Interval Computations
No 3, 1993

On Bounding the Range of Some Elementary Functions in FORTRAN–77

Chenyi Hu, R. Baker Kearfott, and Abdulhamid Awad*

We present the techniques we have used to bound the range of the arcsine, arccosine, arctangent, arccotangent, and hyperbolic sine functions in our portable FORTRAN–77 library INTLIB. The design of this library is based on a balance of simplicity and efficiency, subject to rigor and portability.

О вычислении границ множества значений некоторых элементарных функций в Фортране 77

Ч. Ху, Р. Б. Кирфотт, А. Авад

Представлена техника, используемая нами для ограничения множества значений функций арксинуса, арккосинуса, арктангенса и гиперболического синуса в нашей переносимой библиотеке INTLIB на Фортране 77. При проектировании библиотеки преследовалась цель достижения баланса между простотой и эффективностью, точностью и переносимостью.

*This research was partially supported by National Science Foundation Grants No. CCR-9203730 and No. DMS-9205680.

1 Introduction

INTLIB ([7] and [8]) is a portable FORTRAN–77 library to support interval arithmetic evaluation of the elementary functions. Here, we describe the subroutines in INTLIB for bounding the arcsine, arccosine, arctangent, arccotangent, and hyperbolic sine functions. These subroutines are called IASIN, IACOS, IATAN, IACOT, and ISINH respectively.

Our basic algorithms are based on Taylor's theorem, with special argument reduction and choice of expansions. Our choice of the Taylor approximation allows us to achieve

1) ease of obtaining mathematically rigorous bounds on the truncation error;
2) portability;
3) fast convergence rates, and
4) ease with which the FORTRAN source code can be read and maintained.

There are numerous alternatives, such as polynomial interpolation, rational interpolation or minimax approximation, etc. Polynomial interpolants have error bounds similar to those of Taylor polynomials, and share other similarities, but are slightly more complicated. Rational functions have been used in interval computations to approximate elementary functions; for example, see [2, §2.4], following [12]. However, to our knowledge, in general *all* coefficients change if we increase the degree of approximation. Furthermore, we know of no simple error formulas amenable to interval computation. For example, in [11, problem 19, p. 312], the error term for rational interpolation of a function f by $P(x)/Q(x)$ at x_i, $1 \le i \le n$ is given by

$$e(x) = \frac{p_n(x)}{n!Q^2(x)} \frac{d^n}{dx^n} \left[Q^2(x) f(x)\right] \bigg|_{x=\xi}$$

where $p_n(x) = \prod_{i=1}^{n}(x - x_i)$. Contrast this with the simple error term for the arcsine in equation (4) below.

Related rationales are given in [3] and [9] for employing Taylor series in portable interval libraries for the standard functions. There are, however, some techniques beyond the Taylor approximations that will be useful in our context; see §5 below.

Specifically, the following well-known series expansions underlie the approximations in this paper.

$$\arcsin x = x + \frac{x^3}{2 \cdot 3} + \frac{1 \cdot 3 x^5}{2 \cdot 4 \cdot 5} + \cdots + \frac{(2n)! x^{2n+1}}{(2^n n!)^2 (2n+1)} + \cdots \quad (1)$$

$$\arctan x = x - \frac{x^3}{3} + \frac{x^5}{5} - \cdots + (-1)^{n-1} \frac{x^{2n-1}}{2n-1} + \cdots \quad (2)$$

$$\sinh x = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots + \frac{x^{2n+1}}{(2n+1)!} + \cdots \quad (3)$$

We bound these elementary functions with interval arithmetic.[1] In interval computations, both the number of computations and the widths of the input intervals affect the width of the result interval. Hence, we want to

(a) use narrow intervals $\mathbf{x}$ in the above series and

(b) accelerate the rate of convergence to reduce the number of computations.

