

März 2020

Computeralgebra Rundbrief

> Ausgabe 66

- ▶ Hecke: A Number Theory Package
- ▶ Eine Drohne kann die Wände eines Raumes hören
- ▶ Mit einem CAS Termumformungen motivieren
- ▶ Industrie-Tagung der Fachgruppe



Maple Companion



Auch verfügbar in Deutsch

Warum warten?

Versuchen Sie es doch mal!



Möchten Sie unsere Maple Companion app ausprobieren, haben aber kein Maple?

Holen Sie sich eine 15-tägige Testlizenz und schauen Sie, was die Companion app in Verbindung mit Maple leisten kann!

www.maplesoft.com/trial

Mathe auf Ihrem Smartphone!

Mit der kostenlosen Maple Companion App erhalten Sie Antworten auf Probleme aus den Bereichen Algebra, Elementarmathematik, Rechnungsarten, linearer Algebra, Differenzialgleichungen und mehr, alles mithilfe der Kamera auf Ihrem Handy. Und wenn Sie Maple Nutzer sind, können Sie den Maple Companion auch dazu verwenden, Mathe-Aufgaben in Maple zu importieren, wo Sie gesamte Leistung von Maple zum Lösen, Visualisieren und Erforschen Ihrer Mathe-Aufgabenstellungen nutzen können.

- Mathe-Probleme mit einem Kamera-Klick oder dem eingebauten Mathe-Editor eingeben.
- Integrale finden, Polynome faktorisieren, Matrizen invertieren, Gleichungssysteme und gewöhnliche Differentialgleichungen lösen, und vieles mehr.
- Plots erstellen, um das Verständnis zu vertiefen.
- Mathe-Aufgaben in Maple hochladen, um die gesamte Leistung von Maple zu nutzen.
- Fehler vermeiden, die beim Abschreiben von mathematischen Ausdrücken zu Maple von Hand auftreten können.



Inhaltsverzeichnis

Inhalt	3
Impressum	4
Mitteilungen der Sprecher	5
Tagungen der Fachgruppe	7
Themen und Anwendungen	9
<i>Eine Drohne kann die Wände eines Raumes hören (G. Kemper)</i>	9
Neues über Systeme	12
<i>HECKE: A Number Theory Package (C. Fieker, T. Hofmann, C. Sircana)</i>	12
Computeralgebra in der Schule	15
<i>Mit einem CAS Termumformungen motivieren (J. H. Müller)</i>	15
Berichte über Arbeitsgruppen	18
<i>SFB/TRR 195 Symbolic Tools in Mathematics and their Application (Part 5/5)</i>	18
Publikationen über Computeralgebra	19
Besprechungen zu Büchern der Computeralgebra	20
<i>Benjamin Hutz: An Experimental Introduction to Number Theory (M. Kreuzer)</i>	20
<i>Bican Xia, Lu Yang: Automated Inequality Proving and Discovering (M. Kreuzer)</i>	21
Promotionen in der Computeralgebra	22
Habilitationen in der Computeralgebra	23
Berichte von Konferenzen	24
Hinweise auf Konferenzen	25
Fachgruppenleitung Computeralgebra 2020–2023	27

Impressum

Der Computeralgebra-Rundbrief wird herausgegeben von der Fachgruppe Computeralgebra der GI in Kooperation mit der DMV und der GAMM (verantwortlicher Redakteur: Dr. Fabian Reimers car@mathematik.de)

Der Computeralgebra-Rundbrief erscheint halbjährlich, Redaktionsschluss 15.02. und 15.09. ISSN 0933-5994. Mitglieder der Fachgruppe Computeralgebra erhalten je ein Exemplar dieses Rundbriefs im Rahmen ihrer Mitgliedschaft. Fachgruppe Computeralgebra im Internet: <http://www.fachgruppe-computeralgebra.de>.

Konferenzankündigungen, Mitteilungen, einzurichtende Links, Manuskripte und Anzeigenwünsche bitte an den verantwortlichen Redakteur.

GI (Gesellschaft für
Informatik e.V.)
Wissenschaftszentrum
Ahrstr. 45
53175 Bonn
Telefon 0228-302-145
Telefax 0228-302-167
bonn@gi.de
<https://gi.de>



DMV (Deutsche Mathematiker-
Vereinigung e.V.)
Mohrenstraße 39
10117 Berlin
Telefon 030-20377-306
Telefax 030-20377-307
dmv@wias-berlin.de
<https://www.mathematik.de>



GAMM (Gesellschaft für Angewandte
Mathematik und Mechanik e.V.)
Technische Universität Dresden
Institut für Statik und Dynamik der
Tragwerke
01062 Dresden
Telefon 0351-463-33448
Telefax 0351-463-37086
GAMM@mailbox.tu-dresden.de
<https://www.gamm-ev.de>



Mitteilungen der Sprecher

Liebe Mitglieder der Fachgruppe Computeralgebra,

wie sicherlich kaum jemandem entgangen ist, fand Ende letzten Jahres turnusgemäß die Wahl der Fachgruppenleitung statt. Erstmals bestand neben der bewährten Briefwahl die Möglichkeit zur elektronischen Stimmabgabe, von der auch ein Großteil der Wähler Gebrauch machte. So durften die Wahlleiter Wolfram Koepf und Eva Zerz diesmal über die erfreulich hohe Zahl von 124 Personen berichten, die je bis zu 9 Stimmen abgegeben haben. Diese Stimmen verteilten sich folgendermaßen:

<i>Prof. Dr. Anne Frühbis-Krüger</i>	<i>Oldenburg</i>	<i>100</i>
<i>Prof. Dr. Gregor Kemper</i>	<i>München</i>	<i>95</i>
<i>Prof. Dr. Michael Cuntz</i>	<i>Hannover</i>	<i>94</i>
<i>Xenia Bogomolec</i>	<i>Hannover</i>	<i>91</i>
<i>Prof. Dr. Florian Heß</i>	<i>Oldenburg</i>	<i>88</i>
<i>Prof. Dr. Claus Fieker</i>	<i>Kaiserslautern</i>	<i>76</i>
<i>Prof. Dr. Jürgen Klüners</i>	<i>Paderborn</i>	<i>76</i>
<i>Prof. Dr. Martin Kreuzer</i>	<i>Passau</i>	<i>76</i>
<i>Prof. Dr. Max Horn</i>	<i>Siegen</i>	<i>75</i>
<i>Dr. Fabian Reimers</i>	<i>München</i>	<i>75</i>
<i>Dr. Thomas Hahn</i>	<i>München</i>	<i>65</i>

Die ersten 8 Kandidaten sind damit direkt gewählt, unter den beiden Nächstplatzierten musste das Los entscheiden, das auf Max Horn fiel. Wir danken allen Kandidaten für ihre Bereitschaft, sich zur Wahl zu stellen, den Wahlleitern Wolfram Koepf und Eva Zerz für die Durchführung und Auswertung der Wahl sowie ganz besonders Thomas Hahn für die Vorarbeiten und die technische Unterstützung, die die elektronische Stimmabgabe erst möglich gemacht haben. Gleichzeitig möchten wir an dieser Stelle auch den bisherigen Mitgliedern der Fachgruppenleitung Meinolf Geck und Jan Hendrik Müller, die nicht mehr zur Wahl angetreten waren, für ihr Engagement und die geleistete Arbeit danken.

Neben den gewählten Mitgliedern gehören der Fachgruppenleitung noch drei weitere Mitglieder an, die von den Fachgesellschaften als Vertreter entsandt werden. In der Wahlperiode 2020-2023 werden dies sein:

<i>Prof. Dr. Erika Abraham</i>	<i>Aachen</i>	<i>Vertreterin der GI</i>
<i>Prof. Dr. Florian Heß</i>	<i>Oldenburg</i>	<i>Vertreter der DMV</i>
<i>Prof. Dr. Eva Zerz</i>	<i>Aachen</i>	<i>Vertreterin der GAMM</i>

Mit der Entsendung von Florian Heß in die Fachgruppenleitung als Vertreter der DMV im Januar 2020, musste dieser sein gerade erneut errungenes Amt als gewähltes Mitglied niederlegen, so dass Fabian Reimers für ihn nachrückte. In der konstituierenden Sitzung am 10. Februar erfolgte dann die Wahl des Sprecherteams, das im Vergleich zur letzten Wahlperiode lediglich die Rollen von Sprecher(in) und Stellvertreter(in) tauschte. Die Benennung von Thomas Hahn als Fachexperten Physik machte dann das Team der Fachgruppenleitung fast komplett – einzig ein Fachexperte Schule wurde bisher noch nicht benannt.

Dem aufmerksamen Leser wird an dieser Stelle nicht entgangen sein, dass ein Name, der fast ein Vierteljahrhundert eine Konstante der Fachgruppenleitung war, in der Aufzählung erstmals fehlt: Wolfram Koepf zog sich nach seinem Ausscheiden aus dem aktiven Dienst an der Universität Kassel nun auch von seinem Amt als DMV-Vertreter in der Fachgruppenleitung zurück. Als langjähriger Sprecher der Fachgruppe und als Mitglied der Fachgruppenleitung lenkte er die Geschicke der Fachgruppe mit großem Geschick und viel Übersicht. Er organisierte federführend so viele Fachgruppentagungen in Kassel, dass es uns allen schwer fällt, uns eine solche Tagung ohne ihn und an einem anderen Ort vorzustellen. Doch auch im Hintergrund trug er maßgeblich zum reibungslosen Funktionieren so vieler Kleinigkeiten bei: Webserver der Fachgruppe, Mailinglisten und Mailadressen nach Bedarf, technische Infrastruktur für den Rundbrief. All dies lief ganz unauffällig auf einem Rechner in Kassel, der nach seinem Weggang jetzt bald abgeschaltet werden wird und für dessen Dienste inzwischen unterschiedliche dauerhafte Lösungen gefunden wurden. Lieber Wolfram, wir hoffen, auch weiterhin in heiklen Entschei-

dungen auf Deine Erfahrung zurückgreifen zu dürfen, und auch darauf, Dich noch auf vielen Tagungen und Workshops der Fachgruppe zu treffen.

Doch blicken wir nun in die Zukunft und dabei gar nicht so weit nach vorn. Noch in diesem Jahr wird die Fachgruppe nach langer Zeit einmal wieder eine Industrietagung veranstalten – aus aktuellem Anlaß zum Thema Kryptographie, siehe Seite 7. Auf der Jahrestagung der DMV in Chemnitz werden wir, wie es bereits fast Tradition ist, mit einem Minisymposium vertreten sein, über das Sie auf Seite 8 mehr erfahren. Schließlich laufen auch schon die ersten Vorbereitungen für die Fachgruppentagung im kommenden Jahr an, die ungewohnterweise nicht in Kassel, sondern in München stattfinden wird.

Nach derart vielen Ankündigungen und Verlautbarungen in eigener Sache wird es jetzt aber höchste Zeit, Sie nicht länger von der Lektüre des Heftes abzuhalten, das Ihnen diesmal interessante Beiträge zu der Mathematik hinter der Orientierung von Drohnen, zu dem Zahlentheoriepaket HECKE sowie zu CAS und Termumformungen in der Schule liefert. Wir wünschen Ihnen eine angenehme und anregende Lektüre.

Anne Frühbis-Krüger

Gregor Kemper

Tagungen der Fachgruppe



Industrie-Tagung der Fachgruppe

Oldenburg, 30.09.-01.10.2020 (Fachtagung)
28.09.-29.09.2020 (Schule)

www.fachgruppe-computeralgebra.de

Nach neunjähriger Pause veranstaltet die Fachgruppe Computeralgebra wieder eine Industrie-Fachtagung. Im Zuge der großen Herausforderungen an digitale Sicherheit durch neue Technologien wie KI, hoch performante binäre Systeme und Quantentechnologien haben wir uns entschieden, den Fokus auf Kryptografie zu legen.

Aktuell befindet sich die NIST¹ in der zweiten Runde der Standardisierung der neuen kryptografischen Algorithmen, Post-Quanten sichere Kryptografie genannt. Diese werden besondere Herausforderungen an Implementationen stellen und damit Experten brauchen, die tiefes Verständnis entsprechender Bereiche aus Mathematik, Algorithmik und Computing mitbringen – ein lohnendes, sich rasch entwickelndes Feld mit Zukunftsperspektiven für junge Absolventen und Promovierte.

