Springer Undergraduate Texts in Mathematics and Technology

SUMAT

Wolfram Koepf

Computer Algebra

An Algorithm-Oriented Introduction

Springer

Springer Undergraduate Texts in Mathematics and Technology

Wolfram Koepf

Computer Algebra

An Algorithm-Oriented Introduction

Springer

Wolfram Koepf
Institut für Mathematik
Universität Kassel
Kassel, Germany

ISSN 1867-5506 ISSN 1867-5514 (electronic)
Springer Undergraduate Texts in Mathematics and Technology
ISBN 978-3-030-78016-6 ISBN 978-3-030-78017-3 (eBook)
https://doi.org/10.1007/978-3-030-78017-3

Mathematics Subject Classification: 68W30, 11Y05, 11Y11, 11Y16, 11R04, 12D05, 12-08, 13F20, 13F25, 33F10

Translation and update from the German language edition: *Computeralgebra - Eine algorithmisch orientierte Einführung* by Wolfram Koepf, © Springer-Verlag Berlin Heidelberg 2006. Published by Springer-Verlag GmbH Germany. All Rights Reserved.

Preface

This textbook about computer algebra gives an introduction to this modern field of Mathematics. It was published in German language in 2006, based on lectures given at the University of Kassel and containing the material of two lectures of four hours per week each, with additional two hours of exercise classes. When the German edition appeared, I had already taught this topic several times, and in the meantime I have taught it a dozen times. My students were bachelor and master students of Mathematics and Computer Science, as well as high school teacher students with Math as a major. The requirements for this course are low, namely just some knowledge on linear algebra and calculus is required. Hence the book can also be used by high school teachers—who might use computer algebra equipment in their classrooms—as a source for better understanding the underlying computer algebra algorithms. I tried hard to include the most important concepts and algorithms of computer algebra, whose proofs are demonstrated as elementarily as possible. Since I did not want to require a course in algebra, I refrained from using deep algebraic arguments. Rather, the presentation is quite implementation-oriented, and therefore more algorithmic than algebraic. For example, the Chinese remainder theorem is presented as an algorithm and not as an isomorphism theorem. For these reasons, this book does not replace a higher algebra course, but advertises an algebraic consolidation.

In my introductory computer algebra course, I normally lecture the contents of the first eight chapters. These topics seem to be an indispensable foundation of computer algebra. Of course all lecturers can set their own priorities. For example, if the lecturing time is too short, I tend to leave out some theorems on finite fields and point the students to the proofs in the book instead. In contrast, another lecturer might omit the chapter about coding theory and cryptography, although my experience is that the students like these kinds of applications. Another option is to teach the algorithms for integers and for polynomials (for example, the extended Euclidean algorithm) in parallel instead of sequentially. Besides, if knowledge about higher algebra can be assumed, some proofs get much shorter. In any case, all lecturers have enough possibilities to keep their own style.

In the first two chapters, we present the abilities of a general-purpose computer algebra system, and we show how mathematical algorithms can be programmed in such a system. Next, integer arithmetic is introduced, which defines the core of every computer algebra system. After discussing modular arithmetic with the Chinese remainder theorem, the Little Fermat theorem and primality testing, a chapter about coding theory and cryptography follows in which these number-theoretic foundations are applied. Here, the RSA cryptographic scheme is treated, among others.

Next, we consider polynomial arithmetic, which inherits many algorithms from integer arithmetic. Kronecker's algorithm to factorize an integer-coefficient polynomial comes next. In the chapter about algebraic numbers, modular arithmetic is considered from a new point of view, as a result of which finite fields and resultants can be introduced. In the following chapters about polynomial factorization, modern (and much more efficient) algorithms are treated and implemented.

Chapters 9 to 12 build the basis of a more advanced lecture on computer algebra and are influenced by my own research interests. The selected content is very well suited for such an advanced course, since the algorithms treated in the first part are now applied to topics which should be known to the students from their previous studies and are also of interest in applications: simplification and normal forms, power series, summation formulas, and integration. Whereas power series are already treated in higher calculus and turn out to be necessary in Physics, Computer Science, and probability theory, summation formulas appear in many parts of Mathematics and applications. Definite integration is considered a difficult part of analysis, and it is not immediately clear that this problem can be treated with algebraic methods. As a model for algebraic integration, we shall treat the case of rational functions in detail.

Of course I could not include all relevant topics of modern computer algebra. For example, I decided to omit the theory of Gröbner bases although they play a very important role in modern computer algebra systems, in particular for the solution of non-linear systems of equations. In the meantime, Gröbner bases have surpassed the use of resultants for this purpose. However, in order to treat this topic thoroughly, one needs either higher algebra or the space of a book on its own. Therefore, I preferred to refer to existing literature, for example to the very well suited book by Cox, Little, and O'Shea [CLO1997]. Also, I would have liked to include the LLL algorithm by Lenstra, Lenstra, and Lovász [LLL1982] which has very important and interesting applications, but it had to be omitted due to time and space restrictions. Still, treating these two topics seems to be mandatory for a deep understanding of modern computer algebra.

