# The Mathematica GuideBook

*for Numerics*

Michael Trott

# The Mathematica GuideBook
## *for Numerics*

With 1364 Illustrations

INCLUDES DVD

Springer

Michael Trott
Wolfram Research
Champaign, Illinois

Mathematica is a registered trademark of Wolfram Research, Inc.

Library of Congress Control Number: 2005928494

ISBN-10: 0-387-95011-7 Printed on acid-free paper.
ISBN-13: 978-0387-95011-2
e-ISBN 0-387-28814-7

Printed in the United States of America. (HAM)

9 8 7 6 5 4 3 2 1

springeronline.com

# Preface

*Bei mathematischen Operationen kann sogar eine gänzliche Entlastung des Kopfes eintreten, indem man einmal ausgeführte Zähloperationen mit Zeichen symbolisiert und, statt die Hirnfunktion auf Wiederholung schon ausgeführter Operationen zu verschwenden, sie für wichtigere Fälle aufspart.*

*When doing mathematics, instead of burdening the brain with the repetitive job of redoing numerical operations which have already been done before, it's possible to save that brainpower for more important situations by using symbols, instead, to represent those numerical calculations.*

— Ernst Mach (1883 ) [45]

## Computer Mathematics and Mathematica

Computers were initially developed to expedite numerical calculations. A newer, and in the long run, very fruitful field is the manipulation of symbolic expressions. When these symbolic expressions represent mathematical entities, this field is generally called computer algebra [8]. Computer algebra begins with relatively elementary operations, such as addition and multiplication of symbolic expressions, and includes such things as factorization of integers and polynomials, exact linear algebra, solution of systems of equations, and logical operations. It also includes analysis operations, such as definite and indefinite integration, the solution of linear and nonlinear ordinary and partial differential equations, series expansions, and residue calculations. Today, with computer algebra systems, it is possible to calculate in minutes or hours the results that would (and did) take years to accomplish by paper and pencil. One classic example is the calculation of the orbit of the moon, which took the French astronomer Delaunay 20 years [12], [13], [14], [15], [11], [26], [27], [53], [16], [17], [25]. (The *Mathematica GuideBooks* cover the two other historic examples of calculations that, at the end of the 19th century, took researchers many years of hand calculations [1], [4], [38] and literally thousands of pages of paper.)

Along with the ability to do symbolic calculations, four other ingredients of modern general-purpose computer algebra systems prove to be of critical importance for solving scientific problems:

- a powerful high-level programming language to formulate complicated problems
- programmable two- and three-dimensional graphics
- robust, adaptive numerical methods, including arbitrary precision and interval arithmetic
- the ability to numerically evaluate and symbolically deal with the classical orthogonal polynomials and special functions of mathematical physics.

The most widely used, complete, and advanced general-purpose computer algebra system is *Mathematica. Mathematica* provides a variety of capabilities such as graphics, numerics, symbolics, standardized interfaces to other programs, a complete electronic document-creation environment (including a full-fledged mathematical typesetting system), and a variety of import and export capabilities. Most of these ingredients are necessary to coherently and exhaustively solve problems and model processes occurring in the natural sciences [41], [58], [21], [39] and other fields using constructive mathematics, as well as to properly represent the results. Consequently, *Mathematica*'s main areas of application are presently in the natural sciences, engineering, pure and applied mathematics, economics, finance, computer graphics, and computer science.

*Mathematica* is an ideal environment for doing general scientific and engineering calculations, for investigating and solving many different mathematically expressable problems, for visualizing them, and for writing notes, reports, and papers about them. Thus, *Mathematica* is an integrated computing environment, meaning it is what is also called a "problem-solving environment" [40], [23], [6], [48], [43], [50], [52].

## Scope and Goals

*The Mathematica GuideBooks* are four independent books whose main focus is to show how to solve scientific problems with *Mathematica*. Each book addresses one of the four ingredients to solve nontrivial and real-life mathematically formulated problems: programming, graphics, numerics, and symbolics. The Programming and the Graphics volume were published in autumn 2004.

The four *Mathematica GuideBooks* discuss programming, two-dimensional, and three-dimensional graphics, numerics, and symbolics (including special functions). While the four books build on each other, each one is self-contained. Each book discusses the definition, use, and unique features of the corresponding *Mathematica* functions, gives small and large application examples with detailed references, and includes an extensive set of relevant exercises and solutions.

The *GuideBooks* have three primary goals:

- to give the reader a solid working knowledge of *Mathematica*
- to give the reader a detailed knowledge of key aspects of *Mathematica* needed to create the "best", fastest, shortest, and most elegant solutions to problems from the natural sciences
- to convince the reader that working with *Mathematica* can be a quite fruitful, enlightening, and joyful way of cooperation between a computer and a human.

Realizing these goals is achieved by understanding the unifying design and philosophy behind the *Mathematica* system through discussing and solving numerous example-type problems. While a variety of mathematics and physics problems are discussed, the *GuideBooks* are not mathematics or physics books (from the point of view of content and rigor; no proofs are typically involved), but rather the author builds on *Mathematica*'s mathematical and scientific knowledge to explore, solve, and visualize a variety of applied problems.

The focus on solving problems implies a focus on the computational engine of *Mathematica,* the kernel—rather than on the user interface of *Mathematica,* the front end. (Nevertheless, for a nicer presentation inside the electronic version, various front end features are used, but are not discussed in depth.)

The *Mathematica GuideBooks* go far beyond the scope of a pure introduction into *Mathematica*. The books also present instructive implementations, explanations, and examples that are, for the most part, original. The books also discuss some "classical" *Mathematica* implementations, explanations, and examples, partially available only in the original literature referenced or from newsgroups threads.

In addition to introducing *Mathematica*, the *GuideBooks* serve as a guide for generating fairly complicated graphics and for solving more advanced problems using graphical, numerical, and symbolical techniques in cooperative ways. The emphasis is on the *Mathematica* part of the solution, but the author employs examples that are not uninteresting from a content point of view. After studying the *GuideBooks*, the reader will be able to solve new and old scientific, engineering, and recreational mathematics problems faster and more completely with the help of *Mathematica*—at least, this is the author's goal. The author also hopes that the reader will enjoy using *Mathematica* for visualization of the results as much as the author does, as well as just studying *Mathematica* as a language on its own.

In the same way that computer algebra systems are not “proof machines” [46], [9], [37], [10], [54], [55], [56] such as might be used to establish the four-color theorem ([2], [22]), the Kepler [28], [19], [29], [30], [31], [32], [33], [34], [35], [36] or the Robbins ([44], [20]) conjectures, proving theorems is not the central theme of the *GuideBooks*. However, powerful and general proof machines [9], [42], [49], [24], [3], founded on *Mathematica*' s general programming paradigms and its mathematical capabilities, have been built (one such system is *Theorema* [7]). And, in the *GuideBooks*, we occasionally prove one theorem or another theorem.

In general, the author’s aim is to present a realistic portrait of *Mathematica*: its use, its usefulness, and its strengths, including some current weak points and sometimes unexpected, but often nevertheless quite “thought through”, behavior. *Mathematica* is not a universal tool to solve arbitrary problems which can be formulated mathematically—only a fraction of all mathematical problems can even be formulated in such a way to be efficiently expressed today in a way understandable to a computer. Rather, it is often necessary to do a certain amount of programming and occasionally give *Mathematica* some “help” instead of simply calling a single function like `Solve` to solve a system of equations. Because this will almost always be the case for “real-life” problems, we do not restrict ourselves only to “textbook” examples, where all goes smoothly without unexpected problems and obstacles. The reader will see that by employing *Mathematica’s* programming, numeric, symbolic, and graphic power, *Mathematica* can offer more effective, complete, straightforward, reusable, and less likely erroneous solution methods for calculations than paper and pencil, or numerical programming languages.

Although the *Guidebooks* are large books, it is nevertheless impossible to discuss all of the 2,000+ built-in *Mathematica* commands. So, some simple as well as some more complicated commands have been omitted. For a full overview about *Mathematica*’s capabilities, it is necessary to study *The Mathematica Book* [60] in detail. The commands discussed in the *Guidebooks* are those that an scientist or research engineer needs for solving *typical* problems, if such a thing exists [18]. These subjects include a quite detailed discussion of the structure of *Mathematica* expressions, *Mathematica* input and output (important for the human–*Mathematica* interaction), graphics, numerical calculations, and calculations from classical analysis. Also, emphasis is given to the powerful algebraic manipulation functions. Interestingly, they frequently allow one to solve analysis problems in an algorithmic way [5]. These functions are typically not so well known because they are not taught in classical engineering or physics-mathematics courses, but with the advance of computers doing symbolic mathematics, their importance increases [47].

A thorough knowledge of:

- machine and high-precision numbers, packed arrays, and intervals
- machine, high-precision, and interval arithmetic
- process of compilation, its advantages and limits
- main operations of numerical analysis, such as equation solving, minimization, summation
- numerical solution of ordinary and partial differential equations

is needed for virtually any nontrivial numeric calculation and frequently also in symbolic computations. *The Mathematica GuideBook for Numerics* discusses these subjects.

The current version of the *Mathematica GuideBooks* is tailored for *Mathematica* Version 5.1.

## Content Overview

*The Mathematica GuideBook for Numerics* has two chapters. Each chapter is subdivided into sections (which occasionally have subsections), exercises, solutions to the exercises, and references.

