A modeling system for mathematics

(Ein Modellierungssystem für Mathematik)

O.Prof. Dr. Arnold Neumaier, Universität Wien

1 Summary

This project aims at the development of a flexible modeling system for the specification of models for large-scale numerical work in optimization, data analysis, and partial differential equations. Its input should be provided in a form natural for the working mathematician, while the choice of the numerical solvers and the transformation to the format required by the solvers is done by the interface system. The input format should combine the simplicity of LaTeX source code with the semantic conciseness and modularity of current modeling languages such as AMPL, and it should be as close as possible to the mathematical language people use to explain and communicate their models in publications and lectures.

In order that the system is useful for the intended applications, interfaces translating the model formulated in the proposed system into the input required for current state of the art solvers, and into the dominant current modeling languages are needed and shall be provided. Moreover, certain shortcomings of the current generation of modeling languages, such as the lack of support for the correct treatment of uncertainties and rounding errors, shall be overcome.

The experience gained in this project will be useful in future work in the more general context of mathematical knowledge management.

Contents

2 Scientific Aspects

During the work on the global optimization environment COCONUT, it became apparent that the currently available modeling languages (cf. the recent synopsis by Kallrath [20]), which were very succesful in dramatically simplifying the use of large-scale optimization software for the average user, still have significant limitations in several directions.

Most prominent is the inability of the languages to express high level mathematical structures: For example, partial differential equations must be specified not in their natural form but in a fixed discretization, stochastic optimization problems must be reduced explicitly to sequences of ordinary optimization problems, and (probabilistic, worst case, or rounding-based) uncertainty specification in the available data is either impossible or very awkward. Model specification would be simpler and less error prone if the problems could be specified on their natural level, and the modifications needed to solve them would be done by tools operating on this specification. The modifications could then be adapted to the (increasing) capabilities of the solvers, and different modifications of the same model could be easily tried.

Another limitation for applications in pure mathematics is the lack of precision in the specification of mathematical problems to be solved. This is due to the fact that data conversion is usually accompanied by uncontrolled rounding errors, since traditionally numerical solvers introduced additional rounding errors anyway. However, there is now software with mathematical verification capabilites [28, 29] even for problems whose solution involves floating point computation, and interfacing to them needs full control of all approximations made. This is particularly important for applications in pure mathematics such as checking the validity of the solution of the Kepler problem by Hales [15] involving a large number of computer-assisted proofs allegedly verifying certain optimization problems, or the still unsolved problem of the minimum norm of the Thompson lattice, where the condition to be checked involves integer coefficients with several hundred digits [37].

Instead of augmenting the modeling capabilities of a language each time a new type of application is encountered – as it is currently done –, we propose in this project to create a modeling system with a universal modeling language that allows a semantic representation of arbitrary mathematical constructs. This language shall be interpreted by a system which can “understand” enough of this representation to perform automatic translations to specialized systems with well-defined solution capabilities.

In contrast to the many different and differently structured programming languages and modeling tools in use across the disciplines that use mathematics, the language of mathematics is general and uniform across all disciplines. It is learnt by every mathematician, computer scientist, physicist, and engineer as part of their training, as well as by other users of mathematics. By designing the new modeling language as a slightly formalized version of the traditional mathematical language, users are relieved from unnecessarily doubling of work by first representing the ideas in a general mathematical framework (such as a LaTeX report of the model) and later reducing it to a formal description by coding it in the appropriate formal language (e.g., AMPL). As an extra benefit, the participants in interdisciplinary applications would not need to learn the ideosyncracies of specialized languages but could concentrate on their contribution in the familiar language of mathematics.

The present project aims at the specification of such a modeling system. Of course, partial progress towards such a modeling language is already visible in many current software systems, see Section 2.2. However, none of the systems available has the capabilities required for a universal mathematical prototyping language.

While we shall concentrate in the implementation on the aspects directly relevant for applications in optimization and numerical analysis (where our main expertise resides), we want to make the design of the language and its basic implementation broad enough to enable others to build upon it a flexible, easy-to-use system for applications in any field of mathematics. In particular, we plan to design the system in a way that it should be easy for others to complement the new language by a user-friendly interface to computer algebra systems and theorem provers.

Ultimately, the system could develop into part of a system managing not only large user-specific models in important applications, but also the collected mathematical knowledge of our culture. While this is far beyond the scope of this project, projects going in this direction from a different perspective are already under way; indeed, there are regular conferences on mathematical knowledge management; see, e.g., (`http://www.cs.bham.ac.uk/~mmk/events/MKM07/`).

