] oder [37].

Legendre-transformation $f^\star$

2.24 Definition. Die Abbildung $f \mapsto f^\star$, die einer konvexen Funktion $f : \mathbf{R} \to \mathbf{R}_\infty$ die Funktion

$$f^\star : t \mapsto \max_x\{t\,x - f(x)\}$$

zuordnet, heißt *Legendretransformation* von f. Die Funktion $f^\star$ ist die *Legendretransformierte* von f.

2.25 Bemerkung. Es gilt $(f^\star)^\star = f$, die Legendretransformation ist also involutiv.

2.26 Beispiel. Betrachte die Funktion $f : \mathbf{R} \to \mathbf{R}, x \mapsto \frac{1}{4}x^2 + 1$, dann wird das Maximum über x von $t\,x - f(t)$ an der Stelle mit $f'(x) = t$ also $x(t) = 2\,t$ angenommen. Wir können die Legendretransformierte also direkt angeben: $f^\star(t) = x(t)\,t - f(x(t)) = t^2 - 1$ (vgl Abbildung 2.3). Führen wir das gleiche Verfahren für $g : \mathbf{R} \to \mathbf{R}, x \mapsto x^2 - 1$ durch, so bekommen wir mit analoger Argumentation $g^\star(t) = \frac{1}{4}x^2 + 1$.

Im Folgenden betrachen wir polygonale (also stückweise lineare, stetige Funktion mit endlich vielen „Knicken“) konvexe Funktionen, die also nicht „streng“ konvex sind.

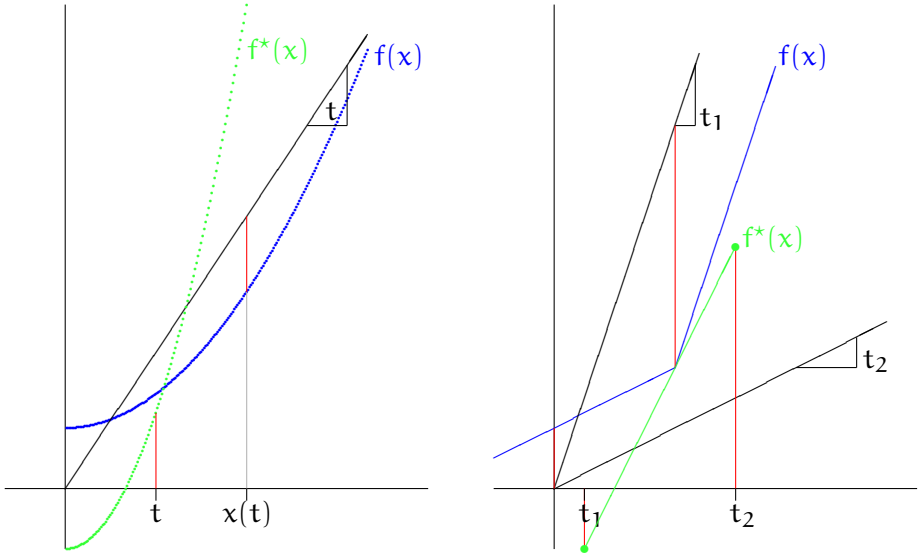

Abbildung 2.3: Veranschaulichung zur Legendretransformation im streng konvexen Fall (links) und im polygonalen Fall (rechts).

2.27 Satz. *Für eine stetige, stückweise lineare, konvexe Funktion* $f : \mathbf{R} \to \mathbf{R}$ *gegeben durch*

$$f(x) = \begin{cases} y_n + s_n x & x < x_n \\ y_{n-1} + s_{n-1} x & x \in [x_n, x_{n-1}] \\ \vdots & \vdots \\ y_1 + s_1 x & x \in [x_2, x_1] \\ y_0 + s_0 x & x > x_1 \end{cases}$$

mit $s_n < s_{n-1} < \ldots < s_1 < s_0$ *und* $x_n < x_{n-1} < \ldots < x_1$ *gelten:*

(i) $f^\star$ *ist auf* $[s_n, s_0]$ *wieder stetig, stückweise linear und konvex.*

(ii) Explizit gilt:

$$f^\star(t) = \begin{cases} \infty & t < s_n \\ t\,x_n - (s_n x_n + y_n) & t \in [s_n, s_{n-1}] \\ \vdots & \vdots \\ t\,x_1 - (s_1 x_1 + y_1) & t \in [s_1, s_0] \\ \infty & t > s_0 \end{cases}$$

Beweis. Sei zunächst $t > s_0$, dann ist

$$f^\star(t) = \max_x\{t\,x - f(x)\} = \max_x\{t\,x - y_0 - s_0\,x\} \overset{x\to\infty}{\to} \infty,$$

analog für $t < s_n$.

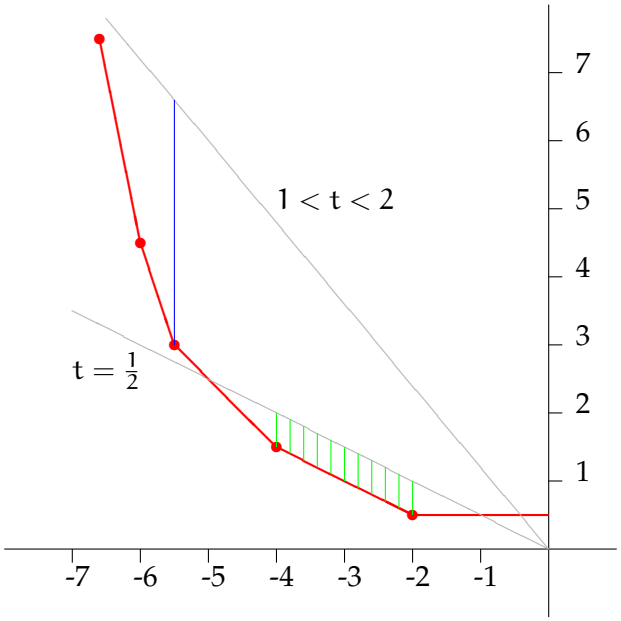

Abbildung 2.4: Veranschaulichung zum Beweis von Satz 2.27

Für $t \in \{s_0, \ldots, s_n\}$, etwa $t = s_i$ wird $\max_x\{t\,x - f(x)\}$ offensichtlich im Intervall $[x_{i+1}, x_i]$ angenommen, da $f(x)$ für $x > x_i$ langsamer, für $x < x_{i+1}$ schneller als $t\,x$ wächst, deswegen ist $f^\star(s_i) = s_i\,x_i - y_{i-1} - s_i\,x_i = -y_{i-1}$. Für $t \in (s_{i+1}, s_i)$ wird $\max_x\{t\,x - f(x)\}$ am linken Rand der Kante mit der Steigung s_i angenommen. Demnach ist dann

$$f^\star(t) = \max_x\{t\,x - f(x)\} = t\,x_{i+1} - (s_{i+1}\,x_{i+1} + y_{i+1}).$$

□

2.4.2 Das Newtonpolygon

Wir betrachen hier wieder $L := \sum_{i=0}^{n} a_i \sigma^i \in \mathbf{F}[x][\sigma]$.

Newtonpolygon $\mathcal{N}_v(L)$, $\mathcal{P}_L$

2.28 Definition. Definiere nun *das Newtonpolygon von* L:

$$\mathcal{N}_v(L) := \text{Konvexe Hülle von } \bigcup_{i=0}^{n} \{(i, y) \mid y \geq v(a_i)\} \subseteq \mathbf{R}^2.$$

Definiere weiterhin – analog zum Bewertungspolygon – die Funktion $\mathcal{P}_{v,L} : \mathbf{R} \to \mathbf{R} \cup \{\infty\}$ als die stetige, stückweise lineare Funktion, die den unteren Rand des Newtonpolygons definiert.

2.29 Satz. *Es gilt für alle* $r \in [0, n]$

$$\mathcal{P}_{v,L}(r) = (-\mathcal{V}_{v,L})^\star(-r).$$

Beweis. Wir erinnern an Definition 2.18:

$$\nu_r(L) := \min\{\nu(a_i) + r \cdot i \mid i = 0, \dots, n\},$$

seien $I := \{i_0, \dots, i_m\}$ die Indizes, für die $\nu(a_{i_j}) + r\,i_j$ zu einem Segment des Bewertungspolygons beiträgt, derart sortiert dass $i_0 < \dots < i_m$. Weiter seien x_j, für $j = 1, \dots, m$ die Schnittpunkte von $\nu(a_{i_j}) + r\,i_j$ mit $\nu(a_{i_{j-1}}) + r\,i_{j-1}$.
Dann ist

$$-\mathcal{V}_{\nu,L}(x) = \begin{cases} -\nu(a_{i_m}) - i_m x & x < x_m \\ -\nu(a_{i_{m-1}}) - i_{m-1} x & x \in [x_m, x_{m-1}] \\ \vdots & \vdots \\ -\nu(a_{i_1}) - i_1 x & x \in [x_2, x_1] \\ -\nu(a_{i_0}) - i_0 & x > x_1 \end{cases}$$

und demnach ist

$$(-\mathcal{V}_{\nu,L})^\star(t) = \begin{cases} \infty & t < -i_m \\ t\,x_m + i_m x_m + \nu(a_{i_m}) & t \in [-i_m, -i_{m-1}] \\ \vdots & \vdots \\ t\,x_1 + i_1 x_1 + \nu(a_{i_1}) & t \in [-i_1, -i_0] \\ \infty & t > -i_0 \end{cases}$$

Für $t = -i_j$ gilt dann $(-\mathcal{V}_{\nu,L})^\star(t) = \nu(a_{i_j})$. □

2.30 Korollar. *Ist $r \in \mathbf{R}$ eine exzeptionelle Stelle des Bewertungspolygons $\mathcal{V}_{\nu,L}$, so hat das Newtonpolygon $\mathcal{N}_\nu(L)$ eine Kante der Steigung $-r$.*

Beweis. Folgt aus den Sätzen 2.27 und 2.29. □

2.31 Beispiel. Sei

$$L := (4x + 1)\sigma^5 + (4x^3 + 2x - 2)\sigma^3 + (x^4 + 2)\sigma + 2x^2 + 1,$$

und benutze die Bewertung

$$\nu_{\deg}(a) = \begin{cases} -\deg a & a \neq 0 \\ \infty & a = 0 \end{cases}$$

dann zeigt Abbildung 2.5 das Newtonpolygon $\mathcal{N}_{\nu_{\deg}}(L)$.

2.32 Bemerkung. Wenn wir von L zum adjungierten Operator L^* übergehen (das ist nicht die Legendretransformierte, siehe Definition 5.1), so ändert sich das Newtonpolygon nicht.

2.33 Bemerkung. Das σ-Newtonpolygon hat zwei unendlich lange vertikale Kanten und endlich viele Kanten mit rationalen Steigungen. Wenn wir von der "Kante r", mit $r \in \mathbf{Q}$, reden, so ist die Kante mit der Steigung r gemeint.

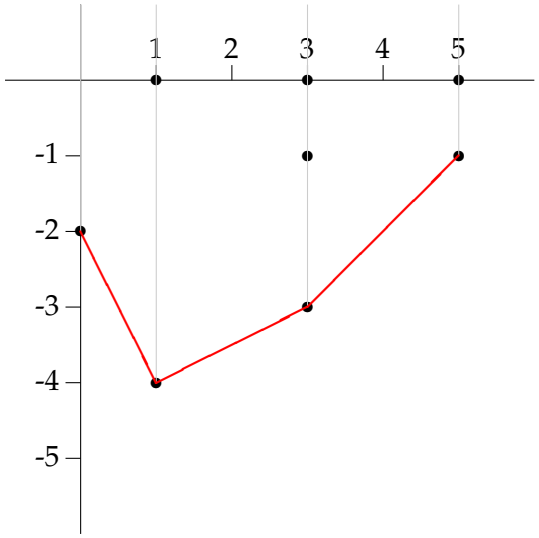

Abbildung 2.5: Newtonpolygon zu Beispiel 2.31

L_p

2.34 Definition. Sei nun $L := \sum_{i=0}^{n} a_i\sigma^i = \sum_{i,j} a_{i,j}\, x^j\, \sigma^i \in \mathbf{F}[x;\sigma]$ und $r \in \mathbf{R}$. In Definition 2.18 haben wir die Steigungs-Bewertung $\nu_r(L)$ eingeführt, sei nun

$$L_r := \sum_{\substack{i,j \\ \nu_r(a_{i,j}\, x^j\, \sigma^i)=\nu_r(L)}} a_{i,j}\, x^j\sigma^i$$

die Summe der Monome, die genau die Bewertung $\nu_r(L)$ haben.

2.35 Bemerkung. Die Monome $a_{i,j}x^j\sigma^i$ von L_r sind genau die, für die $(i, \nu(a_{i,j}x^j))$ auf der Kante r liegt. Gibt es keine Kante r, so ist $L_r = 0$.

2.36 Beispiel. Sei

$$L = \sigma^4 + (x^2-1)\sigma^3 - (4x^3+x)\sigma^2 + (x^4+12)\sigma - (x^2+2),$$

und benutze $\nu = -\deg$, dann ist

$$\begin{aligned}\nu_{(-1)}(L) &= \min\{\nu(a_i) - 1\cdot i \mid i = 0,\dots,4\} \\ &= \min\{-2, -4-1, -3-2, -2-3, 0-4\} = -5\end{aligned}$$

und so ist

$$L_{(-1)} = x^2\sigma^3 - 4x^3\sigma^2 + x^4\sigma.$$

Die zugehörigen Bewertungs- und Newtonpolygone sind in Abbildung 2.6 zu sehen. Beachte, dass die Kante 1 des Newtonpolygons die Länge 2 hat, und sich entsprechend an der exzeptionellen Stelle (-1) des Bewertungspolygons drei Geraden schneiden.

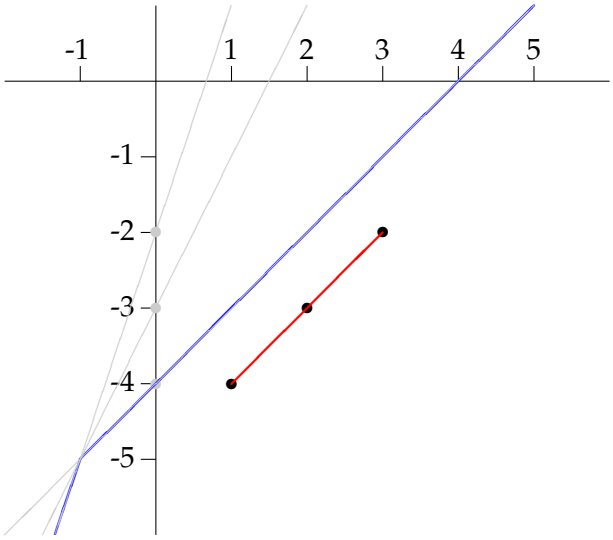

Abbildung 2.6: Newtonpolygon (rot) und Bewertungspolygon (blau, Konstruktion grau) zu $L_{(-1)}$ aus Beispiel 2.36

2.37 Definition. Das *charakteristische Polynom* assoziiert zur Kante $r \in \mathbf{Q}$ von $N_\sigma(L)$ ist für $\sigma = \tau$

Charakteristisches Polynom $P^\sigma_{v,r}(T)$

$$P^\tau_{v,r}(T) = \sum_{\substack{i,j \\ v_r(a_{i,j}x^j\tau^i)=v_r(L)}} a_{i,j}\,T^{i-i_0},$$

bzw. $\sigma = \epsilon$

$$P^\epsilon_{v,r}(T) = \sum_{\substack{i,j \\ v_r(a_{i,j}x^j e^i)=v_r(L)}} a_{i,j}\cdot q^{\frac{1}{2}(i^2-i)r\cdot v(x)}\cdot T^{i-i_0},$$

dabei ist i_0 so gewählt, dass $T \nmid P_r(T)$.

Dies ist ein Polynom in T, der Grad entspricht der „Länge“ der Kante r. Die Gleichung $P^\sigma_{v,r}(T) = 0$ heißt die *charakteristische Gleichung* assoziiert zur Kante r.

Charakteristische Gleichung

2.38 Bemerkung. Die Lösungen der charakteristischen Gleichung $P_{v_{\deg},r}(T) = 0$ liefern gerade die Lösungen der Rekursionsgleichung $L_r \bmod (x-1)$.

2.39 Beispiel (Fortsetzung von Beispiel 2.36). Die charakteristische Gleichung zur Kante 1 von $\mathcal{N}_\sigma(L)$ ist

$$P_1(T) = T^2 - 4T + 1 = (T - (2+\sqrt{3}))\cdot(T-(2-\sqrt{3})).$$

2.40 Satz. *Sei u eine hypergeometrische Lösung von L mit $\frac{\sigma(u(x))}{u(x)} = c\,w = c\,\frac{f}{g}$ rational. Dabei seien $c \in \mathbf{F}, w \in \mathbf{F}(x)$ und $f, g \in \mathbf{F}[x]$ normiert bezüglich[1] ν mit $\gcd(f,g) = 1$ gewählt.*
Dann ist c eine Nullstelle der charakteristischen Gleichung $P^{\sigma}_{\nu,\nu(g)-\nu(f)}$.

Beweis. Sei also u eine Lösung von $L(u) = 0$ mit $\frac{\sigma(u)}{u} = c\,w = c\,\frac{f}{g}$ wie oben. Dann ist $\sigma(u) = c\,u\,w$ und induktiv

$$\sigma^k(u) = c^k\,u\,w\,\sigma(w)\cdots\sigma^{k-1}(w).$$

Damit erhalten wir

$$0 = L(u) = \sum_{i=0}^{n} a_i\sigma^i(u) = \sum_{i=0}^{n} a_i\,c^i \prod_{j=0}^{i-1}\sigma^j(w). \tag{2.2}$$

Betrachten wir nun das Polynom

$$P(T) := \sum_{i=0}^{n} \underbrace{\left(a_i \prod_{j=0}^{i-1}\sigma^j(f)\cdot\prod_{j=i}^{n-1}\sigma^j(g)\right)}_{=:p_i} T^i = \sum_{i=0}^{n} p_i\,T^i \in \mathbf{F}[x][T], \tag{2.3}$$

dann ist $P(a) \in \mathbf{F}[x]$ für $a \in \mathbf{F}$, und c ist eine Nullstelle von P.
Betrachte nun den x-Leitkoeffizienten bezüglich ν in jedem Koeffizienten von T und setze $s_f := \nu(f)$, $s_g := \nu(g)$ und $r := s_f - s_g$. Dann ist

$$\begin{aligned}\nu(p_i) &= \nu\left(a_i \prod_{j=0}^{i-1}\sigma^j(f)\cdot\prod_{j=i}^{n-1}\sigma^j(g)\right)\\ &= \nu(a_i) + \sum_{j=0}^{i-1}\nu(f) + \sum_{j=i}^{n-1}\nu(g)\\ &= \nu(a_i) + i\,s_f + (n-i)s_g = \nu(a_i) + i\,r + n\,s_g.\end{aligned}$$

Wir suchen nun die Menge I_r der i, für die $\nu(p_i)$ minimal wird. Den bezüglich i konstanten Summanden $n\,s_g$ können wir dabei ignorieren.

Behauptung: I_r ist die Menge der Indizes i, für die $\mathrm{lt}_\nu(a_i)\sigma^i$ auf der Kante der Steigung $(-r)$ des zugehörigen Newtonpolygons liegen.
Seien nun $i_0 < i_1$ die Indizes des ersten bzw. letzten a_i auf der Kante $(-r)$. Demnach gilt für jedes $i < i_0$

$$\frac{\nu(a_{i_0}) - \nu(a_i)}{i_0 - i} < (-r) \overset{i_0-i>0}{\Longrightarrow} \nu(a_{i_0}) + r\,i_0 < \nu(a_i) + r\,i,$$

[1] Damit ist gemeint, dass für $\nu = \nu_{\deg}$ gilt $\mathrm{lc}(f) = \mathrm{lc}(g) = 1$ und für $\nu = \nu_{\mathrm{tdeg}}$ gilt $\mathrm{tcoeff}(f) = \mathrm{tcoeff}(g) = 1$.

und für $i > i_1$ gilt

$$\frac{\nu(a_{i_1}) - \nu(a_i)}{i_1 - i} > (-r) \overset{i_1 - i < 0}{\Longrightarrow} \nu(a_{i_1}) + r\,i_1 < \nu(a_i) + r\,i.$$

Diese Argumentation gilt auch für die $i_0 < i < i_1$, für die $\mathrm{lt}_\nu(a_i)$ nicht auf der Kante der Steigung $(-r)$ liegt. Das beweist die Behauptung.

Nun müssen wir den Shift- und den q-Shift-Fall unterscheiden. Betrachte zunächst den Shift-Fall:
Wir haben jetzt

$$P^\tau_{\nu,(-r)}(T) = \sum_{i \in I_r} \mathrm{lc}_\nu(a_i) T^i.$$

Dies ist genau der Teil von P, der den x-Leitkoeffizienten bezüglich ν beisteuert. Demnach muss also $P^\tau_{\nu,(-r)}(c) = 0$ gelten.

Im q-Fall ist die Situation ein klein wenig problematischer, da ein normiertes, nichtkonstantes Polynom unter ϵ nicht normiert bleibt:

$$\mathrm{lc}_\nu(\epsilon(x^k)) = \mathrm{lc}_\nu(q^k x^k) = \left.\begin{cases} q^{-\nu(x^k)} & \text{für } \nu = \nu_{\deg} \\ q^{\nu(x^k)} & \text{für } \nu = \nu_{\mathrm{tdeg}} \end{cases}\right\} = q^{\nu(x)\cdot\nu(x^k)}.$$

Der x-Leitkoeffizient bezüglich ν von p_i ist also i. A. nicht identisch mit dem Leitkoeffizienten von a_i bezüglich ν. Tatsächlich gilt

$$\begin{aligned}
\mathrm{lc}_\nu(p_i) &= \mathrm{lc}_\nu(a_i) \cdot \prod_{j=0}^{i-1} q^{j\,s_f\,\nu(x)} \cdot \prod_{j=i}^{n-1} q^{j\,s_g\,\nu(x)} \\
&= \mathrm{lc}_\nu(a_i) \cdot q^{\frac{1}{2}\left((i^2-i)\,s_f + (n^2-n-i^2+i)\nu(g)\right)\nu(x)} \\
&= \mathrm{lc}_\nu(a_i) \cdot q^{\frac{1}{2}\left((i^2-i)\,s_f + (n^2-n)s_g - (i^2-i)s_g\right)\nu(x)} \\
&= \mathrm{lc}_\nu(a_i) \cdot q^{\frac{1}{2}(i^2-i)\,r\,\nu(x)} \cdot q^{\frac{1}{2}(n^2-n)s_g\,\nu(x)}.
\end{aligned}$$

Der Faktor $q^{\frac{1}{2}(n^2-n)s_g\,\nu(x)}$ hängt dabei nicht von i ab, kann also also als Konstante abdividiert werden. Definiere $c_i := \mathrm{lc}_\nu(a_i) \cdot q^{\frac{1}{2}(i^2-i)\,r\cdot\nu(x)}$. Beachte dabei, dass für $i \in \mathbf{N}$ und $r \in \mathbf{Z}$ auch $\frac{1}{2}(i^2 - i)\,r \cdot \nu(x) \in \mathbf{Z}$ gilt.

Damit ist also auch im q-Fall

$$P^\epsilon_{\nu,(-r)}(T) = \sum_{i \in I_r} c_i T^i,$$

und dies ist genau der Teil von P, der den x-Leitkoeffizienten beisteuert.
Demnach muss also $P^\epsilon_{\nu,(-r)}(c) = 0$ gelten. □

Algorithmus 4: `NewtonPolygon(L, ν)`

Input: $0 \neq L = \sum_{i=0}^{n} a_i \sigma^i \in \mathbf{F}(x)[\sigma]$, ν Bewertung
Output: Eine Menge von Paaren $(r, P^{\sigma}_{\nu,r}(T))$, r Steigung des Newtonpolygons

```
1  i ← 0,   n ← order(L),   S ← ∅
2  while i < n
3  |  if a_i = 0 then next i
4  |  s ← ∞
5  |  for j = i + 1, ..., n
6  |  |  if a_j = 0 then next j
7  |  |  t ← (ν(a_j) − ν(a_i)) / (j − i)
8  |  |  if t < s
9  |  |  |  s ← t
10 |  |  |  pol ← Koeffizient von a_i mit Bewertung ν(a_i)
11 |  |  if t = s
12 |  |  |  pol ← pol + (Koeffizient von a_j mit Bewertung ν(a_j)) · T^(j−i)
13 |  |  |  c ← j
14 |  S ← S ∪ {(s, pol)}
15 |  i ← c
16 return S
```

Die Implementierung im q-Fall unterscheidet sich nur durch den Faktor $q^{\frac{1}{2}(i^2-i)r\cdot\nu(x)}$ (siehe Definition 2.37), deswegen verzichten wir darauf, sie hier extra aufzuführen.

2.5 Implementierung

2.5.1 Dom::ShiftOperators

MuPAD-Session 4: Newtonpolygon, Shift-Fall

Zunächst betrachten wir die Newtonpolygone zu zwei relativ einfachen Operatoren:

```
>> L1 := R((7*x^5+3*x^2-2)*S^2+(x^7+3*x^2+x)*S+6*x^2-8*x-12)
```

$$\left(7x^5+3x^2-2\right)S^2+\left(x^7+3x^2+x\right)S+6x^2-8x-12$$

```
>> R::polygon(L1)
```

$$\{[-5, \text{poly}(x+6,[x])], [2, \text{poly}(7x+1,[x])]\}$$

```
>> L2 := R(x*S+(3*x^2+x-1))
```

$$xS+3x^2+x-1$$

```
>> R::polygon(L2)
```

$$\{[1, \text{poly}(x+3,[x])]\}$$

Nun berechnen wir das Produkt...

≫ L := L1*L2

$$\left(7\,x^6+14\,x^5+3\,x^3+6\,x^2-2\,x-4\right)\ S^3+$$

$$\left(x^8+22\,x^7+91\,x^6+91\,x^5+9\,x^4+42\,x^3+37\,x^2-25\,x-26\right)\ S^2+$$

$$\left(3\,x^9+7\,x^8+3\,x^7+9\,x^4+30\,x^3+8\,x^2-9\,x\right)\ S+$$

$$18\,x^4-18\,x^3-50\,x^2-4\,x+12$$

... und stellen fest, dass die Steigungen aus beiden Polygonen im Produktpolygon vorkommen:

≫ R::polygon(L)

$$\{[-5,\mathrm{poly}\,(3\,x+18,[x])]\,,[1,\mathrm{poly}\,(x+3,[x])]\,,[2,\mathrm{poly}\,(7\,x+1,[x])]\}$$

Auch wenn die Multiplikation der Operatoren nicht kommutativ ist...

