

März 2019

Computeralgebra Rundbrief

> Ausgabe 64

- ▶ Tagung der Fachgruppe 2019
- ▶ Parallelism in Computational Algebraic Geometry
- ▶ CAP: categories, algorithms, programming
- ▶ Viele Trassierungsaufgaben sind unsinnig!



Sie sind noch kein Maple User?

Lernen Sie hier, was Sie und vor allem Ihre Schüler und Studenten verpassen!

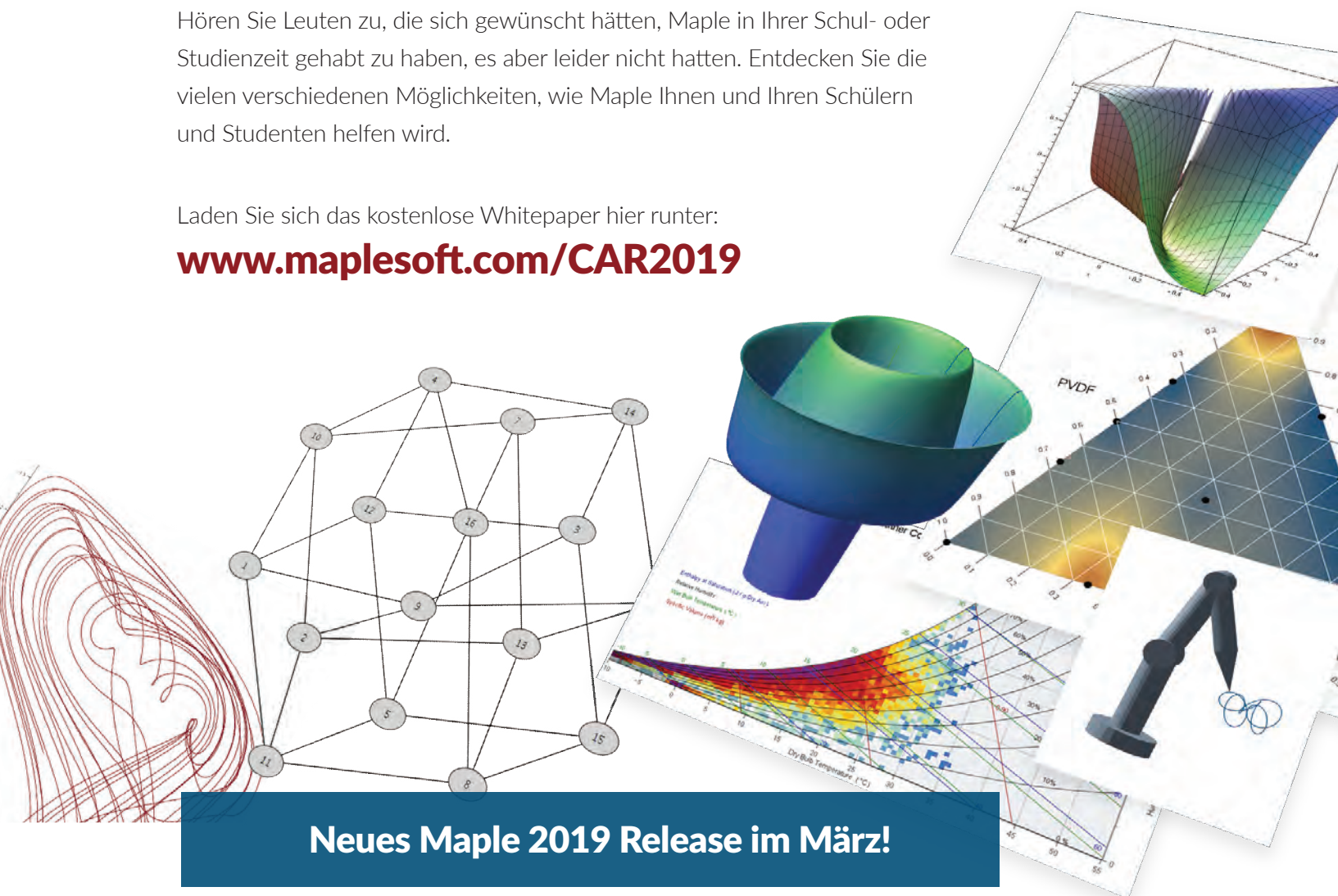
Leistungsstarke Mathematik Software, die gleichzeitig leicht zu bedienen ist

Warum das interessant ist?

Hören Sie Leuten zu, die sich gewünscht hätten, Maple in Ihrer Schul- oder Studienzeit gehabt zu haben, es aber leider nicht hatten. Entdecken Sie die vielen verschiedenen Möglichkeiten, wie Maple Ihnen und Ihren Schülern und Studenten helfen wird.

Laden Sie sich das kostenlose Whitepaper hier runter:

www.maplesoft.com/CAR2019



Neues Maple 2019 Release im März!



Inhaltsverzeichnis

Inhalt	3
Impressum	4
Mitteilungen der Sprecher	5
Tagungen der Fachgruppe	6
Themen und Anwendungen	8
<i>Parallelism in Computational Algebraic Geometry</i> (J. Böhm, A. Frühbis-Krüger, M. Rahn)	8
Neues über Systeme	14
<i>CAP: categories, algorithms, programming</i> (S. Gutsche, S. Posur)	14
Computeralgebra in der Schule	18
<i>Viele Trassierungsaufgaben sind unsinnig!</i> (J. Meyer)	18
Berichte über Arbeitsgruppen	20
<i>SFB/TRR 195 Symbolic Tools in Mathematics and their Application (Part 3/5)</i>	20
Berufungen	21
Publikationen über Computeralgebra	21
Besprechungen zu Büchern der Computeralgebra	22
<i>Hans-Gert Gräbe: EAGLE-Starthilfe Computeralgebra im Abitur</i> (M. Kreuzer)	22
Promotionen in der Computeralgebra	23
Berichte von Konferenzen	24
Hinweise auf Konferenzen	24
Fachgruppenleitung Computeralgebra 2017–2020	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
gs@gi-ev.de
<http://www.gi-ev.de>



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



Mitteilungen der Sprecher

Liebe Mitglieder der Fachgruppe Computeralgebra,

in der letzten Ausgabe des Rundbriefs haben wir von unserem Bemühen berichtet, die Position des Fachexperten Industrie nach dem Ausscheiden von Christoph Thiel neu zu besetzen. Dies ist nun gelungen: Wir konnten Frau Xenia Bogomolec hierfür gewinnen, sie wurde entsprechend auf der Frühjahrs-Sitzung der Fachgruppenleitung als Fachexpertin Industrie ernannt. Sie ist Expertin auf dem hochaktuellen Gebiet der Post-Quanten Kryptographie (siehe der von ihr mitverfasste Artikel im letzten Rundbrief). Bei der Mitwirkung in der Fachgruppe ist ihr Ziel, Verbindungen zwischen Hochschule und Industrie in Computeralgebra zu fördern.

Angesichts dieses Neuzugangs in der Fachgruppenleitung sollten wir aber nicht aus den Augen verlieren, dass die Amtszeit der jetzigen Fachgruppenleitung schon im Frühjahr 2020 endet. Mit anderen Worten: Es ist wieder an der Zeit, die Wahl der nächsten Fachgruppenleitung vorzubereiten. Deshalb bitten wir Sie zu überlegen, ob eine Kandidatur für Sie in Frage käme. Falls ja, teilen Sie uns dies bitte bis zum

31. Juli 2019

per Mail an kemper@ma.tum.de mit. Im Herbst-Rundbrief erfolgt dann die Vorstellung der Kandidatinnen und Kandidaten, wodurch die Wahl eingeläutet wird. Diesmal wird es erstmals die Möglichkeit der elektronischen Stimmabgabe geben, was ebenfalls in der Herbstausgabe vorgestellt werden wird.

Bei unserer diesjährigen Computeralgebra-Tagung der Fachgruppe in Kassel würden wir uns sehr freuen, viele von Ihnen und insbesondere viele Doktoranden/Doktorandinnen und junge PostDocs begrüßen zu dürfen. Wenn Sie diesen Rundbrief in Händen halten, ist es gerade noch rechtzeitig, um einen Vortrag anzumelden; ermuntern Sie insbesondere auch Nachwuchswissenschaftler aus Ihrem Umfeld dazu. Die Hauptvorträge und die bisher eingereichten Vortragsanmeldungen versprechen ein breit gefächertes, interessantes Programm. Näheres dazu und auch zu der Möglichkeit des Antrags auf Reisekostenbeihilfe für die Teilnahme finden Sie ab Seite 6.

Nach so vielen Ankündigungen in eigener Sache, darf die Computeralgebra nicht zu kurz kommen: Der erste Artikel befasst sich mit einem Thema, das sich in der Vergangenheit als notorisch schwierig erwiesen hat: massiv parallele Rechnungen in der Algebraischen Geometrie – daher kommt auch der Untertitel des Artikels “not a contradiction”. Auch der sehr abstrakte Blickwinkel der homologischen Algebra lässt den Leser nicht direkt an Implementierungen denken. Wie falsch dieser erste Eindruck ist, beweist der Artikel über das Software-Projekt CAP aus Siegen. Der Artikel mit Schulbezug nimmt dann beliebte Oberstufenaufgaben mit einem behaupteten Anwendungsbezug unter die Lupe: die Bestimmung von Straßenführungen.

Doch genug der Vorrede. Wir möchten Sie nicht länger aufhalten und wünschen Ihnen eine angenehme und anregende Lektüre dieses Hefts.

Gregor Kemper

Anne Frühbis-Krüger

Tagung der Fachgruppe Computeralgebra, Kassel, 16.5. – 18.5.2019

<http://www.fachgruppe-computeralgebra.de/kassel-2019>

In Fortsetzung der erfolgreichen Tagungen 2003, 2005, 2009, 2012, 2014, 2017 in Kassel und 2007 in Kaiserslautern führt die Fachgruppe vom 16. bis 18. Mai 2019 wieder eine derartige Tagung in Kassel durch. Das Ziel dieser Tagungsreihe ist es einerseits Nachwuchswissenschaftlern zu ermöglichen ihre Ergebnisse vorzustellen, andererseits aber auch einige Hauptvortragende zu gewinnen, die Übersichtsvorträge über wichtige Gebiete der Computeralgebra und über Computeralgebra-Software geben.

Hierzu fordern wir **Nachwuchswissenschaftler (Doktoranden, Post-Docs)** auf, sich bis zum **31.3.2019** mit einem **Vortrag anzumelden**. Details zur Anmeldung und zu weiteren Themen (Reisekostenbeihilfe, 500 € Nachwuchspreis) finden Sie auf der folgenden Seite.

