Accurate and Monotone Approximations of Some Transcendental Functions

Warren E. Ferguson, Jr. Tom Brightman

Cyrix Corporation, P.O. Box 850118, Richardson, Texas 75081-0118

Abstract

A technique for computing monotonicity preserving approximations $F_a(x)$ of a function $F(x)$ is presented. This technique involves computing an extra precise approximation of $F(x)$ that is rounded to produce the value of $F_a(x)$. For example, only a few extra bits of precision are used to make the accurate transcendental functions found on the Cyrix™ FasMath™ line of 80387 compatible math coprocessors monotonic.

1 Introduction

The IEEE Standard for Binary Floating-Point Arithmetic [1] is a landmark in the field of computer arithmetic. This standard establishes the formats used to store floating-point numbers and defines the results that must be returned by the fundamental mathematical operations of addition, subtraction, multiplication, division, square root, and remainder.

This standard, however, omits any discussion of what should be returned as the value of transcendental functions like the sine or the logarithm. One reason for this omission is that no simple algorithms that yield correctly the rounded value of these functions are known. It is therefore natural to focus on easily computable approximations of these functions that preserve their important properties.

This paper shows that monotonic approximations of the sine, cosine, tangent, arctangent, exponential, and logarithm, on restricted domains, can be derived by computing extra precise approximations. For example, an approximation of the sine function on $[-\pi/4,\pi/4]$ that is accurate to 66 bits of precision will necessarily be monotonic when rounded to 64 bits of precision. We have used this technique to establish the monotonicity of a Cyrix FasMath coprocessor's polynomial based approximations of transcendental functions [2]. Since this technique does not depend on how the approximation is determined, then it also can be applied to approximations derived by other means, e.g., CORDIC based approximations.

2 Machine Numbers

The IEEE Standard [1] defines four floating-point number formats: single basic, single extended, double basic, and double extended. Within a given format the representable numbers can be described as

$$x = (-1)^s\, 2^E\, (b_0 \bullet b_1 b_2 \cdots b_{p-1})$$

where $s = 0$ or 1, $E_{min} \le E \le E_{max}$, $b_i = 0$ or 1.

We consider exclusively the situation where arguments, and the approximate values of transcendental functions, are representable as double extended format numbers (machine numbers). Math coprocessors compatible with the 80387 math coprocessor use a double extended format with $E_{max} = +16383$, $E_{min} = -16382$, $p = 64$.

Of particular interest are the following two classes of *machine numbers*: *denormalized numbers* (including zero) and *normalized numbers*. Consecutive nonnegative denormalized numbers $m < \bar{m}$ have the form

$$m = (I)\, 2^{-16445} \quad \text{and} \quad \bar{m} = m + 2^{-16445}$$

where I is an integer that satisfies $0 \le I \le 2^{63}$; note that $-16445 = -16382 - 63$.

CH3015-5/91/0000/0237$01.00 © 1991 IEEE

Normalized numbers can be classified by the *binade* to which they belong; in the binade E consecutive positive normalized numbers $m < \overline{m}$ have the form

$$m = (1+f)2^E \quad \text{and} \quad \overline{m} = m + 2^{E-63}$$

where $2^{63}f$ and E are integers satisfying $0 \le 2^{63}f < 2^{63}$ and $-16382 \le E \le +16383$.

In the following discussion the symbols $\underline{m}$, m, and $\overline{m}$ will always be used to denote consecutive machine numbers ordered so that $\underline{m} < m < \overline{m}$.

3 Transcendental Functions

Cyrix FasMath coprocessors use polynomial based methods [3,4,5] to approximate the five basic transcendental functions: $2^X - 1$ on $[-1/2,1/2]$, $Log_2(1+x)$ on $[1/\sqrt{2}-1, \sqrt{2}-1]$, Sin(x) and Tan(x) on $[-\pi/4, \pi/4]$, and ATan(x) on [0,1].

