ABSTRACTS

for

An International Conference on

Numerical Analysis with Automatic Result Verification

Mathematics, Applications, and Software

February 25 through March 1, 1993

Lafayette, Louisiana

The abstracts are listed in alphabetical order by last name of first author. Abstracts for the Workshop on Interval Methods in Artificial Intelligence appear in this collection, in alphabetical order, following the general abstracts.

Finding by Interval Arithmetic the Number of Zeros of an Analytic Function Inside a Rectangle

O. Aberth

Department of Mathematics, Texas A&M University
College Station, TX 77843-3368 USA

For a given analytic function $f(z)$, there are various methods available for deciding whether a region in the complex plane contains a zero. When $f(z)$ is a polynomial and the region is a circle, the method of D. H. Lehmer can be used. More generally, the number of zeros in the region can be obtained by the more difficult numerical evaluation of the integral

$$\frac{\pi}{2}\int\frac{f'(z)dz}{f(z)}$$

around the boundary of the region. The identical result can also be obtained by evaluating the topological degree. R. B. Kearfott has described a method for evaluating topological degree that converts to an efficient interval arithmetic method when the region is a rectangle with sides parallel to the real and imaginary axes. We describe the method and its use in locating the zeros of an analytic function.

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX

Iterative Methods of Solving Nonsmooth Optimization Problems in Hilbert and Banach Spaces with Nonasymptotic Estimates of Convergence Rate

Ya. Alber

Technion – Israel Institute of Technology
Dept. of Mathematics, 3200 Haifa, Israel

We consider iterative processes for minimization of nonsmooth convex functionals in Hilbert and Banach spaces, for solving equations and variational inequalities with multivalued and even discontinuous operators. We assume that the operator of variational inequality (subgradient of functional) has arbitrary growth "on infinity" and it is uniformly monotone, i.e. uniformly degenerate. Under such weak restrictions we prove a strong convergence of approximate methods, investigate their stability and establish both asymptotic and nonasymptotic esimates for the convergence rate. We show that the behaviour of the iterative processes depends not only on the structure and smoothness of the problems, but also on the geometric characteristics of the Hilbert and Banach spaces.

The Cholesky Method for Interval Data

G. Alefeld and G. Mayer

Institut für Angewandte Mathematik, Universität Karlsruhe
Postfach 6980, W-7500 Karlsruhe, Germany

We apply the well-known Cholesky method to bound the solutions of linear systems with symmetric matrices and right-hand sides both of which are varying within given intervals. We discuss some general properties and derive criteria which guarantee the feasibility and the optimality of the method.

Practical Interval Matrix Computations

F. L. Alvarado

The University of Wisconsin - Madison
Dept. of ECE, 1425 Johnson Dr.
Madison, Wisconsin 53706-1607 USA

This presentation will describe several practical aspects of computing with intervals in large problems. After a general introduction to practical interval computations and the choice of languages for interval computation work, the author will describe a useful general purpose package for the manipulation of large sparse matrices. This popular package, the Sparse Matrix Manipulation System, has been extended recently to interval computations of many kinds and includes tools for the visualization of interval computations. Although the straightforward use of interval computations in this package is not always optimal, the package is a good workbench to illustrate the advantages and pitfalls of interval computation using large sparse matrices and vectors.

In addition to describing and illustrating the interval version of the Sparse Matrix Manipulation System, this presentation will describe several of the unique tools specific to interval computation that were developed, and present a few ideas that may be new to the audience. The first of these ideas is the "singleton partitioned inverse" method. This method is an extrapolation of the partitioned inverse method for solving linear equations. For ordinary matrices, this method has the virtue of being a highly parallel means of performing repeat solutions once a matrix has been factored. The method has another advantage: when dealing with "thin" matrices and interval right hand sides, a slight modification of the partitioned inverse method can yield the hull of the solution set directly for a large class of matrices without the serious degradation in sparsity that results from more traditional interval solution methods.

The second novel concept explored in this presentation is the relationship between the LINPACK matrix condition number estimator and practical algorithms for the determination of the hull of linear equations. The presentation will explore the practicality of this approach to interval computation.

A third novel notion explored in this presentation will be to study the effect of ordering of the matrix on interval growth for sparse matrices. A surprising result is that some fill-in is desirable to help control interval growth.

Finally, the presentation will describe a few application examples to power systems engineering for the determination of secure regions of operation of the power system when uncertainty in the network injections exists.

Interval and Circular Enclosures of the Set of Solutions to Initial Value Problems. The Wrapping Effect.

R. Anguelov

Department of Applied Mathematics
National University of Science and Technology
Box 346, Bulawayo, Zimbabwe

The Initial Value Problem (IVP) for systems of Ordinary Differential Equations

$$\dot{x} = f(t, x) \tag{1}$$

with a set-valued initial condition

$$x(t_0) = x^0 \in X^0 \subset \mathbb{R}^n \tag{2}$$

is considered. The wrapping effect is a well known obstacle in the construction of interval enclosures of the set of solutions $x(x^0; t)$ of the system (1) when $x^0 \in X^0$. We investigate the wrapping effect introducing a new concept of a wrapping function. The wrapping function is an interval function defined by using an IVP associated with problem (1), (2). It is proved that the wrapping function is the limit of the interval enclosures produced by step-by-step numerical methods when the step size tends to zero. Therefore the wrapping effect is the difference between the wrapping function and the optimal interval enclosure of the solution $x(X^0; t) = \{x(x^0; t) : x^0 \in X^0\}$ to the problem (1), (2). This is used to classify the IVP with respect to the wrapping effect. A class of IVP without wrapping effect is determined.

Enclosing of the solution $x(X^0; t)$ of (1), (2) by circle-valued functions of the form $S(t) = (c(t), r(t)) = \{x \in \mathbb{R}^n : \|x - c(t)\| < r(t)\}$ is proposed. The wrapping effect in the case of linear systems of differential equations is studied using a circular wrapping function. A class of IVPs without wrapping effect is determined. Since this class depends on the type of enclosures that are applied the class of IVPs without circular wrapping effect is different from the class of IVPs without interval wrapping effect.

Higher-Order Combined Iterative Methods for Simultaneous Determination of Zeros of Analytic Functions

L. Atanassova

Institut für Dynamische Systeme, Universität Bremen
D-2800 Bremen 33, Germany

Using a modification of Koenig's theorem we present a family of combined iterative methods with extremely rapid convergence for the simultaneous determination all zeros of an analytic function (inside a simple smooth closed contour in the complex plane). The convergence analysis is included. Some algorithms for the inclusion of the zeros of an analytic functions derived from the presented family of combined methods are considered.

Bounds for the R-order of Some Classes of Simultaneous Inclusion Methods for Polynomial Roots

L. Atanssova and J. Herzberger

Institut für Dynamische Systeme, Universität Bremen
D-2800 Bremen 33, Germany
and
Fachbereich Mathematik, Universität Oldenburg
W-2900 Oldenberg, Germany

For a general model for deriving some classes of higher-order methods for the simultaneous inclusion of polynomial zeros, we can calculate lower bounds for their R-order of convergence. It is shown that, in this general approach, the single-step methods are faster convergent than the corresponding total-step methods. Some formulas for the lower bounds are explicitly given.

This general approach makes it possible to easily derive bounds for the R-order of concrete methods if some characteristic parameters are known. With the help of interval arithmetic estimations, these parameters are obtained in most cases by simple applications of the basic rules for the width operator. This will be shown for some cases in the literature.

The results of this approach are important to determine efficient iteration methods. It is shown how one can determine optimal methods in some families of higher-order simultaneous methods. In addition to this, we are now able to treat some families of iteration methods by a simple application of our results.

New Methods for Bracketing Eigenvalues of Self-adjoint Operators

C. Beattie

Department of Mathematics
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061 USA

A new approach for computing tight lower bounds to the eigenvalues of a class of semibounded self-adjoint operators is presented that requires comparatively little a priori spectral information and permits in some circumstances the effective use of nonconforming finite-element trial functions. The method makes use of parameterized intermediate quadratic forms and generalizes in a certain sense the left-definite variant of Lehmann's inclusion intervals. This formulation is developed into a systematic framework based on recent inertia results in the Weinstein-Aronszajn theory. This provides greater flexibility in analysis and results in a final computational task involving sparse, well-structured matrices.

Bounds to Eigenvalues of Parameter Dependent Differential Equations

H. Behnke

Inst. f. Angewandte Mathematik
Technische Universität Clausthal
D-3392 Clausthal-Zellerfeld, Germany

We consider the natural bending vibrations of a free standing blade of a turbine disc. The mathematical model describing this problem [1] results in the following eigenvalue problem with ordinary differential equations:

$$(\Phi_z v'' + \Phi_{yz} w'')'' - \Omega^2(\Theta v')' = \lambda v$$

$$(\Phi_{yz} v'' + \Phi_y w'')'' - \Omega^2(\Theta w')' - \Omega^2 w = \lambda w$$

and boundary conditions

$$v(0) = v'(0) = w(0) = w'(0) = 0, \quad v''(1) = v'''(1) = w''(1) = w'''(1) = 0.$$

Here x is the cartesian coordinate of the blade, v and w are the displacements in the y and z direction, $\Phi_y = \Phi_y(x)$ and $\Phi_z = \Phi_z(x)$ are area moments in y and z direction and $\Omega^2\Theta$ denotes the normal force in the blade due to rotation. The dimensionless eigenvalue λ corresponds to the circular frequency. The eigenvalues $\lambda = \lambda(\Omega)$ depend on the real parameter Ω (the angular velocity) which is assumed to vary in the interval $[0, 30]$.

In the lecture we will show how verified bounds of the form

$$p(\Omega) - \epsilon \leq \lambda(\Omega) \leq p(\Omega) + \epsilon \text{ for all } \Omega \in [0, 30]$$

can be computed for the smallest eigenvalue curves. Here ϵ is a small positive number and p is an explicitly known function. The eigenvalue curves show the interesting curve veering phenomenon; by means of the calculated bounds, we can prove that the smallest eigenvalues curves do not *cross* each other.

"Verified" means, that rounding errors are controlled rigorously by the use of interval arithmetic. An advantage of the described method is, that it can be applied to eigenvalue problems with partial differential equations just as well.

Reference

1. H. Irretier and O. Mahrenholz. "Eigenfrequencies and Mode Shapes of a Free-Standing, Twisted, Tapered and Rotating Blade with Respect to an Elastically Supported Root, *ASME*, pp. 1–9, 1981.

On Interval Mathematical Models of Differentiative Electronic Circuits

I. V. Belousova

ap. 59, 16 / 2, pr. Yu. Gagarina
Saint Petersburg, SU-196211, Russia

The report considers the accuracy of an electrical differentiative circuit with input representing a sum of finite spectrum signal and interferences. The interval analysis approach is used to describe interferences and parasitic elements of the circuit. A mathematical expression for an upper bound on the relative error of differentiation is derived and the accuracy of this expression is explored. Algorithms for parametrical optimization of the circuit using minimization of this upper bound are proposed.

Exact Estimates of the Long Term Stability of Dynamical Systems Applied to the Weakly Nonlinear Design of Large Storage Rings

M. Berz and G. Hoffstaetter

Department of Physics and Astronomy and
National Superconducting Cyclotron Laboratory
Michigan State University, East Lansing, MI 48824 USA

The motion in weakly nonlinear dynamic systems like circular accelerators and storage rings like the planned Superconducting Supercollider can be described very well by high order Taylor maps describing the action on phase space. Recently it has been shown that such transfer maps can be obtained very elegantly and accurately using high order automatic differentiation and differential algebraic techniques in several variables. In typical examples, derivatives of up to order ten in six variables are needed. This goes far beyond the abilities of other methods, which are limited to about order three.

Using nonlinear normal form and other methods, the Taylor maps can be used to obtain families of approximate six dimensional invariants of the motion. In general, the approximate invariants will not be exact unless the motion is integrable, which in reality is rather unlikely. However, the quality of the approximate invariants, i.e. the maximum deviations from invariance, allows a direct estimate of the time it takes particles to get lost, which is a crucial parameter for the design of large scale accelerators like the SSC.

Because the deviations from invariance are very irregular, a conventional estimate of the maximum of the six dimensional deviation function is rather cumbersome and inaccurate. On the other hand, interval methods allow an exact and rather tight estimate of the maximum. Because of the oscillatory structure of the deviation function and the large number of variables, this represents a nontrivial task, which for the first time provides the missing link to allow a rigorous quantitative stability estimate.

All calculations are performed with the COSY language environment which provides an object oriented structured language as well as an executer. The compiler and executer are written in FORTRAN 77 for easy portability. The ability to use dedicated data types for automatic differentiation and interval arithmetic drastically simplifies the practical realization of the concepts discussed above.

Performance of an Occam/Transputer Implementation of Interval Arithmetic

O. Caprani and K. Madsen

University of Aarhus
and
Institute for Numerical Analysis, Tech. Univ.
Bldg. 303, DK-2800 Lyngby, Denmark

Rounded interval arithmetic is very easy to implement by means of directed rounding arithmetic operators. Such operators are available in the IEEE floating point arithmetic of the INMOS T800 transputer. When a few small pieces of assembly language code are used to access the directed rounding operators, the four basic rounded interval arithmetic operators can easily be expressed in the programming language Occam.

The performance of this implementation is assessed, and it is shown that the directed rounding floating point operations are no longer the time consuming part of the computations. Most of the time is spent with transport of operands to and from the on-chip floating point unit, and by the procedure call and parameter passing overhead.

Based on this experience, the implementation can be improved. The improved implementation runs at 0.15 MIOPS (Million Operations Per Second), or 0.30 MFLOPS. Furthermore, it is demonstrated that an advanced interval language compiler may provide a performance of 0.30 MIOPS or 0.59 MFLOPS on the transputer.

Use of a Real-Valued Local Minimum in Parallel Interval Global Optimization

O. Caprani and K. Madsen

University of Aarhus
and
Institute for Numerical Analysis, Tech. Univ.
Bldg. 303, DK-2800 Lyngby, Denmark

We discuss a parallel method for finding the global minimum (and all of the global minimizers) of a continuous non-linear function $f : D \rightarrow \mathbb{R}$, where D is an n-dimensional interval, $D \in \mathbb{IR}^n$. The method is based o the well known methods of Skelboe, Moore and Hansen.

Initially, we use a standard non-linear optimization method to find a local minimizer $\mathbf{x}_p$ (or rather: a prediction of a local minimizer). Then the interval method is used either to verify that the point found by the standard method is the global minimizer, or to detect the opposite and then find the global minimum. In case the standard method has solved the problem posed, i.e. if $\mathbf{x}_p$ is the global minimizer, the verification process is significantly faster than application of the usual interval method.

The new interval method starts by applying the interval Newton method to an interval I_p containing $\mathbf{x}_p$ as its midpoint. I_p is chosen as large as possible under the restriction that the interval Newton method must converge when I_p is used as starting interval. In this way the original problem is reduced to the problem of searching a domain which does not contain the local (and perhaps global) minimizer. This domain is searched by a method similar to the standard interval method, starting by splitting it into $2n$ intervals and hence avoiding I_p.

The new method parallizes very well, and experiments on distributed memory systems with up to 32 processors are reported.

Interval Arithmetic in Maple

A. Connell and R. M. Corless

Department of Applied Mathematics
University of Western Ontario
London, Ontario Canada N6A 5B7

We describe an implementation in the computer algebra language Maple of the draft Basic Interval Arithmetic Subroutines library proposed by George Corliss.

Maple provides a variable-precision numerical environment, in that one can, at any time, set the global (environment) variable Digits to the desired precision. The only limits to Digits are the memory available on the machine. Maple's built in arithmetic and elementary functions are of high quality and efficiency (for software arithmetic). This implementation of interval arithmetic is also variable-precision.

Besides the normal operations of finite interval arithmetic, we allow some manipulations with the symbol "infinity", which is taken to mean positive real infinity by this package, and the symbol "FAIL" which indicates a failure, such as occurs when subtracting infinities. We have not yet implemented "smart" interval arithmetic operators which take into account derivative information.

We were unable to overload the operators $+$, $-$, $*$, and /, but were able to use Maple's so-called "inert" operators "&+", "&−", "&*", and "&/". This results in a package that is easier to use than one that uses pure procedure calls, but is less satisfactory than true operator overloading.

We will discuss the implementation and verification of the arithmetic and elementary functions, and present some simple applications. Finally we will discuss the potential for using the symbolic capabilities of Maple to improve the tightness and efficiency of this package.

Automatic Differentiation Applications

G. F. Corliss

Marquette University and Argonne National Labs
Department of Mathematics, Statistics, and Computer Science
Milwaukee, WI 53233 USA

Automatic differentiation is a technique for the fast, accurate computations of ordinary and partial derivatives required by many self-validating algorithms. Given a program for the function to be differentiated, we nominate independent and dependent variables, and we produce a program that computes the desired derivatives. The ability to compute accurate derivatives efficiently is a critical enabling technology for optimization, nonlinear systems, continuation methods, stiff odes, and sensitivity analysis. Argonne has developed two complementary software tools: ADOL–C and ADIFOR.

ADOL–C is a very flexible tool using operator overloading in C++. It can be used to generate derivatives in either the forward or the reverse modes, Taylor series, or mixed partial derivatives of any order. ADOL–C can use a checkpointing scheme to restrict storage demands to grow only logarithmically in the execution time of the original function.

We describe the application of ADOL–C to two significant applications. The first is a model under development at Sandia of unsaturated flow in a porous medium. The second application of ADOL–C was to a new algorithm using Padé series in an [illegible] implicit way to solve stiff ordinary differential equations. ADOL–C computes the necessary high-order Taylor series and also derivatives of the Taylor coefficients with respect to the first term, thereby enabling the exploration of a completely new class of algorithms. Initial explorations suggest that the new algorithms are very competitive with conventional methods in both speed and accuracy.

The second tool for automatic differentiation is ADIFOR, a Fortran-based source transformation tool. ADIFOR accepts Fortran code for the function to be differentiated and uses advanced compiler technology to generate portable Fortran code to compute the derivatives. We describe the handling of sparse Jacobians by matrix coloring or by sparse saxpy operations, partially separable functions, functions which are defined implicitly, second order derivatives, and functions which may not be differentiable.

P-Matrices and the Rank-One Matrix Polytope Regularity Problem

G. E. Coxson

Department of Electrical and Computer Engineering
University of Wisconsin-Madison
1415 Johnson Dr., Madison, Wisconsin 53706 USA

A real matrix polytope is a bounded affine mapping from a polytope in $\mathbb{R}^k$ to $\mathbb{R}^{n\times n}$. Any such mapping may be described in terms of a starting matrix and k independent additive matrix perturbations each in the form of a fixed matrix multiplied by a parameter restricted to a closed bounded interval. When each of the k perturbation matrices has rank one, the polytope is called a rank-one matrix polytope. In this paper, we assume the polytope domain is a hyperrectangle. Any interval matrix is a rank-one matrix polytope with hyperrectangular domain.

An important problem in the study of interval methods is that of testing necessary and sufficient conditions for robust nonsingularity, or regularity, of interval matrices. This paper shows that, starting from any given rank-one matrix polytope with a hyperrectangular domain, one can construct a matrix which is a P-matrix if and only if the matrix polytope is regular. A P-matrix is defined as a matrix whose principal minors are all positive.

One way to use this P-matrix formulation is to strengthen a computational complexity result of Rohn and Poljak. In a recent paper, they prove that the regularity problem for interval matrices with rank-one radius matrix is co-NP-complete. We show using the P-matrix formulation that the more general problem of testing regularity of rank-one matrix polytopes is co-NP-complete and, as a special case of this, the interval matrix regularity problem is co-NP-complete as well. An additional corollary is that the problem of testing whether a given matrix is a P-matrix is co-NP-complete. These results imply that unless P=NP, which is considered to be highly unlikely, any algorithms for these problems must exhibit worst-case exponential-time computational behavior.

Reliable Recursive Computation of Some Determinant Ratios

A. Cuyt and B. Verdonk

Dept Mathematics and Computer Science
Universiteit Antwerpen (UIA), Universiteitsplein 1
B-2610 Wilrijk-Antwerpen, Belgium

Let $\{S_n\}$ and $\{g_i(n)\}$ $(i \geq 1)$ be complex-valued sequences and consider the quotient of determinants

$$E_k(S_n) = \frac{\begin{vmatrix} S_n & \dots & S_{n+k} \\ g_1(n) & \dots & g_1(n+k) \\ \vdots & & \vdots \\ g_k(n) & \dots & g_k(n+k) \end{vmatrix}}{\begin{vmatrix} 1 & \dots & 1 \\ g_1(n) & \dots & g_1(n+k) \\ \vdots & & \vdots \\ g_k(n) & \dots & g_k(n+k) \end{vmatrix}}$$

The quantities $E_k(S_n)$ play a crucial role in many numerical computations. We refer to convergence acceleration methods based on composite sequence transformations [BREZa], univariate and multivariate approximation and interpolation formulas [CUYT, MUHL] and many other applications. It is therefore essential that reliable algorithms be developed for the recursive computation of the values $E_k(S_n)$. The recursive computation schemes available at this time often exhibit unstable numerical behaviour due to breakdown or near-breakdown of the algorithm. Some singular rules have been developed [ALCU, BREZb] to handle the situation where the denominator vanishes theoretically. Whether or not these rules should also be applied in case of near-breakdown of the algorithm is difficult to determine. The aim is to develop algorithms which yield the values $E_k(S_n)$ with guaranteed accuracy based on a combination of theoretical results and scientific computation tools.

References

[ALCU] Allouche H. and Cuyt A., *A recursive computation scheme for singular tables of multivariate rational interpolants*, Numer. Algor. (to appear).

[BREZa] Brezinski C., *A general extrapolation algorithm*, Numer. Math. **35** (1980), 175–187.

[BREZb] Brezinski C., *Other manifestations of the Schur complement*, Lin. ALg. Appl. **111** (1988), 231–247. Numer. Math. **35** (1980), 175–187.

[CUYT] Cuyt A., *A recursive computation scheme for multivariate rational interpolants*, SIAM J. Num. Anal. **24** (1987), 228–238.

[MUHL] Mühlbach G., *The general Neville-Aitken-Algorithm and some applications*, Numer. Math. **31** (1978), 97–110.

Interval Methods via a Posteriori Error Estimates

B.S. Dobronec

Computing Center, Academy of Sciences, Siberian Branch, Akademgorodok, Krasnoyarsk, Russia

Interval methods for solving ODE's and PDE'S are presented in this paper. The methods are based on a posteriori error estimates.

The proposed methods are as follows: first, solve a given problem and some additional problems by some numerical methods, then use splines to interpolate the resulting numerical solutions. An interval solution consisting of a linear combination of these splines is then sought.

Interval analysis is applied here only for computing defects of the spline solutions and the necessary constants. The technique allows the computation time to be considerably reduced and the precision to be improved.

The advantage of this approach is that known numerical methods for solving ODE's, PDE's and spline approximations are used for its realization. This approach is applied here in particular for "stiff" ODE's, boundary-value problems for higher order equations with a small parameter, elliptic problems, and two-dimensional parabolic equations. An illustrative numerical example is given.

References

1. B.S. Dobronec and V.V. Shaidurov, *Two-Sided Numerical Methods*, Nauka, 1990, 208 p. (in Russian)
2. B.S. Dobronec, Two-sisded solution of ODEs via a posteriori error estimates, *J. CAM* **23** (1988), 53–61.

On Clustering of Boxes in Multidimensional Global Optimization

K. Du and R. B. Kearfott

Dept. of Mathematics, Univ. of Southwestern Louisiana
U.S.L. Box 4-1010, Lafayette, LA 70504-1010 USA

We consider branch and bound methods for enclosing all global minimizers of a nonlinear C^2 or C^1 objective function. In particular, we consider bounds obtained with interval arithmetic, along with the "midpoint test," but no acceleration procedures. Unless the lower bound is exact, the algorithm without acceleration procedures in general gives an undesirable cluster of boxes around each minimizer. Here, we analyze this problem in the multidimensional case, by generalizing arguments we previously presented for the one-dimensional case. Theoretical results are given which show that the problem is highly related to the behavior of the objective function near the global minimizers and to the order of the corresponding interval extension. We make various simplifying assumptions to be able to present a clear analysis. For example, we do not consider non-isolated minima, the effect of interval Newton methods, or minima which occur on boundaries of constructed sub-boxes.

A Computer Test of Interval Matrix Asymptotic Stability

I. V. Dugarova

Department of Applied Mathematics and Cybernetics
University of Tomsk
36, Lenina str., Tomsk, 634050, Russia

The interval matrix stability problem, important in practice, is discussed in [1]-[6]. It is important to create simple computer methods to test such a property for an interval matrix. Since it is difficult to simultaneously study necessary and sufficient conditions for interval matrix stability, most papers are devoted only to the development of a sufficient criterion for stability or instability [1]-[6]. The sufficient condition test considered in the present report is based on Gershgorin's theorem as well as on methods that have been worked out in [1] and [6]. However, the proposed algorithm is distinct from [1]-[3], and is efficient not only for interval matrices with negative diagonal elements. Its advantage is illustrated with numerical examples. Furthermore, as in the papers [3] and [4], another simple sufficient condition for interval matrix instability is presented.

References

1. Heinen, J.A. Sufficient conditions for stability of interval matrices. *Int. J. Control* **39** 6, pp. 1323–1328 (1984)
2. Xu, Daoyi. Simple criteria for stability of interval matrices. *Int. J. Control* **41** 1, pp. 289–295 (1985)
3. Xu, Daoyi. Stability criteria for interval matrices. *Comput. and Comb. Methods in System Theory* Amsterdam: Elsevier Science Publishers B.V., 1986.
4. Ziao, X.-X. and Qian, L.-L. Some new results for stability of interval matrices. *Contr. Theory and Adv. Techn.* 4 2 (1988)
5. Yedavally, R.K. Stability analysis of interval matrices: another sufficient condition. Int. J. Control **43** 3, pp. 767–772 (1986)
6. Argoun, M.B. On the sufficient conditions for the stability of interval matrices. *Int. J. Control* **44** 5, pp. 1245–1251 (1986)

The VPI Software Package

J. Ely

Mathematical Sciences Dept.,
Lewis and Clark College,Portland, OR 97219

The VPI (Variable Precision Interval) software package is a collection of routines written in C++ (by this author) to support variable precision interval arithmetic. It appears to be the oldest of the various C++ packages, having been used as early as 1988, although it has endured many modifications and enhancements since then. Here, the author discusses its capabilities, flaws, evolution (including the development of a vectorized version for the Cray YMP), and a variety of pedagogical and research applications to which the author has put it.

On Ways of Obtaining Guaranteed Solutions for a System of Ordinary Differential Equations

O.B. Ermakov

Department of Mechanics and Mathematics
Computer Center, Saratov State University
Astrakhanskaya 83, Saratov, Russia 410071

In the paper, ways of finding guaranteed solutions for a system of ordinary differential equations are considered. These ways are based on the extrapolation-interpolation Adams methods of any orders and on Taylor series. The methods account for the errors of input data, the errors of method and the rounding errors. To implement the first method for obtaining guaranteed solutions, finite differences and recurrence relations for the coefficients of the Adams formula are used. Both for the second method and for obtaining estimates of errors in the methods, we apply the idea, proposed by R. E. Moore, dealing with recursive computation of Taylor coefficients. This approach allows us to compute the Taylor coefficients very economically. In the realization of these methods, we use capabilities of the PASCAL-XSC compiler (high-accuracy arithmetic, computation of expressions with maximum precision) to take rounding errors into account; doing so will give high-accuracy guaranteed estimates solution of initial problem. A comparision of accuracy and execution time of these methods will be presented.

A Unified Approach to Real Numbers and Intervals

M. H. Escardó, D. M. Claudio and B. R. Tavares Franciosi
CPGCC da UFRGS, Porto Alegre, Brasil

Intervals are constructed by a generalization of Dedekind's construction of real numbers, in such a way that real numbers are particular cases of intervals. Intervals are defined neither as pairs of reals nor as (closed) sets of reals. Instead, intervals are defined as incompletely specified reals, and hence we call them *partial reals* also. Reals are then obtained from complete specifications. Specifically, reals and intervals are constructed by (possibly partial) functions $\alpha : \mathbf{Q} \rightarrow \{0, 1\}$ subject to certain conditions. Intuitively, the rational numbers p such that $\alpha(p) = 0$ are the lower bounds of the real number or interval defined by α, and the rational numbers q such that $\alpha(q) = 1$ are the upper bounds of the real number or interval defined by α. Then α defines a proper interval if α is undefined for some rational numbers, and defines a real number otherwise. The set of partial reals obtained in such a way turns out to be a Scott domain ordered by graph inclusion of functions. In particular, the bottom element of the domain of partial reals is the totally undefined function, which can be written $[-\infty, +\infty]$. Then this domain is also a topological space with Scott topology. It turns out that the Scott topology restricted to real numbers is the usual Hausdorff topology for real numbers. This fact is important because the computable functions on the domain of partial reals must be continuous in the Scott topology in order to be computable. Another way of stating this property is to say that a function on real numbers has a Scott-continuous extension to intervals iff it is Hausdorff-continuous. Moreover, every extension is monotonic, and there is a unique maximal extension, which corresponds to the so-called interval extension. This work also develops the arithmetic for partial reals, in such a way that the arithmetic for reals is a particular case of the arithmetic for partial reals. This unified construction of reals and intervals allows also to develop a unified theory of computability, and to relate real analysis and computability. A partial real defined by α is computable if α itself is computable as a function. An operation on real numbers is then a higher-order function. Scott domain theory allows such higher-order functions, and defines the notion of computability for them. It turns out that the four arithmetic operations and several other functions are computable.

Generalization of the Oettli-Prager-Theorem

H. Fischer

Technische Universität München
Institut fur Angewandte Mathematik und Statistik
Arcisstr.21, 8000 München 2, Germany

The classical Oettli–Prager theorem deals with a system $Ax = b$ of linear equations. It allows to check whether or not an approximate solution is acceptable within prescribed tolerances (intervals) for the coefficient matrix A and the right-hand side b. This theorem can be generalized in two respects: We replace intervals by more general regions (hypernorm-balls). And we replace the matrix A by an operator A that maps a linear space X into a linear space Y, where the dimensions of X and Y are arbitrary. Thus the new theorem can be applied for instance to integral equations.

Asynchronous Interval Iterations

A. Frommer

Fachbereich 7 Mathematik, Bergische Universität Wuppertal
Gaußstraße 20, D-5600 Wuppertal 1, Germany

A large class of methods for enclosing the solution x^* of a real (non-interval) problem, for example an $n \times n$ linear system, result in an iteration,

$$x^{k+1} = Hx^k, k = 0, 1, ..., \qquad (1)$$

where

$$H : \mathbb{IR}^n \to \mathbb{IR}^n,\ \mathbb{IR}^n : \text{space of all interval vectors of dimension } n.$$

Typically, H is an inclusion isotone interval extension of a continuous real operator that admits x^* as its unique fixed point. Thus, if we have an initial interval vector x^0 such that $Hx^0 \subseteq x^0$, Brouwer's fixed point theorem shows that $x^* \in x^0$ and, by inclusion isotonicity, $x^* \in x^k$ for all k.

Instead of the synchronous iteration (1) we propose to consider an *asynchronous* modification, where at a given stage only some components of H are evaluated at a vector whose components are made up of various results of previous stages. This kind of iterative scheme arises very naturally on parallel computers if one tries to minimize delays due to communication and/or synchronization. Although these *asynchronous* iterations are much more general than (1), it turns out that in the interval case its convergence properties are quite the same as for the synchronous iteration. This holds in particular for the standard situation described at the end of the previous paragraph.

The lecture will firstly present some results of the theory. It will then focus on several numerical examples on local memory as well as shared memory parallel computers, which compare the practical performance of synchronous and asynchronous methods. We are using a (non-optimal) interval arithmetic which delivers guaranteed bounds for the desired solutions.