This volume deals with *Mathematica*'s numerical mathematics capabilities, the indispensable tools for dealing with virtually any "real life" problem. Fast machine, exact integer, and rational and verified high-precision arithmetic is applied to a large number of examples in the main text and in the solutions to the exercises.

Chapter 1 deals with numerical calculations, which are important for virtually all *Mathematica* users. This volume starts with calculations involving real and complex numbers with an "arbitrary" number of digits. (Well, not really an "arbitrary" number of digits, but on present-day computers, many calculations involving a few million digits are easily feasible.). Then follows a discussion of significance arithmetic, which automatically keeps track of the digits which are correct in calculations with high-precision numbers. Also discussed is the use of interval arithmetic. (Despite being slow, exact and/or inexact, interval arithmetic allows one to carry out validated numerical calculations.) The next important subject is the (pseudo)compilation of *Mathematica* code. Because *Mathematica* is an interpreted language that allows for "unforeseeable" actions and arbitrary side effects at runtime, it generally cannot be compiled. Strictly numerical calculations can, of course, be compiled.

Then, the main numerical functions are discussed: interpolation, Fourier transforms, numerical summation, and integration, solution of equations (root finding), minimization of functions, and the solution of ordinary and partial differential equations. To illustrate *Mathematica*'s differential equation solving capabilities, many ODEs and PDEs are discussed. Many medium-sized examples are given for the various numerical procedures. In addition, *Mathematica* is used to monitor and visualize various numerical algorithms.

The main part of Chapter 1 culminates with two larger applications, the construction of Riemann surfaces of algebraic functions and the visualization of electric and magnetic field lines of some more complicated two- and three-dimensional charge and current distributions. A large, diverse set of exercises and detailed solutions ends the first chapter.

Chapter 2 deals with exact integer calculations and integer-valued functions while concentrating on topics that are important in classical analysis. Number theory functions and modular polynomial functions, currently little used in most natural science applications, are intentionally given less detailed treatment. While some of the functions of this chapter have analytic continuations to complex arguments and could so be considered as belonging to Chapter 3 of the Symbolics volume, emphasis is given to their combinatorial use in this chapter.

This volume explains and demonstrates the use of the numerical functions of *Mathematica*. It only rarely discusses the underlying numerical algorithms themselves. But occasionally *Mathematica* is used to monitor how the algorithms work and progress.

## The Books and the Accompanying DVDs

Each of the *GuideBooks* comes with a multiplatform DVD. Each DVD contains the fourteen main notebooks, the hyperlinked table of contents and index, a navigation palette, and some utility notebooks and files. All notebooks are tailored for *Mathematica* 5.1. Each of the main notebooks corresponds to a chapter from the printed books. The notebooks have the look and feel of a printed book, containing structured units, typeset formulas, *Mathemat-*

*ica* code, and complete solutions to all exercises. The DVDs contain the fully evaluated notebooks corresponding to the chapters of the corresponding printed book (meaning these notebooks have text, inputs, outputs and graphics). The DVDs also include the unevaluated versions of the notebooks of the other three *GuideBooks* (meaning they contain all text and *Mathematica* code, but no outputs and graphics).

Although the *Mathematica GuideBooks* are printed, *Mathematica* is "a system for doing mathematics by computer" [59]. This was the lovely tagline of earlier versions of *Mathematica,* but because of its growing breadth (like data import, export and handling, operating system-independent file system operations, electronic publishing capabilities, web connectivity), nowadays *Mathematica* is called a "system for technical computing". The original tagline (that is more than ever valid today!) emphasized two points: doing mathematics and doing it on a computer. The approach and content of the *GuideBooks* are fully in the spirit of the original tagline: They are centered around *doing* mathematics. The second point of the tagline expresses that an electronic version of the *GuideBooks* is the more natural medium for *Mathematica*-related material. Long outputs returned by *Mathematica*, sequences of animations, thousands of web-retrievable references, a 10,000-entry hyperlinked index (that points more precisely than a printed index does) are space-consuming, and therefore not well suited for the printed book. As an interactive program, *Mathematica* is best learned, used, challenged, and enjoyed while sitting in front of a powerful computer (or by having a remote kernel connection to a powerful computer).

In addition to simply showing the printed book's text, the notebooks allow the reader to:

- experiment with, reuse, adapt, and extend functions and code
- investigate parameter dependencies
- annotate text, code, and formulas
- view graphics in color
- run animations.

## The Accompanying Web Site

Why does a printed book need a home page? There are (in addition to being just trendy) two reasons for a printed book to have its fingerprints on the web. The first is for (*Mathematica*) users who have not seen the book so far. Having an outline and content sample on the web is easily accomplished, and shows the look and feel of the notebooks (including some animations). This is something that a printed book actually cannot do. The second reason is for readers of the book: *Mathematica* is a large modern software system. As such, it ages quickly in the sense that in the timescale of $10^{1.\,smallInteger}$months, a new version will likely be available. The overwhelmingly large majority of *Mathematica* functions and programs will run unchanged in a new version. But occasionally, changes and adaptations might be needed. To accommodate this, the web site of this book—http://www.MathematicaGuideBooks.org—contains a list of changes relevant to the *GuideBooks*. In addition, like any larger software project, unavoidably, the *GuideBooks* will contain suboptimal implementations, mistakes, omissions, imperfections, and errors. As they come to his attention, the author will list them at the book's web site. Updates to references, corrections [51], hundreds of pages of additional exercises and solutions, improved code segments, and other relevant information will be on the web site as well. Also, information about OS-dependent and *Mathematica* version-related changes of the given *Mathematica* code will be available there.

## Evolution of the Mathematica GuideBooks

A few words about the history and the original purpose of the *GuideBooks*: They started from lecture notes of an *Introductory Course in Mathematica 2* and an advanced course on the *Efficient Use of the Mathematica Programming System*, given in 1991/1992 at the Technical University of Ilmenau, Germany. Since then, after each release of a new version of *Mathematica*, the material has been updated to incorporate additional functionality. This electronic/printed publication contains text, unique graphics, editable formulas, runable, and modifiable programs, all made possible by the electronic publishing capabilities of *Mathematica*. However, because the structure, functions and examples of the original lecture notes have been kept, an abbreviated form of the *GuideBooks* is still suitable for courses.

Since 1992 the manuscript has grown in size from 1,600 pages to more than three times its original length, finally "weighing in" at nearly 5,000 printed book pages with more than:

- 18 gigabytes of accompanying *Mathematica* notebooks
- 22,000 *Mathematica* inputs with more than 13,000 code comments
- 11,000 references
- 4,000 graphics
- 1,000 fully solved exercises
- 150 animations.

This first edition of this book is the result of more than eleven years of writing and daily work with *Mathematica*. In these years, *Mathematica* gained hundreds of functions with increased functionality and power. A modern year-2005 computer equipped with *Mathematica* represents a computational power available only a few years ago to a select number of people [57] and allows one to carry out recreational or new computations and visualizations—unlimited in nature, scope, and complexity— quickly and easily. Over the years, the author has learned a lot of *Mathematica* and its current and potential applications, and has had a lot of fun, enlightening moments and satisfaction applying *Mathematica* to a variety of research and recreational areas, especially graphics. The author hopes the reader will have a similar experience.

## Disclaimer

In addition to the usual disclaimer that neither the author nor the publisher guarantees the correctness of any formula, fitness, or reliability of any of the code pieces given in this book, another remark should be made. No guarantee is given that running the *Mathematica* code shown in the *GuideBooks* will give identical results to the printed ones. On the contrary, taking into account that *Mathematica* is a large and complicated software system which evolves with each released version, running the code with another version of *Mathematica* (or sometimes even on another operating system) will very likely result in different outputs for some inputs. And, as a consequence, if different outputs are generated early in a longer calculation, some functions might hang or return useless results.

The interpretations of *Mathematica* commands, their descriptions, and uses belong solely to the author. They are not claimed, supported, validated, or enforced by Wolfram Research. The reader will find that the author's view on *Mathematica* deviates sometimes considerably from those found in other books. The author's view is more on the formal than on the pragmatic side. The author does not hold the opinion that any *Mathematica* input has to have an immediate semantic meaning. *Mathematica* is an extremely rich system, especially from the language point of view. It is instructive, interesting, and fun to study the behavior of built-in *Mathematica* functions when called with a variety of arguments (like unevaluated, hold, including undercover zeros, etc.). It is the author's strong belief that doing this and being able to explain the observed behavior will be, in the long term, very fruitful for the reader because it develops the ability to recognize the uniformity of the principles underlying *Mathematica* and to make constructive, imaginative, and effective use of this uniformity. Also, some exercises ask the reader to investigate certain "unusual" inputs.

From time to time, the author makes use of undocumented features and/or functions from the `Developer`` and `Experimental`` contexts (in later versions of *Mathematica* these functions could exist in the `System`` context or could have different names). However, some such functions might no longer be supported or even exist in later versions of *Mathematica*.