2.1 Goals and expectations

We expect during the project to design a modeling language which

- is based on traditional mathematical syntax (colloquial mathematics slightly formalized to ensure a unique interpretation),
- provides tools to translate mathematical specifications of problems into a unified internal representation in terms of logical statements about the objects defined, thus 100% preserving the semantics,
- extends and completely restructures our earlier modeling language NOP-2,
- integrates an implemented decision tree for mathematical software which automatically selects the right tools (recognizing a linear system of equations, an optimization problem, a partial differential equation, etc.),
- allows the access of a large variety of systems from a common mathematical platform, in particular the following:
  - LaTeX for creating printable documents describing or summarizing models, in multiple target languages,
  - other modeling languages, in particular AMPL and GAMS,
  - constraint solvers for checking consistency,
  - optimization systems for finding approximate best scenarios,
  - numerical solvers for getting approximate answers,

and implement a working prototype that executes the above funtionality. If time permits (but we are somewhat pessimistic about that) we'd also like to interface to

- some computer algebra system,
- some logical framework for reasoning with the concepts.

2.2 State of the art

Some available systems. Many existing tools provide partial functionality of the kind to be created in the project but only tied to specific applications, or with a limited scope. The following list of software systems is not exhaustive but illustrates important features and limitations of what is currently available. In particular, we assess their potential value as a modeling tool for complex optimization problems.

LaTeX. LaTeX (`http://www.latex-project.org`) is the de facto standard for the communication and publication of scientific documents, widely used by mathematicians, scientists, and engineers. It is fairly user-friendly and produces documents of excellent quality; however, it only specifies the syntax and the quotation structure (references to equations, theorems, papers, footnotes, etc.) of the documents, leaving the semantics inaccessible.

Markup languages. OpenMath (`http://www.openmath.org`) is an extensible language for representing the semantics of mathematical objects as a structured text. Its purpose is to facilitate the exchange of mathematical information between computer programs, databases, or worldwide web documents, and to enable its display in a browser. But there is no intrinsic display form for OpenMath objects. MathML (`http://www.w3.org/Math`) is a low-level specification for describing mathematics as a basis for machine to machine communication, with emphasis on web applications. MathML deals principally with the presentation of mathematical objects (so that MathML generated display looks like nice ordinary mathematics). It only has limited facilities for dealing with content; but it can embed OpenMath constructs. OMDoc (`http://www.omdoc.org/omdoc`) is a semantics-oriented representation format and ontology language for mathematical knowledge extending the markup of OpenMath and MathML to the document and theory level of mathematical documents. The voluminous representations

```
<OMOBJ>
  <OMA>
    <OMS cd="calculus1" name="int"/>
    <OMBIND>
      <OMS cd="fns1" name="lambda"/>
      <OMBVAR> <OMV name="x"/> </OMBVAR>
      <OMA>
        <OMS name="sin" cd="transc1"/>
        <OMV name="x"/>
      </OMA>
    </OMBIND>
  </OMA>
</OMOBJ>
```