The Bernsteinalgorithm with Subdivision

J. Garloff
Fachhochschule Konstanz, Fachbereich Informatik
Postfach 10 05 43, D-7750 Konstanz, Germany

Upper and lower bounds for the range $[\underline{m}, \overline{m}]$ of a multivariate polynomial

$$p(x_1, \ldots, x_n) = \sum_{i_1,\ldots,i_n=0}^{m} a_{i_1\ldots i_n} x^{i_1\ldots i_n}$$

on an n-dimensional interval vector X can be obtained by expanding the polynomial into Bernstein polynomials. For $n = 2$ and $k \geq m$ the expansion is

$$p(x,y) = \sum_{i,j=0}^{k} b_{ij}^{(k)} \cdot p_{ij}^{(k)}(x,y), \qquad (x,y) \in X$$

where

$$p_{ij}^{(k)}(x,y) = \binom{k}{i}\binom{k}{j} x^i(1-x)^{k-i} y^j (1-y)^{k-j}$$

$$b_{ij}^{(k)} = \sum_{s=0}^{i}\sum_{t=0}^{j} \left[\binom{k}{s}\binom{k}{t}\right]^{-1} \binom{i}{s}\binom{j}{t} a_{st} \qquad (a_{st} = 0 \text{ for } s > m \text{ or } t > m).$$

Then for all $k \geq m$ the following relations hold

(1) $$\min_{i,j} b_{ij}^{(k)} \leq \underline{m}$$

(2) $$\max_{i,j} b_{ij}^{(k)} \geq \overline{m}$$

with equality in (1) (resp. (2)) iff $\min b_{ij}^{(k)}$ (resp. $\max b_{ij}^{(k)}$) is assumed at one of the four Bernstein coefficients $b_{00}^{(k)}$, $b_{k0}^{(k)}$, $b_{0k}^{(k)}$, $b_{kk}^{(k)}$.

In this talk we show that degree elevation (raising k) is inferior to subdivision, i.e. subdivide X and calculate the Bernstein coefficients b_{ij} on the subintervalvectors. We concentrate on how the Bernstein coefficients on the subintervalvectors can be computed in a very economical way from the Bernstein coefficients on X.

Then we report on how the Bernstein algorithm may be applied to solve problems of robust control, e.g., checking (Hurwitz) stability of a polynomial with coefficients depending multilinearly on parameters varying in intervals.

The Proof Theorems on a Computer: Interval-Analytic Aspects

N. Glazunov

Glushkov Institute of Cybernetics, AN Ukraine
Prospect Glushkova 40, Kiev 252207 Ukraine

Let $f = f(x, y)$ be a sufficiently smooth real function of two variables. An interval-analytic method for proving theorems about properties of such functions f is discussed, and examples of its applications are given. The main example of application of this method is a proof of the famous Minkowski conjecture concerning the critical determinant of the region $|x|^p + |y|^p < 1, p > 1$ [1]. We consider the Minkowski conjecture in analytical form, which is proposed by Davis [2].

I discuss the following aspects of the method and its applications:
- common description of a method;
- interval-analytic strategies for proving conjectures [3-5];
- choice of domains and intervals;
- parallel computations;
- interval-analytical investigation of singularities;
- software for computer support of the method [6];
- interval-analytic computations on computers;
- another applications, and conclusions.

References

1. Minkowski, H. *Diophantische Approximationen*, Leipzig: Teubner, 1907.
2. Davis C. S. "Notes on a conjecture by Minkowski," *J. London Math. Sci.* **23** (1984), pp. 172–175.
3. Glazunov, N. M. and Malyshev, A. V. "On Minkovski's conjecture about the critical determinant," *Kibernetica* **5** (1985) pp. 10–14.
4. Glazunov N. M. and Malyshev A. V. "The proof of Minkovski's conjecture concerning the critical determinant of the region $|x|^p + |y|^p < 1$ near $p = 2$," *Doklady Akad. Nauk Ukr. SSR*, ser. A. **7** (1986) pp. 9–12.
5. Glazunov, N. M., Golovanov, A. S. and Malyshev, A. V. "Proof of the Minkovski hypothesis about the critical determinant of the $|x|^p + |y|^p < 1$ domain, *Research in the number theory.* **9**, Notes of scientific seminars of LOMI. Vol. 151. Leningrad: Nauka. 1986. pp. 40–53.
6. Glazunov, N. M. "Program Package TCHAI for Functional Research. Evaluation of Real Functions in Rectangular Regions / V. M. Glushkov Inst. of Cyb. Ac. Sci. Ukr. SSR. - Kiev, 1989. -67p.- Deposited in VINITI.

Interval Non-smooth Optimization

M. Gutowski

Institute of Physics, ON-3.2, Polish Academy of Sciences
Al. Lotnikow 32/46, PL-02-668 Warsaw, Poland

Objective functions of many variables arising from various practical problems of experimental data fitting often contain non-smooth terms like absolute value or $\max\{0, f(x)\}$. Non-smooth objective functions are hard to maximize using classical methods. Interval methods, besides Monte Carlo type methods, are among the few capable to find a global extremum in such cases [1]. Unfortunately, there is only a very limited choice of powerful accelerating devices when dealing with non-smooth problems. This is the reason for the rather large number of boxes, which possibly contain the desired solution, being generated during the calculation course. In this contribution we want to present some practical experiences with Ichida and Fujii's algorithm [2] modified in such a way as to lower the chance of dealing with a large number of boxes. There are two important points in our "upgrade". The first is to adopt the proper strategy of selecting a box to be bisected as the next one, without losing convergence properties of the algorithm. The second idea is to check from time to time (when "out of memory" error might be unavoidable) whether the boxes cannot be merged into one or more clusters, thus effectively lowering the number of objects to be dealt with. The method works satisfactorily when the number of unknowns is small.

References

1. H. Ratschek and J. Rokne. "Interval Tools for Global Optimization," *Computers Math. Applic.* **21** 6/7, pp. 41–50 (1991).
2. K. Ichida, Y. Fujii. "An Interval Arithmetic Method of Global Optimization", *Computing*, **23**, pp. 85–97 (1979).

Efficient Solution of Large Systems of Nonlinear Constraints with Inexact Data and Explicit Termination Criteria

G. D. Hager

Department of Computer Science
P.O. Box 2158 Yale Station
Yale University, New Haven, CT 06520

Many robotics applications require propositional decisions to be computed from noisy geometric sensor data. In structured situations, this problem is usually posed by fitting a parametric surface or volumetric model to the data, and then choosing the action appropriate for the resulting parametric description. The equations describing the models are usually highly nonlinear as are the criteria used to choose the appropriate action. In those cases where the errors in sensor data are bounded, making a decision from sensor data can be posed as solving a large set of nonlinear equalities and inequalities until given termination criteria are met.

A typical example is the problem of deciding the graspability of an object based on its shape and size. Objects are observed by a range scanner that delivers a dense (one quarter million bytes) range "image." Objects are described by a *superellipsoid* (an ellipsoid where the exponents are allowed to range from 2 to ∞) augmented with deformations such as bending and twisting. The graspability criteria are described by inequalities on the size and shape parameters of the superellipsoid.

In general, this class of problems leads to large sets of nonlinear equalities and inequalities with the following general properties:

- The set of equations are highly overdetermined. For example, in the case above, 10 to 15 parameters are determined using over 1000 constraint equations.
- The precision to which model parameters can be computed is ultimately governed by the precision of the sensor data which is bounded, but typically several orders of magnitude worse than machine precision. The precision to which the equations can be solved also varies widely based on the density or completeness of the data delivered.
- Model parameters need only be solved to the precision required to reach a specific decision. The constraints describing decision criteria are typically much simpler than the constraints describing the data model.
- There are occasionally data *outliers* that could potentially render the system inconsistent if countermeasures are not taken.

We have built an interval-based bisection algorithm to solve these types of problems. Some interesting aspects of this algorithm are:

- Because of the relatively large data errors and the potential for outliers, we found that Newton-style methods performed far worse than a procedure that uses direct interval-based constraint testing.
- The structure of the procedure makes it possible to prove that, for n parameters, only $2n$ data constraints are relevant to an interval. By implementing a relatively inexpensive selection procedure, it is possible to significantly speed up the constraint solving process. Given this result, it follows that the solution

procedure used by the bisection algorithm can be performed in constant time using $6n^2$ processors operating in parallel.

- The structure of the decision constraints are used to guide the bisection process so that the computation needed to reach a decision is kept to the minimum. Furthermore, the amount of computation performed varies adaptively based on the character of the data and the system of constraints.

We have generally found that interval-based algorithms are an extremely flexible basis for solving these types of problems.

The full paper will describe the basic structure of the bisection algorithm, detail the selection procedure mentioned above, and will also include some experimental results from the grasping example given above. We will also briefly describe the extensions required to use interval-based constraint solving in less structured situations.

Solving Systems of Nonlinear Equations Using Generalized Interval Arithmetic

E. R. Hansen

654 Paco Drive, Los Altos, CA 94022, USA

In an earlier paper [E. R. Hansen, A Generalized Interval Arithmetic, pages 7-18 of K. Nickel (ed.). Interval Mathematics, Springer-Verlag Lecture Notes in Computer Science no. 29, 1975.] the author introduced a generalization of interval arithmetic. It was designed to reduce the growth of intervals which occurs during computation because of dependency. In this paper, we consider application of this arithmetic to solving systems of nonlinear equations.

Estimations and Enclosures of Condition Numbers for Evaluation Algorithms

G. Heindl

Fachbereich Mathematik und Institut für Angewandte Informatik
Bergische Universität, Gauss-Straße 20
D5600 Wuppertal 1, Germany

It is shown that the problem of estimating and enclosing condition numbers for evaluation algorithms can be reduced to the problem of estimating and enclosing the determinants of certain Hessenberg matrices. There are several methods for solving the reduced problem, and any of them can be viewed as a special realization of automatic differentiation.

On Bounding the Range of Some Elementry Functions in Fortran 77

C. Hu, A. Awad and R. B. Kearfott

Department of Computer and Mathematical Sciences
University of Houston-Downtown
Houston, TX 77002
and
University of Southwestern Louisiana

We present Fortran 77 programs to obtain the range of the arcsine, arccosine, arctangent, arccotangent, hyperbolic sine, hyperbolic cosine, hyperbolic tangent and hyperbolic cotangent functions over intervals. These programs are to be part of the software library INTLIB.

Algorithms and test data will be discussed. We will also present the techniques used to narrow the result interval and to speed up the rate of convergence.

On the Solution of Volterra-type Equations with Preassigned Accuracy

V. V. Ivanov

(1) General theory of complete error estimation for the solution of applied problems on computers is briefly considered.
(2) Some conjectures by N. S. Bakhvalov about properties of optimal algorithms are formulated.
(3) General results and validity of the conjectures are illustrated with the solution of Volterra-type equations as the main example.
(4) Software for the solution of applied problems on computers with automatic result verification is briefly described.
(5) Some industrial and scientific applications are given.

References

1. V. M. Glushkov, V. V. Ivanov and V. M. Yanenko, *Developing Systems Modeling*, Moscow: Nauka, 1983, 352 p. (in Russian).
2. V. V. Ivanov, *Methods of Computation on Computers.* Guide book, Kiev: Naukova dumka, 1986, 584 p. (in Russian).
3. V. V. Ivanov et al., *Package of Routines POM-1* (in Russian), Acad. of Sci., Ukr, V. M. Glushkov Inst. of Cybern., Kiev: 1986, 1200 p. (dep. in GOS FAP SSSR, N 50860000156).
4. V. V. Ivanov, Ju. G. Tvalodse and A. Sh. Zhuzhunashvili, On the solution of Volterra-type integral equations with a preassigned accuracy (in Russian), Acad. of Sci., GSSR, N. I. Mushelishvili Institute of Computing mathematics, Tbilisi: 1989, 31 p. (dep. in VINITI23.03.89, N 133-B-90).
5. V. V. Ivanov, Systems development simulation problems and C. Carathéodory's concepts, in the book: *Constantin Carathéodory: International Tribute*, Ed. Th. M. Rassias, World Sci. Syngapur, New Jersey, London, Hong Kong, Vol. 1, 1991, pp. 501–526.

Correct Implementation of Floating Point Algorithms

K.-U. Jahn
Fachbereich Mathematik und Informatik
der TH Leipzig, Karl-Liebknecht-Str. 132
O-7030 Leipzig, Germany

Properties on numerical algorithms are usually proved in idealized abstract spaces. The spaces occurring on a real computer are different from them. So the question arises how the behaviour and the computed results differ from the theoretical ones.

A calculus is introduced to get an answer to this question. The calculus is a generalization of Hoare's for verifying algorithms. For example, there will be given a rule for verifying while-loops when the occurring floating point numbers are replaced by sets of numbers which include the exact values. This rule has the same shape as its original, namely

$$\{I \wedge B\}S\{I\} \vdash \{I_\#\} \text{ while } B_+ \text{ do } \widetilde{S}\{I_\# \wedge \neg B_+\},$$

where $I_\#$ and B_+ are weak and strong generalizations of the loop invariant I and the loop condition B, respectively. $\widetilde{S}$ differs from S in replacing all not exactly computable resp. representable numbers by enclosing intervals.

The properties of the calculus will be proved, and applications to numerical examples will be given.

Distance Measuring and Computing Similarity Grades

K.-U. Jahn

Fachbereich Mathematik und Informatik
der TH Leipzig, Karl-Liebknecht-Str. 132
O-7030 Leipzig, Germany

For pairs (X, Y) of n-dimensional hyperrectangles X, Y and a wide class of metrics d, the corresponding Hausdorff distance $d^H(X, Y)$ can be effectively computed. This was shown in a paper by the author.

Now it will be shown that the computation can be extended to sets representable as finite unions of hyperrectangles, however for a restricted class of metrices. If only a two-sided approximation of the Hausdorff distance is required, which can be arbitrarily improved, then a method can be given which works for each metric.

Using the Hausdorff ~~distance for finite~~ unions of intervals, ~~their distance can~~ be computed effectively for arbitrary bounded sets. This will be applied for computing similarity as well as congruency grades of sets. Further applications to example-based learning and image recognition will be given.

Use of Interval Methods for Solving Minimax Problems

C. Jansson and O. Knüppel
Informatik III, Technische Universität Hamburg-Harburg
Eissendorfer Str. 38, 2100 Hamburg 90, Germany

Minimax problems arise in many practical applications in physics, chemistry, and engineering. For example, in multivariable control-system design and structural engineering, the minimization of important structural properties like robustness can be expressed as minimax problems.

We present a method which computes guaranteed lower and upper bounds for the global optimum value of minimax problems.

The numerical performance of our method is discussed on standard test examples and some real world problems.

Estimation of the Parameters of Nonlinear Models from Experimental Data via Interval Analysis

L. Jaulin and E. Walter

Laboratoire des Signaux et Systèmes, SUPELEC
Plateau de Moulon, 91192 Gif-sur-Yvette Cedex, France

Building mathematical models to understand, predict and control the behavior of a system is a basic activity in most fields of pure and applied science. Frequently, prior knowledge is not sufficient to allow a complete derivation of the model, and experimental data must be used to select a suitable model structure and estimate its unknown parameters.

The most classical approach for parameter estimation is to look for the value of the parameter vector that minimizes a criterion, usually the sum of the squares of the errors between the experimental data and corresponding model outputs. Except in very special cases, such as when the errors are affine in the parameters, no explicit formula is available to compute the optimal parameter vector, and iterative methods are used to compute displacements in the parameter space that improve the value of the criterion. Such an approach has two main drawbacks: (i) it is essentially local and does not guarantee convergence to the global optimum; (ii) it provides no reliable indication on the uncertainty with which the unknown parameters are estimated.

To solve problem (ii), we suggest in this paper to look for *the set of all models* that are *acceptable* instead of looking for *the model* that is *optimal* in the sense of a given criterion. The first step is then to list all the properties that the model should have to be acceptable. A first example of such acceptability conditions is that the residuals between the data and corresponding model outputs lie between some known bounds that express the confidence intervals attached to individual measurements (see e.g. [1] and all other papers of the same special issue). One can also consider that errors are acceptable if the probability that they correspond to realizations of some random variable with known probability distribution is higher than some prior threshold. Other conditions not directly related to the errors can be considered as well and several acceptability conditions may have to be met simultaneously for the model to be acceptable.

Once conditions of acceptability have been defined, one is interested in characterizing the set S of *all* values of the parameter vectors such that the model belongs to set of acceptable models. This is a problem of set inversion that needs be solved globally in order to avoid problem (i). In this paper we shall present a new algorithm, based on interval analysis, that makes it possible to approach S as precisely as desired.

The approach will be illustrated with several examples.

Reference

[1] E. Walter and H. Piet-Lahanier, Estimation of parameter bounds from bounded-error data: a survey. *Mathematics and Computers in Simulation*, **32**, 449-468, 1990.

Use of Interval Arithmetic in Symbolic Computation

J. R. Johnson[1] and W. Krandick[2]

Department of Mathematics and Computer Science
Drexel Univesity, Philadelphia, Pennsylvania 19104 USA
and
Research Institute for Symbolic Computation
Johannes Kepler University, A-4040 Linz, Austria

This project explores the use of interval arithmetic in symbolic computation. Interval arithmetic, a formalism for computing with ranges of numbers, provides an efficient and error free method of combining floating point arithmetic with exact symbolic computation. The primary goal of this project is to develop and analyze exact algebraic algorithms that use approximate interval arithmetic for certain intermediate computations yet still produce guaranteed results. The purpose of such hybrid algorithms is to obtain a practical efficiency gain to time consuming algebraic algorithms such as Collins' Cylindrical Algebraic Decomposition (CAD) based Quantifier Elimination. The CAD algorithm is a powerful method for proving mathematical theorems involving polynomial equations and inequalities. Reduced execution times will allow more complicated and interesting problems to be solved.

The correctness of the CAD algorithm relies upon correctly determining the signs of real algebraic numbers and correctly determining the number of real roots of polynomials with integer and real algebraic number coefficients. This can be accomplished with time consuming exact computation with polynomials and real algebraic numbers. However, in many situations the desired information can be obtained with a verifiable approximate computation. For example, if a real algebraic number is contained in an interval which does not contain zero then its sign can easily be determined from the endpoints of the interval.

There are many algebraic algorithms which can obtain useful information from rough interval bounds and hence benefit from judicious use of interval arithmetic. Example hybrid algorithms include Lehmer's multiprecision integer GCD algorithm, real algebraic number sign computation, algebraic number inequality tests, polynomial root bound computation, polynomial root isolation and refinement, and approximate algebraic polynomial GCD calculation. Preliminary study of some of these algorithms shows the potential merits of this approach.

We will report on the design of an arbitrary precision interval arithmetic package, intended to be used in conjunction with exact algebraic algorithms, and our experience using the package in the development of hybrid interval-symbolic algorithms. The interval package also includes algorithms for exact interval arithmetic with rational endpoints. The goal of the hybrid algorithms is to replace time consuming exact computation, where possible, by approximate interval computation in such a way that the exact results can be recovered. Interval arithmetic is performed with increasing amounts of precision, and only when interval arithmetic fails to provide the desired information, do we resort to exact computation. We will illustrate some techniques for determining the amount of precision necessary

[1]Supported by NSF grant CCR-9211016
[2]Supported by Austrian Science Foundation (Grant M022-PHY)

for certain computations, for systematically increasing the precision, and for successfully combining both interval and exact computation. Empirical results will be presented showing the performance improvement over exact algorithms and the effectiveness of interval arithmetic to provide exact information.

An Interval Step Control for Continuation Methods

R. B. Kearfott and Zh. Xing

Dept. of Mathematics, Univ. of Southwestern Louisiana
U.S.L. Box 4-1010, Lafayette, LA 70504-1010 USA

We present a step control for continuation methods that is deterministic in the sense that (i) it computationally but rigorously verifies that the corrector iteration will converge to a point on the same curve as the previous point (i.e. the predictor / corrector iteration will never jump across paths),and (ii) each predictor step is as large as possible, subject to verification that the curve is unique with the given interval extension. We propose two different interval step controls to guarantee rigorousness. Those are: interval Gauss-seidel iteration for a general n-dimensional continuation problem and interval arithmetic function value checking for a 2-dimensional problem, which greatly increases efficiency. Several theorems guarantee that our algorithms are appropriate, and can not fail (under some assumptions). We present performance data and comparisons with a non interval step control method (PITCON version 6.1). A comparison of plots obtained from both step controls reveals that a non interval step control will behave erratically in situations where the interval step control leads to orderly progression along the curve. Two-dimensional experiments, tests using the Topologist's sine curve, which has changes in curvature of increasing magnitude, suggest that this interval step control is very efficient when rapid changes in curvature occur, and that it is much faster than the non interval step control for such functions, assuming both step controls follow the curve successfully. We apply this interval step control in conjunction with slope arithmetic to CAD Geometric graphic design problems. It solves the problem of approximating the intersection curve of two patches with a guaranty of finding all branches of the intersection curves.

Efficient Iteration with Operation Decomposition

R. B. Kearfott and S.-F. Shi

Dept. of Mathematics, Univ. of Southwestern Louisiana
U.S.L. Box 4-1010, Lafayette, LA 70504-1010 USA

We have recently considered (in *Computing* **47**, pp. 169–191) decomposing a system of nonlinear equations by defining new variables corresponding to the intermediate results in the evaluation process. In this work (ibid.) we applied both a derivative-free component solution process and an interval Gauss–Seidel method to the larger, sparse system of equations so obtained.

A geometrical analysis of the component solution process indicates when convergence of the process is expected, without evaluating iteration matrices and without coordinate bisection. A separate analysis indicates when a linearized Gauss–Seidel step is necessary, and how to make it most effective. In this talk, we will present experimental results on an improved, efficient hybrid algorithm combining the component solution process with an occasional Gauss–Seidel step on a single component.

Interval Arithmetic with Mathematica

J. B. Keiper

Wolfram Research, Inc.
100 Trade Center Drive, Champaign, IL 61820 USA

The arbitrary-precision arithmetic in *Mathematica* displays results with as many digits are are known to be correct. This sort of arithmetic, sometimes referred to as significance arithmetic, has certain advantages over fixed-precision arithmetic. We show how interval arithmetic is the correct model to use for significance arithmetic and how this is done in *Mathematica.* As an implementation of interval arithmetic, significance arithmetic is limited to intervals that are short relative to their magnitude. For this reason *Mathematica* also provides correctly rounded interval arithmetic with the function `RealInterval[ ]`. We show some examples of how this works.

Interval Calculations Programs in the ASIAS Program Package

N. A. Khlebalin and A.V. Lazarev

Department of Automation of Manufacturing,
Moscow Steel and Alloys Institute,
Pervomayskaya 7, Elektrostal,
Moskowskaya oblast, 144000, Russia

Program package ASIAS is used to solve problems of analysis and synthesis of automatic systems with interval initial data (parameter values, initial conditions and characteristics of external disturbances). The distinctive property of the program package is that operations with interval values are performed in accordance with the rules of interval analysis. That is why the interval calculations programs are one of components of the software in the ASIAS program package. They work with the following interval mathematical objects: functions, linear algebraic equations systems, matrices, differential equations.

For interval functions, the package allows determinnation of the range and construction of the borderline functions (major-line and minor-line). For polynomials with pure- and functionally-interval coefficients, ASIAS can localize the sets of roots (oneconnected and multiconnected). For complex fractionally-rationally-interval functions ASIAS can determine the amplitude, phase, logarithmic and other characterictics.

For linear interval algebraic equation systems, ASIAS can determine the various sets related to solutions solutions (united, tolerable, guaranteed solution set).

Matrix algebra is presented by the following programs: derivation of the characteristic polynomial, derivation of the joined matrix, multiplication, raising to a power, calculation of the interval of the determinant values.

For interval differential equations the sets (pipes) of the solutions are constructed.

Most of the solution methods of these pure mathematical problems are not traditional. They use the results obtained in automatic control theory, more specifically in interval automatic systems theory, whose development has been progressing for about 15 years.

The package is written on FORTRAN for the IBM PC. The teaching version is provided with a course of lectures and effect drawing. Program package ASIAS is a developing system. At present, the interval calculations programs are complemented by symbolic operations and translated to the language PASCAL-XSC.

Reference

"Program package ASIAS (Analysis and Synthesis of Interval Automatic Systems)", *Interval Computations* **1** 2 (1991) pp. 83–84.

Renormalization for One Parameter Families of Hierarchical Models

H. Koch[3] and P. Wittwer
University of Texas at Austin, Department of Mathematics
Austin, TX 78712, USA
and
Université de Genève, Département de Physique Théorique
Genève, CH 1211 (Switzerland)

We present a detailed proof of a theorem which was announced at the conference on "Nonlinear Evolution and Chaotic Phenomena", held in Noto, Sicily, Italy. The central object of our study is a nonlinear operator, acting on a Banach space of analytic functions. An extension of interval arithmetics to such spaces is used to prove the existence and local uniqueness of a fixed point for this operator; and the bounds are carried out on a computer. The listings of our computer programs, as well as details of our proofs, are available.

On a certain Banach space $\mathcal{E}$ we consider the fixed point problem for the nonlinear operator $\mathcal{N}$, which is defined by the equation

$$(1) \qquad (\mathcal{N}f)(t^2) = \frac{1}{c}\int_{-\infty}^{\infty} ds \exp(-L^2 s^2)[f((\frac{1}{\alpha\ell}t + s)^2)]^{\ell}.$$

Here, the normalization constant $c = c(f)$ in (1) is chosen such that $(\mathcal{N}f)(0) = 1$, i.e., $c = \int ds \exp(-L^2 s^2)[f(s^2)]^{\ell}$; the other constants will be specified.

We consider not only the Banach space $\mathcal{E}$, but a family of such spaces $\mathcal{E}_\rho$. This is because, for the purpose of doing interval analysis, we represent $\mathcal{N}$ as the product of three operators: One which takes the ℓ–th power of a function, one which convolutes it with the Gaussian measure, and one that scales its argument by a factor $\frac{1}{\alpha\ell.}$ For the first two of these operators, if the domain is chosen to be an open subset of one of the spaces $\mathcal{E}_\rho$, then the range is no longer contained in the same Banach space.

Interest in this fixed point problem stems from the theory of critical phenomena in statistical mechanics and quantum field theory, where $\mathcal{N}$ describes the action of a "block spin renormalization group operator" on a space of hierarchical lattice models. Given the dimension d of the lattice, and an arbitrary integer $L > 1$ defining the block spins, the canonical values for the parameters ℓ and α are $\ell = L^d$ and $\alpha = L^{-\frac{d-2}{2}}$, respectively. In particular, for $d = 3$ (the most interesting case from the point of view of physics) and $L = 2$, one obtains $\ell = 8$ and $\alpha = 1/\sqrt{2}$. These are the values which we use in our analysis (actually, we only assume that α lies in a certain small interval containing $1/\sqrt{2}$).

[3]Supported by the National Science Foundation under Grant No. DMS–9103590.

Interval Methods in Electrical, Electronics and Control Engineering

L. Kolev

23 Tamarind Court, 4 Cheryl Road
Avondale, Harare, Zimbabwe

The paper will present an overview of technical problems arising in Electrical, Electronics and Control Engineering that have been solved using interval methods. More specifically, the following topics will be covered.

1. Tolerance analysis of steady states in linear electric circuits:
   a) direct current circuits and
   b) alternating current circuits.
2. Robust stability of linear electric circuits or control systems.
3. Robust performance of the same class of circuits and systems.
4. Transient analysis of the same class of circuits and systems.
5. Analysis of nonlinear circuits and systems
   a) determination of all steady-states:
      - Constant steady-states
      - T-periodic steady-states
   b) Uniqueness of T-periodic steady-states.

It is shown that the above technical problems can be transformed to corresponding mathematical problems: linear interval systems result from problem 1; global nonlinear optimization problems result from 1b, 2 and 3; ordinary differential equations result from problem 4; systems of nonlinear equations result from problem 5. The latter problems are then effectively solved using appropriate interval methods.

The paper will emphasize advantages of the existing interval methods over the traditional, non-interval methods, as well as some difficulties encountered in applying interval methods (mainly due to lack of interval software packages).

Improved Interval Algorithms for Tolerance and Robustness Analysis

L. Kolev

23 Tamarind Court, 4 Cheryl Road
Avondale, Harare, Zimbabwe

In this paper, it is shown that various engineering problems dealing with tolerance, robust stability and robust performance analysis of linear electric circuits or control systems can be reduced to solving corresponding global nonlinear minimization problems.

Depending on the nature of the engineering problem considered, two classes of minimization problems arise. The first class comprises minimization problems with simple constraints (the state-variable vector $x \in X$ where X is an interval vector). The second class of problems includes additional functional inequality or equality constraints. In both cases all the nonlinear functions involved are multivariate polynomials. Problems of the former type arise in robust stability analysis as well as in "worst-case" tolerance and performance analysis. Tolerance and performance analysis in statistical formulation or determination of the so-called stability margin lead to minimization problems of the latter type.

Several algorithms for tolerance and robustness analysis of linear electric circuits and control systems are suggested. They are based on first and second order interval methods for global minimization. The algorithms also incorporate various simple techniques (not fully exploited previously) which lead to substantial improvement of their convergence rate.

Numerical examples show that the overall efficiency of the presented algorithms is superior to that of other known interval algorithms (based on first-order minimization methods or the Bernstein method).

Efficient, Accurate, Exhaustive, and Robust Method for Solving System of Non-Linear Equations

P. Koparkar
National Centre for Software Technology
Gulmohar Road, Cross Road No. 9
Juhu, Bombay 400049, India

Various real-life objects along with their properties need to be modelled for the purpose of simulation and performance analysis using computers. The objects are modelled by certain mathematical (symbolic) expressions, while the properties are modelled by equations involving these expressions. In engineering applications, the solution sought to the system of equations has requirements different from those of a mathematical solution. Arriving at a solution in one form or another, amenable to further processing, is important. It is essential that the solution is computable on the machine and the method can be automated. The method used for this purpose should be efficient (involving minimal computations), accurate (numerically high precision), exhaustive (detecting all possible solutions) and robust (working without failures). In this paper we present such a method for solving the system of non-linear equations over a finite, bounded domain, when the functions involved are differentiable and their range evaluation is possible either by interval methods or by some other means. With this point of view, methods based on strategies such as numerical iterative refinement or recursive subdivision are more welcome.

Numerical iterative method refines a guess about the solution to arrive at a better one, and iterates this until the guess is acceptable. It offers no guarantee of convergence. Also, it may converge only to a particular solution ignoring others. Thus, it lacks exhaustiveness and robustness. However, such a method can quickly find the solution to the desired accuracy, whenever it converges.

On the other hand, recursive subdivision (bisection) method considers the entire domain at a time, and subdivides it and recurses when necessary. It concentrates only on those subdivided portions that show a potential to contain solutions. It exhausts all solutions in the domain and is robust. However, if more accuracy is desired, it becomes inefficient as the deeper subdivision increases the number of subdomains exponentially.

These two methods are unacceptable when the solutions are computed automatically on the machine, since neither of them shows all the desirable characteristics. The method proposed in this paper starts with the entire domain and applies subdivision to ensure exhaustiveness and robustness. Later it switches over to numerical iterative refinement for efficiency and accuracy. Switching takes place only when the convergence of the refinement method is guaranteed. A computable condition for this guarantee is worked out using range evaluation techniques and results from topology and mathematical analysis. The method devised is of generic nature in that it covers a large spectrum of applications. We conclude the paper with the discussion of implementation of the method on computer.