Let us address (a) first. The range of a continuous monotone function $f(\mathbf{x})$ is either $[f(\underline{x}), f(\overline{x})]$ or $[f(\overline{x}), f(\underline{x})]$. Therefore, we may bound $f(\mathbf{x})$ by replacing both $\underline{x}$ and $\overline{x}$ by the smallest machine representable intervals[2] $\mathbf{l}$ and $\mathbf{u}$ containing $\underline{x}$ and $\overline{x}$, respectively, then bounding $f(\mathbf{l})$ and $f(\mathbf{u})$. If $f(x)$ is a monotonically increasing function, then $f(\mathbf{x}) \subseteq [\underline{f(\mathbf{l})}, \overline{f(\mathbf{u})}]$, where $f(\mathbf{x})$ denotes the exact range of f over $\mathbf{x}$. If $f(x)$ is a monotonically decreasing function, then $f(\mathbf{x}) \subseteq [\underline{f(\mathbf{u})}, \overline{f(\mathbf{l})}]$. Since the widths of $\mathbf{l}$ and $\mathbf{u}$ are small, the overestimation in the bounds for $f(\mathbf{x})$ due to the width of the operand will be limited. This idea is used for all subroutines in this paper, since all functions considered in this paper are monotone.

To guarantee that the computed range contains $f(x)$ for all $x \in \mathbf{x}$, we need to over- or underestimate the series as appropriate. For example, we need to overestimate the series (1) if we replace x by $\mathbf{u}$ and to underestimate (1) if we replace x by $\mathbf{l}$.

[1] Throughout this paper, we use boldface letters to denote intervals; we use $\underline{x}$ and $\overline{x}$ to denote the lower bound and upper bound, respectively, for an interval $\mathbf{x}$.

[2] This can be done by using the subroutine RNDOUT in [6].

We address (b) and error estimations for each of these subroutines in the following sections.

2 IASIN and IACOS

IASIN and IACOS bound the arcsine function and arccosine function respectively. The core subroutine of IASIN evaluates $\mathbf{s}$, the partial sum of series (1), by interval arithmetic, where $\mathbf{s}_i = \sum_{n=0}^{i} \frac{(2n)!\mathbf{x}^{2n+1}}{(2^n n!)^2(2n+1)}$, and $0 \le \mathbf{x} \le 0.5$. Since $\arcsin x$ is an odd function, we assume $\mathbf{x} \ge 0$. If $\mathbf{x} < 0$, we compute $\arcsin \mathbf{x}$ by $\arcsin \mathbf{x} = -\arcsin(-\mathbf{x})$. We assume $\mathbf{x} \le 0.5$ for fast convergence. If $\mathbf{x} > 0.5$, we define $\mathbf{y} = \sqrt{\frac{1-\mathbf{x}}{2}} < 0.5$ and use $\mathbf{y}$ in equation (1). Then, we apply the identity $\arcsin x = \pi/2 - 2\arcsin\sqrt{\frac{1-x}{2}}$ to find $\arcsin \mathbf{x}$.

Suppose i terms are used in evaluating (1), so that the interval truncated from (1) is $\mathbf{e_i} = \sum_{n=i+1}^{\infty} \frac{(2n)!\mathbf{x}^{2n+1}}{(2^n n!)^2(2n+1)}$. Since $\frac{(2n)!}{(2^n n!)^2(2n+1)}$ is a decreasing sequence, we may apply the geometric series with common ratio $r = 0.5 \ge \mathbf{x}$ to overestimate the maximum absolute truncation error $e_i = \max_{e\in\mathbf{e_i}} |e|$. We thus obtain

$$\begin{aligned} e_i &= \sum_{n=i+1}^{\infty} \frac{(2n)!\overline{x}^{2n+1}}{(2^n n!)^2(2n+1)} \\ &\le \frac{(2i+2)!\overline{x}^{2i+3}}{(2^{(i+1)}(i+1)!)^2(2i+3)} \sum_{n=1}^{\infty} \overline{x}^n \\ &\le \frac{2(2i+2)!\overline{x}^{2i+3}}{(2^{(i+1)}(i+1)!)^2(2i+3)}. \end{aligned} \tag{4}$$

If $\underline{s} = 0$, we use $\sin 0 = 0$ to avoid computing equation (1). If $\underline{s} \ne 0$, the maximum relative error is less than $\frac{\max \mathbf{e_i}}{\min \mathbf{s}_i} \le \frac{(2i)!2\overline{x}^{2i+1}}{(2^i i!)^2(2i+1)\underline{s}}$. We terminate accumulation of the sum $\mathbf{s}$ when this relative error bound is less than a tolerance ϵ (denoted TOL0 in the code), related to the relative error in a single floating-point operation. If we need to overestimate $\mathbf{s}$, we add the maximum absolute truncation error to the partial sum. If we need to underestimate the series, we simply truncate all terms from the $i+1$ term.