Argument reduction techniques are used to expand the domain of these basic transcendental functions. For example, FasMaths can return the approximate value of $2^x - 1$ for arguments in [-1,1]. Given such an argument it first uses a polynomial based method to determine the approximate value of $2^{x/2} - 1$ and then recovers the approximate value of $2^x - 1$ via the identity

$$2^x - 1 = (2^{x/2} - 1)(\{2^{x/2} - 1\} + 2) .$$

FasMaths can determine the approximate value of $Log_2(x)$ for all positive values of x. To do so it first determines the value of p for which $2^{-p}x$ belongs to $[1/\sqrt{2}, \sqrt{2}]$. Using this value of p polynomial based methods are used to determine the approximate value of $Log_2(1 + \{2^{-p}x - 1\}) = Log_2(2^{-p}x)$. Finally the identity

$$Log_2(x) = p + Log_2(2^{-p}x)$$

is used to recover the approximate value of $Log_2(x)$.

FasMaths can determine the approximate value of Sin(x), Cos(x), and Tan(x) for arguments x satisfying $|x| \le 2^{63}$. To do so they first use the symmetric partial remainder instruction to determine the last three bits of the quotient q and the exact remainder r, with $|r| \le \pi/4$, when x is divided by an approximation to $\pi/2$ with 68 bits of precision. Once the approximate value of Sin(r) or Tan(r) is determined by polynomial based methods standard trigonometric identities are used to recover the values of Sin(x), Cos(x), and Tan(x). These trigonometric identities require the last three bits of q to determine the appropriate transformation. FasMaths also use the identity

$$Cos(r) = \sqrt{1 - Sin^2(r)}$$

to recover the approximate value of Cos(r) for arguments $|r| \le \pi/4$ from the approximate value of Sin(r). A FasMath's approximation of Sin(r), Cos(r), and Tan(r) preserve the identities

$$Sin(x) = -Sin(-x),$$
$$Cos(x) = Cos(-x), \text{ and}$$
$$Tan(x) = -Tan(-x).$$

The use of the symmetric partial remainder instruction to reduce input arguments also means that FasMaths return rounded-period approximations of trigonometric functions.

Finally, FasMaths can determine the approximate value of ATan(x) for all values of x. To do so it uses the identities

$$ATan(x) = -ATan(-x), \text{ and}$$
$$ATan(x) = \pi/2 - ATan(1/x)$$

to reduce a general argument to one between 0 and 1. Once again, an approximate value of $\pi/2$ with 68 bits of precision is used when the last identity is applied.

4 Measures of Errors

Two of the most frequently cited measures of error associated with an approximation F_a of a nonzero number F are the error *err* and relative error *relerr* defined by

$$err(F_a) \equiv F_a - F \text{ , and} \quad relerr(F_a) \equiv \frac{F_a - F}{F} .$$

Note that $F_a = (1 + relerr(F_a))\, F$.

We say that F_a is a *p-bit approximation* of F if the error amounts to no more than one-half of one bit in the p-th significant bit of the normalized binary representation of F.

To understand the implication of this definition let us suppose that F is a nonnegative normalized number in the binade E,

$$F = (1 + f)\ 2^E \qquad \text{where} \qquad 0 \le f < 1.$$

If

$$F_a = (1 + f + \epsilon)\ 2^E, \text{ with}$$

$$|\,\epsilon\,| \le \frac{1}{2}\ 2^{-p+1} = 2^{-p}\ ,$$

then by definition F_a is a p-bit approximation of F. Note that the error in F_a is represented as an error ϵ in its *significand* f. If F_a is a p-bit approximation of F, then it follows that

$$|\,\text{err}(F_a)\,| \;=\; |\,\epsilon\,|\ 2^E, \text{ and}$$

$$|\,\text{relerr}(F_a)\,| \;= \frac{|\,\epsilon\,|}{1+f} \le 2^{-p}\ .$$

Since $1 \le 1 + f < 2$, then we infer that p-bit approximations that are not (p+1)-bit approximations have relative errors whose magnitudes lie between 2^{-p-1} and 2^{-p}; the larger the significand the smaller the associated relative error. We summarize these observations as follows:

Fact: Let $F = \pm(1+f)\ 2^E$ belong to the binade E. If the magnitude of the relative error relerr(F_a) in an approximation F_a of F is no larger than $2^{-p}/(1+f)$, then F_a is a p-bit approximation of F.