Towards a Verified Solution of Linear Systems Basing on Common Software Libraries

C. F. Korn, Ch. Ullrich

Institut für Informatik, Universität Basel
Mittlere Str. 142, CH-4056 Basel, Switzerland

In the past decades huge numerical software libraries have been developed. Linear systems have deserved a special interest which is reflected in the great amount of available packages. As a major drawback these provide just an approximation of the true solution, its accuracy is normally estimated by the means of "backward analysis". Examples show that these estimations are sometimes very precise, in other cases completely wrong, giving way to misleading interpretations of the computed solution. This problem is avoided when using automatic result verification. Our approach consists in reusing the existing software extending it by a verification capability. This is achieved by adopting following strict separation when computing the inclusion of a linear system:

1. approximation phase: an approximate solution is computed using the routines of existing software libraries.
2. verification phase: based on the delivered approximation an inclusion is computed.

Of course this separation is in principle possible for any numerical problem. This strategy has several consequences. First, the approximation is computed fast (assuming the used library is optimized). Since the first phase is absolutely independent of the second, precomputations for the verification step are not done, especially the exact (but in all known implementations inefficient) dot-product is not used. Second, the inclusion step is executed only if required by the user. Third, the interface of the verification routine must be adapted to the interface of the approximation routine.

The feasibility of this approach will be shown by means of the extension of one particular library. Runtime (and accuracy) measurements will show that the verification of the results can be achieved with low additional costs - even for linear systems of high dimension.

Semimorphic Function Evaluation

W. Kraemer

Univ. Karlsruhe, Kaiserstr. 12
D-7500 Karlsruhe 1, Germany

The construction of reliable computer programs for the approximation of mathematical functions will be discussed. A procedure will be described which allows the computation of floating-point approximations for $f(x)$ with maximum accuracy. This means, that for every valid floating-point argument x, the computed approximation $\tilde{f}(x)$ will be equal to the exact (real) value $f(x)$ rounded to the nearest floating-point number.

If the exact value is the midpoint of two adjacent floating-point numbers the floating-point result will be determined corresponding to an antisymmetric rounding (round$(-a) = -$round(a)).

This leads to:

$$\begin{array}{rcl} f(x) < f(y) & \Longrightarrow & \tilde{f}(x) \le \tilde{f}(y) \\ f(-x) = f(x) & \Longrightarrow & \tilde{f}(-x) = \tilde{f}(x) \\ f(-x) = -f(x) & \Longrightarrow & \tilde{f}(-x) = -\tilde{f}(x) \\ f(x) \text{is a flp-number} & \Longrightarrow & \tilde{f}(x) = f(x) \end{array}$$

In analogy, the computation of (best possible) directed rounded function approximations will be considered. Using such approximations, an optimal floating-point enclosure of the range

$$f(X) := \{f(x) \mid x \in X\}$$

of the function f over the interval X can be determined.

The main advantage of such routines is full *compatibility* on all computer platforms with identical data formats. The routines always produce exactly the same output and lead to portable software.

Standardization of the elementary mathematical functions would be a very desirable task. To require the functions to always deliver the best possible numerical results seems to be the most natural way to do this. The specification of the square root function in the IEEE-754 standard is the only one that already fulfills this requirements.

Around such implementations *reliable mathematics* can be built!

Global Optimal Solutions with Tolerances and Practical Composite Laminate Design

B. P. Kristinsdottir, Z. B. Zabinsky and T. Csendes

Industrial Engineering Dept., Univ. of Washington,
Seattle, Washington 98195 USA
and
Kalma'r Laboratory of Cybernetics, Jo'sef Attila University,
Arpad ter 2, P. O. Box 652, H-6701 Szeged, Hungary

An algorithm for finding a large feasible n-dimensional interval for constrained global optimization is presented. The resultant interval is iteratively enlarged about a seed point while maintaining feasibility. An interval subdivision method may be used to check feasibility of the growing box. An alternative method to check feasibility using a different global optimization algorithm is discussed. The resultant feasible interval is constrained to lie within a given level set, thus ensuring it is close to the optimum. It is proved that the algorithm converges in a finite number of iterations.

The ability to determine such a feasible interval is useful for exploring the neighborhood of the optimum, and can be practically used in manufacturing considerations. The algorithm is applied to a real life engineering design problem to construct manufacturing tolerances for an optimum design of composite materials.

Nonlinear Systems of Equations and Ordinary Differential Equations: Periodic Orbits of Conservative Systems and Non-Integrability

W. Kuhn

School of Mathematics, Georgia Institute of Technology
Atlanta, GA 30332

The paper presents a test for non-integrability of ordinary differential equations. It is based on periodic solutions and the Poincaré map. A standard approach to find a fixed point of the Poincaré map $P(x) = x$ is to use a Newton-like interval iteration. If there exists a first integral (or constant of motion), the Jacobi matrix of the Poincaré map becomes singular and the usual iteration schemes must fail. In other words: If the iteration can be applied successfully, then the ODE has no first integral. The problem now arises how to decide that there are no more than m already known first integrals.

A restatement of the problem is how we can use the singular equation $P(x) = x$ and the m integrals to establish a fixed point iteratively. We show how to construct a new Poincaré map on an $(n - m - 1)$ dimensional manifold (n is the dimension of the ODE), which is no longer singular and whose fixed point represents a periodic orbit. Also, a successful iteration shows that there are only m integrals.

The paper also shows how to work with the program, which does the verification automatically. It only requires the ODE and a (good) initial guess of the periodic orbit in a user-friendly symbolic form.

An application to the general 3-Body Problem is given. For this problem, $n = 12$ and $m = 4$.

Programming Environments for Scientific Computation with Verification: PASCAL-XSC and C-XSC

U. Kulisch

Institut f. Angewandte Mathematik, Univ. Karlsruhe
Kaiserstr. 12, D–7500 Karlsruhe 1, Germany

PASCAL-XSC is a PASCAL extension which significantly simplifies programming in the area of scientific and technical computing. PASCAL-XSC is an updated PASCAL which contains the full Standard PASCAL. It is immediately usable by PASCAL programmers. PASCAL-XSC is easy to learn and ideal for programming education.

In a quite natural way PASCAL-XSC provides a number of concepts which are vital in a contemporary programming environment: a module concept, an operator concept, functions and operators with general result type, overloading of functions, procedures and operators, dynamic arrays, subarray slices, a string concept, overloading of the assignment, of read and of write, and of others. Thus, modules and operators, for instance, for complex arithmetic, rational arithmetic, higher precision or multiple precision arithmetic, or interval arithmetic can easily be developed by the user. Arithmetic expressions and numerical algorithms come considerably closer to the usual mathematical notation. This simplifies programming essentially. Programs are easier to read and to write. They are easier to debug and therefore more reliable. Many programs can be read like a technical report.

C-XSC is a superset of the well-known language C. It is implemented as a numerical class library in the programming language C++. C-XSC may be interpreted and can be used as an arithmetic module extending the properties of the language C.

The programming environments PASCAL-XSC and C-XSC make the computer more powerful arithmetically. They provide a large number of predefined data types for elements of the most commonly used vector spaces such as real and complex numbers, vectors and matrices as well as real and complex intervals, interval vectors, and interval matrices. Operators for elements of these types are predefined and can be called by their usual operator symbol. Thus, PASCAL-XSC and C-XSC make the computer look like a vector processor to the programmer.

All predefined numerical operators are of highest accuracy which means that the computed result differs from the correct result by at most one rounding. 24 mathematical standard functions are provided with their generic names for real and complex arguments as well as for real and complex interval arguments. The computed values are of highest accuracy. While the emphasis in computing is traditionally on speed, in PASCAL-XSC and C-XSC the emphasis is on accuracy and reliability of results. The total time for solving a problem is the sum of the programming effort, the processing time, and the time for the interpretation of results. The XSC-languages reduce this sum considerably.

Numerical mathematics has devised algorithms which deliver highly accurate and automatically verified results by applying mathematical fixed-point theorems. This means that these computations carry their own accuracy control. However, their implementation requires suitable arithmetic support and powerful programming tools which were not previously available. The development of PASCAL-XSC and of C-XSC has aimed at providing these tools within the setting of PASCAL and of C, respectively. Both languages are based on an identical large arithmetic runtime system. PASCAL-XSC and C-XSC are particularly suited for the development of numerical algorithms that deliver highly accurate and automatically verified results. As an example, this is essential in simulation runs where the user has to distinguish between computational artifacts and genuine reactions of the model. Problem-solving routines with automatic result verification have been developed in PASCAL-XSC as well as in C-XSC for a large number of standard problems of Numerical Analysis.

Via the f2c compiler from AT & T and the C interface of PASCAL-XSC and C-XSC all FORTRAN libraries can be linekd and used with PASCAL-XSC and C-XSC programs.

A language description of PASCAL-XSC has been published by Springer-Verlag (ISBN 3-540-55137-9 and ISBN 0-387-55137- 9). Among others, the book contains a major chapter with sample programs, exercises, and solutions as well as a complete set of syntax diagrams, detailed tables, and an index. It can be used directly as a textbook for lectures on the subject of computer programming.

Compilers for PASCAL-XSC as well as the complete language description and user's guide are available in Europe from "Numerik Software GmbH", Rettigstraße 6, D-7570 Baden-Baden, Germany, FAX-Number: +49 721-69 44 18. Licence fees vary depending on machine type and size. In the USA and Canada please contact FB Software, P.O. Box 44 666, Madison, Wisconsin 53744-4666 or C. Abaci, Inc., 208 St. Mary's St., Raleigh, NC 27 605.

The reference manual and language description of C-XSC is also being printed by Springer-Verlag (ISBN 3-540-56328-8). The book will be available at the time of the conference. It contains a chapter with sample programs, and a library of problem-solving functions and programs with automatic result verification for standard problems of Numerical Analysis. The material in this book should be easily accessible for C or C++ programmers. All software concerning C-XSC can be obtained from the addresses mentioned in the preceding paragraph.

Adaptive Predictive Control Using Neural Networks and Interval Optimization

D. Layne, Staff Engineer

Research and Technology Dept., Martin Marietta Astronautics Group
P. O. Box 1036, Littleton, CO 80160 USA

Adaptive nonlinear control methods that are robust to uncertainties and disturbances are needed in many applications, including process control, robotics (flexible arms with variable loads) and large, flexible space structures. In large space structures, the tasks of controlling the rotations, pointing with high accuracy, and stabilizing the vibrations in the absence of damping pose difficult theoretical and computational control problems [2]. Challenges include the following:

(i) Large space structures that frequently change configuration will require on-board system identification.
(ii) Control laws are needed for flexible bodies with rigid attachments.
(iii) Robust control schemes for distributed parameter systems are needed to compensate for unmodeled dynamics and uncertain parameters.

A control architecture that combines Model Reference Adaptive Control (MRAC), neural network estimators, and interval optimization is being developed to address some of these problems. MRAC systems [1] are typically model-following controllers, using gradient minimization of current error between the plant and a reference model. In adaptive predictive control (receding-horizon), a plant estimator is used with the reference model to minimize predicted error.

The envisioned applications have unknown or uncertain plants, so neural networks (trainable input-output mappings) will emulate plant dynamics. Using time-lagged inputs, several neural net models have been trained on simulated observations to perform time-series prediction, including feed-forward nets with back-propagation learning and adaptive spline nets with least-squares learning. The spline nets learned faster with more accuracy on sample vibration suppression problems. The network mappings are designed to be differentiable, and nets may be trained on plant output or on errors between plant and desired reference model. A gradient-based method is being implemented to minimize predicted errors between the adaptable neural net estimator and the ideal reference model [4]. Sensitivities (gradients of network outputs with respect to control inputs) are readily computed in backpropagation nets, and may be computed via automatic differentiation for the more complex spline nets.

Interval optimization [3] of the cost function will increase reliability, avoid local minima and help characterize the nonlinear neural net models. Interval implementation of the iterative neural net training algorithm is not planned. However, a trained network mapping may be extended for use in an interval Newton algorithm to compute a guaranteed, global minimum within a specified region of system parameters. Software simulations are being developed for single-input, single-output test cases.

References

1. Astrom, K. and Wittenmark, B., *Adaptive Control*, Addison-Wesley, 1989.

2. Fleming, W., ed. *Future Directions in Control Theory: A Mathematical Perspective*, SIAM, Philadelphia, 1989.
3. Moore, R., Hansen, E. and Leclerc, A., "Rigorous Methods for Global Optimization," in "Recent Advances in Global Optimization," C. Floudas and P. Pardalos, editors, Princeton University Press, 1992
4. Narendra, K., "Adaptive Control of Dynamical Systems using Neural Networks," in *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches*, D. White and D. Sofge, editors, Van Nostrand Reinhold, 1992

Parallel Interval Global Optimization in C++

A. Leclerc

5771 Ave. Chateau Du Nord
Columbus, Ohio 43229 USA

For quite some time, it has been held that no numerical algorithm could guarantee having found a global solution to the general nonlinear global optimization problem. The reasoning was:

The function to be minimized can only be sampled at a finite number of points. Therefore, there is no way of knowing whether the function dips to some smaller value between sampled points.

Although this argument is probably true using evaluations of functions at points, it is not true of methods which can produce asymptotically accurate lower bounds for the range of values of the function over compact sets.

Interval arithmetic, for example, provides asymptotically accurate upper and lower bounds on ranges of values of functions over continua. Coding interval arithmetic in C++, the author has designed an ideal bounding mechanism which is:

1. capable of producing reliable, "tight", and asymptotically accurate bounds
2. efficient to compute
3. applicable to *any* programmable function
4. easy to generalize and automate
5. convenient for an unsophisticated user to utilize

Using this mechanism, a rigorous algorithm is presented which produces a list of "boxes" enclosing the set of all global minimizers and an interval trapping the minimum value. The algorithm does not require differentiability. However, for differentiable problems, improvements in efficiency are achieved by using interval Newton methods, monotonicity tests, and convexity tests. Numerical examples illustrating the techniques are given.

For further improvements in efficiency, the algorithms are *parallelized*. The parallelization task is accomplished by distributing processes over a network of workstations. Each process performs the interval global optimization algorithm but with a different subregion of the initial search space. Communication overhead is minimized in order to maximize the speedup. Issues of distributed initialization, load balancing, and global termination detection are addressed. Finally, an analysis of the speedup is determined.

Complementing Interval Arithmetic with Logic Programming

J. H. M. Lee and M. H. van Emden

Department of Computer Science
University of Victoria
Victoria, Canada, V8W 3P6

Logic programming realizes the ideal of "computation is deduction," but not when floating-point numbers are involved. In that respect logic programming languages are as flawed as conventional languages, such as Fortran and C: they ignore the fact that floating-point operations are only approximate and that it is not easy to tell how good the approximation is. We aim to extend the benefits of logic programming to computation involving floating-point arithmetic by using interval arithmetic.

Interval arithmetic contributes methods guaranteeing correctness at the level of a simple arithmetic expression. As embedded in imperative languages, however, interval arithmetic lacks verification of an entire algorithm involving conditional statements and iterations. This works against an important goal of interval arithmetic: *reliability*.

We combine interval arithmetic and logic programming, in such a way that rigorously justified claims can be made about the error in numerical computation. Cleary incorporated a relational form of interval arithmetic into Prolog so that variables already bound can be bound again. In this way, the usual logical interpretation of computation no longer holds. Based on Cleary's idea, we develop a technique for narrowing intervals, which is guaranteed to produce results that are logical consequences of the arithmetic axioms and the input intervals. We present a relaxation algorithm for coordinating the applications of the interval narrowing operations to numerical constraints in a network.

We incorporate relational interval arithmetic into two established logic programming languages: CHIP and CLP($\mathcal{R}$). We modify CHIP by allowing "domains" to be intervals of real numbers. Interval narrowing is shown to be an instance of the Look-Ahead Inference Rule used in CHIP. In CLP($\mathcal{R}$), we represent intervals by inequality constraints. Interval narrowing is viewed as a constraint simplification step. Most importantly, the enhanced languages ICHIP and ICLP($\mathcal{R}$) preserve the semantics of logic so that *numerical computations are deductions*, even when floating-point arithmetic is used.

We have constructed a prototype of ICLP($\mathcal{R}$), consisting of a meta-interpreter executed by an existing CLP($\mathcal{R}$) system. We expect our method to be applicable to, besides numerical computing, temporal and spatial reasoning, and automatic theorem proving involving arithmetic.

Strategies for a Boundary-Based Interval Newton's Method

P. Linz and L. Simcik

Division of Computer Science, University of California
Davis, California 95616 USA

We consider here the multi-dimensional root-finding problem

$$f_1(x_1, x_2, \ldots, x_n) = 0$$
$$\vdots$$
$$f_n(x_1, x_2, \ldots, x_n) = 0$$

Krawczyk's method starts at the center of a given region in $\mathbb{R}^n$, where the Jacobian is computed and used to find the region in which roots may possibly be located. The boundary-based interval Newton's method, on the other hand, starts at the boundaries of the region and uses slope information to eliminate regions where no root can be located. To illustrate, consider the one-dimensional case

$$f(x) = 0$$

for $x \in [a, b]$. Suppose that $f(b) > 0$ and that

$$\max_{x \in [a,b]} |f'(x)| = M.$$

Then the region $\left(b - \frac{f(b)}{M}, b\right)$ cannot contain any root and can be eliminated from further consideration. The generalization of this observation to more than one dimension is immediate.

The boundary-based method has a number of advantages over Krawczyk's method. One is that it can shrink the interval even if the Jacobian is singular. Another advantage is that it only requires the derivatives $\frac{\partial f_i}{\partial x_j}$, and not the inverse of the Jacobian, thereby considerably speeding up each iteration. Furthermore, as our research shows, with a careful strategy, the boundary-based method considerably reduces the need for interval splitting, which is a major source of difficulty for Krawczyk's method.

Interval Methods Applied to Variable Bound Reduction in Nonlinear Programs with Application to Infeasibility and Redundancy Diagnosis Part 1: Theory

W. A. Lodwick

University of Colorado at Denver
Department of Mathematics - Campus Box 170
P.O. Box 173364, Denver, Colorado 80217-3364 USA

Methods, algorithms and numerical experiments for variable bound reduction in nonlinear programming problems are developed. In particular, two approaches to compute bounds on variables found in nonlinear functional constraint inequalities are used and tested. The techniques developed herein are applied to infeasibility and redundancy diagnosis.

Part 2: Applications of Interval Methods for Variable Bound Reduction in Radiation Therapy, Refinery Models, Fuzzy Linear Programs and Artificial Intelligence

Problems associated with radiation therapy, refinery models, fuzzy linear programs and constraint logic programming require the computation of bounds on associated variables as a means to determine the set of feasible values which either solves the problem or identifies starting values for an optimizer. Part 2 of the talk will discuss how to compute bounds on variables in nonlinear functional constraints (prior to optimization in the case of a mathematical programming problem and as a means for instantiation in the constraint logic programming case) in radiation therapy problems, refinery models and in constraint logic programming (an extension of PROLOG).

Verified Solution of Ordinary Initial Value Problems with Large Step Sizes and with Step Size Control

R. J. Lohner

Institute for Applied Mathematics
University of Karlsruhe
Kaiserstr. 12, W-7500 Karlsruhe, Germany

In the past, several methods have been designed which compute verified bounds for the solution of ordinary initial value problems fully automatically on a computer. Most of these methods allow arbitrary step sizes to be prescribed at each time step, but their authors usually do not treat the question of step size control. Also, most methods, being explicit, do not allow step sizes which are larger than an Euler step. There seems to be no method which supports both step sizes larger than the Euler step and a built-in step size control.

In this talk we will present an extension of our earlier method which now allows step sizes larger than an Euler step, by using a suitable interval polynomial arithmetic. Additionally, the method is supplemented by a step size control which basically is due to Eijgenraam (1981).

These two extensions yield a much more powerful and flexible enclosure method for initial value problems than the previous method. The extensions will be presented, discussed and demonstrated with many numerical examples.

On the Presentation of Ranges of Monotone Functions Using Interval Arithmetic

S. M. Markov

Division for Mathematical Modelling Institute of Biophysics
Bulgarian Academy of Sciences, Acad. G. Bonchev str., block 25A
BG-1113 Sofia, Bulgaria

We consider the representation of ranges of functions (in many variables) by means of various extended interval arithmetic structures, such as structures involving infinite intervals, improper (Kaucher type) intervals, etc. A program system supporting the computation of ranges of functions and their derivatives using such extended intervals is reported.

Some Interpolation Problems Involving Interval Data

S. M. Markov

Division for Mathematical Modelling Institute of Biophysics
Bulgarian Academy of Sciences, Acad. G. Bonchev str., block 25A
BG-1113 Sofia, Bulgaria

We consider some interpolating problems using polynomial functions in the situation when interval input data are involved. Special attention is given to the situation when the degree of the interpolating polynomials is substantially less than the number of input data points. An algorithm and a program computing the envelopes of the potential families of interpolating polynomials are reported.

Standards for Floating Point and Vector Floating Point Arithmetic

D. W. Matula

Dept. of Comp. Sci. and Engrg.
Southern Methodist University, Dallas, TX 75275 USA

The *IEEE* floating point standard has been virtually universally adopted for PC and workstation platforms and is likely to become an available option on mainframes and supercomputers in their next generation. Several distinct features of the standard have contributed to its success in removing the chaos from previous nonstandardized floating point numeric computation systems. For the primitive arithmetic operations the controlled directed rounding feature dictates unique results. The not-a-number feature provides completeness for the operations over all arguments. The precision hierarchy thoughtfully treats both expanded accuracy and range. We describe these features and their significance in providing efficient hardware support for interval arithmetic and automatic result verification. We present extensions of these ideas beyond the primitive arithmetic operations to the area of transcendental function and vector processing hardware primitive operations. Problems and proposed solutions for handling error monitoring flags in a parallel and/or pipelined vector processing environment without degradation of efficiency for well structured computations will be presented and evaluated.

Epsilon-Inflation in Verification Algorithms

G. Mayer

Institut für Angewandte Mathematik, Universität Karlsruhe
Postfach 6980, W-7500 Karlsruhe, Germany

We present a new criterion for some class of *nonlinear* functions $f : D \subseteq \mathbb{R}^n \to \mathbb{R}^n$ which guarantees that epsilon-inflation combined with fixed point iteration will necessarily result in an inclusion

$$f([x]) \subseteq [x] \qquad (*)$$

in a finite number of steps. Here $[x]$ is some interval vector constructed by the iterative process just mentioned. The theorem generalizes a recently published result of S. M. Rump which was derived for *linear* functions f. It is well known that in the case of $(*)$ Brouwer's fixed point theorem applies to verify a fixed point of f in $[x]$.

We also interprete some ansatz in several algorithms for eigenvalue problems and for initial value problems as an epsilon-inflation, thus showing that the concept of this inflation is very general.

Restrictions on Approximation of the Standard Functions for Constructing Standard Interval Procedures

G. G. Menshikov

Faculty of Applied Math. and Control Processes
St. Petersburg State University
St. Petersburg, Russia

An Approach to Reliable Computations with Minimal Representation

E. A. Musaev

Steklov's Mathematical Institute
St.Petersburg, Russia

Sometimes continuation of an interval computation comes to a dead end as two values to be compared happen to be incomparable. When indefiniteness is due to the selected precision, repeating the computation with a higher precision seems very attractive. On the other hand, it is impossible to determine a priori what precision is necessary. An approach to automatically get the minimal necessary representation is suggested below.

The idea that the computational process could choose the necessary precision is proposed in [1], while the basis of the concrete method suggested here is described in [2,3]. In this model, the program consists of several shells, which differ only in the representations used. To begin, the shell with the simplest representation is initialized. If incomparable values occur, then this shell is frozen, and control is transfered to a shell with a more complex representation (in the case of concurrent processes it could be initialized immediately, but have lower priority). When this shell comes to the point with incomparable values, it corrects them and activates the previous shell, but if the values are still incomparable, a third shell is initialized to get them properly.

In this scheme, the low level shell makes a path for the complex and expensive high level shells, computing, when possible, directions for IF and CASE operators, and iteration counters for loops. On the other hand, the high level shells correct the values computed by the low level shells. We use the the term "wave computations", since the points of control in the shells run one after another like waves on a sea.

A scheme for realization of this idea is proposed. It is described using a language similar to Algol 68, with elements of Pascal and C. It could be also described very well in C++ or other object-oriented languages. When true concurrent processing is available, it is not necessary to freeze all high level processes. It is sufficient to have a guarantee that a process with a higher level will never outstrip any process of lower levels (it is necessary to properly process rendezvous points). A process with a lower level should have higher priority in comparison with a process of higher levels, so the process of level 0 is frozen if and only if it is needed in support of the process of level 1, while the process of level 1 is freozen if and only if it is needed in support of the process of level 2, or it is at the same point as the process of level 0, or the process of level 0 is running now and there are no processor resources. Moreover, besides ease in obtaining real concurrent computation, the advantage of this scheme is an ease in using it with immediate effect.

References

1. Yakovlev A. G., On some possibilities in the organization of localization computations on computers. Interval analysis. Informational materials – Computer Centre, Siberian Division of the Soviet Academy of Sciences, Krasnoyarsk, 1990 - pp. 33–38 - (in Russian).

2. Musaev E. A. Wave computations in interval analysis, Proceedings of aseminar on interval mathematics, Saratov, May 29-31, 1990, Saratov State University, Saratov, 1990, pp. 95–100 (in Russian).
3. Musaev E.A. Non-hierachical wave computations, Proceedings of the conference "Actual problems of modern mathematics", Computer Center of Saratov State University, 1991, pp. 110-112 (in Russian).

Finite Element Method for Nonlinear Elliptic Problems with Result Verification

M. T. Nakao

Department of Mathematics, Faculty of Science
Kyushu University 33, Fukuoka 812, Japan

In the last decade, various kinds of numerics with result verification have been proposed for differential equations. We investigated for years the numerical verification of solutions for partial differential equations (PDEs) based on the finite element method (FEM) and computable error estimates combined with the infinite dimensional fixed point theorem. While the FEM is developed as the most convenient and strong numerical method for PDEs and many error analyses have been done from the theoretical point of view, there are very few works on numerical or explicit error estimates in the a posteriori sense except for the linear coercive case. Our work also implies that we can obtain the finite element solution with a posteriori and guaranteed error bounds, even if we have no information about the existence of an exact solution for the original problem.

In our method, an element u in a certain function space is treated by the computer as the sum of the rounding $R(u)$ and the rounding error $RE(u)$. Here, $R(u)$ means the projection of u into some finite element subspace and $RE(u)$ the set including potential error. When the elliptic problem is defined as the fixed point form: $u = Fu$, the verification condition is represented, through some fixed point theorem, by $R(FU) + RE(FU) \subset U$ instead of $FU \subset U$ for a bounded, closed and convex set U. $R(FU)$ is calculated as the linear combination, with *interval coefficients*, of basis functions in the finite element subspace, and $RE(FU)$ is given by a nonnegative real number corresponding to the error bound for the projection. In order to determine the rounding error $RE(FU)$, we use some computable error estimates of approximate solution for Poisson's equation by FEM.

Since our verification method utilizes a kind of set valued Newton-like iterative method which includes the usual Newton's method for obtaining an approximation, it is possible to verify the exact solution by the same procedure or scheme as in the calculation of approximate solution. We also emphasize that the verification can be done by the use of standard C^0 elements. This fact leads us to easy implementation of the verification program for various domain shapes, e.g. polygon, curved region etc. Moreover, it appeared recently that, by the numerical determination of the constant in a priori error estimates using a computer, our method can also be applied to nonconvex domains, e.g. an L-shape domain, in which the solution has only low regularity.

In this talk, we show the basic verification principle of the method for the nonlinear Dirichlet problem and present some numerical examples. Furthermore, we will briefly mention an extension to evolutional problems.

On Some Generalizations of Interval Arithmetic

V. M. Nesterov

St. Petersburg Institute for Informatics and Automation
Russian Academy of Sciences, Box 52
St. Petersburg 195256, Russia

The problem of computing the range of values of an elementary function $f(x)$, where $x \in \mathbf{I}$ for some interval $\mathbf{I}$, often occurs in practice. Computing the range of values of terms constructed from such functions is very important as well. By elementary functions in the limited sense, we mean the functions exp, ln, sin, cos etc. In a broader sense, we mean a finite set of arbitrary functions forming a basis for some application.

The situation when functions under investigation are continuous and monotone is quite simple [1,2]. The situation when a function tends to infinity near some point (e.g. $\ln x$, $1/x$ near 0) is much more interesting. Also, it is very important to be able to estimate the range of a function over an infinite interval.

In this paper we consider special generalizations of interval arithmetic which enable us to solve the aforementioned problems in some cases. The generalization to infinite intervals is widely known [1,2]. In this paper we propose a method to reduce the arithmetic of infinite intervals to arithmetic of finite intervals. To do this, we introduce a mapping from the set of real numbers without one point but with one additional element - "infinity" to the ordinary set of real numbers. This mapping generates a corresponding mapping between sets of infinite and finite intervals. Use of this approach leads to good results for functions which have only one break point. We also propose a method to estimate the accuracy in the computed range of a term constructed from elementary functions. This method extends results obtained in the paper [1].

A second type of generalization uses a finite set of intervals instead of individual intervals (which we also consider) as an argument of the interval function. This approach enables us to work with partially continuous functions with many break points.

The third group of generalizations is used to estimate the range of functions in more than one variable. These generalizations are based on preliminary investigation of the monotonicity of the function and on discovering the points where the function tends to infinity.

References

1. V. M. Nesterov. On one extension of interval analysis and its application for estimation the range of function values. *Mathematical methods for algorithm construction and analysis*, Leningrad, Nauka, 1990, pp. 109–124. (In Russian).
2. S. M. Markov. Extended interval arithmetic involving infinite intervals, 26p., Private communication.

Rigorous Chaos Verification by Interval Methods

A. Neumaier and T. Rage

Institute of Applied Mathematics
University of Freiburg, Hermann-Herder-Str. 10
D-7800 Freiburg, Germany

Faculty of Physics
University of Freiburg, Hermann-Herder-Str. 3
D-7800 Freiburg, Germany

It is shown how interval analysis can be used to give rigorously valid enclosures of portions of stable manifolds of fixed points of finite-dimensional nonlinear mappings whose Jacobian at the fixed point is real hyperbolic.

The main tool is a semilocal version of a well-known existence theorem for stable manifolds of mappings which gives verifiable conditions under which a *specified* neighbourhood of an approximate manifold contains a specified part of the stable manifold. The proof is based on the fixed point theorem of Banach, applied in a suitable space of Lipschitz continuous functions.

Theorem 1: Let $F : \Omega \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$ be a Lipschitz continuous function having the fixed point $x^* \in \Omega$, and let $A \in \mathbb{IR}^{n\times n}$ be an interval matrix such that

$$F(y) - F(x) \in A\,(y - x) \quad \text{for all } x, y \in \Omega. \tag{1}$$

For some nonsingular matrix $Q \in \mathbb{R}^{n\times n}$, let

$$Q^{-1}(AQ) = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} \tag{2}$$

with interval matrices B_{11}, B_{12}, B_{21} and B_{22} of sizes $p\times p$, $p\times q$, $q\times p$ and $q\times q$, respectively, where $p+q=n$. For some nonsingular matrix $C \in \mathbb{R}^{q\times q}$ and some interval matrix $L \in \mathbb{IR}^{q\times p}$, put

$$D := I + C(LB_{12} - B_{22})\,, \tag{3}$$

$$E := C(LB_{11} - B_{21})\,, \tag{4}$$

$$K := Q\begin{pmatrix} I \\ L \end{pmatrix}, \tag{5}$$

$$M := B_{11} + B_{12}L\,. \tag{6}$$

If the closure conditions

$$\|D\|_p + \|C\|_q\|M\|_p \le \beta < 1 \quad \text{and} \tag{7}$$

$$DL + E \subseteq L\,, \tag{8}$$

hold for suitable norms $\|.\|_p$ in $\mathbb{R}^p$ and $\|.\|_q$ in $\mathbb{R}^q$, then, for any subset Σ of $\mathbb{R}^p$ with

$$0 \in \Sigma\,,\ Mt \subseteq \Sigma \quad \text{for } t \in \Sigma\,, \tag{9}$$

$$x^* + Kt \subseteq \Omega \quad \text{for } t \in \Sigma\,, \tag{10}$$

there are unique Lipschitz continuous functions $x : \Sigma \to \Omega$, $\sigma : \Sigma \to \Sigma$ and $g : \Sigma \to \mathbb{R}^q$ such that

$$F(x(t)) = x(\sigma(t)) \qquad \text{for} \quad t \in \Sigma\,, \tag{11}$$

$$x(t) = x^* + Q\begin{pmatrix} t \\ g(t) \end{pmatrix} \qquad \text{for} \quad t \in \Sigma\,, \tag{12}$$

$$x(0) = x^*\,,\ \ x(s) - x(t) \in K\,(s-t) \quad \text{for} \ \ s,t \in \Sigma\,, \tag{13}$$

$$g(0) = 0\ \ ,\ \ g(s) - g(t) \in L(s-t) \quad \text{for} \ \ s,t \in \Sigma\,, \tag{14}$$

$$\sigma(0) = 0\ \ ,\ \ \sigma(s) - \sigma(t) \in M(s-t) \quad \text{for} \ \ s,t \in \Sigma\,. \tag{15}$$

As an application, we describe a method for obtaining a computer-assisted proof of chaos in discrete dynamical systems, by verifying sufficient conditions for the existence of a transversal homoclinic point.