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
<matrix>
  <matrixrow>
    <cn> 0 </cn> <cn> 1 </cn> <cn> 0 </cn>
  </matrixrow>
  <matrixrow>
    <cn> 0 </cn> <cn> 0 </cn> <cn> 1 </cn>
  </matrixrow>
  <matrixrow>
    <cn> 1 </cn> <cn> 0 </cn> <cn> 0 </cn>
  </matrixrow>
</matrix>
</math>
```

of $\int \sin x\,dx$ in OpenMath and of the matrix $\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$ in MathML, taken from the above web sites, show that a markup format is not suitable as a modeling tool.

Mizar. Mizar (`http://mizar.org`) is a formal language for specifying the syntax and semantics for mathematical texts in a way that they can be verified automatically by computer. A 16 year old journal "Formalized Mathematics" specializes in publishing Mizar-generated mathematics. The Mizar input is difficult to read and create, but the output is reasonably readable, embedding the formulas in English (though often somewhat uneven) sentences. There are no capabilities to specify data-driven applications.

Common mathematical language. Making ordinary mathematical text completely comprehensible for the computer is an exceedingly difficult task, at present virtually impossible without much interactive assistence. Fully automatic partial work is reported in two Ph.D. theses by Simon [53] and Zinn [58], which exhibit the many problems encountered in their attempts to turn some elementary number theory texts into checkable proofs.

Therefore, the informal mathematical language must be somewhat restricted to be automatically analyzable. KAMAREDDINE [22] emphasizes the problems to be overcome for a full formalization acceptable to the typical mathematician, and introduces MathLang (KAMAREDDINE et al. [24, 25], VAN TILBURG [55]), a non-fully-formalized language close to the common mathematical language, for interactively annotating mathematics written in English and translating it into Mizar [23]. Currently, the system seems to be in an explorative stage of development and does not seem to have a structure that would lend itself easily for the specification of large optimization models.

The Z notation. The Z notation (`http://vl.zuser.org`) is a specification language based on Zermelo-Fraenkel set theory and first order predicate logic, available since 2002 as an ISO standard [19]. It is a mathematical specification notation, used by industry as part of the software and hardware development process for the highly reliable definition of algorithmic components, particularly for safety-critical systems. Z is intended to increase human understandability of the specified system and to allow the possibility of formal reasoning and development. Professional quality typeset documents based on Z specifications can be produced by a LaTeX interface provided by the tool set CADiZ (`http://www-users.cs.york.ac.uk/~ian/cadiz/`); this system also provides support for finding proofs related to Z specifications. Various other tools are available, including an interface to the theorem proving environment HOL. However, there is no support for nonalgorithmic mathematics such as abstract infinite sets, integrals, etc.. Moreover, the specifications themselves look more like programs than like mathematics.

Modeling systems. For the application to numerical problems, in particular large-scale optimization, a number of versatile modeling languages are available; see KALLRATH [20]. Widely used examples include AMPL (`http://www.ampl.com`) and GAMS (`http://www.gams.com`). They are very efficient for problems from the applications, with interfaces to all major state of the art optimization packages, providing the derivatives needed by these solvers automatically. But the current modeling languages also have serious limitations, mentioned already in the introduction.

CVX. CVX is a system for disciplined convex programming, allowing the user to specify nonlinear optimization problems with automatic verification of their convexity properties, in an easy to use syntax. While very specialized, it is an example of a system that checks the assumptions of a numerical method before trying it, and by its syntax enforces a healthy disciplined approach to model formulation.

Decision trees for software. The Decision Tree for Optimization Software by MITTELMANN (`http://plato.la.asu.edu/guide.html`) gives online advice about which software to use for which kind of optimization problem. Similar decision trees are discussed in the literature for a variety of problems in scientific computing; see, e.g., ADDISON et al. [1], RAMAKRISHNAN & RIBBENS [44], RAMAKRISHNAN et al. [45], HOUSTIS et al. [17], MAARUSTER et al. [31], BUNUS [5]. Although not a modeling tool, such decision trees have a natural place in a modeling system that automatically selects the solver to use.

