

An Implementation of Sin and Cos Using Gal's Accurate Tables

Pascal Leroy (phl)

2025-02-02

This document describes the implementation of functions `Sin` and `Cos` in Principia. The goals of that implementation are to be portable (including to machines that do not have a fused multiply-add instruction), achieve good performance, and ensure correct rounding.

Overview

The implementation follows the ideas described by [GB91] and uses accurate tables produced by the method presented in [SZ05]. It guarantees correct rounding with a high probability. In circumstances where it cannot guarantee correct rounding, it falls back to the (slower but correct) implementation provided by the CORE-MATH project [SZG22] [ZSG+24]. More precisely, the algorithm proceeds through the following steps:

- perform argument reduction using Cody and Waite's algorithm in double precision (see [Mul+10, p. 379]);
- if argument reduction loses too many bits (i.e., the argument is close to a multiple of $\frac{\pi}{2}$), fall back to `cr_sin` or `cr_cos`;
- otherwise, uses accurate tables and a polynomial approximation to compute `Sin` or `Cos` with extra accuracy;
- if the result has a “dangerous rounding configuration” (as defined by [GB91]), fall back to `cr_sin` or `cr_cos`;
- otherwise return the rounded result of the preceding computation.

Notation and Accuracy Model

In this document we assume a base-2 floating-point number system with M significand bits¹ similar to the IEEE formats. We define a real function m and an integer function e denoting the *significand* and *exponent* of a real number, respectively:

$$x = \pm m(x) \times 2^{e(x)} \quad \text{with} \quad 2^{M-1} \leq m(x) \leq 2^M - 1$$

Note that this representation is unique. Furthermore, if x is a floating-point number, $m(x)$ is an integer.

The *unit of the last place* of x is defined as:

$$u(x) := 2^{e(x)}$$

In particular, $u(1) = 2^{1-M}$ and:

$$\frac{|x|}{2^M} < \frac{|x|}{2^M - 1} \leq u(x) \leq \frac{|x|}{2^{M-1}} \quad (1)$$

We ignore the exponent bias, overflow and underflow as they play no role in this discussion.

Finally, for error analysis we use the accuracy model of [Higo2], equation (2.4): everywhere they appear, the quantities δ_i represent a roundoff factor such that $|\delta_i| < u = 2^{-M}$ (see pages 37 and 38). We also use θ_n and γ_n with the same meaning as in [Higo2], lemma 3.1.

¹In binary64, $M = 53$.

Approximation of $\frac{\pi}{2}$

To perform argument reduction, we need to build approximations of $\frac{\pi}{2}$ with extra accuracy and analyse the circumstances under which they may be used and the errors that they entail on the reduced argument.

Let $z \geq 0$. We start by defining the truncation function $\text{Tr}(\kappa, z)$ which clears the last κ bits of the significand of z :

$$\text{Tr}(\kappa, z) := \lfloor 2^{-\kappa} m(z) \rfloor 2^\kappa u(z)$$

We have:

$$z - \text{Tr}(\kappa, z) = (2^{-\kappa} m(z) - \lfloor 2^{-\kappa} m(z) \rfloor) 2^\kappa u(z)$$

The definition of the floor function implies that the quantity in parentheses is in $[0, 1[$ and therefore:

$$0 \leq z - \text{Tr}(\kappa, z) < 2^\kappa u(z)$$

Furthermore if the bits that are being truncated start with exactly k zeros we have the stricter inequality:

$$2^{\kappa'-1} u(z) \leq z - \text{Tr}(\kappa, z) < 2^{\kappa'} u(z) \quad \text{with} \quad \kappa' = \kappa - k \quad (2)$$

