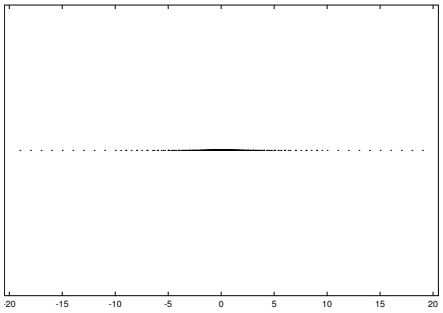
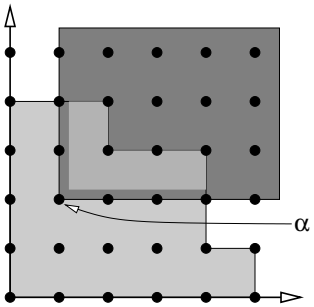


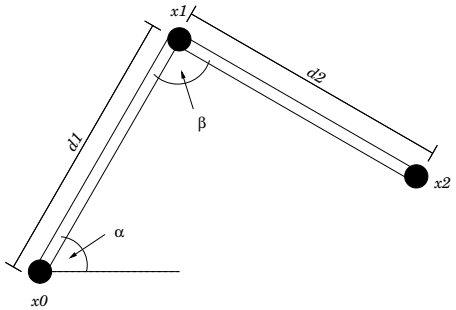


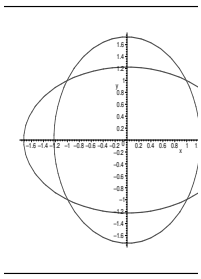
Computeralgebra

Vorlesung, zuerst gehalten im Sommersemester 2001

Tomas Sauer

Version 1.2
Letzte Änderung: 9.11.2010





Statt einer Leerseite . . . | 0

Lasciate ogni speranza, voi ch' entrate [. . .]
Vuolsi così colà dove si puote
ciò che si vuole, e più non dimandare

Laßt, die Ihr eingeht, alle Hoffnung fahren [. . .]
Wo eins ist das Vollbringen und Verlangen,
Dort will man's also! Und nicht weiter frage.

Dante, *La divina comedia, Inferno*, Canto III
Aus (Laaths, 1994), Deutsch von R. Zoozmann.

0 Inhaltsverzeichnis

When a company offers a superior quality 10 megabyte Winchester hard disk for only $695, it's bound to raise a few eyebrows . . .

Anzeige in *nibble*, September 1985

1 Computeralgebra und ein paar Anwendungen

Neben der "normalen" numerischen Rechnung auf Computern hat sich in den letzten Jahren dank der dramatisch gesteigerten Leistungsfähigkeit von Computern auch mehr und mehr die Verwendung von Computeralgebra–Systemen etabliert, wobei die verbreitetsten[1] wohl `Maple` und `Mathematica` sein dürften, man aber zumindest in Deutschland auch noch `MuPad` und `Singular` erwähnen sollte. Kurz ein Wort zu den Systemen:

`Maple` (`www.maplesoft.com`) ist wohl inzwischen das erfolgreichste Computeralgebra–Paket. `Maple` enstand ursprünglich an der University of Waterloo und wird dort auch noch weiterentwickelt, ist aber ein rein kommerzielles Produkt, das heißt, es gibt keine "offiziellen" Erweiterungen oder Ergänzungen ohne dafür zu bezahlen.

`Mathematica` (`www.wolfram.com`) ist später als `Maple` entstanden und war von vornherein als vollständig kommerzielles Produkt ("Wolfram Research") angelegt. Daher gibt es auch keine "offizielle" Bindung an staatliche Forschungseinrichtungen, auch wenn mehr als nur lose Beziehungen zum RISC[2] bestehen. `Mathematica` war von vornherein "bunter" als `Maple` angelegt und ist weitgehend dafür verantwortlich, daß heute Grafik, Sound und Animation als "wesentliche" Bestandteile von Computeralgebrasystemen angesehen werden.

`MuPAD` (`www.mupad.de`) stammt von der Universität Paderborn und wird - zumindest in Binärversion - wohl immer noch frei verteilt, zumindest konnte man es auf den SuSE–Linux–CDs bis Version 6.4[3] noch finden. Die neue Version, 2.0, ist nicht nur recht stabil, sondern glänzt auch durch eine überarbeitete Benutzeroberfläche; dafür ist es aber nicht ganz klar, ob und in welchem Rahmen die Software noch frei verfügbar ist. Eine Besonderheit von `MuPAD` besteht darin, daß es Funktionen für den Einsatz auf verteilten

[1] Hier werden mir die italinischen Computeralgebraiker, die ein System namens `CoCoA` geschaffen haben, wohl widersprechen.

[2] **R**esearch **I**nstitute for **S**ymbolic **C**omputations in Schloß Hagenberg bei Linz, gehört zur Johannes–Kepler–Universität Linz und wird/wurde geleitet von B. Buchberger, dem Erfinder der Gröbnerbasen (Buchberger, 1965).

[3] Lang, lang ist's (mittlerweile) her.

Systemen und Parallelrechnern besitzt, dafür gibt es aber (zumindest bei älteren Versionen) immer noch Schwächen im algebraischen Bereich. Inzwischen[4] gibt es `MuPAD` nicht mehr als eigenständige und frei erhältliche Software, sondern es wurde in `Matlab` eingegliedert und unterliegt daher auch denselben lizenzrechtlichen Restriktionen.

`Singular` (`www.singular.uni-kl.de`) ist eine Entwicklung der Computeralgebra–Gruppe an der Universität Kaiserslautern (Greuel & Pfister, 2002) und hat viele der moderneren algebraischen Konzepte integriert. Es ist in `C++` geschrieben und scheint auch eine `C++`–Klassenbibliothek zu bieten, ein Service, den Programmierer, die effiziente algebraische Routinen in ihre Anwendung integrieren wollen, sicher zu schätzen wissen. Allerdings besitzt Singular *keine* "richtige" Langzahlarithmetik.

`GinaC` von der Universität Mainz[5] steht in guter alter akronymischer Art für *"GiNaC is Not a CAS (Computer Algebra System)"*, was irgendwie schon gelogen ist, denn `GinaC` bietet wirklich einiges, was ein gutes CAS braucht – allerdings nicht in Form einer programmierbaren Oberfläche, sondern als `C++`–Klassenbibliothek. Es gibt zwar eine interaktive Benutzerschnittstelle, `ginsh`, die beim Sourcecode mitgeliefert wird und die Zugriff auf alle Datentypen und Funktionen erlaubt, dafür aber über keinerlei Kontrollstrukturen (Schleifen, Entscheidungen, Verzweigungen) verfügt. Dafür ist aber beispielsweise die Ganzzahlmultiplikation extrem schnell. Was allerdings noch fehlt, sind Routinen für "algebraische Geometrie", also Gröbnerbasen und dergleichen – genau das, was nun wieder in `Singular` so gut implementiert ist.

`CoCoA` ist ein an der Uni Genua entwickeltes System (CoCoATeam,), das sich auf Fragen der algebraischen Geometrie und insbesondere Varianten der Gröbnerbasen spezialisiert hat. An einigen Stellen ist es noch etwas sperrig[6], die Benutzeroberfläche ist bei weitem nicht so poliert wie die der kommerziellen Programme[7] und grafische Möglichkeiten fehlen völlig, aber dafür besitzt es äußerst leistungsfähige Funktionen für konstruktive Idealtheorie. Und es ist frei verfügbar, was es zu einem idealen Instrument zur Vorlesungsbegleitung macht.

Viele Computeralgebra–Pakete bieten neben "exakter" bzw. "symbolischer" Rechnung (entweder in $\mathbb{Q}$ oder in endlichen Ringen) auch noch "hochexakte" numerische Rechnung[8] mit einer beliebig festlegbaren Stellenzahl. Allerdings ist es oftmals schwer nachvollziehbar, was die Systeme intern wirklich tun. Beispielsweise führt `Maple` derartige Berechnungen intern oftmals in *exakter rationaler Arithmetik* durch und rundet erst das Endergebnis – diese Methode zu "mogeln" führt zwar meistens zu (überraschend) exakten Ergebnissen, kann aber andererseits die Laufzeit drastisch verlängern und Instabilitäten eines Verfahrens überspielen.

Abgesehen davon sollte man generell bei der Verwendung von symbolischen "Resultaten" eines Computer-algebra–Systems Vorsicht walten lassen, denn diese können sehr wohl

[4] Stand 12.4.2010

[5] Und zwar von Physikern, nicht von Mathematikern erstellt.

[6] Beispielsweise beim Rechnen mit rationalen Funktionen oder Laurentpolynomen

[7] Man hat das Gefühl, daß bei diesen mindestens 90% der Entwicklungsarbeit in Klickibunti investiert werden.

[8] Zumindest die beiden "großen", also `Maple` und `Mathematica`, `MuPAD` hatte sowas auch.

unvollständig, uninterpretierbar oder sogar schlichtweg falsch sein. Und dies kann auf Implementierungsfehler, zu großzügigen Umgang mit Voraussetzungen[9] oder einfach auf prinzipielle Probleme zurückgehen. Ein einfaches Beispiel ist die Integration des Betrags von Funktionen: Um dies symbolisch und exakt durchführen zu können, muß man alle Vorzeichenwechsel der Funktion kennen, was bereits eine recht komplexe Geschichte ist, und dann werden die Stammfunktionen auch recht schnell recht kompliziert. Ein anderes einfaches Beispiel für die Schwächen symbolischer Rechnung sieht man, wenn man die fünfte Ableitung der Funktion $f(x) = |x|^4$ ausrechnen läßt.

1.1 Exakte Rechnung contra numerische Rechnung

Nachdem auf heutigen Computern allen Werbeargumenten zum Trotz der verfügbare Speicher immer noch endlich ist, kann man auf Computern eben auch nur Zahlen verarbeiten, die mit endlicher Information darstellbar sind. Die einfachsten Beispiele hierfür sind

- *Ganze Zahlen*, die bezüglich einer Basis $B \geq 2$ als

$$p = \pm \sum_{j=0}^{N} p_j \, B^j, \qquad p_j \in \{0, \ldots, B-1\}, \quad N \in \mathbb{N}_0,$$

  mit *Ziffern* $p_0, \ldots, p_N$ dargestellt werden können. Auch wenn diese Zahlen beliebig groß werden können und irgendwann mehr Speicher benötigen können, als auf einem Computer zur Verfügung steht, so kommt doch jede einzelne Zahl für sich mit endlich vielen Ziffern über die Runden. Diese Bemerkung ist so banal wie grundlegend.

- *Brüche* lassen sich nun ganz einfach als Paare ganzer Zahlen

$$(p, q) = \frac{p}{q} = \pm \frac{\displaystyle\sum_{j=0}^{M} p_j \, B^j}{\displaystyle\sum_{j=0}^{N} q_j \, B^j}$$

  darstellen. Trotzdem wird die Sache hier schon ein bisschen interessanter: Sollen die Brüche immer in gekürzter Form dargestellt werden oder nicht? Dies ist bereits das immer wiederkehrende Dilemma zwischen speichereffizienter Darstellung (ein gekürzter Bruch kommt mit einer minimalen Anzahl von Ziffern aus) und Effizienz bei den Rechenoperationen (nach jeder Addition muß mühsam gekürzt werden).

- Eine ganz andere Darstellung sind die *Fließpunktzahlen*[10], die ein sehr variables Zahlen-

[9] Bevor das System zugibt, daß es etwas nicht kann, wird zumindest mal etwas probiert ...

[10] Die beiden ursprünglichen "reinrassigen" Namen für dieses Zahlenformat sind das englische "*floating point number*" und das deutsche "Gleitkommazahlen", aber inzwischen haben sich auch die beiden anderen Kombinationen "Fließkommazahlen" oder "Gleitpunktzahlen" eingebürgert und es ist grundsätzlich erlaubt, was gefällt ...

format mit endlicher Information darstellen. Eine derartige Zahl hat die Form

$$x = \sum_{j=1}^{M} x_j \, B^{-j} \times B^e, \qquad x_j \in \{0, \ldots, B-1\}, \quad e \in E \subset \mathbb{Z},$$

wobei M eine a priori festgelegte Zahl, die sogenannte *Mantissenlänge* ist – die Zahl $\sum x_j \, B^j$ bezeichnet man dann als *Mantisse*, und der *Exponentenbereich* normalerweise ein (nahezu) symmetrisches Intervall ist, das die Null enthält. Die Zahl x heißt *normalisiert*, wenn $x_1 \neq 0$ ist – nicht–normalisierte Zahlen "verschenken" Genauigkeit, indem sie wertvolle Mantissenziffern mit Nullen belegen, die man doch durch einen einfachen *Shift*, also eine Multiplikation mit B^k, loswerden könnte. Das Problem mit dieser Zahlendarstellung, die sehr effiziente Rechnung ermöglicht und in allen verbreiteten Prozessoren eingebaut ist, besteht darin, daß bei arithmetischen Operationen "Extraziffern" auftreten können[11], die man durch Runden beseitigen muss. Und das führt leider dazu, daß weder Assoziativgesetz noch Distributivgesetz gelten, im Gegensatz zu den Brüchen haben wir es also mit keinem Körper mehr zu tun. Ein Teil der Schwierigkeiten kommt zwar vom Normalisieren, siehe z.B. (Higham, 1996; Sauer, 2000a), aber ohne Normalisierung wird's auch nicht besser, weil man irgendwann möglicherweise zu viele Stellen verliert, um noch ein vernünftiges Ergebnis zu erhalten.

Beispiel 1.1 *"Die harmonische Reihe konvergiert"*
Eine beliebte Anekdote unter Mathematikern, die das Vergnügen haben, Vorlesungen über Ingenieurmathematik zu halten, handelt vom Informatikstudenten, der am Tag nachdem man ihm die Divergenz die harmonischen Reihe

$$\sum_{j=1}^{\infty} \frac{1}{j}$$

bewiesen hat, mit einem ellenlangen Computerausdruck erscheint, der "doch zeigt, daß diese Reihe konvergiert" und zwar gegen `15.403683` *in einfacher Genauigkeit. In doppelter Genauigkeit hingegen stürzt das Programm ab (bzw. liefert sogar den Wert* `Inf` *für ∞!). Abgesehen davon, daß einem die zwei verschiedenen Ergebnisse schon zu denken geben sollten, ist das Problem natürlich einfach zu erklären: Irgendwann ist der zu addierende Wert $1/j$ so klein im Vergleich zur bisher aufgelaufenen Summe, daß er bei der Rundung einfach unter den Tisch fällt.*

Führt man dasselbe "Experiment" in exakter rationaler Rechnung durch, so steigt die Folge der Teilsummen natürlich strikt monoton an[12]*, dafür dauern aber die einzelnen Operationen immer länger und der Rechner stürzt ab, lange bevor man Divergenz erkennen kann (wahrscheinlich ist der so erreichte "Grenzwert" sogar kleiner als der bei doppeltgenauer Fließpunktrechnung).*

Nun, dieses Beispiel spricht eigentlich nicht allzusehr für die pauschale Verwendung exakter Arithmetik, da man das Genauigkeitsproblem durch ein beinahe noch dramatischeres Komplexitätsproblem ersetzt. Außerdem verliert man unmittelbar eine große Klasse von effizienten

[11] Hat beispielsweise die Zahl x M Ziffern, so hat x^2 bereits $2M$ Ziffern und gehört damit nicht mehr zu "unserer" Klasse der Fließpunktzahlen.

[12] Was natürlich noch nicht auf Divergenz schließen läßt.

numerischen Verfahren, nämlich alle *iterativen* Verfahren, die die Lösung eines Problems nicht in einer endlichen Anzahl von Schritten[13], sondern diese erst "im Grenzwert" erreichen.

1.2 Rationale Rechnung und Wurzeln

Ein einfaches Beispiel für die Art von Problemen, die beim exakten rationalen Rechnung auftreten, besteht in der Bestimmung und Handhabung von Wurzeln, die in vielen angewandten Problemen, z.B. beim Lösen von symmetrischen, positiv definiten Gleichungssystemen mit dem Cholesky–Verfahren, ganz natürlich auftreten. Interessant wird die Geschichte dadurch, daß diese Wurzeln nun nicht mehr im Körper $\mathbb{Q}$ liegen müssen – inzwischen ist das einst wohlgehütete Geheimnis der Pythagoräer, daß $\sqrt{2}$ irrational ist, zum "Allgemeingut" geworden, das man in praktisch jedem populärwissenschaftlichen Buch findet (Sagan, 1989). Wir werden in diesem Kapitelchen mal ein paar sehr elementare Aspekte des Adjungierens betrachten[14], die uns einen ersten Eindruck von dem geben, was die Computeralgebra so interessant macht, nämlich die Mischung aus Algebra und der ewigen Suche nach effizienten "numerischen" Verfahren.

Trotzdem kann man sich auch hier noch retten, indem man die Wurzel zu $\mathbb{Q}$ *adjungiert*.

Proposition 1.2 (Alegbraische Körpererweiterung) *Sei $r \in \mathbb{Q}$ kein Quadrat[15]. Dann bildet*

$$\mathbb{Q}\left[\sqrt{r}\right] := \{(p,q) : p,q \in \mathbb{Q}\}$$

mit den Verknüpfungen

$$(p,q) \oplus (p',q') = (p+p', q+q')\,, \qquad (p,q) \otimes (p',q') = (pp' + r\,qq', qp' + pq') \tag{1.1}$$

einen Körper.

Bemerkung 1.3 (Adjungierte Wurzeln)

1. *Der Rechenaufwand steigt natürlich immens bei der Verwendung adjungierter Wurzeln! So sind bei einer Addition nun zwei rationale Additionen (mit dem ganzen Brimborium wie "Erweitern" und "Kürzen" der Brüche), bei einer Multiplikation sogar fünf Multiplikationen und zwei Additionen durchzuführen.*

2. *Natürlich kann man die Idee des Adjungierens nicht nur einmal, sondern wiederholt durchführen. Sind nämlich $r_1, \ldots, r_n \in \mathbb{Q}$, so erhält man*

   $$\mathbb{Q}\left[\sqrt{r_1}, \ldots, \sqrt{r_n}\right] = \left(\mathbb{Q}\left[\sqrt{r_1}, \ldots, \sqrt{r_{n-1}}\right]\right)\left[\sqrt{r_n}\right] = \left\{(p,q) \,:\, p,q \in \mathbb{Q}\left[\sqrt{r_1}, \ldots, \sqrt{r_{n-1}}\right]\right\}$$

   und verwendet die Rechenregeln wie in (1.1) mit $r = r_n$.

3. *Daß beim mehrfachen Adjungieren von Wurzeln der Rechenaufwand entsprechend steigt, braucht wohl nicht mehr gesondert erwähnt zu werden.*

[13] Und Endlichkeit allein ist auch noch nicht der Weisheit letzter Schluß – oftmals braucht man noch Abbruchbedingungen, die einem sagen, wann das Ende der Endlichkeit erreicht ist. Wir werden solche Probleme noch kennenlernen.

[14] Kein Mensch macht das wirklich so.

[15] Es gibt also kein $q \in \mathbb{Q}$, so daß $r = q^2$ ist.

4. *Möglicherweise macht man sich eine Menge unnützer Arbeit, wenn man sich r nicht genau genug ansieht: Hat r nämlich einen quadratischen Faktor, d.h., ist $r = p^2 q$, $p, q \in \mathbb{N}$, dann würde es eigentlich genügen, q zu adjungieren. Aber das Auffinden möglicher quadratischer Faktoren von r heißt wahrscheinlich, r zu faktorisieren und das ist nicht ganz so einfach.*

5. *Mit ein wenig mehr Hintergrund wird sich das Adjungieren von Wurzeln sogar als ein sehr einfacher Prozess erweisen. Mehr dazu in Kapitel 2.4.*

Beweis von Proposition 1.2: Die Idee ist naheliegend: Wir identifizieren das Paar (p, q) mit der Zahl $p + q\sqrt{r}$ – solange $\sqrt{r} \notin \mathbb{Q}$ ist diese Darstellung auch eineindeutig, was es uns erlaubt, zwischen den beiden Repräsentationen hin- und herzuwechseln! Damit ergeben sich die Rechenregeln als

$$(p, q) \oplus (p', q') = p + q\sqrt{r} + p' + q'\sqrt{r} = (p + p') + (q + q')\sqrt{r}$$

und

$$\begin{aligned}(p, q) \otimes (p', q') &= \left(p + q\sqrt{r}\right)\left(p' + q'\sqrt{r}\right) = pp' + pq'\sqrt{r} + qp'\sqrt{r} + qq'\sqrt{r}^2 \\ &= (pp' + qq'r) + (pq' + qp')\sqrt{r}.\end{aligned}$$

Aus diesen Beobachtungen ergeben sich nun unmittelbar die Kommutativität sowie das Assoziativ- und das Distributivgesetz. Bleibt noch zu zeigen, daß jedes Element $0 \neq (p, q) \in \mathbb{Q}[\sqrt{r}]$ ein multiplikatives Inverses hat – das berechnet man "einfach" als

$$(p, q)^{-1} = \left(\frac{p}{p^2 - rq^2}, -\frac{q}{p^2 - rq^2}\right) \tag{1.2}$$

angibt – in der Tat ist

$$\left(\frac{p}{p^2 - rq^2}, -\frac{q}{p^2 - rq^2}\right) \otimes (p, q) = \left(\frac{p^2 - rq^2}{p^2 - rq^2}, \frac{pq - pq}{p^2 - rq^2}\right) = (1, 0).$$

Der Nenner in (1.2) ist dabei immer $\neq 0$, weil man sonst

$$0 = p^2 - rq^2 \qquad \Longrightarrow \qquad r = \frac{p^2}{q^2} \qquad \Longrightarrow \qquad \sqrt{r} \in \mathbb{Q}$$

hätte, und hier brauchen wir wirklich, die Voraussetzung, daß r kein Quadrat ist. □

Damit ist ja eigentlich wieder alles ganz einfach. Müssen wir mit Wurzeln von rationalen Zahlen arbeiten, so überprüfen wir "einfach", ob diese Wurzel rational ist, indem wir die Wurzeln von Zähler und Nenner bestimmen und prüfen, ob sie natürliche Zahlen sind, und ist die Wurzel unserer rationalen Zahl irrational, dann adjungieren wir sie eben. Aber: Wie bestimmt man die

Wurzel einer ganzen Zahl r auf *effiziente* Art und Weise? Das altbewährte Heron–Verfahren[16], das heißt die Iteration

$$x_0 = r, \qquad x_{n+1} = \frac{1}{2}\left(x_n + \frac{r}{x_n}\right), \quad n \in \mathbb{N}_0, \tag{1.3}$$

erzeugt zwar eine Folge von rationalen[17] Zahlen, die *quadratisch*[18] gegen die Wurzel konvergiert – aber eben nur konvergiert. Beispielsweise erhalten wir für $r = 4$ die Iteration $x_{n+1} = x_n/2 + 2/x_n$, also die Folge

$$x_0 = 4, \qquad x_1 = \frac{5}{2}, \qquad x_2 = \frac{41}{20}, \qquad x_3 = \frac{3281}{1640}, \qquad \dots$$

die keine Anstalten macht, irgendwann den exakten Wert 2 zu liefern.

Übung 1.1 Zeigen Sie: Die Zahlen x_n, $n \geq 1$, sind von der Form

$$x_n = 2 + \frac{1}{m_n}, \qquad m_{n+1} = 2m_n\left(2m_n + 1\right), \qquad m_1 = 2.$$

Trotzdem kann man das Heron–Verfahren so modifizieren, daß man es als relativ flottes Verfahren zur Bestimmung der Existenz und des Wertes ganzzahliger Wurzeln zu ganzen Zahlen verwenden kann.

Definition 1.4 *Für $x \in \mathbb{Q}$ bezeichne*

$$\lfloor x \rfloor = \max\left\{k \in \mathbb{Z} \ : \ k \leq x\right\}$$

den ganzzahligen Anteil von x, das heißt,

$$x = \lfloor x \rfloor + \underbrace{x - \lfloor x \rfloor}_{\in [0,1) \cap \mathbb{Q}}.$$

Proposition 1.5 *Für jedes $r \in \mathbb{N}$ liefert das* ganzzahlige Heron–Verfahren

$$x_0 = r, \qquad x_{n+1} = \left\lfloor \frac{x_n}{2} + \frac{r}{2x_n} \right\rfloor, \quad n \in \mathbb{N}, \tag{1.4}$$

nach endlich vielen Schritten entweder die ganzzahlige Wurzel $\sqrt{r}$ falls $r \in \mathbb{Z} \cdot \mathbb{Z}$ oder die Abbruchbedingung $x_n^2 < r$.

[16] Benannt nach *Heron von Alexandria*, etwa von 10 n. Chr. bis 75 n. Chr., Mathematiker, Physiker und Erfinder. Neben der Formel für die Dreiecksfläche bringt man seinen Namen auch mit einem numerischen Verfahren zur Berechnung von Quadratwurzeln in Verbindung, das aber bereits in babylonischen Keilschrifttexten nachgewiesen ist, doch dazu später mehr. Außerdem konstruierte er Automaten, darunter selbsttätig zwitschernde Vögel und eine *Dampfmaschine*.

[17] Und damit symbolisch darstellbaren

[18] Also recht zügig.

Beweis: Was man hier ausnutzt, ist die *einseitige* Konvergenz des Heron–Verfahrens, eine Eigenschaft, die ja generell für Anwendungen des Newton–Verfahrens auf konvexe Funktionen zutrifft. Da nämlich für beliebige $0 < x, a \in \mathbb{Q}$

$$\frac{1}{2}\left(x + \frac{a^2}{x}\right) > a \qquad \Longleftrightarrow \qquad x^2 + a^2 > 2ax \qquad \Longleftrightarrow \qquad (x-a) \neq 0$$

gilt, erhalten wir unter der Voraussetzung $x_n \neq \sqrt{r}$, daß auch

$$y_{n+1} := \frac{1}{2}\left(x_n + \frac{r}{x_n}\right) > \sqrt{r}$$

ist[19], also $x_{n+1} = \lfloor y_{n+1} \rfloor \geq \sqrt{r}$, wenn $\sqrt{r} \in \mathbb{N}$, also wenn r eine Quadratzahl ist. Da außerdem für beliebige rationale Zahlen $0 < a, x$ die Äquivalenz

$$\frac{1}{2}\left(x + \frac{a^2}{x}\right) < x \qquad \Longleftrightarrow \qquad x^2 + a^2 < 2x^2 \qquad \Longleftrightarrow \qquad a < x$$

gilt, ist die Folge der x_n, $n \in \mathbb{N}$, *strikt monoton fallend* solange sie sich oberhalb von $\sqrt{r}$ befindet. Damit muß nach m Schritten der Fall $x_m = \lfloor \sqrt{r} \rfloor$ eintreten, und dann ist

$$\begin{array}{ll} x_m^2 = r, & r \in \mathbb{N} \cdot \mathbb{N}, \\ x_m^2 < r, & r \notin \mathbb{N} \cdot \mathbb{N}. \end{array}$$

□

Dieses Beispiel zeigt schon, wo die interessanten Fragestellungen in der Computeralgebra liegen: Wie kann man effiziente "numerische" Verfahren mit den "algebraischen" Gegebenheiten des Gebildes, in dem man rechnet, kombinieren?

Zum Abschluß dieses Abschnitts treiben wir die Effizienz noch auf die Spitze, denn schließlich beinhaltet ja jede Iteration in (1.4) immer noch einiges an Rechenoperationen. Aber auch hier kann man sparen: Verwendet man nämlich die *Binärdarstellung*, das heißt, die Basis $B = 2$ des Zahlensystems, dann ist

$$x = \sum_{j=0}^{n} x_j 2^j = x_n \cdots x_0. \qquad \text{und} \qquad \frac{x}{2} = \sum_{j=0}^{n} x_j 2^{j-1} = x_n \cdots x_1 \,.\, x_0,$$

Division durch 2 entspricht also einem *Rechtsshift* der Ziffern über den "Dualpunkt" hinaus. Entsprechend zerlegen wir r in $r = px + q$, $0 \leq q < x$, und erhalten, daß $\frac{p}{2} = p_m \cdots p_1 \,.\, p_0$ sowie $\frac{q}{2x} < \frac{1}{2}$. Also ist

$$\begin{aligned} \left\lfloor \frac{x}{2} + \frac{r}{2x} \right\rfloor &= (x_n \cdots x_1 + p_m \cdots p_1) + \underbrace{\left\lfloor \frac{x_0 + p_0}{2} + \frac{q}{2x} \right\rfloor}_{=1 \,\Leftrightarrow\, x_0 = p_0 = 1} = x \oslash 2 + p \oslash 2 + x_0 p_0, \\ &= x \oslash 2 + p \oslash 2 + (x_0 + p_0) \oslash 2 = (x + r \oslash x) \oslash 2, \end{aligned} \tag{1.5}$$

[19] Nette Bemerkung am Rande: Wenn das Heronverfahren noch nicht am Ziel ist, dann springt es immer an einen Punkt *rechts* von der Lösung!

wobei "$\oslash$" die ganzzahlige Division unter Vernachlässigung des Rests bedeutet, was ja insbesondere $p = r \oslash x$ liefert. Wie man aus (1.5) erkennt, ist der Divisionsrest q völlig irrelevant.

Damit ergibt sich ein Schritt des ganzzahligen Heron–Verfahrens ganz einfach als

$$x_{n+1} = (x_n \oplus r \oslash x_n) \oslash 2, \tag{1.6}$$

was lediglich eine ganzzahlige Addition, eine ganzzahlige Division und eine ganzzahlige Division durch 2 (im Falle einer binären Darstellung also ein "Shift" nach rechts) benötigt. Insbesondere muß also nichts mehr rational gerechnet und abschließend mühevoll gerundet werden.

Übung 1.2 Wie lauten für $x, y \in \mathbb{Z}$ die Terme $x_1, \ldots, x_n$ (und welchen Wert hat n) in der Entwicklung $(x + y) \oslash 2 = x_1 \oslash 2 + \cdots + x_n \oslash 2$? $\diamondsuit$

1.3 Simple Roboter und polynomiale Gleichungssysteme

Eine weitere Anwendung des symbolischen Rechnens besteht im *formalen* Operieren mit algebraischen Objekten, die keine Zahlen mehr sind, beispielsweise mit *Polynomen*. Zur Erinnerung: Zu einem Grundkörper $\mathbb{K}$ bezeichnet $\Pi = \mathbb{K}[x]$, $x = (x_1, \ldots, x_n)$, den Ring der Polynome in n Variablen, das heißt, alle *endlichen* Linearkombinationen von *Monomen* der Form x^α, $\alpha \in \mathbb{N}_0^n$. Ist nun $F \subset \Pi$ eine *endliche* Menge von Polynomen, dann interessiert man sich für die Lösungen $x^* \in \overline{\mathbb{K}}^n$ des polynomialen Gleichungssystem

$$F(x) = 0, \qquad \text{d.h.} \qquad f(x) = 0, \quad f \in F.$$

Diese Lösungen liegen leider nicht immer im Grundkörper $\mathbb{K}$, sondern in einer geeigneten algebraischen Erweiterung – so hat beispielsweise das Polynom $f(x) = x^2 + 1$ rationale Koeffizienten, aber die beiden komplexen Nullstellen $\pm i$ – was man gegebenenfalls geeignet interpretieren muß.

Die Lösungsmengen solcher polynomialer Gleichungssysteme und deren Struktur spielen auch in der Praxis eine bedeuetende Rolle, beispielsweise bei der Bewegungsplanung von Robotern. Als einfaches Modell betrachten wir einmal einen *planaren Zweigelenkroboter*, wie in Abb. 1.1. Dieser Roboter besitzt ein Gelenk an der Stelle $x_0 \in \mathbb{R}^2$, an dem ein Arm der Länge d_1 befestigt ist. An dessen anderem Ende, dem (beweglichen) Punkt x_1 befindet sich ein weiteres Gelenk, an dem auch der zweite Arm mit der Länge d_2 befestigt ist. Das andere Ende dieses Arms ist der Punkt x_2, an dem sich die "Nutzlast" (das kann ein Stift, ein Schweißgerät oder ein Skalpell sein) befindet. Gesucht sind, für vogegebenes $x_2 \in \mathbb{R}^2$ (der Aufhängepunkt x_0 ist von Haus aus fest) die Winkel α und β an den beiden Gelenken. Da

$$\cos\alpha = \frac{\langle x_1 - x_0, e_1 \rangle}{d_1} \qquad \text{und} \qquad \cos\beta = \frac{\langle x_0 - x_1, x_2 - x_1 \rangle}{d_1 d_2}$$

genügt es, die Lage des Punktes x_1 festzustellen – die Winkel bekommen wir dann unmittelbar. Andererseits wird x_1 durch die Abstandsbedingungen

$$\begin{array}{rcl} \|x_1 - x_0\|_2 &=& d_1, \\ \|x_1 - x_2\|_2 &=& d_2, \end{array} \qquad \text{bzw.} \qquad \begin{array}{rcl} \|x_1 - x_0\|_2^2 &=& d_1^2, \\ \|x_1 - x_2\|_2^2 &=& d_2^2, \end{array}$$

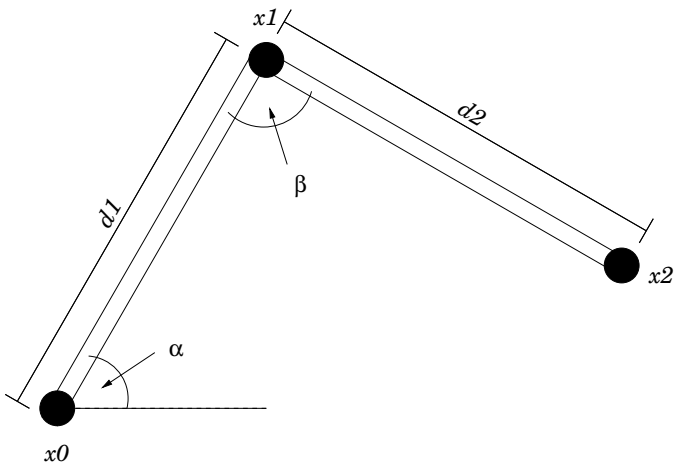


Abbildung 1.1: Ein Roboter mit zwei beweglichen Armen und zwei Gelenken. "Gearbeitet" wird mit dem Punkt x_2, die "Steuerungsgrößen" sind die beiden Winkel α und β.

(denn wer will schon Wurzeln) festgelegt, was zwei quadratische Gleichungen in den beiden Unbekannten $x_1 = (x_{11}, x_{12})$ liefert. Lösen wir sie schnell für den einfachen Spezialfall[20] $x_0 = (0, 0)$ und $d_1 = d_2 = 1$, wo die Gleichungen die Form

$$\begin{aligned} 1 &= x_{11}^2 + x_{12}^2 \\ 1 &= x_{11}^2 - 2x_{21}x_{11} + x_{21}^2 + x_{12}^2 - 2x_{22}x_{12} + x_{22}^2 \end{aligned}$$

annehmen. Subtrahiert man nun die erste Gleichung von der zweiten, dann erhält man

$$0 = -2x_{21}x_{11} + x_{21}^2 - 2x_{22}x_{12} + x_{22}^2 = \|x_2\|_2^2 - 2x_{21}x_{11} - 2x_{22}x_{12},$$

was uns unter der Voraussetzung $x_2 \neq 0$ mindestens eine der beiden Identitäten

$$x_{11} = \frac{\|x_2\|_2^2 - 2x_{22}x_{12}}{2x_{21}} \qquad \text{oder} \qquad x_{12} = \frac{\|x_2\|_2^2 - 2x_{21}x_{11}}{2x_{22}}$$

liefert, je nachdem ob $x_{21} \neq 0$ oder $x_{22} \neq 0$ oder beide. Setzt man das dann in die erste Gleichung ein, so erhält man eine quadratische Gleichung in nunmehr nur noch einer Unbekannten, die

1. zwei *reelle* Lösungen (symmetrisch um die Verbindungslinie zwischen x_0 und x_2) hat, wenn der Roboter x_2 erreichen kann.

[20] Das heißt, wir haben es mit einem Roboter zu tun, dessen Arme gleichlang sind. Die Länge können wir natürlich immer zu 1 normieren und ebenso den "festmonierten" Punkt zum Ursprung unseres Koordinatensystems ernennen.

2. eine (doppelte) reelle Lösung hat, wenn $\|x_2\| = 2$, das heißt, wenn sich der Roboter voll ausstrecken muß, um x_2 zu erreichen.

3. zwei *komplexe* Lösungen, wenn x_2 außerhalb der Reichweite des Roboters liegt.

Bleibt noch der Fall $x_2 = 0$: Hier ist jede Wahl von x_1 auf dem Einheitskreis korrekt, der Roboter ist "zusammegeklappt", eine Konfiguration, die wegen der räumlichen Ausdehnung des Roboters nicht angenommen werden kann.

Natürlich ist das nur ein sehr einfaches, eigentlich zu einfaches Beispiel und die Steuerung von und Bahnplanung für Roboter muß noch sehr viel mehr Beschränkungen und Nebenbedingungen berücksichtigen, aber zumindest sieht man schon mal, daß polynomiale Gleichungssysteme und systematische Methoden um deren Lösungen und die Struktur der Lösungsräume (hier: der Unterschied zwischen $x_2 = 0$ und $x_2 \neq 0$) zu bestimmen, auch in der Praxis eine nicht zu vernachlässigende Bedeutung besitzen.

1.4 Geteilte Geheimnisse und Interpolation

Ein weiteres nettes Beispiel für symbolisches Rechnen aus (Gathen & Gerhard, 1999), das ein bißchen in die Richtung "Kryptographie" geht, ist das folgende Problem:

> *Eine geheime Information, codiert in eine Zahl $g \in \mathbb{N}$, soll so in N Daten verschlüsselt ("geteilt") werden, daß das Geheimnis zwar aus allen N Informationen, nicht aber aus $N-1$ der Werte rekonstruiert werden kann.*
>
> *Allgemeiner könnte man fordern, daß zur Rekonstruktion des Geheimnisses $M \leq N$ der N Informationen, nicht aber $M-1$ ausreichend sind.*

Zugegeben, ein bißchen erinnert das Ganze an die alten Edgar–Wallace–Filme ("Die Tür mit den sieben Schlössern"), aber man kann sich durchaus auch ernsthafte Anwendungen vorstellen. Wie dem auch sein, die mathematische Methode ist hier ein alter Bekannter aus Numerik–Vorlesungen, nämlich *Polynominterpolation*. Sei nämlich $\mathbb{K}$ ein Körper, so daß $g \in \mathbb{K}$, z.B., $\mathbb{K} = \mathbb{Q}$ oder $\mathbb{K} = \mathbb{F}_p := \mathbb{Z}/\langle p \rangle$, wobei $p > g$ eine Primzahl ist, und $\mathbb{K}[x]$ wieder der Ring der (univariaten, also lediglich $n = 1$) Polynome über $\mathbb{K}$.

Lemma 1.6 *Es sei $N \in \mathbb{N}$ und $\mathbb{K}$ ein Körper. Für jede Wahl von $N+1$ disjunkten Punkten*[21] *$x_0, \ldots, x_N \in \mathbb{K}$ und Werten $y_0, \ldots, y_n$ gibt es genau ein Polynom $f \in \mathbb{K}[x]$ vom Grad $\leq N$, so daß*

$$f(x_j) = y_j, \qquad j = 0, \ldots, N.$$

Beweis: Um die Existenz zu beweisen, verwendet man normalerweise die *Lagrange–Basispolynome*

$$\ell_j(x) := \prod_{k \neq j} \frac{x - x_k}{x_j - x_k}, \qquad j = 0, \ldots, N,$$

[21] Das bedeutet natürlich, daß "unser" Körper $\mathbb{K}$ mindestens $N+1$ Elemente haben muß.

die alle vom Grad $\leq N$ sind und die Eigenschaft $\ell_j(x_k) = \delta_{jk}$, $j,k = 0,\ldots,N$, haben, denn dann ist

$$f(x) = \sum_{j=0}^{N} y_j\, \ell_j(x).$$

Die Eindeutigkeit ergibt sich sofort, wenn man berücksichtigt, daß die Differenz zweier Lösungen $f - \widetilde{f}$, ein Polynom vom Grad $\leq N$ ist, das an den $N+1$ Punkten $x_0,\ldots,x_N$ verschwindet, also das Nullpolynom sein muß[22]. □

Übung 1.3

1. Zeigen Sie, daß ein Polynom $f \in \mathbb{K}[x]$ über einem Körper $\mathbb{K}$ genau dann an einem Punkt $\xi \in \mathbb{K}$ verschwindet, wenn es einen Linearfaktor $x - \xi$ enthält:

   $$f(\xi) = 0 \qquad \Leftrightarrow f(x) = (x-\xi)\; g(x), \quad g \in \mathbb{K}[x].$$

   Hinweis: Verwenden Sie Polynomdivision mit Rest.

2. Folgern Sie daraus, daß ein Polynom vom Grad N, das an $N+1$ disjunkten Punkten verschwindet, das Nullpolynom sein muß.

♢

Das legt auch schon die Strategie für die Ver- und Entschlüsselung des geteilten Geheimnisses nahe: Wir wählen

1. eine Primzahl $p > \max\{g, N+1\}$,
2. $N-1$ (zufällige) Werte $f_1,\ldots,f_{N-1} \in \mathbb{F}_p$,
3. N disjunkte Punkte $x_1,\ldots,x_N \in \mathbb{F}_p \setminus \{0\}$ – einen für das Geheimnis und einen für jeden Geheimnisträger.

Die letzte Bedingung ist der Grund, warum $p > N+1$ gefordert wurde, denn der Körper $\mathbb{F}_p$ muß ja mindestens die N verschiedenen Werte $x_1,\ldots,x_n$ und den Wert 0 enthalten[23]. Dann verwendet man das Polynom

$$f(x) = g + \sum_{j=1}^{N-1} f_j\, x^j, \qquad \text{d.h.} \qquad g = f(0).$$

[22] In der Numerik wird diese Tatsache gerne über den Satz von Rolle bewiesen – der aber setzt voraus, daß wir einen Körper $\mathbb{K}$ verwenden, über dem man Analysis betreiben kann, also etwa $\mathbb{K} = \mathbb{R}$. Und stetige Funktionen setzen ein gewisses Maß an Topologie voraus, um vernünftig definiert zu werden. Trotzdem gilt die Aussage aber, siehe Übung 1.3

[23] Das Nullelement eines Körpers, ganz egal, welches Symbol wir dafür verwenden, ist immer Bestandteil der Körperdefinition und damit festgelegt.

Die j–te Teilinformation, d.h., der j–te Schlüssel $j = 1, \ldots, N$, besteht nun in dem Wert $g_j = f(x_j)$ und die Rekonstruktion ergibt sich als

$$g = f(0) = \sum_{j=1}^{N} g_j \, \ell_j(0),$$

wobei die Werte $\ell_j(0)$ für die Rekonstruktion vorberechnet werden können. Da $f(0)$ das "Geheimnis" darstellt, ist es natürlich nicht sinnvoll, daß keiner der Punkte x_j, $j = 0, \ldots, N$, der Nullpunkt ist, denn sonst gibt man einem "Diktator" alle Information und den anderen nur vollkommen irrelevantes Wissen.

Ist dieser Code nun sicher? Naja, wenn wie mal annehmen, daß nur $g_1, \ldots, g_{N-1}$ bekannt sind und mit

$$\widehat{f}(x) := \sum_{j=1}^{N-1} g_j \, \ell_j(x) = \sum_{j=1}^{N-1} g_j \prod_{\substack{k=1 \\ k \neq j}}^{N-1} \frac{x - x_k}{x_j - x_k}$$

dasjenige Polynom bezeichnen, das die Werte $g_1, \ldots, g_{N-1}$ an den Stellen $x_1, \ldots, x_{N-1}$ interpoliert, dann hat jedes Polynom, das an $x_1, \ldots, x_{N-1}$ die Werte $g_1, \ldots, g_{N-1}$ annimmt, die Form

$$f_y(x) = \widehat{f}(x) + y \underbrace{(x - x_1) \cdots (x - x_{N-1})}_{=:\omega(x)}, \qquad y \in \mathbb{F}_p,$$

und da $\omega(0) = x_1 \cdots x_{N-1} \neq 0$, ist

$$\{f_y(0) \ : \ y \in \mathbb{F}_p\} = \mathbb{F}_p,$$

das heißt, *alle* konstanten Terme ließen sich aus der unvollständigen Information ableiten. Andererseits gibt es aber nur *genau einen* richtigen Wert, nämlich den, den man erhält wenn man verlangt, daß

$$f_y(x_N) = g_N, \qquad \text{also} \qquad y = \frac{\widehat{f}(x_N)}{\omega(x_N)}$$

ist. Also ist das Geheimnis sicher, denn die Kenntnis eines Polynoms vom Grad $N-1$ an $N-1$ Punkten aus $\mathbb{F}_p \setminus \{0\}$ erlaubt keinen Rückschluss auf den Wert des Polynoms an $x = 0$.

Der Fall, daß bereits $M \leq N$ Leute das Geheimnis entschlüsseln können ist nun eine einfache Folgerung: Wir nehmen eben kein Polynom vom Grad $N-1$, sondern lediglich eines vom Grad $M - 1 \leq N - 1$. Allerdings wird die Rekonstruktion etwas aufwendiger – aber nicht wirklich, man könnte beispielsweise ja den Algorithmus von Aitken–Neville[24] dafür verwenden – das ist billiger als entweder für alle $\binom{N}{M}$ möglichen Kombinationen die Lagrange–Fundamentalpolynome vorzuberechnen, oder über diese das Interpolationspolynom zu bestimmen und dann auszuwerten.

[24] Dieser Algorithmus berechnet zu vorgegebenen Interpolationspunkten und Daten und zu einem weiteren Punkt, den *Wert* des Interpolationspolynoms an dieser Stelle, ohne das Interpolationspolynom selbst zu bestimmen. Siehe z.B. (Sauer, 2000a, S. 123–126).

In theory there is no difference between theory and practice. In practice there is.

Yogi Berra

2 Fundamentale Algorithmen

In diesem Kapitel wollen wir uns die Prinzipien der Darstellung von Zahlen und die zugehörigen Rechenoperationen ansehen – insbesondere natürlich wieder die Frage, wie man diese Sachen *effizient* macht.

2.1 Ganze Zahlen

Fangen wir also an mit einer "flexiblen" Darstellung von ganzen Zahlen (fast) beliebiger Größe und den effizienten Implementierungen der Operationen

- Addition/Subtraktion,
- Multiplikation,
- Division und Restbestimmung,

Die grundlegende Größe auf einem modernen Digitalcomputer ist das sogenannte *Wort*, das ist eine Gruppe von Bits, die der Rechner "auf einen Schlag" verarbeiten kann. Die Anzahl der Bits in einem Wort bezeichnet man als *Wortlänge* w; in "aktuellen", handelsüblichen Prozessoren beträgt momentan die Wortlänge normalerweise 32 oder 64 Bit. In einem Wort kann man die Zahlen

$$\mathbb{Z}_B = \mathbb{Z}/\langle B\rangle = \{0, \ldots, B-1\}\,, \qquad B = 2^w,$$

darstellen.

Definition 2.1 *(Multiprecision–Zahlen)*

1. *Eine ganze Zahl p der* Länge[25] $N = \ell(p)$ *in der "Multiprecision–Arithmetik" $\mathbb{M}_w$ ist von der Form*

$$p = \pm \sum_{j=0}^{N} p_j\, B^j, \qquad p_j \in \mathbb{Z}_B, \quad 0 \le N < 2^{w-1}$$

[25] Die Länge einer Zahl ist also die Anzahl der Ziffern minus 1.

und wird mit $N+2$ Worten dargestellt als

$\frac{1-\operatorname{sgn} p}{2}\, 2^{w-1} + N$	p_0	$\cdots$	p_N

.

2. *Eine Zahl $p \in \mathbb{M}_w$ heißt* normal(isiert)*, wenn $p_{\ell(p)} \neq 0$.*

Bemerkung 2.2 *Diese Darstellung von "ganzen Zahlen* beliebiger *Größe" ist erst einmal ein bißchen verwunderlich, denn die Größe der darstellbaren Zahlen ist natürlich beschränkt, denn dadurch, daß die Anzahl der Ziffern im ersten Wort der Darstellung codiert ist, kann solch eine Zahl nur aus $B/2-1$ Worten bestehen.*

Trotzdem ist das in der Praxis keine Einschränkung, denn normalerweise wird der Speicher eines Computers ohnehin nur über ein Wort adressiert, das heißt, die Beschränkung bedeutet, daß eine Zahl maximal 50% des adressierbaren Hauptspeichers belegen darf. Und wenn man mal so eine Zahl hat, dann ist man sowieso in Schwierigkeiten. Außerdem ist das bereits auf einem "altmodischen" 32–Bit–Rechner schon eine ganze Menge: Hier besteht eine Multiprecision–Zahl aus maximal $2^{31}-1$ Zahlen zu je $4 = 2^2$ Bytes, also hat man mindestens[26] *$2^{33} \sim 8 \times 10^9$ Bytes, also etwa 8GB pro Zahl zur Verfügung. Bei einem 64–Bit–Prozessor*[27] *sind es dann schon mehr als 10^{18} Bytes pro Zahl und das sind nun wirklich Größenordnungen weit jenseits von Gut und Böse ...*

Beginnen wir mit der einfachsten Operation, nämlich der Addition, die "natürlich" komponentenweise und mit Übertrag ausgeführt wird (damit ist der Rechenaufwand also linear in der Anzahl der Stellen). Hierzu bemerken wir, daß sich die Summe von zwei Zahlen $a, b \in \mathbb{Z}_B$ als

$$a + b = c + \gamma\, B, \qquad c \in \mathbb{Z}_B, \quad \gamma \in \{0, 1\}$$

ergibt, wobei γ als das *Carry–Bit* bezeichnet wird. Normalerweise wird dieses bei der Ganzzahladdition ermittelt und im Prozessor gespeichert[28]. Um also zwei Multiprecision–Zahlen p, q, der Einfachhheit halber nehmen wir an, daß $\ell(p) \leq \ell(q)$, zu berechnen, verwenden wir folgenden Algorithmus.

Algorithmus 2.3 *(Addition)*
Gegeben: $p = \sum p_j B^j, q = \sum q_j B^j \in \mathbb{M}_w$, $\ell(p) \leq \ell(q)$.

1. *Setze $\gamma = 0$.*
2. *Für $j = 0, \ldots, \ell(q)$ setze*
$$r_j + \gamma\, B \leftarrow p_j + q_j + \gamma,$$
wobei $p_j = 0$, $j = \ell(p)+1, \ldots, \ell(q)$.

[26] "Hauptsatz der Informatik": $2^{10} = 1024 \sim 1000 = 10^3$.
[27] Z.B. einem Sparc–Prozessor der Firma SUN.
[28] Es gibt sogar eigene Maschinensprachenbefehle, um dieses Bit zu setzen und zu löschen. Eigentlich ist die Addition eine Operation der Form $(a, b, \gamma) \to (c, \gamma)$, das heißt, man läßt das Carry–Bit einfach für die nächste Operation stehen.

3. *Ist* $\gamma = 1$*, dann setze* $r_{\ell(q)+1} = 1$.

Ergebnis: $p \oplus q =: r = \sum r_j \, B^j$.

Bemerkung 2.4 1. *Man sieht natürlich sofort, daß die Länge* $\ell(r)$ *von* r *entweder* $\ell(q)$ *oder* $\ell(q)+1$ *ist, je nachdem, wie das Carry–Bit am Ende aussieht.*

2. *Die Subtraktion funktioniert "analog". Genauer addiert man in diesem Fall Zweierkomplemente der Ausgangszahl (das entspricht mehr oder weniger einem einfachen Umdrehen der Bits in der Zahl) und behandelt wieder die Carry–Bits angemessen. Die Details wollen wir hier aber weglassen.*

Als nächstes sehen wir uns die *Multiplikation* an. Dabei erhalten wir, daß

$$pq = \left(\sum_{j=0}^{\ell(p)} p_j \, B^j \right) \left(\sum_{k=0}^{\ell(q)} q_k \, B^k \right) = \sum_{\ell=0}^{\ell(p)+\ell(q)} \left(\sum_{j+k=\ell} p_j q_k \right) B^\ell =: \sum_{\ell=0}^{\ell(p)+\ell(q)} r_\ell \, B^\ell,$$

wobei

$$r_\ell = \sum_{j=\max\{\ell-\ell(q),0\}}^{\min\{\ell,\ell(p)\}} p_j q_{\ell-j}, \qquad \ell = 0, \ldots, \ell(p)+\ell(q). \tag{2.1}$$

Wenn man genau hinsieht ist (2.1) nichts als die *Faltung* der beiden Folgen $(p_j \,:\, j \in \mathbb{Z})$ und $(q_j \,:\, j \in \mathbb{Z})$, wobei die "nicht vorhandenen" Ziffern einfach auf Null gesetzt werden. Allerdings: Die so berechneten "Ziffern" r_ℓ, $\ell = 0, \ldots, \ell(p)+\ell(q)$, liegen nicht mehr in $\{0, \ldots, B-1\}$ sondern können entschieden größer sein[29], also muß man sich noch um Übertrag kümmern. Damit liefe die Multiplikation "Falten und Übertragen" zuerst einmal auf die Bestimmung von $\ell(p)+\ell(q)$ Koeffizienten (als Multiprecision–Zahlen, denn je mehr Möglichkeiten es gibt, ℓ als $j+k$ darzustellen[30], desto größer können die Zahlen werden), die dann, wieder in Multiprecision–Arithmetik aufaddiert werden müssen – die Multiplikation mit B^ℓ ist ja "nur" ein Schiebeprozess und damit (fast) vernachlässigbar[31]. Für ein etwas effizienteres Verfahren nehmen wir an, daß unser System eine "doppeltgenaue" Multiplikation unterstützt, also

$$a \cdot b = c + d \, B, \qquad c, d \in \mathbb{Z}_B$$

berechnen kann. Gibt's sowas nicht, müßte man a und b in Halbwörter zerhacken, diese einfach genau multiplizieren (das Ergebnis passt jetzt in ein Wort) und dann alles passend zusammensetzen. Damit können wir uns ansehen, wie man die Multiplikation mit einer *Ziffer* bestimmt.

[29] Hier ist das Rechnen mit Polynomen, das ganz analog verläuft, entscheidend einfacher.

[30] Und das sind bekanntlich $\binom{\ell+1}{j}$.

[31] Eigentlich ist die "komplexitätstheoretische" Ansicht, man müsse nur die Rechenoperationen zählen und könnte andere Aspekte wie Speicherverwaltung, Speicherzugriff etc. vernachlässigen im Zeitalter von sehr effizienten Hochleistungsrechnern nicht mehr haltbar, siehe (McGeoch, 2001). Aber irgendwie muß man ja zu *einfachen* Vergleichsmaßstäben kommen.

Algorithmus 2.5 *(Multiplikation mit Ziffer)*
Gegeben: $p = \sum p_j\, B^j \in \mathbb{M}_w$, *und* $a \in \mathbb{Z}_B$.

1. *Setze* $d = 0$ *und* $\gamma = 0$.
2. *Für* $j = 0, \ldots, N$ *berechne*

$$\begin{aligned} r + d'\, B &\leftarrow p_j \cdot a \\ r_j + \gamma\, B &\leftarrow r + d + \gamma \\ d &\leftarrow d' \end{aligned}$$

3. *Ist* $d > 0$ *oder* $\gamma > 0$ *setze*

$$r_{\ell(p)+1} \leftarrow d + \gamma$$

Ergebnis: $p \otimes a := r = \sum r_j\, B^j$.

Bemerkung 2.6 *Da der "finale" Überlauf* d *höchstens von der Größenordnung*

$$\left\lfloor \frac{(B-1)\,(B-1)}{B} \right\rfloor = \left\lfloor \frac{B^2 - 2B + 1}{B} \right\rfloor = B - 2$$

ist, ist also $d + \gamma \leq B - 1$ *und die Anzahl der Ziffern erhöht sich maximal um* 1.

Aus unseren Verfahren zur Multiplikation mit einer Multiprecision–Zahl mit einer Ziffer und zur Addition zweier Multiprecision–Zahlen können wir nun das Multiplikationsverfahren zusammensetzen.

Algorithmus 2.7 *(Multiplikation)*
Gegeben: $p, q \in \mathbb{M}_w$.

1. *Setze* $r' = 0$, $p' = |p|$, $q' = |q|$.
2. *Für* $j = \ell(q), \ldots, 0$ *berechne*

$$r' \leftarrow (r' \otimes B) \oplus \left(p' \otimes q'_j\right) \tag{2.2}$$

3. *Setze* $r = (\operatorname{sgn} p)\ (\operatorname{sgn} q)\ r'$.

Ergebnis: $p \otimes q := r$.

Bemerkung 2.8
1. *Die erste Multiplikation in (2.2) ist wieder geschenkt, da sie nur eine Schiebeoperation ist.*
2. *Wenn sich jemand bei Algorithmus 2.7 an das Horner–Schema zur Auswertung von Polynomen erinnert fühlt, dann ist das richtig und auch kein Zufall.*

3. *Der Aufwand dieses Multiplikationsverfahrens ist $O(\ell(p)\ell(q))$, also quadratisch in N, wenn $\ell(p) \sim \ell(q) \sim N$. Die "besten" Verfahren schaffen es dagegen mit einer Größenordnung von $O(N \log N \log\log N)$ elementaren Operationen, aber sie benötigen einiges an Tricks, nämlich Polynommultiplikation, Schnelle Fouriertransformation und mehrfach modulares Rechnen. Wir kommen aber noch dazu.*

Bleiben also zur Vervollständigung der Grundrechenarten die Division und die Bestimmung des Divisionsrests – letztere wird entscheidend für unser *modulares Rechnen* in Abschnitt 2.3 werden. Wir werden aber auch sehen, daß wir für eine effiziente Realisierung der Division schon etwas mehr (auch mathematischen) Aufwand treiben müssen.

Aber auch bei der Division denkt man zuerst wieder an die Methode, die man "in der Schule lernt", also das sukzessive abdividieren "von vorne nach hinten". Sehen wir uns das mal an einem ganz einfachen Beispiel an.

Beispiel 2.9 *Berechnung von* $12346 : 151$ *mit* $B = 10$*. Nachdem* $123 < 151$ *müssen wir also zuerst*

$$1234 : 151 = \underbrace{8 \cdot 151}_{1208} + 26$$

und dann

$$266 : 151 = \underbrace{1 \cdot 151}_{151} + 115$$

also $1234 = 81 \cdot 151 + 115$*, oder, im "altbewährten" Schema*

$$\begin{array}{rcccccccc} & 1 & 2 & 3 & 4 & 6 & : & 151 & = & 81 \\ \hline - & 1 & 2 & 0 & 8 & & & & & \\ \cline{1-5} & & & 2 & 6 & & & & & \\ & & & 2 & 6 & 6 & & & & \\ & & - & 1 & 5 & 1 & & & & \\ \cline{2-5} & & & 1 & 1 & 5 & & & & \end{array}$$

Seien also $p, q \in \mathbb{M}_w$ gegeben wobei natürlich $\ell(p) \geq \ell(q)$ sein sollte, denn ansonsten ist ja $q = 0 \cdot p + q$ die gewünschte Zerlegung. Der Algorithmus läuft dann wie folgt ab.

Algorithmus 2.10 *(Naive Division)*
Gegeben: $p, q \in \mathbb{M}_w$, $\ell(p) \geq \ell(q)$.

1. *Setze* $p' = |p|$, $q' = |q|$.

2. *Setze*

$$\begin{aligned} r &\leftarrow \sum_{j=0}^{\ell(q)-1} p'_{\ell(p)-\ell(q)+j+1}\, B^j, \\ r &\leftarrow 0. \end{aligned}$$

3. *Für* $j = 0, \ldots, \ell(p) - \ell(q)$

   (a) *("Nächste Ziffer") Setze*

   $$r \leftarrow (r \otimes B) \oplus p'_{\ell(p)-\ell(q)-j} \tag{2.3}$$

   (b) *("Divisorenraten") Bestimme* $a \in \mathbb{Z}_B$ *so daß*

   $$0 \leq r \ominus (q' \otimes a) < q'. \tag{2.4}$$

   (c) *("Abdividieren") Setze*

   $$\begin{aligned} r &\leftarrow r \ominus (q' \otimes a)\,, \\ s &\leftarrow (s \otimes B) \oplus a. \end{aligned}$$

4. *Setze*

   $$\begin{aligned} s &\leftarrow (\operatorname{sgn} p)\ (\operatorname{sgn} q)\ s, \\ r &\leftarrow (\operatorname{sgn} q)\ r. \end{aligned}$$

Ergebnis: $p = s \cdot q + r$, *wobei* $|r| < |q|$

So schön und einfach das Ganze ist – dieses Verfahren hat auch einiges an Nachteilen zu bieten.

Bemerkung 2.11 *1. Die "Rechenoperation" in (2.3) macht keine Schwierigkeiten, denn es handelt sich lediglich um einen Schiebeprozess und ein Kopieren der nächsten Stelle von* p *in die "freigewordene" Position.*

2. Das "Raten" des Divisors a *in (2.4) ist eine Kunst für sich! Man kann es zwar mal mit dem Quotienten der beiden "Leitziffern" von* s *und* q *versuchen, aber, wie die zweite Operation in Beispiel 2.9 zeigt, man merkt erst nach einer Multiplikation (mit Ziffer) und einer Subtraktion, ob man gut geraten hat oder nicht. Gegebenenfalls muß eben* a *nach oben oder nach unten korrigiert werden.*

3. Der Rechenaufwand ist aber nicht zu verachten: Man muß $\ell(p) - \ell(q)$ *Multiprecision–Ziffer–Multiplikationen ausführen und dann eine Multiprecision–Subtraktion – beides ist mit einem Aufwand von* $O(\ell(q))$ *elementaren Operationen verbunden. Der Gesamtaufwand ist dann also* $O((\ell(p) - \ell(q))\,\ell(q))$ *elementare Operationen.*

Vor allem der große Rechenaufwand, aber auch die etwas haarigen Details machen dieses Verfahren nicht zum Mittel der Wahl! Was aber sonst tun? Nun, wir erinnern uns wieder mal an die Methoden der Numerischen Mathematik und verwenden das *Newton–Verfahren*, das Standardverfahren zum Lösen nichtlinearer Gleichungen der Form $f(x) = 0$, $f \in C^1(\mathbb{R})$.

Definition 2.12 *Die* Newton–Iteration *berechnet eine Folge x_j, $j \in \mathbb{N}_0$, über die Regel*

$$x_{j+1} = x_j - \frac{f(x_j)}{f'(x_j)}, \qquad x_0 \in \mathbb{R}.$$

Es ist bekannt, siehe z.B. (Sauer, 2000a), daß, wann immer $f \in C^2(\mathbb{R})$ eine *einfache* Nullstelle x^* besitzt, d.h. $f(x^*) = 0$, $f'(x^*) \neq 0$, es eine Umgebung U von x^* gibt, so daß die Newton–Iteration für alle Startwerte $x_0 \in U$ *quadratisch* gegen x^* konvergiert, das heißt, daß

$$|x_{j+1} - x^*| \sim |x_j - x^*|^2 .$$

Allerdings besteht das Hauptproblem der Newton–Iteration gerade darin, einen guten Startwert zu bestimmen (oder zu raten). Warum ist nun das Newton–Verfahren gerade für die Bestimmung von $1/a$ so gut? Ganz einfach, betrachtet man nämlich die Funktion

$$f(x) = \frac{1}{x} - a,$$

die genau die Nullstelle $x^* = \frac{1}{a}$ besitzt, dann ist $f'(x) = -\frac{1}{x^2}$ und die Iterationsvorschrift

$$x_{j+1} = x_j - \frac{x_j^{-1} - a}{-x_j^{-2}} = x_j + x_j - a x_j^2 = 2x_j - a x_j^2 \tag{2.5}$$

kommt lediglich mit Additionen[32] und Multiplikationen, also mit Operationen, die wir bereits kennen, aus. Bevor wir unseren schnellen Divisionsalgorithmus aufstellen, brauchen wir zuerst einmal ein paar Eigenschaften der Iteration (2.5).

Proposition 2.13 *Sei $a > 0$.*

1. *Die Newton–Iteration (2.5) konvergiert gegen $\frac{1}{a}$ wenn $\frac{1}{2a} < x_0 < \frac{3}{2a}$.*
2. *Es ist*
$$\left| x_{j+1} - \frac{1}{a} \right| = a \left| x_j - \frac{1}{a} \right|^2, \qquad j \in \mathbb{N}_0. \tag{2.6}$$
3. *Ist $\rho \in (-1, 1)$ so, daß $x_0 = \frac{1+\rho}{a}$, dann ist*
$$\left| x_j - \frac{1}{a} \right| \leq \frac{|\rho|^{2^j}}{a}, \qquad j \in \mathbb{N}_0.$$
4. *Für die Startwerte aus 1. ist*
$$\left| x_j - \frac{1}{a} \right| \leq \frac{2^{-2^j}}{a}, \qquad j \in \mathbb{N}_0. \tag{2.7}$$

[32] OK, eigentlich ist es natürlich eine Subtraktion, aber darauf soll's und nicht ankommen.

Beweis: Um 1. zu beweisen, müssen wir zeigen, daß die Abbildung $\phi(x) = 2x - ax^2$ das Intervall $\left[\frac{1}{2a}, \frac{3}{2a}\right]$ auf sich selbst abbildet und daß dort $|\phi'| < 1$ ist. Letzteres folgt aus

$$\phi'(x) = 2 - 2ax \qquad \Longrightarrow \qquad -1 < \phi'(x) < 1 \quad \Longleftrightarrow \quad \frac{1}{2a} < x < \frac{3}{2a}.$$

Außerdem ist

$$\phi\left(\frac{1}{2a}\right) = \frac{1}{a} - a\frac{1}{4a^2} = \frac{3}{4a} = \frac{3}{a} - a\frac{9}{4a^2} = \phi\left(\frac{3}{2a}\right),$$

und das Minimum $\frac{1}{a}$ wird von ϕ an der Stelle $\frac{1}{a}$ angenommen. Also ist

$$\phi\left(\left[\frac{1}{2a}, \frac{3}{2a}\right]\right) \subset \left[\frac{1}{2a}, \frac{3}{2a}\right].$$

Für 2. bemerken wir, daß

$$\begin{aligned}\left(x_j - \frac{1}{a}\right)^2 &= \frac{(ax_j - 1)^2}{a^2} = \frac{1}{a^2}\left((ax_j)^2 - 2ax_j + 1\right) = \frac{1}{a}\left(\frac{1}{a} - \left(2x_j - ax_j^2\right)\right) \\ &= \frac{1}{a}\left(\frac{1}{a} - x_{j+1}\right).\end{aligned}$$

Der Beweis von 3. ergibt sich aus

$$\left|x_0 - \frac{1}{a}\right| = \left|\frac{1 + \rho - 1}{a}\right| = \frac{|\rho|}{a} \le \frac{1}{a}$$

und, unter Verwendung von 2. und 3. für j,

$$\left|x_{j+1} - \frac{1}{a}\right| \le a\left|x_j - \frac{1}{a}\right|^2 \le a\left(\frac{\rho^{2^j}}{a}\right)^2 = \frac{1}{a}\left(\rho^{2^j}\right)^2 = \frac{\rho^{2^{j+1}}}{a}$$

was, per Induktion, den Beweis vervollständigt.
Die Konvergenzrate (2.7) aus 4. ergibt sich schließlich daraus, daß für $\frac{1}{2a} < x_0 < \frac{3}{2a}$ die Größe ρ aus 3. die Beziehung $|\rho| < \frac{1}{2}$ erfüllt. □

Um nun aber die gute Konvergenzgeschwindigkeit des Newton–Verfahrens nutzen zu können, brauchen wir einen guten Startwert und das ist nicht mehr ganz so offensichlich wie beim Wurzelziehen in Abschnitt 1.2. Außerdem können wir nicht mehr wirklich mit ganzen Zahlen rechnen, sondern müssen zu "Pseudo–Fließpunkt–Zahlen" übergehen. Aber immer der Reihe nach.