Computer algebra systems and numerical prototyping languages. Packages such as Maple (`http://www.maplesoft.com`), Mathematica (`http://www.wolfram.com/products/mathematica/index.html`), MuPAD (`http://www.mupad.de`), etc. are computer algebra systems with lots of built in functionality. The systems Matlab (`http://www.mathworks.com/products/matlab`), Scilab (`http://www.scilab.org`), etc. are programming environments for efficient numerical computation and prototyping, easy to use for the working mathematician. All these are widely used but don't give the user access to the mathematical meaning of the constructs, so that one cannot use the defined relations for anything but for computations.

Theorema. The Theorema package (`http://www.risc.uni-linz.ac.at/research/theorema`) is a Mathematica-based extension of current computer algebra systems by facilities for supporting mathematical proving. It allows to build interactively proofs for mathematical theorems, with proofs that are readable and can be expanded or shortened in a user-controlled manner. Input and output appear in a form acceptable for mathematicians, though often somewhat stilted. There are no capabilities for specifying data-driven applications.

Logical frameworks. There are a large number of logical frameworks and theorem proving systems which allow the verification of (carefully specified) proofs for mathematical statements, primarily of properties of programming languages and logics. Widely used systems include Coq (`http://pauillac.inria.fr/coq`), HOL light (`http://www.cl.cam.ac.uk/~jrh13/hol-light`), PVS (`http://pvs.cl.sri.com`), and Twelf (`http://www.cs.cmu.edu/~twelf`). Many of these systems can be used to prove nontrivial mathematics; see, e.g., WIEDIJK [56] (100 formally proved theorems, ranging from the fundamental theorem of algebra to Brouwer's fixed point theorem), and WIEDIJK [57] (a comparison of provers for the irrationality of $\sqrt{2}$) But proofs are generally unreadable manually (except by expert users). Typically, statements and proofs must be provided in a program-like structure far removed from mathematical practice. This makes these systems unsuitable for mathematical modeling applications.

Grammatical framework. The Grammatical Framework (`http://www.cs.chalmers.se/~aarne/GF`) is a special-purpose programming system whose core is a generic grammar processor capable to recognize and generate important fragments of many languages. It particularly addresses multilinguality, semantic conditions of well-formedness, language modularity and the reuse of grammars in different formats and as software components. It provides enough functionality for the restricted natural language support required for a formal language close to the traditional informal mathematical usage. Other language-related references of potential interest for the project are CARLSON et al. [6] for multilingual mathematics and Greenstone (`http://www.greenstone.org`) for multilingual software for building and distributing digital library collections.

2.3 Methods

Based on the preceding discussion of what exists, we can paraphrase the goal of the project: Combine the advantages of LaTeX for writing and viewing mathematics with the user-friendliness of mathematical modeling systems such as AMPL for the flexible definition of large-scale numerical applications, with the high-level discipline of CVX for solving convex programming problems, and with the semantical clarity of the Z notation for the precise specification of concepts and statements.

Emphasis must be given to two concerns:

1. Ease of debugging and modifying models through the use of a mostly self-explaining language and comfortable interactive tools,

2. Efficiency for large-scale applications (especially those related to optimization and differential equations) involving complex models with large, structured data sets.

Since we are not aiming at verifying any statements in the proposed modeling system but only at representing their meaning faithfully enough to feed solvers, many of the difficult problems encountered in attempts to fully formalizing mathematics are absent. On the other hand, we are aware of the progress made in this direction and of what can be learnt from it for our own project.

We have already much experience with large-scale applications of numerical methods in protein folding [33, 38], regularization [26, 27, 30, 34], global optimization [2, 18, 35, 41], quantum mechanics [13, 32], with software projects of significant size [7, 8, 39, 40, 50], with the design of modeling languages [36, 42, 47, 48], and with translation tools between formal languages used as input to various external systems, among others the modeling languages AMPL and GAMS [41, 49].

The methods we need within the project are mostly well-known and well-tested in specific application domains, in particular in compiler design. However, they need to be adapted to the particular goals of the project. The challenge is in integrating these known methods into an easy-to-use system.