This variation by a factor of 2 in the upper bound on the magnitude of the relative error of p-bit approximations, called precision wobble [5], is one reason why base 2 arithmetic is preferred over base 16 arithmetic. Base 16 (hexadecimal) arithmetic has a precision wobble 8 times larger than that of base 2 (binary) arithmetic.

5 Accurate Approximations

Consider an approximation $F_a(x)$ of a function F(x). If F(x) belongs to the binade E(x), then we can represent F(x) as

$$F(x) = \pm\{1 + f(x)\}\ 2^{E(x)},$$

where $1 \le 1 + f(x) < 2$. If we express the error in the approximation of F(x) by $F_a(x)$ as an error $\epsilon(x)$ associated with the significand of F(x),

$$F_a(x) = \pm\{1 + f(x) + \epsilon(x)\}\ 2^{E(x)},$$

then we find that

$$|\,\text{relerr}(F_a(x))\,| = \frac{|\,\epsilon(x)\,|}{1 + f(x)}\ .$$

If $F_a(x)$ is a p-bit approximation of F(x), then $|\,\epsilon(x)\,| \le 2^{-p}$ and so

$$|\,\text{relerr}(F_a(x))\,| \le \frac{2^{-p}}{1 + f(x)}\ .$$

Note that the upper bound on the magnitude of the relative error of p-bit approximations wobbles not when the argument x crosses a binade's boundary, but rather when the value F(x) crosses a binade's boundary.

For each of the five basic transcendental functions, the upper-most dashed lines in Figures 2 depict the upper bound on the relative error that must be satisfied by any 64-bit precise approximation.

6 Monotone Approximations

Informally, the monotonic behavior of an approximation can be established by making use of three facts: the monotonic behavior exhibited by the function, the discrete nature of the machine numbers, and the accuracy of the approximation.

For example, consider the sine function on $[0,\pi/4]$. On this domain Sin(x) is a monotone increasing function whose slope is never less than $\text{Cos}(\pi/4) \simeq 0.7$. If $m < \bar{m}$ are two consecutive machine numbers that belong to the binade E, then Sin(m) and Sin($\bar{m}$) will differ by at least $0.7*(\bar{m} - m) \simeq (0.7)\ 2^{E-63}$. If the error in the approximation to Sin(x) is less than half of this difference throughout the binade E, then the approximation will also behave monotonically.

This monotonicity argument implies that the approximation is determined using a higher precision arithmetic than that required in the final result. This is relatively easy to achieve for FasMaths because they implement the full IEEE 754 Standard, and such an implementation requires that the five basic operations of addition, subtraction, multiplication, division and square root be computed internally to a higher precision. A mathematically precise version of this monotonicity argument is presented in the following theorem.

Theorem: Let $F(x)$ be a monotonic function defined on the interval [a,b]. Let $F_a(x)$ be an approximation of $F(x)$ whose associated relative error admits the bound

$$|\text{relerr}(F_a(x))| \leq R$$

for some constant $R < 1$. If for every pair $m < \bar{m}$ of consecutive machine numbers in [a,b]

$$R < R(m,\bar{m}) \equiv \frac{|F(\bar{m}) - F(m)|}{|F(\bar{m})| + |F(m)|},$$

then $F_a(x)$ exhibits on the set of machine numbers in [a,b] the same monotonic behavior exhibited by $F(x)$ on [a,b].

Proof: Let $m < \bar{m}$ be any pair of consecutive machine numbers in [a,b]. From the assumptions we find that

$$|F_a(m) - F(m)| + |F_a(\bar{m}) - F(\bar{m})|$$

$$= |\text{relerr}(F_a(m))F(m)| + |\text{relerr}(F_a(\bar{m}))\, F(\bar{m})|$$

$$= R\,\{\,|F(m)| + |F(\bar{m})|\,\}$$

$$\leq R(m,\bar{m})\,\{\,|F(m)| + |F(\bar{m})|\,\}$$

$$\leq |F(\bar{m}) - F(m)|.$$

As illustrated in Figure 1, this inequality shows that the error bar centered at $F(m)$ with half-width $R\,|F(m)|$ containing $F_a(m)$, and the error bar centered at $F(\bar{m})$ with half-width $R\,|F(\bar{m})|$ containing $F_a(\bar{m})$, cannot overlap. Therefore the function F on [a,b], and the approximation F_a on the machine numbers in [a,b], must exhibit the same monotonic behavior. □

Of course the user never sees the value of the underlying higher precision approximation, but only its value truncated to double extended format. To insure that the value presented to the user also behaves monotonically it is important that this truncation to double extended format preserves monotonicity. It is simple to verify that if the truncation trunc() used is consistently truncation by rounding or truncation by chopping, then monotonicity is preserved because $x < y$ implies that $\text{trunc}(x) \leq \text{trunc}(y)$. Therefore when the assumptions of the previous theorem hold, then the truncated approximation is weakly monotonic whenever the function is strictly monotonic.