This leads to the following upper bound for the unit of the last place of the truncation error:

$$u(z - \text{Tr}(\kappa, z)) < 2^{\kappa'-M+1} u(z)$$

which can be made more precise by noting that the function u is always a power of 2:

$$u(z - \text{Tr}(\kappa, z)) = 2^{\kappa'-M} u(z) \quad (3)$$

Two-Term Approximation

In this scheme we approximate $\frac{\pi}{2}$ as the sum of two floating-point numbers:

$$\frac{\pi}{2} \simeq C_1 + \delta C_1$$

which are defined as:

$$\begin{cases} C_1 &:= \text{Tr}\left(\kappa_1, \frac{\pi}{2}\right) \\ \delta C_1 &:= \left\lfloor \frac{\pi}{2} - C_1 \right\rfloor \end{cases}$$

Equation (2) applied to the definition of C_1 yields:

$$2^{\kappa'_1-1} u\left(\frac{\pi}{2}\right) \leq \frac{\pi}{2} - C_1 < 2^{\kappa'_1} u\left(\frac{\pi}{2}\right)$$

where $\kappa'_1 \leq \kappa_1$ accounts for any leading zeroes in the bits of $\frac{\pi}{2}$ that are being truncated. Accordingly equation (3) yields, for the unit of the last place:

$$u\left(\frac{\pi}{2} - C_1\right) = 2^{\kappa'_1-M} u\left(\frac{\pi}{2}\right)$$

Noting that the absolute error on the rounding that appears in the definition of δC_1 is bounded by $\frac{1}{2} u\left(\frac{\pi}{2} - C_1\right)$, we obtain the absolute error on the two-term approximation:

$$\left| \frac{\pi}{2} - C_1 - \delta C_1 \right| \leq \frac{1}{2} u\left(\frac{\pi}{2} - C_1\right) = 2^{\kappa'_1-M-1} u\left(\frac{\pi}{2}\right) \quad (4)$$

and the following upper bound for δC_1 :

$$\begin{aligned} |\delta C_1| &< \frac{\pi}{2} - C_1 + \frac{1}{2} u\left(\frac{\pi}{2} - C_1\right) \\ &< 2^{\kappa'_1} u\left(\frac{\pi}{2}\right) + 2^{\kappa'_1-M-1} u\left(\frac{\pi}{2}\right) = 2^{\kappa'_1} (1 + 2^{-M-1}) u\left(\frac{\pi}{2}\right) \end{aligned} \quad (5)$$

This scheme gives a representation with a significand that has effectively $2M - \kappa'_1$ bits and is such that multiplying C_1 by an integer less than or equal to 2^{κ_1} is exact.

Three-Term Approximation

In this scheme we approximate $\frac{\pi}{2}$ as the sum of three floating-point numbers:

$$\frac{\pi}{2} \simeq C_2 + C'_2 + \delta C_2$$

which are defined as:

$$\begin{cases} C_2 &:= \text{Tr}\left(\kappa_2, \frac{\pi}{2}\right) \\ C'_2 &:= \text{Tr}\left(\kappa_2, \frac{\pi}{2} - C_2\right) \\ \delta C_2 &:= \left\llbracket \frac{\pi}{2} - C_2 - C'_2 \right\rrbracket \end{cases}$$

Equation (2) applied to the definition of C_2 yields:

$$2^{\kappa'_2-1} \mathfrak{u}\left(\frac{\pi}{2}\right) \leq \frac{\pi}{2} - C_2 < 2^{\kappa'_2} \mathfrak{u}\left(\frac{\pi}{2}\right) \quad (6)$$

where $\kappa'_2 \leq \kappa_2$ accounts for any leading zeroes in the bits of $\frac{\pi}{2}$ that are being truncated. Accordingly equation (3) yields, for the unit of the last place:

$$\mathfrak{u}\left(\frac{\pi}{2} - C_2\right) = 2^{\kappa'_2-M} \mathfrak{u}\left(\frac{\pi}{2}\right)$$

