Lecture Notes in Computational Science and Engineering

42

Alfred Schmidt
Kunibert G. Siebert

Design of Adaptive Finite Element Software

The Finite Element Toolbox ALBERTA

With 30 Figures

Springer

Alfred Schmidt

Zentrum für Technomathematik
Fachbereich Mathematik/Informatik
Universität Bremen
Bibliothekstr. 2
28359 Bremen, Germany
e-mail: schmidt@math.uni-bremen.de

Kunibert G. Siebert

Institut für Mathematik
Universität Augsburg
Universitätsstraße 14
86159 Augsburg, Germany
e-mail: siebert@math.uni-augsburg.de

Library of Congress Control Number: 2004113298

Mathematics Subject Classification (2000):
65M50, 65M60, 65N22, 65N30, 65N50, 65Y99, 68U20

ISSN 1439-7358
ISBN 3-540-22842-X Springer Berlin Heidelberg New York

Cover design: Friedhelm Steinen-Broo, Estudio Calamar, Spain
Cover production: *design & production*, Heidelberg
Typeset by the authors using a Springer TEX macro package
Production: LE-TEX Jelonek, Schmidt & Vöckler GbR, Leipzig

Printed on acid-free paper 46/3142/YL - 5 4 3 2 1 0

Preface

During the last years, scientific computing has become an important research branch located between applied mathematics and applied sciences and engineering. Nowadays, in numerical mathematics not only simple model problems are treated, but modern and well-founded mathematical algorithms are applied to solve complex problems of real life applications. Such applications are demanding for computational realization and need suitable and robust tools for a flexible and efficient implementation. Modularity and abstract concepts allow for an easy transfer of methods to different applications.

Inspired by and parallel to the investigation of real life applications, numerical mathematics has built and improved many modern algorithms which are now standard tools in scientific computing. Examples are adaptive methods, higher order discretizations, fast linear and non-linear iterative solvers, multi-level algorithms, etc. These mathematical tools are able to reduce computing times tremendously and for many applications a simulation can only be realized in a reasonable time frame using such highly efficient algorithms.

A very flexible software is needed when working in both fields of scientific computing and numerical mathematics. We developed the toolbox ALBERTA[1] for meeting these requirements. Our intention in the design of ALBERTA is threefold: First, it is a toolbox for fast and flexible implementation of efficient software for real life applications, based on the modern algorithms mentioned above. Secondly, in an interplay with mathematical analysis, ALBERTA is an environment for improving existent, or developing new numerical methods. And finally, it allows the direct integration of such new or improved methods in existing simulation software.

Before having ALBERTA, we worked with a variety of solvers, each designed for the solution of one single application. Most of them were based on data structures specifically designed for one single application. A combination of different solvers or exchanging modules between programs was hard to do.

[1]The original name of the toolbox was ALBERT. Due to copyright reasons, we had to rename it and we have chosen ALBERTA.

Facing these problems, we wanted to develop a general adaptive finite element environment, open for implementing a large class of applications, where an exchange of modules and a coupling of different solvers is easy to realize.

Such a toolbox has to be based on a solid concept which is still open for extensions as science develops. Such a solid concept can be derived from a mathematical abstraction of problem classes, numerical methods, and solvers. Our mathematical view of numerical algorithms, especially finite element methods, is based on our education and scientific research in the departments for applied mathematics at the universities of Bonn and Freiburg. This view point has greatly inspired the abstract concepts of ALBERTA as well as their practical realization, reflected in the main data structures. The robustness and flexible extensibility of our concept was approved in various applications from physics and engineering, like computational fluid dynamics, structural mechanics, industrial crystal growth, etc. as well as by the validation of new mathematical methods.

ALBERTA is a library with data structures and functions for adaptive finite element simulations in one, two, and three space dimension, written in the programming language ANSI-C. Shortly after finishing the implementation of the first version of ALBERTA and using it for first scientific applications, we confronted students with it in a course about finite element methods. The idea was to work on more interesting projects in the course and providing a strong foundation for an upcoming diploma thesis. Using ALBERTA in education then required a documentation of data structures and functions. The numerical course tutorials were the basis for a description of the background and concepts of adaptive finite elements.

The combination of the abstract and concrete description resulted in a manual for ALBERTA and made it possible that it is now used world wide in universities and research centers. The interest from other scientists motivated a further polishing of the manual as well as the toolbox itself, and resulted in this book.

These notes are organized as follows: In Chapter 1 we describe the concepts of adaptive finite element methods and its ingredients like the domain discretization, finite element basis functions and degrees of freedom, numerical integration via quadrature formulas for the assemblage of discrete systems, and adaptive algorithms.

The second chapter is a tutorial for using ALBERTA without giving much details about data structures and functions. The implementation of three model problems is presented and explained. We start with the easy and straight forward implementation of the Poisson problem to learn about the basics of ALBERTA. The examples with the implementation of a nonlinear reaction-diffusion problem and the time dependent heat equation are more involved and show the tools of ALBERTA for attacking more complex problems. The chapter is closed with a short introduction to the installation of the

ALBERTA distribution enclosed to this book in a UNIX/Linux environment. Visit the ALBERTA web site

`http://www.alberta-fem.de/`

for updates, more information, FAQ, contributions, pictures from different projects, etc.

The realization of data structures and functions in ALBERTA is based on the abstract concepts presented in Chapter 1. A detailed description of all data structures and functions of ALBERTA is given in Chapter 3. The book closes with separate lists of all data types, symbolic constants, functions, and macros.

The cover picture of this book shows the ALBERTA logo, combined with a locally refined cogwheel mesh [17], and the norm of the velocity from a calculation of edge tones in a flute [4].

Starting first as a two-men-project, ALBERTA is evolving and now there are more people maintaining and extending it. We are grateful for a lot of substantial contributions coming from: Michael Fried, who was the first brave man besides us to use ALBERT, Claus-Justus Heine, Daniel Köster, and Oliver Kriessl. Daniel and Claus in particular set up the GNU configure tools for an easy, platform-independent installation of the software.

We are indebted to the authors of the gltools, especially Jürgen Fuhrmann, and also to the developers of GRAPE, especially Bernard Haasdonk, Robert Klöfkorn, Mario Ohlberger, and Martin Rumpf.

We want to thank the Department of Mathematics at the University of Maryland (USA), in particular Ricardo H. Nochetto, where part of the documentation was written during a visit of the second author. We appreciate the invitation of the Isaac Newton Institute in Cambridge (UK) where we could meet and work intensively on the revision of the manual for three weeks.

We thank our friends, distributed all over the world, who have pointed out a lot of typos in the manual and suggested several improvements for ALBERTA.

Last but not least, ALBERTA would not have come into being without the stimulating atmosphere in the group in Freiburg, which was the perfect environment for working on this project. We want to express our gratitude to all former colleagues, especially Gerhard Dziuk.

Bremen and Augsburg, October 2004

Alfred Schmidt and Kunibert G. Siebert

Contents