Before applying this theorem to the five basic transcendental functions it is convenient to simplify the expression for $R(m,\bar{m})$ as follows. First note that each of these transcendental functions is monotone increasing with a single zero at a machine number m_0 in the domain. The monotonic behavior of the approximation at this zero is ensured by the fact that $R < 1$, for this fact implies that the approximation is zero only when the function is zero and that the approximation has the same sign as the function. Since m_0 is a machine number, then the function has the same sign at consecutive machine numbers $m < \bar{m}$. Therefore, the denominator in the expression for $R(m,\bar{m})$ can be simplified to

$$|F(\bar{m})| + |F(m)| = |F(\bar{m}) + F(m)|.$$

Note that

$$F(\bar{m}) + F(m) = \begin{cases} 2F(m)\ \{1 + \dfrac{F(\bar{m}) - F(m)}{2\,F(m)}\}; & F(\bar{m}) > F(m) > 0 \\ 2F(\bar{m})\ \{1 + \dfrac{F(m) - F(\bar{m})}{2\,F(\bar{m})}\}; & F(m) < F(\bar{m}) < 0 \end{cases}$$

where the fraction inside the curly braces is a positive quantity. Using these facts the expression for $R(m,\bar{m})$ can be simplified to yield

$$R(m,\bar{m}) = \left|\frac{F(\bar{m}) - F(m)}{F(\bar{m}) + F(m)}\right| = \frac{S(m,\bar{m})}{1 + S(m,\bar{m})}$$

where

$$S(m,\bar{m}) \equiv \frac{|F(\bar{m}) - F(m)|}{2\min\{\,|F(m)|, |F(\bar{m})|\,\}}.$$

If we interpret $S(m,\bar{m})$ as ∞ when either m or $\bar{m}$ is equal to m_0, and $R(m,\bar{m})$ as 1 when $S(m,\bar{m})$ is ∞, then this simplified expression for $R(m,\bar{m})$ remains valid when either m or $\bar{m}$ is equal to m_0.

In Table 1 we present the expression for S for each of the five basic transcendental functions. Note that the expressions associated with $2^x - 1$ and $\text{Log}_2(1+x)$ have been split into two cases, one case for consecutive negative machine numbers $\underline{m} < m < 0$ and another case for consecutive positive machine numbers $0 < m < \bar{m}$. These expressions for S have been written so that they clearly show their dependence on the average and difference of the consecutive machine numbers. Table 2 displays highly accurate approximations of S that were obtained from the expressions in Table 1.

These approximations of S were obtained by replacing functions depending on the difference of two consecutive machine numbers by the first term of the relevant MacLaurin expansion. The high accuracy of these approximations is a result of the fact that the difference between consecutive machine numbers is never greater than 2^{-63} on the domains under consideration.

For each of the five basic transcendental functions, the dotted line of Figures 2 depict upper bounds on the relative error that must be satisfied by any monotonic approximation whose arguments are obtained from double extended format machine numbers. These curves were obtained from the relationship between R and S and the expressions for S displayed in Table 2.

Note that the upper bounds on the relative error of approximations wobble when the value of the function crosses a binade boundary, while the upper bounds on the relative error of monotonic approximations wobble when the argument of the function crosses a binade boundary.

The bottom-most dash-dotted curves Figure 2 depict rigorous bounds on the relative error in the approximations used by FasMaths. In all cases the upper bound on this error lies below the bounds required of accurate and monotonic double extended format approximations. In particular, note that the relative errors lie below the curve characterizing approximations that have full double extended format (64-bit) accuracy. This observation demonstrates mathematically that the approximate values of the basic transcendental functions computed by a FasMath are either the correctly rounded value or one of its two immediate double extended format neighbors.