Sei also nun $q \in \mathbb{M}_w$ gegeben und nehmen wir der Einfachheit an, daß q normalisiert und positiv[33] ist, daß also

$$q = \sum_{j=0}^{\ell(q)} q_j B^j \ge \underbrace{q_{\ell(q)}}_{\ge 1} B^{\ell(q)} \ge B^{\ell(q)}$$

[33] Vorzeichen spielen ja beim Dividieren eine eher untergeordnete Rolle.

ist. Und natürlich ist $q < B^{\ell(q)+1}$.

Um das Inverse $\frac{1}{q}$ über das Newton–Verfahren annähern zu können, wählen wir $a \in \{2, \ldots, B\}$ so, daß

$$(a-1)\,q < B^{\ell(q)+1} < aq; \tag{2.8}$$

daß $a > 1$ sein muß folgt sofort aus der Tatsache, daß $q < B^{\ell(q)+1}$ ist. Wäre nun zufällig $aq = B^{\ell(q)}$, dann ist ja $\frac{1}{q} = a\,B^{-\ell(q)}$ und wir können uns das Iterieren sparen[34]. Dieses Auffinden von a entspricht übrigens gerade *einer* Ausführung des "Divisorenratens" in der "naiven" Division, siehe Algorithmus 2.10. Damit die Notation nicht ganz so ausufert setzen wir $n := \ell(q)+1$ und folgern aus (2.8), daß

$$\left(a\,B^{-n} - B^{-n}\right) q = \left((a-1)\,B^{-n}\right) q < 1 = \frac{1}{q}\,q < \left(a\,B^{-n}\right) q,$$

also, nach Division durch q,

$$a\,B^{-n} - B^{-n} < \frac{1}{q} < a\,B^{-n}$$

und daher

$$\frac{1}{q} < a\,B^{-n} < \frac{1}{q} + B^{-n} \tag{2.9}$$

Dies ist bereits der Schlüssel zur Wahl des Startwerts: Ist nämlich $q < \frac{1}{2}\,B^n$, das heißt $B^{-n} < \frac{1}{2q}$, dann setzen wir[35] $x_0 = a\,B^{-\ell(q)-1}$ und erhalten aus (2.9), daß

$$\frac{1}{q} < \underbrace{a\,B^{-\ell(q)-1}}_{=x_0} < \frac{1}{q} + \frac{1}{2q} = \frac{3}{2q}. \tag{2.10}$$

Ist hingegen $q > \frac{1}{2}\,B^n$, also $B^{-n} > \frac{1}{2q}$, dann ist bereits $2q > B^n$ und somit $a = 2$. Wählen wir jetzt $x = B^{-\ell(q)-1}$, dann ist

$$\frac{1}{q} - \frac{1}{2q} = \frac{1}{2q} < x_0 < \frac{1}{q}, \tag{2.11}$$

was uns zusammen mit (2.10) die Abschätzung $\left|x_0 - \frac{1}{q}\right| < \frac{1}{2q}$ und damit die schnelle Konvergenz des Newton–Verfahrens liefert. Ach ja, und ist $q = \frac{B^{\ell(q)+1}}{2}$, dann ist $\frac{1}{q} = 2 \cdot B^{-\ell(q)-1}$ und das Invertieren ist wieder geschenkt.

Bemerkung 2.14 *Die Wahl*

$$x_0 = \begin{cases} a\,B^{-\ell(q)-1}, & 2q < B^{\ell(q)+1}, \\ B^{-\ell(q)-1}, & 2q > B^{\ell(q)+1}, \end{cases}$$

ist sehr einfach zu treffen: Man muß sich nämlich nur das höchste Bit der höchstwertigen Ziffer[36] *von q ansehen. Mit anderen Worten: In Wirklichkeit müssen wir sogar nur die Entscheidung*

$$x_0 = \begin{cases} a\,B^{-\ell(q)}, & q_{\ell(q)} < 2^{w-1}, \\ B^{-\ell(q)}, & q_{\ell(q)} \geq 2^{w-1}, \end{cases} \tag{2.12}$$

[34] Dasselbe Argument greift natürlich auch im Fall $aq = B^{\ell(q)+1}$.

[35] Unter Verwendung der "alten" Notation.

[36] Also das höchste Bit in der Binärdarstellung.

treffen.

Nachdem wir also nun unseren Startwert gefunden haben, rechnen wir nun die Iteration

$$x_{j+1} = -x_j \otimes (q \otimes x_j \ominus 2) \tag{2.13}$$

mit beliebig genauen, normalisierten Fließpunktzahlen weiter, also mit Zahlen der Form

$$(r, e) = r \times B^e, \qquad r \in \mathbb{M}_w, \quad e \in \mathbb{Z}.$$

wobei wir immer annehmen wollen, daß r normalisiert ist – schließlich besteht wenig Sinn darin, überflüssige Nullen herumzuschleppen. Die Multiplikation zweier solcher Zahlen ergibt sich dann als

$$(r, e) \otimes (r', e') = (r \otimes r', e + e')$$

und die Addition (mit geeignetem Schieben) im Falle von $e \geq e'$ als[37]

$$(r, e) \oplus (r', e') = \left(B^{e-e'} r \oplus r', e'\right),$$

wobei nötigenfalls noch normalisiert werden muß.

Was jetzt also noch fehlt ist eine *Abbruchbedingung* für die Newton–Iteration! Wir werden aber sogar sehen, daß wir die Anzahl der notwendigen Iterationen sogar *a priori* beschränken können, und daß diese Zahl sogar verhältnismäßig klein ist. Dazu erinnern wir uns zuerst an (2.7)[38] und erhalten, unter unserer "kanonischen" Voraussetzung $0 < q < p$, daß

$$|p - pqx_j| = pq \left| \frac{1}{q} - x_j \right| \leq pq \frac{2^{-2^j}}{q} \leq \frac{p}{2^{2^j}} \leq \frac{B^{\ell(p)+1}}{2^{2^j}} = \frac{2^{w\,(\ell(p)+1)}}{2^{2^j}} = 2^{w\,(\ell(p)+1)-2^j}, \tag{2.14}$$

was für $2^j > w\ (\ell(p) + 1)$, also spätestens für

$$j = 1 + \lfloor \log_2 w + \log_2 (\ell(p) + 1) \rfloor \tag{2.15}$$

kleiner als 1 wird. Für diesen Wert von j bezeichne $s = \left\lfloor px_j + \frac{1}{2} \right\rfloor \in \mathbb{N}$ die ganzzahlige Rundung von px_j – offensichtlich ist $|s - px_j| \leq \frac{1}{2}$. Dann ist

$$|p - s \cdot q| \leq |p - pqx_j| + q \underbrace{|px_j - s|}_{<\frac{1}{2}} < 1 + \frac{q}{2} \leq q$$

und somit hat $r = p - s \cdot q$ die Eigenschaft, daß $|r| < q$. Ist $r < 0$, so ersetzen wir s durch $s - 1$ und r durch $q + r$, was dann zwischen 0 und $q - 1$ liegt, und erhalten die Darstellung

$$p = sq + r, \qquad s > 0, \quad 0 \leq r < q,$$

die genau die *Ganzzahldivision mit Rest* ist.

[37] Daß die Multiplikation $B^{e-e'} r$ nicht als Langszahlmultiplikation geschrieben ist, ist volle Absicht, denn diese Operation kann als einfache Schiebeoperation realisiert werden, bei der noch nicht einmal die hinten angehängten Nullen wirklich gespeichert werden müssen.

[38] Achtung: Das a von dort ist jetzt unser q!

Bemerkung 2.15 *1. Zur Berechnung des "Näherungsquotienten" müssen nach (2.15) maximal*

$$1 + \lfloor \log_2 w + \log_2 (\ell(p) + 1) \rfloor$$

Iterationen durchgeführt werden, eine Zahl die vor *Beginn des Newton–Verfahrens bekannt ist. So wird aus einem iterativen Verfahren ein Algorithmus . . .*

2. Da $\ell(p) < 2^{w-1}$ ist, beschränkt sich der maximale Aufwand somit auf höchstens $w + \log_2 w$ Multiprecision–Operationen. Und das ist nun wirklich nicht übermäßig viel.

3. Die Iteration (2.13) verlangt noch nach einer genaueren Betrachtung, denn die Stellenzahl von x_j kann ja prinzipiell so groß werden, daß man in den Multiplikationen eine Aufwand betreibt, der schlichtweg inakzeptabel wird. Und in der Tat wächst die Stellenzahl auch ganz schön flott: Sei $s_j = \ell(x_j)$ die Länge der Mantisse von x_j, dann hat $a \otimes x_j$ ja höchsten $s_j + 1$ Stellen und $x_j \otimes (a \otimes x_j)$ maximal $2s_j + 1$ Stellen. Somit ist $s_{j+1} \leq 2s_j + 1$ und damit[39]

$$s_j \leq \left(2^{j+1} - 1\right) s_0, \qquad j \in \mathbb{N}_0,$$

Und da unser Startwert x_0 "einziffrig" gewählt war und deswegen $s_0 = 1$ ist, haben wir nach den $1 + \log_2 (w\,\ell(p) + w)$ Iterationsschritten also insgesamt gerade einmal höchstens

$$2^{2+\log_2(w\,\ell(p)+w)} - 1 \leq 4w\ (\ell(p) + 1)$$

Stellen angehäuft, was immer noch ein $O(\ell(p))$ ist – es kann also nichts schlimmes passieren, insbesondere wenn man bedenkt, daß man sowieso mindestens $\ell(p)$ Stellen im "finalen" x_j benötigt, wenn $p\,x_j$ auch nur ein halbwegs vernünftiges Ergebnis sein soll.

4. Nochmals: Das Ganze funktioniert nur deswegen so gut und mit moderatem Aufwand, weil das Newton–Verfahren quadratisch *konvergiert und diese quadratische Konvergenz hier voll ausgenützt wird.*

5. Die Effizienz der Newton–Iteration lebt nun im wesentlichen von der Effizienz der Multiplikation – je schneller wir also multiplizieren können, desto schneller können wir dann auch dividieren.

Übung 2.1 Formulieren Sie den Algorithmus zur "schnellen" Division über das Newton–Verfahren im Detail.

[39] Das ist eine ganz einfache Induktion.

2.2 Der euklidische Algorithmus

Nachdem wir nun also ganze Zahlen teilen und den Divisionsrest, also den Wert "p modulo q" mehr oder weniger effizient bestimmen können, wollen wir uns an das Rechnen mit solchen Resten machen. Um etwas mehr Freiheit zu bekommen, insbesondere, um das Rechnen mit ganzen Zahlen und Polynomen auf einen Schlag erledigen zu können, brauchen wir ein bißchen algebraische Terminologie – eigentlich nur Abstrahierungen gemeinsamer Eigenschaften.

Definition 2.16 *1. Ein Element $a \in R$ eines Rings R* teilt *$b \in R$, in Zeichen $a \mid b$, wenn es ein $c \in R$ gibt, so daß $ac = b$.*

2. *Ein kommutativer Ring R mit Einselement heißt* Integritätsbereich[40] *oder* Integritätsring*, wenn er* nullteilerfrei *ist, das heißt, wenn es kein $0 \neq a, b \in R$ gibt, so daß $ab = 0$.*

3. *Ein Integritätsring R heißt* euklidischer Ring*, wenn es außerdem eine "Bewertungsfunktion" oder* Euklidische Funktion *$d : R \to \mathbb{N}_0 \cup \{-\infty\}$ gibt, so daß es für alle $p, q \in R$, $q \neq 0$, einen* Quotienten *$s \in R$ und einen* Rest *$r \in R$ gibt, so daß*

$$p = sq + r, \qquad d(r) < d(q).$$

Wir schreiben dann auch $s =: p/q$ und $r =: (p)_q$.

Bemerkung 2.17 *1. Beide Eigenschaften sind wichtig für das praktische Rechnen in Ringen. Die Nullteilerfreiheit sorgt dafür, daß man "kürzen" darf, das heißt, ist $a \neq 0$ und $ab = ac$, dann ist auch $b = c$, während die zweite Eigenschaft uns die Existenz einer Division mit Rest gibt, wobei der Rest* einfacher *ist als der Divisor.*

2. *Jede euklidische Funktion hat die Eigenschaft, daß $d(0) < d(a)$ für alle $a \in R \setminus \{0\}$. Gäbe es nämlich ein $a \in R \setminus \{0\}$, so daß $d(a) \leq d(R)$, dann erhalten wir mit $p = q = a$, daß eine euklidische Darstellung von a die Form*

$$a = sa + \underbrace{(1-s)a}_{=r}, \qquad s \in R,$$

haben muß, aber ganz egal wie wir s wählen, würde jeder dieser Reste $d((1-s)r) \geq d(a)$ erfüllen und damit wäre der Ring nicht euklidisch.

3. *Es hat zwar nicht jede euklidische Funktion die Eigenschaft*

$$d(a \cdot b) \geq d(a), \qquad a, b \in R \setminus \{0\}, \tag{2.16}$$

die man von den "normalen" euklidischen Funktionen für $\mathbb{Z}$ und $\mathbb{F}[x]$ kennt und fast für "selbstverständlich" hält, aber für jeden Ring Integritätsring R gibt es so eine Bewertungsfunktion, und zwar die minimale euklidische Funktion*, die als punktweises Minimum* aller *möglichen euklidischen Funktionen definiert ist, siehe (Gathen & Gerhard, 1999, Übung 3.5). Wir können und werden also immer annehmen, daß wir die* minimale *euklidische Funktion verwenden.*

[40] Auf Englisch "*integral domain*".

4. *Der Wert $d(a) = -\infty$ ist nur für $a = 0$ zulässig – muß aber nicht angenommen werden.*

Beispiel 2.18 *(Euklidische Ringe)*

1. *Die ganzen Zahlen $\mathbb{Z}$ bilden zusammen mit der Funktion $d = |\cdot|$ einen euklidischen Ring.*
2. *Die Polynome $\mathbb{K}[x]$ bilden einen euklidischen Ring mit der Funktion $d = \deg$, wobei $\deg 0 = -\infty$.*
3. *Ein Körper $\mathbb{K}$ ist ein euklidischer Ring mit $d = (1 - \delta_0)$.*
4. *Eine etwas obskurere euklidische Funktion auf $\mathbb{Z}$ ist $d(3) = 2$ und $d = |\cdot|$ sonst. Diese euklidische Funktion*[41] *erfüllt* nicht *die Bedingung (2.16), da $d(-1 \cdot 3) = d(-3) = 3 > 2 = d(3)$ ist. Trotzdem ist aber immer noch $d(0)$ minimal ...*

In einem beliebigen Ring (hier brauchen wir also keinen euklidischen Ring) können wir den *größten gemeinsamen Teiler* $\text{ggT}\,(a, b)$ und das *kleinste gemeinsame Vielfache* $\text{kgV}\,(a, b)$ von zwei Elementen $a, b \in R$ definieren, nämlich durch die Forderungen[42]

$$\begin{array}{lll} \text{ggT}\,(a,b) \,|\, a, & \text{ggT}\,(a,b) \,|\, b, & c \,|\, a,\ c \,|\, b \Longrightarrow c \,|\, \text{ggT}\,(a,b), \\ a \,|\, \text{kgV}\,(a,b), & b \,|\, \text{kgV}\,(a,b)b, & a \,|\, c,\ b \,|\, c \Longrightarrow \text{kgV}\,(a,b) \,|\, c. \end{array} \tag{2.17}$$

Dabei sind jeweils die ersten beiden Eigenschaften für "gemeinsamer Teiler" bzw. "gemeinsames Vielfaches" zuständig, während die dritte Eigenschaft für das "größter" bzw. das "kleinste" sorgt. Nachdem nicht so ganz klar ist, ob das Nullelement eines Rings ein anderes Element teilt oder nicht, *definieren* wir außerdem, daß

$$\text{ggT}\,(a, b) = 0 \qquad \text{falls} \qquad a = 0 \text{ oder } b = 0, \tag{2.18}$$

was natürlich auch den Fall $a = b = 0$ einschließen soll[43].

Wie gesagt, definieren können wir ggT und kgV auf beliebigen Ringen, zum Ausrechnen ist jedoch ein euklidischer Ring hilfreich.

Algorithmus 2.19 *(Euklidischer Algorithmus)*
Gegeben: *$a, b \in R$, R euklidischer Ring.*

1. *Solange $b \neq 0$ setze*

$$\begin{array}{rcl} c & \leftarrow & (a)_b \\ a & \leftarrow & b \\ b & \leftarrow & c \end{array}$$

[41] Euklidisch ist sie dadurch, daß der Divisionsrest bei Division in $\{-1, 0, 1\}$ gewählt wird.
[42] Per Definitionem ist $\text{ggT}\,(a, 0) = 0$.
[43] Hier sind wir ebenfalls wieder konsistent mit der anschaulichen Tatsache, daß $\text{ggT}\,(a, a) = a$ ist.

Ergebnis: $a = \mathrm{ggT}\,(a, b)$.

Proposition 2.20 *Der euklidische Algorithmus terminiert für alle $a, b \in R$ nach endlich vielen Schritten und berechnet* $\mathrm{ggT}\,(a, b)$.

Beweis: Wir schreiben den Algorithmus als

$$r_{j+1} = (r_{j-1})_{r_j}\,, \qquad j \in \mathbb{N}, \qquad r_0 = a,\, r_1 = b.$$

Da $d\,(r_{j+1}) < d\,(r_j)$ bricht das Verfahren nach endlich vielen Schritten ab. Die Beziehung können wir auch schreiben als

$$r_{j+1} = r_{j-1} - s_j\, r_j, \qquad s_j \in R, \qquad j \in \mathbb{N}, \tag{2.19}$$

woraus induktiv sofort folgt, daß $\mathrm{ggT}\,(a, b) \,|\, r_j$, $j \in \mathbb{N}_0$. Ist nun $r_{n+1} = 0$ und $r_n \neq 0$, dann gilt $r_n \,|\, r_{n-1}$, also teilt r_n wegen

$$r_{j-1} = s_j r_j + r_{j+1}, \qquad j = n-1, \ldots, 1,$$

auch $r_{n-2}, r_{n-3}, \ldots, r_1 = b, r_0 = a$ und damit ist $r_n = \mathrm{ggT}\,(a, b)$. □

Bemerkung 2.21

1. *Indem wir den Divisionsrest bei Division durch q nicht in der Menge $\{0, \ldots, q-1\}$ sondern in*

$$\left\{-\frac{q}{2}+1, \ldots, \frac{q}{2}\right\}, \qquad q \in 2\mathbb{Z} \setminus \{0\},$$

$$\left\{-\frac{q-1}{2}, \ldots, \frac{q-1}{2}\right\}, \qquad q \in 2\mathbb{Z}+1,$$

wählen, erhalten wir einen schnelleren Abfall der Bewertungsfunktion als im "normalen" euklidischen Algorithmus. Außerdem sehen wir dabei, daß sich die Bewertungsfunktion in jedem Iterationsschritt nicht um 1 *verringert, wie im Beweis verwendet, sondern halbiert, so daß der euklidische Algorithmus hier sogar in* $\log_2 q$ *Schritten* garantiert.

2. *Diese schnelle Konvergenz ist auch der Fall, wenn man die Standardbewertung mit nichtnegativem Rest verwendet. Ist nämlich $p = qs + r$, dann ist für $q \leq \frac{p}{2}$ trivialerweise auch $r < q \leq \frac{p}{2}$, ist hingegen $q > \frac{p}{2}$, dann muss $s = 1$ sein und somit ist $r = p - q < \frac{p}{2}$. Also gilt immer $r < \frac{p}{2}$ und wir haben wieder Terminierung nach einer logarithmischen Anzahl von Schritten.*

Beispiel 2.22 *Betrachten wir die Berechnung von* $\mathrm{ggT}\,(21, 13) = 1$. *Mit dem "naiven" Verfahren erhalten wir die Folge*

j	r_j	r_{j+1}
0	21	13
1	13	8
2	8	5
3	5	3
4	3	2
5	2	1

mit vorzeichenbehaftetem Divisionsrest hingegen ergibt sich

j	r_j	r_{j+1}
0	21	13
1	13	-5
2	-5	-2
3	-2	1

was immerhin etwas schneller ist.

Bemerkung 2.23 *Besonders viele Iterationsschritte benötigt der euklidische Algorithmus für die* Fibonaccizahlen $x_{n+1} = x_n + x_{n-1}$, $x_0 = x_1 = 1$, *siehe (Knuth, 1998).*

Dieser euklidische Algorithmus funktioniert zwar sehr gut mit ganzen Zahlen, aber z.B. mit Polynomen müssen wir noch mehr fordern, um den ggT zweier Polynome eindeutig zu bekommen. Da nämlich in $\mathbb{K}[x]$ "einfach so" mit Körperelementen multipliziert wird, ist jedes Element der Form $k \cdot \text{ggT}\ (f, g)$, $k \in \mathbb{K} \setminus \{0\}$ ebenfalls ein ggT von $f, g \in \mathbb{K}[x]$. Hier bietet sich Normieren an:

> *Wir definieren als* ggT (f, g) *dasjenige Polynom, das von der Form* $x^n + \cdots$, *also* monisch *ist.*

Um dieses Konzept auf einen beliebigen euklidischen Ring zu übertragen, brauchen wir wieder ein bißchen mehr Terminologie.

Definition 2.24 *Sei* R *ein kommutativer Ring mit* 1.

1. *Ein Element* $a \in R$ *heißt* Einheit, *wenn es in* R *(multiplikativ) invertierbar ist, das heißt, wenn es ein* $b \in R$ *gibt, so daß* $ab = 1$. *Mit* R^* *bezeichnen wir die Menge aller Einheiten in* R.

2. *Zwei Elemente* $a, b \in R$ *heißen* assoziiert, *in Zeichen* $a \sim b$, *wenn* $a \in R^* \cdot b$.

3. *Die* Normalform ν *in einem* euklidischen *Ring* R *ist eine Abbildung* $\nu : R \to R$ *mit der Eigenschaft, daß*

$$\textit{(i)}\ a \sim \nu(a), \qquad \textit{(ii)}\ \nu(a) = \nu(b) \iff a \sim b, \qquad \textit{(iii)}\ \nu(a \cdot b) = \nu(a) \cdot \nu(b). \tag{2.20}$$

4. *Zu einer gegebenen Normalform ist der* Leitkoeffizient $\lambda(a) \in R^*$ *von* $a \in R$ *als Lösung von*

$$a = \lambda(a) \cdot \nu(a)$$

definiert.

Übung 2.2 Es sei d die minimale euklidische Funktion eines euklidischen Rings. Zeigen Sie:

1. Es ist $d(1) = d(a) \leq d(b)$ für alle $a \in R^*$ und $b \in R \setminus \{0\}$.
2. $d(1) \in \{0, 1\}$.

♢

Bemerkung 2.25 *1. Die Einheiten in $\mathbb{Z}$ sind ± 1, die Einheiten in $\mathbb{K}[x]$ bereits $\mathbb{K} \setminus \{0\}$. Im Ring der* Laurentpolynome

$$f(x) = \sum_{j \in \mathbb{Z}} f_j \, x^j, \qquad \# \{j \ : \ f_j \neq 0\} < \infty$$

sind sogar alle Monome der Form $k \, x^j$, $j \in \mathbb{Z}$, $k \in \mathbb{K} \setminus \{0\}$ Einheiten.

2. *Die Normalform stellt eine* konsistente *Auswahl eines besonderen Elements aus der Menge $R^* \, a$ dar. Außerdem folgt aus (2.20), daß $\nu(R^*) = 1$: Da die Normalformen zueinander assoziierter Elemente von R identisch ist, haben alle Elemente von R^** dieselbe *Normalform $a = \nu(R^*) = \nu(1) \in R^*$ und für diese gilt*

$$a = \nu(1) = \nu(1 \cdot 1) = \nu(1) \cdot \nu(1), = a^2,$$

also $a = 1$, weil unser euklidischer Ring ein Integritätsbereich ist.

3. *Wählt man als Normalform auf $\mathbb{Z}$ die Abbildung $\nu = |\cdot|$, dann ist $\lambda(p) = \operatorname{sgn} p$.*
4. *Ist $g = \operatorname{ggT}(a, b)$, dann erfüllt auch jedes Element von $R^* \, b$ die Bedingungen aus (2.17). Erfüllen umgekehrt $g, g' \in R$ diese Bedingungen, dann gilt $g \mid g'$ und $g' \mid g$, also $g/g' \in R^*$ mit $(g/g') = g'/g$. Und da $g = (g/g') \; g'$ ist also $g \sim g'$. Mit anderen Worten: Alle ggTs sind zueinander assoziiert.*
5. *Das erlaubt es uns, nun einen* eindeutigen ggT *zu definieren, indem man stets die Normalform $\nu(\operatorname{ggT}(a, b))$ eines "irgendwie" berechneten* ggT *wählt.*

Die Grundidee des nächsten Verfahrens, des *erweiterten euklidischen Algorithmus*, besteht nun darin, den euklidischen Algorithmus so zu modifizieren, daß nur *normalisierte* Teiler produziert werden, denn dann ist der ggT der am Schluß übrigbleibt, ebenfalls normalisiert und somit **der** ggT. Wir formulieren das ganze diesmal mit Zählindex, denn das wird beim Beweis einfacher. Ach ja, und wenn wir schon dabei sind, dann machen wir uns gleich noch ein bißchen mehr Arbeit.

Algorithmus 2.26 *(Erweiterter Euklidischer Algorithmus)*
Gegeben: *$a, b \in R$, R euklidischer Ring mit Normalform.*

1. *Normalisiere und initialisiere:*

$$\begin{array}{lll} r_0 \leftarrow \nu(a), & p_0 \leftarrow \lambda^{-1}(a), & q_0 \leftarrow 0, \\ r_1 \leftarrow \nu(b), & p_1 \leftarrow 0, & q_1 \leftarrow \lambda^{-1}(b). \end{array} \tag{2.21}$$

2. *Setze* $j \leftarrow 1$.

3. *Solange* $r_j \neq 0$

   (a) *Division mit Rest:*
   $$s \leftarrow r_{j-1} \,/\, r_j, \qquad r_{j+1} \leftarrow (r_{j-1})_{r_j}\,.$$

   (b) *Update von* p *und* q
   $$p_{j+1} \leftarrow (p_{j-1} - s\,p_j)\ \lambda^{-1}(r_{j+1})\,, \qquad q_{j+1} \leftarrow (q_{j-1} - s\,q_j)\ \lambda^{-1}(r_{j+1})\,, \quad (2.22)$$

   (c) *Normalisiere:*
   $$r_{j+1} \leftarrow \nu(r_{j+1})\,.$$

   (d) *Setze* $j \leftarrow j+1$.

4. *Setze* $j \leftarrow j-1$.

Ergebnis: $r_j = \mathrm{ggT}\,(a,b)$ *und* $(p,q) = (p_j, q_j) \in R^2$*, so daß*

$$p\,a + q\,b = \mathrm{ggT}\ (a,b)\,.$$

Bemerkung 2.27 *1. Die* Bézout–Koeffizienten p, q *sind ein mehr als schönes Abfallprodukt des erweiterten euklidischen Algorithmus. Sind nämlich* a, b *teilerfremd, also* $\mathrm{ggT}\,(a,b) = 1$*, dann ist* $q\,b \equiv 1$ *modulo* a *und damit ist* $q \equiv b^{-1}$ *modulo* a*. So ganz nebenbei haben wir also auch das Problem gelöst, wie wir in dem endlichen Körper* $\mathbb{F}_p = \mathbb{Z}/\langle p\rangle$*,* $p \in \mathbb{N}$ *prim, invertieren.*

2. Man könnte die Bézout–Koeffizienten auch mit dem "klassischen" (?) Verfahren bestimmen, den euklidischen Algorithmus "von hinten aufzurollen", indem man die Zerlegungen $r_{j+1} = r_{j-1} - s_j r_j$ *wieder einsetzt:*

$$\begin{aligned}\mathrm{ggT}\,(a,b) &= r_n = r_{n-2} - s_{n-1}r_{n-1} = r_{n-2} - s_{n-1}\,(r_{n-3} - s_{n-2}r_{n-2})\\ &= -s_{n-1}r_{n-3} + (1 + s_{n-1}s_{n-2})\ r_{n-2}\\ &= p_j\,r_{j-1} + q_j\,r_j, \qquad j = n-1,\ldots,1,\end{aligned}$$

wobei man die Rekursionsformeln

$$p_{j-1} = q_j \qquad \textit{und} \quad q_{j-1} = p_j - s_{j-1}q_j$$

hat. Um diese "Rückwärtsrekursion" aber rechnen zu können, müßte man alle Werte r_j*,* s_j*, die man im Laufe des euklidischen Algorithmus erhalten hat, zwischenspeichern, was sicherlich ineffizient ist.*

3. Was in Bemerkung 2.21 gesagt wurde, gilt sinngemäß natürlich auch für den erweiterten euklidischen Algorithmus (auf $\mathbb{Z}$*), nur darf man dann fleißig normieren.*

Satz 2.28 *Sei R ein euklidischer Ring mit Normalform und $a, b \in R$. Der erweiterte euklidische Algorithmus*

1. *terminiert nach einer endlichen Anzahl von Schritten,*
2. *berechnet den normalisierten* ggT*,*
3. *berechnet Zahlen p, q, so daß*

$$p\,a + q\,b = \text{ggT}\ (a, b)\,. \tag{2.23}$$

Beweis: 1. und 2. sind exakt wie in Proposition 2.20 – daß der ggT normalisiert ist, folgt daraus, daß wir in jedem Schritt r_j explizit normalisieren.
(2.23) ist eine Konsequenz aus der allgemeineren Invariante

$$p_j\,a + q_j\,b = r_j, \qquad j \in \mathbb{N}_0, \tag{2.24}$$

die wir per Induktion beweisen wollen; hierbei bezeichnet r_j bereits die *normalisierte* Variante. Für $j = 0, 1$ ist (2.24) aus (2.21) offensichtlich:

$$r_0 = \nu(a) = \lambda^{-1}(a)\,a + 0\,b, \qquad \text{und} \qquad r_1 = \nu(b) = 0\,a + \lambda^{-1}(b)\,b.$$

Ansonsten setzen wir

$$\lambda_j = \lambda\left((r_{j-1})_{r_j}\right), \qquad j \in \mathbb{N}$$

und verwenden (2.22) für

$$\begin{aligned} p_{j+1}\,a + q_{j+1}\,b &= \lambda_j^{-1}\left((p_{j-1} - s\,p_j)\ a + (q_{j-1} - s\,q_j)\ b\right) \\ &= \lambda_j^{-1}\left(p_{j-1}a + q_{j-1}b - s_j(p_j a + q_j b)\right) \\ &= \lambda_j^{-1}\left(r_{j-1} - s_j r_j\right) = \lambda_j^{-1}\left(r_{j-1}\right)_{r_j} = \lambda_j^{-1}\lambda\left((r_{j-1})_{r_j}\right)\ \nu\left(r_{j-1}\right)_{r_j} \\ &= r_{j+1} \end{aligned}$$

□

Beispiel 2.29 *Jetzt aber erst einmal ein Beispiel für den erweiterten euklidischen Algorithmus.Sei $R = \mathbb{Z}$, $a = 126$, $b = 35$. Wir betrachten das folgende Tableau*

# It.	r_{j-1}	r_j	p_j	q_j
0	0	126	1	0
1	126	35	0	1

Nun berechnen wir, daß $126 = 3 \times 35 + 21$, also ist $r_2 = 35$, sowie

$$p_2 = 1 - 3 \times 0 = 1 \qquad \textit{und} \qquad q_2 = 0 - 1 \times 3 = -3,$$

was unser Tableau zu

# It.	r_{j-1}	r_j	p_j	q_j
0	0	126	1	0
1	126	35	0	1
2	35	21	1	-3

erweitert. Und in der Tat: $126 - 3 \times 35 = 21$. *Ziehen wir das so weiter durch, so erhalten wir*

# It.	r_{j-1}	r_j	p_j	q_j
0	0	126	1	0
1	126	35	0	1
2	35	21	1	-3
3	21	14	-1	4
4	14	7	2	-7

was uns auch die Darstellung $\mathrm{ggT}\,(126, 35) = 7 = 2 \times 126 - 7 \times 35$ *liefert.*

Algorithmus 2.30 *("Beschleunigter" erweiterter euklidischer Algorithmus ohne Normierung für ganze Zahlen)*
Gegeben: $a, b \in \mathbb{Z}$.

1. *Setze* $p = q' = 0, p' = q = 1$.
2. *Solange* $b \neq 0$
   (a) *Bestimme* $r, s \in \mathbb{Z}$ *so, daß*

$$a = sb + r, \qquad 0 \leq r \in \left\{ -\left\lfloor \frac{b}{2} \right\rfloor, \left\lfloor \frac{b}{2} \right\rfloor \right\}.$$

   (b) *Setze*

$$\begin{aligned} a &\leftarrow b \\ b &\leftarrow r \\ (p, p') &\leftarrow (p', p' - s \cdot p) \\ (q, q') &\leftarrow (q', q' - s \cdot q) \end{aligned}$$

Ergebnis: $\mathrm{ggT}\,(a, b)$ *in Variable* b *sowie* p, q *mit der Eigenschaft* $p\,a + q\,b = \mathrm{ggT}\,(a, b)$.

Beispiel 2.31 *Um auch mal ein Beispiel für die Normalisierungsgeschichte zu haben, jetzt mal ein Beispiel für den euklidischen Algorithmus für zwei Polynome, nämlich*

$$f = 3x^3 + 6x^2 + 3x \qquad \textit{und} \qquad g = \frac{1}{2}x^2 - \frac{1}{2}$$

Nach (2.21) wird unser Tableau nun wie folgt initalisiert:

# It.	r_{j-1}	r_j	p_j	q_j
0	0	x^3+2x^2+x	$\frac{1}{3}$	0
1	x^3+2x^2+x	x^2-1	0	2

Jetzt also zur Division mit Rest, die uns

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

also, nach Normalisierung, $r_2 = x+1$ *und*

$$p_2 = \left(\frac{1}{3} - 0\cdot(x+2)\right)/2 = \frac{1}{6} \qquad \textit{und} \qquad q_2 = \left(0-2\cdot(x+2)\right)/2 = -(x+2)$$

liefert. In der Tat ist

$$\begin{aligned} p_2\, f + q_2\, g &= \frac{1}{6}\left(3x^3+6x^2+3x\right) - (x+2)\left(\frac{1}{2}x^2-\frac{1}{2}\right) \\ &= \frac{1}{2}x^3+x^2+\frac{1}{2}x-\frac{1}{2}x^3-x^2+\frac{1}{2}x+1 = x+1, \end{aligned}$$

dem normalisierten Rest. So erhalten wir also die Rechnung

# It.	r_{j-1}	r_j	p_j	q_j
0	0	x^3+2x^2+x	$\frac{1}{3}$	0
1	x^3+2x^2+x	x^2-1	0	2
2	x^2-1	$x+1$	$\frac{1}{6}$	$-(x+2)$

und da $x+1 \,|\, x^2-1$ *sind wir fertig.*

2.3 Modulares Rechnen

In diesem Kapitel betrachten wir das Rechnen in Restklassenringen und dessen Anwendungen - vor allem werden wir sehen, daß es überraschend viele Anwendungen geben wird, in gewissem Sinne ist modulares Rechnen sogar **das** Hilfsmittel exakter Arithmetik.

Definition 2.32 *Sei* R *ein Ring und* $m \in R$*.*

1. *Wir sagen, zwei Elemente* $a, b \in R$ *sind* kongruent modulo m*, in Zeichen* $a \equiv_m b$*, wenn* $a - b \in \langle m\rangle := m\,R$*.*

2. *Für* $a \in R$ *ist die* Restklasse *oder* Äquivalenzklasse *zu* a *modulo* m *die Menge*

$$[a] := [a]_m := \{b \in R \;:\; a \equiv_m b\}\,.$$

3. *Der* Restklassenring $R/\langle m\rangle = R/mR$ *ist die Menge aller Restklassen mit den Operationen*

$$[a]\cdot[b] = [a\cdot b] \qquad \textit{und} \qquad [a]+[b] = [a+b]\,.$$

Bemerkung 2.33 *1. Der "Standardfall" ist natürlich $R = \mathbb{Z}$ und die Verwendung des Divisionsrests. Dann ist, für $p \in \mathbb{Z}$, der Restklassenring*

$$\mathbb{Z}/\langle p\rangle = \mathbb{Z}_p = \{0,\ldots,p-1\}$$

und wir haben für $a, b \in \mathbb{Z}_p$ die Rechenoperationen

$$a+b := (a+b)_p \qquad \textit{und} \qquad a\cdot b := (a\cdot b)_p\,.$$

2. *Diese Vorgehensweise können wir auf jeden euklidischen Ring übertragen, bei dem der Divisionsrest "eindeutig" geregelt ist, also insbesondere auf die Polynome!*

3. *Ist der Modulus m eine Einheit, dann gibt es genau eine Restklasse! Denn in diesem Fall ist $mR = mm^{-1}R = 1R = R$ und je zwei beliebige Elemente des Ringes sind kongruent modulo m.*

Eine schöne Eigenschaft von Restklassenringen besteht darin, daß wir ihre Einheiten sehr einfach angeben können und daß sich *algorithmisch* entscheiden läßt, ob ein Element eine Einheit ist.

Proposition 2.34 *Sei R ein euklidischer Ring mit Normalform*[44] *und $m \in R$. Dann ist die Äquivalenzklasse $[a]$ genau dann eine Einheit in $R/\langle m\rangle$, wenn* $\mathrm{ggT}\,(a,m) = 1$.

Beweis: Ist $\mathrm{ggT}\,(a,m) =: b \neq 1$, dann gibt es $p, q \in R$, $[q] \neq [0]$, so daß $a = bp$, $m = bq$, und damit ist $[q]\cdot[a] = [qa] = [bpq] = [pm] = [0]$, weswegen $[a]$ keine Einheit sein kann. Ist andererseits $\mathrm{ggT}\,(a,m) = 1$, dann verwenden wir unseren erweiterten euklidischen Algorithmus 2.26 und erhalten $p, q \in R$, so daß

$$1 = \mathrm{ggT}\,(a,m) = pa + qm \equiv_m p\,a,$$

so daß $a^{-1} = p$ in $R/\langle m\rangle$. □

Besonders schön sind natürlich Ringe, in denen man nach Herzenslust auch dividieren darf, also Ringe, in denen alle von Null verschiedenen Elemente Einheiten sind.

Definition 2.35 *Sei R ein euklidischer Ring mit Normalform.*

1. *Ein Element $a \in R$ heißt* irreduzibel*, wenn*

$$\mathrm{ggT}\,(a,b) = 1, \qquad b \in R\setminus\{0,a\}.$$

[44] Diese Eigenschaften brauchen wir im wesentlichen, um auch wirklich rechnen zu können.

2. *Man bezeichnet den Ring R als* Körper, *wenn $R^* = R \setminus \{0\}$.*

Korollar 2.36 *Sei R ein euklidischer Ring mit Normalform. Für $m \in R$ ist $R/\langle m\rangle$ genau dann ein Körper, wenn m irreduzibel ist.*
Insbesondere ist $\mathbb{F}_p := \mathbb{Z}_p = \mathbb{Z}/\langle p\rangle$ ein Körper wenn p eine Primzahl ist.

Übung 2.3 Zeigen Sie: Ist m reduzibel, dann hat $R/\langle m\rangle$ einen Nullteiler.

Beispiel 2.37 *Bei Polynomen hängt Irreduzibilität vom Grundkörper ab! So ist beispielsweise das Polynom $f(x) = x^2 - 2$ über $\mathbb{Q}[x]$ irreduzibel und damit ist $\mathbb{Q}[x]/\langle x^2-2\rangle$ ein Körper, wohingegen es über $\mathbb{R}$ in $f(x) = \left(x+\sqrt{2}\right)\left(x-\sqrt{2}\right)$ zerfällt, also ist $\mathbb{R}[x]/\langle x^2-2\rangle$ kein Körper.*
Im ersten, rationalen Fall ist

$$\mathbb{Q}[x] \simeq \text{span}_{\,\mathbb{Q}}\,\{1, x\}$$

und die Inversen der beiden Basiselemente erhalten wir, indem wir mit dem erweiterten euklidischen Algorithms jeweils den ggT *berechnen, also*

# It.	r_{j-1}	r_j	p_j	q_j
0	0	x^2-2	1	0
1	x^2-2	x	0	1
2	x	1	$-\frac{1}{2}$	$\frac{1}{2}\,x$

das heißt

$$1 = -\frac{1}{2}\left(x^2-2\right) + \frac{1}{2}\,x\,x \equiv_{x^2-2} = \left(\frac{1}{2}x\right)\,x, \qquad \Longrightarrow \qquad x^{-1} \equiv_{x^2-2} \frac{1}{2}\,x.$$

Beispiel 2.38 *("Hashing")*
Eine ganz einfache Anwendung modularen Rechnens ist das "Hashing" oder "Fingerprinting": Man sortiert große Datensätze (z.B. digitalisierte Fingerabdrücke, DNS–Daten) nicht systematisch, sondern speichert zusätzlich die Information "modulo p" für eine oder mehrere Primzahlen. Solche Datensätze kann man als riesige Zahlen $a, b \in \mathbb{N}$ interpretieren und anstatt sofort nachzuprüfen, ob $a = b$ gilt, testet man zuerst einmal, ob

$$a \equiv_{p_1} b, \quad a \equiv_{p_2} b, \quad \ldots \quad a \equiv_{p_n} b$$

für sinnigerweise unterschiedliche (Prim-) Zahlen $p_1, \ldots, p_n$ erfüllt ist. Diese Test kann man parallelisieren und die "fingerprints" $(a)_{p_j}$, $j = 1, \ldots, n$, vorberechnen. Geht dann auch nur einer der Tests schief[45]*, dann kann man sich den "großen" Vergleich sparen.*

[45] Was hier sozusagen der Erfolgsfall ist.

Wir können aber auch beide Methoden, um mittels Korollar 2.36 einen Körper zu konstruieren, kombinieren, um einen einen endlichen Körper zu erhalten, der eine Primzahlpotenz von Elementen enthält und in dem wir effizient rechnen können, indem wir den erweiterten euklidischen Algorithmus für ganze Zahlen mit dem erweiterten euklidischen Algorithmus für Polynome kombinieren.

Proposition 2.39 *Sei $p \in \mathbb{N}$ eine Primzahl und $f \in \mathbb{F}_p[x]$ ein irreduzibles Polynom vom Grad n. Dann ist $\mathbb{F}_p[x]/\langle f\rangle$ ein endlicher Körper mit p^n Elementen.*

Beweis: Daß $\mathbb{F}_p[x]/\langle f\rangle$ ein Körper ist ist, folgt sofort mit einer zweimaligen Anwendung von Korollar 2.36. Da außerdem für $f(x) = x^n + \cdots \in \Pi_n$ die Beziehung[46] $\mathbb{K}/\langle f\rangle \simeq \Pi_{n-1} \simeq \mathbb{K}^n$ gilt, ist also auch $\mathbb{F}_p/\langle f\rangle \simeq \mathbb{F}_p^n$ und daher

$$\#\mathbb{F}_p/\langle f\rangle = (\#\mathbb{F}_p)^n = p^n.$$

□

2.4 Adjungieren von Wurzeln

Nach diesen "Vorarbeiten" kehren wir nochmals zu den *adjungierten Wurzeln* aus Proposition 1.2 zurück und sehen uns diesen Prozess nun aber genauer an. Motiviert ist das Ganze durch (Stewart, 1975, S. 89–91), wo Rechnen modulo Polynome verwendet wird, um die komplexen Zahlen einzuführen ohne den reellen Zahlen einen Stellenwert zu geben, den sie nach Meinung des Autors nicht verdienen. Computeralgebraisch ist es ein schöner Trick, um sich um die Nichtauflösbarkeit polynomialer Gleichung vom Grad ≥ 4 "herumzumogeln" und trotzdem mit den Ergebnissen weiterrechnen zu können. Vornehmer könnte man es auch als die *algorithmische Version der algebraischen Körpererweiterungen* .

Sei also R ein Ring und $f \in R[x]$ ein *irreduzibles* Polynom, dessen Nullstellen oder *Wurzeln* wir zur R adjungieren wollen. Formal ist Irreduzibilität eigentlich noch nicht einmal eine Einschränkung, denn wäre das Polynom f faktorisierbar, dann müssten wir halt die Wurzeln, die zu den Faktoren gehören, separat andjungieren. Wie man das Polynom faktorisiert ist natürlich eine andere Geschichte, aber immerhin merkt man die Existenz einer Faktorisierung spätestens dann, wenn sich ein Polynom nicht invertieren lässt.

Mathematisch verwenden wir wieder einmal Korollar 2.36, das uns sagt, daß

$$\mathbb{K} := R[x]/\langle f\rangle \simeq \Pi_{n-1}(R), \qquad f(x) = x^n + \cdots,$$

ein Körper ist, sofern nur f irreduzibel ist. Ein Element $p \in \mathbb{K}$ können wir dann als Polynom $p \in \Pi_{n-1}(R)$ oder eben auch als Vektor[47] $p \in R^n$ darstellen[48], also

$$p(x) = p_{n-1}\, x^{n-1} + \cdots + p_0, \qquad p_j \in R, \quad j \in \mathbb{Z}_n.$$

[46] Dieser Isomorphismus hängt nur von *Grad* von f, nicht von f selbst ab, das heißt, alle diese Körper sind zueinander isomorph – aber man muß natürlich die Rechenregeln entsprechend anpassen.

[47] Eigentlich sind Vektorräume nur über Körpern definiert, wenn wir ganu präzise sein wollten, müssten wir hier von einem *Modul* sprechen, aber wenn wir ehrlich sind, dann geht es uns eigentlich nur um n–Tupel, nicht um das, was man arithmetisch mit ihnen anstellen kann.

[48] Daß wir hier dieselbe Notation für verschiedene Darstellungen von p verwenden, ist ein verzeihlicher Mißbrauch von Notation – schließlich ist es ja immer genau dasselbe Objekt.

Addition und Subtraktion in $\mathbb{K}$ sind nun einfach, nämlich

$$p \oplus q = (p_{n-1}, \ldots, p_0) + (q_{n-1}, \ldots, q_0) = (p_{n-1} + q_{n-1}, \ldots, p_0 + q_0)\,,$$

die Multiplikation ist $p \otimes q = (pq)_f$ und die Division ist

$$p \oslash q = p \otimes a, \qquad qa + fb = \mathrm{ggT}\,(q, f) = 1,$$

wie in (2.23), wobei a, b natürlich über den *erweiterten euklidischen Algorithmus* 2.26 berechnet werden. Um die Wurzeln eines faktorisierbaren[49] Polynoms $f = f_1 \cdots f_m$ zu adjungieren, geht man einfach schrittweise vor:

$$\mathbb{K}_j = \mathbb{K}_{j-1}[x]/\langle f_j \rangle, \quad j = 1, \ldots, m, \qquad \mathbb{K}_0 = R.$$

Beispiel 2.40 *Das Adjungieren "klassischer" Wurzeln $\sqrt{a}$ zu $\mathbb{Q}$ besteht nun also im Übergang zum Körper $\mathbb{K} := \mathbb{Q}/\langle x^2 - a \rangle$ und jedes Element in $\mathbb{K}$ ist nun natürlich ein Paar $(\alpha, \beta) \simeq \alpha\, x + \beta$. Nun ist $\mathbb{Q} \simeq (0, \mathbb{Q})$ und wegen $x^2 \equiv_f a$ ist auch*

$$(1, 0) \otimes (1, 0) \simeq x \cdot x = x^2 \equiv_f a,$$

das heißt, wir haben genau die Darstellung als $\beta + \alpha\,\sqrt{a}$ und plötzlich ist zwischen der Adjungierung von $\sqrt{2}$ und i zu $\mathbb{Q}$ absolut kein Unterschied mehr, im einen Fall haben wir halt $a = 2$ und im anderen $a = -1$, aber ob a positiv oder negativ ist, das spielt in $\mathbb{Q}$ nun wirklich keine entscheidende Rolle.

Bemerkung 2.41 *In Computeralgebrasystemen werden die so adjungierten Wurzeln normalerweise mit dem Ausdruck `RootOf` dargestellt, wobei hinter dem Schlüsselwort die Koeffizienten des Polynoms f angezeigt werden. Hat man nun mehrere Sätze von Wurzeln assoziiert, so kann es natürlich durchaus passieren, daß diese Koeffizienten selbs aus einem algebraischen Erweiterungskörper stammen und daher selbst wieder "`RootOf`"–Ausdrücke enthalten. Typisch Computeralgebra halt: So ein Ergebnis ist zwar korrekt, aber völlig unbrauchbar. Aber man muss fair sein, denn was soll das arme Programm auch sonst machen? Es ist eher die Schwere des Problems als die Schwäche der Lösung, die einem hier das Leben schwermacht.*

2.5 Mehrfach modulare Arithmetik

Es ist also eine besonders schöne Sache, modular modulo einer Primzahl zu rechnen, oder modulo Primzahl und irreduziblem Polynom wie in Proposition 2.39, aber der Nachteil bei dieser Vorgehensweise besteht darin, daß man große Primzahlen erst mal haben muß – das Auffinden irreduzibler Polynome ist da möglicherweise schon einfacher. Nun gilt aber für beliebiges $m \in \mathbb{N}$

$$q \in \mathbb{Z}_m^* \qquad \Longleftrightarrow \qquad \mathrm{ggT}\,(g, m) = 1 \qquad \Longleftrightarrow \qquad p_j \not| q, \quad j = 1, \ldots, n,$$

[49] Natürlich sollte jeder der Faktoren mindestens Grad 2 haben, denn ansonsten geHören die Nullstellen ja schon zu unserem Ausgangsring.

wobei $p_1, \ldots, p_n$ die *Primfaktoren* von m sind, das heißt, es gibt Zahlen $e_j > 0$, $j = 1, \ldots, n$, so daß

$$m = \prod_{j=1}^{n} p_j^{e_j} =: \prod_{j=1}^{n} q_j.$$

In anderen Worten,

$$q \in \mathbb{Z}_m^* \qquad \Longleftrightarrow \qquad q \not\equiv_{p_j} 0, \quad j = 1, \ldots, n, \tag{2.25}$$

Nehmen wir nun mal an, daß alle Primzahlen, bzw. Primzahlpotenzen, in die wir m zerlegen können, jeweils in *ein* Wort passen, dann könnten wir ja versuchen, für ein $q \in \mathbb{Z}_m^*$ bei der Berechnung von q^{-1} zuerst einmal $(q^{-1})_{q_j}$, $j = 1, \ldots, n$, zu berechnen und dann aus dieser Information die Zahl q^{-1} zusammenzusetzen. Wir stehen also vor dem folgenden Problem:

> *Wie kann man die Zahl q aus den Divisionsresten $r_j := (q)_{q_j}$, $j = 1, \ldots, n$, rekonstruieren? Genauer, wie findet man ein $q' \in \mathbb{N}$, so daß $q \equiv_m q'$ und $q \equiv_{q_j} q'$, $j = 1, \ldots, n$.*

Nun, die Antwort ist erstaunlich einfach und verwendet – nicht ganz zufällig – dieselbe Idee wie bei der Polynominterpolation in Lemma 1.6. Dazu konstruieren wir "Lagrange–Zahlen" $\ell_j \in \mathbb{Z}_m$ mit der Eigenschaft, daß

$$\ell_j \equiv_{q_k} \delta_{jk}, \qquad j, k = 1, \ldots, n, \tag{2.26}$$

denn dann gilt

$$q' = \left(\sum_{j=1}^{n} r_j \, \ell_j \right)_m \qquad \Longrightarrow \qquad q' = \underbrace{r_0 \, m}_{\equiv_{q_j} 0} + \sum_{k=1}^{n} r_k \, \underbrace{\ell_k}_{\equiv_{q_j} \delta_{jk}} \equiv_{q_j} r_j.$$

Und die Berechnung der ℓ_j ist auch einfach: Da die Zahlen[50] $m_j := p_j^{-e_j} m = \frac{m}{q_j}$, $j = 1, \ldots, n$, die Eigenschaft

$$\text{ggT} \left(m_j, p_j^{e_j} \right) = \text{ggT} \, (m_j, q_j) = \text{ggT} \, (m_j, p_j) = 1, \qquad j = 1, \ldots, n, \tag{2.27}$$

haben, können wir $(m_j)_{q_j}$ in $\mathbb{Z}_{q_j}$ mit dem erweiterten euklidischen Algorithmus invertieren und erhalten Zahlen $s_j \in \mathbb{Z}$, $j = 1, \ldots, n$, so daß $s_j \, m_j \equiv_{q_j} 1$, $j = 1, \ldots, n$, und damit sind

$$\ell_j := (s_j \, m_j)_m \,, \qquad j = 1, \ldots, n, \tag{2.28}$$

die gesuchten Faktoren. Damit haben wir also das folgende Rekonstruktionsverfahren.

Algorithmus 2.42 *(CRT[51]–Algorithmus für $\mathbb{Z}$)*
Gegeben: *Paarweise teilerfremde Zahlen[52] $q_1, \ldots, q_n$ und Werte $r_j \in \mathbb{Z}_{q_j}$, $j = 1, \ldots, n$.*

[50] Diese Zahlen sind dann die Produkte der "anderen" Primfaktoren, wenn man einmal einen festgehalten hat.

[51] *CRT* steht für "Chinese Remainder Theorem", die englische Bezeichnung für den chinesischen Restsatz.

[52] Diese Zahlen brauchen keine Potenzen unterschiedlicher Primzahlen sein, paarweise teilerfremd oder *koprim* reicht völlig. Entscheidend ist ja nur, daß ggT $(m_j, q_j) = 1$, $j = 1, \ldots, n$, denn dann kann man in $\mathbb{Z}_{q_j}$ invertieren.

1. *Setze* $m = q_1 \cdots q_n$.

2. *Bestimme mit dem erweitertem euklidischen Algorithmus Zahlen* $a_j, b_j \in \mathbb{Z}$, *so daß*

$$a_j \frac{m}{q_j} + b_j\, q_j = \text{ggT}\ (m/q_j, q_j) = 1,$$

und setze

$$\ell_j \leftarrow \left(a_j \frac{m}{q_j} \right)_m, \qquad j = 1, \ldots, n.$$

3. *Setze*

$$q \leftarrow \left(\sum_{j=1}^{n} r_j\, \ell_j \right)_m$$

Ergebnis: $q \in \mathbb{Z}_m$ *so daß* $q \equiv_{q_j} r_j$, $j = 1, \ldots, n$.

Bemerkung 2.43 1. *Algorithmus 2.42 hat eine ziemliche praktische Bedeutung: Wir können eine große Zahl in* $\mathbb{Z}_m$ *in ihre Reste modulo der einfachen Primfaktoren von* m *zerlegen, die Rechnung in diesen wesentlich kleineren Körpern durchführen und brauchen schließlich "nur" das Ergebnis wieder zusammenzusetzen.*

2. *Was für praktische Anwendungen ebenfalls sehr ansprechend ist, ist die Tatsache, daß die "Zwischenrechnungen" in den individuellen Restklassenringen* $\mathbb{Z}_{q_j}$, $j = 1, \ldots, n$, *nun komplett voneinander unabhängig und damit* parallelisierbar *sind! Wählt man außerdem die Zahlen* q_j *alle so, daß* $\ell\,(q_j) = 0$ *(in einer Multiprecision–Arithmetik), dann habe alle Primfaktoren nur eine* B*-adische Stelle und daher besteht das ganze Restklassenrechnen nur aus* elementaren *Rechenoperationen.*

3. *Stehen die Zahl* m *und damit ihre Primfaktoren* a priori *fest, dann können die Zahlen* ℓ_j, $j = 1, \ldots, n$, *als "Invarianten" der Arithmetik vorberechnet und in einer Tabelle gespeichert werden. Alles was man dann zur Rekonstruktion einer Zahl braucht sind* n *Multiplikationen und* $n - 1$ *Additionen. Ist wieder* $\ell\,(p_j) = 0$, $j = 1, \ldots, n$, *dann benötigt jede Multiplikation (á la Algorithmus 2.5)* $O\,(\ell(m))$ *Operationen und die Additionen nochmals* $O\,(n\,\ell(m))$ *Operationen. Das führt insbesondere zu einem relativ "billigen" Verfahren zur Berechnung des multiplikativen Inversen in* $\mathbb{Z}_m$.

4. *Das Verfahren funktioniert, wie man sich leicht überlegt, nicht nur für ganze Zahlen, sondern für beliebige euklidische Ringe modulo Elementen, die in disjunkte irreduzible Faktoren zerlegt werden können.*

5. *Der Name "CRT–Algorithmus" ist aus (Gathen & Gerhard, 1999) geborgt und stammt*[53] *daher, daß er den* Chinesischen Restsatz *("Chinese Remainder Theorem") realisiert.*

[53] Algebraiker haben es bestimmt schon erkannt.

Satz 2.44 *(Chinesischer Restsatz)*
Seien $p_1, \ldots, p_n$ paarweise teilerfremde Elemente des euklidischen Rings R und sei $m = p_1 \cdots p_n$. Dann ist

$$R/\langle m\rangle \simeq R/\langle p_1\rangle \times \cdots \times R/\langle p_n\rangle \tag{2.29}$$

und

$$(R/\langle m\rangle)^* \simeq (R/\langle p_1\rangle)^* \times \cdots \times (R/\langle p_n\rangle)^* . \tag{2.30}$$

Beweis: Sei wieder $m_j = m/p_j = \prod_{k\neq j} p_k$ und

$$\ell_j := \left(\left(m_j^{-1}\right)_{p_j} m_j \right)_m , \qquad j = 1, \ldots, n.$$

Da für $q, q' \in R$

$$q \equiv_m q' \qquad \Longleftrightarrow \qquad m \mid (q - q') \qquad \Longleftrightarrow \qquad p_j \mid (q - q') , \quad j = 1, \ldots, n,$$

gilt für die lineare Abbildung

$$L : \left\{ \begin{array}{rcl} R & \to & R/\langle p_1\rangle \times \cdots \times R/\langle p_n\rangle, \\ q & \mapsto & (r_1, \ldots, r_n) = ((q)_{p_1}, \ldots, (q)_{p_n}), \end{array} \right.$$

daß $\ker L = m\, R = \langle m\rangle$. Die Surjektivität von L liefert uns eine enstprechende Varainte von Algorithmus 2.42, die uns zu jedem Satz von Resten r_j, $j = 1, \ldots, n$, ein Element $q \in R$, nämlich $q = \sum_{j=1}^n r_j\, \ell_j$, liefert, so daß $q \equiv_{p_j} r_j$, $j = 1, \ldots, n$. Damit gilt (2.29) und L ist der gewünschte Homomorphismus. □

Übung 2.4 Formulieren Sie den CRT–Algorithmus für beliebige euklidische Ringe.

Bemerkung 2.45 *Der Bezug zwischen Chinesischem Restsatz und Polynominterpolation wird am deutlichsten, wenn wir $R = \mathbb{R}[x]$ und $p_j = p_j(x) = (x - x_j)$, $j = 0, \ldots, n$, für paarweise verschiedene $x_j \in \mathbb{R}$ wählen, denn dann ist offensichtlich*[54]

$$\text{ggT}\ (p_j, p_k) = \delta_{jk}\, (x - x_j) + (1 - \delta_{jk}) , \qquad j, k = 0, \ldots, n,$$

also entweder 1 *oder der gesamte Linearfaktor. Da*

$$m(x) = \prod_{j=0}^{n} (x - x_j) ,$$

ist $\mathbb{R}[x]/\langle m\rangle \simeq \Pi_n$. Auf der anderen Seite ist $\mathbb{R}[x]/\langle p_j\rangle = \mathbb{R}$, $j = 0, \ldots, n$, da wir jedes $f \in \mathbb{R}[x]$ als

$$f(x) = f\,(x_j) + (x - x_j)\, g(x), \qquad g(x) = \frac{f(x) - f\,(x_j)}{x - x_j},$$

[54] Und nach Normalisierung!

darstellen können ("synthetische Division"). Damit ist

$$L(f) = \left(f\left(x_0\right), \ldots, f\left(x_n\right)\right).$$

Außerdem ist

$$m_j(x) = \prod_{k \neq j} (x - x_k) \qquad \Longrightarrow \qquad \left(m_j\right)_{p_j} = m_j\left(x_j\right) = \prod_{k \neq j} (x_j - x_k) \neq 0,$$

was in $\mathbb{R} = \mathbb{R}[x]/\langle x - x_j \rangle$ *invertierbar ist und somit sind die "Rekonstruktionselemente" im Ring* R *gerade*

$$\ell_j(x) = \prod_{k \neq j} \frac{x - x_k}{x_j - x_k} \quad \in \Pi_n = \mathbb{R}[x]/\langle m \rangle.$$

The key of pure mathematics, as supplied chiefly in mediæval and modern time, and mostly by the labours of private philosophers in their own studies, sometimes to absolute truth, sometimes to such close approaches thereto, as to be certain up to the last figure of any fraction yet arrived at ...

Piazzi Smyth, *The Great Pyramid, its secrets and mysteries revealed*, 1880

3 Rationale Arithmetik

In diesem Kapitel wollen wir uns nun mit *rationalen* Arithmetiken beschäftigen; der einfachste Ansatz ist natürlich die "naive" rationale Arithmetik, bei der Brüche $\frac{p}{q} \in \mathbb{Q}$ als Paare $(p, q) \in \mathbb{M}_w \times \mathbb{M}_w$ dargestellt werden, wobei man sich wahlweise beim Zähler oder beim Nenner auch das Vorzeichen schenken könnte (oder es separat in den Bruch codieren könnte). Rationales Rechnen kann erstaunlicherweise auch bei der *effektiven* Behandlung ganzzahliger Probleme eine Rolle spielen, was wir uns im ersten Abschnitt ansehen wollen.

3.1 Die Determinante einee ganzzahligen Matrix

Wir betrachten das scheinbar unscheinbare und unschuldige Problem, die Determinante einer Matrix $A \in \mathbb{Z}^{n\times n}$ zu bestimmen – wer mag, kann auch gleich an $A \in R^{n\times n}$ denken, wobei R ein euklidischer Ring ist – und zwar, und hier liegt die Crux, mit einer Anzahl von Rechenoperationen, die nur *polynomial* von n abhängt. Dabei wollen wir *keine* Struktur der Matrix A voraussetzen, das heißt, sie kann durchaus dicht besetzt und asymmetrisch sein.

Als erstes bemerken wir, daß der *Laplacesche Entwicklungssatz*

$$\det A = (-1)^j \sum_{k=1}^{n} (-1)^k \, a_{jk} \, \det A_{jk}, \qquad j = 1, \ldots, n,$$

mit

$$\det A = a_{11}, \quad A \in \mathbb{Z}^{1\times 1},$$

wobei $A_{jk} \in \mathbb{Z}^{n-1\times n-1}$ ist, die durch Streichung der j–ten Zeile und der k–ten Spalte entsteht, *nicht* weiterhilft, genausowenig wie die *Leibnitz–Formel*, die sogar einen expliziten Ausdruck

für die Determinante gibt. Die Anzahl $f(n)$ der Rechenoperationen für eine $n \times n$–Matrix erfüllt nämlich die Rekursion

$$f(n) = nf(n-1) + n + n - 1 = n\, f(n-1) + 2n - 1, \qquad f(1) = 1,$$

und damit ist $f(n) \geq n!$, $n \in \mathbb{N}$, und von einem polynomialen Zeitaufwand kann also nicht die Rede sein.

Nun gut, wie macht man es also dann? Die Standardmethode in der numerischen Linearen Algebra besteht darin, die Matrix so zu zerlegen, daß man ihre Determinante sehr einfach bestimmen kann. Das "Mittel der Wahl" ist hier meist die *Gauß–Elimination*, die[55] Permutationsmatrizen[56] $P, Q \in \mathbb{Z}^{n\times n}$, sowie eine obere Dreiecksmatrix $U \in \mathbb{Q}^{n\times n}$ und eine untere Dreiecksmatrix $L \in \mathbb{Q}^{n\times n}$ mit Diagonalelementen $\ell_{jj} = 1$, $j = 1, \ldots, n$, bestimmt, so daß

$$PAQ = LU.$$

Da Permutationsmatrizen Determinante ± 1 haben, ist also

$$\det A = \pm \det(PAQ) = \pm \det(LU) = \pm \underbrace{\det L}_{=1} \det U = \pm \prod_{j=1}^{n} u_{jj},$$

und das Vorzeichen läßt sich recht einfach bestimmen, es ist insbesondere nicht nötig, die Determinanten der Permutationsmatrizen auszurechnen. Die gute Nachricht ist nun, daß die Bestimmung von P, Q, L und U lediglich $O(n^3)$ Rechenoperationen benötigt, die schlechte Nachricht ist aber, daß man nun dividieren muß und daß man so *rationale* Zahlen erhält, die mit größerem Aufwand verbunden sind. Zur Erinnerung: bei der Gauß–Elimination erzeugt man Matrizen $A^{(k)}$, $k = 1, \ldots, n$, $A^{(0)} = A$, der Gestalt

$$A^{(k)} = \begin{bmatrix} U^{(k)} & * \\ 0 & B^{(k)} \end{bmatrix}, \qquad U^{(k)} \in \mathbb{Q}^{k\times k}, \quad B^{(k)} \in \mathbb{Q}^{n-k\times n-k},$$

indem man zuerst Permutationsmatrizen $P, Q \in \mathbb{Z}^{n-k\times n-k}$ findet, so daß

$$\begin{bmatrix} I_k & 0 \\ 0 & P \end{bmatrix} A^{(k)} \begin{bmatrix} I_k & 0 \\ 0 & Q \end{bmatrix} = \begin{bmatrix} U^{(k)} & * \\ 0 & C^{(k)} \end{bmatrix} =: \widetilde{A}^{(k)}$$

die Eigenschaft $c_{11}^{(k)} \neq 0$ hat[57] und setzt dann

$$A^{(k+1)} = \begin{bmatrix} I_k & & & \\ & 1 & & \\ & -\left(c_{21}^{(k)}/c_{11}^{(k)}\right) & 1 & \\ & \vdots & & \ddots \\ & -\left(c_{n-k,1}^{(k)}/c_{11}^{(k)}\right) & & 1 \end{bmatrix} \begin{bmatrix} I_k & \\ & P \end{bmatrix} A^{(k)} \begin{bmatrix} I_k & \\ & Q \end{bmatrix},$$

[55] Das ist jetzt die Variante mit Totalpivotsuche. Spaltenpivotsuche würde auch ausreichen, aber aus verschiedenen Gründen, die hoffentlich noch klarer werden, soll hier die Totalpivotsuche gewählt werden.