Die Fachtagung soll Vertretern der Industrie die Möglichkeit geben, ihre aktuellen Arbeitsgebiete darzustellen und Kontakte zu Forschern und wissenschaftlichem Nachwuchs zu knüpfen, während die Schule interessiertem Nachwuchs einen Einblick in die mit der neuen Kryptografie verbundenen Themen geben und sie auf die Thematik der Fachtagung vorbereiten soll. Die Inhalte der Vorträge reichen von rein algorithmischen Fragen über Computing-Aspekte bis zu Kontext-Themen, wie z. B. der Notwendigkeit von Post-Quanten Kryptografie zur Realisierung von dynamischen QKD-Netzwerken.

Wir haben industrielle Vertreter aus verschiedensten Bereichen eingeladen, um dem Nachwuchs die Weite der Einsatzmöglichkeiten außerhalb der konventionellen wissenschaftlichen Pfade nahe zu bringen.

Bereits zugesagt haben ihre Teilnahme die folgenden Industrievertreter:

- Christian Wirth, www.privacybyblockchaindesign.com
- Cybersec Innovation Partners, www.cybersecip.com
- Marc Kaplan, VeriQloud, veriqcloud.com
- Stiepan Kovac, QRCrypto SA, qrcrypto.ch
- Dr. Peter Nonnenmann, DHBW, www.karlsruhe.dhbw.de

Xenia Bogomolec (Hanover)
Anne Frühbis-Krüger (Oldenburg)
Florian Heß (Oldenburg)

¹National Institute of Standard and Technologies, US Department of Commerce

Minisymposium Computeralgebra

DMV-Jahrestagung 2020, Chemnitz, 14.-17.9.2020

www.tu-chemnitz.de/mathematik/dmv2020/minisymp.php

Viele spannende neue Entwicklungen haben sich auch in den letzten Jahren wieder in der Computeralgebra ergeben. Der algorithmische Blickwinkel und der Einsatz vielfältiger moderner Computeralgebrasoftware hat z. B. einer Reihe von Gebieten der reinen Mathematik (z. B. innerhalb von Algebra, diskreter Mathematik, Geometrie und Zahlentheorie) neue experimentelle Zugänge erschlossen, was die Forschung um etliche interessante Facetten bereichert und in Rückkopplung mit der Entwicklung von CAS inzwischen eine eigene Dynamik gewonnen hat.

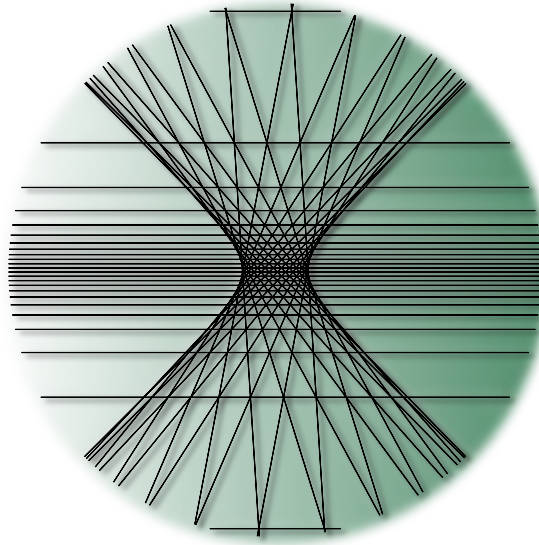


Abbildung 1: Aus einem der Anwendungsgebiete: ein simpliziales Arrangement auf einer kubischen Kurve

Wie schon in vergangenen Jahren wird die Fachgruppe Computeralgebra diesen Entwicklungen auch auf der DMV Jahrestagung in Chemnitz wieder mit einem thematisch breit aufgestellten Minisymposium Rechnung tragen. Dabei setzen wir bei der Auswahl der Vortragenden bewusst auf eine gute Mischung zwischen Nachwuchswissenschaftlern und erfahreneren Kollegen.

Als Vortragende des Minisymposiums konnten wir die folgenden Kolleginnen und Kollegen gewinnen:

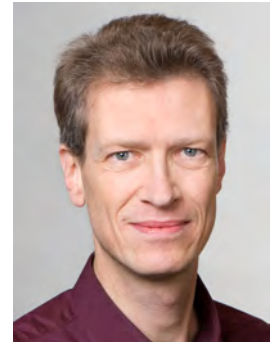
- Mohamed Barakat (Siegen): Algebraic Geometry, Cohomologische Methoden
- Melanie Harms (Aachen): System- und Kontrolltheorie
- Paul Mücksch (Bochum): Algebraische Kombinatorik, Hyperplane Arrangements
- Sergio Siccha (Kaiserslautern): Gruppentheorie
- Rainer Sinn (Berlin): Diskrete Geometrie
- Isabell Stenger (Saarbrücken): Klassische Algebraische Geometrie
- Timo de Wolff (Braunschweig): Applied Algebra, Optimization, Computational Algebraic Geometry

Michael Cuntz (Hannover)
Anne Frühbis-Krüger (Oldenburg)
Gregor Kemper (München)
Max Horn (Kaiserslautern)

Eine Drohne kann die Wände eines Raumes hören

Gregor Kemper, Technische Universität München

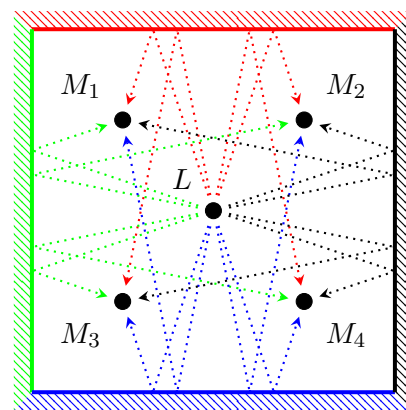
kemper@ma.tum.de



Einleitung

Man stelle sich eine Drohne vor, auf der vier Mikrophone montiert sind und die außerdem einen Lautsprecher trägt. Der Lautsprecher sendet einen hochfrequenten Schallimpuls aus, der an den Wänden des Raums, in dem die Drohne sich befindet, reflektiert wird. Die Mikrophone hören die Echos und messen die Laufzeiten zwischen Aussenden des Schalles und Eintreffen der Reflexion. Kann die Drohne anhand dieser Laufzeiten die Lage der Wände berechnen? Auch dann, wenn die Anzahl der Wände nicht bekannt ist und auch horizontale Wände (anders gesagt: Decken und Böden) sowie schräge Wände erlaubt sind? Mit dieser Frage haben sich Mireille Boutin und ich in einer kürzlich erschienenen Arbeit [2] beschäftigt. In diesem Artikel soll erklärt werden, wie Computeralgebra uns bei der Lösung des Problems geholfen hat.

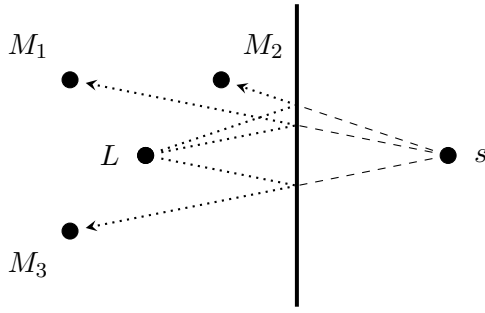
Für die Modellierung der Situation benutzen wir „Strahlenakustik“, also die bei hohen Frequenzen gerechtfertigte Annahme, dass sich der Schall wie Licht in der Strahlenoptik verhält. Außerdem berücksichtigen wir bei der Lageberechnung der Wände nur direkte Reflexionen („Echos erster Ordnung“), also keine Echos, die mehrfach zwischen Wänden hin und her geworfen wurden. Unsere Drohne muss also zwischen Echos erster Ordnung und Echos „höherer Ordnung“ unterscheiden können, wofür es (allerdings noch nicht perfekte) Techniken gibt. Eine weitere Annahme muss selbstverständlich noch getroffen werden: dass von einer Wand alle Echos erster Ordnung von sämtlichen vier Mikrophenen gehört werden—sonst ist die Lagebestimmung dieser Wand mit unserer Methode prinzipiell unmöglich. Die folgende 2D-Skizze stellt die Situation dar.



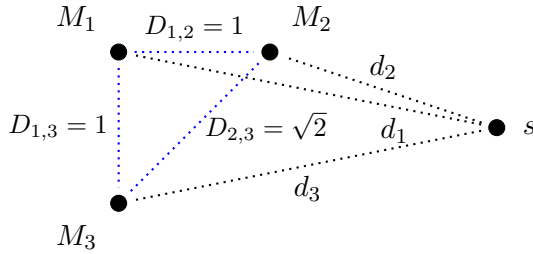
Hierbei sind die M_i und L die Positionen der Mikrophone bzw. des Lautsprechers, und die gepunkteten Linien stellen die Schallwege dar. Die Drohne selbst ist nicht eingezeichnet. In Wirklichkeit wird es mehr als vier Wände geben, und weniger Symmetrie. Die gemessenen Laufzeiten entsprechen den Strecken, die der Schall zurücklegt. Im Gegensatz zur Einfärbung der Schallwege in unserer Skizze klingen die Echos von den verschiedenen Wänden für jedes der Mikrophone allerdings gleich. Die Hauptaufgabe besteht nun darin, die empfangenen Echos einander zuzuordnen: Es muss eine Methode gefunden werden um herauszufinden, welche der von den verschiedenen Mikrophenen gehörten Echos von ein und derselben Wand stammen. Ist diese Zuordnung einmal gelungen, dann ist die Bestimmung der Wände (genauer: der Punkte, an denen die „Schallstrahlen“ reflektiert wurden) eine einfache geometrische Aufgabe, die der Positionsbestimmung mit GPS ähnelt.

Relationen

Für die Zuordnung der Echos erweisen sich nun Relationen zwischen den Schallstrecken als entscheidendes Hilfsmittel. Wir konzentrieren uns auf eine einzige Wand und betrachten nur drei Mikrophone, die im Zweidimensionalen genügen. (In vier Dimensionen bräuchte man fünf Mikrophone und so weiter.) In der folgenden Skizze ist der „Spiegelpunkt“ s eingezeichnet, also die an der Wand reflektierte Lautsprecherposition.



Die gemessenen Laufzeiten verhalten sich so, als käme der Schall direkt vom Spiegelpunkt s . Die Mikrophone und der Spiegelpunkt bilden eine Vier-Punkt-Konfiguration, bei der die Abstände d_1, d_2, d_3 die durch die Laufzeiten gemessenen Schallstrecken sind, und die Abstände $D_{1,2}, D_{1,3}, D_{2,3}$ zwischen den Mikrophen bekannt sind. Die Abstände sind in folgender Skizze markiert.



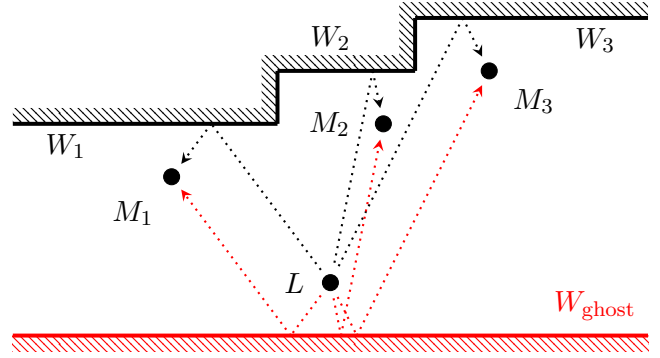
Nun sind die sechs Abstände einer ebenen Vier-Punkt-Konfiguration (ebenso wie die zehn Abstände einer räumlichen Fünf-Punkt-Konfiguration) nicht unabhängig, sondern erfüllen eine sogenannte Cayley-Menger-Relation. Im Fall unserer Skizze lautet diese

$$f_D(d_1^2, d_2^2, d_3^2) := (d_1^2 - d_2^2)^2 + (d_1^2 - d_3^2)^2 - 2(d_2^2 + d_3^2) + 2 = 0. \quad (1)$$

Bei gleichbleibender relativer Position der Mikrophone erfüllen die Schallstrecken d_i diese Relation, solange die Reflexion an ein und derselben Wand stattfindet. Diese Beobachtung führt direkt zu unserem Matching-Algorithmus, der die empfangenen Echos einander zuordnet. Wir begeben uns wieder in den tatsächlichen Fall von vier Mikrophen in 3D. Für alle Kombinationen (d_1, d_2, d_3, d_4) von über die Laufzeiten gemessenen Schallstrecken testet unser Algorithmus, ob $f_D(d_1^2, d_2^2, d_3^2, d_4^2) = 0$ gilt. Wenn ja, dann nimmt er an, dass die d_i von Reflexionen an ein und derselben Wand herkommen, die dann mit Standardmethoden bestimmt und von dem Algorithmus ausgegeben wird.