≫ LL := L2*L1

$$\left(7\,x^6+35\,x^5+70\,x^4+73\,x^3+41\,x^2+8\,x\right)\ S^3+$$

$$\left(x^8+28\,x^7+28\,x^6+28\,x^5+44\,x^4+27\,x^3+5\,x^2+3\,x+2\right)\ S^2+$$

$$\left(3\,x^9+x^8-x^7+9\,x^4+12\,x^3+2\,x^2-15\,x\right)\ S+$$

$$18\,x^4-18\,x^3-50\,x^2-4\,x+12$$

... sind Satz 2.20 und 2.29 natürlich in beiden Reihenfolgen richtig:

≫ R::polygon(LL)

$$\{[-5,\mathrm{poly}\,(3\,x+18,[x])]\,,[1,\mathrm{poly}\,(x+3,[x])]\,,[2,\mathrm{poly}\,(7\,x+1,[x])]\}$$

2.5.2 Dom::qShiftOperators

MuPAD-Session 5: Newtonpolygon, q-Shift-Fall

Zunächst betrachten wir die Newtonpolygone zu zwei relativ einfachen Operatoren. Standardmäßig wird die Bewertung $-\deg$ *benutzt:*

≫ L1 := qR(q*Eq^2+3*q*x*Eq+1)

$$q\,Eq^2+((3\,q)\ x)\ Eq+1$$

≫ qR::polygon(L1)

$$\{[-1,\mathrm{poly}\,((3\,q)\ x+1,[x])]\,,[1,\mathrm{poly}\,(q\,x+(3\,q)\,,[x])]\}$$

≫ L2 := qR((3*x+q)*Eq+(q^2+1)*x+1)

$$(3\,x+q)\ Eq+\left(q^2+1\right)\ x+1$$

≫ qR::polygon(L2)

$$\left\{\left[0, \text{poly}\left(3\,x + \left(q^2+1\right), [x]\right)\right]\right\}$$

Man sieht, dass sich die Polygone addieren – wie Satz 2.20 und 2.29 postulieren:

≫ L := L1 * L2

$$\left(\left(3\,q^3\right)\,x + q^2\right)\,Eq^3 + \left(\left(9\,q^2\right)\,x^2 + \left(q^5+q^3+3\,q^2\right)\,x + q\right)\,Eq^2 +$$
$$\left(\left(3\,q^4+3\,q^2\right)\,x^2 + (3\,q+3)\,x + q\right)\,Eq + \left(q^2+1\right)\,x + 1$$

≫ qR::polygon(L)

$$\left\{\left[-1, \text{poly}\left(\left(3\,q^4+3\,q^2\right)\,x + \left(q^2+1\right), [x]\right)\right],\right.$$
$$\left[0, \text{poly}\left(\left(9\,q^2\right)\,x + \left(3\,q^4+3\,q^2\right), [x]\right)\right],$$
$$\left.\left[1, \text{poly}\left(\left(3\,q^3\right)\,x + \left(9\,q^2\right), [x]\right)\right]\right\}$$

Nun benutzen wir die x*-adische Bewertung* ldegree*:*

≫ qR::polygon(L1, ldegree)

$$\left\{\left[\frac{1}{2}, \text{poly}\,(1, [x])\right]\right\}$$

≫ qR::polygon(L2, ldegree)

$$\{[1, \text{poly}\,(3\,x+1, [x])]\}$$

≫ qR::polygon(L, ldegree)

$$\left\{\left[\frac{1}{2}, \text{poly}\left(\left(q^5+q^3+3\,q^2\right)\,x^2 + 1, [x]\right)\right],\right.$$
$$\left.\left[1, \text{poly}\left(\left(q^5+q^3+3\,q^2\right), [x]\right)\right]\right\}$$

Kapitel 3

Polynomielle und rationale Lösungen von Schiefpolynomen

In diesem Kapitel werden wir sehen, wie effizient polynomielle und rationale Lösungen von Operatoren berechnet werden können.

Auch wenn die Lösungen eines Operators nicht das eigentliche Thema dieser Arbeit sind, sind doch – wie bei Polynomen – Lösungen und Rechtsfaktoren eng verwandt, gilt doch $(\sigma - \frac{\sigma(u)}{u})(u) = 0$.

Die Algorithmen in diesem Abschnitt basieren im Wesentlichen auf [1], [63] und [3], eine Zusammenstellung findet sich in [84].

Bei der Darstellung der Nennerschranke benutzen wir die vorher eingeführte σ-Äquivalenz, die eine einfachere Darstellung erlaubt.

3.1 Polynomielle Lösungen

Sei $L = \sum_{i=0}^{n} a_i \sigma^i \in \mathcal{S}$. Ohne Einschränkung können wir annehmen, dass die Koeffizienten $a_i \in \mathbf{F}[x]$ sind.
Ist $N \in \mathbf{N}$ eine Schranke für den Grad einer polynomiellen Lösung $u(x)$ von L, so können wir einen allgemeinen Ansatz machen:

$$u(x) = \alpha_N x^N + \ldots + \alpha_1 x + \alpha_0,$$

mit unbestimmten $\alpha_i \in \mathbf{F}$, für $0 \leq i \leq N$. Setzen wir dieses $u(x)$ nun in L ein, so erhalten wir

$$L(u) = \sum_{i=0}^{N} a_i \sigma^i(u(x)) = \sum_{i=0}^{n} a_i \sum_{j=0}^{N} \alpha_j \sigma^i(x^j), \quad \text{wobei } \sigma^i(u(x)) \in k[x].$$

Nach Ausmultiplizieren und Sortieren nach x-Potenzen bekommen wir

$$L(u) = \sum_{i=0}^{N+\deg_x(L)} \lambda_i x^i \in \mathbf{F}[x],$$

wobei die λ_i selbst linear in $\alpha_0, \ldots, \alpha_N$ sind.

Die Forderung $L(u) = 0$ liefert dann mit Koeffizientenvergleich ein lineares Gleichungssystem für die α_i:

$$\lambda_i = 0, \quad \text{für} \quad 0 \le i \le \deg_x(L) + N.$$

Algorithmus 5: PolynomialSolutions(L)

Input: $L = \sum_{i=0}^{n} a_i \sigma^i \in \mathbf{F}[x; \sigma]$
Output: Basis aller polynomiellen Lösungen von L

```
1 N ← PolyDegreeBound(L)
2 c ← Σ_{i=0}^{N} c_i x^i
3 c' ← L(c)
4 EQ ← {coeff(c', x^i) = 0 | i = 0, ..., deg_x(c')}
5 Sol ← Lösungen des LGS EQ für c_0, ..., c_N
6 return {subs(c, s) | s ∈ Sol}
```

3.1.1 Gradschranke

Für die Berechnung der Gradschranke im Shift-Fall folgen wir [63]. Sei $L = \sum_{i=0}^{n} a_i \tau^i$ mit $a_i = \sum_{j=0}^{m} a_{i,j} x^j$, wobei mindestens eines der a_i tatsächlich Grad m hat. Wir benutzen nun den Ansatz $u = \sum_{l=0}^{N} u_l x^l$. Dann ist

$$\tau^i(u) = \sum_{l=0}^{N} u_l \sum_{k=0}^{l} \binom{l}{k} x^k i^{l-k} = \sum_{k=0}^{N} \Bigg(\underbrace{\sum_{l=k}^{N} \binom{l}{k} u_l i^{l-k}}_{=:\tilde{u}_{i,k}} \Bigg) x^k, \tag{3.1}$$

und damit

$$\begin{aligned} L(u) &= \sum_{i=0}^{n} a_i \sum_{k=0}^{N} \tilde{u}_{i,k} x^k = \sum_{i=0}^{n} \sum_{j=0}^{d_i} a_{i,j} x^j \sum_{k=0}^{N} \tilde{u}_{i,k} x^k \\ &= \sum_{j=0}^{m} \sum_{k=0}^{N} x^{j+k} \sum_{i=0}^{n} (a_{i,j} \tilde{u}_{i,k}) \end{aligned}$$

die Koeffizienten von $L(u)$ als Polynom in x sind Polynome in N. Setzten wir nun (3.1) in $L(u) = 0$ ein und sortieren nach Potenzen von x, so erhalten wir nach Koeffizientenvergleich zunächst $\sum_{i=0}^{n} a_{i,m} = 0$, anderenfalls kann es keine polynomiellen Lösungen geben. Der nächst niedrigere Koeffizient

von $L(u)$ ist nach Vereinfachen

$$c_d = \sum_{j=0}^{d} \binom{N}{j} \sum_{i=0}^{n} a_{i,j}\, i^j.$$

mit $d = 1$. Falls c_d für $d = 1$ identisch verschwindet, haben wir keine Informationen gewonnen und müssen d um eins erhöhen. Ist $c_d \neq 0$, so ist die größte nichtnegative ganzzahlige Nullstelle von c_d eine obere Schranke für den Grad einer polynomiellen Lösung.
Das führt zu folgendem Algorithmus:

Algorithmus 6: `PolyDegreeBound(L)`

Input: $L = \sum_{i=0}^{n} a_i \sigma^i \in \mathbf{F}[x;\sigma]$, $a_i = \sum_{j=0}^{m} a_{i,j}\, x^j$
Output: $N \in \mathbf{N}$, so dass jede polynomielle Lösung von L Grad $\leq N$ hat, oder (-1)

```
1  s ← 0, d ← 0
2  while s = 0 do
3  |  d ← d + 1
4  |  s ← Σ_{j=0}^{d} (N choose j) Σ_{i=0}^{n} a_{i,j} i^j
5  S ← N ∩ solve(0 = s, N)
6  if ∅ = S then return (−1)
7  return max S
```

Für mehr Details siehe [63] oder [52].

Im q-Shift-Fall ist die Gradschranke mit den Mitteln aus Kapitel 2, genauer mit Satz 2.40 sehr einfach zu berechnen:

3.1 Satz. *Sei $P_0^\epsilon(T)$ ist das charakteristische Polynom, wie es in Definition 2.37 definiert wurde. Dann ist*

$$N := \max\{n \in \mathbf{N} \mid P_0^\epsilon(q^n) = 0\}$$

eine obere Schranke für den Grad einer polynomiellen Lösung von L. Die benutzte Bewertung ist hier $\nu = \nu_{\deg}$.

Beweis. Ist $f = \sum_{i=0}^{d} f_i\, x^i$ eine polynomielle Lösunug von L, dann ist

$$\operatorname{cert}_\epsilon(f) = \frac{q^d f_d\, x^d + \dots}{f_d\, x^d + \dots} = q^d\, \frac{x^d + \frac{f_{d-1}}{q\, f_d}\, x^{d-1} + \dots}{x^d + \frac{f_{d-1}}{f_d} x^{d-1} + \dots},$$

und damit muss q^d nach Satz 2.40 eine Nullstelle von $P_0^\epsilon(T)$ sein. □

In [5] wird ein effizienterer Algorithmus vorgestellt, der auch eine niedrigere Gradschranke liefert.

3.2 σ-Äquivalenz

Im Folgenden wird es häufig sinnvoll und notwendig sein, auftretende Polynome (bzw. deren Nullstellen) nur "modulo des Endomorpismus" zu betrachten. Das wurde in vielen Arbeiten etwa zur symbolischen Summation etwa in Form der Dispersion (vgl. Definition 3.22) benutzt.
Die Bezeichnung σ-Äquivalenz findet sich explizit z. B. in [66] oder [30], wir führen sie hier allgemeiner aus.

σ-Äquivalenz

3.2 Definition. Zwei Polynome $f, g \in \mathbf{F}[x]$ heißen σ-*äquivalent* $f \sim_\sigma g$, wenn es ein $k \in \mathbf{Z}$ gibt mit $f(x) = \sigma^k(g(x))$.
Wir schreiben dann auch $f \sim_\sigma g$.

Ist $\sigma = \tau : x \mapsto x + 1$, so ist es einfach zu entscheiden, ob zwei Polynome σ-äquivalent sind:

Algorithmus 7: ShiftEquivalence(f, g)

Input: $f, g \in \mathbf{F}[x]$
Output: $k \in \mathbf{Z}$ so dass $f(x) = g(x + k)$ falls so ein k existiert, oder fail

```
1  if lc(f) ≠ lc(g) or deg(f) ≠ deg(g) then return fail
2  n ← deg(f)
3  k ← (coeff(f, n−1) − coeff(g, n−1)) / (n ∗ lc(f))
4  if k ∉ Z then return fail
5  if f(x) ≠ f(x + k) then return fail
6  return k
```

Für den q-Shift ϵ ergibt sich ein ähnlicher Algorithmus.

$[p]_\sigma$

3.3 Satz und Definition. σ-Äquivalenz ist eine Äquivalenzrelation. Die σ-Äquivalenzklasse eines Polynoms $p \in \mathbf{F}[x]$ bezeichnen wir mit $[p]_\sigma$.

$$[p]_\sigma := \{\sigma^j(p(x)) \mid j \in \mathbf{Z}\} = \{f \in \mathbf{F}[x] \mid f \sim_\sigma p\}.$$

$\log_p^\sigma(f)$

Ist $p \sim_\sigma f$, so definieren wir

$$\log_p^\sigma(f) := \text{das eindeutig bestimmte } k \in \mathbf{Z} \text{ mit } \sigma^k(p) = f.$$

3.4 Beispiel.

(i) Betrachte $\mathcal{S}_\tau$. Setze $p = x^2 + 1$. Dann ist

$$[p]_\tau = \{(x + k)^2 + 1 \mid k \in \mathbf{Z}\}.$$

Für $f = x^2 + 4x + 5$ ist $f \sim_\tau p$ und $\log_p^\tau(f) = 2$ bzw. $\log_f^\tau(p) = -2$.

(ii) Betrachte nun $\mathcal{S}_\epsilon$ und seien $p = (2+q^2)x^2 - (q^2-2+3q)x - q + 1$ und $f = (\frac{2}{q^2} + 1)x^2 - (q - \frac{2}{q} + 3)x - q + 1$. Dann ist $f \sim_\epsilon p$ und $\log_p^\epsilon(f) = -1$ bzw. $\log_f^\epsilon(p) = 1$.

Um alle in einem Polynom auftretenden σ-Äquivalenzklassen zu finden, benutzen wir den folgenden Algorithmus:

Algorithmus 8: ShiftEquivalenceClasses(f)

Input: $f \in \mathbf{F}[x]$ normiert
Output: $\{p_1, \dots, p_r\} \subseteq \mathbf{F}[x]$ so dass jeder irreduzible Faktor von f σ-äquivalent zu genau einem p_i ist.

```
1  faktorisiere f = f_1^{e_1} ··· f_s^{e_s} in irreduzible Faktoren
2  F ← ∅
3  for i ← 1, ..., s
4  |  new ← true
5  |  for p ∈ F while new = true
6  |  |  if ShiftEquivalence(f_i, p) ≠ fail then
7  |  |  |  new ← false
8  |  |  |  if ShiftEquivalence(f_i, p) > 0
9  |  |  |  |  F ← (F \ {p}) ∪ {f_i}
10 |  if new = true then
11 |  |  F ← F ∪ {f_i}
12 return F
```

Innerhalb der σ-Äquivalenzklassen definieren wir eine Ordnungsrelation:

σ-Ordnung
$f \succcurlyeq_\sigma g$

3.5 Satz und Definition. Die Relation $\succcurlyeq_\sigma \subseteq M \times M$ mit $M \in \mathbf{F}[x]/\sim_\sigma$ definiert durch

$$f \succcurlyeq_\sigma g \;:\Longleftrightarrow\; (\exists k \in \mathbf{N}_0)[\sigma^k(g) = f]$$

ist

(i) reflexiv,

(ii) antisymmetrisch und

(iii) transitiv.

Damit ist $\succcurlyeq_\sigma$ eine Teilordnung. Wir nennen sie die σ-*Ordnung*. Tatsächlich ist $\succcurlyeq_\sigma$ sogar eine lineare Ordnung.
Für endliche Teilmengen $M' \subseteq M \in \mathbf{F}(x)/\sim_\sigma$ definieren wir damit auch

$$\max_\sigma(M') \quad \text{und} \quad \min_\sigma(M').$$

3.6 Bemerkung. Im Fall endlicher Grundkörper könnten die Klassen $[p]_\sigma$ endlich und damit zyklisch sein, dann ist $\succcurlyeq_\sigma$ natürlich keine lineare Teilordnung mehr.

3.7 Beispiel. (Fortsetzung von Beispiel 3.4)

(i) Betrachte wieder $\mathcal{S}_\tau$, es ist $f = x^2 + 4x + 5 \succ_\tau x^2 + 1$.

(ii) Betrachte nun $\mathcal{S}_\epsilon$, hier ist $(2 + q^2)x^2 - (q^2 - 2 + 3q)x - q + 1 \succ_\epsilon (\frac{2}{q^2} + 1)x^2 - (q - \frac{2}{q} + 3)x - q + 1$.

3.8 Bemerkung. Algorithmus 8 ShiftEquivalenceClasses liefert stets das σ-minimale vorkommende Element aus den σ-Äquivalenzklassen zurück. Dies wird in den Zeilen 8 f erreicht.
Damit wird der Algorithmus auch deterministisch.
Wir brauchen diese Eigenschaft später für Algorithmus 10 DenomBound.

Für Elemente von **F** führen wir noch folgende Definition ein:

σ-Kongruenz
$a \equiv_\sigma b$

3.9 Definition. Seien $a, b \in \mathbf{F}$. Wir sagen, a und b seien *kongruent unter* σ oder σ*-kongruent*, wenn $x - a \sim_\sigma x - b$ gilt. Dies bezeichnen wir mit $a \equiv_\sigma b$.

3.10 Beispiel.

(i) Unter τ über **C** sind die Kongruenzklassen gerade $\alpha + \mathbf{Z}$ mit $\alpha \in \mathbf{C}$.

(ii) Unter ϵ über **C** sind die Kongruenzklassen gerade $\alpha \cdot q^{\mathbf{Z}}$ mit $\alpha \in \mathbf{C}$.

Bewertung
$\nu_p(f)$, $\nu_{[p]_\sigma}(f)$

3.11 Definition. Für ein irreduzibles Polynom $p \in \mathbf{F}[x]$ und ein Polynom $f \in \mathbf{F}[x]$ definiere die *Bewertung von* f *an der Stelle* p als

$$\nu_p(f) := \max\{e \in \mathbf{N}_0 \mid p^e \text{ teilt } f\}.$$

Weiter definiere die Verallgemeinerung

$$\nu_{[p]_\sigma}(f) := \sum_{g \in [p]_\sigma} \nu_g(f) \;=\; \sum_{(s,e) \in \mathrm{Log}_p^\sigma(f)} e.$$

Für eine rationale Funktion $F = \frac{g}{h} \in \mathbf{F}(x)$ definiere

$$\nu_p(F) := \nu_p(g) - \nu_p(h) \quad \text{und} \quad \nu_{[p]_\sigma}(F) := \nu_{[p]_\sigma}(g) - \nu_{[p]_\sigma}(h).$$

3.3 Rationale Lösungen

In diesem Abschnitt betrachten wir stets einen Operator der Ordnung $n \geq 1$ mit polynomiellen Koeffizienten:

$$L = \sum_{i=0}^{n} a_i \sigma^i \in \mathcal{S}, \quad a_0, \ldots, a_n \in \mathbf{F}[x], \quad a_n \neq 0$$

und suchen eine rationale Lösung $u = \frac{v}{w}$ mit $v, w \in \mathbf{F}[x]$ mit $\mathrm{ggT}(v, w) = 1$.

Schranke

3.12 Definition. (i) Für ein Polynom p bezeichnen wir jedes Polynom f als *Schranke von* p, das von p geteilt wird.

$\mathcal{T}_f$

(ii) Für ein Polynom $f \in \mathbf{F}[x]$ bezeichne mit $\mathcal{T}_f$ die Menge der über **F** irreduziblen Teiler von f:

$$\mathcal{T}_f := \{\, g \in \mathbf{F}[x] \mid g \text{ teilt } f \text{ und } g \text{ ist irreduzibel und normiert} \,\}.$$

(iii) Für zwei Polynome $p, f \in \mathbf{F}[x]$ mit p irreduzibel, bezeichne mit $\mathcal{T}_f^p$ die zu p σ-äquivalenten Teiler von f: $\mathcal{T}_f^p$

$$\mathcal{T}_f^p := \{\, g \in \mathcal{T}_f \mid g \sim_\sigma p \,\}.$$

(iv) Für ein Polynom p bezeichne mit $\widehat{p} := \frac{1}{\operatorname{lc} p} p$ das normierte Polynom mit den gleichen Nullstellen. $\widehat{p} = \frac{1}{\operatorname{lc} p} p$

3.13 Beispiel. Betrachte $\mathcal{S}_\tau$. Sei

$$\begin{aligned} f &= x^8 - 3x^7 + 4x^5 + 2x^4 - 8x^3 + 4x^2 + 16x - 16 \\ &= (x-1)\cdot(x-2)^2 \cdot (x^2 + 2x + 2) \cdot (x^3 + 2). \end{aligned}$$

Dann sind

$$\begin{aligned} \mathcal{T}_f &= \{x-1,\ x-2,\ x^2+1,\ x^3+2\}, \\ \mathcal{T}_f^x &= \{x-1,\ x-2\}, \\ \mathcal{T}_f^{x^2+1} &= \{x^2 + 2x + 2\}. \end{aligned}$$

3.14 Bemerkung. Um die rationalen Lösungen eines Operators zu berechnen, ist es hinreichend, eine Schranke für den Nenner der rationalen Lösung anzugeben. Im Interesse der Effizienz suchen wir eine Schranke möglichst geringen Grades.

3.15 Lemma. *Sei $u = \frac{v}{w}$ eine rationale Lösung von* L. *Ist $p \in \mathcal{T}_w$ ein irreduzibler Teiler von w mit Vielfachheit μ, so gelten:*

(i) $\{\widehat{\sigma(p)}, \ldots, \widehat{\sigma^n(p)}\} \cap \mathcal{T}_w = \emptyset \implies p^\mu \mid a_0$,

(ii) $\{\widehat{\sigma^{-1}(p)}, \ldots, \widehat{\sigma^{-n}(p)}\} \cap \mathcal{T}_w = \emptyset \implies p^\mu \mid \sigma^{-n}(a_n)$,

(iii) $\{\widehat{\sigma(p)}, \ldots, \widehat{\sigma^n(p)}, \widehat{\sigma^{-1}(p)}, \ldots, \widehat{\sigma^{-n}(p)}\} \cap \mathcal{T}_w = \emptyset$
$\implies p^\mu \mid \gcd(a_0, \sigma^{-n}(a_n))$.

Beweis. Wegen $a_n\sigma^n(u) + a_{n-1}\sigma^{n-1}(u) + \ldots + a_1\sigma(u) + a_0 \cdot u = 0$ gelten

$$a_0 \cdot u = -\Big(a_n\sigma^n(u) + a_{n-1}\sigma^{n-1}(u) + \ldots + a_1\sigma(u)\Big) \quad \text{und}$$

$$\sigma^{-n}(a_n) \cdot u = -\Big(\sigma^{-n}(a_{n-1})\sigma^{-1}(u) + \ldots + \sigma^{-n}(a_1)\sigma^{-n+1}(u) + \sigma^{-n}(a_0 \cdot u)\Big).$$

Nach Voraussetzung haben die rechten Seiten jeweils *keine* Teiler in $[p]_\sigma$, aber u hat den Pol p^μ, demnach müssen $p^\mu \mid a_0$ und $p^\mu \mid \sigma^{-n}(a_n)$ gelten. Teil (iii) folgt aus (i) und (ii). □

Das motiviert die folgende Definition:

3.16 Definition. Sei $L = \sum_{i=0}^n a_i\sigma^i$ wie bisher. Die Bilder der Nullstellen von a_0 und a_n unter σ-Potenzen heißen *Problemstellen* von L. Problemstellen

3.17 Bemerkung. Gilt $p = \widehat{\sigma(p)}$, so gewinnen wir durch Lemma 3.15 keine Informationen darüber, dass bzw. ob $p \mid \gcd(a_0, \sigma^{-n}(a_n))$ gilt.

Spezielle Singularitäten

3.18 Definition. Sei $\alpha \in \mathbf{K}$, gilt $x - \alpha = \widehat{\sigma(x) - \alpha}$, so nennen wir α eine *spezielle Singularität*.
Den Punkt ∞ wollen wir immer eine spezielle Singularität nennen.

3.19 Beispiel. (i) Der Shiftoperator $x \mapsto x + 1$ hat nur ∞ als spezielle Singularität

(ii) Der q-Shiftoperator $x \mapsto qx$ hat 0 und ∞ als spezielle Singularitäten.

Für den Rest dieses Abschnitts bis Bemerkung 3.27 nehmen wir deshalb an, dass w keine Nullstelle in einer speziellen Singularität hat. Im Shift-Fall ist das gar keine Einschränkung, im q-Shift-Fall schließen wir nur x-Potenzen aus.

3.20 Lemma. *Sei p ein irreduzibler Teiler von w. Dann gelten*

(i) $\max_\sigma(\mathcal{T}_w^p) \in \mathcal{T}_{a_0}^p$ *und*

(ii) $\min_\sigma(\mathcal{T}_w^p) \in \mathcal{T}_{\sigma^{-n}(a_n)}^p$.

Beweis. Sei $p_0 = \max_\sigma(\mathcal{T}_w^p)$. Dann gilt für alle $k \in \mathbf{N}^+$: $p \nmid \sigma^k(w)$ nach der Definition von $\max_\sigma$ und $\mathcal{T}_w^p$. Also gilt nach Lemma 3.15 $p_0 \mid a_0$ und damit $p_0 \in \mathcal{T}_{a_0}^p$.
Der Beweis von (ii) ist analog. □

3.21 Korollar. *Sei p ein irreduzibles Polynom. Mit den vorherigen Aussagen und*

$$m_0^p := \min_\sigma \mathcal{T}_{\sigma^{-n}(a_n)}^p \quad \textit{und} \quad m_1^p := \max_\sigma \mathcal{T}_{a_0}^p$$

gelten:

(i) $p_0 \in \mathcal{T}_w^p \implies m_0 \preccurlyeq_\sigma p_0 \preccurlyeq_\sigma m_1$,

(ii) $m_1 \prec_\sigma m_0 \implies \mathcal{T}_w^p = \emptyset$ *und*

(iii) $$\mathcal{T}_w \subseteq \bigcup_{[p]_\sigma \in \mathbf{F}[x]/\sim_\sigma} \left\{ \sigma^k(p) \,\middle|\, \log_p^\sigma(m_0) \le k \le \log_p^\sigma(m_1) \right\}.$$

Beweis. (i) folgt aus Lemma 3.20, (ii) und (iii) folgen aus (i). □

σ-Dispersionsmenge $DS_\sigma(f,g)$

3.22 Definition. Für zwei Polynome $f, g \in \mathbf{F}[x]$ definieren wir die σ-*Dispersionsmenge von* f *und* g als

$$DS_\sigma(f,g) := \left\{ k \in \mathbf{Z} \,\middle|\, \deg \gcd(\sigma^k(f), g) > 0 \right\}.$$

σ-Dispersion $dis_\sigma(f,g)$

Ist $DS_\sigma(f,g) \neq \emptyset$, so definieren wir die σ-*Dispersion von* f *und* g als

$$dis_\sigma(f,g) := \max(DS_\sigma(f,g)).$$

3.23 Beispiel.

(i) Betrachte $\sigma = \tau$ und

$$\begin{aligned} f &= x^8 - 3x^7 + 4x^5 + 2x^4 - 8x^3 + 4x^2 + 16x - 16 \\ &= (x-1)\cdot(x-2)^2\cdot(x^2+2x+2)\cdot(x^3+2), \\ g &= x^5 + x^4 - 14x^3 - 22x^2 + 49x + 105 \\ &= (x-3)\cdot(x^2+4x+5)\cdot(x^2-7), \end{aligned}$$

dann sind $DS_\tau(f,g) = \{-2, 1\}$ und $dis_\tau(f,g) = 1$.