All algorithms considered in this book are implemented and presented in the general-purpose computer algebra system *Mathematica*. Nevertheless, lecturers are absolutely not bound to this system, since the book's definitions and theorems are completely independent of a chosen computer algebra system. Consequently, all computer algebra sessions can be downloaded as worksheets in the three systems *Mathematica*, *Maple*, and *Maxima* from the book's internet page `www.computer-algebra.org`. *Maxima* is freely available and was used by the author when teaching the same course several times at AIMS Cameroon.[1] The three mentioned systems are general-purpose systems that treat computer algebra in a wide range and are therefore perfectly suitable for this book. They all have the feature that computations can be saved as sessions that can subsequently be reloaded. All instructions which are saved in such a notebook (or worksheet) can be easily computed by the system again after loading the worksheet, which allows the lecturer to use the worksheets for demonstration purposes. In my lectures, I always load the worksheets and let the system process one command line after the other in real-time with the help of a projector, so that the students can see how the system works and generates the computational results. While demonstrating the system's behavior, the lecturer should always present the background information that is needed for the correct and smart use of the system. This procedure gives the students the chance to directly follow the algorithms, to observe whether they are fast or slow, and to suggest their own examples. Furthermore, this scheme gives the students the opportunity to use the algorithms for testing purposes in the sense of experimental Mathematics.

For this reason, I found it important not to describe the algorithms—as in most other books—by so-called pseudo code, but in contrast as executable programs. All sessions considered in the

[1] The *African Institute of Mathematical Sciences*, founded in South Africa, now has branches in South Africa, Senegal, Ghana, Cameroon, Tanzania, and Rwanda, and it is still expanding.

framework of this book can be downloaded on www.computer-algebra.org as *Mathematica*, *Maple*, or *Maxima* code, which you can then run on your own computer with the respective computer algebra system. My feeling is that this should be quite useful for many readers. Concerning the exercise classes, in all three systems the students are able to write down their exercise solutions, containing both computations and texts, in respective notebooks (worksheets), which they can hand in electronically for grading. Afterwards, the tutor can return the corrected worksheets to the students.

In the first decade of the current century, I was the chairman of the German Fachgruppe Computeralgebra (www.fachgruppe-computeralgebra.de). In this position, I organized several meetings about *Computer Algebra in Education*, where I was often asked by high school teachers about literature on the essentials of computer algebra for the non-specialist. The current book gives an elementary introduction to the algorithmic foundation of computer algebra systems that is suitable for high school teachers. Since the book is rather elementary and contains a detailed index, it can also be used as a reference manual on algorithms of computer algebra.

As a reference for the first 9 chapters of the present book, I could resort to the books [Chi2000], [GCL1992] and [GG1999]. The book by Lindsay Childs, already written in 1979, is surely one of the first books about computer algebra algorithms, although its title *A Concrete Introduction to Higher Algebra* does not suggest this. Another very important reference is the book by Geddes, Czapor, and Labahn from 1992, a very careful and readable description of the capabilities of *Maple*. Finally, in 1999 the book by Joachim von zur Gathen and Jürgen Gerhard was published, which could well be called the computer algebra bible. However, because of its opulence and the requirement to be familiar with higher algebra, this book seems not directly suitable as an introductory lecture.

The German version of this book would not have been possible without the help of numerous colleagues, staff, and students who helped me with words and deeds, smoothed my text on several occasions, and corrected many small and large errors in my first draft. My special thanks goes to my research colleague Dr. Dieter Schmersau—who unfortunately passed away in 2012—with whom I had endless discussions about the whole material. Further suggestions for improvement have been given by Imran Hafeez, Dr. Peter Horn, Dr. Detlef Müller, Prof. Dr. Reinhard Oldenburg, Dr. Torsten Sprenger, and Jonas Wolf.

One of the typical problems when working with software is that every new release gives you a headache, since some things do not work as they did before. Therefore, when I decided in 2016 to translate my text into English, it was also clear that I had to adapt the computer algebra sessions to the current releases. For example, *Mathematica* had meanwhile decided to incorporate many packages into the core language, so that loading of packages is no longer necessary. Even worse, loading a package as described might even result in errors! Also, some language changes had to be taken care of. Consequently, please be aware that I cannot guarantee that all the sessions will work as described in future releases.

Last but not least, I would like to thank the University of Kassel for granting me three sabbaticals that I could use to write the book and its translation. Also, I would like to thank Clemens Heine from Springer Heidelberg, who was already my project partner for the German version and who now made the English edition possible, and Dr. Loretta Bartolini from Springer New York for providing excellent support during the publication process. Finally, I want to thank Tobias Schwaibold for his thorough copy editing prior to the publication of the English edition.

I should mention that I did not foresee how much work is involved in such a translation and how much time it would occupy when I signed my contract in 2017. Unfortunately, it turned out that the progress was very slow, since I had to manage many different activities in parallel, in particular teaching, research, hosting guests from all over the world (mostly from Africa), and being active in the executive committee of the *Deutsche Mathematiker-Vereinigung* (www.mathematik.de). Among other things, I organized the two workshops *Introduction to Computer Algebra and Applications* and *Introduction to Orthogonal Polynomials and Applications*, together with my Cameroonian colleague Mama Foupouagnigni, which took place in 2017 and 2018 and were generously funded by the *VolkswagenStiftung* (www.aims-volkswagen-workshops.org).

Now that the English translation is ready to be published, I would like to thank Alta Jooste from Pretoria, South Africa, very much, who read the English version extremely carefully and gave many valuable suggestions for improvement. This was of great help to me regarding the completion of the manuscript. Furthermore, I would like to thank one of my Master students, Thomas Izgin, who provided an interesting argument to simplify the proof of Theorem 12.9. My sincere thanks also go to my former PhD students Torsten Sprenger, Henning Seidler, and Kornelia Fischer who had found mistakes in the German Edition that could now be eliminated.

Kassel, April 20th, 2021 Wolfram Koepf

E-Mail: koepf@mathematik.uni-kassel.de
www.mathematik.uni-kassel.de/~koepf

Contents