Geisterwände

Allerdings kann es passieren, dass Schallstrecken von verschiedenen Wänden zufällig die Relation erfüllen. Unser Algorithmus wird dann fälschlicherweise Wände bestimmen, die gar nicht existieren. So etwas nennt man Geisterwände, wie in folgender Skizze dargestellt.



Die schwarzen Wände W_1, W_2, W_3 zusammen mit den schwarzen Schallwegen sind die tatsächlich existierenden, während die ebenso langen roten Schallwege von der nicht existenten Geisterwand kämen. In dieser Situation wird jeder Algorithmus, dessen einziger Input die Laufzeiten sind, irrtümlicherweise die Geisterwand ausgeben. Dies führt zu der Frage, wie häufig das Phänomen von Geisterwänden ist. Hier stellen wir uns auf den Standpunkt, dass die Wände unseres Raums beliebig aber fest gegeben sind, aber die Position der Drohne, die die Mikrophone trägt, variiert. Wie sprechen von einer *guten Position*, wenn der Matching-Algorithmus keine Geisterwände ausgibt. Nun können wir das Hauptresultat von [2] formulieren.

Satz 1 Gegeben sei ein Raum, also eine Konfiguration von beliebig vielen Wänden, die auch horizontal oder schräg sein dürfen. In diesen Raum platzieren wir zufällig eine Drohne, die mit vier Mikrophen und einem Lautsprecher bestückt ist, wobei die Mikrophone nicht koplanar sein dürfen. Dann befindet sich die Drohne mit Wahrscheinlichkeit 1 in einer guten Position. Genauer befinden sich sämtliche schlechten Positionen in einer fünfdimensionalen Untervarietät des sechsdimensionalen Konfigurationsraums $\mathbb{R}^3 \times \text{SO}(3)$ aller möglicher Drohnenpositionen.

Man kann auch sagen, dass man mit einer infinitesimalen Bewegung die Drohne aus einer möglichen schlechten Position herauslösen kann. Da die Mikrophone fest auf der Drohne montiert sind, ist dies eine stärkere Aussage als für den Fall, dass sich die Mikrophone unabhängig voneinander bewegen können.

Um zu verstehen, welche Rolle die Computeralgebra in diesem Satz spielt, ist ein Blick auf die Beweismethoden erforderlich.

Rolle der Computeralgebra

Der Beweis von Satz 1 lässt sich auf folgende Schritte herunterbrechen.

1. Wir definieren eine Teilmenge $\mathcal{U} \subseteq \mathbb{R}^3 \times \text{SO}(3)$ von „sehr guten“ Drohnen-Positionen. Die Menge \mathcal{U} hängt von der gegebenen Konfiguration der Wände und der gegebenen Konfiguration der Mikrophone und des Lautsprechers auf der Drohne ab, und sie ist Zariski-offen. Es genügt daher $\mathcal{U} \neq \emptyset$ zu zeigen.

2. Man reduziert den Satz auf den Fall von vier Wänden.
3. Durch geeignete Koordinatisierung kann man die gegebene Konfiguration der vier Wände zusammen mit der gegebenen Konfiguration der Mikrophone und des Lautsprechers durch einen einzigen Punkt im \mathbb{R}^{14} darstellen.
4. Die Bedingung „ $\mathcal{M} = \emptyset$ “ an eine Konfiguration von Wänden, Mikrophonen und Lautsprechern definiert eine Untervarietät $\mathcal{V} \subseteq \mathbb{R}^{14}$. Außerdem betrachten wir die Untervarietät $\mathcal{V}' \subseteq \mathbb{R}^{14}$ gegeben durch die Bedingung: „Die Mikrophone sind koplanar“. Nun ist $\mathcal{V} \subseteq \mathcal{V}'$ zu zeigen.
5. Die Varietät \mathcal{V}' wird durch ein einziges Polynom g gegeben, und auch für \mathcal{V} lässt sich ein definierendes Ideal J relativ leicht aufstellen. Damit ist der Beweis auf die idealtheoretische Aussage

$$g \in \sqrt{J} \quad (2)$$

reduziert. Diese lässt sich prinzipiell mit Hilfe von Gröbnerbasen nachprüfen.

Damit läuft der Beweis auf den Nachweis von (2) hinaus. Das Aufstellen von g und J ist mit Hilfe eines Computeralgebrasystems nicht schwer. Wir haben MAGMA [1] benutzt. Allerdings arbeiten wir in einem Polynomring mit 14 Variablen, was den Test von (2) ohne zusätzliche Tricks und ad hoc Methoden unmöglich macht. Wir waren schließlich nach einigem Probieren erfolgreich, indem wir Homogenisierung, trunkierte Gröbnerbasen und eine geeignete Monomordnung (nicht grevlex) benutzten.

Es ist wichtig zu betonen, dass die für den Beweis erforderlichen Rechnungen zwar aufwändig sind, die

Drohne aber für das Matching der Echos und die Bestimmung der Wände nur sehr einfache Rechnungen ausführen muss.

Ausblick

Es würde die Anwendungsmöglichkeiten erweitern, wenn man Analoga von Satz 1 für den Fall hätte, dass die Mikrophone auf Bodenfahrzeugen, wie (selbstfahrende) Autos, Gabelstapler, oder mobile Roboter, montiert sind. Ein solches Ergebnis ist deutlich schwieriger zu erzielen, weil es bedeutet, dass drei Freiheitsgrade ausreichen, um durch eine infinitesimale Bewegung aus einer möglichen schlechten Position zu entweichen. Wir erwarten nicht, dass in der eingeschränkteren Situation Satz 1 in voller Allgemeinheit gilt.

Eine weitere Frage, die auf dem Weg zur tatsächlichen Anwendbarkeit zu klären ist, ist die Behandlung von (unvermeidlichen) Messfehlern. Da der Matching-Algorithmus eine Relation von der Art wie (1) benutzt, muss die Gleichheit durch eine sinnvolle näherungsweise Gleichheit ersetzt werden. Außerdem muss in den meisten Anwendungsszenarien mit von außen eingestreuten falschen Signalen gerechnet werden.

Literatur

- [1] Wieb Bosma, John J. Cannon, Catherine Playoust, *The Magma Algebra System I: The User Language*, J. Symb. Comput. **24** (1997), 235–265.
- [2] Mireille Boutin and Gregor Kemper, *A Drone Can Hear the Shape of a Room*, SIAM J. Appl. Algebra Geometry **4** (2020), 123–140.



HECKE: A Number Theory Package

C. Fieker, C. Sircana (Kaiserslautern University)
T. Hofmann (University des Saarlands)

fieker@mathematik.uni-kl.de
sircana@mathematik.uni-kl.de
thofmann@math.uni-sb.de



HECKE & OSCAR

What is HECKE or OSCAR? OSCAR is a new computer algebra system developed through the SFB-TRR 195 “Symbolic Tools”. The idea is to fuse the well established, mature projects GAP, POLYMAKE and SINGULAR as well as the comparatively new project HECKE [2] into a comprehensive combined system (OSCAR) where users are free to utilize all parts of the underlying “cornerstones” through a consistent interface that is driven by mathematics only. In the near future we envision more articles detailing other components of OSCAR as they are developed. A core feature of OSCAR is the use of a new programming language: OSCAR, that is, the glueing layer and the mathematical modelling, is written in JULIA [1]. The programming language JULIA is a relatively new language, developed originally at the MIT mathematics department for fast numerical mathematics. It features a just-in-time compiler and a strongly typed language with multiple dispatch. We’ll come back to what this means later.

HECKE itself started about 5 years ago with a small nucleus written in C: The ANTIC library for arithmetic in number fields [4]. This nucleus was then extended in JULIA and now comprises about 130,000 lines of code, covering classical algorithmic number theory, some geometry of numbers, class field theory and, more recently, also quadratic forms and associative algebras.

While developing HECKE, a strong emphasis was put on support of large degree fields: Most of the clas-

sical algorithms, as available in for example in MAGMA or PARI/GP were conceived and developed at a time when most computations applied to fields of degree 10 or less. On the other hand currently, driven partly by increase in processing power and partly by demands of cryptography (and other applications), computations in fields of degree larger than 100 are desirable, aiming for degree 1000.

The JULIA language through the strong typing and the JIT compiler allows for essentially the same execution speed as pure C code while at the same time being interactive, through its REPL. The downside of this is that the first time a function is executed, there is a delay while the compilation happens.

Installing

Installation of HECKE is made very easy through JULIA. First one needs to install JULIA in version at least 1.0¹. Then, once JULIA is installed and started, we do

```
julia> using Pkg
julia> Pkg.add("Hecke")
```

Now Hecke is installed. If you want to use the latest, cutting edge version, do

```
julia> using Pkg
julia> Pkg.add("Hecke#master")
```

instead. Now in order to load HECKE, one needs to enter

```
julia> using Hecke
```

¹available from <https://julialang.org/>

Note, that HECKE is build on and requires other packages as well, but they will be installed automatically.

Basics

Number fields are finite dimensional extensions of the rationals and this is the view taken by HECKE. Algorithmically we distinguish two cases depending on the base field

- the field K is given explicitly as an extension of \mathbb{Q} , that is, $K = \mathbb{Q}[t]/(f)$ for some suitable irreducible polynomial f ,
- the field K is given as an extension of an already constructed number field k ,

as well as two cases depending on the presentation

- the field K is given via some primitive element (or, more precisely, its minimal polynomial),
- the field K has multiple generators, that is, $K = \mathbb{Q}[\sqrt{2}, \sqrt{3}, \sqrt{5}]$, a so called non-simple field.

Internally, non-simple fields and their elements are represented using sparse multivariate polynomials, while fields given via primitive elements are densely represented. Of course, all fields can be converted to a simple extension of \mathbb{Q} , and while this is asymptotically the best presentation, many problems naturally require the other presentations.

```
julia> using Hecke
...
julia> Qt, t = QQ["t"];
julia> K, a =
    number_field(t^3+13t^2-13t+13)
(Number field over Rational Field
 with defining polynomial ...
julia> defining_polynomial(K)
t^3+13*t^2-13*t+13
julia> ans(a)
0
```

(Note that a ; after a command suppresses the output, and ans refers to the return value of the previous command.)

But also

```
julia> E, g =
    number_field([t^2-2, t^2-3, t^2-5],
    ["s2", "s3", "s5"]);
julia> [x^2 for x in g]
Array{NfAbsNSElem, 1}: 2 3 5
```

Of course, this non-simple field can be converted to the dense presentation, that is, we can find a primitive element:

```
julia> Es, phi = simple_extension(W);
julia> phi(gen(Es))
s2 + s3 + s5
```

So, phi is a map: $E_s \rightarrow E$, the element $\text{gen}(E_s)$ is the primitive element of the “new” field E_s and, as expected, in the “old” field the primitive element is the sum of the generators:

```
julia> sum(g)
s2 + s3 + s5
```

There are shortcuts available for common field constructions such as `quadratic_field` and `cyclotomic_field`.