[56] Eine *Permutationsmatrix* ist eine Matrix, die in jeder Zeile und jeder Spalte genau einmal den Wert 1 und sonst nur den Wert 0 besitzt.

[57] Die genaue Wahl dieser Permutationen hängt von der gewählten *Pivotstrategie* ab.

wobei bei jeder Matrix nur die von Null verschiedenen Einträge dargestellt werden. Und nun kommt die wirklich schlechte Nachricht: seien λ_k und μ_k die maximalen (Wort–) Längen von Zählern und Nennern, die in $A^{(k)}$ auftreten, dann sind

$$\lambda_0 = \max_{j,k=1,\ldots,n} \ell\left(a_{jk}\right), \quad \mu_0 = 0, \qquad \lambda_{k+1} \leq \lambda_k + \mu_k, \quad \mu_{k+1} \leq \lambda_k + \mu_k,$$

also

$$\lambda_1, \mu_1 \leq \lambda_0, \qquad \lambda_2, \mu_2 \leq 2\lambda_0, \qquad \ldots \qquad \lambda_{k+1} \leq 2^k \lambda_0.$$

Natürlich sind das nur obere Abschätzungen und man kann sogar[58] zeigen, daß man so eliminieren kann, daß die Länge der rationalen Zahlen nur polynomial wächst[59], aber trotzdem verdirbt uns diese Abschätzung zuerst mal die Freude an unserer "polynomialen" Gauß–Elimination: zwar ist die *Anzahl* der Rechenoperationen polynomial beschränkt in n, aber der Aufwand der einzelnen Rechenoperationen wächst *exponentiell* in n.

Wie kann man aber nun wieder dieses Problem umgehen? Nun, die Determinante einer ganzzahligen Matrix ist eine *ganze Zahl* und die rationalen Zahlen, denen man "unterwegs" begegnet sind ja eigentlich nur künstlich, denn die Nenner müssen sich ja am Ende wieder wegkürzen. Wir können also

1. eine Primzahl p wählen, die Gaußelimination in $\mathbb{F}_p$ berechnen, und erhalten dann ein Ergebnis $q \in \mathbb{F}_p$, das natürlich die Eigenschaft $q \equiv_p \det A$ hat. Wählen wir die Repräsentanten von $\mathbb{F}_p$ als[60]

   $$\mathbb{F}_p = \left\{ -\frac{p-1}{2}, \ldots, \frac{p-1}{2} \right\},$$

   also die symmetrischen Reste[61], dann erhalten wir das korrekte Ergebnis, wenn $p \geq 2\left|\det A\right| + 1$ ist – wir müssen also "nur" das Ergebnis kennen, das wir glücklicherweise durch die *Hadamard–Formel*

   $$\left|\det A\right| \leq n^{n/2} \max_{jk} \left|a_{jk}\right|^n \tag{3.1}$$

   abschätzen können und dann eine hinreichend große Primzahl parat haben und schon ist alles polynomial in $\ell(p)$. Und da $\ell(p) = \log_{2^w} p = \frac{1}{w} \log_2 p$ ist, liefert uns (3.1), daß

   $$\ell(p) \sim \frac{1}{w} \log_2 \left(n^{n/2} \|A\|_\infty^n \right) = \frac{1}{w} \left(\frac{n}{2} \log_2 n + n \log_2 \|A\|_\infty \right),$$

   und so ist schließlich der Gesamtaufwand polynomial[62] in n beschränkt.

2. mehrere Primzahlen $p_1, \ldots, p_n$ wählen und das Ergebnis parallel in den Körpern $\mathbb{F}_{p_1}, \ldots, \mathbb{F}_{p_n}$ berechnen und dann mit dem chinesischen Restsatz bis auf Vielfache von $m = p_1 \cdots p_n$ rekonstruieren.

[58] Wohl mit geeigneten Pivotstrategien, z.B. indem man das Pivotelement so wählt, daß der Zähler möglichst "kurz" ist.

[59] Was das Beispiel doch etwas abschwächt ...

[60] Das setzt voraus, daß p eine *ungerade* Primzahl ist, eine Situation, die nicht so unwahrscheinlich ist.

[61] Schließlich kann so eine Determinante ja auch negativ werden.

[62] Man erspare es mir, den endgültigen Exponenten auszurechnen.

Bemerkung 3.1 *In vielen Fällen ist man nur an der Frage interessiert, ob eine Matrix $A \in \mathbb{Z}^{n\times n}$ invertierbar ist oder nicht; anstatt das "große" Problem direkt anzugehen, kann man systematisch in $\mathbb{F}_p$ für verschiedene Werte von p rechnen. Kommt auch nur einmal ein Wert $\neq 0$ heraus, ist die Matrix invertierbar, ansonsten erhält man mit einer immer höheren Gewißheit (modulo des Produkts aller bisher getesteten Primzahlen), daß die Matrix singulär ist.*

Manchmal wird das Leben aber auch dadurch leichter, daß man die Matrix richtig zerlegt – das hat nun nicht ganz unbedingt mit rationaler Arithmetik zu tun, ist aber "trotzdem" schöne Mathematik.

Definition 3.2 *Sei R ein kommutativer Ring mit 1. Eine Matrix $A \in R^{n\times n}$ heißt* unimodular, *wenn* $\det A \in R^*$.

Bemerkung 3.3

1. *Eine Matrix $A \in \mathbb{Z}^{n\times n}$ ist unimodular, wenn* $\det A = \pm 1$.

2. *Permutationsmatrizen und Matrizen der Form*

$$A = \begin{bmatrix} a_1 & & & \\ q_2 & a_2 & & \\ \vdots & & \ddots & \\ q_n & & & a_n \end{bmatrix}, \qquad a_1, \ldots, a_n \in R^*,\ q_2, \ldots, q_n \in R,$$

   sind unimodular.

3. *Eine unimodulare Matrix besitzt eine Inverse in $R^{n\times n}$: Nach der Cramerschen Regel ist nämlich*

$$\left(A^{-1}\right)_{jk} = (\det A)^{-1} \det\left[a_1 \cdots a_{k-1}\, e_j\, a_{k+1} \cdots a_n\right],$$

   wobei $a_1, \ldots, a_n \in R^n$ die Spaltenvektoren von A sind. Umgekehrt ist aber auch jede invertierbare Matrix in R unimodular: Da $\det A^{-1} \det A = \det\left(A^{-1}A\right) = \det I = 1$ *ist und mit $A^{-1} \in R^{n\times n}$ auch* $\det A \in R$ *gilt, liegt auch* $(\det A)^{-1} = \det A^{-1}$ *in R und somit* $\det A \in R^*$.

Satz 3.4 *Sei R ein euklidischer Ring und $A \in R^{n\times n}$. Dann gibt es unimodulare Matrizen $P, Q \in R^{n\times n}$ und eine Diagonalmatrix $D \in R^{n\times n}$, so daß*

$$A = PDQ. \tag{3.2}$$

Bemerkung 3.5 *Man kann die Diagonalmatrix D zusätzlich so wählen, daß $d_{jj} | d_{j+1,j+1}$, dann spricht man von der* Smith–Normalform[63] *der Matrix A, siehe z.B. (Marcus & Minc, 1969, Ch. I, 3.22, S. 44).*

[63] Henry John Stephen Smith, 1826–1883, dessen Arbeit in der Zahlentheorie als *"the most complete and elegant monument ever erected to the theory of numbers"* bezeichnet wurde. Dieser Smith hat nichts mit (Charles) Piazzi Smyth aus dem Zitat zu tun! Letzterer, "astronomer-royal" aus Edinburgh ist einer der Entdecker, oder sollte man sagen Erfinder, der in der "Great Pyramid" (Cheops–Pyramide) codierten universellen astronomischen und sonstigen Verhältnisse und Maßeinheiten, die er in (Smyth, 1880) veröffentlichte, siehe auch (Gardner, 1957, S. 176–185).

Beweis von Satz 3.4: Die *algorithmische* Bestimmung der Smith–Zerlegung geht wie folgt:

1. Man beginnt mit der Matrix A und bestimmt $j, k \in \{1, \ldots, n\}$ so, daß

$$d\,(a_{jk}) = \min\left\{d\,(a_{rs}) \;:\; r, s \in \{1, \ldots, n\},\, a_{rs} \neq 0\right\}$$

2. Man wählt die beiden unimodularen Permutationsmatrizen $\widetilde{P}_1, \widetilde{Q}_1$ als Vertauschung der Zeilen 1 und j bzw. 1 und k, dann hat die Matrix $\widetilde{A} = \widetilde{P}_1 A \widetilde{Q}_1$ die Eigenschaft, daß

$$\widetilde{a}_{11} = a_{jk}, \qquad \Longrightarrow \qquad d\,(\widetilde{a}_{11}) \leq \min_{r,s\in\{1,\ldots,n\}} d\,(\widetilde{a}_{rs})\,.$$

3. Man bestimmt Werte p_j und q_j, $j = 2, \ldots, n$, so daß

$$\widetilde{a}_{j1} = p_j \widetilde{a}_{11} + r_j \qquad \text{und} \qquad \widetilde{a}_{1j} = q_j \widetilde{a}_{11} + s_j, \qquad j = 2, \ldots, n,$$

   wobei, weil wir es ja mit einem *euklidischen* Ring zu tun haben,

$$\left.\begin{matrix} d\,(r_j) \\ d\,(s_j) \end{matrix}\right\} < d\,(\widetilde{a}_{11})\,, \qquad j = 2, \ldots, n. \tag{3.3}$$

4. Man setzt

$$A^{(1)} = \underbrace{\begin{bmatrix} 1 & & & \\ -p_2 & 1 & & \\ \vdots & & \ddots & \\ -p_n & & & 1 \end{bmatrix}}_{P_1'} \widetilde{A} \underbrace{\begin{bmatrix} 1 & -q_2 & \ldots & -q_n \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}}_{Q_1'} = \underbrace{P_1' \widetilde{P}_1}_{=:P_1} A \underbrace{\widetilde{Q}_1 Q_1'}_{=:Q_1} = P_1 A Q_1.$$

   Diese Matrix ist unimodular ähnlich zu A.

Dieses Spiel setzt man nun iterativ fort. Da, nach (3.3), die Ungleichung

$$\min_{r,s\in\{1,\ldots,n\}} d\left(a_{rs}^{(j)}\right) =: d\left(A^{(j)}\right) < d\left(A^{(j-1)}\right), \qquad j = 1, 2, \ldots$$

gilt, muß diese Kette nach endlich vielen Schritten abbrechen, das heißt, es gibt einen Index j, so daß $r_j = s_j = 0$ ist und damit

$$A^{(j)} = \left[\begin{array}{c|c} a_{11}^{(j)} & 0 \\ \hline 0 & B^{(j)} \end{array}\right], \qquad B^{(j)} \in R^{n-1\times n-1}.$$

Und dann wenden wir dasselbe Verfahren auf $B^{(j)}$ an ... □

Bemerkung 3.6 *Die Vorzeichen der Matrizen P und Q in (3.2) ergeben sich durch einfaches Zählen, wie oft Zeilen und Spalten vertauscht werden mußten – können also leicht ermittelt werden.*

3.2 Rationales Rechnen mit endlicher Genauigkeit I – die Idee

Nach diesem kleinen Exkurs in die lineare Algebra jetzt aber wieder zurück zum exakten Rechnen mit rationalen Zahlen. Dabei wollen wir, wie im vorherigen Beispiel, das Problem, daß Zähler und Nenner bei "vollständiger" Darstellung einer rationalen Zahl in unermeßliche wachsen können, dadurch umgehen, daß wir modular rechnen. Die folgende Vorgehensweise aus (Gregory & Krishnamurthy, 1984) wollen wir uns in diesem Abschnitt ansehen. Hier zuerst die "große–Primzahl" oder "Single Modulus" Variante, die für eine Primzahl $p \in \mathbb{N}$ wie folgt verfährt:

1. Jeder Bruch $\frac{a}{b}$ wird als
$$f_p\left(\frac{a}{b}\right) = \left(a \cdot b^{-1}\right)_p = \left((a)_p \cdot \left(b^{-1}\right)_p\right) \in \mathbb{F}_p$$
dargestellt.
2. Dann wird alles in $\mathbb{F}_p$ gerechnet.
3. Das Ergebnis wird wieder in einen Bruch zurückkonvertiert.

Natürlich ist die Abbildung $f : \mathbb{Q} \to \mathbb{F}_p$ nicht eineindeutig – schließlich hat $\mathbb{F}_p$ ja nur endlich viele Elemente – und natürlich können Brüche der Form $\frac{a}{kp}$, $k \in \mathbb{Z}\backslash\{0\}$ nicht konvertiert werden. Deswegen ein paar Begriffe.

Definition 3.7 *1. Den Definitionsbereich von $f_p : \mathbb{Q} \to \mathbb{F}_p$ in $\mathbb{Q}$ bezeichnen wir mit*
$$\mathbb{Q}_p := \left\{x = \frac{a}{b} \in \mathbb{Q} \ : \ \text{ggT}\ (a,b) = \text{ggT}\ (b,p) = 1\right\}.$$

2. Für $n > 0$ sind die Farey–Brüche *der Ordnung n definiert als*
$$\mathbb{Q} \supset F_n := \left\{x = \frac{a}{b} \in \mathbb{Q} \ : \ \text{ggT}\,(a,b) = 1,\ 0 \le |a| \le n,\ 0 < |b| \le n\right\}.$$

Beispiel 3.8 *Hier ein paar Sätze von Farey–Brüchen:*

$$\begin{aligned}
F_1 &= \{0\} \cup \pm\{1\} \\
F_2 &= \{0\} \cup \pm\left\{\frac{1}{2}, 1, 2\right\} \\
F_3 &= \{0\} \cup \pm\left\{\frac{1}{3}, \frac{1}{2}, \frac{2}{3}, 1, \frac{3}{2}, 2, 3\right\} \\
F_4 &= \{0\} \cup \pm\left\{\frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}, 1, \frac{4}{3}, \frac{3}{2}, 2, 3, 4\right\} \\
F_5 &= \{0\} \cup \pm\left\{\frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, 1, \frac{5}{4}, \frac{4}{3}, \frac{3}{2}, \frac{5}{3}, 2, \frac{5}{2}, 3, 4, 5\right\}
\end{aligned}$$

Die Verteilung der Farey–Brüche ist für zwei Beispiele in Abb. 3.1 dargestellt.

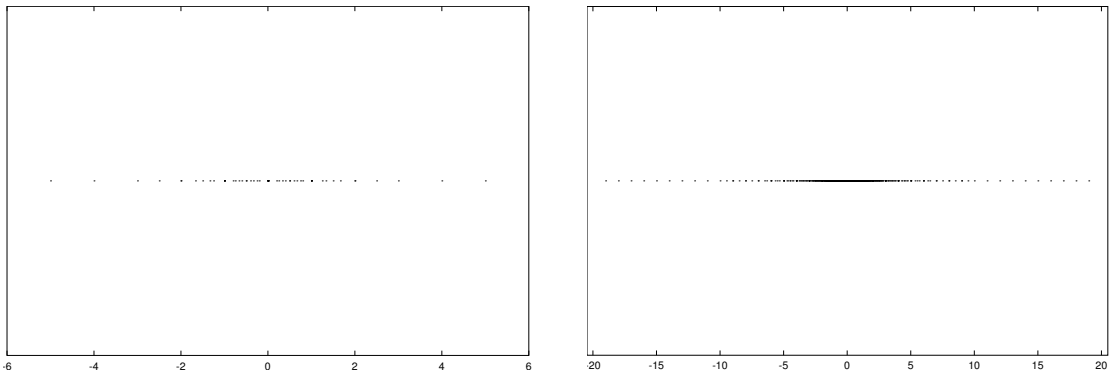


Abbildung 3.1: Verteilung der Farey–Brüche F_5 und F_{20}. Ja, das sollen Punkte sein!

Der Schlüssel zum Erfolg mit unserer rationalen Arithmetik besteht nun in der Beobachtung, daß die Farey–Brüche *injektiv* (und damit umkehrbar) in $\mathbb{F}_p$ eingebettet werden können, solange nur n klein genug ist im Vergleich zu p.

Satz 3.9 *Seien $n \in \mathbb{N}$ und eine Primzahl $p \in \mathbb{N}$ so gewählt, daß*

$$2n^2 < p, \tag{3.4}$$

dann ist $F_n \subset \mathbb{Q}_p$ und $f_p|_{F_n}$ ist injektiv.

Für den Beweis benötigen wir die folgende Beobachtung.

Lemma 3.10 *Seien $x, x' \in \mathbb{Q}_p$, $x = \frac{a}{b}$, $x' = \frac{a'}{b'}$. Dann ist*

$$f_p(x) = f_p(x') \qquad \Longleftrightarrow \qquad ab' \equiv_p a'b.$$

Beweis:

$$f_p(x) = f_p(x') \qquad \Longleftrightarrow \qquad ab^{-1} \equiv_p a'(b')^{-1} \qquad \Longleftrightarrow \qquad ab' \equiv_p a'b.$$

□

Beweis von Satz 3.9: Seien $x, x' \in F_n$ in gekürzten Darstellungen $x = \frac{a}{b}$, $x' = \frac{a'}{b'}$. Dann ist

$$|ab' - a'b| \leq |ab'| + |a'b| = |a| \cdot |b'| + |a'| \cdot |b| \leq 2n^2.$$

Wäre nun $f_p(x) = f_p(x')$, dann ist nach Lemma 3.10 $ab' \equiv_p a'b$, also $ab' - a'b = kp$ für ein $k \in \mathbb{Z}$ und da $p > 2n^2$ ist, muß $k = 0$ sein, also $x = x'$. □

Satz 3.9 liefert nun natürlich unsere "Strategie" zum rationalen Rechnen mit endlicher[64] Genauigkeit, nämlich in der Menge der Farey–Brüche der Ordnung n. Man gibt sich also ein $n > 0$ vor, wählt

[64] Schließlich sind wir ja "nur" in $\mathbb{F}_p$.

eine hinreichend große Primzahl $p > 2n^2$ und rechnet in $\mathbb{F}_p$ – das können wir ja inzwischen. Ist nun das Ergebnis wieder ein Farey–Bruch, dann können wir den auch wieder rekonstruieren, wenn nicht, dann haben wir, wie immer wenn man mit *endlicher* Genauigkeit rechnet, Pech gehabt ...

Bemerkung 3.11 *Beim Rechnen mit endlicher Genauigkeit, also mit Zahlenbereichen, die nur endlich viele Elemente umfassen und deren Größe* a priori *feststeht, muß man tätig werden, sobald das Ergebnis einer Rechenoperation den darstellbaren Bereich verläßt. Bei der Fließpunktrechnung wird das durch* Rundung *erledigt, die eine benachbarte darstellbare Zahl mit möglichst geringer relativer Abweichung verwendet – diese Fehler sind aber normalerweise irreversibel und selbst wenn das exakte Endergebnis exakt darstellbar wäre, wird es im Normalfall mit (kleinen[65]) Fehlern behaftet sein. Dies ist anders bei den Farey–Brüchen: Ist das Endergebnis darstellbar, so ist es auch exakt, ganz egal wie oft bei Zwischenrechnungen der darstellbare Bereich verlassen wurde, ist das Endergebnis hingegen* nicht *darstellbar, dann ist es auch total falsch, denn die Verteilung der Farey–Brüche ist nicht metrisch.*

3.3 Rationales Rechnen mit endlicher Genauigkeit II – die Algorithmen

Was wir jetzt also noch brauchen, um "unsere" Arithmetik realisieren zu können, sind die algorithmischen Realisierungen von f_p und vor allem von f_p^{-1}, das heißt:

1. wie finden wir zu einem Farey–Bruch aus F_n die zugehörige Darstellung in $\mathbb{F}_p$?
2. wie finden wir zu einer Zahl $q \in \mathbb{F}_p$ den zugehörigen Farey–Bruch?

Für das erste Problem haben wir bereits eine Lösung parat, nämlich den erweiterten euklidischen Algorithmus, mit dem wir die Zahlen u und v berechnen können, so daß für ein $b \in \mathbb{Z} \setminus p\mathbb{Z}$

$$1 = \text{ggT}\ (b,p) = ub + vp \equiv_p ub \qquad \Longrightarrow \qquad u \equiv_p b^{-1} \qquad \Longrightarrow \qquad \frac{a}{b} \equiv_p au \equiv_p (au)_p\,.$$

Allerdings kann man sich das Leben ein bißchen leichter machen, wenn man den erweiterten euklidischen Algorithmus wie in (Gregory & Krishnamurthy, 1984) etwas anders hinschreibt, nämlich in Matrixform.

Algorithmus 3.12 *(Euklidischer Algorithmus, Version 3)*
Gegeben: *Matrix* $A = \begin{bmatrix} a_1^T \\ a_2^T \end{bmatrix} \in R^{2\times n}$, *R euklidischer Ring mit* 1.

1. *Setze*
$$s \leftarrow a_{11}/a_{21}, \qquad y \leftarrow a_1 - sa_2.$$
2. *Solange* $y_1 \neq 0$

[65] Zumindest hofft man das.

(a) Setze

$$a_1 \leftarrow a_2, \qquad a_2 \leftarrow y.$$

(b) Setze

$$s \leftarrow a_{11}/a_{21}, \qquad y \leftarrow a_1 - s a_2.$$

3. Setze $y \leftarrow a_2$.

Ergebnis: *Vektor* y.

Bemerkung 3.13 *1. Wie man sieht, enthält* y_1 *immer den Divisionsrest* $(a_{11})_{a_{21}}$, *so daß also am Ende der Wert* $\mathrm{ggT}\,(a_{11}, a_{21})$ *in* a_{21} *zu finden sein wird.*

2. Startet man Algorithmus 3.12 mit der Matrix

$$A = \begin{bmatrix} a & 1 & 0 \\ b & 0 & 1 \end{bmatrix},$$

dann erhalten wir die nicht–normalisierende Version von Algorithmus 2.26 und somit erfüllt der Ergebnisvektor y *die Bedingung*

$$\mathrm{ggT}\,(a, b) = y_1 = a\, y_2 + b\, y_3.$$

3. Für unsere Zwecke, das heißt für die Berechnung von $(b^{-1})_p$, *brauchen wir ja* y_2 *nicht und können so eine Spalte weglassen und mit der Matrix*

$$A = \begin{bmatrix} p & 0 \\ b & 1 \end{bmatrix} \qquad \Longrightarrow \qquad y = \left(1 = \mathrm{ggT}\,(p, b), \left(b^{-1}\right)_p\right)$$

beginnen.

Aus Bemerkung 3.13, 3., erhalten wir sofort das *Zweischritt–Verfahren*, um $\left(\frac{a}{b}\right)_m$ zu berechnen: Zuerst bestimmt man $c := (b^{-1})_p$ und dann $(a \cdot c)_p$. Es geht aber noch einfacher. Dazu bezeichnen wir mit

$$y^{(j+1)} = y^{(j-1)} - s_j y^{(j)}, \qquad s_j = y_1^{(j-1)}/y_1^{(j)}, \qquad j = 1, 2, \ldots,$$

initialisiert mit

$$y^{(0)} = \begin{bmatrix} p \\ 0 \end{bmatrix}, \qquad y^{(1)} = \begin{bmatrix} b \\ 1 \end{bmatrix},$$

die Zwischenergebnisse des erweiterten euklidischen Algorithmus, wobei

$$y^{(n)} = \begin{bmatrix} \mathrm{ggT}\,(p, b) \\ (b^{-1})_p \end{bmatrix} = \begin{bmatrix} 1 \\ (b^{-1})_p \end{bmatrix}.$$

Da $y_2^{(n)} \equiv_p b^{-1}$ haben wir somit, daß

$$\begin{aligned} \frac{a}{b} \equiv_p a\, y_2^{(n)} &= a\left(y_2^{(n-2)} - s_{n-1} y_2^{(n-1)}\right) = \left(a \cdot y_2^{(n-2)}\right) - s_{n-1}\left(a \cdot y_2^{(n-1)}\right) \\ &=: \widetilde{y}_2^{(n-2)} - s_{n-1}\widetilde{y}_2^{(n-1)}. \end{aligned}$$

Setzen wir nun $\widetilde{y}^{(1)} = [b, a]^T$, anstelle von $[b, 1]^T$, dann liefert die Iteration des erweiterten euklidischen Algorithmus also gerade die Werte $\widetilde{y}^{(j)} = \left[y_1^{(j)}, a \cdot y_2^{(j)}\right]$, also insbesondere $\left(\frac{a}{b}\right)_p$ als Endergebenis. Das ist die *"Einschritt–Methode"*, die wir wie folgt zusammenfassen können.

Satz 3.14 *Startet man Algorithmus 3.12 mit der Matrix*

$$A = \begin{bmatrix} p & 0 \\ b & a \end{bmatrix},$$

dann erhält man als Ergebnis den Vektor

$$y = \begin{bmatrix} 1 \\ \left(\frac{a}{b}\right)_p \end{bmatrix}.$$

Beispiel 3.15 *Sehen wir uns doch mal an, wie so ein paar Farey–Brüche modulo* 37 *aussehen. Die Voraussetzung* $2n^2 < p$ *läßt also* $n = 1, 2, 3, 4$ *zu.*

$n = 1$**:**

-1	0	1
36	0	1

$n = 2$**:**

-2	-1	$-\frac{1}{2}$	0	$\frac{1}{2}$	1	2
35	36	18	0	19	1	2

$n = 3$**:**

-3	-2	$-\frac{3}{2}$	-1	$-\frac{2}{3}$	$-\frac{1}{2}$	$-\frac{1}{3}$	0	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{2}{3}$	1	$\frac{3}{2}$	2	3
34	35	17	36	24	18	12	0	25	19	13	1	20	2	3

$n = 4$**:** *Hier wird die Tabelle etwas länglich, sehen wir uns dafür lieber mal die Berechnung von* $\left(\frac{4}{3}\right)_{37}$ *an:*

37	0
3	4
1	-48

$$\Longrightarrow \qquad \left(\frac{4}{3}\right)_{37} = 26.$$

Also, wie gesagt: die "Codierung" eines Farey–Bruchs in $\mathbb{F}_p$ ist der einfache Teil, aber was ist mit der Umkehrung. Nun, nachdem "unser" Hilfsmittel der erweiterte euklidische Algorithmus ist, probieren wir ihn doch einfach mal mit 26:

37	0
26	1
11	-1
4	**3**
3	-7
1	10

(3.5)

Es ist kein Zufall, daß wir unseren Bruch $\frac{4}{3}$ in dieser Liste finden, und es ist auch kein Zufall, daß es der einzige *Farey–Bruch* in dieser Liste ist.

Satz 3.16 *Sei $2n^2 < p$. Startet man für $q \in \mathbb{F}_p \cap f_p(F_n)$ den erweiterten euklidischen Algorithmus mit der Matrix*

$$A = \begin{bmatrix} p & 0 \\ q & 1 \end{bmatrix},$$

dann gibt es ein $j \geq 1$ so daß $y^{(j)} = [a, b]^T$, wobei $\frac{a}{b} = f_p^{-1}(q)$ und $\frac{a}{b}$ ist der einzige Fareybruch der Ordnung n in dieser Liste.

Der Beweis dieses Satzes wird etwas aufwendiger, dafür werden wir aber auch ein bißchen mehr über den[66] (erweiterten) euklidischen Algorithmus lernen. Vorher aber noch ein paar Bemerkungen.

Bemerkung 3.17

1. *Dieses Resultat ermöglicht es uns auch, zu erkennen, ob überhaupt $q \in \mathbb{F}_p \cap f_p(F_n)$: Da irgendwann ja der euklidische Algorithmus terminiert, die zu durchsuchende Liste also endlich ist, finden wir entweder einen Fareybruch der Orndnung n (das heißt $q \in \mathbb{F}_p \cap f_p(F_n)$) oder eben nicht (und das heißt $q \in \mathbb{F}_p \setminus f_p(F_n)$.*

2. *Die obige Aussage gilt für* jedes *$n < \sqrt{p/2}$ – es bringt also gar nichts, nur Fareybrüche zu verwenden, deren Ordnung echt kleiner als der maximal mögliche Wert von n ist. Anders gesagt, die Qualität dieser Arithmetik hängt eigentlich nur von der verwendeten Primzahl p ab.*

Jetzt aber an die Arbeit – tragen wir mal langsam die Zutaten für den Beweis zusammen.

Lemma 3.18 *Seien*

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad \text{und} \qquad B = \begin{bmatrix} c & d \\ e & f \end{bmatrix},$$

wobei $e = (a)_c$ und $f = b - (a/c) \cdot d$. Ist, für $m > 0$, $\det A \equiv_m 0$, dann ist auch $\det B \equiv_m 0$.

Beweis: Es ist

$$\begin{aligned} \det B &= cf - de = c(b - (a/c)\, d) - d\,(a)_c = bc - \underbrace{(c \cdot (a/c))}_{=a-(a)_c} d - d\,(a)_c \\ &= bc - ad + d\,(a)_c - d\,(a)_c = bc - ad = -\det A \equiv_m 0. \end{aligned}$$

□

Proposition 3.19 *Die Vektoren $y^{(j)}$, $j \geq 1$, die bei Algorithmus 3.12 mit der Startmatrix*

$$A = \begin{bmatrix} p & 0 \\ q & 1 \end{bmatrix}$$

erzeugt werden, erfüllen

$$\frac{y_1^{(j)}}{y_2^{(j)}} \equiv_p q. \tag{3.6}$$

[66] Gar nicht mal so trivialen.

Beweis: Da[67] $\det \left[y^{(0)}\, y^{(1)} \right] = \det A = p \equiv_p 0$ ist nach Lemma 3.18 auch für $j = 1, 2, \ldots$

$$\det \left[y^{(j)}\, y^{(j+1)} \right] \equiv_p \det \left[y^{(j-1)}\, y^{(j)} \right] \equiv_p \cdots \equiv_p \det \left[y^{(0)}\, y^{(1)} \right] \equiv_p \det A \equiv_p 0,$$

also $y_1^{(j+1)} y_2^{(j)} \equiv_p y_2^{(j+1)} y_1^{(j)}$, oder eben

$$\frac{y_1^{(j+1)}}{y_2^{(j+1)}} \equiv_p \frac{y_1^{(j)}}{y_2^{(j)}} \equiv_p \cdots \equiv_p \frac{y_1^{(1)}}{y_2^{(1)}} = \frac{q}{1} = q.$$

□

Was wir also dank Proposition 3.19 schon mal festhalten können, ist die Tatsache, daß *wenn* wir einen Farey–Bruch in der Liste der Zwischenergebnisse von Algorithmus 3.12 finden, er auch $f_p^{-1}(q)$ sein muß – mehr als ein Farey–Bruch kann ja nach Satz 3.9 nicht in $f_p^{-1}(q)$ liegen. Was wir also noch zeigen müssen, ist daß wir bei dem Prozeß den Farey–Bruch auch wirklich finden können.

Zu diesem Zweck müssen wir uns einer klassischen und heute leider nicht mehr so gebräuchlichen Theorie, nämlich der der *Kettenbrüche* bedienen, die sich durch fast die gesamte Geschichte der Mathematik zieht: Laut (Knuth, 1998)[68] basierte die früheste Vorstellung der alten Griechen von reellen Zahlen auf Kettenbrüchen, aber auch die Verwendung von *orthogonalen Polynomen* in der *Quadraturtheorie*[69] wurde von Gauß in seiner Originalarbeit (Gauss, 1816) vollwertig durch Kettenbrüche, zu deren Verständnis er auch einiges beigetragen hat, ersetzt[70]. Das Standardwerk über Kettenbrüche ist aber wohl (Perron, 1954), hier orientieren wir uns aber an (Hardy & Wright, 1954, Kapitel X). Daneben gibt es noch das sehr schöne "Büchlein" (Khinchin, 1964) und (inzwischen) auch das Vorlesungsskript (Sauer, 2005). Wundert es da im Kontext dieser Vorlesung wirklich noch jemanden, daß Kettenbrüche im wesentlichen über beliebigen euklidischen Ringen definiert werden können und fast auch sollten?

Definition 3.20 *Für ganze Zahlen $a_0, \ldots, a_n$ ist der(einfache)* Kettenbruch*, geschrieben als $[a_0, a_1, \ldots, a_n]$ (rekursiv) definiert als*

$$[a_0, a_1, \ldots, a_n] = a_0 + \frac{1}{[a_1, \ldots, a_n]} = \cdots = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots\ a_{n-1} + \cfrac{1}{a_n}}}} \tag{3.7}$$

Satz 3.21 *Jede rationale Zahl $x = \frac{a}{b}$ kann durch einen endlichen Kettenbruch dargestellt werden.*

[67] Nicht vergessen: $\det A^T = \det A$!

[68] Er verweist seinerseits auf (Becker, 1933).

[69] Das ist numerische Intgeration.

[70] Dieser Ansatz ist in (Sauer, 2000b) ausgearbeitet und das ist keine Eigenwerbung – ich weiß sonst keine Literatur, in der man "Quadratur nach Art des Gaußes" in "heutiger" Terminologie und Notation finden kann.

Beweis: Wir nehmen an, daß a, b bereits gekürzt sind, das heißt, daß $\mathrm{ggT}\,(a,b) = 1$ ist, ansonsten kann man ja (mit Euklid) den größten gemeinsamen Teiler ermitteln und dann kürzen. Dann betrachten wir den "normalen" euklidischen Algorithmus mit $r_0 = a$, $r_1 = b$, und der Darstellung $r_{j+1} = r_{j-1} - s_j r_j$, $s_j = r_{j-1}/r_j$, und erhalten

$$\frac{a}{b} = \frac{r_0}{r_1} = \frac{s_1\, r_1 + r_2}{r_1} = s_1 + \frac{r_2}{r_1} = s_1 + \frac{1}{\frac{r_1}{r_2}} \tag{3.8}$$

Nun führen wir Induktion über die Anzahl n der Schritte im euklidischen Algorithmus durch. Ist $n = 1$, dann ist $r_2 = 0$ und (3.8) liefert uns, daß $\frac{a}{b} = a = [s_1]$, für $n > 1$ hingegen liefern uns (3.8), die Induktionsannahme und (3.7), daß

$$\frac{a}{b} = s_1 + \frac{1}{\frac{r_1}{r_2}} = s_1 + \frac{1}{[s_2, \ldots, s_n]} = [s_1, \ldots, s_n]\,.$$

□

Satz 3.21, bzw. sein Beweis, hat die offensichtliche Konsequenz, daß wir sogar wissen, *wie* man die Kettenbruchdarstellung eines Bruches, also einer rationalen Zahl, *algorithmisch* bestimmen kann. Und es sollte uns schön langsam nicht mehr wundern, daß hier wieder einmal der euklidische Algorithmus ins Spiel kommt.

Definition 3.22 *Sei $x \in \mathbb{Q}$ und sei $x = [a_0, \ldots, a_n]$ die*[71] *zugehörige Kettenbruchentwicklung. Dann bezeichnet man*

$$K_k(x) := [a_0, \ldots, a_k]\,, \qquad k \in \mathbb{N}, \qquad a_{n+1} = a_{n+2} = \cdots = 0,$$

*als die k–*te Konvergente *von x.*

Ist nun $x = [a_0, \ldots, a_n]$ in einen Kettenbruch entwickelt, dann können wir ja mal ein paar Konvergenten berechnen:

$$\begin{aligned} K_0(x) &= a_0, \\ K_1(x) &= a_0 + \frac{1}{a_1} = \frac{a_0 a_1 + 1}{a_1} \\ K_2(x) &= \left[a_0, a_1 + \frac{1}{a_2}\right] = \frac{a_0\left(a_1 + \frac{1}{a_2}\right) + 1}{a_1 + \frac{1}{a_2}} = \frac{a_0 a_1 a_2 + a_0 + a_2}{a_1 a_2 + 1}. \end{aligned}$$

Lemma 3.23 *Die Zahlen p_k, q_k, $k = 0, \ldots, n$, definiert durch die Rekursionsformel*

$$\begin{array}{rclcrclcrcl} p_k &=& p_{k-2} - a_k p_{k-1}, & & p_{-1} &=& 1, & & p_{-2} &=& 0 \\ q_k &=& q_{k-2} - a_k q_{k-1}, & & q_{-1} &=& 0, & & q_{-2} &=& -1. \end{array} \tag{3.9}$$

liefern die k–te Konvergente von $x = [a_0, \ldots, a_n]$, also $K_k(x) = \frac{p_k}{q_k}$.

[71] Es gibt immer mehrere Kettenbruchdarstellungen einer rationalen Zahl, insbesondere ist $[a_0, \ldots, a_n] = [a_0, \ldots, a_n - 1, 1]$, siehe (Hardy & Wright, 1954, Theorem 158). Aber wir können hier immer die Kettenbruchentwicklung, die uns der euklidische Algorithmus wie im Beweis von Satz 3.21 liefert, als *die* Kettenbruchentwicklung definieren.

Beweis: Induktion über k, der Fall $k = 0$ ergibt sich mit (3.9) als

$$p_0 = 0 - a_0 = -a_0, \qquad q_0 = -1 - 0 \times a_0 = -1, \qquad \Longrightarrow \qquad \frac{p}{q} = a_0 = K_0(x).$$

Ist außerdem (3.23) für ein $k \geq 0$ bewiesen, dann ergibt sich, falls $a_{k+1} \neq 0$,

$$\begin{aligned} K_{k+1}(x) &= [a_0, \ldots, a_{k+1}] = \left[a_0, \ldots, a_k + \frac{1}{a_{k+1}}\right] = \frac{-\left(a_k + \frac{1}{a_{k+1}}\right) p_{k-1} + p_{k-2}}{-\left(a_k + \frac{1}{a_{k+1}}\right) q_{k-1} + q_{k-2}} \\ &= \frac{-(a_k a_{k+1} + 1)\, p_{k-1} + a_{k+1}\, p_{k-2}}{-(a_k a_{k+1} + 1)\, q_{k-1} + a_{k+1}\, q_{k-2}} = \frac{a_{k+1} \overbrace{(-a_k p_{k-1} + p_{k-2})}^{=p_k} - p_{k-1}}{a_{k+1} \underbrace{(-a_k q_{k-1} + q_{k-2})}_{=q_k} - q_{k-1}} \\ &= \frac{a_{k+1} p_k - p_{k-1}}{a_{k+1} q_k - q_{k-1}} = \frac{p_{k-1} - a_{k+1} p_k}{q_{k-1} - a_{k+1} q_k} = \frac{p_{k+1}}{q_{k+1}} \end{aligned}$$

□

Nun sollte uns aber die Rekursionsformel in (3.9) bekannt vorkommen, es ist nämlich genau dieselbe wie in der Matrixversion des erweiterten euklidischen Algorithmus. Initialisieren wir nun noch richtig, dann haben wir auch schon eine Möglichkeit gefunden, Konvergenten einer rationalen Zahl zu berechnen.

Korollar 3.24 *Sei $x = \frac{a}{b} \in \mathbb{Q}$. Startet man Algorithmus 3.9 mit der Matrix*

$$A = \begin{bmatrix} a & 0 & -1 \\ b & 1 & 0 \end{bmatrix},$$

dann ist

$$K_k(x) = \frac{y_2^{(k+2)}}{y_3^{(k+2)}}, \qquad k \in \mathbb{N}_0. \tag{3.10}$$

Beispiel 3.25 *Probieren wir es doch einmal mit $x = \frac{37}{26}$. Das Schema, das wir nun erhalten ist*

37	0	−1	
26	1	0	
11	−1	−1	$\to 1$
4	3	2	$\to \frac{3}{2}$
3	−7	−5	$\to \frac{7}{5}$
1	10	7	$\to \frac{10}{7}$
0	37	26	$\to \frac{37}{26}$

Man sollte nicht unterschlagen, daß die Konvergenten ziemlich flott konvergieren und so ihrem Namen alle Ehre machen, denn die Fehler sind

$$\frac{9}{26} \sim 0.34615, \qquad -\frac{1}{13} \sim -0.076923, \qquad \frac{3}{130} \sim 0.023077, \qquad -\frac{1}{182} \sim -0.0054945$$

Man kann sogar zeigen, daß die Konvergenten sehr schnell konvergieren und daß alles, was eine bestimmte Konvergenzgeschwindigkeit hat, auch bereits eine Konvergente sein muß. Bewiesen werden diese Tatsachen beispielsweise in (Hardy & Wright, 1954, Kapitel X).

Satz 3.26 *Sei $x \in \mathbb{Q}$.*

1. *Für jede Konvergente $\frac{p}{q} = K_k(x)$ gilt*

$$\left| x - \frac{p}{q} \right| < \frac{1}{q^2}. \tag{3.11}$$

2. *Unter je zwei aufeinananderfolgenden Konvergenten $K_k(x)$ und $K_{k+1}(x)$ befindet sich eine, bezeichnet als $\frac{p}{q}$, so daß sogar*

$$\left| x - \frac{p}{q} \right| < \frac{1}{2q^2} \tag{3.12}$$

   gilt.

3. *Jeder Bruch $\frac{p}{q}$, der (3.12) erfüllt, muß eine Konvergente sein, das heißt, es gibt ein $k \in \mathbb{N}_0$, so daß $\frac{p}{q} = K_k(x)$.*

So interessant es wäre, diesen Satz zu beweisen[72], es wird langsam Zeit, uns an unser Ziel zu erinnern, nämlich an den Beweis von Satz 3.16, genauer, an den Nachweis der Tatsache, daß der erweiterte euklidische Algorithmus auch tatsächlich unseren Fareybruch findet.

Beweis von Satz 3.16: Zuerst erinnern wir uns an unsere "alte" Notation: Es war p die Primzahl und $q \in \mathbb{Z}_p$ die Codierung des Fareybruchs $\frac{a}{b}$. Da $\frac{a}{b} \equiv_p q$, also $a \equiv_p bq$ gibt es eine (eindeutige) Zahl $c \in \mathbb{Z}$, so daß

$$a = b\,q - c\,p. \tag{3.13}$$

Da $\frac{a}{b} \in F_n$ und $2n^2 < p$, erhalten wir nun, daß

$$\left| \frac{q}{p} - \frac{c}{b} \right| = \left| \frac{bq - cp}{bp} \right| = \frac{|a|}{|bp|} \leq \frac{1}{|b|} \frac{n}{2n^2+1} \leq \frac{1}{b^2} \frac{|b|n}{2n^2+1} \leq \frac{1}{b^2} \cdot \underbrace{\frac{n^2}{2n^2+1}}_{<\frac{1}{2}} < \frac{1}{2b^2},$$

also ist, nach Satz 3.26, $\frac{c}{b}$ eine Konvergente von $\frac{q}{p}$. Nun ist $\frac{q}{p}$ zwar nicht ganz das, was wir wollen, aber da die ersten drei Zeilen eines erweiterten Matrix–Euklid nach Art von Korollar 3.24 für die Berechnung der Konvergenten von $\frac{q}{p}$ für $q \in \mathbb{Z}_p$ die Form

q	0	-1
p	1	0
q	0	-1

[72] Der Beweis ist noch nicht einmal so lang und eigentlich recht elementar.

haben, müssen die Konvergenten von $\frac{p}{q}$ und die von $\frac{q}{p}$ eng verwandt sein. In der Tat, wenn wir die zweite und die dritte Spalte vertauschen, was dem Reziprokwert der Konvergenten entspricht, und beide Vorzeichen umdrehen, was überhaupt nichts verändert, dann erhalten wir, daß die Konvergenten von $\frac{p}{q}$ gerade die Reziprowerte der Konvergenten von $\frac{q}{p}$ sind. Also, und darauf wollten wir hinaus, ist $\frac{b}{c}$ eine Konvergente von $\frac{p}{q}$ und deshalb gibt es einen Index $j \geq 2$ im Matrix–Euklid mit der Startmatrix

$$A = \begin{bmatrix} p & 0 & -1 \\ q & 1 & 0 \end{bmatrix}, \qquad \text{so daß} \qquad \frac{b}{c} = K_{j-2}\left(\frac{p}{q}\right) = \frac{y_2^{(j)}}{y_3^{(j)}}.$$

Damit haben wir aber auch unseren Fareybruch erwischt, denn jetzt erhalten wir mit der "Euklid–Invarianz"

$$y_1^{(j)} = q\, y_2^{(j)} - p\, y_3^{(j)},$$

siehe Gleichung (2.23) in Satz 2.28, 3, und mit (3.13), daß

$$\frac{a}{b} = q - \frac{c}{b}p = q - \frac{y_3^{(j)}}{y_2^{(j)}}\, p = q - \frac{p\, y_3^{(j)}}{y_2^{(j)}} = q - \frac{q\, y_2^{(j)} - y_1^{(j)}}{y_2^{(j)}} = \frac{y_1^{(j)}}{y_2^{(j)}}.$$

Damit ist gezeigt, daß wir unseren Fareybruch finden. Da nach Proposition 3.19 alle Brüche in der vom euklidischen Algorithmus erzeugten Liste kongruent zu q modulo p sind, ist $\frac{a}{b}$ nach Satz 3.9 auch der einzige Fareybruch der Ordnung n in dieser Liste. □

Zum Abschluß, und um zu sehen, daß das Ganze auch wirklich funktioniert, sehen wir uns schließlich noch ein Beispiel an.

Beispiel 3.27 *Wir wollen uns einmal F_5 ansehen und wählen dafür die kleinste Primzahl größer als $2 \times 5^2 = 50$, also $p = 53$. Die durchzuführende Rechnung wird einfach*

$$-\frac{4}{3} \oslash \left(\frac{1}{5} \oplus \frac{1}{3}\right)$$

sein. Die Codierung der Zahlen ergibt sich also als

53	0
3	−4
2	68
1	−72
	→ 34

53	0
5	1
3	−10
2	11
1	−21
	→ 32

53	0
3	1
2	−17
1	18
	→ 18

und die Rechnung in $\mathbb{F}_{53}$ ist daher

$$\left(34 \times \left((32+18)^{-1}\right)_p\right)_p.$$

Nachdem wir $32 + 18 = 50$ *noch im Kopf können, brauchen wir also noch das multiplikative Inverse von* 50 *in* $\mathbb{F}_{53}$*, das uns der erweiterte Euklid mit dem Tableau*

53	0
50	1
3	-1
2	17
1	-18
	$\to 35$

liefert. Also ist unser Endergebnis $(34 \times 35)_{53} = (1190)_{53} = 24$*, was wir nun wieder in* F_5 *rücktransformieren wollen:*

53	0
24	1
5	$\mathbf{-2}$
4	9
1	-11

und das Ergebnis ist in der Tat $\frac{5}{-2} = -\frac{5}{2}$*. Übrigens, das Zwischenergebnis* $\frac{1}{5} + \frac{1}{3} = \frac{8}{15} \equiv_{53} 35$ *ist natürlich* nicht *in* F_5*, was sich auch im "gescheiterten" Invertierungsversuch*

53	0
35	1
18	-1
17	2
1	-3

zeigt. Aber Moment mal: Hier wird doch ein Fareybruch gefunden, nämlich der Bruch $-\frac{1}{3}$*! Das ist schon richtig, und der ist auch, entsprechend der Theorie, der einzige Fareybruch in der Liste, aber trotzdem halt das falsche Ergebnis, das aber die Eigenschaft hat, daß* $-\frac{1}{3} \equiv_{53} \frac{15}{8}$*. Wir können also im allgemeinen* nicht *entscheiden, ob der Rechenbereich verlassen wurde oder nicht.*

3.4 Rationales Rechnen mit endlicher Genauigkeit III – mehrere Moduli

Lassen wir die Methoden und Beweise aus den Abschnitten 3.2 und 3.3 noch einmal Revue passieren, dann stellen wir fest, daß die gesamten Algorithmen zur Berechnung von f_p und f_p^{-1} auch dann funktionieren, wenn wir die Primzahl p durch eine Zahl m ersetzen. Nur kann es natürlich die obligatorischen Probleme bei der Berechnung des multiplikativen Inversen modulo m geben.

Genauer, seien $p_1, \ldots, p_\ell$ die Primfaktoren von m, also

$$m = p_1^{e_1} \cdots p_\ell^{e_\ell}, \qquad e_j \in \mathbb{N}, \quad j = 1, \ldots, \ell,$$

dann gilt $a \in \mathbb{Z}_m^*$ genau dann, wenn ggT $(a, p_j) = 1$, $j = 1, \ldots, \ell$. Nachdem wir mit höheren Potenzen der Primfaktoren ohnehin nichts erreichen können, wollen wir annehmen, daß $e_1 = \cdots = e_\ell = 1$. Dann ist natürlich

$$\mathbb{Q}_m = \left\{ x = \frac{a}{b} \ : \ \text{ggT}\ (a,b) = \text{ggT}\ (b, p_j) = 1,\ j = 1, \ldots, \ell \right\}$$

und wir haben, daß

$$x, y \in \mathbb{Q}_m \qquad \Longrightarrow \qquad \{x + y, x - y, x \cdot y\} \in \mathbb{Q}_m,$$

nur bei der Division können wir in Schwierigkeiten kommen, da

$$x = \frac{a}{b} \in \mathbb{Q}_m^{-1} \qquad \Longleftrightarrow \qquad \text{ggT}\ (a, p_j) = 1, \quad j = 1, \ldots, \ell.$$

Bemerkung 3.28 *Sei $m = p_1 \cdots p_\ell$. Die Rechnung mit Fareybrüchen ist mit Sicherheit unproblematisch, wenn wir die Beziehungen*

$$2n^2 < m = p_1 \cdots p_\ell \qquad \textit{und} \qquad n < p_j, \quad j = 1, \ldots, \ell, \tag{3.14}$$

erfüllt sind. Dann könnten wir parallel in $\mathbb{Z}_{p_j}$ rechnen und das Ergebnis zuerst mit dem chinesischen Restsatz in $\mathbb{Z}_m$ rekonstruieren und dann eben wieder zu "rekonvertieren".

Nun ist (3.14) ja eine schöne Sache, aber leider beschränkt es die Zahl der Primfaktoren und damit Parallelisierungsmöglichkeiten ganz ordentlich. Es ist natürlich zuerst einmal ziemlich naheliegend, daß die Primfaktoren von m so "gleichgroß" wie möglich sein sollten.

Proposition 3.29 *Sei $m = p_1 \cdots p_\ell$ das Produkt der Primzahlen $p_1, \ldots, p_\ell$ und sei $n \in \mathbb{N}$ die größte Zahl mit der Eigenschaft daß $2n^2 < m$.*

1. *Ist $\ell = 2$ und sind $p_1 < p_2$ zwei* aufeinanderfolgende *Primzahlen, dann ist $n < p_1$.*
2. *Ist $\ell > 2$ und gilt $p_1 < \cdots < p_\ell$, dann ist $p_1 \leq n$.*

Mit anderen Worten: Man kann nur mit zwei Primfaktoren "vernünftig" rechnen.

Beweis: Für 1 bemerken wir[73], daß $p_2 < 2p_1$ und damit ist

$$2n^2 < m = p_1\, p_2 < 2p_1^2 \qquad \Longrightarrow \qquad n < p_1.$$

Für 2 halten wir zuerst fest, daß die Maximalitätsbedingung an n bedeutet, daß

$$n \leq \sqrt{\frac{p_1 \cdots p_\ell - 1}{2}} < n + 1 \qquad \Longrightarrow \qquad \frac{p_1 \cdots p_\ell - 1}{2} < (n+1)^2,$$

also

$$p_1 \cdots p_\ell < 2n^2 + 4n + 3.$$

[73] Das Bertrandsche Postulat, siehe (Hardy & Wright, 1954, S. 343).

Wäre $p_2 \leq n$, dann wäre $p_1 < p_2 \leq n$ und die Behauptung wäre bewiesen; andererseits erhalten wir mit $p_\ell > \cdots > p_2 \geq n$, daß

$$p_1 < p_1 \frac{p_2 \cdots p_\ell}{n^{\ell-1}} < \frac{2n^2+4n+3}{n^{\ell-1}} < \frac{2}{n^{\ell-3}} + \frac{4}{n^{\ell-2}} + \frac{3}{n^{\ell-1}} \leq 2 + \frac{4}{3} + \frac{1}{3} = \frac{11}{3},$$

unter Verwendung der minimalen Werte $n = \ell = 3$. Damit ist aber $p_1 \leq 3 \leq n$. □

Beispiel 3.30 *Sehen wir uns mal ein Beispiel für "bimodulares" Rechnen an und wählen wir* $n = 8$ *und* $m = 143 = 11 \cdot 13$*, sowie die Rechnung*

$$\left(\frac{5}{3} \oplus \frac{1}{5}\right) \otimes \left(\frac{2}{7} \ominus \frac{3}{8}\right).$$

Die Konvertierung, jetzt in Paare *modulo* 11 *und* 13 *liefert für die ersten beiden Zahlen,* $\frac{5}{3}$ *und* $\frac{1}{5}$ *die Werte*

11	0	13	0
3	5	3	5
2	−15	1	−20
1	20		
	→ 9		→ 6

11	0	13	0
5	1	5	1
1	−2	3	−2
		2	3
		1	−5
	→ 9		→ 8

sowie für $\frac{2}{7}$ *und* $-\frac{3}{8}$ *die Ergebnisse*

11	0	13	0
7	2	7	2
4	−2	6	−2
3	4	1	4
1	−6		
	→ 5		→ 4

11	0	13	0
8	−3	8	−3
3	3	5	3
2	−9	3	−6
1	12	2	9
		1	−15
	→ 1		→ 11

Achtung: Im letzten Tableau haben wir bereits die Zahl $-\frac{3}{8}$ *codiert, so daß wir nun* addieren *können. Das heißt, wir berechnen jetzt*

$$([9,6] \oplus [9,8]) \otimes ([5,4] \oplus [1,11]) = [18,14] \times [6,15] = [7,1] \times [6,2] = [42,2] = [9,2]$$

Zur Bestimmung des Ergebnisse modulo 143 *brauchen wir die "Lagrange–Zahlen" aus Algorithmus 2.42,*

$$\ell_1 = \left(\left(\frac{1}{13}\right)_{11} \times 13\right)_{143}, \qquad \ell_2 = \left(\left(\frac{1}{11}\right)_{13} \times 11\right)_{143},$$

also wieder einmal mit Euklid

11	0
13	1
11	0
2	1
1	−5
	→ 6

13	0
11	1
2	−1
1	6
	→ 6

$$\Longrightarrow \qquad \begin{aligned} \ell_1 &= (6 \times 13)_{143} = 78, \\ \ell_2 &= (6 \times 11)_{143} = 66. \end{aligned}$$

Damit ist das Ergebnis unserer Rechnung also $(9 \times 78 + 2 \times 66)_{143} = (834)_{143} = 119$ *und die Rücktransformation ergibt*

143	0
119	1
24	−1
23	5
1	**−6**

$$\Longrightarrow \qquad \left(\frac{5}{3} + \frac{1}{5}\right)\left(\frac{2}{7} - \frac{3}{8}\right) = \frac{28}{15} \times \frac{-5}{56} = -\frac{1}{6}$$

Gut, mit zwei Primfaktoren geht's also. Was aber tun mit mehr als zwei Primfaktoren? Natürlich könnte man immer nur die Fareybrüche F_n mit $n < p_1$ wählen, wobei p_1 wieder der kleinste Primfaktor von $m = p_1 \cdots p_\ell$ ist, aber dann können wir uns ja genauso gut auf $m = p_1 \cdot p_2$ beschränken, die anderen Primfaktoren sind ja nur unnötiger Ballast.
Ist hingegen $p_1 < \cdots < p_k < n < p_{k+1} < \cdots < p_\ell$ für ein $k \in \{1, \ldots, \ell\}$, und $2n^2 < m$, dann ist

$$F_n \cap (\mathbb{Q} \setminus \mathbb{Q}_m) \neq 0.$$

Natürlich könnte man diese Brüche einfach ausschließen und man überlegt sich leicht, daß $\mathbb{Q}_m$ unter Addition, Subtraktion und Multiplikation abgeschlossen ist, aber nun gibt es Brüche in $F_n \cap \mathbb{Q}_m$, deren Reziprokwert nicht mehr in $\mathbb{Q}_m$ liegt, nämlich diejenigen, deren Zähler durch einen unserer Primfaktoren von m teilbar ist. Und schließt man die auch noch aus, verliert man Abgeschlossenheit unter Addition: Sei $m = 3 \cdot p_2 \cdots P_\ell$, dann ist $\frac{1}{b} + \frac{2}{b} = \frac{3}{b}$ schon so ein "schlechter" Bruch.

Beispiel 3.31 *Ein besonders schönes Beispiel ist* $m = 210 = 2 \cdot 3 \cdot 5 \cdot 7$*, denn dann ist* $m = 10$ *und* $\mathbb{Z}_m^* \cap \{-10, \ldots, 10\} = \{\pm 1\}$*, es gibt also eigentlich keinen Fareybruch und jede Division ist sofort zum Scheitern verurteilt!*

Trotzdem gibt es einen Trick, mit dem man auch mit mehreren Moduli arbeiten kann, siehe (Gregory & Krishnamurthy, 1984, S. 56–62). Sei $x = \frac{a}{b} \in F_n$ ein (gekürzter Bruch). Nun können sowohl a als auch b natürlich Potenzen von $p_1 \cdots p_\ell$ enthalten, die wir nun explizit herausziehen, das heißt wir schreiben

$$x = \frac{a}{b} = \frac{a'}{b'} \prod_{j=1}^{\ell} p_j^{e_j}, \qquad \text{ggT}\,(a', p_j) = \text{ggT}\,(b', p_j) = 1, \quad e_j \in \mathbb{Z}, \qquad j = 1, \ldots, \ell. \tag{3.15}$$

Insbesondere ist also $x'_j := p_j^{-e_j}\, x$ auch ein Fareybruch[74], geschrieben als $x'_j = \frac{a_j}{b_j}$, $j = 1, \ldots, \ell$. Unsere "Vorwärtsabbildung" ordnet nun jedem $x \in F_n$ den Vektor

$$f_m(x) = \left[\left(\left(x'_j\right)_{p_j}, e_j\right) \;:\; j = 1, \ldots, \ell\right] \quad \in \quad (\mathbb{Z}_{p_1} \times \mathbb{Z}) \times \cdots \times (\mathbb{Z}_{p_\ell} \times \mathbb{Z}) \tag{3.16}$$

zu.

Beispiel 3.32 *(Fortsetzung von Beispiel 3.31)*
Sehen wir uns einmal ein paar "typische" Fareybrüche an:

$$\begin{aligned}
2 &= \frac{2}{1} = [(1,1),(2,0),(2,0),(2,0)] \\
10 &= [((5)_2, 1), ((10)_3, 0), ((2)_5, 1), ((10)_7, 0)] = [(1,1),(1,0),(2,1),(3,0)] \\
\frac{1}{2} &= 2^{-1} = [(1,-1),(2,0),(3,0),(4,0)] \\
\frac{5}{7} &= \left[\left((5)_2 \cdot \left(7^{-1}\right)_2, 0\right), \left((5)_3 \cdot \left(7^{-1}\right)_3, 0\right), \left(\left(7^{-1}\right)_5, 1\right), ((5)_7, -1)\right] \\
&= \left[\left((1\cdot 1)_2, 0\right), \left(\left(2\cdot 1^{-1}\right)_3, 0\right), \left(\left(2^{-1}\right)_5, 1\right), (5,-1)\right] \\
&= [(1,0),(2,0),(3,1),(5,-1)]\,.
\end{aligned}$$

Die multiplikativen Rechenregeln für solche Vektoren sind ja nun ziemlich einfach:

$$\begin{aligned}
[(u_j, v_j) \;:\; j = 1, \ldots, \ell] \;\otimes\; &\left[\left(u'_j, v'_j\right) \;:\; j = 1, \ldots, \ell\right] \\
= \;&\left[\left(\left(u_j \cdot u'_j\right)_{p_j}, v_j + v'_j\right) \;:\; j = 1, \ldots, \ell\right] \\
[(u_j, v_j) \;:\; j = 1, \ldots, \ell] \;\oslash\; &\left[\left(u'_j, v'_j\right) \;:\; j = 1, \ldots, \ell\right] \\
= \;&\left[\left(\left(u_j \cdot \left(u'_j\right)^{-1}\right)_{p_j}, v_j - v'_j\right) \;:\; j = 1, \ldots, \ell\right],
\end{aligned}$$

aber bei der Addition muß man ein bißchen aufpassen, was wir uns mal in einer Komponente, sagen wir Nummer j, ansehen wollen: Seien also (u_j, v_j) und $\left(u'_j, v'_j\right)$ gegeben und sei, der Einfachheit halber, $v_j \le v'_j$. Dann ist

$$u_j\, p^{v_j} + u'_j\, p^{v'_j} = \left(u_j + u'_j\, p^{v'_j - v_j}\right) p^{v_j} \equiv_{p_j} p^{v_j} \cdot \begin{cases} u_j, & v_j < v'_j, \\ \left(u_j + u'_j\right)_{p_j}, & v_j = v'_j. \end{cases}$$

Beispiel 3.33 *(Fortsetzung von Beispiel 3.32)*
Rechnen wir doch mal $2 + \frac{1}{2}$ *und* $1 - \frac{5}{7}$ *und* $9/10$ *in dieser Arithmetik. Im ersten Fall ist*

$$\begin{aligned}
2 + \frac{1}{2} &= [(1,1),(2,0),(2,0),(2,0)] \oplus [(1,-1),(2,0),(3,0),(4,0)] \\
&= [(1,-1),((2+2)_3,0),((3+2)_5,0),((2+4)_7,0)] \\
&= [(1,-1),(1,0),(0,0),(6,0)]\,,
\end{aligned}$$

[74] Denn wir kürzen ja einen Faktor $p_j^{|e_j|}$ entweder aus dem Zähler (der Fall $e_j > 0$) oder aus dem Nenner (der Fall $e_j < 0$) von x.

im zweiten Fall erhalten wir

$$\begin{aligned} 1-\frac{5}{7} &= [(1,0),(1,0),(1,0),(1,0)] \ominus [(1,0),(2,0),(3,1),(5,-1)] \\ &= [(1,0),(1,0),(1,0),(1,0)] \oplus [(1,0),(1,0),(2,1),(2,-1)] \\ &= [((1+1)_2,0),((1+1)_3,0),(1,0),(2,-1)] \\ &= [(0,0),(2,0),(1,0),(2,-1)], \end{aligned}$$

und schließlich

$$\begin{aligned} 9/10 &= [(1,0),(1,2),(4,0),(2,0)] \oslash [(1,1),(1,0),(2,1),(3,0)] \\ &= [(1,0),(1,2),(4,0),(2,0)] \otimes [(1,-1),(1,0),(3,-1),(5,0)] \\ &= [((1\cdot 1)_2,0+(-1)),((1\cdot 1)_3,2+0),((4\cdot 3)_5,0+(-1)),((2\cdot 5)_7,0+0)] \\ &= [(1,-1),(1,2),(2,-1),(3,0)]. \end{aligned}$$

Bleibt also noch die "Rückrechnung". Sei also so ein Vektor $[(u_j,v_j)\ :\ j=1,\ldots,\ell]$ gegeben. Wären $v_1=\cdots=v_\ell=0$, dann ist die Sache einfach: Wir verwendet ganz einfach unseren chinesischen Restsatz. Ansonsten müssen wir bei unserer Rekonstruktion der zugehörigen Zahl aus $\mathbb{Z}_m$ eine Fallunterscheidung machen

$v_j<0$: Diese Zahl ist in $\mathbb{Z}/\langle m\rangle$ nicht darstellbar[75], es sei denn, wir multiplizieren sie mit $p_j^{-v_j}$ und merken uns diesen Faktor.

$v_j>0$: Die durch diesen Vektor dargestellte Zahl ist ein Vielfaches von p_j, also trägt diese Zahl den Anteil $0\cdot\ell_j$ bei der Rekonstruktion bei und wir können sie mehr oder weniger vergessen, was wir dann auch tun werden.

Das führt zum folgenden Verfahren.

Algorithmus 3.34 *(Konvertierung multimodularer Zahlen)*
Gegeben: *Darstellung*

$$\alpha=[(u_j,v_j)\ :\ j=1,\ldots,\ell]\in(\mathbb{Z}_m\times\mathbb{Z})^\ell.$$

1. *Setze*

$$I_+:=\{j\ :\ v_j>0\},\qquad I_-:=\{j\ :\ v_j<0\},\qquad I_0:=\{j\ :\ v_j=0\}$$

und

$$m^*\leftarrow\left(\prod_{j\in I_0}p_j\right)\cdot\left(\prod_{j\in I_-}p_j\right)=m\cdot\left(\prod_{j\in I_+}p_j\right)^{-1}.$$

[75] Hier sind wir mal ein bißchen genauer: $\mathbb{Z}_m$ steht für die Menge $\{0,\ldots,m-1\}$, während $\mathbb{Z}/\langle m\rangle$ die Restklassen mit entsprechenden Rechenoperationen bezeichnet.

2. *Setze*
$$p_- \leftarrow \prod_{k\in I_-} p_k^{-v_k}.$$

3. *Für $j \in I_0$ setze*
$$u_j \leftarrow \left(p_- \cdot u_j\right)_{p_j}.$$

4. *(CRT–Algorithmus modulo m^*): Für $j \in I_0 \cup I_-$ setze*
$$m_j^* \leftarrow \frac{m^*}{p_j}, \qquad \ell_j \leftarrow \left(\left(m_j^*\right)_{p_j}^{-1} m_j^*\right)_{m^*}.$$

5. *Setze*
$$q \leftarrow \left(\sum_{j\in I_0\cup I_-} u_j \cdot \ell_j\right)_{m^*}$$

6. *Wandle q in einen Farey–Bruch um.*

Ergebnis: $q = p_-\widetilde{q}$.

Bemerkung 3.35 *(Multimodulare Invertierung)*

1. *Bei dieser Arithmetik, insbesondere bei der Invertierungsmethode hier, werden zuerst Vielfache der Primfaktoren von m aus dem Nenner herausgezogen. Das heißt aber insbesondere auch, daß nun nicht nur die Fareybrüche F_n, $n = \lfloor\sqrt{m/2}\rfloor$, dargestellt werden können, sondern alle Brüche der Form*
$$F_n \cdot \prod_{j=1}^{\ell} p_j^{-e_j}, \qquad 0 \le e_j, \quad j = 1, \ldots, \ell.$$

2. *Mit den Zählern ist das ein bißchen anders! Sobald nämlich mindestens einer der Exponenten v_j positiv ist, ist $m^* < m$ und die Rekonstruktion[76] der durch α dargestellten Zahl erfolgt nicht mehr in $\mathbb{Z}_m$ sondern in $\mathbb{Z}_{m^*} \subset \mathbb{Z}_m$. So erhält man beispielsweise für*
$$49 \simeq [(1,0),(1,0),(4,0),(1,2)] \to 19,$$
was modulo $m^* = 30$ ja auch völlig korrekt ist.

