Structure and Problem Solving in ISL

R. Baker Kearfott
Department of Mathematics
University of Louisiana at Lafayette
mailto:rbk@louisiana.edu
with George Corliss, Ned Nedialkov,
John Pryce, and Spencer Smith

Presented at Schloß Dagstuhl Seminar 06021,
"Reliable Implementation of Real Number Algorithms: Theory and Practice,"
January 10, 2006
Slides: http://interval.louisiana.edu/preprints/2006_Dagstuhl.pdf

Context of this Project

- Validated computations have generally matured over the past 40 years.
  - Standardization is advancing on low-level computational aspects.
  - Algorithms and computational procedures are maturing.
  - There is a significant history of package building.
- There are numerous commercial examples of general-purpose libraries for non-validated scientific computing.
  - NAG
  - Harwell
  - LAPACK (for linear systems only)
  - IMSL, etc.

Goals

- Facilitate validated computation within the community of experts
- Foster wider use and acceptance within the scientific community in general

To achieve these goals, it is useful to

- standardize behavior of the interval arithmetic itself;
- unify low-level utilities and procedure calls across different problem-solving areas;
- specify at a level that allows easy porting between different languages (e.g. between MATLAB, C++ and Fortran);
- provide package simplicity and clarity in the documentation.

To do this, we *must*

- incorporate previous work;
- engender wide participation within our professional community.

Structure: Building on Previous Work

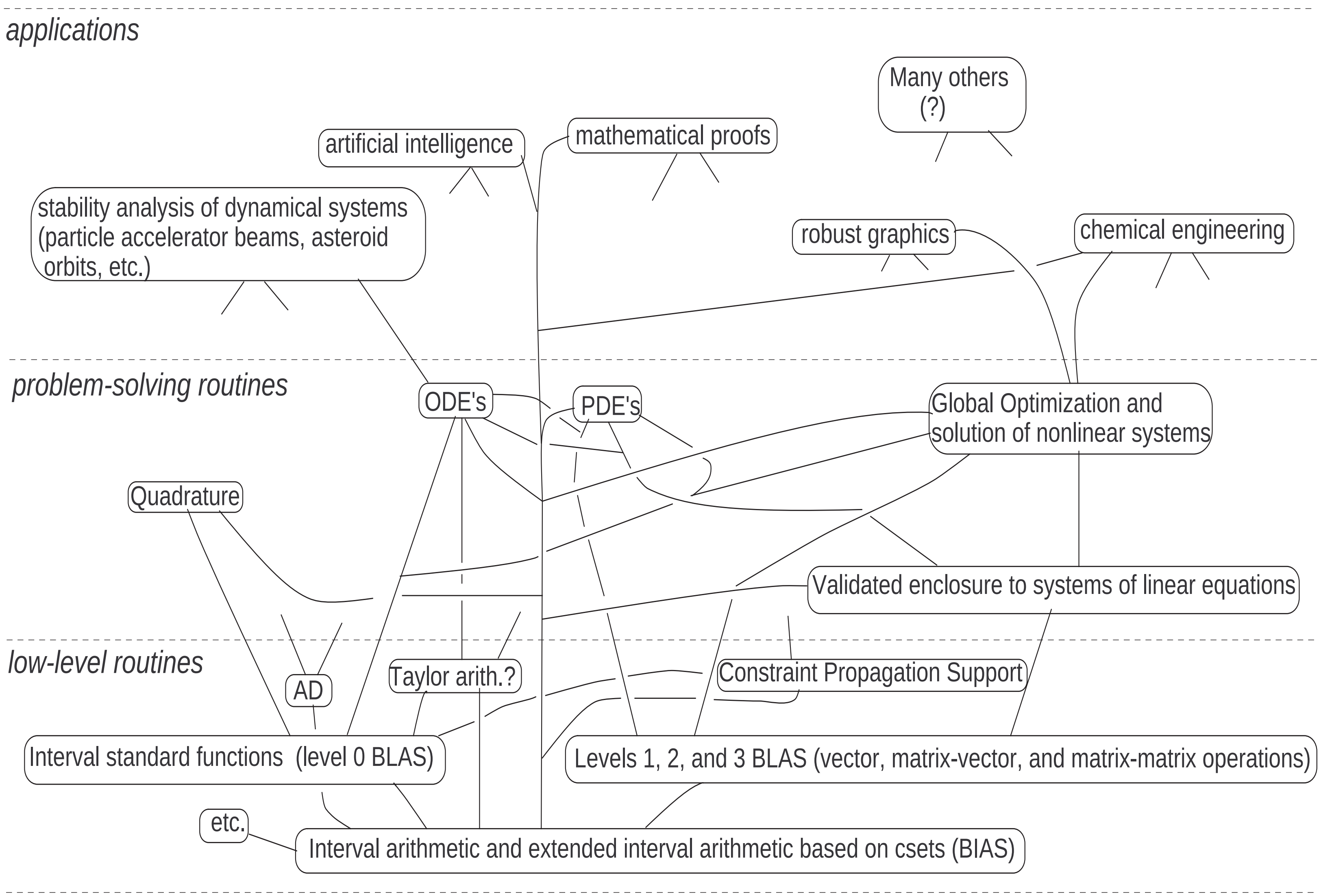
We envision ISL to be hierarchical. Thus, we will build on