Similarly, equation (2) applied to the definition of C'_2 yields:

$$\begin{aligned} 2^{\kappa'_2-1} \mathfrak{u}\left(\frac{\pi}{2} - C_2\right) &\leq \frac{\pi}{2} - C_2 - C'_2 < 2^{\kappa'_2} \mathfrak{u}\left(\frac{\pi}{2} - C_2\right) \\ 2^{\kappa'_2+\kappa'_2-M-1} \mathfrak{u}\left(\frac{\pi}{2}\right) &\leq < 2^{\kappa'_2+\kappa'_2-M} \mathfrak{u}\left(\frac{\pi}{2}\right) \end{aligned}$$

where $\kappa'_2 \leq \kappa_2$ accounts for any leading zeroes in the bits of $\frac{\pi}{2} - C_2$ that are being truncated. Note that normalization of the significand of $\frac{\pi}{2} - C_2$ effectively drops the zeroes at positions κ_2 to κ'_2 and therefore the computation of C'_2 applies to a significand aligned on position κ'_2 .

It is straightforward to transform these inequalities using (6) to obtain bounds on C'_2 :

$$2^{\kappa'_2} \left(\frac{1}{2} - 2^{\kappa'_2-M}\right) \mathfrak{u}\left(\frac{\pi}{2}\right) < C'_2 < 2^{\kappa'_2} (1 - 2^{\kappa'_2-M-1}) \mathfrak{u}\left(\frac{\pi}{2}\right)$$

Equation (3) applied to the definition of C'_2 yields, for the unit of the last place:

$$\begin{aligned} \mathfrak{u}\left(\frac{\pi}{2} - C_2 - C'_2\right) &= 2^{\kappa'_2-M} \mathfrak{u}\left(\frac{\pi}{2} - C_2\right) \\ &= 2^{\kappa'_2+\kappa'_2-2M} \mathfrak{u}\left(\frac{\pi}{2}\right) \end{aligned}$$

Noting that the absolute error on the rounding that appears in the definition of δC_2 is bounded by $\frac{1}{2} \mathfrak{u}\left(\frac{\pi}{2} - C_2 - C'_2\right)$, we obtain the absolute error on the three-term approximation:

$$\left| \frac{\pi}{2} - C_2 - C'_2 - \delta C_2 \right| \leq \frac{1}{2} \mathfrak{u}\left(\frac{\pi}{2} - C_2 - C'_2\right) = 2^{\kappa'_2+\kappa'_2-2M-1} \mathfrak{u}\left(\frac{\pi}{2}\right) \quad (7)$$

and the following upper bound for δC_2 :

$$|\delta C_2| < 2^{\kappa'_2+\kappa'_2-M} (1 + 2^{-M-1}) \mathfrak{u}\left(\frac{\pi}{2}\right) \quad (8)$$

This scheme gives a representation with a significand that has effectively $3M - \kappa'_2 - \kappa'_2$ bits and is such that multiplying C_2 and C'_2 by an integer less than or equal to 2^{κ_2} is exact.

Argument Reduction

Given an argument x , the purpose of argument reduction is to compute a pair of floating-point numbers $(\tilde{x}, \delta\tilde{x})$ such that:

$$\begin{cases} \tilde{x} + \delta\tilde{x} \cong x \pmod{\frac{\pi}{2}} \\ \tilde{x} \text{ is approximately in } \left[-\frac{\pi}{4}, \frac{\pi}{4}\right] \\ |\delta\tilde{x}| \leq \frac{1}{2} u(\tilde{x}) \end{cases}$$

Argument Reduction for Small Angles

If $|x| < \left\lfloor \frac{\pi}{4} \right\rfloor$ then $\tilde{x} = x$ and $\delta\tilde{x} = 0$.