We can now extend the field K further. In order to do so we define a polynomial ring over K and adjoin a root of an irreducible polynomial.

```
julia> Ks, s = K["s"];
julia> F, f = number_field(s^3-2a+a);
```

Class Field Theory

One of the original goals for HECKE was to perform computations in constructive class field theory [3]. To continue with the fields from above:

```
julia> class_group(K)
(GrpAb: Z/6, ClassGroup map...)
```

So the class group is non-trivial, hence we should be able to find a non-trivial Hilbert class field:

```
julia> H = hilbert_class_field(K)
Class field defined
mod (<1, 1>, InfPlc[])
of structure ... : Z/6
```

```
julia> number_field(H)
non-simple Relative number field over
...
with defining polynomials ...
[_$1^2+(2*_a^2 + 30*_a + 23),
_$2^3+(39*_a^2 + 546*_a)*_$2
+(-13*_a-182)]
```

As you can see (or notice when trying it out), the creation of the Hilbert class field was immediate—and it did nothing. The conversion to a number field however, then found defining equations. It is well known that in the Hilbert class field all ideals from the base field are principal. To check that, We first create an absolute field H_a , that is, a simple extension of \mathbb{Q} , which is isomorphic to the Hilbert class field. As a second step, we determine its class group and verify that ideals of K become principal.

```
julia> Ha, psi, inj =
    absolute_field(ans);
julia> CH, rho_H = class_group(Ha)
(GrpAb: Z/1, ...)
julia> CK, rho_K = class_group(K);
julia> for c in CK
    I = rho_K(c)
    J = inj(I)
    @show isprincipal(J)[1]
end
(isprincipal(J))[1] = true
...
```

This shows, indeed, that all ideals become principal! In general, to compute class fields, we start by computing ray class groups:

```
julia> R, mR =
    ray_class_group(7*maximal_order(K))
```

The presentation returned makes it hard to see the structure (however this is very useful computationally), but we can compute the SNF of this abelian group:

```
julia> snf(R)[1]
GrpAb: Z/6 x Z/26
julia> ray_class_field(mR)
Class field defined mod (<7, 7>, ...
with structure: Z/6 x Z/36
```

Of course, while here conversion to a number field is possible (the degree is not too large), in general we want to study subfields which correspond to quotients of R :

```
julia> q, mq =
    quo(R, [3*g for g in gens(R)]);
julia> ray_class_field(mR, mq)
Class field defined mod (<7, 7>, ...
... with structure: Z/3^2

julia> for s in subgroups(R, quotype=[3])
    q, mq = quo(R, s[1])
    A = ray_class_field(mR, mq)
    println(number_field(A))
end
non-simple Relative number field over
...
```

Given that K is a cubic field, the splitting field can at most be a quadratic extension, ramified at only the primes that already ramify in K . Let's see if we can find them!

```
julia> ZK = maximal_order(K)
julia> d = discriminant(ZK)
julia> R, mR = ray_class_group(d*ZK,
    real_places(K), n_quo = 2)

julia> for s in subgroups(R, quotype=[2])
    q, mq = quo(R, s[1])
    global A =
        ray_class_field(mR, mq)
    if isnormal(A)
        break
    end
```

```
end
julia> con = conductor(A)
(<179, _$-8>
Norm: 179
Minimum: 179
..., InfPlc[...])
```

Since the Galois group of K is dihedral, there should be a quadratic extension B of \mathbb{Q} contained in A . By Kronecker-Weber, B should be a subfield of a cyclotomic field:

```
julia> C =
    cyclotomic_field(ClassField,
        minimum(con[1]))
julia> B = maximal_abelian_subfield(A, QQ)
julia> issubset(B, C)
true
```

This is obviously only showing a very small part of the functionality of HECKE, even of the class field theory. This is meant to give an indication or feeling about the use of both HECKE and, eventually OSCAR. This example, and a couple more are also available as Jupyter notebooks from

<https://github.com/thofma/HeckeTutorials.jl>

The source code of HECKE itself is also freely available from

<https://github.com/thofma/Hecke.jl>

References

- [1] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, *Julia: A fresh approach to numerical computing*, SIAM review **59** (2017), no. 1, 65–98.
- [2] C. Fieker, W. Hart, T. Hofmann, and F. Johansson, *Nemo/Hecke: Computer Algebra and Number Theory Packages for the Julia Programming Language*, ISSAC'17—Proceedings of the 2017 ACM International Symposium on Symbolic and Algebraic Computation, ACM, New York, 2017, pp. 157–164.
- [3] C. Fieker, T. Hofmann, and C. Sircana, *On the construction of class fields*, Proceedings of the Thirteenth Algorithmic Number Theory Symposium, Open Book Ser., vol. 2, Math. Sci. Publ., Berkeley, CA, 2019, pp. 239–255.
- [4] W. Hart, *ANTIC: Algebraic Number Theory in C*, Computeralgebra Rundbrief **56** (2015), 10–11.



Mit einem CAS Termumformungen motivieren

J. H. Müller
(Rivius Gymnasium Attendorn)

mueller@rivius-gymnasium.de

Einführung

Im vorliegenden Artikel wird beschrieben, wie Termumformungen auf eine einfache Art mit Hilfe eines CAS motiviert werden können.

Termumformungen früh motivieren

Terme sollten in verschiedenen äquivalenten Darstellungsarten schon früh im Unterricht thematisiert werden. Im Kontext von Formeln, etwa zur Berechnung des Flächeninhalts eines Dreiecks mit bekannter Grundseite g und Höhe h , kann der Flächeninhalt auf verschiedene Arten in Form von

$$g \cdot h/2, \quad g/2 \cdot h, \quad \frac{1}{2} \cdot g \cdot h$$

oder

$$(g \cdot h)/2$$

geschrieben werden. Trotz der Äquivalenz dieser Terme, können sie herleitungsbedingt geometrisch unterschiedlich interpretiert werden. Zerteilt man ein Dreieck auf verschiedene Arten in flächeninhaltsgleiche

Rechtecke, ergeben sich zerlegungsbedingt verschiedene äquivalente Flächeninhaltsformeln. Abb. 1 zeigt zwei Zerlegungsmöglichkeiten, die auf die Terme $g \cdot h/2$ und $g/2 \cdot h$ führen können, deren Äquivalenz anschließend mit Hilfe von Rechenregeln und Rechengesetzen nachgewiesen werden kann.

Warum hier ein CAS einsetzen?

Neben der Möglichkeit zur Vernetzung von Geometrie und Algebra existiert hier eine Möglichkeit zum Einsatz von CAS und kann anhand von Computergrafiken motiviert werden: Die Beschreibung von Oberflächen mit Hilfe von Dreiecken wird u.a. in Computerspielen genutzt und "Tessellation" genannt. Je leistungsfähiger die Grafikkarte in einem Computer ist, desto detaillierter kann diese Unterteilung sein und desto realistischer wirken die darzustellenden Figuren. Aktuelle Grafikkarten verarbeiten etwa 7 Mrd. Dreiecke pro Sekunde (Stand 2019, vgl. <https://t1p.de/ujx0>). Das entspricht bei einer Bildfrequenz von 60 Bildern pro Sekunde über 100 Millionen Dreiecken pro Bild.

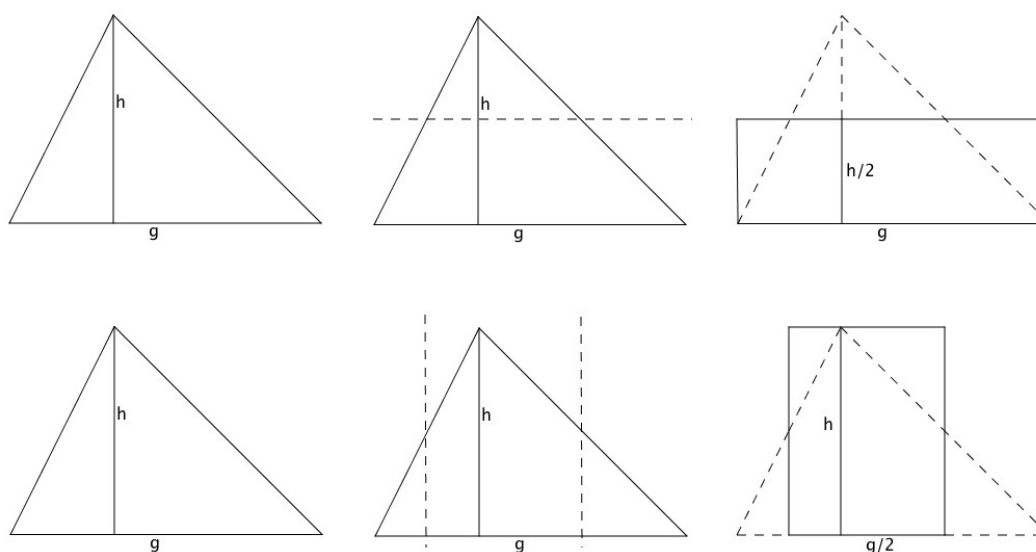


Abbildung 1: Zerlegungsmöglichkeiten eines Dreiecks

Lernenden kann man mit Hilfe eines CAS gut verdeutlichen, dass bei dieser enormen Anzahl an Dreiecken die Programmierung von Formeln einen enormen Einfluss auf die benötigte Rechenzeit hat, um etwa Spiele flüssig aussehen zu lassen. Dies kann vereinfacht mit Hilfe der gewonnenen Formeln und zwei zu programmierenden Schleifen verdeutlicht werden, um etwa die Oberfläche von einer Million Dreiecken mit Hilfe verschiedener Formeln zu berechnen und die dafür benötigte Rechenzeit zu vergleichen. Abb. 2 zeigt verschiedene mit dem CAS wxMaxima programmierte Schleifen und in eckigen Klammern die hierfür jeweils benötigte Rechenzeit in Sekunden.

```
for g:1 thru 1000 do
  for h:1 thru 1000 do
    g*h/2$
  time(%);
[ 4.435 ]

for g:1 thru 1000 do
  for h:1 thru 1000 do
    g/2*h$
  time(%);
[ 4.604 ]

for g:1 thru 1000 do
  for h:1 thru 1000 do
    0.5*g*h$
  time(%);
[ 3.406 ]

for g:1 thru 1000 do
  for h:1 thru 1000 do
    g*0.5*h$
  time(%);
[ 3.382 ]

for g:1 thru 1000 do
  for h:1 thru 1000 do
    g*h*0.5$
  time(%);
[ 3.334 ]

for g:1 thru 1000 do
  for h:1 thru 1000 do
    (g*h)*0.5$
  time(%);
[ 3.695 ]
```

Abbildung 2: Laufzeitmessung von Schleifen mit wxMaxima

Sehr auffällig - aber aufgrund der aufwändigeren Gleitpunktarithmetik weniger überraschend - ist die deutlich erhöhte Rechenzeit unter Nutzung von Division anstelle von Multiplikation. Auch Experimente zum (überflüssigen aber von Lernenden gern genutzten) Einsatz von Klammern in dieser Formel zeigen, dass dies die Rechenzeit negativ beeinflusst. Experimente zur Position von Konstanten in einer Formel liefern in diesem

Beispiel wiederum keine konsistent besseren Laufzeit-Ergebnisse. Auf diese Art kann der Einsatz eines CAS auch als Motivation für weiterführende Fragen zur Vernetzung von (fachübergreifenden) Themengebieten dienen: Um wie viel Prozent kann die Rechenzeit verkürzt werden, wenn man das schlechteste mit dem besten Ergebnis vergleicht? Wie viele Dreiecke wurden mit den zwei Schleifen berechnet? Verdoppelt sich die Rechenzeit, wenn man die Anzahl der Dreiecke verdoppelt? Wie kann man die Anzahl der Dreiecke mit Hilfe der Schleifen verdoppeln? Welche Formeln könnte man noch ausprobieren und wie kann man geometrisch die Gültigkeit der Formeln veranschaulichen?

CAS in der analytischen Geometrie

Ob der vorgestellte Einsatz der Laufzeitmessung im Kontext zur Berechnung von Flächeninhalten von Dreiecken lohnenswert erscheint, kann kritisch diskutiert werden, soll die Idee lediglich erläutern und wurde vom Autor an dieser Stelle auch nur in Klassen eingesetzt und mit Schülern besprochen, die interessiert waren. Besonders gewinnbringend erscheint die Idee spätestens in der Oberstufe im Themengebiet Geometrie im Kontext von Computerspielen etwa zur Berechnung des Schnittpunktes von Geraden mit Ebenen, um räumliche Punkte so auf einem zweidimensionalen Bildschirm zu projizieren, dass ein dreidimensionaler Eindruck entsteht. Abb. 4 am Ende des Artikels zeigt die Idee der pseudo-dreidimensionalen Darstellung am Beispiel eines Tetraeders mit Hilfe der Zentralprojektion.