   Multiplizieren wir erst mal mit $\frac{1}{7}$ durch, schreiben wir also
$$[(1,0),(1,0),(4,0),(1,2)] = 7 \cdot [(1,0),(1,0),(2,0),(1,1)] \quad \to 7 \cdot 7 = 49$$
so erhalten wir auch die 49 zurück. Wie gesagt, an den Details kann man hier noch feilen ...

[76] Via CRT–Algorithmus.

3. *Eine vernünftige Strategie scheint es also zu sein, "Zählerfaktoren" so lange herauszuziehen (und entsprechend damit durchzumultiplizieren), bis man eine Darstellung*

$$x = \prod_{j=1}^{\ell} p_j^{e_j} \cdot [(u_j, v_j) \ : \ j = 1, \ldots, \ell], \qquad v_j \in \{0, 1\}$$

hat.

4. *Die multimodulare Arithmetik ist in (Gregory & Krishnamurthy, 1984, S. 56–62) dargestellt, allerdings ohne Details. Für diese wird*[77] *auf ein "forthcoming paper" von Carl Gregory*[78] *mit David Matula verwiesen:*

> Because a rigorous treatment of this method will appear in a paper by Matula and Gregory (soon after this book is published) we do not include the derivation of the method here.

Nach Kenntnisstand 16. Mai 2001 ist diese Arbeit bis heute nicht erschienen, was auch immer uns das sagen will . . .

Beispiel 3.36 *(Fortsetzung von Beispiel 3.33) Lassen wir uns doch mal überraschen und sehen wir uns an, was die beiden Ergebnisse der Rechnungen in Beispiel 3.33 so liefern.*

1. *Wir erhalten für das Ergebnis von* $2 + \frac{1}{2}$*, daß*

$$[(1,-1),(1,0),(0,0),(6,0)] \ = \ \frac{1}{2}\,[(1,0),(2,0),(0,0),(5,0)] = \frac{1}{2} \cdot 5 = \frac{5}{2}.$$

2. *Im zweiten Fall, also für* $1 - \frac{5}{7}$*, war das Ergebnis*

$$\begin{aligned}[(0,0),(2,0),(1,0),(2,-1)] \ &= \ \frac{1}{7}\,[(1\cdot 0,0),(1\cdot 2,0),(2\cdot 1,0),(2,0)] \\ &= \ \frac{1}{7}\,[(0,0),(2,0),(2,0),(2,0)] = \frac{1}{7} \cdot 2 = \frac{2}{7}.\end{aligned}$$

3. *Im dritte Fall, für* $9/10$*, bekommen wir schließlich*

$$\begin{aligned}&[(1,-1),(1,2),(2,-1),(3,0)] \\ &= \ \frac{1}{2\cdot 5}\,[((5)_2 \cdot 1,0)\,,((2)_3 \cdot (5)_3 \cdot 1,2)\,,(2\cdot 2,0)\,,((2)_7 \cdot (5)_7 \cdot 3,0)] \\ &= \ \frac{1}{10}\,[(1,0),(1,2),(4,0),(2,0)] \\ &= \ \frac{1}{10}\Big(1\cdot(5\cdot 7)_2^{-1}\cdot(5\cdot 7) + 4\cdot(2\cdot 7)_5^{-1}\cdot(2\cdot 7) + 2\cdot(2\cdot 5)_7^{-1}\cdot(2\cdot 5)\Big)_{70} \\ &= \ \frac{1}{10}\left(1\cdot(1)_2^{-1}\cdot 35 + 4\cdot(4)_5^{-1}\cdot 14 + 2\cdot(3)_7^{-1}\cdot 10\right)_{70} \\ &= \ \frac{1}{10}\Big(35 + \underbrace{4\cdot 4\cdot 14}_{=224} + \underbrace{2\cdot 5\cdot 10}_{=100}\Big)_{70} = \frac{1}{10}\,(359)_{70} = \frac{1}{10}\,9 = \frac{9}{10}.\end{aligned}$$

[77] Das Buch erschien 1984!

[78] Angeblich der Sohn des einen Autors von (Gregory & Krishnamurthy, 1984)

Es wäre sicherlich mal interessant, eine laufende Implementierung einer derartigen Arithmetik zu sehen und an praktischen Beispielen[79] zu testen. Um "vernünftig" zu funktionieren müßte eine solche Arithmetik natürlich auf einer Vielzahl von Primzahlen basieren, die "gerade noch" in ein Maschinenwort passen.

[79] Determinantenberechnungen, numerische lineare Algebra mit fastsingulären Matrizen, ...

La mia scrittura è piana,
e la speranza di costor non falla,
se ben si guarda con la mente sana[a].

Dante, "La divina comedia", *Purgatore*, Canto VI

[a] Das was ich schrieb liegt offen;
und jenen wird die Hoffnung nicht zuschanden,
wenn man gesunden Geists den Sinn getroffen.
(Deutsch von R. Zoozmann), aus (Laaths, 1994)

4 Rechnen mit (univariaten) Polynomen

Jetzt wollen wir uns also mit Polynomen beschäftigen. Dabei kommt es uns natürlich zugute, daß wir uns einige Sachen schon im Kontext der euklidischen Ringe angesehen haben. Trotzdem ist für Polynome trotzdem noch einiges anders.

Wir werden in diesem Kapitel zumeist Polynome in $R[x]$ betrachten, das heißt, Polynome, deren Koeffizienten "nur" in einem Ring R liegen brauchen. Und das macht übrigens einen ganz gewaltigen Unterschied: $\mathbb{R}[x]$ ist (bekanntlich) ein euklidischer Ring, $\mathbb{Z}[x]$ aber nicht mehr – man braucht nur mal versuchen, Division mit Rest mit den beiden Polynomen $f(x) = 3x^2$ und $g(x) = 2x^2$ zu betreiben ...

4.1 Schnelle Polynommultiplikation I – Karatsuba

Das erste Verfahren zur "beschleunigten" Berechnung des Produkts zweier Polynome basiert auf einer einfachen Idee, die sich übrigens auch hinter der sogenannten *schnellen Matrixmultiplikation* versteckt. Multiplizieren wir nämlich zwei Polynome $f(x) = ax+b$ und $g(x) = cx+d$, dann erhalten wir das Produkt

$$(f \cdot g)(x) = ac\,x^2 + (ad + bc)\,x + bd$$

mit

1. den vier Multiplikationen $a \otimes c$, $a \otimes d$, $b \otimes c$ und $b \otimes d$,
2. der Addition $ad \oplus bc$.

Es geht aber auch anders, nämlich mit

1. den zwei Additionen $a \oplus b$, $c \oplus d$,
2. den *drei* Multiplikationen $a \otimes c$, $b \otimes d$ und $u = (a \oplus b) \otimes (c \oplus d)$
3. den zwei Additionen $u - a \otimes c - b \otimes d$.

Gut, das allein ist noch nicht so ganz der große Knaller, wird sich aber gleich als lohnend erweisen, denn jetzt kommt die zentrale Idee der meisten "schnellen" Algorithmen:

Zweiterpotenzen und Rekursion

Sei nämlich mal $n = 2^k$ und $f(x) = ax^n + \cdots$ sowie $g(x) = bx^n + \cdots$, also $f, g \in \Pi_n$ dann schreiben wir

$$f(x) = f_1(x)x^{n/2} + f_0(x), \qquad g(x) = g_1(x)x^{n/2} + g_0(x), \qquad f_0, f_1, g_0, g_1 \in \Pi_{n/2} = \Pi_{2^{k-1}}, \tag{4.1}$$

und erhalten, daß

$$(fg)(x) = f_1(x)g_1(x)\,x^n + (f_0(x)g_1(x) + f_1(x)g_0(x))\;x^{n/2} + f_0(x)g_0(x),$$

wobei wir die "Faktoren" mit drei Multiplikationen, zwei Additionen von Polynomen vom Grad $n/2$, zwei Additionen von Polynomen vom Grad n und drei Additionen von Polynomen vom Grad $2n$ durchführen können. Nachdem der Aufwand bei der Addition zweier Polynome vom Grad n aus $n+1$ Operationen in R besteht, haben wir also einen Gesamtaufwand von

$$2 \times \left(\frac{n}{2} + 1\right) + 2\,(n+1) + 3\,(2n+1) = 9n + 7 \leq 16n$$

Operationen in R. Außerdem müssen wir ja noch (rekursiv!) die drei Multiplikationen vom Grad $n/2$ durchführen, so daß unser Gesamtaufwand $E(n)$ die Rekursionsformel

$$E(n) \leq 3E\,(n/2) + 16n =: 3E\,(n/2) + S(n), \tag{4.2}$$

erfüllt. Da $S(2n) = 2S(n)$ erhalten wir somit, daß für $n = 2^j$, also $j = \log_2 n$,

$$\begin{aligned}
E\left(2^j\right) &\leq 3E\left(2^{j-1}\right) + S\left(2^j\right) \leq 3\left(3E\left(2^{j-2}\right) + S\left(2^{j-1}\right)\right) + S\left(2^j\right) \leq \cdots \\
&\leq 3^j E(1) + \sum_{k=0}^{j-1} 3^k S\left(2^{j-k}\right) = 3^j E(1) + S\left(2^j\right) \sum_{k=0}^{j-1} \frac{3^k}{2^k} \\
&= 3^j E(1) + S(n)\frac{(3/2)^j - 1}{3/2 - 1} \leq E(1)\; 2^{(\log_2 3)\,j} + 16\,n\;\frac{2 \cdot 2^{(\log_2 3 - 1)\,j}}{3-2} \\
&= E(1)\; 2^{(\log_2 3)\,(\log_2 n)} + 32\,n\; 2^{(\log_2 3 - 1)\,(\log_2 n)} \\
&= E(1)\; n^{\log_2 3} + 32n\; n^{\log_2 3 - 1} = O\left(n^{\log_2 3}\right),
\end{aligned}$$

also

$$E\,(n) = O\left(n^{\log_2 3}\right) \sim O\left(n^{1.59}\right). \tag{4.3}$$

Da $\log_2 3 \sim 1.59$, haben wir jetzt also die wirklich signifikant bessere Komplexitätsschranke $O\left(n^{1.59}\right)$ für die Multiplikation zweier Polynome vom Grad $n = 2^j$ – zum Vergleich: vorher war es $O\left(n^2\right)$. Damit sieht also der Algorithmus von Karatsuba (Karatsuba & Ofman, 1963) zur schnellen Multiplikation von Polynomen wie folgt aus.

Algorithmus 4.1 *(Karatsuba–Multiplikation)*
Gegeben: *Polynome f, g vom Grad $n = 2^j$.*

1. *Bestimme*[80] f_0, f_1, g_0, g_1*, so daß*
$$f(x) = f_1(x)\, x^{n/2} + f_0(x), \qquad g(x) = g_1(x)\, x^{n/2} + g_0(x).$$
2. *Berechne*
$$a \leftarrow f_1 \oplus f_0, \qquad b \leftarrow g_1 \oplus g_0.$$
3. *Berechne (rekursiv)*
$$\begin{aligned} c &\leftarrow a \otimes b \\ d &\leftarrow f_1 \otimes g_1 \\ e &\leftarrow f_0 \otimes g_0 \end{aligned}$$
4. *Berechne*
$$u \leftarrow c \ominus d \ominus e.$$
5. *Berechne*[81]
$$h(x) \leftarrow d(x) \cdot x^n \oplus u(x) \cdot x^{n/2} \oplus e(x).$$

Ergebnis: $h(x) = f(x) \cdot g(x)$.

Bemerkung 4.2
1. *Es sieht zuerst einmal so aus, als hätte der Karatsuba–Algorithmus nur für Zweierpotenzen die gute Komplexitätsabschätzung. Was aber, wenn n keine Zweierpotenz ist? Auch nicht so schlimm, denn daß sich im Intervall $[n, 2n)$ immer eine Zweierpotenz $m = 2^j$ finden läßt*[82]*, können wir unsere Polynome f, g (vom gleichen Grad) einfach auf die nächste Zweierpotenz auffüllen und erhalten, daß der Aufwand $\leq O\left(m^{\log_2 3}\right) \leq O\left((2n)^{\log_2 3}\right) = O\left(n^{\log_2 3}\right)$ mit der moderaten Konstante 3 ist.*
2. *Bei der Multiplikation zweier Polynome von signifikant* verschiedenem *Grad füllt man natürlich nur auf die jeweils nächste Zweierpotenz auf und bricht ab, wenn eines der beiden Polynome konstant geworden ist, denn dann hat man es nur noch mit der Multiplikation mit einem Körperelement zu tun – eine Operation mit traditionell linearem Aufwand.*

[80] Das ist nur ein "Halbieren" der Koeffizienten, mit keinen Rechenoperationen verbunden.

[81] Jetzt bauen wir das Ergebnis zusammen! Die "Multiplikationen" mit Monomen sind wieder nur Schiebeoperationen.

[82] Der Beweis ist denkbar einfach: Wir schreiben $n = 2^j + n_{j-1}2^{j-1} + \cdots$, dann ist $n \leq 2^{j+1} \leq 2n$, oder wir bemerken, daß $n \leq 2^{\lceil \log_2 n \rceil} \leq 2n$.

3. *Man kann dieses Verfahren sofort auch in ein Multiplikationsverfahren für Multiprecision–Zahlen umwandeln: Hier werden dann eben die Zahlen in zwei gleichgroße Happen zerlegt, diese rekursiv multipliziert und schließlich wieder zusammengesetzt.*

Übung 4.1 Formulieren Sie das Karatsuba–Verfahren für die Multiplikation von Multiprecision–Zahlen.
Ein analoges Verfahren gibt es übrigens auch zur Multiplikation von (Block–)Matrizen, siehe (Golub & van Loan, 1996, S. 31–33) und (ausführlicher) (Higham, 1996, S. 446–463), wobei der Trick darin besteht, die Multiplikation von 2×2–Matrizen mit 7 anstelle von 8 Multiplikationen zu berechnen. Zusammen mit einer rekursiven Zerlegung verringert sich dann der Aufwand von $O\left(n^3\right)$ der "naiven" Methode auf $O\left(n^{\log_2 7}\right) \sim O\left(n^{2.807}\right)$; wie man darauf kommt, dürfte jetzt nicht mehr so schwer nachzuvollziehen sein.

4.2 Schnelle Polynommultiplikation II – DFT und FFT

Das wesentliche Hilfsmittel zur schnellen Multiplikation von Polynomen wird die *schnelle Fouriertransformation*[83] (FFT) zur Bestimmung der *diskreten Fouriertransformation* (DFT) einer Funktion, in diesem Fall eines Polynoms, sein.

Definition 4.3 *Sei R ein kommutativer Ring mit 1.*

1. *Wir betten $\mathbb{N}$ in R ein, indem wir die Zahl $n \in \mathbb{N}$ mit*

$$n \cdot 1 = \underbrace{1 + \cdots + 1}_{n} \in R$$

   identifizieren.

2. *Ein Element $\omega \in R$ heißt n–te* Einheitswurzel*, wenn $\omega^n = 1$.*

3. *Eine n–te Einheitswurzel $\omega \in R$ heißt* primitiv*, wenn*[84] *$n \in R^*$ ist und wenn gilt: für alle Primfaktoren p von n ist $\omega^{n/p} - 1$ kein Nullteiler in R.*

Beispiel 4.4 *(Einheitswurzeln)*

1. *In $R = \mathbb{C}$ ist $z = e^{i\pi/4} = e^{2i\pi/8}$ eine achte Einheitswurzel, da $z^8 = e^{2i\pi} = 1$. Da außerdem $8 \in \mathbb{C}^* = \mathbb{C} \setminus \{0\}$ und $z^4 - 1 = e^{i\pi} - 1 = -1 - 1 = -2$, ist z auch eine primitive achte Einheitswurzel.*

2. *In $\mathbb{Z}/\langle 8\rangle = \mathbb{Z}/8\mathbb{Z}$ gibt es keine primitiven Einheitswurzeln! Zwar sind $3, 5, 7$ Einheitswurzeln und zwar zweite,*

$$3^2 = 9 \equiv_8 5^2 = 25 \equiv_8 7^2 = 49 \equiv_8 1,$$

   aber da $2 \notin (\mathbb{Z}_8/\langle 8\rangle)^*$, brauchen wir uns für die Nullteilerfrage gar nicht interessieren.

[83] Es sollte uns inzwischen nicht mehr überraschen, daß "analytische" Verfahren auch bei "algebraischen" Problemen erfolgreich Anwendung finden - wie übrigens natürlich (!) auch umgekehrt. Mathematik als solche widersetzt sich eben doch dem Schubladendenken.

[84] Im Sinne der Einbettung von Teil 1 dieser Definition.

3. *Im Körper $\mathbb{F}_{17}$ ist 3 eine primitive 16–te Einheitswurzel. Die Zahl 2 hingegen ist "nur" eine primitive achte Einheitswurzel: $2^4 = 16 \equiv_{17} -1$, also $2^8 = 1$. Dennoch ist 2 natürlich auch eine 16te Einheitswurzel*[85]*, aber eben keine primitive.*

4. *Etwas interessanter wird es in $\mathbb{Z}/\langle 14\rangle$. Hier ist 3 eine sechste Wurzel, da*[86]

$$3^2 = 9, \qquad 3^3 = 13, \qquad 3^4 = 11, \qquad 3^5 = 5, \qquad 3^6 = 1,$$

aber natürlich keine *primitive Einheitswurzel, da weder die Potenz 6 eine Einheit in $\mathbb{Z}/\langle 14\rangle$ ist, und außerdem gilt für beide Primteiler $2, 3$ von 6, daß*

$$3^{6/2} - 1 = 3^3 - 1 = 13 - 1 = 12 \qquad \textit{und} \qquad 3^{6/3} - 1 = 3^2 - 1 = 9 - 1 = 8$$

Nullteiler*, und zwar von Null verschiedene, in $\mathbb{Z}/\langle 14\rangle$ sind. Bei 11 ist es noch schöner: da $11^3 \equiv_{14} 1$, ist 11 eine dritte Einheitswurzel, 3 ist eine Einheit in $\mathbb{Z}/\langle 14\rangle$, aber leider ist $11 - 1 = 10$ trotzdem ein Nullteiler. Dasselbe gilt übrigens auch für 9.*

Da $(\mathbb{Z}/\langle 14\rangle)^* = \{1, 3, 5, 9, 11\}$ gibt es also dritte und sechste Einheitswurzeln[87]*, aber keine primitiven Einheitswurzeln in $\mathbb{Z}/\langle 14\rangle$.*

5. *Schön langsam beginnt man zu zweifeln, ob es überhaupt primitive Einheitswurzeln in "Nichtkörpern" gibt*[88]*. Ein einfaches Beispiel ist $\omega = 8$ und $n = 2$ in $\mathbb{Z}/\langle 9\rangle$, etwas mehr suchen muß man für $\omega = 20$, $n = 13$ und $\omega = 23$, $n = 13$ sowie $\omega = 26$, $n = 2$ in $\mathbb{Z}/\langle 27\rangle$.*

Bemerkung 4.5 *1. Jede Einheitswurzel ist eine Einheit: $1 = \omega^n = \omega^{n-1} \cdot \omega$ bedeutet ja, daß $\omega^{-1} = \omega^{n-1}$.*

2. *Da $(\omega^{-1})^n = (\omega^n)^{-1} = 1$, ist auch ω^{-1} eine n–te Einheitswurzel.*

3. *Auch Primitivität bleibt erhalten, da für jeden Primteiler p von n*

$$\left(\omega^{-1}\right)^{n/p} - 1 = \omega^{-n/p}\left(1 - \omega^{n/p}\right)$$

und weil die rechte Seite kein Nullteiler ist, kann auch die linke Seite keiner sein.

In Wirklichkeit gilt aber die "kein Nullteiler"–Eigenschaft von primitiven Einheitswurzeln nicht nur für Potenzen $\omega^\ell - 1$, wobei $\ell = n/p$ ist, sondern für alle $\ell < n$.

Lemma 4.6 *Sei $\omega \in R$ eine primitive n-te Einheitswurzel*[89] *in R. Dann gilt:*

[85] Denn $2^{16} = \left(2^8\right)^2 = 1^2 = 1$.

[86] Immer modulo 14 gesehen.

[87] Und zwar echte!

[88] In einem Körper ist eine Einheitswurzel der Ordnung n primitiv, wenn es kein $k \leq n$ gibt, so daß $\omega^k = 1$. Die nichtprimitiven Einheitswurzeln sind also Einheitswurzeln, die durch "Potenzieren der 1" entstanden sind.

[89] Das soll natürlich auch automatisch unsere "Standardsituation", daß R ein kommutativer Ring mit 1 ist implizieren.

1. *Für alle* $j = 1, \ldots, n-1$ *ist* $\omega^j - 1$ *kein Nullteiler in* R.
2. *Für alle* $j = 1, \ldots, n-1$ *ist*

$$\sum_{k=0}^{n-1} \omega^{jk} = 0. \tag{4.4}$$

Beweis: Die entscheidende Beweistechnik ist die einfache[90] Identität

$$(a-1)\sum_{j=0}^{m-1} a^j = a^m - 1, \qquad a \in R. \tag{4.5}$$

Für 1. wählen wir $1 < j < n$ und setzen $\ell = \text{ggT}\ (j, n)$. Da $\ell | n$ und $\ell < n$ hat $\frac{n}{\ell} \in \mathbb{N}$ mindestens einen nichttrivialen Primfaktor p, das heißt, es gibt eine Zahl $m \in \mathbb{N}$, so daß $\frac{n}{\ell} = mp$, also $m = \frac{n}{\ell p}$. Nach (4.5) ist

$$\left(\omega^\ell - 1\right)\sum_{j=0}^{m-1} \omega^{\ell j} = \omega^{\ell m} - 1 = \omega^{(\ell n)/(\ell p)} = \omega^{n/p} - 1$$

und da die rechte Seite kein Nullteiler ist, kann auch $\omega^\ell - 1$ kein Nullteiler sein, denn sonst hätten wir für ein $a \in R \setminus \{0\}$ den Widerspruch

$$0 = a \cdot \left(\omega^\ell - 1\right) = a\ \left(\omega^\ell - 1\right)\sum_{j=0}^{m-1} \omega^{\ell j} = a\left(\omega^{n/p} - 1\right).$$

Um nun schließlich von ℓ zu j zu kommen wählen wir Bézout–Koeffizienten $s, t \in \mathbb{Z}$ so daß $\ell = sj + tn$ und $s > 0$[91]. Dann ist, wieder mit (4.5),

$$\omega^\ell - 1 = \omega^{sj+tn} - 1 = \omega^{sj} \underbrace{\omega^{nt}}_{=(\omega^n)^t = 1^t = 1} - 1 = \omega^{sj} - 1 = \left(\omega^j - 1\right)\sum_{k=0}^{s-1} \omega^{kj},$$

also ist auch $\omega^j - 1$ ebenfalls kein Nullteiler in R.
Hier steckt auch schon die Idee für den Beweis von (4.4): Da, nochmals unter Verwendung von (4.5),

$$0 = 1 - 1 = \omega^0 - 1 = \omega^{nj}\omega^{-nj} - 1 = \left(\omega^j\right)^n \underbrace{\left(\omega^n\right)^{-j}}_{=1^{-j}=1} - 1 = \left(\omega^j - 1\right)\sum_{k=0}^{n-1} \omega^{kj}$$

ist und da nach Teil 1. $\omega^j - 1$ *kein* Nullteiler ist, muß also (4.4) gelten. □

Jetzt aber zur Definition der diskreten Fouriertransformation.

[90] Aber in ihrer Bedeutung nicht zu unterschätzende.

[91] Ist (s, t) ein Paar von Bézout–Koeffizienten, dann ist, für jedes $k \in \mathbb{Z}$ auch $(s + kn, t - kj)$ ein Paar von Bézout–Koeffizienten und diese Parametrisierung liefert auch alle solchen Paare. In der Wavelet–Gemeinde hat diese tiefliegende Beobachtung unter dem Namen *"Lifting scheme"* große Popularität erlangt.

Definition 4.7 *Sei $\omega \in R$ eine primitive n–te Einheitswurzel und $f \in R[x]$. Dann ist die* diskrete Fouriertransformation *(DFT) $f^\wedge(\omega) \in R^n$ von f definiert als*

$$f^\wedge(\omega) = \left(f\left(\omega^j\right) \;:\; j = 0, \ldots, n-1\right).$$

Bemerkung 4.8 *(DFT)*

1. *Die DFT operiert eigentlich nur auf $R[x]/\langle x^n - 1\rangle$: Ist nämlich*

$$f(x) = p(x)\,(x^n - 1) + q(x), \qquad q \in \Pi_{n-1},$$

dann ist

$$f\left(\omega^j\right) = p\left(\omega^j\right)\overbrace{\left(\underbrace{(\omega^n)^j}_{=1} - 1\right)}^{=0} + q\left(\omega^j\right) = q\left(\omega^j\right).$$

Mit anderen Worten[92]*:*

$$f \equiv_{x^n-1} g \qquad \Longrightarrow \qquad f^\wedge(\omega) = g^\wedge(\omega). \tag{4.6}$$

2. *Damit können wir also, was die DFT angeht, für jedes Polynom die Identifizierung*[93]

$$\begin{array}{ccccccc} R[x] & \simeq & R[x]/\langle x^n - 1\rangle & \simeq & \Pi_{n-1} & \simeq & R^n \\ f(x) & \simeq & \left[(f(x))_{x^n-1}\right] & \simeq & \displaystyle\sum_{j=0}^{n-1} f_j x^j & \simeq & f = [f_j \;:\; j \in \mathbb{Z}_n] \end{array} \tag{4.7}$$

verwenden. Ob wir nun das Symbol f für den Vektor oder das Polynom benutzen, wird aus dem Kontext klar werden[94]*.*

3. *Der ganze Ansatz basiert auf der* Existenz *von primitven $n$ten Einheitswurzeln – und die muß ja erst mal gewährleistet sein. Tatsächlich gilt die folgende Aussage:*

> In einem endlichen Körper $\mathbb{F}_q$, $q = p^m$, p eine Primzahl, gibt es genau dann eine primitive $n$te Einheitswurzel, wenn $n \mid q - 1$.

Wenn wir uns die DFT ansehen, dann wird also einem Polynom vom Grad $n-1$ ein Vektor von n Funktionswerten des Polynoms zugeordnet, was schon wieder verdächtig nach Lagrange–Interpolation riecht. Und tatsächlich erhalten wir, unter Verwendung der Identifizierung (4.6),

[92] Eigentlich nicht "Worten" sondern "Symbolen"

[93] Als Abkürzung für die Menge $\{0, \ldots, n-1\}$ wird jetzt $\mathbb{Z}_n$ verwendet werden. Aber nur für die Zahlenmenge.

[94] Kleine Fußnote am Rande: Wie stellt man in der Praxis, am Computer, ein Polynom dar? Richtig, als Vektor seiner Koeffizienten bezüglich einer Basis – diese muß aber nicht notwendigerweise aus den Monomen bestehen.

daß

$$\begin{aligned} f^\wedge(\omega) &= \left(f\left(\omega^j\right) \,:\, j \in \mathbb{Z}_n\right) = \left(\left(\sum_{k\in\mathbb{Z}_n} f_k\, x^k\right)\left(\omega^j\right) \,:\, j \in \mathbb{Z}_n\right) \\ &= \left(\sum_{k\in\mathbb{Z}_n} \omega^{jk} f_k \,:\, j \in \mathbb{Z}_n\right) = \left[\omega^{jk} \,:\, j,k \in \mathbb{Z}_n\right]\,[f_j \,:\, j \in \mathbb{Z}_n] \\ &= \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{n-2} & \dots & \omega \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} =: V_\omega\, f. \end{aligned}$$

Die Matrix V_ω bezeichnet man als die zugehörige *Vandermonde–Matrix* und ihre Inverse löst das *Interpolationsproblem:*

Finde $g \in \Pi_{n-1}$ so daß $g\left(\omega^j\right) = \left(f^\wedge(\omega)\right)_j$, $j \in \mathbb{Z}_n$.

Proposition 4.9 *Ist $\omega \in R$ eine primitive n–te Einheitswurzel, dann ist die Matrix V_ω invertierbar und ihre Inverse ist*[95] $n^{-1}V_{\omega^{-1}}$. *Insbesondere kann man die inverse DFT $f^\vee : R^n \to R[x]$ berechnen:*

$$f^\vee(x) = n^{-1} \sum_{j\in\mathbb{Z}_n} \left(V_{\omega^{-1}} f\right)_j x^j = \left(n^{-1} f^\wedge\left(\omega^{-1}\right)\right)(x), \qquad f \in R^n. \tag{4.8}$$

Beweis: Als n–te Einheitswurzel ist ω ja invertierbar[96], also ist $V_{\omega^{-1}}$ wohldefiniert. Nun ist für $j,k \in \mathbb{Z}_n$

$$\left(V_\omega V_{\omega^{-1}}\right)_{jk} = \sum_{\ell\in\mathbb{Z}_n} \left(V_\omega\right)_{j\ell} \left(V_{-\omega}\right)_{\ell k} = \sum_{\ell\in\mathbb{Z}_n} \omega^{j\ell}\, \omega^{-\ell k} = \sum_{\ell\in\mathbb{Z}_n} \omega^{\ell(j-k)} = \begin{cases} n, & j = k, \\ 0, & j \neq k, \end{cases}$$

was wir erhalten, indem wir (4.4) auf ω bzw. ω^{-1} anwenden, je nachdem, ob $j > k$ oder $k > j$ ist. Also ist $V_\omega\, V_{-\omega} = nI$ oder eben $V_\omega^{-1} = n^{-1}V_{-\omega}$. Der Rest ist offensichtlich. □

Die Berechnung der DFT auf naive Art und Weise benötigt also n Auswertungen des Polynoms f an den Stellen ω^j, $j \in \mathbb{Z}_n$. Gemäß (4.7) können wir sogar annehmen, daß $f \in \Pi_{n-1}$ ist[97] und dann kostet uns eine Auswertung von f an einer Stelle x, zum Beispiel mit dem Hornerschema $O(n)$ Operationen im Ring R – und entscheidend besser geht es auch nicht, denn jeder Koeffizient muß ja bei mindestens einer Rechenoperation mitspielen dürfen. Also kommen wir bei individueller Auswertung insgesamt auf $O\left(n^2\right)$ Operationen.

[95] Und jetzt wird auch klar, warum n eine Einheit in R sein mußte.

[96] Siehe Bemerkung 4.5

[97] Alle Terme höherer Ordnung spielen keine Rolle.

Sei, und das sollte uns nicht mehr überraschen, jetzt wieder $n = 2^\ell$ und $\omega \in R$ eine primitive n–te Einheitswurzel. Dann zerlegen wir das Polynom $f \in \Pi_{n-1} \simeq R[x]/\langle x^n - 1\rangle$ per Division mit Rest in

$$f(x) = \left(x^{n/2} + 1\right) g(x) + r_+(x) = \left(x^{n/2} - 1\right) g(x) + r_-(x), \qquad g, r_\pm \in \Pi_{n/2-1}. \tag{4.9}$$

Diese Zerlegung ist sehr effizient zu berechnen:

$$f(x) = \sum_{j\in\mathbb{Z}_n} f_j\, x^j \qquad \Rightarrow \qquad \left\{ \begin{array}{rcl} g(x) & = & \displaystyle\sum_{j\in\mathbb{Z}_{n/2}} f_{j+n/2}\, x^j, \\ r_\pm(x) & = & \displaystyle\sum_{j\in\mathbb{Z}_{n/2}} \left(f_j \mp g_j\right)\, x^j, \end{array} \right. \tag{4.10}$$

also ist

$$r_\pm(x) = \sum_{j\in\mathbb{Z}_{n/2}} \left(f_j \mp f_{j+n/2}\right)\, x^j \tag{4.11}$$

mit jeweils $n/2$ Operationen in R zu berechnen. Setzen wir nun die j–ten Potenzen von ω in f ein und unterscheiden wir zwischen geraden ($j = 2k$) und ungeraden ($j = 2k+1$) Werten von j, dann erhalten wir, daß

$$\begin{array}{rcl} f\left(\omega^{2j}\right) & = & \overbrace{\left(\omega^{nj} - 1\right)}^{=0} g\left(\omega^{2j}\right) + r_-\left(\omega^{2j}\right) = r_-\left(\omega^{2j}\right) \\ f\left(\omega^{2j+1}\right) & = & \underbrace{\left(\omega^{nj}\omega^{n/2} + 1\right)}_{=\omega^{n/2}+1=0} g\left(\omega^{2j+1}\right) + r_+\left(\omega^{2j+1}\right) = r_+\left(\omega^{2j+1}\right). \end{array}$$

Im zweiten Fall haben wir berücksichtigt, daß ω eine *primitive* $n = 2^\ell$–te Einheitswurzel ist, also ist $\omega^{n/2} - 1$ kein Nullteiler in R und daher haben wir, daß

$$0 = \omega^n - 1 = \left(\omega^{n/2} - 1\right)\left(\omega^{n/2} + 1\right) \qquad \Longrightarrow \qquad \omega^{n/2} + 1 = 0.$$

Als nächstes machen wir noch eine einfache aber nützliche Bemerkung. Notwendig ist diese Aussage weil wir nicht vergessen dürfen, daß zwischen dem Grad n des Polynoms und der Ordnung n der primitiven Einheitswurzel eine Beziehung besteht, bestehen muß, denn n ist ja in beiden Fällen *dasselbe* Symbol und damit auch dieselbe Zahl! Und wenn wir jetzt mit $r_\pm$ zwei Polynome vom Grad $n/2$ haben, dann dürfen wir auch nur Einheitswurzeln der Ordnung $n/2$ für die DFT verwenden, denn sonst ist unsere schöne Rekursion zum Teufel.

Lemma 4.10 *Sei $n = 2^\ell$ und ω eine primitive n–te Einheitswurzel in R. Dann ist ω^2 eine primitive $(n/2)$–te Einheitswurzel in R.*

Beweis: Da $2^\ell = n \in R^*$ ist auch $(n/2)^{-1} = 2n^{-1} \in R$, also $n/2 \in R^*$. Außerdem ist 2 der einzige echte Primteiler[98] von n und $n/2$ und nach Voraussetzung ist

$$\left(\omega^2\right)^{(n/2)/2} - 1 = \left(\omega^2\right)^{n/4} - 1 = \omega^{n/2} - 1$$

[98] Es sei denn, wir hätten $n = 2$, aber dann ist es noch einfacher!

kein Nullteiler in R. Daß $(\omega^2)^{n/2} = \omega^n = 1$ ist, bedarf ja keiner gesonderten Erwähnung mehr[99]. □

Damit haben wir auch schon die Hälfte der Arbeit delegiert: Die Werte $f(\omega^{2j})$, $j \in \mathbb{Z}_{n/2}$, erhalten wir, indem wir (wieder rekursiv) die Auswertung für $r_-(\omega^{2j})$ aufrufen. Mit den ungeraden Werte $2j+1$, $j \in \mathbb{Z}_{n/2}$ ist das aber auch nicht viel schlimmer, da

$$r_+\left(\omega^{2j+1}\right) = \sum_{k\in\mathbb{Z}_{n/2}} \left(f_k - f_{k+n/2}\right)\omega^{k(2j+1)} = \sum_{k\in\mathbb{Z}_{n/2}} \left(\left(f_k - f_{k+n/2}\right)\omega^k\right)\omega^{2jk} = r_+^{\omega}\left(\omega^{2j}\right)$$

ist und so auch wieder rekursiv aufgerufen werden kann. Und das führt auch schon zur *schnellen Fouriertransformation* (FFT).

Algorithmus 4.11 *(FFT)*
Gegeben: $n = 2^\ell$, *primitive* n*–te Einheitswurzel* $\omega \in R$ *und* $f \in \Pi_{n-1}$.

1. *Berechne*

$$r(x) \leftarrow \sum_{j\in\mathbb{Z}_{n/2}} \left(f_j + f_{j+n/2}\right)x^j \qquad \textit{und} \qquad r_\omega(x) \leftarrow \sum_{j\in\mathbb{Z}_{n/2}} \omega^j\left(f_j - f_{j+n/2}\right)x^j,$$

   bzw.[100]

$$r \leftarrow \left[f_j + f_{j+n/2} \;:\; j \in \mathbb{Z}_{n/2}\right] \qquad \textit{und} \qquad r_\omega \leftarrow \left[\omega^j\left(f_j - f_{j+n/2}\right) \;:\; j \in \mathbb{Z}_{n/2}\right].$$

2. *Berechne rekursiv*

$$a \leftarrow r^\wedge\left(\omega^2\right) \qquad \textit{und} \qquad b \leftarrow r_\omega^\wedge\left(\omega^2\right)$$

3. *"Mische" die Vektoren*

$$c \leftarrow \left[a_0, b_0, a_1, b_1, \ldots, a_{n/2-1}, b_{n/2-1}\right]$$

Ergebnis: $c = f^\wedge(\omega)$.

Bleibt also noch die Frage nach dem Rechenaufwand $F(n)$ des Verfahrens. Nun, im $n$ten Schritt brauchen wir,

1. zur Berechnung der Koeffizienten von r: $\frac{n}{2}$ Operationen,
2. zur Berechnung der Koeffizienten von r_ω: $2\frac{n}{2} = n$ Operationen,
3. zur Berechnung von ω^2: 1 Operation,
4. für die Rekursion: $2F(n/2)$ Operationen,

[99] Weswegen wir es an dieser Stelle ja auch nicht erwähnen ...
[100] So würde man es wohl bei einer "praktischen" Implementierung machen.

alles natürlich bezüglich Rechnung im Ring R. Damit ist also[101]

$$F(n) = 2\,F\left(\frac{n}{2}\right) + \frac{3}{2}n + 1, \quad n > 1, \qquad F(1) = 0. \tag{4.12}$$

Zum Rechnen wird aber die Abschätzung

$$F(n) \leq 2\left(F\left(\frac{n}{2}\right) + n\right), \quad n > 1, \qquad F(1) = 1, \tag{4.13}$$

etwas angenehmer sein, denn nun erhalten wir[102]

$$\begin{aligned} F(n) &= F\left(2^\ell\right) \leq 2F\left(2^{\ell-1}\right) + 2^{\ell+1} \leq 4F\left(2^{\ell-2}\right) + 2^{\ell+1} + 2^{\ell+1} \leq \cdots \leq 2^\ell F(1) + \ell 2^{\ell+1} \\ &= 2^{\log_2 n} + 2\log_2 n 2^{\log_2 n} = n + 2n\,\log_2 n. \end{aligned}$$

Mit etwas mehr Sorgfalt[103] kommt man schließlich auf die Aufwandsabschätzung

$$F(n) = \frac{3}{2}\,n\,\log_2 n + O(n) = O(n\,\log_2 n) \tag{4.14}$$

Operationen in R für die Berechnung der FFT.

Nun sind DFT und FFT ja sicher schöne Sachen, aber was haben sie bitte mit unserem Ausgangsproblem, nämlich der (hoffentlich schnellen) *Multiplikation* zweier Polynome zu tun. In der Tat eine ganze Menge, wenn man die folgenden Begriffe in Betracht zieht.

Definition 4.12 *(Faltungen)*

1. *Seien $f \in R^m$, $g \in R^n$. Die* Faltung *$f * g \in R^{m+n}$ ist definiert als*

$$(f * g)_j := \sum_{k+\ell=j} f_k\,g_\ell = \sum_{k\in\mathbb{Z}_m} f_k\,g_{j-k} = \sum_{\ell\in\mathbb{Z}_n} f_{j-\ell}g_\ell, \qquad j \in \mathbb{Z}_{n+m}$$

   wobei $f_j = g_j = 0$ gesetzt wird, wann immer j ein unzulässiger Index ist[104].

2. *Seien $f, g \in R^n$. Die* Faltung modulo n, *$f *_n g \in R^n$ ist definiert als*

$$f *_n g = \left[(f * g)_j + (f * g)_{j+n} \;:\; j \in \mathbb{Z}_n\right].$$

   Allgemein ist

$$f *_n g = \left[\sum_{k+\ell\equiv_n j} f_k\,g_\ell \;:\; j \in \mathbb{Z}_n\right]. \tag{4.15}$$

Und die Faltung stellt nun auch die gewünschte Beziehung zwischen der Multiplikation von Polynomen und der DFT dar.

[101] Die Auswertung eines konstanten Polynoms an einer Stelle gibt's bekanntlich umsonst ...
[102] Mit praktisch derselben Rechnung wie bei Freund Karatsuba in (4.3).
[103] Und unter Ausnutzung von $F(1) = 0$.
[104] Also zu $\mathbb{Z} \setminus \mathbb{Z}_m$ bzw. $\mathbb{Z} \setminus \mathbb{Z}_n$ gehört.

Proposition 4.13 *Seien $f, g \in R^n$ und $\omega \in R$ eine primitive n–te Einheitswurzel*

1. *Es ist*

$$f(x) \cdot g(x) = (f * g)(x) \qquad \textit{und} \qquad f(x) \cdot g(x) \equiv_{x^n-1} (f *_n g)(x). \tag{4.16}$$

2. *Es ist*

$$(f *_n g)^\wedge(\omega) = f^\wedge(\omega) \odot g^\wedge(\omega), \tag{4.17}$$

wobei hier "$\odot$" die komponentenweise *Multiplikation der Vektoren bezeichnet.*

3. *Damit ist*

$$f *_n g = n^{-1} \left(f^\wedge(\omega) \odot g^\wedge(\omega)\right)^\wedge \left(\omega^{-1}\right). \tag{4.18}$$

Beweis: Da

$$\left(\sum_{j\in\mathbb{Z}_n} f_j x^j\right) \cdot \left(\sum_{k\in\mathbb{Z}_n} g_k x^k\right) = \sum_{j,k\in\mathbb{Z}_n} f_j g_k\, x^{j+k} = \sum_{\ell\in\mathbb{Z}_{2n}} \left(\sum_{j+k=\ell} f_j g_k\right) x^\ell = \sum_{\ell\in\mathbb{Z}_{2n}} (f*g)_\ell\, x^\ell,$$

folgt (4.16) unmittelbar. Demnach ist aber

$$(f *_n g)(x) = f(x) \cdot g(x) + q(x)\,(x^n - 1)$$

und somit

$$\begin{aligned} (f *_n g)^\wedge(\omega) &= \left[f\left(\omega^j\right) \cdot g\left(\omega^j\right) + q\left(\omega^j\right)\left(\omega^{jn} - 1\right) \;:\; j \in \mathbb{Z}_n\right] \\ &= \left[f\left(\omega^j\right) \cdot g\left(\omega^j\right) \;:\; j \in \mathbb{Z}_n\right] = f^\wedge(\omega) \odot g^\wedge(\omega), \end{aligned}$$

was uns (4.17) liefert. (4.18) folgt dann sofort aus (4.17) und (4.8). □

Ausgehend von (4.18) können wir nun ein Verfahren zur schnellen Multiplikation zweier Polynome modulo $x^n - 1$ angeben.

Algorithmus 4.14 *(Schnelle Multiplikation modulo $x^n - 1$)*
Gegeben: *$n = 2^\ell$, primitve n–te Einheitswurzel $\omega \in R$ und $f, g \in R[x]/\langle x^n - 1\rangle$.*

1. *Berechne mittels der FFT, Algorithmus 4.11,*

$$a \leftarrow f^\wedge(\omega) \qquad \textit{und} \qquad b \leftarrow g^\wedge(\omega)$$

2. *Berechne das komponentenweise Produkt*

$$c \leftarrow a \odot b$$

3. *Berechne, wieder mit Algorithmus 4.11,*

$$d \leftarrow n^{-1} c^{\wedge}\left(\omega^{-1}\right).$$

Ergebnis: $d(x) = f(x) \cdot g(x)$.

Der Aufwand ist jetzt wieder moderat: Wir haben in Schritt 1. 2 FFTs zu berechnen, also $O\left(n \log_2 n\right)$ Operationen in R, in Schritt 2. n Multiplikationen in R und in Schritt 3. nochmal eine DFT und n Multiplikationen, also insgesamt wieder ein Gesamtaufwand von $O\left(n \log_2 n\right)$ Operationen in R.
Um nun zwei Polynome $f, g \in \Pi_{n-1}$ miteinander zu multiplizieren, verwenden wir wieder die Zerlegung

$$f(x) = f_1(x)\, x^{n/2} + f_0(x), \qquad g(x) = g_1(x)\, x^{n/2} + g_0(x), \qquad f_0, f_1, g_0, g_1 \in \Pi_{n/2-1},$$

führen die drei[105] Multiplikationen

$$f_1 \cdot g_1, \qquad f_0 \cdot g_0, \qquad (f_0 + f_1) \cdot (g_0 + g_1)$$

mit Algorithmus 4.14 aus und bauen die Ergebnisse, die ja nur Grad $n-1$ haben und somit jetzt *exakt* sind, dann wieder passend zusammen.

Übung 4.2 Formulieren Sie das schnelle Multiplikationsverfahren für zwei Polynome und weisen Sie nach, daß sein Aufwand wieder $O\left(n \log_2 n\right)$ ist.

Es gibt im übrigen einen Ring, in dem die FFT immer funktioniert, weil er 2^ℓ–te Einheitswurzeln für beliebiges $\ell \in \mathbb{N}$ besitzt, nämlich die komplexen Zahlen $\mathbb{C}$. Und dort, bzw. für trigonometrische Polynome als Partialsummen von komplexen Fourierreihen, wurde auch die FFT ursprünglich in (Cooley & Tukey, 1965) eingeführt[106] bzw. wiederentdeckt, siehe (Cooley, 1987; Cooley, 1990). Für allgemeine Ringe, insbesondere für endliche Restklassenringe, aber auch für endliche Körper, ist die Sache mit der Existenz der 2^ℓ–ten Einheitswurzeln natürlich wesentlich komplizierter und unangenehmer.

4.3 Schnelle Polynommultiplikation III – Root it yourself

Die schnelle Polynommultiplikation via DFT und FFT à la Algorithmus 4.14 ist ja schön und gut, funktioniert aber halt eben nicht in allen Ringen, denn sie setzt ja die Existenz einer n–ten primitiven Einheitswurzel voraus, wobei $n = 2^\ell$ auch noch eine Zweierpotenz sein muß. Das ist aber bereits eine Forderung, die alle Ringe der Form $\mathbb{Z}/\langle 2k\rangle$, $k \in \mathbb{N}$, von vornherein ausschließt. Das heißt, wir beschränken uns schon mal auf Ringe R, in denen 2 eine Einheit ist – davon gibt's ja auch noch genug. Und wenn's in so einem Ring keine n–te Einheitswurzel gibt, naja, dann basteln wir uns halt einfach eine; vornehmer: Wir *adjungieren* eine geeignete Einheitswurzel.

[105] Schließlich müssen wir ja nicht alles vergessen, was wir an "guten" Dingen im Abschnitt 4.1 gelernt haben.
[106] Auf ganzen vier Seiten!

Lemma 4.15 *Sei R ein kommutativer Ring mit 1, so daß $2 \in R^*$ und sei $n = 2^\ell$, $\ell \geq 1$. Dann ist in dem Ring $R_n := R[x]/\langle x^n + 1\rangle \supset R$ das Element $\omega = \omega(x) = x$ eine primitive $(2n)$te Einheitswurzel.*

Beweis: Da $x^n \equiv_{x^n+1} -1$, ist $\omega(x) = x$ eine $2n$te Einheitswurzel. Da außerdem $2 \in R^* \subseteq R_n^*$ der einzige Primteiler von $2n = 2^{\ell+1} \in R^* \subseteq R_n^*$ ist, ergibt

$$\omega^{2n/2}(x) - 1 = x^n - 1 \equiv_{x^n+1} -1 - 1 = -2 \in R_n^*,$$

daß ω tatsächlich eine *primitive* $2n$te Einheitswurzel ist. □

Bemerkung 4.16 *Der Ring R_n aus Lemma 4.15 ist normalerweise* kein *Körper*[107]*, nicht einmal wenn R ein Körper ist. Ein einfaches Beispiel ist $R = \mathbb{C}$ und $n = 2$, denn dann ist $x^2 + 1 = (x + i)(x - i)$ reduzibel.*

Die gute Nachricht von Lemma 4.15 ist also, daß wir die Existenz der gesuchten Einheitswurzel in jedem Ring erzwingen können, die schlechte Nachricht ist hingegen, daß wir in einem *polynomialen* Restklassenring rechnen müssen, was zu ein paar Detailproblemen führt, die wir uns jetzt ansehen wollen.

Seien nun also $f, g \in R[x]$ zwei Polynome, so daß $\deg(f \cdot g) < n = 2^\ell$. Das ist keine echte Einschränkung, denn so ein n kann man immer bestimmen. Und da $f \cdot g = (f \cdot g)_{x^n+1}$, genügt es natürlich, modulo $x^n + 1$ zu rechnen.
Wir setzen

$$m = 2^{\lfloor \ell/2 \rfloor} \qquad \text{und} \qquad m' = \frac{n}{m} = 2^{\lceil \ell/2 \rceil} \qquad \Longrightarrow \qquad n = m\, m';$$

diese beiden Zahlen m, m' sind die "Wurzeln" von n und es gilt

$$m' = \begin{cases} m, & \ell \in 2\,\mathbb{N}, \\ 2m, & \ell \in 2\,\mathbb{N} + 1 \end{cases} \tag{4.19}$$

Nun zerlegen wir f und g in polynomiale "Blöcke" der Länge m:

$$f(x) = \sum_{j \in \mathbb{Z}_{m'}} f_j(x)\, x^{mj} \qquad g(x) = \sum_{j \in \mathbb{Z}_{m'}} g_j(x)\, x^{mj}, \qquad f_j, g_j \in \Pi_{m-1}, \quad j \in \mathbb{Z}_{m'}. \tag{4.20}$$

Anders gesagt: Die *Anzahl* der Blöcke und deren *Länge* stimmen so gut es geht überein – bis eben auf diesen unvermeidbaren Faktor 2, der auftritt, wenn ℓ ungerade ist. Damit definieren wir künstlich *bivariate* Polynome $F, G \in R[x, y]$, indem wir

$$F(x, y) = \sum_{j \in \mathbb{Z}_{m'}} f_j(x)\, y^j \qquad \text{und} \qquad G(x, y) = \sum_{j \in \mathbb{Z}_{m'}} g_j(x)\, y^j$$

[107] So schlimm ist das aber nicht! Für unsere FFT und deren Invertierung brauchen wir ja nicht, daß alle Elemente des Rings, in dem wir rechnen, invertierbar sind, sondern nur, daß unserer "magische" Einheitswurzel invertierbar ist, und die ist es nach Konstruktion.

setzen, und erhalten, daß

$$f(x) = F\left(x, x^m\right), \qquad \text{und} \qquad g(x) = G\left(x, x^m\right).$$

Das bivariate Produkt $F \cdot G$ brauchen wir nun "nur" in $R[x,y]/\left\langle y^{m'}+1\right\rangle$ zu berechnen, denn aus der "Division–mit–Rest–Zerlegung" $F(x,y)\cdot G(x,y) = H(x,y)+Q(x,y)\left(y^{m'}+1\right)$ folgt, daß

$$\begin{aligned} f(x)\cdot g(x) \quad &= \quad F\left(x,x^m\right)\cdot G\left(x,x^m\right) = H\left(x,x^m\right) + Q\left(x,x^m\right)\underbrace{\left(x^{mm'}+1\right)}_{=x^n+1} \\ &\equiv_{x^n+1} \quad H\left(x,x^m\right). \end{aligned}$$

Bemerkung 4.17 *Für unsere Ausgangspolynome f, g, deren Produkt ja Grad $< n$ hat, ist diese modulare Rechnung rein formal. Beim* rekursiven *Aufruf dieser Multiplikationsroutine wird aber ganz gezielt und explizit modulo x^n+1 gerechnet und dann macht die Sache Sinn!*

Durch die zusätzliche Einschränkung $\deg_y H(x,y) < m'$ wählen wir somit einen eindeutigen Repräsentanten aus der Äquivalenzklasse $[H(x,y)]$ modulo $y^{m'}+1$ aus, für den außerdem

$$\deg_x H(x,y) \le \max_{j,k\in\mathbb{Z}_{m'}} \deg f_j\cdot g_k = \max_{j,k\in\mathbb{Z}_{m'}} \underbrace{\deg f_j}_{<m} + \underbrace{\deg g_k}_{<m} < 2m-1$$

gilt – schließlich ergibt sich ja der Koeffizient zu y^ℓ als

$$\sum_{j+k=\ell} f_j(x)\, g_k(x)$$

und der Grad der Summe von Polynomen ist $\le$ dem Grad der einzelnen Summanden[108]. Mit

$$H^*(x,y) = \left(H(x,y)\right)_{x^{2m}+1},$$

erhalten wir somit, daß

$$F(y)\cdot G(y) \equiv_{y^{m'}+1} H^*(y), \qquad \text{in} \qquad R_{2m}[y] = \left(R[x]/\left\langle x^{2m}+1\right\rangle\right)[y]. \tag{4.21}$$

Wenden wir schließlich Lemma 4.15 auf die Zweierpotenz $2m$ an, dann erhalten wir eine primitive Einheitswurzel $\eta(x) = x$ der Ordnung $4m$ in $R_{2m} = R[x]/\langle x^{2m}+1\rangle$. Wegen (4.19) enthält R_{2m} auch eine $(2m')$–te primitive Einheitswurzel ξ und zwar entweder $\xi = \eta^2$ oder $\xi = \eta$, je nachdem, ob ℓ gerade oder ungerade ist. Ersetzen wir y durch ξy in (4.21), was möglich ist, da $\xi \in R_{2m}^*$, dann erhalten wir die äquivalente Formulierung

$$F\left(\xi y\right)\cdot G\left(\xi y\right) \equiv_{(\xi y)^{m'}+1} H^*\left(\xi y\right) \tag{4.22}$$

[108] Der Fall "$<$" kann natürlich eintreten, und zwar, wenn sich die Leitterme wegheben.

was wir wegen $\xi^{m'} = -1$, also

$$(\xi y)^{m'} + 1 = -y^{m'} + 1 = -\left(y^{m'} - 1\right),$$

auch als

$$F(\xi y) \cdot G(\xi y) \equiv_{y^{m'}-1} H^*(\xi y) \tag{4.23}$$

schreiben können.
Doch nun sind wir im Geschäft, denn in R_{2m} sind jetzt alle Voraussetzungen für eine schnelle Multiplikation modulo $y^{m'} - 1$ erfüllt! Denn da ξ eine primitive $(2m')$–te Einheitswurzel in R_{2m} ist, ist auch $\omega = \xi^2$ eine m'–te Einheitswurzel in R_{2m} – und das ist die Grundvoraussetzung für eine FFT-Multiplikation.

Algorithmus 4.18 *(Schnelle Polynommultiplikation in $R_n = R[x]/\langle x^n + 1\rangle$)*
Gegeben: *R kommutativer Ring mit 1 so daß $2 \in R^*$, $n = 2^\ell$, $\ell > 2$, und $f, g \in \Pi_{n-1}$.*

1. *Setze*
$$m \leftarrow 2^{\lfloor \ell/2 \rfloor}, \qquad m' \leftarrow n/m.$$

2. *Setze (in R_{2m})*
$$\xi = \xi(x) \leftarrow \begin{cases} x^2, & \ell \in 2\mathbb{N}, \\ x, & \ell \in 2\mathbb{N}+1. \end{cases}$$

3. *Für $j \in \mathbb{Z}_{m'}$ setze*
$$f_j(x) \leftarrow \sum_{k \in \mathbb{Z}_m} f_{mj+k}\, x^k, \qquad g_j(x) \leftarrow \sum_{k \in \mathbb{Z}_m} g_{mj+k}\, x^k$$
und dann
$$F(x,y) = \sum_{j \in \mathbb{Z}_{m'}} f_j(x)\, y^j, \qquad G(x,y) = \sum_{j \in \mathbb{Z}_{m'}} g_j(x)\, y^j.$$

4. *Berechne mit der schnellen Polynommultiplikation, Algorithmus 4.14 (unter Verwendung der m'ten primitiven Einheitswurzel $\omega = \xi^2$ in R_{2m}), das Produkt*
$$H^*(y) \leftarrow \left(F(\xi y) \cdot G(\xi y)\right)_{y^{m'}-1}, \qquad F, G, H^* \in R_{2m}[y],$$
und verwende Algorithmus 4.18 rekursiv für die Rechnung im Ring R_{2m}.

5. *Setze*
$$H(x,y) \leftarrow H^*\left(x, \xi^{-1} y\right).$$

6. *Setze*
$$h(x) \leftarrow \left(H(x, x^m)\right)_{x^n+1}$$

Ergebnis: $h(x) \equiv_{x^n+1} f(x) \cdot g(x)$.

Wenn man nun eine sorgfältige Analyse des Rechenaufwands betreibt, dann erhält man (Gathen & Gerhard, 1999, Theorem 8.22, S. 228)[109], daß man mit

$$\frac{9}{2}\, n\, \log_2 n\, \log_2 \log_2 n + O\left(n\, \log_2 n\right)$$

Operationen in R auskommt, das "implizite" Adjungieren der primitiven $2n$ten Wurzel x in R_n kostet also lediglich den multiplikativen Faktor $\log_2 \log_2 n$.

Um uns klarzumachen, mit was für einer Sorte Algorithmus wir es da eigentlich zu tun haben, schauen wir uns jetzt einmal ein langes und explizites Beispiel an.

Beispiel 4.19 *Multiplizieren wir doch einmal in $\mathbb{F}_5$ die beiden Polynome*

$$f(x) = x^4 + 2x + 3, \qquad g(x) = 2x^3 + x^2 + 4x + 2.$$

Da $\deg fg = 7$ wählen wir $n = 8$, also $\ell = 3$. Damit liefern also die einzelnen Schritte von Algorithmus 4.18 die folgenden Resultate:

1. $m = 2$, $m' = 4$,
2. $\xi(x) = x$, *da* $\ell = 3 \in 2\mathbb{N} + 1$.
3. *Damit erhalten wir die Zerlegungen*

$$\begin{array}{lcllcllcl} f_0(x) & = & 2x + 3, & f_1(x) & = & 0, & f_2(x) & = & 1, \\ g_0(x) & = & 4x + 2, & g_1(x) & = & 2x + 1 & g_2(x) & = & 0, \end{array}$$

   sowie $f_3 = g_3 = 0$ und somit

$$\begin{aligned} F(x, y) &= y^2 + 2x + 3, \\ G(x, y) &= (2x + 1)\, y + 4x + 2. \end{aligned}$$

4. *Jetzt geht also los mit der FFT! Mit der m'ten primitiven Einheitswurzel $\omega = \xi^2$, also $\omega(x) = x^2$, erhalten wir, daß*

$$\begin{aligned} F_*^{\wedge}(\omega) &:= F\left(\xi\cdot\right)^{\wedge}(\omega) = \left[F\left(\xi \cdot \omega^j\right) \,:\, j \in \mathbb{Z}_{m'}\right] = \left[F\left(\xi\, \xi^{2j}\right) \,:\, j = 0, \ldots, 3\right] \\ &= \left[F\left(x, x^{2j+1}\right) \,:\, j = 0, \ldots, 3\right], \end{aligned}$$

   und analog für G_*. Diese Vektoren können wir explizit[110] *als*

$$F_*^{\wedge}(\omega) = \begin{bmatrix} x^2 + 2x + 3 \\ x^6 + 2x + 3 \\ x^{10} + 2x + 3 \\ x^{14} + 2x + 3 \end{bmatrix} = \begin{bmatrix} x^2 + 2x + 3 \\ 4x^2 + 2x + 3 \\ x^2 + 2x + 3 \\ 4x^2 + 2x + 3 \end{bmatrix}$$

[109] Ist es eigentlich wahrscheinlich, daß in einem mathematischen Buch die Nummer eines Satzes mit der rückwärtsgelesenen Seitenzahl übereinstimmt?

[110] Unter Verwendung von $x^4 \equiv -1 \equiv 4$ im Ring $\mathbb{F}_5[x]/\langle x^4 + 1\rangle$. Die "richtige" FFT spielen wir später auch noch einmal durch.

bzw.

$$G_*^{\wedge}(\omega) = \begin{bmatrix} 2x^2+x+4x+2 \\ 2x^4+x^3+4x+2 \\ 2x^6+x^6+4x+2 \\ 2x^8+x^7+4x+2 \end{bmatrix} = \begin{bmatrix} 2x^2+2 \\ x^3+4x \\ 3x^2+3x+2 \\ 4x^3+4x+4 \end{bmatrix}$$

angeben. Die komponentenweise Multiplikation[111] *in* $\mathbb{F}_5[x]/\langle x^4+1\rangle$ *liefert dann, daß*

$$F_*^{\wedge}(\omega) \cdot G_*^{\wedge}(\omega) = \begin{bmatrix} 4x^3+3x^2+4x+4 \\ 4x^3+3x^2+3x+3 \\ 4x^3+2x^2+3x+3 \\ 3x^3+4x^2+4x+4 \end{bmatrix} = (H^*)^{\wedge}(\omega) =: h^*. \tag{4.24}$$

Die Koeffizienten von H^* *erhalten wir dann aus der inversen Fouriertransformation der Koeffizienten von* h*, also als*

$$4^{-1}\,(h^*)^{\wedge}\left(\omega^{-1}\right) \qquad \textit{da} \qquad \omega^{-1} = \left(x^2\right)^{-1} \equiv -x^2 \equiv 4x^2, \quad 4^{-1} \equiv 4.$$

Hier machen wir uns das Leben etwas leichter und verwenden die FFT[112] *mit* ω^{-1}*. In der Tat erhalten wir aus* h *die beiden Vektoren*[113]

$$\begin{aligned} r_- &= \begin{bmatrix} h_0^*+h_2^* \\ h_1^*+h_3^* \end{bmatrix} = \begin{bmatrix} 3x^3+2x+2 \\ 2x^3+2x^2+2x+2 \end{bmatrix}, \\ r_+ &= \underbrace{\begin{bmatrix} 1 \\ 4x^2 \end{bmatrix}}_{=[\omega^{-0},\omega^{-1}]^T} \odot \begin{bmatrix} h_0^*-h_2^* \\ h_1^*-h_3^* \end{bmatrix} = \begin{bmatrix} 1 \\ 4x^2 \end{bmatrix} \odot \begin{bmatrix} x^2+x+1 \\ x^3+4x^2+4x+4 \end{bmatrix} \\ &= \begin{bmatrix} x^2+x+1 \\ x^3+x^2+x+4 \end{bmatrix}, \end{aligned}$$

die wir bei der Berechnung von $r_-^{\wedge}(\omega^{-2})$ *und von* $r_+^{\wedge}(\omega^{-2})$ *nochmals aufspalten in*

$$\begin{aligned} r_{--} &= 2x^2+4x+4, & r_{-+} &= \underbrace{\left(\omega^{-2}\right)^0}_{=1}(x^3+3x^2) = x^3+3x^2, \\ r_{+-} &= x^3+2x^2+2x, & r_{++} &= \underbrace{\left(\omega^{-2}\right)^0}_{=1}(4x^3+2) = 4x^3+2. \end{aligned}$$

Nach dem erforderlichen Mischen erhalten wir somit, daß

$$H^* = 4\begin{bmatrix} r_{--} \\ r_{+-} \\ r_{-+} \\ r_{++} \end{bmatrix} = 4\begin{bmatrix} 2x^2+4x+4 \\ x^3+2x^2+2x \\ x^3+3x^2 \\ 4x^3+2 \end{bmatrix} = \begin{bmatrix} 3x^2+x+1 \\ 4x^3+3x^2+3x \\ 4x^3+2x^2 \\ x^3+3 \end{bmatrix},$$

[111] Hier erfolgt nun der rekursive Aufruf, siehe Beispiel 4.20.

[112] Und bei dieser Gelegenheit sieht man dann auch gleich, wie die FFT wirklich funktioniert.

[113] Natürlich immer unter Verwendung von $-1 \equiv_5 4$.

also

$$H^*(y) = \left(2x^4 + x^3\right) y^3 + \left(4x^3 + 2x^2\right) y^2 + \left(4x^3 + 3x^2 + 3x\right) y + \left(3x^2 + x + 1\right),$$

5. *und daher*

$$\begin{aligned} H(x,y) &= H^*\left(x, \xi^{-1} y\right) = H^*\left(x, x^{-1} y\right) \\ &= \left(2x^4 + x^3\right) x^{-3} y^3 + \left(4x^3 + 2x^2\right) x^{-2} y^2 + \left(4x^3 + 3x^2 + 3x\right) x^{-1} y \\ &\quad + \left(3x^2 + x + 1\right) \\ &= (2x+1) y^3 + (4x+2) y^2 + \left(4x^2 + 3x + 3\right) y + \left(3x^2 + x + 1\right). \end{aligned}$$

6. *Somit erhalten wir*

$$\begin{aligned} h(x) &\equiv_{x^n+1} H\left(x, x^2\right) \\ &= (2x+1) x^6 + (4x+2) x^4 + \left(4x^2 + 3x + 3\right) x^2 + \left(3x^2 + x + 1\right) \\ &= 2x^7 + x^6 + 4x^5 + x^4 + 3x^3 + x^2 + x + 1, \end{aligned}$$

was man "zu Fuß" mit

$$\left(x^4 + 2x + 3\right) \cdot \left(2x^3 + x^2 + 4x + 2\right)$$

auch bekommen hätte.

Beispiel 4.20 *Als Beispiel für die rekursive Berechnung in Schritt 4 aus Beispiel 4.19 berechnen wir nun*

$$\Big(\underbrace{\left(4x^2 + 2x + 3\right)}_{=f(x)} \cdot \underbrace{\left(4x^3 + 4x + 4\right)}_{=g(x)}\Big)_{x^4+1},$$

natürlich immer noch in $\mathbb{F}_5$*. Wir haben also jetzt* $n = 4$ *und damit* $\ell = 2$ festgelegt*, sie werden nicht mehr aus dem Grad des Produkts ermittelt*[114]

1. $m = m' = 2$.
2. $\xi(x) = x^2$.
3. *Die Zerlegungen sind jetzt*

$$\begin{array}{rclcrclcrcl} f_0(x) &=& 2x+3, & & f_1(x) &=& 4, & \Longrightarrow & F(x,y) &=& 4y + 2x + 3, \\ g_0(x) &=& 4x+4, & & g_1(x) &=& 4x, & \Longrightarrow & G(x,y) &=& 4xy + 4x + 4. \end{array}$$

4. *Jetzt bestimmen wir, wieder in* $R_{2m} = \mathbb{F}_5[x]/\langle x^4 + 1\rangle$ *das Produkt*

$$\begin{aligned} H^*(y) &= \left(F(\xi y) \cdot G(\xi y)\right)_{y^2-1} = \left(F\left(x, x^2 y\right) \cdot G\left(x, x^2 y\right)\right)_{y^2-1} \\ &= \left(\left(4x^2 y + 2x + 3\right) \cdot \left(4x^3 y + 4x + 4\right)\right)_{y^2-1} \\ &= \left(4xy^2 + \left(x^3 + x^2 + 2 + 2x^3\right) y + 3x^2 + 2\right)_{y^2-1} \\ &= \left(3x^3 + x^2 + 2\right) y + 3x^2 + 4x + 2. \end{aligned}$$

[114] Sonst wäre der ganze Geschwindigkeitsvorteil zum Teufel.