Argument Reduction Using the Two-Term Approximation

If $|x| \leq 2^{\kappa_1} \left\lfloor \frac{\pi}{2} \right\rfloor$ we compute:

$$\begin{cases} n &= \left\lfloor \left\lfloor x \left\lfloor \frac{2}{\pi} \right\rfloor \right\rfloor \right\rfloor \\ y &= x - n C_1 \\ \delta y &= \left\lfloor n \delta C_1 \right\rfloor \\ (\tilde{x}, \delta\tilde{x}) &= \text{TwoDifference}(y, \delta y) \end{cases}$$

The first thing to note is that $|n| \leq 2^{\kappa_1}$. We have:

$$|x| \leq 2^{\kappa_1} \left\lfloor \frac{\pi}{2} \right\rfloor = 2^{\kappa_1} \frac{\pi}{2} (1 + \delta_1)$$

and:

$$\left\lfloor x \left\lfloor \frac{2}{\pi} \right\rfloor \right\rfloor = x \frac{2}{\pi} (1 + \delta_2)(1 + \delta_3) \quad (9)$$

from which we deduce the upper bound:

$$\begin{aligned} |n| &\leq \left\lfloor 2^{\kappa_1} \frac{\pi}{2} (1 + \delta_1) \frac{2}{\pi} (1 + \delta_2)(1 + \delta_3) \right\rfloor \\ &\leq \left\lceil 2^{\kappa_1} (1 + \gamma_3) \right\rceil \end{aligned}$$

If $2^{\kappa_1} \gamma_3$ is small enough (less than 1/2), the rounding cannot cause n to exceed 2^{κ_1} . In practice we choose a relatively small value for κ_1 , so this condition is met.

Now if x is close to an odd multiple of $\frac{\pi}{4}$ it is possible for misrounding to happen. There are two kinds of misrounding, with different bounds.

A misrounding of the first kind occurs if, assuming $n > 0$:

$$x < \left(n - \frac{1}{2}\right) \frac{\pi}{2} \quad \text{and} \quad \left\lfloor x \left\lfloor \frac{2}{\pi} \right\rfloor \right\rfloor > n - \frac{1}{2}$$

Using equation (9) we find that this misrounding is only possible if:

$$x > \frac{\pi}{2} \left(n - \frac{1}{2}\right) \frac{1}{(1 + \delta_2)(1 + \delta_3)} \geq \frac{\pi}{2} \left(n - \frac{1}{2}\right) \frac{1}{1 + \gamma_2}$$

In which case the computation of n results in:

$$n \frac{\pi}{2} - x < \frac{\pi}{4} \left(1 + \frac{\gamma_2}{1 + \gamma_2} (2n - 1)\right)$$

In this case, misrounding causes the absolute value of the reduced angle to increase and it may thus exceed $\frac{\pi}{4}$ by as much as:

$$\frac{\pi}{4} \frac{\gamma_2}{1 + \gamma_2} (2^{\kappa_1+1} - 1) \quad (10)$$

TODO(phl): In this section and the next, document the “better” bounds computed with *Mathematica*.

The accurate tables must be constructed so that the last interval covers angles misrounded in that manner².

A misrounding of the second kind occurs if, assuming $n \geq 0$:

$$x > \left(n + \frac{1}{2}\right) \frac{\pi}{2} \quad \text{and} \quad \left\lfloor \left\lfloor x \left\lfloor \frac{2}{\pi} \right\rfloor \right\rfloor \right\rfloor < n + \frac{1}{2}$$

A derivation similar to the one above gives the following condition for this misrounding to be possible. Using equation (9):

$$x < \frac{\pi}{2} \left(n + \frac{1}{2}\right) \frac{1}{(1 + \delta_2)(1 + \delta_3)} \leq \frac{\pi}{2} \left(n + \frac{1}{2}\right) (1 + \gamma_2)$$

we derive the bound:

$$x - n \frac{\pi}{2} < \frac{\pi}{4} (1 + \gamma_2 (2n + 1))$$

In this case, misrounding causes the absolute value of the reduced angle to decrease by as much as:

$$\frac{\pi}{4} \gamma_2 (2^{\kappa_1+1} + 1)$$

This is however not a concern for the accurate tables and the polynomials as it cannot cause the reduced angle to become negative.

Using the bound on $|n|$ and the fact that C_1 has κ_1 trailing zeroes, we see that the product $n C_1$ is exact. The subtraction $x - n C_1$ is exact by Sterbenz's Lemma. Finally, the last step performs an exact addition³ using algorithm 4 of [HLBo8].