## Acknowledgements

Over the decade, the *GuideBooks* were in development, many people have seen parts of them and suggested useful changes, additions, and edits. I would like to thank Horst Finsterbusch, Gottfried Teichmann, Klaus Voss, Udo Krause, Jerry Keiper, David Withoff, and Yu He for their critical examination of early versions of the manuscript and their useful suggestions, and Sabine Trott for the first proofreading of the German manuscript. I also want to thank the participants of the original lectures for many useful discussions. My thanks go to the reviewers of this book: John Novak, Alec Schramm, Paul Abbott, Jim Feagin, Richard Palmer, Ward Hanson, Stan Wagon, and Markus van Almsick, for their suggestions and ideas for improvement. I thank Richard Crandall, Allan Hayes, Andrzej Kozlowski, Hartmut Wolf, Stephan Leibbrandt, George Kambouroglou, Domenico Minunni, Eric Weisstein, Andy Shiekh, Arthur G. Hubbard, Jay Warrendorff, Allan Cortzen, Ed Pegg, and Udo Krause for comments on the prepublication version of the *GuideBooks*. I thank Bobby R. Treat, Arthur G. Hubbard, Murray Eisenberg, Marvin Schaefer, Marek Duszynski, Daniel Lichtblau, Devendra Kapadia, Adam Strzebonski, Anton Antonov, and Brett Champion for useful comments on the *Mathematica* Version 5.1 tailored version of the *GuideBooks*.

My thanks are due to Gerhard Gobsch of the Institute for Physics of the Technical University in Ilmenau for the opportunity to develop and give these original lectures at the Institute, and to Stephen Wolfram who encouraged and supported me on this project.

Concerning the process of making the *Mathematica GuideBooks* from a set of lecture notes, I thank Glenn Scholebo for transforming notebooks to $T_EX$ files, and Joe Kaiping for $T_EX$ work related to the printed book. I thank John Novak and Jan Progen for putting all the material into good English style and grammar, John Bonadies for the chapter-opener graphics of the book, and Jean Buck for library work. I especially thank John Novak for the creation of *Mathematica* 3 notebooks from the $T_EX$ files, and Andre Kuzniarek for his work on the stylesheet to give the notebooks a pleasing appearance. My thanks go to Andy Hunt who created a specialized stylesheet for the actual book printout and printed and formatted the 4×1000+ pages of the *Mathematica GuideBooks*. I thank Andy Hunt for making a first version of the homepage of the *GuideBooks* and Amy Young for creating the current version of the homepage of the *GuideBooks*. I thank Sophie Young for a final check of the English. My largest thanks go to Amy Young, who encouraged me to update the whole book over the years and who had a close look at all of my English writing and often improved it considerably. Despite reviews by many individuals any remaining mistakes or omissions, in the *Mathematica* code, in the mathematics, in the description of the *Mathematica* functions, in the English, or in the references, etc. are, of course, solely mine.

Let me take the opportunity to thank members of the Research and Development team of Wolfram Research whom I have met throughout the years, especially Victor Adamchik, Anton Antonov, Alexei Bocharov, Arnoud Buzing, Brett Champion, Matthew Cook, Todd Gayley, Darren Glosemeyer, Roger Germundsson, Unal Goktas, Yifan Hu, Devendra Kapadia, Zbigniew Leyk, David Librik, Daniel Lichtblau, Jerry Keiper, Robert Knapp, Roman Mäder, Oleg Marichev, John Novak, Peter Overmann, Oleksandr Pavlyk, Ulises Cervantes–Pimentel, Mark Sofroniou, Adam Strzebonski, Oyvind Tafjord, Robby Villegas, Tom Wickham–Jones, David Withoff, and Stephen Wolfram for numerous discussions about design principles, various small details, underlying algorithms, efficient implementation of various procedures, and tricks concerning *Mathematica*. The appearance of the notebooks profited from discussions with John Fultz, Paul Hinton, John Novak, Lou D'Andria, Theodore Gray, Andre Kuzniarek, Jason Harris, Andy Hunt, Christopher Carlson, Robert Raguet–Schofield, George Beck, Kai Xin, Chris Hill, and Neil Soiffer about front end, button, and typesetting issues.

I'm grateful to Jeremy Hilton from the Corporation for National Research Initiatives for allowing the use of the text of Shakespeare's *Hamlet* (to be used in Chapter 1 of *The Mathematica GuideBook to Numerics*).

It was an interesting and unique experience to work over the last 12 years with five editors: Allan Wylde, Paul Wellin, Maria Taylor, Wayne Yuhasz, and Ann Kostant, with whom the *GuideBooks* were finally published. Many book-related discussions that ultimately improved the *GuideBooks*, have been carried out with Jan Benes from TELOS and associates, Steven Pisano, Jenny Wolkowicki, Henry Krell, Fred Bartlett, Vaishali Damle, Ken Quinn, Jerry Lyons, and Rüdiger Gebauer from Springer New York.

The author hopes the *Mathematica GuideBooks* help the reader to discover, investigate, urbanize, and enjoy the computational paradise offered by *Mathematica*.

Wolfram Research, Inc. Michael Trott
April 2005

## References

1 A. Amthor. *Z. Math. Phys.* 25, 153 (1880).

2 K. Appel, W. Haken. *J. Math.* 21, 429 (1977).

3 A. Bauer, E. Clarke, X. Zhao. *J. Automat. Reasoning* 21, 295 (1998).

4 A. H. Bell. *Am. Math. Monthly* 2, 140 (1895).

5 M. Berz. *Adv. Imaging Electron Phys.* 108, 1 (2000).

6 R. F. Boisvert. *arXiv:cs.MS*/0004004 (2000).

7 B. Buchberger. *Theorema Project* (1997). ftp://ftp.risc.uni-linz.ac.at/pub/techreports/1997/97-34/ed-media.nb

8 B. Buchberger. *SIGSAM Bull.* 36, 3 (2002).

9 S.-C. Chou, X.-S. Gao, J.-Z. Zhang. *Machine Proofs in Geometry*, World Scientific, Singapore, 1994.

10 A. M. Cohen. *Nieuw Archief Wiskunde* 14, 45 (1996).

11 A. Cook. *The Motion of the Moon*, Adam-Hilger, Bristol, 1988.

12 C. Delaunay. *Théorie du Mouvement de la Lune*, Gauthier-Villars, Paris, 1860.

13 C. Delaunay. *Mem. de l' Acad. des Sc. Paris* 28 (1860).

14 C. Delaunay. *Mem. de l' Acad. des Sc. Paris* 29 (1867).

15 A. Deprit, J. Henrard, A. Rom. *Astron. J.* 75, 747 (1970).

16 A. Deprit. *Science* 168, 1569 (1970).

17 A. Deprit, J. Henrard, A. Rom. *Astron. J.* 76, 273 (1971).

18 P. J. Dolan, Jr., D. S. Melichian. *Am. J. Phys.* 66, 11 (1998).

19 S. P. Ferguson, T. C. Hales. *arXiv:math.MG/* 9811072 (1998).

20 B. Fitelson. *Mathematica Educ. Res.* 7, n1, 17 (1998).

21 A. C. Fowler. *Mathematical Models in the Applied Sciences,* Cambridge University Press, Cambridge, 1997.

22 H. Fritsch, G. Fritsch. *The Four-Color Theorem*, Springer-Verlag, New York, 1998.

23 E. Gallopoulus, E. Houstis, J. R. Rice (eds.). *Future Research Directions in Problem Solving Environments for Computational Science: Report of a Workshop on Research Directions in Integrating Numerical Analysis, Symbolic Computing, Computational Geometry, and Artificial Intelligence for Computational Science*, 1991. http://www.cs.purdue.edu/research/cse/publications/tr/92/92-032.ps.gz

24 V. Gerdt, S. A. Gogilidze in V. G. Ganzha, E. W. Mayr, E. V. Vorozhtsov (eds.). *Computer Algebra in Scientific Computing*, Springer-Verlag, Berlin, 1999.

25 M. C. Gutzwiller, D. S. Schmidt. *Astronomical Papers: The Motion of the Moon as Computed by the Method of Hill, Brown, and Eckert*, U.S. Government Printing Office, Washington, 1986.

26 M. C. Gutzwiller. *Rev. Mod. Phys.* 70, 589 (1998).

27 Y. Hagihara. *Celestial Mechanics* vII/1, MIT Press, Cambridge, 1972.

28 T. C. Hales. *arXiv:math.MG/* 9811071 (1998).

29 T. C. Hales. *arXiv:math.MG/* 9811073 (1998).

30 T. C. Hales. *arXiv:math.MG/* 9811074 (1998).

31 T. C. Hales. *arXiv:math.MG/* 9811075 (1998).

32 T. C. Hales. *arXiv:math.MG/* 9811076 (1998).

33 T. C. Hales. *arXiv:math.MG/* 9811077 (1998).

34 T. C. Hales. *arXiv:math.MG/* 9811078 (1998).

35 T. C. Hales. *arXiv:math.MG*/0205208 (2002).

36 T. C. Hales in L. Tatsien (ed.). *Proceedings of the International Congress of Mathematicians* v. 3, Higher Education Press, Beijing, 2002.

37 J. Harrison. *Theorem Proving with the Real Numbers*, Springer-Verlag, London, 1998.

38 J. Hermes. *Nachrichten Königl. Gesell. Wiss. Göttingen* 170 (1894).

39 E. N. Houstis, J. R. Rice, E. Gallopoulos, R. Bramley (eds.). *Enabling Technologies for Computational Science*, Kluwer, Boston, 2000.