The appendix illustrates the type of analysis used to derive the worst case bound on the relative error of monotonic approximations to Sin(x).

7 Conclusion

The techniques presented in this paper has been used to prove mathematically that the approximations delivered by FasMath coprocessors posses the desirable properties of accuracy and monotonicity [2]. While FasMaths use polynomial based approximations, the techniques presented in this paper are generally applicable to any method for which one can compute upper bounds on the relative error of the approximation. In particular, it could be used to justify the monotonic behavior of approximations derived by CORDIC based methods.

One interesting application of the relative error curves depicted in Figures 2 can be described as follows. The gap between the total error curve and the bound on accurate approximations suggests that the larger the gap, the more frequently a FasMath should return a correctly rounded answer, see also [6]. By using Brent's MP package, owners of FasMaths can verify this claim. Remember, however, that by the nature of its derivation the curves depicting the relative error in a FasMath's approximations are usually a significant overestimate of the actual relative error.

Accuracy and monotonicity results for transcendental functions can also be found in [7]. Recently Dr. W. H. Kahan [8] has informed us that he has presented at UC Berkeley techniques similar to those presented in this paper.

Appendix

Since R() is an increasing function of S(), then R() achieves its least value when S() achieves its least value. For the sine function the bound on the relative error of monotonic approximations is given as

$$\mathrm{S}(\mathrm{m},\bar{\mathrm{m}}) = \frac{\bar{\mathrm{m}} - \mathrm{m}}{2\,\mathrm{Tan(m)}}.$$

Consider the case when $\mathrm{m} < \bar{\mathrm{m}}$ are consecutive machine numbers in the binade E. The numerator of S is constant at value 2^{E-63} while the denominator of S is an increasing function of m. Therefore within the binade E the least value of S is assumed at its right endpoint.

The relevant domain for the sine function is $[0,\pi/4]$. The machine numbers in this domain consist of the whole of the binades with $-16382 \le E \le -2$ and that portion of the binade with $E = -1$ consisting of machine numbers less than $\pi/4$. We therefore conclude that

$$\min \mathrm{S}(\mathrm{m},\bar{\mathrm{m}}) =$$

$$= 2^{-65} \min\left\{ \min_{\substack{x = 2^{E+1} \\ -16382 \le E \le -2}} \left(\frac{x}{\mathrm{Tan}(x)}\right), \frac{1}{\mathrm{Tan}(\pi/4)} \right\}.$$

Since the function x/Tan(x) is a decreasing function of x for $x \leq 1/2$, then the least value of x/Tan(x) is achieved at the largest value of x. Consequently

$$\min S(m,\bar{m}) = 2^{-65} \min \left\{ \frac{1/2}{\mathrm{Tan}(1/2)}, \frac{1}{\mathrm{Tan}(\pi/4)} \right\}$$
$$= \frac{2^{-66}}{\mathrm{Tan}(1/2)} \simeq (1.83)\, 2^{-66}$$

and so to a high degree of approximation

$$\min R(m,\bar{m}) \simeq \min S(m,\bar{m}) \simeq (1.83)\, 2^{-66}.$$

It is interesting to note that the informal argument would have predicted that the most stringent bound on $R(m,\bar{m})$ would occur when $m \simeq \pi/4$ rather than when $m = 1/2$.

References

[1] "IEEE Standard for Binary Floating-Point Arithmetic", ANSI/IEEE Std 754-1985, The Institute of Electrical and Electronics Engineers, Inc., New York, New York, 10017.

[2] *FasMath 83D87 Accuracy Report*, Cyrix Corporation, Richardson, Texas, May 1989.

[3] *Computer Approximations*, John F. Hart et. al., Robert E. Krieger Publishing Company, Malabar, Florida, 1978.

[4] *Software Manual for the Elementary Functions*, William J. Cody, Jr. and William Waite, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1980.

[5] *Computer Evaluation of Mathematical Functions*, C. T. Fike, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1968.

[6] "Evaluating Elementary Functions in a Numerical Coprocessor Based on Rational Approximations", Israel Koren and Ofra Zinaty, IEEE Trans. Computers, Vol. 39, No. 8, August 1990, pp. 1030-1037.