To compute the overall error on argument reduction⁴, first remember that, from equation (4), we have:

$$C_1 + \delta C_1 = \frac{\pi}{2} + \zeta \quad \text{with} \quad |\zeta| \leq 2^{\kappa'_1 - M - 1} u\left(\frac{\pi}{2}\right)$$

The error computation proceeds as follows:

$$\begin{aligned} y - \delta y &= x - n C_1 - n \delta C_1 (1 + \delta_4) \\ &= x - n(C_1 + \delta C_1) - n \delta C_1 \delta_4 \\ &= x - n \frac{\pi}{2} - n(\zeta + \delta C_1 \delta_4) \end{aligned}$$

from which we deduce an upper bound on the absolute error of the reduction:

$$\begin{aligned} \left| y - \delta y - \left(x - n \frac{\pi}{2}\right) \right| &\leq 2^{\kappa_1} 2^{\kappa'_1} (2^{-M-1} + 2^{-M} + 2^{-2M-1}) u\left(\frac{\pi}{2}\right) \\ &= 2^{\kappa_1 + \kappa'_1 - M} \left(\frac{3}{2} + 2^{-M-1}\right) u\left(\frac{\pi}{2}\right) \\ &< 2^{\kappa_1 + \kappa'_1 - M + 1} u\left(\frac{\pi}{2}\right) \end{aligned}$$

where we have used the upper bound for δC_1 given by equation (5).

²In practice this is not a stringent constraint because the distance between accurate table entries is much larger than the quantity given by (10).

³The more efficient QuickTwoDifference is not usable here. First, note that $|y|$ is equal to $u(x)$ if we take x to be the successor or the predecessor of $n C_1$ for any n . Ignoring rounding errors we have:

$$|\delta y| \geq n 2^{\kappa'_1 - 1} u\left(\frac{\pi}{2}\right) \geq 2^{\kappa'_1 + M - 2} u\left(\frac{\pi}{2}\right) u(n)$$

where we used the bound given by equation (1). Now the computation of n can result in a value that is either in the same binade or in the binade below that of x . Therefore $u(n) \geq \frac{1}{2} u(x)$ and the above inequality becomes:

$$|\delta y| \geq 2^{\kappa'_1 + M - 3} u\left(\frac{\pi}{2}\right) u(x)$$

plugging $u\left(\frac{\pi}{2}\right) = 2^{1-M}$ we find:

$$|\delta y| \geq 2^{\kappa'_1 - 2} u(x)$$

Therefore, as long as $\kappa'_1 > 2$, there exist arguments x for which $|\delta y| > |y|$.

⁴Note that this error analysis is correct even in the face of misrounding.

The exact TwoDifference yields a pair such that $|\delta\tilde{x}| \leq \frac{u(\tilde{x})}{2}$ which implies the bounds:

$$\begin{aligned} \text{Relative:} \quad & |\delta\tilde{x}| \leq \frac{|\tilde{x}|}{2^M} \\ \text{Absolute:} \quad & |\delta\tilde{x}| \leq \frac{u\left(\frac{\pi}{4}\right)}{2} = 2^{-M-1} \end{aligned}$$

Furthermore, misrounding of the first kind and the above error on the reduction may combine to cause $|\tilde{x}|$ to move above $\frac{\pi}{4}$ by as much as:

$$\frac{\pi}{4} \frac{\gamma_2}{1 + \gamma_2} (2^{\kappa_1+1} - 1) + 2^{\kappa_1+\kappa'_1-M+1} u\left(\frac{\pi}{2}\right)$$

This must be taken into account when building the polynomials.