Here is the algorithm to bound arcsin $\mathbf{x}$. In the algorithm, we use a logical variable *Negative* to indicate if $\mathbf{x} < 0$ or not. We use another logical variable *Over* to indicate if we need to over estimate $\mathbf{s}$ or not. The variable Code records argument reductions on $\mathbf{x}$, so that the result may be transformed correctly. Some irrational numbers such as $\pi/2$, $\sqrt{2}$, … are used in our algorithms. They are defind as interval constants in INTLIB.

In Algorithm 1, Algorithm 2, and Algorithm 3, Step 3 is analogous to Step 2. However, the rounding direction differs, and depends on what argument reduction was previously done. We do not include details for Step 3 of these algorithms here. However, we have a complete writeup of Step 3 (and indeed, the FORTRAN source) for those interested in implementing these algorithms.

Algorithm 1.

1. Input the interval $\mathbf{x} = [\underline{x}, \overline{x}]$.

2. Compute the lower bound of arcsin $\mathbf{x}$.

    (a) If $\underline{x} < 0$, then *Negative* $\leftarrow$ *True* , $\underline{x} \leftarrow -\underline{x}$, *Over* $\leftarrow$ *True* ,
    else *Negative* $\leftarrow$ *False* , *Over* $\leftarrow$ *False* .

    (b) Represent $\underline{x}$ by an interval $\mathbf{l}$.

    (c) If $\mathbf{l} > 0.5$, then Code $\leftarrow 12$, $\mathbf{l} \leftarrow \sqrt{(1-\mathbf{l})/2}$,
    else Code $\leftarrow 11$.

    (d) Compute $\mathbf{s} = \mathbf{l} + \frac{\mathbf{l}^3}{1\cdot 3} + \cdots + \frac{(2i)!\mathbf{l}^{2i+1}}{(2^i i!)^2(2i+1)}$ until $\dfrac{(2i+2)!2\bar{l}^{2i+3}}{(2^{(i+1)}(i+1)!)^2(2i+3)\underline{s}} < \epsilon$.

    (e) If *Over* $=$ *True* then $\mathbf{s} \leftarrow \mathbf{s} + \frac{(2i+2)!2\mathbf{l}^{2i+3}}{(2^{i+1}(i+1)!)^2(2i+3)}$.

    (f) If Code $= 12$ then $\mathbf{s} \leftarrow \pi/2 - \mathbf{s}$.

    (g) If *Negative* $=$ *True* then $\underline{\arcsin \mathbf{x}} \leftarrow -\overline{s}$ else $\underline{\arcsin \mathbf{x}} \leftarrow \underline{s}$.

3. Compute the upper bound of arcsin $\mathbf{x}$, analogously to Step 2.

Since $\arccos x = \dfrac{\pi}{2} - \arcsin x$, we may first find arcsin $\mathbf{x}$ then subtract it from $\pi/2$ to obtain arccos $\mathbf{x}$. Therefore, we do not present IACOS in detail.

3 IATAN and IACOT

IATAN and IACOT bound ranges of the inverse tangent function and inverse cotangent function, respectively. The series (2) with coefficient $\mathcal{O}(1/n)$ converges too slowly. For faster convergence, we multiply both sides of (2) by x^2. We obtain

$$x^2 \arctan x = x^3 - \frac{x^5}{3} + \frac{x^7}{5} - \cdots + (-1)^{n-1} \frac{x^{2n+1}}{2n-1} + \cdots. \qquad (5)$$

By adding (2) and (5), we have

$$\begin{aligned}(1+x^2) \arctan x &= x + \left(1 - \frac{1}{3}\right) x^3 - \cdots \\ &\quad + (-1)^{n+1} \left(\frac{1}{2n-1} - \frac{1}{2n+1}\right) x^{2n+1} + \cdots. \qquad (6)\end{aligned}$$

Dividing both sides of (6) by $1+x^2$, we obtain

$$\arctan x = \frac{x + 2\left(\frac{1}{1\cdot 3}x^3 - \frac{1}{3\cdot 5}x^5 + \cdots + (-1)^{n+1}\frac{x^{2n+1}}{(2n-1)(2n+1)} + \cdots\right)}{1+x^2}. \qquad (7)$$