40 E. N. Houstis, J. R. Rice. *Math. Comput. Simul.* 54, 243 (2000).

41 M. S. Klamkin (eds.). *Mathematical Modelling*, SIAM, Philadelphia, 1996.

42 H. Koch, A. Schenkel, P. Wittwer. *SIAM Rev.* 38, 565 (1996).

43 Y. N. Lakshman, B. Char, J. Johnson in O. Gloor (ed.). *ISSAC 1998*, ACM Press, New York, 1998.

44 W. McCune. *Robbins Algebras Are Boolean*, 1997. http://www.mcs.anl.gov/home/mccune/ar/robbins/

45 E. Mach (R. Wahsner, H.-H. von Borszeskowski eds.). *Die Mechanik in ihrer Entwicklung*, Akademie-Verlag, Berlin, 1988.

46 D. A. MacKenzie. *Mechanizing Proof: Computing, Risk, and Trust*, MIT Press, Cambridge, 2001.

47 B. M. McCoy. *arXiv:cond-mat*/0012193 (2000).

48 K. J. M. Moriarty, G. Murdeshwar, S. Sanielevici. *Comput. Phys. Commun.* 77, 325 (1993).

49 I. Nemes, M. Petkovšek, H. S. Wilf, D. Zeilberger. *Am. Math. Monthly* 104, 505 (1997).

50 W. H. Press, S. A. Teukolsky. *Comput. Phys.* 11, 417 (1997).

51 D. Rawlings. *Am. Math. Monthly* 108, 713 (2001).

52 *Problem Solving Environments Home Page.* http://www.cs.purdue.edu/research/cse/pses

53 D. S. Schmidt in H. S. Dumas, K. R. Meyer, D. S. Schmidt (eds.). *Hamiltonian Dynamical Systems*, Springer-Verlag, New York, 1995.

54 S. Seiden. *SIGACT News* 32, 111 (2001).

55 S. Seiden. *Theor. Comput. Sc.* 282, 381 (2002).

56 C. Simpson. *arXiv:math.HO*/0311260 (2003).

57 A. M. Stoneham. *Phil. Trans. R. Soc. Lond.* A 360, 1107 (2002).

58 M. Tegmark. *Ann. Phys.* 270, 1 (1999).

59 S. Wolfram. *Mathematica: A System for Doing Mathematics by Computer*, Addison-Wesley, Redwood City, 1992.

60 S. Wolfram. *The Mathematica Book*, Wolfram Media, Champaign, 2003.

# Contents

# Introduction and Orientation to *The Mathematica GuideBooks*

## *0.1 Overview*

### 0.1.1 Content Summaries

The *Mathematica GuideBooks* are published as four independent books: *The Mathematica GuideBook to Programming, The Mathematica GuideBook to Graphics, The Mathematica GuideBook to Numerics*, and *The Mathematica GuideBook to Symbolics.*

■ The Programming volume deals with the structure of *Mathematica* expressions and with *Mathematica* as a programming language. This volume includes the discussion of the hierarchical construction of all *Mathematica* objects out of symbolic expressions (all of the form *head*[*argument*]), the ultimate building blocks of expressions (numbers, symbols, and strings), the definition of functions, the application of rules, the recognition of patterns and their efficient application, the order of evaluation, program flows and program structure, the manipulation of lists (the universal container for *Mathematica* expressions of all kinds), as well as a number of topics specific to the *Mathematica* programming language. Various programming styles, especially *Mathematica*' s powerful functional programming constructs, are covered in detail.

■ The Graphics volume deals with *Mathematica*'s two-dimensional (2D) and three-dimensional (3D) graphics. The chapters of this volume give a detailed treatment on how to create images from graphics primitives, such as points, lines, and polygons. This volume also covers graphically displaying functions given either analytically or in discrete form. A number of images from the *Mathematica* Graphics Gallery are also reconstructed. Also discussed is the generation of pleasing scientific visualizations of functions, formulas, and algorithms. A variety of such examples are given.

■ The Numerics volume deals with *Mathematica*'s numerical mathematics capabilities—the indispensable sledgehammer tools for dealing with virtually any "real life" problem. The arithmetic types (fast machine, exact integer and rational, verified high-precision, and interval arithmetic) are carefully analyzed. Fundamental numerical operations, such as compilation of programs, numerical Fourier transforms, minimization, numerical solution of equations, and ordinary/partial differential equations are analyzed in detail and are applied to a large number of examples in the main text and in the solutions to the exercises.

■ The Symbolics volume deals with *Mathematica*'s symbolic mathematical capabilities—the real heart of *Mathematica* and the ingredient of the *Mathematica* software system that makes it so unique and powerful. Structural and mathematical operations on systems of polynomials are fundamental to many symbolic calculations and are covered in detail. The solution of equations and differential equations, as well as the classical calculus operations, are exhaustively treated. In addition, this volume discusses and employs the classical orthogonal polynomials and special functions of mathematical physics. To demonstrate the symbolic mathematics power, a variety of problems from mathematics and physics are discussed.

The four *GuideBooks* contain about 25,000 *Mathematica* inputs, representing more than 75,000 lines of commented *Mathematica* code. (For the reader already familiar with *Mathematica*, here is a more precise measure: The `LeafCount` of all inputs would be about 900,000 when collected in a list.) The *GuideBooks* also have more than 4,000 graphics, 150 animations, 11,000 references, and 1,000 exercises. More than 10,000 hyperlinked index entries and hundreds of hyperlinks from the overview sections connect all parts in a convenient way. The evaluated notebooks of all four volumes have a cumulative file size of about 20 GB. Although these numbers may sound large, the *Mathematica GuideBooks* actually cover only a portion of *Mathematica*'s functionality and features and give only a glimpse into the possibilities *Mathematica* offers to generate graphics, solve problems, model systems, and discover new identities, relations, and algorithms. The *Mathematica* code is explained in detail throughout all chapters. More than 13,000 comments are scattered throughout all inputs and code fragments.

## 0.1.2 Relation of the Four Volumes

The four volumes of the *GuideBooks* are basically independent, in the sense that readers familiar with *Mathematica* programming can read any of the other three volumes. But a solid working knowledge of the main topics discussed in *The Mathematica GuideBook to Programming*—symbolic expressions, pure functions, rules and replacements, and list manipulations—is required for the Graphics, Numerics, and Symbolics volumes. Compared to these three volumes, the Programming volume might appear to be a bit "dry". But similar to learning a foreign language, before being rewarded with the beauty of novels or a poem, one has to sweat and study. The whole suite of graphical capabilities and all of the mathematical knowledge in *Mathematica* are accessed and applied through lists, patterns, rules, and pure functions, the material discussed in the Programming volume.

Naturally, graphics are the center of attention of the *The Mathematica GuideBook to Graphics*. While in the Programming volume some plotting and graphics for visualization are used, graphics are not crucial for the Programming volume. The reader can safely skip the corresponding inputs to follow the main programming threads. The Numerics and Symbolics volumes, on the other hand, make heavy use of the graphics knowledge acquired in the Graphics volume. Hence, the prerequisites for the Numerics and Symbolics volumes are a good knowledge of *Mathematica*'*s* programming language and of its graphics system.

The Programming volume contains only a few percent of all graphics, the Graphics volume contains about two-thirds, and the Numerics and Symbolics volume, about one-third of the overall 4,000+ graphics. The Programming and Graphics volumes use some mathematical commands, but they restrict the use to a relatively small number (especially `Expand`, `Factor`, `Integrate`, `Solve`). And the use of the function `N` for numerical ization is unavoidable for virtually any "real life" application of *Mathematica*. The last functions allow us to treat some mathematically not uninteresting examples in the Programming and Graphics volumes. In addition to putting these functions to work for nontrivial problems, a detailed discussion of the mathematics functions of *Mathematica* takes place exclusively in the Numerics and Symbolics volumes.

The Programming and Graphics volumes contain a moderate amount of mathematics in the examples and exercises, and focus on programming and graphics issues. The Numerics and Symbolics volumes contain a substantially larger amount of mathematics.

Although printed as four books, the fourteen individual chapters (six in the Programming volume, three in the Graphics volume, two in the Numerics volume, and three in the Symbolics volume) of the *Mathematica GuideBooks* form one organic whole, and the author recommends a strictly sequential reading, starting from Chapter 1 of the Programming volume and ending with Chapter 3 of the Symbolics volume for gaining the maximum benefit. The electronic component of each book contains the text and inputs from all the four *GuideBooks*, together with a comprehensive hyperlinked index. The four volumes refer frequently to one another.

## 0.1.3 Chapter Structure

A rough outline of the content of a chapter is the following:

■ The main body discusses the *Mathematica* functions belonging to the chapter subject, as well their options and attributes. Generically, the author has attempted to introduce the functions in a "natural order". But surely, one cannot be axiomatic with respect to the order. (Such an order of the functions is not unique, and the author intentionally has "spread out" the introduction of various *Mathematica* functions across the four volumes.) With the introduction of a function, some small examples of how to use the functions and comparisons of this function with related ones are given. These examples typically (with the exception of some visualizations in the Programming volume) incorporate functions already discussed. The last section of a chapter often gives a larger example that makes heavy use of the functions discussed in the chapter.

■ A programmatically constructed overview of each chapter functions follows. The functions listed in this section are hyperlinked to their attributes and options, as well as to the corresponding reference guide entries of *The Mathematica Book.*

■ A set of exercises and potential solutions follow. Because learning *Mathematica* through examples is very efficient, the proposed solutions are quite detailed and form up to 50% of the material of a chapter.

■ References end the chapter.