Initially we shall define a restricted subset of LaTeX as input format, and create a front end that translates restricted LaTeX documents into a simple graph representation based on the Vienna Graph Template Library (VGTL) [46]. This will enable us to start working on the backend to solvers by using and extending the functionality of the translators in the COCONUT environment.

In a second stage, we shall collect and analyze a large number of optimization models and constraint satisfaction models from the literature [3, 4, 9, 10, 12, 16, 20, 21, 43, 54], described in informal mathematical terms, and derive from these a flexible language which enables us to encode these models with minimal deviations from the informal description as it would be encoded in LaTeX. This will replace the initial LaTeX dialect as input format.

To make the information collected in the graph representation available for actual use, we plan to create a number of interface tools to other syastems:

- an interactive editor for creating and modifying models,
- a translator from the VGTL representation back into the input language, for testing correctness.
- a translator from the VGTL representation into an efficient internal representation amenable to symbolic analysis,
- a translator to (and the design of) a written format of the internal representation for exchange with other systems and a corresponding reader,
- a translator from AMPL into the written internal format, to make the COCONUT test set of optimization models [51, 52] available in the new system,
- the implementation of a decision tree for optimization and (if time permits) other numerical tasks, and methods to extract the required information from the models,
- interfaces to some of the solvers in the decision tree.

We shall also design and provide a documentation facility for LaTeX output in natural language in several languages, at least English and German. The documentation language should be such that, in full documentation mode, the model can be automatically and unambiguously reconstructed from the documentation, at least from the English version.

Finally, a lot of work will go into testing the various parts and ensuring that they perform as required.

2.4 Research plan and project schedule

During the **first year** we will work along two independent lines. First, and most important, we shall design a grammatical extension of a simplified subset of LaTeX as preliminary input language. This subset will need a more rigid structuring than standard LaTeX to simplify the semantical analysis. Second, we shall collect an assortment of (preferably optimization oriented) applications for testing the versatility and convenience of the input format, such as

- standard numerical tasks
  - nonlinear systems
  - differential equations
- logic problems

  - the Collatz problem
  - problems from the TPTP Problem Library for Automated Theorem Proving [54]
- specifically for optimization:
  - some complex optimization models currently implemented in AMPL or GAMS [14, 52]
  - semiinfinite optimization
  - local optimization with differential constraints
  - linear matrix inequalities
  - specifying relaxations
  - discrete variable support
    * linear and quadratic assignment problem
    * traveling salesman problem
    * other problems from the OR Library [3]

At the end of the first year we plan to finish the implementation of a prototype reader of the preliminary input format to an internal data structure.

The **second year** shall focus on the implementation and tests of various modules (beginning with intelligent LaTeX, AMPL output, and a COCONUT interface) using the internal data structures.

We shall put special emphasis on the ability to produce verified input and output relevant for computer-assisted proofs [37] and on flexible variable and data formats to be competitive with other modeling systems.

In parallel we shall use the experience gained in working with the preliminary input format to gradually evolve its structure and the associated language parser. During this process we shall improve the semantical analysis tool and adapt the language and the internal format. To ensure that we end up with a flexible language close to mathematical practice, we shall also collect and study informal phrases from diverse textbooks, as they are used to introduce a formal context in mathematical writing. We shall create a database for managing changing contexts and notational conventions, one of the important properties of informal mathematical writing.