Im Unterricht können zunächst anhand konkreter räumlicher Koordinaten eines Fluchtpunktes

$$F(f_1, f_2, f_3)$$

und eines Eckpunktes

$$E(e_1, e_2, e_3)$$

des Tetraeders die Koordinaten der Projektion P_E von E etwa in die y - z -Ebene berechnet und visualisiert werden. Durch den zunehmenden Einsatz von Variablen anstelle konkreter Koordinatenwerte und einer Verallgemeinerung der Projektionsebene zu

$$e : ax + by + cz = d$$

nähert sich die Berechnung der Koordinaten P_E der Idee einer sogenannten Spiele-Engine an, also einem Programm, das für die visuelle Darstellung eines Computerspielablaufs genutzt wird.

Abb. 3 zeigt, wie man syntaktisch mit Hilfe von wxMaxima die Koordinaten von P_E berechnen lassen kann. Die Rechenausdrücke der Koordinaten bieten vielfältige gewinnbringende Diskussionsanlässe für Vermutungen, wie die Rechenzeit verringert werden kann. Gute Beiträge oder Ideen wären etwa, dass gemeinsame Rechenausdrücke getrennt und nur einmalig berechnet werden (wie etwa den Faktor s in allen drei Koordinaten), ob es gewinnbringend ist im Nenner von s zu faktorisieren oder ob Subtraktionen in ihrer Anzahl minimiert werden können oder sollten.

```

gl:solve(a*(e1+s*(f1-e1))+b*(e2+s*(f2-e2))
+c*(e3+s*(f3-e3))=d,s);
[  $s = -\frac{c e_3 + b e_2 + a e_1 - d}{c f_3 + b f_2 + a f_1 - c e_3 - b e_2 - a e_1}$  ]

lsg:map(rhs,gl);
[  $-\frac{c e_3 + b e_2 + a e_1 - d}{c f_3 + b f_2 + a f_1 - c e_3 - b e_2 - a e_1}$  ]

x:f1+lsg*(f1-e1);
[  $f_1 - \frac{(c e_3 + b e_2 + a e_1 - d)(f_1 - e_1)}{c f_3 + b f_2 + a f_1 - c e_3 - b e_2 - a e_1}$  ]

y:f2+lsg*(f2-e2);
[  $f_2 - \frac{(c e_3 + b e_2 + a e_1 - d)(f_2 - e_1)}{c f_3 + b f_2 + a f_1 - c e_3 - b e_2 - a e_1}$  ]

z:f3+lsg*(f3-e3);
[  $f_3 - \frac{(c e_3 + b e_2 + a e_1 - d)(f_3 - e_3)}{c f_3 + b f_2 + a f_1 - c e_3 - b e_2 - a e_1}$  ]

```

Abbildung 3: Geraden-Ebenschnittberechnung mit wxMaxima

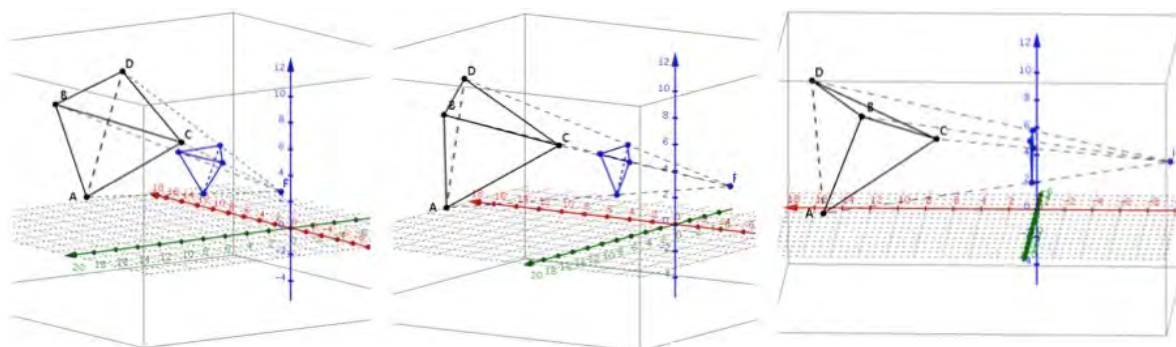


Abbildung 4: Eine Tetraederprojektion in die y-z-Ebene mit Geogebra

Zusammenfassung

Die Idee der Zeitmessung im Rahmen von Schleifenprogrammierung anhand der beiden Beispiele beschreibt eine einfach zu realisierende alternative Idee zur Motivierung von Termumformungen mit Hilfe eines CAS. Sie ist zwar nicht zwingend an ein CAS gebunden, ein CAS vereinfacht aber - wie im zweiten Beispiel gezeigt - die Berechnung, Darstellung und gemeinsame Analyse komplexer Termen auf schulischem Niveau. Zudem eröffnet die beschriebene Idee eine motivierende Möglichkeit Schüler(gruppen) mittels Termumformungen um eine möglichst geringe Laufzeit wetteifern zu lassen. Leider verfügen viele CAS-Handhelds nicht über die vorgestellte Laufzeitmessfunktion. Dies wäre zukünftig wünschenswert.

SFB/TRR 195 Symbolic Tools in Mathematics and their Application (Part 5/5)

Algebraic Geometry

Algebraic geometry is one of the five core areas in the SFB/TRR 195. SINGULAR, which is one of the four cornerstone computer algebra systems that will be further developed within the SFB/TRR 195, allows for polynomial computations, with special emphasis on commutative and non-commutative algebra, algebraic geometry, and singularity theory.

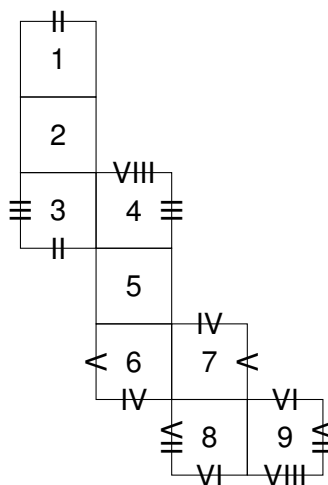
Systems of polynomial equations arise throughout mathematics, science and engineering. Their solution sets are the central objects of study in algebraic geometry. With the rapid increase of abstraction initiated by Grothendieck, algebraic geometry relies on more and more involved mathematical language and formalism. Making such concepts constructive and computationally accessible for the international user community is one of the goals of the SFB/TRR 195. In a project of Mohamed Barakat, together with Sebastian Posur and Kamal Saleh, the theory and implementation of constructive category theory was pushed forward significantly; in particular, many categorical constructions like additive closures, Freyd categories, Serre quotients, stable categories, homotopy categories, derived categories and their differentially graded enhancements together with various derived equivalences were realized for the first time in a computer algebra system.

A project of Frank-Olaf Schreyer has considerably improved the computation of syzygies (free resolutions) as well as the computation of the cohomology and the direct images of coherent sheaves. Among others, this led to a computational verification of the generic Green conjecture for canonical curves of small genus g also in finite characteristic $p \geq \lfloor \frac{g-1}{2} \rfloor$ (David Eisenbud and Schreyer).

To advance the constructive treatment of moduli spaces is another goal of the SFB/TRR 195. In a project of Wolfram Decker and Frank-Olaf Schreyer, Isabel Stenger significantly pushed forward our understanding of numerical Godeaux surfaces, which are minimal surfaces of general type with the smallest numerical invariants. The key algebraic feature in Stenger's constructions is a generalization of the celebrated structure result of Buchsbaum and Eisenbud. In the same project, supervised by Schreyer, Hanieh Keneshlou studied curves with several pencils in the geometry of the Brill-Noether divisor in the moduli space of curves, with focus on genus $g = 11$. The work of both Keneshlou and Stenger is considerably based on the use of computer algebra.

One approach to produce points in the moduli space of curves is via the dessins d'enfants introduced by Grothendieck in the 1980s. A variant of this idea is given by so-called origamis which play a key role in the

theory of translation surfaces. They are the topic of a project by Gabriela Weitze-Schmithüsen: origamis are surfaces obtained from gluing copies of the Euclidean unit square. They define particularly accessible examples of Teichmüller curves which are special algebraic curves in the moduli space. The computer algebra packages developed in the project provide a deeper understanding of origamis, their Teichmüller curves and their Veech groups.



An origami with 9 squares whose Veech group is a subgroup of $SL(2, \mathbb{Z})$ of index 38736. Edges with the same labels are glued.

A first important milestone in the SINGULAR project of Janko Böhm, Wolfram Decker and Mathias Schulze, in collaboration with Anne Frühbis-Krüger, was the development of a framework for massively parallel computations in algebraic geometry by combining SINGULAR with the workflow management system GPI-SPACE developed at the Fraunhofer ITWM Kaiserslautern. Together with newly designed algorithms, this led to a game-changing increase in computing power with successful applications in algebraic geometry, tropical geometry and high energy physics. Another important milestone of the project was the integration of SINGULAR with the language JULIA, which is crucial for the future design of SINGULAR and its integration into the visionary next generation system OSCAR, whose development is a central task of the SFB/TRR 195.

One of the main goals of our future projects in algebraic geometry is the advancement of the constructive treatment of higher dimensional varieties, their moduli and their classification.

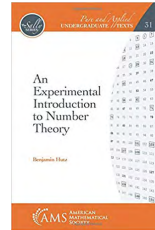
Wolfram Decker (Kaiserslautern)

Publikationen über Computeralgebra

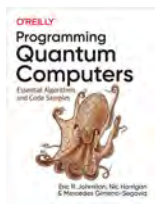
Neuerscheinungen:



Wilhelm Haager,
Computeralgebra mit Maxima,
2., aktualisierte Auflage,
Carl Hanser Verlag,
München 2019,
320 Seiten,
ISBN 978-3-446-44868-1



Benjamin Hutz,
*An Experimental Introduction to
Number Theory*,
AMS, Providence 2018,
313+xii Seiten,
ISBN 978-1-470-43097-9



Eric R. Johnston, Nic Harrigan und
Mercedes Gimeno-Segovia,
Programming Quantum Computers,
O'Reilly Media, Sebastopol 2019,
336 Seiten,
ISBN 978-1-492-03968-3



Edmund Weitz,
*Konkrete Mathematik (nicht nur) für
Informatiker*,
Springer Spektrum,
Heidelberg 2018,
955 Seiten,
ISBN 978-3-658-21564-4



Thomas Westermann,
*Mathematische Probleme lösen mit
Maple*,
6. Auflage, Springer Vieweg,
Wiesbaden 2020,
207 Seiten,
ISBN 978-3-662-60543-1

Die Rubrik Publikationen ist nicht allein auf eine Liste von Neuerscheinungen und Neuauflagen beschränkt. Sie lebt vor allem von fundierten Rezensionen von Fachgruppenmitgliedern für Fachgruppenmitglieder, die wir an dieser Stelle gerne abdrucken. Sollte eines der oben genannten Bücher, insbesondere eine der Neuerscheinungen, Ihr Interesse geweckt haben, und Sie möchten dieses für den Computeralgebra-Rundbrief besprechen, nehmen Sie bitte Kontakt zu Jürgen Klüners oder Martin Kreuzer (klueners@math.uni-paderborn.de, martin.kreuzer@uni-passau.de) auf.

Benjamin Hutz

An Experimental Introduction to Number Theory

Bei diesem Buch handelt es sich um eine Einführung in die algorithmische Zahlentheorie auf Bachelor-Niveau. Die Themen werden „experimentell“ eingeführt, das heißt, die Leser sollen durch Computerexperimente Muster erkennen, Vermutungen aufstellen und so herausfinden, für welche Behauptungen es sich lohnt, nach einem Beweis zu suchen.

Der Zielgruppe entsprechend beginnt der Autor mit sehr elementaren Themen: Primfaktorzerlegung ganzer Zahlen, ggT und euklidischer Algorithmus. Übungsaufgaben sind äußerst reichlich vorhanden und teilen sich in Berechnungsaufgaben, theoretische Aufgaben und „Erforschungsaufgaben“ auf. Besonders letztere bieten ein reichhaltiges Angebot und eignen sich sehr gut als Basis für Proseminare, Seminare oder Tutorien.