Note that the first few chapters of the Programming volume deviate slightly from this structure. Chapter 1 of the Programming volume gives a general overview of the kind of problems dealt with in the four *GuideBooks.* The second, third, and fourth chapters of the Programming volume introduce the basics of programming in *Mathematica*. Starting with Chapters 5 of the Programming volume and throughout the Graphics, Numerics, and Symbolics volumes, the above-described structure applies.

In the 14 chapters of the *GuideBooks* the author has chosen a "we" style for the discussions of how to proceed in constructing programs and carrying out calculations to include the reader intimately.

## 0.1.4 Code Presentation Style

The typical style of a unit of the main part of a chapter is: Define a new function, discuss its arguments, options, and attributes, and then give examples of its usage. The examples are virtually always *Mathematica* inputs and outputs. The majority of inputs is in `InputForm` are the notebooks. On occasion `StandardForm` is also used. Although `StandardForm` mimics classical mathematics notation and makes short inputs more readable, for "program-like" inputs, `InputForm` is typically more readable and easier and more natural to align. For the outputs, `StandardForm` is used by default and occasionally the author has resorted to `InputForm` or `FullForm` to expose digits of numbers and to `TraditionalForm` for some formulas. Outputs are mostly not programs, but nearly always "results" (often mathematical expressions, formulas, identities, or lists of numbers rather than program constructs). The world of *Mathematica* users is divided into three groups, and each of them has a nearly religious opinion on how to format *Mathematica* code [1], [2]. The author follows the `InputForm` cult(ure) and hopes that the *Mathematica* users who do everything in either `StandardForm` or `TraditionalForm` will bear with him. If the reader really wants to see all code in either `StandardForm` or `TraditionalForm`, this can easily be done with the **Convert To** item from the **Cell** menu. (Note that the relation between `InputForm` and `StandardForm` is not symmetric. The `InputForm` cells of this book have been line-broken and aligned by hand. Transforming them into `StandardForm` or `TraditionalForm` cells works well because one typically does not line-break manually and align *Mathematica* code in these cell types. But converting `StandardForm` or `TraditionalForm` cells into `InputForm` cells results in much less pleasing results.)

In the inputs, special typeset symbols for *Mathematica* functions are typically avoided because they are not monospaced. But the author does occasionally compromise and use Greek, script, Gothic, and doublestruck characters.

In a book about a programming language, two other issues come always up: indentation and placement of the code.

■ The code of the *GuideBooks* is largely consistently formatted and indented. There are no strict guidelines or even rules on how to format and indent *Mathematica* code. The author hopes the reader will find the book's formatting style readable. It is a compromise between readability (mental parsabililty) and space conservation, so that the printed version of the *Mathematica GuideBook* matches closely the electronic version.

■ Because of the large number of examples, a rather imposing amount of *Mathematica* code is presented. Should this code be present only on the disk, or also in the printed book? If it is in the printed book, should it be at the position where the code is used or at the end of the book in an appendix? Many authors of *Mathematica* articles and books have strong opinions on this subject. Because the main emphasis of the *Mathematica GuideBooks* is on *solving* problems *with Mathematica* and not on the actual problems, the *GuideBooks* give all of the code at the point where it is needed in the printed book, rather than "hiding" it in packages and appendices. In addition to being more straightforward to read and conveniently allowing us to refer to elements of the code pieces, this placement makes the correspondence between the printed book and the notebooks close to 1:1, and so working back and forth between the printed book and the notebooks is as straightforward as possible.

## 0.2 Requirements

### 0.2.1 Hardware and Software

Throughout the *GuideBooks*, it is assumed that the reader has access to a computer running a current version of *Mathematica* (version 5.0/5.1 or newer). For readers without access to a licensed copy of *Mathematica*, it is possible to view all of the material on the disk using a trial version of *Mathematica*. (A trial version is downloadable from http://www.wolfram.com/products/mathematica/trial.cgi.)

The files of the *GuideBooks* are relatively large, altogether more than 20 GB. This is also the amount of hard disk space needed to store uncompressed versions of the notebooks. To view the notebooks comfortably, the reader's computer needs 128 MB RAM; to evaluate the evaluation units of the notebooks 1 GB RAM or more is recommended.

In the *GuideBooks*, a large number of animations are generated. Although they need more memory than single pictures, they are easy to create, to animate, and to store on typical year-2005 hardware, and they provide a lot of joy.

## 0.2.2 Reader Prerequisites

Although prior *Mathematica* knowledge is not needed to read *The Mathematica GuideBook to Programming*, it is assumed that the reader is familiar with basic actions in the *Mathematica* front end, including entering Greek characters using the keyboard, copying and pasting cells, and so on. Freely available tutorials on these (and other) subjects can be found at http://library.wolfram.com.

For a complete understanding of most of the *GuideBooks* examples, it is desirable to have a background in mathematics, science, or engineering at about the bachelor's level or above. Familiarity with mechanics and electrodynamics is assumed. Some examples and exercises are more specialized, for instance, from quantum mechanics, finite element analysis, statistical mechanics, solid state physics, number theory, and other areas. But the *GuideBooks* avoid very advanced (but tempting) topics such as renormalization groups [6], parquet approximations [27], and modular moonshines [14]. (Although *Mathematica* can deal with such topics, they do not fit the character of the *Mathematica GuideBooks* but rather the one of a *Mathematica Topographical Atlas* [a monumental work to be carried out by the *Mathematica*–Bourbakians of the 21st century]).

Each scientific application discussed has a set of references. The references should easily give the reader both an overview of the subject and pointers to further references.

## 0.3 What the GuideBooks Are and What They Are Not

### 0.3.1 Doing Computer Mathematics

As discussed in the Preface, the main goal of the *GuideBooks* is to demonstrate, showcase, teach, and exemplify scientific problem solving with *Mathematica*. An important step in achieving this goal is the discussion of *Mathematica* functions that allow readers to become fluent in programming when creating complicated graphics or solving scientific problems. This again means that the reader must become familiar with the most important programming, graphics, numerics, and symbolics functions, their arguments, options, attributes, and a few of their time and space complexities. And the reader must know which functions to use in each situation.

The *GuideBooks* treat only aspects of *Mathematica* that are ultimately related to "doing mathematics". This means that the *GuideBooks* focus on the functionalities of the kernel rather than on those of the front end. The knowledge required to use the front end to work with the notebooks can easily be gained by reading the corresponding chapters of the online documentation of *Mathematica*. Some of the subjects that are treated either lightly or not at all in the *GuideBooks* include the basic use of *Mathematica* (starting the program, features, and special properties of the notebook front end [16]), typesetting, the preparation of packages, external file operations, the communication of *Mathematica* with other programs via *MathLink*, special formatting and string manipulations, computer- and operating system-specific operations, audio generation, and commands available in various packages. "Packages" includes both, those distributed with *Mathematica* as well as those available from the *Mathematica* Information Center (http://library.wolfram.com/infocenter) and commercial sources, such as MathTensor for doing general relativity calculations (http://smc.vnet.net/MathTensor.html) or FeynCalc for doing high-energy physics calculations (http://www.feyncalc.org). This means, in particular, that probability and statistical calculations are barely touched on because most of the relevant commands are contained in the packages. The *GuideBooks* make little or no mention of the machine-dependent possibilities offered by the various *Mathematica* implementations. For this information, see the *Mathematica* documentation.

Mathematical and physical remarks introduce certain subjects and formulas to make the associated *Mathematica* implementations easier to understand. These remarks are not meant to provide a deep understanding of the (sometimes complicated) physical model or underlying mathematics; some of these remarks intentionally oversimplify matters.

The reader should examine all *Mathematica* inputs and outputs carefully. Sometimes, the inputs and outputs illustrate little-known or seldom-used aspects of *Mathematica* commands. Moreover, for the efficient use of *Mathematica*, it is very important to understand the possibilities and limits of the built-in commands. Many commands in *Mathematica* allow different numbers of arguments. When a given command is called with fewer than the maximum number of arguments, an internal (or user-defined) default value is used for the missing arguments. For most of the commands, the maximum number of arguments and default values are discussed.

When solving problems, the *GuideBooks* generically use a “straightforward” approach. This means they are not using particularly clever tricks to solve problems, but rather direct, possibly computationally more expensive, approaches. (From time to time, the *GuideBooks* even make use of a “brute force” approach.) The motivation is that when solving new “real life” problems a reader encounters in daily work, the “right mathematical trick” is seldom at hand. Nevertheless, the reader can more often than not rely on *Mathematica* being powerful enough to often succeed in using a straightforward approach. But attention is paid to *Mathematica*-specific issues to find time- and memory-efficient implementations—something that should be taken into account for any larger program.

As already mentioned, all larger pieces of code in this book have comments explaining the individual steps carried out in the calculations. Many smaller pieces of code have comments when needed to expedite the understanding of how they work. This enables the reader to easily change and adapt the code pieces. Sometimes, when the translation from traditional mathematics into *Mathematica* is trivial, or when the author wants to emphasize certain aspects of the code, we let the code “speak for itself”. While paying attention to efficiency, the *GuideBooks* only occasionally go into the computational complexity ([8], [40], and [7]) of the given implementations. The implementation of very large, complicated suites of algorithms is not the purpose of the *GuideBooks*. The *Mathematica* packages included with *Mathematica* and the ones at *MathSource* (http://library.wolfram.com/database/MathSource) offer a rich variety of self-study material on building large programs. Most general guidelines for writing code for scientific calculations (like descriptive variable names and modularity of code; see, e.g., [19] for a review) apply also to *Mathematica* programs.