Als **Hauptvortragende** konnten wir folgende Wissenschaftler gewinnen:

- **Matthias Junge (Oldenburg):** *Asymptotisch schnelle Arithmetik in der Picardgruppe algebraischer Kurven*
Wir präsentieren den asymptotisch schnellsten Algorithmus zum Rechnen in der Picardgruppe algebraischer Kurven, welche nicht notwendigerweise glatt sein müssen. Unser Algorithmus vereinigt die Laufzeiten der bisher schnellsten Algorithmen für glatte Kurven konstanter Gonalität (Heß) und glatter Kurven mit Gonalität in der Größenordnung des Geschlechts (Khuri-Makdisi). Darüber hinaus arbeitet unser Algorithmus mit weitaus allgemeineren Kurven. Im Falle von integralen, projektiven Kurven erzielen wir eine Laufzeit von $\mathcal{O}(n^{\omega-1}g)$, wobei g das arithmetische Geschlecht und n die Gonalität der Kurve bezeichnet.
- **Markus Kirschmer (Aachen):** *Quaternäre quadratische Formen*
Nach einem klassischen Ergebnis von Gauß entsprechen die quadratischen Formen in zwei Variablen über \mathbb{Z} bekanntlich den Idealen quadratischer Erweiterungen von \mathbb{Z} . Analog dazu korrespondieren auch die quadratischen Formen in vier Variablen über \mathbb{Z} bestimmten Idealen in Quaternionenordnungen. In dem Vortrag möchte ich diese Korrespondenz auf beliebige algebraische Zahlkörper ausdehnen.
Weiter werde ich zeigen, wie die Arithmetik in Quaternionenordnungen ausgenutzt werden kann, um die Isometrieklassen im Geschlecht einer quaternären quadratischen Form effizient zu bestimmen.
- **Hannah Markwig (Tübingen):** *Ebene tropische Kurven und ihre Berechnung*
Tropikalisierung bezeichnet einen Degenerationsprozess, unter dem algebraische Varietäten auf sogenannte tropische Varietäten übergehen, das sind bestimmte Polyederkomplexe. Dabei können wesentliche Eigenschaften erhalten bleiben. Dadurch erhalten wir die Möglichkeit, mit Hilfe von Methoden aus der konvexen Geometrie algebraische Varietäten zu studieren. Allerdings hängt die Tropikalisierung von der Einbettung ab. Eine treue Tropikalisierung bezeichnet eine, bei der „möglichst viele“ geometrische Eigenschaften erhalten bleiben. In diesem Vortrag werden Algorithmen zur Bestimmung treuer Tropikalisierungen ebener Kurven vorgestellt.
- **Bernd Sturmfels (Leipzig):** *Sixty-four Curves of Degree Six*
This lecture is an invitation to real algebraic geometry, along with computational aspects, ranging from bitangents and K3 surfaces to eigenvectors and ranks of tensors. We present an experimental study – with many pictures – of smooth curves of degree six in the real plane. This is joint work with Nidhi Kainhsa, Mario Kummer, Mahsa Sayyari and Daniel Plaumann. The number 64 refers to the Rokhlin-Nikulin classification of sextic curves.
- **Rebecca Waldecker (Halle):** *Kanonische Bilder*
Gegeben sind eine Menge M und eine Untergruppe H der symmetrischen Gruppe auf M . Wie können wir für zwei Elemente a, b aus M entscheiden, ob es eine Permutation in H gibt, die a auf b abbildet? Wie können wir für zwei gleich große Teilmengen A und B von M entscheiden, ob es eine Permutation in H gibt, die A auf B abbildet? Wie können wir entscheiden, ob zwei gegebene Elemente aus H in H konjugiert sind? Die Antwort ist ganz einfach: Wir probieren alle Elemente von H nacheinander durch und werden dann fündig oder eben nicht. Leider ist das nicht besonders effizient! Kanonische Bilder ermöglichen eine elegante Lösung für dieses Problem und haben daher zahlreiche Anwendungen.

Webseite: <http://www.fachgruppe-computeralgebra.de/kassel-2019>

Termin und Ort: Die Tagung findet in der Zeit vom 16. – 18. Mai 2019 an der Universität Kassel statt. Sie wird am 16. Mai 2019 um 13:00 Uhr eröffnet (Anreisetag) und endet am 18. Mai 2019 gegen 12:30 Uhr (Abreisetag).

Anmeldung: Die Anmeldung eines Vortrags ist bis 31. März 2019 möglich. Die Anmeldung ohne Vortrag ist bis 30. April 2019 möglich. Bitte benutzen Sie dazu das auf der Webseite der Tagung bereitgestellte Formular.

Konferenzgebühren: Jedes Nichtmitglied der Fachgruppe entrichtet vor Ort einen Unkostenbeitrag in Höhe von 20 € für die Kaffeepausen, alternativ kann man vor Ort zum Jahresbeitrag von 9 € Mitglied der Fachgruppe werden.

Hotel-Kontingente: Bis Ende März sind spezielle Hotel-Kontingente verfügbar. Details hierzu entnehmen Sie bitte dem Anmeldeformular.

Reisekostenbeihilfe: Die Fachgruppe Computeralgebra kann in begrenztem Umfang Mittel als Reisekostenbeihilfe zur Verfügung stellen. Bewerbungen auf Reisekostenbeihilfe mit einem erklärenden Anschreiben, einer Referenzperson sowie einer Aufstellung der benötigten Mittel bitten wir einzureichen.

Nachwuchspreis: Die Fachgruppe Computeralgebra vergibt für den besten Vortrag eines Nachwuchswissenschaftlers wieder einen mit 500 € dotierten Nachwuchspreis. Verbunden mit dem Geldpreis ist die Einladung auf der nächsten Tagung der Fachgruppe einen Hauptvortrag zu halten.



Tagung der Fachgruppe 2017 in Kassel

Massively parallel computations in algebraic geometry – not a contradiction^a

Janko Böhm (TU Kaiserslautern)

Anne Frühbis-Krüger (Leibniz Universität Hannover)

Mirko Rahn (Fraunhofer ITWM Kaiserslautern)

boehm@mathematik.uni-kl.de

anne@math.uni-hannover.de

mirko.rahn@itwm.fraunhofer.de



^a This work has been partially supported through SPP 1489 and through Project II.5 of SFB-TRR 195 “Symbolic Tools in Mathematics and their Application” of the German Research Foundation (DFG) and was developed with active and financial support from the “R&D Lab : BIG Data Analysis and High Performance Computing” in the Fraunhofer “High Performance Center Simulation and Software Based Innovation”

The design and implementation of parallel algorithms is a fundamental task in computer algebra. The combination of the computer algebra system SINGULAR and the workflow management system GPI-Space provides an infrastructure for massively parallel computations in commutative algebra and algebraic geometry, which has already shown very promising first results. In this note, we give an overview on the current capabilities of this framework by looking into three sample applications: determining smoothness of algebraic varieties, computing GIT-fans in geometric invariant theory, and determining tropicalizations. These applications employ algorithmic methods originating from commutative algebra, sheaf structures on manifolds, local geometry, convex geometry, group theory, and combinatorics, illustrating the potential of the framework in further problems in computer algebra.

This work grew out of a cooperation of the project II.5 “SINGULAR: A new level of abstraction and performance” within SFB/TRR 195 and the “R&D Lab: BIG Data Analysis and High Performance Computing” within Fraunhofer ITWM. The joint project is led by Janko Böhm, Wolfram Decker (both TU Kaiserslautern), Anne Frühbis-Krüger (Leibniz Universität Hannover), Franz-Josef Pfreundt and Mirko Rahn (both Fraunhofer ITWM). In this note, we discuss the general infrastructure and sample applications which were studied in the PhD thesis of Lukas Ristau [23] (TU Kaiserslautern and Fraunhofer ITWM; supervised by Wolfram Decker and Anne Frühbis-Krüger), the Master’s thesis of Christian Reinbold [21] (Leibniz Univer-

sität Hannover and Fraunhofer ITWM; supervised by Janko Böhm and Anne Frühbis-Krüger) and the Bachelor’s thesis of Dominik Bendle [1] (TU Kaiserslautern; supervised by Janko Böhm). The general infrastructure and the smoothness test were implemented in the context of Ristau’s thesis, the fan traversal and GIT-fan computation in Reinbold’s thesis and the computation of tropical varieties in Bendle’s thesis.

Introduction

Parallel computations play an increasingly important role in the development of computer infrastructure. In particular, high-performance clusters have the potential for a game-changing increase in computing power. This raises the question of development and efficient modeling of parallel algorithms in any field relying on large scale calculations. In the context of numerical simulations, massively parallel computations on clusters are nowadays an indispensable tool used for example in weather forecasts or seismic data analysis. In symbolic computing and, in particular, computational algebraic geometry, parallelism also starts to take a center stage [6, 5, 4, 18]. Massively parallel computations have been applied to problems in combinatorics, e.g. in [19]. However, in algebraic geometry, due to the unpredictability of time and memory consumption of key algorithmic tools like Buchberger’s Algorithm for computing Gröbner bases, the use of parallel algorithms has been limited to rather specific contexts. With the goal of a widespread use of parallelism in computer algebra,

an effort to adopt the approach of separating the actual computation from the coordination layer has been started. From an approach of this kind, fields like numerical simulation have already benefited significantly in recent years. In our context, the computer algebra system SINGULAR provides the computational back-end, for the coordination layer the workflow management system GPI-space is used, which employs Petri nets to model the respective algorithms.

Parallel Computing

From the technical point of view, there is more than one paradigm to achieve parallelism. In general, computer scientists identify two different models: in a shared memory based approach several threads have access to the same data in memory, whereas distributed models run independent processes, possibly on different computers, and communicate their results to the other processes as needed. Not being subject to the obvious restriction of running on the same machine, the latter approach is the suitable one for massively parallel computations, that is for applications scaling up to several hundreds or thousands of parallel computations. On the other hand, it also holds many technical challenges to be met by the expertise of computer scientists, in particular, the need for a clever coordination of the computations to ensure scaling of the performance with the number of cores.

Framework Based on GPI-space and SINGULAR

For the coordination layer we use the workflow management system GPI-space, which is developed by Fraunhofer ITWM (Kaiserslautern) [20], and allows for automated parallel execution of algorithms. GPI-space provides a scalable runtime system suitable for huge dynamic environments like scientific computing clusters, but works equally well on sets of computers with heterogeneous hardware, or an individual server. It manages available resources and is tolerant to failure of a node during computation. GPI-Space introduces an application independent global memory layer, that is used to manage the data-flow across a parallel machine during the computation. This memory driven computing approach gives applications access to very large memories of different character (DRAM NVMe). A key feature is the Petri net based workflow engine, which allows for automatic parallelization and dependency tracking, and will be addressed in the subsequent section.

As back-end we use SINGULAR [15], which is a computer algebra system developed for polynomial computation in commutative algebra, algebraic geometry and singularity theory. Its main workhorse is the Groebner basis engine, around which most of its core algorithms are built. SINGULAR exists as a stand-alone software with user interface and as a library version `libSingular`, which is designed for use within other

systems like SAGE or OSCAR. For use in conjunction with GPI-space, we rely on `libSingular` in its existing form, which for this purpose did not undergo any specific changes.

Petri Nets and Parallelism

While sequential algorithms are usually described step by step, Petri nets intrinsically reflect the structure of the algorithms and the state of the computation. As a result they allow for exploiting the parallel structure automatically. A Petri net looks like a directed graph with two kinds of vertices, places (represented as circles in the figures) and transitions (shown as rectangular boxes). The latter contain the elementary functional units, the places can hold marked tokens. These should be understood to represent pieces of data (in the sense of a so-called *colored* Petri net). Transitions link places, consuming one token from each input place and putting one token onto each output place. So a transition can only fire, if all input places hold a token. Figure 1 shows a small example of a Petri net.

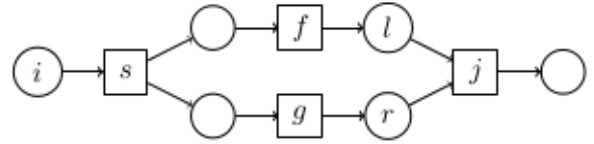


Figure 1: Example of a Petri net.

A token placed in place i causes transition s to fire which produces one token in each output place. These tokens are then treated independently by transitions f and g , which place their output tokens on l and r respectively. The transitions f and g can be executed in parallel, that is, the Petri net allows for task parallelism. The last transition j needs to consume one token from each of l and r . Note that the transition j can only fire, if there is at least one token in each of the places l and r . If several tokens are available on the places l and r the Petri net does not determine which of them will be consumed first by the transition j . By meeting this rule, we can allow several parallel instances of each transition to run in parallel without destroying the integrity of the computation (data parallelism). By the use of so-called *conditions*, it is possible to ensure that, depending on properties of the tokens, only certain transitions can fire.

Applications

Smoothness Test

One of the central tasks of computational algebraic geometry is to explicitly construct objects which exhibit prescribed properties, e.g., general members of moduli

spaces or counterexamples to conjectures. In this context, often the question of deciding smoothness of geometric objects arises, since singularities change intrinsic properties of these objects. Thinking in pictures, an algebraic set is smooth, if it looks in a sufficiently small neighborhood of each of its points like an affine space (that is, like a \mathbb{C}^k), see Figure 2.