At the start of the **third year** we shall fix a grammar for a formalized version of mathematical language definition and stabilize the language interpreter. We shall study the context dependent features of mathematical terminology, and try to find ways of resolving ambiguities automatically from context, or interactively if an automatic resolution is infeasible. We shall add multiple (output) language support for the documentation of models specified in the new (input) language, We shall also create an interactive tool for editing partially specified models and modifying previous models, Using the application database collected in year one we shall produce templates for models formulating applications in various areas, with emphasis on optimization and constraint satisfaction problems. In addition we shall write comprehensive documentation of the language and its use in order to make the resulting package easily usable. On a project web page, linked to from the principle investigator's widely visible web pages, we shall provide the interpreter for free download on a GPL basis [11] and an online interface for trying sample problems using the software.

2.5 National and international cooperations

We expect to collaborate with Bob Fourer (Northwestern University, USA) on AMPL-related questions, with Mike Grant (Stanford University, USA) on problems related to specifying convex problems in CVX, with Eildert Groeneveld (Bundesforschungsanstalt fr Landwirtschaft Neustadt, Germany) on specifying large models from genetics in VCE, and with Pavel Solin (University of Texas at El Paso, USA) on specifying optimization problems involving partial differential equations in a hp-FEM finite element method.

3 Scientific Environment

The project will be carried out at the research group CMA (Computational MAthematics (`http://www.mat.univie.ac.at/~neum/cma.html`)), located at the Mathematics Faculty of the University of Vienna. There will be some synergy effects with the FWF project on COCONUT (mainly aimed at improving the COCONUT solver and its structural core) and the FSP project on "Advanced modeling in global optimization" (aimed at exploring new paradigma for solving global optimization problems), both located at CMA.

Apart from Prof. Arnold Neumaier, the principal investigator, Prof. Hermann Schichl from the CMA will actively participate in the project.

Prof. Neumaier is one of the leading scientists in the field of global optimization, and has an extensive background in applications of numerical analysis (protein folding, regularization), statistical data analysis (covariance estimation, artificial intelligence) and computer-assisted proofs.

Prof. Schichl is the main developer of the COCONUT environment, a system for global optimization which takes AMPL input, creates an internal DAG format using the Vienna Graph Template Library (VGTL) [46], and from it a number of representations in different formats (AMPL, GAMS, LGO, KNITRO, MatLab, Mathematica) to connect either to the COCONUT solver, or to other state of the art local or global optimization packages. In particular, it incorporates automatic differentiation facilities and related graph transformations. Work on the COCONUT environment provided a lot of expertise in the management of large software systems and the construction of compilers and translators.

Prof. Neumaier and Prof. Schichl jointly developed the modeling language NOP [42, 47, 48]; the need for a very flexible modeling language as proposed in the present project became apparent during the work on NOP, the COCONUT environment, and the trends visible in the development of other modeling languages.

4 Dissemination; expected long-term consequences

The results of our research shall be published in regular journals and/or conference proceeding. In addition, there will be a project web page where we shall provide the interpreter for free download on a GPL basis [11] and an online interface for trying sample problems using the software produced during the project.

A huge amount of modeling is performed by mathematically educated people in many areas of application. Thus, we expect that our new modeling system will relieve them of most of the cumbersome translation or implementation work needed to transfer their mathematical knowlegde to a formal system; in the ideal case, these modelers will not even need to learn a new formal language, since they usually know LaTeX and the mathematical language. Furthermore, using the system it will be finally possible to rigorosly and conveniently formulate mathematical programming problems which have to be solved in a verified way.

There will be general implications for society, as well, since the project centers on model creation for numerical analysis and optimization, one of the most important aspects for economics. Advanced algorithms and better user interfaces could make it possible to improve construction of technical devices, making them optimally, safer, powerful, and possibly cheaper.

As a long-term consequence we hope that our system may gradually develop into a standard for communicating formal mathematics. This project contributes at a low level towards such a standard. Our roadmap (for future projects) includes among other the following important additional steps:

- get results from the solvers and put them into a nice format, automatically generating human readable reports in natural language,
- get output from a logic system and check it on the level of our modeling system,