The programs given in a chapter typically make use of *Mathematica* functions discussed in earlier chapters. Using commands from later chapters would sometimes allow for more efficient techniques. Also, these programs emphasize the use of commands from the current chapter. So, for example, instead of list operation, from a complexity point of view, hashing techniques or tailored data structures might be preferable. All subsections and sections are “self-contained” (meaning that no other code than the one presented is needed to evaluate the subsections and sections). The price for this “self-containedness” is that from time to time some code has to be repeated (such as manipulating polygons or forming random permutations of lists) instead of delegating such programming constructs to a package. Because this repetition could be construed as boring, the author typically uses a slightly different implementation to achieve the same goal.

## 0.3.2 Programming Paradigms

In the *GuideBooks,* the author wants to show the reader that *Mathematica* supports various programming paradigms and also show that, depending on the problem under consideration and the goal (e.g., solution of a problem, test of an algorithm, development of a program), each style has its advantages and disadvantages. (For a general discussion concerning programming styles, see [3], [41], [23], [32], [15], and [9].) *Mathematica* supports a functional programming style. Thus, in addition to classical procedural programs (which are often less efficient and less elegant), programs using the functional style are also presented. In the first volume of the *Mathematica GuideBooks*, the programming style is usually dictated by the types of commands that have been discussed up to that point. A certain portion of the programs involve recursive, rule-based programming. The choice of programming style is, of course, partially (ultimately) a matter of personal preference. The *GuideBooks*' main aim is to explain the operation, limits, and efficient application of the various *Mathematica* commands. For certain commands, this dictates a certain style of programming. However, the various programming styles, with their advantages and disadvantages, are not the main concern of the *GuideBooks*. In working with *Mathematica*, the reader is likely to use different programming styles depending if one wants a quick one-time calculation or a routine that will be used repeatedly. So, for a given implementation, the program structure may not always be the most elegant, fastest, or "prettiest".

The *GuideBooks* are not a substitute for the study of *The Mathematica Book* [45] http://documents.wolfram.com/mathematica). It is impossible to acquire a deeper (full) understanding of *Mathematica* without a *thorough* study of this book (reading it twice from the first to the last page is highly recommended). It *defines* the language and the spirit of *Mathematica*. The reader will probably from time to time need to refer to parts of it, because not all commands are discussed in the *GuideBooks*. However, the story of what can be done with *Mathematica* does not end with the examples shown in *The Mathematica Book*. The *Mathematica GuideBooks* go beyond *The Mathematica Book*. They present larger programs for solving various problems and creating complicated graphics. In addition, the *GuideBooks* discuss a number of commands that are not or are only fleetingly mentioned in the manual (e.g., some specialized methods of mathematical functions and functions from the `Developer`` and `Experimental`` contexts), but which the author deems important. In the notebooks, the author gives special emphasis to discussions, remarks, and applications relating to several commands that are typical for *Mathematica* but not for most other programming languages, e.g., `Map`, `MapAt`, `MapIndexed`, `Distribute`, `Apply`, `Replace`, `ReplaceAll`, `Inner`, `Outer`, `Fold`, `Nest`, `NestList`, `FixedPoint`, `FixedPointList`, and `Function`. These commands allow to write exceptionally elegant, fast, and powerful programs. All of these commands are discussed in *The Mathematica Book* and others that deal with programming in *Mathematica* (e.g., [33], [34], and [42]). However, the author's experience suggests that a deeper understanding of these commands and their optimal applications comes only after working with *Mathematica* in the solution of more complicated problems.

Both the printed book and the electronic component contain material that is meant to teach in detail how to use *Mathematica* to solve problems, rather than to present the underlying details of the various scientific examples. It cannot be overemphasized that to master the use of *Mathematica,* its programming paradigms and individual functions, the reader must experiment; this is especially important, insightful, easily verifiable, and satisfying with graphics, which involve manipulating expressions, making small changes, and finding different approaches. Because the results can easily be visually checked, generating and modifying graphics is an ideal method to learn programming in *Mathematica*.

## 0.4 Exercises and Solutions

### 0.4.1 Exercises

Each chapter includes a set of exercises and a detailed solution proposal for each exercise. When possible, all of the purely *Mathematica*-programming related exercises (these are most of the exercises of the Programming volume) should be solved by every reader. The exercises coming from mathematics, physics, and engineering should be solved according to the reader's interest. The most important *Mathematica* functions needed to solve a given problem are generally those of the associated chapter.

For a rough orientation about the content of an exercise, the subject is included in its title. The relative degree of difficulty is indicated by level superscript of the exercise number ($^{L1}$ indicates easy, $^{L2}$ indicates medium, and $^{L3}$ indicates difficult). The author's aim was to present understandable interesting examples that illustrate the *Mathematica* material discussed in the corresponding chapter. Some exercises were inspired by recent research problems; the references given allow the interested reader to dig deeper into the subject.

The exercises are intentionally not hyperlinked to the corresponding solution. The independent solving of the exercises is an important part of learning *Mathematica*.

### 0.4.2 Solutions

The *GuideBooks* contain solutions to each of the more than 1,000 exercises. Many of the techniques used in the solutions are not just one-line calls to built-in functions. It might well be that with further enhancements, a future version of *Mathematica* might be able to solve the problem more directly. (But due to different forms of some results returned by *Mathematica*, some problems might also become more challenging.) The author encourages the reader to try to find shorter, more clever, faster (in terms of runtime as well complexity), more general, and more elegant solutions. *Doing* various calculations is the most effective way to learn *Mathematica*. A proper *Mathematica* implementation of a function that solves a given problem often contains many different elements. The function(s) should have sensibly named and sensibly behaving options; for various (machine numeric, high-precision numeric, symbolic) inputs different steps might be required; shielding against inappropriate input might be needed; different parameter values might require different solution strategies and algorithms, helpful error and warning messages should be available. The returned data structure should be intuitive and easy to reuse; to achieve a good computational complexity, nontrivial data structures might be needed, etc. Most of the solutions do not deal with all of these issues, but only with selected ones and thereby leave plenty of room for more detailed treatments; as far as limit, boundary, and degenerate cases are concerned, they represent an outline of how to tackle the problem. Although the solutions do their job in general, they often allow considerable refinement and extension by the reader.

The reader should consider the given solution to a given exercise as a proposal; quite different approaches are often possible and sometimes even more efficient. The routines presented in the solutions are not the most general possible, because to make them foolproof for every possible input (sensible and nonsensical, evaluated and unevaluated, numerical and symbolical), the books would have had to go considerably beyond the mathematical and physical framework of the *GuideBooks*. In addition, few warnings are implemented for improper or improperly used arguments. The graphics provided in the solutions are mostly subject to a long list of refinements. Although the solutions do work, they are often sketchy and can be considerably refined and extended by the reader. This also means that the provided solutions to the exercises programs are not always very suitable for solving larger classes of problems. To increase their applicability would require considerably more code. Thus, it is not guaranteed that given routines will work correctly on related problems. To guarantee this generality and scalability, one would have to protect the variables better, implement formulas for more general or specialized cases, write functions to accept different numbers of variables, add type-checking and error-checking functions, and include corresponding error messages and warnings.

To simplify working through the solutions, the various steps of the solution are commented and are not always packed in a `Module` or `Block`. In general, only functions that are used later are packed. For longer calculations, such as those in some of the exercises, this was not feasible and intended. The arguments of the functions are not always checked for their appropriateness as is desirable for robust code. But, this makes it easier for the user to test and modify the code.

## 0.5 The Books Versus the Electronic Components

### 0.5.1 Working with the Notebooks

Each volume of the *GuideBooks* comes with a multiplatform DVD, containing fourteen main notebooks tailored for *Mathematica* 4 and compatible with *Mathematica* 5. Each notebook corresponds to a chapter from the printed books. (To avoid large file sizes of the notebooks, all animations are located in the Animations directory and not directly in the chapter notebooks.) The chapters (and so the corresponding notebooks) contain a detailed description and explanation of the *Mathematica* commands needed and used in applications of *Mathematica* to the sciences. Discussions on *Mathematica* functions are supplemented by a variety of mathematics, physics, and graphics examples. The notebooks also contain complete solutions to all exercises. Forming an electronic book, the notebooks also contain all text, as well as fully typeset formulas, and reader-editable and reader-changeable input. (Readers can copy, paste, and use the inputs in their notebooks.) In addition to the chapter notebooks, the DVD also includes a navigation palette and fully hyperlinked table of contents and index notebooks. The *Mathematica* notebooks corresponding to the printed book are fully evaluated. The evaluated chapter notebooks also come with hyperlinked overviews; these overviews are not in the printed book.

When reading the printed books, it might seem that some parts are longer than needed. The reader should keep in mind that the primary tool for working with the *Mathematica* kernel are *Mathematica* notebooks and that on a computer screen and there “length does not matter much”. The *GuideBooks* are basically a printout of the notebooks, which makes going back and forth between the printed books and the notebooks very easy. The *GuideBooks* give large examples to encourage the reader to investigate various *Mathematica* functions and to become familiar with *Mathematica* as a system for doing mathematics, as well as a programming language. Investigating *Mathematica* in the accompanying notebooks is the best way to learn its details.