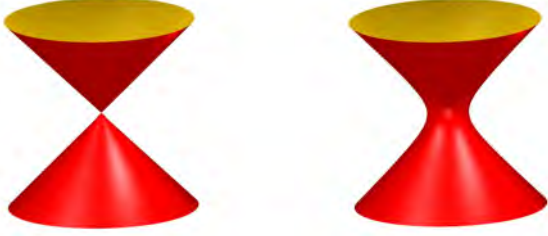


Figure 2: A singular and a smooth quadric.

It is classically known that smoothness can be decided by considering the Jacobian ideal generated by all minors of the Jacobian matrix of the correct (that is, codimension) size. Unfortunately, for many problems arising in current research, the Jacobian ideal or even a single minor can be too large to be handled with current computational means; for instance, a recently constructed numerical Godeaux surface (a minimal surface of *general type* with geometric genus and irregularity zero and $K^2 = 1$, constructed in the PhD-thesis of Isabel Stenger [24]) is minimally described by 38 generators in 13 variables, which leads to billions of minors.

Taking a completely different approach to deciding smoothness (or in this case more precisely regularity), one can follow ideas of Hironaka’s resolution of singularities and use his termination criterion for the resolution process. This approach has been described by the first and second author in [9]. In a nutshell, given an affine algebraic set X , this criterion locally uses a descending induction on the dimension of ambient spaces $W_i \supset X$ and checks at each level and point $p \in W_i$ the value of a certain invariant, the order of the defining ideal of the given affine algebraic set X in $\mathcal{O}_{W_i,p}$. If at some level i and point p the order exceeds one, the point p is singular. From an algorithmic point of view, we can then terminate the computation with a certificate of non-smoothness; if the order at every level and point has the value one until we reach the lowest level where the ideal of X equals the whole ring $\mathcal{O}_{W_k,p}$, we obtain a certificate of smoothness. Transferring this approach from the local analytic setting at individual points to the algebraic setting making use of Zariski open charts, the descending induction requires the computation of a suitable finite open covering at each level. This leads to a tree of charts with a potentially large number of leaves growing with the level i . Moreover, the deeper the level, the more difficult becomes the computation of the open covering of the current ambient space.

In practice, it is most useful to first proceed with the descending induction in order to pass from a com-

putationally very hard global problem to many computationally simpler local problems. However, before the combinatorial complexity becomes too large, one rather passes in each open set to a relative version of the Jacobian criterion when reaching a given codimension c . Since then both the number of generators of the ideal of X and the codimension are smaller, the computation of the ideal of minors (in each of the open sets) is a significantly easier task than in the beginning.

A simplified version of the Petri net used for modeling the algorithm is shown in Figure 3. Here tokens represent tuples consisting of X , a principal open subset U of affine space and a local ambient algebraic set $W \supset X \cap U$. The input token is placed on i . Depending on whether the codimension c has been reached for a given token (which is tested for by using conditions), either the relative Jacobian criterion (transition *Jac*) or the descent in dimension (transition *desc*) fires. These transitions add to each of their output tokens the knowledge whether a singularity has been detected. If so, the respective transition *sing* fires and places a token on the output place o , indicating that the variety is singular. Otherwise the token is deleted by *sm* or replaced on i , in the respective cases. If there are no tokens left in the Petri net, we know that the input algebraic set in the input principal open subset was smooth. For details on the implementation see [8].

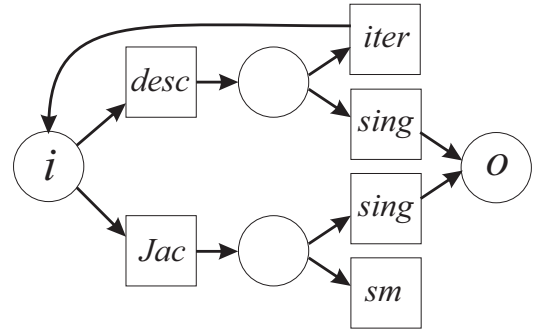


Figure 3: Basic structure of the Petri net for the smoothness test.

It is important to note that, due to the massively parallel execution of the Petri net, the computation will automatically determine, from all possible choices of charts, an open covering of X which leads to the (non-) smoothness certificate in the fastest possible way. This in fact leads to a super-linear speedup with the number of cores: While the computation on 16 cores takes about 53000 seconds, the computation on 128 cores finishes after 3100 seconds, i.e. using the 8-fold number of cores we achieve a speedup by a factor of 17 (with timings measured on a cluster of ITWM with 192 nodes, each with 16 Intel Xeon E5-2670 cores and 64 GB of RAM). The maximal speedup is limited by the total number of leaves of the tree of charts produced by the algorithm.

Another big advantage of the new approach is its lower memory consumption: For the above mentioned Godeaux surface, no individual transition consumed

more than 3,1 GB, whereas the Groebner basis calculation at the end of the classical sequential algorithm (global application of the Jacobian criterion) easily grows beyond 350 GB.

Computing GIT-fans

Geometric Invariant Theory (GIT) aims at associating a reasonable quotient $X//G$ to an algebraic variety X on which an algebraic group G acts. This setup occurs, e.g., when a parameterizing space for classes of geometric objects is to be constructed, where the action of the group G on X arises from isomorphisms between the objects. The homogeneous space X/G is not a good candidate for $X//G$, as it may not be an algebraic variety. For an affine variety X , one can rather define the quotient $X//G$ via the ring of invariant functions of X . However, this quotient may show very little structure: For example, the quotient $\mathbb{C}^2//\mathbb{C}^*$ with the action given by component-wise multiplication is just a point. If we allow for choosing an open subset of X , we obtain a much richer geometry, e.g., $(\mathbb{C}^2 \setminus \{(0,0)\})//\mathbb{C}^* = \mathbb{P}^1$. For given X , there are usually many choices of open subsets $U \subseteq X$ such that different choices lead to different birationally equivalent quotients $U//G$.

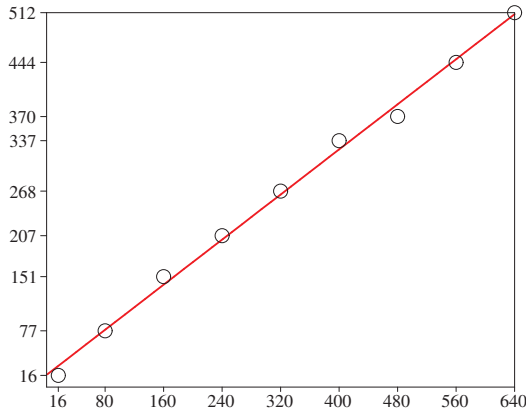


Figure 4: Scaling with the number of cores for computing the Mori chamber decomposition of $\overline{M}_{0,6}$.

To describe this behaviour, Dolgachev and Hu [17] introduced the *GIT-fan*, which is a polyhedral fan¹ describing this variation of GIT-quotients. In [12] a parallel algorithm for computing GIT-fans, which is also designed to make use of symmetries of the setup, has been described. The algorithm is based on symbolic methods from commutative algebra (Gröbner bases), convex geometry (double description) and group theory (orbit decomposition according to the finite symmetry group). Its most important substructure is a traversal of a complete fan which starts at some maximal dimensional cone of the GIT-fan, passes through codimension one faces to all its neighbors, as far as they are not known yet, and iterates. For a short account on the algorithm

¹A polyhedral fan is a finite set of strongly convex rational polyhedral cones such that their faces are again elements of the set and the intersection of any two cones is a face of both.

see [7]. In the Master's thesis of Christian Reinhold [21], this algorithm has been modeled in terms of a Petri net and has been implemented using our framework. We have applied this implementation to compute the Mori chamber decomposition of the cone of movable divisors of the Deligne-Mumford moduli space $\overline{M}_{0,6}$ of 6-pointed stable curves of genus 0. Figure 5 shows the computation time plotted against the number of cores in use (on the cluster described above). We observe an impressive linear scaling up to 640 cores, the maximum number we have tried so far, see Figure 4.

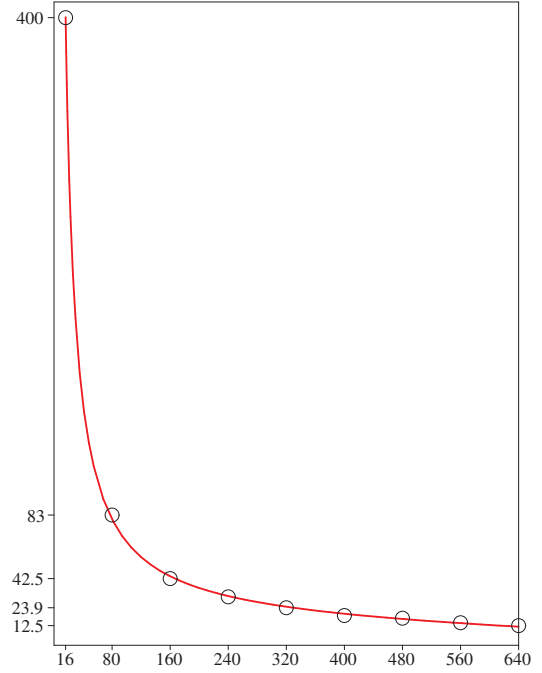


Figure 5: Computation times (in minutes) for the Mori chamber decomposition of $\overline{M}_{0,6}$.

Computing tropical varieties

Tropical geometry is a piecewise linear combinatorial version of algebraic geometry. From an element f of a polynomial ring $R = K[x_1, \dots, x_n]$ over a field K with valuation (usually the Puiseux series field in a variable t) one obtains a piecewise linear function by replacing the field operations with those of the tropical semi-ring

$$a \oplus b := \min(a, b), \quad f \odot g := a + b,$$

and coefficients by their valuation. Given an ideal $I \subset R$ and hence an algebraic variety $V(I)$, one can associate to I the tropical variety $T(I)$ defined as the common corner locus of the tropical polynomials corresponding to elements of I . By the Bieri-Groves theorem [2], the tropical variety corresponds to the set of valuation-tuples of K -points of I . See Figure 6 for a visualization of (fibers in) the family of plane elliptic curves defined by

$$I = \langle t \cdot (x^3 + y^3 + 1) + xy \rangle \in K[x, y]$$

and the associated tropical variety.

Computation of tropical varieties is usually a difficult task, both due to the combinatorial complexity of the resulting tropical variety and the computations of the individual faces, which are based on Buchberger’s algorithm for finding a Gröbner basis. The standard algorithm for this problem starts with one face of the tropical variety and passes to all neighbors by making use of the fact that tropical varieties of prime ideals are irreducible and connected in codimension one [3].

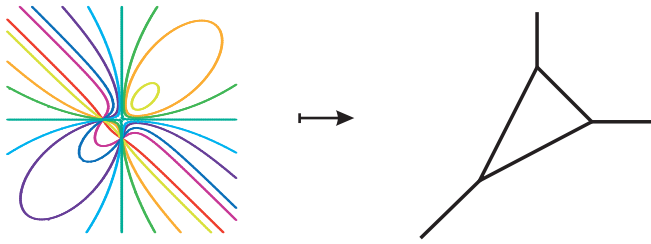


Figure 6: Tropicalization of a family of elliptic curves.

Our massively parallel algorithm for computing tropical varieties is based on the fan traversal through codimension one faces (i.e., on a graph expansion problem), see [1]. Neighboring faces are computed by determining the tropical link via reduction to the curve case and considering Puiseux expansions, see [22]. The traversal is derived from the fan traversal discussed in the subsection on GIT-fans. As in the setting there, the algorithm can make use of symmetries. However, the algorithm is more involved since, although tropical varieties can also be described as fans, they are not of maximal dimension. As a result, when passing through a codimension one face, there will typically exist more than one neighboring cone of maximal dimension.

Our implementation computes the previously unknown tropical Grassmannian $\mathcal{G}_{3,8}$ on 768 cores in less than 20 minutes with a parallel efficiency comparable to the GIT-fan algorithm. This result opens up, for example, the possibility of investigating the relation of the tropical Grassmannian to the Dressian using methods from matroid theory, and the study of the positive Grassmannian.