[7] "High Accuracy Standard Functions for Real and Complex Intervals", K. Braune and W. Krämer, in *Computerarithmetic: Scientific Computation and Programming Languages*, E. Kaucher, U. Kulisch, Ch. Ullrich (ed.), Teubner, Stuttgart, pp. 81-114, 1987.

[8] Personal communication, W. H. Kahan, February, 1990.

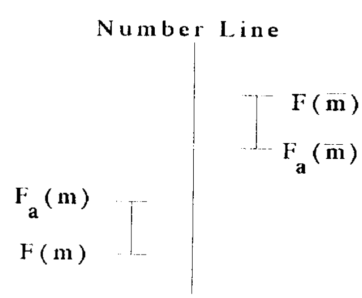


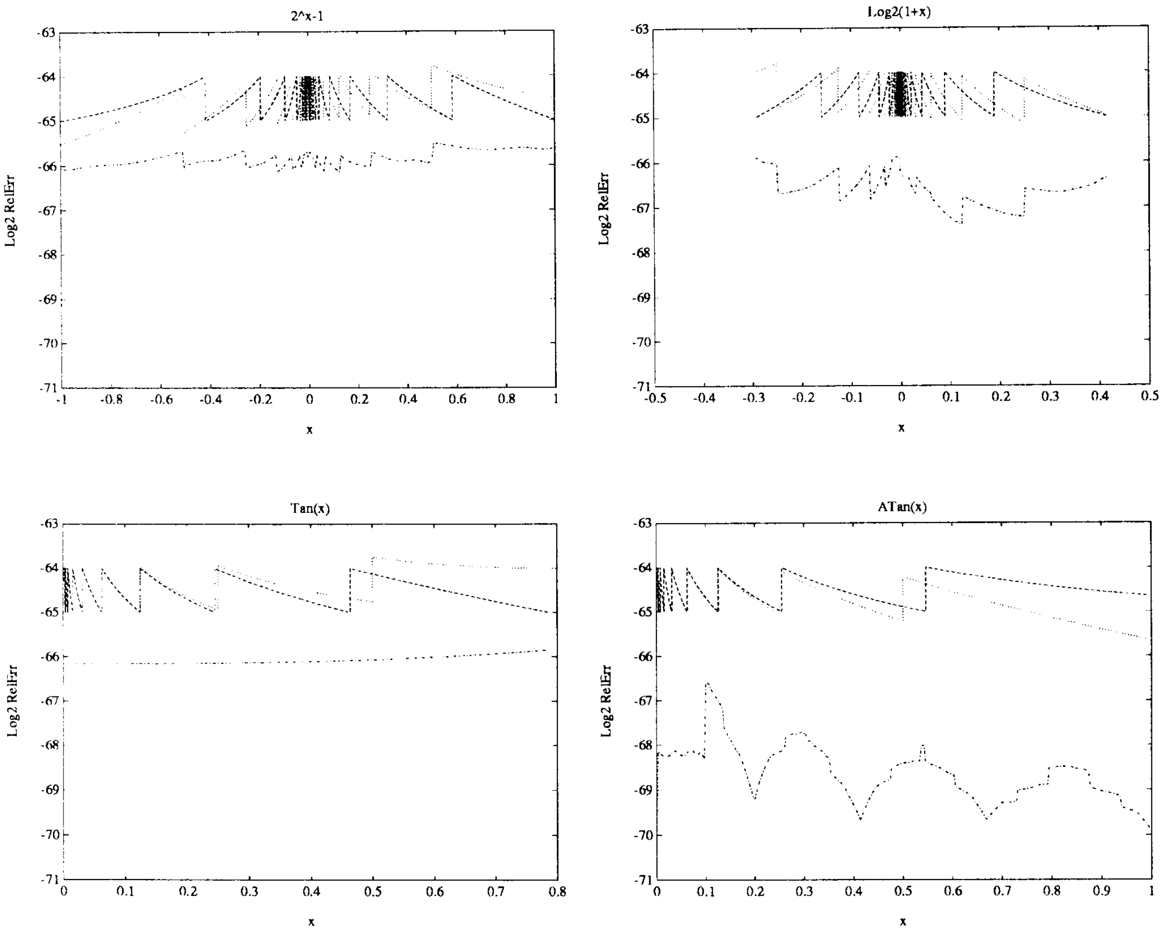
Figure 1: Relationship between F and F_a.

