Algorithms of Scientific Computing

Overview and General Remarks

Michael Bader
Summer Term 2014
Classification of the Lecture

Students of Informatics:
- Informatics Bachelor & Master
- Informatics: Games Engineering (Bachelor)
- Information Systems (Wirtschaftsinformatik)

Students of (Computational) Engineering:
- Computational Science and Engineering (CSE): application area (cat. E1)
- Engineering Science

Students of Physics:
various “flavours” → who?

Students of Mathematics:
Bachelor, Master, as minor?

... anyone else? *Warm Welcome!*
Tutorials

**Tutors:**
- Denis Jarema (FFT, space-filling curves)
- Kilian Röhner (hierarchical methods, sparse grids)

**Time & Day:**
- by default, tutorials will be on Wednesdays (10-12, MI 02.07.023)
- skipped on Apr 23 (student assembly)

**“Style”:**
- worksheets with applications & examples
- no compulsory part
- iPython Notebook
Algorithms in Scientific Computing
Scientific Computing


**Attempt of a definition:**

Scientific Computing is . . .

- (numerical) simulation of problems from science or engineering using High Performance Computing (Bungartz, TUM)
- the interdisciplinary conjunction of mathematical and computer science methods as well as different applications of the natural sciences and engineering disciplines, e.g. (TU Darmstadt)
- the subfield of computer science concerned with constructing mathematical models and quantitative analysis techniques and using computers to analyze and solve scientific problems (Wikipedia, 2012)
- an *interdisciplinary discipline*
- the focus at our chair SCCS
Algorithms in Scientific Computing?

Central Question:
What do I “get” from this lecture?
• ... in particular in the field of Scientific Computing?
• ... in general in the field of Informatics?
⇒ What could/should/do I want to learn in
• ... Informatics?
• ... Computing?

Cross-topical aspects: What central ...
• problems
• techniques, methods
• analytical questions
... of Informatics/Computing/... play a (major) role?
Cross-Topical Aspects
Representation of Information

Claim:

Informatics is the science (or art) of storing information such that it can be used (processed) efficiently.

Examples for information and storage technique:

- tables (data bases of all kind)
- trees, graphs (path searching, . . .)
- multi-dimensional fields (raster data, etc.)

Our topic:

How do we store continuous data (mathematical functions)?
For Comparison: Representation of Scalars

A brief history of the representation of numbers:

- “tally marks”: |, ||, |||, |||| (still successfully used to count drinks in bars & restaurants)
- number symbols such as I, V, X, MMIV: compact but tedious for computing
- positional notation (decimal numbers, binary system, etc.): ease of arithmetics up to machine computing

Crucial ideas:

- Hierarchy (different “value” of digits depending on their position)
- Structure (concept of 0 as a placeholder!)

Our topic:

“Coefficients and basis functions”
Representation of Mathematical Functions

Possibilities of representation (historical):

- *analytical functions*: \( f(x) = e^x \sin(x) \)

- *tabulated values*
  (z.B. logarithm tables, newly rastered data/sampling)

- *interpolation* (also piecewise):
  (polygonal chain/curve, polynomial interpolation, spline interpolation, trigonometrical interpolation, . . .)

**Goal**: access and use information efficiently!

- more compact storage
- identification of certain properties (information)
- generally: more efficient algorithms for processing/computations
Multi-Dimensional Data

Examples for multi-dimensional data structures:

- Matrices
- Image data (images, tomography, movies, ...)
- Discretization based on grids (discretization of physical models / partial differential equations)
- Coordinates of any kind (often going along with graphs)
- Tables (relational databases)
- In financial mathematics: baskets of stocks/options/...

Core topic: linearization/sequentialization

- Storage of data structures in memory
- Data processing (traversal)
- goals: preserve neighborhood (data locality), fast index computation, “continuity”, “symmetry”, etc.
Problems on Multi-Dimensional Data

Data Representation:
- interpolation and approximation
- data compression
- related: quantification of error

Classification and Learning:
- classify: predict unknown function values based on known data
- learn: extract function from noisy data

Numerical Simulation:
- compute approximate solution to unknown function
- function given as solution of a partial/ordinary differential equation
Recursive Algorithms and Hierarchical Data Structures

“Traditional” style of algorithms in scientific computing:

- FORTRAN programs; procedural/iterative programming
- strongly based on loops and arrays

Nowadays:

- **Recursive and hierarchical:** w.r.t. algorithms (partitioning of problem) and data structures (trees, object orientation)
- **Adaptive:** invest effort, where most benefit can be achieved
- **“Optimal Complexity”:** high approximation order, etc.
- **Distributed:** Computing on parallel and distributed systems
- **Hardware-oriented:** → High Performance Computing

⇒ generally applicable concepts and ideas
Schedule

Fast Fourier Transform:
- discrete Fourier transform as 2D, 3D interpolation
- FFT as divide-and-conquer algorithm
- transform for data compression (images, audio and video data)

Hierarchical basis and sparse grids
- adaptive integration and Archimedes’ quadrature
- hierarchical basis functions
- the curse of dimensionality → sparse grids
- wavelets

Space trees and space-filling curves
- sequential data structures and traversal of octrees
- definition and construction of space-filling curves
- parallelisation and partitioning