Conclusion and Outlook

Using GPI-Space in conjunction with SINGULAR and possibly other computer algebra systems with C-library interface provides an efficient infrastructure for massively parallel computations in computer algebra. By modeling algorithms in terms of Petri nets, parallel structures of the problem can be exploited in a transparent and straight-forward manner. Our infrastructure can make use of the potential of large clusters, but is also suitable for use on smaller multi-core servers and personal computers. Based on the example applications considered so far, we believe that a multitude of algorithmic problems in computer algebra can benefit significantly from our approach. Current research addresses,

for example, the computation of Hironaka resolutions of singularities [16], Zeta-functions, positive tropical varieties, integration-by-parts identities for Feynman integrals in high-energy physics via the algorithm developed in [13, 14], and generating functions for tropical numerical invariants via the algorithms introduced in [10, 11].

References

- [1] Bendle, D.: *Massively Parallel Computation of Tropical Varieties*. Bachelor’s Thesis, TU Kaiserslautern (2018).
- [2] Bieri, R.; Groves, J.R.J.: *The geometry of the set of characters induced by valuations*. Journal für die reine und angewandte Mathematik **347** (1984), 168 – 195.
- [3] Bogart, T.; Jensen, A.N.; Speyer, D.; Sturmfels, B.; Thomas, R.R.: *Computing tropical varieties*, J. Symbolic Comput. **42** (2007), 54 – 73.
- [4] Böhm, J.; Decker, W.; Fieker, C.; Pfister, G.: *The use of bad primes in rational reconstruction*, Math. Comp. **84**, 3013–3027 (2015).
- [5] Böhm, J.; Decker, W.; Laplagne, S.; Pfister, G.: *Local to global algorithms for the Gorenstein adjoint ideal of a curve* in Böckle et al. (ed.) Algorithmic and Experimental Methods in Algebra, Geometry, and Number Theory, Springer (2018), 51–96.
- [6] Böhm, J.; Decker, W.; Laplagne, S.; Pfister, G.; Steenpaß, A.; Steidel, S.: *Parallel Algorithms for Normalization*. J. Symbolic Comput. **51** (2013), 99–114.
- [7] Böhm, J.; Decker, W.; Keicher, S.; Ren, Y.: *Current Challenges in Developing Open Source Computer Algebra Systems*. LNCS **9582** (2016).
- [8] Böhm, J.; Decker, W.; Frühbis-Krüger, A.; Pfreundt, F.-J.; Rahn, M.; Ristau, L.: *Towards Massively Parallel Computations in Algebraic Geometry*. Preprint (2018). <https://arxiv.org/abs/1808.09727>
- [9] Böhm, J.; Frühbis-Krüger, A.: *A smoothness test for higher codimensions*. J. Symbolic Comput. **86** (2018), 153–165.
- [10] Böhm, J.; Goldner, Ch.; Markwig, H.: *Tropical mirror symmetry in dimension one*. Preprint (2018). <https://arxiv.org/abs/1809.10659>
- [11] Böhm, J.; Bringmann, K.; Buchholz, A.; Markwig, H.: *Tropical mirror symmetry for elliptic curves*, J. Reine Angew. Math. **732** (2017), 211–246.
- [12] Böhm, J.; Keicher, S.; Ren, Y.: *Computing GIT-fans with symmetry and the Mori chamber decomposition of $\overline{M}_{0,6}$* . Preprint (2016). <https://arxiv.org/abs/1603.09241>
- [13] Böhm, J.; Georgoudis, A.; Larsen, K.J.; Schöenemann, H.; Zhang, Y.: *Complete integration-by-parts reductions of the non-planar hexagon-box via*

- module intersections*. J. High Energ. Phys. **2018:24** (2018).
- [14] Böhm, J.; Georgoudis, A.; Larsen, K.J.; Schulze, M.; Zhang, Y.: *Complete sets of logarithmic vector fields for integration-by-parts identities of Feynman integrals*, Phys. Rev. D **98** (2018) 025023.
- [15] Decker, W.; Greuel, G.-M.; Pfister, G.; Schönemann, H.: *SINGULAR 4-1-0 — A computer algebra system for polynomial computations*. <http://www.singular.uni-kl.de>.
- [16] Frühbis-Krüger, A.; Ristau, L.; Schober, B.: *Embedded desingularization for arithmetic surfaces – toward a parallel implementation*. Preprint (2017). <https://arxiv.org/abs/1712.08131>
- [17] Dolgachev, I.V.; Hu, Y. and Yi Hu, *Variation of geometric invariant theory quotients. (With an appendix: “An example of a thick wall” by Nicolas Ressayre).*, Publ. Math., Inst. Hautes Étud. Sci. **87** (1998), 5–56.
- [18] Jensen, A.; Sommars, J.; Verschelde, J., *Computing Tropical Prevarieties in Parallel*, Preprint (2017), <https://arxiv.org/abs/1705.00720>
- [19] Jordan, C.; Joswig, M.; Kastner, L., *Parallel Enumeration of Triangulations*, Electronic Journal of Combinatorics **25**, (2018)
- [20] Pfreundt, F.-J. ; Rahn, M.; et al.: *GPI-space*, Fraunhofer ITWM Kaiserslautern, <http://www.gpi-space.de/>.
- [21] Reinbold, Ch.: *Computation of the GIT-fan using a massively parallel implementation*. Master’s Thesis, Leibniz Universität Hannover (2018).
- [22] Ren, Y; Hofmann, T.: *Computing Tropical Points and Tropical Links*. Discrete and Computational Geometry **60** (2018), 627–645.
- [23] Ristau, L.: *Using Petri nets to parallelize algebraic algorithms*, Dissertation, TU Kaiserslautern (2019)
- [24] Stenger, I.: *A Homological Approach to Numerical Godeaux Surfaces*, Dissertation, TU Kaiserslautern (2018)

Workshop-Förderung der Fachgruppe:

Sie veranstalten einen Workshop zu einem Thema aus dem Bereich der Computeralgebra und könnten mit einer kleinen finanziellen Unterstützung den Workshop deutlich interessanter oder effektiver gestalten? Die Fachgruppe Computeralgebra unterstützt Workshops mit bis zu 1000,- Euro.

Anträge können bis mit einer kurzen Beschreibung des Workshops (ca. 1 DIN A4 Seite; kurze Beschreibung des Gebiets, Thema des Workshops, Zielgruppe, Budget-Planung) und einer Darstellung, inwiefern diese Förderung einen deutlich erkennbaren Beitrag zum Gelingen des Workshops und zur Nachwuchsförderung liefert, an den Sprecher der Fachgruppe gerichtet werden: **kemper@ma.tum.de**, bitte „**Workshop-Förderung**“ im Betreff angeben.



CAP: categories, algorithms, programming

S. Gutsche (University of Siegen)

S. Posur (University of Siegen)

sebastian.gutsche@uni-siegen.de

sebastian.posur@uni-siegen.de



What is CAP?

CAP stands for categories, algorithms, programming and is an open source software project for constructive category theory written in GAP (groups, algorithms, programming) [4]. CAP's core package includes tools to facilitate the implementation of categories in GAP. The other CAP packages mainly include implementations of various category constructors, i.e., operations that output concrete instances of categories. The idea of category constructors lies at the heart of CAP: construct fairly complex computational contexts from easily graspable building blocks.

The development of CAP's core package started in December 2013 by the authors of this article followed by major contributions of Øystein Skartsæterhagen in 2015. CAP's core package along with three other packages¹ are distributed via the current GAP release², more packages can be found on GitHub³.

As a quick way to learn about CAP and test its features, you can try CAP in a Jupyter notebook interactively in Binder⁴.

Applications of CAP

CAP is used for redesigning the categorical foundations of the homalg project [7] and in the development of a new version of QPA, a package for computations with quivers and path algebras [11]. CAP has been used in the search for equivariant vector bundles on projective space [10], in the computation of associators in skeletal representation categories [9], in modeling coherent sheaves on toric varieties [6] and for performing cohomology computations within this context [2]. Furthermore, Dupont and Juteau implemented an algorithm in

CAP for the computation of periods associated to hyperplane arrangements decorated with two colors [3].

The purpose of CAP

Every computer algebra system (CAS) faces the question of how to organize the mathematical entities that it wishes to support. Desirable features of such an organization are:

1. Unambiguous specifications: providing input/output types for the operations.
2. High composability: offering sufficiently many operations that allow to quickly write programs for the computational exploration of new mathematical ideas.
3. Intuitive usage: being designed for mathematicians as human beings.

CAP's approach to achieve these features is the organization of all its data in the language of category theory. Every object or morphism created within a CAP session belongs to exactly one category, which can be thought of as being part of its type.

Choosing category theory as an organizational principle allows the definition of clearly specified operations for concepts of interest in computer algebra like subobjects or quotients, image factorizations, (co)kernels, tensor products, or Hom-functors. Moreover, a system that strictly adheres to purely categorical specifications automatically benefits from the impressive expressiveness of category theory, allowing to compose complicated systems from fairly easy building blocks. Last, category theory nowadays is a widely accepted link between different areas of mathematics, and thus a suitable choice of a common language.

¹These three packages are: `LinearAlgebraForCAP` (computing with finite dimensional vector spaces), `ModulePresentationsForCAP` (computing with finitely presented modules), `GeneralizedMorphismsForCAP` (performing diagram chases in homological algebra).

²Version 4.10.0, as of January 2019

³The CAP project GitHub repository: https://github.com/homalg-project/CAP_project

⁴Trying CAP in a Jupyter notebook: <https://mybinder.org/v2/gh/sebastianpos/cap-aachen2018/master>

How CAP works

CAP is ultimately designed for computational purposes. It can be used as a calculator that operates on objects and morphisms of some specific instance of a category. We will demonstrate the usage of CAP by means of a simple guiding example: $\text{vec}_{\mathbb{Q}}$, the category of finite dimensional \mathbb{Q} -vector spaces. Later, we discuss more advanced examples.

Computer-friendly models of categories

We consider a category \mathbf{C} to be **computable** if we have data structures for both the objects in $\text{Obj}_{\mathbf{C}}$ and the morphisms in $\text{Hom}_{\mathbf{C}}(A, B)$, along with algorithms for composing morphisms, deciding their equality, and constructing identities $\text{id}_A \in \text{Hom}_{\mathbf{C}}(A, A)$, where $A, B \in \text{Obj}_{\mathbf{C}}$.

Example We can construct an example of a computable category as follows: we let the objects simply be given by \mathbb{N}_0 . We define the set of homomorphisms from $m \in \mathbb{N}_0$ to $n \in \mathbb{N}_0$ as $\mathbb{Q}^{m \times n}$. Composition is induced by matrix multiplication (with swapped arguments), identities are given by identity matrices. We call this category $\text{mat}_{\mathbb{Q}}$, the category of matrices⁵ over \mathbb{Q} .

There exists an equivalence of categories

$$\text{mat}_{\mathbb{Q}} \simeq \text{vec}_{\mathbb{Q}}$$

that identifies $n \in \mathbb{N}_0$ with the row space $\mathbb{Q}^{1 \times n}$. Due to the simplicity of its data structures, the category $\text{mat}_{\mathbb{Q}}$ is undoubtedly very computer-friendly, and as long as we only care about categorical notions, working within $\text{mat}_{\mathbb{Q}}$ is as good as working within $\text{vec}_{\mathbb{Q}}$, just like replacing a group G with an isomorphic group H is harmless as long as we only care about isomorphism-invariant properties of G .

Here, we see a first benefit of categorical organization: the notion of an equivalence of categories makes us very flexible in our choice of data structures, and lets us, in the context of our guiding example, think about elements in \mathbb{N}_0 as if they were finite dimensional vector spaces.

Categorical language

Most of CAP's primitives are constructive interpretations of definitions found in standard textbooks on category theory. For example, the concept of kernels within an Ab-category⁶ is realized by three operations:

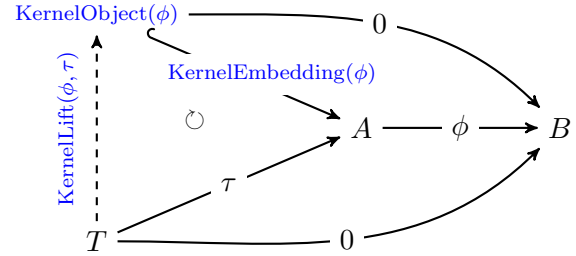
1. Given $\phi : A \rightarrow B$, compute an object $\text{KernelObject}(\phi)$.
2. Given $\phi : A \rightarrow B$, compute a morphism $\text{KernelEmbedding}(\phi) : \text{KernelObject}(\phi) \rightarrow A$ s.t. $\phi \circ \text{KernelEmbedding}(\phi) = 0$.