Letting $s = \frac{1}{1\cdot 3}x^3 - \frac{1}{3\cdot 5}x^5 + \cdots + (-1)^{n+1}\frac{x^{2n+1}}{(2n-1)(2n+1)} + \cdots$, (7) becomes

$$\arctan x = \frac{x + 2s}{1+x^2}. \qquad (8)$$

The series s converges much faster than (2) does, since its coefficient $a_n = \mathcal{O}(1/n^2)$. In our program, we use equation (8) with interval arithmetic to bound arctan $\mathbf{x}$. We assume $\mathbf{x} \geq 0$, because the inverse tangent function is an odd function; if $\mathbf{x} < 0$, we use arctan $\mathbf{x} = -$arctan $(-\mathbf{x})$. We also assume $|\mathbf{x}| \leq 1$. If $|\mathbf{x}| > 1$, we apply $\arctan x = \pi/2 - \arctan \frac{1}{x}$ to find arctan $\mathbf{x}$.

The series s also converges at $|x| = 1$. However, to further increase the rate of convergence, we only evaluate series (8) when $0 \leq \mathbf{x} \leq \frac{\sqrt{3}}{3}$. If $\mathbf{x} > \frac{\sqrt{3}}{3}$, we let $\mathbf{y} = \frac{\sqrt{3}\mathbf{x}-1}{\sqrt{3}+\mathbf{x}}$. It is easy to prove that $0 \leq \mathbf{y} \leq \frac{\sqrt{3}}{3}$ when $\frac{\sqrt{3}}{3} \leq \mathbf{x} \leq 1$. We then apply the identity $\arctan x = \pi/6 + \arctan \frac{\sqrt{3}x-1}{\sqrt{3}+x}$ to find arctan $\mathbf{x}$.

The maximum absolute truncation error of s from the i-th term is less than $\frac{\overline{x}^{2i+1}}{(2i-1)(2i+1)}$. This is because s is an alternating series. We use $\frac{\overline{x}^{2i+3}}{(2i+1)(2i+3)}/\underline{s}$ to estimate the maximum relative error. If the relative error less than ϵ, we either terminate the computation or add one more term to $\mathbf{s}$, depending on the sign of the last term and whether we need to overestimate $\mathbf{s}$ or underestimate it. When we need to overestimate the series $\mathbf{s}$, we should chop the series after the i-th term if i is an odd number. When we need to underestimate the series $\mathbf{s}$, we should chop the series after the i-th term if i is an even number.

Here is the algorithm to bound arctan $\mathbf{x}$. In the algorithm, a logical variable *Negative* is used to indicate if $\mathbf{x} < 0$ or not. Another logical variable *Even* indicates whether or not we need to overestimate $\mathbf{s}$. The variable Code records argument reduction transformations that have been applied to $\mathbf{x}$.

Algorithm 2.

```
1. Input the interval $\mathbf{x} = [\underline{x}, \overline{x}]$.
2. Compute the lower bound for arctan $\mathbf{x}$.
    (a) Transform $\underline{x}$ to make $0 \le \underline{x} \le \frac{\sqrt{3}}{3}$.
        i. If $\underline{x} < 0$, then Negative $\leftarrow$ True , $\underline{x} \leftarrow -\underline{x}$,
           else Negative $\leftarrow$ False .
        ii. If $\underline{x} > 1$ then $\underline{x} \leftarrow \frac{1}{\underline{x}}$.
        iii. If $\underline{x} > \frac{\sqrt{3}}{3}$ then $\underline{x} \leftarrow \frac{\sqrt{3}\underline{x}-1}{\sqrt{3}+\underline{x}}$.
        iv. We record the transformations by the following:
            If neither (b) nor (c) has been performed, Code $\leftarrow$ 11.
            If only (c) has been performed, Code $\leftarrow$ 12.
            If only (b) has been performed, Code $\leftarrow$ 21.
            If both (b) and (c) have been performed, Code $\leftarrow$ 22.
    (b) Determine the value of Even according to the above transformation.
        If we need to overestimate s then Even $\leftarrow$ False ,
        else Even $\leftarrow$ True .
```

Even $\leftarrow$

Negative \ Code	11	12	21	22
False	*True*	*True*	*False*	*False*
True	*False*	*False*	*True*	*True*

.

```
    (c) Represent $\underline{x}$ by an interval $\mathbf{l}$.
    (d) Compute $\mathbf{s} = \frac{\mathbf{l}^3}{1\cdot 3} - \frac{\mathbf{l}^5}{3\cdot 5} + \cdots + (-1)^{i-1}\frac{\mathbf{l}^{2i+1}}{(2i-1)(2i+1)}$ until $\frac{\overline{l}^{2i+3}}{(2i+1)(2i+3)}/\underline{s} < \epsilon$.
    (e) If Even = True and $i$ is an odd number, then add the next term on s.
```

(f) If *Even* $=$ *False* and i is an even number, then add the next term on $\mathbf{s}$.