- create a programming language for self-verifying programming,
- encode undergraduate mathematics textbook knowledge into our modeling system,

An easy to use system for communicating formal mathematics, mainly based on the traditional mathematical language, may also simplify the teaching of formal precision in the education of students in mathematics and computer science.

References

[1] C.A. Addison, W.H. Enright, P.W. Gaffney, I. Gladwell, and P.M. Hanson. Algorithm 687: a decision tree for the numerical solution of initial value ordinary differential equations. *ACM Trans. Math. Software*, 17:1–10, 1991.

[2] C.S. Adjiman, S. Dallwig, C.A. Floudas, and A. Neumaier. A global optimization method, alphaBB, for general twice-differentiable constrained NLPs - i. theoretical advances. *Computers and Chemical Engineering*, 22:1137–1158, 1998.

[3] J. E. Beasley. OR-Library, 2005. WWW Site. `http://people.brunel.ac.uk/~mastjjb/jeb/info.html`.

[4] Anthony Brooke, David Kendrick, and Alexander Meeraus. *GAMS - A User's Guide (Release 2.25).* Boyd & Fraser Publishing Company, Danvers, Massachusetts, 1992.

[5] P. Bunus. A simulation and decision framework for selection of numerical solvers in scientific computing. In *Proc. 39th ann. Symp. Simulation, IEEE Computer Society, Washington*, pp. 178–187, 2006.

[6] L. Carlson, J. Saludes, and A. Strotmann. State of the art in multilingual and multicultural creation of digital mathematical content, 2005. Web Advanced Learning Technologies, Manuscript. `http://webalt.math.helsinki.fi/content/e16/e301/e305/Deliverable1.2_eng.pdf`.

[7] COCONUT. The COCONUT project home page. `http://www.mat.univie.ac.at/coconut`.

[8] H. Schichl et al. The COCONUT Environment for global optimization, 2005. Software Package. `http://www.mat.univie.ac.at/coconut-environment`.

[9] C. A. Floudas and P.M. Pardalos. *A Collection of Test Problems for Constrained Global Optimization Algorithms*, vol. 455 of *Lecture Notes Comp. Sci.* Springer, Berlin, 1990.

[10] C.A. Floudas, P.M. Pardalos, C.S. Adjiman, W.R. Esposito, Z.H. Gümüs, S.T. Harding, J.L. Klepeis, C.A. Meyer, and C.A. Schweiger. *Handbook of Test Problems in Local and Global Optimization.* Kluwer, Dordrecht, 1999. `http://titan.princeton.edu/TestProblems/`.

[11] Free Software Foundation. GNU general public license, 2007. WWW-Document. `http://www.gnu.org/copyleft/gpl.html`.

[12] Robert Fourer, David M. Gay, and Brian W. Kernighan. *AMPL — A Mathematical Programming Language.* Thomson, second ed., 2003.

[13] P. Frantsuzov, A. Neumaier, and V.A. Mandelshtam. Gaussian resolutions for equilibrium density matrices. *Chem. Phys. Letters*, 381:117–122, 2003. quant-ph/0306124.

[14] GLOBAL. GLOBAL library, 2002. WWW-Document. http://www.gamsworld.org/global/globallib.htm.

[15] T.C. Hales. A proof of the Kepler conjecture. *Ann. Math.*, 162:1065–1185, 2005.

[16] J.N. Hooker. *Logic-based Methods for Optimization: Combining Optimization and Constraint Satisfaction.* Wiley, New York, 2000.

[17] E.N. Houstis, A.C. Catlin, N. Dhanjani, and J.R. Rice. MyPYTHIA: a recommendation portal for scientific software and services. *Concurrency Computat.: Pract. Exper.*, 14: 1481–1505, 2002.

[18] W. Huyer and A. Neumaier. Global optimization by multilevel coordinate search. *J. Global Optimization*, 14:331–355, 1999.

[19] ISO. International standard ISO/IEC 13568:2002 information technology – Z formal specification notation – syntax, type system and semantics, 2002.

[20] J. Kallrath. *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis.* Vieweg, 2002.

[21] J. Kallrath, editor. *Modeling Languages in Mathematical Optimization.* Kluwer, Dordrecht, 2003.

[22] F. Kamareddine. Motivations for MathLang, 2005. Slides. http://www.macs.hw.ac.uk/~fairouz/talks/talks2005/mathlang-general-talk.pdf.

[23] F. Kamareddine, M. Maarek, K. Retel, and J.B. Wells. Gradual computerisation/formalisation of mathematical texts into Mizar, 2004. Manuscript. http://www.macs.hw.ac.uk/~fairouz/papers/drafts/mizar.pdf.

[24] F. Kamareddine, M. Maarek, and J.B. Wells. *Flexible encoding of mathematics on the computer*, pp. 160–174. Springer, Berlin, 2004.

[25] F. Kamareddine, M. Maarek, and J.B. Wells. MathLang: An experience driven language of mathematics. *Electronic Notes in Theoretical Computer Science*, 93C:123–145, 2004.

[26] L.C. Kaufman and A. Neumaier. PET regularization by envelope guided conjugate gradients. *IEEE Trans. Medical Imag.*, 15:385–389, 1996.

[27] L.C. Kaufman and A. Neumaier. Regularization of ill-posed problems by envelope guided conjugate gradients. *J. Comput. Graph. Stat.*, 6:451–463, 1997.

[28] R.B. Kearfott. GlobSol: History, composition, and advice on use. In Ch. Bliek et al., editor, *Global Optimization and Constraint Satisfaction*, pp. 17–31. Springer, Berlin, 2003.

[29] Y. Lebbah. ICOS (Interval COnstraints Solver), 2003. WWW-Document. http://www-sop.inria.fr/coprin/ylebbah/icos/.