3. Given $\phi : A \rightarrow B$ and $\tau : T \rightarrow A$ such that

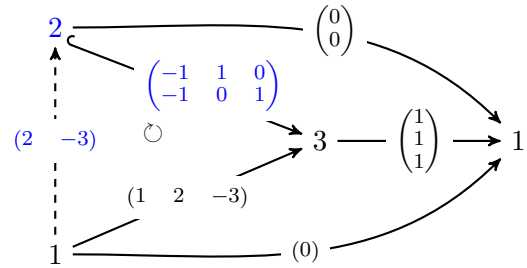
$$\phi \circ \tau = 0,$$

compute a uniquely determined morphism $\text{KernelLift}(\phi, \tau) : T \rightarrow \text{KernelObject}(\phi)$ s.t. $\text{KernelEmbedding}(\phi) \circ \text{KernelLift}(\phi, \tau) = \tau$.

We depict the three algorithms in a diagram:



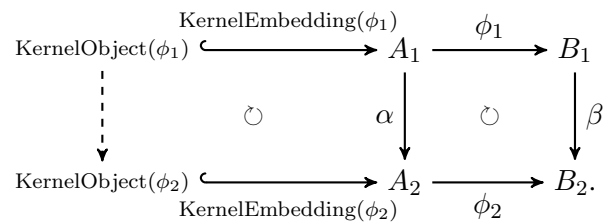
Here is an example of how the three algorithms could act on a specific input in $\text{mat}_{\mathbb{Q}}$ (remember, the objects are simply elements in \mathbb{N}_0):



We see that within $\text{mat}_{\mathbb{Q}}$ the operation KernelObject merely yields the dimension of the row syzygies, whereas KernelEmbedding outputs an actual basis. Finally, KernelLift expresses any matrix containing row syzygies as a linear combination w.r.t. the basis given by KernelEmbedding .

Other examples of categorical notions supported by CAP are limits like direct products, terminal objects, equalizers, and pullbacks, colimits like cokernels, coproducts, initial objects, coequalizers, and pushouts, universal constructions like images and coimages, and additional categorical structures like tensor products and internal homs.

One needs a little practice in order to get used to expressing familiar set-theoretic constructions in the language of category theory. We give an example. Consider the following diagram



⁵Note that matrices form the morphisms in this category, not the objects.

⁶An Ab-category is a category whose homomorphism sets are abelian groups s.t. the composition is bilinear. In particular, there exists a zero morphism (simply denoted by 0) between any two objects.

The task is to find the dashed arrow which renders the left square commutative assuming we are given the solid arrows. If this is a diagram of abelian groups, one can argue set-theoretically: for an element $a \in A_1$ in the kernel of ϕ_1 , we have $\phi_2(\alpha(a)) = \beta(\phi_1(a)) = 0$. Thus, $\alpha(a)$ lies in the kernel of ϕ_2 and we can construct the dashed arrow by restricting the source and range of α .

In the language of category theory, we can simply state that, due to the commutativity of the right square, the following closed formula is well-defined and yields the desired dashed arrow:

$$\text{KernelLift}(\phi_2, \alpha \circ \text{KernelEmbedding}(\phi_1)). \quad (1)$$

Note that this formula is easy to implement, we don't need any accessing of "underlying elements". Moreover, this formula is valid in any Ab-category with kernels.

Finitely presented modules

To give another example of such a category, we take a look at finitely presented modules⁷ over a ring R .

Definition An R -module M is **finitely presented** if there exist $m, n \in \mathbb{N}_0$, $r_1, \dots, r_m \in R^{1 \times n}$ such that $M \cong \frac{R^{1 \times n}}{\langle r_1, \dots, r_m \rangle}$.

Finitely presented modules form a subcategory $R\text{-fpmod}$ of the category of all R -modules. For designing a computable model of $R\text{-fpmod}$, we first need to find a data structure for the objects. Depending on the information we are given about R , we have several competing choices.

Example Any matrix $M \in R^{m \times n}$ can be interpreted as the finitely presented module that is defined by

$$\frac{R^{1 \times n}}{\langle \text{Rows of } M \rangle}.$$

This is the data structure used by `homalg` [7].

Example Assume that R is a coherent ring, i.e., the row syzygies of any matrix with entries in R are finitely generated. Then any pair of matrices $M \in R^{m \times n}$, $N \in R^{o \times n}$ gives rise to the module

$$\frac{\langle \text{Rows of } M \rangle}{\langle \text{Rows of } M \rangle \cap \langle \text{Rows of } N \rangle}$$

which is finitely presented due to the coherence of R . This is the data structure used for example by `MACAULAY2` [5].

Example Assume that R is a principal ideal domain (PID). Then any proper chain of elements $d_1 | \dots | d_m$ and any $n \in \mathbb{N}_0$ define the finitely presented module

$$R^{1 \times n} \oplus \left(\bigoplus_{i=1}^m \frac{R}{\langle d_i \rangle} \right)$$

⁷We will only consider left modules in this article.

⁸Reminder: this includes data structures for objects and morphisms, as well as algorithms for categorical operations.

⁹An additive category is an Ab-category with finite direct sums.

and all finitely presented modules arise in this way due to the structure theorem for PIDs.

If we impose in all three examples the extra condition of R being a computable ring (see [1]), then we can build up three different computable models of $R\text{-fpmod}$ from these three different data structures of the objects. In this case, CAP actually encourages the implementation of three different categories, one for each data structure. CAP provides interfaces for the creation of functors which can be thought of as tools for the conversion of data structures. Moreover, the categorical language that operates within each different model is always the same: a categorical term as presented in equation (1) can always be interpreted in any of the three models above, regardless of the underlying data structures.

Category constructors

Categories in CAP are first class citizens, i.e., they can be part of the input/output of operations. We call any operation that outputs a category a **category constructor**. Category constructors are helpful for quickly building up computable models⁸ of categories.

Example CAP offers a category constructor for **Serre quotients**: given an abelian category \mathbf{A} and a Serre subcategory \mathbf{C} (given by a membership function), there is an abelian category $\frac{\mathbf{A}}{\mathbf{C}}$ universal with the property that objects of \mathbf{C} are treated as zero objects within \mathbf{A} . This category can be employed to perform computations with coherent sheaves on toric varieties [6].

Example Given an abelian category \mathbf{A} , its **generalized morphism category** $\mathbf{G}(\mathbf{A})$ can be employed to perform diagram chases constructively [9]. All operations in $\mathbf{G}(\mathbf{A})$ are derived from the operations in \mathbf{A} .

If one wishes to perform diagram chases with coherent sheaves on toric varieties, one could simply concatenate the Serre quotient constructor with the generalized morphism constructor.

During our development and usage of CAP, we learned to appreciate more and more the impressive power of category theory as an organizational principle. It lets one build up complex mathematical entities step by step from easily graspable concepts. Each step, if implemented with the necessary categorical rigor, interacts smoothly and coherently with the previous and the next step.

Example A good example for the benefits of category constructors are **Freyd categories**: given an additive category⁹ \mathbf{A} , we can turn its morphisms to the objects of a new category $\mathcal{A}(\mathbf{A})$. Morphisms in $\mathcal{A}(\mathbf{A})$ are then defined as commutative squares considered up to homotopy [8]. Depending on the input category \mathbf{A} , we get computable models of different categories:

1. If R is a computable ring and mat_R the category of matrices over R , then $\mathcal{A}(\text{mat}_R) \simeq R\text{-fpmo}$, and we have recovered our first model for finitely presented modules.
2. If S is a G -graded computable ring for an additively written abelian group G , then we let grmat_S denote the category of graded matrices¹⁰ over S . Now, $\mathcal{A}(\text{grmat}_S)$ is equivalent to the category of finitely presented graded S -modules.
3. The iterated construction $\mathcal{A}(\mathcal{A}(\text{mat}_R)^{\text{op}})$ is equivalent to the functor category generated by all finitely presented functors from $R\text{-fpmo}$ to the category of abelian groups. In the case where R is computable and commutative, Tor- and Ext-functors are finitely presented and we can calculate the sets of natural transformations between any pair of such functors [8].

We see that category constructors can take massive advantage of reusing code for building up quite different mathematical entities.

Enhancing CAP

GAP programmers who want to implement their own computable category can make use of CAP's helpful derivation mechanism: if you install only a subset of the available categorical primitives, CAP will try to install as much of the remaining primitives as it can. For example, CAP knows the usual textbook derivation of pullbacks from kernels and direct sums: the pullback object of $A \xrightarrow{\alpha} B \xleftarrow{\gamma} C$ is given (up to isomorphism) by

$$\text{KernelObject}\left(\begin{pmatrix} \alpha \\ -\gamma \end{pmatrix} : A \oplus C \longrightarrow B\right).$$

CAP's current implementation of the computable model of $R\text{-fpmo}$ (using single matrices as its data structure for the objects) was implemented by directly providing algorithms for 35 primitives, the total number of 181 available primitives are due to the derivation mechanism.

The future of CAP

We plan to extend our arsenal of category constructors within the realm of additive categories, dg categories, A_∞ -categories, monoidal categories, and toposes. Moreover, we are interested in the study and effective application of type theory for our categorical approach to computer algebra.

We believe that computer algebra systems can greatly profit from a categorical organization both in

their expressiveness and computational power. It is our goal to explore and reveal with CAP as many of these profits as possible.

References

- [1] Mohamed Barakat and Markus Lange-Hegermann. An axiomatic setup for algorithmic homological algebra and an alternative approach to localization. *J. Algebra Appl.*, 10(2):269–293, 2011. (arXiv:1003.1943).
- [2] Martin Bies. *Cohomologies of coherent sheaves and massless spectra in F-theory*. PhD thesis, Heidelberg U., 2018-02. (arXiv:1802.08860).
- [3] Clement Dupont. *Periods of hyperplane arrangements and motivic coproduct*. PhD thesis, Université Pierre et Marie Curie - Paris VI, 2014. (<https://tel.archives-ouvertes.fr/tel-01083524/document>).
- [4] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.9.1*, 2018. (<http://www.gap-system.org>).
- [5] Daniel R. Grayson and Michael E. Stillman. Macaulay2, a software system for research in algebraic geometry. <http://www.math.uiuc.edu/Macaulay2/>.
- [6] Sebastian Gutsche. *Constructive category theory and applications to algebraic geometry*. PhD thesis, University of Siegen, 2017. (<http://dokumentix.ub.uni-siegen.de/opus/volltexte/2017/1241/>).
- [7] homalg project authors. The homalg project – Algorithmic Homological Algebra. (<http://homalg-project.github.io>), 2003–2017.
- [8] Sebastian Posur. A constructive approach to Freyd categories. *ArXiv e-prints*, December 2017. (arXiv:1712.03492).
- [9] Sebastian Posur. *Constructive Category Theory and Applications to Equivariant Sheaves*. PhD thesis, University of Siegen, 2017. (<http://dokumentix.ub.uni-siegen.de/opus/volltexte/2017/1179/>).
- [10] Sebastian Posur. Constructing equivariant vector bundles via the bgg correspondence. *Journal of Symbolic Computation*, 91:57 – 73, 2019. MEGA 2017, Effective Methods in Algebraic Geometry, Nice (France), June 12-16, 2017.
- [11] The QPA-team. *QPA – Quivers and path algebras*, 2012. (<http://www.math.ntnu.no/~oyvinso/QPA/>).

¹⁰An object in grmat_S is a finite list of elements in G , and a morphism between two such lists $(a_i)_i \in G^m$ and $(b_j)_j \in G^n$ for $m, n \in \mathbb{N}_0$ is a matrix $(s_{ij})_{ij} \in S^{m \times n}$ whose entries s_{ij} are either zero or homogeneous of degree $b_j - a_i$. Composition is given by matrix multiplication (with swapped arguments).