(g) $\mathbf{s} \leftarrow \frac{\mathbf{l}+2\mathbf{s}}{1+\mathbf{l}^2}$.

(h) Transform $\mathbf{s}$ according to the argument reduction transformations to make $0 \le \underline{x} \le \frac{\sqrt{3}}{3}$.
If Code $=12$, $\mathbf{s} \leftarrow \pi/6 + \mathbf{s}$.
If Code $=21$, $\mathbf{s} \leftarrow \pi/2 - \mathbf{s}$.
If Code $=22$, $\mathbf{s} \leftarrow \pi/3 - \mathbf{s}$.

(i) If *Negative* $=$ *True* then $\underline{\arctan \mathbf{x}} \leftarrow -\overline{s}$,
else $\underline{\arctan \mathbf{x}} \leftarrow \underline{s}$.

3. Compute the upper bound for arctan $\mathbf{x}$, analogously to Step 2.

Since $\operatorname{arccot} x = \dfrac{\pi}{2} - \arctan x$, to find arccot $\mathbf{x}$ we first find arctan $\mathbf{x}$ then subtract it from $\pi/2$.

4 ISINH

We discuss ISINH in this section. The series (3) converges for all real numbers. To further increase the rate of convergence in our program, we assume $0 \le \mathbf{x} \le 1$. We use $\sinh \mathbf{x} = -\sinh(-\mathbf{x})$ when $\mathbf{x} < 0$. In our program, we repeatedly apply the identity $\sinh x = 3\sinh(x/3) + 4\sinh^3(x/3)$ to evaluate $\sinh \mathbf{x}$ if $1 < |\mathbf{x}| \le 3^3 = 27$. For arguments with $|\mathbf{x}| > 27$, we use INTLIB's exponential function IEXP. IEXP, in turn, has error checking for arguments that would overflow.

Here is our algorithm.

Algorithm 3.

1. Input the interval $\mathbf{x} = [\underline{x}, \overline{x}]$.

2. Compute the lower bound for $\sinh \mathbf{x}$.

(a) If $\underline{x} < 0$, then *Negative* $\leftarrow$ *True* , $\underline{x} \leftarrow -\underline{x}$, *Over* $\leftarrow$ *True* ,
else *Negative* $\leftarrow$ *False* , *Over* $\leftarrow$ *False* .

(b) If $0 \le \underline{x} \le 1$, then Code $\leftarrow 11$.
If $1 < \underline{x} \le 3$, then Code $\leftarrow 12$, $\underline{x} \leftarrow \underline{x}/3$.
If $3 < \underline{x} \le 9$, then Code $\leftarrow 13$, $\underline{x} \leftarrow \underline{x}/9$.
If $9 < \underline{x} \le 27$, then Code $\leftarrow 14$, $\underline{x} \leftarrow \underline{x}/27$.
If $27 < \underline{x}$, then branch to IEXP.

(c) Represent $\underline{x}$ by an interval $\mathbf{l}$.

(d) Compute $\mathbf{s} = \mathbf{l} + \frac{\mathbf{l}^3}{3!} + \frac{\mathbf{l}^5}{5!} + \cdots + \frac{\mathbf{l}^{2i+1}}{(2i+1)!}$ until $\frac{2\bar{l}^{2i+3}}{(2i+3)!\underline{s}} < \epsilon$.

(e) If *Over* = *True* , $\mathbf{s} \leftarrow \mathbf{s} + \frac{2\mathbf{l}^{2i+3}}{(2i+3)!}$.

(f) For $k = 12$ to Code do: $\mathbf{s} \leftarrow 3\mathbf{s} + 4\mathbf{s}^3$.

(g) If *Negative* = *True* then $\underline{\sinh \mathbf{x}} \leftarrow -\overline{s}$,
else $\underline{\sinh \mathbf{x}} \leftarrow \underline{s}$.

3. Compute the upper bound for $\sinh \mathbf{x}$, analogously to Step 2.

We have also have an algorithm, ICOSH, for the hyperbolic cosine function. However, we have not yet perfected it for inclusion in INTLIB at the time we submit this paper.