(ii) Mit $\sigma = \epsilon$ und

$$\begin{aligned} f &= (q^2+q)x^7 + (-q^4-q^3-2q^2-3q)x^5 + (-q-1)x^4 \\ &\qquad + (q^3+2q)x^3 + (q^3+q^2+2q+3)x^2 - (q^2+2) \\ &= ((q+1)x^2-1)\cdot(x^2-(q^2+2))\cdot(qx^3-1), \\ g &= (q^3+q^2)x^4 + (-q-\tfrac{2}{q}-\tfrac{2}{q^2}-q^4-1)x^2 + (q^2+2) \\ &= (\tfrac{1}{q}+\tfrac{1}{q^2})x^2 - 1)\cdot(q^4x^2-(q^2+2)) \end{aligned}$$

sind $DS_\epsilon(f,g) = \{-1, 2\}$ und $dis_\epsilon(f,g) = 2$.

3.24 Korollar. *Mit den obigen Notationen gelten*

(i) $$dis_\sigma(a,b) = \max\Big\{ \log_p^\sigma\big(\max_\sigma \mathcal{T}_a^p\big) - \log_p^\sigma\big(\min_\sigma \mathcal{T}_b^p\big) \,\Big|\, [p]_\sigma \in (\mathcal{T}_a/\sim_\sigma) \cap (\mathcal{T}_b/\sim_\sigma) \Big\},$$

(ii) $DS_\sigma(a_0, \sigma^{-n}(a_n)) = \emptyset \implies w = 1$.

Beweis. (i) folgt aus der Definition, (ii) folgt aus Lemma 3.20. □

3.25 Satz. *Seien*

$$L = \sum_{i=0}^{n} a_i \sigma^i, \quad a_0, \ldots, a_n \in \mathbf{F}[x], \quad a_n \neq 0$$

und

$$N := dis_\sigma(a_0, \sigma^{-n}(a_n)) \geq 0.$$

Dann ist

$$w' := \gcd\left(\prod_{i=0}^{N}\sigma^i(a_0),\quad \prod_{i=0}^{N}\sigma^{-n-i}(a_n)\right) \tag{3.2}$$

eine Schranke für den Nenner jeder rationalen Lösung von L.

Beweis. Sei nun $u = \frac{v}{w}$ eine maximal gekürzte Darstellung einer rationalen Lösung von L. Dann gelten wegen $a_n\sigma^n(u) = -\sum_{i=0}^{n-1} a_i\sigma^i(u)$:

$$\begin{aligned}
\sigma^n(u) &= -\frac{1}{a_n}\sum_{i=0}^{n-1} a_i\sigma^i(u)\\
\sigma^{n-1}(u) &= -\frac{1}{\sigma^{-1}(a_n)}\sum_{i=0}^{n-1} \sigma^{-1}(a_i)\sigma^{i-1}(u)\\
\sigma^{n-2}(u) &= -\frac{1}{\sigma^{-2}(a_n)}\sum_{i=0}^{n-1} \sigma^{-2}(a_i)\sigma^{i-2}(u)\\
&\vdots \qquad\quad \vdots\\
\sigma^{n-N}(u) &= -\frac{1}{\sigma^{-N}(a_n)}\sum_{i=0}^{n-1} \sigma^{-N}(a_i)\sigma^{i-N}(u)
\end{aligned}$$

Substituieren wir nun die Formeln für $\sigma^{n-1}(u), \dots, \sigma^{n-N}(u)$ in $\sigma^n(u)$, so erhalten wir

$$\sigma^n(u) = \frac{1}{\prod_{i=0}^{N}\sigma^{-i}(a_n)}\sum_{i=0}^{n-1}\hat{a}_i\,\sigma^{i-N}(u), \qquad \text{wobei } \hat{a}_i \in \mathbf{F}[x].$$

Nach der Definition von N und Korollar 3.21 haben die Nenner von $\sigma^n(u)$ und $\sigma^{n-N-1}(u), \dots, \sigma^{-N}(u)$ keine gemeinsamen Teiler. Demnach wird der Nenner $\sigma^n(w)$ von $\sigma^n(u)$ durch $\prod_{i=0}^{N}\sigma^{-i}(a_n)$ beschränkt, also gelten – mit analoger Argumentation:

$$w \mid \prod_{i=0}^{N}\sigma^{-n-i}(a_n) \qquad \text{und} \qquad w \mid \prod_{i=0}^{N}\sigma^{i}(a_0)$$

und damit

$$w \mid \gcd\left(\prod_{i=0}^{N}\sigma^i(a_0),\quad \prod_{i=0}^{N}\sigma^{-n-i}(a_n)\right).$$

□

Mit Satz 3.25 können wir nun rationale Lösungen berechnen:

3.26 Satz. *Seien*

$$L = \sum_{i=0}^{n} a_i \sigma^i, \quad a_0, \dots, a_n \in \mathbf{F}[x], \quad a_n \neq 0,$$

$N := \operatorname{dis}_\sigma(a_0, \sigma^{-n}(a_n)) \geq 0$ *und*

$$w' := \gcd\left(\prod_{i=0}^{N} \sigma^i(a_0), \quad \prod_{i=0}^{N} \sigma^{-n-i}(a_n)\right).$$

Dann korrespondiert jede rationale Lösung $u = \frac{v}{w}$ *von* L *zu einer polynomiellen Lösung von*

$$L' := \sum_{i=0}^{n} \frac{a_i}{\sigma^i(w')} \sigma^i.$$

Beweis. Sei $u = \frac{v}{w}$ eine maximal gekürzte Lösung von L. Dann existiert ein $v' \in \mathbf{F}[x]$ mit $u = \frac{v}{w} = \frac{v'v}{w'}$, also $w' = v'w$. Weiter gilt

$$\begin{aligned} 0 = L(u) = L\left(\tfrac{v}{w}\right) = L\left(\tfrac{v'v}{v'w}\right) = L\left(\tfrac{v'v}{w'}\right) \\ = \sum_{i=0}^{n} a_i \sigma^i\left(\frac{v'v}{w'}\right) = \sum_{i=0}^{n} \frac{a_i}{\sigma^i(w')} \sigma^i(v'v) = L'(v'v), \end{aligned}$$

und damit ist $v'v$ eine polynomielle Lösung von L'. □

Seit Lemma 3.20 haben wir vorausgesetzt, dass w keine Nullstellen in speziellen Singularitäten hat. Ganz konkret haben wir also die Nullstelle 0 im q-Fall bisher nicht betrachtet. Das soll nun nachgeholt werden. Dazu benutzen wir Ergebnisse aus Kapitel 2.

3.27 Bemerkung. Hat eine rationale Lösung $u = \frac{v}{w}$ eines q-Shiftoperators L einen Pol der Ordnung μ in 0, so kann das Zertifikat[1] $\operatorname{cert}_\sigma(u) := \frac{\sigma(u(x))}{u(x)}$ geschrieben werden als

$$\frac{v(qx)\, w(x)}{w(qx)\, v(x)} = q^{-\mu} \cdot (1 + \mathcal{O}(x)).$$

Nach Satz 2.40 ist die maximale q-Potenz, die in einer Nullstelle der charakteristischen Gleichung zur Steigung 0 des Newtonpolygons zur x-adischen Bewertung vorkommt, eine obere Schranke für μ.

[1] Siehe Definition 4.1 ff auf Seite 61.

3.4 Effiziente Berechnung der rationalen Lösungen

Zur effizienten Berechnung ist die Formel

$$w' := \gcd\left(\prod_{i=0}^{N}\sigma^i(a_0),\quad \prod_{i=0}^{N}\sigma^{-n-i}(a_n)\right) \tag{3.3}$$

denkbar ungeeignet, denn es müssen zwei Produkte potentiell großen Grades berechnet werden und dann deren ggT. Tatsächlich lässt sich die Berechnung durch die Benutzung der $\log^\sigma$-Schreibweise erheblich vereinfachen.
Faktorisiere

$$a := a_0\sigma^{-n}(a_n) = p_1^{e_1}\cdots p_l^{e_l}$$

mit $p_1,\dots,p_l \in \mathbf{F}[x]$ irreduzibel und $e_1,\dots,e_l \in \mathbf{N}_{>0}$. Sei

$$\begin{aligned}\mathcal{P}_a &:= \{\sigma\text{-minimale Vetreter jeder }\sigma\text{-Äquivalenzklasse von } \{p_1,\dots,p_l\}\}\\ &= \mathrm{ShiftEquivalenceClasses}(a).\end{aligned}$$

3.28 Definition. Für ein beliebiges Polynom $f \in \mathbf{F}[x]$ definiere

$$\mathrm{Log}_p^\sigma(f) := \{(s,e)\in\mathbf{Z}\times\mathbf{N} \mid e \text{ maximal mit } \sigma^s(p)^e \mid f\}. \tag{3.4}$$

3.29 Beispiel. (Fortsetzung von 3.23)

(i) Betrachte $\sigma = \tau$ und

$$f = (x-1)\cdot(x-2)^2\cdot(x^2+2x+2)\cdot(x^3+2),$$

dann sind etwa

$$\mathrm{Log}_x^\tau(f) = \{(-1,1),\ (-2,2)\} \quad\text{und}\quad \mathrm{Log}_{x^2+1}^\tau(f) = \{(1,1)\}.$$

(ii) Mit $\sigma = \epsilon$ und

$$g = ((\tfrac{1}{q}+\tfrac{1}{q^2})x^2-1)\cdot(q^4x^2-(q^2+2))$$

ist z. B.

$$\mathrm{Log}_{(q+1)x^2-1}^\epsilon(f) = \{(-1,1)\}.$$

Algorithmus 9: SigmaLogarithmRep(p, f)

Input: $f \in \mathbf{F}[x]$, $p \in \mathbf{F}[x]$ irreduzibel
Output: Eine Menge $\{(s_1,e_1),\dots,(s_l,e_l)\} \in \mathbf{Z}\times\mathbf{N}$ wie in Gleichung (3.4)

```
1  S ← ∅
2  faktorisiere f = f_1^{e_1} ··· f_r^{e_r}
3  for i ← 1, ..., r
4  |  s ← ShiftEquivalence(p, f_i)
5  |  if s ≠ fail
6  |  |  S ← S ∪ {(s, e_i)}
7  return S
```

Damit erhalten wir eine Darstellung

$$a = \prod_{p \in \mathcal{P}_a} \prod_{(s,e) \in \mathrm{Log}_p^\sigma(a)} \sigma^s(p)^e.$$

Definiere nun

$$S := \bigcup_{p \in \mathcal{P}_a} \mathrm{Log}_p^\sigma(a), \quad s_0 := \min\{s \mid (s,e) \in S\}, \quad s_1 := \max\{s \mid (s,e) \in S\}.$$

Dann können wir die Produkte in (3.3) umschreiben als

$$\prod_{i=0}^{N} \sigma^i(a_0) = \prod_{i=s_0}^{s_1} \prod_{p \in \mathcal{P}_a} p^{e_p^i}, \quad \text{mit} \quad e_p^i := \sum_{\substack{(s,e) \in \mathrm{Log}_p^\sigma(a_0) \\ s \geq i}} e,$$

$$\prod_{i=0}^{N} \sigma^{-n-i}(a_n) = \prod_{i=s_0}^{s_1} \prod_{p \in \mathcal{P}_a} p^{\bar{e}_p^i}, \quad \text{mit} \quad \bar{e}_p^i := \sum_{\substack{(s,e) \in \mathrm{Log}_p^\sigma(a_n) \\ s \leq i}} e,$$

und wir erhalten schließlich

$$w' = \gcd\left(\prod_{i=0}^{N} \sigma^i(a_0), \prod_{i=0}^{N} \sigma^{-n-i}(a_n)\right) = \prod_{i=s_0}^{s_1} \prod_{p \in \mathcal{P}_a} p^{\min\{e_p^i, \bar{e}_p^i\}}.$$

Algorithmus 10: DenomBound(L)

Input: $L = \sum_{i=0}^{n} a_i \sigma^i$, $a_0, \ldots, a_n \in \mathbf{F}[x]$, $\gcd(a_0, \ldots, a_n) = 1$, $a_n \neq 0$
Output: Eine Schranke $w' \in \mathbf{F}[x]$ für den Nenner jeder rationalen Lösung von L.

```
1  a ← a_0 · σ^{-n}(a_n)
2  P_a ← ShiftEquivalenceClasses(a)
3  s_0 ← ∞,   s_1 ← −∞
4  for all p ∈ P_a
5  |  L_p(a_0) ← SigmaLogarithmRep(a_0)
6  |  L_p(a_n) ← SigmaLogarithmRep(σ^{-n}(a_n))
7  |  for all (s, e) ∈ L_p(a_0) ∪ L_p(a_n)
8  |  |  s_0 ← min{s_0, s},   s_1 ← max{s_1, s}
9  b ← 1
10 for i ← s_0, ..., s_1
11 |  for all p ∈ P_a
12 |  |  e_p^i ← Σ_{(s,e)∈L_p(a_0), s≥i} e,   ē_p^i ← Σ_{(s,e)∈L_p(a_n), s≤i} e,
13 |  |  b ← b · p^{min{e_p^i, ē_p^i}}
14 return b
```

Im q-Fall muss das Ergebnis dieser Berechnung noch mit der über das Newtonpolygon gemäß Bemerkung 3.27 bzw. Satz 2.40 berechneten x-Potenz multipliziert werden.
Damit haben wir den Algorithmus zur Berechnung der rationalen Lösungen hergeleitet:

Algorithmus 11: RationalSolutions(L)

Input: $L = \sum_{i=0}^{n} a_i \sigma^i$, $a_0, \ldots, a_n \in \mathbf{F}[x]$, $\gcd(a_0, \ldots, a_n) = 1$, $a_n \neq 0$
Output: Basis der rationalen Lösungen von L

1 $w \leftarrow \text{DenomBound}(L)$
2 $L' \leftarrow \text{lcm}(\sigma^0(w), \ldots, \sigma^n(w)) \cdot \left(\sum_{i=0}^{n} \frac{a_i}{\sigma^i(w)} \sigma^i\right)$
3 $S \leftarrow \text{PolynomialSolutions}(L')$
4 return $\{\frac{u}{w} \mid u \in S\}$

3.5 Implementierung

3.5.1 Polynomielle Lösungen

MuPAD-Session 6: Polynomielle Lösungen

Wir konstruieren einen Shiftoperator, der ein bestimmtes Polynom als Lösung hat:

```
>> p := x^3+2*x^2-4*x+7:

>> L := R(p*S-subs(p,x=x+1))
```

$$(x^3 + 2x^2 - 4x + 7)\,S + (-x^3 - 5x^2 - 3 * x - 6)$$

```
>> Sol := R::polysols(L)
```

$$\{x^3 + 2x^2 - 4x + 7\}$$

```
>> L(op(Sol))
```

$$0$$

```
>> L := R(S^2+x*S-15)*L
```

$$\left(x^3 + 8x^2 + 16x + 15\right) S^3 + \left(x^4 + 4x^3 - 8x^2 - 29x - 40\right) S^2 +$$

$$\left(-x^4 - 23x^3 - 46x^2 + 45x - 105\right) S + 15x^3 + 75x^2 + 45x + 90$$

```
>> Sol := R::polysols(L)
```

$$\{x^3 + 2x^2 - 4x + 7\}$$

```
>> L(op(Sol))
```

$$0$$

Das q-Analogon dazu:

```
>> L := qR(p*Eq-subs(p,x=q*x))
```

$$\left(x^3 + 2x^2 - 4x + 7\right) Eq + \left(-q^3\right) x^3 + \left(-2q^2\right) x^2 + (4q)\, x - 7$$

```
>> Sol := qR::qpolysols(L)
```

$$\{x^3 + 2x^2 - 4x + 7\}$$

```
>> L(op(Sol))
```

$$0$$

```
>> L := qR(Eq^2+x*Eq-15)*L
```

$$\left(q^6 x^3 + \left(2\,q^4\right)\,x^2 + \left(-4\,q^2\right)\,x + 7\right)\,Eq^3 +$$
$$\left(q^3 x^4 + \left(2\,q^2 - q^9\right)\,x^3 + \left(-2\,q^6 - 4\,q\right)\,x^2 + \left(4\,q^3 + 7\right)\,x - 7\right)\,Eq^2 +$$
$$\left(\left(-q^6\right)\,x^4 + \left(-2\,q^4 - 15\right)\,x^3 + \left(4\,q^2 - 30\right)\,x^2 + 53\,x - 105\right)\,Eq +$$
$$\left(15\,q^3\right)\,x^3 + \left(30\,q^2\right)\,x^2 + (-60\,q)\,x + 105$$

```
>> Sol := qR::qpolysols(L)
```

$$\{x^3 + 2\,x^2 - 4\,x + 7\}$$

```
>> L(op(Sol))
```

$$0$$

3.5.2 Rationale Lösungen

MuPAD-Session 7: Rationale Lösungen

Wie oben betrachten wir jetzt rationale Lösungen im Shift-Fall:

```
>> L := R((x+6)*S-x)
```

$$(x+6)\,S - x$$

```
>> Sol := R::ratsols(L)
```

$$\left\{\frac{1}{x\,(x+1)\,(x+2)\,(x+3)\,(x+4)\,(x+5)}\right\}$$

```
>> L(op(Sol))
```

$$0$$

```
>> r := (x^2+2*x-7)/(x^3+4*x-11):

>> L := R(r*S-subs(r,x=x+1))
```

$$\left(\frac{x^2+2\,x-7}{x^3+4\,x-11}\right)\,S + -\frac{x^2+4\,x-4}{x^3+3\,x^2+7\,x-6}$$

```
>> Sol := R::ratsols(L)
```

$$0$$

```
>> L := R((x+6)*S-x)*L
```

$$\left(\frac{x^3+10\,x^2+20\,x-24}{x^3+3\,x^2+7\,x-6}\right)\,S^2 +$$

$$\left(-\frac{2x^6+20x^5+62x^4+38x^3-86x^2-418x-66}{x^6+6x^5+20x^4+18x^3-2x^2-156x-55}\right) S+$$

$$\frac{x^3+4x^2-4x}{x^3+3x^2+7x-6}$$

≫ R::polysols(L)

$$\{\}$$

≫ Sol := R::ratsols(L)

$$\left\{\frac{\frac{x^7}{12}+x^6+4x^5+\frac{25x^4}{6}-\frac{121x^3}{12}-\frac{151x^2}{6}-14x}{x\ (x+1)\ (x+2)\ (x+3)\ (x+4)\ (x^3+4x-11)}\right\}$$

≫ L(op(Sol))

$$0$$

und wieder das q*-Analogon:*

≫ L := qR(r*Eq-subs(r,x=q*x))

$$\left(\frac{(x^2+2x-7)}{(x^3+4x-11)}\right) Eq+\frac{(-q^2)\ x^2+(-2q)\ x+7}{q^3x^3+(4q)\ x-11}$$

≫ Sol := qR::ratsols(L)

$$\left\{\frac{x^2+2x-7}{x^3+4x-11}\right\}$$

≫ L(op(Sol))

$$0$$

≫ L := qR::lclm(qR(q^6*x*Eq^2+x*Eq-x),L)

$$Eq^3+$$

$$\left(\frac{(q^{23}-q^{27}-q^{18}+q^{17})\ x^{11}+\ldots+(9317\,q^6-9317\,q^{12})}{(q^{28}-q^{24}+q^{23})\ x^{11}+\ldots+(726\,q^7\ldots-726\,q^6)\ x+9317\,q^{12}}\right) Eq^2+$$

$$\left(\frac{(q^{18}-q^{21}-q^{22}-q^{16})\ x^{11}+\ldots+(6050\,q^9+\ldots)\ x-18634\,q^6}{(q^{28}-q^{24}+q^{23})\ x^{11}+\ldots+(726\,q^7-\ldots)\ x+9317\,q^{12}}\right) Eq+$$

$$\frac{(q^{20}-q^{16}+q^{15})\ x^{11}+\ldots+(726\,q-3388\,q^7-\ldots)\ x+9317\,q^5}{(q^{27}-q^{23}+q^{22})\ x^{11}+\ldots+(726\,q^6-2662\,q^{13}-\ldots)\ x+9317\,q^{11}}$$

≫ Sol := qR::ratsols(L)

$$\left\{\frac{x^2+2x-7}{x^3+4x-11}\right\}$$

≫ expand(L(op(Sol)))

$$0$$

Kapitel 4

σ-Hypergeometrische Lösungen

Nachdem wir gesehen haben, wie polynomielle und rationale Lösungen eines Operators berechnet werden können, wenden wir uns in diesem Kapitel den nächst-komplexeren Lösungen zu. Die hypergeometrischen Lösungen sind genau die, die zu Rechtsfaktoren erster Ordnung korrespondieren (vgl. Satz 4.3), und wir beschränken uns hier auch auf die Berechnung dieser Rechtsfaktoren.

Im Wesentlichen werden hier zwei verschiedene Algorithmen vorgestellt:

- *der van-Hoeij-Algorithmus, der hier vereinfacht und auf den* q*-Shift-Fall übertragen wird;*
- *der Petkovšek-Algorithmus, der hier – vor allem im* q*-Shift-Fall – massiv beschleunigt werden kann.*

4.1 σ-Hypergeometrische Terme

In Kapitel 2 haben wir den Begriff der *hypergeometrischen Lösung* bereits kurz eingeführt (vgl. Seite 27), hier folgt eine etwas ausführlichere Definition.

4.1 Definition. Sei $u(x)$ ein Ausdruck, der neben rationalen Funktionen über $\mathbf{F}$ möglicherweise spezielle Funktionen eines geeigneten Erweiterungskörpers von $\mathbf{F}(x)$ enthält. $u(x)$ heißt σ-*hypergeometrisch*, wenn

hypergeometrische Terme $\mathbb{H}^{\sigma}_{\mathbf{F},x}$

$$\operatorname{cert}_{\sigma}(u) := \frac{\sigma(u(x))}{u(x)} \in \mathbf{F}(x)$$

gilt. Die Menge der σ-hypergeometrischen Terme über $\mathbf{F}$ in x bezeichnen wir mit $\mathbb{H}^{\sigma}_{\mathbf{F},x}$. Die rationale Funktion $\operatorname{cert}_{\sigma}(u)$ bezeichnen wir dann als σ-*Zertifikat von* u.

Zertifikat $\operatorname{cert}_{\sigma}(u)$

Zwei hypergeometrische Terme u_1, u_2 sind *vom gleichen Typ*, wenn $\frac{u_1}{u_2} \in \mathbf{F}(x)$.

4.2 Bemerkung. (i) Vom gleichen Typ zu sein, ist eine Äquivalenzrelation.

(ii) Sind u_1, u_2 vom gleichen Typ, so ist $u_1 + u_2$ ein hypergeometrischer Term oder Null.

(iii) Die hypergeometrischen Terme sind abgeschlossen unter Multiplikation, aber *nicht* abgeschlossen unter Addition.

4.3 Satz. *Ist* u *eine hypergeometrische Lösung von* L, *so ist* $\sigma - \mathrm{cert}_\sigma(u)$ *ein Rechtsfaktor erster Ordnung von* L.

Beweis. Nach Bemerkung 1.11 existieren $Q, R \in \mathcal{S}$, so dass $L = Q \cdot (\sigma - \mathrm{cert}_\sigma(u)) + R$ und $\mathrm{order}(R) = 0$ oder $R = 0$. Betrachte nun

$$0 = L(u) = (Q \cdot (\sigma - \mathrm{cert}_\sigma(u)) + R)(u) = R \cdot u,$$

also muss $R = 0$ gelten. □

4.4 Bemerkung. Andererseits korrespondiert sogar jeder Rechtsfaktor erster Ordnung zu einer hypergeometrischen Lösung über einem geeigneten Erweiterungskörper bzw. über $\overline{\mathbf{F}}$. Da wir in dieser Arbeit aber die benötigen Γ- bzw. Γ_q-Funktionen nicht eingeführt haben, zeigen wir dies nur in einem Beispiel.

4.5 Beispiel. Ist etwa $u(x) = \Gamma(x+1)\,\Gamma(x+\frac{2}{3})$ eine Lösung, so ist das Zertifikat

$$\mathrm{cert}_\tau(u) = \frac{\Gamma(x+2)\,\Gamma(x+\frac{5}{3})}{\Gamma(x+1)\,\Gamma(x+\frac{2}{3})} = (x+1)(x+\tfrac{2}{3}),$$

analog lässt sich das Zertifikat von jedem Produkt von Γ-Potenzen als rationale Funktion schreiben.
Umgekehrt kann aus jedem Zertifikat, dessen Zähler und Nenner in Linearfaktoren zerfallen, der hypergeometrische Term rekonstruiert werden. Siehe dazu etwa [52] und [80].

4.2 Der lokale Typ

Wir wollen hier die prinzipielle Struktur des Zertifikats eines hypergeometrischen Terms analysieren.
Sei $u(x)$ ein hypergeometrischer Term und $r = \mathrm{cert}_\sigma(u)$ sein Zertifikat. Faktorisiert man r über $\mathbf{F}$, so erhält man

$$r = c \cdot \varphi_1^{\alpha_1} \cdots \varphi_m^{\alpha_m} \cdot \psi_1^{-\beta_1} \cdots \psi_{m'}^{-\beta_{m'}},$$

mit $c \in \mathbf{F}^\times,\ \alpha_i, \beta_j \in \mathbf{N},\ \varphi_i, \psi_j \in \mathbf{F}[x]$ irreduzibel, $\deg \varphi_i = d_i,\ \deg \psi_j = e_j$, und

$$\varphi_i(x) = x^{d_i} + \varphi_{i,1}x^{d_i-1} + \ldots + \varphi_{i,d_i}$$
$$\psi_j(x) = x^{e_j} + \psi_{j,1}x^{e_j-1} + \ldots + \psi_{j,e_j}.$$

Der lokale Typ eines hypergeometrischen Terms beschreibt die Null- und Polstellen des Zertifikats modulo σ. In Definition 3.18 haben wir den Begriff der speziellen Singularität eingeführt, insofern müssen wir einerseits die endlichen Singularitäten und andererseits die speziellen Singularitäten betrachten.