Numerical results are described for two 2-dimensional systems, namely the Chirikov standard map and the Poincaré mapping of a continuous-time dynamical system.

In the latter case, the interval enclosure of the Poincaré mapping and its Jacobian was calculated by the initial value problem solver AWA of R. Lohner. The very long runtime of this program makes it desirable to develop faster enclosure methods for ODE's which simultaneously enclose the solution and its derivative.

A Neural Network for Optimizing Radiation Therapy Dosage

F. Newman and H. Cline
University of Colorado Health Sciences Center
Dept. of Radiology, Campus Box C-278
4200 E. 9th Ave., Denver, Colorado 82062 USA

An associative memory technique has been developed for encoding, storing and retrieving two-dimensional radiation therapy treatment plans. The objective of this technique is to automate the search for a feasible set of radiation beams to be used as a starting point for constrained dose optimization. The mathematical model is a simple linear relationship. Each patient/tumor image is encoded as a vector x. Each set of radiation beams corresponding to a particular patient is encoded as a vector y. The set of all feasible beams for any patient consists of a 360 degree array of adjacent pencil beams. The intensities and arrangements of the pencil beams are varied to deliver a feasible (perhaps optimal) dose distribution. The feasible beam arrangement must be input by human intervention at this stage of development.

Training consists of:

1) forming a set of representative encoded patient/tumor vectors, $\{x \subset\}_{i=1}^{n}$
2) determining the feasible beam vector y_i to be associated with each patient vector, x_i
3) orthogonalizing the patient vectors with respect to each other and the beam vectors with respect to each other using the Gram–Schmidt technique
4) stacking the vectors (x_i's and y_i's) as columns into matrices X and Y.

Recall consists of presenting a novel patient/tumor image x' and finding a solution $y' = YX^Tx'$ that has been shown by Teuvo Kohonen et al to be an optimal solution in a least-squares, minimum norm sense. A feasible beam configuration is obtained as a linear combination of stored feasible beams using correlation matrix formalism. That is, a heteroassociation is formed.

Future work will entail training with beam vectors from feasible regions using interval methods so that subjective user decisions are eliminated from the training process and a closed, automated loop is formed.

On Combination of Interval Analysis and Object-Oriented Approach in Solving Systems of Non-linear Differential Equations

A. V. Nikitin

ap. 162, 39 / 1, pr. M. Toreza
Saint Petersburg, SU-194223, Russia

This paper presents a combined interval analysis and object-oriented approach to solving systems of nonlinear differential equations. A new interval version of an iterative Newton-like algorithm, based on the Senyo and Vengersky modification of the Krawczyk interval iterative method is proposed. Implementation of this algorithm using an object-oriented programming technique, including C++ class libraries, is discussed. Perfomance of this implementation in a program for transient analysis of non-linear circuits and networks is presented.

Numerical Methods for Enclosing Solutions of Ordinary Differential Equations with Interval Data

V. A. Novicov and A. N. Rogalev
Computing Center of the Siberian Department of the Russian Academy of Science Academgorodok, Krasnoyarsk 660036 Russia
and
Department of Computational Mathematics Krasnojarsk State University

Consider the initial value problem

$$dy/dx = f(x,y), \quad y \in R^n, \tag{1}$$

$$y(x_0) = y_0 \in Y_0 \tag{2}$$

where Y_0 is an interval in R^n. We wish to compute an interval valued function $Y(x) = [Y_{\text{Lower}}(x), Y_{\text{Upper}}(x)]$ for which the true solution $y(x, x_0) \in Y(x)$ for all $x \in [x_0, x_T], y_0 \in Y_0$, and prove convergence. The construction of upper and lower bounds Y_{Lower}, Y_{Upper} is based on analytical representation of a spline solution for IVP (1), (2) and interval techniques to bound the global error.

Consider the grid $x_0 < x_1 < \cdots < x_n = x_T$ and step size $h = \max_i(x_i - x_{i-1})$
Definition. Let the interval valued function $Y(x)$ satisfy conditions

$$Y(x) \supset \{y(x_0, y_0, x) \mid y_0 \in Y_0\} \tag{3}$$

$$\zeta(Y(x), \{y(x_0, y_0, x) \mid y_0 \in Y_0\} \to 0 \text{ as } h \to 0, \tag{4}$$

where ζ is the Hausdorff distance, then we call $Y(x)$ a convergent external interval estimate.

The right side of (1) may be a linear or nonlinear function.
1. First we suppose that $f(x,y)$ is a linear function. The interval estimate $Y(x)$ is respresented as

$$Y(x) = S(x) + h^m[-r(x), r(x)], \tag{5}$$

where $S(x)$ is united extension of spline solution $s(x)$ over $y_0 \in Y_0$, i.e.

$$S(x) = \cup_{y_0 \in Y_0} s(x_0, y_0, x),$$

and $r(x)$ is a bound on the global error.

The components of the vector spline function $s(x)$ are represented in the form

$$s_K(x) = \sum_{j=0}^{q} c_{ij}^K \frac{(x-x_i)^j}{j!} + \sum_{j=0}^{d} c_{i,q+j}^K \frac{(x-x_i)^{q+j}}{(q+j)!} \tag{6}$$

These components satisfy collocation conditions in grid points

$$c_{ij}^K = s_K^{(j)}(x_i), j = 0, \cdots, q,$$

$$c_{0j}^K = y_K^{(j)}(x_0), j = 0, \cdots, q,$$

$$s_K^{(j)} \mid_{x=x_{i+1}} = f_K^{(j-1)}(x_{i+1}, s(x_{i+1})), j = 1, \cdots, d, i = 1, \cdots, n.$$

Parameters of the spline solution (degree m, defect d) are chosen so that stability and convergence is guaranteed.
2. If a nonlinear function $f(x, y)$ is treated, then the interval estimate $Y(x)$ has same form (5). The spline solution $s(x)$ is constructed in the form

$$s_K(x_0) = y_K^0,$$

(7) $$s_K(x_i) = s_K(x_{i-1}) + \sum_{j=0}^{m} f_K^{(j)}(x_{i-1}, s(x_{i-1}))\frac{(x - x_i)^{j+1}}{(j+1)!},$$

$$x \in [x_{i-1}, x_i], i = 1, \cdots, n.$$

Realisation of the numerical method includes:
analytical computation $s(x)$ in (8),
linearisation of $s(x)$ in analytical form over initial values $y_1^0, y_2^0, \cdots, y_n^0$,
interval estimation of the global error.

We prove that the Hausdorff distance between the interval function $Y(x)$ and the set of true solutions tends to zero as the step size h tends to zero.
Theorem. The interval valued function

$$Y(x) = S(x) + h^m[-r(x), r(x)],$$

where

(8) $$r(x) = \frac{d!q!}{m!(m+1)!} \frac{\max_\eta |y^{(m+1)}(\eta)|}{\sum_{j=1}^{d}(|b_j| + |a_j|)h^{j-1}} \frac{\exp(xLt) - 1}{Lt},$$

$$t = 2(|b_1| + |a_1|),$$

$$a_j = \frac{(m-j)!}{m!}\binom{q}{j}, b_j = \frac{(m-j)!}{m!}(-1)^{j+1}\binom{d}{j}$$

for the linear function $f(x, y)$,

and

$$Y(x) = S(x) + h^m[-r(x), r(x)] + W^2(Y_0)[-r_1(x), r_1(x)],$$

$$r(x) = (e^{\|A\|} - 1)b_2/(\|A\|(m+1)!),$$

$$r_1(x) = (e^{\|A\|} - 1)b_1/\|A\|, a_{ij} = L(e + 1/(m+1)!),$$

$$b_1 = en(n-1)G/4, b_2 = n(n-1)GW^2(Y_0) + \omega(y_K^{(i+1)}, h)$$

where $W(Y)$ is the width of the interval vector Y,
ω is a modulus of continuity,
L is a Lipshitz constant,
G is a bound for the second dervative of the function $f_K^{(j)}(x, s(x))$,
$k = 1, \cdots, n; j = 1, \cdots, m$;

for the nonlinear function $f(x, y)$,

converge to the set of true solutions IVP (1), (2) in the sense (3)-(4) for linear IVP and converge to the set of true solutions IVP (1), (2) as $h \to 0$ and $W(Y_0) \to 0$ for nonlinear IVP (1), (2).
Test example. Consider the system of Moore

$$\begin{aligned} y_1' &= \ \ y_2, \\ y_2' &= -y_1 \end{aligned}$$

with initial values

$$\begin{aligned} y_1(x_0) &= [1-\epsilon, 1+\epsilon], \\ y(x_0) &= [-\epsilon, \epsilon], \end{aligned}$$

The step size was chosen $\pi/30$ and $\pi/300$.
Results of computations at $x = 2000\pi$ showed that value of the spline solution $s(x)$ is equal to value of true solution to within the range of accuracy of the arithmetical operations. The interval function $Y(x)$ includes the set of true solutions, and the value of the interval function $S(x)$ is equal to set of true solutions $\{y(x_0, y_0, x) \mid y_0 \in Y_0\}$.

A Family of Linear Programming Preconditioners for the Interval Gauss–Seidel Operator

M. Novoa III

1928 Hickory Ave., Apt. B, Harahan, LA 70123 USA

In [1], Kearfott presented an algorithm employing linear programming to compute preconditioners for the interval Gauss–Seidel operator. These preconditioners were, in a certain sense, optimal. Additional preconditioners were presented in [2]. In this work, we present theoretical results for these, as well as additional linear programming preconditioners.

References

1. Kearfott, R. B., "Preconditioners for the Interval Gauss–Seidel Method," *SIAM J. Numer. Anal.* **27** 3, (June, 1990), pp. 804–822.
2. Kearfott, R. B., Hu, C. Y., and Novoa, M. III "A Review of Preconditioners for the Interval Gauss–Seidel Method," *Interval Computations* **1** 1, (1991), pp. 59–85.

An Incremental Existence Test Based on Interval Newton Methods

M. Novoa III
1928 Hickory Ave., Apt. B, Harahan, LA 70123 USA

Application of an interval Newton operator can prove existence of a root of a nonlinear system of equations within a rectangular region, or box. In [1], Kearfott proposed that, in the context of a generalized bisection algorithm, one could check existence componentwise, accumulating information about the separate components of derived boxes obtained from repeated applications of the interval Newton and bisection operators. In this work, we have developed a general framework for such an incremental existence test. This framework allows application of both the Gauss–Seidel and Gaussian elimination interval Newton operators, as well as hybrids. Furthermore, the framework may be used with both interval Lipschitz and interval slope matrices.

Reference

1. Kearfott, R. B., "Interval Newton / Generalized Bisection When There are Singularities near Roots," *Annals of Operations Research* **25** (1990), pp. 181–196.

A Solution of Linear Resistive Networks by Interval Computation

K. Okumura and K. Sakanashi

Department of Electrical Engineering, Kyoto University
Kyoto 606, Japan

When we try to obtain solutions of linear circuit equations whose coefficients are given by interval numbers, we usually use the Monte Carlo method. As is well known, however, the Monte Carlo method requires substantial time to compute the solutions as the number of trials increases, in order to obtain an accurate range for the solutions. To overcome this difficulty we will use interval computation.

This paper presents a method for obtaining solutions of linear resistive networks whose components are given by interval resistances and interval current and/or voltage sources.

First, a formulation of the network equation suitable for interval operation is proposed. In the theory of the linear networks whose parameters are given by real numbers, the cutset or tieset equations are usually used. From the point of view of interval computation, it is not necessarily acceptable to apply these equations to the linear network with parameters set to interval numbers, since the elements of the coefficient matrix are given by linear combinations of the parameters. This causes us to have a bad estimation of the interval solutions of voltages and currents. When the network parameters are given by interval numbers, it is important to formulate the network equation in a manner suitable for interval computation. The authors propose formulation of the network equation in a hybrid form.

Second, we try to solve the hybrid network equation by means of the interval Gaussian algorithm. The condition for the interval Gaussian algorithm to be carried out is not necessarily satisfied in the hybrid equation as as well as the usual cutset or tieset equations. Hence we propose to use Hansen's scaling method to implement the interval Gaussian algorithm more practically.

Third, in order to have a more accurate range of solutions, we propose solving the pair of hybrid equations based on maximally distant trees. So far in the network with parameters of real numbers, the cutset or tieset equations are formulated based on one tree such as the normal tree. The interval solutions of the hybrid equation based on one tree are shown to have a fairly good estimation of the tree branch voltages. On the other hand, the interval solutions of the co-tree branch voltages do not always have a good estimation, because we need more interval computation from the tree voltages. Hence, we formulate the another hybrid equation based on the tree maximally distant from the former one. The intersection of the interval solutions of the both hybrid equations gives us a good estimation of solutions.

Finally, we give numerical examples comparing with the technique with the Monte Carlo method. A comparison with the results by the usual cutset equation is also made. The results by the hybrid equation using maximally distant trees are shown to be in fairly good agreement with those by Monte Carlo method. The computation time by the proposed method is far shorter than the Monte Carlo method.

Survey of Validating Computations in Pure Mathematics

P. S. Pankov

Institute of Mathematics, Republic of Kirghizstan Academy of Sciences
Lenenskii prosp. 265a, Bishkek (Frunze),720071 Kirghizstan

1) Some results obtained without strict estimation of computational errors.

- Stability of particular linear solutions of the three body problem (*Ryabo, Yu. A., 1953*, mechanical computer was used) and further investigations (*Bryuno, A. D.*).
- Verification of Riemann's hypothesis on the ζ-function within bounded domains (many authors, since early 1950-s).
- All convex polyhedra with faces being regular polygons (*Zalgaller, V. A., 1967*).
- Existence of periodic solutions of concrete differential equations, taking account of computational errors (*Urabe, M., 1970, Stokes, A., 1970, Sinay, Ja. G., 1979*).
- Convexity of the zero sequence of the function $(si(x) + si(x + \pi))$. (*Dzyadyk, V. K., Stepanetz, A. I., 1972*).
- Some cases of Minkovski's conjecture (*Malyshev, A. V., since 1976, Glasunoc, N. M.* used interval analysis).

2) The general scheme to reduce any problem to proving a strict inequality on a compact set and to use interval analysis was proposed in [1]. Some results obtained rigorously:

- A universal insert for convex figures of unit width (*1978*).
- A wider coefficient domain that unites known ones where an asymptotically stable finite-dimensional subspace of solutions of delay-differential equations exists (*Dolmatov S. L., 1979*).
- Proof of Polya's conjecture (consequence of the aforementioned Riemann hypothesis) (*Matiyasevich, Yu. V., 1982*).
- Proof of Feigenbaum's conjecture (*Lanford O. E., 1982*) and proof of universality of area-preserving maps (*Eckmann J. -P., Koch H., Wittwer P., 1984*).
- Improvements of known estimates in: univalency of analytical functions; spline approximations; Lebesgue's problem on universal cover of figures of unit diameter; densest packing of equal balls in three-dimensional space [2].
- Relativistic stability of matter (*Fefferman, C., de la Llave, R., 1985*). Stability in small denominator problems (*Rana D., 1987*).
- Existence of rotating boundary layer (with *Imanaliev, M. I., 1987*).

References

1. *Pankov P. Cybernetics, 1978*, 14, no. 3.
2. *Pankov P., Bayachorova B., Yugai S. Cybernetics, 1982*, 18, no. 6.

Enclosures for Solutions of Operator Equations with Applications to Nonlinear Boundary Value Problems

M. Plum

Mathematisches Institut der Universität zu Köln
Weyertal 86-90, D-5000 Köln 41, Germany

Consider the operator equation

$$U \in X \quad , \quad L_0[U] + \mathcal{F}(U) = 0 \quad , \tag{1}$$

where $L_0 : X \to Z$ is linear and bounded, and $\mathcal{F} : Y \to Z$ is Fréchet differentiable, with $X \subseteq Y \subseteq Z$ denoting three real Banach spaces such that the embedding $E_X^Y : X \to Y$ is compact and $E_Y^Z : Y \to Z$ is bounded. We make the general *regularity assumption* that, for some real σ, $L_0 + \sigma \cdot E_X^Z : X \to Z$ is one–to–one and onto.

We will present a method for proving the existence of a solution of problem (1) within an explicit and "close" $\|\cdot\|_Y$–neighborhood of some *approximate* solution $\omega \in X$, provided that

i) the *defect* $L_0[\omega] + \mathcal{F}(\omega)$ of ω can be bounded, in the norm $\|\cdot\|_Z$, by a sufficiently small constant δ ;

ii) the operator $L := L_0 + \mathcal{F}'(\omega)E_X^Y : X \to Z$, i.e., the linearization of problem (1) at ω, is one–to–one (and thus, as can be shown by Fredholm's Alternative Theorem, also onto), and the inverse L^{-1} (more precisely, $E_X^Y L^{-1}$) can be bounded *explicitly* in a suitable way. In our practical examples, we realize such bounds by computing a constant K such that, for all $u \in X$,

$$\|u\|_Y \leq K \|L[u]\|_Z \quad . \tag{2}$$

We give an elementary condition in terms of δ and K (and of some simple quantification of the Fréchet differentiability of $\mathcal{F}$) which implies the desired existence and enclosure result via Schauder's Fixed Point Theorem, and which is satisfied for sufficiently small δ.

The formally simplest application of the method are systems of nonlinear equations in $\mathbb{R}^n$ (where $X = Y = Z := \mathbb{R}^n$, $L_0 \equiv 0$). In this case, however, the bound (2) for the *matrix* inverse L^{-1} appears to be unnecessarily rough, since very accurate enclosure methods for matrix inverses exist.

Our main goal is to apply the method to nonlinear *elliptic boundary value problems* on some bounded regular domain $\Omega \subset \mathbb{R}^n$. Here, X is some appropriate closed subspace of $H_2(\Omega)$ containing the boundary conditions, $Z := L_2(\Omega)$, $L_0 := -\Delta$, $\mathcal{F}(u)(x) := F(x, u(x), \nabla u(x))$, where F is some given nonlinearity, and the "intermediate" space Y is chosen suitably (for instance, $Y := C(\bar{\Omega})$ if F is independent of the gradient ∇u).

Obviously, also *integro–differential equations* and even more general *functional differential equations* may be treated, at least in principle. Moreover, our approach covers "augmented" problems occurring in parameter–dependent problems with turning points, after change of parameters.

In practice, concrete methods for the computation of δ and K (see i) and ii) above) are needed. In our applications to boundary value problems (also to turning point problems), we use a *quadrature formula* (with remainder term bound) to compute δ, while the main numerical work which has to be done for the calculation of K consists in the computation of bounds for *eigenvalues* of L or of L^*L.

The method is illustrated by several numerical examples.

Validation of Multiple and Tightly Clustered Roots

L. B. Rall

Dept. of Mathematics, Univ. of Wisconsin
Madison, Wisconsin 53705 USA

Multiple and tightly clustered roots of polynomials and analytic functions may be difficult to approximate accurately and present problems with regard to validation. Techniques based on Newton's method and the principle of the argument provide ways to locate, detect, approximate, and validate approximations of such roots.

New Results on Global Optimization with Automatic Result Verification

D. Ratz

Institut für Angewandte Mathematik, Universität Karlsruhe
Kaiserstraße 12, D-7500 Karlsruhe, Germany

We present some new results on an interval method for global optimization with automatic result verification. The method based on the algorithm of E. Hansen is able to compute verified enclosures for all global minimizers and the global minimum of a function $f : \mathbb{R}^n \to \mathbb{R}$.

We give an overview of the different techniques used in our algorithm, and we describe the modifications and additional features improving the efficiency of our implementation compared to former versions. A modified box splitting in the Gauss-Seidel-Step, the use of special preconditioners introduced by Kearfott, and the test of local uniqueness for the global minimizer are some of the main topics of the talk.

Test results for standard global optimization problems are discussed for different variants of our method in its portable PASCAL–XSC implementation. These results demonstrate, that in many cases the efficiency of the algorithm is better than that of traditional methods, which do *not* give an automatic result verification.

The Brouwer Fixed Point Theorem and Verified Inclusions

G. Rex

Mathematisches Institut
Fachbereich Mathematik/Informatik
Universität Leipzig, Augustusplatz 10
D-O-7010 Leipzig, Germany

Miranda (1941) has shown that a generalized intermediate value theorem is equivalent to the Brouwer fixed point theorem. It is shown that Miranda's version of Brouwer's fixed point theorem is directly constructive for the derivation of verified inclusion operators for a solution z of a system of equations $f(x) = 0$. A radius vector for an interval vector enclosure of z is computed on the basis of the Perron–Frobenius theory [1, Theorem 3]. This new access is discussed for both linear and nonlinear f. The new results are extensions of [1].

References

1. Rex, G. Zur Einschliessung der Lösung eines linearen Gleichungssystems – Ein konstruktiver Zugang über den Fixpunktsatz von Brouwer – to appear in *ZAMM* **6** (1993).

Verified Inclusions of Solutions of Linear Interval Equations

J. Rohn

Faculty of Math. and Physics, Charles University
Malostranske nam. 25, 11800 Praha 1, Czechoslovakia

We shall describe a method for computing with arbitrary accuracy an interval enclosure of the solution set of a system of linear interval equations

$$A^I x = b^I \tag{1}$$

with an inverse stable $n \times n$ interval matrix A^I (i.e. $|A^{-1}| > 0$ for each $A \in A^I$). The method is based on the fact that in this case an interval vector X^I encloses the solution set of (1) if and only if it intersects $2n$ explicitly described unbounded convex polytopes.

A Step Size Control for Lohner's Enclosure Algorithm for ODE's and Applications

W. Rufeger

School of Mathematics, Georgia Institute of Technology
P. O. Box 37331, Ga. Tech. Station, Atlanta, GA 30222 USA

For nonlinear ODE's discretizations are traditionally the most important practical methods to get approximations of the true solution(s). But in general the computability of a difference solution does **not** imply the existence of the true solution. There is no theory available for what the difference-approximations really describe: the behavior of a true solution, a diverting (ghost) solution or nothing at all.

Consequently, it is desirable to compute a set of values ("the enclosure") which is guaranteed to contain the values of the true solution(s) while simultaneously verifying the existence of the true solution.

The Enclosure Algorithm developed in Karlsruhe by Lohner and Adams

1. yields guaranteed interval enclosures for all times $t \in [0, T]$ and the computed bounds are mathematically safe from all kinds of errors,
2. rests
   a) on the methods of interval arithmetic and on the Kulisch Computer Arithmetic (e.g. the computer language PASCAL-XSC)
   b) on an explicit one-step Taylor method of arbitrary order p with simultaneous enclosure of the local discretization error
   c) on the application of the Banach and the Brouwer fixed point theorems,
3. and is supplemented here by an automatic step size control, mainly in view of the computability of an enclosure in a close neighborhood of a pole, e.g. in the Restricted Three Body Problem.

The step size control is based on an estimate of the excess of the computed enclosure $[y(t_{j+1})]$ as compared with the set of true solutions starting at $[y(t_j)]$. For this purpose a few approximations of solutions $y^*(t, y_j)$ are used which start on the boundary of $[y(t_j)]$.

Lohner's Enclosure Algorithm with step size control has been used for the detection of "diversions" or "spurious difference solutions" or "ghost solutions" for the Restricted Three Body Problem and for the Lorenz Equations. A further application for this algorithm is the verfication of the existence of periodic solutions in a closed region (strip) S (whose lateral extension can be made negligibly small within graphical accuracy) for the stiffly coupled simplified Oregonator, a numerically ill-conditional mathematical model in chemical kinetics.

Verified solution of Large Linear Systems

S. Rump

Informatik III –Programmiersprachen und Algorithmen
Technische Universität Hamburg, Eissendorfer Straße 38
2100 Hamburg 90, Germany

Some new methods will be presented for computing verified inclusions of the solution of large linear systems. The matrix of the linear system if typically of band or sparse structure. There are no prerequisites to the matrix such as being M-matrix, symmetric or positive definite. Examples will be presented.

Precise Zeros of Analytic Functions Using Range Arithmetic

M. J. Schaefer

Universität Tübingen, Wilhelm-Schickard-Institut
Sand 13, 7400 Tübingen, Germany

Range arithmetic is a special kind of interval arithmetic in which a number is similar to an ordinary floating point number but has additionally a small range field and a variable number of mantissa digits. The range field keeps track of uncertainties in computed results due to roundoff and truncation and its small size enhances the efficiency of this arithmetic. Formal interval calculations, when needed, are best implemented using two ranged numbers to represent a single interval.

We present a practical algorithm for the computation of all zeros of a function f in a given rectangle, to a number of guaranteed decimal places requested by the user at the start of the program. This function must be analytic inside the rectangle and on its boundary, and must not have any zeros on the boundary. The precision of computation is varied dynamically for maximum efficiency. The algorithm is based on a discrete version of the argument principle and is shown to converge when carried out in range arithmetic and memory constraints can be ignored. Some numerical examples are presented as well.

The algorithm proceeds in a manner similar to an algorithm presented by P. Henrici and I. Gargantini for the simultaneous approximation of all zeros of a polynomial, which in turn was based on H. Weyl's proof of the fundamental theorem of algebra. Our algorithm uses rectangles rather than squares and maintains independent lists of adjacent rectangles, where it is known that the rectangles in a list collectively contain at least one zero of the function f. Rectangles in one list do not border on rectangles in other lists. If the rectangles in a list contain exactly one zero, an attempt is made to locate it rapidly using Newton's method and to verify its location by surrounding the point in question by a small square and tracing its boundary in the same way as described below for rectangles. The size of the square is chosen to satisfy the user's accuracy requirements.

The number of zeros contained in a rectangle R can usually be obtained by the following method: values of f at the corners of R are computed and classified according to their location (entirely within a quadrant, overlapping an axis, etc.). If the values of f at the endpoints of a side L can be put in a box not containing the origin, an estimate of $f(L)$ is obtained and then checked to see if it covers the origin. If the origin is not covered, the argument range of f along L is well determined, and if it does (or if $f(L)$ was never computed in the first place), L is recursively bisected and the computations for each half repeated. There is a limit to the maximum permissible depth of recursive function calls which depends on the current precision of computation. Of course, all four sides of R are processed in this way.

A list of rectangles is managed like this: each rectangle in turn is bisected along its longer side and its two emerging subrectangles checked for zeros. Rectangles known not to contain zeros are of course discarded, others appended to the end of the list. (If too many rectangles accumulate which contain an undetermined number of zeros, the precision is increased.) It is important to organize the computations and storage of results in such a way that expensive interval computations are not repeated. For this reason each rectangle maintains four pointers pointing to binary trees that store the results of the recursive procedure calls described in the previous paragraph. Two adjacent rectangles will share a tree if their overlapping sides are of equal length; otherwise the shorter of the two will point to a subtree of the other's tree. From time to time the rectangles in a list are inspected to determine whether they still represent a connected region; if not, the list is split into two or more independent lists. Also, if the region is sufficiently small the corresponding list need no longer be processed.

Vector Processor Support for Semimorphic Arithmetic

L. Schmidt

Institut für Angewandte Mathematik, Universität Karlsruhe
Kaiserstrasse 12, W–7500 Karlsruhe, Germany

For many years, languages for scientific computation like Pascal–XSC[4], ACRITH–XSC[5], and C–XSC have been in widespread use. They are available on many hardware platforms, ranging from personal computers to mainframes. Vector and parallel computers, however, were not covered until now, mainly because there was no implementation of semimorphic arithmetic which could take advantage of the pipelining and parallelization features. This talk and the related paper will focus on arithmetic for vector processors.

To obtain the best possible results in terms of performance, the available hardware (i. e. the vector processor pipelines) must be used extensively. We will present algorithms for the basic arithmetic operations (addition, subtraction, multiplication, division) which only use built-in floating-point operations. They are well suited for vectorization *and* deliver reliable results of maximum accuracy. For the computation of inner products, we discuss an algorithm using n long accumulators to hold the *exact* sums of n dot products computed in parallel. A short excursion to pipelined interval and complex arithmetic will close this part of the talk.

Implementation of the presented algorithms was done in FORTRAN 77, and is portable with the exception of very few data format dependent routines. All arithmetical array operations defined in ACRITH–XSC, and more, are included in the library. A fully compatible interface to ACRITH–XSC is provided including, dynamic array and subarray support.

Performance comparisons will take a large part of the talk. The implementation of the algorithms presented is compared with the numerically *equivalent* implementation of the ACRITH–XSC runtime library. To show the benefits of the vector processor support for more complex problems, a linear system solver with automatic result verification is included in the comparisons. The measurements were done on an IBM 4361 mini-mainframe, an IBM 3090/300 VF mainframe with vector facility, and a Fujitsu VP 2600 high performance (5 GFLOPS) vector processor.

[4]distributed by Numerik Software GmbH, D–7570 Baden–Baden, Germany
[5]IBM Program Product, released in 1990, order no.: 5684–129

Application of a Parallel Interval Newton/Generalized Bisection Algorithm to Equation-Based Chemical Process Flowsheeting

C. A. Schnepper and M. A. Stadtherr

Halliburton Services, 1710 West Plato Rd. #603
Duncan, Oklahoma 73533 USA
and
University of Illinois at Urbana–Champaign

Computer-aided process simulation, design, and optimization are important tools in the design and control of manufacturing processes in the chemical and petroleum industries. The diagram of a process, showing the units and the connections between them, is often referred to as a process flowsheet, and, especially in the chemical engineering literature, the associated simulation or optimization problems are referred to as process flowsheeting problems. In contrast to the traditional sequential modular approach to flowsheeting, in which the problem is divided into a sequence of smaller subproblems which are solved repeatedly until converging to a solution of the overall problem, equation-based flowsheeting uses a single, very large, sparse system of nonlinear equations to model a process.

In equation-based flowsheeting, the nonlinear equation solving algorithm is typically some quasi-Newton based approach. Such algorithms are not always reliable when the initial guess is not good, and they are not designed for finding multiple roots, if they exist. Various approaches have been used to address this problem, including trust region methods, homotopy methods, and methods based on imbedded nonlinear programming problems. Bisection approaches have not received serious consideration in this context because the number of variables involved in process flowsheeting makes such approaches infeasible on serial computers. However, by taking advantage of parallel computer architectures, interval bisection techniques can be made feasible, providing a globally convergent solution method capable of locating multiple solutions.

We describe here the application of a new parallel interval Newton/generalized bisection algorithm for solving the large, sparse, nonlinear equation systems arising in chemical process flowsheeting. The algorithm is designed for implementation on MIMD computers with a combination of local and shared memory. It is based on the simultaneous application of root inclusion tests to multiple interval regions, and it is designed for use with the sparse matrices associated with equation-based chemical process flowsheeting models.