Viele Trassierungsaufgaben sind unsinnig!

J. Meyer
(Hameln)

j.m.meyer@t-online.de

Zusammenfassung

Die schulüblichen Trassierungsaufgaben beachten oftmals nicht die Abhängigkeit von der Orientierung des Koordinatensystems. Dreht man die vorgegebenen Elemente, so resultieren recht unterschiedliche Kurventypen.

Einleitung

In früheren Zeiten waren Kurvendiskussionen ein wesentlicher Bestandteil des schriftlichen Abiturs in Mathematik. Man ist davon abgekommen, weil man eingesehen hat, dass Kurvendiskussionen häufig sinnarm waren und weil sie schematisch durchgeführt werden konnten und man somit nicht überprüfen konnte, ob die Schülerinnen und Schüler überhaupt irgendetwas inhaltlich verstanden hatten.

In den letzten Jahren fanden sich Trassierungsaufgaben in einigen Schulbuch- und Abituraufgaben; dabei handelt es sich um das „Problem“, zu zwei vorgegebenen geradlinigen Straßenabschnitten eine möglichst „gute“ Verbindung herzustellen, d. h. die Verbindungskurve soll sich stetig, knick- und krümmungsruckfrei an die vorhandenen Straßen anschließen. Setzt man die Kurve als Graph eines möglichst niedriggradigen Polynoms an, so ist ein lineares Gleichungssystem zu lösen. Dass auch hier das Lösungsverfahren leicht schematisierbar war, trug zur Beliebtheit solcher Trassierungsaufgaben bei. Außerdem meinte man, dadurch den Realitätsbezug des Mathematikunterrichts gesteigert zu haben. Diese Meinung besteht zu Unrecht: Solche Aufgaben haben mit der Realität gar nichts zu tun!

Der einfachste Fall

Sehen wir uns den einfachsten Fall an: Zwei zueinander parallele Geradenstücke sind gegeben. Man kann das Koordinatensystem so wählen, dass die beiden Strecken symmetrisch zum Ursprung sind. Damit hat das Zielpolynom f eine einfache Struktur. Da es in beiden

Verbindungspunkten nur drei Bedingungen gibt, kommt man mit Grad 5 aus. Beginnt eines der Geradenstücke im Punkt

$$P_0 = \begin{pmatrix} u_0 \\ v_0 \end{pmatrix},$$

so müssen $f(u_0) = v_0$ und $f'(u_0) = m_0$ (als Steigung der Strecke) sowie $f''(u_0) = 0$ sein. Dieses Gleichungssystem wird man nicht per Hand lösen wollen. Abb. 1 zeigt das Resultat für $v_0 = m_0 = 1$ und $u_0 = 2$.

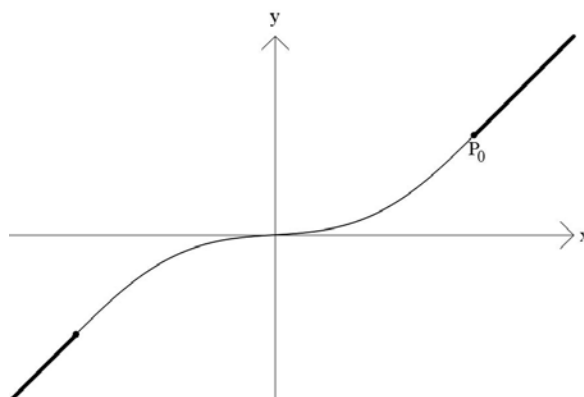


Abbildung 1: Eine überzeugend wirkende Lösung

So überzeugend die Lösung aussehen mag, so sinnlos ist sie auch. Dies merkt man, wenn man die vorgegebenen Elemente um den Ursprung mit dem Winkel φ dreht. Mit der Drehmatrix

$$M = \begin{pmatrix} \cos \varphi & -\sin \varphi \\ \sin \varphi & \cos \varphi \end{pmatrix}$$

wird P_0 auf

$$P_\varphi = M \cdot P_0 = \begin{pmatrix} u_\varphi \\ v_\varphi \end{pmatrix}$$

abgebildet.

Wie bekommt man die neue Steigung m_φ ? Zwar ist

$$m_\varphi = \tan(\arctan(m_0) + \varphi),$$

man kann jedoch auch folgendermaßen vorgehen:
Auf der Strecke mit Anfangspunkt P_0 und Steigung m_0 liegt auch der Punkt

$$Q_0 = \begin{pmatrix} u_0 + 1 \\ v_0 + m_0 \end{pmatrix}.$$

Wegen

$$M \cdot Q_0 - M \cdot P_0 = M \cdot \begin{pmatrix} 1 \\ m_0 \end{pmatrix} = \begin{pmatrix} \cos \varphi - m_0 \cdot \sin \varphi \\ \sin \varphi + m_0 \cdot \cos \varphi \end{pmatrix}$$

hat die neue Steigung den Wert

$$m_\varphi = \frac{\sin \varphi + m_0 \cdot \cos \varphi}{\cos \varphi - m_0 \cdot \sin \varphi}.$$

Nun müssen für das Zielpolynom f die Bedingungen $f(u_\varphi) = v_\varphi$ und $f'(u_\varphi) = m_\varphi$ sowie $f''(u_\varphi) = 0$ erfüllt werden.

Einige Resultate

Weiterhin seien $v_0 = m_0 = 1$ und $u_0 = 2$.

Für $\varphi = 40^\circ$ bekommt man den in Abb. 2 gezeigten Straßenverlauf, der sich deutlich von dem der Abb. 1 unterscheidet.

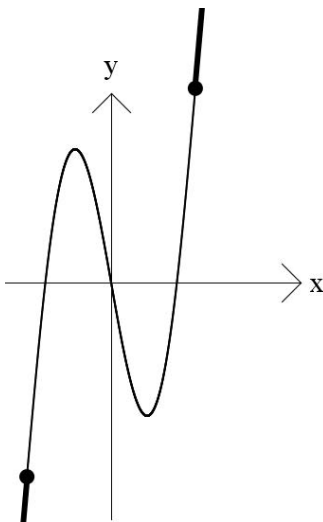


Abbildung 2: Der Verlauf nach Drehung um 40°

Wiederum anders ist es in Abb. 3, die zu $\varphi = -40^\circ$ gehört.

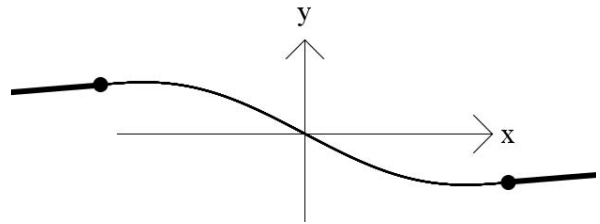


Abbildung 3: Der Verlauf nach Drehung um -40°

Der obige Ansatz setzt ganz wesentlich ein Koordinatensystem und dessen Ausrichtung voraus. Unterschiedlich gedrehte Koordinatensysteme führen zu unterschiedlichen Lösungen. Da es im Straßenbau kein ausgezeichnetes Koordinatensystem gibt, *muss* man auf Kurven zurückgreifen, die keine Funktionsgraphen sind, sondern sich allein durch ihre inneren Eigenschaften beschreiben lassen, etwa dadurch, dass deren Krümmung linear mit dem auf der Kurve zurückgelegten Weg wächst. Die *Klothoide* hat diese Eigenschaft; sie ist allerdings für den Mathematikunterricht zu kompliziert.

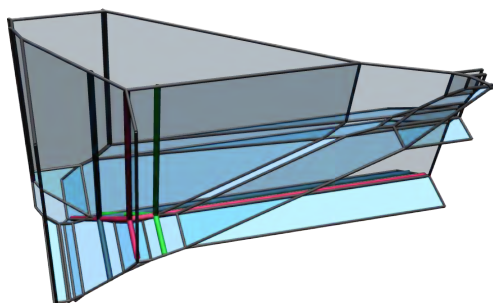
SFB/TRR 195 Symbolic Tools in Mathematics and their Application (Part 3/5)

Tropical and Polyhedral Geometry

Tropical and polyhedral geometry is one of the core areas in the SFB-TRR 195. Polymake, which is one of the cornerstone computer algebra systems that continue to be developed within the SFB-TRR 195, allows computations with objects from polyhedral geometry such as polytopes, polyhedra, cones, fans and polyhedral complexes.

Tropical geometry provides a bridge linking algebraic geometry with polyhedral geometry: the degeneration method referred to as tropicalization produces a polyhedral complex from an algebraic variety, from which important properties can be read off. This bridge can then be used to transfer methods from one area of research to the other.

For computations in tropical geometry, we have to rely on the one hand on computations involving ideals, as common for computations in algebraic geometry, and on the other hand on computations involving polyhedral complexes. The desire to provide an integrated computer algebra system combining the functionality of various existing computer algebra systems - as envisioned by the system OSCAR which is being developed in the SFB-TRR 195 - is thus inherent in tropical geometry.

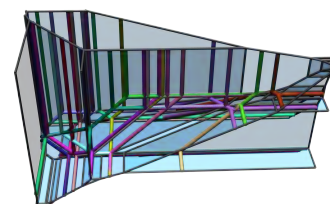


The tropical and polyhedral projects of the SFB-TRR 195 share an algorithmic and experimental approach to important theoretical questions in the area. Our leading goal is to advance the study of higher-dimensional tropical varieties and their moduli spaces, in particular the interplay between abstract tropical varieties and their embeddings. In a project aiming at new Polymake features important for tropical geometry and their theoretical background (Michael Joswig), we have for example obtained a major enhancement for the computation of secondary fans. The latter allowed the computation of the moduli space of tropical cubic surfaces in \mathbb{R}^3 which was out of reach of the existing tools. With this, new examples of (families of) tropical lines in a tropical cubic surface were discovered, as shown in the pictures.

In a project dealing with moduli spaces of tropical varieties (Michael Joswig, Hannah Markwig, Thomas

Markwig), we have constructed moduli spaces of plane tropical curves, illuminating the relation between the space of abstract curves of a fixed genus and the space of embedded curves. In a project dealing with mirror symmetry of elliptic curves (Janko Böhm, Hannah Markwig), we make use of the bridge between polyhedral and algebraic geometry provided by tropical geometry by proving relations involving generating functions for counts of algebraic curves in surfaces and Feynman integrals by means of tropical geometry. A computational method which plays a role in several of these projects is parallelization. It is used e.g. to list triangulations for secondary fans, or to compute Feynman integrals. This connecting theme provides not only a common computational framework for the tropical and polyhedral projects, but also links to other projects in the SFB-TRR 195 and to the development of OSCAR.

The research continues in all of the projects above, and will shed more light on higher-dimensional tropical varieties in the future.



The tropical and polyhedral projects are well-centered in the SFB-TRR 195 and feature connections to all other core areas of the SFB-TRR 195. The research in the tropical projects obviously profits from collaborations with the algebraic geometers of the SFB-TRR 195, but also from the connections to number theory in the context of semistable reduction (which can be viewed as an ingredient to abstract tropicalization) and to the representation theory of the symmetric group in the context of mirror symmetry. More surprisingly, the theory of Hurwitz numbers also provides a link to the fifth core area, random matrices and free probability theory. This is the center of attention of a project dealing with random matrices and Hurwitz numbers (Hannah Markwig, Roland Speicher), in which we have generalized existing variants of the Hurwitz problem and study their combinatorial properties. Further aims of the SFB-TRR 195 within the fifth core area will be presented in the coming issue.

Hannah Markwig (Tübingen)

Berufungen

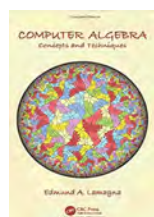
Dr. Simon Brandhorst hat den Ruf auf eine Juniorprofessur für Konstruktive Methoden der Algebra an der Universität des Saarlandes angenommen und am 14.12.2018 angetreten.