4.2.1 Lokaler Typ an endlichen Stellen

Lokaler Typ in $[p]_\sigma$ $\mathrm{ltyp}_{[p]_\sigma}(u(x))$

4.6 Definition. Sei $p \in \mathbf{F}[x]$ irreduzibel, und $[p]_\sigma \in \mathbf{F}[x]/\sim_\sigma$ die σ-Äquivalenzklasse von p. Der *lokale Typ in* $[p]_\sigma$ von u ist definiert als

$$\mathrm{ltyp}_{[p]_\sigma}(u(x)) = \sum_{\substack{i=1..m \\ \varphi_i \in [p]_\sigma}} d_i\alpha_i - \sum_{\substack{j=1..m' \\ \psi_j \in [p]_\sigma}} e_j\beta_j.$$

4.2.2 Lokaler Typ in speziellen Singularitäten

Wir können das Zertifikat in den speziellen Singularitäten in Potenzreihen entwickeln und auf diesem Wege Informationen über das Verhalten des Zertifikats gewinnen. Betrachte zunächst die spezielle Singularität ∞, die sowohl im Shift- als auch im q-Shift-Fall auftritt.
Wir können das Zertifikat $r(x) := \mathrm{cert}_\sigma(u)$ schreiben als

$$\begin{aligned} r(x) &= c_\infty \cdot \frac{x^t + x^{t-1}\sum_{i=1}^{m} \alpha_i\,\varphi_{i,1} + \ldots + \prod_{i=1}^{m} \varphi_{i,d_i}^{\alpha_i}}{x^{t'} + x^{t'-1}\sum_{j=1}^{m'} \beta_j\,\psi_{j,1} + \ldots + \prod_{j=1}^{m'} \psi_{j,e_j}^{\beta_j}} \\ &= c_\infty \cdot x^{s_\infty} \cdot (1 + \mathcal{O}(x^{-1})), \end{aligned}$$

wobei $c_\infty \in \mathbf{F}^\times$ und $s_\infty := t - t' \in \mathbf{Z}$. Genauer ist

$$s_\infty = \sum_{i=1}^{m} \deg(\varphi_i)\,\alpha_i - \sum_{j=1}^{m'} \deg(\psi_j)\,\beta_j.$$

Offensichtlich dominieren c_∞ und s_∞ das asymptotische Verhalten von r in ∞.

Lokaler Typ in ∞
$\operatorname{ltyp}_\infty(u(x))$

4.7 Definition. Mit den obigen Bezeichnungen definieren wir den *lokalen Typ in* ∞ des hypergeometrischen Terms $u(x)$ als

$$\operatorname{ltyp}_\infty(u(x)) := (c_\infty, s_\infty) \in \mathbf{F}^\times \times \mathbf{Z}.$$

Im q-Fall bekommen wir zusätzlich die spezielle Singularität 0, können also zusätzlich um 0 entwickeln:

$$r(x) = c_0 \cdot x^{s_0}(1 + \mathcal{O}(x)),$$

wobei $c_0 \in \mathbf{F}^\times$ und $s_0 \in \mathbf{Z}$. Genauer ist

$$s_0 = \sum_{i=1}^{m} \nu_{\mathrm{tdeg}}(\varphi_i)\,\alpha_i - \sum_{j=1}^{m'} \nu_{\mathrm{tdeg}}(\psi_j)\,\beta_j.$$

Lokaler Typ in 0
$\operatorname{ltyp}_0(u(x))$

4.8 Definition. Mit den obigen Bezeichnungen definieren wir den *lokalen Typ in* 0 des hypergeometrischen Terms $u(x)$ als

$$\operatorname{ltyp}_0(u(x)) := (c_0, s_0) \in \mathbf{F}^\times \times \mathbf{Z}.$$

4.9 Bemerkung. Seien $u_1, u_2 \in \mathbb{H}^\sigma_{\mathbf{F},x}$ und

$$\operatorname{ltyp}_\infty(u_1) = (c_1, s_1), \quad \text{und} \quad \operatorname{ltyp}_\infty(u_2) = (c_2, s_2)$$

ihre lokalen Typen in ∞, dann ist

$$\operatorname{ltyp}_\infty(u_1 \cdot u_2) = (c_1 \cdot c_2, s_1 + s_2).$$

Ist $\sigma = \epsilon$, so gilt die Aussage auch für ltyp_0.

4.2.3 Fuchs-Relationen

Zwischen den lokalen Typen an endlichen Stellen und den lokalen Typen in den speziellen Singularitäten besteht ein Zusammenhang.

4.10 Satz (Fuchs-Relationen)**.** *Sei* u *ein hypergeometrischer Term,* $\operatorname{ltyp}_\infty(u) = (c_\infty, s_\infty)$*, dann gilt in beiden Fällen:*

$$s_\infty = \sum_{[p]_\sigma \in \mathbf{F}[x]/\sim_\sigma} \operatorname{ltyp}_{[p]_\sigma}(u(x)), \tag{4.1}$$

und im q*-Shift-Fall mit* $\operatorname{ltyp}_0 = (c_0, s_0)$ *gilt zusätzlich:*

$$s_0 = \operatorname{ltyp}_{[x]_\sigma}(u(x)). \tag{4.2}$$

Beweis. s_∞ ist gerade die Differenz der Grade von Zähler und Nenner des Zertifikats. $\operatorname{ltyp}_{[p]_\sigma}(u(x))$ gibt die Graddifferenz *pro* σ*-Äquivalenzklasse* an, und da über alle Klassen summiert wird, muss Gleichheit gelten. Im q-Shift-Fall ist s_0 die Vielfachheit der Nullstelle bzw. des Pols 0. □

4.11 Satz. *Zwei hypergeometrische Terme u_1, u_2 sind genau dann vom selben Typ, wenn für jedes $[p]_\sigma \in \mathbf{F}[x]/\sim_\sigma$ gilt $\mathrm{ltyp}_{[p]_\sigma}(u_1) = \mathrm{ltyp}_{[p]_\sigma}(u_2)$ und $\mathrm{ltyp}_\infty(u_1) = \mathrm{ltyp}_\infty(u_2)$ (und ggfs. $\mathrm{ltyp}_0(u_1) = \mathrm{ltyp}_0(u_2)$.)*

4.12 Korollar. *Wenn wir für jedes $[p]_\sigma \in \mathbf{F}[x]/\sim_\sigma$ und ∞ (und 0) den lokalen Typ eines hypergeometrischen Terms kennen, so kennen wir den Typ des Terms.*

Das motiviert die folgende Definition:

globaler Typ $\mathrm{gtyp}(u)$

4.13 Definition. Mit der obigen Definition definieren wir den *globalen Typ* eines hypergeometrischen Terms $u \in \mathbb{H}^\sigma$ als

$$\begin{aligned}\mathrm{gtyp}(u) = &\Big\{\big(\infty, \mathrm{ltyp}_\infty(u)\big)\Big\} \cup \Big\{\big(0, \mathrm{ltyp}_0(u)\big)\Big\} \cup \\ &\Big\{\big([p]_\sigma, \mathrm{ltyp}_{[p]_\sigma}(u)\big)\Big| [p]_\sigma \in \mathbf{F}[x]/\sim_\sigma,\ \mathrm{ltyp}_{[p]_\sigma}(u) \neq 0\Big\}.\end{aligned}$$

4.3 Das symmetrische Produkt

symmetrisches Produkt $L_1 \circledS L_2$

4.14 Definition. Für $L_1, L_2 \in \mathbf{F}(x)[\sigma]$ definiere das *symmetrische Produkt* $L_1 \circledS L_2$ als den normierten Operator L minimalen Grades, so dass für alle u_1, u_2 mit $L_1(u_1) = L_2(u_2) = 0$ gilt $L(u_1\, u_2) = 0$.

4.15 Bemerkung. Sei

$$P = \sum_{i=0}^{d} f_i\, \sigma^i, \quad Q = \sum_{j=0}^{e} g_j\, \sigma^j, \quad f_i, g_j \in \mathbf{F}(x),$$

ohne Einschränkung $f_d = g_e = 1$. Seien weiter $u, v \in \mathbb{H}^g_{\mathbf{F},x}$ mit $P(u) = Q(v) = 0$. Dann gelten

$$\sigma^d(u) = -f_{d-1}\, \sigma^{d-1}(u) - \ldots - f_0 u, \tag{4.3}$$

$$\sigma^e(v) = -g_{e-1}\, \sigma^{e-1}(v) - \ldots - g_0 v. \tag{4.4}$$

Nun machen wir den Ansatz

$$S = \sum_{k=0}^{d\cdot e} s_k\, \sigma^k \quad \text{und} \quad S(u \cdot v) = 0.$$

Da σ ein Automorphismus ist, können wir in

$$S(u\, v) = \sum_{k=0}^{d\cdot e} s_k\, \sigma^k(u\, v) = \sum_{k=0}^{d\cdot e} s_k\, \sigma^k(u)\, \sigma^k(v)$$

alle auftretenden Ausdrücke der Form $\sigma^{d+j}(u), \sigma^{e+i}(v)$ mit Hilfe von (4.3) und (4.4) ersetzen.
Bezeichne das Ergebnis dieser Ersetzung mit $\tilde{S}(uv) \in \mathbf{F}[\sigma(u), \sigma(v)]$. Dann hat $\tilde{S}(uv)$ die Ordnung $e \cdot d - 2$ und nach Koeffizientenvergleich $\tilde{S}(uv) = 0$

bekommen wir ein homogenes lineares Gleichungssystem mit $e \cdot d$ Unbekannten und $e \cdot d - 1$ Gleichungen.

4.16 Bemerkung. Ist $Q = \sigma - g_0$ von erster Ordnung, so können wir eine explizite Fomel für $P \circledS Q$ angeben:

$$P \circledS (\sigma - g_0) = \frac{1}{h_n} \sum_{i=0}^{d} h_i \, \sigma^i$$

mit

$$h_d(x) = f_d(x)$$
$$h_i(x) = \begin{cases} f_i(x) \cdot \prod_{j=i}^{d-1} g_0(x+j) & \text{im Shift-Fall,} \\ f_i(x) \cdot \prod_{j=i}^{d-1} g_0(q^j x) & \text{im q-Shift-Fall,} \end{cases} \quad \text{für } i = 0, \ldots, d-1.$$

Algorithmus 12: SymmetricProduct(P, Q)

Input: $P = \sum_{i=0}^{d} f_i \, \sigma^i$, $Q = \sum_{j=0}^{e} g_j \, \sigma^j \in \mathbf{F}(x)[\sigma]$
Output: $R = P \circledS Q \in \mathbf{F}(x)[\sigma]$

```
d ← order P, e ← order Q
s_P ← −f_{d−1} U^{d−1} − ... − f_1 U − f_0, U Unbestimmte
s_Q ← −g_{e−1} V^{d−1} − ... − g_1 V − f_0, V Unbestimmte
for J ← max{d, e}, ..., d · e
    S ← (U V)^J + Σ_{i=0}^{d·e−1} s_i U^i V^i, s_i Unbestimmte
    S̃ ← S
    while deg_U(S̃) ≥ d
        Ersetze S̃ ← S̃ |_{U^{deg_U(S̃)} → s_P^{deg_U(S̃)−d−1}}
    while deg_V(S) ≥ e
        Ersetze S̃ ← S̃ |_{V^{deg_V(S)} → s_Q^{deg_V(S)−e−1}}
    Bestimme LGS durch Koeffizientenvergleich S̃ = 0
    Berechne 𝒮 ← linearSolve(LGS)
    If 𝒮 ≠ ∅
        R ← subs(S, 𝒮)
        Beende die For-Schleife
return R
```

4.4 Vom lokalen Typ zur Lösung – der van Hoeij-Algorithmus

Angenommen, ein Orakel gäbe uns den globalen Typ einer unbekannten Lösung u von L, etwa

$$T = \{(\infty, (c, s)), \ ([p_1]_\sigma, k_1), \ldots, ([p_l]_\sigma, k_l)\},$$

mit $[p_1]_\sigma, \ldots, [p_l]_\sigma \in \mathbf{F}[x]/\sim_\sigma$.Wie kann man nun eine hypergeometrische Lösung u dieses Typs konstruieren?

Sei

$$r := c \cdot (p_1)^{k_1} \cdots (p_l)^{k_l}$$

Dann hat jede Lösung $\tilde{u}$ von $\sigma - r$ den Typ T, und nach Definition 4.1 sehen wir $\frac{\tilde{u}}{u} \in \mathbf{F}(x)$.
Demnach ist $\frac{1}{\tilde{u}}$ eine Lösung von $\sigma - \frac{1}{r}$ und damit ist $\frac{u}{\tilde{u}}$ eine *rationale* Lösung von $L\text{Ⓢ}(\sigma - \frac{1}{r})$.
Damit haben wir das Problem, eine hypergeometrische Lösung *eines gegebenen Typs* zu finden, auf die Berechnung einer rationalen Lösung eines modifizierten Operators reduziert.

Es bleibt also die – hochgradig nicht-triviale – Frage zu beantworten, wie man den Kandidaten T für den globalen Typ einer hypergeometrischen Lösung finden kann.

4.4.1 Lokaler Typ an endlichen Stellen

Die Ideen für die Berechnung der Nennerschranke (3.2) (siehe Seite 56) aus Abschnitt 3.4 geben uns Schranken für die Multiplizität der Pole jeder Lösung.

4.17 Lemma. *Sei* $L = \sum_{i=0}^{n} a_i \sigma^i$ *mit* $a_i \in \mathbf{F}[x]$. *Für jede hypergeometrische Lösung* $u(x)$ *mit* $\mathrm{cert}_\sigma(u) = c\frac{f}{g}$, $c \in \mathbf{F}$, $f, g \in \mathbf{F}[x]$ *normiert und teilerfremd, gelten* $f \mid a_0$ *und* $\sigma^{n-1}(g) \mid a_n$.

Beweis. Ist $g\,\sigma - f$ mit $f, g \in \mathbf{F}[x]$ und $\gcd(f, g) = 1$ der zu einer hypergeometrischen Lösung gehörende *polynomielle* Rechtsfaktor erster Ordnung, so existiert ein Linksfaktor $\tilde{L} = \sum_{i=0}^{n-1} b_i \sigma^i$, so dass $L = \tilde{L} \cdot (g\,\sigma - f) = b_{n-1}\sigma^{n-1}(g)\sigma^n + \ldots - b_0 f$. □

Als Konsequenz erhalten wir:

4.18 Satz. *Setze für die Äquivalenzklasse* $[p]_\sigma$ *jedes normierten Polynoms* p, *das als Teiler von* $a := a_0\sigma^{-n+1}(a_n)$ *auftritt:*

$$E^{\min}_{[p]_\sigma} := \quad -\sum_{(s,e)\in \mathrm{Log}^\sigma_p(a_n)} e \; = -\nu_{[p]_\sigma}(a_n),$$

$$E^{\max}_{[p]_\sigma} := \quad \sum_{(s,e)\in \mathrm{Log}^\sigma_p(a_0)} e \; = \nu_{[p]_\sigma}(a_0).$$

Dann gilt für jedes $[p]_\sigma$ *mit* $p \in \mathcal{T}_{a_0 \cdot a_n}$ *und jede hypergeometrische Lösung* u

$$E^{\min}_{[p]_\sigma} \le \nu_{[p]_\sigma}(\mathrm{cert}_\sigma(u)) \le E^{\max}_{[p]_\sigma}.$$

Definiere deswegen $V_{[p]_\sigma}(L) := \{E^{\min}_{[p]_\sigma}, \ldots, E^{\max}_{[p]_\sigma}\}$.

Bewertungs-
schranke $V_{[p]_\sigma}(L)$

Um alle Möglichkeiten durchzutesten benötigen wir die Berechnung von $\prod_{\substack{[p]_\sigma \in \mathbf{F}[x]/\sim_\sigma \\ \mathrm{ltyp}_{[p]_\sigma}(u) \neq 0}} |E^{\max}_{[p]_\sigma} - E^{\max}_{[p]_\sigma}| + 1$ rationalen Lösungen.

Algorithmus 13: valbound(L, p)

Input: $0 \neq L = \sum_{i=0}^{n} a_i \sigma^i \in \mathbf{F}(x)[\sigma]$, $p \in \mathbf{F}[x]$
Output: $V_{[p]_\sigma}(L)$

```
1  S_0 ← SigmaLogarithmRep(p, a_0)
2  S_n ← SigmaLogarithmRep(p, a_n)
3  E_p^min ← −∑_{(s,e)∈S_n} e
4  E_p^max ← ∑_{(s,e)∈S_0} e
5  return {E_p^min, ..., E_p^max}
```

Der Aufwand kann massiv beschränkt werden, wenn man die Fuchs-Relationen (4.1) bzw. (4.2) benutzt, um bestimmte Zusammenstellungen von lokalen Typen direkt auszusortieren. Dazu benötigen wir allerdings eine Möglichkeit, den lokalen Typ in ∞ aller möglichen Lösungen zu bestimmen.

4.4.2 Lokaler Typ in ∞ bzw. 0

Auf Seite 64 haben wir gesehen, dass sich das Zertifikat jeder Lösung $u(x)$ schreiben lässt als

$$\mathrm{cert}_\sigma(u(x)) = c_\infty \cdot x^{s_\infty} \cdot (1 + \mathcal{O}(x^{-1})),$$

und haben $\mathrm{ltyp}_\infty(u) := (c_\infty, s_\infty)$ gesetzt.

Hier suchen wir nun eine Menge $V_\infty(L)$, so dass für jede hypergeometrische Lösung u von L gilt

$$\mathrm{ltyp}_\infty(u) \in V_\infty(L).$$

Nach Satz 2.22 stimmt die Bewertung des Zertifikats einer Lösung stets mit dem negativen einer Steigung einer Kante des Newtonpolygons überein. Benutzen wir die Bewertung $\nu_{\deg}$, so bedeutet das, dass s_∞ als Kante im Newtonpolygon auftauchen muss, und andererseits nur die ganzzahligen Steigungen für s_∞ in Frage kommen.

Nach Satz 2.40 kommen für die zugehörigen c_∞ nur die Nullstellen der charakteristischen Gleichung $P_{\nu_{\deg},s}$ in Frage, wir können die möglichen lokalen Typen in ∞ also direkt vom Newtonpolygon zur Bewertung $\nu_{\deg}$ ablesen.

Algorithmus 14: LocalTypeInInfinity(L)

Input: $0 \neq L = \sum_{i=0}^{n} a_i \sigma^i \in \mathbf{F}(x)[\sigma]$, $p \in \mathbf{F}[x]$
Output: Menge S, so dass der lokale Typ in ∞ jeder Lösung von L in S liegt.

```
1  P ← NewtonPolygon(L, −deg)
2  S ← ∅
3  for all (r, pol) ∈ P
4  |  if r ∉ Z then next
5  |  for all c ∈ solve(pol = 0)
6  |  |  S ← S ∪ {(r, c)}
7  return S
```

Im q-Fall bekommen wir für die zweite spezielle Singularität 0 mit identischer Argumentation, dass wir s_0 aus dem Newtonpolygon zur Bewertung ν_{tdeg} und die zugehörigen c_0 aus der charakteristischen Gleichung ablesen können.
Der Algorithmus LocalTypeIn0(L) unterscheidet sich nur dadurch, dass ν_{tdeg} als Bewertung benutzt wird.

4.4.3 Algorithmus

Damit haben wir alle Zutaten zusammen, um den Algorithmus zur Berechnung von Rechtsfaktoren erster Ordnung zu formulieren.

Algorithmus 15: FirstOrderRightFactorHoeij(L)

Input: $0 \neq L = \sum_{i=0}^{n} a_i \sigma^i \in \mathbf{F}[x][\sigma]$, L hat keine rationalen Lösungen
Output: Die Menge aller normierten Rechtsfaktoren erster Ordnung

```
1  F = (f_1, ..., f_l) ← ShiftEquivalenceClasses(a_0 · a_n)
2  P ← LocalTypeAtInfinity(L)
3  Sol ← ∅
4  for all (e_1, ..., e_l) ∈ valbound(L, f_1) × ... × valbound(L, f_l)
5  |  for all (s, c) ∈ P
6  |  |  if 0 = s − Σ_{i=0}^{l} deg(f_i) · e_i
7  |  |  |  r ← Π_{i=0}^{l} f_i^{e_i}
8  |  |  |  r ← c · lc(denom(r))/lc(numer(r)) · r
9  |  |  |  L_1 ← SymmetricProduct(L, σ − 1/r)
10 |  |  |  RS ← RationalSolutions(L_1)
11 |  |  |  for all t ∈ RS
12 |  |  |  |  Sol ← Sol ∪ {σ − σ(t)/(t·r)}
13 return Sol
```

Im q-Fall wird man vor Zeile 7 in Algorithmus 15 noch die entsprechenden Tests für den lokalen Typ in 0 einschieben.

4.4.4 Beispiel

Betrachte den Operator

$$\begin{aligned} L &:= a_2\,\epsilon^2 + a_1\,\epsilon + a_0 \\ &:= \left(q^2 x^4 + (-2\,q^2)\,x^3 + (-q^4 - 3\,q^2)\,x^2 + (2\,q^4)\,x + 3\,q^4\right)\epsilon^2 \\ &+ \left((q^3+q)\,x^5 + (2\,q^3 - q^4 - 2\,q^2)\,x^4 + (1 - 3\,q^3 - 3\,q^4)\,x^3\right. \\ &+ (q^2 - 3\,q^5 - 2\,q + 2)\,x^2 + (2 - 3\,q^2 - 2\,q - 2\,q^3)\,x - (3\,q^4 + 2\,q)\big)\,\epsilon \\ &+ \left(q\,x^6 + q^2 x^5 + x^4 + (3\,q)\,x^3 + (2\,q^2)\,x^2 + 2\,x + 2\,q\right). \end{aligned}$$

Faktorisieren wir a_0 und a_2, so erhalten wir

$$\begin{aligned} a_0 &= (x^3+2)\cdot(q\cdot x^2+1)\cdot(q+x) \\ a_2 &= -q^2\cdot(x+1)\cdot(x-3)\cdot(q-x)\cdot(q+x) \end{aligned}$$

und die vorkommenden Shift-Äquivalenzklassen sind

$$\begin{aligned} &f_1 = x+1, \quad f_2 = x-3, \quad f_3 = (x-q), \\ &f_4 = x^3+2, \quad f_5 = x^2 + 1/q. \end{aligned}$$

Für die möglichen lokalen Typen in den Stellen $[f_1]_\sigma, \ldots, [f_5]_\sigma$ liefert uns dann Satz 4.18:

$$\begin{aligned} V_{[x+1]_\epsilon} &= \{-2,\ldots,1\}, & V_{[x-3]_\epsilon} &= \{-1,\ldots,0\}, \\ V_{[x-q]_\epsilon} &= \{-1,\ldots,0\}, & V_{[x^3+2]_\epsilon} &= \{0,\ldots,1\}, \\ V_{[x^2+1/q]_\epsilon} &= \{0,\ldots,1\}. \end{aligned}$$

Berechnen wir die möglichen lokalen Typen einer Lösung in 0 bzw. ∞:

$$\begin{aligned} P_0 &= \left\{(1,\ 0), \quad \left(\frac{2}{3\,q^3},\ 0\right)\right\}, \\ P_\infty &= \left\{(-1,\ -1), \quad \left(-\frac{1}{q^2},\ -1\right)\right\}. \end{aligned}$$

Nun multiplizieren wir jede Kandidaten-Kombination der f_i aus, wenn die Graddifferenz zwischen Zähler und Nenner nicht -1 oder wenn die Differenz der ν_{tdeg}-Bwertung von Zähler und Nenner nicht 0 ist, kann der zugehörige hypergeometrische Term nicht den gleichen Typ wie eine Lösung haben.

Von den 48 möglichen Produkten bestehen diesen Test nur die folgenden zehn:

$$-\frac{q^2 x^5 + 2\,q^2 x^2 + q\,x^3 + 2\,q}{-3\,q^{11} + q^9 x - 3\,q^8 x + q^6 x^2 + 3\,q^5 x^2 - q^3 x^3 + 3\,q^2 x^3 - x^4},$$

$$-\frac{x^3+2}{q^6 - x^2}, \qquad -\frac{x^3+2}{-3\,q^5 + q^3 x + 3\,q^2 x - x^2}, \qquad \frac{q^2 x^2 + q}{x - q^3},$$

$$-\frac{q^5x^2+q^4+q^2x^3+qx}{-3q^5+q^3x+3q^2x-x^2},\qquad \frac{x^3+2}{q^6+2q^3x+x^2},$$

$$-\frac{x^3+2}{3q^5-q^3x+3q^2x-x^2},\frac{q^2x^2+q}{q^3+x},\qquad \frac{q^2x^2+q}{x-3q^2},\qquad q^3+x$$

Nach Multiplikation mit dem möglichen c_∞ (hier -1 oder $-\frac{1}{q^2}$) muss der Quotient der niedrigsten Koeffizienten noch (bis auf q-Shifts) mit 1 oder $\frac{2}{3q^3}$ übereinstimmen, so dass nur noch die 6 Kandidaten

$$-\frac{q^2x^5+2q^2x^2+qx^3+2q}{-3q^{11}+q^9x-3q^8x+q^6x^2+3q^5x^2-q^3x^3+3q^2x^3-x^4},$$
$$-\frac{x^3+2}{-3q^5+q^3x+3q^2x-x^2},\qquad \frac{q^2x^2+q}{x-q^3}$$
$$-\frac{x^3+2}{3q^5-q^3x+3q^2x-x^2},\qquad \frac{q^2x^2+q}{q^3+x},\qquad q^3+x$$

für $c_\infty \in \{-1,-\frac{1}{q^2}\}$ übrig bleiben. Für jedes dieser $r \in \mathbf{F}(x)$ setzen wir nun $L_{c_\infty,r} := \epsilon - \frac{1}{c_\infty r}$ und berechnen die rationalen Lösungen von $L\circledS L_{c_\infty,r}$. Für

$$L_{(-1),r} = \epsilon + \frac{x^2+\left(q^3-3q^2\right)x-3q^5}{x^3+2}$$

bekommen wir dann die rationale Lösung

$$u = -\frac{x^6+\left(2q^2-2q\right)x^5+(\ldots)x^4+\left(-2q^5-6q^4\right)x^3+\left(-3q^6\right)x^2}{q^5x^6+(q^8-q^7-2q^6)x^5+(\ldots)x^4+(\ldots)x^2+(\ldots)x+(9q^{16})}$$

Das führt uns zu dem rationalen Faktor

$$\epsilon - \frac{\epsilon(u)}{u}\cdot(-1)r = \epsilon + \frac{-x^3-2}{-x^2+(3q-q^2)x+3q^3}.$$

4.5 Der Petkovšek-Algorithmus

Zunächst zitieren wir hier einige Aussagen aus [63], [64], [6].

4.19 Satz. *Jede rationale Funktion* $0 \neq f \in \mathbf{F}(x)$ *kann auf eindeutige Art und Weise geschrieben werden als*

$$r(x) = c\,\frac{A(x)\cdot\sigma(C(x))}{B(x)\cdot C(x)}, \tag{4.5}$$

mit $c \in \mathbf{F}$ *und normierten Polynomen* $A, B, C \in \mathbf{F}[x]$. *Für* A, B, C *gelten*

$$1.\ \gcd(A(x), \sigma^k(B(x))) = 1 \ \textit{für alle}\ k \in \mathbf{N}, \tag{4.6}$$

$$2.\ \gcd(A(x), C(x)) = 1, \tag{4.7}$$

$$3.\ \gcd(B(x), \sigma(C(x))) = 1, \tag{4.8}$$

$$4.\ C(0) \neq 0. \tag{4.9}$$

Die Idee für den Petkovsek-Algorithmus entspricht teilweise dem Vorgehen im Beweis von Satz 2.40.