Function	Interval	S
Sin(x)	$[0,\pi/4]$	$\text{Cos}(\frac{m+\overline{m}}{2})\,\frac{\text{Sin}(\frac{\overline{m}-m}{2})}{\text{Sin}(m)}$
Tan(x)	$[0,\pi/4]$	$\frac{1}{\text{Cos}(\overline{m})}\,\frac{\text{Sin}(\overline{m}-m)}{2\ \text{Sin}(m)}$
ATan(x)	$[0,1]$	$\frac{\text{ATan}(\frac{\overline{m}-m}{1+m\overline{m}})}{2\ \text{ATan}(m)}$
2^x-1	$[-1,0]$	$2^{-\frac{m-\underline{m}}{2}}\,\frac{\text{Sinh}(\frac{m-\underline{m}}{2}\ \text{Log}_e(2))}{2^{-m}-1}$
2^x-1	$[0,1]$	$2^{\frac{\overline{m}-m}{2}}\,\frac{\text{Sinh}(\frac{\overline{m}-m}{2}\ \text{Log}_e(2))}{1-2^{-m}}$
$\text{Log}_2(1+x)$	$[\frac{1}{\sqrt{2}}-1,0]$	$\frac{\text{Log}_e(1-\frac{m-\underline{m}}{1+m})}{2\ \text{Log}_e(1+m)}$
$\text{Log}_2(1+x)$	$[0,\sqrt{2}-1]$	$\frac{\text{Log}_e(1+\frac{\overline{m}-m}{1+m})}{2\ \text{Log}_e(1+m)}$

Table 1: Expressions for S.

Function	Interval	S
Sin(x)	$[0,\pi/4]$	$\frac{\overline{m}-m}{2\ \text{Tan}(m)}$
Tan(x)	$[0,\pi/4]$	$\frac{\overline{m}-m}{\text{Sin}(2m)}$
ATan(x)	$[0,1]$	$\frac{\overline{m}-m}{2\ (1+m^2)\ \text{ATan}(m)}$
2^x-1	$[-1,0]$	$\frac{m-\underline{m}}{2\ \text{Log}_2(e)\ (2^{-m}-1)}$
2^x-1	$[0,1]$	$\frac{\overline{m}-m}{2\ \text{Log}_2(e)\ (1-2^{-m})}$
$\text{Log}_2(1+x)$	$[\frac{1}{\sqrt{2}}-1,0]$	$\frac{m-\underline{m}}{2\ (1+m)\ \text{Log}_e(\frac{1}{1+m})}$
$\text{Log}_2(1+x)$	$[0,\sqrt{2}-1]$	$\frac{\overline{m}-m}{2\ (1+m)\ \text{Log}_e(1+m)}$

Table 2: Accurate approximations of S.



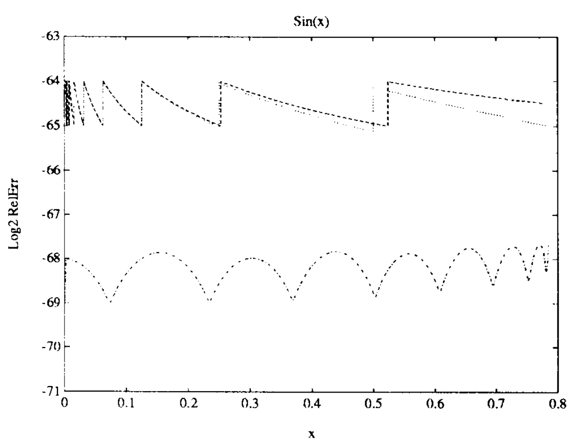


Figure 2: Relative error bounds for the transcendental fuctions 2^x-1, Log2(1+x), Tan(x), ATan(x), and Sin(x). In each graph: (1) the bottom most dash-dotted curve is an upper bound on the relative error in FasMath approximations prior to being rounded to double extended precision, (2) the top most dotted curve bounds relative errors of approximations that are monotonic, and (3) the top most dashed curve bounds relative errors of approximations that have 64-bit precision.