The algorithm was tested successfully on several relatively small flowsheeting problems. Tests were performed using between 2 and 32 nodes of a BBN TC2000 parallel computer. These initial results demonstrate the potential of the approach and point the direction toward handling larger problems and implementations on other parallel machines.

Verified Surface/Surface Intersection of Parametric Bézier-Surfaces

P. Schramm

Institut für Angewandte Mathematik, Universität Karlsruhe
Postfach 6980, W-7500 Karlsruhe, Germany

The calculation of intersection curves between (parameterized) surfaces is one of the major tasks in Computer Aided Design. The algorithms available for solving this problem can be subdivided into the following classes: Lattice evaluation, marching methods, recursive subdivision and algebraic methods [1]. All these algorithms have in common that the evaluated intersection curve is given as a series of discrete points approximating it.

A new approach is made in computing the intersection as a parameterized curve in the parameter plane of each surface:

If two surfaces $X(u,v)$ and $Y(s,t)$, $X, Y : [0,1]^2 \to \mathbb{R}^3$ are given, the problem of intersection is solved by computing $u(\alpha)$, $v(\alpha)$, $s(\alpha)$, $t(\alpha)$, $\alpha \in [\underline{\alpha}, \bar{\alpha}]$ where u, v, s and t are polynomials of the same degree n.

The method of obtaining this result is a functional fixed point iteration in a Tschebyscheff Functoid $\{S_n(M), \boxplus, \boxminus, \boxast, \boxslash\}$ (resp. $\{IS_n(M), \diamondplus, \diamondminus, \diamondast, \diamondslash\}$ for a verified inclusion) [2] where M is the space of the real (interval) polynomials and S_n (resp. IS_n) is the Tschebyscheff rounding from M to $S_n(M)$ (resp. $IS_n(M)$) the space of real (interval) polynomials with a maximum degree of n. The fixed point equation system to be solved is given by

$$\begin{pmatrix} IS_n(X(u(\alpha), v(\alpha)) - Y(s(\alpha), t(\alpha)))_x \\ IS_n(X(u(\alpha), v(\alpha)) - Y(s(\alpha), t(\alpha)))_y \\ IS_n(X(u(\alpha), v(\alpha)) - Y(s(\alpha), t(\alpha)))_z \\ IS_n(\int_0^\alpha \sqrt[2]{\dot{X}^2(u(\xi), v(\xi))}d\xi - \alpha) \end{pmatrix} = \begin{pmatrix} u(\alpha) \\ v(\alpha) \\ s(\alpha) \\ t(\alpha) \end{pmatrix}$$

where the integrand $\sqrt[2]{\dot{X}^2(u(\xi), v(\xi))}$ of the fourth equation (parametrization condition for $(u(\alpha), v(\alpha))^T$) is expanded into a Taylor series.

This method has the advantage of calculating an approximation with respect to an interval inclusion of the intersection curve which gives the true topology of the real intersection curve. Up to that moment the algorithm was only applied to Bézier surfaces.

References

1. M.J. Pratt and A.D. Geisow. Surface/Surface Intersection Problems; in: J. A. Gregory (ed.) *The Mathematics of Surfaces*, Clarendon Press 1986.
2. E.W. Kaucher and W.L. Miranker, *Self-Validating Numerics for Function Space Problems*, Academic Press 1984.

On Hardware Implementation of some Exactly Rounded Elementary Functions

M. Schulte and E. Swartzlander

Deptartment of Electrical and Computer Engineering
University of Texas at Austin
Austin, Texas 78712-1084 USA

An algorithm is described which produces exactly rounded results for the functions of reciprocal, square root, 2^x, and $\log_2(x)$. Hardware designs based on this algorithm are presented for floating point numbers with 16 and 24 bit significands. These designs perform a polynomial approximation in which the coefficients are originally computed using the minimax criterion. The coefficients are then adjusted to guarantee exactly rounded results for all inputs. To reduce the number of terms in the approximation, the input interval is divided into subintervals of equal size and different coefficients are used for each subinterval. A method is presented for determining the accuracy of the pre-rounded result which will guarantee exact rounding. Area and performance estimates indicate that for numbers with 16 bit significands, the functions can be computed in approximately 55ns on a 5mm by 4mm chip. For numbers with 24 bit significands, the functions can be computed in approximately 80ns on a 10mm by 10mm chip. The algorithm present in this paper can be extended to provide extremely accurate results for other elementary functions and other number formats.

Interval Arithmetic Block Cyclic Reduction for Arbitrary Block Dimension with Applications to Domain Decomposition Methods

H. Schwandt

Technische Universität Berlin
Fachbereich 3 (Mathematik)
D-1000 Berlin 12, Germany

Block cyclic reduction methods have been widely used for the efficient solution of large linear systems of equations with special regular sparse coefficient structures resulting mostly from discretizations of partial elliptic boundary value problems. The Buneman algorithm for matrices of the form $(-S, A, -T)$ and its variations are well known examples for this kind of methods.

In the context of enclosure methods, interval arithmetic versions of the Buneman algorithm have a significant importance as they seem to be up to now the only interval methods which are not only efficient, but which can also guarantee, under appropriate conditions, satisfactory or even optimal inclusions for systems of equations with interval coefficients whose structure can be derived from the above mentioned class of applications.

The cyclic reduction principle underlying the Buneman algorithm(s) originally implied some restrictions on the matrix size. While the size of the matrices A, S, T is arbitrary, the number of block rows (the block dimension) is restricted to be of the form 2^{n-1} for matrices derived from Dirichlet problems (or 2^n and 2^{n+1} for Neumann and periodic problems, resp.). This restriction causes problems in particular if there are problem dependent limitations on the step size in the discretization or simply memory problems for very large problems. Therefore, Sweet (1974, 1977) has proposed a modification of the point (noninterval) Buneman algorithm for arbitrary block dimensions.

In the present context an interval arithmetic variant for arbitrary block dimensions is presented. Under the aspect of optimal inclusions, however, there still remain restrictions. We extend the range of admissible block dimensions for optimal inclusions and we estimate the width of the enclosures in the non optimal cases. As an example for the application of the increased flexibility concerning the block dimension we mention the integration of the new interval Buneman variant in a Schur based interval domain decomposition method.

In view of a practical application, we discuss the vectorization and parallelization of the resulting algorithms and we briefly introduce the implementation of a vectorizing simulation of an interval arithmetic on CRAY computers and a simulation on IEEE based machines. Numerical examples are included.

Solving Interval Linear Systems with Nonnegative Matrices

S. P. Shary

Computing Center, Siberian Department of the Russian Academy of Sciences, Academgorodok, 660036 Krasnoyarsk, Russia

Let the interval linear algebraic system (ILAS)

$$\mathbf{A}x = \mathbf{b}$$

be given with the interval $n \times n$ - matrix $\mathbf{A}$ and the interval n - vector $\mathbf{b}$. A classical interval analysis problem is the problem of "outer" componentwise evaluation of the *united solution set*

$$\mathbf{X}^*(A, \mathbf{b}) = \{x \in \mathbb{R}^n \mid (\exists A \in \mathbf{A})(\exists b \in \mathbf{b})(Ax = b)\},$$

formed by solutions of all point systems $Ax = b$ with $A \in \mathbf{A}$ and $b \in \mathbf{b}$. Traditionally, it is formulated as follows

find an interval vector $\mathbf{V}$ that contains
the united solution set of the given ILAS

If $\mathbf{V}$ is the least inclusive, i.e., its components are the projections of $\mathbf{X}^*(A, \mathbf{b})$ on the coordinate axis, then it is called the *optimal solution* of the stated problem. Determining such optimal $\mathbf{V}$ appears intractable in general, so researchers usually confine themselves to constructing algorithms to solve this problem optimally only for particular classes of interval linear systems. In this work we present new effective (polynomially complex) numerical methods for computing the *optimal* solutions of the ILAS with nonnegative matrices, based on fine geometric properties of these systems' united solution sets.

Fast Automatic Differentiation for Vector Processors

D. Shiriaev

Institut f. Angewandte Mathematik, Univ. Karlsruhe
Kaiserstr. 12, D-7500 Karlsruhe 1, Germany

Automatic differentiation is a non-approximative method allowing fast and exact evaluation of derivatives of any degree. Automatic differentiation does not incur any truncation errors and could yield exact results if the calculations could be carried out with infinite precision. The only errors introduced are those associated with using real arithmetic, instead of rational arithmetic.

The execution time requirement for gradient computation using the basic algorithm for fast automatic differentiation is only a small multiple of that for the program which computes the underlying function values. Unfortunately, all available systems implementing this algorithm suffer from a storage requirement that grows proportionally to the execution time of the original program. The storage requirement can therefore be significantly large for computation intensive problems, thus limiting the size of the problems that can be solved with this method.

We discuss a semimorphic extension of the general scheme of fast automatic differentiation from scalar arithmetic to the customary higher numerical spaces. Our main goal is to provide a basis for achieving high accuracy and for reducing the storage and temporal requirements of fast automatic differentiation. The set of basic operations is extended to vector operations, including accurate dot product, improving the numerical, space and time performance of the method. Using the compound vector operations, the control-flow structure of the underlying program is simplified and reflects better the nature of the problem, increasing reliability, ease of programming and maintenance of the programs. Vector operations provide a basis for parallelization and vectorization, and, if they are hardware supported, the speed of resulting program can be significantly increased. Inclusion of the accurate dot product as a basic operation of fast automatic differentiation improves significantly the numerical behavior of the algorithm and provides a basis for the achieving high or even maximal accuracy.

The new method is compared with another implementations of automatic differentiation on the example of the Helmholz energy function, illustrating the effectiveness of the proposed ideas.

Complexity of Fixed Points - A Review

K. Sikorski

Department of Computer Science
University of Utah, Salt Lake City, Utah 84112 USA

We address the problem of approximating fixed points of nonlinear mappings and survey all recent complexity results.

We consider two error criteria: 1. absolute and 2. residual.

For contractive functions and the absolute criterion we exhibit an optimal algorithm (i.e. and algorithm minimizing in the worst case the number of function evaluations). This algorithm is based on the bisection-envelope construction in the one dimensional case. In multivariate case the Banach's simple iteration algorithm is optimal if the dimension of the domain of functions is large. For moderately large dimension we present an ellipsoid algorithm based on the ellipsoid construction of Kchachian. This algorithm establishes an upper bound on the complexity. We conjecture a formula for a lower bound, however the actual complexity is still not known.

For noncontractive functions and the absolute error criterion, the bisection algorithm is optimal in the univariate case. In the multivariate case we show that the problem is unsolvable with finite cost. This holds even in the two dimensional case and nonexpanding functions (Lipschitz with constant 1).

For noncontractive functions and the residual error criterion the problem is tractable, however its complexity is an exponential function of the dimension.

Correction Terms for Quadrature Formulas

W. Solak

Institute of Mathematics
Academy of Mining and Metallurgy, Ul. Mickiewicza 30
Cracow, Poland

Reference [3] provides information concerning correction terms with parameter for the following quadrature formulas

$$G^{\beta}_{n+5}[f] = T_{n+1}[f] + h(24\beta)^{-1}[-3f(a) + 4f(a+\beta h) - f(a+24\beta h)$$
$$- 3f(b) + 4f(b-\beta h) - f(b-2\beta h)]$$

and

$$L^{\beta}_{n+6}[f] = M_n[f] + h(24\beta)^{-1}[2f(b-0.5\beta h) - 3f(b-1.5\beta h) + f(b-2.5\beta h)$$
$$+ 2f(a+0.5\beta h) + 2f(a+0.5\beta h) - 3f(a+1.5\beta h) + f(a+2.5\beta h)]$$

where

$$T_{n+1}[f] = h\sum_{i=1}^{n-1} f(a+ih) + 0.5h[f(a)+f(b)]$$

and

$$M_n[f] = h\sum_{i=0}^{n-1} f(a+(i+0.5)h),\ \textit{where } b-a=nh$$

In particular, in the case $\beta = 1$, formula (1) reduces to Gregory's formula see [1], [2] and formula (2) to Laplace formula [1] for the integral

$$I[f] = \int_a^b f(x)dx$$

In the general case, use of the above formulas shortens calculations of numerical integration. Further consideration of this problem is discused in [3].

Formulas (1) and (2) are of the fourth order and are given only the first correction terms, so-called of the first degree.

The principal purpose of this paper is to obtain estimates of errors in approximate integration with a parameters in formula (1) and

INTEGRATION FORMULAS

If a function $f \in C^{2k}[a,b]$, then the integral (3) can be represented by the first Euler-Maclaurin formula (see [1], p. 152)

$$\int_a^b f(x)dx = T_{n+1}[f] - \sum_{i=1}^{k-1} \frac{B_{2i}}{(2i)!}[f^{(2i-1)}(b) - f^{(2i-1)}(a)]h^{2i}$$

$$-f^{(2k)}\xi\frac{B_{2k}(b-a)}{(2k)!}h^{2k}$$

for some ξ between a and b.

A useful variant of the first Euler-Maclaurin formula is (see [4])

$$I[f] = G^{\beta}_{n+5}[f] + \frac{1-20\beta^2}{720}h^4[f'''(b) - f'''(a)]$$

$$+\frac{\beta^3 h^5}{48}[f^{(4)}(a) + f^{(4)}(b)] + 0(h^6)$$

and

$$I[f] = L^{\beta}_{n+6}[f] - \frac{7-80\beta^2}{4!240}h^4[f'''(b) - f'''(a)]$$

$$-\frac{\beta^3 h^5}{90}[f^{(4)}(a) + f^{(4)}(b)] + 0(h^6)$$

If $\beta_1 \in (0, 20^{-0.5})$, $\beta_2 \in (20^{-0.5}, 1)]$ and $f'''(b) \neq f'''(a)$ then exists N such that for $n \geq N$ we have

$$G^{\beta_1}_{n+5}[f] \leq I[f] \leq G^{\beta_2}_{n+5}[f] \; (G^{\beta_2}_{n+5}[f] \leq I[f] \leq G^{\beta_1}_{n+5}[f])$$

for these formulas (1).

If $\beta_1 \in (0, (7/80)^{0.5})$, $\beta_2 \in ((7/80)^{0.5}, 1)$ and $f'''(b) \neq f'''(a)$ then exist N and for $N \geq N$ we have

$$L^{\beta_1}_{n+6}[f] \leq I[f] \leq L^{\beta_2}_{n+6}[f] \; (L^{\beta_2}_{n+6}[f] \geq I[f] \geq L^{\beta_1}_{n+6}[f])$$

for these formulas (2).

References

1. Brass H. *Quadraturverfahren* Vandenhoek and Ruprecht, Göttingen, 1977.
2. Hildebrand F.B. *Introdruction to Numerical Analysis*, McGraw-Hill, New York, 1956.
3. Solak W. and Szydełko Z. "Quadrature rules with Gregory–Laplace end corrections", *Journal of CAM* **36** (1991), 251–253.
4. Solak W. "Boundary corrections for numeric integration formulae," *Opuscula Mathematica* Z. 8, Cracow 1991

Algebraic Algorithms with Automatic Stability Analysis

H. J. Stetter

Institut für Angewandte und Numerische Mathematik
Technische Universität Wien, Wiedner Hauptstraße 6-10
A-1040 Wien, Austria

In algebraic algorithms as they are implemented in standard Computer Algebra systems like REDUCE, Mathematica, etc., computations use rational numbers with potentially "unlimited" wordlengths for numerators and denominators. The use of floating-point arithmetic in such algorithms has appeared counter-productive to most computer algebraists. On the other hand, however, some algebraic algorithms have been widely used in Numerical Analysis with excellent success and with a potentially enhanced information return, in spite of floating-point computation. The best-known example is Gaussian elimination for linear equation solving.

This unexpected fact arises from the more sophisticated implementation which is indispensible with approximate computation: A potential breakdown of the accuracy has to be monitored which leads to a consideration of the algorithm for all data in some vicinity of the specified problem. Thus potential nearby degenerations of various sorts are discovered. In Scientific Computing, such information is often more essential than a 100% accurate result for the specified data.

Naturally, it is this supplementary monitoring in the algorithm – and not the use of floating-point arithmetic – which may make the floating-point versions of algebraic algorithms more suitable for Scientific Computing than the "strict" versions. Therefore, we have begun to design "embedded" versions of a number of classical algebraic algorithms, like the Euclidean algorithm for the g.c.d. of univariate polynomials or Buchberger's algorithm for the computation of Groebner Bases for sets of multivariate polynomials.

In these versions, quantities which are below a specified threshold relative to the data level of the problem are disregarded for the further computation; but the effect of this perturbation is stored and its propagation (both forward and backward) is accumulated. The result of such an algorithm describes the most degenerate situation which may occur in a (specifiable) small neighborhood of the specified problem. If, e.g., a minute change of some coefficients of a set of multivariate polynomials leads to a positive-dimensional manifold of joint zeros, this fact will be uncovered, whereas present "exact" implementations would give no indication of this fact.

While we have, at first, continued to use rational arithmetic throughout in an effort to have exact assertions about neighborhoods etc. and to separate sources of perturbation, we have now considered the use of floating-point arithmetic plus the use of intervals at crucial points. Some first results of this approach will be reported. It is claimed that such versions of algebraic algorithms will have a higher applicability in Scientific Computing than the standard rational arithmetic versions.

Numerical Integration in Two Dimensions with Automatic Result Verification

U. Storck

Institut für Angewandte Mathematik
Universität Karlsruhe (TH), Germany

In scientific and engineering problems, the values of multi-dimensional integrals are frequently needed. There are many different methods for numerical integration, especially in one and two dimensions. Particularly in the two-dimensional case, however, the remainder term, assuming it is taken into account, is not given in a form suitable for numerical computation. In addition, the round-off errors are rarely taken into account. Therefore, a reliable statement about the accuracy is not possible in general, and the numerical results are often doubtful.

In order to obtain an error estimate for a numerical result, two methods for calculating integrals of the form

$$J = \int_a^b \int_c^d f(x_1, x_2) dx_2 dx_1$$

with automatic result verification are presented. We will call these procedures the single Romberg extrapolation and the double Romberg extrapolation. The approximations of both algorithms are determined by Romberg extrapolation and can be presented by a linear combination of grid-values of the integrand. In order to obtain tight enclosures of the approximations the employment of the precise scalar product and interval arithmetics is necessary. The corresponding remainder terms are mainly determined by the Taylor coefficients of the integrand. For calculating enclosures of these terms, we use automatic differentiation, interval arithmetics and the precise scalar product. In both algorithms, the quality of the remainder term chiefly determines the error of the result, i. e. the width of the enclosure of the integral. We therefore examine in detail the representations of the remainder terms in dependency on the chosen step size sequences. Finally a short comparison of the two integration methods and further aspects for the integration in higher dimensions are given.

Interval Analysis Isn't Fuzzy Is It?

M. J. Tretter

Department Business Analysis, Texas A&M University
College Station, TX 77843-4217

There are some intriguing relationships between fuzzy set theory and interval analysis. Recent research by Zadeh ("Random sets and fuzzy interval analysis", *Fuzzy Sets and Systems* **42** (1991) pp. 87–101) offers a combined approach. This paper will look at extending this relationship. Applications will include Moore's 1984 interval analysis approach to risk analysis and a fuzzy/interval approach to geometric programming.

Truncated Newton's method with Automatic Differentiation and Interval Error Bounds

R. J. Van Iwaarden

2208 Williams St.
Denver, Colorado 80210 USA

The use of automatic differentiation (AD) to solve systems of nonlinear equations using truncated Newton's method will be presented with numerical examples, computed error bounds using interval analytic techniques and run times illustrating it's potential for applications. Little or no previous knowledge of AD is assumed and open questions regarding AD will be presented.

Computation of Integrals of Uncertain Vector Functions

V.M. Veliov
International Institute for Applied Systems Analysis
Schlossplatz 1, A-2361 Laxenburg, Austria

We address the following problem of integration of an uncertain function $f : [0,1] \to R^n$. Suppose that the only information about $f(\cdot)$ is that $f(t) \in \operatorname{conv}\{f_1(t), \dots, f_p(t)\}$, $t \in [0,1]$, where $f_1(\cdot), \dots, f_p(\cdot)$ are known functions. Then $\int_0^1 f(t)\,dt$ can, in principle, have any value from the set

$$I = \left\{ \int_0^1 \varphi(t)\,dt;\ \varphi(t) \in \operatorname{conv}\{f_1(t), \dots, f_p(t)\},\ \varphi(\cdot) - \text{integrable} \right\}.$$

The problem is to approximate the convex set I with any given accuracy. This is a particular case of the more general problem of approximation of the set

$$J = \int_0^1 F(t)W\,dt = \left\{ \int_0^1 F(t)w(t)\,dt;\ w(t) \in W,\ w(\cdot) - \text{integrable} \right\},$$

where $F(\cdot)$ is an $(n \times r)$-matrix function and W is a convex compact set in $\mathbb{R}^r$.

Two interrelated issues arise: 1) to develop the theory of quadrature formulae for set-valued integrals as (1); 2) to choose tools for representation/approximation of convex sets in R^n. Both of them will be discussed in the talk, but here we sketch only the first one, which is well understood for $n = 1$, but is much more complicated in the multidimensional case.

Let $0 \le \tau_1 < \dots < \tau_m \le 1$ and $a_1, \dots, a_m \in \mathbb{R}$ generate the linear quadrature formula $\sum_{i=0}^m a_i g(\tau_i h))$, which is exact for real polynomials g of degree 1. Let $t_0 = 0$, $t_1 = h$, $\dots$, $t_N = Nh = 1$ be the N-points uniform grid in $[0,1]$. It turns out that the linear composite formula

$$\bar{J}_N = \sum_{k=0}^{N-1} (\sum_{i=0}^{m} a_i F(t_k + \tau_i h)W)$$

approximates J with accuracy const$/N^2$ with respect to the Hausdorff distance between sets (the constant also can be estimated), provided that F is Lipschitz continuous and $\dot{F}$ is of bounded variation.

In general, there are no linear quadrature formulae that approximate J with accuracy better than const$/N^2$ even in the class of analytic matrices F and sets W with smooth boundaries. Conditions for F and W under which such formulae exist are known, but are too restrictive and not satisfied in many applications. However, certain higher than second order *nonlinear* quadrature formulae turns out to exist under essentially weaker conditions. They are in the form of

$$\tilde{J}_N = \operatorname{conv} \sum_{k=0}^{N-1} (A_k^N(F)\mathcal{D}(W)),$$

where A_k^N depend on F linearly and $\mathcal{D}$ is a certain nonlinear mapping, independent of N and k. Thus, given W, the image $\mathcal{D}(W)$ can be evaluated (in some cases analytically) and used in any formula of the above type. We discuss some particular cases and numerical aspects.

Linear-Time Algorithms that Locate Local Maxima and Minima of a Function from Approximate Measurement Results

K. Villaverde and V. Kreinovich

Computer Science Department
University of Texas at El Paso
El Paso, TX 79968, USA

The problem of locating local maxima and minima of a function from approximate measurement results is vital for many physical applications: in spectral analysis, elements are identified by locating local maxima of the spectra; in radioastronomy, sources and their components are located by locating local maxima of the brightness; elementary particles are identified by locating local maxima of the experimental curves.

In mathematical terms, we know n numbers $x_1 < ... < x_n$, and n intervals $I_i = [y_i^-, y_i^+], i = 1, ..., n$, where $y_i^- = y_i - \varepsilon$, $y_i^+ = y_i + \varepsilon$, and we know that the values $f(x_i)$ of the unknown function $f(x)$ at the points x_i belong to I_i. The set $\mathcal{F}$ of all the functions $f(x)$ that satisfy this property can be considered as a *function interval* (this definition was, in essence, first proposed by R. Moore himself). We say that an interval I *locates a local maximum* if any function $f \in \mathcal{F}$ attains a local maxima at some point from I. So, the problem is to generate intervals $I_1, ..., I_k$ that locate local maxima.

Evidently, if I locates a local maximum, then any bigger interval $J \supset I$ also locates them. We want to find the *smallest* possible locations I. We propose an algorithm that finds the smallest possible locations in linear time (i.e., in time that is $\le Cn$ for some n).

Remarks.

1. By looking for the smallest possible location, we want an *optimal* interval estimate in the sense of [R80] and [RR80] (see also [K86]).
2. There exist various algorithms that locate the *global maxima* of an intervally defined function (see, e.g., [M79], [DS83], [RR88], [M91]). For these algorithms, local maxima are the main obstacle that has to be overcome, and not the final result, so we cannot apply these algorithms to locate *all local* maxima.
3. Local maxima and mimima are also used in the methods that accelerate the convergence of the measurement result to the real value of a physical variable, and thus allow the user to estimate this value without waiting for the oscillations to stop [N88].

REFERENCES

[DS83] Dennis, J. E., and R. B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*, Prentice-Hall, Englewood Cliffs, N.J., 1983.

[K86] Kolacz, H. In: *Interval Mathematics 1985*, Springer Lecture Notes in Computer Science, Vol. 212, 1986, pp. 67–79.

[M79] Moore, R. E. *Methods and applications of interval analysis*, SIAM, Philadelphia, 1979.

[M91] Moore, R.E. *Computers and Mathematical Applications*, Vol. 21, 1991, No. 6/7, pp. 25–39.

[N88] Nickel, K. In: *Reliability in Computing*, Academic Press, N.Y., 1988, pp. 151–169.

[R80] Rall, L. B. In: *Interval Mathematics 1980,* Academic Press, N.Y., 1980, pp. 489–498.

[RR80] Ratschek, H., and J. Rokne. In: *Interval Mathematics 1980,* Academic Press, N.Y., 1980, pp. 499–508.

[RR88] Ratschek, H., and J. Rokne. *New computer methods for global optimization*, Ellis Horwood, Chicester, 1988.

Verified Computation of Pole Assignment by Complete Modal Synthesis

P. Walerius and G. Ludyk

Inst. f. Automatisierungstechnik, Universität Bremen
Kufsteiner Str. / Geb. NW 1, W-2800 Bremen 33, Germany

In control engineering Pole Assignment is a method to synthesize a control law. The aim of synthesis is to give a system some chosen properties. A fundamental requirement is to guarantee stability.

In this work, the controller design for multivariable linear systems will be realized by using state variable feedback or output feedback. Under the condition that the system is controllable, the dynamic behaviour that is determined by its eigenvalues can be changed arbitrarily. The Complete Modal Synthesis delivers an explicit controller formula for calculation of the unknown feedback matrix. The proposed methods for verified computation of the feedback matrix is based on this formula which is transformed into a system of nonlinear equations where the unknown variables are the coefficients of the feedback vector.

By application of inclusion procedures for the solution of nonlinear systems, the computation of the feedback vector for the pole placement problem can be carried out with high precision and verification.

The computed results with the new method are in the form of precise inclusion intervals of the feedback vectors. It is furthermore always guaranteed that the exact solution lies in the computed intervals.

FORTRAN–XSC
A Portable and Versatile Fortran: 90 Module Library for Highly Accurate and Reliable Scientific Computing

W. V. Walter

Institut f. Angewandte Mathematik, Univ. Karlsruhe
Kaiserstr. 12, D–7500 Karlsruhe 1, Germany

The new Fortran: 90 standard offers a multitude of enhancements and extensions of the old FORTRAN: 77 language. Many new features and concepts such as dynamic arrays, array operators, pointers, modules, and user-defined data types and operators are essential for contemporary programming. However, the mathematical properties of the arithmetic operators and the mathematical elementary functions, in particular any accuracy requirements, still remain unspecified. In this sense, the Fortran: 90 standard is still arithmetically and numerically deficient.

FORTRAN–XSC is a versatile toolbox intended for use in a wide range of numerical applications. The library consists of a number of Fortran: 90 modules providing accurate scalar, vector and matrix arithmetic for real and complex numbers and intervals, accurate conversion routines for numerical constants and input/output data, multiple precision arithmetic, a reliable and highly accurate implementation of the Fortran: 90 intrinsics SUM, DOTPRODUCT MATMUL and of the BLAS (Basic Linear Algebra Subprograms), and more. The user has full control of the rounding mode to be used in an operation. Every operation is guaranteed to be accurate to 1 ulp (unit in the last place).

FORTRAN–XSC is particularly useful for the development of self-validating numerical algorithms. Such algorithms deliver results of high accuracy which are verified to be correct by the computer, so there is no need to perform an error analysis by hand. For example, self-validating numerical techniques have been successfully applied to a variety of engineering problems in soil mechanics, optics of liquid crystals, ground-water modelling and vibrational mechanics where conventional floating-point methods have failed.

FORTRAN–XSC is written in pure standard-conforming Fortran: 90 and is fully portable. The module library automatically adapts to the native floating-point system of the machine used and exploits the hardware arithmetic as much as possible. The only requirement is that the arithmetic of the machine be faithful, that is, that the elementary operations provide least-bit accuracy.

Interval Test: An Application of Interval Arithmetic in Data Dependence Analysis

Zh. Xing and W. Shang

Center for Advanced Computer Studies
University of Southwestern Louisiana
Lafayette, Louisiana 70504 USA

Data dependence analysis is a basic step in detecting loop level parallelism in numerical programs. Testing if there is a dependence in a loop can be converted to checking if there exist integral points in a polyhedron described by a set of linear equations and inequalities. Several methods on the data dependence test have been proposed. Some of those methods only consider *single dimension test*, i.e., they either consider cases where the polyhedron is described by only one linear equation and some inequalities, or for the cases where multiple linear equations are involved, they test those linear equations separately and independently, which often results in an inaccurate test. Some *simultaneous data dependence test* methods consider all linear equations and inequalities together to have more accurate results, but they have exponential complexity of the number of loops. In this paper, interval operations are used. This greatly simplifies the previous work and also gives a new efficient single dimension test called *interval test*. Consequently, simultaneous data dependence analysis is achieved by transforming equivalently a data dependence test problem with any number of linear equations to a test problem with only one linear equation and applying some single dimension test methods such as the GCD test, Banerjee's test, I-test and interval test. The method reported in this paper overcomes the inexactness of traditional test methods which check dimension by dimension rather than simultaneously checking, and improves many previously proposed methods in aspects of accuracy, application and efficiency.

Possibilities for Further Development of SC-Languages

A.G. Yakovlev

Moscow Institute of New Technologies in Education
Nizhnyaya Radischevskaya 10, Moscow, 109004, Russia

In recent years an entire family of so-called SC-languages (languages for Scientific Computations) along with program systems implementing these languages has been developed. The family includes ACRITH-XSC [1] (FORTRAN-SC [2]), PASCAL-SC [3] and -XSC [4], MODULA-SC [5], C-XSC [6], etc. Characteristic features of the languages are: high accuracy of numerical operations, in particular, of specially implemented scalar products and elementary functions, flexibility in controlling rounding directions for numerical results to be computed, availability of various means for programming interval computations and so on. Also, the languages have a number improvements of a universal nature. For instance, they usually support modularity. In general, the languages are primarily based on comfortable and effective programming of localizational (self-validating) numerical computations. However, the contemporary technical base and achievements in software design allow some additional concepts to be implemented, increasing the advantages of SC-languages. Among these concepts are

- **multi-aspectness**, allowing operation on different representations of the same mathematical objects either simultaneously or separately [7-8];
- **parallelism** of operations, computing methods and their versions, allowing not only increased speed, but also the accuracy and reliability of localizational computations [7, 9];
- •recomputation, based on the idea of repeated localization of intermediate results. In particular, this permits all available computing resources to be concentrated on increasing the accuracy of the final result accuracy [7, 9];
- **analytical evaluations** that become implementable because of a special organization of a symbolic-numeric interface. This permits implementation of combined (numerical-analytical) computing methods [8];
- **classifying** that allows increased expressiveness and tractability of programming when programs with a complicated logical structure have to be written [10];
- **special objects** and **exceptions** allowing fuller use of the information contained in initial data, and allowing avoidance of many by explicit tests of conditions which would obfuscate a program text. [11]; etc.