5. *Also ist*[115]

$$\begin{aligned} H(x,y) &= H^*\left(x,\xi^{-1}y\right) = H^*\left(x,4x^2\,y\right) = \left(3x^3+x^2+2\right)4x^2\,y+3x^2+4x+2 \\ &= \left(3x^2+3x+1\right)y+3x^2+4x+2, \end{aligned}$$

6. *was uns schließlich*[116] *unser Endergebnis*

$$\begin{aligned} h(x) &= H\left(x,x^2\right) = \left(3x^2+3x+1\right)x^2+3x^2+4x+2 \\ &= 2+3x^3+x^2+3x^2+4x+2 = 3x^3+4x^2+4x+4 \end{aligned}$$

liefert.

Und genau dieses Polynom steht ja auch in der letzten Zeile des "Ergebnisvektors" in (4.24), wo es allerdings "von Hand" und nicht rekursiv berechnet wurde.

Die einzige Einschränkung an den Ring R bestand nun darin, daß 2 eine Einheit in R sein mußte – das lag aber im wesentlichen daran, daß wir in jedem Schritt der FFT in Polynom in zwei identische Polynome aufgespalten haben; und was man mit 2 machen kann, kann man zumeist auch mit beliebigen anderen Zahlen, insbesondere mit 3 machen. Und tatsächlich gibt es auch eine 3adische FFT (Gathen & Gerhard, 1999, Exercise 8.26, S. 239–240) und eine 3adische Version der schnellen Polynommultiplikation (Gathen & Gerhard, 1999, Exercise 8.30, S. 240), die auf Schönhage (Schönhage, 1977) zurückgeht. Und damit kann man schon eine ganze Menge erreichen, nämlich beispielsweise alle Ringe der Form $\mathbb{Z}/\left\langle p^m\right\rangle$, wobei p eine Primzahl ist[117]; am wichtigsten ist natürlich der Fall $p=2$, denn diese Ringe sind die "natürlichen" Ganzzahlbereiche auf Rechnern.

4.4 Schnelle Ganzzahlmultiplikation – Schönhage und Strassen

Jetzt können wir schließlich zu unserer schnellen Multiplikation von ganzen Zahlen zurückkommen und diese auf die schnelle Polynommultiplikation zurückführen, indem wir eine Multiprecision–Zahl

$$p=\sum_{j\in\mathbb{Z}_N} p_j B^j$$

als Wert des Polynoms $p(x)\in\mathbb{Z}[x]/\left\langle x^N-1\right\rangle$ an der Stelle $x=B$ auffassen. Anders gesagt,

$$p\otimes q = \left.\left(p(x)\cdot q(x)\right)\right|_{x=B}.$$

Sind nun p,q zwei Multiprecision–Zahlen der Länge N, dann sind $p(x)$ und $q(x)$ zwei Polynome vom Grad N mit Koeffizienten in $\mathbb{Z}_B$, die wir nun dank Algorithmus 4.18 mit einem Aufwand von $O\left(N\,\log_2 N\,\log_2\log_2 N\right)$ Ringoperationen berechnen könnten[118], wenn $2\in\mathbb{Z}^*$

[115] Immer noch in $R_{2m}=\mathbb{F}_5[x]/\left\langle x^4+1\right\rangle$.

[116] Auch auf die Gefahr, mich zu wiederholen: die Rechnungen erfolgen immer noch in $\mathbb{F}_5[x]/\left\langle x^4+1\right\rangle$.

[117] Ist $p=2$, wählt man die triadische Form, ansonsten die dyadische.

[118] Man beachte den Konjunktiv!

wäre. Die *Auswertung* dieses Polynoms ist eine vergleichsweise simple Geschichte, denn die Multiplikationen im *Hornerschema*

$$\begin{aligned} h_0 &\leftarrow p_N, \\ h_j &\leftarrow p_{N-j} + h_{j-1} \cdot B, \qquad j = 1, \ldots, N, \\ p(B) &= h_N, \end{aligned}$$

sind ja nur Shifts! Außerdem müssen wir bei der Addition berücksichtigen, daß die Koeffizienten[119]

$$(p \cdot q)_j = \sum_{k+\ell=j} p_j q_k \leq N B^2 \leq B^3, \qquad j \in \mathbb{Z}_{N+1},$$

des Produkts $p \cdot q$ aus maximal drei Worten bestehen, und wir somit in der Tat mit $O\left(N \log_2 N\right)$ Wortoperationen auskommen, da wir im Hornerschema N Additionen mit Zahlen der Länge $O\left(\log_2 N\right)$ durchzuführen haben.

Die Idee von Schönhage und Strassen (Schönhage & Strassen, 1971) besteht nun darin, daß man wieder zuerst einmal die Zahlen[120] $p, q \in \mathbb{M}_w$ als Polynome $p(x) \in \Pi_{\ell(p)}$ und $q(x) \in \Pi_{\ell(q)}$ auffasst. Dann wählt man $n = 2^\ell$ so, daß (wie gehabt) $\deg p \cdot q < n$ sowie $n > w$ gilt und interpretiert p, q als Polynome in $R[x]$, wobei $R = \mathbb{Z}/\langle 2^n + 1 \rangle$. Dieser Ring hat eine schöne Eigenschaft.

Lemma 4.21 *Sei $n = 2^\ell \in \mathbb{N}$. Dann ist $\omega = 2$ ist eine primitive $2n$–te Einheitswurzel in $R = \mathbb{Z}/\langle 2^n + 1 \rangle$.*

Definition 4.22 *Die Zahl $F_n = 2^{2^n} + 1$, $n \in \mathbb{N}$, heißt n–te* Fermat–Zahl[121]

Damit sind also alle Bedingungen für eine FFT–basierte schnelle Multiplikation mittels Algorithmus 4.14 gegeben und natürlich ist dank unserer Annahme $\deg p \cdot q < n$ das Ergebnis dann letztendlich auch korrekt. Die nächste Idee ist dann wieder wie bei der Herleitung von Algorithmus 4.18 eine Zerlegung von $n = 2^\ell$ in $n = m\,m'$, wobei $m' \in \{m, 2m\}$. Wir zerhacken unsere Zahlen der Länge n also wieder in m' Blöcke der Länge m und erhalten[122] die Koeffizienten des Ergebnisses als diskrete Faltung der Koeffizienten der Ausgangszahlen bzw. der Ausgangspolynome. Für diese diskrete Faltung verwendet man nun eine FFT, bei der Zahlen in etwa halber Länge miteinander multipliziert werden müssen, was man mit einem rekursiven Aufruf der Multiplikationsroutine löst.

Bemerkung 4.23 *(Schönhage & Strassen)*

[119] Zur Erinnerung: Jede Zahl hat maximal $2^w = B$ Ziffern, das war die Einschränkung, die keine ist, aus Definition 2.1.

[120] Zur Terminologie siehe Definition 2.1 auf S. 16.

[121] Wenn ich micht recht erinnere, dann hat Fermat vermutet, daß alle Zahlen dieser Form Primzahlen wären, denn $F_0 = 3$, $F_1 = 5$, $F_2 = 17$, $F_3 = 257$, $F_4 = 65537$, $F_5 = 4294967297 = 641 \cdot 6700417$ – dieses erste Gegenbeispiel stammt von Euler, der diese Faktorisierung *von Hand* bestimmt hat. Übrigens reichen 32–Bit Ganzzahlen für F_5 schon nicht mehr aus.

[122] Bis auf Übertrag, aber der bleibt ja, was den Aufwand angeht, im Rahmen, wie unsere Betrachtungen über die Auswertung von Polynomen an der Stelle $B = 2^w$ gezeigt haben.

1. *Das Verfahren von Schönhage & Strassen ist seit 1971 auf dem Markt und bis heute ist noch keine schnellere Methode bekannt! Auf der anderen Seite scheinen die FFT–basierten Verfahren noch nicht besonders in die kommerziellen Computeralgebra–Programme integriert zu sein. Das ist übrigens nicht so verwunderlich: Bevor man an dem sensiblen internen Kern herumspielt, fügt man lieber "sichtbare" Features wie Audio, Video und Animation hinzu – sowas verkauft sich nun einmal besser. Und ansonsten hofft man auf immer leistungsstärkere Rechner.*

2. *Diese Philosophie ist nachvollziehbar. Die Einführung komplett neuer Verfahren, auf denen ja immerhin das gesamte System aufsetzt, ist immer ein großes Risiko und führt normalerweise zu jeder Menge unerwarteter Fehler.*

3. *Laut (Gathen & Gerhard, 1999) war*[123] `Magma V 2.4`[124] *das einzige Computeralgebra–Programm, das FFT–basierte Verfahren zur Multiplikation von Polynomen und ganzen Zahlen verwendet. Inzwischen ist die FFT–Multiplikation von ganzen Zahlen zumindest auch in* `GinaC` *verfügbar und tatsächlich sind die Multiplikationsroutinen dort um einiges schneller als beispielsweise die in* `MuPAD`*. Das kann natürlich alles heute schon wieder ganz anders sein.*

4. *Es sieht generell so aus, als ob* $O\left(n \log_2 n\right)$ *eine natürliche Komplexitätsschranke für Probleme ist, zu denen es ein nichttriviales, naives Verfahren mit einem Aufwand von* $O\left(n^2\right)$ *gibt.*

5. *Überhaupt ist (Schönhage & Strassen, 1971) sehr lesenswert! Die Einleitung gibt auch die Geschichte der "Jagd" nach schnellen Multiplikationsverfahren wieder, die mit Karatsuba (Karatsuba & Ofman, 1963) beginnt und über Verfahren von Toom (Toom, 1963), Schönhage (Schönhage, 1966) bzw. Cook (Cook, 1966) mit mit einem O–Aufwand von jeweils*

Toom	*Schönhage*	*Cook*
$N2^{C\sqrt{\ln N}}$	$N2^{\sqrt{2\log_2 N}}(\log_2 N)^{3/2}$	$N2^{\sqrt{2\log_2 N}}\log_2 N$

   schließlich bei dem (bisher) optimalen Aufwand $O\left(N \log_2 N \log_2\log_2 N\right)$ *landet.*

6. *In (Schönhage & Strassen, 1971) wird auch darauf hingewiesen, daß Knuth unabhängig die Idee hatte, die FFT zur Multiplikation zweier ganzer Zahlen zu verwenden. Man sollte dabei bedenken, daß (Knuth, 1998), genauer gesagt, die erste Auflage von 1969 auch die Arbeit von Schönhagen und Strassen beeinflußt hat.*

7. *Einen Punkt sollte man aber weder vergessen noch unterschätzen: Das Verfahren von Schönhage und Strassen hat optimale* asymptotische *Komplexität, aber das sind eben nur Aussagen der Form* $O\left(\ldots\right)$*, und Verfahren, die sich oftmals erst ab einer bestimmten*

[123] Richtiger: Zum Zeitpunkt als dieses Buch geschrieben wurde.
[124] Was auch immer das ist.

Größe der Zahl lohnen, bei "kleinen" Zahlen sind oftmals der Aufwand und der "Overhead" für die Rekursion und die Aufbereitung der Daten zu groß. Tatsächlich scheint es so zu sein, daß in vielen Implementierungen für die letzten Schritte der Rekursion der einfachere und dann schnellere Karatsuba verwendet wird.

Beweis von Lemma 4.21: Daß $2 \in R^*$ und $2^n \equiv_{2^n+1} -1$ ist, ist für uns inzwischen ja schon fast Routine. Also ist 2 *immer* eine Einheitswurzel der Ordnung $2n$.
Ist nun $n = 2^\ell$, dann ist $2n = 2^{\ell+1}$ und der einzige Primteiler hiervon ist 2, das heißt, es genügt, einzusehen, daß

$$\omega^{2n/2} - 1 = \omega^n - 1 = 2^n - 1 = -1 - 1 = -2 \in R^*$$

kein Nullteiler ist. Bleibt also nur noch zu zeigen, daß $2n \in R^*$. Aber auch das ist einfach, wenn man bedenkt, daß

$$1 \equiv_{2^n+1} 2^{2n} = \underbrace{2^{\ell+1}}_{=2n} \underbrace{2^{2n-\ell-1}}_{=(2n)^{-1}}.$$

□

Bemerkung 4.24 *(Lemma 4.21)*
Man kann sogar zeigen (Gathen & Gerhard, 1999, Exercise 8.32), daß 2 genau dann *eine* $2n$*–te* primitive *Einheitswurzel in* R *ist, wenn* n *von der Form* 2^ℓ, $\ell \in \mathbb{N}$*, ist. Für unsere Zwecke reicht aber die einfachere Form von Lemma 4.21 völlig aus.*

"Oh nein!" – entgegnete die Rechenmaschine. "[...] Außerdem wünsche ich, daß Du mich nicht anders ansprichst, als mit dem Titel 'Rechengroßmarschall'. Im Gespräch kannst Du auch sagen: 'Eure Ferromagnetifizenz'. "

S. Lem, *Robotermärchen*

5 Multivariate Polynome I – Grundlagen

Jetzt also mal zu etwas ganz anderem, nämlich zu multivariaten Polynomen und wie man mit diesen etwas "abstrakter" rechnet.

Definition 5.1 *Sei $\mathbb{K}$ ein Körper*[125].

1. *Die Elemente $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}_0^n$ bezeichnet man als* Multiindizes. *Für diese gelten die "Standardnotationen"*

$$\alpha! = \alpha_1! \cdots \alpha_n! \qquad \textit{und} \qquad |\alpha| = \alpha_1 + \cdots + \alpha_n,$$

   sowie

$$x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}, \qquad x = (x_1, \ldots, x_n) \in \mathbb{K}^n.$$

2. *Mit*

$$\Pi = \mathbb{K}[x] = \mathbb{K}[x_1, \ldots, x_n] = (\cdots (\mathbb{K}[x_1])[x_2] \cdots)[x_n]$$

   bezeichnet man die Algebra aller Polynome

$$f(x) = \sum_{\alpha \in \mathbb{N}_0^n} f_\alpha \, x^\alpha, \qquad \#\{\alpha \in \mathbb{N}_0^n \ : \ f_\alpha \neq 0\} < \infty,$$

   in den n Variablen $x_1, \ldots, x_n$.

Die Speicherung der Koeffizienten multivariater Polynome und das Rechnen mit multivariaten Polynomen sind um einiges komplexer als die algorithmische Behandlung univariater Polynome. Allein die Frage, wie man die Koeffizienten eines multivariaten Polynoms im Rechner in einen "linearen" Vektor anordnen sollte (Boor, 2000), hängt sehr stark davon ab, welche Operationen man mit den Polynomen ausführen will; im Zusammenhang mit dem Hornerschema (Peña & Sauer, 2000a; Peña & Sauer, 2000b) und der Polynominterpolation (Sauer, 1995) verbrauchen allein solche Konvertierungen, wenn man sie auf "naive" Art und Weise durchführt, bis zu 90% der Rechenzeit.

[125] Standardbeispiele sind $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ oder aber endliche Körper

5.1 Unser Dauerbeispiel

Das "typische" Problem, mit dem wir uns "herumschlagen" wollen, ist die Lösung nichtlinearer, genauer polynomialer, Gleichungssysteme:

> *Zu einer gegebenen endlichen Teilmenge* $F \subset \Pi$ *finde man* alle *gemeinsamen Nullstellen, d.h., eine Menge* $X \subset \overline{\mathbb{K}}^n$*, so daß*
>
> $$F(X) = 0.$$

Ein besonders einfaches Beispiel, anhand dessen man aber dennoch alle Effekte illustrieren kann, ist das folgende.

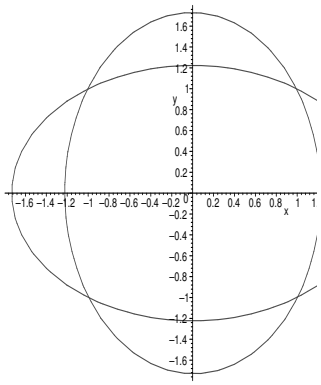


Abbildung 5.1: Die beiden Ellipsen aus Beispiel 5.2. .

Beispiel 5.2 *Wir betrachten die gemeinsamen Nullstellen der beiden Ellipsen*

$$\begin{aligned} f(x_1, x_2) &= \frac{1}{3}x_1^2 + \frac{2}{3}x_2^2 - 1, \\ g(x_1, x_2) &= \frac{2}{3}x_1^2 + \frac{1}{3}x_2^2 - 1. \end{aligned}$$

Diese Nullstellen sind, wie man in Abb 5.1 sieht, die vier Punkte $(\pm 1, \pm 1)$*. Nun aber machen wir das Ganze etwas interessanter, indem wir* g *etwas rotieren, also durch*

$$g_\varphi := g\left(R_\varphi \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right), \qquad R_\varphi = \begin{bmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{bmatrix}$$

ersetzen – dies soll beispielsweise das Auftreten von Rundungs-, Meß- oder Verfahrensfehlern simulieren. Dabei gehen wir aber davon aus, daß der "Störungswinkel" φ sehr klein *ist, daß also $\varphi \sim 0$ gilt.*

Damit sind die Nullstellen immer noch einfach und gut voneinander getrennt.

Natürlich ist das ein Beispiel, das man ohne weiteres "von Hand" lösen kann, insbesondere, wenn $\varphi = 0$ ist. Trotzdem werden wir jetzt erst mal einiges an Theorie aufhäufeln, bevor wir uns wieder dem Beispiel zuwenden werden.

5.2 Graduierte Ringe

In einer Variablen ist der Begriff des Grads eines Polynoms intuitiv klar, man nimmt einfach den höchsten auftretenden Exponenten; die Frage, welcher Exponent der größte ist, stellt sich eigentlich gar nicht, denn für $\mathbb{N}$ gibt es eigentlich nur eine, **die** natürliche Ordnung – zumindest, wenn man fordert, daß diese Ordnung mit der Addition verträglich ist. Wir halten also das univariate Credo fest:

- Der Grad eines Polynoms ist eine Zahl.
- Der Grad eines Polynoms ist der höchste auftretende Exponent.

Damit sind wir aber für $n > 1$ in Schwierigkeiten: Der höchste auftretende Exponent ist jetzt ein *Multiindex* und diese kann man, wie wir bald sehen werden, auf verschiedene Arten anordnen. Bestehen wir hingegen darauf, daß der Grad eine *Zahl* ist, dann müssen wir auf den Exponenten verzichten. Trotzdem hat der Gradbegriff eine wichtige Eigenschaft, auf die wir nicht verzichten wollen und die auf jeden Fall erhalten bleiben sollte, nämlich

$$\deg(f \cdot g) = \deg f + \deg g, \qquad f, g \in \Pi. \tag{5.1}$$

Abstract gesprochen passiert hier etwas: Der Gradbegriff verknüpft die *multiplikative* Struktur der Polynome mit der *additiven* Struktur von $\mathbb{N}_0$ (im Falle univariater Polynome). Grade sollten also etwas sein, was man addieren kann.
Um den Begriff eines *graduierten Rings* gemäß (Eisenbud, 1994) einführen zu können der die Eigenschaft (5.1) "vernünftig"[126] verallgemeinert, brauchen wir erst mal ein bißchen Notation.

Definition 5.3 *Ein* Monoid Γ *ist eine kommutative (additive) Halbgruppe mit*[127] *neutralem Element* 0.

Und tatsächlich besteht die Definition eines graduierten Rings nun darin, die additive Struktur des Graduierungsmonoid auf die multiplikative Struktur eines Rings zu "transplantieren". Wie wir uns jetzt schon vorstellen können, werden verschiedene[128] Graduierungsmonoide auch zu echt verschiedenen Gradbegriffen für denselben Ausgangsring R führen.

[126] Man kann durchaus darüber diskutieren, ob dieser Begriff schon allgemein genug ist, z.B. bei Laurentpolynomen.

[127] Die Menge $2\mathbb{N}$ ist auch eine Halbrguppe aber kein Monoid!

[128] Also echt verschiedene, das heißt nicht–isomorphe!

Definition 5.4 *Ein kommutativer Ring R mit Einselement heißt* graduierter Ring*, wenn es einen (Graduierungs-) Monoid Γ gibt, so daß*

1. *R die direkte Summenzerlegung*
$$R = \bigoplus_{\gamma\in\Gamma} R_\gamma$$
in additive Untergruppen $R_\gamma \subseteq R, \quad \gamma \in \Gamma$, von R besitzt.

2. *die Summanden die Eigenschaft*
$$R_\gamma \cdot R_{\gamma'} \subseteq R_{\gamma+\gamma'}, \qquad \gamma, \gamma' \in \Gamma$$
besitzen.

Die Elemente der direkten Summanden R_γ, $\gamma \in \Gamma$, bezeichnet man als homogene Elemente *von R und schreiben*
$$R^0 = \bigcup_{\gamma\in\Gamma} R_\gamma$$
für die Gesamtheit aller homogenen Elemente von R.

Beispiel 5.5 *(Beispiele für Graduierungen von $\mathbb{K}[x_1, \ldots, x_n]$)*

1. *Graduierung über den* Totalgrad*: Hier ist $\Gamma = \mathbb{N}_0$ und*
$$\Pi_k := \text{span}_{\mathbb{K}} \{x^\alpha \ : \ |\alpha| = k\}.$$

2. *Graduierung in* Monome*: Hier ist $\Gamma = \mathbb{N}_0^n$ und*
$$\Pi_\alpha = \text{span}_{\mathbb{K}} \{x^\alpha\}.$$

3. *Für $n = 1$ fallen diese beiden Graduierungsbegriffe zusammen.*

4. *Eine etwas allgemeinere Graduierung erhält man wie folgt: Seien $v_j \in \mathbb{K}^n$, $j = 1, \ldots, n$,* linear unabhängige *Vektoren und sei $\ell_j(x) = v_j^T x$, $j = 1, \ldots, n$. Dann bilden die "homogenen" Räume*
$$\Pi_k = \text{span}_{\mathbb{K}} \{\ell^\alpha \ : \ |\alpha| = k\}, \qquad \Pi_\alpha = \text{span}_{\mathbb{K}} \{\ell^\alpha\}, \qquad k \in \mathbb{N}_0, \quad \alpha \in \mathbb{N}_0^n,$$
eine Graduierung, wobei $\ell^\alpha = \ell_1^{\alpha_1} \cdots \ell_n^{\alpha_n}$.

Nachdem wir nun also den Begriff des "homogenen" Terms verallgemeinert haben, brauchen wir nun für unseren Gradbegriff einen "größten" homogenen Term eines Ringelements.

Definition 5.6 *Eine Ordnung "$<$" auf einem Monoid Γ heißt* Wohlordnung[129]*, wenn*

[129] Die Schreibweise "heißt wohl Ordnung", die die Falschschreibreform nahelegen würde, bedeutet ja bekanntlich etwas anderes bzw. Anderes.

1. *sie eine* totale Ordnung *ist, das heißt*

$$\gamma, \gamma' \in \Gamma \quad \gamma \neq \gamma' \qquad \Longrightarrow \qquad \gamma < \gamma' \quad \textit{oder} \quad \gamma' < \gamma.$$

2. *sie* kompatibel mit der Halbgruppenoperation "$+$" *ist, das heißt*

$$\gamma < \gamma' \qquad \Longrightarrow \qquad \gamma + \eta < \gamma' + \eta, \qquad \eta \in \Gamma.$$

3. *jede* strikt absteigende *Folge* $\gamma_1 > \gamma_2 > \cdots$ *endlich sein muß.*

Eine einfache Bemerkung über Wohlordnungen auf Monoiden ist, daß wir immer ihr minimales Element kennen.

Lemma 5.7 *Sei "$<$" eine Wohlordnung auf dem Monoid* Γ*. Dann ist* $0 < \gamma$ *für alle* $\gamma \in \Gamma \setminus \{0\}$.

Beweis: Angenommen, es gäbe ein $\gamma < 0$. Dann ist

$$\gamma = \gamma + 0 > \gamma + \gamma =: 2\gamma = 2\gamma + 0 > 2\gamma + \gamma = 3\gamma > \cdots$$

und die strikt absteigende Kette $k\gamma$, $k \in \mathbb{N}$, bricht nie ab, was ein Widerspruch zu Definition 5.6 ist. □

Definition 5.8 *Eine Graduierung auf* $\mathbb{K}[x_1, \ldots, x_n]$ *heißt* Termordnung*, wenn* $\Gamma = \mathbb{N}_0^n$ *und* $\Pi_\alpha = \text{span } \{x^\alpha\}$, $\alpha \in \mathbb{N}_0^n$.

Bemerkung 5.9 *(Wohlordnungen)*

1. *Die einzige Wohlordnung auf* $\mathbb{N}_0$ *ist die kanonische Wohlordnung, denn aus* $0 < 1$ *folgt* $k < k+1 < k+2 < \cdots$ *für alle* $k \in \mathbb{N}_0$.

2. *Auf* $\mathbb{Z}$ *gibt es* keine *Wohlordnung: Für jedes* $k \in \mathbb{Z}$ *müßte* $k > 0$ *sein wie auch* $-k > 0$*, da aber* $0 = -k + k$ *ist die Kompatibilität mit der Addition verletzt.*

3. *Auf* $\mathbb{N}_0^n$ *gibt es jede Menge Wohlordnungen, aus denen man sich die (im jeweiligen Fall) geeignetste aussuchen kann. Die Klassiker sind:*

   (a) Lexikografische Termordnung *("lex"): für* $\alpha \neq \beta \in \mathbb{N}_0^n$ *ist*

$$\alpha \prec_l \beta \qquad \Longleftrightarrow \qquad \alpha_j = \beta_j, \quad j = 1, \ldots, k-1, \quad \alpha_k < \beta_k.$$

   (b) Graduiert lexikografische Termordnung *("gradlex"): für* $\alpha \neq \beta \in \mathbb{N}_0^n$ *ist*

$$\alpha \prec_g \beta \qquad \Longleftrightarrow \qquad |\alpha| < |\beta| \quad \textit{oder} \quad |\alpha| = |\beta|,\ \alpha \prec_l \beta.$$

   Bei der "gradlex"–Ordnung benutzt man also die lexikografische Ordnung als "tie breaker"[130] *für Multiiindizes gleichen Betrags.*

[130] So sagen wirklich viele Leute, nicht nur Tennisfreaks.

Definition 5.10 *Ein Ring R heißt* geordnet graduiert*, wenn zusätzlich zum Graduierungsmonoid Γ auch eine Wohlordnung "$<$" gegeben ist.*

Lemma 5.11 *Sei R ein geordnet graduierter Ring. Dann ist $R^* \subseteq R_0$.*

Beweis: Sei $r \in R^*$ mit inversem Element $s = r^{-1}$ und seien

$$r = \sum_{\gamma\in\Gamma} r_\gamma \qquad \text{und} \qquad s = \sum_{\gamma\in\Gamma} s_\gamma$$

die homogenen Zerlegungen. Dann ist, für $\eta \in \Gamma$,

$$R_\eta = 1 \cdot R_\eta = (rs)\, R_\eta = \sum_{\gamma,\gamma'\in\Gamma} \underbrace{r_\gamma s_{\gamma'} R_\eta}_{\in R_{\gamma+\gamma'+\eta}},$$

und da $\gamma + \gamma' > 0$ falls $(\gamma, \gamma') \neq (0, 0)$, folgt wieder aus der direkten Summendarstellung, daß $r_\gamma = s_\gamma = 0$ für $\gamma \in \Gamma \setminus \{0\}$. □

Oftmals sind Graduierungen interessant, bei denen R_0 so klein ist wie möglich. Also geben wir ihnen einen besonderen Namen.

Definition 5.12 *Eine Graduierung heißt* strikt*, wenn $R_0 = R^*$.*

5.3 Polynomgrade

Jetzt aber zurück zu unserem konkreten Ring $\Pi = \mathbb{K}[x_1, \ldots, x_n]$! In unserer "neuen" Terminologie war der Grad eines univariaten Polynomials ja nichts anderes als der *größte Index*, der zu einer *nichttrivialen*, das heißt von Null verschiendenen, *homogenen* Komponente des Polynoms gehört. Und **diesen** Ansatz können wir jetzt mit den bereitgestellten Mitteln verallgemeinern. Alles, was wir dazu brauchen ist ein wohlgeordneter Graduierungsmonoid für Π.

Definition 5.13 *Sei Γ ein wohlgeordneter Graduierungsmonoid*[131] *für $\mathbb{K}[x_1, \ldots, x_n]$. Für*[132]

$$\Pi \ni f = \sum_{\gamma\in\Gamma} f_\gamma \qquad f_\gamma \in \Pi_\gamma,$$

definiert man

1. *den* (Γ–)Grad *von f als*

$$\delta_\Gamma(f) := \max\{\gamma \in \Gamma \ : \ f_\gamma \neq 0\} \quad \in \Gamma.$$

[131] Dieser Begriff impliziert immer die Existenz einer direkten Summenzerlegung in homogene Komponenten. Diese Zerlegung kann übrigens, das nur zur Erinnerung, für *denselben* Monoid sehr wohl unterschiedlich ausfallen, siehe Beispiel 5.5.

[132] In bewährter Manier ist auch diese Summe natürlich endlich, enthält genauer gesagt nur endlich viele von Null verschiedene Terme.

2. *den* (Γ–)Leitterm *von* f *als*

$$\lambda_\Gamma(f) := f_{\delta_\Gamma(f)} \quad \in \Pi^0.$$

Übung 5.1 Zeigen Sie: Die Begriffe aus Definition 5.13 erfüllt $\delta_\Gamma(f \cdot g) = \delta_\Gamma(f) + \delta_\Gamma(g)$ und $\lambda_\Gamma(f \cdot g) = \lambda_\Gamma(f) \cdot \lambda_\Gamma(g)$.

Bemerkung 5.14
1. *Wir können den Grad bzw. Leitterm auch als Abbildungen von* Π *nach* Γ *bzw.* Π^0 *auffassen.*
2. *Genaugenommen hängen der Grad und damit auch Leitterm vom Monoid* Γ und *von der Ordnung "<" ab. So haben beispielsweise für das Polynom* $f(x,y) = 2x^2y^2 + 3x^3$ *bezüglich der lexikografischen "lex"–Termordnung, daß* $\delta(f) = (3,0)$ *und* $\lambda(f) = 3x^3$*, wohingegen wir für die graduiert*[133] *lexikografische "gradlex"–Termordnung* $\delta(f) = (2,2)$ *und* $\lambda(f) = 2x^2y^2$ *erhalten.*

Um ein bißchen ein Gefühl zu bekommen, was solche Graduierung alles leisten können, wollen wir uns jetzt noch ein paar Beispiele ansehen.

Beispiel 5.15 *(Gewichteter Totalgrad)*
Man wählt einen Vektor $0 \neq \omega \in \mathbb{N}_0^n$*, dann* $\Gamma = \mathbb{N}_0$ *als Monoid und*

$$\Pi_k = \operatorname{span}_{\mathbb{K}} \left\{ x^\alpha \ : \ \omega^T \alpha = k \right\}, \qquad k \in \mathbb{N}_0.$$

Ist hier $\omega = \mathbf{1} = [1, \ldots, 1]^T$*, dann hat man "normale" Graduierung nach Totalgrad, auch als* H–Grading[134] *bezeichnet.*
Es ist übrigens nicht *verboten, daß* $\omega_j = 0$ *ist für einen oder mehrere Werte von* j*, nur eben nicht für alle. Ist beispielsweise* $\omega_2 = \cdots = \omega_n = 0$*, dann ist der Grad eines Polynoms der Grad als Polynom in* x_1*; insbesondere besteht also* Π_0 *aus allen Linearkombinationen der Monome* $x_2^{\alpha_2} \cdots x_n^{\alpha_n}$*. Allerdings ist dies dann keine strikte Graduierung mehr.*

Beispiel 5.16 *(Matrixgewichtung)*
Was wir mit Vektoren machen können, geht natürlich auch mit Matrizen: Sei $m \in \mathbb{N}$ *und* $0 \neq M \in \mathbb{N}_0^{m \times n}$*, sowie* $\prec$ *eine Wohlordnung auf* $\mathbb{N}_0^m$*, dann setzen wir*

$$\Pi_\beta = \operatorname{span}_{\mathbb{K}} \left\{ x^\alpha \ : \ M\alpha = \beta \right\}, \qquad \beta \in \mathbb{N}_0^m.$$

Hierbei kann es durchaus passieren, daß $\Pi_\beta = \{0\}$ *für manche Werte von* β*. Hier noch ein paar Spezialfälle:*

1. *Ist* $m = 1$*, so sind wir in der Situation von Beispiel 5.15.*

[133] Zuerst kommt der Totalgrad und dann, als "Tie break" die lexikografische Ordnung.
[134] Hier steht "H" natürlich für "homogen".

2. *Setzen wir $M = 2I$, so ist $\Pi_\beta = \{0\}$ wann immer $\beta_j \in 2\mathbb{N}+1$ für mindestens ein $j \in \{1,\dots,n\}$.*

3. *Setzen wir $m = n+1$, und wählen wir $\prec$ als die lexikografische Ordnung auf $\mathbb{N}_0^m$, dann erhalten wir mit der Matrix*
$$M = \begin{bmatrix} 1 & \dots & 1 \\ 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix}$$
die "gradlex"–Graduierungen.

4. *Tatsächlich kann man, wie Robbiano gezeigt hat,* alle *Termordnungen mittels Matrixmultiplikation auf die lexikografische Termordnung zurückführen.*

5. *Ist generell $\prec_m$ eine Wohlordnung auf $\mathbb{N}_0^m$ und ist $M \in \mathbb{N}_0^{m\times n}$, dann ist die Ordnung $\prec_n$ auf $\mathbb{N}_0^n$, definiert durch*
$$\alpha \prec_n \beta \qquad \Longleftrightarrow \qquad M\alpha \prec_m M\beta, \qquad \alpha,\beta \in \mathbb{N}_0^n,$$
genau dann eine Wohlordnung, wenn $\ker M = \{\alpha \in \mathbb{Z}^n \,:\, M\alpha = 0\} = \{0\}$*; diese Bedingung ist notwendig und hinreichend für die* Vergleichbarkeit*, also für eine* totale *Ordnung.*

Beispiel 5.17 *Man kann sogar mit $\mathbb{R}$, genauer gesagt mit einem endlich erzeugten Teilmonoid von $\mathbb{R}$ graduieren: Hierzu sei $\omega \in \mathbb{R}_+^n$ ein Vektor dessen Komponenten über $\mathbb{Q}$ linear unabhängig sind, d.h.*
$$\left\{q \in \mathbb{Q}^n \,:\, \omega^T q = 0\right\} = \{0\}.$$
Solche Vektoren gibt es zuhauf[135]*, z.B. $\omega = \left(1, \sqrt{2}, \pi\right)^T$. Und dann ist*
$$\alpha \prec \beta \qquad \Longleftrightarrow \qquad \underbrace{\omega^T\alpha}_{\in\mathbb{R}} < \underbrace{\omega^T\beta}_{\in\mathbb{R}}$$
sogar eine Termordnung.

Definition 5.18 *Eine Graduierung heißt* monomial*, wenn alle homogenen Teilräume Π_γ, $\gamma \in \Gamma$, als $\mathbb{K}$–Vektorräume von Monomen aufgespannt werden.*

[135] Preisfrage: Welche Dimension hat $\mathbb{R}$ als $\mathbb{Q}$–Vektorraum?

Diese Voraussetzung setzt ihn in eine Leidenschaft, die den ganzen Grund seiner Seele eröffnet, alle Geburten seiner Phantasie, alle Resultate seines stillen Denkens ans Licht bringt und deutlich zu erkennen gibt, wie sehr ihn diese Ideale beherrschen.

F. Schiller, *Briefe über Don Carlos*

6 Multivariate Polynome II – Ideale

Die Theorie der Polynomideale hat natürlich eine ganze Menge mit der Lösung des Gleichungssystems $F(X) = 0$ zu tun. Ist nämlich $F \subset \Pi$ eine endliche Menge von Polynomen, dann haben wir offensichtlich, daß

$$\begin{aligned} F(X) = 0 &\implies f(X) = 0, \qquad f \in F, \\ &\implies (g_f \cdot f)(X) = 0, \qquad g_f \in \Pi, \quad f \in F \\ &\implies \left(\sum_{f \in F} g_f \cdot f \right)(X) = 0, \qquad g_f \in \Pi, \quad f \in F \\ &\implies \langle F \rangle (X) = 0, \end{aligned}$$

und da $F \subset \langle F \rangle$ gilt hier überall sogar Äquivalenz.

Definition 6.1 *Ein* Ideal $\mathscr{I}$ *in* Π *ist eine Teilmenge von* Π *mit den Eigenschaften*

$$\mathscr{I} + \mathscr{I} = \mathscr{I} \qquad \textit{und} \qquad \Pi \cdot \mathscr{I} = \mathscr{I}.$$

1. *Sei* $F \subset \Pi$. *Dann ist das von* F *erzeugte Ideal* $\langle F \rangle$ *definiert als*

$$\langle F \rangle := \left\{ \sum_{f \in F} g_f \, f \ : \ g_f \in \Pi, \, f \in F \right\}.$$

2. *Sei* $X \subset \mathbb{K}^n$. *Dann ist das zur Varietät gehörige (radikale) Ideal* $\mathscr{I}(X)$ *definiert als*

$$\mathscr{I}(X) := \{ f \in \Pi \ : \ f(X) = 0 \}.$$

6.1 "Gute" Idealbasen

Wie wir gesehen haben, hängt das Gleichungssystem $F(X) = 0$ eigentlich nicht von F sondern von $\langle F \rangle$ ab[136], wir lösen also eigentlich nicht $F(X) = 0$ sondern $\langle F \rangle (X) = 0$. Das heißt

[136] Das ist typisch für eine ganze Menge von Problemen, in die multivariate Polynome verwickelt sind.

aber nun wieder auch, daß für zwei Teilmengen $F, G \subset \Pi$ mit der Eigenschaft $\langle F \rangle = \langle G \rangle$ die Gleichungssysteme $F(X) = 0$ und $G(X) = 0$ äquivalent sind.

Definition 6.2 *Eine Teilmenge $F \subset \mathscr{I}$ heißt* Basis *eines Ideals $\mathscr{I}$, wenn $\mathscr{I} = \langle F \rangle$ ist, das heißt,*

$$g \in \mathscr{I} \qquad \Longleftrightarrow \qquad g = \sum_{f \in F} g_f \, f, \qquad g_f \in \Pi, \; f \in F.$$

Das folgende Resultat ist die Grundlage für das praktische Rechnen mit Polynomidealen, denn es sagt, daß sich *jedes* Polynomideal mit *endlicher* Information darstellen läßt.

Satz 6.3 *(Hilbertscher Basissatz)*
Jedes Ideal in Π hat eine endliche *Basis.*

Wir werden diesen Satz hier nicht beweisen, sondern gläubig hinnehmen, getreu der Devise

> *The proof of the Hilbert Basis Theorem is not mathematics; it is theology.*
>
> (P. Gordan)[137]

hinter der eine sehr interessante Geschichte um Personen, Persönlichkeiten und Einflußnahme auf Publikationsorgane steckt[138].

Für unsere Zwecke reicht es aber, zu wissen, daß wir jedes Polynomideal als endliche Menge von Polynomen beschreiben und somit auch am Rechner darstellen kann. Die generelle Strategie zur Lösung unserer Gleichungssysteme besteht also darin, zu einer gegebenen endlichen Menge $F \subset \Pi$ eine "gute" Basis G des Ideals $\langle F \rangle = \langle G \rangle$ zu bestimmen, so daß man das Problem $G(X) = 0$ einfacher lösen kann – das heißt, eigentlich wären wir ja fast schon damit zufrieden, das Problem überhaupt lösen zu können. Die Basen, die sich hierbei als besonders sinnvoll erweisen werden, heißen Γ–Basen und sind eng mit dem Gradbegriff verknüpft.

Definition 6.4 *Sei Γ ein Graduierungsmonoid*[139] *für Π. Eine Teilmenge $G \subset \mathscr{I} \subseteq \Pi$ heißt* Γ–Basis *von $\mathscr{I}$ wenn*

$$f \in \mathscr{I} \qquad \Longleftrightarrow \qquad f = \sum_{g \in G} f_g \, g, \qquad \delta_\Gamma(f) \geq \delta_\Gamma\left(f_g \, g\right), \; g \in G. \tag{6.1}$$

Die Darstellung von g auf der rechten Seite von (6.1) bezeichnet man als Γ–Darstellung.

[137] In (Eisenbud, 1994) wird dieses Zitat ebenfalls dem "reigning king of invariants", zugeschrieben. In `http://www-groups.dcs.st-and.ac.uk/~history/index.html` findet man es zweimal, einmal für Paul Gordan und einmal für Camille Jordan. Es scheint also eine mündlich–phonetische Überlieferung mathematischer Mythen und Legenden zu geben.

[138] An dieser Stelle sei nochmals auf die sehr interessante und lesenswerte Kurzbiografie von D. Hilbert in `http://www-groups.dcs.st-and.ac.uk/~history/index.html` verwiesen, in der sich auch der Bezug zu Gordan (bis vor kurzem stand dort noch "Jordan"!) und einiges über die Rolle von F. Klein im Zusammenhang mit der Publikation des Hilbertschen Basissatzes (Hilbert, 1890) finden läßt.

[139] Hier nehmen wir von nun an immer an, daß auf dem Graduierungsmonoid auch eine Wohlordnung vorliegt, damit wir auch von "Grad" und "Leitterm" reden können.

1. *Ist* $\Gamma = \mathbb{N}_0^n$ *und verwendet man eine Termordnung, so spricht man von* Gröbnerbasen[140].

2. *Ist* $\Gamma = \mathbb{N}_0$ *und graduiert man nach dem Totalgrad, dann bezeichnet man die resultierenden Basen als* H–Basen[141].

Bemerkung 6.5 *Das Entscheidende an einer* Γ*–Basis ist, daß es eine Darstellung von* f *bezüglich* G *gibt, bei der kein Term auf der rechten Seite einen höheren Grad hat als* f*. Eine solche Darstellung ist* nichtredundant*! Andernfalls müßten sich nämlich in der Summe homogene Terme höheren Grads aufheben, die Summe selbst wäre also redundant. Aber: nicht jede Basis ist eine* Γ*–Basis.*

Satz 6.6 *Für jeden Graduierungsmonoid* Γ *besitzt jedes Polynomideal* $\mathscr{I} \subset \Pi$ *eine endliche* Γ*–Basis* G*.*

Der Beweis dieses Satzes wird noch einige Zeit auf sich warten lassen, nämlich bis zu Abschnitt 6.3, dafür werden wir aber auch ein *konstruktives* und algorithmisches Verfahren angeben, diese Basen zu berechnen.

6.2 Division mit Rest und Normalformen

Der Schlüssel zum Erfolg wird (wieder einmal) ein Verfahren zur Division mit Rest sein, nur müssen wir jetzt etwas anders vorgehen, denn unser Ring ist leider nicht mehr euklidisch. Trotzdem läßt sich das Konzept der Division mit Rest verallgemeinern, wenn wir sie nur aus dem "richtigen" Blickwinkel betrachten. Bekanntlich läßt sich ja zu gegebenem $f \in \mathbb{K}[x]$ jedes Polynom $g \in \mathbb{K}[x]$ als

$$g(x) = p(x)\, f(x) + r(x), \qquad \deg r < \deg f, \tag{6.2}$$

schreiben, wobei die Darstellung durch die Gradforderung an r eindeutig wurde. Allerdings war es ja gerade die Existenz einer solchen Gradfunktion und einer solchen Zerlegung, die einen euklidische Ring auszeichnete, siehe Definition 2.16. Andererseits könnten und werden wir aber auch den folgenden Standpunkt einnehmen:

1. Das Polynom $p(x)\, g(x)$ ist ein Element des von g erzeugten (Haupt–)Ideals $\langle g \rangle$.

2. Nehmen wir an, daß $f(x) = f_n x^n + \cdots$. Dann heißt ja $\deg r < \deg f$ nichts anderes, als daß r keinen nichttrivialen Term der Form $r_n x^n, r_{n+1} x^{n+1}, \ldots$ enthält, also keinen Term, der sich durch $f_n x^n$ teilen läßt[142]; und $f_n x^n$ ist aber wieder nichts anderes als der *Leitterm* $\lambda(f)$ von f.

[140] "Erfunden" von B. Buchberger (Buchberger, 1965) im Jahre 1965, siehe auch (Buchberger, 1970; Buchberger, 1998a) sowie (Buchberger, 1998b). Die Anekdoten, wer was wann und wo erfunden hat und inwieweit Gröbner selbst involviert war, variieren im übrigen von Erzähler zu Erzähler.

[141] Und diese wurden bereits von F. S. Macaulay (Macaulay, 1916) im Jahre 1916 eingeführt. Allerdings gibt es, wie wir sehen werden, rechnerische Schwierigkeiten mit den H–Basen und alle Beispiele, die Macaulay vorgeführt hat, waren in Wirklichkeit Gröbnerbasen, siehe (Möller & Sauer, 2000a).

[142] Nicht vergessen: $f_n \in \mathbb{K} \setminus \{0\}$ ist eine Einheit, also hat Teilbarkeit nur was mit dem Monom x^n zu tun!

Die Idee ist nun also ganz einfach: Wir dividieren nun mit Rest durch eine endliche Menge $F \subset \Pi$, bzw. eigentlich durch das von F erzeugte Ideal $\langle F \rangle$ und ersetzen die Gradbedingung auf geeignete Weise durch eine Teilbarkeitsforderung.

Beispiel 6.7 *Beginnen wir einfach, indem wir ein $\alpha \in \mathbb{N}_0^n$ fixieren und $F = \{x^\alpha\}$ setzen. Dann können wir ein beliebiges Polynom*

$$g(x) = \sum_{\beta \in \mathbb{N}_0^n} g_\beta \, x^\beta \qquad \in \Pi$$

als

$$g(x) = \underbrace{\left(\sum_{\beta \in \alpha + \mathbb{N}_0^n} g_\beta \, x^{\beta - \alpha} \right) x^\alpha}_{\in \langle F \rangle} + \sum_{\beta \in \mathbb{N}_0^n \setminus \left(\alpha + \mathbb{N}_0^n\right)} g_\beta \, x^\beta$$

schreiben. Das Monom $f(x) = x^\alpha$ zerlegt also den "Träger" von g in zwei Teile, nämlich in die Terme[143]*, die durch x^α teilbar sind und eben diejenigen, die es nicht sind. Diese Zerlegung ist in Abb. 6.1 dargestellt.*

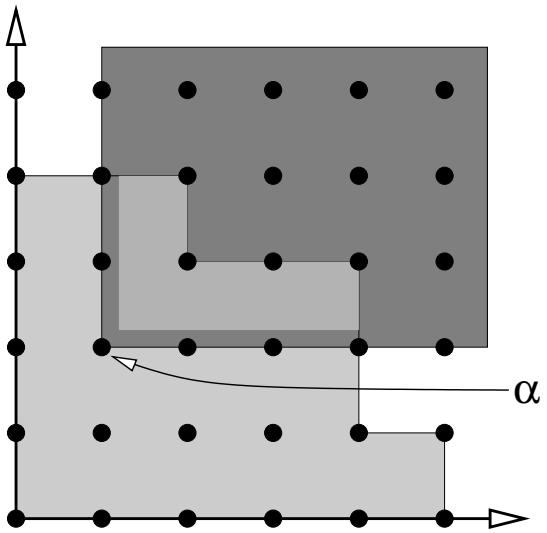


Abbildung 6.1: Diejenigen Exponenten, die zum von $\alpha = (1, 2)$ aufgespannten Kegel $\alpha + \mathbb{N}_0^2$ gehören und diejenigen, die nicht dazugehören. Das liefert also die Zerlegung eines Polynoms g, dessen "Träger" eingezeichnet ist, in "Division" und "Rest".

[143] Ein *Term* ist ein Ausdruck von der Form $c\,x^\alpha$, $0 \neq c \in \mathbb{K}$, $\alpha \in \mathbb{N}_0^n$, also ein Vielfaches eines Monoms.

Definition 6.8 *Sei $F \subset \Pi$, $\#F < \infty$, und sei der Graduierungsmonoid Γ als $\Gamma = \mathbb{N}_0^n$ gewählt*[144].

1. *Mit*
$$\lambda(F) = \lambda_\Gamma(F) = \{\lambda_\Gamma(f) \ : \ f \in F\}$$
bezeichnen wir die Menge der in F auftretenden Leitterme.

2. *Wir sagen F, bzw. das Ideal $\langle F \rangle$* teilt *$g \in \Pi$ mit* Rest *r, wenn es Polynome $g_f \in \Pi$ gibt, so daß*
$$g = \sum_{f \in F} g_f \, f + r \tag{6.3}$$
und kein Element[145] *von $\lambda(F)$ irgendeine der homogenen Komponenten von r teilt.*

3. *Wir nennen eine Zerlegung der Form (6.3) von g eine* G–Darstellung[146] *von g bezüglich F, wenn außerdem noch $\delta_\Gamma(g) \geq \delta_\Gamma(g_f \, f)$ sowie $\delta_\Gamma(g) \geq \delta_\Gamma(r)$ gilt.*

Bemerkung 6.9 *Weder die "Koeffizienten" g_f noch der Rest r sind in (6.3) eindeutig bestimmt! Wählen wir beispielsweise*
$$F = \left\{xy - 2, x^2 + 2y - 1\right\} =: \{f_1, f_2\} \quad \subset \mathbb{K}[x, y],$$
und $g(x) = x^2 y$, dann erhalten wir die beiden Darstellungen
$$\begin{array}{rcrcrcrcl} g(x,y) & = & g_1(x,y) & f_1(x,y) & + & g_2(x,y) & f_2(x) & + & r(x,y) \\ x^2 y & = & x & (xy-2) & + & 0 & (x^2+2y-1) & + & 2x \\ & = & 0 & (xy-2) & + & y & (x^2+2y-1) & + & y - 2y^2. \end{array}$$
Verwendet man die lexikografische Termordnung (mit $x > y$), dann sind beide Darstellungen sogar G–Darstellungen – wir brauchen also nicht zu hoffen, daß wir durch Gradbeschränkung Eindeutigkeit erhalten könnten.

Beginnen wir mit einem Verfahren, das eine G–Darstellung eines gegenen Polynoms g bezüglich einer endlichen Menge F berechnet – die praktische Durchführung der Division mit Rest.

Algorithmus 6.10 *(Division mit Rest, Version Termordnung)*
Gegeben: *$g \in \Pi$, endliche Teilmenge $F \subset \Pi$ und* Termordnung Γ.

1. *Setze $j \leftarrow 0$ und*
$$g_0 \leftarrow g, \qquad r \leftarrow 0, \qquad g_f \leftarrow 0, \quad f \in F.$$

[144] Das heißt, alle "homogenen" Räume werden von genau einem Monom aufgespannt, die über eine Termordnung angeordnet sind.
[145] Das sind jetzt Terme, nicht vergessen!
[146] "G" wie "Gröbner"!

2. *Solange* $g_j \neq 0$

   (a) *Gibt es* $f \in F$, *so daß* $\lambda(f) \mid \lambda(g_j)$?

   ja: *Setze*

$$g_f \leftarrow g_f + \underbrace{\frac{\lambda(g_j)}{\lambda(f)}}_{\in \Pi}, \qquad g_{j+1} \leftarrow g_j - \frac{\lambda(g_j)}{\lambda(f)} f \tag{6.4}$$

   nein: *Setze*

$$r \leftarrow r + \lambda(g_j), \qquad g_{j+1} \leftarrow g_j - \lambda(g_j). \tag{6.5}$$

   (b) $j \leftarrow j + 1$.

Ergebnis: *Polynome* $q_f \in \Pi$, $f \in F$, *und* $r \in \Pi$, *so daß*

$$g = \sum_{f \in F} g_f f + r, \qquad \delta(g_f f) \leq \delta(g),\ \delta(r) \leq \delta(g). \tag{6.6}$$

Proposition 6.11 *Algorithmus 6.10 terminiert nach endlich vielen Schritten und liefert die Darstellung (6.6), wobei* r *ein Rest im Sinne von Definition 6.8 ist.*

Beweis: Da in jedem Schritt des Algorithmus der Leitterm von g_j eliminiert wird, einmal durch Subtraktion eines geeigneten Vielfachen[147] von f, einmal direkt, haben wir, daß

$$\delta(g_{j+1}) < \delta(g_j), \qquad j = 0, 1, 2, \ldots$$

und nach endlich vielen Schritten müssen wir beim Nullpolynom angekommen sein.
Die "wichtigere" Eigenschaft, nämlich (6.6) folgt aus der "Invarianten"

$$g = g_j + \sum_{f \in F} g_f f + r, \qquad j = 0, 1, 2, \ldots \tag{6.7}$$

die für $j = 0$ aufgrund der Initialisierung trivialerweise erfüllt ist. Nun ist aber im j–ten Schritt des Algorithmus im ersten Fall[148]

$$\begin{aligned} g_{j+1} + \sum_{f \in F} \tilde{g}_f f + r &= \left(g_j - \frac{\lambda(g_j)}{\lambda(f)} f\right) + \sum_{f' \in F \setminus \{f\}} g_{f'} f' + \left(g_f + \frac{\lambda(g_j)}{\lambda(f)}\right) f \\ &= g_j + \sum_{f \in F} g_f f + r \end{aligned}$$

und im zweiten Fall

$$g_{j+1} + \sum_{f \in F} \tilde{g}_f f + \tilde{r} = (g_j - \lambda(g_j)) + \sum_{f \in F} \tilde{g}_f f + (r + \lambda(g_j)) = g_j + \sum_{f \in F} g_f f + r,$$

womit sich (6.7) per Induktion beweisen lässt. □

[147] Wer's nicht glaubt bemerkt einfach, daß $\lambda\left(\frac{\lambda(g_j)}{\lambda(f)} f\right) = \lambda(g_j)$.

[148] Hier bezeichnet g_f den Wert der Funktion im $j$ten Schritt und $\tilde{g}_f$ den Wert im $(j+1)$ten Schritt **nach** dem Update in (6.4) bzw. (6.5).

Beispiel 6.12 *Sehen wir uns mal die Vorgehensweise dieses Algorithmus anhand der Polynome* $F = \{xy - 2, x^2 + 2y - 1\}$ *und der graduiert lexikografischen Termordnung mit* $x > y$ *aus Bemerkung 6.9 an.*

1. *Sei* $g(x, y) = x^3 + y^3 + 1$. *Dann erhalten wir*[149]

j	g_j	$\lambda(g_j)$	g_1	g_2	r
0	$x^3 + y^3 + 1$	x^3	0	0	0
1	$y^3 - 2xy + x + 1$	y^3	0	$\boxed{x}$	0
2	$-2xy + x + 1$	$-2xy$	0	x	$\boxed{y^3}$
3	$x + 5$	x	$\boxed{-2}$	x	y^3
4	5	5	-2	x	$\boxed{y^3 + x}$
5	0	0	-2	x	$\boxed{y^3 + x + 5}$

was uns die G–Darstellung

$$g = -2(xy - 2) + x\left(x^2 + 2y - 1\right) + y^3 + x + 5$$

von g *bezüglich* F *liefert.*

2. *Mit* $g(x, y) = x^2y$ *ergibt sich entweder*

j	g_j	$\lambda(g_j)$	g_1	g_2	r
0	x^2y	x^2y	0	0	0
1	$2x$	$2x$	$\boxed{x}$	0	0
2	0	0	x	0	$\boxed{2x}$

oder

j	g_j	$\lambda(g_j)$	g_1	g_2	r
0	x^2y	x^2y	0	0	0
1	$-2y^2 + y$	$-2y^2$	0	$\boxed{y}$	0
2	y	y	0	y	$\boxed{-2y^2}$
3	0	0	0	y	$\boxed{-2y^2 - y}$

je nachdem, ob man im ersten Schritt f_1 *oder* f_2 *verwendet.*

Bemerkung 6.13 *Damit sehen wir das grundlegende Problem der multivariaten Division mit Rest schon etwas klarer. Wir können nicht erwarten, daß für beliebige* $F \subset \Pi$ *die G–Darstellung oder der Divisionsrest eindeutig sind. Und wir sehen auch wo die Mehrdeutigkeit herkommt: Sobald es in einem Schritt von Algorithmus 6.10* mehrere *Polynome* $f \in F$ *gibt, so daß*

[149] Eingerahmt sind immer die Variablen, die im $j$ten Schritt des Algorithmus verändert wurden.

$\lambda(f) \mid \lambda(g_j)$, enstehen Mehrdeutigkeiten, die auch zu Mehrdeutigkeiten in der G–Darstellung und, noch schlimmer, im Divisionsrest führen können.
Natürlich könnte man den Algorithmus selbst eindeutig machen: Wenn wir F nicht als "einfache" Teilmenge, sondern als "geordnete" Teilmenge ansehen (was wir immer haben, wenn wir ihre Komponenten in irgendeiner Reihenfolge durchnumerieren), dann kann man immer das "kleinste" $f \in F$ wählen, dessen Leitterm den Leitterm von g_j teilt und damit wäre der Algorithmus wohldefiniert! Allerhings hängt in diesem Fall das Ergebnis dann möglicherweise von der Reihenfolge *der Elemente in F ab.*

Was aber ist nun so wichtig daran, einen *eindeutigen* Divisionsrest zu erhalten? Nun, hätten wir einen eindeutigen Divisionsrest, dann könnten wir das *Ideal Membership Problem* entscheiden, nämlich die Frage

Gehört ein gegebenes $g \in \Pi$ zum Ideal $\langle F \rangle$?

Denn natürlich liefert $r = 0$ eine Darstellung von g bezüglich $\langle F \rangle$, das heißt, $r = 0$ bedeutet insbesondere $g \in \langle F \rangle$. Und wäre die Restbestimmung eindeutig, dann müßte der Divisionsalgorithmus in diesem Fall auch $r = 0$ liefern. Daß es nicht für *jede* Basis klappt, das wissen wir schon, aber die gute Nachricht ist: Es funktioniert für *gute* Basen!

Satz 6.14 *Ist G eine Gröbnerbasis für das Ideal $\langle G \rangle$ und hat $f \in \Pi$ die beiden G–Darstellungen*

$$f = \sum_{g \in G} f_g \, g + r = \sum_{g \in G} f'_g \, g + r',$$

dann ist $r = r'$.

Wir werden diesen Satz an dieser Stelle nicht beweisen, sondern später, in Satz 6.22, gleich für beliebige Γ–Basen[150]. Jetzt erst mal noch eine Begrifflichkeit.

Definition 6.15 *Sei $F \subset \Pi$ und sei G eine Gröbnerbasis*[151] *für $\langle F \rangle$. Für $h \in \Pi$ bezeichnen wir den eindeutigen Rest r in Algorithmus 6.10 als* Normalform *von h bezüglich G, in Zeichen $\nu_G(h)$.*

Korollar 6.16 *Ist G eine Gröbnerbasis*[152]*, dann ist*

$$f \in \langle G \rangle \qquad \Longleftrightarrow \qquad \nu_G(f) = 0.$$

Da $\nu_G(f)$ eine algorithmisch bestimmbare Größe ist, kann man so das Ideal membership problem *entscheiden.*

[150] Ja, das Vertagen wird langsam zur Gewohnheit ... Aber die Beweise kommen noch, versprochen! Und dann sind sie kurz und elementar.

[151] Was immer die Existenz einer zugehörigen Termordnung voraussetzt.

[152] Die Worte "für das Ideal $\langle G \rangle$" lassen wir, da redundant, ab sofort weg.

Unser nächstes Ziel besteht nun darin, den Divisionsalgorithmus auf beliebige Graduierungen, also auf einen beliebigen Gradbegriff zu übertragen. Doch hier stoßen wir auf Probleme:

1. Monome teilen einander oder sie tun es nicht – in diesem Fall sind sie "richtig" verschieden. Homogene Polynome im klassischen Sinne teilen einander fast nie, sind auf der anderen Seite aber auch nur sehr selten ganz und gar "unteilbar", z.B. die beiden homogenen Formen $x^2 + y^2$ und $x^3 + y^3$.

2. Außerdem gibt es für jedes Monom x^α, $0 \neq \alpha \in \mathbb{N}_0^n$ immer mindestens ein anderes Monom x^β, das dieses Monom teilt[153]. Auch das ist bei homogenen Polynomen nicht mehr der Fall, wo beispielsweise $x^2 + y^2$ nicht mehr durch lineare Formen teilbar ist.

Trotzdem kann man dieses Problem tatsächlich für beliebige Graduierungen angehen, indem man ein Konzept einführt (Sauer, 2001), das einen *graduellen* Teilbarkeitsbegriff verwendet – wir unterscheiden also nicht mehr nur zwischen "teilt" und "teilt nicht", sondern "teilt mehr" oder "teilt weniger".

Da wir für diesen Ansatz innere Produkte verwenden werden, soll nun $\mathbb{K}$ ein in $\mathbb{C}$ eingebetteter Körper sein, also $\mathbb{K} = \mathbb{Q}$, eine algebraische Erweiterung von $\mathbb{Q}$, $\mathbb{K} = \mathbb{R}$ oder eben $\mathbb{K} = \mathbb{C}$. Ein *inneres Produkt* oder *Skalarprodukt* $(\cdot, \cdot)$ ist dann eine nichtentartete Sesquilinearform mit den Eigenschaften

$$\begin{array}{rclcrcl} (f+f',g) & = & (f,g)+(f,g)\,, & & (\lambda\, f,g) & = & \lambda\,(f,g) \\ (f,g+g') & = & (f,g)+(f,g')\,, & & (f,\lambda\, g) & = & \overline{\lambda}\,(f,g) \\ (f,g) & = & \overline{(g,f)} & & (f,f) & \neq & 0. \end{array}$$

Übrigens: um einen *Hilbertraum* zu erhalten, bei dem man eine Norm über das Skalarprodukt definieren kann, muß man natürlich $(f,f) > 0$ voraussetzen, aber für unsere Zwecke genügt es, wenn das innere Produkt *definit* ist, denn dann ist $W \cap W^\perp = \{0\}$ für jeden Teilraum $W \subset \Pi$. Mehr Information über innere Produkte über beliebigen Körpern (auch Schiefkörpern) findet man in (Brieskorn, 1985, S. 282ff).

Definition 6.17 *Seien Γ ein beliebiger Graduierungsmonoid und $(\cdot,\cdot) : \Pi \times \Pi \to \mathbb{K}$ ein inneres Produkt auf den Polynomen.*

1. *Zu einer beliebigen (endlichen) Menge $F \subset \Pi$ und $\gamma \in \Gamma$ bezeichnen wir mit*[154]

$$V_\gamma(F) := \left\{ \sum_{f\in F} g_f\, \lambda(f) \;:\; g_f \in \Pi_{\gamma-\delta(f)} \right\} \subseteq \Pi_\gamma \tag{6.8}$$

den von $\lambda(F)$ erzeugten "homogenen" Teilraum von Π_γ und damit auch von Π.

[153] Genauer: Es teilen alle Monome x^β, solange nur $\beta_j \leq \alpha_j$, $j = 1, \ldots, n$, ist.

[154] Die Subskripte Γ bei Grad und Leitterm lassen wir ab jetzt weg, da wir nicht auf halbem Wege den Graduierungsmonoid wechseln, sondern als *a priori* festgelegte "Konstante" ansehen.

2. *Für $\gamma \in \Gamma$ bezeichnen wir mit*

$$W_\gamma(F) := V_\gamma^\perp(F) := \Pi_\gamma \ominus V_\gamma(F) = \{g \in \Pi_\gamma \ : \ (g, V_\gamma(F)) = 0\}$$

das orthogonale Komplement *von $V_\gamma(F)$ in Π_γ und schreiben*

$$V(F) = \bigoplus_{\gamma\in\Gamma} V_\gamma(F), \qquad W(F) = \bigoplus_{\gamma\in\Gamma} W_\gamma(F)$$

für die Menge aller Polynome, deren "homogene" Komponenten auf zum jeweiligen $V_\gamma(F)$ orthogonal sind.

3. *Wir sagen, $F \subset \Pi$ teilt $g \in \Pi$ mit Rest $r \in \Pi$, wenn*

$$g = \sum_{f\in F} g_f \, f + r, \qquad r \in W(F). \tag{6.9}$$

Bemerkung 6.18 *1. Offensichtlich ist $V_\gamma(F)$ ein Vektorraum, wenn man bedenkt, daß ja immer $\mathbb{K} \subseteq \Pi_0$ ist, daß also mit $g \in \Pi_{\gamma-\delta(f)}$ auch $\mathbb{K} \cdot g$, zu $\Pi_{\gamma-\delta(f)}$ gehört.*

2. *Hier sieht man schon, wie man sich verhalten könnte, wenn wir es mit einem Körper zu tun hätten, der kein inneres Produkt besitzt. Man bräuchte dann eine (surjektive)* Projektion[155] *P_γ von Π_γ* auf *$V_\gamma(F)$ und könnte dann mit $Q_\gamma = I - P_\gamma$ eine direkte Summenzerlegung*

$$\Pi_\gamma = P_\gamma \, \Pi_\gamma \oplus Q_\gamma \, \Pi_\gamma$$

bilden, denn schließlich ist ja

$$Q_\gamma \, P_\gamma = P_\gamma \, Q_\gamma = P_\gamma \, (I - P_\gamma) = P_\gamma - P_\gamma^2 = P_\gamma - P_\gamma = 0.$$

Allerdings besteht dann das Problem in der praktischen *Bestimmung dieser Projektion. Deswegen machen wir uns lieber das Leben leicht und wählen einen "praxisrelevanten"[156] einfachen Körper $\mathbb{K}$.*

3. *Im Falle einer Termordnung bringen die Begriffe aus Definition 6.17 nichts neues! Denn da für $\beta \in \mathbb{N}_0^n$*

$$V_\beta(F) = \begin{cases} \Pi_\beta, & \beta \in \bigcup\limits_{f\in F} (\delta(f) + \mathbb{N}_0^n), \\ \{0\}, & \beta \notin \bigcup\limits_{f\in F} (\delta(f) + \mathbb{N}_0^n), \end{cases}$$

gilt, siehe Abb. 6.2, ist $r \in W(F)$ genau dann, wenn kein Term aus $\lambda(F)$ einen der Terme von r teilt – also alles wie gehabt.

[155] Das heißt also, daß $P_\gamma^2 = P_\gamma$.

[156] Das ist nicht nur eine faule Ausrede! Das Lösen der Gleichungssysteme $F(X) = 0$ findet im wesentichen in $\mathbb{C}$ oder $\mathbb{R}$ statt.