- increasing agreement concerning how low-level interval arithmetic should go;
- maturing refinements in understanding of interval arithmetic with infinite intervals;
- both interval and non-interval work in linear algebra;
- the extensive previous research in problem-solving areas such as
  - validated enclosure to solutions of linear systems
  - quadrature, global optimization and nonlinear systems, ODE's, PDE's, fuzzy logic, etc.

R. B. Kearfott, Dagstuhl, 10 Jan., 2006

A Tentative (and evolving) Structure

The structure is hierarchical. Part of our work is to unify this structure.



On Interval Arithmetic

1. Basic properties of interval arithmetic are substantially agreed upon at this point.
   - Previous work on a Fortran interval arithmetic standard has resulted in a document[a] and agreement on most issues.
   - Present work on a C++ interval arithmetic standard is progressing nicely.
2. Correct interval I/O is now possible to supply in virtually all languages without too much difficulty (e.g. with David Gay's `gdtoa`).
3. Various small issues remain, such as
   - representation of the empty set and operations with the empty set.
   - evaluation of a standard function when part (but not all of the input domain is outside the domain of definition of the function

   These issues will largely be resolved with the cset theory of Pryce, with emerging consensus, and with careful documentation.

We have examined several high-quality packages; we intend to utilize these, modifying them as necessary.

[a]although not formally adopted

On Extended Interval Arithmetic

1. Interval arithmetic involving division by zero or other function ranges that are infinite is useful in various contexts:
   - in interval Newton methods, when the complement of an interval is intersected with an interval.
   - in constraint propagation, when overestimation gives infinite extents, but subsequent intersection results in ordinary intervals.
2. Starting from work of Kahan (1968, "A more complete interval arithmetic," Tech. rep. Univ. Toronto), various extended interval arithmetics have been developed.
   - The extended floating arithmetic in the IEEE 754 floating point standard was motivated partially to support such an extended interval arithmetic.
3. Various revisions and corrections of this extended arithmetic have been proposed since Kahan.
   - Proposed systems have not given expected or consistent results in all problem-solving contexts.
   - Even if correct, existing systems are not optimally sharp, often giving $[-\infty, \infty]$.

ISL Actions: Extended Interval Arithmetic

We intend to

- use Pryce's cset theory to define the arithmetic clearly and unambiguously;
- document the assumptions clearly where necessary, in the implementation incorporated into ISL, so users know exactly what to expect;
- enable sharper bounds, especially when intersected with finite intervals;
- forge a consensus within our community;
- provide a reference implementation in our library, eventually to be incorporated into all problem-solving routines.

The BLAS

1. The Basic Linear Algebra Subroutines, were originally by Lawson, Hanson, Kincaid and Krough (1979, see http://portal.acm.org/citation.cfm?doid=355841.355848), and were developed and championed by Dongarra et al in the LINPACK (see http://www.netlib.org/linpack/) and later LAPACK (see http://www.netlib.org/lapack/) packages.

2. Their nearly universal acceptance (due to simplicity and utility) has made them into a de-facto standard. Their standardization process has more recently proceeded formally in the BLAST fora, with a resulting multi-programming-language standards document (see http://www.netlib.org/blas/blast-forum/).

3. The Interval BLAS, championed by Chenyi Hu (see http://www.cs.uca.edu/faculty_files/Hu.htm), is in the "Journal of Development."

BLAS Issues to be Resolved

- Most issues were resolved.
- Issues remain concerning exceptions in interval vector and matrix operations when operations with the empty set or operations with infinities are encountered.
- There was not a consensus among interval arithmetic experts at the time the overall BLAS standard was adopted.
- The issues are important in the overall BLAS standardization effort, since a focus is efficiency of vector and matrix operations in advanced architectures.

Action on Interval BLAS in ISL

1. We will provide a reference (not necessarily optimal) implementation of the BLAS.
2. Our analysis and implementation of the fundamentals of extended interval arithmetic, including arithmetic involving the empty set, should clarify procedures for handling exceptions in interval vector and matrix operations (or else indicate how it can be done exception-free).
3. Participation of a large part of our community and fostering simplicity and clarity will lead to:
   - wide use of the library,
   - consensus on procedure,
   - standardization,
   - ease of porting,
   - widespread application.