To support some of these concepts, new language constructs and methods of their implementation have been developed. Other concepts can be implemented by constructs existing in other programming languages.

In general, introduction of the proposed concepts into SC- languages allows increased accuracyas necessary, increased speed and reliability of algorithms, increased expressiveness and tractibility of programming, and also more flexibly to adjust the relation "quality of the final result / usage of computing resources."

References

1. ACRITH-XSC. IBM high accuracy arithmetic - extended scientific computation. Version 1, release 1. IBM Deutschland GmbH, 1990.
2. Bleher J.H., Rump S.M., Kulisch U., Metzger M., Ullrich Ch., Walter W. FORTRAN-SC. A study of FORTRAN extension for engineering/scientific computation with access to ACRITH. *Computing* **39**, 93–110 (1987).
3. PASCAL-SC: A computer language for scientific computation (G.Bohlender, C.Ullrich, J.Wolff von Gudenberg, L.B.Rall, eds.). Academic Press (Perspectives in Computing, vol. 17), Boston etc., 1987.
4. Klatte R., Kulisch U., Neaga M., Ratz D., Ullrich Ch. PASCAL- XSC: Language reference with examples. Springer Verlag, Berlin etc., 1991.
5. Falco Korn C., Koenig S., Gutzwiller S. MODULA-SC. A precompiler to MODULA-2. In *Intern. Symp. on Computer Arith- metic and Self-Validating Numerical Methods, Basel, October 2-6, 1989*, p. 41. University of Basel, 1989.
6. Lawo Ch. C-XSC. A programming environment for verified scientific computing and numerical data processing. Karlsruhe, Institute of Applied Mathematics, 1992.
7. Yakovlev A.G. On some possibilities in organization of localizational (interval) computations on electronic computers. Inf.-operat. material (interval analysis), preprint 16, Computer Center, Siberian Branch of the USSR Academy of Sciences, Krasnoyarsk, pp. 33-38 (1990) (in Russian).
8. Yakovlev A.G. Multi-aspectness in programming of localizational (interval) computations. *Proc. Seminar on Interval Mathematics, May 29-31, 1990, Saratov*, pp. 113-120 (1990) (in Russian).
9. Yakovlev A.G. Specific parallelism of localizational computations. Proc. *All-Union Conf. on Actual Problems of Applied Mathematics, Saratov, May 20-22, 1991, Saratov*, pp. 151–158 (1991) (in Russian).
10. Yakovlev A.G. Classification approach to programming localizational (interval) computations. *Interval computations* **1**, 1992 (to appear).
11. Yakovlev A.G. Machine arithmetic of multi-intervals. *Voprosy Kibernetiki* **125**, pp. 66–81 (1987) (in Russian).

On Some Problems of Best Approximation in Interval-Segment Analysis

V. S. Zyuzin

Department of Mechanics and Mathematics
Computer Center, Saratov State University
Astrakhanskaya 83, Saratov, Russia 410071

The definitions of algebraic polynomials which deviate least from zero and meet some requirements are introduced in terms of interval-segment analysis. There is a special algebraic polynomial in many variables deviating least from zero which has one linear dependence. With this polynomial and interval Taylor series, we can construct polynomials close to the best ones.

A Method for Finding Complex Intervals Containing Zeros of Nonlinear Equations

V. S. Zyuzin and L. V. Kupriyanova

Department of Mechanics and Mathematics
Computer Center, Saratov State University
Astrakhanskaya 83, Saratov, Russia 410071

In this communication, a method and a program in PASCAL-SC for finding complex intervals containing the zeros of nonlinear functions will be presented. The communication continues the work "On one way of finding intervals containing zeros of nonlinear equations" presented at the conference "INTERVAL-92." In contrast to other known methods, the method developed here allows determination of all intervals containing both real and complex zeros on the whole complex plane. Numerical examples will be presented.

ABSTRACTS

for

A Workshop on

Interval Methods in Artificial Intelligence

February 25 through March 1, 1993

Lafayette, Louisiana

to be held in conjunction with the conference on

Numerical Analysis with Automatic Result Verification

The abstracts are listed in alphabetical order by last name of first author.

UNICALC - AN INTELLIGENT SOLVER FOR PROBLEMS WITH IMPRECISE AND SUBDEFINITE DATA

A.B. Babichev, O.B. Kadyrowa, T.P. Kashevarova, A.L. Semenov
Novosibirsk Branch of the Russian Research Institute of Artificial Intelligence
Novosibirsk 630090, Russia, email semenov@isi.itfs.nsk.su

A new calculus apparatus based on one approach to knowledge representation and processing was suggested in [1]. To implement this apparatus the UniCalc solver which can both solve new classes of problems and deal with more usual tasks in a very unusual but efficient way was developed.

UniCalc is designated to solve arbitrary algebraic and algebraic-differential systems joining equations, inequalities and logic expressions. A system may be subdetermined or overdetermined and it's parameters (coefficients, variables, constants, Cauchy initial values) may be given imprecise (as intervals). Both real and integers intervals may be used to find solutions of corresponding types. Any initial approximation to solution is not required. The solution of an algebraic system is represented by a parallelepiped containing all system roots. The appropriate message is issued if a system has no roots. A point (with given accuracy) will be output for a system with unique root.

The solver is a multiwindow integrated environment providing all necessary capabilities: an input and modification of systems to be solved, calculations, an output of solutions in desired form and so on. An input language allows to formulate a problem in the usual math notation and built-in multiwindow editor has all necessary functions for simultaneous input and editing of several systems. All kinds of problems (precise and interval, real and integer) are solved by the unique original algorithm. The solver contains a compiler, some preprocessors (including a preprocessor for symbolic manipulation) and the kernel: a data-flow processor. The processor implements a concept of "generalized computational model" [1] being a data-flow model of knowledge processing. This model is a non-deterministic parallel asynchronous process of subdefinite calculations. To deal with subdefinite data the processor uses algorithms of interval arithmetic.

To estimate efficiency of the UniCalc a lot of different problems was solved. We solved linear and nonlinear algebraic systems of equations and inequalities, different kinds of integer problems, optimization problems, interval problems, systems of ordinary differential equations and so on. In general, results of testing were very successful especially for complicated nonlinear systems and integer problems. Also we have solved an optimization problem for Rozenbrock's functions (up to 15-th order) and Powell's function.

Current UniCalc release runs on IBM PC/MS DOS and requires 450Kb of RAM. This configuration permits to solve systems up to 250 variables and up to 300 expressions.

References

1. A. S. Narin'yani. Subdefiniteness in the Knowledge Representation and Processing System. Izvestiya USSR Acad. of Sciences, Technical Cybernetics, 1986, No. 5, pp. 3–28 (In Russian).

WHY LINEAR REGRESSION METHODS WORK SO WELL FOR NON-LINEAR PROBLEMS?

M. Baker[1], D. Doser[1], V. Kreinovich[2], V. Kozlenko[3]
Department of Geology[1] and Department of Computer Science[2]
University of Texas at El Paso, El Paso, TX 79968
[3]Parashutnaya 12-345, St. Petersburg 197341, Russia

In many real-life situations, we must estimate the value of a physical quantity y that is difficult to measure directly. So, to avoid direct measurements, we measure whatever variables we can, and then based on the measured value $x_1, ..., x_n$, try to estimate y. These situations is most frequent in geology, when it is very costly to measure the properties of the deep layers, and much cheaper to measure the waves reflected from those layers.

In some situations, we know the relationship between x_i and y, so we can use this known model to estimate y based on x_i. n geology, this relationship is usually highly non-linear. In many situations, however, this dependency between x_i and y must be determined experimentally. There exist many statistical methods that help to discover such a dependency, the simplest of them (linear regression methods) help to discover linear relationships $y = \sum_i a_i x_i$. For the cases when we are not sure whether the relationship is linear or not, standard statistical procedures require that we first try simple linear regression methods, and then, if linear methods do not work, try more complicated non- linear methods.

In geology, since the relationship between different parameters is highly non-linear, we would expect that in the majority of the cases linear regression methods would fail. Unexpectedly, in the majority of cases, linear methods succeed! The same strange phenomenon occurs in economics: when we, e.g., analyze the dependency of the workers-per-manager ratio on the parameters that characterize the business, we also get a pretty good fit for linear regression in an evidently non-linear situation. Why are linear methods working so well in non-linear situations?

In the present report, we present an answer to this question. Of course, if all the variables x_i are independent, and the actual dependency of y on x_i is non-linear, then linear regression methods cannot work. But in many cases, the values x_i are not independent. For example, in geological measurements, if in reality, we have a 3-layer situations, then we need, say, 9 parameters to describe these 3 layers. So, if we measure the values of reflections in 30 different point, the results of these measurements are evidently interdependent. Let us show how interdependency leads to the applicability of linear regression formulas. Let us consider the simplest case, when all the variables x_i depend on one parameter s. The dependency $x_i(s)$ is non-linear for all i. Since we measure x_i with a certain precision ε, we actually have the interval dependency that associates with each s the interval $[x_i(s) - \varepsilon, x_i(s) + \varepsilon]$. Hence, we can use the approximate formula for $x_i(s)$ as soon as it lies inside that interval. In particular, if the actual dependency is smooth (which is most often the case), we can retain several terms in the Taylor expansion of $x_i(s)$, and thus consider the case when all these dependencies are of the type $x_i(s) = a_i^{(0)} + a_i^{(1)} s + a_i^{(2)} s^2 + ... + a_i^{(k)} s^k$ for some k. Likewise, $y(s) = a^{(0)} + a^{(1)} s + a^{(2)} s^2 + ... + a^{(k)} s^k$. So, y and each of x_i belong to a $(k+1)-$dimensional linear space of all linear combinations of $k+1$ functions $1, s, ..., s^k$. Hence, if $n > k$, we have $n+1 > k+1$ vectors in an $(k+1)-$dimensional space. Therefore, these vectors are linearly dependent, i.e., y can be represented as a linear combination of x_i.

The same conclusion can be obtained when x_i depend on several parameters s_j.

Our conclusion: if linear regression works in a non-linear situation, one does not need to search for an error. Moreover, if this is the situation, then we can be sure that the variables x_i are interdependent, so we can look for the dependencies between them.

TOWARDS INTERVAL SEMANTICS FOR LOGIC PROGRAMS

Chitta Baral
Computer Science Department
University of Texas at El Paso, El Paso, TX 79968

For logic programs, one of the most reasonable semantics is the so called *stable model* semantics, that was proposed by M. Gelfond and V. Lifschitz. According to this semantics, if we ask a query Q, the answer is "yes" if all reasonable sets of beliefs include Q, "no" if none of the reasonable belief sets include Q, and "unknown" if Q is included in some sets of beliefs but not in all of them. A belief set is a set of atomic statements in which someone believes. It is called reasonable if, crudely speaking, the following is true: if we start with this set of beliefs B, and apply all the rules from a given logic program, then the resulting set of beliefs $T(B)$ coincides with B. In other words, everything that can be concluded from this set of beliefs is already there. Such a reasonable set of beliefs is called a *stable model.*

One of the main problems with stable sets is that for some logic programs there is no stable set at all. The simplest example is a program $p \leftarrow \textit{not } p$. The reason why it has no stable set is that in stable model semantics, it is interpreted as "if we do not believe in p, then we believe in p", which makes no sense. In more complicated cases, the reason is not so evident. One can argue that such programs represent inconsistent knowledge. This may be true, but such programs do appear when we formalize human reasoning. Since our goal is to formalize commonsense reasoning, and not to criticize it, it seems desirable to generalize stable model semantics so that it would be applicable to such programs as well.

Such a generalization (called *stable model semantics*) was proposed in [1]. In the situations where no stable model exists, we can have several people with different sets of beliefs. None of these sets is stable. Therefore, after we apply the rules, each of them changes his set of beliefs. But sometimes, if one chooses a sufficiently broad spectrum of opinions, one can guarantee that (although each expert changes his opinion) the set of all their opinions remains the same. So, in this case, although each set of beliefs is not stable, but the class $\mathcal{S}$ of these sets can be stable (in the sense that $T(\mathcal{S}) = \mathcal{S}$). In this case, we can say that an answer to a query is "yes" Q is true in all the belief sets that belong to each stable class, "no" if Q is false in all these belief sets, and in all other cases it is "unknown". It is proved that an arbitrary program has a stable class, so this semantics is well defined.

This semantics has a slight drawback: if Q is true in some of the belief sets, and false in some others, the answer is always "unknown". However, if Q is true in 90% of belief sets, then we have more reasons to believe in Q than if it is true in 10% of belief sets. In the present paper, we show how to add this distinction to stable class semantics.

The idea of this addition is simple: Let us again consider the group of people with different views, but now we will not only consider the set of those views, but also count how many of them hold to each view. This set will be called stable if after applying T not only the classes remain the same, but the numbers as well. For such stable classes, for each query Q, we can compute the ratio of those who believe in Q. We prove that such stable classes exist for an arbitrary logic program, and that for each program, the set of ratios that correspond to different stable sets form an interval. "Yes" and "no" answers correspond to degenerate intervals [1,1] and [0,0], but, say, [0.1,0.2] and [0.8,0.9] are evidently different cases of what was before classified as "unknown".

Reference.

C. Baral. Ph.D. Dissertation. University of Maryland, College Park, 1991 (and references therein).

INTERVAL RESTRICTIONS HELP IN DESIGNING GRAPHICAL INTERFACES FOR MACHINE CONTROL

Ray Bell
Computer Science Department
University of Texas at El Paso, El Paso, TX 79968

Modern computer-aided design (for example, design of a car or a plane) starts with the designer's testing and comparing different shapes and choosing the best ones. There exist many different graphic tools that help; these tools make the task of forming a graphical representation really easy. For example, graphics is included in wide-spread Turbo C and Turbo Pascal compilers. With many software packages, even kids can easily design graphical images.

After one or several shapes are chosen based on computer simulations, it is necessary to make models of these shapes to test their aerodynamical, optical, and other physical properties. There exist machines that cut wood or metal following a computer program. Unfortunately, we cannot directly translate the image generated by a graphical program into the instruction for these machines. The reason is that these machines require the manufacturing program to be written in a special low-level language, and they do not understand the high-level languages that are used for graphics. As always with low level languages, these programs are difficult to debug. For a normal program debugging only increases time. For a manufacturing program, in addition to wasting the programmers' time, we have to make several wrong cuts before we finally fix the program. Therefore, it is desirable to develop something like a compiler from a graphic language to the language of machine control. It is difficult to write a compiler, because graphical packages deal with 2-D images (2-D projections of a 3-D shape), and for manufacturing, we need to know a 3-D shape itself.

So what we really need is an *interface* between the graphical software and the machine control software.

We describe the general idea of such an interface, and the results of an actual implementation of this idea (on the basis of Turbo Pascal graphics).

The idea is as follows. It is very easy to describe a 2-D projection using a graphic package. To describe a 3-D image, we must somehow represent the third coordinate of each point using the same 2-D representation. In other words, to describe the shape of a surface $z(x,y)$, we must somehow store the values of $z(x,y)$ that corresponds to each pixel (x,y). In graphic packages, the only information that we can store for each pixel is its color. So, we must use color of a point (x,y) to encode the 3-rd coordinate of a point whose projection is (x,y). In the majority of terminals, there are not so many colors: the number of colors is much smaller than the number of possible different real numbers z that can be represented in a computer.

But for design, we do not need supermicron precision. So, in addition to the desired shape $z_d(x,y)$, a value $\delta > 0$ is given, and for each x and y, such that the resulting shape $z(x,y)$ must belong to the interval $[z_d(x,y) - \delta, z_d(x,y) + \delta]$. If we take this fact into consideration, then for reasonable δ, the number of essentially different depths z is smaller than the number of colors on a monitor, and therefore, we can use a convenient color coding to represent 3-rd dimension.

The resulting system is easy to use, and it takes as much time (a few minutes) to program machine control to cut a shape, as it takes to draw this shape on the screen. Besides, since the same program is used for drawing and cutting, there is practically no need for debugging: if something is wrong, we will see it graphically.

Reference

R. Bell. *Graphical Interfaces for Machine Control*, Technical Program, ACM Rio Grande Chapter Winter Meeting, 1991, p. 6.

REASONING WITH INTERVALS AND PROBABILITY DENSITY FUNCTIONS: A CONSTRAINT PROPAGATION APPROACH

Daniel Berleant
Computer Systems Engineering, 313 Engineering Building,
University of Arkansas, Fayetteville, AR 72701, (501) 575-5590, djb@engr.uark.edu

Consider an multiplication constraint among 3 interval valued quantities. The reasoning process for using that constraint to narrow its operand quantities is quite different from solving an interval equation $AB = C$. For one thing, given C and $AB = C$, narrowing A implies widening B. Yet a multiplication *constraint* among interval-valued quantities A, B, and C means that $ab = c$, and narrowing A implies narrowing B.

In this paper we explain the difference between constraint propagation with interval-valued labels, and evaluation of interval equations. A body of Artificial Intelligence work on constraints and intervals of significant size is reviewed, and contrasted with analogous work in the interval mathematics field.

Viewing constraint propagation as the well-known AI algorithm called Waltz filtering, a novel proof of Waltz filtering using techniques of program verification is given, which applies to interval-valued and other label sets of infinite magnitude, unlike most other proofs of the validity of Waltz filtering. Termination of the algorithm is also dealt with. We suggest using two different epsilons in rounding interval calculations, one to insure correctness, as suggested by Moore [1979], and one to ensure termination, as suggested by Davis [1987], and briefly describe how the size of the termination epsilon affects termination time.

We then develop a method of working with operands some of which are intervals and others of which are probability density functions. Pilot work has applied this to an M-of-N problem in dependability analysis. The pdfs are discretized into a set of intervals and combined using the "numerical combination of random variables" (or "histogram discretization") method popularized originally by Colombo and Jaarsma [1980]. The histogram discretization method is extended to the case of some operands being intervals, by treating the interval as a pdf discretized as a histogram containing one bar.

Describing the results in the general case requires using the concept of first order stochastic dominance, but in the context of finding the probabilities of qualitative behaviors, interval bounds on those probabilities may be found instead.

STEP-SIZE REFINEMENT: A NEW INTERVAL ALGORITHM FOR FLEXIBLE SIMULATION OF ODE'S

Daniel Berleant
Computer Systems Engineering, 313 Engineering Building,
University of Arkansas, Fayetteville, AR 72701, (501) 575-5590, djb@engr.uark.edu

By preprocessing an ODE with qualitative simulation to describe its qualitative behaviors, each behavior may be further simulated by inserting quantitative time points between the qualitative time points. This approach leads naturally to measurement interpretation, in which not only time point values but values of other variables are inserted into the behavior description. As simulation proceeds, the quality of the inferences increases. Unlike most methods of numerical simulation, this approach will not produce worsening results with excessively small step sizes.

WHEN SOME INPUTS ARE PDF'S AND OTHERS ARE INTERVALS

Daniel Berleant
Computer Systems Engineering, 313 Engineering Building,
University of Arkansas, Fayetteville, AR 72701, (501) 575-5590, djb@engr.uark.edu

Monte Carlo simulation is well-established for applications in which all uncertain inputs are pdfs, and for applications in which all uncertain inputs are intervals. However, dealing with situations in which some inputs are pdfs and some are intervals is difficult using Monte Carlo techniques. We attack this problem by histogram discretization of pdfs, in which the domain is partitioned into intervals [Ingram et al. 1968; Colombo and Jaarsma 1980]. The result is many subproblems, each dealing with interval inference only. Recombination in general requires use of first order stochastic dominance to describe the results, however, if our interest is in bounding probabilities of qualitative behaviors, recombination can instead give us intervals bounding the probabilities of those behaviors. We show these ideas in the context of a simple example. While scaling up to larger problems leads to computational complexity problems, simple problems are also of widespread applicability, and in fact Klir [1991] argues that simple models are often better than their more complex counterparts.

WALTZ FILETRING IS CORRECT

Daniel Berleant
Computer Systems Engineering, 313 Engineering Building,
University of Arkansas, Fayetteville, AR 72701, (501) 575-5590, djb@engr.uark.edu

We use program verification techniques to prove that an AI technique called Waltz filtering is correct. A proof of this sort brings into sharp relief the assumptions that must be made for Waltz filtering to be correct. These assumptions are that constraints must not be time varying and that shrinking the label set at one node must not result in expanding the label set at another node. With interval-valued label sets, the magnitude of the label sets are infinite, and termination becomes an issue in the Waltz filtering settling process. Davis (in his 1987 paper in Artificial Intelligence Journal) and Moore [1979] propose changing bounds only if the change exceeds some epsilon. Davis addresses termination, whereas Moore addressed correctness of inferences, in standard floating point computer arithmetic. We suggest using 2 different epsilons, one for each purpose, and briefly analyze how time-to-termination depends on the epsilon chosen for this purpose.

SUB-DEFINITE CALENDAR SCHEDULING

S. B. Borde
Russian Research Institute for Artificial Intelligence,
P.O. Box 111, 103001, Moscow, Russia, e-mail nar@isi.itfs.nsk.su

Situations when the time parameters for specific works cannot be specified definitely are typical for practical calendar scheduling. If this is due to incomplete information about the time when some work has to be fulfilled, then we shall call such scheduling to be sub-definite.

An approach to tackle this problem is based on the temporal logic (the so-called T–model proposed by E. Y. Kandrashina) and the theory of sub-definite models (proposed by A. S. Narin'yani) developed in the Russian Research Institute of Artificial Intelligence. The T–model provides means to specify the relations between individual events. Time is modeled as a straight directed line T. Traditional notions such as time point (t–point), time interval (t–interval) and more complex ones are included into the T–model.

Sub-definite model (or n–model from Russian "nedoopredelionnyj") generalizes the notion of a computational model. The inference process on the n–model is determined by a special kind of a data-driven virtual machine.

In the case of the discrete T–model, the t–point is modeled by integer. The sub-definite t–point can be presented by the interval of integers $[a, b], a \leq b$.

To represent the event lifetime, the T–model introduces the notion of the t–interval. The t–interval A is defined as a triplet of units $(s(A), f(A), d(A))$, where $s(A)$ and $f(A)$ are non-identical t–points, one being the start point of the t–interval and another being its finish point respectively, and $d(A)$ is the duration of A, which are bound with the obvious correspondence.

The start and finish points of the sub-definite t–interval A are sub-definite t–points and the duration is sub-definite quantity. They are bound by the functionally interpretable n–relation:

$$d(A) = f(A) - s(A). \tag{1}$$

The n–schedule is treated as n–model $M = (X, R)$, where X is a set of sub-definite t–intervals and R is a set of relations binding the sub-definite t–intervals of X. The sub-definite t–interval models any activity that is continuously performed during some time interval. R includes the relations (1) for every sub-definite t-interval of X and all relations binding sub-definite t-intervals.

The relations binding sub-definite t–intervals are n–extensions of relations binding t–intervals in T–model. An example of such relations is:

(a) $EMBEDDING(A1, A2) \leftrightarrow (s(A1) \leftarrow s(A2))\&(f(A2) \leftarrow f(A1))$

(b) $SUCCESSION(A1, A2) \leftrightarrow f(A1) \leftarrow s(A2)$

(c) $NON-SIMULTANEITY(A1, A2) \leftrightarrow SUCCESSION(A1, A2) \vee SUCCESSION(A2, A1)$,

where A1 and A2 stand for sub-definite t–intervals.

The inference process on the n–schedule results in the reduction gaps between the boundaries of intervals of start and finish points and durations of sub-definite t–intervals. The lower boundary of an interval cannot exceed its upper boundary. If this condition is violated, it implies that the n–schedule is inconsistent.

The Time-EX system intended for elaboration, optimization and management of calendar schedules was developed on the basis of this approach. The most important advantage of the Time-EX system is that a user deals not with a particular explicit variant of the schedule but rather with a n–schedule representing in compact and clear form all the schedules which satisfy limitations. The user can ever allow the final variant to remain sub-definite enough to ensure a sufficient time lag to maneuver under changing conditions during the schedule implementation.

A HIGH LEVEL LANGUAGE TO DEAL WITH MULTISETS: DISCRETE ANALOG OF INTERVALS

Daniel E. Cooke
Computer Science Department, University of Texas at El Paso, El Paso, TX 79968

If a physical quantity can (in principle) take arbitrary real values, and we do not know the precise value, then usually, possible values form an interval. For example, if we measure voltage V with precision ε, and the measurement result is equal to $\tilde{V}$, this means that the actual voltage belongs to an interval $[\tilde{V}-\varepsilon, \tilde{V}+\varepsilon]$.

One of the main objectives of interval mathematics is to analyze the following situation. Suppose that we know the interval of possible values $X_1, ..., X_n$ for several quantities $x_1, ..., x_n$, and we know that some other quantity is related to x_i by a formula $y = f(x_1, ..., x_n)$. If we take different values $x_i \in X_i$, we will end up with different values of y. How to describe the set Y of possible values of y? The existing methods of interval mathematics work pretty well in this situation.

A similar problem appears when the set of possible values of x_i is discrete. For example, we may know that x_i takes only integer values, but we are not sure what the value is, so there are several possible values. Such situations occur in quantum physics, where many variables (spin, angular momentum, charges, etc) can take only the values from some discrete set (spectrum). In this case, if we know the approximate value of x_i, and know the precision, then the possible values of x_i form an (ordered) finite set.

If we know the finite set X of possible values for x, and we know that some other variable y is related to x by a formula $y = f(x)$, then we would like to determine the set of possible values Y for y.

Of course, for every such situation, we can write a special program that computes this set Y, without inventing any new formalism. But, just like in the case of interval analysis, we would like to be able to do the following: to have a software tool that, given an expression for $f(x)$, and the description of X, would generate the set Y. Such a tool is proposed in the present report.

This tool works along the same lines as interval computations: the algorithm that computes $f(x)$ can be represented as a sequence of elementary arithmetic operations $(+, -, *, \text{etc.})$. As results of these steps, we get functions $f_1(x), f_2(x), ..., f_n(x)$ (where n is the number of computational steps). For example, for $f(x) = x - x^2$ we have $n = 2$, $f_1(x) = x * x$, and $f(x) = f_2(x) = x - f_1(x)$. We want to apply the same operations, in the same order, but to finite sets, and not to individual numbers. We start with the set X that is somehow ordered $(X = \{x_1, ..., x_m\})$. On k-th intermediate step, we want to compute the set of values $X_k = \{f_k(x_1), f_k(x_2), ..., f_k(x_m)\}$. This can be done, if we extend operations with numbers to elementwise operations with sets, so that, e.g., $\{a_1, a_2, ...\} + \{b_1, b_2, ...\} = \{a_1 + b_1, a_2 + b_2, ...\}$.

We propose a special language BagL, in which the user can write down any expression he wants, and the computer will automatically apply the same expression to numbers, sets, whatever. An interpreter for this language is written in PROLOG, and we are currently working on an independent (and more efficient) interpreter.

We are currently working on other potential applications, including processing satellite data.

For many problems, the programs in BagL are easier to write and comprehend, and they do not require fixing a datatype.

HOW TO DESIGN AN EXPERT SYSTEM THAT FOR A GIVEN QUERY Q, COMPUTES THE INTERVAL OF POSSIBLE VALUES OF PROBABILITY $p(Q)$ THAT Q IS TRUE

Luis Cortes
Computer Science Department, University of Texas, El Paso, TX 79968

The main objective of my research is to develop a method that for every knowledge base, in which uncertainty of the statements is expressed by probabilities, estimates the probability that a given query is true.

Since probability that a statement is true is known only approximately (corresponding error estimates can be determined by statistical methods), we must be able to generate not only the probability $p(Q)$ of a positive answer to a query Q, but also the error estimate for this probability. In other words, since the initial probabilities are not known precisely, we would like to generate an interval $[p-\varepsilon, p+\varepsilon]$ of possible values of the probability $p(Q)$.

So, the methods must take as an input:

- a *knowledge base*, i.e., a finite set of rules and facts E_i, with associated probabilities p_i and error estimate ε_i for these probabilities, and
- a *query* Q, and from this data, we must produce:
- the probability $p(Q)$ of a positive answer to this query, and
- an error estimate ε for this probability.

I have outlined an algorithm that gives such estimates, and currently I am working on implementing this algorithm in Prolog. The idea of this algorithm is that by means of random simulation we generate several random "worlds" (we organize a random simulation in such a way that each of the initial statements E_i is true or false with probability p_i). In each of these worlds there are no probabilities any more: it is just a knowledge base that contains some of the initial rules and facts E_i. Given such a world and a query Q, we can apply Prolog to find out whether Q is true in this particular world. After we repeat this procedure several times, we can estimate $p(Q)$ as the fraction of the worlds in which Q turned out to be true.

This idea works if we know the probabilities p_i precisely. If we know only approximate values $\tilde{p}_i$ of p_i (and also know the precision ε_i of these estimates), then in order to find the resulting error in $p(Q)$ we must:

- repeat this procedure several times with different simulated values $p_i = \tilde{p}_i + \nu\varepsilon_i$ (where ν is a simulated random variable), thus calculating several different estimates $\tilde{p}(q)$ for $p(Q)$, and then
- apply well-known methods of mathematical statistics to estimate the confidence interval for the true probability $p(Q)$.

One of the advantages of this Monte-Carlo method is that it is easily parallelizable, because each simulation can be done on a separate processor. Therefore, if we have sufficiently many processors, we can essentially decrease the total computation time.

MODELS OF FUZZY QUANTUM SPACES: WHAT IS THE QUANTUM ANALOGUE OF AN INTERVAL?

Anatolij Dvurečenskij
Mathematical Institute, Slovak Academy of Science
Štefánikova 49, CS-814 73 Bratislava, Slovakia

One of main reasons why intervals are important in image processing is that they represent uncertainty of the measurements: if we measure an arbitrary physical quantity x and get the result $\tilde{x}$, then because of the possible errors the actual value x may be different from $\tilde{x}$. The only thing that we know about x is that it belongs to an interval $[\tilde{x} - \varepsilon, \tilde{x} + \varepsilon]$, where ε is the maximal possible error of this measurement device. This number ε must be provided and guaranteed by the manufacturer of this device (if arbitrarily big errors were possible, then for any given $\tilde{x}$, arbitrary value of x would be possible, and therefore, this measurement would give us no information at all about the actual value).

These arguments are applicable to classical physics, where the value of each quantity is a number. In some cases, however, we have to take into consideration quantum effects. In quantum mechanics, the value of a physical quantity in a physical state is in general non-deterministic, and can be only represented by a probability distribution. The state itself can be represented as a so-called wave function (in the simplest cases, it is a complex-valued function).

Quantum measurements are also not precise. So, if for some observable x and some state s, the quantum measurement lead us to a probability distribution ρ, what can the actual distribution be? In other words, *what is the quantum analogue of an interval?*

In the present talk, we describe the quantum formalism that describes not only the uncertainty caused by measurement errors, but also the uncertainty of experts' estimates. For real values (that correspond to measurements in classical physics), this second type of uncertainty can be represented by fuzzy logic. Therefore, we called our models *fuzzy quantum (F-quantum) spaces.*

The standard model of quantum mechanics is the one in which the states are represented by vectors in a Hilbert space, and yes-no observables (i.e., observables, that have two possible values "yes" and "no") correspond to closed subspaces of a Hilbert space. For quantum field theory, this model is not sufficient, and it has been generalized to so-called *quantum logics.*

For our purposes, we use a model introduced by P. Suppes. This model is called a *quantum probability space* and is defined as a pair $(\Omega, \mathcal{Q})$, where Ω is a set, and $\mathcal{Q}$ is a collection of a subsets of Ω that so closed with respect to countable disjoint unions and complementation. Generalizing this notion, we introduced an *F-quantum probability space* as a pair (Ω, M), where $M \subset [0,1]^{\Omega}$ is a set of fuzzy subsets of the universe Ω that is closed with respect to countable disjoint unions and complements. More precisely, $1_{\Omega} = 1 \in M$; if $f \in M$, then $1 - f \in M$; if for a sequence $\{f_i\} \in M$, $\min(f_i, f_j) \leq 1/2$ for all $i \neq j$, then $\cup f_i = \sup f_i \in M$; and, finally, $1/2 \notin M$.