[30] W.A. Lodwick, A. Neumaier, and F. Newman. Optimization under uncertainity: methods and applications in radiation therapy. In *FUZZ-IEEE'2001, Proc. 10th IEEE Int. Conf. Fuzzy Systems, Vol. 3*, pp. 1219–1222, Australia, 2001. Melbourne.

[31] S. Maaruster, V. Negru, D. Petcu, and C. Sandru. Intelligent front-end for solving nonlinear systems of differential and algebraic equations. *J. Math. Sci.*, 108:1139–1151, 2002.

[32] V.A. Mandelshtam and A. Neumaier. Further generalization and numerical implementation of pseudo-time Schrödinger equations for quantum scattering calculations. *J. Theor. Comput. Chemistry*, 1:1–15, 2002. physics/0204049.

[33] A. Neumaier. Molecular modeling of proteins and mathematical prediction of protein structure. *SIAM Rev.*, 39:407–460, 1997.

[34] A. Neumaier. Solving ill-conditioned and singular linear systems: A tutorial on regularization. *SIAM Review*, 40:636–666, 1998.

[35] A. Neumaier. Complete search in continuous global optimization and constraint satisfaction. In A. Iserles, editor, *Acta Numerica 2004*, pp. 271–369. Cambridge University Press, 2004.

[36] A. Neumaier. Mathematical model building. In J. Kallrath, editor, *Modeling Languages in Mathematical Optimization*, vol. 88 of *Applied Optimization*. Kluwer, Boston, 2004. Chapter 3.

[37] A. Neumaier. Computer-assisted proofs. In *IEEE SCAN 2006 proceedings*, 2007. to appear.

[38] A. Neumaier, S. Dallwig, W. Huyer, and H. Schichl. New techniques for the construction of residue potentials for protein folding. In P. Deuflhard et al., editor, *Algorithms for Macromolecular Modelling*, vol. 4 of *Lecture Notes Comput. Sci. Eng.*, pp. 212–224, Berlin, 1999. Springer.

[39] A. Neumaier and E. Groeneveld. Restricted maximum likelihood estimation of covariances in sparse linear models. *Genet. Sel. Evol.*, 30:3–26, 1998.

[40] A. Neumaier and T. Schneider. Estimation of parameters and eigenmodes of multivariate autoregressive models. *ACM Trans. Math. Softw.*, 27:27–57, 2001.

[41] A. Neumaier, O. Shcherbina, W. Huyer, and T. Vinko. A comparison of complete global optimization solvers. *Math. Programming B*, 103:335–356, 2005.

[42] Arnold Neumaier. NOP - A Compact Input Format for Nonlinear Optimization Problems. In I. M. Bomze, T. Csendes, R. Horst, and P.M. Pardalos, editors, *Developments in Global Optimization*, pp. 1–18. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1997.

[43] Y. Pochet and L.A. Wolsey. *Production Planning by Mixed Integer Programming.* Springer, New York, 2006.

[44] N. Ramakrishnan and C.J. Ribbens. Mining and visualizing recommendation spaces for elliptic pdes with continuous attributes. *ACM Trans. Math. Software*, 26:254–273, 2000.

[45] N. Ramakrishnan, J.R. Rice, and E.N. Houstis. GAUSS: an online algorithm selection system for numerical quadrature. *Adv. Engin. Software*, 33:27–36, 2002.

[46] H. Schichl. VGTL (Vienna Graph Template Library) version 1.0, reference manual. Technical Report Appendix to "Upgraded State of the Art Techniques implemented as Modules", Deliverable D13, the COCONUT project, July 2003. Version 1.1, October 2003. `http://www.mat.univie.ac.at/coconut-environment`.

[47] H. Schichl and A. Neumaier. The NOP-2 modeling language. In J. Kallrath, editor, *Modeling Languages in Mathematical Optimization*, vol. 88 of *Applied Optimization.* Kluwer, Boston, 2004. Chapter 15.

[48] H. Schichl, A. Neumaier, and S. Dallwig. The NOP-2 modeling language. *Ann. Oper. Research*, 104:281–312, 2001.

[49] H. Schichl, O. Shcherbina, and A. Pownuk. External converters. Deliverable D14, COCONUT project, July 2003. in "Set of Combination Algorithms for State of the Art Modules".

[50] T. Schneider and A. Neumaier. Algorithm 808: ARfit - a Matlab package for the estimation of parameters and eigenmodes of multivariate autoregressive models. *ACM Trans. Math. Softw.*, 27:58–65, 2001.

[51] O. Shcherbina. The COCONUT benchmark, a benchmark for global optimization and constraint satisfaction, 2007. WWW-Document. `http://www.mat.univie.ac.at/~neum/glopt/coconut/Benchmark/Benchmark.html`.

[52] O. Shcherbina, A. Neumaier, Djamila Sam-Haroud, Xuan-Ha Vu, and Tuan-Viet Nguyen. Benchmarking global optimization and constraint satisfaction codes. In Ch. Bliek, Ch. Jermann, and A. Neumaier, editors, *Global Optimization and Constraint Satisfaction*, Berlin, 2003. Springer.

[53] D.L. Simon. *Checking Number Theory Proofs in Natural Language.* PhD thesis, Univ. of Texas, Austin, 1990.

[54] G. Sutcliffe and C. Suttner. The TPTP problem library for automated theorem proving, 2007. WWW Site. `http://www.cs.miami.edu/~tptp/`.

[55] P. van Tilburg. Exploring the core of MathLang, 2006. Manuscript. `http://paul.luon.net/reports/HWU-MACS-MathLang.pdf`.

[56] F. Wiedijk. Formalizing 100 theorems, 2007. WWW-Document. `http://www.cs.ru.nl/~freek/100/index.html`.

[57] F. Wiedijk. The seventeen provers of the world, 2007. WWW-Document. `http://www.cs.ru.nl/~freek/comparison/index.html`.

[58] C. Zinn. *Understanding Informal Mathematical Discourse.* PhD thesis, Univ. Erlangen, 2004.