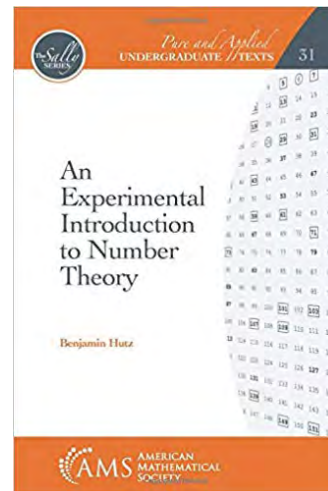
Nach weiteren, für die elementare Zahlentheorie grundlegenden Kapiteln über modulare Arithmetik (kleiner fermatscher Satz, chinesischer Restsatz) und quadratische Reste schiebt der Autor einen ersten angewandten Abschnitt über zahlentheoretische Kryptographie (Diffie-Hellman Schlüsselaustausch, RSA) ein.

Danach folgt noch ein elementares Kapitel über arithmetische Funktionen (Eulersche Phi-Funktion, Teilersummen, Möbius-Inversion), bevor der mathematische Anspruch deutlich zunimmt. Mit den quadratischen Zahlkörpern wird ein grundlegendes Thema der algebraischen Zahlentheorie diskutiert, bevor es mit diophantischen Approximationen, Kettenbrüchen und Konvergenzen etwas analytisch geprägter weiter geht.

In einem weiteren Kapitel werden modulare Methoden und Hensels Lemma auf berühmte klassische diophantische Gleichungen angewendet, unter anderem auf pythagoräische Tripel, Fermats „letzten Satz“, Pell-sche Gleichungen und das Waring-Problem bzw. den Vierquadrate-Satz.

Die letzten drei Kapitel erhöhen die Schagzahl dann noch einmal und bieten schöne Einblicke und Ausblicke auf moderne zahlentheoretische Bereiche, nämlich elliptische Kurven, dynamische Systeme in der Zahlentheorie und diophantische Gleichungen für Polynome. Auch hier bleibt sich der Autor treu und fördert ein Höchstmaß an entdeckendem Lernen.

Die experimentellen Untersuchungen und algorithmischen Aufgaben können mit den meisten Computeralgebrasystemen problemlos gelöst werden. Der Autor empfiehlt SageMath oder PARI/GP als kostenlose Möglichkeiten.



Das Buch ist sehr klar und übersichtlich präsentiert. Es deckt einen großen Bereich der elementaren Zahlentheorie sowie die Anfangsgründe der algebraischen und der analytischen Zahlentheorie ab. Es besticht besonders durch ein weit gespanntes Netz an Aufgaben, die zum Einsatz eines Computeralgebrasystems einladen und dessen Vorteile erlebbar machen.

Der einzige Punkt, der einer uneingeschränkten Anwendung dieses Buch entgegenstehen könnte ist, dass es in englischer Sprache verfasst ist, welche auf dem beabsichtigten Bachelor-Niveau nicht unbedingt gefordert werden kann. Angesichts der guten Englischkenntnisse der heutigen Bachelor-Studenten sollte dies im Regelfall nur eine sehr geringe Einschränkung sein. Das Buch ist jedem algorithmisch und zahlentheoretisch interessierten Studierenden oder Dozenten wärmstens zu empfehlen.

Martin Kreuzer (Passau)

Bican Xia und Lu Yang

Automated Inequality Proving and Discovering

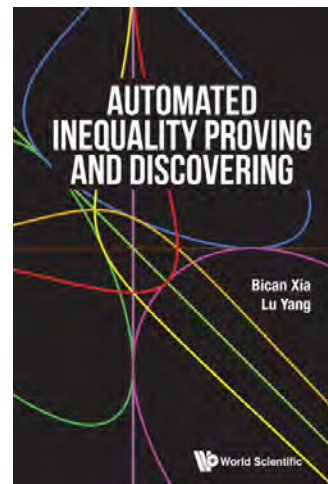
Das Hauptthema dieses Buchs ist, "automatische" Beweise für Ungleichungen $f(x_1, \dots, x_n) \geq 0$ mit einem Polynom $f \in \mathbb{R}[x_1, \dots, x_n]$ zu finden. Es handelt sich also im Wesentlichen um ein Thema, das der algorithmischen reellen algebraischen Geometrie zuzuordnen ist. So ist es kein Wunder, dass auch die grundlegenden Methoden, die zur Anwendung kommen und in den ersten drei Kapiteln eingeführt werden, aus diesem Gebiet stammen: Resultanten und Subresultanten, simpliziale Zerlegungen und trianguläre Zerlegungen. Für die zugrunde liegende Theorie aus der algorithmischen reellen algebraischen Geometrie verweisen die Autoren dabei auf die einschlägigen klassischen Werke.

Die nächsten drei Kapitel drehen sich dann um das Zählen reeller Nullstellen sowie die Isolierung und die Klassifizierung derselben. Der Bezug dieser Themen zum Beweisen von Ungleichungen bleibt dabei zunächst dünn und unklar: man könnte eher feststellen, dass Beiträge zum approximativen Lösen und zur reellen Quantorenelimination bereitgestellt werden. Der letzte Teil des sechsten Kapitels enthält aber dann doch noch sehr schöne Anwendungen, zum Beispiel auf automatische Beweise geometrischer Ungleichungen, auf die algebraische Analyse biologischer Systeme und auf Erreichbarkeits-Berechnungen für cyber-physikalische Systeme.

Die mathematisch anspruchsvollsten und fortschrittensten Methoden werden in Kapitel 7 eingeführt. Sie basieren auf der notorisch schwierigen zylindrischen algebraischen Zerlegung semialgebraischer reeller Varietäten. Dabei werden sukzessive Resultanten berechnet, um die positive Definitheit von Polynomen per Induktion über die Zahl der Unbestimmten zu entscheiden. Auch bei diesen Ansätzen ist die Anwendung auf praktisch interessante Ungleichungen eher indirekt.

Für die Praxis deutlich relevantere Verfahren werden jedoch in den letzten vier Kapiteln präsentiert. Bei der Dimensionsreduktion in Kapitel 8 werden Algorithmen diskutiert, die die Zahl der Unbestimmten auf ein Minimum verkleinern. Das resultierende Maple-Programm `BOTTEMA` hat z.B. über 1000 algebraische und geometrische Ungleichungen automatisch bewiesen und dabei über 100 offene Probleme gelöst.

Viele dieser Anwendungen werden ausführlich dargestellt. Weitere, auch klassisch bereits bekannte Verfahren sind die Summen-von-Quadraten Darstellungen in Kapitel 9 sowie die Symmetriebrechung durch sukzessive Differenzen-Substitutionen in Kapitel 10. Diese Methoden werden genau erläutert und durch viele Beispiele illustriert. Im letzten Kapitel behandeln die Autoren dann einige Ansätze, die über das Tarski-Modell hinausgehen, z.B. die Anwendung symmetrischer Polynome.



Das Buch stellt ein wichtiges und wertvolles Kompendium an Algorithmen bereit, um Polynomungleichungen im reellen Fall algorithmisch zu beweisen. Es verlangt aber besonders im ersten Teil ein gutes Grundwissen des Lesers in reeller Algebra und reeller algebraischer Geometrie. Wenn man sich durch diesen Teil hindurchgekämpft hat (oder ihn einfach überspringt), sind die Anwendungen in den hinteren Kapiteln aber umso beeindruckender und anregender. Für jeden Wissenschaftler, der sich auf algorithmische Art und Weise mit Ungleichungen beschäftigen möchte, liefert das Buch eine umfassende und sehr wertvolle Grundlage, die ein weites Gebiet an Anwendungen für die Computeralgebra erschließt.

Martin Kreuzer (Passau)

Henning Schatz: Automatic computation of continued fraction representations as solutions of explicit differential equations

Betreuer: Wolfram Koepf (Kassel)

Zweitgutachter: Werner Seiler (Kassel)

Februar 2020

Abstract:

Continued fractions, expressions of the form

$$b_0 + \frac{a_1}{b_1 + \frac{a_2}{b_2 + \frac{a_3}{b_3 + \ddots}}},$$

are not exactly a new field of mathematical study. They were already known and used by Euler, for example, to prove the irrationality of e . More recently, the study of continued fractions gained prominence, starting with Oscar Perron in 1913, continuing through Wall to Jones and Thron to Lorentzen and Waadeland. Despite that, continued fractions were sparsely found in collections and handbooks of special functions, until the release of the *Handbook of Continued Fractions for Special Functions* in 2008 [1], which collected all known continued fraction representations of most special constants and functions into a single reference work.

The main focus of this thesis is to present a variation of an algorithm first presented by Maulat and Salvy, with which it is possible to algorithmically guess as well as prove continued fraction expansions of analytical expressions with the help of ordinary differential equations.

To that end, Chapter 1 gives an overview of this thesis and its contents. Chapter 2 lays the groundwork, giving definitions and basic properties in relation to continued fractions, as well as tools with which to check continued fractions for convergence. It also contains a short excursion to the Riemann zeta function ζ and, more specifically, continued fraction expansions of $\zeta(3)$, both previously known as well as new ones, the latter derived from known continued fraction expansions of the tetragamma function. The final stretch contains basic definitions from the field of hypergeometric summation that are relevant to the changes to the algorithm of Maulat and Salvy, as well as a basic overview of the approaches of both Petkovšek and van Hoeij to the problem of finding all hypergeometric term solutions of a given holonomic recurrence equation.

Chapter 3 first presents the theoretical underpinnings of the guess and prove method. The main changes compared to the work of Maulat and Salvy is support for differential equations of order higher than one as well as applying van Hoeij's algorithm instead of a second guessing step. The van Hoeij algorithm is extended and used, since in the verification step of the guess and prove algorithm two-term right factors of a holonomic recurrence satisfied by some sequence H_n are of interest. A two-term right factor of order m corresponds to an m -fold hypergeometric term solution. As it turns out, it suffices to consider two-term right factors of a holonomic recurrence satisfied by any subsequence H_{ln+i} .

After the presentation of the main algorithm follow detailed demonstrations for $\tan x$ and $\exp x$, as well as less detailed examples mostly from the *Handbook of Continued Fractions for Special Functions*. This also includes two new

continued fraction representations of $\exp x$ derived from the generating function of the Euler polynomials. It follows a section concentrating on how to find differential equations satisfied by a given expression, which of course has applications for the presented algorithm, but can be of interest elsewhere. This section also contains some examples concerning implicit differential equations.

Finally, the fully automated algorithm presented in Chapter 3 was implemented using *Maple 18* in the package `guessandprove.mpl`, which is an integral part of this thesis, hence the appendix contains instructions and examples for the use of this package.

Reference

- [1] Annie Cuyt, Vigdis Brevik Petersen, Brigitte Verdonk, Haakon Waadeland and William B. Jones: *Handbook of Continued Fractions for Special Functions*, 2008, Springer.

Jan Horáček: Algebraic and Logic Solving Methods for Cryptanalysis

Betreuer: Martin Kreuzer (Universität Passau)

Zweitgutachter: Armin Biere (JKU Linz)

Februar 2020

Abstract:

Algebraic solving of polynomial systems and satisfiability of propositional logic formulas are not two completely separate research areas, as it may appear at first sight. In fact, many problems coming from cryptanalysis, such as algebraic fault attacks, can be rephrased as solving a set of Boolean polynomials or as deciding the satisfiability of a propositional logic formula. Thus one can analyze the security of cryptosystems by applying standard solving methods from computer algebra and SAT solving. This doctoral thesis is dedicated to studying solvers that are based on logic and algebra separately, as well as integrating them into one such that the combined solvers become more powerful tools for cryptanalysis.

The dissertation is divided into three parts. In the first part, some theory and basic techniques for algebraic and logic solving are recalled. The focus is mainly on DPLL-based SAT solving and techniques that are related to border bases and Gröbner bases. In particular, the Border Basis Algorithm and its specialized version for Boolean polynomials are discussed.

The second part deals with connecting solvers based on algebra and logic. The ultimate goal is to combine the strength of different solvers into one. Namely, the XOR reasoning from algebraic solvers is fused with the light, efficient design of SAT solvers. As a first step in this direction, various conversions from sets of clauses to sets of Boolean polynomials, and vice versa, are designed such that solutions and models are preserved via the conversions. In particular, based on a block-building mechanism, a new algorithm for the CNF to ANF conversion geared towards producing fewer and lower degree polynomials is constructed. The above conversions allow one to integrate both solvers via a communication interface.