We define a *state* m as a mapping $m : M \to [0,1]$ such that, first, $m(a \cup a^{\perp}) = 1$ for an arbitrary $a \in M$, and, second, if $\min(a_i, a_j) \leq 1/2$ for all $i \neq j$, then $m(\cup_i a_i) = \sum_i m(a_i)$. An *observable* (an analogue of a random variable) is defined as a mapping x from the set of all Borel subsets of R into M such that, first, $x(R - E) = 1 - x(E)$, and, second, if $E_i \cup E_j = \Phi$ for $i \neq j$, then $x(\cup_i E_i) = \cup_i x(E_i)$.

Our main result is the representation theorem that allows us to describe arbitrary states and observables. Crudely speaking, states are in 1-1 correspondence with measures on $K(M)$, and observables - with measurable functions from $K(M)$ to R. Here, by $K(M)$, we denoted the set of all subsets $A \subseteq \Omega$ for which there is an $a \in M$ with $\{\omega : a(\omega) > 1/2\} \subseteq A \subseteq \{\omega : a(\omega) \geq 1/2\}$.

These results enable us to introduce the calculus of observables.

ERROR ESTIMATES FOR THE RESULTS OF INTELLIGENT DATA PROCESSING, ESPECIALLY NEURAL NETWORKS

S. Gulati[1], L. Gemoets[2], K. Villaverde[3]
[1]Neural Computation and Nonlinear Science Group
Space Microelectronics Device Technology Section, Jet Propulsion Laboratory
4800 Oak Grove Drive, Pasadena, CA 91109-8099, email sgulati@jpl-cray.jpl.nasa.gov
[2]Department of Computer Information Systems,
University of Texas at El Paso, El Paso TX 79968
[3]Department of Computer Science
University of Texas at El Paso, El Paso, TX 79968, email karen@cs.ep.utexas.edu

In data processing, it is vitally important to estimate the error of the result. For traditional algorithms of numerical mathematics (like numerical integration of differential equations, numerical solution of integral equations, etc), there usually exist methods of error estimate that have been developed inside numerical mathematics, and that give reasonably precise error estimates sufficiently fast.

Traditional methods work well when we analyze well known physical situations, for which sufficiently precise models are known. But this is often not the case in the situations like space exploration, where we do not have enough data to determine a precise model. Instead, we have to rely on experts' knowledge and/or simulated experts' experience (e.g., on artificial neural networks, that simulate the human brain). For these intelligent data processing methods (just like for traditional ones) the errors with which we measure the input data $x_1, ..., x_n$, lead to an error Δy in the result y. Unlike traditional data processing methods, however, for intelligent data processing, there are no known methods that would estimate the resulting error Δy. So, we have to use general methods of interval mathematics.

These methods are often too time-consuming. Besides, e.g., neural network methods can be easily implemented in hardware and thus made faster (at JPL, we developed hardware that simulates millions operations per second), and if we apply interval algorithms, we must do it in software and thus considerably slow down the entire data processing. In the current report, we discuss the methods that would allow, crudely speaking, to make neural networks perform interval computations. The resulting methods allow us to implement both the data processing and its error estimates in hardware, thus saving much time.

FAST INTERVAL ERROR ESTIMATES FOR THE RESULTS OF NON-LINEAR LEAST SQUARES METHOD: APPLICATION TO PAVEMENT ENGINEERING

C. Ferregut[1], S. Nazarian[1], K. Vennalaganti[1], Ching-Chuan Chang[2], V. Kreinovich[2]
[1]Department of Civil Engineering and [2]Computer Science Department
University of Texas at El Paso, El Paso, TX 79968

One of the main applications of interval mathematics to engineering problems is as follows: we want to know the value of some characteristic y that is difficult to measure directly (e.g., lifetime of a pavement, efficiency of an engine, etc). To estimate y, we must know the relationship between y and some directly measurable physical quantities $x_1, ..., x_n$. From this relationship, we extract an algorithm f that allows us, given x_i, to compute y: $y = f(x_1, ..., x_n)$. So, we measure x_i, apply an algorithm f, and get the desired estimate.

Usually, we know the precisions with which we measure all the x_i (i.e., the intervals of possible values of x_i). The problem is: how to estimate the precision, with which we know y (i.e., the interval of possible values of y)?

Standard techniques of interval mathematics require that we decompose the algorithm f into elementary operations, and then apply interval operations instead of usual ones. An operation with intervals consist of 2 or 4 operations with numbers. Therefore, by applying interval mathematics, we increase the computation time at least 2–4 times. In addition, the error estimates that we attain this way are often "overshoots" (i.e., much bigger than the biggest possible errors).

In the frequent situations, where the errors are relatively small, and thus we can neglect terms that are quadratic in errors, we can apply Monte-Carlo methods and sensitivity analysis. These methods also require that we run the algorithm f several times, thus drastically increasing the running time.

In order to diminish the total running time, let us recall where the relationships between x_i and y usually come from. Usually, there exists a *model* of the physical phenomenon, that has several adjustable parameters z_j (that are usually not directly measurable). In engineering language, a "model" means that we have (rather simple) formulas that, given z_j, allow us to compute all physical characteristics, including x_i and y: $x_i = F_i(\{z_j\})$ and $y = G(\{z_j\})$. Therefore, when we know x_i, we first determine the parameters z_j by model fitting (i.e., by solving the equations $x_i = F_i(\{z_j\})$), and then use these values z_j to compute y. This y is $f(x_1, ..., x_n)$.

The most time-consuming part of this algorithm is computing z_j. Since the functions F_i are usually highly non-linear, this is done by non-linear least square methods.

In the case when one can neglect quadratic terms, we can therefore simplify error estimation as follows: Here, $\Delta x_i = \sum_j (\partial F_i/\partial z_j)\Delta z_j$. The "forward" functions F_i are easy to compute, and so are the derivatives. So, as soon as we know z_j, we can easily compute the derivatives, invert the matrix, and get the expression for Δz_j in terms of Δx_i. Substituting these expressions into $\Delta y = \sum_j (\partial G/\partial z_j)\Delta z_j$, we get the expression of Δy in terms of Δx_i, from which we can easily get the desired estimates for Δy.

Since the most time-consuming part of the algorithm f is repeated only once, we get the estimate with practically no increase in total running time.

As an example of this methodology, we give pavement lifetime estimates.

QUASIORTHOGONAL DIMENSION AND CLASSIFICATION BY NEURAL NETWORKS

Paul C. Kainen[1], Věra Kůrková[2]
[1]Industrial Math, 3044 N St., N.W., Washington, D.C. 20007, KAINEN@CS.UMD.EDU
[2]Institute of Information and Computer Science
P.O.Box 5, 18207 Prague 8, Czechoslovakia, SVT@CSPGCS11.BITNET

The majority of classification methods are based on the following idea. We choose a set of characteristics. To every object, we put into correspondence a vector $\vec{x}$ formed by the values of these characteristics $x_1, ..., x_n$. Thus, each object is represented by a point in an $n-$dimensional space. If the characteristics are properly chosen (i.e., they are independent, and their scales are compatible), then the Euclidean distance $d(\vec{x}, \vec{y}) = \sqrt{\sum(x_i - y_i)^2}$ becomes a reasonable measure of similarity between the objects that correspond to $\vec{x}$ and $\vec{y}$. If they are similar, then this distance is small; if they are radically different, then this distance is big. In this representation, the classification problem can be formulated in the following way: We have a set of points in an $n-$dimensional space. How to divide this set into classes so that points within each class are close to each other and points from different subsets are far away from each other. There exist many methods of solving this problem, from simple statistical ones (that try to find linear hyperplanes that separate the classes), to neural network models, that can (in principle) successfully solve the classification problem for classes of arbitrary shape. In all these methods, when n increases, the computational complexity rapidly increases.

We wrote "in principle", because the big problem with these methods is that for large n the computations time of these algorithms increases very fast, and these algorithms become practically non-feasible.

In the present paper, we show that this computations time can be drastically decreased if we take into consideration the fact that the measurements that give the values x_i are only approximate. Therefore, for each object, we know only the interval that contains the actual value x_i. In view of that, the distances between different points are also only approximately known.

It turns out that this fact can be used to drastically reduce the dimension of the space of objects (and thus reduce the computations time). Namely, we can map an $n-$dimensional space into a space of fewer dimensions in such a way that the distances $d(f(\vec{x}), f(\vec{y}))$ between the images are close ($\varepsilon-$close) to the distances $d(\vec{x}, \vec{y})$ between the original points. So, if we know the distances only with precision ε, we can apply this mapping and get a new representation of our original problem with a smaller value of n.

To design and analyze such mappings, we introduce a new notion of quasiorthogonal dimension. In Euclidean space, dimension n can be defined as the biggest number of mutually orthogonal unit vectors $\vec{e}_1, ..., \vec{e}_n$ (i.e., vectors, for which $\vec{e}_i \cdot \vec{e}_j = 0$ if $i \neq j$). In our case, we must look for unit vectors $\vec{e}_i$ that are only approximately orthogonal, i.e., for which $\vec{e}_i \cdot \vec{e}_j \in [-\varepsilon, \varepsilon]$. The biggest number N of such vectors is called a *quasiortogonal dimension.* This value N can be much bigger than n, thus allowing to map an $N-$dimensional space into $n-$dimensional one. that (forming a base).

Using graph theory techniques, we derive lower bounds on the growth of N, and show that for fixed ε, it grows exponentially with n. For example, for $\varepsilon = 0.05$ and $n = 10,000$, $N \geq 1,700,000$. These interval-based estimates lead to a considerable reduction in complexity for neural networks used in classification tasks.

A few other applications of this new notion of dimension will be discussed.

INTERVAL-VALUED INFERENCE AND INFORMATION RETRIEVAL IN MEDICAL KNOWLEDGE-BASED SYSTEM CLINAID

Ladislav J. Kohout[1], Isabel Stabile[2]
[1]Department of Computer Science B-173
Florida State University, Tallahassee, Florida 32306, USA
[2]Academic Unit of Obstetrics and Gynaecology
The Royal London Hospital College, University of London, U.K.
and Center for Biomedical Research and Toxicology, Florida State University, U.S.A.

In a series of papers and a monograph [1], we have described the conceptual structures as well as the basic architecture of knowledge-based system CLINAID. Its architecture is aimed at supporting not only diagnosis but also other types of clinical activity and decision making in diverse clinical and/or hospital environments. In general, CLINAID generic architecture is aimed at support of knowledge-based decision making with risk and under uncertainty. Such a system has to operate in a multi-environmental situation and make decisions within a multiplicity of contexts.

The basic architecture consists of the following cooperating units (basic shell substratum):

1. Diagnostic Unit (comprized of several parallel cooperating centres)
2. Treatment Recommendation Unit.
3. Patient Clinical Record Unit.
4. Co-ordination and Planning Unit.

The majority of extant medical expert systems deal with a limited medical context, the largest domain of knowledge being just a single medical field, e.g. Internal medicine in CADUCEUS. The inherent limitation of such medical expert systems is in its essence conceptual and logical: their knowledge bases and inference engines cannot mix easily the knowledge from several fields without some adverse effects. CLINAID deals with this problem by introducing a multi-centre architecture in the Diagnostic Unit. The medical data and knowledge of each medical specialist field exhibit different logical properties. This in turn leads to the several kinds of many-valued logics on which the relational inference and data manipulation is based. The semantic justification of these logics is provided by a theoretical device called the *checklist paradigm* [2] which gives an epistemological justification for the interval-valued inference as well as for knowledge and data retrieval techniques utilizing information structures with interval credibility weights.

As CLINAID attempts to be a comprehensive medical consultation system, its knowledge has to contain a large amount of medical expert knowledge. The Diagnostic unit of CLINAID deals with a number of body systems [1]. In this paper we shall use the *cardiovascular* and *reproductive* body systems to demonstrate the need for interval based methods of inference and information retrieval. In particular, we shall pay attention to the question of how the character of *medical knowledge*, and the nature of the *retrieval* or *inference* task influence the *choice* of the base logic for the interval-based inference.

References

[1]. L.J. Kohout, J. Anderson, and W. Bandler. *Knowledge-Based Systems for Multiple Environments.* Ashgate Publ. (Gower), Aldershot, U.K., 1992.

[2]. L.J. Kohout and W. Bandler. *Use of fuzzy relations in knowledge representation, acquisition and processing.* In: L.A. Zadeh and J. Kacprzyk, editors, *Fuzzy Logic for the Management of Uncertainty*, John Wiley, New York, 1992.

DECISION MAKING AND GAME THEORY IN CASE OF INTERVAL UNCERTAINTY

Olga M. Kosheleva
Computer Science Department, University of Texas at El Paso, El Paso, TX 79968

In traditional decision making, to each possible outcome a there corresponds a value $u(a)$ called its *utility*, so that if an alternative x leads to the outcomes $a_1, ..., a_n$ with probabilities $p_1, ..., p_n$, and alternative y leads to a_i with probability q_i, we choose x over y if $\sum_i p_i u_i > \sum_i q_i u_i$ (here we denoted $u_i = u(a_i)$). The values of utility are determined by asking the experts to compare hypothetic outcomes with different probabilities p_i (such hypothetic outcomes are called *lotteries*).

Traditional utility theory is based on the following idealization: we can ask the expert's opinion about arbitrarily many different lotteries, and for every two lotteries, he will give us a precise answer: either that the first one is better, or that the second one is better, or that they are absolutely of the same value to him. In such a case, we can define all the values of the utilities uniquely modulo an arbitrary linear transformation $u(a) \to u'(a) = ku(a) + l$. Therefore, if we fix two outcomes a_0 and a_1 such that $a_0 < a_1$, and assume that $u(a_0) = 1$ and $u(a_1) = 1$, then, in this ideal setting, we can uniquely determine $u(a)$ for all outcomes a.

In real life, we can ask an expert to compare only finitely many pairs of lotteries, and for some of these pairs he may say "I don't know". With this information only, we cannot determine u_i uniquely. The corresponding finite set of inequalities defines a convex polytop in an $n-$dimensional space of possible vectors $\vec{u} = (u_1, ..., u_n)$. Therefore, for each outcome a $(= a_1, ..., a_n)$, we get an interval $[u^-(a), u^+(a)]$ of possible values. This interval can be computed by applying linear programming methods.

How to make decisions if we know only intervals for utilities? Let us first consider the simplest case, when we have to choose between n outcomes $a_1, ..., a_n$, for which only intervals are known. We want to choose an outcome a for which $u(a) \to \max$, where $u(a) \in [u^-(a), u^+(a)]$. For different functions $u(a)$, maximum may be attained for different a. So, what are the possible best choices? In this case, the answer is as follows: a is a possible best choice iff $u^+(a) \geq \max_b u^-(b)$, where max is taken over all outcomes b.

A more complicated case is when we have a zero-sum game, in which the first player has n pure strategies $1, ..., i, ..., n$, and the second player has m pure strategies $1, ..., j, ..., m$. Zero-sum games are easy to solve if we know precisely the values of utilities u_{ij} when the players choose strategies i and j. Then each player has to use a maximin strategy, i.e., choose $i-$th strategy with probability x_i, where $\min_{\vec{y}} u(\vec{x}, \vec{y}) \to \max_{\vec{x}}$ (here we denoted $u(\vec{x}, \vec{y}) = \sum_{ij} x_i y_j u_{ij}$). In real life, we know only intervals $[u_{ij}^-, u_{ij}^+]$ for payoff values. Depending on what u_{ij} form these intervals we choose, we get different optimal (maximim) strategies. How to describe the set of all strategies that can be optimal? The answer is as follows: a strategy $\vec{x}$ can be optimal, if $\min_{\vec{y}} u^+(\vec{x}, \vec{y}) \geq \max_{\vec{z}} \min_{\vec{y}} u^-(\vec{z}, \vec{y})$. Therefore, the set of possible values of the game coincides with the interval $[\max_{\vec{x}} \min_{\vec{y}} u^-(\vec{x}, \vec{y}), \max_{\vec{x}} \min_{\vec{y}} u^+(\vec{x}, \vec{y})]$.

We also describe how intervals influence the notions of solutions of a cooperative $N-$person game (Shapley value, Nash's scheme), and for optimism, pessimism, Laplace and other criteria for the case of complete uncertainty.

INTERVAL ESTIMATES FOR CLOSURE-PHASE AND CLOSURE-AMPLITUDE IMAGING IN RADIO ASTRONOMY

O. Kosheleva[1], A. Bernat[1], A. Finkelstein[2], V. Kreinovich[1]
[1]Computer Science Department, University of Texas at El Paso, El Paso, TX 79968
[2]Institute for Applied Astronomy, Russian Academy of Sciences, St. Petersburg, Russia

Closure-phase and closure-amplitude imaging are methods for reconstructing a radioimage from the results of approximate measurements.

The specific feature of very distant objects is that, although they may be physically large, due to the enormous distance their angular (visible) size on the sky is extremely small. For example, quasars have details of milliarcsecond size (0.001 of an arc second). Normal telescopes are unable to see such tiny details, because their *angular resolution* $\Delta\phi$ (ability to distinguish nearby points) is limited. A good approximation to this resolution is given by: $\Delta\phi \approx \lambda/D$, where λ is the wavelength of interest and D is the diameter of the telescope. Thus, in order to improve the angular resolution, it is necessary to increase D. But to distinguish details of distant radio sources ($\lambda \approx$ 21cm) we need D equal to several thousand kilometers. Of course it is technically impossible to build such a big telescope.

We can overcome this difficulty if we take into consideration the fact that in actual antennas the signal is received by several parts and we observe the superposition of the signals received by different parts. So, although we cannot build a single large antenna, we can simulate one if we place several antennas at different locations, collect the signals, send them to one place and there simulate the superposition by using a computer. This is called a Very Long Baseline Interferometry (VLBI).

As a result of this simulation, we have a signal that simulates what we would have received from a single large antenna. This signal is sinusoidal (with frequency equal to the frequency of observations) and can therefore be characterized by its amplitude and phase. These values are related to the function that describes the brightness $I(\vec{x})$ of the source that we are observing at point $\vec{x}$ (this function is called an *image*, or a *brightness distribution*). In the idealized situation, when we neglect noise and measurement error, the measured amplitude is proportional to the absolute value $A(\vec{b}) = |F(I)(\vec{b})|$ of the Fourier transform $F(I)$ of the desired image I, where $\vec{b}$ is equal to the projection of the vector $\vec{R}$ between two antennas onto the plane that is perpendicular to the source. Likewise the phase $\theta(\vec{b})$ equals the phase of the (complex) Fourier transform. So in this case, if we observe the same source on different pairs of antennas with different $\vec{b}$, we obtain both the amplitude A and the phase θ of $F(I)$, and thus reconstruct the complex value of $F(I)$ for every $\vec{b}$ as $A\exp(i\theta)$. By applying the inverse Fourier transform, we determine the desired image I.

To avoid additive errors, instead of using the measured phases, we form the "closure phases" $\theta(\vec{x}) + \theta(\vec{y}) - \theta(\vec{x} + \vec{y})$, and from them reconstruct θ.

If we know the measurements precision (i.e., the intervals for the measured values), what is the precision of the result of this reconstruction? This problem cannot be solved by standard interval methods because one of the measured quantities, the phase, takes its values on a circle, not on a real line.

In the present paper we give the desired estimates. The main result is that if we measure the phase $\theta(\vec{x})$ with precision ε (so that the closure phase is known with precision 3ε), then from these measurements we can reconstruct θ with precision 6ε. Similar estimates are given for closure amplitude.

INTERVAL, MEAN VALUE, STANDARD DEVIATION, WHAT ELSE? GROUP-THEORETIC APPROACH TO DESCRIBING UNCERTAINTY

Vladik Kreinovich
Computer Science Department, University of Texas at El Paso, El Paso, TX 79968

Traditionally, probability theory is used to describe the uncertainty of the measurements. The reason for that is very simple: for an arbitrary measuring device, if we make sufficiently many measurements, and compare them with the values measured by a standard, we can get the probability distribution on the set of all possible errors. In real life, for the majority of real measuring devices, we have no time and no money to perform such a long experiment, so usually, we make a few measurement (typically about 40). From these measurements, it is impossible to determine the precise probabilistic distribution. What we can determine is some characteristic of that distribution. In probability theory, zillions of different characteristics are known. However, in real measurements, only three of them are widely used: mean value, standard deviation, and the interval of possible values. If we know the errors $e_1, ..., e_n$, then these characteristics are estimated, crudely speaking, as $\bar{e} = n^{-1} \sum_i e_i$, $\sigma = \sqrt{n^{-1} \sum_i (e_i - \bar{e})^2}$, and $[\min e_i, \max e_i]$.

The fact that interval is among these basic characteristics is the main reasons why interval estimates are so widely spread, and why interval methods are so useful.

The question is: why only these three? Has there been a rather arbitrary choice, negotiated later into ISO and IEEE standards, or intervals are really better characteristics in some sense?

To answer this question, we formulate what a characteristic can be, and what "better" means. A characteristic can be viewed as a means to store the results of ≈ 40 different measurements in one number. To simplify computations, it is therefore reasonable to act as follows: after m measurements, we have some preliminary value $e^{(m)}$ of this characteristics, that compresses the values $e_1, ..., e_m$. After we make a next measurement and get the value e_{m+1}, we should use the compressed value $e^{(m)}$ only, and not drag along all m measurements. So, on each stage, we deal with two numbers only: $e^{(m)}$ and e_{m+1}. So, to define a characteristics, it is necessary to define a binary operation $*$ that transforms $e^{(m)}$ and e_{m+1} into the next estimate $e^{(m+1)}$. It is also reasonable to assume that the resulting value of a characteristic should not depend on the order of these measurements. From this we can conclude that $a * b = b * a$ and $a * (b * c) = (a * b) * c$, i.e., that $*$ defines a commutative semigroup.

We assume that on the set of such semigroup operations an ordering is defined (whose meaning is that the characteristic that corresponds to one operation is "better" in some reasonable sense than the characteristic that is obtained from using the second operation). We demand that this ordering chooses the unique "best" characteristics, and that the ordering should not be change if we change the unit in which all the measurements are performed (i.e., use inches instead of centimeters). Our main theorem is that for such criteria, the best characteristic coincides either with $\max e_i$, or with $\min e_i$, or with one of the momenta $\sum_i e_i^p$ for some p (so, as particular cases, we get mean values and second momenta). If we add invariance with respect to some non-linear transformations, then the only optimal characteristics are the interval ones min and max.

A similar result explains why in multi-dimensional case, ellipsoids turn out to be the best approximation for uncertainty domains.

INTERVAL ESTIMATES HELP TO MAKE INTELLIGENT CONTROL MORE EFFICIENT AND RELIABLE

Bob Lea
NASA Johnson Space Center, Houston TX 77058, USA
email blea@gothamcity.jsc.nasa.gov

In case we do not have the precise knowledge of a controlled system, we are unable to apply traditional control theory. Such situations occur, e.g., when we are devising a control for a future Martian rover, or a control for any other space mission into the unknown. In such cases, we often have a skillful operator who is known to make good control decisions in uncertain environments. This operator can communicate his skills only in terms of natural- language rules. There exists a methodology called *intelligent control* that enables to translate these rules into an actual control (see, e.g., [1-4]).

This methodology is extremely successful in various applications, ranging from controlling appliances to controlling trains, cars and space missions. However, this methodology encounters several problems related to the fact that it generates a numeric value of the recommended control, and does not specify with what precision we must follow this recommendation (i.e., does not give an interval of possible controls). Therefore:

- it is difficult to choose the necessary precision of the controlling devices;
- in case the recommendation is close to a 0 (no-control) value, it is difficult to tell whether we need to apply any control at all;
- in case we have several alternative rule-based controls, it is difficult to decide whether their recommendations are in good agreement (thus increasing the reliability of these recommendations), or they contradict to each other, so in this situation, we cannot rely on our automatic systems.

For all these purposes, we need to know this interval precisely: e.g., in the last application, if we underestimate this interval, we will sometimes erroneously reject close (and thus reliable) controls, and thus not utilize all the possibilities of an automated control system. If we overestimate this interval, we may erroneously accept the unreliable recommendations.

We give the precise estimates for the interval of possible controls, and show how to use these estimates in order to increase the efficiency and reliability of intelligent control.

REFERENCES

[1] V. Kreinovich, R. Lea, et al. *What non-linearity to choose? Mathematical foundations of fuzzy control.* Proceedings of the 1992 International Fuzzy Systems and Intelligent Control Conference, Louisville, KY, 1992, pp. 349–412.

[2] R. N. Lea. *Automated space vehicle control for rendezvous proximity operations.* Telemechanics and Informatics, 1988, Vol. 5, pp. 179–185.

[3] R. N. Lea, Y. K. Jani and H. Berenji. *Fuzzy logic controller with reinforcement learning for proximity operations and docking.* Proceedings of the 5th IEEE International Symposium on Intelligent Control, 1990, Vol. 2, pp. 903–906.

[4] R. N. Lea, M. Togai, J. Teichrow and Y. Jani. *Fuzzy logic approach to combined transnational and rotational control of a spacecraft in proximity of the Space Station.* Proceedings of the 3rd International Fuzzy Systems Association Congress, 1989, pp. 23–29.

IF WE MEASURE A NUMBER WE GET AN INTERVAL, WHAT IF WE MEASURE A FUNCTION OR AN OPERATOR?

Joe Lorkowski, Vladik Kreinovich
Computer Science Department
University of Texas at El Paso, El Paso, TX 79968

Why are intervals appropriate for describing measurements? Suppose that we measure a physical quantity x (e.g., length l). The actual value of this quantity is an arbitrary real number. But there is no way to design an ideal measuring device, that would always give a precise value of x. The result $\tilde{x}$ that is produced by a real measuring device is always approximate. This is well known, and the producers of measuring devices supply them with the precision estimates. In other words, they give a value δ, and they guarantee that the difference $x - \tilde{x}$ between the actual and the measured values does not exceed δ. So, if we apply a measuring device, and get $\tilde{x}$ as a result, then the possible values of the physical quantity x form an interval $[\tilde{x} - \delta, \tilde{x} + \delta]$.

Suppose now that we know that a physical quantity y is a function of the physical quantity x, in other words, we know that $y = f(x)$ for some function $f(x)$, but we do not know what this function is. How to determine this function? Again, we can measure only finitely many values, with finite precision, so, after finitely many measurements, we get a set of possible functions $f(x)$. This set can be called a *function interval.*

The first person to consider function intervals was R. Moore himself in his pioneer papers. Since then several different definitions of a function interval have been proposed.

The situation can become even more complicated. For example, if we analyze how physical fields evolve, then in addition to functions, we must describe functionals and operators, i.e., mappings that transform a function (current values $f(\vec{x})$ of the physical field) into a predicted future value of this field. Again, since we can perform only finitely many measurements, in any moment of time, our measurement results are consistent with the whole bunch of different functionals. So, at any moment of time, we have a set of functionals: a *functional interval.*

This problem is especially important for quantum mechanics, where to describe even a single particle, we need an entire field $\psi(\vec{x})$ (called a *wave function*).

Even more complicated mathematical structures naturally appear in quantum field theory and especially in quantum theory of space- time.

One can apply different ideas to describe function intervals, functional intervals, etc. But it is desirable to develop a general formalism that would allow us, given a natural definition of an interval for the sets X and Y, to design an appropriate definition of an interval for the set Y^X of all the functions from X to Y.

We propose such a definition, and show on several examples that it is physically natural. As a basis of our definition, we take the theory of semantic domains that was developed by Dana Scott as part of his description of efficiency in mathematics.

ZONE LOGIC FOR MANUFACTURING AUTOMATION: INTERVALS INSTEAD OF OPTIMIZATION, GOALS INSTEAD OF ALGORITHMS

Roger Lovrenich
Teramar Technologies, 211 Teramat Way, El Paso, TX 79922

The performance of the existing automated plants is far below the original bright expectations. Can we improve it by applying mathematical optimization techniques? They work well for a single robot whose state can be described by a few parameters. But there is no known way to optimize the performance of an automated plant, where dozens or hundreds of automatic robots and machines are performing complicated tasks. There are even results that show that such optimization problems like task scheduling are intractable (NP-complete). Does this mean that we can abandon all the hopes to drastically improve the efficiency of the existing automated plants? Hopefully, not.

The main reason why the automated plants are not that efficient as we want them to be is not because the trajectories of the robotic arms are not precisely mathematically optimal. The main reason is that about half of the time the entire system is down, and needs restarting. And when the system is down, then usually the metal parts moving in the wrong directions crush into each other and create a mess (that takes a long time to clean up).

We have developed a new approach to manufacturing automation that we called a *Zone Logic.* In this approach, there is no optimization involved. Instead, two things are done. First is to avoid crushes. For every parameter of every machine, we use the experts' experience to divide the set of its possible values into reasonable intervals (that correspond to different stages of the manufacturing process). For each pair of potentially interacting robots or machines, we give the list of possible combinations of these intervals that are still OK (they are called "zones"). The values of these parameters are constantly monitored (or, to be more precise, the local processor attached to each robot is constantly checking what interval does the current value belong to), and if the current combination of these parameters is outside the permitted zones, both robots stop, and a signal is sent to the operator.

As for manufacturing itself, instead of describing an algorithm (who does what), we describe the goal of the system, i.e., we describe which stages are permitted after which (e.g., after the robot that applies a crude cutting the object can go to fine polishing, but not vice versa). The system chooses the first available next permitted zone, and if none is permitted, stops the system and sends a message to the operator.

This description of goals and not algorithms is similar to the one used in languages like PROLOG, and our algorithm is somewhat similar to the methods that are used in PROLOG compilers.

These are just two basic ideas. There are several others that make the whole system workable, efficient, and fool-proof.

As a result, we get a working systems. It was implemented in several automotive plants, and lead to a drastic increase in efficiency.

MEASURING SURFACE WITH A COORDINATE MEASURING MACHINE: INTERVALS OF PRECISION

Thomas J. McLean, David H. Xu
Department of Mechanical & Industrial Engineering,
The University of Texas at El Paso, El Paso, TX 79968

Coordinate Measuring Machines (CMMs) coupled with computers have provided new, powerful capabilities in the field of manufacturing quality control.

When a finite set of measurements is made on a surface of interest, no information is gathered about the rest of the surface. The coordinates of these points are only a sample of the surface from which the size and form of the workpiece are to be estimated. At present it is standard practice to regard the sample as the workpiece itself [1]. So the calculated parameters of the features are not derived from the whole surface of the workpiece but only from a set of points [1].

It is therefore desirable to get an estimate not for the measured points only, but for the shape of the entire surface. In mathematical terms, our goal is to provide a thin "sandwich": two surfaces between which the actual surface lies.

For such an objective, the following questions arise:

1) What is the optimum sample size for each standard geometric elements (line, plane, circle, cylinder, cone, etc), given a desired specific confidence level?
2) Where and how should measurements be conducted to measure these points on the surface? Should the sampling be complete random, randomized block or equal space sequential sampling? Which is better?
3) What is the confidence level of the sampled data representing the infinite surface?

There is no standard or guidance for CMM users as to how many points are required to assure adequate confidence that the sample parameters are consistent with the parameters of the entire surface being evaluated and where and how to make these measurements.

The present report gives the answers to the above questions: we describe what sampling size and pattern are optimal for various standard geometric elements, and what algorithms we should use for reconstructing the "sandwich".

It turns out that among the existing surface reconstruction algorithms, minimum zone evaluation [2] (the best known algorithm for computing tolerance zone for sampled points), often underestimates the intervals for the entire surface. On the other hand, the least squares method with proper sampling strategies on CMMs leads to the true tolerance zone of the interested surface.