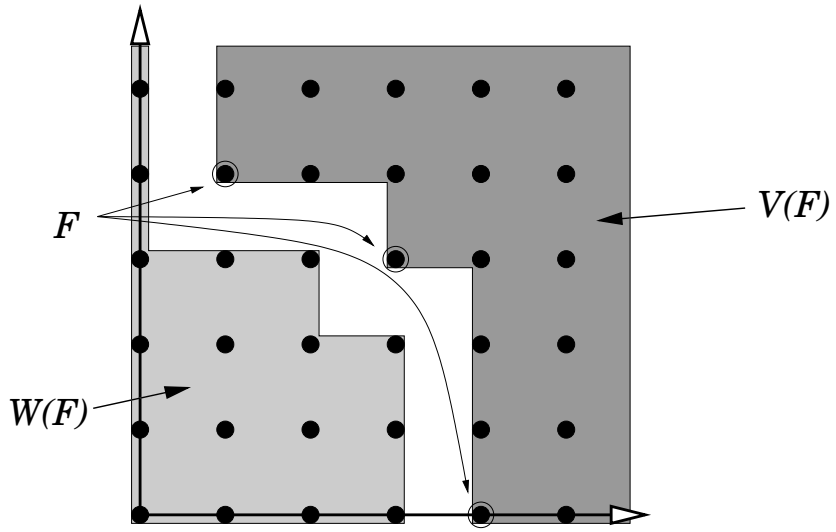


Abbildung 6.2: Die Mengen $V(F)$ und $W(F)$ für die drei Monome xy^4, x^3y^3 und x^4, genauer gesagt, eine endlicher Ausschnitt davon..

4. *Im Falle einer Termordnung ist also das verwendete innere Produkt irrelevant. Im Allgemeinen ist das natürlich nicht der Fall. "Gute" innere Produkte sind beispielsweise für*

$$f(x) = \sum_{\alpha\in\mathbb{N}_0^n} f_\alpha x^\alpha, \qquad g(x) = \sum_{\alpha\in\mathbb{N}_0^n} g_\alpha x^\alpha,$$

   (a) *das innere Produkt der monomialen Koeffizienten*[157]

$$(f,g) := \sum_{\alpha\in\mathbb{N}_0^n} f_\alpha\, g_\alpha.$$

   (b) *das innere Produkt*

$$(f,g) := \sum_{\alpha\in\mathbb{N}_0^n} \alpha!\, f_\alpha\, g_\alpha = (f(D)g)\,(0),$$

   wobei

$$f(D) = \sum_{\alpha\in\mathbb{N}_0^n} f_\alpha \frac{\partial^{|\alpha|}}{\partial x^\alpha}$$

   ein partieller Differentialoperator mit konstanten Koeffizienten ist.

[157] Wir verzichten hier auf komplexe Konjugation, die man im Falle $\mathbb{K} = \mathbb{C}$ eigenlich noch hinzufügen müßte, denn da sind innere Produkte ja nur Sesquilinearformen.

Beispiel 6.19 *Bei der H–Graduierung wird die Sache schon interessanter. Betrachten wir nämlich die Situation $F = \{x^2 + y^2\}$, das innere Produkt $(f, f') = f(D)f'(0)$ und $g = x^3 + y^3$. Dann hat $W_3(F)$ die Basis*

$$V_3(F) = \text{span}_{\mathbb{K}} \left\{ x^3 + xy^2, x^2y + y^3 \right\},$$

deren Elemente sogar orthogonal *sind. Dann ist*

$$W_3(F) = \text{span}_{\mathbb{K}} \left\{ x^3 - 3xy^2, 3x^2y - y^3 \right\}$$

und die Zerlegung ist

$$g(x) = \underbrace{\left(\frac{3}{4}x + \frac{3}{4}y \right) \left(x^2 + y^2 \right)}_{\in V_3(F)} + \underbrace{\left(\frac{1}{4}x^3 - \frac{3}{4}x^2y - \frac{3}{4}xy^2 + \frac{1}{4}y^3 \right)}_{\in W_3(F)}.$$

Jetzt haben wir aber alle Bausteine beisammen, um eine allgemeine Version von Algorithmus 6.10 für beliebige Graduierungen angeben und untersuchen zu können (damit auch die Theorie mal vorankommt).

Algorithmus 6.20 *(Division mit Rest, allgemeine Version)*
Gegeben: $g \in \Pi$, *endliche Teilmenge* $F \subset \Pi$ *und Graduierung* Γ.

1. *Initialisiere*

$$j \leftarrow 0, \qquad g_0 \leftarrow g, \qquad r \leftarrow 0, \qquad g_f \leftarrow 0, \quad f \in F.$$

2. *Solange* $g_j \neq 0$

   (a) *Setze*

$$\gamma \leftarrow \delta\left(g_j\right), \qquad h_j \leftarrow \lambda\left(g_j\right).$$

   (b) *(Orthogonale Projektion) Bestimme* $h_f \in \Pi_{\gamma - \delta(f)}$ *so daß*[158]

$$r_j := h_j - \underbrace{\sum_{f \in F} h_f \, \lambda(f)}_{\in V_\gamma(F)} \perp V_\gamma(F).$$

   (c) *Setze*

$$g_{j+1} \leftarrow g_j - \sum_{f \in F} h_f \, f - r_j, \qquad r \leftarrow r + r_j, \qquad g_f \leftarrow g_f + h_f, \quad f \in F.$$

   (d) *Setze* $j \leftarrow j + 1$.

[158] Man beachte: Diese Berechnung erfolgt komplett im *homogenen* Raum Π_γ.

Ergebnis: *Polynome $g_f \in \Pi$, $f \in F$, und $r \in \Pi$, so daß*

$$g = \sum_{f \in F} g_f \, f + r, \qquad \delta(g) \geq \begin{cases} \delta\left(g_f \, f\right), \\ \delta(r), \end{cases} \qquad r \in W(F). \tag{6.10}$$

Proposition 6.21 *Algorithmus 6.20 terminiert nach endlich vielen Schritten und liefert die Γ–Darstellung (6.10).*

Beweis: Der Beweis ähnelt natürlich sehr stark dem von Proposition 6.11[159]. Da[160]

$$\lambda\left(\sum_{f \in F} h_f \, f - r_j\right) = \sum_{f \in F} h_f \, \lambda(f) - \lambda\left(r_j\right) = h_j = \lambda\left(g_j\right),$$

reduziert sich der Grad in jedem Schritt, das heißt $\delta\left(g_{j+1}\right) < \delta\left(g_j\right)$.
Die Korrektheit, das heißt, die Gültigkeit von (6.10), folgt wieder aus der Invarianz

$$g = g_j + \sum_{f \in F} g_f \, f + r, \qquad j = 0, 1, 2, \ldots,$$

die für $j = 0$ wieder trivialerweise richtig ist und sich mit

$$\begin{aligned} g_{j+1} &= g_j - \sum_{f \in F} h_f \, f - r_j = g - \sum_{f \in F} g_f \, f - r - \sum_{f \in F} h_f \, f - r_j \\ &= g - \sum_{f \in F} \underbrace{\left(g_f + h_f\right)}_{=: \widetilde{g}_f} f - \underbrace{r + r_j}_{=: \widetilde{r}} \end{aligned}$$

induktiv beweisen läßt, wobei wieder $\widetilde{g}_f$ und $\widetilde{r}$ die entsprechenden Wert nach dem $j$ten Schritt bezeichnen. □

Natürlich brauchen wir uns hier erst recht nicht einzubilden, daß wir Eindeutigkeit der Γ–Darstellung oder des Restes erhalten würden, es sei denn, wir haben es mit einer Γ–Basis zu tun. Hier ist sie also, die versprochene Verallgemeinerung von Satz 6.14, nur diesmal **mit**[161] Beweis.

Satz 6.22 *Sei G eine Γ–Basis und seien zwei Γ–Darstellungen*

$$f = \sum_{g \in G} f_g \, g + r = \sum_{g \in G} f'_g \, g + r'$$

von f gegeben. Dann ist $r = r'$.

[159] Man könnte auch sagen, er ist fast identisch.

[160] Achtung: Das ist keine generelle Eigenschaft des Leitterms, sondern liegt daran, daß die *homogenen* Faktoren h_f passend gewählt wurden, nämlich so, daß alles denselben Grad hat: $\delta\left(h_f \, f\right) = \delta\left(h_{f'} \, f'\right)$, $f, f' \in F$.

[161] Ja, sowas kommt gelegentlich auch vor.

Beweis: Wir nehmen an, es wäre $r \neq r'$. Offenbar gehört das Polynom

$$0 \neq q := r - r' = \sum_{g \in G} \left(f_g - f'_g\right)\, g$$

zum Ideal $\mathscr{I} = \langle G \rangle$ und lässt sich daher auch als Γ–Darstellung

$$q = \sum_{g \in G} q_g\, g, \qquad \delta\left(q_g\, g\right) \leq \delta(q) \tag{6.11}$$

schreiben, denn schließlich war ja G als Γ–Basis vorausgesetzt[162]. Betrachten wir nur die Leitterme in (6.11), dann ist

$$0 \neq \lambda(q) = \sum_{g \in G'} \lambda\left(q_g\right)\, \lambda(g) \in V_{\delta(q)}\left(G\right), \qquad G' = \{g \in G \,:\, \lambda\left(q_g\, g\right) = \lambda(q)\} \neq \emptyset.$$

Insbesondere ist $\lambda(q) \in V(G)$. Auf der anderen Seite sind aber $r, r' \in W(G)$, also auch $q = r - r'$ und damit ist insbesondere $\lambda(q) \in W(G)$. Also ist

$$\lambda(q) \in W(G) \cap V(G) = \{0\},$$

was einen Widerspruch zu $q \neq 0$ darstellt. □

Zum Abschluß dieses Kapitels jetzt noch in Generalisierung von Definition 6.15 die allgemeine Definition der Normalform.

Definition 6.23 *Sei G eine Γ–Basis von $\mathscr{I}$. Dann ist die* Normalform *von $f \in \Pi$ bezüglich G oder $\mathscr{I}$, in Zeichen $\nu_G(f)$ oder $\nu_{\mathscr{I}}(f)$, definiert als der Divisionrest in Algorithmus 6.20.*

Um zu zeigen, daß $\nu_{\mathscr{I}}$ tatsächlich nur vom Ideal $\mathscr{I}$ abhängt, mußsen wir noch zeigen, daß zwei (möglicherweise) verschiedene Γ–Basen dennoch denselben Divisionsrest liefern.

Lemma 6.24 *Seien G, G' zwei Γ–Basen für das Ideal $\mathscr{I} = \langle G \rangle = \langle G' \rangle$. Dann ist $\nu_G(f) = \nu_{G'}(f)$ für alle $f \in \Pi$.*

Beweis: Für $g' \in G' \subset \langle G \rangle$ sei

$$g' = \sum_{g \in G} h_{g',g}\, g, \qquad \delta\left(h_{g',g}\, g\right) \leq \delta\left(g'\right),$$

die zugehörige Γ–Darstellung, die existieren muß, da G eine Γ–Basis ist. Sei nun

$$f = \sum_{g' \in G'} f_{g'}\, g' + \nu_{G'}(f), \qquad \delta\left(f_{g'}\, g'\right) \leq \delta(f),$$

[162] Irgendwo müssen wir diese Voraussetzung ja auch verwenden.

die Γ–Darstellung von f bezüglich G'. Dann ist

$$\begin{aligned} f &= \sum_{g' \in G'} f_{g'}\, g' + \nu_{G'}(f) = \sum_{g' \in G'} f_{g'} \left(\sum_{g \in G} h_{g',g}\, g \right) + \nu_{G'}(f) \\ &= \sum_{g \in G} \underbrace{\left(\sum_{g' \in G'} f_{g'} h_{g',g} \right)}_{=: f_g} g + \nu_{G'}(f) \end{aligned}$$

ebenfalls eine Γ–Darstellung, da

$$\begin{aligned} \delta\left(f_g\, g\right) &= \delta(g) + \delta\left(\sum_{g' \in G'} f_{g'} h_{g',g} \right) \leq \delta(g) + \max_{g' \in G'} \left(\delta\left(h_{g',g}\right) + \delta\left(f_{g'}\right) \right) \\ &\leq \max_{g' \in G'} \left(\delta(g) + \delta\left(h_{g',g}\right) + \delta\left(f_{g'}\right) \right) \leq \max_{g' \in G'} \left(\delta\left(g'\right) + \delta\left(f_{g'}\right) \right) \leq \delta\left(f\right). \end{aligned}$$

Nach Satz 6.22 ist dann aber $\nu_G(f) = \nu_{G'}(f)$. □

6.3 Konstruktion von Γ–Basen

Jetzt aber geht es endlich an den (konstruktiven) Existenzbeweis für Γ–Basen, also für den lang angekündigten Satz 6.6, nämlich:

> *Sei $\mathscr{I} \subseteq \Pi$ ein Ideal und sei Γ ein Graduierungsmonoid. Dann hat $\mathscr{I}$ eine endliche Γ–Basis*

Das entscheidende Hilfsmittel für den Beweis ist die folgende algorithmische Charakterisierung von Γ–Basen über den Divisionsalgorithmus.

Proposition 6.25 *Eine endliche Menge $G \subset \Pi$ ist genau dann eine Γ–Basis, wenn für jeden Vektor $q = (q_g \ : \ g \in G) \in \Pi^G$ von Polynomen mit der Eigenschaft*

$$\delta\left(q \cdot G\right) := \delta\left(\sum_{g \in G} q_g\, g \right) < \max_{g \in G} \delta\left(q_g\, g\right) \tag{6.12}$$

der Divisionsalgorithmus 6.20 für $\left(q \cdot G\right)_G$ den Rest 0 liefert.

Beweis: Ist G eine Γ–Basis, so besitzt "natürlich" das Polynom $q \cdot G \in \langle G \rangle$ eine Γ–Darstellung bezüglich G mit Rest 0 und nach Satz 6.22 muß dies dann auch das Ergebnis des Divisionsverfahrens sein.
Das heißt, die eigentliche "Arbeit" wird bei der Umkehrung fällig. Hier verwenden wir einen Ansatz, der in (Möller, 1988) für die Konstruktion von Gröbnerbasen angegeben wurde. Hierbei bezeichne für $f \in \Pi$ der Ausdruck $r_G(f)$ den Divisionsrest von Algorithmus 6.20. Nehmen

wir also an, G hat die Eigenschaft, daß $r_G(q \cdot G) = 0$ wann immer $\delta(q \cdot G) < \max \delta(q_g\, g)$, $q \in \Pi^G$. Sei nun also $f \in \langle G \rangle$ mit einer Darstellung[163]

$$f = \sum_{g \in G} f_g\, g, \qquad f_g \in \Pi, \quad g \in G. \tag{6.13}$$

Unser Ziel wird es sein, eine Γ–Darstellung von f als

$$f = \sum_{g \in G} f'_g\, g, \qquad \delta\left(f'_g\, g\right) \leq \delta(f), \quad g \in G,$$

zu konstruieren, denn wenn das für *alle* $f \in \langle G \rangle$ gelingt, dann ist ja gemäß Definition 6.4 G eine Γ–Basis. Nehmen wir also an, daß (6.13) *keine* Γ–Darstellung ist und setzen

$$\gamma := \max_{g \in G} \delta(f_g\, g) > \delta(f) \qquad \text{sowie} \qquad J := \{g \in G \,:\, \delta(f_g\, g) = \gamma\}.$$

Damit erfüllt aber

$$q = (q_g \,:\, g \in G) \qquad \text{mit} \qquad q_g = \begin{cases} \lambda(f_g), & g \in J, \\ 0 & g \notin J, \end{cases} \quad g \in G, \tag{6.14}$$

die Bedingung daß

$$\delta(q \cdot G) < \gamma = \max_{g \in G} \delta(q_g\, g)$$

und hat, dank des Divisionsalgorithmus und der Annahme an G eine Γ–Darstellung

$$q \cdot G = \sum_{g \in G} q'_g\, g + \underbrace{r_G(q \cdot G)}_{=0} = \sum_{g \in G} q'_g\, g, \qquad \delta\left(q'_g\, g\right) < \gamma.$$

Damit ist

$$\begin{aligned} f &= \sum_{g \in G} f_g\, g = \sum_{g \in G \setminus J} f_g\, g + \sum_{g \in J} (f_g - \lambda(f_g))\; g + \underbrace{\sum_{g \in J} \lambda(f_g)\; g}_{=q \cdot g} \\ &= \sum_{g \in G \setminus J} f_g\, g + \sum_{g \in J} (f_g - \lambda(f_g))\; g + \sum_{g \in G} q'_g\, g =: \sum_{g \in G} f^1_g\, g \end{aligned}$$

und da alle drei Darstellungen auf der rechten Seite Grad $< \gamma$ haben, erhalten wir, daß

$$\delta(f) \leq \delta\left(f^1_g\, g\right) < \gamma, \qquad g \in G.$$

Nun fahren wir fort und setzen analog $\gamma_1 := \max_g \delta\left(f^1_g\, g\right)$ und wenn $\gamma_1 > \delta(f)$ ist, dann verwenden wir dasselbe Argument wieder und erhalten $\gamma_2 < \gamma_1$ und entsprechende Koeffizienten f^2_g, $g \in G$, und so weiter. Da wir es mit einer *Wohlordnung* zu tun haben, muß dieser Prozess

[163] Die normalerweise keine Γ–Darstellung sein wird.

nach endlich vielen, sagen wir N, Schritten abbrechen und dann muß $\gamma_N = \delta(f)$ sein[164] und wir $f'_g := f_g^N$, $g \in G$, liefert die gewünschte Γ–Darstellung, womit G als Γ–Basis geoutet ist. $\square$

Der Beweis von Proposition 6.25 zeigt uns, daß wir die Aussage dort noch verschärfen können: Schließlich haben wir in (6.14) den Vektor q sogar so gewählt, daß alle Einträge *homogene* Polynome, also nicht nur $q \in \Pi^G$, sondern $q \in (\Pi^0)^G$. Die Forderung $\delta(q \cdot G) < \gamma$ heißt dann aber, daß

$$q \cdot \lambda(G) = \sum_{g \in J} q_g \, \lambda(g) = 0.$$

Und solche Vektoren haben einen Namen.

Definition 6.26 *Sei $F \subset \Pi$ endlich.*

1. *Ein Vektor $s \in \Pi^F$ heißt* Syzygie[165] *bezüglich F, wenn $s \cdot F = 0$.*
2. *Den* Syzygienmodul *von F bezeichnen wir mit*

$$S(F) = \left\{ s \in \Pi^F \ : \ 0 = s \cdot F := \sum_{f \in F} s_f \, f \right\}.$$

Bemerkung 6.27 *Sei $F \subset \Pi$ endlich.*

1. *$S(F)$ ist ein* Modul *über Π, d.h., für $s, s' \in S(F)$ und $g, g' \in \Pi$ ist auch $g\,s + g'\,s' \in S(F)$.*
2. *Der Modul $S(F)$ ist endlich erzeugt, siehe (Gröbner, 1970), das heißt, es gibt eine* endliche *Menge $S \subset S(F)$, so daß*

$$S(F) = \left\{ \sum_{s \in S} q_s \, s \ : \ q_s \in \Pi \right\}.$$

Eine solche Menge S heißt Basis *des Syzygienmoduls.*

Lemma 6.28 *G ist genau dann eine Γ–Basis, wenn $r_G(s \cdot G) = 0$ für alle $s \in S$ gilt, wobei S eine Basis des Syzygienmoduls $S(\lambda(G))$ ist.*

Beweis: Nach Proposition 6.25 ist G genau dann eine Γ–Basis, wenn $r_G(s \cdot G) = 0$ für alle $s \in S(\lambda(F))$ erfüllt ist. Bleibt also nur zu zeigen, daß es hierfür genügt, sich eine Basis S dieses Syzygienmoduls anzusehen.

[164] Denn sonst könnte man den Grad ja nochmals reduzieren!

[165] Gebildet aus den griechischen Worten "$\sigma\upsilon\varsigma$" = "Zusammen" und "$\zeta\upsilon\gamma o\nu$" = "Joch", also etwas "zusammengespanntes". Der Begriff wurde ursprünglich (und bereits bei den Griechen) für Plantenstellungen (lat. "Konjunktionen") verwendet.

Ist $r_G(s \cdot G) \neq 0$ für ein $s \in S$, so kann trivialerweise $r_G(s \cdot G) = 0$, $s \in S(\lambda(F))$, nicht mehr erfüllt sein. Ist umgekehrt

$$S(\lambda(F)) \ni t = \sum_{s \in S} q_s\, s, \qquad q_s \in \Pi,$$

eine beliebige Syzygie von Leittermen, so existiert nach der Annahme, daß jedes Element von s zu Null reduziert die Darstellung

$$t \cdot G = \sum_{g \in G} t_g\, g = \sum_{g \in G} \sum_{s \in S} q_s s_g\, g = \sum_{s \in S} q_s \sum_{g \in G} s_g\, g = \sum_{s \in S} q_s \sum_{g \in G} f_{s,g}\, g = \sum_{g \in G} \underbrace{\sum_{s \in S} q_s f_{s,g}}_{=: f_g}\, g$$

wobei die f_g als Vielfache der Divisionsfaktoren auch vom Divisionsalgorithmus bestimmt werden. Somit ist $r_g(t \cdot G) = 0$. □

Das letzte Lemma liefert uns dann auch schon die Grundidee zur Bestimmung einer Γ–Basis:

1. Wir bestimmen eine endliche Basis S von $S(\lambda(F))$.
2. Ist $r_F(s \cdot F) = 0$ für alle $s \in S$, dann ist F eine Γ–Basis und wir sind fertig.
3. Ansonsten gibt es ein $s^* \in S$, so daß

$$0 \neq f^* := r_F(s^* \cdot F) = \underbrace{s^* \cdot F}_{\in \langle F \rangle} - \underbrace{\sum_{f \in F} g_f\, f}_{\in \langle F \rangle} \quad \in \langle F \rangle.$$

4. Nun ist $\langle F \cup \{f^*\} \rangle = \langle F \rangle$ und

$$r_{F \cup \{f^*\}}(s^* \cdot F) = 0,$$

   wir können also durch vergrößern der Basis F immer die Reduktionsbedingung erzwingen.

5. Also beginnen wir mit $F \cup \{f^*\}$ von vorne.

Das ist eigentlich dann auch schon der berühmte *Buchberger–Algorithmus*, (Buchberger, 1965), der, wie der Name schon sagt, von Bruno Buchberger 1965 angegeben wurde.

Algorithmus 6.29 *(Buchberger–Algorithmus)*
Gegeben: *endliche Menge* $F \subset \Pi$.

1. *Führe die Operationen*
   (a) *Bestimme eine endliche Basis* S *von* $S(\lambda(F))$.

(b) Berechne

$$G \leftarrow \{r_F(s \cdot F) \ : \ s \in S\} \setminus \{0\}.$$

(c) Setze

$$F \leftarrow F \cup G.$$

aus, bis $G = \emptyset$ ist.

Ergebnis: Γ*–Basis* F

Satz 6.30 *Algorithmus 6.29 terminiert nach endlich vielen Schritten und liefert eine* Γ*–Basis.*

Bevor wir diese Aussage beweisen, aus der unser "Fernziel", Satz 6.6, unmittelbar folgt, sollten wir aber erst einmal festhalten, daß sich Algorithmus 6.29 darauf verläßt, daß man eine Basis des Syzygienmoduls bestimmen kann. Dies ist ziemlich schwierig für Nicht–Termordnungen[166], aber einfach für Termordnungen.

Lemma 6.31 *Sei $F \subset \Pi$ endlich und Γ eine Termordnung. Dann wird der Syzygienmodul*[167] *$S(\lambda(F))$ erzeugt von den* S–Polynomen *$s(f, f')$, $f, f' \in F$, $f \neq f'$, deren von Null verschiedene Komponenten durch*

$$s(f,f')_f = \frac{\lambda(f')}{x^\alpha}, \qquad s(f,f')_{f'} = -\frac{\lambda(f)}{x^\alpha}, \qquad \alpha = \min\{\delta(f), \delta(f')\}, \tag{6.15}$$

definiert sind, wobei das Minimum in (6.15) komponentenweise zu verstehen ist.

Die S–Polynome sind eigentlich die *einfachsten* Syzygien, die man sich vorstellen kann, denn es sind lediglich Syzygien zwischen *zwei* Polynomen, die man noch dazu ganz einfach und direkt per Hand ausrechnen kann. Daß sie trotzdem die gesuchte Basis des Syzygienmoduls bilden ist umso erfreulicher.

Beweis von Lemma 6.31: Da sich jede Syzygie in ihre homogenen Komponenten zerlegen läßt, können wir annehmen, daß $s \in S(F)$ eine *homogene* Syzygie ist, das heißt, daß es ein $\beta \in \mathbb{N}_0^n$ gibt, so daß

$$s_f \lambda(f) \in \Pi_\beta, \qquad \text{also} \qquad s_f \lambda(f) = c_j x^\beta, \qquad f \in F.$$

Ist nun $s \in S(F)$ eine nichttriviale Syzygie, dann gibt es mindestens *zwei* Polynome $f, f' \in F$, so daß $s_f \cdot s_{f'} \neq 0$. Das heißt aber insbesondere, daß

$$\beta \in (\delta(f) + \mathbb{N}_0^n) \cap (\delta(f') + \mathbb{N}_0^n) \subset \alpha + \mathbb{N}_0^n.$$

[166]Ein Methode zur Bestimmung einer Basis des Syzygienmoduls einer *beliebigen* Menge $F \subset \Pi$ von Polynomen ist z.B. in (Buchberger, 1985) angegeben. Der Wermutstropfen dabei ist aber, daß man dafür erst mal eine Gröbnerbasis berechnen muß. Prinzipiell reicht das zwar, denn wie's für Gröbnerbasen geht werden wir gleich sehen, aber algorithmisch und prinzipiell ist das doch sehr unbefriedigend: man kann wohl kaum sagen, daß beispielsweise H–Basen eine tolle Verallgemeinerung von oder vielleicht sogar ein Ersatz für Gröbnerbasen sind, wenn man zu ihrer Berechnung Gröbnerbasen benötigt.

[167]Nicht vergessen: Was uns interessiert ist nicht $S(F)$ sondern $S(\lambda(F))$, also der Syzygienmodul der *Leitterme* von F.

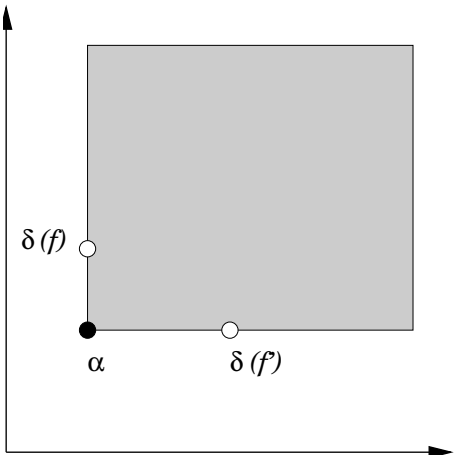


Abbildung 6.3: Die geometrische Interpretation von α in (6.15) als größter Multiindex, der einen Kegel erzeugt, in dem sowohl $\delta(f)$ als auch $\delta\,(f')$ liegt. Damit kann man x^α als ggT $(\lambda\,(f)\,,\lambda\,(f'))$ auffassen.

Es gibt also einen Term $g' \in \Pi_{\beta-\delta(f')}$, so daß

$$s_f\,\lambda(f) = g'\,\lambda\,(f') = d\,x^\beta, \qquad d \in \mathbb{K},$$

und damit ist

$$s\cdot\lambda(F) = \sum_{h\in F\setminus\{f\}} s_h\lambda(h) + \underbrace{s_f\lambda(f) - g'\,\lambda\,(f')}_{=g\,s_{f,f'}} + g'\,\lambda\,(f') =: s'\cdot\lambda(F) + (g\,s_{f,f'})\cdot\lambda(F)$$

und damit ist $s' = s - g\,s_{f,f'} \in S(F)$ und hat einen von Null verschiedenen Eintrag weniger. Durch Iteration dieser Vorgehensweise erhält man dann die Zerlegung in S–Polynome. □

Um zu beweisen, daß der Buchberger–Algorithmus auch wirklich terminiert schnell *noch* eine Beschreibung der Γ–Basen. Dazu benötigen wir den Begriff des *homogenen Ideals* als ein Ideal $\mathscr{H}$, zu dem mit $f \in \mathscr{H}$ auch alle *homogenen Bestandteile* von f zum Ideal gehören:

$$\mathscr{H} \ni f = \sum_{\gamma\in\Gamma} f_\gamma \qquad \Rightarrow \qquad f_\gamma \in \mathscr{H}, \quad \gamma \in \Gamma,$$

siehe (Gröbner, 1970). Besitzt $\mathscr{H}$ eine Basis $F \subset \Pi^0$ von *Formen*[168], also homogenen Polynomen, dann ist $\mathscr{H}$ offensichtlich ein homogenes Ideal. Ist umgekehrt $\mathscr{H}$ ein homogenes Ideal und F eine Basis von $\mathscr{I}$, dann gehört auch die Menge

$$F^0 := \{f_\gamma \;:\; f \in F,\, \gamma \in \Gamma\} \subset \Pi^0$$

zu $\mathscr{H}$ und bildet damit eine *homogene Basis* oder *Formenbasis* von $\mathscr{H}$. Fazit:

[168] Daher auch der ältere Name "Formenideal".

Lemma 6.32 *Ein Ideal $\mathscr{H} \subset \Pi$ ist genau dann ein homogenes Ideal, wenn es eine Basis aus homogenen Polynomen hat.*

Im Gegensatz zum "normalen" Ideal[169] bilden die homogenen Polynome im Ideal auch eine Art Ideal, bei der man nur mit *homogenen* Polynomen multipliziert und homogene Polynome gleichen Grades addiert.

Lemma 6.33 *Sei $\mathscr{I} = \langle G \rangle \subset \Pi$ ein Ideal. Dann gilt:*

1. *Die Ideale $\langle \lambda(G) \rangle$ und $\langle \lambda(\mathscr{I}) \rangle$ sind homogene Ideale.*
2. *G ist genau dann eine Γ–Basis für $\mathscr{I}$, wenn $\langle \lambda(\mathscr{I}) \rangle = \langle \lambda(G) \rangle$ ist.*

Beweis: 1. Beide Ideale werden von homogenen Polynomen erzeugt und sind daher nach Lemma 6.32 homogen.
2. ist auch nicht viel schlimmer: Ist G eine Γ–Basis und hat $f \in \mathscr{I}$ die Γ–Darstellung

$$f = \sum_{g \in G} f_g \, g,$$

dann ist

$$\lambda(f) = \sum_{\{g \,:\, \delta(f_g \, g) = \delta(f)\}} \lambda\left(f_g\right) \, \lambda(g) \quad \in \langle \lambda(G) \rangle.$$

Ist umgekehrt $f \in \mathscr{I}$ und $\lambda(f) = h \cdot \lambda(G)$, dann gehört $f - h \cdot G$ ebenfalls zu $\mathscr{I}$, hat aber niedrigeren Grad. Auf diese Weise erhält man die homogenen Komponenten einer Γ–Darstellung von f. □

Beweis von Satz 6.30 und damit Satz 6.6: Daß wir eine Γ–Basis gefunden haben, wenn der Buchberger–Algorithmus terminiert, folgt sofort aus Proposition 6.25, die ja gerade Γ–Basen dadurch charakterisiert hat, daß alle Syzygien zu null reduziert werden.
Bleibt also die Terminierung. Ist $0 \neq g := r_F\left(s \cdot F\right)$, dann heißt das ja insbesondere, daß $\lambda(g) \in W_{\delta(g)}(F)$, also $g \in \langle F \rangle$, aber $\lambda(g) \notin \langle \lambda(F) \rangle$. Ersetzen wir jetzt also F durch $F' = F \cup \{g\}$, dann ist $\langle F' \rangle = \langle F \rangle$, aber $\langle \lambda\left(F\right) \rangle$ ist eine *echte* Teilmenge von $\langle \lambda\left(F'\right) \rangle$. Solch eine aufsteigende Kette von homogenen Idealen muß aber[170] nach endlich vielen Schritten abbrechen und dann kann es kein $s \in S(F)$ mehr geben, so daß $r_F\left(s \cdot F\right) \neq 0$, denn sonst könnten wir das homogene Ideal ja nochmal vergrößern.
Der langen Rede kurzer Sinn: Der Buchberger–Algorithmus erweitert jede endliche Basis eines Ideals in endlich vielen Schritten zu einer endlichen Γ–Basis desselben Ideals. □

Bemerkung 6.34 *Natürlich ist "unsere" Version des Buchberger–Algorithmus, Algorithmus 6.29, in keinster Weise effektiv und schon gleich gar nicht optimal. Mal ganz abgesehen davon, daß man nicht* alle *S–Polynome braucht, sondern nur die Hälfte, denn $s\left(f, f'\right)$ und $s\left(f', f\right)$ sind ja dasselbe, ist die geschickte Wahl und Verwaltung der S–Polynome eines der zentralen*

[169] Wahrscheinlich hat "normales Ideal" noch eine ganz andere Bedeutung.
[170] *Noetherscher Ring* und so . . ..

algorithmischen *Probleme bei der Berechnung von Gröbnerbasen*[171]*. Was man sich aber leicht überlegen kann, ist die Tatsache, daß es bestimmt nicht clever ist, Redundanzen mit sich herumzuschleppen. Deswegen nennt man eine Γ–Basis G*

1. minimal*, wenn für jedes $g \in G$ die Menge $G \setminus \{g\}$ keine Γ–Basis mehr ist,*
2. reduziert*, wenn*

$$r_{G\setminus\{g\}}(g) = g, \qquad g \in G.$$

So eine (minimale) reduzierte Γ–Basis kann man immer erhalten, indem man zuerst alle Redundanzen rauswirft[172] *und dann jedes $g \in G$ durch $r_{G\setminus\{g\}}(g)$ ersetzt.*
Man kann zeigen, daß es zu jeder Termordnung bis auf Normierung genau eine reduzierte Gröbnerbasis gibt, z.B. (Cox et al.*, 1996), was aber für allgemeine Γ–Basen* nicht *mehr gilt. In (Möller & Sauer, 2000a) sind beispielsweise alle reduzierten H–Basen für ein vorgegebenes Ideal charakterisiert.*

[171] Die Bestimmung allgemeinerer Γ–Basen ist bisher noch nirgendwo implementiert, was nicht sonderlich verwunderlich ist, sind sie doch ziemlich neuen Datums.

[172] Ein Element g einer Γ–Basis G ist ja nun wieder genau dann redundant, wenn $r_{G\setminus\{g\}}(g) = 0$. Das heißt aber insbesondere, daß jede reduzierte Γ–Basis auch minimal ist.

Let me see, then, what thereat is, and this mystery explore [. . .] Quoth the Raven "Nevermore"

E. A. Poe, *The Raven*

Lösen von Gleichungssystemen 7

So, jetzt also zurück zu unserem Problem das Gleichungssystem $F(X) = 0$ zu lösen, oder, vornehmer ausgedrückt, die Varietät

$$\mathscr{V}(F) = \mathscr{V}(\langle F\rangle) = \left\{x \in \overline{\mathbb{K}} \ : \ F(x) = \langle F\rangle(x) = \{0\}\right\}$$

zu bestimmen. Um uns das Leben leichter zu machen und weil uns für weitergehende Untersuchungen die Zeit fehlt[173], beschränken wir uns auf den Fall, daß (äquivalent)

- die Varietät $\mathscr{V}(F)$ nur aus *endlich vielen* Punkten besteht.
- der Quotientenraum
$$\Pi/\langle F\rangle \simeq \nu_{\langle F\rangle}(\Pi)$$
endlichdimensional ist.
- $\langle F\rangle$ ein *nulldimensionales* Ideal ist.

Wir verwenden diese Äquivalenz hier als *Definition* eines nulldimensionalen Ideals; intuitiv entspricht ein eindimensionales Ideal einer (algebraischen) Kurve, ein zweidimensionales Ideal einer Fläche und so weiter. Eine Warnung ist hier allerdings angebracht: Die Anzahl der Punkte in $\mathscr{V}(F)$ ist nicht automatisch gleich der Dimension von $\Pi/\langle F\rangle$, denn Punkte können ja *Vielfachheiten* haben – man denke nur an $n = 1$ und $f(x) = x^k$, da ist $\mathscr{V}(F) = \{0\}$, aber $\Pi/\langle F\rangle = \Pi_{n-1}$. Wir werden aber in Abschnitt 7.3 sehen, wie man dieses Problem in den Griff bekommt, das heißt, wie man Vielfachheiten bestimmt oder, vornehmer gesagt, wie man von $\langle F\rangle$ zum *Radikal*

$$\sqrt{\langle F\rangle} = \left\{f \in \Pi \ : \ f^k \in \langle F\rangle \text{ für ein } k \in \mathbb{N}\right\}.$$

Die Gleichung $\sqrt{F}(X) = 0$ hat dieselbe Lösung X wie $F(X) = 0$, aber die Nullstellen sind jetzt nur noch *einfache* Nullstellen.

Diese Umformung an sich kann schon sehr hilfreich sein, wenn man beispielsweise iterative Verfahren wie das *Newton–Verfahren*[174] (Sauer, 2000b) verwenden will. Dieses Verfahren

[173] Ist das nicht eine tolle Ausrede?

[174] Das ist **das** iterative Verfahren zum Lösen nichtlinearer Gleichungen solange die Anzahl der Gleichungen und die Anzahl der Variablen übereinstimmen. Da man dafür die Jacobimatrix von F braucht, sind Polynome natürlich dankbare Kandidaten, denn die kann man ja relativ leicht ableiten und auswerten.

ist sehr empfindlich gegen mehrfache Nullstellen, genauer gesagt, es konvergiert eigentlich gar nicht gegen mehrfache Nullstellen, und da is es schon sehr hilfreich, wenn man das Gleichungssystem so manipulieren kann, daß einfache Nullstellen garantiert sind. Aber mehr dazu später.

Wir wollen jetzt die wesentlichen Methoden betrachten, wie wir durch idealtheoretische Manipulationen ein Gleichungssystem so umformen können, daß wir es einfacher lösen können.

7.1 Eliminationsideale

Die Idee der Eliminationsideale geht bereits auf Kronecker zurück und basiert auf der "naheliegenden" Idee, nach einzelnen Variablen aufzulösen und diese zu eliminieren, bis man nur noch eine Gleichung in einer Variablen übrigbehält. Hier werden die Gröbnerbasen bezüglich einer lexikografischen Termordnung eine entscheidende Rolle spielen und das war auch die "ursprüngliche" Anwendung der Gröbnerbasen, siehe (Trinks, 1978).

Definition 7.1 *Es sei $\mathscr{I}$ ein Ideal in Π. Für $k = 1, \ldots, n$ heißt*

$$\mathscr{I}_k := \mathscr{I} \cap \mathbb{K}[x_1, \ldots, x_k]$$

das k–te Eliminationsideal *von $\mathscr{I}$.*

Die Bestimmung der Eliminationsideale ist ermöglicht es dann, das Gleichungssystem auf "altväterliche" Art und Weise zu lösen:

- Das Ideal $\mathscr{I}_1$ wird von einem univariaten Polynom erzeugt. Die Nullstellen dieses Polynoms sind die x_1–Komponenten aller Lösungen von $F(X) = 0$.
- Für jede dieser Lösungen betrachtet man dann das Ideal $\mathscr{I}_2$ und setzt die Lösung für x_1 ein. Das gibt dann wieder "nur" univariate Polynome und man kann wieder die (möglicherweise leere) Menge aller x_2–Komponenten zu jeder Wahl für x_1 berechnen.
- Diese Paare setzt man dann in $\mathscr{I}_3$ ein, berechnet Lösungstripel und so weiter und so weiter.
- Da $\mathscr{I} = \mathscr{I}_n$ hat man so am Ende alle Lösungen bestimmt.

Mit anderen Worten: Wir brauchen "nur" noch eine Methode, Basen für die Eliminationsideale aus einer Basis für $\mathscr{I}$ zu bestimmen. Wie bekommen wir nun sowas? Nun, intuitiv könnte man sagen, daß wenn wir mit unserem Gradbegriff x_n und seinen Potenzen einen möglichst *großen* Grad zuordnen, dann wird unsere entsprechende Γ–Basis versuchen, zuerst so viele Elemente kleineren Grades (also in $x_1, \ldots, x_{n-1}$) zu bestimmen wie möglich und erst dann, wenn es nicht mehr anders geht, Potenzen von x_n hinzunehmen. Und so einen Gradbegriff kennen wir: Die *lexikografische* Termordnung.

Satz 7.2 *Sei $\mathscr{I} \subset \Pi$ ein Ideal und G eine Gröbnerbasis von $\mathscr{I}$ bezüglich der lexikografischen Termordnung mit $x_1 < x_2 < \cdots < x_n$. Dann ist*

$$\mathscr{I}_k = \langle G \cap \mathbb{K}[x_1, \ldots, x_k] \rangle \qquad \textit{in } \mathbb{K}[x_1, \ldots, x_k].$$

Beweis: Wir setzen $G_k = G \cap \mathbb{K}[x_1, \ldots, x_k]$ und wählen ein $k \in 1, \ldots, n$. Sei also $f \in \mathscr{I}_k \subseteq \mathscr{I}$, dann hat f, weil ja G eine Gröbnerbasis von $\mathscr{I}$ eine G–Darstellung

$$f = \sum_{g \in G} f_g \, g, \qquad \delta(f_g \, g) \leq \delta(f). \tag{7.1}$$

Ist nun aber $g \in G \setminus G_k$, dann enthält g (mindestens) eine der Variablen $x_{k+1}, \ldots, x_n$ und damit ist $\delta(g) > \delta(f)$ und das zugehörige f_g in der G–Darstellung (7.1) muß 0 sein. Damit ist also

$$f = \sum_{g \in G_k} f_g \, g$$

und mit demselben Argument wie oben folgt aus $\delta(f_g) < \delta(f)$, daß $f_g \in \mathbb{K}[x_1, \ldots, x_k]$ sein muß und damit hat f also eine G–Darstellung in $\mathbb{K}[x_1, \ldots, x_k]$ bezüglich G_k. Und genau das war die Behauptung. □

Und damit wird die Bestimmung der Eliminationsideale, also "im Prinzip" auch die Lösung des Gleichungssystems $F(X) = 0$ zurückgeführt auf die Bestimmung einer Gröbnerbasis bezüglich der lexikografischen Termordnung. Buchberger selbst bezeichnet dieses Verfahren übrigens als

> "Kombination von euklidischem Algorithmus und Gauß–Elimination."

Beispiel 7.3 *(Fortsetzung von Beispiel 5.2)*
Im Falle unserer beiden Ellipsen ist es nicht schwer, die lexikografische Gröbnerbasis als

$$\begin{aligned} g_1(x) &= \left(\cos^2\varphi + 8\right)\, x^4 + 6\left(\cos^2\varphi - 4\right)\, x^2 + 9\cos^2\varphi \\ g_2(x,y) &= -4\cos\varphi\, \sin\varphi\, y - \frac{\cos^2\varphi + 8}{3}\, x^3 + \left(8 - 5\cos^2\varphi\right)\, x \end{aligned}$$

zu bestimmen. Beim Lösen und Rücksubstituieren mit 10 *Dezimalstellen Genauigkeit erhält man dann das beeindruckende Ergebnis*

φ	x_1/x_3	x_2/x_4
10^{-1}	[-.933368, -1.031703]	[.933368, 1.031703]
	[-1.066627, .964963]	[1.066627, -.964963]
10^{-2}	[-.933368, -1.031703]	[.933368, 1.031703]
	[-1.066627, .964963]	[1.066627, -.964963]
10^{-3}	[-.993333, -1.003317]	[.993333, 1.003317]
	[-1.006667, .996651]	[1.006667, -.996651]
10^{-4}	[-.999333, -1.000334]	[.999333, 1.000334]
	[-1.000667, .999667]	[1.000667, -.999667]
10^{-5}	[-.999933, -1.000034]	[.999933, 1.000034]
	[-1.000067, .999969]	[1.000067, -.999969]
10^{-6}	[-.999991, -1.369300]	[.999991, 1.369300]
	[-1.000009, 1.369350]	[1.000009, -1.369350]
10^{-7}	[1.000000, 0.000000]	[1.000000, 0.000000]
	[-1.000000, 0.000000]	[-1.000000, 0.000000]

Das heißt: Je kleiner die Störung wird, desto bescheidener wird das Ergebnis der Rechnung!

Läßt sich dieses Desaster denn nun auch irgendwie erklären? Allerdings, denn betrachtet man einmal wie y in $g_2(x, y)$ von x abhängt, dann erhält man, daß

$$y = \frac{1}{4 \sin\varphi \, \cos\varphi} \left(\frac{\cos^2\varphi + 8}{3} \, x^3 - \left(8 - 5\cos^2\varphi\right) \, x \right),$$

und der Term $\sin\varphi$ im Nenner sorgt für eine *dramatische* Fehlerverstärkung.
Damit hat die Idee der Eliminationsideale einige gravierende Nachteile:

1. Die lexikografische Gröbnerbasis ist, das kann man zeigen, von allen Gröbnerbasen am schwierigsten zu bekommen, ihre Bestimmung erfordert den höchsten Aufwand.

2. "Das" univariate Polynom in $\mathscr{I}_1$, das ja den "Startpunkt" für das "Knacken" des Gleichungssystems darstellt, wird normalerweise sehr hohen Grad haben: Haben beispielsweise alle Lösungen verschiedene x_1–Koordinaten, was sozusagen der "Normalfall" ist, dann ist der Grad dieses Polymoms gleich der Anzahl dieser Lösungen. Damit wird die Nullstellenbestimmung mit *symbolischen* Methoden normalerweise unmöglich[175] und die *numerische* Nullstellenbestimmung schwierig, instabil und ungenau.

3. Und weil das noch nicht genug ist, kann das Einsetzen auch noch zu richtig solider Fehlerverstärkung führen, siehe Beispiel 7.3.

4. Und das ist kein isolierter Fall! Es ist eine wohlbekannte und klassische Tatsache, siehe z.B. (Wilkinson, 1984) oder (Farouki & Rajan, 1987), daß die Nullstellen eines Polynoms auf recht empfindliche Art und Weise von den Koeffizienten abhängen, vor allem dann, wenn der Grad des Polynoms nicht mehr allzu klein ist.

[175] Und liefert, beispielsweise in Maple, die allseits beliebten `RootOf`–Ausdrücke.

Wir brauchen also etwas anderes . . .

7.2 Eigenwertmethoden

Wir gehen nun die Lösung des Problems $F(X) = 0$ ganz anders an und führen sie auf die Lösung eines *Eigenwertproblems* zurück. Diese Idee, die von Stetter (Stetter, 1995) aufgebracht wurde[176], siehe auch (Möller & Stetter, 1995), verwendet das multivariate Gegenstück zu den sogenannten *Frobenius–Begleitmatrizen*, siehe z.B. (Sauer, 2000b). Allerdings funktionieren diese Ansätze bisher nur für *nulldimensionale* Ideale, also für Gleichungssysteme, die nur *endlich viele* Lösungen haben.
Um die Idee vernünftig und einfacher erklären zu können, nehmen wir jetzt sogar an, daß $\langle F\rangle$ ein *nulldimensionales Radikal* ist, daß wir also nicht nur endlich viele Lösungen von $F(X)$ haben, sondern daß alle diesen gemeinsamen Nullstellen von F obendrein *einfache* Nullstellen sind. Unter diesen Voraussetzungen können wir ein klein wenig multivariate Polynominterpolation betreiben.

Proposition 7.4 *Sei $\mathscr{I} := \langle F\rangle$ ein nulldimensionales Radikal, $X = \mathscr{V}(\mathscr{I})$ die zugehörige Varietät, also $\langle F\rangle = I(X)$ und sei Γ ein Graduierungsmonoid.*

1. *Der Vektorraum $N_F = \nu_{\langle F\rangle}(\Pi) \subset \Pi$ ist ein Interpolationsraum, das heißt, für jedes Polynom $g \in \Pi$ gibt es genau ein $h \in N_F$ so daß $g(X) = h(X)$.*

2. *Für jedes $x \in X$ gibt es genau ein Polynom $\ell_x \in N_F$, so daß*

$$\ell_x(x') = \delta_{x,x'}, \qquad x, x' \in X.$$

3. *Für jedes $f : X \to \mathbb{K}$ ist dann*

$$L_F f := \sum_{x\in X} f(x)\,\ell_x$$

   das zugehörige Interpolationspolynom in N_F.

4. *Die Polynome ℓ_x, $x \in X$, bilden eine Basis von N_F.*

Beweis: Sei G eine Γ–Basis von $\langle F\rangle$ bezüglich des Graduierungsmonoids Γ. Wegen der Linearität der orthogonalen Projektionen ist die Abbildung $r_G : \Pi \to N_F$ eine lineare Projektion und da $\nu_{\langle F\rangle}$ als r_G definiert ist, also auch $\nu_{\langle F\rangle}$.

[176] Der sich in den Jahren kurz vor seiner Pensionierung noch mal komplett umorientierte, nämlich von gewöhlichen Differentialgleichungen zu Gröbnerbasen und der als einer der ersten auf die Probleme bei der algebraisch–numerischen Behandlung von polynomialen Gleichungssystemen hinwies. Das gibt's inzwischen auch als Buch, (Stetter, 2005)

Für jedes $h \in \Pi$ gilt $h - \nu_{\langle F\rangle}(h) \in \mathscr{I} = I(X)$ und daher ist $\left(h - \nu_{\langle F\rangle}(h)\right)(X) = 0$. Hätten außerdem $h, h' \in N_F$ die Eigenschaft, daß $(h - h')(X) = 0$, dann ist

$$0 = r_G(h - h') = \nu_{\langle F\rangle}(h - h') = \underbrace{\nu_{\langle F\rangle}(h)}_{=h} - \underbrace{\nu_{\langle F\rangle}(h')}_{=h'},$$

also $h = h'$ woraus die Eindeutigkeit des Interpolationspolynoms, also auch 1. folgt.
Für 2. definieren wir einfach für $x \in X$ die Polynome

$$\ell_x = \nu_{\langle F\rangle}\left(\prod_{x' \neq x} \frac{v_{x,x'}^T(\cdot - x')}{v_{x,x'}^T(x - x')}\right), \qquad v_{x,x'} \in \mathbb{K}^n, \quad v_{x,x'}^T(x - x') \neq 0, \tag{7.2}$$

und diese Polynome interpolieren *eindeutig* $\delta_{x,x'}$. Darüberhinaus ist 3. offensichtlich und 4. folgt aus der Tatsache, daß jedes $h \in N_F$ die Form

$$h = L_F h = \sum_{x \in X} h(x)\,\ell_x$$

hat. □

Und jetzt kehren wir zurück an den Anfang der Gröbnerbasenstory, nämlich zu der Aufgabe, die Buchberger seinerzeit (von Gröbner) gestellt wurde:

Die Bestimmung der Multiplikationstafeln für nulldimensionale Ideale.

Was zum ... ist das nun schon wieder? Nun, zu einem gegebenen $h \in \Pi$ betrachten wir die Abbildung $\Phi_h : N_F \to N_F$, gegeben durch

$$\Phi_h f = \nu_{\langle F\rangle}(f \cdot h), \qquad f \in N_F, \tag{7.3}$$

also die Multiplikation mit h gefolgt von der Bildung der Normalform. Für festes $h \in \Pi$ und $f, f' \in N_F$ sowie $c, c' \in \mathbb{K}$ ist nun

$$\begin{aligned}
\Phi_h(c\,f + c'\,f') &= \nu_{\langle F\rangle}(h \cdot (c\,f + c'\,f')) = \nu_{\langle F\rangle}(h\,c\,f + h\,c'\,f') \\
&= \nu_{\langle F\rangle}(h\,c\,f) + \nu_{\langle F\rangle}(h \cdot c'\,f') = c\,\nu_{\langle F\rangle}(f \cdot h) + c'\,\nu_{\langle F\rangle}(f' \cdot h) \\
&= c\,\Phi_h f + c'\,\Phi_h f',
\end{aligned}$$

Φ_h ist also eine *lineare* Abbildung von N_F in sich.

Definition 7.5 *Sei $P = (p_x : x \in X)$ eine Basis*[177] *von N_F und $h \in \Pi$. Dann bezeichnen wir die Matrix $M(h) = M(h, P) \in \mathbb{K}^{X \times X}$, definiert durch*

$$\Phi_h p_x = \sum_{x' \in X} M(h)_{x,x'}\,p_{x'}, \qquad x \in X,$$

als Multiplikationstafel *für die Multiplikation mit h auf N_F bezüglich der Basis P.*

[177] Achtung: Die Polynome p_x müssen *nichts* mit dem Punkt x zu tun haben! Es ist und bleibt lediglich eine "Numerierung".

Wir halten zuerst fest, daß die Multiplikationstafel eine *lineare* Funktion in h ist, daß also

$$M\left(c\,h + c'\,h'\right) = c\,M\left(h\right) + c'M\left(h'\right), \qquad h, h' \in \Pi, \quad c, c' \in \mathbb{K}, \tag{7.4}$$

ist.
Und jetzt kommt der Clou des Ganzen:

Satz 7.6 *Die Eigenwerte und zugehörigen Eigenvektoren der Multiplikationstafel $M(h)$ sind $h(x)$ und ℓ_x, $x \in X$.*

Beweis: Für $x \in X$ und $h \in \Pi$ ist

$$\begin{aligned}\Phi_{h-h(x)}\ell_x &= \Phi_h\ell_x - h(x)\,\ell_x = \nu_{\langle F\rangle}\left(h\,\ell_x\right) - h(x)\,\ell_x = \sum_{x'\in X} \underbrace{\left(h\,\ell_x\right)\left(x'\right)}_{=\delta_{x,x'}h(x')}\,\ell_{x'} - h(x)\,\ell_x \\ &= h(x)\,\ell_x - h(x)\,\ell_x = 0,\end{aligned}$$

was uns, wie gewünscht, $\Phi_h\ell_x = h(x)\,\ell_x$ liefert. □

Bemerkung 7.7 *Ein wichtiger Aspekt von Satz 7.6 ist die Tatsache, daß die Eigen*vektoren *der Matrizen $M(h)$ von dem Polynom h* unabhängig *sind – es sind immer die Polynome ℓ_x, $x \in X$. Ohne diese Tatsache wäre es nämlich unmöglich, aus den "individuellen" Eigenwertproblemen die Gesamtlösung zu kombinieren! Wir könne ja nach Satz 7.6 aus den Matrizen $M_j := M\left(x_j\right)$, $j = 1, \ldots, n$, die Vektoren $X_j = \left(x_j : x \in X\right) \in \overline{\mathbb{K}}^X$ der Koordintenprojektionen bestimmen*[178]*, aber welche Komponenten dieser Vektoren zusammengehören, das kann man eben nur über die zugehörigen Eigenvektoren entscheiden.*

Übrigens, nur zur Erinnerung: Die Eigenwerte der Matrizen M_j sind auch unabhängig *von der gewählten Basis P, denn ein Basiswechsel ist nur eine Ähnlichkeitstransformation und der sind Eigenwerte vollkommen egal.*

Das liefert uns nun ein neues Verfahren, um $F(X) = 0$ zu lösen, nämlich indem man das Problem, wie von Stetter vorgeschlagen, in ein Eigenwertproblem transformiert. Dazu bemerken wir zuerst, daß die *Nulldimensionalität* von $\langle F\rangle$ insbesondere bedeutet, daß $W(X) = \nu_{N_F}\left(\Pi\right)$ endlichdimensional ist[179] auch, daß es ein $\gamma^* \in \Gamma$ gibt, so daß $V_\gamma(F) = \Pi_\gamma$, $\gamma \geq \gamma^*$, gilt[180]und somit kann man eine Basis von $N_F = W(F)$ durch sukzessive Bildung von Komplementen von $V_\gamma(F)$ in Π_γ, $\gamma < \gamma^*$, recht einfach berechnen. Damit erhalten wir das folgende Lösungsverfahren:

1. Bestimme eine Basis P von N_F.

[178] Wer noch nie Erfahrung mit Eigenwertverfahren gemacht: Den halbwegs effizienten unter ihnen ist es egal, in welcher Reihenfolge sie die Eigenwerte liefern und das hängt normalerweise von der Ausgangsmatrix ab.

[179] Das folgt daraus, daß für $g \in W(F)$ die Identität $g = \nu_{\langle F\rangle}(g)$ gilt und somit $\nu_{\langle F\rangle}$ eine (surjektive) Projektion *auf* $W(F)$ ist.

[180] Eigentlich ist sogar $V_\gamma \neq \Pi_\gamma$ nur für endlich viele $\gamma \in \Gamma$, das ist für die H–Graduierung dasselbe aber beispielsweise im Fall der lexikographischen Termordnung entschieden mehr.

2. Berechne die Multiplikationstafeln $M_j = M(x_j)$ der Koordinatenpolynome[181] bezüglich P.

3. Bestimme die Eigenwerte von M_j und "verbinde" sie über die zugehörigen Eigenvektoren.

Und jetzt, endlich, liefert uns unsere Verallgemeinerung auch einen echten Gewinn:

1. Die mittels allgemeiner, termordnungsfreier H–Basen erhaltenen Eigenwertprobleme sind normalerweise stabiler und insensitiver gegen Störungen. Beispielsweise enthalten bei Verwendung von Gröbnerbasen die Multiplikationstafeln von Beispiel 5.2 Werte, die für $\varphi \to 0$ gegen ∞ divergieren (um das auszugleichen, werden gleichzeitig Spalten linear abhängig!), was bei Verwendung der termordnungsfreien H–Basis nicht passiert, siehe (Möller & Sauer, 2000c).

2. Im Standardfall der "complete intersection" (das heißt, wenn der Satz von Bézout zutriff), in dem $\#F = n$ ist und

$$\{x \in \mathbb{K}^n \ : \ \lambda(F)(x) = 0\} = \{0\},$$

ist F bereits *automatisch* eine H–Basis und man kann sofort mit der Bestimmung des Quotientenraums und der Multiplikationstafeln beginnen, siehe (Möller & Sauer, 2000a).

7.3 Bestimmung des Radikals

Jetzt haben wir's also fast geschafft, unser Problem $F(X) = 0$ zu lösen. Was aber tun, wenn das Originalproblem *mehrfache* Nullstellen enthält? Mehrfache Nullstellen fürchten Numeriker normalerweise wie der sprichwörtliche Teufel das nicht minder sprichwörtliche Weihwasser – in der Tat setzen die meisten Iterationsverfahren[182] voraus, daß die Ergebnisse "einfach" sind; ansonsten braucht man normalerweise ziemlich trickreiche Verfahren, um die Vielfachheiten in den Griff zu bekommen. Die Eigenwertstrukturen von Multiplikationstafeln wurden in (Möller & Stetter, 1995) beschrieben – die Vielfachheiten der Nullstellen werden zu Vielfachheiten der Eigenwerte und die Struktur der Vielfachheit[183] überträgt sich auf die Struktur des zugehörigen invarianten Raums, aber meistens in einer Weise, die Eigenwertverfahren nicht so sehr mögen. Deswegen wollen wir versuchen, die *Vielfachheiten* der Nullstellen lozuwerden.

In unserer algebraischen Sprechweise heißt das dann, daß wir anstelle des Ideals $\langle F \rangle$ sein *Radikal*

$$\sqrt{\langle F \rangle} = \left\{ g \in \Pi \ : \ g^k \in \langle F \rangle, \, k \geq k_0 \right\}$$

[181] Hier täte es auch jedes andere System von Polynomen aus denen sich die Komponenten von x rekonstruieren lassen. Eine (in manchen Fällen vielversprechende) Idee besteht zum Beispiel darin, n linear unabhängige lineare Polynome zu wählen und so "Vielfachheiten" bei den x- oder y–Koordinaten zu vermeiden, die dem Eigenwertverfahren Schwierigkeiten bereiten könnten, siehe auch Beispiel 5.2.

[182] Beispielsweise Newton für die Lösung nichtlinearer Gleichungssysteme oder auch das QR–Verfahren zur Bestimmung von Eigenwerten.

[183] Im Gegensatz zum univariaten Fall ist die Vielfachheit der Nullstelle eine multivariaten Funktion nicht mehr nur eine Zahl sondern eine strukturelle Größe, genauer, ein endlichdimensionaler, differentiationsinvarianter Polynomraum.

auf Nullstellen abklopfen müssen. Aber woher nehmen? Amüsanterweise helfen uns wieder die Multiplikationstafeln (d.h., im wesentlichen die Division mit Rest), diesmal über die *Trace–Methode* aus (Gonzales-Vega *et al.*, 1999a).

Definition 7.8 *Sei P eine Basis von $N_F = \nu_{\langle F\rangle}(\Pi)$, $\langle F\rangle$ nulldimensional und sei X die Lösung von $F(X) = 0$.*

1. *Für $x \in X$ bezeichne $\mu(x)$ die* Vielfachheit *der Nullstelle x, d.h.,*

$$\mu(x) = \dim\{q \in \Pi \ : \ (q(D)f)(x) = 0,\ f \in \langle F\rangle\}. \tag{7.5}$$

2. *Für $h \in \Pi$ definieren wir die* Trace–Matrix $T(h) \in \mathbb{K}^{P\times P}$ *als*

$$T(h) = [\text{trace}\, M\,(h \cdot p \cdot p') \ : \ p, p' \in P]. \tag{7.6}$$

Bemerkung 7.9 *Die Definition (7.5) des Begriffs der Vielfachheit als skalarer Wert gibt die Dimension der Vielfachheit als* strukturelle *Größe an. Die Vielfacheit einer Nullstelle eines Ideals ist nämlich der differentiationsinvariante Vektorraum*

$$\mathscr{Q}_x(F) = \{q \in \Pi \ : \ q(D)f(x) = 0,\ f \in \langle F\rangle\},$$

siehe (Boor & Ron, 1991; Marinari et al.*, 1996).* Differentiationsinvariant *bedeutet hierbei, daß mit q auch alle Ableitung von q zu $\mathscr{Q}_x(F)$ gehören.*
Zwischen skalarer und struktureller Vielfachheit besteht durchaus ein Unterschied: Während an einfachen Nullstellen immer der Funtionswert Null sein muss und an doppelten Nullstellen Funktionswert und eine Richtungsableitung verschwinden, gibt es bei dreifachen Nullstelle zwei Möglichkeiten. Entweder es verschwinden zwei linear unabhängige Richtungsableitungen oder es verschwinden eine erst und eine zweite Richtungsableitung, diese aber dann in dieselbe *Richtung.*

Übung 7.1 Zeigen Sie: Ist $\#F = 1$, dann hat jede Nullstelle immer mindestens Vielfachheit n. ♢

Was bringt uns nun die Trace–Matrix? Nun, betrachten wir einen Vektor $v = (v_p \ : \ p \in P)$ und das zugehörige Polynom $q = v \cdot P \in N_F$, dann erhalten wir unter Berücksichtigung von (7.4) und der Tatsache, daß die Spur einer Matrix die Summe der Eigenwerte ist, für $v = (v_p \ : \ p \in P) \in \mathbb{K}^P$ die Rechnung

$$\begin{aligned} v^H T(h)\, v &= \sum_{p,p'\in P} T(h)_{p,p'}\overline{v_p} v_{p'} = \sum_{p,p'\in P} (\text{trace}\, M\,(h\cdot p\cdot p'))\ \overline{v_p} v_{p'} \\ &= \text{trace}\, M\left(\sum_{p,p'\in P} h\cdot \overline{v_p}\, p\cdot v_{p'}\, p'\right) = \text{trace}\, M\left(h\cdot |q|^2\right), \qquad q = v\cdot P = \sum_{p\in P} v_p\, p \in P, \\ &= \sum_{x\in X} \mu(x)\, h(x)\ |q(x)|^2, \end{aligned}$$

also

$$v^H\, T(h)\, v = \sum_{x\in X} \mu(x)\, h(x)\ \left|(q\cdot P)\right|^2 . \tag{7.7}$$

Und das liefert uns auch schon die Aussage, wie wir diejenigen Polynome $q \in P$ mit $q(X) = 0$ finden können.

Satz 7.10 *Für ein Polynom $q = v \cdot P$, $v \in \mathbb{K}^P$, gilt*

$$q(X) = 0 \qquad \Longleftrightarrow \qquad T(1)\, v = 0. \tag{7.8}$$

Beweis: Sei $q(X) = 0$. Für ein beliebiges $w \in \mathbb{K}^n$ setzen wir $g := w \cdot P$ und erhalten, daß

$$w^T\, T(1)\, v = \sum_{x\in X} \mu(x)\, g(x)\, \underbrace{q(x)}_{=0} = 0,$$

was uns $T(1)\, v = 0$ liefert.
Sei nun umgekehrt $T(1)\, v = 0$ und seien $v_x \in \mathbb{K}^P$ die linear unabhängigen[184] Vektoren, so daß $\ell_x = v_x \cdot p$, $x \in X$, dann ist für $x \in X$

$$0 = v_x^T\, 0 = v_x^T\, T(1)\, v = \sum_{x'\in X} \mu\left(x'\right)\, \underbrace{\ell_x\left(x'\right)}_{=\delta_{x,x'}}\, q\left(x'\right) = q(x),$$

was uns somit $q(X) = 0$ liefert. □

Algorithmus 7.11 *(Bestimmung des Radikals)*
Gegeben: *endliches $F \subset \Pi$, so daß $\langle F\rangle$ nulldimensional ist.*

1. *Bestimme eine Γ–Basis G von $\langle F\rangle$.*
2. *Bestimme den endlichdimensionalen Raum*

   $$N_F = \nu_G\left(\Pi\right)$$

   und eine Basis P davon.
3. *Berechne die Multiplikationstabellen*

   $$M\left(p\cdot p'\right), \qquad p, p' \in P,$$

   und die Matrix $T(1)$.
4. *Bestimme*[185] *eine Basis $V \subseteq \mathbb{K}^{|P|}$ von*

   $$\ker T(1) = \left\{v \in \mathbb{K}^{|P|}\ :\ T(1)\, v = 0\right\}.$$

[184] Verständnisfrage: Warum müssen diese Vektoren linear unabhängig sein?
[185] Beispielsweise durch Gaußelimination oder eine QR–Zerlegung.

5. Berechne eine Γ*–Basis* G^* *von*

$$\langle F \cup \{v \cdot P \ : \ v \in V\}\rangle.$$

Ergebnis: Γ*–Basis* G^* *von* $\sqrt{\langle F\rangle}$.

Mit Hilfe dieses Verfahrens können wir uns nun endlich um die Lösung des Gleichungssystems $F(X) = 0$ kümmern, indem wir zuerst mit obiger Methode eine Basis des *Radikals* $\sqrt{\langle F\rangle}$ bestimmen und dann über ein Eigenwertverfahren das zugehörige "einfache" Problem lösen.