Publikationen über Computeralgebra

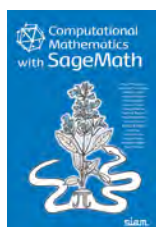
Neuerscheinungen:



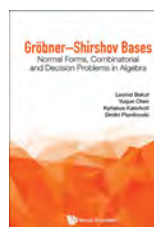
Hans-Gert Gräbe,
EAGLE-Starthilfe
Computeralgebra im Abitur,
Edition am Gutenbergplatz,
Leipzig, 2018, 114 Seiten,
ISBN 978-3959221047



Edmund A. Lamagna,
Computer Algebra:
Concepts and Techniques,
CRC Press, 2018, 372 Seiten,
ISBN 978-1138093140



Paul Zimmermann et. al.,
Computational Mathematics
with SageMath,
SIAM, 2018, 464 Seiten,
ISBN 978-1611975451



Leonid Bokut, Yuqun Chen, Kyriakos
Kalorkoti, Dmitri Piontlovski,
Gröbner-Shirshov Bases,
World Scientific,
Feb. 2019, 450 Seiten,
ISBN 978-9814619486

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 Florian Heß oder Martin Kreuzer (florian.hess@uni-oldenburg.de, martin.kreuzer@uni-passau.de) auf.

Hans-Gert Gräbe

EAGLE-Starthilfe Computeralgebra im Abitur

Edition am Gutenbergplatz, Leipzig 2018, 114 Seiten, ISBN 978-3-95922-104-7, € 14,50

In vielen Bundesländern ist mittlerweile der Einsatz von Computeralgebra-Systemen (CAS) im Mathematikunterricht der Oberstufe an der Tagesordnung, und oftmals wird auch im Abitur ein Satz von CAS-Aufgaben angeboten. In dem vorliegenden Büchlein geht mit Hans-Gert Gräbe ein ausgewiesener Computeralgebra-Experte und erfahrener Dozent die Aufgabe an, die Welt der computerunterstützten Mathematik an Hand des Open Source Systems *Maxima* in Form einer “Start-hilfe” einem Zielpublikum aus interessierten Schülern und Lehrern näher zu bringen. Dabei liefert er ganz gezielt nicht eine weitere Rezeptsammlung zur Abiturvorbereitung und auch keinen Diskussionsbeitrag zu dem Thema, ob man ein CAS überhaupt vernünftig in eine schriftliche Abiturprüfung einbinden kann und sollte. Stattdessen werden Einsatzmöglichkeiten der Computeralgebra vorgeführt, wie man sie sinnvoll zum Erwerb der von diversen Lehrplänen und Bildungsstandards versprochenen Kompetenzen der Abiturienten einbringen kann.

Dazu werden im zweiten Kapitel die Grundkonzepte eines CAS vorgestellt und mit den entsprechenden mathematischen Konzepten in Bezug gesetzt. Danach folgt in Kapitel 3 der zentrale Teil des Werks: die Aufgaben des sächsischen Abiturs zum Leistungskurs Mathematik 2009 werden mit einem Lösungsplan in Angriff genommen, der eine sinnvolle CAS-Unterstützung mit einschließt. Dies gelingt in allen drei Teilprüfungsbereichen: Analysis, Geometrie/Algebra und Stochastik.

Das Buch ist in sehr klarer und verständlicher Weise geschrieben, die sich auch der avisierten Leserschaft gut erschließen sollte. Prägnant formulierte Merksätze in farblich abgesetzten Kästchen, klarer, gut lesbarer

und stets vollständiger Source-Code sowie eine schulnahe und nie überfrachtete Notation tragen zum Lesevergnügen bei. Dabei wird das Bändchen aber nie zur Bettlektüre, denn jegliche Problemlösung und viele Nachdenkaufgaben wollen am Computer implementiert und ausprobiert werden.



Mit diesem Büchlein hat Hans-Gert Gräbe eine rundum gelungene “Starthilfe” geschaffen, die nicht nur Schülern und Lehrern einen ganz frischen Blick auf die CAS-Welt ermöglicht, sondern die auch wichtige Anregungen liefert, wie man den Mathematik- und den Informatikunterricht in der Oberstufe mit CAS-Unterstützung neu denken und neu organisieren könnte. Daher sei dieses Werk auch und gerade Lehrplanentwicklern und Didaktik-Experten zum Durcharbeiten und Durchdenken dringend empfohlen.

Martin Kreuzer (Passau)

Daniel Heinlein: Integer linear programming techniques for constant dimension codes and related structures

Betreuer: Sascha Kurz (Bayreuth), Alfred Wassermann (Bayreuth)

Zweitgutachter: Michael Stoll (Bayreuth), Leo Storme (Gent)

November 2018

Abstract: The lattice of subspaces of a finite dimensional vector space over a finite field \mathbb{F}_q is combined with the subspace distance $d(U, W) = \dim(U + W) - \dim(U \cap W)$ a metric space. A constant dimension code (CDC) is a set of k -dimensional subspaces of \mathbb{F}_q^v with pairwise subspace distance at least d , which is equivalent to an upper bound on the pairwise intersection. CDCs play a vital role in the context of random linear network coding, in which data is transmitted from a sender to multiple receivers such that participants of the communication forward random linear combinations of the data. The two main problems are the determination of the cardinality of largest CDCs and their classification (up to symmetry). This thesis provides constructions, classifications, and bounds in special cases, which can e.g. be seen on the associated homepage

<http://subspacecodes.uni-bayreuth.de/>.

We increase the ratio of “best known lower bound” / “best known upper bound” to at least 61.6% for all parameters. In particular, we present a method for an exhaustive search for subgroups with special properties and couple this search with Kramer-Mesner computations to find the currently best known CDC in the binary Fano setting, i.e., a set of 333 planes in the \mathbb{F}_2^7 mutually intersecting in at most a point. By a sophisticated computer-aided classification of maximum cardinality CDCs in \mathbb{F}_2^8 with $k = 4$ and $d = 6$ we determine the third known maximum cardinality and second known classification for nontrivial parameters with $d < 2 \min\{k, v - k\}$.

Johannes Hoffmann: Left saturation closure - theory and algorithms

Betreuer: Viktor Levandovskyy (Aachen)

Zweitgutachter: Eva Zerz (Aachen)

Januar 2019

Abstract: Die Theorie der Ore-Lokalisierung von Ringen und Moduln findet Verwendung in vielen Bereichen der nicht-kommutativen Algebra, zum Beispiel Ringtheorie, Di-

mensionstheorie, nicht-kommutative Geometrie und viele andere. In dieser Dissertation verfolgen wir mehrere Ziele. Wir beginnen mit einer gründlichen Einführung zum Thema Ore-Lokalisierung, aufbauend auf einer axiomatischen Definition und der klassischen Konstruktion, die von Ores Arbeiten inspiriert ist. Ore-Lokalisierung ist problemlos mit der Ringaddition verträglich, allerdings ist deren Existenz für die Konstruktion überhaupt nicht notwendig. Aus diesem Grund untersuchen wir, wie sich die Theorie verändert, wenn wir statt Ringen Monoide betrachten.

Der Hauptbeitrag dieser Dissertation ist das Konzept des *Linkssaturationsabschlusses* oder kurz *LSat* und die sich daraus ergebenden Anwendungen: Für eine Linksnennermenge S in einem beliebigen Ring R stellt $\text{LSat}(S)$ eine kanonische Form von S bezüglich R -fixierender Homomorphismen von Lokalisierungen von R dar. Weiterhin spielt $\text{LSat}(S)$ eine bedeutende Rolle in Charakterisierungen von links-invertierbaren Elementen und Linksidealien sowie weiterer struktureller Eigenschaften der Lokalisierung $S^{-1}R$. Wir beweisen, dass $\text{LSat}(S)$ genau dann eine Linksnennermenge in R ist, wenn $S^{-1}R$ Dedekind-endlich ist. In diesem Fall ist $S^{-1}R$ auf kanonische Art und Weise isomorph zu $\text{LSat}(S)^{-1}R$ und zusätzlich liefert $\text{LSat}(S)$ eine vollständige Beschreibung der Einheitengruppe von $S^{-1}R$. Wir untersuchen Klassen von Ringen, die diese Eigenschaft erfüllen: In Ore-Bereichen charakterisieren wir maximale und prämaximale Linksnennermengen und in Faktorisierungsbereichen beweisen wir, dass jede saturierte Linksnennermenge eindeutig durch die darin enthaltenen irreduziblen Elemente bestimmt ist. Wir betrachten beispielhaft mehrere Linksnennermengen in Weyl-Algebren und die dazugehörigen Lokalisierungen.

Im Anschluss erweitern wir unsere strukturellen Untersuchungen auf Ore-lokalisierte Moduln. Insbesondere zeigen wir, dass das etablierte Konzept des lokalen Abschlusses von Untermoduln ein weiterer Spezialfall des Linkssaturationsabschlusses ist. Als Spezialfall des lokalen Abschlusses betrachten wir lokale Torsion, was wiederum eine Verallgemeinerung des klassischen Torsionsbegriffs ist, und zeigen weitere Verbindungen zwischen den zwei Konzepten auf.

Im letzten Teil dieser Dissertation geben wir algorithmische Lösungen für diverse Fragestellungen im Rahmen von Ore-Lokalisierungen von G -Algebren, kurz OLGAs. Wir beschreiben unser Konzept für grundlegende Arithmetik in OLGAs und deren Implementierung im Computeralgebra-System SINGULAR:PLURAL. Abschließend entwickeln wir mehrere Algorithmen, um den lokalen Abschluss in wichtigen Spezialfällen zu berechnen.

Berichte von Konferenzen

Core Computational Methods

Providence, USA, 17.09. – 21.09.2018

icerm.brown.edu/programs/sp-f18/w1

Der Workshop “Core Computational Methods” leitete das spezielle Halbjahr “Nonlinear Algebra” am ICERM Institut an der Brown University, Providence, Rhode Island (USA) ein. Sein Thema war das Zusammenspiel der drei Hauptbereiche, aus denen die Algorithmen der nichtlinearen Algebra stammen: numerische algebraische Geometrie, symbolische Berechnungen und kombinatorische Methoden. Unter der fachmännischen Leitung von Jesus de Loera, Mike Stillman, Wolfram Decker und Andrew Sommese fand sich ein hochkarätiges Teilnehmerfeld ein, das zahlreiche sehr aktuelle und interessante neue Entwicklungen präsentierte. Einige (nicht repräsentative) Highlights aus der Sicht des Berichterstatters waren neuartige Gröbner-Basen aus der tro-

pischen Geometrie (Diane Maclagan), eine Anwendung der tropischen PCA auf phylogenetische Bäume (Ruriko Yoshida) und extrem schnelle und überraschende numerische Methoden für *exakte* Berechnungen in der algebraischen Geometrie (Chris Peterson).

Eine Besonderheit der Workshops am ICERM Institut ist, dass die Vorträge als Videoaufzeichnung jederzeit abrufbar sind, in diesem Fall unter der Adresse

<https://icerm.brown.edu/programs/sp-f18/w1/#lecturevideos>

Die Mitarbeiter am ICERM Institut sind äußerst hilfsbereit und kompetent, so dass eine Teilnahme an einem der Programme dort eine große Bereicherung und sehr zu empfehlen ist.

Martin Kreuzer (Passau)

Hinweise auf Konferenzen

PCA 2019

St. Petersburg, Russland, 15.04. – 20.04.2019

pca-pdmi.ru/2019/

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'2019 is the 12-th 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.

Computeralgebra-Tagung der Fachgruppe

Kassel, 16.05. – 18.05.2019

www.fachgruppe-computeralgebra.de/kassel-2019

In Fortsetzung der erfolgreichen Tagungen 2003, 2005, 2009, 2012, 2014, 2017 in Kassel und 2007 in Kaiserslautern führt die Fachgruppe vom 16. bis 18. Mai 2019 wieder eine derartige Tagung in Kassel durch. Das Ziel dieser Tagungsreihe ist es einerseits Nachwuchswissenschaftlern zu ermöglichen ihre Ergebnisse vorzustellen, andererseits aber auch einige Hauptvortragende zu gewinnen, die Übersichtsvorträge über wichtige Gebiete der Computeralgebra und über Computeralgebra-Software geben.