To reach an even tighter integration, proof systems that combine resolution and polynomial calculus are considered. These are the two most used proof systems in logic and algebraic solving. Based on such a proof system, called SRES, new types of solving algorithms demonstrate the synergy between Gröbner-like and DPLL-like solving. At the end of the second part, some experiments based on a new benchmark illustrate that the new method has the potential to outperform classical SAT solvers.

In the third part of the thesis, practical attacks on vari-

ous cryptographic primitives are executed. For instance, SAT solvers are applied to algebraic fault attacks on the symmetric ciphers LED and derivatives of the block cipher AES. The main goal there is to derive so-called fault equations automatically from the hardware description of the cryptosystem and thus automatize the attack. To give some extra power to a SAT solver that inverts the hash functions SHA-1 and SHA-2, it is tweaked using a programmatic interface such that the propagation of the solver and thus the attack itself are improved.

Habilitationen in der Computeralgebra

Daniel Duviol Tcheutia: Algorithmic Methods for Mixed Recurrence Equations, Zeros of Classical Orthogonal Polynomials and Classical Orthogonal Polynomial Solutions of Three-Term Recurrence Equations

Betreuer: Wolfram Koepf (Kassel)

Weitere Gutachter: Mama Foupouagnigni (Yaoundé, Kamerun), Francisco Marcellán (Madrid, Spanien)

Oktober 2019

Abstract: Using an algorithmic approach, we derive classes of mixed recurrence equations satisfied by classical orthogonal polynomials. Starting from certain structure relations satisfied by classical orthogonal polynomials or their connection formulae, we show that our mixed recurrence equations are structurally valid. However, they couldn't be easily obtained with classical methods and for this reason, our algorithmic approach is important. The main algorithmic tool used here is an extended version of Zeilberger's algorithm. As application of the mixed recurrence equations,

1. we investigate interlacing properties of zeros of sequences of classical orthogonal polynomials;
2. we prove quasi-orthogonality of certain classes of polynomials and determine the location of the extreme zeros of the quasi-orthogonal polynomials with respect to the end points of the interval of orthogonality of the polynomial sequence, where possible;
3. we find bounds for the extreme zeros of classical orthogonal polynomials.

Every orthogonal polynomial system $\{p_n(x)\}_{n \geq 0}$ satisfies a three-term recurrence relation of the type

$$p_{n+1}(x) = (A_n x + B_n)p_n(x) - C_n p_{n-1}(x) \\ (n = 0, 1, 2, \dots, p_{-1} \equiv 0),$$

with $C_n A_n A_{n-1} > 0$. Moreover, Favard's theorem states that the converse is also true. A general method to derive the coefficients A_n, B_n, C_n in terms of the polynomial coefficients of the divided-difference equations satisfied by orthogonal polynomials on a quadratic or q -quadratic lattice is revisited. The Maple implementations `rec2ortho` of [3] or `retode` of [2] were developed to identify classical orthogonal polynomials knowing their three-term recurrence relations. The two implementations `rec2ortho` and `retode` do not handle classical orthogonal polynomials on a quadratic or q -quadratic lattice. We extend the Maple implementation `retode` of [2] to cover classical orthogonal polynomials on quadratic or q -quadratic lattices and to answer as application an open problem submitted by [1] during the 14th International Symposium on Orthogonal Polynomials, Special Functions and Applications.

References

- [1] Alhaidari, A. D.: Open problem in orthogonal polynomials, <https://arxiv.org/abs/1709.06081v1>, 2017.
- [2] Koepf, W. and Schmiersau, D.: Recurrence equations and their classical orthogonal polynomial solutions. *Appl. Math. Comput.* **128**, 2002, 303–327.
- [3] Koornwinder, T. H. and Swarttouw, R.: `rec2ortho`: an algorithm for identifying orthogonal polynomials given by their three-term recurrence relation as special functions, 1996–1998. staff.fnwi.uva.nl/t.h.koornwinder/art/software/rec2ortho/.

Lehrstuhljubiläum und Nikolauskonferenz 2019

Aachen, 05.12. – 07.12.2019

www.math.rwth-aachen.de/~LDfM50

www.math.rwth-aachen.de/Nikolaus2019

Die traditionelle Nikolauskonferenz am Lehrstuhl D für Mathematik der RWTH Aachen war in diesem Jahr um ein Kolloquium zur Feier des 50-jährigen Bestehens des Lehrstuhls erweitert worden. Die von den Lehrstuhlinhabern Joachim Neubüser (1969–1997) und Gerhard Hiss (seit 1997) gesetzten Forschungsschwerpunkte sind die algorithmische Gruppen- und Darstellungstheorie, und davon handelten die Vorträge, die von Bettina Eick und Frank Himstedt am 5.12. gehalten wurden.

Die Nikolauskonferenz selbst fand dann am Freitagnachmittag (6.12.) und am Samstag (7.12.) statt. In diesem Jahr hatte die Tagung über 45 externe Teilnehmerinnen und Teilnehmer, hinzu kamen etliche Teilnehmerinnen und Teilnehmer aus Aachen.

In 17 Vorträgen von je 20 Minuten ging es um algebraische Gruppen, endliche einfache Gruppen, Endomorphismenringe, ganzzahlige Gruppenringe, Permutationsgruppen, kristallographische Gruppen, Coxetergruppen, elliptische Kurven und Trivial-Source-Moduln. Wie in den Vorjahren war darauf geachtet worden, sowohl theoretische als auch experimentelle/computergestützte Aspekte vorzustellen.

Zwischen den Vorträgen hatten die Organisatoren (federführend: Frank Lübeck) genügend Zeit für Diskussionen vorgesehen, die auch während der gemeinsamen Abendessen fortgesetzt wurden, unter dem Motto “Research Cambridge Style”.

Thomas Breuer (Aachen)

Algebra and Algorithms

Djerba, Tunesien, 04.02. – 06.02.2020

sites.google.com/view/algebraalgorithms2020

Organisiert von Ihsen Yengui und Peter Schuster fand diese in mancher Hinsicht ungewöhnliche Konferenz auf der Ferieninsel Djerba statt. Ungewöhnlich war beispielsweise die Herangehensweise an algebraische Probleme vom Standpunkt der konstruktiven Algebra. Diese kann, wie Yengui auf der Konferenz-Homepage erklärt, als abstrakte Version von Computeralgebra angesehen werden, oder auch also eine Art „Preprocessing“, bei dem der rechnerische, explizite und konstruktive Gehalt von algebraischen Konzepten und Aussagen erst einmal aufgedeckt wird. Auf abstrakterem Niveau ist die konstruktive Algebra zwischen der Algebra und der mathematischen Logik verortet und hat Verbindungen zur Typentheorie. So stellte Peter Schuster in seinem Vortrag heraus, dass das Axiomenschema „ $A \vee \neg A$ “ (oder auch: *tertium non datur*) in der konstruktiven Algebra abzulehnen ist.

Ungewöhnlich war auch, dass der Austragungsort ein Ferienhotel war. Dies bot den großen Vorteil, dass alle Teilnehmer am selben Ort untergebracht und verpflegt wurden, an dem auch die Vorträge stattfanden, was der Kontaktbildung und dem fachlichen Austausch sehr zuträglich war. Insgesamt waren 25 Vorträge angesetzt, so dass fast alle Teilnehmer der Konferenz zu Wort kamen, ohne dass sich die Tagung in mehrere Sektionen aufspalten musste. Die Organisation der Konferenz war perfekt, so dass sich die Teilnehmer voll auf die inhaltlichen Aspekte konzentrieren konnten.

Gregor Kemper (München)



50 Jahre Lehrstuhl D

GAP Days Spring 2020

St Andrews, Schottland, 23.03. – 27.03.2020

www.gapdays.de/gapdays2020-spring

GAP Days are meetings where developers and users with GAP programming experience are invited to discuss, influence, and contribute to the future development of GAP. To streamline each meeting, the organisers usually suggest a few main topics to work on during the week.

The meetings are also suitable for advertising recent developments in GAP itself, and its packages, via short talks. However, the focus of the meetings is on development, with talks playing only a minor role. As enough GAP experts will be around for advice and technical support, the meetings offer particularly good opportunities for people to work on their own packages. At the end of the week, we would be happy to hear about any progress you made.

PCA 2020

St. Petersburg, Russland, 20.04. – 25.04.2020

pca-pdmi.ru/2020/

The annual conference Polynomial Computer Algebra is devoted to polynomial algorithms in Computer Algebra. This field has a lot of applications both in theoretical and applied mathematics as well as in Computer Science. The conference PCA'2020 is the 13th in the series. The first one PCA'2008 commemorated Eugene Pankratiev who was a brilliant specialist in the field of Computer Algebra and Differential Algebra.

FoCM 2020

Vancouver, Kanada, 15.06. – 24.06.2020

focm-society.org/2020/

Foundations of Computational Mathematics (FoCM) 2020 will be held at the Downtown Vancouver campus of Simon Fraser University from June 15 - June 24, 2020. The conference will follow a format similar to that of previous FoCM meetings: two plenary invited lectures each day, coupled with theme-centred workshops. Most workshop talks will also be by invitation.

Each workshop will extend over three days, with the conference consisting of a total of three periods. Although some participants choose to attend just one or two periods, participants are encouraged to attend the conference for its entire duration. In addition to the plenaries and workshops, there will be a poster session during each of the three periods.

SC² 2020

Paris, Frankreich, 05.07.2020

www.sc-square.org/CSA/workshop5.html

Symbolic Computation is concerned with the efficient algorithmic determination of exact solutions to complicated mathematical problems. Satisfiability Checking has recently started to tackle similar problems but with different algorithmic and technological solutions.

The two communities share many central interests, but researchers from these two communities rarely interact. Also, the lack of common or compatible interfaces for tools is an

obstacle to their fruitful combination. Bridges between the communities in the form of common platforms and roadmaps are necessary to initiate an exchange, and to support and direct their interaction. The aim of this workshop is to provide an opportunity to discuss, share knowledge and experience across both communities.

ICMS 2020

Braunschweig, 13.07. – 16.07.2020

www.icms-conference.org/2020

The “International Congress of Mathematical Software” (ICMS) is a community of researchers and practitioners centered around “mathematical software” as a scientific activity.

ACA 2020

Athen, Griechenland, 14.07. – 18.07.2020

math.unm.edu/aca.html

The 26th Conference on Applications of Computer Algebra (ACA) will be held in Athens, Greece. This event will take place from July 14 to July 18, 2020.

The ACA conference series is devoted to promoting all kinds of computer algebra applications, and encouraging the interaction of developers of computer algebra systems and packages with researchers and users (including scientists, engineers, educators, and mathematicians).

Topics include, but are not limited to, computer algebra in the sciences, engineering, communication, medicine, pure and applied mathematics, education, business and computer science.

ISSAC 2020

Kalamata, Griechenland, 20.07. – 23.07.2020

www.issac-conference.org/2020

The International Symposium on Symbolic and Algebraic Computation (ISSAC) is the premier conference for research in symbolic computation and computer algebra. ISSAC 2020 will be the 45th meeting in the series, which started in 1966 and has been held annually since 1981. The conference presents a range of invited speakers, tutorials, poster sessions, software demonstrations and vendor exhibits with a center-piece of contributed research papers.

ISSAC 2020 will be held on 20-23 July 2020, at Kalamata, Messinia, Greece.

SYNASC 2020

Timișoara, Rumänien, 03.09. – 05.09.2020

synasc.ro/2020

International Symposium on Symbolic and Numeric Algorithms for Scientific Computing is an international conference that aims to stimulate the interaction between the two scientific communities of symbolic and numeric computing and to exhibit interesting applications of the areas both in theory and in practice. The choice of the topic is motivated by the belief of the organizers that the dialogue between the two communities is very necessary for accelerating the progress in making the computer a truly intelligent aid for mathematicians and engineers.

CASC 2020

Linz, Österreich, 14.09. – 18.09.2020

www.casc-conference.org

The 22nd International Workshop on Computer Algebra in Scientific Computing, CASC 2020, will be held in Linz, Austria, September 14 - 18, 2020.