To start viewing the notebooks, open the table of contents notebook TableOfContents.nb. *Mathematica* notebooks can contain hyperlinks, and all entries of the table of contents are hyperlinked. Navigating through one of the chapters is convenient when done using the navigator palette GuideBooksNavigator.nb.

When opening a notebook, the front end minimizes the amount of memory needed to display the notebook by loading it incrementally. Depending on the reader’s hardware, this might result in a slow scrolling speed. Clicking the “Load notebook cache” button of the GuideBooksNavigator palette speeds this up by loading the complete notebook into the front end.

For the vast majority of sections, subsections, and solutions of the exercises, the reader can just select such a structural unit and evaluate it (at once) on a year-2005 computer (≥512 MB RAM) typically in a matter of minutes. Some sections and solutions containing many graphics may need hours of computation time. Also, more than 50 pieces of code run hours, even days. The inputs that are very memory intensive or produce large outputs and graphics are in inactive cells which can be activated by clicking the adjacent button. Because of potentially overlapping variable names between various sections and subsections, the author advises the reader not to evaluate an entire chapter at once.

Each smallest self-contained structural unit (a subsection, a section without subsections, or an exercise) should be evaluated within one *Mathematica* session starting with a freshly started kernel. At the end of each unit is an input cell. After evaluating all input cells of a unit in consecutive order, the input of this cell generates a short summary about the entire *Mathematica* session. It lists the number of evaluated inputs, the kernel CPU time, the wall clock time, and the maximal memory used to evaluate the inputs (excluding the resources needed to evaluate the `Program` cells). These numbers serve as a guide for the reader about the to-be-expected running times and memory needs. These numbers can deviate from run to run. The wall clock time can be substantially larger than the CPU time due to other processes running on the same computer and due to time needed to render graphics. The data shown in the evaluated notebooks came from a 2.5 GHz Linux computer. The CPU times are generically proportional to the computer clock speed, but can deviate within a small factor from operating system to operating system. In rare, randomly occurring cases slower computers can achieve smaller CPU and wall clock times than faster computers, due to internal time-constrained simplification processes in various symbolic mathematics functions (such as `Integrate`, `Sum`, `DSolve`, …).

The Overview Section of the chapters is set up for a front end and kernel running on the same computer and having access to the same file system. When using a remote kernel, the directory specification for the package `Overview.m` must be changed accordingly.

References can be conveniently extracted from the main text by selecting the cell(s) that refer to them (or parts of a cell) and then clicking the "Extract References" button. A new notebook with the extracted references will then appear.

The notebooks contain color graphics. (To rerender the pictures with a greater color depth or at a larger size, choose Rerender Graphics from the Cell menu.) With some of the colors used, black-and-white printouts occasionally give low-contrast results. For better black-and-white printouts of these graphics, the author recommends setting the `ColorOutput` option of the relevant graphics function to `GrayLevel`. The notebooks with animations (in the printed book, animations are typically printed as an array of about 10 to 20 individual graphics) typically contain between 60 and 120 frames. Rerunning the corresponding code with a large number of frames will allow the reader to generate smoother and longer-running animations.

Because many cell styles used in the notebooks are unique to the *GuideBooks*, when copying expressions and cells from the *GuideBooks* notebooks to other notebooks, one should first attach the style sheet notebook GuideBooksStylesheet.nb to the destination notebook, or define the needed styles in the style sheet of the destination notebook.

## 0.5.2 Reproducibility of the Results

The 14 chapter notebooks contained in the electronic version of the *GuideBooks* were run mostly with *Mathematica* 5.1 on a 2 GHz Intel Linux computer with 2 GB RAM. They need more than 100 hours of evaluation time. (This does not include the evaluation of the currently unevaluatable parts of code after the Make Input buttons.) For most subsections and sections, 512 MB RAM are recommended for a fast and smooth evaluation "at once" (meaning the reader can select the section or subsection, and evaluate all inputs without running out of memory or clearing variables) and the rendering of the generated graphic in the front end. Some subsections and sections need more memory when run. To reduce these memory requirements, the author recommends restarting the *Mathematica* kernel inside these subsections and sections, evaluating the necessary definitions, and then continuing. This will allow the reader to evaluate all inputs.

In general, regardless of the computer, with the same version of *Mathematica*, the reader should get the same results as shown in the notebooks. (The author has tested the code on Sun and Intel-based Linux computers, but this does not mean that some code might not run as displayed (because of different configurations, stack size settings, etc., but the disclaimer from the Preface applies everywhere). If an input does not work on a particular machine, please inform the author. Some deviations from the results given may appear because of the following:
■ Inputs involving the function `Random[...]` in some form. (Often `SeedRandom` to allow for some kind of reproducibility and randomness at the same time is employed.)
■ *Mathematica* commands operating on the file system of the computer, or make use of the type of computer (such inputs need to be edited using the appropriate directory specifications).

■ Calculations showing some of the differences of floating-point numbers and the machine-dependent representation of these on various computers.
■ Pictures using various fonts and sizes because of their availability (or lack thereof) and shape on different computers.
■ Calculations involving `Timing` because of different clock speeds, architectures, operating systems, and libraries.
■ Formats of results depending on the actual window width and default font size. (Often, the corresponding inputs will contain `Short`.)

Using anything other than *Mathematica* Version 5.1 might also result in different outputs. Examples of results that change form, but are all mathematically correct and equivalent, are the parameter variables used in underdetermined systems of linear equations, the form of the results of an integral, and the internal form of functions like `InterpolatingFunction` and `CompiledFunction`. Some inputs might no longer evaluate the same way because functions from a package were used and these functions are potentially built-in functions in a later *Mathematica* version. *Mathematica* is a very large and complicated program that is constantly updated and improved. Some of these changes might be design changes, superseded functionality, or potentially regressions, and as a result, some of the inputs might not work at all or give unexpected results in future versions of *Mathematica*.

## 0.5.3 Earlier Versions of the Notebooks

The first printing of the Programming volume and the Graphics volumes of the *Mathematica GuideBooks* were published in October 2004. The electronic components of these two books contained the corresponding evaluated chapter notebooks as well as unevaluated versions of preversions of the notebooks belonging to the Numerics and Symbolics volumes. Similarly, the electronic components of the Numerics and Symbolics volume contain the corresponding evaluated chapter notebooks and unevaluated copies of the notebooks of the Programming and Graphics volumes. This allows the reader to follow cross-references and look up relevant concepts discussed in the other volumes. The author has tried to keep the notebooks of the *GuideBooks* as up-to-date as possible. (Meaning with respect to the efficient and appropriate use of the latest version of *Mathematica*, with respect to maintaining a list of references that contains new publications, and examples, and with respect to incorporating corrections to known problems, errors, and mistakes). As a result, the notebooks of all four volumes that come with later printings of the Programming and Graphics volumes, as well with the Numerics and Symbolics volumes will be different and supersede the earlier notebooks originally distributed with the

Programming and Graphics volumes. The notebooks that come with the Numerics and Symbolics volumes are genuine *Mathematica* Version 5.1 notebooks. Because most advances in *Mathematica* Version 5 and 5.1 compared with *Mathematica* Version 4 occurred in functions carrying out numerical and symbolical calculations, the notebooks associated with Numerics and Symbolics volumes contain a substantial amount of changes and additions compared with their originally distributed version.

## 0.6 Style and Design Elements

### 0.6.1 Text and Code Formatting

The *GuideBooks* are divided into chapters. Each chapter consists of several sections, which frequently are further subdivided into subsections. General remarks about a chapter or a section are presented in the sections and subsections numbered 0. (These remarks usually discuss the structure of the following section and give teasers about the usefulness of the functions to be discussed.) Also, sometimes these sections serve to refresh the discussion of some functions already introduced earlier.

Following the style of *The Mathematica Book* [45], the *GuideBooks* use the following fonts: For the main text, Times; for *Mathematica* inputs and built-in *Mathematica* commands, Courier plain (like `Plot`); and for user-supplied arguments, Times italic (like $userArgument_1$). Built-in *Mathematica* functions are introduced in the following style:

`MathematicaFunctionToBeIntroduced[`*typeIndicatingUserSuppliedArgument(s)*`]`
is a description of the built-in command `MathematicaFunctionToBeIntroduced` upon its first appearance. A definition of the command, along with its parameters is given. Here, *typeIndicatingUserSupplied-Argument(s)* is one (or more) user-supplied expression(s) and may be written in an abbreviated form or in a different way for emphasis.

The actual *Mathematica* inputs and outputs appear in the following manner (as mentioned above, virtually all inputs are given in `InputForm`).

```
(* A comment. It will be/is ignored as Mathematica input:
   Return only one of the solutions *)
Last[Solve[{x^2 - y == 1, x - y^2 == 1}, {x, y}]]
```

When referring in text to variables of *Mathematica* inputs and outputs, the following convention is used: Fixed, nonpattern variables (including local variables) are printed in Courier plain (the equations solved above contained the variables `x` and `y`). User supplied arguments to built-in or defined functions with pattern variables are printed in Times italic. The next input defines a function generating a pair of polynomial equations in *x* and *y*.

```
equationPair[x_, y_] := {x^2 - y == 1, x - y^2 == 1}
```

*x* and *y* are pattern variables (usimng the same letters, but a different font from the actual code fragments `x_` and `y_`) that can stand for any argument. Here we call the function `equationPair` with the two arguments `u + v` and `w - z`.

```
equationPair[u + v, w - z]
```

Occasionally, explanation about a mathematics or physics topic is given before the corresponding *Mathematica* implementation is discussed. These sections are marked as follows:

#### Mathematical Remark: Special Topic in Mathematics or Physics

A *short* summary or review of mathematical or physical ideas necessary for the following example(s).