5 Additional improvements

The Ph. D. dissertations [3] and [9] have recently become available to us. These works provide a careful consideration of implementation of the elementary functions for arbitrary floating-point systems. As here, much (though not all) of these works are based on Taylor approximations. These works emphasize high or maximal precision (approximation by the nearest machine number) in the function values. That requires higher than working precision to attain. Nonetheless, much of their development can be beneficially applied to INTLIB. For instance, formulas appear for the total roundoff error in the evaluation of a polynomial using Horner's scheme in terms of roundoff error in a single operation, as well as for the roundoff error occurring from certain types of argument reductions. With such formulas, we may do the computations in steps 2d and 3d of algorithms 1, 2, and 3 in the machine's floating-point arithmetic, instead of in our portable interval arithmetic. Though some rearrangement of the computations will be necessary, this change will clearly *greatly* increase the speed of these algorithms.

Another *possible* improvement is the use of Chebyshev economization; see e.g. [4, pp. 456–459]. The additional error terms introduced by dropping the trailing terms in the Chebyshev basis can be easily bounded using interval arithmetic. However, the resulting polynomial coefficients are no longer as simply representable, causing possible minor difficulties. For example, some of the roundoff error bounds in [3] and [9] depend on the zero-th order coefficient of the polynomial being exactly representable in the machine. Also, computing these coefficients complicates maintenance of the routines, should it be desirable to increase or decrease the degree of approximation.

6 Summary

Based on the above algorithms, we have developed subroutines: IASIN, IACOS, IATAN, IACOT, and ISINH in FORTRAN–77 ([1]). These subroutines have been tested and exhaustively documented. They are included in the readily available and portable software library INTLIB ([7]). INTLIB has been submitted to ACM Transactions on Mathematical Software ([8]). We are continuing to test and improve it. A test version is available through the Internet via FTP at `ftp.ucs.usl.edu` in the directory `/pub/interval_math/intlib`.

Acknowledgements

We wish to acknowledge the referees for their careful reading and useful suggestions. We also wish to thank George Corliss for editorial changes that made the paper, and in particular for making Step 2b of Algorithm 2 easier to read.

References

[1] *American national standard programming language FORTRAN ANSI X3.9–1978.* ANSI, New York, 1978.

[2] Bauch, H., Jahn, K. U., Oelschlägel, D., Süsse, H., and Wiebigke, V. *Intervallmathematik – Theorie und Anwendungen.* Teubner, Leipzig, 1987.

[3] Braune, K. D. *Hochgenaue Standardfunktionen für reele und komplexe Punkte und Intervalle in beliebigen Gleitpunktrastern.* Ph. D. dissertation, Universität Karlsruhe, 1987.

[4] Burden, R. L. and Faires, D. J. *Numerical analysis, fourth edition.* PWS – Kent, Boston, 1988.

[5] Cody, W. J. and Waite, W. *Software manual for the elementary functions.* Prentice-Hall, Englewood Cliffs, NJ, 1980.

[6] Kearfott, R. B. and Novoa, M. *INTBIS, a portable interval Newton/bisection package (algorithm 681).* ACM Trans. Math. Software **16** (2) (1990), pp. 152–157.

[7] Kearfott, R. B., Dawande, M., Du, K. S., and Hu, C. Y. *INTLIB: a portable FORTRAN–77 elementary function library.* Interval Computations 3 (5) (1992), pp. 96–105.

[8] Kearfott, R. B., Dawande, M., Du, K. S., and Hu, C. Y. *INTLIB: a reasonably portable interval elementary function library.* Submitted to ACM Trans. Math. Software, 1993.

[9] Krämer, W. *Inverse Standardfunktionen für reelle und komplexe Intervallargumente mit a priori Fehlerabschätzungen.* Ph. D. dissertation, Universität Karlsruhe, 1987.

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

[11] Ralston, A. *A first course in numerical analysis.* Mc-Graw-Hill, New York, 1965.

[12] Wolff von Gudenberg, J. *Einbettung allgemeiner Rechnerarithmetik in Pascal mittels eines Operatorkonzeptes und Implementierung der Standardfunktionen mit optimaler Genauigkeit.* Ph. D. dissertation, University of Karlsruhe, 1980.

C. Hu, A. Awad
Department of Computer and
Mathematical Sciences
University of Houston-Downtown
Houston, TX 77002
USA

R. B. Kearfott
Department of Mathematics
University of Southwestern Louisiana
Lafayette, LA 70504
USA