Siehe ausführliche Ankündigung auf Seite 6.

MEGA 2019

Madrid, Spanien, 17.06. – 21.06.2019

eventos.ucm.es/12097/detail/mega-2019.html

MEGA is the acronym for Effective Methods in Algebraic Geometry (and its equivalent in Italian, French, Spanish, German, Russian, etc.). MEGA conferences have emerged

from the leading research activity in Effective Algebraic Geometry developed in Europe over the last decades. They have established themselves as an important vehicle in the exploration of the broad interactions between Algebraic Geometry and Computational and Applied domains of Mathematics. The conferences have also increasingly been attracting young talented researchers, showing the strength and vitality of the community. The MEGA conferences aim at promoting the development of the subject through academic exchange between International researchers; at presenting the recent progresses of the domain through the presentation of selected talks; at updating the community knowledge through the invitation of talented researchers from Europe and outside; and at attracting and supporting young students and researchers to join the current research area.

CAI 2019

Niš, Serbien, 30.06. – 04.07.2019

www.pmf.ni.ac.rs/CAI2019

CAI is the biennial conference serving the community interested in the intersection of theoretical computer science, algebra, and related areas.

In 2019 it will feature invited presentations and a selective single-track program of contributed papers describing original and unpublished research.

CAI 2019 is hosted at the University of Niš, Serbia, and organized by the Faculty of Sciences and Mathematics of this university.

CICM 2019

Prag, Tschechien, 08.07. – 12.07.2019

www.cicm-conference.org/2019

Digital and computational solutions are becoming the prevalent means for the generation, communication, processing, storage and curation of mathematical information. Separate

communities have developed to investigate and build computer based systems for computer algebra, automated deduction, and mathematical publishing as well as novel user interfaces. While all of these systems excel in their own right, their integration can lead to synergies offering significant added value. The Conference on Intelligent Computer Mathematics (CICM) offers a venue for discussing and developing solutions to the great challenges posed by the integration of these diverse areas.

AAG 2019

Bern, Schweiz, 09.07. – 13.07.2019

mathsites.unibe.ch/siamag19

The SIAM Conference on Applied Algebraic Geometry (AAG '19) will take place in Bern, Switzerland. The purpose of the SIAM Activity Group in Algebraic Geometry is to bring together researchers who use algebraic geometry in industrial and applied mathematics. „Algebraic geometry“ is interpreted broadly to include at least: algebraic geometry, commutative algebra, noncommutative algebra, symbolic and numeric computation, algebraic and geometric combinatorics, representation theory, and algebraic topology. These methods have already seen applications in: biology, coding theory, cryptography, combustion, computational geometry, computer graphics, quantum computing, control theory, geometric design, complexity theory, machine learning, nonlinear partial differential equations, optimization, robotics, and statistics. We welcome participation from both theoretical mathematical areas and application areas not on this list which fall under this broadly interpreted notion of algebraic geometry and its applications.

SC-square Workshop 2019

Bern, Schweiz, 10.07.2019

www.sc-square.org/CSA/workshop4.html

The 4th International Workshop on Satisfiability Checking and Symbolic Computation will be held on Wednesday 10 July 2019 in Bern, Switzerland. It will be minisymposium at The SIAM Conference on Applied Algebraic Geometry (AAG 2019).

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.

ISSAC 2019

Peking, China, 15.07. – 18.07.2019

www.issac-conference.org/2019

The International Symposium on Symbolic and Algebraic Computation (ISSAC) is the premier conference for research in symbolic computation and computer algebra. ISSAC 2019 will be the 44th 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 2019 will be held on 15-18 July 2019, at Beihang University, Beijing, China

ACA 2019

Montreal, Kanada, 16.07. – 20.07.2019

aca2019.etsmtl.ca

The 25th Conference on Applications of Computer Algebra (ACA) will be held in Montréal, Canada. This event will take place from Tuesday July 16 to Saturday July 20, 2019. École de technologie supérieure will host the international conference for a second time, 10 years after ACA 2009.

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.

CASC 2019

Moskau, Russland, 26.08. – 30.08.2019

www.casc-conference.org/2019

The 21st International Workshop on Computer Algebra in Scientific Computing, CASC 2019, will be held in the city of Moscow, Russia, August 26 - 30, 2019.

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 2019

Karlsruhe, 23.09. – 26.09.2019

dmv2019.math.kit.edu

Wir laden Sie herzlich zur DMV-Jahrestagung 2019 in Karlsruhe ein! Zur Eröffnung wird Hélène Esnault die Cantor-Medaille 2019 verliehen. Während der Tagung findet die jährliche DMV-Mitgliederversammlung statt.

Sie können das wissenschaftliche Programm mitgestalten, indem Sie ein Minisymposium anmelden oder mit einem Vortrag in den Sektionen beitragen. Bachelor- und Masterstudierende können sich an der DMV-Studierendenkonferenz beteiligen, und zum Thema ‚Mathematik in Industrie und Gesellschaft‘ findet wieder ein Mittagseminar statt.

GI-Jahrestagung 2019

Kassel, 23.09. – 26.09.2019

www.informatik2019.de

Die Jahrestagung INFORMATIK 2019 wird vom 23. bis 26. September 2019 an der Universität Kassel stattfinden. Die 49. GI-Jahrestagung INFORMATIK 2019 bringt Informatikerinnen und Informatiker aus Bildung, Wissenschaft und Wirtschaft zusammen, um aktuelle Entwicklungen zu diskutieren, neue Kontakte zu knüpfen und gemeinsame Aktivitäten zu planen.

Anlässlich des 50. Geburtstages der Gesellschaft für Informatik lautet das Tagungsmotto: „50 Jahre Gesellschaft für Informatik – Informatik für Gesellschaft“.

Antrag auf Mitgliedschaft in der Fachgruppe Computeralgebra der GI in Kooperation mit der DMV und GAMM und auf Bezug des Computeralgebra-Rundbriefs

Bitte zurücksenden an:

Prof. Dr. Wolfram Koepf
Universität Kassel
FB Mathematik/Informatik
Heinrich-Plett-Str. 40
D-34132 Kassel



Name:	Vorname:
Akadem. Grad:	Geburtsjahr:
<i>Privatanschrift:</i>	
Straße/Postfach:	PLZ Ort:
Telefon:	Telefax:
<i>Dienstanschrift:</i>	
Firma/Institut:	Abteilung:
Straße/Postfach:	PLZ Ort:
Telefon:	Telefax:
E-Mail:	
Gewünschte Postanschrift: <input type="checkbox"/> Privatanschrift <input type="checkbox"/> Dienstanschrift	
Gewünschte Regionalgruppenzuordnung: (http://regionalgruppen.gi.de)	

- ☐ Ich bin persönliches Mitglied der GI und beantrage die Mitgliedschaft in der Fachgruppe Computeralgebra sowie den Bezug des Rundbriefs
- ☐ Ich beantrage assoziierte Mitgliedschaft in der GI und Mitgliedschaft in der Fachgruppe Computeralgebra sowie den Bezug des Rundbriefs
- ☐ ab 1. Januar
- ☐ rückwirkend zum 1. Januar des laufenden Jahres (bis zum 30. September möglich).

Ich ordne mich folgender Jahresbeitragsklasse zu:

- ☐ 7,50 Euro für Mitglieder der ☐ GI ☐ DMV ☐ GAMM,

Mitgliedsnummer:

- ☐ 7,50 Euro. Ich beantrage gleichzeitig Mitgliedschaft in der ☐ GI ☐ DMV ☐ GAMM und bitte um Zusendung der dazu erforderlichen Unterlagen.

- ☐ 9,00 Euro für Nichtmitglieder. Ich bitte um Zusendung von Informationen über ☐ GI ☐ DMV ☐ GAMM.

- ☐ Ich bitte lediglich um Aktualisierung meiner Adressdaten sowie meiner Angaben über die Zusendung von Informationen.

Ich nehme zur Kenntnis, dass die Aufnahme in die Fachgruppe Computeralgebra zum 1.1. erfolgt und dass die Mitgliedschaft zum 31.12. mit Frist 30.11. schriftlich gekündigt werden kann.

Datennutzung

Meine oben angegebenen personenbezogenen Daten werden im Rahmen meiner Mitgliedschaft soweit gesetzlich erlaubt oder aufgrund meiner Einwilligung durch die GI oder durch Dritte nach Weitergabe durch die GI wie folgt genutzt:

- ☐ für alle GI-gesellschaftsinternen Aussendungen,
- ☐ für von der GI ausgewählte Informationen mit Bezug zur Informatik, z.B. Weiterbildungsangebote, Informatikveranstaltungen oder -kongresse mit und ohne GI-Beteiligung sowie Publikationen mit Informatikbezug.

Wenn Sie uns Ihre E-Mail-Adresse angegeben haben, wird die Kommunikation soweit möglich elektronisch ausgeführt.

- ☐ Der Nutzung meiner E-Mail-Adresse zu Zwecken, die über die satzungsgemäßen Ziele der GI hinausgehen (wie z.B. Werbung, Markt- und Meinungsforschung) stimme ich zu.

Natürlich können Sie Ihre Zustimmung jederzeit widerrufen oder Ihre E-Mail-Adresse in unserem System löschen lassen, kurze Nachricht an mitgliederservice@gi.de, per Post oder Fax genügt.

Datum: Unterschrift:

Rückfragen: Telefon +49 (0)228-302-151/-149 Telefax +49 (0)228-302-167 E-Mail: mitgliederservice@gi.de <http://gi.de>

Fachgruppenleitung Computeralgebra 2017–2020

**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-m11.ma.tum.de/~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/>

**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>

**Fachreferent 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>

**Fachreferent CA an der Hochschule:**

Prof. Dr. Jürgen Klüners
Mathematisches Institut der Universität Paderborn
Warburger Str. 100, 33098 Paderborn
05251-60-2646, -3516 (Fax)
klueners@math.uni-paderborn.de
<http://www2.math.uni-paderborn.de/people/juergen-klueners.html>

**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>

**Fachexperte Redaktion Rundbrief:**

Dr. Fabian Reimers
Zentrum Mathematik – M11
Technische Universität München
Boltzmannstr. 3, 85748 Garching
089-289-17474
reimers@ma.tum.de
<http://www-m11.ma.tum.de/reimers>

**Stellvertretende Sprecherin:**

Prof. Dr. Anne Frühbis-Krüger
Institut für Algebraische Geometrie
Welfengarten 1, 30167 Hannover
0511-762-3592
fruehbis-krueger@math.uni-hannover.de
<http://www.iag.uni-hannover.de/~anne>

**Fachexpertin Industrie:**

Xenia Bogomolec
Coding Services Hannover
Engelbosteler Damm 15
30167 Hannover
0173-3031816
indigomind@protonmail.ch

**Fachreferent Sonderforschungsbereich 195:**

Prof. Dr. Meinolf Geck
Universität Stuttgart
Institut für Algebra und Zahlentheorie
Pfaffenwaldring 57, 70569 Stuttgart
0711 685-65367
meinolf.geck@mathematik.uni-stuttgart.de
<http://www.mathematik.uni-stuttgart.de/~geckmf/>

**Fachreferent Themen, Anwendungen und Publikationen:**

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

**Vertreter der DMV:**

Prof. Dr. Wolfram Koepf
Institut für Mathematik
Universität Kassel
Heinrich-Plett-Str. 40, 34132 Kassel
0561-804-4207, -4646 (Fax)
koepf@mathematik.uni-kassel.de
<http://www.mathematik.uni-kassel.de/~koepf>

**Fachreferent Schule und Didaktik:**

StD Jan Hendrik Müller
Rivius-Gymnasium der Stadt Attendorf
Westwall 48, 57439 Attendorf
02722-5953 (Sekretariat)
jan.mueller@math.uni-dortmund.de
www.mathebeimueller.de

**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*™ TECHNOLOGIE



BEGEISTERT FÜR MINT.



Vertraute Funktionen.
Neue Möglichkeiten.
Schnellere Interaktivität.

Entdecken Sie den neuen
TI-Nspire™ CX II-T CAS Graphik-
rechner inklusive Software.

education.ti.com/de/nspire