The tools of Scientific Computing play an important role in the natural sciences and engineering. Computer Algebra Systems and the underlying algorithms for Symbolic Computation play an increasingly important role within Scientific Computation. The CASC workshop series has been running for over two decades to explore the interaction of these topics, their implementation, and their application.

DMV-Jahrestagung 2020

Chemnitz, 14.09. – 17.09.2020

www.mathematik.de/dmv/jahrestagungen

The 2020 annual meeting of the Deutsche Mathematiker-Vereinigung (German Mathematical Society) will be held at the Technische Universität Chemnitz in September 2020. Two workshops with a long tradition at the TU Chemnitz, the Chemnitz Finite Element Symposium and the Chemnitz Symposium on Inverse Problems, will take place within DMV 2020.

GI-Jahrestagung 2020

Karlsruhe, 29.09 – 01.10.2020

informatik2020.de

Die 50. GI-Jahrestagung wird am 29.09. - 01.10.2020 im Kongresszentrum Karlsruhe unter dem Motto "Back to the Future" stattfinden.



Antrag auf Mitgliedschaft in der Fachgruppe Computeralgebra

Die Fachgruppe Computeralgebra sieht es als ihre Aufgabe an, Lehre, Forschung, Entwicklung, Anwendungen, Informationsaustausch und Zusammenarbeit auf dem Gebiet der Computeralgebra in Deutschland zu fördern.

Eine Mitgliedschaft in der Fachgruppe Computeralgebra gibt es bereits ab 7,50 € pro Jahr (für Mitglieder von DMV, GI oder GAMM; ansonsten 9 €).

Vorteile einer Mitgliedschaft:

- Sie fördern durch Ihren Beitrag die Workshops, Seminare, Tagungen und andere Aktivitäten auf dem Gebiet der Computeralgebra, die die Fachgruppe organisiert und unterstützt.
- Sie erhalten zweimal im Jahr den Computeralgebra-Rundbrief mit vielen interessanten Informationen rund um die Computeralgebra frei Haus.
- Sie verleihen unserer Stimme an Gewicht, die wir aktiv in Diskussionen um die Stellung der Computeralgebra in der Ausbildung in Schule und Hochschule einbringen.

Wir würden uns sehr über Ihre Unterstützung freuen. Die Mitgliedschaft in der Fachgruppe steht allen offen. Weiter Informationen zur Mitgliedschaft und einen Aufnahmeantrag finden Sie auf unserer Webseite unter folgender Adresse, oder scannen Sie einfach den QR-Code.

<https://fachgruppe-computeralgebra.de/aufnahmeantrag>



Fachgruppenleitung Computeralgebra 2020–2023

**Sprecherin:**

Prof. Dr. Anne Fruehbis-Krueger
Carl-von Ossietzky Universität Oldenburg
Institut für Mathematik
Carl-von-Ossietzky-Straße 11, 26129 Oldenburg
0441 798-3233
anne.fruehbis-krueger@uni-oldenburg.de
<https://uol.de/anne-fruehbis-krueger>

**Stellvertretender Sprecher:**

Prof. Dr. Gregor Kemper
Zentrum Mathematik – M11
Technische Universität München
Boltzmannstr. 3, 85748 Garching
089 289-17454, -17457 (Fax)
kemper@ma.tum.de
<http://www.groups.ma.tum.de/algebra/kemper>

**Vertreterin der GI:**

Prof. Dr. Erika Abraham
Fachgruppe Informatik
RWTH Aachen University
Ahornstr. 55, 52056 Aachen
0241 80-21242, -22243 (Fax)
abraham@cs.rwth-aachen.de
<https://ths.rwth-aachen.de/people/erika-abraham/>

**Fachreferentin Industrie:**

Xenia Bogomolec
Coding Services Hannover
Engelbosteler Damm 15, 30167 Hannover
0173 3031816
indigomind@protonmail.ch
<https://quant-x-sec.com>

**Fachreferent CA an der Hochschule:**

Prof. Dr. Michael Cuntz
Leibniz Universität Hannover
Institut für Algebra, Zahlentheorie und Diskrete Math.
Welfengarten 1, 30167 Hannover
0511 762-4252
cuntz@math.uni-hannover.de
<http://www.iazd.uni-hannover.de/~cuntz>

**Fachreferent CA-Systeme und -Bibliotheken:**

Prof. Dr. Claus Fieker
Fachbereich Mathematik
Technische Universität Kaiserslautern
Gottlieb-Daimler-Straße, 67663 Kaiserslautern
0631 205-2392, -4427 (Fax)
fieker@mathematik.uni-kl.de
<http://www.mathematik.uni-kl.de/~fieker>

**Fachexperte Physik:**

Dr. Thomas Hahn
Max-Planck-Institut für Physik
Föhringer Ring 6, 80805 München
089 32354-300, -304 (Fax)
hahn@feynarts.de
<http://wwwth.mpp.mpg.de/members/hahn>

**Vertreter der DMV:**

Prof. Dr. Florian Heß
Carl-von Ossietzky Universität Oldenburg
Institut für Mathematik, 26111 Oldenburg
0441 798-2906, -3004 (Fax)
florian.hess@uni-oldenburg.de
<https://uol.de/florian-hess>

**Fachreferent CA-Systeme und -Bibliotheken:**

Prof. Dr. Max Horn
Department Mathematik
Universität Siegen
Walter-Flex-Straße 3, 57072 Siegen
0271 740-2868
max.horn@uni-siegen.de
<https://www.quendi.de/de/mathe>

**Fachreferent Themen, Anwendungen und Publikationen:**

Prof. Dr. Jürgen Klüners
Mathematisches Institut der Universität Paderborn
Warburger Str. 100, 33098 Paderborn
05251 60-2646, -3516 (Fax)
klueners@math.uni-paderborn.de
<https://math.uni-paderborn.de/ag/klueners/>

**Fachreferent Themen, Anwendungen und Publikationen:**

Prof. Dr. Martin Kreuzer
Fakultät für Informatik und Mathematik
Universität Passau
Innstr. 33, 94030 Passau
0851 509-3120, -3122 (Fax)
martin.kreuzer@uni-passau.de
<http://www.fim.uni-passau.de/~kreuzer>

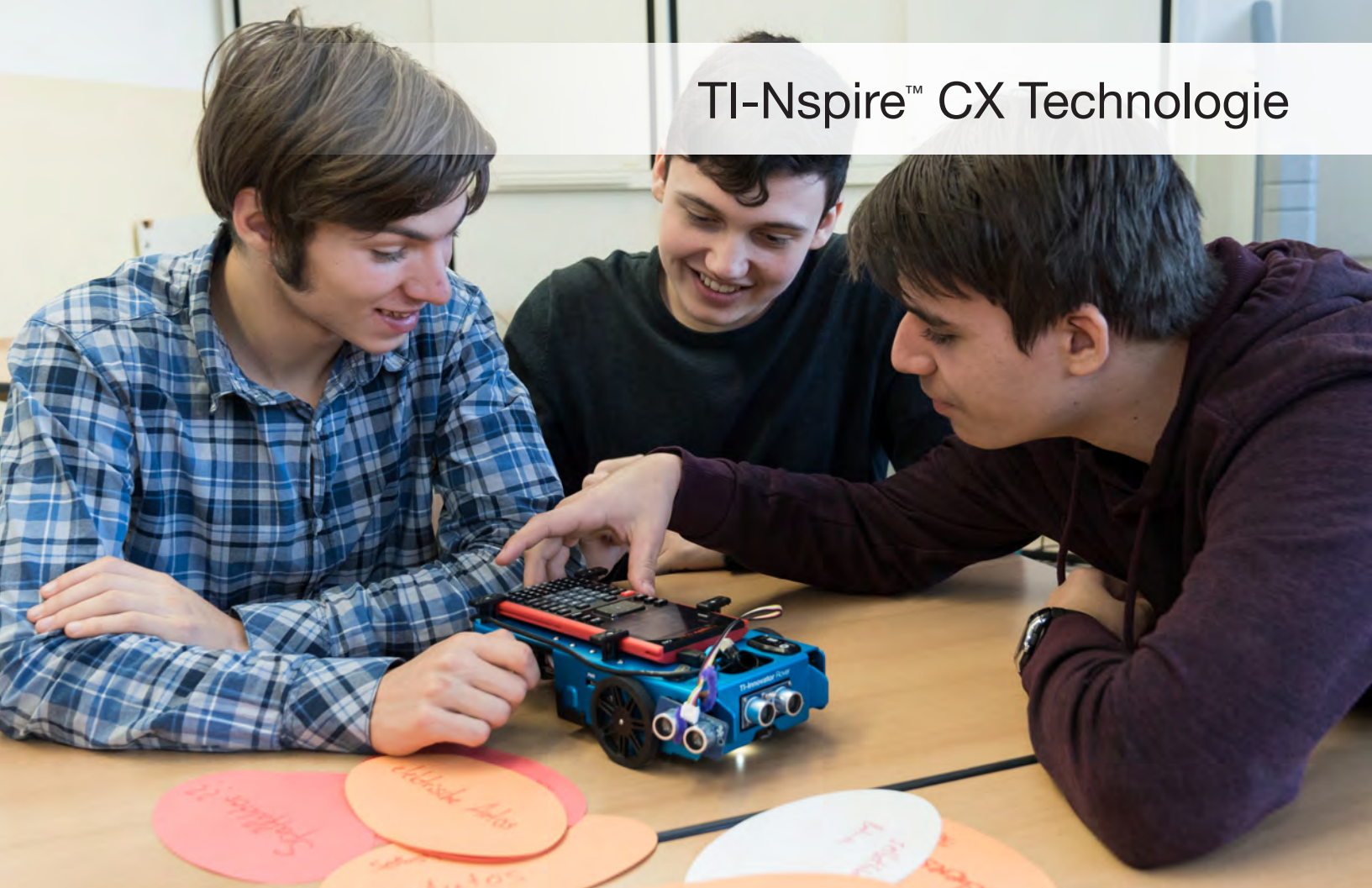
**Fachreferent Redaktion Rundbrief:**

Dr. Fabian Reimers
Zentrum Mathematik – M11
Technische Universität München
Boltzmannstr. 3, 85748 Garching
089 289-17474
reimers@ma.tum.de
<http://www.groups.ma.tum.de/algebra/reimers>

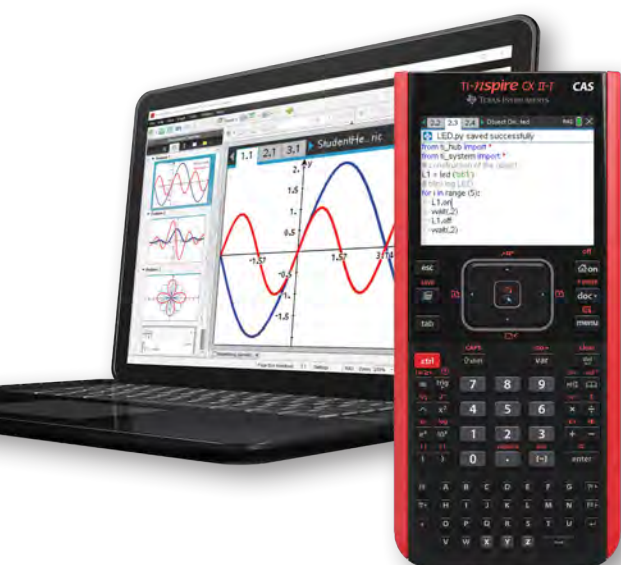
**Vertreterin der GAMM:**

Prof. Dr. Eva Zerz
Lehrstuhl D für Mathematik
RWTH Aachen
Pontdriesch 14/16, 52062 Aachen
0241 80-94544, -92108 (Fax)
eva.zerz@math.rwth-aachen.de
<http://www.math.rwth-aachen.de/~Eva.Zerz/>

TI-Nspire™ CX Technologie



BEGEISTERT FÜR MINT.



Die bekannte Programmiersprache Python ist jetzt in die TI-Nspire™ CX II-T CAS Graphikrechner und Software integriert.

Verfügbar im Herbst 2020.

education.ti.com/de/python