References

[1] A. Weckenmann and M. Heinrichowski, "Problems with Software for Running Coordinate Measuring Machines", *Precision Engineering*, Vol. 7(2), pp. 87–91, 1985.

[2] T. S. R. Murthy and S. Z. Abdin, "Minimum Zone Evaluation of Surfaces", *Int. J. of Machines Tool Design and Research*, Vol. 20, pp. 123–136, 1980.

CONSTRUCTIVE PROOF OF KOLMOGOROV'S THEOREM, NEURAL NETWORKS AND INTERVALS

Ray Mines[1], Mutsumi Nakamura[1], Vladik Kreinovich[2]
[1]Department of Mathematics, New Mexico State University, Las Cruces, NM 88003
[2]Department of Computer Science, University of Texas at El Paso, El Paso, TX 79968

David Hilbert announced, in 1900, a list of 23 problems as a challenge for the 20th century. One of them (No. 13) was to prove that not all functions of several variables can be represented as compositions of functions of 1 and 2 variables. This problem remained a challenge until 1957, when (rather unexpectedly) Kolmogorov proved that an arbitrary continuous function $f(x_1, ..., x_n)$ on an n- dimensional cube (of arbitrary dimension n) can be represented as a composition of addition and of functions, $g_i(z)$, of one variable.

This result was a purely theoretical, there were no thoughts of possible applications. Moreover, Kolmogorov's proof was a pure existence proof. Because the proof used indirect methods, it gave no hint of an algorithm for the construction of the needed functions. Thus one cannot plot their graphs. At that time, computers were still in their infancy, and it was not practical to compute many functions. So the nonconstructive nature of the proof was not viewed as a serious drawback; especially since there was no use for such a decomposition.

The first possible application of Kolmogorov's theorem was discovered, in 1987, by R. Hecht-Nielsen, who noticed that Kolomogorov's theorem proves that an arbitrary function can be implemented by a 3-layer neural network (with neurons whose in-out characteristics are the functions $g_i(z)$). Because of Hecht-Nielsen's observation, the nonconstructive nature of Kolmogorov's proof is a serious drawback.

We know that such a network exists, but we have no idea how to design and/or to train such a network.

The design problem is somewhat simplified by the fact that manufacturing is always non-precise. Therefore, even if we know the functions $g_i(z)$ precisely, we can only guarantee that the actual in-out characteristics of the manufactured hardware neurons belongs to the interval $[g_i(z) - \delta, g_i(z) + \delta]$, where δ denotes the manufacturing precision. In view of this, we see that for a given design, and for a given manufacturing precision, we will only be able to approximate a given function $f(x_1, ..., x_n)$ to within some precision ε. In other words, for each set of input values $x_1, ..., x_n$, the set of outputs that correspond to different manufacturing implementations of the g_i-neurons, form an interval $[f(x_1, ..., x_n) - \varepsilon, f(x_1, ..., x_n) + \varepsilon]$.

It was proved, in 1988, that for every interval of this type, one can produce a design of a neural network (with the appropriate ideal functions $g_i(z)$), and precision δ such that the implemintation of this design with this precision, the result will fit into the given interval of output values.

In the idealized settinf of Kolmogorov's theorem, we have for each function $f(x_1, ..., x_n)$, there is a design that fits exactly ($\varepsilon = 0$). The problem is that there is no way to find this perfect design. In the 1988 constructive result, for each ε, we can produce a design algorithmically, but for different ε we get different designs.

While we gain constructability, we lose elegance. Is this necessary? Is is possible to find an algorithmic design that would work for all ε?

Our answer is "yes". We give a constructive proof of Kolmogorov's theorem, and thus prove that we can construct a design that serves for all ε.

References

A. N. Kolmogorov, *Doklady Akad. Nauk USSR*, 1957, Vol. 114, pp. 953–956.

R. Hecht-Nielsen, *Proceedings of the International Conference on Neural Networks*, 1987, Vol. III, pp. 11–13.

K. Hornik et al. *Neural Networks*, 1989, Vol. 2, pp. 359–366.

NE-FACTORS: DIFFERENT PRAGMATICS OF AN INTERVAL IN KNOWLEDGE REPRESENTATION

A.S.Narin'yani
Russian Research Institute of Artificial Intelligence,
P.O. Box 111, 103001, Moscow, Russia, email nar@isi.itfs.nsk.su

Intervals have several known and clear applications in knowledge representation: e.g., in geometry, in time models, etc. Less clear are the applications of intervals to modelling such basic features of the real world as imprecision, uncertainty, subdefiniteness, fuzziness and some others. In my previous papers these features, which are antonyms to precision, definiteness, etc. of formal objects in the traditional mathematics, were named NE- factors from the Russian negative prefix "ne-" being the equivalent to "im-","in-","non-","un-" in English.

It is necessary to stress that NE-factors are not formalization of semantics of the corresponding lexemes of natural language but rather these lexemes are labels for some particular characteristics of real world models. To be included in the knowledge techniques, these characteristics should be distinguishable and accurately defined.

Because of the fact that the notions fuzziness and uncertainty have been investigated rather intensively, we try to introduce informally the remaining two NE-factors:

imprecision is a natural feature of every real parameter: when an interval represents an imprecise value, it means that any smaller interval has no physical meaning in this situation, because of the very nature of the corresponding parameter.

sub-definiteness expresses an approximate estimation of a value up to our information about it (due to rough measurement, incomplete data, etc.); in contrast to the previous case the sub-definite interval can become smaller when additional information is provided. I need the above introduction to use the rest of this one-page abstract to formulate the following three statements.

The first statement is aimed at the AI part of the audience: there exists a general theory of subdefinite models that covers not only the interval representation for real numbers, but rather different representations for all data types within any possible formal system: sub-definite integers, sets, multi-valued logic variables, etc.

Another statement is aimed at those participants who are closer to computations: interval mathematics is not limited to interval algorithms, it can be extended to more promising computational interval models covering both sub-definite and imprecise data types.

The third statement is addressed to both parts of audience: a powerful technology has been developed by the joint team of Russian Research Institute of Artificial Intelligence and the AI Laboratory of Institute of Informatics Systems to construct wide spectrum of active data types, i.e. sub-definite and imprecise ones, and to use them for building qualitatively new data/knowledge processing systems, basing on highly parallel virtual data-driven machines.

HOW TO CONTROL IF EVEN EXPERTS ARE NOT SURE: MINIMIZING INTERVALS OF UNCERTAINTY

Hung T. Nguyen[1], Vladik Kreinovich[2], Bob Lea[3], Dana Tolbert[2]

[1] *Department of Systems Science, Tokyo Institute of Technology*
4259 Nagatsuta-cho, Midori-ku, Yokohama 227 Japan
[2] *Computer Science Department, University of Texas at El Paso, El Paso, TX 79968*
[3] *Mail Box PT4, NASA Johnson Space Center, Houston, TX 77058*

In case we do not have the precise knowledge of a controlled system, we are unable to apply traditional control theory. In such cases, we can find an expert who is good at control, extract as many rules as possible from him, and try to transform these rules into the precise control strategy. Zadeh and Mamdani initiated a methodology for such a translation that is called *fuzzy control.*

In order to apply this methodology, we must:

1) describe the expert's uncertainty about every natural-language term A (such as "small") that he uses while describing the control rules; this is done by ascribing to every possible value x of the related physical quantity a value $\mu_A(x)$ from the interval [0,1] that describes to what extent this expert believes that x satisfies the property A (e.g., $\mu_{small}(0.3)$ is his degree of belief that 0.3 is small). The resulting function μ_A is called a *membership function*;
2) experts' rules contain natural-language words combined by logical connectives (e.g., "*if* x is small, *and* $\dot{x}$ is medium, *then* u must be small"). Therefore, we must be able to estimate the experts' degree of belief in $A\&B$, $A \vee B$, $\neg A$ (where $\neg$ stands for "not") from the known values of degrees of belief of A and B. In other words, we must describe the fuzzy analogues of &, $\vee$, and $\neg$ to combine the original membership functions into a membership function $\mu_C(u)$ for control;
3) finally, we must transform this membership functions into an actual control value by a proper *defuzzification* procedure.

This methodology works fine if the experts are absolutely sure of what they are doing. But in real life they are not that confident in their own controlling abilities. As a result, the degrees of certainty that correspond to one of the same expert can differ drastically, and fuzzy control algorithms translate these different degrees of uncertainty into different control strategies.

In such situation, it is reasonable to choose a fuzzy control methodology in such a way that the resulting control is the least vulnerable to this kind of uncertainty, i.e., that the *resulting intervals of uncertainty for control values are as small as possible.*

We show that this minimization demand leads to min and max for &− and $\vee$−operations, and to $1 - x$ for negation.

ON THE NONMONOTONIC BEHAVIOR OF EVENT CALCULI FOR DERIVING MAXIMAL TIME INTERVALS

Alessandro Provetti
CIRFID - Universita' di Bologna
Via Galliera 3/a Bologna. I-40100 ITALY
provetti@cirfid.unibo.it

The Calculus of Events (EC) has been proposed by [Sergot & Kowalski 86] as a formalism for dealing with time and actions. EC is based on an ontology of events (that are considered more primitive than time) and of properties, i.e. descriptions of the world that are true over certain intervals of time (intervals). These intervals are constrained by couples of events that affect them. Also intervals open on the left have to be taken into account. Starting from a description of events that have occurred in the domain (input data in a database context) Events Calculus derives intervals where certain properties hold or decide whether a property holds at a specific date. These intervals are maximal and convex and we argue that they are the most informative result that can be extracted from such a description.

Since it has been conceived in the framework of Logic Programming. EC can derive intervals efficiently by running its axiomatic definitions a PROLOG program (with few changes). The average computational cost of deriving an interval for a given property is roughly $O(3n)$, where n is the size of input, that is, basically, the number of events.

Our concern, however, is to make the systems able to derive the set of intervals that a human reasoner would derive from the same description. To this extent, we are interested in deriving intervals constrained by events that may or may not be time-stamped.

It seems a reasonable intuition that the degree of "orderedness" of the set of events E is a key factor in determining the size of the set of intervals Ans that EC can derive. Ans depends on the input set and on the kind of ordering over events: $Ans(E, <_i)$.

We define an ordering over orderings: " $<_a$" < " $<_b$" with respect to E if for any X, Y from E, $X <_a Y \rightarrow X <_b Y$. Our observation is that from " $<_i$" < " $<_j$" with respect to E, it does not always follow that $Ans(E, <_i) < And(E, <_j)$.

This is also the case for the maximal order $<_M$, i.e., when $< E, <_M>$ is totally ordered.

This abnormal behavior is in principle avoided in Event Calculus systems for database applications or in general applications, where a "no" or "don't know" answer is preferred to a (potentially incorrect) positive answer.

Nonetheless, we look at the interval in $Ans(E, <_i) - And(E, <_j)$ as accounting for the nonmonotonic behavior of the human reasoner in the following situation: a person who is told a narrative where events are partially ordered sometimes has to retract his/her own belief about the actual sequence of events. The relation between these observations and the formal theory of belief revision as developed, for instance, by [Gandenfors 89], has not yet been carried out.

CONTROL PROBLEMS WITH INTERVAL RESTRICTIONS

Sergei Savitsky
Montebello apts. # F4, 5900 Enterprise, El Paso, TX 79912

In many situations, the objective of the control is to keep the parameters of the controlled system (*plant*) close to the desired values.

For example, once a trajectory of the navy ship is determined, it is necessary to deviate from this trajectory as little as possible. On the other hand, from the viewpoint of the engine safety, it is necessary to keep the changes in the number of revolutions per second as small as possible. The values of the parameters that describe the control (like the amount of fuel injected) should also be close to some initially ascribed values.

In such cases, the actual values $\vec{x}(t)$ of the parameters that describe a plant are close to the desired (ideal) values $\vec{x}_p(t)$, and the values of the control $\vec{u}(t)$ should be close to the ideal control values $\vec{u}_p(t)$. In general, a plant is described by non-linear equations $\vec{x}(t+1) = \vec{F}(\vec{x}(t), \vec{u}(t))$. But since $\vec{x}(t) \approx \vec{x}_p(t)$ and $\vec{u}(t) \approx \vec{u}_p(t)$, it is possible to restrict ourselves only to the terms that are linear in $\vec{X}(t) = \vec{x}(t) - \vec{x}_p(t)$ and $\vec{U}(t) = \vec{u}(t) - \vec{u}_p(t)$, and thus arrive at the linear system: $\vec{X}(t+1) = \vec{X}_0(t) + \Phi(t)\vec{X}(t) + V(t)\vec{U}(t)$, where $\vec{X}_0(t)$ is a known vector, and $\Phi(t)$ and $V(t)$ are a priori known matrices.

In these terms, the control problem can be formulated as follows: for a given initial state $X(0)$, find the controls $\vec{U}(t)$ so that for all t and for all components of the vectors $\vec{X}(t)$ and $\vec{U}(t)$, $|X_i(t)| \le \varepsilon_i$ and $|U_j(t)| \le \delta_j$, where ε_i and δ_j are a priori given positive real numbers. In other words, $x_i(t) \in [x_{pi}(t) - \varepsilon_i, x_{pi}(t) + \varepsilon_i]$ and $u_j(t) \in [u_{pj}(t) - \delta_j, u_{pj}(t) + \delta_j]$.

¿From mathematical viewpoint, this problem is a particular case of the linear programming problem, and therefore, one can apply both standard (simplex) and new (ellipsoid) linear programming techniques to solve it. But even the newest algorithms require n^3 computational steps, where n is the total number of variables. Here, n is proportional to the total time T, so to compute what to do in the moment T we need to perform $T^3 \gg T$ computations. In other words, we do not have time to solve the problem precisely.

At first glance, it may seem like the problem is not solvable at all. Maybe it is: if we take ε_i and δ_j as a must. But actually, the real demands are somewhat fuzzy: that the deviations be small. So, the values that the experts give are the result of a (rather arbitrary) "defuzzification", and there will be no harm, if we violate these inequalities a little bit.

This remark prompts the following heuristic algorithm: we look for a control that minimizes the quadratic functional

$$J(U) = \sum_{t=0}^{T}(\sum_i q_i X_i^2(t) + \sum_j k_j U_j^2)$$

for some positive "weights" q_i and k_j. Minimizing a quadratic functional leads to an easily solvable linear problem. If the resulting control violates the above inequalities for X_i, we decrease q_i; if it violates he inequalities for U_j, we decrease k_j. If after several iterations we still cannot fit into all the intervals, we ask the experts to loosen the constraints, and start the procedure anew.

This procedure was used to control Russian navy and civilian ships, and lead to an essential improvement of the control. A similar procedure was applied to a more complicated problem of controlling the chemical plant, and also lead to a drastic increase in productivity.

NEURAL NETWORKS THAT ARE NOT SENSITIVE TO THE PARAMETERS OF NEURONS

Ongard Sirisaengtaksin
Department of Applied Mathematical Sciences
University of Houston-Downtown, Houston, TX 77002

For non-linear systems, it is often very difficult to design a control that satisfies certain given goals. It is even more difficult to design a control that is optimal in some reasonable sense (e.g., uses the minimal amount of fuel, attains its goals in the shortest time, etc). One of the methods that (in many cases) helps to design such a control is to train a neural network in such a way that for a given input $\vec{x}$, its output is close to the ideal control value u.

Sometimes this method helps, sometimes it does not. What is the reason? If after, say, 3000 iterations the network is still not appropriately trained, does it mean that it cannot be trained in principle, or that we were not sufficiently patient (and after more iterations, we would have got the desired control)?

An answer to this question was given by several authors who proved that 3-layer neural networks can approximate any continuous function $f(\vec{x})$ with any given precision (Hecht-Nielsen, Cybenko, Funahashi, Hornik, Stinchcombe, White). These results are extremely valuable for control: they show that for a plant whose state can be described by finitely many parameters, for an arbitrary control $u(\vec{x})$, and for an arbitrary precision $\varepsilon > 0$, we can implement this control with a given precision using 3-layer neural networks. In other words, these results mean that in principle, an arbitrary control can be obtained by using neural networks.

We said "in principle", because these results are based on the assumption that we can design neurons with precisely known in-out characteristics $s(x)$. In reality, however, it is technically impossible to design a hardware device whose in-out characteristics $\tilde{s}(x)$ precisely coincides with a prescribed function $s(x)$. What we can guarantee is as follows: we can fix a precision $\delta > 0$, and the biggest possible signal X, and guarantee that the in-out characteristics $\tilde{s}(x)$ of a manufactured (hardware) neuron belongs to an interval $[s(x) - \delta, s(x) + \delta]$ for all x from $-X$ to X.

Our main result is that with these non-precise ("interval") neurons, it is possible for an arbitrary continuous function f, and a real number $\varepsilon > 0$, to produce a design (scheme) of a neural network and the necessary precision δ in such a way that even if we build all the neurons with this precision δ, the in-out characteristics of the resulting network will be $\varepsilon-$close to f.

Similar results allow us to build designs that are not sensitive to the parameters of the neurons, to compute functionals (i.e., plants with distributed parameters, in which the state of the plant is described by a function, and the control takes that function as an input). It is also proved possible to train a network in such a way that it would take the description of a plant, and the desired objective as inputs, and generate the appropriate control.

References

R. Hecht-Nielsen. "Kolmogorov's mapping neural network existence theorem", *IEEE International Conference on Neural Networks*, San Diego, SOS Printing, 1987, pp. 11–14.

K. Hornik, M. Stinchcombe, and H. White. "Multilayer feedforward networks are universal approximators", *Neural Networks*, 1989, Vol. 2, pp. 359–366.

QUASIEUCLIDEAN ARITHMETIC INTERVAL SPACE AND ITS PROPERTIES

Yu. G. Stoyan
Institute for Problems in Machinery, Academy of Sciences of Ukraine
2/10 Pozharsky St., Kharkov, 310046, Ukraine

In this report, we define a set $I_s(R)$ of centered interval points of the kind $[x, \nu_x] = (a, b)$, where $x = (a + b)/2$, $\nu_x = (b - a)/2$. On this set $I_s(R)$, we introduce the concepts of a positive, negative, and zero interval, as well as the linear order. We also define on this set the operations of addition, subtraction and multiplication which are equivalent to the operations of addition, subtraction and multiplication from interval mathematics (i.e., defined on $I(R)$). Besides, we consider the operations of absolute value, and p-th root. The metric on $I_s(R)$ is defined as $id([x, \nu_x], [y, \nu_y] = |[x, \nu_x] - [y, \nu_y]|$. Based on this metric, we define the concepts of an open (closed) ε-neighborhood, interval-open and interval-closed sets, interval Cauchy series, interval compactness, etc. We proved that in this topology, the set $I_s(R)$ is complete, separable, and connected.

Then we form an n-dimensional interval space $I_s(R^n)$ consisting of n-dimensional points whose n coordinates are interval points from the space $I_s(R)$. On this set $I_s(R^n)$, we introduce operations of addition, subtraction, and multiplication by a number. With these operations, $I_s(R^n)$ becomes a quasilinear space. We also define a quasiscalar product on this space, that, in its turn, allows us to introduce an interval norm and an interval metric that take values in $I_s(R)$. We show that the quasilinear space $I_s(R^n)$ with this quasiscalar product is complete, separable, and connected. We analyze geometric properties of this space $I_s(R^n)$ for $n = 2$ and $n = 3$.

DEMPSTER-SHAFER THEORY, LOGIC PROGRAMS AND GENERALIZATION OF STABLE SEMANTICS TO INTERVAL UNCERTAINTY

V. S. Subrahmanian[1], Raymond T. Ng[2]
[1]Department of Computer Science, University of Maryland, College Park, MD 20742
[2]Department of Computer Science, University of British Columbia
Vancouver, B.C., V6T 1Z2 Canada

Standard logic programs consists of the statements of the type "if P_1, ..., P_n are true, then P is true". In real life, the experts' knowledge is often not that precise. For example, the experts know that birds normally fly. It does not mean "if X is a bird ($bird(X)$), then X flies ($fly(X)$)", because we know that some birds do not fly. One way to express this statement is to assign some probability to it, say, 90% of the birds fly.

In a few well-analyzed situations we can assign such a probability, but in the majority of the situations the expert can only give crude estimates for that probability. For example, an expert can say that there is an over 90% chance that a bird can fly. This means that if $bird(X)$ is true, then the probability that $fly(X)$ is true belongs to an interval [0.9,1].

So, in general, we must consider intervals as our estimates of the probability that some statement Q is true. This interval description includes (as the particular cases) the situations when we know whether Q is true or not, when we know nothing about Q, and when we know the precise value of the probability of Q. Namely, if we are sure that Q is true, then the interval is [1,1]. If we are sure that Q is false, then the interval of possible values of probability is [0,1]. If we know the probability p that Q is true, then this interval is $[p, p]$. If we know nothing about Q, then the interval is [0,1]. In general, we can get any interval $I \subset [0, 1]$.

So, when we state the facts of the logic program, we state them in the form $P : [a, b] \leftarrow$, where P is an atomic statement, and $[a, b]$ is the corresponding interval of possible probabilities of P. Similarly, it is reasonable to consider implications between such facts as rules, i.e., consider as rules statements of the type $P : [a, b] \leftarrow P_1 : [a_1, b_1], ..., P_n : [a_n, b_n]$. By a logic program we will understand an arbitrary combination of such facts and rules.

The entire purpose of the knowledge is to be able to answer queries, So, suppose that we ask a query Q. Since we are not sure whether the facts and rules from the knowledge base are true or not (we know only their probabilities), we, in the generic case, cannot hope to get the precise "yes" or "no" answer to that query. The only information that we can get is the information about the probability of Q. Since we do not know precisely the probabilities of the statements and facts from the knowledge base, it is not reasonable to expect the precise value of this probability $p(Q)$. So, we can expect only the interval of possible values of probability that Q is true.

So, two questions naturally arise: how to define this interval? In other words, what is the semantics of such logic programs? And, second, how to compute this interval?

As for semantics, we want to define it in such a way that for the standard logic programs (where the only allowed intervals are [1,1]) this semantics will turn into stable model semantics (that we believe to be the most reasonable semantics for standard logic programs). To define a generalization of stable model semantics to generalized logic programs, we apply Dempster-Shafer formalism that has formulas for combining uncertainty (and therefore, if we know that $A \leftarrow B$ and $A \leftarrow C$, and we know the intervals for B and C, we can compute the resulting interval for A). We also show how to compute the resulting intervals.

INTERVAL VALUED FUZZY SETS AND LOGICS

I. Burhan Türkşen
Principal Investigator
Manufacturing Research Corporation of Ontario
and
Department of Industrial Engineering
University of Ontario, Totonto, Ontario, M5S 1A4, Canada

It has beem experimentally determined that experts use of linguistic connections AND, OR, ..., IF ... THEN, etc, does not directly correspond to the well-known $t-$norms and $t-$conorms (Zimmermann and Zysno, 1980). An interval-valued fuzzy set based on normal forms was proposed to represent expert meaning of linguistic connectives AND, OR, ..., IF ... THEN (Turksen 1986).

Recently, it has been shown that "Compensatory AND" proposed by Zimmermann falls within the boundaries of interval-values fuzzy sets proposed by Turksen. The fuzzy logics of interval-valued fuzzy sets generate bounds on approximate reasoning. These bounds contain some of the point-valued approximate reasoning results. Issues related to point-valued versus interval valued fuzzy sets and logics will be reviewed and future challenges will be presented.

INTERVAL COMPUTATIONS ON REAL AND REALISTIC PARALLEL MACHINES: EXPERIMENTAL AND THEORETICAL ANALYSIS

Elsa Villa[1], Andrew Bernat[2]
[1]Department of Mathematics, El Paso Community College, P.O. Box 20500
El Paso, TX 79998, email elsav@laguna.epcc.edu
[2]Department of Computer Science, University of Texas at El Paso
El Paso, TX 79968, email abernat@cs.ep.utexas.edu

It is well known that when in a numerical algorithm, we switch from real numbers to intervals, the number of necessary arithmetic operations increases. So, on a single processor machine, the running time increases as well.

This problem is not very crucial for simple data processing algorithms, but for algorithms that come from Artificial Intelligence, where runing time is often already huge, further increase may be a disaster.

A natural way to decrease running time is to use several processors working in parallel. In [1], we showed that in principle, parallelism can indeed decrease the running time drastically.

One of the main obstacles to parallelism is that if we use too many processors, then communications "eat up" the advantages of parallelism. As a result, when we use, say, 100 processors, and the computations are completely parallelizable, we still rarely achieve a decrease in running time that is bigger than 10 times.

In [1], we have developed and experimentally tested a computer architecture that allows the users to achieve the biggest possible time decrease.

Unfortunately, the actual parallel computers are usually not tailored to the interval problems. So, in the present report, we show how (and when) parallelizing helps to get interval estimates on several different parallelization schemes.

In addition to theoretical analysis, we confirm the theoretical results experimentally: either on real parallel computers, or on their realistic simulations.

Reference

[1] V. Kreinovich, A. Bernat, E. Villa, Y. Mariscal. "Parallel computers estimate errors caused by imprecise data," *Interval Computations*, 1991, Vol. 1, No. 2, pp. 31–46.

LINEAR-TIME ALGORITHMS THAT LOCATE LOCAL MAXIMA AND MINIMA OF A FUNCTION FROM APPROXIMATE MEASUREMENT RESULTS

Karen Villaverde, Vladik Kreinovich
Computer Science Department, University of Texas at El Paso
El Paso, TX 79968, USA

Abstract. The problem of locating local maxima and minima of a function from approximate measurement results is vital for many physical applications: in spectral analysis, elements are identified by locating local maxima of the spectra; in radioastronomy, sources and their components are located by locating local maxima of the brightness; elementary particles are identified by locating local maxima of the experimental curves.

In mathematical terms, we know n numbers $x_1 < ... < x_n$, and n intervals $I_i = [y_i^-, y_i^+], i = 1, ..., n$, where $y_i^- = y_i - \varepsilon$, $y_i^+ = y_i + \varepsilon$, and we know that the values $f(x_i)$ of the unknown function $f(x)$ at the points x_i belong to I_i. The set $\mathcal{F}$ of all the functions $f(x)$ that satisfy this property can be considered as a *function interval* (this definition was, in essence, first proposed by R. Moore himself). We say that an interval I *locates a local maximum* if any function $f \in \mathcal{F}$ attains a local maxima at some point from I. So, the problem is to generate intervals $I_1, ..., I_k$ that locate local maxima.

Evidently, if I locates a local maximum, then any bigger interval $J \supset I$ also locates them. We want to find the *smallest* possible locations I. We propose an algorithm that finds the smallest possible locations in linear time (i.e., in time that is $\leq Cn$ for some n).

Remarks.

1. By looking for the smallest possible location, we want an *optimal* interval estimate in the sense of [R80] and [RR80] (see also [K86]).
2. There exist various algorithms that locate the *global maxima* of an intervally defined function (see, e.g., [M79], [DS83], [RR88], [M91]). For these algorithms, local maxima are the main obstacle that has to be overcome, and not the final result, so we cannot apply these algorithms to locate *all local* maxima.
3. Local maxima and mimima are also used in the methods that accelerate the convergence of the measurement result to the real value of a physical variable, and thus allow the user to estimate this value without waiting for the oscillations to stop [N88].

REFERENCES

[DS83] Dennis, J. E., and R. B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*, Prentice- Hall, Englewood Cliffs, N.J., 1983.

[K86] Kolacz, H. In: *Interval Mathematics 1985*, Springer Lecture Notes in Computer Science, Vol. 212, 1986, pp. 67–79.

[M79] Moore, R. E. *Methods and applications of interval analysis*, SIAM, Philadelphia, 1979.

[M91] Moore, R.E. *Computers and Mathematical Applications*, Vol. 21, 1991, No. 6/7, pp. 25–39.

[N88] Nickel, K. In: *Reliability in Computing*, Academic Press, N.Y., 1988, pp. 151–169.

[R80] Rall, L. B. In: *Interval Mathematics 1980,* Academic Press, N.Y., 1980, pp. 489–498.

[RR80] Ratschek, H., and J. Rokne. In: *Interval Mathematics 1980,* Academic Press, N.Y., 1980, pp. 499–508.

[RR88] Ratschek, H., and J. Rokne. *New computer methods for global optimization*, Ellis Horwood, Chicester, 1988.

INTELLIGENT CONTROL WITH INTERVAL RESTRICTIONS

John Yen[1], Nathan Pfluger[1], Reza Langari[3]
[1]Department of Computer Science, [2]Department of Mechanical Engineering
Texas A&M University, College Station, TX 77840

Intelligent control is a methodology that transforms the experts' rules (formulated in natural language) into the actual control. As a result, for each set of parameters $x_1, ..., x_n$ that describe a state of the controlled system (*plant*), we compute the value of the control u that is reasonable to apply in this state. This computation is usually done in two steps: on the first step, for every real number u, we estimate the degree $\mu(u)$, with which it is reasonable to apply a control u. On the second step, we convert the resulting function $\mu(u)$ (called *membership function*) into a single value $\bar{u}$. Usually, a center of gravity (or some similar characteristics) is taken for $\bar{u}$.

In many cases, this scheme works pretty well. However, there are reasonable cases when it does not work. Suppose, for example, that a robot is approaching an obstacle. It can either turn to the left or to the right. Some experts would recommend turning to the left, some of them would prefer turning to the right. If we place both rules into the initial set of rules that is used to develop a fuzzy control, and apply a previously described procedure, then we end up with a membership function $\mu(x)$ for the controlled angle. We assume that this membership function reflects the experts' knowledge correctly. In particular, since it is senseless not to turn, the value $\mu(0)$ is either equal to 0, or close to 0. For big positive or big negative angles (corresponding to the angles that are reasonable to apply) the values $\mu(x)$ are positive and sufficiently big.

Since turns to the right ($x > 0$) and to the left ($x < 0$) seem to be equally reasonable, it is reasonable to expect that we will have a symmetric membership function, i.e., a function $\mu(x)$ such that $\mu(x) = \mu(-x)$. However, in this case, the above-described center-of-gravity procedure leads to the value $\bar{x} = 0$. So, if we follow it, the robot will run directly into an obstacle.

Therefore, in addition to the membership function, it is necessary to take into consideration that not all the values of u are possible: there exist intervals of forbidden values of u. In the present talk, we propose such a method and show that it works fine.

The general idea was proposed in [1–3]. We want to avoid the cases, when the value $\mu(x)$ for the resulting control x is too small. So we must establish some threshold value p, and consider only the values x, for which $\mu(x) \geq p$. The above-mentioned paradox occurs, when the values x for which $\mu(x) \geq p$ form several disjoint regions. So in this case we choose a region that is most "probable" (in some reasonable sense), then restrict the function $\mu(x)$ to this region, and apply a usual (e.g., center- of-gravity) "defuzzification" to the result of this restriction.

We to choose a region with the biggest area $\int \mu(x)\,dx$, therefore we call this method a *Centroid of Largest Area* (CLA) Defuzzification.

The main reason for choosing area and not any other characteristic is that using area really helps the robot to avoid the obstacles. Other (more theoretical) arguments in favor of this approach will be given.

References

[1] J. Yen and N. Pfluger. *Path planning and execution using fuzzy logic*, In: *AIAA Guidance, Navigation and Control Conference*, New Orleans, LA, 1991, Vol. 3, pp. 1691–1698.

[2] J. Yen and N. Pfluger. *Designing an adaptive path execution system.* In: IEEE International Conference on Systems, Man and Cybernetics, Charlottesville, VA, 1991.

[3] J. Yen, N. Pfluger, and R. Langari. *A defuzzification strategy for a fuzzy logic controller employing prohibitive information in command formulation*, Proceedings of IEEE International Conference on Fuzzy Systems, San Diego, CA, March 1992.