In the computation of the trigonometric functions, we need $\tilde{x} + \delta\tilde{x}$ to provide enough accuracy that the final result is correctly rounded most of the time. The above error bound shows that, if \tilde{x} is very small (i.e., if x is very close to a multiple of $\frac{\pi}{2}$), the two-term approximation may not provide enough correct bits. Formally, say that we want to have $M + \kappa_3$ correct bits in the mantissa of $\tilde{x} + \delta\tilde{x}$. The error must be less than $2^{-\kappa_3}$ half-units of the last place of the result:

$$2^{\kappa_1+\kappa'_1-M+1} u\left(\frac{\pi}{2}\right) \leq 2^{-\kappa_3-1} u(\tilde{x}) \leq 2^{-\kappa_3-M} |\tilde{x}| \quad (11)$$

which leads to the following condition on the reduced angle:

$$|\tilde{x}| \geq 2^{\kappa_1+\kappa'_1+\kappa_3+1} u\left(\frac{\pi}{2}\right) = 2^{\kappa_1+\kappa'_1+\kappa_3-M+2}$$

The rest of the implementation assumes that $\kappa_3 = 18$ to achieve correct rounding with high probability. If we choose $\kappa_1 = 8$ we find that $\kappa'_1 = 5$ (because there are three consecutive zeroes at this location in the significand of $\frac{\pi}{2}$) and the desired accuracy is obtained as long as $|\tilde{x}| \geq 2^{-20} \simeq 9.5 \times 10^{-7}$.

Argument Reduction Using the Three-Term Approximation

If $|x| \leq 2^{\kappa_2} \left\lceil \frac{\pi}{2} \right\rceil$ we compute:

$$\begin{cases} n &= \left\lceil \left\lceil x \left\lceil \frac{2}{\pi} \right\rceil \right\rceil \right\rceil \\ y &= x - n C_2 \\ y' &= n C'_2 \\ \delta y &= \llbracket n \delta C_2 \rrbracket \\ (z, \delta z) &= \text{QuickTwoSum}(y', \delta y) \\ (\tilde{x}, \delta\tilde{x}) &= \text{LongSub}(y, (z, \delta z)) \end{cases}$$

The products $n C_2$ and $n C'_2$ are exact thanks to the κ_2 trailing zeroes of C_2 and C'_2 . The subtraction $x - n C_2$ is exact by Sterbenz's Lemma. QuickTwoSum performs an exact addition using algorithm 3 of [HLBo8]; it is usable in this case because clearly $|\delta y| < |y'|$. LongSub is the obvious adaptation of the algorithm LongAdd presented in section 5 of [Lin81], which implements precise (but not exact) double-precision arithmetic.

It is straightforward to show, like we did in the preceding section, that:

$$|n| \leq \lceil 2^{\kappa_2} (1 + \gamma_3) \rceil$$

and therefore that $|n| \leq 2^{\kappa_2}$ as long as $2^{\kappa_2} \gamma_3 < 1/2$. Similarly, the misrounding bound (10) is applicable with κ_2 replacing κ_1 .

To compute the overall error on argument reduction, first remember that, from equation (7), we have:

$$C_2 + C'_2 + \delta C_2 = \frac{\pi}{2} + \zeta_1 \quad \text{with} \quad |\zeta_1| \leq 2^{\kappa'_2 + \kappa''_2 - 2M - 1} u\left(\frac{\pi}{2}\right)$$

Let ζ_2 be the relative error introduced by LongAdd. Table 1 of [Lin81] indicates that $|\zeta_2| < 2^{2-2M}$. The error computation proceeds as follows:

$$\begin{aligned} y - y' - \delta y &= (x - n C_2 - n C'_2 - n \delta C_2 (1 + \delta_4))(1 + \zeta_2) \\ &= \left(x - n \frac{\pi}{2} - n(\zeta_1 + \delta C_2 \delta_4)\right)(1 + \zeta_2) \\ &= x - n \frac{\pi}{2} - n(\zeta_1 + \delta C_2 \delta_4)(1 + \zeta_2) + \left(x - n \frac{\pi}{2}\right) \zeta_2 \end{aligned}$$

from which we deduce an upper bound on the absolute error of the reduction, noting that $|x - n \frac{\pi}{2}| \leq \frac{\pi}{4}(1 + \gamma_2(2^{\kappa_2+1} + 1))$ as per (10):