Problem Solving Routines

- Problem-solving routines are the numerical routines above the low-level routines and BLAS but below user-applications.
- These routines correspond to chapters in a numerical analysis text.
- Examples of collections of problem-solving routines include the Karlsruhe toolboxes.
- Problem-solving routines include various helper routines more specific than the low-level routines, but possibly of use on their own in further algorithmic research.
- For ISL to be powerful enough for a maximum of applications,
  - participation of experts among us will be encouraged and facilitated,
  - existing work should be incorporated, and
  - formal testing and quality assurance (as to be introduced by Spencer Smith) should be included.

Two Examples: Problem-Solving

Context
Goals
Structure
Prob. Solving
Examples
Linear Systems
Lin. Sys. Helpers
Optimization
Opt. Helpers
Conclusion

- Validated enclosure of systems of linear equations.
- Global Optimization and validated solution of nonlinear systems.

We will describe

- work on which we hope to build,
- tentative structure, and
- helper routines

for these two examples.

Validated Enclosure of Linear Systems

- Siegfried Rump has produced much good work over the years, and has embodied it in INTLAB (a MATLAB toolbox; see http://www.ti3.tu-harburg.de/~rump/intlab/); we hope to build upon much of that.
- We can also incorporate our improved extended arithmetic to bound infinite solution sets for non-regular problems.
- We can include routines for computing exact bounds (for small systems).
- Longer term: We can include routines for specially structured systems, as with packages for non-validated floating point solvers.

Helper Routines: Linear Systems

Context

Goals

Structure

Prob. Solving

Examples

Linear Systems

Lin. Sys. Helpers

Optimization

Opt. Helpers

Conclusion

1. Non-validated floating point routines from other packages
   - Linear system solvers
   - Linear and quadratic programming solvers
2. Preconditioner set-up and solve routines
   - Inverse midpoint
   - Optimal width
   - Mignitude optimal
   - S-preconditioners, left-optimal, and right-optimal preconditioners
   - Various heuristic combinations
3. Solver routines
   - Computation of exact hulls
   - Gaussian elimination
   - Application of a Gauss–Seidel step
   - Gauss–Seidel iteration and validation
4. Epsilon-inflation routines
5. Etc.

Nonlinear Systems; Global Optimization

Context
Goals
Structure
Prob. Solving
Examples
Linear Systems
Lin. Sys. Helpers
Optimization
Opt. Helpers
Conclusion

- Previous work includes that of Jansson et al, Arnold Neumaier and the COCOS project (see http://www.mat.univie.ac.at/users/neum/public_html/glopt/coconut/), GlobSol, constraint solvers such as ICOS (see http://www-sop.inria.fr/coprin/ylebbah/icos/), etc.
- These systems represent many years of work.
- Initially, nonlinear systems and global optimization capability in ISL may need to be represented by one or more of these packages, essentially unchanged.
- With work, we will improve the portability, maintainability, etc. of these packages by integrating them more fully with the lower-level support routines of GlobSol.
- Research into algorithms is continuing, and will result in significant improvements in years to come.

Validated Optimization: Helpers

Context

Goals

Structure

Prob. Solving

Examples

Linear Systems

Lin. Sys. Helpers

Optimization

Opt. Helpers

Conclusion

- Many lower-level ISL routines
  - constraint propagation support
  - automatic differentiation
  - (possibly) Taylor arithmetic
  - various routines from the linear systems chapter

  will be useful.
- Routines for rigorous linear underestimators, such as described in Borradaile and van Hentenryck (see http://www.mat.univie.ac.at/~neum/glopt/mss/BorH04.pdf) or Hongthong et al (see http://interval.louisiana.edu/preprints/estimates_of_powers.pdf)
- Interval Newton routines (partially or totally based on linear system solving routines)
- Routines to support processing lists of boxes
- Routines implementing stopping criteria
- Routines to do tessellation (bisection, taking the complement of a box in a list of boxes, etc.)
- One or more routine representing the overall branch and bound process.
- Etc.

Conclusion

1. We envision a comprehensive, universally used library[a]. In this way, we hope to go beyond previous efforts.
2. Problem solving routines from all areas of numerical analysis will eventually be represented.
3. We view the effort as promoting standardization, portability, and re-use.
4. To attain these goals with a practical number of resources, the following are important:
   - Cooperation, discussion, consensus, and participation within our professional community
   - Participation from professional software vendors, such as NAG.
   - Support from home institutions and granting agencies (travel grants, release time, etc.)

Quality, comprehensive libraries are not compiled by a single person or small group of people over a short time.

[a] This is in contrast to offering general languages, such as in the COCONUT project or GAMS, or offering graphical user interfaces such as in various commercial packages