7.4 Ein Beispiel

Wir wollen uns diese Methode nun einmal anhand eines einfachen Beispiels ansehen, nämlich den Schnitt, genauer gesagt, den *Berührpunkt* zwischen einem Kreis um den Ursprung und einem um den Punkt $[1, 0]$ bestimmen. Die Gleichungen sind also

$$0 = f_1(x, y) = x^2 + y^2 - r_1^2, \qquad 0 = f_2(x, y) = (x - 1)^2 + y^2 - r_2^2.$$

Eine Berührung zeichnet sich ja dadurch aus, daß $\sqrt{\langle F\rangle} \neq \langle F\rangle$, also $\det T(1) = 0$ ist. Eine H–Basis des Ideals erhalten wir, indem wir f_2 von f_1 subtrahieren, was uns $h_1(x, y) = 2x + r_1^2 - r_2^2 + 1$ liefert; dieses Polynom muss auf alle Fälle in die Basis. Bilden wir außerdem

$$h_2(x, y) = f_1(x, y) - \frac{x}{2}h_1(x, y) = y^2 - \underbrace{\frac{r_1 - r_2 + 1}{2}}_{=:R} x - r_1^2 = y^2 - Rx - r_1^2,$$

dann sieht man leicht, daß $\frac{1}{2}h_1 = x + R$ und $h_2(x, y)$ eine H–Basis[186] bilden – bereits alle quadratischen Terme x^2, xy und y^2 finden sich im Leittermideal der beiden Polynome. Der Quotientenraum $\Pi/\langle F\rangle = \text{span}\,\{1, y\}$ ist auch einfach und für $T(1)$ benötigen wir nur die Multiplikationstabellen[187]

$$M(1) = I, \qquad M(x) = \begin{bmatrix} 0 & 1 \\ 0 & -R \end{bmatrix}, \qquad M\left(x^2\right) = \begin{bmatrix} 0 & -R \\ 0 & R^2 \end{bmatrix},$$

also

$$T(1) = \begin{bmatrix} \text{trace}\, M(1) & \text{trace}\, M(x) \\ \text{trace}\, M(x) & M\left(x^2\right) \end{bmatrix} = \begin{bmatrix} 2 & -R \\ -R & R^2 \end{bmatrix} \qquad \Rightarrow \qquad \det T(1) = R^2.$$

Damit liegt also genau dann ein Berührpunkt vor, wenn $R^2 = 0$ ist, also wenn

$$0 = R = r_1^2 + r_2^2 - 1 \qquad \Leftrightarrow \qquad r_1^2 + r_2^2 = 1$$

ist – genau das, was man sich geometrisch auch so vorstellen würde.

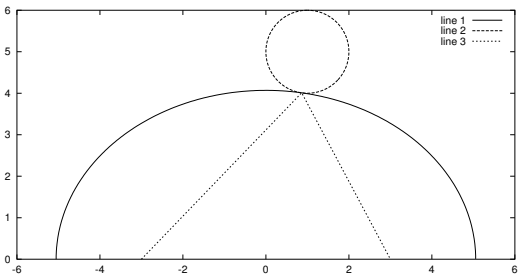


Abbildung 7.1: Eine Ellipse und ein Kreis berühren sich in einem Punkt. Wie ist die große Halbachse zu wählen?

7.5 Noch ein Beispiel

Etwas interessanter wird die Sache schon, wenn wir nach dem Berührpunkt zwischen einem Kreis und einer Ellipse suchen. Dazu skalieren wir das Problem so, daß die Ellipse ihre beiden Brennpunkte in $[-1, 0]$ und $[1, 0]$ hat, was die Gleichung

$$\frac{x^2}{\alpha^2} + \frac{y^2}{4+\alpha^2} = 1 \qquad \Leftrightarrow \qquad 0 = x^2 + \frac{\alpha^2}{4+\alpha^2}\, y^2 - \alpha^2 =: f_1(x, y) \tag{7.9}$$

liefert. Der Kreis hingegen wird mit seinem Mittelpunkt $[c_1, c_2]$ und seinem Radius r als

$$0 = (x - c_1)^2 + (y - c_2)^2 - r^2 = x^2 + y^2 - 2\,(c_1\, x + c_2\, y) + \underbrace{c_1^2 + c_2^2 - r^2}_{=:R} =: f_2(x, y) \tag{7.10}$$

geschrieben. Was uns nun interessiert ist die Frage, für welche Werte[188] von α sich der Kreis und die Ellipse berühren, also eine *doppelte* Nullstelle von

$$0 = F(X), \qquad F = \{f_1, f_2\}$$

vorliegt, wann also $\sqrt{\langle F \rangle} \neq \langle F \rangle$ ist.

Zuerst bemerken wir, daß f_1, f_2 bereits eine H–Basis bilden, da die Leitterme

$$\lambda\,(f_1)\,(x, y) = x^2 + \frac{\alpha^2}{4+\alpha^2}\, y^2, \qquad \lambda\,(f_2)\,(x, y) = x^2 + y^2$$

genau dann beide gleich Null sind, wenn

$$0 = \frac{4}{4+\alpha^2}\, y^2 \qquad \Rightarrow \qquad y = 0 \qquad \Rightarrow \qquad x = 0$$

[186] Sogar eine graduiert lexikografische Gröbnerbasis.

[187] Zum Selbst–Nachrechnen: $\nu(x^2) = -Rx$, $\nu\left(x^3\right) = R^2 x$.

[188] Und es müssen genau zwei sein: Einmal wird der Kreis wie in Abb. 7.1 von der Ellipse berührt, es gibt aber auch noch den anderen Fall, daß der Kreis im Inneren der Ellipse liegt.

ist. Die einzigen gemeinsame Nullstelle der Leitterme ist also die triviale und wegen (Möller & Sauer, 2000a, Theorem 5.3) ist F dann auch schon eine H–Basis. Damit ist aber, ganz egal, ob wir als inneres Produkt

$$(f,g) = \sum_{\alpha} f_\alpha \, g_\alpha \qquad \text{oder} \qquad (f,g)_* = (f(D)g)\,(0)$$

verwenden, das Polynom xy orthogonal zu F und

$$V_2(F) = \left\{ a\,x^2 + b\,y^2 \ : \ a,b \in \mathbb{R} \right\},$$

sowie $V_k(F) = \Pi_k^0$, $k \geq 3$, und

$$\Pi / \langle F \rangle = \text{span}\,\{1,x,y,xy\} =: \text{span}\,P. \tag{7.11}$$

Um die H–Basis etwas "handlicher" zu bekommen, ersetzen wir die Polynome durch die "alternative" H–Basis bestehend aus

$$\begin{aligned} h_1(x,y) &= \frac{4+\alpha^2}{4}\left[f_2(x,y) - f_1(x,y)\right] = \frac{4+\alpha^2}{4}\left[\frac{4}{4+\alpha^2}y^2 - 2(c_1x + c_2y) + R + \alpha^2\right] \\ &= y^2 + \left(1 + \frac{\alpha^2}{4}\right)\left[R + \alpha^2 - 2\,(c_1x + c_2y)\right] \end{aligned}$$

und

$$\begin{aligned} h_2(x,y) &= -\frac{\alpha^2}{4}\left[f_2(x,y) - \frac{4+\alpha^2}{\alpha^2}f_1(x,y)\right] \\ &= -\frac{\alpha^2}{4}\left[-\frac{4}{\alpha^2}x^2 - 2\,(c_1x + c_2y) + R + 4 + \alpha^2\right] \\ &= x^2 - \frac{\alpha^2}{4}\left[R + 4 + \alpha^2 - 2\,(c_1x + c_2y)\right], \end{aligned}$$

bei denen die Leitterme Monome sind.

So, aber jetzt geht die Arbeit los, denn wir müssen die Multiplikationstabellen

$$M(1), M(x), M(y), M(xy), M\left(x^2\right), M\left(y^2\right), M\left(x^2y\right), M\left(xy^2\right), M\left(x^2y^2\right)$$

zur Bestimmung der Matrix

$$T(1) = \begin{bmatrix} \text{trace}\,M\,(1\cdot 1) & \text{trace}\,M\,(1\cdot x) & \text{trace}\,M\,(1\cdot y) & \text{trace}\,M\,(1\cdot xy) \\ \text{trace}\,M\,(x\cdot 1) & \text{trace}\,M\,(x\cdot x) & \text{trace}\,M\,(x\cdot y) & \text{trace}\,M\,(x\cdot xy) \\ \text{trace}\,M\,(y\cdot 1) & \text{trace}\,M\,(y\cdot x) & \text{trace}\,M\,(y\cdot y) & \text{trace}\,M\,(y\cdot xy) \\ \text{trace}\,M\,(xy\cdot 1) & \text{trace}\,M\,(xy\cdot x) & \text{trace}\,M\,(xy\cdot y) & \text{trace}\,M\,(xy\cdot xy) \end{bmatrix}$$

berechnen, immer bezüglich der Basis P. Natürlich ist $M(1) = I$ und

$$M(x) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ * & * & * & * \\ 0 & 0 & 0 & 1 \\ \spadesuit & \spadesuit & \spadesuit & \spadesuit \end{bmatrix},$$

wobei in "$*$" die Koeffzienten der Normalform von x^2, in ♠ die der Normalform von x^2y stehen. Unter Verwendung von h_2 ist nun

$$\nu\left(x^2\right) = -\frac{\alpha^2 c_1}{2}x - \frac{\alpha^2 c_2}{2}y + \frac{\alpha^2\left(R+4+\alpha^2\right)}{4}$$

und

$$\begin{aligned}\nu\left(x^2y\right) &= \frac{\alpha^2\left(R+4+\alpha^2\right)}{4}y - \frac{\alpha^2 c_1}{2}xy - \frac{\alpha^2}{2}\nu\left(y^2\right)\\ &= \frac{\alpha^2\left(R+4+\alpha^2\right)}{4}y - \frac{\alpha^2}{2}xy + \frac{\alpha^2}{2}\left(1+\frac{\alpha^2}{4}\right)\left[R+\alpha^2-2\left(c_1x+c_2y\right)\right]\\ &= -\frac{\alpha^2}{2}xy + \frac{\alpha^2\left[R+\left(4+\alpha^2\right)\left(1-c_2\right)\right]}{4}y - \frac{\alpha^2\left(4+\alpha^2\right)c_1}{4}x + \frac{\alpha^2\left(4+\alpha^2\right)\left(R+\alpha^2\right)}{8}.\end{aligned}$$

Also ist

$$M(x) = \begin{bmatrix} 0 & 1 & 0 & 0\\ \frac{\alpha^2\left(R+4+\alpha^2\right)}{4} & -\frac{\alpha^2 c_1}{2} & -\frac{\alpha^2}{2} & 0\\ 0 & 0 & 0 & 1\\ \frac{\alpha^2\left(4+\alpha^2\right)\left(R+\alpha^2\right)}{8} & \frac{\alpha^2\left(4+\alpha^2\right)c_1}{4} & \frac{\alpha^2\left[R+\left(4+\alpha^2\right)\left(1-c_2\right)\right]}{4} & -\frac{\alpha^2 c_1}{2}\end{bmatrix}$$

und

$$\text{trace}\, M(x) = -\alpha^2 c_1.$$

Nun haben wir uns natürlich jede Menge unnötige Arbeit gemacht, denn eigentlich brauchen wir natürlich nicht die ganze Matrix, sondern "nur" deren Diagonalelemente. Für die restliche Reduktionen verwenden wir dann auch den Computer, beispielsweise die Funktion `normalf` aus dem `Groebner`–Paket von Maple. Die Diagonalvektoren der Matrizen ergeben sich so als[189]

$$\begin{aligned}M(y) &\simeq \left[\nu_1(y), \nu_x(xy), \nu_y\left(y^2\right), \nu_{xy}\left(xy^2\right)\right]\\ &= \left[0, 0, \frac{4+\alpha^2}{2}c_2, \frac{4+\alpha^2}{2}c_2\right] \to \left(4+\alpha^2\right)c_2\\ M(xy) &\simeq \left[\nu_1(xy), \nu_x\left(x^2y\right), \nu_y\left(xy^2\right), \nu_{xy}\left(x^2y^2\right)\right]\\ &= \left[0, -\frac{4+\alpha^2}{4}\alpha^2 c_1c_2, -\frac{4+\alpha^2}{4}\alpha^2 c_1c_2, -\frac{4+\alpha^2}{2}\alpha^2 c_1c_2\right] \to -\left(4+\alpha^2\right)\alpha^2 c_1c_2\end{aligned}$$

Von da an wird es noch fieser, und wir geben nur noch die Spuren an:

$$\begin{aligned}\text{trace}\, M\left(x^2\right) &= \alpha^2\left[R+4+\alpha^2+\frac{\alpha^2 c_1^2}{2} - \frac{c_2^2\left(4+\alpha^2\right)}{2}\right]\\ \text{trace}\, M\left(y^2\right) &= -\frac{4+\alpha^2}{2}\left[\alpha^2 c_1^2 - c_2^2\left(\alpha^2+4\right) + 2\alpha^2 + 2R\right].\end{aligned}$$

[189] ν_f bezeichnet den Koeffizienten zur Basisfunktion f in der Normalform.

Und weil diese Ausdrücke auch immer schlimmer werden, überlassen wir den Rest gänzlich der Computeralgebra, substituieren $a = \alpha^2$ und überlassen dem Programm die Bestimmung von

$$\det T(1) = \frac{1}{16}\, a^2\, (4+a)^2\, g(a), \qquad g \in \Pi_4. \tag{7.12}$$

Bevor wir uns g explizit widmen, bemerken wir, daß die beiden Fälle $\alpha = 0$ und $\alpha^2 = -2$ für "realistische" Halbachsen sowieso nicht erfüllt werden. Also bleibt

$$\begin{aligned}
g(a) \;=\; & \frac{1}{16}\left(c_1^2+c_2^2+4+4c_2\right)\left(c_1^2+c_2^2+4-4c_2\right)\, a^4 \\
& + \left(8 - 2c_1^2 + \frac{3}{2}\,Rc_1^2 + 4\,R - 10\,c_2^2 - \frac{3}{2}\,c_2^2R + c_2^2c_1^2 + \frac{1}{4}c_1^2c_2^2R - \frac{3}{4}\,c_2^4c_1^2\right) \\
& \quad - \frac{3}{4}\,c_2^2c_1^4 + \frac{1}{8}\,Rc_1^4 + \frac{1}{8}\,Rc_2^4 - \frac{1}{4}\,c_1^6 - 2\,c_1^4 + 3\,c_2^4 - \frac{1}{4}\,c_2^6\Big)\, a^3 \\
& + \big(24\,R - 40\,c_2^2 - 32\,c_1^2 + 22\,c_2^4 + 6\,R^2 + 16 - 22\,c_2^2R - 4\,Rc_1^2 - 8\,c_1^4 + 22\,c_2^2c_1^2 \\
& \quad + \frac{3}{2}\,c_1^2c_2^2R - \frac{3}{2}\,c_2^2R^2 + \frac{3}{2}\,R^2c_1^2 - 6\,c_2^4c_1^2 + \frac{7}{2}\,Rc_2^4 + \frac{1}{16}\,R^2c_2^4 - 3\,c_2^2c_1^4 - 2\,Rc_1^4 \\
& \quad + \frac{1}{16}\,R^2c_1^4 - 3\,c_2^6 + \frac{1}{8}\,c_1^2c_2^2R^2\Big)\, a^2 \\
& + \big(32\,R - 32\,c_2^2 - 64\,c_1^2 + 24\,R^2 - 72\,c_2^2R - 32\,Rc_1^2 + 80\,c_2^2c_1^2 - 14\,c_2^2R^2 \\
& \quad + 22\,Rc_2^4 - 2\,R^2c_1^2 - 12\,c2^4c_1^2 - \frac{1}{2}\,R^3c_2^2 + \frac{1}{2}\,c_1^2R^3 + \frac{1}{2}\,R^2c_2^4 + 48\,c_2^4 + 4\,R^3 \\
& \quad - 12\,c_2^6 + 2\,c_1^2c_2^2R + \frac{1}{2}\,c1^2c_2^2R^2\Big)\, a \\
& + (R+4+4\,c_2)\,(R+4-4\,c_2)\left(R-c_2^2\right)^2
\end{aligned}$$

Eine Berührung erfolgt nun genau dann, wenn a eine *positive* Nullstelle dieses Polynoms ist – aber bereits mit der expliziten Bestimmung der vier Nullstellen von g ist `Maple` mal wieder überfordert. Allerdings spricht nichts dagegen, für *explizite* Werte von c_1, c_2 und R mit einem numerischen Verfahren wie Newton nach den positiven Nullstellen zu suchen, die große liefert die Berührung von außen, die kleine die Berührung von innen.

Es gibt auch keine Spezialfälle, in denen man vereinfachen könnte, denn der Leitterm

$$\left[c_1^2 + (c_2+2)^2\right]\left[c_1^2 + (c_2-2)^2\right]$$

ist außer für die beiden Punkte[190] $[0, \pm 2]$ immer strikt positiv und der konstante Koeffizient verschwindet nur, wenn

$$0 = R + 4 \pm 4c_2 = c_1^2 + c_2^2 - r^2 + 4 \pm 4c_2 = c_1^2 - r^2 + (c_2 \pm 2)^2$$

bzw. wenn

$$0 = R - c_2^2 = c_1^2 - r^2 \qquad \Leftrightarrow \qquad c_1^2 = r^2$$

ist. Wer will, kann sich ja mal die geometrischen Situationen ansehen, die davon impliziert sind.

[190] Die sind offenbar etwas ganz besonderes, symmetrische Situation **und** Abstand zum Mittelpunkt genauso groß wie der Abstand der Brennpunkte.

7.6 Zählen reeller Nullstellen

Zum Abschluss werden wir, was den Körper angeht, nochmal ein wenig spezifischer und betrachten die Nullstellen *reeller* Polynome, die aber halt leider auch komplex werden können. Tatsächlich interessiert man sich in realistischen Problemen oftmals nur für die reellen Lösungen - die komplexen Lösungen sind nur für Schwierigkeiten zuständig, siehe(Sauer & Wagenführ, 2006).

Ideal wäre es, wenn man das *reelle Ideal* zu einem Gleichungssystem bestimmen könnte, also das *kleinste* Ideal $\mathscr{I}$, so daß

$$\langle F \rangle \subseteq \mathscr{I} \qquad \text{und} \qquad \mathbb{R}^n \supseteq V(\mathscr{I}) = \{x \in \mathbb{C}^n \ : \ \mathscr{I}(x) = 0\}\,. \tag{7.13}$$

Dies ist bisher[191] aber immer noch ein offenes Problem, weswegen wir uns erst einmal mit weniger zufrieden geben wollen, nämlich mit dem *Zählen* der reellen Nullstellen eines gegebenen Ideals, wobei uns noch einmal die Trace-Matrix zu Hilfe kommen wird - der ganze Ansatz findet sich in (Cohen *et al.*, 1999, S. 134-5), genauer in (Gonzales-Vega *et al.*, 1999b), und besser ausgearbeitet, in (Basu *et al.*, 2003), geht aber bereits auf Hermite zurück. Doch zuerst ein bisschen Notation und Terminologie.

Definition 7.12 *Die* Signatur $S(A)$ *einer hermiteschen*[192] *Matrix* A *ist die die Differenz zwischen der Anzahl der positiven und der Anzahl der negativen Eigenwerte*[193]*:*

$$S(A) = \#\{\lambda \in \sigma(A) \ : \ \lambda > 0\} - \#\{\lambda \in \sigma(A) \ : \ \lambda < 0\}\,. \tag{7.14}$$

Das wesentliche Hilfsmittel zur Signatur einer Matrix und auch ihre wesentliche Anwendung ist der *(Sylvestersche) Trägheitsatz*[194], den wir - für unsere Zwecke passend und auf quadratische Formen angepasst - in (Gantmacher, 1959a, S. 297) finden können. Dazu erinnern wir uns, daß man eine reelle quadratische Form, also die Abbildung $v \mapsto v^T A v$, $v \in \mathbb{R}^n$, auf viele Arten als Summe von Quadraten darstellen kann, also als

$$v^T A v = \sum_{j=1}^{r} \alpha_j \left(v^T w_j\right)^2, \qquad w_j \in \mathbb{R}^n,\ \alpha_j \in \mathbb{R}, \quad j = 1, \ldots, r \leq n; \tag{7.15}$$

die „klassische“ derartige Darstellung erhält man, wenn man die α_j als die Eigenwerte von A und die w_j als die zugehörigen *orthogonalen* Eigenvektoren wählen, es gibt aber auch eine Vielzahl von anderen Darstellungen.

Bemerkung 7.13 *Die Darstellung (7.15) der quadratischen Form zu* A *entspricht einer Zerlegung von* A *in symmetrische Matrizen vom Rang* 1*:*

$$A = \sum_{j=1}^{r} \alpha_j \, w_j w_j^T. \tag{7.16}$$

[191] Stand 14.1.2008

[192] Das schliesst natürlich den Fall reeller symmetrischer Matrizen ein. Auf jeden Fall sind aber für hermitesche Matrizen die Eigenwerte reell und damit ist das, was folgt, auch wirklich wohldefiniert. Ansonsten heisst das Stichwort *Sylvesterscher Trägheitssatz*.

[193] Hier bezeichnet $\sigma(A)$ das *Spektrum* von A, also die Menge der Eigenwerte von A.

[194] Auf Englisch *“law of inertia”*

Die *Trägheit* der Matrix ist nun die Anzahl der positiven, negativen und verschwindenden Koeffizienten in dieser Darstellung, also das Tripel[195]

$$(I_+, I_-, I_0), \qquad I_\pm = \#\{j \ : \ \pm\alpha_j > 0), \quad I_0 = n - I_+ - I_-.$$

Der Trägheitssatz besagt nun nichts anderes als daß die Trägheit von der jeweiligen Darstellung *unabhänig* ist, daß man zwar die α_j und w_j anders wählen kann, nicht aber ihr Vorzeichenverhalten.

Satz 7.14 (Trägheitssatz) *Für jede Darstellung der Matrix $A \in \mathbb{R}^{n\times n}$ in der Form (7.15) mit* linear unabhänigen w_j, $j = 1, \ldots, r$, *hängt die Trägheit, also die Anzahl der positiven und negativen Koeffizienten α_j nur von A, aber nicht von der gewählten Darstellung ab.*

Wir verwenden jetzt wieder die Trace-Matrix aus (7.6), und zwar als quadratische Form auf N_F:

$$N_F \ni f = v \cdot P \mapsto v^T T(h) v.$$

Nachdem wir ja schon wissen, wie wir von einem Ideal $\langle F\rangle$ zum zugehörigen Radikal kommen, beschränken wir uns hier auf radikale Ideale. Schließlich sind Vielfachheit und Körpererweiterung ja auch "unterschiedliche Baustellen". Dann gilt die folgende Aussage, die sehr stark an Sturmsche Ketten erinnert, siehe z.B. (Gantmacher, 1959b).

Satz 7.15 *Für ein nulldimensionales radikales Ideal $\langle F\rangle$ und $h \in \Pi$ gilt*

$$S(T(h)) = \#\{x \in \mathbb{R}^n \ : \ F(x) = 0,\ h(x) > 0\} - \#\{x \in \mathbb{R}^n \ : \ F(x) = 0,\ h(x) < 0\}. \tag{7.17}$$

Setzen wir in diesem Satz ganz spezifisch $h = 1$, dann erhalten wir ein einfaches Rezept, um die reellen Lösungen eines polynomialen Gleichungssystems zu zählen.

Korollar 7.16 *Ist $\langle F\rangle$ ein nulldimensionales Radikal, dann ist*

$$S(T(1)) = \#\{x \in \mathbb{R}^n \ : \ F(x) = 0\}.$$

Beweis von Satz 7.15: Sei X die Lösungsmenge von $F(X) = 0$ und seien $X_\mathbb{R} := X \cap \mathbb{R}^n$ sowie $X_\mathbb{C} := X \setminus X_\mathbb{R}$. Nun ist wieder[196], wie in (7.7),

$$\begin{aligned} v^T T(h)\, v &= \sum_{x\in X} h(x)\ [(v\cdot P)(x)]^2 = \sum_{x\in X_\mathbb{R}} h(x)\ [(v\cdot P)(x)]^2 + \sum_{x\in X_\mathbb{C}} h(x)\ [(v\cdot P)(x)]^2 \\ &=: v^T T_\mathbb{R}\, v + v^T T_\mathbb{C}\, v. \end{aligned}$$

Seien nun

$$m_x := \nu_{\langle F\rangle}\left(\prod_{x'\in X_\mathbb{R}\setminus\{x\}} \frac{(x-x')^T(\cdot - x')}{(x-x')^T(x-x')}\right), \qquad x \in X_\mathbb{R},$$

[195] Die Terme mit $\alpha_j = 0$ müssen ja in der Darstellung (7.15) nicht auftauchen, deswegen diese Definition von I_0.

[196] Alle Vielfachheiten $\mu(x)$, $x \in X$, sind nach Voraussetzung 1.

ein Satz von Polynomen mit $m_x(x') = \delta_{x,x'}$, dann können wir diese mittel q_x, $x \in X_{\mathbb{C}}$, zu einer Basis von N_F ergänzen und die Polynome

$$m_x := q_x - \sum_{x' \in X_{\mathbb{R}}} q_x(x')\, m_{x'}, \qquad x \in X_{\mathbb{C}},$$

sind nun *reelle* Polynome, die an $X_{\mathbb{R}}$ verschwinden, aber zusammen mit den m_x, $X \in X_{\mathbb{R}}$, immer noch eine Basis bilden

$$N_F = \text{span}\,\{m_x : x \in X\}.$$

Da aber nun

$$T_{\mathbb{R}}\, v_x = 0, \qquad m_x = v_x \cdot P, \quad x \in X_{\mathbb{C}},$$

ist auch aber

$$S(T(h)) = S(T_{\mathbb{R}}) + S(T_{\mathbb{C}}). \tag{7.18}$$

Seien nun, $x, \overline{x} \in X_{\mathbb{C}}$ ein konjugiert komplexes Lösungspärchen und[197] $h(x) = (\alpha + i\beta)^2$, also $h(\overline{x}) = (\alpha - i\beta)^2$, sowie

$$(v \cdot P(x)) = (v \cdot \Re P(x) + iv \cdot \Im P(x))$$

und

$$(v \cdot P(\overline{x})) = (v \cdot \Re P(x) - iv \cdot \Im P(x)).$$

Insgesamt ist dann für $x \in X_{\mathbb{C}}$

$$\begin{aligned}
&h(x)\,[(v \cdot P)(x)]^2 + h(\overline{x})\,[(v \cdot P)(\overline{x})]^2 \\
&\quad = \quad [(\alpha + i\,\beta)\,(v \cdot \Re P(x) + i\,v \cdot \Im P(x))]^2 + [(\alpha - i\,\beta)\,(v \cdot \Re P(x) - i\,v \cdot \Im P(x))]^2 \\
&\quad = \quad [v \cdot a_x + i\,v \cdot b_x]^2 + [v \cdot a_x - i\,v \cdot b_x]^2 \\
&\quad = \quad \left[(v \cdot a_x)^2 + 2i\ (v \cdot a_x)\,(v \cdot b_x) - (v \cdot b_x)^2\right]\left[(v \cdot a_x)^2 - 2i\ (v \cdot a_x)\,(v \cdot b_x) - (v \cdot b_x)^2\right] \\
&\quad = \quad 2\left[(v \cdot a_x)^2 - (v \cdot b_x)^2\right],
\end{aligned}$$

und damit ist

$$T_{\mathbb{C}} = \sum_{x \in X_{\mathbb{C}}} (v \cdot a_x)^2 - \sum_{x \in X_{\mathbb{C}}} (v \cdot b_x)^2.$$

In dieser Summe tauchen jedes a_x und jedes b_x zweimal auf, nämlich für x und $\overline{x}$; um diesen Effekt loszuwerden, wählen wir $X'_{\mathbb{C}} \subset X_{\mathbb{C}}$ so, daß $X_{\mathbb{C}} = X'_{\mathbb{C}} \cup \overline{X'_{\mathbb{C}}}$, und erkennen so, daß die zu $T_{\mathbb{C}}$ gehörige symmetrische Matrix $A_{\mathbb{C}}$ die Form

$$A_{\mathbb{C}} = \sum_{x \in X'_{\mathbb{C}}} a_x\, a_x^T - \sum_{x \in X'_{\mathbb{C}}} b_x\, b_x^T$$

[197] Im Komplexen kann man ja ganz einfach Wurzeln ziehen, warum also nicht hier?

haben muss. Der Rang[198] $R(A_{\mathbb{C}})$ dieser Matrix ist offensichtlich[199] $\leq X_{\mathbb{C}}$, mit Gleichheit genau dann, wenn die a_x, b_x allesamt linear unabhängig sind. Nun ist aber[200] $R(A_{\mathbb{R}}) = \#X_{\mathbb{R}}$ und außerdem

$$\#X_{\mathbb{R}} + \#X_{\mathbb{C}} = \#X = R(A) \leq R(A_{\mathbb{R}}) + R(A_{\mathbb{C}}) = \#X_{\mathbb{R}} + R(A_{\mathbb{C}}),$$

also auch $R(A_{\mathbb{C}}) \geq \#X_{\mathbb{C}}$ und somit $R(A_{\mathbb{C}}) \geq \#X_{\mathbb{C}}$ - die a_x, b_x sind linear unabhängig. Damit können wir aber endlich den Trägheitssatz, Satz 7.14, anwenden, der uns $S(T_{\mathbb{C}}) = 0$ liefert. Fazit:

$$S(T(h)) = S(T_{\mathbb{R}})$$

und eine Basistransformation zu $\{m_x : x \in X\}$ lässt zwar die Signatur unverändert, aber liefert eine Zerlegung in die beiden Teilräume

$$V_+ = \text{span}\,\{v_x : h(x) > 0\} \qquad \text{und} \qquad V_- = \text{span}\,\{v_x : h(x) < 0\}$$

auf denen die quadratische Form $T_{\mathbb{R}}$ nun strikt positiv bzw. strikt negativ ist, was sich natürlich auch auf die Anzahl der Eigenwerte der Einschränkung auswirkt. □

[198] Und das ist auch gleich die Definition der Notation $R(A)$!

[199] Denn wir haben ja nur $2\,\#X'_{\mathbb{C}} = \#X_{\mathbb{C}}$ Summanden vom Rang 1 hier!

[200] Wegen der m_x, $x \in X$!

Chaos is found in greatest abundance whereever order is being sought. It always defeats order, because it is better organized.

T. Pratchett, *Interesting times*

8 Von der Interpolation zum Ideal

Daß Idealtheorie und Interpolation an einer endlichen Punktmenge $X \subset \mathbb{K}^d$ eine Menge miteinander zu tun haben müssen, das sollte und längst klar sein; schließlich liefert uns ja die Normalformabbildung $\nu_{\mathscr{I}(X)} : \Pi \to \Pi/\mathscr{I}(X)$ einen Interpolanten. Und das ist ja auch ganz nett, wenn eine Basis von $\mathscr{I}(X)$ gegeben ist, denn dann können wir zu jedem Polynom $f \in \Pi$ mit $\nu_{\mathscr{I}(X)} f$ den zugehörigen Interpolanten berechnen, und zwar sogar *ohne* die Punktmenge X zu kennen. Aber: Wie kommen wir von X zu $\mathscr{I}(X)$? Wir werden sehen, daß die Idealbasis ein "Abfallprodukt" der Konstruktion des Interpolanten ist.

8.1 Gradreduzierende Interpolation und Newtonbasen

Beginnen wir doch einmal damit, uns die grundlegenden Begriffe zu überlegen.

Definition 8.1 *Sei $X \subset \mathbb{K}^n$ eine endliche Menge von Punkten. Ein Teilraum $\mathscr{P}$ von Π heißt*

1. Interpolationsraum*, wenn es zu jedem $f \in \Pi$ ein* eindeutiges *Polynom $L_{\mathscr{P}} f \in \mathscr{P}$ gibt, so daß*
$$f(X) = L_{\mathscr{P}} f(X)$$
ist.

2. Minimalgrad–Interpolationsraum*, wenn $\mathscr{P}$ ein Interpolationsraum ist, aber kein Unterraum $\mathscr{Q} \subset \Pi$ mit $\delta(\mathscr{Q}) < \delta(\mathscr{P})$. Hierbei ist*
$$\delta(\mathscr{Q}) := \max_{q \in \mathscr{Q}} \delta(q).$$

3. gradreduzierender Interpolationsraum*, wenn*
$$\delta\left(L_{\mathscr{P}} f\right) \leq \delta(f), \qquad f \in \Pi.$$

Für diese Begriffe können wir sofort ein paar einfache Eigenschaften und Beziehungen herleiten.

Proposition 8.2 *1. $\mathscr{P} \subset \Pi$ ist genau dann ein Interpolationsraum, wenn es Polynome $\ell_x \in \mathscr{P}$, $x \in X$, gibt, so daß $\ell_x(x') = \delta_{x,x'}$, $x, x' \in X$, ist.*

2. Die Lagrange–Fundamentalpolynome *ℓ_x, $x \in X$, bilden eine Basis von $\mathscr{P}$.*

3. Jeder gradreduzierende Interpolationsraum ist auch von minimalem Grad.

Beweis: 1. Ist $\mathscr{P}$ ein Interpolationsraum, dann setzen wir, wie in (7.2), für $x \in X$

$$\ell_x := L_{\mathscr{P}} \left(\prod_{x' \neq x} \frac{v_{x,x'}^T (\cdot - x')}{v_{x,x'}^T (x - x')} \right), \qquad v_{x,x'} \in \mathbb{K}^n, \quad v_{x,x'}^T (x - x') \neq 0, \tag{8.1}$$

was auch schon die gesuchten Polynome sind. Existieren umgekehrt diese Polynome, dann ist wieder zu $f \in \Pi$ das Polynom

$$L_{\mathscr{P}} f := \sum_{x \in X} f(x)\, \ell_x \quad \in \mathscr{P}$$

der gesuchte Interpolant.
2. Gäbe es ein $p \in \mathscr{P} \setminus \operatorname{span}\{\ell_x \,:\, x \in X\}$, dann wäre

$$0 \neq \widetilde{p} := p - \sum_{x \in X} p(x)\, \ell_x$$

ein Polynom in $\mathscr{P}$ mit $\widetilde{p}(X) = 0$ und damit die Eindeutigkeit der Interpolation in $\mathscr{P}$ verletzt.
3. Nehmen wir an, $\mathscr{P}$ wäre gradreduzierend und $\mathscr{Q} \subset \Pi$ ebenfalls ein Interpolationsraum mit $\delta(\mathscr{Q}) < \delta(\mathscr{P})$. Da beide Interpolationsräume sind, gibt es nach Teil 1. Lagrange–Basen $\ell_x \in \mathscr{P}$ und $\ell'_x \in \mathscr{Q}$, $x \in X$, und mindestens für ein $x \in X$ muss

$$\delta(\ell_x) = \delta(\mathscr{P}) > \delta(\mathscr{Q}) \geq \delta(\ell'_x)$$

sein. Da aber $\ell_x = L_{\mathscr{P}} \ell'_x$ ist, widerspricht diese Gradungleichung der Voraussetzung, daß $\mathscr{P}$ gradreduzierend ist. □

Die Existenz gradreduzierender Interpolationsräume ist gesichert[201], denn jede Γ–Basis G des Ideals $\mathscr{I}(X)$ führt zu einer wohldefinierten Normalformberechnung $\nu_{\mathscr{I}(X)}(f)$ und schon aufgrund der puren Konstruktion als Γ–Darstllung

$$f = \sum_{g \in G} f_g \; + \nu_{\mathscr{I}(X)}(f)$$

sorgt dafür, daß $\delta\left(\nu_{\mathscr{I}(X)}(f)\right) \leq \delta(f)$, womit $\mathscr{P}_X := \nu_{\mathscr{I}(X)}(\Pi)$ "der" gradreduzierende Interpolationsraum ist.

Bemerkung 8.3 *Weder gradreduzierende, noch gradminimale Interpolationsräume sind im Normalfall eindeutig.*

[201] Wir reden also hier definitiv **nicht** über die leere Menge.

Ein Wort zur Literatur: Gradreduzierende, "problemangepasste" Interpolationsräume wurden von C. de Boor und A. Ron "eingeführt" (Boor & Ron, 1990; Boor & Ron, 1991; Boor & Ron, 1992a; Boor & Ron, 1992b), obwohl man genausogut sagen könnte, daß Möller und Buchberger (Möller & Buchberger, 1982) sich die Sache aus der Perspektive der Gröbnerbasen schon eher angesehen haben, siehe hierzu auch (Möller, 1998). Die nicht ganz so nützlichen "gradminimalen" Interpolationsräume stammen aus (Sauer, 1997), von wo auch der Begriff der Newton–Basis herrührt[202]. Der Bezug zwischen dem Ansatz von de Boor und Ron[203] und Gröbnerbasen wurde in (Sauer, 1998) hergestellt. Generelle Übersichtsarbeiten neueren Datums zum Thema Interpolation sind (Gasca & Sauer, 2000b; Gasca & Sauer, 2000a; Lorentz, 2000; Sauer, 2006).

Um den Begriff der Newton–Basis ein wenig zu motivieren, erinnern wir uns kurz an den *univariaten* Fall. Hier hat es sich gezeigt, daß die Lagrange–Basis

$$\ell_x = \prod_{x' \in X \setminus \{x\}} \frac{\cdot - x'}{x - x'}, \qquad x \in X,$$

numerisch extrem schlecht zu handhaben ist[204], und schon Newton wusste, daß es besser ist, eine andere Basis zu wählen, nämlich

$$1,\ \cdot - x_0,\ (\cdot - x_0)(\cdot - x_1), \ldots$$

Man ordnet also die Punkte an und wählt dann Polynome so, daß sie an den Punkten "niedrigerer Ordnung" verschwinden: Das Polynom $p_0 = 1$ verschwindet nirgends, $p_1 = \cdot - x_0$ an x_0, das Polynom $p_2 = (\cdot - x_0)(\cdot - x_1)$ an x_0 und x_1 und so weiter. Und genau das ist die Idee des Newton–Ansatzes:

Den Grad der Polynome so niedrig wie möglilch halten.

Um diese Idee ins Mehrdimensionale und auf beliebige Graduierungen übertragen zu können, brauchen wir allerdings etwas Terminologie.

Definition 8.4 *Für einen Teilraum $\mathcal{Q}$ von Π und $\gamma \in \Gamma$ bezeichnen wir mit*

$$\Gamma_{\mathcal{Q}} := \bigcap \left\{ \Gamma' \subseteq \Gamma \ :\ \mathcal{Q} \subset \bigoplus_{\gamma \in \Gamma'} \Pi_\gamma \right\}$$

die Indexmenge aller in $\mathcal{Q}$ auftretenden homogenen Anteile.

Lemma 8.5 *Ist $\dim \mathcal{Q} < \infty$, dann ist auch $\#\Gamma_{\mathcal{Q}} < \infty$.*

[202] Wir kommen sofort dazu.

[203] Hier wird einzig und ausschließlich der Totalgrad verwendet – eigentlich ja auch eine feine Sache, auch wenn sich zumindest de Boor neuerdings den Gröbnerbasen zuwendet, (Boor, 2007).

[204] Wobei sich natürlich die Frage stellt, ob Polynominterpolation überhaupt ein probates Mittel zur numerischen Behandlung von Interpolationsproblemen ist. Wenn die Anzahl der Punkte recht groß ist, ist die Antwort ein sehr definitives "Nein".

Beweis: Ein Polynom ist eine *endliche* Linearkombination von Monomen, jedes Monom eine endliche[205] Linearkombination von Γ–homogenen Monomen. Also ist $\#\Gamma_{\mathbb{K}\cdot q} < \infty$ für jedes $q \in \mathscr{Q}$. Ist dann $Q \subset \mathscr{Q}$ eine (wiederum endliche) Basis des Teilraums, dann ist

$$\Gamma_{\mathscr{Q}} = \bigcup_{q \in Q} \Gamma_{\mathbb{K}\cdot q}$$

und damit ebenfalls wieder endlich. □

Gut, wir brauchen also für einen endlichdimensionalen Unterraum von Π immer nur endlich viele homogene Komponenten. Besonders interessant sind sicherlich Polynomräume, die im Gegensatz zu beispielsweise

$$\mathscr{Q} = \text{span}\left\{x^2 + y^2 + 1\right\}$$

von ihren homogenen Komponenten erzeugt werden.

Definition 8.6 *Ein Teilraum $\mathscr{Q}$ von Π heißt* homogen erzeugt*, wenn*

$$\mathscr{Q} = \bigoplus_{\gamma \in \Gamma_{\mathscr{Q}}} (\mathscr{Q} \cap \Pi_\gamma) \tag{8.2}$$

ist.

Wir werden uns im weiteren auf homogen erzeugte Interpolationsräume beschränken[206], und das zuerst einmal damit rechtfertigen, daß es solche Räume nicht nur gibt, sondern daß unser "Prototyp" eines Interpolationsraums, also der Vektorraum der Normalformen, gerade diese Eigenschaft hat.

Proposition 8.7 *Die Interpolationsräume der Form $\mathscr{P}^* = \nu_{\mathscr{I}(X)}(\Pi)$ sind homogen erzeugt.*

Beweis: Einfach! Sei $p = \sum_\gamma p_\gamma$ ein Polynom aus $\mathscr{P}$, dann gehören alle homogenen Komponenten[207] zum enstprechenden $W_\gamma(G)$, wobei G eine Γ–Basis für $\mathscr{I}(X)$ ist. Führen wir also für eine homogene Komponente p_γ den Divisionsalgorithmus durch, dann ist die Projektion auf $V_\gamma(G)$ natürlich das Nullpolynom und daher $\nu_{\mathscr{I}(X)}(p_\gamma) = p_\gamma$, also $p_\gamma \in \nu_{\mathscr{I}(X)}(\Pi) = \mathscr{P}$. □

Nehmen wir also an, wir haben es mit einem homogen erzeugten Interpolationsraum $\mathscr{P}$ zu tun, dann können wir die endliche Menge $\Gamma_{\mathscr{P}}$ von Indizes der Größe nach ordnen[208], also

$$\Gamma_{\mathscr{P}} = \left\{\gamma^0, \ldots, \gamma^m\right\}, \qquad m = \#\Gamma_{\mathscr{P}} - 1.$$

Damit verbinden sich für $k = 0, \ldots, m$ ganz natürlich Teilräume

$$\mathscr{P}_k^0 = \mathscr{P} \cap \Pi_k, \qquad \text{und} \qquad \mathscr{P}_k = \bigoplus_{j=0}^{k} \mathscr{P}_j^0 \tag{8.3}$$

[205] Normalerweise triviale, zumindest wenn es sich um eine der "handelsüblichen" monomialen Graduierungen handelt.

[206] Es gibt auch andere, aber die sagen uns nicht so zu.

[207] Wie es sich für einen anständigen Divisionsrest gehört, siehe Definition 6.17.

[208] Die Ordnung auf dem Monoid Γ ist ja eine totale Ordnung.

von $\mathscr{P}$. Mit dieser Notation können wir die beiden Konzepte des Newton–Ansatzes, aufsteigende Grade und Verschwinden an Punkten niedrigerer Ordnung, auch auf das Mehrdimensionale übertragen – sofern wir es noch um eine idealtheoretische Komponente ergänzen.

Definition 8.8 (Newton–Basis) *Eine Teilmenge*

$$N = \bigcup_{k=0}^{m} N_k, \qquad N_k \subset \mathscr{P}_k,$$

von $\mathscr{P}$ heißt Newton–Basis *von $\mathscr{P}$, wenn*

1. *es eine Zerlegung $X = X_0 \cup \cdots \cup X_m$ gibt, so daß*

$$N_k(X_j) = 0, \quad 0 \le j < k \le m, \qquad N_k(X_k) = I, \quad k = 0, \ldots, m. \tag{8.4}$$

2. *eine Zerlegung*

$$\Pi_\gamma = (\lambda(N) \cap \Pi_\gamma) \oplus \lambda(\mathscr{I}(X)), \qquad \gamma \in \Gamma, \tag{8.5}$$

existiert.

Bemerkung 8.9 *1. Bedingung (8.4) ist genau die Verallgemeinerung der bereits angesprochenen Eigenschaft des Newton–Ansatzes: Die Basispolynome N_k der Ordnung k verschwinden auf allen Punkten niedriger Ordnung, also an $X_0, \ldots, X_{k-1}$, auch wenn das jetzt nicht mehr einzelne Punkte, sondern "Blöcke" von Punkten sind. Die zweite Forderung in (8.4) ist allerdings eine etwas andere Normalisierung! Während die univariate Newton–Basis so normiert ist, daß alle Polynome die* monische *Form $x^k + \cdots$ haben, normieren wir sie hier an den Punkten X_k zu* 1.

2. *Die Bedingung (8.5) hingegen gibt es in einer Variablen nicht, sie ist die (notwendige) idealtheoretische Erweiterung.*

8.2 Konstruktion einer speziellen Newton–Basis

In diesem Abschnitt wollen wir nachweisen, daß der Normalformenraum $\mathscr{P}^*$ eine Newton–Basis besitzt und vor allem auch zeigen, wie man diese konstruiert. Fangen wir an mit einer Bemerkung über die dazugehörige Indexmenge.

Lemma 8.10 *Die Menge $\Gamma^* := \Gamma_{\mathscr{P}^*}$ erfüllt*[209]

$$\Gamma^* = \{\gamma \in \Gamma \ : \ W_\gamma(\mathscr{I}(X)) \neq \{0\}\}.$$

[209] Hierbei setzten wir $W_\gamma(\mathscr{I}(X)) = W_\gamma(G)$ für eine Γ–Basis von $\mathscr{I}(X)$.

Beweis: Zu jedem $\gamma \in \Gamma^*$ gibt es mindestens ein nichttriviales Polynom $0 \neq p \in \Pi_\gamma \cap \mathscr{P}^*$ – schließlich ist nach Proposition 8.7 $\mathscr{P}^*$ ja homogen erzeugt. Nun ist aber $p = \nu_{\mathscr{I}(X)}(p) \in W_\gamma(\mathscr{I}(X))$ und daher

$$\{0\} \neq \{p\} \subset \Gamma_{\mathscr{I}(X)} := \{\gamma \in \Gamma \,:\, W_\gamma(\mathscr{I}(X)) \neq \{0\}\},$$

also $\Gamma^* \subseteq \Gamma_{\mathscr{I}(X)}$. Ist umgekehrt $\gamma \in \Gamma_{\mathscr{I}(X)}$, dann gibt es ein $0 \neq f \in W_\gamma(\mathscr{I}(X))$ und die Normalform dazu ist nun wieder $f = \nu_{\mathscr{I}(X)}(f) \in \mathscr{P}^*$, also ist auch $\gamma \in \Gamma^*$, das heißt, wir haben auch $\Gamma_{\mathscr{I}(X)} \subseteq \Gamma^*$. □

So, dann legen wir mal los! Die endliche Menge Γ^* läßt sich als $\Gamma^* = \{\gamma^0, \ldots, \gamma^m\}$, $m = \#\Gamma^* - 1$, schreiben, wobei $\gamma^0 < \gamma^1 < \cdots < \gamma^m$ ist[210]. Nun beginnen wir mit $\mathscr{P}_0 = \mathscr{P}^* \cap \Pi_{\gamma^0}$, wählen eine Basis P_0 von $\mathscr{P}_0$ und bilden die Matrix

$$P_0(X) = [p(x) \,:\, p \in P_0,\, x \in X]$$

Gäbe es nun einen Koeffizientenektor $v = [v_p \,:\, p \in P_0]$, so daß

$$0 = v^T P_0(X) = \left[\sum_{p \in P_0} v_p\, p(x) \,:\, x \in X\right] = [q(x) \,:\, x \in X] = q(X), \qquad q = v \cdot P_0 \in \mathscr{P}_0,$$

dann hätten wir ein nichttriviales Polynom aus $\mathscr{P}_0 \subseteq \mathscr{P}^*$ gefunden, das an allen Punkten von X verschwindet, was der Tatsache widerspricht, daß $\mathscr{P}^*$ ein Interpolationsraum ist. Also sind die[211] "Zeilen" von $P_0(X)$ linear unabhängig und[212] es gibt eine Teilmenge $X_0 \subseteq X$, $\#X_0 = \dim \mathscr{P}_0$, so daß $P_0(X_0)$ eine quadratische, invertierbare Matrix ist. Deren Inverse, $P_0(X_0)^{-1}$, hat Zeilen[213], die man als Koeffizientenvektor von Polynomen auffassen kann, so daß

$$N_0 := P_0(X_0)^{-1}\, P_0 \tag{8.6}$$

ein Vektor[214] von Polynomen mit der offensichtlichen Eigenschaft

$$N_0(X_0) = P_0(X_0)^{-1}\, P_0(X_0) = I.$$

Da wir zwischen den Elementen der Newton–Basis N_0 und denen der Punktmenge X_0 dadurch eine eineindeutige Zuordung haben, können wir wahlweise $N_0 = [n_x^* \,:\, x \in X_0]$ durch X_0 oder $X_0 = [x_n \,:\, n \in N_0^*]$ durch N_0^* indizieren, je nachdem, was uns gerade angemessener erscheinen mag.

Das war auch schon der erste Schritt. Um weiterzumachen definieren wir $X_1' = X \setminus X_0$ als Menge der "noch freien" Punkte[215], wählen eine beliebige Basis $\widetilde{P}_1$ von $\mathscr{P}_1 := \mathscr{P}^* \cap \Pi_{\gamma^1}$ und sorgen durch

$$P_1 = \widetilde{P}_1 - N_0^T\, \widetilde{P}_1(X_0), \qquad \text{d.h.} \qquad p := \widetilde{p} - \sum_{x \in X_0} \widetilde{p}(x)\, n_x, \quad \widetilde{p} \in \widetilde{P}_1,$$

[210] Die Freuden der totalen Ordnung …

[211] Mit p indizierten!

[212] Da $\dim \mathscr{P}_0 \leq \dim \mathscr{P} = \#X$ hat die Matrix weniger Zeilen als Spalten.

[213] Das ist bei einer Matrix nun eher weniger überraschend.

[214] Ob Vektor oder Menge – der einzige Unterschied ist, daß die Reihenfolge zählt oder nicht.

[215] Die Punkte in X_0 sind ja den Polynomen in N_0 zugeordnet worden.

dafür, daß

$$P_1(X_0) = \widetilde{P}_1(X_0) - \underbrace{N_0(X_0)}_{=I}\widetilde{P}_1(X_0) = \widetilde{P}_1(X_0) - \widetilde{P}_1(X_0) = 0$$

ist, so daß $P_1 \in \mathscr{P}_0 + \mathscr{P}_1 \subseteq \mathscr{P}^*$ auch schon wieder die erste Hälfte der Bedingung (8.4) erfüllt. Mit demselben Argument wie oben stellen wir nun wieder fest, daß die Matrix $P_1(X_1')$ den maximalen Rang $\#P_1 = \dim \mathscr{P}_1$ haben muss, denn $v^T P_1(X_1') = 0$ ergäbe ja wieder einen Koeffizientenvektor v zu einem Polynom aus $\mathscr{P}^*$, das an X verschwindet. Also gibt es wieder Punkte $X_1 \subseteq X_1'$, so daß $P_1(X_1')$ eine quadratische invertierbare[216] Matrix ist und

$$N_1 := P_1(X_1)^{-1} P_1 \tag{8.7}$$

ist der nächste Steinchen in unserem Newton–Mosaik.

Bleibt noch der allgemeine Schritt mit Index k, aber da passiert nichts neues mehr. Wir setzen

$$X_k' := X \setminus \bigcup_{j=0}^{k-1} X_j,$$

wählen eine Basis $\widetilde{P}_k$ von $\mathscr{P}_k := \mathscr{P}^* \cap \Pi_{\gamma^k}$ und sorgen durch[217]

$$P_k := \widetilde{P}_k - \left[\widetilde{P}_k(X_0) \dots \widetilde{P}_k(X_{k-1})\right] \underbrace{\begin{bmatrix} N_0(X_0) & \dots & N_0(X_{k-1}) \\ & \ddots & \vdots \\ & & N_{k-1}(X_{k-1}) \end{bmatrix}^{-1}}_{=\begin{bmatrix} I & * & * \\ & \ddots & * \\ & & I \end{bmatrix}} \begin{bmatrix} N_0 \\ \vdots \\ N_{k-1} \end{bmatrix}$$

wieder dafür, daß

$$P_k(X_j) = 0, \qquad j = 0, \dots, k-1.$$

Da auch $P_k(X_k)$ maximalen Rang $\#P_k = \dim \mathscr{P}_k$ hat, gibt es schließlich $X_k \subseteq X_k' \subseteq X$, so daß $P_k(X_k)$ invertierbar ist und wir erhalten auch

$$N_k := P_k(X_k)^{-1} P_k. \tag{8.8}$$

Kommt irgendjemandem diese Prozedur eigentlich bekannt vor? Ja, eigentlich ist das nichts anderes als die gute alte Gram–Schmidt–Orthogonalisierung, siehe z.B. (Golub & van Loan, 1996); man kann den Vorgang aber auch als "blockweise" Gauß–Elimination in der Vandermonde–Matrix $P(X)$ sehen, wobei P eine graduierte Basis von $\mathscr{P}^*$ ist, (Boor, 1994)

[216] Eigentlich redundant, denn jede invertierbare Matrix *muss* invertierbar sein.

[217] Für Anhänger der numerischen linearen Algebra: das ist nichts anderes als die gute alte *Rücksubstitution*; die Koeffizienten bezüglich der Newton–Basen sind wiederum enge Verwandte der *dividierten Differenzen*, wenn auch ohne Division, siehe (Sauer & Xu, 1995b; Sauer & Xu, 1995a).

Bemerkung 8.11 *Die Auswahl der Punktmengen X_k, $k = 0, \ldots, m$, ist im Normalfall* ***nicht*** *eindeutig, ganz im Gegenteil, der generische Fall ist, daß man jede beliebige Teilmenge von X'_k auswählen kann, solange nur deren Kardinalität passt. Die konkrete Wahl von X_k hat sehr viel mit* Pivotstrategien *zu tun, die man ja auch aus der numerischen linearen Algebra kennt und kann die numerische Stabilität und Qualität des Verfahrens durchaus beeinflussen.*

Satz 8.12 *Die oben konstruierten Polynome $N = [N_k \,:\, k = 0, \ldots, m]$ bilden eine Newton–Basis von von $\mathscr{P}^*$.*

Beweisen brauchen wir hier nichts mehr! Die Eigenschaft (8.4) haben wir durch die Konstruktion *erzwungen*, die Eigenschaft (8.5) war immer automatisch erfüllt, da wir uns nur im Normalformenraum bewegt haben.

Definition 8.13 *Die "kanonische" Newton–Basis der Normalformenraums $\mathscr{P}^*$ zu $X \subset \mathbb{K}^n$ bezeichnen wir mit $N^* = [N_k^* \,:\, k = 0, \ldots, m]$.*

8.3 Newton ist Gradreduktion!

Was aber ist nun so toll an Newton–Basen? Ganz einfach, sie sind das Hauptwerkzeug für die gradreduzierende Interpolation, und zwar schlechthin, nicht nur für die Normalformen.

Satz 8.14 *Ein homogen erzeugter Teilraum $\mathscr{P}$ ist genau dann ein gradreduzierender Interpolationsraum zu $X \subset \mathbb{K}^n$, wenn er eine Newton–Basis besitzt.*

Beweis: Der Trick des Beweises besteht darin, alles auf die "kanonische" Newton–Basis N^* des Normalformenraumes $\mathscr{P}^*$, siehe Definition 8.13, zurückzuspielen, die wir im letzten Abschnitt konstruiert haben.
"$\Rightarrow$": Zu unserem gradreduzierenden Interpolationsraum setzen wir einfach $N = L_{\mathscr{P}}(N^*)$ und da $N(X) = N^*(X)$ ist, ist die Eigenschaft (8.4) auch schon erfüllt. Schreiben wir $f \in \Pi$ als $f = g + \nu_{\mathscr{I}(X)}(f)$, $g \in \mathscr{I}(X)$, dann ist

$$L_{\mathscr{P}} f = L_{\mathscr{P}} \left(g + \nu_{\mathscr{I}(X)}(f) \right) = \underbrace{L_{\mathscr{P}} g}_{=0} + L_{\mathscr{P}} \nu_{\mathscr{I}(X)}(f) = L_{\mathscr{P}} \nu_{\mathscr{I}(X)}(f)$$

und somit wegen der Gradreduktion von $L_{\mathscr{P}}$ und der Normalform

$$\delta\left(L_{\mathscr{P}} f\right) = \delta\left(L_{\mathscr{P}} \nu_{\mathscr{I}(X)}(f)\right) \leq \delta\left(\nu_{\mathscr{I}(X)}(f)\right) = \delta\left(\nu_{\mathscr{I}(X)}(L_{\mathscr{P}} f)\right) \leq \delta\left(L_{\mathscr{P}} f\right),$$

also $\delta\left(L_{\mathscr{P}} f\right) = \delta\left(\nu_{\mathscr{I}(X)}(f)\right) =: \gamma$. Beide Leitterme haben also denselben Grad $\gamma = \gamma_k$ und es ist

$$q_x := \lambda(n_x) - \lambda(n_x^*) \in V_\gamma\left(\mathscr{I}(X)\right), \qquad x \in X_k.$$

Daraus erhalten wir aber sofort, daß

$$\begin{aligned}
\Pi_\gamma &= \text{span}\,\{\lambda(n_x^*) \,:\, x \in X_k\} + V_\gamma\left(\mathscr{I}(X)\right) \\
&= \text{span}\,\{\lambda(n_x) - q_x \,:\, x \in X_k\} + V_\gamma\left(\mathscr{I}(X)\right) \\
&\subseteq \text{span}\,\{\lambda(n_x) \,:\, x \in X_k\} + \underbrace{\text{span}\,\{q_x \,:\, x \in X_k\}}_{\subseteq V_\gamma(\mathscr{I}(X))} + V_\gamma\left(\mathscr{I}(X)\right) \\
&= (\lambda(N) \cap \Pi_\gamma) + V_\gamma\left(\mathscr{I}(X)\right) \subseteq \Pi_\gamma,
\end{aligned}$$

also

$$\Pi_\gamma = (\lambda(N) \cap \Pi_\gamma) + V_\gamma\left(\mathscr{I}(X)\right),$$

was genau (8.5) ist.
"$\Leftarrow$": Daß $\mathscr{P} = \operatorname{span} N$ ein Interpolationsraum ist, sieht man ein, indem man die Matrix

$$N(X) = \begin{bmatrix} N_0(X_0) & N_0(X_1) & \dots & N_0(X_m) \\ N_1(X_0) & N_1(X_1) & \ddots & \vdots \\ \vdots & \ddots & \ddots & N_{m-1}(X_m) \\ N_m(X_0) & \dots & N_m(X_{m-1}) & N_m(X_m) \end{bmatrix} = \begin{bmatrix} I & * & \dots & * \\ 0 & I & \ddots & \vdots \\ \vdots & \ddots & \ddots & * \\ 0 & \dots & 0 & I \end{bmatrix}$$

als invertierbare obere Dreiecksmatrix identifiziert, so daß man das Interpolationspolynom zu $f \in \Pi$ als

$$L_{\mathscr{P}} f = [f(X_0) \dots f(X_m)]\ N(X)^{-T} \begin{bmatrix} N_0 \\ \vdots \\ N_m \end{bmatrix}$$

schreiben kann. Bleibt noch die Gradreduktion, und dafür brauchen wir (8.5) und setzen $N^* = \nu_{\mathscr{I}(X)}(N)$. Diese Normalformen haben dieselben Interpolationseigenschaften wie N, bilden also ebenfalls eine Newton–Basis und erfüllen $N^*(X) = N(X)$, das heißt, die Koeffizienten von $L_{\mathscr{P}} f$ und $L_{\mathscr{P}^*}$ bezüglich der Newton–Basen sind dieselben, nämlich $N(X)^{-1} f(X) = \left(N(X)^*\right)^{-1} f(X)$. Nun sagt uns aber (8.5) zusammen mit

$$\lambda\left(N_\gamma^*\right) - \lambda\left(N_\gamma\right) \in V_\gamma\left(\mathscr{I}(X)\right), \qquad \Gamma \in \Gamma_{\mathscr{P}},$$

daß

$$\gamma = \delta\left(N_\gamma^*\right) = \delta\left(N_\gamma^*\right), \qquad \gamma \in \Gamma_{\mathscr{P}} = \Gamma^*,$$

und weil die Normalformen gradreduzierend sind, muss auch $\mathscr{P}$ ein gradreduzierender Interpolationsraum sein. □

8.4 Berechnung der Idealbasis

Die Idee ist einfach: Wir wiederholen nur die Konstruktion der Newton–Basis aus Abschnitt 8.2, jetzt allerdings mit dem feinen Unterschied, daß wir $\Gamma_{\mathscr{P}}$ noch nicht kennen, sondern mitbestimmen müssen. Dazu erst mal eine kleine, allgemeine Vorbemerkung[218].

Lemma 8.15 *Jede Teilmenge Γ' eines wohlgeordneten Graduierungsmonoids hat ein kleinstes Element.*

Beweis: Für γ^0 betrachten wir die Menge $\Gamma'_1 = \{\gamma \in \Gamma' : \gamma < \gamma^0\}$. Ist $\Gamma'_1 = \emptyset$, dann ist γ^0 das gewünschte Minimalelement, ansonsten wählen wir ein $\gamma^1 \in \Gamma'_1$, das jetzt natürlich $\gamma^1 < \gamma^0$ erfüllt. Generell, im $k$ten Schritt, setzen wir $\Gamma'_{k+1} = \left\{\gamma \in \Gamma' : \gamma < \gamma^k\right\}$ und wählen, falls $\Gamma'_k \neq \emptyset$ ist, ein $\gamma^{k+1} \in \Gamma'_{k+1}$. Dieser Prozess liefert uns eine strikt absteigende Kette

[218] Eigentlich mehr der Vollständigkeit halber.

$\gamma^0 > \gamma^1 > \gamma^2 > \cdots$, die[219] nach endlich vielen Schritten abbrechen muss, was nur dann passieren kann, wenn $\Gamma'_{k+1} = \emptyset$ für irgendein k ist, und das zugehörige γ^k ist das gesuchte Minimum. □

Beginnen wir also mit unserer Konstruktion, und zwar mit $\gamma = \min \Gamma$, was ja nach Lemma 8.15 immer existieren muß[220], wählen eine möglicherweise unendliche[221] Basis P von Π_γ und betrachten die Matrix $P(X)$, die einen Rang hat, der irgendwo zwischen 0 und $\#X$ liegt. Damit finden wir aber in $P(X)$ eine *quadratische* und *invertierbare* Teilmatrix von maximalem Rang; zu dieser Matrix gehören Teilmengen[222] $P_\gamma \subseteq P$ und $X_\gamma \subseteq X$, so daß $P_\gamma(X_\gamma)$ eine (maximale) invertierbare Teilmatrix von $P(X)$ ist, und nach einer Umordnung von P und X haben wir mit $\overline{P_\gamma} := P \setminus P_\gamma$ und $\overline{X_\gamma} := X \setminus X_\gamma$ die Blockdarstellung

$$P(X) = \begin{bmatrix} P_\gamma(X_\gamma) & P_\gamma\left(\overline{X_\gamma}\right) \\ \overline{P_\gamma}(X_\gamma) & \overline{P_\gamma}\left(\overline{X_\gamma}\right) \end{bmatrix} = \begin{bmatrix} I & 0 \\ \overline{P_\gamma}(X_\gamma)\, P_\gamma(X_\gamma)^{-1} & I \end{bmatrix} \begin{bmatrix} P_\gamma(X_\gamma) & P_\gamma\left(\overline{X_\gamma}\right) \\ 0 & * \end{bmatrix}.$$

Aber auch der "$*$" unten rechts muss eine Nullmatrix sein, denn wäre er keiner, dann wäre der Rang der Gesamtmatrix größer als der von $P_\gamma(X_\gamma)$, also auch größer als der von $P(X)$ und somit ist[223]

$$\begin{bmatrix} P_\gamma(X_\gamma) & P_\gamma\left(\overline{X_\gamma}\right) \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} I & 0 \\ -\overline{P_\gamma}(X_\gamma)\, P_\gamma(X_\gamma)^{-1} & I \end{bmatrix} \begin{bmatrix} P_\gamma(X_\gamma) & P_\gamma\left(\overline{X_\gamma}\right) \\ \overline{P_\gamma}(X_\gamma) & \overline{P_\gamma}\left(\overline{X_\gamma}\right) \end{bmatrix}. \tag{8.9}$$

Also hat

$$Q_\gamma := \overline{P_\gamma} - \overline{P_\gamma}(X_\gamma)\, P_\gamma(X_\gamma)^{-1} P_\gamma$$

die schöne Eigenschaft, daß $Q_\gamma(X) = 0$ ist. Fassen wir zusammen, was wir haben:

- Teilmengen P_γ und X_γ, die eindeutige Interpolation ermöglichen.
- Eine Teilmenge Q_γ von P mit $Q_\gamma(X) = 0$, also $Q_\gamma \subset \mathscr{I}(X)$.
- Die beiden Polynommengen erzeugen Π_γ, das heißt, $\Pi_\gamma = P_\gamma \oplus Q_\gamma$.

Und das gibt auch schon unsere Zerlegung nach dem "Aschenputtel–Prinzip": Die eine Teilmenge, P_γ liefert uns ein Stückchen Newton–Basis N,

$$N_\gamma := P_\gamma(X_\gamma)^{-1} P_\gamma,$$

[219] Das ist dann wohl das Wohl der Wohlordnung - ist ja wohl in Ordnung!

[220] Wir wissen sogar mehr! Dieses γ muß nach Lemma 5.7 sein, aber um den Eindruck von Allgemeinheit zu erwecken und die generelle Idee zu sehen, ignorieren wir diese Tatsache einfach. Also: pssst!

[221] Für "normale" Graduierungen sind die Räume sogar endlichdimensional. Aber das ist nur ein praktischer Vorteil.

[222] **Achtung:** Weder P_γ noch X_γ muss eindeutig sein und wird es in interessanten Fällen auch nicht sein. Die hier vorzunehmende Auswahl – eigentlich nichts anderes als die gute alte Pivotsuche, die aus der numerischen linearen Algebra bekannt sein sollte – beeinflußt möglicherweise sehr stark die Schnelligkeit und numerische Stabilität des Verfahrens.

[223] Nicht vergessen: $\begin{bmatrix} 1 & 0 \\ x & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 \\ -x & 1 \end{bmatrix}$.

die andere, Q_γ, einen Teil einer Idealbasis Q; was nun das Töpfchen und was das Kröpfchen ist, ist ausschließlich eine Frage des Kontexts, also ob wir interpolieren oder Idealtheorie treiben wollen. Nehmen wir also N_γ zu N dazu, packen Q_γ zu Q und ersetzen X durch $X' = X \setminus X_\gamma$, eigentlich ganz genauso, wie wir es bei N^* auch gemacht haben.