$$\begin{aligned} &\left|y - y' - \delta y - \left(x - n \frac{\pi}{2}\right)\right| \\ &\leq 2^{\kappa_2 + \kappa'_2 + \kappa''_2} (2^{-2M-1} + 2^{-2M} + 2^{-3M-1})(1 + 2^{2-2M}) u\left(\frac{\pi}{2}\right) + 2^{2-2M} \frac{\pi}{4} (1 + \gamma_2(2^{\kappa_2+1} + 1)) \\ &< 2^{\kappa_2 + \kappa'_2 + \kappa''_2 - 2M} \left(\frac{3}{2} + 2^{-M-1}\right) (1 + 2^{2-2M}) u\left(\frac{\pi}{2}\right) + 2^{-2M} \pi \left(1 + 3 \times 2^{\kappa_2} u\left(\frac{\pi}{2}\right)\right) \\ &< 2^{\kappa_2 - 2M} (2^{\kappa'_2 + \kappa''_2 + 1} + 3) u\left(\frac{\pi}{2}\right) + 2^{-2M} \pi \end{aligned}$$

where the second inequality uses $\gamma_2(2^{\kappa_2+1} + 1) < 3u(2^{\kappa_2+1})$.

A sufficient condition for the reduction to guarantee κ_3 extra bits of accuracy is for this error to be less than $2^{-\kappa_3-1} u(\tilde{x})$ which itself is less than $2^{-\kappa_3-M} |\tilde{x}|$. Therefore we want:

$$\begin{aligned} |\tilde{x}| &\geq 2^{\kappa_3-M} \left(2^{\kappa_2} (2^{\kappa'_2 + \kappa''_2 + 1} + 3) u\left(\frac{\pi}{2}\right) + \pi\right) \\ &= 2^{\kappa_3-M} (2^{\kappa_2-M+1} (2^{\kappa'_2 + \kappa''_2 + 1} + 3) + \pi) \end{aligned}$$

and it is therefore sufficient to have:

$$|\tilde{x}| \geq 2^{\kappa_3-M} (2^{\kappa_2 + \kappa'_2 + \kappa''_2 - M + 2} + 4)$$

If we choose $\kappa_3 = 18$ as above, and $\kappa_2 = 18$ we find that $\kappa'_2 = 14$ and $\kappa''_2 = 15$. Therefore, the desired accuracy is obtained as long as $|\tilde{x}| \geq 65 \times 2^{-39} \simeq 1.2 \times 10^{-10}$.

Fallback

If any of the conditions above is not met, we fall back on the CORE-MATH implementation.

Accurate Tables and Their Generation

Polynomial Approximations

The *Mathematica* function `GeneralMiniMaxApproximation` produces a minimax polynomial $p(h(x))$ approximating a function $f(x)$ by minimizing the quantity $\frac{f(x) - p(h(x))}{g(x)}$. By choosing $g(x)$ appropriately, we can obtain an approximation that minimizes either the absolute or relative error on the result.

TODO(phl): Document the tables and all the tricks that went into their generation.

Sin Near Zero

For the sin function near zero the accurate tables method is not usable because the correction term is not small compared to the tabulated value of the function (which would be zero)⁵. Instead we use a polynomial approximation that minimizes the relative error on the result. Since $\sin x$ is an even function and since its dominant term is x , we are looking for an approximation having the form:

$$\sin x \simeq x + x^3 p_{s0}(x^2)$$

over the interval $[0, \Delta]$, where Δ is chosen so that $\Delta^2 \ll 1$. By leaving the sum unevaluated, the two terms x and $x^3 p_{s0}(x)$ may be used to perform the rounding test (see below).

TODO(phl): Document the rounding test and xref here.