From time to time, *Mathematica* is used to analyze expressions, algorithms, etc. In some cases, results in the form of English sentences are produced programmatically. To differentiate such automatically generated text from the main text, in most instances such text is prefaced by "∘" (structurally the corresponding cells are of type "PrintText" versus "Text" for author-written cells).

Code pieces that either run for quite long, or need a lot of memory, or are tangent to the current discussion are displayed in the following manner.

```
mathematicaCodeWhichEitherRunsVeryLongOrThatIsVeryMemoryIntensive⋱
OrThatProducesAVeryLargeGraphicOrThatIsASideTrackToTheSubjectUnder⋱
Discussion
(* with some comments on how the code works *)
```

To run a code piece like this, click the Make Input button above it. This will generate the corresponding input cell that can be evaluated if the reader's computer has the necessary resources.

The reader is encouraged to add new inputs and annotations to the electronic notebooks. There are two styles for reader-added material: "ReaderInput" (a *Mathematica* input style and simultaneously the default style for a new cell) and "ReaderAnnotation" (a text-style cell type). They are primarily intended to be used in the Reading environment. These two styles are indented more than the default input and text cells, have a green left bar and a dingbat. To access the "ReaderInput" and "ReaderAnnotation" styles, press the system-dependent modifier key (such as Control or Command) and 9 and 7, respectively.

## 0.6.2 References

Because the *GuideBooks* are concerned with the solution of mathematical and physical problems using *Mathematica* and are not mathematics or physics monographs, the author did not attempt to give complete references for each of the applications discussed [38], [20]. The references cited in the text pertain mainly to the applications under discussion. Most of the citations are from the more recent literature; references to older publications can be found in the cited ones. Frequently URLs for downloading relevant or interesting information are given. (The URL addresses worked at the time of printing and, hopefully, will be still active when the reader tries them.) References for *Mathematica*, for algorithms used in computer algebra, and for applications of computer algebra are collected in the Appendix A.

The references are listed at the end of each chapter in alphabetical order. In the notebooks, the references are hyperlinked to all their occurrences in the main text. Multiple references for a subject are not cited in numerical order, but rather in the order of their importance, relevance, and suggested reading order for the implementation given.

In a few cases (e.g., pure functions in Chapter 3, some matrix operations in Chapter 6), references to the mathematical background for some built-in commands are given—mainly for commands in which the mathematics required extends beyond the familiarity commonly exhibited by non-mathematicians. The *GuideBooks* do not discuss the algorithms underlying such complicated functions, but sometimes use *Mathematica* to "monitor" the algorithms.

References of the form *abbreviationOfAScientificField/yearMonthPreprintNumber* (such as quant-ph/0012147) refer to the arXiv preprint server [43], [22], [30] at http://arXiv.org. When a paper appeared as a preprint and (later) in a journal, typically only the more accessible preprint reference is given. For the convenience of the reader, at the end of these references, there is a Get Preprint button. Click the button to display a palette notebook with hyperlinks to the corresponding preprint at the main preprint server and its mirror sites. (Some of the older journal articles can be downloaded free of charge from some of the digital mathematics library servers, such as http://gdz.sub.uni-goettingen.de, http://www.emis.de, http://www.numdam.org, and http://dieper.aib.uni-linz.ac.at.)

As much as available, recent journal articles are hyperlinked through their digital object identifiers (http://www.doi.org).

## 0.6.3 Variable Scoping, Input Numbering, and Warning Messages

Some of the *Mathematica* inputs intentionally cause error messages, infinite loops, and so on, to illustrate the operation of a *Mathematica* command. These messages also arise in the user's practical use of *Mathematica*. So, instead of presenting polished and perfected code, the author prefers to illustrate the potential problems and limitations associated with the use of *Mathematica* applied to "real life" problems. The one exception are the spelling warning messages `General::spell` and `General::spell1` that would appear relatively frequently because "similar" names are used eventually. For easier and less defocused reading, these messages are turned off in the initialization cells. (When working with the notebooks, this means that the pop-up window asking the user "Do you want to automatically evaluate all the initialization cells in the notebook?" should be evaluated should always be answered with a "yes".) For the vast majority of graphics presented, the picture is the focus, not the returned *Mathematica* expression representing the picture. That is why the `Graphics` and `Graphics3D` output is suppressed in most situations.

To improve the code's readability, no attempt has been made to protect all variables that are used in the various examples. This protection could be done with `Clear`, `Remove`, `Block`, `Module`, `With`, and others. Not protecting the variables allows the reader to modify, in a somewhat easier manner, the values and definitions of variables, and to see the effects of these changes. On the other hand, there may be some interference between variable names and values used in the notebooks and those that might be introduced when experimenting with the code. When readers examine some of the code on a computer, reevaluate sections, and sometimes perform subsidiary calculations, they may introduce variables that might interfere with ones from the *GuideBooks*. To partially avoid this problem, and for the reader's convenience, sometimes `Clear[`*sequenceOfVariables*`]` and `Remove[`*sequenceOfVariables*`]` are sprinkled throughout the notebooks. This makes experimenting with these functions easier.

The numbering of the *Mathematica* inputs and outputs typically does not contain all consecutive integers. Some pieces of *Mathematica* code consist of multiple inputs per cell; so, therefore, the line numbering is incremented by more than just 1. As mentioned, *Mathematica* should be restarted at every section, or subsection or solution of an exercise, to make sure that no variables with values get reused. The author also explicitly asks the reader to restart *Mathematica* at some special positions inside sections. This removes previously introduced variables, eliminates all existing contexts, and returns *Mathematica* to the typical initial configuration to ensure reproduction of the results and to avoid using too much memory inside one session.

## 0.6.4 Graphics

In *Mathematica* 5.1, displayed graphics are side effects, not outputs. The actual output of an input producing a graphic is a single cell with the text `-Graphics-` or `-Graphics3D-` or `-GraphicsArray-` and so on. To save paper, these output cells have been deleted in the printed version of the *GuideBooks*.

Most graphics use an appropriate number of plot points and polygons to show the relevant features and details. Changing the number of plot points and polygons to a higher value to obtain higher resolution graphics can be done by changing the corresponding inputs.

The graphics of the printed book and the graphics in the notebooks are largely identical. Some printed book graphics use a different color scheme and different point sizes and line and edge thicknesses to enhance contrast and visibility. In addition, the font size has been reduced for the printed book in tick and axes labels.

The graphics shown in the notebooks are PostScript graphics. This means they can be resized and rerendered without loss of quality. To reduce file sizes, the reader can convert them to bitmap graphics using the Cell⟶ Convert To⟶Bitmap menu. The resulting bitmap graphics can no longer be resized or rerendered in the original resolution.

To reduce file sizes of the main content notebooks, the animations of the *GuideBooks* are not part of the chapter notebooks. They are contained in a separate directory.

## 0.6.5 Notations and Symbols

The symbols used in typeset mathematical formulas are not uniform and unique throughout the *GuideBooks*. Various mathematical and physical quantities (such as normals, rotation matrices, and field strengths) are used repeatedly in this book. Frequently the same notation is used for them, but depending on the context, also different ones are used, e.g. sometimes bold is used for a vector (such as $\mathbf{r}$) and sometimes an arrow (such as $\vec{r}$). Matrices appear in bold or as doublestruck letters. Depending on the context and emphasis placed, different notations are used in display equations and in the *Mathematica* input form. For instance, for a time-dependent scalar quantity of one variable $\psi(t; x)$, we might use one of many patterns, such as `ψ[t][x]` (for emphasizing a parametric $t$-dependence) or `ψ[t, x]` (to treat $t$ and $x$ on an equal footing) or `ψ[t, {x}]` (to emphasize the one-dimensionality of the space variable $x$).

Mathematical formulas use standard notation. To avoid confusion with *Mathematica* notations, the use of square brackets is minimized throughout. Following the conventions of mathematics notation, square brackets are used for three cases: a) Functionals, such as $\mathcal{F}_t[f(t)](\omega)$ for the Fourier transform of a function $f(t)$. b) Power series coefficients, $[x^k](f(x))$ denotes the coefficient of $x^k$ of the power series expansion of $f(x)$ around $x = 0$. c) Closed intervals, like $[a, b]$ (open intervals are denoted by $(a, b)$). Grouping is exclusively done using parentheses. Upper-case double-struck letters denote domains of numbers, $\mathbb{Z}$ for integers, $\mathbb{N}$ for nonnegative integers, $\mathbb{Q}$ for rational numbers, $\mathbb{R}$ for reals, and $\mathbb{C}$ for complex numbers. Points in $\mathbb{R}^n$ (or $\mathbb{C}^n$) with explicitly given coordinates are indicated using curly braces $\{c_1, \ldots, c_n\}$. The symbols $\wedge$ and $\vee$ for And and Or are used in logical formulas.

For variable names in formula- and identity-like *Mathematica* code, the symbol (or small variations of it) traditionally used in mathematics or physics is used. In program-like *Mathematica* code, the author uses very descriptive, sometimes abbreviated, but sometimes also slightly longish, variable names, such as `buildBril⋱ louinZone` and `FibonacciChainMap`.