4.20 Bemerkung. Sei also u eine Lösung von $L(u) = 0$ mit

$$\frac{\sigma(u)}{u} = r = c\,\frac{A(x)\cdot\sigma(C(x))}{B(x)\cdot C(x)}$$

rational wie in (4.5). Dann ist $\sigma(u) = r\,u$ und induktiv

$$\sigma^k(u) = u\prod_{i=0}^{k}\sigma^i(r) = c^k u\prod_{i=0}^{k}\sigma^i\left(\frac{A(x)\cdot\sigma(C(x))}{B(x)\cdot C(x)}\right).$$

Setzen wir dies in $L(u) = 0$ ein, multiplizieren mit dem Hauptnenner und teilen alles durch u, so erhalten wir mit den Bedingungen (4.6)–(4.9)

$$0 = \sum_{i=0}^{n}\underbrace{\left(a_i\,c^i\prod_{j=0}^{i-1}\sigma^j(A)\cdot\prod_{j=i}^{n-1}\sigma^j(B)\right)}_{=:p_i}\sigma^i\left(\frac{\sigma(C(x))}{C(x)}\right) = \sum_{i=0}^{k}p_i\,\sigma^i(C(x)).$$

Bei gegebenen c, A und B ist

$$\sum_{i=0}^{k}p_i\,\sigma^i$$

also eine Rekursionsgleichung, für die C eine polynomielle Lösung ist.

4.21 Bemerkung. Kandidaten für c können bei gegebenen A, B mit Satz 2.40 als Nullstellen der zugehörigen charakteristischen Gleichung berechnet werden.

A bzw. B teilen nach Satz 4.18 a_0 bzw. $\sigma^{-n+1}(a_n)$, es gibt also nur eine endliche Menge von Möglichkeiten.

Offensichtlich ist die Anzahl der polynomiellen Lösungen, die berechnet werden müssen, exponentiell im polynomiellen Grad von a_0 und a_n (bzw. in der Anzahl der Teiler).
Eine wesentliche Erkenntnis dieser Arbeit besteht darin, dass durch die Benutzung der lokalen Typen in ∞ (und im q-Fall zusätzlich in 0) eine Möglichkeit bieten, den Algorithmus massiv zu beschleunigen, da viele Kandidaten für A und B a priori aussortiert werden können.

4.22 Satz. *Sei* $L = \sum_{i=0}^{n} a_i \sigma^i$ *ein Operator der Ordnung* $n > 1$, *seien weiter* $A, B \in \mathbf{F}[x]$ *normiert mit* $A \mid a_0$ *bzw.* $B \mid \sigma^{-n+1}(a_n)$ *und* $c \in \mathbf{F}$. *Hat*

$$0 = \sum_{i=0}^{n} \left(a_i c^i \prod_{j=0}^{i-1} \sigma^j(A) \cdot \prod_{j=i}^{n-1} \sigma^j(B) \right) \sigma^i \tag{4.10}$$

eine polynomielle Lösung $C \in \mathbf{F}[x]$, *so dass*

$$c \frac{A(x) \cdot \sigma(C(x))}{B(x) \cdot C(x)}$$

das Zertifikat einer hypergeometrischen Lösung $u(x)$ *ist. Dann gilt:*

(i) Das Newtonpolygon von L *zur Bewertung* $\nu_{\deg}$ *hat eine Kante mit Steigung* $\deg(A) - \deg(B)$.

Ist $\sigma = \epsilon$ *der* q*-Shift, so gelten außerdem:*

(ii) Das Newtonpolygon von L *zur Bewertung* ν_{tdeg} *hat eine Kante mit Steigung* $\mathrm{tdeg}(B) - \mathrm{tdeg}(A)$.

(iii) $c_0 := \frac{\mathrm{tcoeff}(A)}{\mathrm{tcoeff}(B)}$ *ist* ϵ*-kongruent zu einer Nullstelle der zugehörigen charakteristischen Gleichung.*

Beweis. Die Aussagen folgen aus 2.22 und 2.40, da der Faktor $\frac{\sigma(C(x))}{C(x)}$ die Graddifferenz von Zähler und Nenner nicht verändert. Für den q-Fall wird auch die ν_{tdeg}-Differenz von Zähler und Nenner nicht, und c_0 nur um eine q-Potenz verändert. □

Damit bekommen wir folgenden Algorithmus:

Algorithmus 16: FirstOrderRightFactorPetkovsek(L)

Input: $0 \neq L = \sum_{i=0}^{n} a_i \sigma^i \in \mathbf{F}(x)[\sigma]$, L hat keine rationalen Lösungen
Output: Die Menge aller normierten Rechtsfaktoren erster Ordnung

```
1  A ← {a ∈ F[x] normiert, a teilt a_0}
2  B ← {b ∈ F[x] normiert, b teilt σ^{-n+1}(a_n)}
3  P_∞ ← LocalTypeAtInfinity(L)
4  Sol ← ∅
5  for all a ∈ A
6  |  for all b ∈ B
7  |  |  s_∞ ← deg(a) − deg(b)
8  |  |  P_∞^s ← {(s, c) ∈ P_∞ | s = s_∞}
9  |  |  if 0 < |P_∞^s|
10 |  |  |  for all c ∈ {c | (s, c) ∈ P_∞^s}
11 |  |  |  |  L̃ ← Σ_{i=0}^{n} (a_i c^i Π_{j=0}^{i-1} σ^j(a) · Π_{j=i}^{n-1} σ^j(b)) σ^i
12 |  |  |  |  for all t ∈ PolynomialSolutions(L̃)
13 |  |  |  |  |  Sol ← Sol ∪ {c (a·σ(t))/(b·t)}
14 return Sol
```

Im q-Fall wird man vor Zeile 10 in Algorithmus 16 noch die entsprechenden Tests für den lokalen Typ in 0 einschieben.

4.23 Beispiel. Wir betrachten den gleichen Operator wie im Beispiel für den van-Hoeij-Algorithmus:

$$\begin{aligned} L &:= a_2\,\epsilon^2 + a_1\,\epsilon + a_0 \\ &:= \left(q^2 x^4 + (-2q^2)x^3 + (-q^4 - 3q^2)x^2 + (2q^4)x + 3q^4\right)\epsilon^2 \\ &+ \left((q^3+q)x^5 + (2q^3 - q^4 - 2q^2)x^4 + (1 - 3q^3 - 3q^4)x^3\right. \\ &+ \left.(q^2 - 3q^5 - 2q + 2)x^2 + (2 - 3q^2 - 2q - 2q^3)x - (3q^4 + 2q)\right)\epsilon \\ &+ \left(q x^6 + q^2 x^5 + x^4 + (3q)x^3 + (2q^2)x^2 + 2x + 2q\right). \end{aligned}$$

Auf Grund der Faktorisierung

$$\begin{aligned} a_0 &= (x^3+2)\,(q\,x^2+1)\,(x+q), \\ \epsilon^{-1}(a_2) &= (x+q)\,(-x+q^2)\,(x+q^2)\,(-x+3) \end{aligned}$$

müssten eigentlich $2^3 \cdot 2^4 = 128$ Fälle durchprobiert werden. Nach den Tests bezüglich des lokalen Typs in 0 und ∞ bleiben davon nur dreizehn übrig. Nur für diese müssen die Teiler a und b wirklich ausgerechnet, und der Operator $\tilde{L}$ konstruiert werden.

m	Shift-Fall					q-Shift-Fall				
	Laufzeit			Kandidaten		Laufzeit			Kandidaten	
	A 15	Petk.	A 16	total	red.	A 15	Petk.	A 16	total	red.
1	339	173	84	16	6	1028	720	107	16	2
2	367	165	90	16	7	1589	1882	425	32	6
3	1021	584	274	64	26	54s	16s	1437	128	10
4	1158	1072	618	128	56	54s	44s	1317	256	6
5	2394	2307	1136	256	98	65s	110s	1276	512	5
6	2658	8271	2666	1024	210	3526s	1084s	8s	2048	9
7	6099	24s	11s	2048	792	3623s	2637s	13s	4096	21
8	6s	44s	12s	4096	792	11950s	7933s	22s	8192	22
9	14s	203s	51s	16384	3003	*	53982s	24s	32768	15
10	15s	537s	221s	32768	11440	*	*	76s	65536	32

Laufzeiten in MuPAD für die Faktorisierung des Operators L_m. A 15 bezeichnet `FirstOrderRightFactorHoeij`, „Petk." bezeichnet den original Petkovšek-Algorithmus und A 16 bezeichnet `FirstOrderRightFactorPetkovsek`. * bedeutet, dass die Berechnung nach mehr als 24 Stunden abgebrochen wurde.
In der „total"-Spalte ist die Anzahl der Kandidaten für den Petkovšek-Algorithmus angegeben, „red." ist die Anzahl der Kandidaten, die die Test bezüglich des lokalen Typs in 0 bzw. ∞ überstehen.

Tabelle 4.1: Vergleich Timings für Rechtsfaktoren erster Ordnung.

4.6 Timings

Um die Laufzeiten zu vergleichen, haben wir eine Folge von Operatoren konstruiert, wobei im Shift-Fall

$$L_m := \left(\left(\prod_{i=1}^{\lfloor \frac{m}{2} \rfloor} (x-i) \right) \tau + x^2 + m \right) \cdot \left(\left(\prod_{i=1}^{\lfloor \frac{m+1}{2} \rfloor} \left(x - \frac{i+1}{i} \right) \right) \tau + \prod_{i=1}^{\lfloor \frac{m}{3} \rfloor} (x+i+1) \right),$$

und im q-Shift Fall

$$L_m := \left(\left(\prod_{i=1}^{\lfloor \frac{m}{2} \rfloor} (x-q^i) \right) \epsilon + x^2 + m \right) \cdot \left(\left(\prod_{i=1}^{\lfloor \frac{m+1}{2} \rfloor} \left(x - \frac{i+1}{i} \right) \right) \epsilon + \prod_{i=1}^{\lfloor \frac{m}{3} \rfloor} (x+3\,q^{i+1}) \right).$$

Im Sinne der hier vorgestellten Algorithmen sind dies besonders unerfreuliche Operatoren, da der Leit- und der Fußkoeffizient in viele Linearfakto-

ren zerfallen.
Die Laufzeiten – und der Gewinn der Änderungen am Petkovšek-Algorithmus sind in Tabelle 4.1 zusammengefasst.
Im Shift-Fall ist die Maple-Implementierung hypergeomsols aus dem LREtools-Paket von Mark van Hoeij extrem schnell, für alle Beispiele hier liegt die Laufzeit unter einer Sekunde.
Im q-Shift-Fall wurde auf die Ergenisse von Maple verzichtet, da QHypergeometricSolution aus dem Paket QDifferenceEquations die Lösungen ab $m = 2$ nicht mehr findet, bzw. – bei Benutzung der Option full – eine Fehlermeldung ausgibt.

4.7 Implementierung

4.7.1 Dom::ShiftOperators

MuPAD-Session 8: Rechtsfaktoren erster Ordnung – Shift-Fall

Wir nehmen zwei Operatoren erster Ordnung her, die wir ausmultiplizieren und wieder faktorisieren:

```
>> L1 := R((x^2+1)*S-(x+1)*(x+2)):
>> L2 := R((x^4-1)*S+(x+3)*(x-1)):
>> L := L1*L2;
```

$$\left(x^6+4x^5+7x^4+8x^3+6x^2+4x\right)S^2+$$
$$\left(-x^6-3x^5-x^4+4x^3+2x^2+7x+2\right)S+$$
$$-x^4-5x^3-5x^2+5x+6$$

```
>> R::factorHoeij(L)
```

$$\left\{S+\frac{x+3}{x^3+x^2+x+1}\right\}$$

Das kleinste gemeinsame Linksvielfache sollte beide Operatoren als Rechtsfaktoren haben:

```
>> L3 := R::lclm(L1, L2)
```

$$S^2+\left(-\frac{x^6+11x^5+49x^4+113x^3+141x^2+85x+12}{x^6+8x^5+28x^4+57x^3+72x^2+54x+20}\right)S+$$
$$-\frac{x^5+11x^4+48x^3+105x^2+115x+48}{x^7+6x^6+17x^5+31x^4+38x^3+35x^2+22x+10}$$

```
>> R::factorHoeij(L3)
```

$$\left\{S-\frac{x^2+3x+2}{x^2+1},S+\frac{x+3}{x^3+x^2+x+1}\right\}$$

Ein Beispiel mit höherer Gesamtordnung:

```
>> L4 := R((x+1)*S^5-(x+2)*S+(x-1)):
>> L5 := L4 * L1
```

$$\left(x^3+11\,x^2+36\,x+26\right)\,S^6+\left(-x^3-14\,x^2-55\,x-42\right)\,S^5+$$

$$\left(-x^3-4\,x^2-6\,x-4\right)\,S^2+\left(2\,x^3+6\,x^2+17\,x+11\right)\,S+$$

$$(-x^3-2\,x^2+x+2)$$

```
>> R::factorHoeij(L5)
```

$$\left\{S-\frac{x^2+3\,x+2}{x^2+1}\right\}$$

Natürlich funktioniert der Petkovšek-Algorithmus hier auch:

```
>> R::petkovsek(L5)
```

$$\left\{S-\frac{x^2+3\,x+2}{x^2+1}\right\}$$

4.7.2 Dom::qShiftOperators

MuPAD-Session 9: Rechtsfaktoren erster Ordnung – q-Shift-Fall

Wir nehmen zwei Operatoren erster Ordnung her, die wir ausmultiplizieren und wieder faktorisieren:

```
>> L1 := qR((x^2+q)*Eq-(x+q)*(x+2)):
>> L2 := qR((x^4-q^2)*Eq+(x+3*q)*(x-1)):
>> L := L1*L2;
```

$$\left(q^4\,x^6+q^5\,x^4+\left(-q^2\right)\,x^2-q^3\right)\,Eq^2+$$

$$\left((-q-2)\,x^5-\ldots+\left(4\,q^3+q^2\right)\,x+\left(2\,q^3-3\,q^2\right)\right)\,Eq+$$

$$(-4\,q-1)\,x^3-1\,x^4+\left(2-4\,q-3\,q^2\right)\,x^2+\left(8\,q-3\,q^2\right)\,x+6\,q^2$$

```
>> qR::factorHoeij(L)
```

$$\left\{Eq+\frac{(1-3\,q)\,x-1\,x^2+3\,q}{q^2-1\,x^4}\right\}$$

Das kleinste gemeinsame Linksvielfache sollte beide Operatoren als Rechtsfaktoren haben:

```
>> L3 := qR((x-q)*Eq+x-2):
>> L4 := qR((x+1)*Eq-x+3):
>> L5 := qR::lclm(L3, L4)
```

$$Eq^2+\left(\frac{\left((-q^3-3\,q^2-2\,q)\,x^3+\ldots+(9\,q^2-4)\right)}{\left((-2\,q^2)\,x^4+(q^3+6\,q^2-2\,q)\,x^3+\ldots+(3\,q^2-2\,q)\right)}\right)\,Eq+$$

$$\frac{\left(2\,q^2\right)\,x^4+\ldots+\left(10-39\,q-6\,q^2\right)\,x+\left(18\,q-12\right)}{\left(-2\,q^2\right)\,x^4+\ldots+\left(3\,q^3-6\,q^2-2\,q\right)\,x+\left(3\,q^2-2\,q\right)}$$

≫ qR::factorHoeij(L5)

$$\left\{\mathrm{Eq}+\frac{2-1\,x}{q-1\,x},\mathrm{Eq}+\frac{3-1\,x}{x+1}\right\}$$

Natürlich funktioniert der Petkovšek-Algorithmus hier auch:

≫ qR::qPetkovsek(L5)

$$\left\{\mathrm{Eq}+\frac{2-1\,x}{q-1\,x},\mathrm{Eq}+\frac{3-1\,x}{x+1}\right\}$$

Ein Beispiel mit höherer Gesamtordnung:

≫ L6 := qR((x+1)*Eq^5+Eq^2-(x+1)*(q-x^2)):

≫ L7 := L6*L2

$$\left(q^{20}\,x^5+q^{20}\,x^4+\left(-q^2\right)\,x-q^2\right)\,\mathrm{Eq}^6+$$

$$\left(q^{10}\,x^3+\left(q^{10}+3\,q^6-q^5\right)\,x^2+\left(3\,q^6-q^5-3\,q\right)\,x-3\,q\right)\,\mathrm{Eq}^5+$$

$$\left(q^8\,x^4-q^2\right)\,\mathrm{Eq}^3+\left(q^4\,x^2+\left(3\,q^3-q^2\right)\,x-3\,q\right)\,\mathrm{Eq}^2+$$

$$\left(x^7+x^6+\left(-q\right)\,x^5+\left(-q\right)\,x^4+\left(-q^2\right)\,x^3+\left(-q^2\right)\,x^2+q^3\,x+q^3\right)\,\mathrm{Eq}+$$

$$x^5+\left(3\,q\right)\,x^4+\left(-q-1\right)\,x^3+\left(-3\,q^2-3\,q\right)\,x^2+q\,x+3\,q^2$$

≫ qR::qPetkovsek(L7)

$$\left\{\mathrm{Eq}+\frac{\left(1-3\,q\right)\,x-1\,x^2+3\,q}{q^2-1\,x^4}\right\}$$

Ein wirklich unhandliches Beispiel mit höherer Ordnung. (qR::polyfy multipliziert mit dem gemeinsamen Nenner der Koeffizienten.)

≫ L8 := L6*qR::polyfy(L5)[1]

$$\left(\left(-2\,q^{22}\right)\,x^5+\ldots+\left(3\,q^8-\ldots-2\,q\right)\,x+\left(3\,q^2-2\,q\right)\right)\,\mathrm{Eq}^7+$$

$$\left(\left(-q^{18}-3\,q^{17}-2\,q^{16}\right)\,x^4+\ldots+\left(9\,q^2-4\right)\right)\,\mathrm{Eq}^6+\ldots$$

$$\left(\left(-q^3-3\,q^2-2\,q\right)\,x^6+\ldots+\left(6\,q^3+21\,q^2+2\,q\right)\,x+\left(12\,q-18\,q^2\right)\right)$$

≫ qR::qPetkovsek(L8)

$$\left\{\mathrm{Eq}+\frac{2-1\,x}{q-1\,x},\mathrm{Eq}+\frac{3-1\,x}{x+1}\right\}$$

Diese Berechnung dauert gut 20 Minuten.

Kapitel 5

Faktorisierung

Im letzten Kapitel haben wir gesehen, wie effizient Rechtsfaktoren erster Ordnung berechnet werden können. In diesem Kapitel soll es nun um folgendes gehen:

- *Linksfaktoren berechnen,*
- *Faktoren höherer Ordnung berechnen.*

Die Ideen für dieses Kapitel finden sich – wenngleich nicht in dieser Allgemeinheit – z. B. in [46] und [70].

5.1 Der adjungierte Operator

Hier ist es notwendig, die Fälle $\sigma : x \mapsto x + 1$ und $\sigma : x \mapsto qx$ getrennt zu betrachten, da sich die folgende Abbildung * nicht generisch definieren lässt.

Adjungierter Operator L^*

5.1 Definition. Für einen Operator $L \in \mathbf{F}(x)[\sigma]$ definieren wir den *adjungierten Operator* L^* durch

$$
\begin{aligned}
^* : \quad & \mathbf{F}(x)[\sigma] \longrightarrow \mathbf{F}(x)[\sigma] \\
& \sum_{i=0}^{m} a_i(x)\,\sigma^i \longmapsto \sum_{i=0}^{m} \sigma^i\, a_i(-x) = \sum_{i=0}^{m} a_i(-x-i)\,\sigma^i \quad \text{im Shift-Fall,} \\
& \sum_{i=0}^{m} a_i(x)\,\sigma^i \longmapsto \sum_{i=0}^{m} \sigma^i\, a_i(x^{-1}) = \sum_{i=0}^{m} a_i(q^{-i}x^{-i})\sigma^i \qquad \text{im q-Fall}
\end{aligned}
$$

5.2 Satz. *Die Abbildung* $^* : \mathbf{F}(x)[\sigma] \to \mathbf{F}(x)[\sigma]$ *ist ein involutiver Anti-Automorphismus der Algebra, d.h. für alle* $L, R \in \mathbf{F}(x)[\sigma]$ *gilt*

1. $(L^*)^* = L$,
2. $(L \cdot R)^* = R^* \cdot L^*$.

Beweis. Es ist hinreichend, die Eigenschaften für die Erzeugenden x^k von $\mathbf{F}(x)$ zu zeigen. Zunächst betrachten wir den Shift-Fall. Um zu sehen, dass * involutiv ist, sei $j \in \mathbf{Z}, k \in \mathbf{N}$

$$\begin{aligned}((x^j\sigma^k)^*)^* &= (\sigma^k(-x)^j)^* = ((-x-k)^j\sigma^k)^* \\ &= \sigma^k(x-k)^j = (x+k-k)^j\sigma^k = x^j\sigma^k.\end{aligned}$$

Für die Anti-Automorphismus-Eigenschaft von * betrachte $i, k \in \mathbf{Z}$ und $j, l \in \mathbf{N}$, dann

$$\begin{aligned}(x^i\sigma^j \cdot x^k\sigma^l)^* &= (x^i(x+j)^k\sigma^{j+l})^* = (-x-j-l)^i(-x-l)^k\sigma^{j+l} \\ &= (-x-l)^k\sigma^l \cdot (-x-j)^i\sigma^j = \sigma^l(-x)^k \cdot \sigma^j(-x)^i \\ &= (x^k\sigma^l)^* \cdot (x^i\sigma^j)^*.\end{aligned}$$

Betrachte nun den q-Fall – seien $j \in \mathbf{Z}, k \in \mathbf{N}$, dann ist

$$((x^j\sigma^k)^*)^* = (q^{-jk}x^{-j}\,\sigma^k)^* = q^{-jk}\,\sigma^k\,x^j = q^{-jk}q^{jk}x^j\,\sigma^k = x^j\sigma^k$$

und mit $i, k \in \mathbf{Z}$ und $j, l \in \mathbf{N}$ ist

$$\begin{aligned}(x^i\sigma^j)^* \cdot (x^k\sigma^l)^* &= q^{-(ij+kl+kj)}x^{-(i+k)}\,\sigma^{j+l} \\ &= q^{il}q^{-(ij+il+kj+kl)}x^{-(i+k)}\sigma^{j+l} \\ &= q^{il}q^{-(i+k)(j+l)}x^{-(i+k)}\sigma^{j+l} \\ &= \left(q^{il}x^{i+k}\sigma^{j+l}\right)^* = \left(x^k\sigma^l \cdot x^i\sigma^j\right)^*\end{aligned}$$

□

5.3 Bemerkung. Satz 5.2 erlaubt uns, links und rechts zu vertauschen. Können wir also von einem Operator L Rechtsfaktoren irgendeiner Ordnung berechnen, so könnnen wir durch den Übergang zu L^* auch Linksfaktoren dieser Ordnung berechnen.
Insbesondere sind die Algorithmen 15 `FirstOrderRightFactorHoeij` bzw. 16 `FirstOrderRightFactorPetkovsek` für Rechtsfaktoren erster Ordnung gemeinsam mit der Adjunktion hinreichend, um Operatoren der Ordnung 3 vollständig zu zerlegen.

5.2 Faktoren der Ordnung zwei

Hier stellen wir einen Algorithmus vor, um Rechtsfaktoren der Ordnung zwei eines Operators vierter Ordnung zu berechnen. Dieser Algorithmus lässt sich auf Rechtsfaktoren beliebiger Ordnung von Operatoren beliebiger Ordnung verallgemeinern, was wir in Abschnitt 5.3 tun. Da die Ideen in Ordnung zwei/vier aber besser deutlich werden, behandeln wir zunächst diesen Spezialfall.