Unser Q besteht ja aus Polynomen, die auf ganz X verschwinden, weswegen jedes Vielfache eines solchen Polynoms auch zu $\mathscr{I}(X)$ gehört, so daß wir uns die Räume $V_\eta(Q)$, $\eta > \gamma$ gar nicht mehr anzusehen brauchen, denn das sind Leitterme von Polynomen im Ideal. Damit ist unser nächster Grad aber

$$\gamma' := \min \{\eta \geq \gamma \ : \ W_\eta(Q) \neq \{0\}\}$$

und wir wählen P als Basis von $W_{\gamma'}(Q)$; mit Hilfe der Newton–Basis können wir dafür sorgen, daß $P(X_\eta) = 0$, $\eta \leq \gamma$, ohne den Grad eines Polynoms in P zu verändern, denn die homogenen Terme, mit denen wir gestartet sind, hatten ja den Grad $\gamma' > \gamma$. Und jetzt sehen wir uns die Matrix $P(X')$ an und finden wieder Interpolations- bzw. Idealpolynome. Dabei kann es durchaus passieren, daß $P_{\gamma'} = X_{\gamma'} = \emptyset$ oder aber $Q_{\gamma'} = \emptyset$ ist, aber nicht beides gleichzeitig, denn zusammen erzeugen $P_{\gamma'}$ und $Q_{\gamma'}$ ja schließlich den *nichttrivialen* Vektorraum $W_{\gamma'}(Q)$. Danach erfolgt wieder der "Update"

$$\begin{aligned} N &= N \cup P_{\gamma'} \left(X_{\gamma'}\right)^{-1} P_{\gamma'} \\ Q &= Q \cup Q_{\gamma'} \\ X' &= X' \setminus X_{\gamma'} \end{aligned}$$

und wir gehen in eine neue Iterationsrunde.

In jedem Schritt wird nun entweder X' verkleinert oder $\langle \lambda(Q) \rangle_h$ vergrößert, oder aber sogar beides. Andererseits ist aber beides nur *endlich oft*[224] möglich und deswegen terminiert der Algorithmus nach endlich vielen Schritten, was aber nur eintreten kann, wenn es irgendwann ein "finales" $\gamma \in \Gamma$ gibt, so daß

$$\Pi_\eta = V_\eta(Q), \qquad \eta > \gamma,$$

ist. Nach Konstruktion ist andererseits aber auch

$$\lambda(\mathscr{I}(X)) \cap \left(\bigoplus_{\eta \leq \gamma} \Pi_\eta \right) \subset \bigoplus_{\eta \leq \gamma} V_\eta(Q),$$

denn jedes Polynom aus dem Ideal, dessen Leitterm noch nicht zu einem $V_\gamma(Q)$ gehört hat, haben wir ja explizit zu unserem Ideal hinzugenommen. Zusammen mit $Q \subset \mathscr{I}(X)$ ergibt dies insgesamt[225], daß

$$\lambda(\mathscr{I}(X)) \subseteq V_\gamma(Q) \subseteq \lambda(\mathscr{I}(X)),$$

und wir haben unser abschließendes Resultat bewiesen.

Satz 8.16 *Die Menge $Q \subset \mathscr{I}(X)$ ist eine Γ–Basis für das Ideal $\mathscr{I}(X)$.*

[224] Was X angeht – das ist eine endliche Menge, was Q angeht – Noether lässt grüßen.

[225] Unter Berücksichtigung der Tatsache, daß $\langle \lambda(Q) \rangle_h = \bigoplus_{\gamma \in \Gamma} V_\gamma(Q)$ ist und mit Lemma 6.33

Division and multiplication were discovered. Algebra was invented and provided in interesting diversion for a minute or two. And then he felt the fog of numbers drift away, and looked up and saw the sparkling, distant mountains of calculus.

T. Pratchett, *Men at arms*

9 Computeralgebra und Wavelets

Zum Abschluss wollen wir uns noch kurz eine etwas unerwartete Anwendung von Computeralgebra, genauer gesagt, von Idealbasen ansehen, und zwar im Zusammenhang mit *Wavelets*. Um genau zu sein - Wavelets werden hier nie wirklich auftauchen, aber doch zumindest die wichtigste Zutat der (diskreten) der Waveletanalyse, nämlich *verfeinerbare Funktionen*, deren Approximationsverhalten sich durch die Nullstellen von Laurentidealen beschreiben lässt, und so werden wir auf dem Umweg über torische Ideale auch die Problematik der Idealbasen für Laurentpolynome kennenlernen. Aber immer schön der Reihe nach.

9.1 Verfeinerbare Funktionen

Eine Funktion $\phi : \mathbb{R}^n \to \mathbb{R}$ heißt *verfeinerbar*, wenn sie eine Lösung der Funktionalgleichung

$$\phi = \phi * a\,(\Xi\cdot) = \sum_{\alpha\in\mathbb{Z}^n} \phi\,(\Xi\cdot -\alpha)\; a_\alpha \tag{9.1}$$

ist, wobei $a = (a_\alpha : \alpha \in \mathbb{Z}^n)$, die sogenannte *Maske* der Verfeinerungsgleichung, eine „multiunendliche“ Folge[226] mit nur endlich vielen von Null verschiedenen Komponenten ist und $\Xi \in \mathbb{Z}^{n\times n}$ eine *expandierende Matrix*.

Definition 9.1 *Eine Matrix $A \in \mathbb{Z}^{d\times d}$ heißt* expandierend*, wenn alle ihre Eigenwerte strikt größer als* 1 *sein, also wenn*[227]

$$\lim_{k\to\infty} \left\|A^k\right\| = 0$$

für irgendeine vernünftige Matrixnorm gilt.

[226] Der leicht bescheuerte Name kommt daherm, daß im Fall $n = 1$ Folgen der Form $a = (a_j : j \in \mathbb{Z})$ als *doppeltunendlich* bezeichnet werden.

[227] Sozusagen rückwärts definiert ...

Oftmals ist es praktisch, auf (9.1) die (formale) Fouriertransformation

$$\widehat{f}(\xi) = \int_{\mathbb{R}^n} f(t)\, e^{-i\xi^T t}\, dt, \qquad \xi \in \mathbb{R}^d,$$

anzuwenden, die ganz analog zu (4.17) eine Faltung in eine Multiplikation verwandelt.

Lemma 9.2 *Für $u \in \mathbb{R}^n$ und eine invertierbare Matrix $A \in \mathbb{R}^{n\times n}$ gilt*

$$\left[f\left(\cdot - u\right)\right]^\wedge(\xi) = e^{-i\xi^T u}\, \widehat{f}(\xi) \tag{9.2}$$

$$\left[f\left(A\cdot\right)\right]^\wedge(\xi) = \frac{1}{|\det A|} \widehat{f}\left(A^{-T}\xi\right). \tag{9.3}$$

Beweis: Zwei ganz elementare Rechnungen:

$$\left[f\left(\cdot - u\right)\right]^\wedge(\xi) = \int_{\mathbb{R}^n} f\left(t-u\right)\, e^{-i\xi^T t}\, dt = \int_{\mathbb{R}^n} f\left(t\right)\, e^{-i\xi^T (t+u)}\, dt = e^{-i\xi^T u}\, \widehat{f}(\xi)$$

liefert (9.2) und

$$\begin{aligned}\left[f\left(A\cdot\right)\right]^\wedge(\xi) &= \int_{\mathbb{R}^n} f\left(At\right)\, e^{-i\xi^T t}\, dt = \frac{1}{|\det A|}\int_{\mathbb{R}^n} f(t)\, e^{-i\xi^T A^{-1} t}\, dt \\ &= \frac{1}{|\det A|}\int_{\mathbb{R}^n} f(t)\, e^{-i\left(A^{-T}\xi\right)t}\, dt = \frac{1}{|\det A|}\widehat{f}\left(A^{-T}\xi\right)\end{aligned}$$

ergibt (9.3). □

Mit Hilfe von Lemma 9.2 erhalten wir nun durch Fouriertransformation beider Seiten von (9.1), daß

$$\begin{aligned}\widehat{\phi}(\xi) &= \left[\sum_{\alpha\in\mathbb{Z}^n} \phi\left(\Xi\cdot - \alpha\right)\, a_\alpha\right]^\wedge(\xi) = \sum_{\alpha\in\mathbb{Z}^n} a_\alpha\, \left[\phi\left(\Xi\cdot - \alpha\right)\right]^\wedge(\xi) \\ &= \frac{1}{|\det \Xi|}\sum_{\alpha\in\mathbb{Z}^n} a_\alpha\, \left[\phi\left(\cdot - \alpha\right)\right]^\wedge\left(\Xi^{-T}\,\xi\right) = \frac{1}{|\det \Xi|}\sum_{\alpha\in\mathbb{Z}^n} a_\alpha\, e^{-i\xi^T\Xi^{-1}\alpha}\widehat{\phi}\left((\Xi^{-T}\,\xi\right) \\ &= \widehat{a}\left(\Xi^{-T}\xi\right)\, \widehat{\phi}\left(\Xi^{-T}\xi\right),\end{aligned}$$

wobei man das *trigonometrische Polynom*

$$\widehat{a}(\xi) := \frac{1}{|\det \Xi|}\sum_{\alpha\in\mathbb{Z}^n} a_\alpha\, e^{-i\alpha^T\xi}$$

als *Fouriertransformierte*[228] der Folge a auffassen kann. Es geht aber auch noch anders: Definieren wir zu a das *Symbol*[229] als das Laurentpolynom

$$a^*(z) = \sum_{\alpha\in\mathbb{Z}^n} a_\alpha\, z^\alpha, \qquad z \in \mathbb{C}_\times^n := \left(\mathbb{C}\setminus\{0\}\right)^n, \tag{9.4}$$

[228] Achtung: Das ist zwar eine Fouriertransformierte eines diskreten Objekts, aber nicht die diskrete Fouriertransformierte - letztere bildet ja beanntlich *endliche* Vektoren auf Vektoren gleicher Länge ab.

[229] Das Symbol ist bis auf den Exponenten -1 die sogenannte *z-Transformation* des Signals a, siehe z.B. (Föllinger, 2000; Sauer, 2003), also eigentlich eine z^{-1}-Transformation.

dann haben wir die offensichtliche Beziehung

$$\widehat{a}(\xi) = \frac{1}{|\det \Xi|}\, a^*\left(e^{-i\xi}\right), \qquad \xi \in \mathbb{R}^d, \tag{9.5}$$

und somit die Fouriertransformierte von (9.1) als

$$\widehat{\phi}(\xi) = \frac{1}{|\det \Xi|}\, a^*\left(e^{-i\xi}\right)\, \widehat{\phi}\left(\Xi^{-T}\xi\right) = \widehat{a}\left(\Xi^{-T}\xi\right)\, \widehat{\phi}\left(\Xi^{-T}\xi\right), \qquad \xi \in \mathbb{R}^d. \tag{9.6}$$

Eines allerdings sollte man bei der ganzen Geschichte beachten:

> *Mit ganz wenigen Ausnahmen*[230] *sind verfeinerbare Funktionen nicht* explizit *sondern lediglich* implizit*, also lediglich durch die Maske a gegeben. Daher ist es ein wesentliches Ziel, Eigenschaften verfeinerbarer Funktionen aus der Maske ablesen zu können.*

9.2 Approximationsordnung und Polynomerhaltung

Zu einer gegebenen Funktion ϕ ist $\phi\left(h^{-1}\cdot - \alpha\right)$ die um den Faktor $h > 0$ gestauchte und um $h\,\alpha$ verschobene Kopie von ϕ. Versuchen wir nun, eine Funktion $f \in C\left(\mathbb{R}^n\right)$ durch solche gestauchten und verschobenen Kopien zu approximieren, dann müssen wir eine Koeffizientenfolge c bestimmen, so daß

$$\|f - \phi * c\,(h\cdot)\| = \sum_{x\in\mathbb{R}^n} \left| f(x) - \sum_{\alpha\in\mathbb{Z}^n} \phi\left(hx - \alpha\right)\, c_\alpha \right| \tag{9.7}$$

möglichst klein wird. Die Frage ist natürlich: Wie klein? Nun, hier tauchen wieder unsere Polynome auf, denn wenn ϕ Polynome erzeugen kann, dann ist die *Approximationsordnung*, also die Rate, mit der (9.7) für $h \to 0$ ebenfalls gegen Null geht, besonders gut.

Definition 9.3 *Der von ϕ erzeugte* translationsinvariante Raum *ist als*

$$\mathbb{S}\left(\phi\right) := \left\{ \sum_{\alpha\in\mathbb{Z}^n} \phi\left(\cdot - \alpha\right)\, c_\alpha \;:\; c_\alpha \in \mathbb{R},\, \alpha \in \mathbb{Z}^n \right\}$$

definiert.

Grundlage der ganzen Theorie über Approximationsordnungen, die im Übrigen bereits auf Schoenberg im Kontext der guten alten B-Splines zurückgeht, ist die folgende Abschätzung.

Satz 9.4 *Hat $\phi \in C\left(\mathbb{R}^n\right)$ kompakten Träger und ist $\Pi_n \subset \mathbb{S}(\phi)$ für ein $n \in \mathbb{N}_0$, dann ist für $f \in C^{n+1}\left(\mathbb{R}^n\right)$*

$$d\left(f, \mathbb{S}_h(\phi)\right) := \inf_c \|f - \phi * c(h\cdot)\| \le C_\phi h^{n+1}\, \left\|f^{(n+1)}\right\|, \tag{9.8}$$

wobei die Konstante C_ϕ nur von ϕ, nicht aber von h oder f abhängt.

[230] Wie's der Zufall will, gehören dan auch mal wieder die Splines dazu.

Die Idee des Beweises[231] ist sehr einfach: Wir bilden, für einen festgehaltenen Punkt $x \in \mathbb{R}^n$, das Taylorpolynom der Ordnung n von f,

$$f(\cdot) = \sum_{|\alpha|\leq n} \frac{\partial^n f}{\partial x^\alpha}(x)\, \frac{(\cdot - x)^\alpha}{\alpha!} + \sum_{|\alpha|=n} \frac{\partial^{n+1} f}{\partial x^\alpha}(\xi_\alpha)\, \frac{(\cdot - x)^\alpha}{\alpha!},$$

stellen das Taylorpolynom in $\mathbb{S}(\phi)$ dar, und erhalten einen Fehler von der Größenordnung $\|\cdot - x\|_\infty^{n+1}$. Wegen des kompakten Trägers von ϕ wird die Abweichung an der Stelle x aber nur von endlich vielen der verschobenen und gestauchten Kopien von ϕ beeinflusst, was die Konstante C_ϕ definiert[232].

Stellt sich also die Frage, wie man nun erkennen kann, daß $\Pi_n \subset \mathbb{S}(\phi)$ ist und wie die zugehörigen Koeffizienten aussehen. Nun, man kann ja mal raten und einfach $c_\alpha = \alpha^\beta$ für ein $\beta \in \mathbb{N}_0^n$ mit $|\beta| \leq n$ setzen - das ist dann also die *monomiale Folge*. Für diese Folge ergeben die Poissonsche Summenformel[233] und die Ableitungsformel für die Fouriertransformierte[234], siehe (Katznelson, 1976; Yosida, 1965), daß

$$\phi * c = \sum_{\alpha\in\mathbb{Z}^n} \phi(\cdot - \alpha)\, \alpha^\beta = \sum_{\alpha\in\mathbb{Z}^n} i^{|\alpha|} \sum_{\gamma\leq\beta} \binom{\beta}{\gamma} \frac{\partial^{|\gamma|}\widehat{\phi}}{\partial\xi^\gamma}(2\pi\alpha)\,(ix)^{\beta-\gamma}\, e^{2\pi i\,\alpha^T x}.$$

Jeder Term für $\alpha \neq 0$ in der Summe auf der rechten Seite ist aber ein *trigonometrisches Polynom* und insbesondere 1-periodisch. Damit der Ausdruck also überhaupt ein Polynom sein kann, muss der zugehörige Koeffizient

$$\sum_{\gamma\leq\beta} \binom{\beta}{\gamma} \frac{\partial^{|\gamma|}\widehat{\phi}}{\partial\xi^\gamma}(2\pi\alpha)\,(ix)^{\beta-\gamma}, \qquad \alpha \neq 0$$

den Wert Null haben, was mit der Forderung

$$\frac{\partial^{|\gamma|}\widehat{\phi}}{\partial\xi^\gamma}(2\pi\alpha) = 0, \qquad \alpha \neq 0, \quad |\gamma| \leq n, \tag{9.9}$$

recht einfach zu erreichen ist. Ist außerdem noch wenigstens

$$\widehat{\phi}(0) \neq 0, \tag{9.10}$$

dann ist, nach Ersetzung von γ durch $\beta - \gamma$,

$$\phi * c(x) = \sum_{\gamma\leq\beta} \binom{\beta}{\gamma} \frac{\partial^{|\beta-\gamma|}\widehat{\phi}}{\partial\xi^{\beta-\gamma}}(0)\,(ix)^\gamma = \widehat{\phi}(0)\, x^\beta + q(x),$$

[231] Vollständig gibt's ihn, wenn auch nur in einer Variablen, in (Strang & Nguyen, 1996) oder in (Sauer, 2003; Sauer, 1999), vor allem aber bereits in (Schoenberg, 1973).

[232] Im wesentlichen sagt uns die, wieveiele von diesen Funktionen gebraucht werden, was wiederum direkt mit der Trägergröße von ϕ zusammenhängt.

[233] Die beweisen wir jetzt mal nicht.

[234] Dito

auch tatsächlich ein Polynom mit Leitterm β, da das Polynom $q(x)$ Linearkombination von Monomen x^γ mit $\gamma \leq \beta$ und $\gamma \neq \beta$ ist, bei denen also an mindestens einer Koordinate j aus $\gamma_j < \beta_j$ gelten muss[235]. Das sind sie dann auch schon, die berühmten *Strang-Fix-Bedingungen* aus (Strang & Fix, 1973).

Satz 9.5 *Erfüllt ϕ die* Strang-Fix-Bedingungen der Ordnung p*,*

1. $\widehat{\phi}(0) \neq 0$*,*
2. $\dfrac{\partial^{|\beta|}\widehat{\phi}}{\partial\xi^\beta}(2\pi\alpha) = 0$*,* $\alpha \neq 0$*,* $|\beta| \leq p$*,*

dann ist $\Pi_p \subset \mathbb{S}(\phi)$.

Keine Angst - wir haben die Analysis, sofern man hier bei diesen eher formalen und algebraischen Manipulationen überhaupt von „richtiger“ Analysis sprechen kann, fast geschafft! Unser Ziel ist es ja, herauszufinden, wann die *verfeinerbare* Funktion ϕ, Approximationsordnung n hat, also die Eigenschaft $\Pi_n \subset \mathbb{S}(\phi)$. Da sind die Strang-Fix-Bedingungen aus Satz 9.5 ja sehr nützlich, aber die „Philosophie“ der verfeinerbaren Funktionen sagt ja, daß diese nur sehr selten explizit bekannt sind, sondern daß man eigentlich nur die Maske a kennt und Eigenschaften von ϕ über diese beschreiben will. Um diesem hehren Ziel näherzukommen substituieren wir einmal (9.6) in die Strang-Fix-Bedingungen der Ordnung 0, die

$$0 = \widehat{\phi}(2\pi\alpha) = \widehat{a}\left(2\pi\Xi^{-T}\alpha\right)\,\widehat{\phi}\left(2\pi\Xi^{-T}\alpha\right), \qquad \alpha \in \mathbb{Z}^n \setminus \{0\}, \tag{9.11}$$

fordern. Als nächstes zerlegen wir $\mathbb{Z}^n$ in Restklassen modulo Ξ^T:

$$\mathbb{Z}^n = \bigcup_{\mu\in M} \mu + \Xi^T\mathbb{Z}^n, \qquad M = \Xi^T[0,1)^n \cap \mathbb{Z}^n.$$

Daß das alles passt, zeigt uns das nächste Resultat.

Lemma 9.6 *Der Quotientenraum $\mathbb{Z}^n/\Xi^T\mathbb{Z}$ enthält $|\det\Xi|$ Restklassen und M ist eine Menge von Repräsentatoren.*

Beweis: Sei $\Xi^T = PDQ$ die Smith-Zerlegung von Ξ^T gemäß Satz 3.4, dann bedeutet $\alpha = \beta + \Xi^T\gamma$ nichts anderes als

$$P^{-1}(\alpha - \beta) = DQ\gamma.$$

Nun sind P, Q unimodular und daher beide Bijektionen von $\mathbb{Z}^n$ nach $\mathbb{Z}^n$, was uns die equivalente Identität $\alpha' - \beta' = D\gamma'$ für zwei Elemente α', β' aus derselben Äquivalenzklasse liefert. Da $D \in \mathbb{Z}^{n\times n}$ eine Diagonalmatrix ist, sind die Repräsentanten für die Äquivalenzklasse dann gerade

$$\{0\} \cup \bigcup_{j=1^n} \{k\epsilon_j \,:\, 1 \leq k < d_{jj}\}$$

[235]Ich möchte hier nicht “$\gamma < \beta$” schreiben, da sich das leicht mit $\gamma_j < \beta_j$ für *alle* Indizes $j = 1,\ldots,n$, verwechseln liesse.

und davon gibt es gerade $|\det D| = |\det \Xi|$ Stück. Daß es zu jedem $\alpha \in \mathbb{Z}^n$ ein $\mu \in M$ mit $\alpha = \mu + \Xi^T\beta$ gibt, sieht man, indem man $\beta = \left\lfloor \Xi^{-T}\alpha \right\rfloor$, also $\Xi^{-T}\alpha = \beta + y$, $y \in [0,1)^n$, wählt, was dann auch sofort

$$\alpha = \Xi^T(\beta + y) = \mu + \Xi^T\beta, \qquad \mu = \Xi^T y = \alpha - \Xi^T\beta \in \mathbb{Z}^n,$$

liefert. Schließlich müssen wir noch zeigen, daß keine zwei Elemente von M zu derselben Restklasse gehören. Nehmen wir also an, daß $\mu - \mu' = \Xi^T\beta$ für $\beta \in \mathbb{Z}^n$ und $\mu = \Xi^T y$, $\mu' = \Xi^T y'$, dann ist wegen der Invertierbarkeit von Ξ

$$0 = \Xi^T(\beta - y + y') \qquad \Rightarrow \qquad \beta = y - y' \in (-1,1)^n \cap \mathbb{Z}^n = \{0\},$$

also $y = y'$. □

So, jetzt aber zurück zu (9.11), wo wir nun α durch die Darstellung $\alpha = \mu + \Xi^T\beta$ in unserer Restklassenarithmetik ersetzen und, wegen der 2π–Periodizität von $\widehat{a}$,

$$\begin{aligned} 0 &= \widehat{a}\left(2\pi\Xi^{-T}\mu + 2\pi\beta\right)\,\widehat{\phi}\left(2\pi\Xi^{-T}\mu + 2\pi\beta\right) \\ &= \widehat{a}\left(2\pi\Xi^{-T}\mu\right)\,\widehat{\phi}\left(2\pi\Xi^{-T}\mu + 2\pi\beta\right), \qquad \beta \in \mathbb{Z}^n, \end{aligned}$$

erhalten. Sind nun die (multiunendlichen) Vektoren

$$\Phi_\mu := \left[\widehat{\phi}\left(2\pi\Xi^{-T}\mu + 2\pi\beta\right) \,:\, \beta \in \mathbb{Z}^n\right], \qquad \mu \in M \setminus \{0\}, \tag{9.12}$$

ungleich dem Nullvektor[236], dann ist (9.11) plötzlich äquivalent zu

$$\widehat{a}\left(2\pi\Xi^{-T}\mu\right) = 0, \qquad \mu \in M \setminus \{0\}. \tag{9.13}$$

Das lässt sich nun iterieren und liefert so die folgende Beschreibung der Strang–Fix–Bedingungen über die Maske a.

Satz 9.7 *Ist*[237] $\Phi_\mu \neq 0$, $\mu \in M$, *dann erfüllt die verfeinerbare Funktion* ϕ *die Strang–Fix–Bedingungen der Ordnung* p *genau dann, wenn*

$$\frac{\partial^{|\beta|}\widehat{a}}{\partial\xi^\beta}\left(2\pi\Xi^{-T}\mu\right) = 0, \qquad |\beta| \le p, \quad \mu \in M \setminus \{0\}. \tag{9.14}$$

Beweis: Für $p = 0$ haben wir den Satz in (9.13) schon bewiesen, also verwenden wir jetzt Induktion über p. Für $p \ge 1$ und $|\gamma| = p$ erhalten wir so über die Leibniz–Regel, daß

$$\begin{aligned} \frac{\partial^p\widehat{\phi}}{\partial\xi^\gamma}(2\pi\alpha) &= \frac{\partial^p}{\partial\xi^\gamma}\left(\widehat{\phi}(\cdot)\,\widehat{a}(\cdot)\right)\left(2\pi\Xi^{-T}\alpha\right) \\ &= \sum_{\eta\le\gamma}\binom{\gamma}{\eta}\frac{\partial^{p-|\eta|}\widehat{\phi}}{\partial^{\gamma-\eta}\xi}\left(2\pi\Xi^{-T}\alpha\right)\underbrace{\frac{\partial^{|\eta|}\widehat{a}}{\partial^\eta\xi}\left(2\pi\Xi^{-T}\alpha\right)}_{=0,\,|\eta|<p} \\ &= \widehat{\phi}\left(2\pi\Xi^{-T}\alpha\right)\,\frac{\partial^{|\gamma|}\widehat{a}}{\partial^\gamma\xi}\left(2\pi\Xi^{-T}\alpha\right), \end{aligned}$$

[236] Wegen der Forderung $\widehat{\phi}(0) \neq 0$ ist übrigens automatisch auch $\Phi_0 \neq 0$ …

[237] Man kann zeigen, daß diese Bedingung im allgemeinen nötig ist, um die Richtigkeit des Satzes zu gewährleisten, das heißt, es gibt Beispiele für für Funktionen mit $\Phi_\mu = 0$ für ein $\mu \in M$, für die dann auch die Aussage des Satzes nicht mehr gilt. Das Stichwort hier heißt *Stabilität*.

und dasselbe Zerlegungsargument für α wie oben liefert (9.14). □

Aus diesem Resultat für trigonometrische Polynome kann man nun relativ einfach[238] auch eines für *algebraische Polynome* bekommen.

Korollar 9.8 *Ist $\Phi_\mu \neq 0$, $\mu \in M$, dann erfüllt die verfeinerbare Funktion ϕ die Strang–Fix–Bedingungen der Ordnung n genau dann, wenn*

$$\frac{\partial^{|\beta|} a^*}{\partial z^\beta}(z_\mu) = 0, \qquad |\beta| \leq p, \quad \mu \in M \setminus \{0\}, \tag{9.15}$$

wobei $z_\mu := e^{-2\pi\Xi^{-T}\mu}$ ist.

So, und das ist nun perfekte Idealtheorie: a^* muss zu einer Menge von Polynomen im Ring Λ der *Laurentpolynome* gehören, die durch Nullstellenbedingungen definiert ist, also in einem *Ideal*!

9.3 Quotientenideale

Um eine kompakte Beschreibung der durch (9.15) definierten Ideale zu bekommen, brauchen wir noch einen weiteren Begriff.

Definition 9.9 *Zu Idealen $\mathscr{I}, cJ \subseteq R$ in einem Ring R ist das* Quotientenideal $\mathscr{I} : \mathscr{J}$ *definiert als*

$$\mathscr{I} : \mathscr{J} := \{f \in R \,:\, f \cdot \mathscr{J} \subseteq \mathscr{I}\}.$$

Ein Quotientenideal[239] ist wieder ein Ideal, denn wenn $f\mathscr{J} \subset \mathscr{I}$ und $g\mathscr{J} \subset \mathscr{I}$, dann ist natürlich auch

$$(f+g)\mathscr{J} = f\mathscr{J} + g\mathscr{J} \subset \mathscr{I} + \mathscr{I} = \mathscr{I}$$

und für beliebiges p und ist außerdem

$$pf\mathscr{J} \subset p\mathscr{I} \subset \mathscr{I} \qquad \Rightarrow \qquad pf \in \mathscr{I} : \mathscr{J}.$$

Da für alle $f \in \mathscr{I}$ ja auch $f\mathscr{J} \subseteq f \cdot R \subseteq \mathscr{I}$ ist, schließlich ist $\mathscr{I}$ ein Ideal, gilt insbesondere

$$\mathscr{I} \subseteq \mathscr{I} : \mathscr{J}. \tag{9.16}$$

Geometrisch entspricht das Quotientenideal der Differenz der Varietäten, allerdings unter Berücksichtigung der Vielfachheit und nur im Sinne des *Zariski-Abschlss*. Was das genau ist, findet sich in (Cox *et al.*, 1996), „einfach“ gesagt ist der Zariski-Abschluss einer Menge die kleinste Varietät, die diese Menge enthält. Anstatt das jetzt alles formal anzugehen, sehen wir uns einfach mal Beispiele an.

[238] Ein bisschen was ist schon noch zu tun, denn bei jedem Differenzieren erhält man ja einen linearen Term dazu, der dann mitberücksichtigt werden muss. Aber schlimm ist das nicht, siehe z.B. (Cotronei & Sauer, 2007) für den Fall $\Xi = 2I$.

[239] Bitte nicht verwechseln mit dem *Quotientenraum* $\Pi/\mathscr{I}$ *zu* einem Ideal $\mathscr{I}$!

Beispiel 9.10 (Quotientendideale)

1. *Im Hauptidealring univariater Polynome ist $\langle fg\rangle : \langle g\rangle = \langle f\rangle$, also genau so, wie man sich einen „anständigen“ Qoutienten vorstellt. Damit „entfernt“ auch $\left\langle (x-\xi)^k\right\rangle : \left\langle (x-\xi)^\ell\right\rangle$ einfach ein bisschen was von der Vielfachheit der Nullstelle.*

2. *Bei „richtigen“ Varietäten ist das anders. Betrachten wir $\mathscr{I} = \langle xy-1\rangle$ in $\mathbb{K}[x,y]$, dann ist die Hyperbel $y = \frac{1}{x}$ gerade die zugehörige Varietät $V(\mathscr{I})$. Mit $\mathscr{J} = \langle x-1, y-1\rangle$, also $V(\mathscr{J}) = \{1\} \subset V(\mathscr{I})$ ist aber $\mathscr{I} : \mathscr{J} = \mathscr{I}$ und daher*

$$V(\mathscr{I} : \mathscr{J}) = V(\mathscr{I}) \neq V(\mathscr{I}) \setminus V(\mathscr{J}).$$

Man kann's auch anders sehen: Varietäten mit „einpunktigem“ Loch gibt's halt nicht.

Übung 9.1 Zeigen Sie: Hat $p \in \mathbb{K}[x,y]$ die Eigenschaft, daß $(x-1)p(x,y), (y-1)p(x,y) \in \langle xy-1\rangle$, dann ist auch schon $p \in \langle xy-1\rangle$. ♢

Wo das Quotientenideal allerdings wirklich hilfreich ist, das sind die nulldimensionalen Ideale, und die können wir mit unseren Kenntnissen auch in Angriff nehmen.

Proposition 9.11 *Sind $\mathscr{I}$, $\mathscr{J}$ nulldimensionale, dann ist*

$$V(\mathscr{I} : \mathscr{J}) = V(\mathscr{I}) \setminus V(\mathscr{J}). \tag{9.17}$$

Beweis: Ist $\mathscr{I}$ ein nulldimensionales Ideal, dann ist der Quotientenraum $\mathscr{P} = \Pi/\mathscr{I}$ endlichdimensional und hat Multiplikationstabellen. Mit deren Hilfe können wir wie in Satz 7.10 zum Radikal übergehen. Das hat Multiplikationstabellen und deren gemeinsame Eigenwerte definieren eine endliche Menge $X \subset \overline{\mathbb{K}}^n$ im algebraischen Abschluss von $\mathbb{K}$, so daß $\sqrt{\mathscr{I}} = I(X)$, also auch $V(\mathscr{I}) = X$ ist. Analog erhalten wir $Y \subset \overline{\mathbb{K}}^n$ mit $V(\mathscr{J}) = Y$.

Wählen wir nun ein beliebiges Polynom f, das an $X \setminus Y$ verschwindet und multiplizieren wir dieses mit $g \in \mathscr{J}$, so folgt wegen $g(Y) = 0$ natürlich auch $(fg)(X) = 0$, also $fg \in \mathscr{I}$ und damit $f \in \mathscr{I} : \mathscr{J}$, weswegen[240]

$$\mathscr{I} : \mathscr{J} \supseteq I(X \setminus Y) \qquad \Leftrightarrow \qquad V(\mathscr{I} : \mathscr{J}) \subseteq X \setminus Y$$

sein muss. Für jeden Punkt $x \in X \setminus Y$ gibt es aber[241] ein $g \in \mathscr{J}$ mit $g(x) \neq 0$ und deswegen hat ein f mit $f(x) \neq 0$ die Eigenschaft, daß $(f \cdot \mathscr{J})(x) \neq 0$ ist wesewegen $f \notin \mathscr{I} : \mathscr{J}$ ist. Also ist auch

$$\mathscr{I} : \mathscr{J} \subseteq I(X \setminus Y) \qquad \Leftrightarrow \qquad V(\mathscr{I} : \mathscr{J}) \supseteq X \setminus Y,$$

was den Beweis komplettiert. □

[240] Es gilt bekanntlich die - gar nicht einmal schwer nachzuweisende Equivalenz zwischen $\mathscr{I} \subseteq \mathscr{I}$ und $V(\mathscr{I}) \supseteq V(\mathscr{J})$, siehe (Cox *et al.*, 1996).

[241] Sonst wäre nämlich $x \in Y$, was schon rein semiotisch nicht sinnvoll ist.

Bemerkung 9.12 *Die Aussage von Proposition 9.11 lässt sich sogar verfeinern, sie gilt dann für Vielfachheiten: Wenn eine Nullstelle in $V(\mathscr{J})$ mit Vielfachheit*[242] *ℓ und in $V(\mathscr{I})$ mit Vielfachheit k auftaucht, dann hat sie im Quotientenideal Vielfachheit $(k-\ell)_+$ - klar, kleiner als Null kann die Vielfachheit einer Nullstelle nicht werden.*

So, jetzt nur noch ein klein wenig Notation und wir sind genau da, wo wir hinwollen.

Definition 9.13

1. *Zu einer Matrix $A \in \mathbb{Z}^{n\times k}$ mit Spaltenvektoren a_j, $j = 1,\ldots,k$, definieren wir das Ideal*

$$\left\langle z^A - 1\right\rangle := \langle z^{a_j} - 1 \ : \ j = 1,\ldots,k\rangle. \tag{9.18}$$

2. *Die k-te Potenz eines Ideals $\mathscr{I}$ ist das Ideal*

$$\mathscr{I}^k = \left\langle f^k \ : \ f \in \mathscr{I}\right\rangle,$$

das von den k-ten Potenzen der Elemente von f erzeugt *wird.*

Damit haben wir alles in der Hand, was wir brauchen, um die gesuchte Eigenschaft idealtheoretisch zu beschreiben.

Satz 9.14 *Ist $\Phi_\mu \neq 0$, $\mu \in M$, dann erfüllt ϕ die Strang-Fix-Bedingungen der Ordnung p genau dann, wenn*

$$a^* \in \left(\left\langle z^\Xi - 1\right\rangle : \langle z-1\rangle\right)^{p+1} = \left\langle z^\Xi - 1\right\rangle^{p+1} : \langle z-1\rangle^{p+1}. \tag{9.19}$$

Natürlich wird der Beweis auf den Bedingungen (9.15) beruhen, aber um ihn führen zu können, werden wir uns erst einmal die Ideale in Λ ein wenig ansehen müssen; Satz 9.14, sein Beweis und die anderen Konzepte dieses Kapitels stammen übrigens aus (Möller & Sauer, 2004).

9.4 Laurentideale und deren polynomialer Anteil

Eigentlich erscheint die Sache so einfach: Die Monome sind *Einheiten* in Λ und daher können wir durch Multiplikation mit geeigneten Monomen jedes Laurentideal in ein Polynomideal transformieren. Nur müssen wir dabei ziemlich aufpassen, wie schon das Beispiel des Laurentpolynoms

$$f(x,y) = x\,y^{-1} - 1 = y^{-1}\,(x-y)$$

zeigt, das ja bis auf Einheit nichts anderes als $\widetilde{f}(x,y) = x - y$ ist. Aber: Das äquivalente Polynom $\widetilde{f}$ hat eine Nullstelle an $x = y = 0$, die für das Laurentpolynom tabu ist. So einfach geht es also nicht, wir müssen uns etwas anderes ansehen!

Definition 9.15 (Laurentpolynome und -ideale)

[242] Hier ist die skalare Vielfachheit k, die *Ordnung* von der die Funktionen dort verschwinden, also $q(D)f(x) = 0$, $q \in \Pi_k$.

1. *Der Ring der* Laurentpolynome *ist definiert als*
$$\Lambda = \{z^\alpha \Pi \ : \ \alpha \in \mathbb{Z}^n\} = \mathbb{K}\left[x_1, \ldots, x_n, y_1, \ldots, y_n\right] / \langle x_j y_j - 1 \ : \ j = 1, \ldots, n\rangle.$$
2. *Ein* Laurentideal *ist ein Ideal in* Λ.
3. *Der* Polynomteil $P(\mathscr{J})$ *eines Laurentideals* $\mathscr{J} \subseteq \Lambda$ *ist* $P(\mathscr{J}) := \mathscr{J} \cap \Pi$.
4. *Eine* (polynomiale) Basis F *des Laurentideals* $\mathscr{J}$ *ist eine endliche*[243] *Menge* $F \subset \Pi$ *so daß* $P(\mathscr{J}) = \langle F\rangle_\Pi$ *und somit auch* $\mathscr{J} = \langle F\rangle_\Lambda$.

Übung 9.2 Zeigen Sie: Der Polynomteil eines Laurenideals ist ein Polynomideal. ♢

Polynomteile von Laurentidealen sind nicht „Allerweltsideale“, sondern ganz besondere Ideale.

Proposition 9.16 *Ein Polynomideal* $\mathscr{I} \subseteq \Pi$ *ist genau dann Polynomteil eines Laurentideals* $\mathscr{J}$*, wenn für* $f \in \Pi$ *und* $1 \leq j \leq n$
$$z_j f(z) \in \mathscr{I} \qquad \Rightarrow \qquad f \in \mathscr{I} \tag{9.20}$$
gilt.

Beweis: Sei $\mathscr{I} = P(\mathscr{J}) \subset \mathscr{J}$. Da $z_j^{-1} \in \Lambda$ für $j = 1, \ldots, n$, haben wir
$$z_j f(z) \in \mathscr{I} \subset \mathscr{J} \quad \Rightarrow \quad f = z_j^{-1}\left(z_j f(z)\right) \in \mathscr{I} \cap \Pi = P(\mathscr{J}) = \mathscr{I}.$$
Sei umgekehrt $\mathscr{I}$ ein Polynomideal, das (9.20) erfüllt, $F \subset \mathscr{I}$ eine Basis von $\mathscr{I}$ und $\mathscr{J} := \langle F\rangle_\Pi$, so daß $\mathscr{I} \subseteq P(\mathscr{J})$. Wäre nun $P(\mathscr{J}) \neq \mathscr{I}$, dann gibt es Laurentpolynome $g_f \in \Lambda$, $f \in F$, so daß
$$g = \sum_{f \in F} g_f \, f \in P(\mathscr{J}) \setminus \mathscr{I}$$
wäre. Wählen wir nun aber ein Monom $m \in \Pi$, so daß $m\, g_f \in \Pi$, $f \in F$, dann ist
$$m\, g = \sum_{f \in F} (m\, g_f)\ f \in \langle F\rangle_\Pi = \mathscr{I},$$
und eine wiederholte Anwendung von (9.20) liefert den Widerspruch $g \in \mathscr{I}$. □

Diese Proposition liefert uns bereits einen Weg, Polynomteile eines Laurentideals $\mathscr{J}$ zu bestimmen: Wir beginnen mit einer Basis von $\mathscr{J}$, transformieren diese durch Multiplikation von geeigneten Monomen zu Polynomen und sehen nun nach, ob (9.20) für das von ihnen erzeugte Ideal $\mathscr{I}$ erfüllt ist, was wir übrigens in unserer schönen neuen Quotientenidealnotation[244] auch als
$$\mathscr{I} : \langle z_j\rangle = \mathscr{I}, \qquad j = 1, \ldots, n, \tag{9.21}$$
schreiben können. Gilt in (9.21) aber die strikte Inklusion „$\supset$“, dann erweitern wir $\mathscr{I}$ passend, indem wir die Basis von $\mathscr{I}$ zu einer Basis von $\mathscr{I} : \langle z_j\rangle$ erweitern; ein Algorithmus[245] zur Bestimmung der Basis eines Quotientenideals findet sich beispielsweise in (Cox *et al.*, 1996).

[243] Ja, der gute alte Hilbert gilt immer noch.
[244] Das ist ein Wort!
[245] Es wäre schön, den auch noch vorzustellen, aber dann fehlt uns die Zeit.

Beispiel 9.17 *Bei der sogenannten* $\sqrt{3}$–Subdivision *interessiert man sich für das Ideal* $\mathscr{J} = \langle xy^{-2}-1, x^2y^{-1}-1\rangle$ *und die zugehörigen polynomialisierten Basiselemente* $x-y^2$ *sowie* x^2-y *haben eine gemeinsame Nullstelle an* $x=y=0$*. Aber das merken wir eben auch im Polynomteil, der sich als*

$$P(\mathscr{J}) = \left\langle y^2-x, x^2-y, xy-1\right\rangle = \left\langle y^2-x, x^2-y\right\rangle + \langle xy-1\rangle$$

bestimmt, wobei das Zusatzpolynom sich gerade um die unerwünschte Nullstelle kümmert.

Proposition 9.18 (Laurentideale)

1. *Ein nulldimensionales Polynomideal* $\mathscr{I}$ *ist genau dann Polynomteil eines Laurentideals wenn* $\mathscr{I}(z) \neq 0$ *ist für alle* $z \in \mathbb{C}^n$ *mit* $\prod z_j = 0$*.*

2. *Für Laurentideale* $\mathscr{J}, \mathscr{J}'$ *gilt*

$$P(\mathscr{J} : \mathscr{J}') = P(\mathscr{J}) : P(\mathscr{J}') \tag{9.22}$$

und

$$P\left(\mathscr{J}^k\right) = P(\mathscr{J})^k, \qquad k \in \mathbb{N}. \tag{9.23}$$

Beweis: Als nulldimensionales Ideal hat $\mathscr{I}$ eine endlich Nullstellenmenge X und damit eine *Primärzerlegung* in

$$\mathscr{I} = \bigcap_{x\in X} \langle z-x\rangle^{k_x}, \qquad k_x \in \mathbb{N}, \quad x \in X.$$

Wegen (9.21) ist $\mathscr{I}$ genau dann Polynmteil eines Laurentideals, wenn $\langle z-x\rangle^{k_x} : \langle z_j\rangle = \langle z-x\rangle^{k_x}$, $j=1,\ldots,n$, $x\in X$, also

$$z_k \notin \sqrt{\langle z-x\rangle^{k_x}} = \langle z-x\rangle \qquad \Rightarrow \qquad x_k \neq 0$$

ist – und das ist dann auch schon die erste Behauptung.

Für (9.22) wählen wir $f \in P(\mathscr{J}) : P(\mathscr{J}')$, so daß $f\,P(\mathscr{J}') \subset P(\mathscr{J})$ und somit $\langle f\rangle_\Lambda \mathscr{J}' \subset \mathscr{J}$. Also ist

$$f \in (\mathscr{J} : \mathscr{J}') \cap \Pi = P(\mathscr{J} : \mathscr{J}')$$

das heißt, $P(\mathscr{J}) : P(\mathscr{J}') \subseteq P(\mathscr{J} : \mathscr{J}')$. Umgekehrt ist für jedes $f \in P(\mathscr{J} : \mathscr{J}')$ natürlich

$$\Pi \supset f\,P(\mathscr{J}') \subset f\,\mathscr{J}' \subset \mathscr{J} \quad \Rightarrow \quad f\,P(\mathscr{J}') \subset P(\mathscr{J})$$

und somit auch $P(\mathscr{J}) : P(\mathscr{J}') \supseteq P(\mathscr{J} : \mathscr{J}')$.

Da $\mathscr{J}^k \subseteq \mathscr{J}$ für jedes beliebige Ideal $\mathscr{J}$ in Λ und $k \in \mathbb{N}$, ist $V\left(\mathscr{J}^k\right) \supseteq V(\mathscr{J})$. Andererseits ist mit $f \in \mathscr{J}$ auch $f^k \in \mathscr{J}^k$ und $f^k(x) = 0$ impliziert $f(x) = 0$, das heißt, $x \in V(\mathscr{J})$, so daß auch $V\left(\mathscr{J}^k\right) \supseteq V(\mathscr{J})$ gilt, also

$$V\left(\mathscr{J}^k\right) = V(\mathscr{J}) \subset \mathbb{C}^n_\times,$$

letzteres wegen Teil 1). Nochmal Teil 1), diesmal in die andere Richtung, liefert dann, daß $P\left(\mathscr{J}^k\right)$ Polynomteil eines Laurentideals ist, weswegen, nach Proposition 9.16,

$$P\left(\mathscr{J}^k\right) : \langle z_j \rangle = P\left(\mathscr{J}^k\right), \qquad j = 1, \ldots, n$$

Ist außerdem F eine Basis von $P(\mathscr{J})$ und damit auch von $\mathscr{J}$, dann sind die Polynome

$$\left\{ \prod_{f \in F} f^{k_f} \ : \ \sum_{f \in F} k_f = k \right\} \subset P\left(\mathscr{J}\right)^k$$

auch eine Basis von $\mathscr{J}^k$ und nach Proposition 9.16 ergibt das schließlich (9.23). □

9.5 Das Nullstellenideal

Die Hauptarbeit liegt im Beweis der folgenden Beobachtung.

Proposition 9.19 *Für eine expandierende Matrix Ξ und $z_\mu := e^{-2\pi \Xi^{-T} \mu}$, $\mu \in M$, gilt*

$$\{f \ : \ f(z_\mu) = 0,\, \mu \in M\} = \left\langle z^\Xi - 1 \right\rangle. \tag{9.24}$$

Das folgende Lemma, das wir ohne Beweis angeben[246], ist das LU–Gegenstück zur Smith–Normalform.

Lemma 9.20 *Zu jeder invertierbaren Matrix $A \in \mathbb{Z}^{n \times n}$ gibt es eine unimodulare Matrix $X \in \mathbb{Z}^{n \times n}$ eine obere Dreiecksmatrix U mit $u_{jj} > 0$ und $u_{jk} \leq 0$, $j \neq k$, so daß $A = XU$ ist.*

Beweis von Proposition 9.19: Wir beginnen mit "$\supseteq$", wofür wir nur zu bemerken brauchen, daß für $\mu \in M$ und $j = 1, \ldots, n$

$$z_\mu^{\xi_j} \ = \ z_\mu^{\Xi \epsilon_j} = \left(e^{-2\pi \Xi^{-T} \mu} \right)^{\Xi \epsilon_j} = e^{-2\pi \epsilon_j^T \Xi^T \Xi^{-T} \mu} = e^{-2\pi \mu_j} = 1$$

ist.

Für die Umkehrung schreiben wir $\Xi = XU$ und führen die Variable $y = z^X$ ein, so daß wegen der ganzzahligen Invertierbarkeit von X

$$z^\Xi - 1 = 0 \qquad \Leftrightarrow \qquad y^U - 1 = 0,$$

so daß die Polynome

$$p_j(y) = y_j^{u_{jj}} - \prod_{k=j+1}^{n} y_k^{u_{jk}}, \qquad j = 1, \ldots, n,$$

[246] Aber der funktioniert genauso wie in Satz 3.4.

ebenfalls an den $|\det \Xi|$ Punkten $Y = Z^X := \left\{z_\mu^X \,:\, \mu \in M\right\}$, $Z = \{z_\mu \,:\, \mu \in M\}$, verschwinden. Damit[247] bilden die p_j eine lexikographische Gröbnerbasis für das Polynomideal

$$\{f \in \Pi \,:\, f(Y) = 0\} = P\left(\mathscr{J}'\right), \qquad \mathscr{J}' = \{f \in \Lambda \,:\, f(Y) = 0\},$$

und damit auch eine Basis für das Laurentideal $\mathscr{J}'$. Sei nun $f \in \Lambda$ mit $f(Z) = 0$, also $g(Y) = 0$ mit $g(y) = f\left(y^{X^{-1}}\right)$, und somit $g(y) = \boldsymbol{q}^T(y)\,\left[y^U - 1\right]$, dann ist

$$\begin{aligned} f(z) &= f\left(z^{X\,X^{-1}}\right) = f\left(y^{X^{-1}}\right) = g(y) = \boldsymbol{q}^T(y)\,\left[y^U - 1\right] = \boldsymbol{q}^T\left(z^X\right)\,\left[z^{XU} - 1\right] \\ &= \widetilde{\boldsymbol{q}}(z)\,\left[z^\Xi - 1\right], \end{aligned}$$

also $f \in \left\langle z^\Xi - 1\right\rangle$ und das alles in der wunderbaren Welt der Laurentpolynome. Damit ist aber auch "'$subseteq$" nachgewiesen. □

Übung 9.3 Zeigen Sie: Die Polynome

$$f_j(x) = x^{k_j} + p_j\left(x_{j+1}, \ldots, x_n\right), \qquad k_j > 0, \qquad j = 1, \ldots, n,$$

bilden eine lexikographische Gröbnerbasis und $\dim \Pi/\langle f_1, \ldots, f_n\rangle = k_1 \cdots k_n$. ♢

Beweis von Satz 9.14: Für $p \in \mathbb{N}_0$ sei $\mathscr{J}_p$ das Laurenpolynom, das die Bedingungen von (9.15) erfüllt und $\mathscr{I}_p := P\left(\mathscr{J}_p\right)$. Dann ist

$$\mathscr{J}_0 = \left\langle z^\Xi - 1\right\rangle : \langle z - 1\rangle_\Lambda$$

und, nach (9.22),

$$\mathscr{I}_0 = P\left(\left\langle z^\Xi - 1\right\rangle : \langle z - 1\rangle_\Lambda\right) = P\left(\left\langle z^\Xi - 1\right\rangle\right) : \langle z - 1\rangle_\Pi$$

Die Primzerlegung, siehe (Cox *et al.*, 1996), von $\mathscr{I}_0$ ist

$$\mathscr{I}_0 = \bigcap_{\mu \in M\setminus\{0\}} \langle z - z_\mu\rangle$$

und wegen der Komaxmalität der Ideale, (Cox *et al.*, 1996, S. 189), gilt für $k \in \mathbb{N}$ im Sinne der Polynomideale

$$\mathscr{I}_0^k = \left(\bigcap_{\mu \in M\setminus\{0\}} \langle z - z_\mu\rangle\right)^k = \bigcap_{\mu \in M\setminus\{0\}} \langle z - z_\mu\rangle^k,$$

und daher, wieder mit (9.22)

$$\begin{aligned} \mathscr{I}_p &= \mathscr{I}_0^{p+1} = \left(P\left(\left\langle z^\Xi - 1\right\rangle\right) : \langle z - 1\rangle\right)^{p+1} = P\left(\left[\left(\left\langle z^\Xi - 1\right\rangle\right) : \langle z - 1\rangle\right]^{p+1}\right) \\ &= P\left(\left\langle z^\Xi - 1\right\rangle^{p+1} : \langle z - 1\rangle^{p+1}\right) = P\left(\mathscr{J}_p\right), \end{aligned}$$

also $\mathscr{J}_p = \left\langle z^\Xi - 1\right\rangle^{p+1} : \langle z - 1\rangle^{p+1}$ wie behauptet. □

[247] Siehe die folgende Übung.

Uns ist in alten mæren
wunders viel geseit
von Helden lobebæren
von grôzer arebeit

Das Nibelungenlied

A Literatur

Basu, S., Pollack, R., Roy, M.-F. (2003). *Algorithms in Real Algebraic Geometry*, volume 10 of *Algorithms and Computation in Mathematics*. Springer.

Becker, O. (1933). Quellen und Studien zur Geschichte. *Math., Astron., Physik*, **B2**:311–333.

Boor, C. (2000). Computational aspects of multivariate polynomial interpolation: Indexing the coefficients. *Advances Comput. Math.*, **12**:289–301.

Boor, C. d. (1994). Gauss elimination by segments and multivariate polynomial interpolation. In Zahar, R. V. M., editor, *Approximation and Computation: A Festschrift in Honor of Walter Gautschi*, pages 87–96. Birkhäuser Verlag.

Boor, C. d. (2007). Interpolation from spaces spanned by monomials. *Advances Comput. Math.*, **26**:63–70.

Boor, C. d., Ron, A. (1990). On multivariate polynomial interpolation. *Constr. Approx.*, **6**:287–302.

Boor, C. d., Ron, A. (1991). On polynomial ideals of finite codimension with applications to box spline theory. *J. Math. Anal. and Appl.*, **158**:168–193.

Boor, C. d., Ron, A. (1992a). Computational aspects of polynomial interpolation in several variables. *Math. Comp.*, **58**(198):705–727.

Boor, C. d., Ron, A. (1992b). The least solution for the polynomial interpolation problem. *Math. Z.*, **210**:347–378.

Brieskorn, E. (1985). *Lineare Algebra und Analytische Geometrie II*. Vieweg.

Buchberger, B. (1965). *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenrings nach einem nulldimensionalen Polynomideal*. PhD thesis, Innsbruck.

Buchberger, B. (1970). An algorithmic criterion for the solvability of algebraic systems of equations (German). *Aequationes Math.*, **4**(3):374–383.

Buchberger, B. (1985). Gröbner bases: An algorithmic method in polynomial ideal theory. In Bose, N. K., editor, *Multidimensional Systems Theory*, pages 184–232. D. Reidel Publishing Company.

Buchberger, B. (1998a). An algorithmic criterion for the solvability of a system of algebraic equations. In Buchberger (1965), pages 535–545. Translation of (Buchberger, 1970) by M. Abrahamson and R. Lumbert.

Buchberger, B. (1998b). An introduction to gröbner bases. In Buchberger, B., Winkler, F., editors, *Groebner Bases and Applications (Proc. of the Conf. 33 Years of Groebner Bases)*, volume 251 of *London Math. Soc. Lecture Notes*, pages 3–31. Cambridge University Press. to appear.

CoCoATeam. CoCoA: a system for doing Computations in Commutative Algebra. Available at `http://cocoa.dima.unige.it`.

Cohen, A. M., Cuypers, H., Sterk, M., editors (1999). *Some Tapas of Computer Algebra*, volume 4 of *Algorithms and Computations in Mathematics*. Springer.

Cook, S. A. (1966). *On the Minimum Computation Time of Functions*. PhD thesis, Harvard University.

Cooley, J. W. (1987). The re–discovery of the Fast Fourier Transform. *Mikrochimica Acta*, **3**:33–45.

Cooley, J. W. (1990). How the FFT gained acceptance. In Nash, S. G., editor, *A History of Scientific Computing*, pages 133–140. ACM–Press and Addison–Wesley.

Cooley, J. W., Tukey, J. W. (1965). An algorithm for machine calculation of complex Fourier series. *Math. Comp.*, **19**:297–301.

Cotronei, M., Sauer, T. (2007). Full rank filters and polynomial reproduction. *Comm. Pure Appl. Anal.*, **6**:667–687.

Cox, D., Little, J., O'Shea, D. (1996). *Ideals, Varieties and Algorithms*. Undergraduate Texts in Mathematics. Springer–Verlag, 2. edition.

Cox, D., Little, J., O'Shea, D. (1998). *Using Algebraic Geometry*, volume 185 of *Graduate Texts in Mathematics*. Springer Verlag.

Eisenbud, D. (1994). *Commutative Algebra with a View Toward Algebraic Geometry*, volume 150 of *Graduate Texts in Mathematics*. Springer.

Farouki, R. T., Rajan, V. T. (1987). On the numerical condition of polynomials in Bernstein form. *Comput. Aided Geom. Design*, **4**:191–216.

Föllinger, O. (2000). *Laplace-, Fourier- und z–Transformation*. Hüthig.

Gantmacher, F. R. (1959a). *Matrix Theory. Vol. I.* Chelsea Publishing Company. Reprinted by AMS, 2000.

Gantmacher, F. R. (1959b). *Matrix Theory. Vol. II.* Chelsea Publishing Company. Reprinted by AMS, 2000.

Gardner, M. (1957). *Fads & fallacies*. Dover Publications. Originally published in 1952, *In the name of science*, by G. P. Putnam's Sons.

Gasca, M., Sauer, T. (2000a). On the history of multivariate polynomial interpolation. *J. Comput. Appl. Math.*, **122**:23–35.

Gasca, M., Sauer, T. (2000b). Polynomial interpolation in several variables. *Advances Comput. Math.*, **12**:377–410. to appear.

Gathen, J. v. z., Gerhard, J. (1999). *Modern Computer Algebra*. Cambridge University Press.

Gauss, C. F. (1816). Methodus nova integralium valores per approximationem inveniendi. *Commentationes societate regiae scientiarum Gottingensis recentiores*, **III**.

Golub, G., van Loan, C. F. (1996). *Matrix Computations*. The Johns Hopkins University Press, 3rd edition.

Gonzales-Vega, L., Rouillier, F., Roy, M.-F. (1999a). Symbolic recipes for polynomial system solving. In Cohen *et al.* (1999), chapter 2, pages 34–65.

Gonzales-Vega, L., Rouillier, F., Roy, M.-F., Trujillo, G. (1999b). Symbolic recipes for real solution. In Cohen *et al.* (1999), chapter 2, pages 121–162.

Gregory, R. T., Krishnamurthy, E. V. (1984). *Methods and Applications of Error–free Computation*. Texts and Monographs in Computer Science. Springer–Verlag, New York, Berlin, Heidelberg, Tokio.

Greuel, G.-M., Pfister, G. (2002). *A Singular Introduction to Commutative Algebra*. Springer.

Gröbner, W. (1968). *Algebraische Geometrie I*. Number 273 in B.I–Hochschultaschenbücher. Bibliographisches Institut Mannheim.

Gröbner, W. (1970). *Algebraische Geometrie II*. Number 737 in B.I–Hochschultaschenbücher. Bibliographisches Institut Mannheim.

Hardy, G. H., Wright, E. M. (1954). *An Introduction to the Theory of Numbers*. Oxford University Press, 3rd edition.

Higham, N. J. (1996). *Accuracy and stability of numerical algorithms*. SIAM.

Hilbert, D. (1890). Über die Theorie von algebraischen Formen. *Math. Ann.*, **36**:473–534.

Karatsuba, A., Ofman, Y. (1963). Multiplication of multidigit numbers on automata. *Sovjet Physics–Doklady*, **7**:595–596.

Katznelson, Y. (1976). *An Introduction to Harmonic Analysis*. Dover Books on advanced Mathematics. Dover Publications, 2. edition.

Khinchin, A. Y. (1964). *Continued fractions*. University of Chicago Press, 3rd edition. Reprinted by Dover 1997.

Knuth, D. E. (1998). *The Art of Computer Programming. Seminumerical Algorithms*. Addison–Wesley, 3rd edition.

Laaths, E., editor (1994). *Dante – La vita nuova, La Divina Comedia*. Weltbild Verlag. Zweisprachige Ausgabe: Italienisch / Deutsch.

Lorentz, R. A. (2000). Multivariate Hermite interpolation by algebraic polynomials: a survey. *J. Comput. Appl. Math.*, **122**:167–201. Numerical analysis 2000, Vol. II: Interpolation and extrapolation.

Macaulay, F. S. (1916). *The Algebraic Theory of Modular Systems*. Number 19 in Cambridge Tracts in Math. and Math. Physics. Cambridge Univ. Press.

Marcus, M., Minc, H. (1969). *A Survey of Matrix Theory and Matrix Inequalities*. Prindle, Weber & Schmidt. Paperback reprint, Dover Publications, 1992.

Marinari, M. G., Möller, H. M., Mora, T. (1996). On multiplicities in polynomial system solving. *Trans. Amer. Math. Soc.*, **348**(8):3283–3321.

McGeoch, C. (2001). Experimental analysis of algorithms. *Notices of the AMS*, **48**:304–311.

Möller, H. M. (1988). On the construction of Gröbner bases using syzygies. *J. Symbolic Comput.*, **6**:345–359.

Möller, H. M. (1998). Gröbner bases and Numerical Analysis. In Buchberger, B., Winkler, F., editors, *Groebner Bases and Applications (Proc. of the Conf. 33 Years of Groebner Bases)*, volume 251 of *London Math. Soc. Lecture Notes*, pages 159–178. Cambridge University Press. to appear.

Möller, H. M., Buchberger, B. (1982). The construction of multivariate polynomials with preassigned zeros. In Goos, G., Hartmanis, J., editors, *Computer Algebra, EUROCAM '82, European Computer Algebra Conference*, volume 144 of *Lecture Notes in Computer Science*, pages 24–31. Springer Verlag.

Möller, H. M., Sauer, T. (2000a). H–bases for polynomial interpolation and system solving. *Advances Comput. Math.*, **12**(4):335–362. to appear.

Möller, H. M., Sauer, T. (2000b). H–bases I: The foundation. In Cohen, A., Rabut, C., Schumaker, L. L., editors, *Curve and Surface fitting: Saint–Malo 1999*, pages 325–332. Vanderbilt University Press.

Möller, H. M., Sauer, T. (2000c). H–bases II: Applications to numerical problems. In Cohen, A., Rabut, C., Schumaker, L. L., editors, *Curve and Surface fitting: Saint–Malo 1999*, pages 333–342. Vanderbilt University Press.

Möller, H. M., Sauer, T. (2004). Multivariate refinable functions of high approximation order via quotient ideals of Laurent polynomials. *Advances Comput. Math*, **20**:205–228.

Möller, H. M., Stetter, H. J. (1995). Multivariate polynomial equations with multiple zeros solved by matrix eigenproblems. *Numer. Math.*, **70**:311–329.

Peña, J. M., Sauer, T. (2000a). On the multivariate Horner scheme. *SIAM J. Numer. Anal.*, **37**:1186–1197.

Peña, J. M., Sauer, T. (2000b). On the multivariate Horner scheme II: Running error analysis. *Computing*, **65**:313–322.

Perron, O. (1954). *Die Lehre von den Kettenbrüchen I.* B. G. Teubner, 3rd edition.

Sagan, C. (1989). *Unser Kosmos*. Droemersche Verlagsanstalt Th. Knaur Nachf. Deutsche Taschenbuchausgabe.

Sauer, T. (1995). Computational aspects of multivariate polynomial interpolation. *Advances Comput. Math.*, **3**(3):219–238.

Sauer, T. (1997). Polynomial interpolation of minimal degree. *Numer. Math.*, **78**(1):59–85.

Sauer, T. (1998). Polynomial interpolation of minimal degree and Gröbner bases. In Buchberger, B., Winkler, F., editors, *Groebner Bases and Applications (Proc. of the Conf. 33 Years of Groebner Bases)*, volume 251 of *London Math. Soc. Lecture Notes*, pages 483–494. Cambridge University Press.

Sauer, T. (1999). Splinekurven und –flächen im CAGD. Vorlesungsskript, Friedrich–Alexander–Universität Erlangen–Nürnberg, Justus–Liebig–Universität Gießen. `http://www.math.uni-giessen.de/tomas.sauer`.

Sauer, T. (2000a). Numerische Mathematik I. Vorlesungsskript, Friedrich–Alexander–Universität Erlangen–Nürnberg, Justus–Liebig–Universität Gießen. `http://www.math.uni-giessen.de/tomas.sauer`.

Sauer, T. (2000b). Numerische Mathematik II. Vorlesungsskript, Friedrich–Alexander–Universität Erlangen–Nürnberg, Justus–Liebig–Universität Gießen. `http://www.math.uni-giessen.de/tomas.sauer`.

Sauer, T. (2001). Gröbner bases, H–bases and interpolation. *Trans. Amer. Math. Soc.*, **353**:2293–2308.

Sauer, T. (2003). Digitale Signalverarbeitung. Vorlesungsskript, Justus–Liebig–Universität Gießen. `http://www.math.uni-giessen.de/tomas.sauer`.

Sauer, T. (2005). Kettenbrüche und Approximation. Vorlesungsskript, Justus–Liebig–Universität Gießen. `http://www.math.uni-giessen.de/tomas.sauer`.

Sauer, T. (2006). Polynomial interpolation in several variables: Lattices, differences, and ideals. In Buhmann, M., Hausmann, W., Jetter, K., Schaback, W., Stöckler, J., editors, *Multivariate Approximation and Interpolation*, pages 189–228. Elsevier.

Sauer, T., Wagenführ, D. (2006). Polynomial systems, H–bases, and an application from kinematic transforms. *Monografias del Seminario Matemático García de Galdeano*, **33**:185–196.

Sauer, T., Xu, Y. (1995a). A case study in multivariate Lagrange interpolation. In Singh, S., editor, *Approximation Theory, Wavelets and Applications*, NATO–ASI Proceedings, pages 443–452. Kluwer Academic Publishers.

Sauer, T., Xu, Y. (1995b). On multivariate Lagrange interpolation. *Math. Comp.*, **64**:1147–1170.

Schoenberg, I. J. (1973). *Cardinal Spline Interpolation*, volume 12 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM.

Schönhage, A. (1966). Multiplikation großer Zahlen. *Computing*, **1**:182–196.

Schönhage, A. (1977). Schnelle Multiplikation von Polynomen über Körpern der Charakteristik 2. *Acta Informatica*, **7**:395–398.

Schönhage, A., Strassen, V. (1971). Schnelle Multiplikation großer Zahlen. *Computing*, **7**:281–292.

Smyth, P. (1880). *The Great Pyramid. Its secrets and mysteries revealed*. Gramercy Books. Reprinted 1978.

Stetter, H. J. (1995). Matrix eigenproblems at the heart of polynomial system solving. *SIGSAM Bull.*, **30**(4):22–25.

Stetter, H. J. (2005). *Numerical Polynomial Algebra*. SIAM.

Stewart, I. (1975). *Concepts of Modern Mathematics*. Penguin Books. Dover reprint 1995.

Strang, G., Fix, G. (1973). A Fourier analysis of the finite element variational method. In *Constructive aspects of functional analysis*, pages 793–840. C.I.M.E, Il Ciclo 1971.

Strang, G., Nguyen, T. (1996). *Wavelets and Filter Banks*. Wellesley–Cambridge Press.

Toom, A. L. (1963). Die Komplexität eines logischen Netzes, das die Multiplikation ganzer Zahlen realisiert. *Dokl. Akad. Nauk SSSR*, **150**:496–498.

Trinks, W. (1978). Über B. Buchbergers Verfahren, Systeme algebraischer Gleichungen zu lösen. *J. Number Theory*, **10**:475–488.

Wilkinson, J. H. (1984). The perfidious polynomial. In Golub, G. H., editor, *Studies in Numerical Analysis*, volume 24 of *MAA Studies in Mathematics*, pages 1–28. The Mathematical Association of America.

Yosida, K. (1965). *Functional Analysis*. Grundlehren der mathematischen Wissenschaften. Springer–Verlag.

Index B