Sei

$$L = \sigma^4 + a_3\,\sigma^3 + a_2\,\sigma^2 + a_1\,\sigma + a_0 \in \mathbf{F}(x)[\sigma]$$

normiert von Ordnung 4 und $R = \sigma^2 + \rho_1\sigma + \rho_0$ ein Rechtsfaktor von L der Ordnung 2. Weiter nehmen wir an, dass L keine Rechts- oder Linksfaktoren erster Ordnung hat.
Sei $M := \mathbf{F}(x)[\sigma]/(L)$, dann ist $\dim_{\mathbf{F}(x)} M = 4$ und M ist ein $\mathbf{F}(x)[\sigma]$-Linksmodul mit den Erzeugenden $\sigma^0, \ldots, \sigma^3$.
Die Aktion von σ auf M ist gegeben durch

$$\begin{aligned}
&\sigma^0 \mapsto \sigma^1, \qquad \sigma^1 \mapsto \sigma^2, \qquad \sigma^2 \mapsto \sigma^3,\\
&\sigma^3 \mapsto -a_3\sigma^3 - a_2\sigma^2 - a_1\sigma - a_0\sigma^0,\\
&x \mapsto \sigma(x).
\end{aligned}$$

Sei weiter N der $\mathbf{F}(x)$-Vektorraum, der von $(R \bmod L)$ und $(\sigma R \bmod L)$ erzeugt wird, also

$$N = \{\alpha\,R + \beta\,\sigma\,R \bmod L | \alpha, \beta \in \mathbf{F}(x)\}.$$

Dann gilt:

5.4 Lemma. N *ist ein Untervektorraum von* M *und sogar ein* $\mathbf{F}(x)[\sigma]$*-Linksuntermodul von* M.

Beweis. Dass N ein $\mathbf{F}(x)$-Vektorraum der Dimension zwei ist, ist offensichtlich.
Um zu zeigen, dass N auch ein $\mathbf{F}(x)[\sigma]$-Links-Untermodul von M ist, müssen wir überprüfen, dass für alle $F \in N$ und $G \in \mathbf{F}(x)[\sigma]$, gilt $G \cdot F \in N$. Dazu ist es hinreichend, dies für $\sigma \cdot R \in N$ und $\sigma^2 \cdot R \in N$ nachzurechnen. Für ersteres ist dies ebenfalls offensichtlich.
Für zweiteres sei L' der normierte Operator der Ordnung 2, der durch $L = L'R$ bestimmt ist. $\sigma \cdot \sigma R$ hat Ordnung 4. Reduktion modulo L liefert einen Operator der Ordnung 3. Da sowohl L als auch $\sigma^2 R$ normiert sind, bekommen wir

$$\begin{aligned}
\sigma^2 R \bmod L &= \sigma^2 R - L = \sigma^2 R - L'R = (\overbrace{\sigma^2 - L'}^{\text{Ordnung } 1})R\\
&= (\alpha\sigma + \beta)R = \alpha\sigma R + \beta R
\end{aligned}$$

für $\alpha, \beta \in \mathbf{F}(x)$, was die Aussage beweist. □

Das äußere Produkt $\bigwedge^2 M$ hat die Dimension $\binom{4}{2} = 6$ und die Basis

$$\begin{aligned}
&b_0 := \sigma^0 \wedge \sigma^1, \qquad b_1 := \sigma^0 \wedge \sigma^2, \qquad b_2 := \sigma^0 \wedge \sigma^3,\\
&b_3 := \sigma^1 \wedge \sigma^2, \qquad b_4 := \sigma^1 \wedge \sigma^3, \qquad b_5 := \sigma^2 \wedge \sigma^3.
\end{aligned}$$

Die Aktion von σ auf $\bigwedge^2 M$ ist gegeben durch

$$\begin{aligned}
&b_0 \mapsto b_3, \qquad b_1 \mapsto b_4, \qquad b_3 \mapsto b_5,\\
&b_2 \mapsto \sigma^1 \wedge (-a_3\sigma^3 - a_2\sigma^2 - a_0) = a_0 b_0 - a_2 b_3 - a_3 b_4,\\
&b_4 \mapsto \sigma^2 \wedge (-a_3\sigma^3 - a_1\sigma - a_0) = a_0 b_1 + a_1 b_3 - a_3 b_5,\\
&b_5 \mapsto \sigma^3 \wedge (-a_2\sigma^2 - a_1\sigma - a_0) = a_0 b_2 + a_1 b_4 + a_2 b_5, \quad \text{und}\\
&x \mapsto \sigma(x).
\end{aligned}$$

oder durch die Matrix

$$\mathfrak{A} := \begin{pmatrix}
0 & 0 & a_0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & a_0 & 0\\
0 & 0 & 0 & 0 & 0 & a_0\\
1 & 0 & -a_2 & 0 & a_1 & 0\\
0 & 1 & -a_3 & 0 & 0 & a_1\\
0 & 0 & 0 & 1 & -a_3 & a_2
\end{pmatrix}.$$

5.5 Lemma. *Sei* $w := R \wedge \sigma R$*, dann ist* w *ein Eigenvektor von* σ *bzw.* $\mathfrak{A}$*.*

Beweis. Es gilt

$$\begin{aligned}
\sigma(w) &= \sigma(R \wedge \sigma R) = (\sigma R \wedge \alpha\sigma R + \beta R)\\
&= (\sigma R \wedge \alpha\sigma R) + (\sigma R \wedge \beta R) = (\sigma R \wedge \beta R)\\
&= -\beta(R \wedge \sigma R) = -\beta\, w,
\end{aligned}$$

und damit ist w ein Eigenvektor. □

Drücke nun w durch die Basis $b_0, \ldots, b_5$ aus:

$$\begin{aligned}
w = R \wedge \sigma R &= (\sigma^2 + \rho_1\sigma + \rho_0) \wedge (\sigma^3 + \rho_1\sigma^2 + \rho_0\sigma) && (5.1)\\
&= (\sigma^2 \wedge \sigma^3) + (\sigma^2 \wedge \rho_1\sigma^2) + (\sigma^2 \wedge \rho_0\sigma) && (5.2)\\
&\quad + (\rho_1\sigma \wedge \sigma^3) + (\rho_1\sigma \wedge \rho_1\sigma^2) + (\rho_1\sigma \wedge \rho_0\sigma) && (5.3)\\
&\quad + (\rho_0\sigma^0 \wedge \sigma^3) + (\rho_0\sigma^0 \wedge \rho_1\sigma^2) + (\rho_0\sigma^0 \wedge \rho_0\sigma) && (5.4)\\
&= b_5 + \rho_1 b_4 + (\rho_1^2 - \rho_0) b_3 + \rho_0 b_2 + \rho_0\rho_1 b_1 + \rho_0^2 b_0 && (5.5)
\end{aligned}$$

5.6 Satz. *Sei*

$$v = \sum_{i=0}^{5} \beta_i b_i \in \bigwedge^2 M \quad \textit{mit } \beta_i \in \mathbf{F}(x) \tag{5.6}$$

ein Eigenvektor von σ *bzw.* $\mathfrak{A}$ *und* $\beta_5 = 1,\ \beta_3 = \beta_4^2 - \beta_2,\ \beta_1 = \beta_2\beta_4,\ \beta_0 = \beta_2^2$*, dann ist*

$$\sigma^2 + \beta_4\sigma + \beta_2$$

ein Rechtsfaktor von L*.*

Beweis. Angenommen, wir hätten einen Eigenvektor $\mathfrak{v}$ wie oben berechnet. Wenn dieser durch $\mathfrak{v} = R \wedge \sigma R$ zu einem Rechtsfaktor R korrespondiert, finden wir durch Gleichsetzen (5.5) and (5.6)

$$\beta_5 = 1, \quad \beta_4 = \rho_1, \quad \beta_3 = (\rho_1^2 - \rho_0), \quad \beta_2 = \rho_0, \quad \beta_1 = \rho_0\rho_1, \quad \beta_0 = \rho_0^2.$$

Damit erhalten wir als notwendige und hinreichende Bedingung

$$\left.\begin{array}{ll} \beta_5 & = 1 \\ \beta_3 & = \beta_4^2 - \beta_2 \\ \beta_1 & = \beta_2\beta_4 \\ \beta_0 & = \beta_2^2 \end{array}\right\} \Longrightarrow \beta_0\beta_5 + \beta_2\beta_3 - \beta_1\beta_4 = 0. \tag{5.7}$$

Skalieren wir also $\mathfrak{v}$ so, dass $\beta_5 = 1$, so können wir die Koeffizienten $\rho_0 = \beta_2$ und $\rho_1 = \beta_4$ ablesen, wenn die Bedingungen (5.7) gelten. Die rechte Seite der Implikation in (5.7) ist eine notwendige Bedingung. □

Damit bleibt die Frage zu beantworten, wie man den Eigenvektor von $\mathfrak{A}$ in $\bigwedge^2 M$ berechnet.

5.7 Definition. Wir nennen einen Vektor $\mathfrak{c} \in \bigwedge^2 M$ *zyklisch*, wenn

zyklischer Vektor

$$\langle \mathfrak{c}, \sigma\mathfrak{c}, \sigma^2\mathfrak{c}, \ldots, \sigma^5\mathfrak{c} \rangle = \bigwedge^2 M.$$

Wir zitieren hier ohne Beweis (vgl. etwa [14]):

5.8 Satz. *In $\bigwedge^2 M$ existieren zyklische Vektoren.*

Algorithmus 17: CyclicVector($\bigwedge^2 M$)

Input: $\bigwedge^2 M$
Output: ein zyklischer Vektor $\mathfrak{c}$

1 wähle zufällig $\mathfrak{c} \in \bigwedge^2 M$
2 Wenn $\mathfrak{c}$ nicht zyklisch ist, gehe zu 1
3 return $\mathfrak{c}$

5.9 Bemerkung. Tatsächlich wird man natürlich zunächst keine wirklich zufälligen Vektoren wählen, sondern versuchen, möglichst einfache Vektoren zu bekommen. Entsprechend ist es sinnvoll, mit Basisvektoren zu beginnen, etwa $\mathfrak{b}_0, \mathfrak{b}_1, \mathfrak{b}_2, \ldots$ und dann zu „kleinen" Linearkombinationen überzugehen.
Praktisch hat dieses Vorgehen nach wenigen Schritten (zumeist schon im ersten Versuch) Erfolg.

5.10 Satz und Definition. Sei $\mathfrak{c} \in \bigwedge^2 M$ ein zyklischer Vektor. Es existiert ein eindeutig bestimmter normierter Operator $O \in \mathbf{F}(x)[\sigma]$ der Ordnung 6, so dass $O(\mathfrak{c}) = 0$ in $\bigwedge^2 M$.
Diesen Operator nennen wir den *Minimaloperator* $\operatorname{minop}_{\bigwedge^2 M}(\mathfrak{c})$ *von* $\mathfrak{c}$.

Minimaloperator $\operatorname{minop}_{\bigwedge^2 M}(\mathfrak{c})$

Beweis. Sei c ein zyklischer Vektor, $v_i = \sigma^i(c), 0 \leq i \leq 5$ eine Basis von $\bigwedge^2 M$ und $v_6 = \sigma^6(c)$. Dann existieren nach linearer Algebra eindeutig bestimmte $\alpha_0, \ldots, \alpha_6 \in \mathbf{F}(x)$ mit $\alpha_6 = 1$, so dass $0 = \sum_{i=0}^{6} \alpha_i v_i$. Damit ist

$$\operatorname{minop}_{\bigwedge^2 M}(c) := \sum_{i=0}^{6} \alpha_i \sigma^i.$$

□

Algorithmus 18: MinimalOperator(c)

Input: $c \in \bigwedge^2 M$
Output: $\operatorname{minop}_{\bigwedge^2 M}(c)$

1 Setze $v_i \leftarrow \sigma^i(c)$ für $i = 0, \ldots, 6$
2 Wenn $\{v_0, \ldots, v_5\}$ linear abhängig sind, return fail
3 Berechne $\alpha_0, \ldots, \alpha_5 \in \mathbf{F}(x)$, so dass $v_6 + \sum_{i=0}^{5} \alpha_i v_i = 0$
4 return $\sigma^6 + \alpha_5 \sigma^5 + \ldots + \alpha_0$

5.11 Bemerkung. Um die Koeffizienten möglichst klein zu halten, ist es sinnvoll, nicht nur σ, sondern auch σ^{-1} zu benuzten, und in jedem Schritt in die Richtung des geringeren Koeffizientenwachstums zu „shiften".
Praktisch wird man die Algorithmen A17 und A18 in einem zusammenfassen, denn die Entscheidung ob c zyklisch ist ist im Grunde die Berechnung des Minimaloperators.

5.12 Satz. *Jeder Rechtsfaktor der Ordnung 2 von* L *korrespondiert zu einem Rechtsfaktor erster Ordnung von* $\operatorname{minop}_{\bigwedge^2 M}(c)^*$.

Beweis. Seien also c ein zyklischer Vektor und $v_i = \sigma^i(c), 0 \leq i \leq 5$ eine Basis von $\bigwedge^2 M$ und $N := \operatorname{minop}_L(c)^*$, mit order $N = 6$.
Ist nun $\sigma - r$ mit $r \in \mathbf{F}(x)$ ein Linksfaktor erster Ordnung von N und $P \in \mathbf{F}(x)[\sigma]$ definiert durch $N = (\sigma - r) \cdot P$, so schreibe $P = \sum_{i=0}^{5} \pi_i \sigma^i$ und berechne $v := P(c) = \sum_{i=0}^{5} \pi_i v_i$.
Wir wissen, dass $N(c) = 0$ in $\bigwedge^2 M$, und $N = (\sigma - r) \cdot P$ wobei weder $(\sigma - r)$ noch P identisch null sind. Damit ist auch v ungleich null und $(\sigma - r)(v) = 0$, und damit gilt $\sigma(v) = r \cdot v$ in $\bigwedge^2 M$, und v ist ein Eigenvektor von σ bzw. $\mathfrak{A}$.
Drücke nun v in der Basis b_i aus, also

$$v = \sum_{i=0}^{5} \beta_i b_i \quad \text{mit } \beta_i \in \mathbf{F}[x].$$

Ist nun $\beta_0\beta_5 + \beta_2\beta_3 - \beta_1\beta_4 = 0$ und gelten die Relationen aus (5.7), dann ist nach Satz 5.6

$$K := \sigma^2 + \frac{\beta_4}{\beta_5}\sigma + \frac{\beta_2}{\beta_5}$$

ein Rechtsfaktor zweiter Ordnung von L. □

Algorithmus 19: FactorOrder4(L)

Input: $L \in \mathbf{F}(x)[\sigma]$ von Ordnung 4
Output: Ein Rechtsfaktor zweiter Ordnung R von L, wenn er existiert

```
1  c ← CyclicVector(⋀² M)
2  N ← MinimalOperator(c)
3  Sol ← ∅
4  Für jeden Linksfaktor erster Ordnung L₁ von N
5  |  schreibe N = L₁P
6  |  v ← P(c)
7  |  schreibe v = Σ_{i=0}^{5} β_i b_i
8  |  Wenn β₀β₅ + β₂β₃ − β₁β₄ = 0 und die Bedingungen (5.7) gelten
9  |  |  Sol ← Sol ∪ {σ² + (β₄/β₅)σ + β₂/β₅}
10 return Sol
```

5.2.1 Beispiel

Sei hier $\mathbf{F} = \mathbf{Q}$ und $\tau : x \to x + 1$ der gewöhnliche Shift.

Sei

$$L := \tau^4 + (-x^2 - 5x - 5)\,\tau^3 + (x^3 + 3x^2 + 4x + 5)\,\tau^2 + (-x^3 + x^2 - 5x - 5)\,\tau + (5x - 5).$$

L hat keine polynomiellen, rationalen oder hypergeometrischen Lösungen.

Auf $\bigwedge^2 \mathbf{F}(x)[\tau]/(L)$ wirkt τ folgendermaßen:

$$\begin{aligned}
&\tau^0 \wedge \tau^1 \mapsto \tau^1 \wedge \tau^2, \quad \tau^0 \wedge \tau^2 \mapsto \tau^1 \wedge \tau^3, \quad \tau^1 \wedge \tau^2 \mapsto \tau^2 \wedge \tau^3,\\
&\tau^0 \wedge \tau^3 \mapsto (x^2 + 5x + 5)\tau^1 \wedge \tau^3 + (5x - 5)\tau^0 \wedge \tau^1 \\
&\qquad\qquad - (x^3 + 3x^2 + 4x + 5)\tau^1 \wedge \tau^2,\\
&\tau^1 \wedge \tau^3 \mapsto (x^2 + 5x + 5)\tau^2 \wedge \tau^3 + (5x - 5)\tau^0 \wedge \tau^2 \\
&\qquad\qquad - (x^3 - x^2 + 5x + 5)\tau^1 \wedge \tau^2,\\
&\tau^2 \wedge \tau^3 \mapsto (5x - 5)\tau^0 \wedge \tau^3 - (x^3 - x^2 + 5x + 5)\tau^1 \wedge \tau^3 \\
&\qquad\qquad + (x^3 + 3x^2 + 4x + 5)\tau^2 \wedge \tau^3
\end{aligned}$$

Als zyklischen Vektor wählen wir $c = \tau^0 \wedge \tau^2$ und bekommen

$$\begin{aligned}
&\tau^0 c = \tau^0 \wedge \tau^2, \qquad \tau^1 c = \tau^1 \wedge \tau^3\\
&\tau^2 c = (5x - 5)\,(\tau^0 \wedge \tau^2) + (-x^3 + x^2 - 5x - 5)\,(\tau^1 \wedge \tau^2) \\
&\qquad + (x^2 + 5x + 5)\,(\tau^2 \wedge \tau^3),
\end{aligned}$$

$$\begin{aligned}\tau^3\mathfrak{c} &= (5x^3+30x^2+20x-55)\,(\tau^0\wedge\tau^3)\\ &\quad+(-x^5-6x^4-9x^3-29x^2-85x-55)\,(\tau^1\wedge\tau^3)\\ &\quad+(x^5+10x^4+35x^3+64x^2+73x+45)\,(\tau^2\wedge\tau^3)\\ &\dots\\ \tau^6\mathfrak{c} &\quad\text{hat Koeffizienten-Grad 14 und Polynom-Koeffizienten mit}\\ &\qquad 10\text{ Dezimalen}\end{aligned}$$

Berechnen wir nun den minimalen Operator zu $\mathfrak{c}$. Die Forderung $\sum_{i=0}^{6}\gamma_i\cdot\tau^i\mathfrak{c}=0$ liefert nach Sortieren nach $(\tau^i\wedge\tau^j)$, $i<j$ ein lineares Gleichungssystem für die γ_i:

$$\begin{aligned}\gamma_0 &= -\frac{125x^{17}+4375x^{16}+\ldots+16278582375x^2+5204361250x}{-x^{14}-21x^{13}+\ldots+9589240x^2+4999845x+1096775},\\ \gamma_1 &= -\frac{-25x^{19}-975x^{18}-\ldots+41290651500x^2+7889864000x}{-x^{14}-21x^{13}-\ldots+9589240x^2+4999845x+1096775},\\ &\vdots\qquad\vdots\\ \gamma_5 &= -\frac{-x^{17}-35x^{16}-\ldots+584677420x+114511100}{-x^{14}-21x^{13}-\ldots+9589240x^2+4999845x+1096775},\\ \gamma_6 &= 1\end{aligned}$$

damit ist

$$\operatorname{minop}(\mathfrak{c})=\sum_{i=0}^{6}\gamma_i\tau^i.$$

Der minimale Operator hat einen Linksfaktor erster Ordnung:

$$\tau-\frac{x^{27}-56x^{26}+\ldots+2365600985000x-474637746000}{-x^{26}+53x^{25}+\ldots-1532715812225x+168064317125}$$

Nach Abdividieren dieses Faktors ist

$$\begin{aligned}P=\tau^5&+\left(-\frac{x^{15}+31x^{14}+\ldots-664480x-493675}{x^{12}+20x^{11}+\ldots-9760x-11400}\right)\cdot\tau^4\\ &+\left(\frac{x^{17}+35x^{16}+\ldots-4434480x-4018500}{x^{12}+20x^{11}+\ldots-9760x-11400}\right)\cdot\tau^3\\ &+\left(-\frac{5x^{17}+\ldots+5224025x-4746875}{x^{12}+20x^{11}+\ldots+20560x^2-9760x-11400}\right)\cdot\tau^2\\ &+\left(\frac{25x^{16}+875x^{15}+\ldots+84587000x^2+11191000x}{x^{12}+20x^{11}+\ldots-9760x-11400}\right)\cdot\tau\\ &+\left(-\frac{125x^{14}+3875x^{13}+\ldots-68211375x^2-19154375x}{x^{12}+20x^{11}+\ldots-9760x-11400}\right)\end{aligned}$$

Stellen wir nun $P(\mathfrak{c})$ bezüglich der Basis $\tau^i \wedge \tau^j$ dar, so erhalten wir nach Normierung:

$$\begin{pmatrix} 25 \\ -5x^2 - 10x - 5 \\ \boxed{5} \\ x^4 + 2x^3 + x^2 - 5 \\ \boxed{-x^2} \\ 1 \end{pmatrix}$$

und lesen die Koeffizienten des Rechtsfaktors 2. Ordnung ab:

$$R = \tau^2 \boxed{-x^2} \tau + \boxed{5}.$$

5.3 Faktoren höherer Ordnung

Die Ideen aus dem vorigen Abschnitt lassen sich zu einem vollständigen Faktorisierungsalgorithmus verallgemeinern. Sei

$$L = \sigma^n + a_{n-1}\sigma^{n-1} + \ldots + a_1\sigma + a_0 \in \mathbf{F}(x)[\sigma]$$

Ein Operator der Ordnung $n > 4$, der einen normierten Rechtsfaktor $R = \sum_{i=0}^{r} \rho_i \sigma^i$ der Ordnung $r \leq \lfloor \frac{n}{2} \rfloor$ hat, und sei L' gegeben durch $L = L'R$ mit $l := \operatorname{order} L' = n - r \geq \lfloor \frac{n}{2} \rfloor$.
Setze wieder $M := \mathbf{F}(x)[\sigma]/(L)$, dann ist $\dim_{\mathbf{F}(x)} M = n$ und M ist ein $\mathbf{F}(x)[\sigma]$-Linksmodul.
Sei weiterhin N der $\mathbf{F}(x)$-Vektorraum, der durch $(R \mod L)$, $(\sigma R \mod L), \ldots, (\sigma^{l-1} R \mod L)$ erzeugt wird, also

$$N = \{\alpha_0 R + \alpha_1 \sigma R + \ldots + \alpha_{l-1}\sigma^{l-1} R \bmod L \mid \alpha_0, \ldots, \alpha_{l-1} \in \mathbf{F}(x)\}.$$

5.13 Lemma. N *ist ein Unterraum von* M *und sogar ein* $\mathbf{F}(x)[\sigma]$*-Linksuntermodul von* M.

Beweis. Wie im Beweis von Lemma 5.4 betrachten wir nur $\sigma \cdot \sigma^{l-1}R$:

$$\begin{aligned} \sigma^l R \bmod L = \sigma^l R - L = \sigma^l R - L'R &= (\overbrace{\sigma^l - L'}^{\text{order } l-1})R \\ &= (\alpha_{l-1}\sigma^{l-1} + \ldots + \alpha_1\sigma + \alpha_0)R \end{aligned}$$

mit passenden $\alpha_0, \ldots, \alpha_{l-1} \in \mathbf{F}(x)$, was die Aussage beweist. □

Da $\sigma^l - L'$ Ordnung $l - 1 > 1$ hat, korrespondiert der Faktor nicht zu einem Eigenvektor wie im Ordnung 2-Fall. Betrachten wir die Situation im äußeren Produkt $\bigwedge^l M$ mit der Dimension $\binom{n}{l}$:

Sei $w := R \wedge \sigma R \wedge \ldots \wedge \sigma^{l-1}R \in \bigwedge^l M$, dann ist

$$\begin{aligned}\sigma w &= R \wedge \sigma R \wedge \ldots \wedge \sigma^{l-1}R \\ &= \sigma R \wedge \sigma^2 R \wedge \ldots \wedge (\alpha_{l-1}\sigma^{l-1} + \ldots + \alpha_1\sigma + \alpha_0)R \\ &= (-1)^l \cdot \alpha_0 \cdot (R \wedge \sigma R \wedge \ldots \wedge \sigma^{l-1}R).\end{aligned}$$

Demnach korrespondiert ein Rechtsfaktor der Ordnung r zu einem Eigenvektor in $\bigwedge^l M$, der dann mit der Methode des zyklischen Vektors als Rechtsfaktor erster Ordnung eines Operators $\binom{n}{l}$-ter Ordnung berechnet werden kann.

5.3.1 Bemerkungen zur Implementierung

Die Aktion von σ auf $\bigwedge^l M$ muss im allgemeinen Fall berechnet werden, dazu müssen einige einfache Rechnungen in $\bigwedge^l M$ durchgeführt werden. Außerdem muss Gleichung (5.6) bzw. die Rekonstruktion des Rechtsfaktors aus dem Eigenvektor berechnet werden.
Im q-Fall sind die in Algorithmus 19 `FactorOrder4` auftretenden Ausdrücke so groß, dass MuPAD in vernünftiger Zeit keine Faktorisierungen findet.
In [46] und [70] werden auch Algorithmen vorgestellt, die das Faktorisierungsproblem auf die Lösung eines *Systems* von Gleichungen reduzieren. Perspektivisch sollten diese Algorithmen noch implementiert werden.

5.4 Implementierung

5.4.1 Dom::ShiftOperators

MuPAD-Session 10: Faktorisierung – Shift-Fall

Zunächst experimentieren wir mit dem adjungierten Operator:

```
>> L1 := R(S^2 + (3*x + 1)*S + (-x^2)):
>> L2 := R(S + (x^2 + x - 1)):
>> L := L1*L2
```

$$S^3 + \left(x^2 + 8x + 6\right) S^2 + \left(3x^3 + 9x^2 + 6x + 1\right) S + -x^4 - x^3 + x^2$$

```
>> LL := R::adjoint(L)
```

$$S^3 + \left(x^2 - 4x - 6\right) S^2 + \left(-3x^3 + 3x + 1\right) S + -x^4 + x^3 + x^2$$

```
>> R::rightdivide(LL, R::adjoint(L1))
```

$$\left[S + x^2 - x - 1, 0\right]$$

```
>> LL2 := R::adjoint(L2*L1)
```

$$S^3 + \left(x^2 - 1\right) S^2 + \left(-3x^3 - 6x^2 + x + 2\right) S + -x^4 + x^3 + x^2$$

≫ R::factorHoeij(LL2)

$$\left\{S + x^2 - x - 1\right\}$$

≫ R::adjoint(L2)

$$S + x^2 - x - 1$$

Der Operator aus dem Beispiel-Abschnitt 5.2.1:

≫ L3 := R(S^4 + (- x^2 - 5*x - 5)*S^3 + (x^3 + 3*x^2 + 4*x + 5)*S^2 + (- x^3 + x^2 - 5*x - 5)*S + (5*x - 5))

$$S^4 + \left(-x^2 - 5\,x - 5\right)\,S^3 + \left(x^3 + 3\,x^2 + 4\,x + 5\right)\,S^2 + \left(-x^3 + x^2 - 5\,x - 5\right)\,S + 5\,x - 5$$

≫ R::rightFactorOrder(L3,2)

$$\left\{S^2 + \left(-x^2\right)\,S + 5\right\}$$

≫ R::rightdivide(L3, op(%))

$$\left[S^2 + (-x - 1)\,S + x - 1, 0\right]$$

Nun bauen wir das Produkt zweier Operatoren dritter und zweiter Ordnung zusammen und faktorisieren das Produkt wieder

≫ L4 := R(S^3+(x-1)*S-x+2):

≫ L5 := R(S^2-(x+1)*S+12):

≫ L6 := L4 * L5

$$S^5 + (-x - 4)\,S^4 + (x + 11)\,S^3 + \left(-x^2 - 2\,x + 4\right)\,S^2 + \left(x^2 + 11\,x - 14\right)\,S + 24 - 12\,x$$

≫ R::rightFactorOrder(L6, 2)

$$\left\{S^2 + (-x - 1)\,S + 12\right\}$$

Für diese Faktorisierung sind Geduld, Zeit, ein schneller Rechner und reichlich Speicher gefragt:

≫ L7 := R(S^3 + (- x - 1)*S + 12):

≫ L8 := L4 * L7

$$S^6 - 5\,S^4 + (14 - x)\,S^3 + \left(-x^2 - x + 2\right)\,S^2 + \left(x^2 + 11\,x - 14\right)\,S + 24 - 12\,x$$

≫ R::rightFactorOrder(L8, 3)

$$\left\{S^3 + (-x - 1)\,S + 12\right\}$$

Tatsächlich musste ein Linksfaktor erster Ordnung von einem Operator $\binom{6}{3} = 20$ster Ordnung berechnet werden. Das hat ca. 22 Stunden gedauert und rund 13GB Speicher benötigt.

5.4.2 Dom::qShiftOperators

MuPAD-Session 11: Faktorisierung – q-Shift-Fall

Zunächst experimentieren wir mit dem adjungierten Operator:

```
>> L1 := qR(Eq^2+(q^2+x^2)*Eq+(x+1-q)):
>> L2 := qR((x+q)*Eq-(x^2-q*x-3)):
>> L := L1*L2
```

$$\left(q^2 x + q\right) Eq^3 + \left(q\, x^3 + \left(q - q^4\right) x^2 + \left(2\, q^3\right) x + \left(q^3 + 3\right)\right) Eq^2 +$$

$$\left(\left(-q^2\right) x^4 + q^2 x^3 + \left(4 - q^4\right) x^2 + \left(q^4 + 1\right) x + \left(2\, q^2 + q\right)\right) Eq +$$

$$(2\, q - 1)\, x^2 - 1\, x^3 + \left(-q^2 + q + 3\right) x + (3 - 3\, q)$$

```
>> qR::leftdivide(qR::adjoint(L), qR::adjoint(L2))
```

$$\left[Eq^2 + \left(\frac{(q^4 x^2 + 1)}{(q^2 x^2)}\right) Eq + \frac{(1 - q)\, x + 1}{x}, 0\right]$$

Im q-Fall funktioniert die Berechnung von Rechtsfaktoren zweiter Ordnung auch:

```
>> L2 := qR(Eq^2-q+1):
>> L1 := qR(Eq^2+q*x*Eq-q):
>> L := L2*L1
```

$$Eq^4 + \left(q^3 x\right) Eq^3 + (1 - 2\, q)\, Eq^2 + \left(\left(q - q^2\right) x\right) Eq + q^2 - q$$

```
>> qR::rightFactorOrder(L, 2)
```

$$\left\{Eq^2 + (q\, x)\, Eq - q\right\}$$

```
>> LL := L1*L2
```

$$Eq^4 + (q\, x)\, Eq^3 + (1 - 2\, q)\, Eq^2 + \left(\left(q - q^2\right) x\right) Eq + q^2 - q$$

```
>> qR::rightFactorOrder(LL, 2)
```

$$\left\{Eq^2 + 1 - q\right\}$$

Kapitel 6

Lineare Algebra mit Polynom-Matrizen

Eine der häufigsten Aufaben, die die im Rahmen der in dieser Arbeit vorgestellten Algorithmen benötigen, ist das Lösen linearer Gleichungssysteme über $\mathbf{F}(x)$, wobei $\mathbf{F}$ ein Körper der Charakteristik 0 ist.

Während die grundsätzliche Lösbarkeit solcher Systeme ein wohlbekanntes Thema für jeden Erstsemester-Mathematikstudenten ist, ist der Gauß-Algorithmus in diesem Fall zwar effektiv, aber überhaupt nicht effizient.

In diesem Kapitel werden nun Algorithmen vorgestellt, die das Problem deutlich effizienter lösen.

Konkret war die Motivation für dieses Kapitel, dass es mit MuPAD praktisch unmöglich war, den minimalen Operator mit Algorithmus 18 `MinimalOperator` *(vgl. Seite 84) tatsächlich zu berechnen. Anders als in [73] sind die Ideen hier nicht problemspezifisch, können also bei jeder Art von polynomiellen linearen Gleichungssystemen benutzt werden.*

Ein weiterer Vorteil der hier vorgestellten Interpolationsstrategie ist, dass sie sich kanonisch auf mehrere CPU parallelisieren lässt. Dies wurde allerdings noch nicht implementiert.

Die (naheliegende) Idee, statt auf Evaluation und Interpolation zu setzen, modulo geeigneter Primzahlen zu reduzieren, war in MuPAD leider langsam, auch wenn die Idee theoretisch vielversprechend ist. Deswegen gehen wir auf diesen Ansatz nicht weiter ein.

Seien im Folgenden stets $1 < n \in \mathbf{N}$ und

$$M := \begin{pmatrix} m_{1,1} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n} \end{pmatrix} \in \mathbf{F}[x]^{n \times n} \text{ mit } m_{i,j} \in \mathbf{F}[x].$$

Der Vergleich der Algorithmen erfolgte auf einem MacPro mit 3GHz CPUs und 16GB Speicher, die genutzten Systeme sind MuPAD 4.0.6 bzw. die aktuelle Entwicklerversion sowie Maple 11 und Mathematica 6.

6.1 Determinante

6.1 Lemma. *Mit den Definitionen von oben gilt*

$$\deg(\det M) \le \min\Big\{\sum_{i=1}^{n} \max\{\deg(m_{i,1}),\dots,\deg(m_{i,n})\},\ \sum_{i=1}^{n} \max\{\deg(m_{1,i}),\dots,\deg(m_{n,i})\}\Big\}.$$

Beweis. Nach Definition gilt

$$\det M = \sum_{\sigma\in S_n} (-1)^{\operatorname{sgn}(\sigma)} \prod_{i=1}^{n} m_{i,\sigma(i)} \in \mathbf{F}[x],$$

damit ist

$$\deg(\det M) \le \max_{\sigma\in S_n}\Big\{\sum_{i=1}^{n} m_{i,\sigma(i)}\Big\} \le \sum_{i=1}^{n} \max\{\deg(m_{i,1}),\dots,\deg(m_{i,n})\},$$

da die Determinante sich durch Transponieren nicht ändert, gilt außerdem $\deg(\det M) \le \sum_{i=1}^{n} \max\{\deg(m_{1,i}),\dots,\deg(m_{n,i})\}$. □

Im Allgemeinen ist die Schranke scharf, auch wenn sich natürlich andere Beispiele konstruieren lassen.

Algorithmus 20: DetDegreeBound(M)

Input: $M := (m_{i,j})_{1\le i,j\le n} \in \mathbf{F}[x]^{n\times n}$ mit $m_{i,j} \in \mathbf{F}[x]$
Output: $B \in \mathbf{N}$ mit $\deg(\det M) \le B$

```
1  B1 ← Σ_{i=1}^{n} max{deg(m_{i,1}), ..., deg(m_{i,n})}
2  B2 ← Σ_{i=1}^{n} max{deg(m_{1,i}), ..., deg(m_{1,i})}
3  return min{B1, B2}
```

Bezeichne die Schranke aus Lemma 6.1 mit B, dann ist also $\det(M)$ ein Polynom vom Grad maximal B, welches wir durch Interpolation berechnen können, indem wir den Einsetzungshomomorphismus für $a \in \mathbf{F}$

$$\varphi_a : \mathbf{F}[x] \to \mathbf{F}, \quad f \mapsto f(a)$$

auf den Matrizenring erweitern:

$$\varphi_a : \mathbf{F}[x]^{n\times n} \to \mathbf{F}^{n\times n}, \quad (m_{i,j})_{1\le i,j\le n} \mapsto (m_{i,j}(a))_{1\le i,j\le n}.$$

Da die Abbildung $\det : \mathbf{F}[x]^{n\times n} \to \mathbf{F}[x]$ nur Multiplikationen und Additionen benutzt, kommutiert sie mit φ_a, es gilt also

$$\det(\varphi_a(M)) = \varphi_a(\det(M)).$$

Im Folgenden verwenden wir zur besseren Lesbarkeit für den Einsetzungshomomorphismus $\varphi_a(M)$ die suggestivere Schreibweise $M(a)$.

Sei nun $A = (a_0, \ldots, a_B) \in \mathbf{F}^{B+1}$ mit $|\{a_0, \ldots, a_B\}| = B + 1$ und

$$\varphi_A : \mathbf{F}[x] \to \mathbf{F}^{B+1}, \quad f \mapsto (f(a_0), \ldots, f(a_B)).$$

und $\iota : \mathbf{F}^{B+1} \to \mathbf{F}[x]$ die Interpolationsabbildung, dann kommutiert das folgende Diagramm, da der Grad der Determinante $\leq B$ ist:

$$\begin{array}{ccc} \mathbf{F}[x]^{n\times n} & \xrightarrow{\det} & \mathbf{F}[x] \\ \downarrow{\scriptstyle \varphi_A} & & \uparrow{\scriptstyle \iota} \\ (\mathbf{F}^{n\times n})^{B+1} & \xrightarrow{\det^{B+1}} & \mathbf{F}^{B+1} \end{array} \tag{6.1}$$

Wir können also $\det(M)$ berechnen, indem wir M an $B + 1$ Stellen auswerten, die entsprechenden *numerischen* Determinanten ausrechnen, und dann das Interpolationspolynom durch $\Big((a_0, \det(M(a_0))), \ldots, (a_B, \det(M(a_B))\Big)$ bestimmen.
Das liefert nun folgenden Algorithmus:

Algorithmus 21: PolyDet(M)

Input: $M := (m_{i,j})_{1\leq i,j\leq n} \in \mathbf{F}[x]^{n\times n}$ mit $m_{i,j} \in \mathbf{F}[x]$
Output: $\det(M) \in \mathbf{F}[x]$

```
1  B ← DetDegreeBound(M)
2  Wähle A = (a_0, ..., a_B) ∈ F^{B+1} mit i ≠ j ⇒ a_i ≠ a_j
3  S ← ∅
4  for i ← 0, ..., B
5  |  S ← S ∪ {(a_i, det(M(a_i)))}
6  Berechne d ← Interpolationspolynom durch Punkte von S
7  return d
```

6.2 Beispiel. Sei

$$M := \begin{pmatrix} 1 & x^2 \\ x^2+1 & x-2 \end{pmatrix},$$

dann ist die Schranke für den Grad der Determinante $B = 4$, wir benötigen also 5 Auswertungspunkte, wähle $A := (0, -1, 1, 2, -2)$, das liefert die Matrizen

$$M_0 = \begin{pmatrix} 1 & 0 \\ 1 & -2 \end{pmatrix}, \qquad M_1 = \begin{pmatrix} 1 & 1 \\ 2 & -1 \end{pmatrix}, \qquad M_2 = \begin{pmatrix} 1 & 1 \\ 2 & -3 \end{pmatrix},$$
$$M_3 = \begin{pmatrix} 1 & 4 \\ 5 & 0 \end{pmatrix}, \qquad M_4 = \begin{pmatrix} 1 & 4 \\ 5 & -4 \end{pmatrix},$$

mit den Determinanten

$$d_0 = -2, \quad d_1 = -3, \quad d_2 = -5, \quad d_3 = -20, \quad d_4 = -24$$

Das Interpolationspolynom durch die Punkte $(0, -2)$, $(1, -3)$, $(-1, -5)$,

$(2, -20)$ und $(-2, -24)$ ist

$$-x^4 - x^2 + x - 2 = 1 \cdot (x - 2) - (x^2 + 1) \cdot x^2.$$

Dieser Algorithmus lässt sich natürlich auf Matrizen mit rationalen Funktionen als Einträgen erweitern:

Algorithmus 22: RatDet(M)

Input: $M := (m_{i,j})_{1 \le i,j \le n} \in \mathbf{F}(X)^{n \times n}$ mit $m_{i,j} \in \mathbf{F}(X)$
Output: $\det M \in \mathbf{F}(x)$

```
1  M̃ ← (0)_{1≤i,j≤n} ∈ F[x]
2  for i = 1, ..., n
3  |  d_i ← lcm(denom(m_{i,1}), ..., denom(m_{i,n}))
4  |  for j ← 1, ..., n
5  |  |  Setze den i, j-Eintrag von M̃ auf d_i · m_{i,j} ∈ F[x]
6  d̃ ← PolyDet(M̃)
7  d ← d̃ / (d_1 ··· d_n)
8  return d
```

Diese algorithmische Idee lässt sich auf andere Invarianten wie z. B. auf das charakteristische Polynom wörtlich übertragen.

Komplexität

6.3 Satz. *Die Berechnung der Determinante einer Matrix* $M \in \mathbf{F}[x]^{n \times n}$*, deren Einträge Grad* $\le d$ *haben, erfolgt mit Algorithmus 21* PolyDet *in* $\mathcal{O}(n^4 d^2)$ *Operationen in* $\mathbf{F}$*.*

Beweis. Die Berechnung der Gradschranke kann in $\mathcal{O}(n)$ Schritten erfolgen, es gilt $B \le nd$. Die Auswertung an den Stellen a_i kostet mit dem Hornerschema jeweils $\mathcal{O}(n^2 d)$ Operationen. Die Berechnung der "numerischen" Determinanten kostet $\mathcal{O}(n^3)$ Operationen. Die Berechnung des Interpolationspolynoms erfolgt in $\mathcal{O}((nd)^2)$ Operationen.
Das liefert insgesamt $\mathcal{O}(n + nd \cdot (n^2 d + n^3) + (nd)^2) = \mathcal{O}(n^3 d^2 + n^4 d + (nd)^2) = \mathcal{O}(n^4 d^2)$ Operationen. □

Timimgs. Zum Vergleich sind die Zeiten in ms für $d \times d$-Matrizen mit zufälligen polynomiellen Einträgen des angegebenen Grades gegenübergestellt in Tabelle 6.1.

Für die Berechnung der Determinante existiert in Maple die Option `method=univar`, mit der ein identischer Ansatz benutzt wird. Auch in Mathematica kann die Berechnung der Determinante mit dem modularen Ansatz drastisch beschleuningt werden.

d	Grad 3				Grad 10				Grad 20			
	sparse		dense		sparse		dense		sparse		dense	
	MuP	A 21	MuP	A 21	MuP	A 21	MuP	A 21	MuP	A 21	MuP	A 21
3	5	7	12	12	7	35	40	44	7	66	84	107
4	61	29	80	25	88	71	244	97	105	154	554	259
5	132	42	190	43	265	144	644	176	387	354	1376	500
6	294	67	386	70	657	240	1332	311	1067	620	3015	931
7	462	104	727	109	1384	429	2581	519	2213	1047	5827	1677
8	846	144	1217	165	2546	687	4419	844	4732	1944	10385	2893
9	1439	233	1959	244	4363	1047	7324	1329	8897	3289	17s	4898
10	2280	331	2988	350	7287	1669	11s	2049	15s	5261	28s	7970
11	2882	472	4373	490	11s	2629	17s	3092	26s	8646	43s	12s
12	4645	633	6181	676	18s	3903	25s	4541	39s	15s	65s	19s
13	6976	875	8524	922	26s	5504	35s	6586	60s	23s	95s	29s
14	9940	1180	11s	1241	38s	7983	49s	9390	95s	35s	134s	43s
15	10s	1526	15s	1646	50s	11s	68s	13s	124s	48s	184s	63s
20	41s	5370	50s	5724	187s	50s	247s	56s	538s	263s	722s	313s
25	102s	15s	132s	16s	590s	164s	708s	189s	1796s	966s	2407s	1151s
30	268s	37s	295s	40s	1493s	482s	1688s	521s	4544s	3035s	5468s	3292s
35	527s	84s	591s	87s	3204s	1164s	3623s	1247s	10043s	7506s	13296s	8180s
40	1014s	167s	1098s	175s	6403s	2540s	7218s	2679s	21679s	16997s	25360s	18099s

d ist die Dimension der Matrix, der Grad gibt den maximalen polynomiellen Grad der Einträge an. Im sparse-Fall haben die Polynome zwei, im dense-Fall $d + 1$ Koeffizienten. MuP bezeichnet den in MuPAD implementierten Determinanten-Algorithmus, A 21 bezeichnet den Algorithmus `PolyDet`.

Tabelle 6.1: Vergleich Timings für Determinanten-Berechnung.

6.2 Lineare $n \times n$ Gleichungssysteme

Sei nun weiterhin $1 < n \in \mathbf{N}$ und

$$M := \begin{pmatrix} m_{1,1} & \cdots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n} \end{pmatrix} \in \mathbf{F}[x]^{n\times n}, \qquad b := \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \in \mathbf{F}[x]^n$$

mit $m_{i,j}, b_i \in \mathbf{F}[x]$. Weiter fordern wir $\det(M) \neq 0$.

Wir wollen nun den Interpolationsansatz benutzen, um das Gleichungssystem

$$M \cdot Y = b \tag{6.2}$$

für einen Vektor $Y = (Y_1, \ldots, Y_n)^t \in \mathbf{F}(x)^n$ zu lösen.

Die Lösung des Systems (6.2) kann mit Hilfe der Cramerschen Regel in geschlossener Form angegeben werden:

$$Y_i = \frac{\det(M_i)}{\det(M)}, \tag{6.3}$$

wobei M_i aus M dadurch entsteht, dass die i-te Spalte von M durch b ersetzt wird.
Damit haben wir mit Lemma 6.1 a priori Schranken für die Zähler- und Nennergrade jeder Lösung. Ist nun B eine obere Schranke für den Grad aller Zähler und des Nenners, so können wir – ebenfalls wie oben – $A = (a_0, \ldots, a_B) \in \mathbf{F}^{B+1}$ mit $|\{a_0, \ldots, a_B\}| = B+1$ wählen, wobei wir die weitere Bedingung $\det(M)(a_i) \neq 0$ für die Lösbarkeit benötigen.
Bezeichne mit $Y = (Y_1, \ldots, Y_n)^t$ die Lösung von (6.2) und mit $Y^{(i)} = (Y_1^{(i)}, \ldots, Y_n^{(i)})$ die Lösung des „modularen" Systems $M(a_i) \cdot Y^{(i)} = b(a_i)$. Dann gilt $Y_i(a_j) = Y_i^{(j)}$ für alle a_j. Anders als im Fall der Determinante können wir hier aber nicht direkt Polynominterpolation anwenden, da die Koeffizienten der Lösung aus $\mathbf{F}(x)$ sind.
Sei nun etwa $\mathbf{F} = \mathbf{Q}$, dann können wir für alle $1 \leq i \leq n$, $0 \leq j \leq B$ schreiben

$$Y_i^{(j)} = \frac{N_i^{(j)}}{D_i^{(j)}}, \quad Y_i = \frac{N_i}{\det(M)} \quad \text{mit teilerfremden } N_i^{(j)},\ D_i^{(j)} \in \mathbf{Z},\ N_i \in \mathbf{Z}[X].$$

Dann ist aber nicht notwendigerweise $N_i^{(j)} = N_i(a_j)$ und $D_i^{(j)} = \det(M)(a_j)$, aber es gilt sicher $D_i^{(j)} \mid \det(M)(a_j)$.
Dann ist

$$N_i(a_j) = \underbrace{N_i^{(j)} \cdot \frac{\det(M)(a_j)}{D_i^{(j)}}}_{=:\tilde{N}_i^{(j)}} \in \mathbf{Z}.$$

Damit können wir also die „isolierten" Zähler $\tilde{N}_i^{(j)}$ ausrechnen und mit Hilfe von Polynominterpolation zusammensetzen:

Algorithmus 23: PolyLinearSolve(M, b)

Input: $M := (m_{i,j})_{1 \leq i,j \leq n} \in \mathbf{F}[x]^{n \times n}$, $\det(M) \neq 0$, $b = (b_1, \ldots, b_n)^t$ mit $m_{i,j}, b_i \in \mathbf{F}[x]$
Output: $Y \in \mathbf{F}(x)^n$ mit $M \cdot Y = b$

```
1  B ← max{DetDegreeBound(M_1), ..., DetDegreeBound(M_n)}
2  D ← PolyDet(M)
3  Wähle A = (a_0, ..., a_B) ∈ F^{B+1} mit i ≠ j ⇒ a_i ≠ a_j und D(a_j) ≠ 0
4  S_1 ← ∅, ..., S_n ← ∅
5  for i ← 0, ..., B
6  |  Berechne Lösung y von M(a_i) · y = b(a_i)
7  |  for j ← 1, ..., n
8  |  |  S_j ← S_J ∪ { (a_i, numer(y_j) · D(a_i)/denom(y_i)) }
9  for j ← 1, ..., n
10 |  Berechne Y_j ←Interpolationspolynom durch Punkte von S_j
11 return (Y_1/D, ..., Y_n/D)
```

Die genaue Form der erhaltenen Lösung hängt natürlich von der Wahl des

Interpolationsalgorithmus ab. In MuPAD ist eine Newton-Interpolation implementiert, deren Resultat in expandierter Form zurückgeliefert wird.

6.4 Beispiel. Betrachte

$$M := \begin{pmatrix} 1 & x^2 \\ x^2+1 & x-2 \end{pmatrix} \quad \text{und} \quad b := \begin{pmatrix} x-1 \\ x+1 \end{pmatrix}.$$

Die Schranke für den Grad der Nenner der Lösung ist $B = 3$, Die Determinante von M ist $D = -x^4 - x^2 + x - 2$, wie wir in Beispiel 6.2 berechnet haben.
Wähle $A = (0, 1, -1, 2, -2)$, D hat in keinem dieser Punkte eine Nullstelle. Nun lösen wir die fünf Systeme

$$\begin{aligned}
a_0 = 0: &\quad \begin{pmatrix} 1 & 0 \\ 1 & -2 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix} && \Rightarrow y_1 = -1,\ y_2 = -1 \\
a_1 = 1: &\quad \begin{pmatrix} 1 & 1 \\ 2 & -1 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \end{pmatrix} && \Rightarrow y_1 = \frac{2}{3},\ y_2 = -\frac{2}{3} \\
a_2 = -1: &\quad \begin{pmatrix} 1 & 1 \\ 2 & -3 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \end{pmatrix} && \Rightarrow y_1 = -\frac{6}{5},\ y_2 = -\frac{4}{5} \\
a_3 = 2: &\quad \begin{pmatrix} 1 & 4 \\ 5 & 0 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \end{pmatrix} && \Rightarrow y_1 = \frac{3}{5},\ y_2 = \frac{1}{10} \\
a_4 = -2: &\quad \begin{pmatrix} 1 & 4 \\ 5 & -4 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} -3 \\ -1 \end{pmatrix} && \Rightarrow y_1 = -\frac{2}{3},\ y_2 = -\frac{7}{12}
\end{aligned}$$

Die Determinanten an den Punkten aus A sind:

$$D(0) = -2, \quad D(1) = -3, \quad D(-1) = -5,$$
$$D(2) = -20, \quad D(-2) = -24$$

Die Interpolationspolynome durch $\{(0,2), (1,-2), (-1,6), (2,12), (-2,16)\}$ bzw. $\{(0,2), (1,2), (-1,4), (2,-2), (-2,-14)\}$ sind

$$Y_1 = -x^3 - 3x + 2, \qquad Y_2 = -x^3 + x^2 + 2,$$

und damit ist der gesuchte Lösungsvektor

$$\begin{pmatrix} \frac{x^3+3x-2}{x^4+x^2-x+2} \\ \frac{x^3-x^2-2}{x^4+x^2-x+2} \end{pmatrix}$$

Komplexität

6.5 Satz. *Seien* $M \in \mathbf{F}[x]^{n \times n}$, $b \in \mathbf{F}[x]^n$ *mit polynomiellen Einträgen vom Grad maximal* $d \in \mathbf{N}$. *Dann berechnet Algorithmus 23* `PolyLinearSolve` *die Lösung*

Dimension			4	6	8	10	15	20	25	30	40
Grad 5	nativ	MuPAD	173	850	2823	8261	*	*	*	*	*
		Maple	18	95	378	1296	41s	609s	2476s	*	*
		Mathem.	11	133	2023	31s	*	*	*	*	*
	Alg. 23	MuPAD	203	573	1338	2794	12s	43s	134s	312s	1342s
		Maple	78	189	431	892	4288	15s	42s	100s	412s
		Mathem.	17	26	64	142	608	1880	4701	10s	37s
Grad 15	nativ	MuPAD	545	2820	9660	29s	*	*	*	*	*
		Maple	79	535	3211	16s	1227s	5528s	*	*	*
		Mathem.	19	230	3722	57s	*	*	*	*	*
	Alg. 23	MuPAD	726	2415	6547	15s	98s	439s	1481s	3914s	20405s
		Maple	468	1822	5573	14s	99s	387s	1118s	2538s	11266s
		Mathem.	47	161	420	1080	5229	17s	48s	112s	445s

Vergleich für verschiedene Matrixdimensionen und Polynomgrade, die polynomiellen Einträge haben stets 5 Terme. Bei den *-Einträgen sind MuPAD, Maple bzw. Mathematica am 2 GB- (MuPAD), 4 GB- (Maple) bzw. 16 GB-Speicherlimit (Mathematica) gescheitert und mit einem Out-Of-Memory Error ausgestiegen. Alg 23 bezeichnet `PolyLinearSolve`.

Tabelle 6.2: Vergleich Timings für nicht-homogene, quadratische LGS

$Y \in \mathbf{F}[x]^n$ *von*

$$M \cdot Y = b$$

mit $\mathcal{O}(n^4\, d^2)$ *Operationen in* **F**.

Beweis. Die Berechnung der Determinante D benötigt $\mathcal{O}(n^4 d^2)$ Operationen, für die Wahl der Evaluationspunkte in Zeile 23 müssen die Nullstellen von D ausgeschlossen werden, das kann effizient durch Probieren erreicht werden.
Dann müssen $B + 1 \leq nd + 1$ lineare Gleichungssysteme über **F** gelöst werden, was insgesamt $\mathcal{O}(nd \cdot n^3)$ Operationen kostet.
Die Berechnung der n Interpolationspolynome kostet $\mathcal{O}(n \cdot (nd)^2)$ Operationen.
In der Summe erhält man also $\mathcal{O}(n^4 d^2 + n^4\, d + n^3\, d^2) = \mathcal{O}(n^4 d^2)$ Operationen in **F**. □

Timings Zum Vergleich sind die Zeiten in ms für $d \times d$-Matrizen mit zufälligen polynomiellen Einträgen des angegebenen Grades in Tabelle 6.2 gegenübergestellt.
„nativ“ bezeichnet in MuPAD `matlinsolve`, in Maple `linsolve` und in Mathematica `LinearSolve`. Die Implementierungen von Algorithmus 23 in Maple und Mathematica sind prototypisch und verzichten auf ausgefeilte Fehlerbehandlung etc.
Die Implementierungen finden sich in den Verzeichnissen „Maple“ bzw. „Mathematica“ auf der beiliegenden CD.

6.3 Homogene lineare Gleichungssysteme

Gegeben sei eine Matrix $M \in \mathbf{F}[x]^{n\times k}$ mit $n < k$, gesucht die Basis des Kernes. Algorithmus 23 `PolyLinearSolve` können wir hier nicht anwenden, da wir für die Nenner der Lösung zwingend die Determinante benötigen. Es bleiben also zwei Möglichkeiten:
Die Matrix durch

- Hinzufügen linear unabhängiger Zeilen oder
- Streichen linear abhängiger Spalten

regulär zu machen.

6.3.1 Regularisierung durch Hinzufügen linear unabhängiger Zeilen

Wir nehmen an, dass $\operatorname{rank}(M) = n$ ist. Seien $m_1, \ldots, m_n$ die Zeilenvektoren von M. Ist nun $m_{k-n+1}, \ldots, m_k$ eine Basis von $\operatorname{Kern} M$, so sind $m_1, \ldots, m_n, m_{k-n+1}, \ldots, m_k$ linear unabhängig. Wir können also M zu einer quadratischen Matrix mit vollem Rang ergänzen, indem wir die Zeilen $m_{k-n+1}, \ldots, m_k$ zu M hinzufügen.
Jetzt benutzen wir das folgende Lemma:

6.6 Lemma. *Sei* $a \in \mathbf{F}$ *so gewählt, dass* $\operatorname{rank}(M(a)) = \operatorname{rank}(M) = n$ *gilt. Seien ferner* $\tilde{m}_{k-n+1}, \ldots, \tilde{m}_k$ *eine Basis des Kerns von* $M(a)$*. Dann hat*

$$\tilde{M} := (m_1, \ldots, m_n, \tilde{m}_{k-n+1}, \ldots, \tilde{m}_k)^t$$

vollen Rang.

Beweis. Die Vektoren $\{m_1, \ldots, m_n, \tilde{m}_{k-n+1}, \ldots, \tilde{m}_k\}$ sind die Urbilder der linear unabhängigen Vektoren $\{m_1(a), \ldots, m_n(a), \tilde{m}_{k-n+1}, \ldots, \tilde{m}_k\}$ unter dem Einsetzungshomomorphismus.
Da die Urbilder linear unabhängiger Vektoren unter Homomorphismen stets linear unabhängig sind, folgt die Aussage. □

Damit sind die $(k-n)$ linearen Gleichungssysteme

$$\tilde{M} \cdot Y = b_j, \quad b_j = (\underbrace{0, \ldots, 0}_{n\text{-fach}}, \underbrace{0, \ldots, 0}_{j-1\text{-fach}}, 1, \underbrace{0, \ldots, 0}_{k-n-j\text{-fach}})^t, \quad 1 \le j \le k-n \tag{6.4}$$

quadratisch, nicht-homogen und eindeutig lösbar.

6.7 Satz. *Die Lösungen der* $(n-k)$ *Systeme* (6.4) *spannen den Kern von* M *auf.*

Beweis. Für jedes $1 \leq j \leq k-n$ erfüllt die Lösung $u_j \in \mathbf{F}(x)^k$ von $\tilde{M} \cdot u_j = b_j$ die Gleichung $M \cdot u_j = 0$ nach Konstruktion.
Andererseits steht u_j orthogonal auf allen $\tilde{m}_i$ mit $i \neq j$, demnach müssen die u_j linear unabhangig sein.
Da der Kern nach Voraussetzung genau Dimension $(k-n)$ hat, haben wir eine Basis des Kerns gefunden. □

Das liefert folgenden Algorithmus:

Algorithmus 24: PolyLinearNullspaceAR(M)

Input: $M = (m_{i,j})_{\substack{1 \leq i \leq n \\ 1 \leq j \leq k}} \in \mathbf{F}[x]^{n \times k}$, $n < k$, $\operatorname{rank}(M) = n$
Output: $u_1, \ldots, u_{k-n} \in \mathbf{F}(x)^n$ mit $\langle u_1, \ldots, u_{k-n} \rangle = \operatorname{Kern}(M)$

```
1  Wähle a ∈ F mit rank(M(a)) = n
2  Berechne m̃_{k-n+1}, ..., m̃_k Basis von Kern(M(a))
3  M̃ ← füge m̃_{k-n+1}, ..., m̃_k als Zeilen zu M hinzu
4  for i ← 1, ..., k - n
5  |  b_i ← (δ_{j,n+i})^t_{1≤j≤k}
6  |  u_i ← PolyLinearSolve(M̃, b_i)
7  return (u_1, ..., u_{n-k})
```

6.8 Bemerkung. Für eine wirkliche Implementierung ergeben sich einige offensichtliche Verbesserungen für eine spezielle Version von PolyLinearSolve:

(i) Die Evaluationspunkte müssen nur einmal gewählt werden.

(ii) Die Determinante von $\tilde{M}$ muss nur einmal berechnet werden.

(iii) Beim ersten Aufruf von PolyLinearSolve$(\tilde{M}, b_1)$ kann an jedem Evaluationspunkt eine LR-Zerlegung berechnet werden, so dass die weiteren Lösungen schneller berechnet werden können.

(iv) Auf die Division durch die Determinante im letzten Schritt von PolyLinearSolve kann verzichtet werden. Damit bekommt man eine Basis des Nullraumes mit polynomiellen Einträgen.

6.3.2 Regularisierung durch Streichen linear abhängiger Spalten

Nachdem die Ergebnisse mit Algorithmus 24 PolyLinearNullspaceAR zwar erfreulich sind, aber doch hinter den Erwartungen zurückblieben, haben wir nach einem Weg gesucht, die Systeme kleiner zu halten.
Sei also $M \in \mathbf{F}[x]^{n \times k}$ mit $n < k$ und Rang n und seien weiter ohne Einschränkung die ersten n Spalten von M linear unabhängig. Bezeichne mit $\tilde{M} \in \mathbf{F}[x]^{n \times n}$ die reguläre Matrix aus diesen Spalten und mit $\tilde{b}_1, \ldots, \tilde{b}_{k-n}$ die Vektoren aus den weiteren Spalten der Matrix M. Die $k-n$ linearen, quadratischen Gleichungssyteme

$$\tilde{M}\, y = \tilde{b}_1, \quad \ldots, \quad \tilde{M}\, y = \tilde{b}_{k-n} \tag{6.5}$$

sind demnach eindeutig lösbar.

Sind $y_j = (y_{j,1}, \ldots, y_{j,n})^t$ für $j = 1, \ldots, k-n$ die Lösungen von (6.5), dann sind

$$u_j = (y_{j,1}, \ldots, y_{j,n}, 0, \ldots, 0, \underbrace{-1}_{\text{j-te Pos.}}, 0, \ldots, 0)^t$$

im Kern von M. Da die u_j (wegen der -1 an den Positionen $n+j$) linear unabhängig sind, spannen sie den Kern von M auf.

Einen Teil der Verbesserungen aus Bemerkung 6.8 führen wir hier explizit durch.

Algorithmus 25: PolyLinearNullspaceDC(M)

Input: $M = (m_{i,j})_{\substack{1 \le i \le n \\ 1 \le j \le k}} \in \mathbf{F}[x]^{n \times k}$, $n < k$, $\operatorname{rank}(M) = n$
Output: $u_1, \ldots, u_{k-n} \in \mathbf{F}(x)^n$ mit $\langle u_1, \ldots, u_{k-n} \rangle = \operatorname{Kern}(M)$

```
B ← 1 + Σ_{i=1}^{n} max{deg(m_{i,j}) | j = 1, ..., k}
I ← {i_1, ..., i_n} charakteristische Spaltenindizes
Ĩ = {ĩ_1, ..., ĩ_{k-n}} ← {1, ..., k} \ I
M̃ ← (m_{j,i_k})_{1≤j,k≤n},   b̃_ĩ ← Spaltenvektoren ĩ ∈ Ĩ
J ← 0
for l ← 1, ..., B
    repeat
        a_l ← (−1)^J ⌊(J+1)/2⌋
        J ← J + 1
        d_l ← det(M̃(a_l))
    until d_l ≠ 0
    L_l, R_l, P_l ← LR-Zerlegung von M̃(a_l)
    for j ← 1, ..., k − n
        y_{j,l} ← Lösung von M(a_l) y = −b̃_{ĩ_j}, benutze dabei L_l, R_l, P_l
d ← Interpolationspolynom durch {(a_1, d_1), ..., (a_B, d_B)}
for j ← 1, ..., k − n
    ỹ ← komponentenweises Interpolationspolynom durch
                                        {(a_1, y_{j,1}), ..., (a_B, y_{j,B})}
    t ← 0^n
    for k ← 1, ..., n
        t_{i_k} ← ỹ_k
    t_{ĩ_j} ← d
    u_j ← t
return (u_1, ..., u_{n−k})
```

Timings. Zum Vergleich sind die Zeiten in ms für Matrizen mit zufälligen polynomiellen Einträgen des angegebenen Grades in Tabelle 6.3 gegenübergestellt.

d	MuPAD nativ		MuPAD Alg. 24		MuPAD Alg. 25		Maple nativ	
	$k = 1$	$k = 3$	$k = 1$	$k = 3$	$k = 1$	$k = 3$	$k = 1$	$k = 3$
4	283	19	260	114	179	108	17	6
5	775	162	502	298	332	257	21	7
6	1739	650	891	629	585	520	63	14
7	3502	1713	1493	1230	984	944	123	49
8	6747	3697	2433	2026	1566	1576	247	97
9	11s	7457	3750	3309	2453	2511	437	211
10	19s	13s	5639	5148	3693	3767	762	388
11	31s	22s	8232	7611	5461	5627	1331	653
12	48s	37s	11s	10s	7825	8100	2088	1160
13	71s	60s	18s	17s	11s	12s	3217	1854
14	104s	87s	25s	22s	16s	17s	4809	2951
15	150s	127s	32s	30s	22s	21s	6977	4569
16	204s	178s	43s	39s	29s	29s	9969	6929
17	274s	246s	57s	51s	38s	38s	14s	9713
18	380s	337s	75s	67s	50s	50s	20s	15s
19	507s	456s	97s	87s	66s	66s	30s	24s
20	661s	648s	125s	111s	85s	84s	46s	33s
25	2225s	2216s	399s	338s	268s	264s	176s	138s
30	5684s	5583s	1009s	847s	704s	686s	552s	426s
35	12911s	12087s	2322s	1899s	1616s	1571s	1492s	1153s
40	27722s	26878s	4811s	3838s	3334s	3250s	3716s	2820s

Berechnung einer Basis des Kernes einer $d \times d$-Matrix, die Dimension des Kernes ist k.

Tabelle 6.3: Vergleich Timings für Homogene LGS

6.4 Matrizen über multivariaten Polynomringen

Die Ideen aus den vorigen Abschnitten lassen sich mit übersichtlichem Aufwand auf den multivariaten Fall erweitern. Die auftretenden Besonderheiten sollen hier kurz besprochen werden.
Sei dazu $M \in \mathbf{F}[x_1, \ldots, x_r]^{n \times n}$, wobei die Polynome maximal Grad d_i in x_i haben.

(1) Man bekommt r Schranken $B_i \leq n\, d_i$.

(2) Die „modulare“ Auswertung erfolgt auf einem r-dimensionalen Gitter.

(3) Die Komplexität ist exponentiell in r.

(4) Man kann nicht mehr a priori entscheiden, ob die Determinante an bestimmten Punkten des Gitters verschwindet, die Suche nach guten Evaluationspunkten kann bei unglücklichen Nullstellen aufwendig werden.

6.5 Implementierung

MuPAD-Session 12: Lineare Algebra, das liinalg-domain

Wir konstruieren eine Matrix mit zufälligen polynomiellen Einträgen

```
>> r := () -> expr(polylib::randpoly([x], Degree=5, Terms=3)):
>> rM := (n,m,r) -> matrix([[r() $ i=1..m] $ j=1..n]):
>> M := rM(3,3,r)
```

$$\begin{pmatrix} x-724\,x^4-101\,x^2+534 & -480\,x^4-522\,x-286 & 892\,x^2-367\,x-393 \\ x203\,x^5-930\,x+773 & 904\,x^4-791\,x^2+59 & 343\,x^5-695\,x^2+397\,x \\ x-383\,x^4-643\,x^2+636 & 412\,x^5-175\,x^4+397\,x & -716\,x^5-631\,x^2+556x \end{pmatrix}$$

```
>> liinalg::det(M)
```

$$\text{poly}(32545744\,x^{14}+488218156\,x^{13}+88876228\,x^{12}-651916253\,x^{11}+1150455233\,x^{10}-$$
$$738172011\,x^9-949036140\,x^8+1330271901\,x^7-1577574901\,x^6-511013506\,x^5+$$
$$1716714251\,x^4+574987801\,x^3-970622156\,x^2-02581181\,x+155183636,[x])$$

```
>> b := rM(3,1,r);
```

$$\begin{pmatrix} 391\,x^5-228\,x^2-717\,x \\ 11\,x^5-720\,x^2-316 \\ 546\,x^2+78\,x+508 \end{pmatrix}$$

```
>> y := liinalg::matlinsolve(M, b)
```

$$\begin{pmatrix} -\frac{55254556\,x^{15}+233390929\,x^{14}-369666728\,x^{12}+5840616\,x^{11}-\ldots+38470060}{32545744\,x^{14}+488218156\,x^{13}+88876228\,x^{12}-651916253\,x^{11}+\ldots+155183636} \\ -\frac{-56831068\,x^{15}+45663055\,x^{14}+68494164\,x^{12}+459407057\,x^{11}-\ldots+327130044}{32545744\,x^{14}+488218156\,x^{13}+88876228\,x^{12}-651916253\,x^{11}+\ldots+155183636} \\ -\frac{-32701676\,x^{15}+10609107\,x^{14}-136005252\,x^{13}+18611276\,x^{12}+\ldots-185791808}{32545744\,x^{14}+488218156\,x^{13}+88876228\,x^{12}-651916253\,x^{11}+\ldots+155183636} \end{pmatrix}$$

```
>> normal(M*y-b)
```

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Jetzt ein homogenes System:

```
>> M2 := rM(3,5,r):
>> y2 := liinalg::nullspace(M2):
>> map(op(y2), y->normal(M2*y))
```

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

```
>> M2 := linalg::stackMatrix(M2, linalg::row(M2,2)):
>> y2 := liinalg::nullspace(M2):
>> map(op(y2), y->normal(M2*y))
```

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Index

Literaturverzeichnis

[1] ABRAMOV, S. A.: *Rational Solutions of linear Difference and* q*-Difference Equations with Polynomial Coefficients*. In: *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, 1995.

[2] ABRAMOV, S. A., KH. K. LE und Z. LI: *Ore Polynomial Rings in one Variable in Computer Algebra*. Sovrem. Mat. Prilozh., (13, Algebra):24–39, 2004.

[3] ABRAMOV, SERGEI A. und M. A. BARKATOU: *Rational Solutions of First Order Linear Difference Systems*. In: *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, Seiten 124–131, 1998.

[4] ABRAMOV, SERGEI A., M. A. BARKATOU und MARK VAN HOEIJ: *Apparent Singularities of Linear Difference Equations with Polynomial Coefficients*. Applicable Algebra in Engineering, Communication and Computing, 17(2):117–133, 2006.

[5] ABRAMOV, SERGEI A., MANUEL BRONSTEIN und MARKO PETKOVSEK: *On Polynomial Solutions of Linear Operator Equations*. In: *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, 1995.

[6] ABRAMOV, SERGEI A., PETER PAULE und MARKO PETKOVŠEK: q*-Hypergeometric Solutions of* q*-Difference Equations*. In: *Proceedings of the 7th Conference on Formal Power Series and Algebraic Combinatorics (Noisy-le-Grand, 1995)*, Band 180, Seiten 3–22, 1998.

[7] ABU SALEM, FATIMA, SHUHONG GAO und ALAN G. B. LAUDER: *Factoring Polynomials via Polytopes*. In: *Proceedings of the 2004 International Symposium on Symbolic and Algebraic Computation*, Seiten 4–11. ACM, New York, 2004.

[8] ADAMS, C. RAYMOND: *On The Irregular Cases of the Linear Ordinary Difference Equation*. Transactions of the American Mathematical Society, 30(3):507–541, 1928.

[9] ADAMS, C. RAYMOND: *On the linear ordinary* q*-Difference Equation*. Ann. of Math. (2), 30(1-4):195–205, 1928/29.

[10] ADAMS, C. RAYMOND: *Linear* q*-Difference Equations*. Bulletin of the American Mathematical Society, 37:361–400, 1931.

[11] APEL, J. und U. KLAUS: *Computer Algebra Handbook*, Kapitel 4.2.15. Springer-Verlag, 2002.

[12] ARNOLD, V.I.: *Mathematical methods of classical mechanics.* Nummer 60 in *Graduate Texts in Mathematics.* Springer-Verlag, 2. Auflage, 1997.

[13] BARKATOU, A. und ANNE DUVAL: *Sur les séries formelles solutions d'équations aux différences polynomiales*. Annales de l'institut Fourier, 44(2):495–524, 1994.

[14] BARKATOU, M. A.: *An Algorithm for computing a Companion Block Diagonal Form for a System of Linear Differential Equations*. Appl. Algebra Engrg. Comm. Comput., 4(3):185–195, 1993.

[15] BARKATOU, M. A. und F. RICHARD-JUNG: *Formal Solutions of Linear Differential and Difference Equations*. Programmirovamie (a Journal of the Russian Academy of Science), (1), 1997.

[16] BEKE, EMANUEL: *Die Irreducibilität der homogenen linearen Differentialgleichung*. Mathematische Annalen, Seiten 278–294, 1894.

[17] BEKE, EMANUEL: *Die symmetrischen Functionen bei den linearen homogenen Differentialgleichungen*. Mathematische Annalen, Seiten 295–300, 1894.

[18] BIRKHOFF, GEORGE D.: *Formal Theory of irregular linear Difference Equations*. Acta Math., 54(1):205–246, 1930.

[19] BJÖRK, JAN-ERIK: *Rings of Differential Operators*. North-Holland mathematical library. North-Holland Publishing Company, 1979.

[20] BÖING, HARALD und WOLFRAM KOEPF: *Algorithms for* q*-hypergeometric Summation in Computeralgebra*. J. Symbolic Comput., 28(6):777–799, 1999. Orthogonal polynomials and computer algebra.

[21] BRONSTEIN, M. und M. PETKOVŠEK: *An Introduction to Pseudo–Linear Algebra*. Theoretical Computer Science, 157:3–33, 1996.

[22] BRONSTEIN, MANUEL: *Linear ordinary Differential Equations: Breaking through the Order 2 Barrier*. In: *Proceedings of the 1992 International Symposium on Symbolic and Algebraic Computation*, Seiten 42–48. ACM Press, 1992.

[23] BRONSTEIN, MANUEL: *An improved Algorithm for Factoring linear ordinary Differential Operators*. In: *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, Seiten 336–340. ACM Press, 1994.

[24] BRONSTEIN, MANUEL und MARKO PETKOVSEK: *On Ore Rings, Linear Operators and Factorization*. Programming and Computer Science, 20(1):14–26, 1994.

[25] BUESO, JOSÉ L., JOSÉ GÓMEZ-TORRECILLAS und ALAIN VERSCHOREN: *Algorithmic Methods in Non-Commutative Algebra: Applications to Quantum Groups*. Kluwer, July 2003.

[26] CHYZAK, FRÉDÉRIC: *Fonctions holonomes en calcul formel*. Doktorarbeit, École polytechnique, 1998. INRIA, TU 0531. 227 pages.

[27] CHYZAK, FRÉDÉRIC und BRUNO SALVY: *Non-commmutative Elimination in Ore Algebras proves multivariate Identities*. Journal of Symbolic Computation, 26(2):187–227, 1998.

[28] CLUZEAU, THOMAS: *Factorization of Differential Systems in Characteristic* p. In: *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation*, Seiten 58–65. ACM Press, 2003.

[29] CLUZEAU, THOMAS und MARK VAN HOEIJ: *A Modular Algorithm to Compute the Exponential Solutions of a Linear Differential Operator*. Journal of Symbolic Computation, 38:1043–1076, 2004.

[30] CLUZEAU, THOMAS und MARK VAN HOEIJ: *Computing Hypergeometric Solutions of Linear Recurrence Equations*. Applicable Algebra in Engineering, Communication and Computing, 17:83–115, 2006.

[31] COHN, PAUL. M.: *Free Rings and their Relations*, Band 19 der Reihe *London Mathematical Society Monographs*. Academic Press Inc. [Harcourt Brace Jovanovich Publishers], London, Second Auflage, 1985.

[32] COHN, PAUL M.: *Skew Fields, Theory of General Division Rings*. Cambridge University Press, 1995.

[33] DUVAL, ANNER: *Lemmes de Hensel et Factorisation Formelle pour les Opérateurs aux Différences*. Funkcialaj Ekvacioj, 26:349–368, 1983.

[34] FREDET, ANNE: *Factorization of linear Differential Pperators in Exponential Extensions*. In: *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation*, Seiten 103–110. ACM Press, 2003.

[35] GAO, SHUHONG: *Factoring Multivariate Polynomials via Partial Differential Equations*. Mathematics of Computation, 2002. to appear.

[36] GATHEN, JOACHIM VON ZUR und JÜRGEN GERHARD: *Modern Computer Algebra*. Cambridge University Press, Cambridge, UK, 1999.

[37] GELFAND, I. M. und S. V. FOMIN: *Calculus of Variations*. Revised English edition translated and edited by Richard A. Silverman. Prentice-Hall Inc., Englewood Cliffs, N.J., 1963.

[38] GIESBRECHT, MARK: *Factoring in Skew-Polynomial Rings over Finite Fields*. Journal of Symbolic Computation, 1998.

[39] GIESBRECHT, MARK und YANG ZHANG: *Factoring and Decomposing Ore Polynomials over* $\mathbf{F}_q(t)$. In: *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation*, Seiten 127–134. ACM Press, 2003.

[40] GOODEARL, K. R. und R. B. WARFIELD, JR.: *An Introduction to Noncommutative Noetherian Rings*, Band 16 der Reihe *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 1989.

[41] GRAHAM, R. L., D. E. KNUTH und O. PATASHNIK: *Concrete Mathematics*. Addison Wesley, 1989.

[42] HOEIJ, MARK VAN: *Factorization of Linear Differential Operators*. Doktorarbeit, Universiteit Nijmegen, 1996.

[43] HOEIJ, MARK VAN: *Factrization of Differential Operators with Rational Function Coefficients*. Journal of Symbolic Computation, 24:537–561, 1997.

[44] HOEIJ, MARK VAN: *Formal Solutions and Factorization of Differential Operators with Power Series Coefficients*. Journal of Symbolic Computation, 1997.

[45] HOEIJ, MARK VAN: *Finite Singularities and Hypergeometric Solutions of Linear Recurrence Equations*. Journal of Pure and Applied Algebra, Seiten 109–131, 1998.

[46] HOEIJ, MARK VAN: *Factorization and Hypergeometric Solutions of linear Recurrence Systems*. Slides for a Talk ath the Manuel Bronstein's conference, July 2006.

[47] JACOBSON, NATHAN: *The Theory of Rings*. Mathematical Surveys. American Mathematical Society, 2nd Auflage, 1943.

[48] JACOBSON, NATHAN: *Finite-dimensional Division Algebras over Fields*. Springer-Verlag, Berlin, 1996.

[49] KAUERS, MANUEL and CHRISTOPH KOUTSCHAN: *A Mathematica Package for q-holonomic Sequences and Power Series*. Technical Report 2007-16, SFB F013, 2007.

[50] KEDLAYA, KIRAN S.: p-*adic Differential Equations*, 2007.

[51] KNUTH, DONALD E.: *The Art of Computer Programming, vol.2, Seminumerical Algorithms*. Addison Wesley, 2nd edition, 1997.

[52] KOEPF, WOLFRAM: *Hypergeometric Summation, An Algorithmic Approach to Summation and Special Function Identities*. Advanced Lectures in Mathematics. Friedr. Vieweg & Sohn, Braunschweig, 1998.

[53] KOEPF, WOLFRAM: *Computeralgebra, Eine algorithmisch orientierte Einführung*. Springer-Verlag, 2006.

[54] KOLCHIN, ELLIS R.: *Differential Algebra and Algebraic Groups*. Academic Press, 1973.

[55] KOVACIC, JERALD J.: *An Algorithm for Solving Second Order Linear Homogeneous Differential Equations*. Journal of Symbolic Computation, (2):3–43, 1986.

[56] LANG, SERGE: *Algebra*. Addison Wesley, 3rd edition, 1999.

[57] LEVANDOVSKYY, VIKTOR and HANS SCHÖNEMANN: *Plural: a Computer Algebra System for Noncommutative Polynomial Algebras*. In *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation*, pages 176–183. ACM Press, 2003.

[58] MELENK, H. and J. APEL: *Reduce package NCPOLY: Computation in Non-Commutative Polynomial Ideals.* Technical report, Konrad-Zuse-Zentrum Berlin (ZIB), 1994.

[59] ORE, OYSTEIN: *Theory of Non-Commutative Polynomials*. The Annals of Mathematics, 34:480–508, 1993.

[60] PAULE, PETER: *Greatest Factorial Factorization and Symbolic Summation*. Journal of Symbolic Computation, 20:235–268, 1995.

[61] PERDRY, HERVÉ: *Henselian valued Fields: a Constructive Point of View*. MLQ Math. Log. Q., 51(4):400–416, 2005.

[62] PESCH, MICHAEL: *Gröbner Bases in Skew Polynomial Rings*. Shaker Verlag, 1997.

[63] PETKOVŠEK, MARKO: *Hypergeometric Solutions of Linear Recurrences with Polynomial Coefficients*. J. Symbolic Comput., 14(2-3):243–264, 1992.

[64] PETKOVSEK, MARKO and BRUNO SALVY: *Finding all Hypergeometric Solutions of Linear Differential Equations*. In *Proceedings of the 1993 International Symposium on Symbolic and Algebraic Computation*, 1993.

[65] PETKOVŠEK, MARKO, HERBERT S. WILF, and DORON ZEILBERGER: A = B. A K Peters Ltd., Wellesley, MA, 1996. With a foreword by Donald E. Knuth, With a separately available computer disk.

[66] PIRASTU, ROBERTO and VOLKER STREHL: *Rational summation and Gosper-Petkovšek representation*. J. Symbolic Comput., 20(5-6):617–635, 1995. Symbolic computation in combinatorics Δ_1 (Ithaca, NY, 1993).

[67] PUT, MARIUS VAN DER and MARC REVERSAT: *Galois Theory of* q*-Difference Equations*. n/a, 2007.

[68] PUT, MARIUS VAN DER and MICHAEL F. SINGER: *Galois Theory of Linear Differential Equations*. To Appear, 2002.

[69] PUT, MARIUS VAN DER and MICHAEL F. SINGER: *Galois Theory of Difference Equations*, volume 1666 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1997.

[70] PUT, MARIUS VAN DER and MICHAEL F. SINGER: *Galois Theory of Linear Differential Equations*, volume 328 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 2003.

[71] RAMIS, JEAN-PIERRE: *Théorèmes d'indices Gevrey pour les équations différentielles ordinaires*. Memoirs of the American Mathematical Society, 48(269), 1984.

[72] RIESE, A.: *Contributions to Symbolic q-Hypergeometric Summation*. PhD thesis, RISC, J. Kepler University Linz, 1997.

[73] RIESE, AXEL: *Fine-Tuning Zeilberger's Algorithm: The Methods of Automatic Filtering and Creative Substituting*. Symbolic Computation, Number Theory, Special Functions, Physics and Combinatorics, 4:243–254, 2001.

[74] ROBBA, P.: *Lemmes de Hensel pour les opérateurs différentiels. Application à la réduction formelle des équations différentielles*. L'Enseignement Mathématique, 26(3-4):279–311, 1980.

[75] ROWEN, LOUIS H.: *Ring Theory. Vol. I*, volume 127 of *Pure and Applied Mathematics*. Academic Press Inc., Boston, MA, 1988.

[76] ROWEN, LOUIS H.: *Ring Theory. Vol. II*, volume 128 of *Pure and Applied Mathematics*. Academic Press Inc., Boston, MA, 1988.

[77] SCHWARZ, FRITZ: *A Factorization Algorithm for linear ordinary Differential Equations*. In *Proceedings of the ACM-SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation*, pages 17–25. ACM Press, 1989.

[78] SINGER, MICHAEL: *Testing Reducibility of linear Differential Operators: a Group-Theoretic Perspective*. Appl. Algebra Engrg. Comm. Comput., 7(2):77–104, 1996.

[79] SOMMELING, RON: *Characteristic Classes for irregular Singularities*. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, pages 163–168. ACM Press, 1994.

[80] SPRENGER, TORSTEN: *Algorithmen für* q*-holonome Funktionen und* q*-hypergeometrische Reihen*. PhD thesis, Universität Kassel, 2008 (to appear).

[81] TSAREV, S. P.: *An Algorithm for complete Enumeration of all Factorizations of a linear ordinary Differential Operator*. In *Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation*, pages 226–231. ACM Press, 1996.

[82] TSAREV, S. P.: *Factorization of linear partial Differential Operators and Darboux Integrability of nonlinear PDEs*. ACM Press SIGSAM Bulletin, 32(4):21–28, 1998.

[83] VAN DER PUT, MARIUS and MARC REVERSAT: *Galois Theory of* q*-Difference Equations*, 2005.

[84] WEIXLBAUMER, C.: *Solutions of Difference Equations With Polynomial Coefficients*. Master's thesis, RISC, J. Kepler University Linz, 2001.

[85] YU.GRIGOR'EV, D.: *Complexity of Factoring and Calculation the GCD of Linear Ordinary Differential Operators*. Journal of Symbolic Computation, 10:7–37, 1990.

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Dissertation selbständig und ohne unerlaubte Hilfe angefertigt und andere als die in der Dissertation angegebenen Hilfsmittel nicht benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen sind, habe ich als solche kenntlich gemacht. Kein Teil dieser Arbeit ist in einem anderen Promotions- oder Habilitationsverfahren verwendet worden.

Kassel, den 2. April 2008