We are therefore calling GeneralMiniMaxApproximation with:

$$\begin{cases} h(x) &:= x^2 \\ f(x) &:= \frac{\sin x - x}{x^3} \\ g(x) &:= \frac{\sin x}{x^3} \end{cases}$$

which results in a polynomial p_{s0} which minimizes the relative error $\frac{\sin x}{x}$; the degree of p_{s0} is chosen so that the error is less than $u(\Delta^2)$.

In practice we choose $\Delta = 2^{-10}$, and compute a degree-1 polynomial which induces a relative error smaller than $2^{-75.538}$ (before rounding the coefficients to machine numbers).

Around Table Entries

Let (x_k, s_k, c_k) be an accurate table entry. x_k is close to $2k\Delta$ and the accurate table interval containing x_k is $[(2k-1)\Delta, (2k+1)\Delta]$. The implementation of sin and cos starts by choosing, from the argument x (which in the case of sin, is not close to zero), a k such that $x \in [(2k-1)\Delta, (2k+1)\Delta]$ and computing $h = x - x_k$. We are therefore looking for approximations of the form:

$$\begin{aligned} \sin h &\simeq h + h^3 p_s(h^2) \\ \cos h &\simeq 1 + h^2 p_c(h^2) \end{aligned}$$

which must cover the interval $[0, h_{max}]$ with:

$$h_{max} := \max_k \{x_k - (2k-1)\Delta, (2k+1)\Delta - x_k\}$$

It is difficult to find a theoretical justification for the error function used in these approximations, as the polynomials p_s and p_c contribute in complicated ways to the final error on sin and cos. In practice it appears that an absolute error bound results in a slightly lower final relative error, so for sin we call GeneralMiniMaxApproximation with:

$$\begin{cases} h(x) &:= x^2 \\ f(x) &:= \frac{\sin x - x}{x^3} \\ g(x) &:= \frac{1}{x^3} \end{cases}$$

and for cos:

$$\begin{cases} h(x) &:= x^2 \\ f(x) &:= \frac{\cos x - 1}{x^2} \\ g(x) &:= \frac{1}{x^2} \end{cases}$$

⁵It would be possible to have one set of tables per binade with progressively denser intervals, but that would have a terrible performance as the tables would end up being very large.

TODO(phl): This cannot be a
TODO(phl): and in these intervals!

In practice we chose above $\Delta = 2^{-10}$ and the accurate tables construction yielded $h_{max} < \Delta + 2^{-17.834}$. The minimax computation results in a 1-degree polynomial p_s with an absolute error smaller than $2^{-85.746}$ and a 1-degree polynomial p_c with an absolute error smaller than $2^{-72.693}$ (before rounding the coefficients to machine numbers).

TODO(phl): xref

Core Implementation

This section documents the core of the algorithms used to compute sin and cos. They take as input the result of argument reduction, $(\tilde{x}, \delta\tilde{x})$ and produce a pair $(y, \delta y)$ which is passed to the rounding test described in [Mul+10, p. 397] to decide whether we need to fall back to the CORE-MATH implementation. Most of the time (with a probability close to $1 - 2^{-18}$) y is the correctly-rounded result. We assume that \tilde{x} is positive ($\delta\tilde{x}$ may be positive or negative).

We use $\llbracket expr \rrbracket \dots$ to denote evaluation where appropriate rounding happens on each literal or operation of the expression $expr$. This notation is typically used for error intervals computed using the functions `IEEEEvaluateWithRelativeError` and `IEEEEvaluateWithAbsoluteError` in file `mathematica/ieee754_floating_point_evaluation.wl`. In particular, these functions take into account that the evaluation of the polynomials p_{s0} , p_s , and p_c has two sources of errors: the rounding of the coefficients to machine numbers, and the error due to the floating-point operations; this may leading to asymmetrical error intervals.

We assume the availability of an FMA operation, and the rounding direction *roundTiesToEven*.

TODO(phl): Try to tune the high-degree coefficient after rounding the low-degree one.

Reduced Angle

Argument reduction took the input angle x and produced a pair $(\tilde{x}, \delta\tilde{x})$ approximating the angle reduced modulo $\frac{\pi}{4}$. That approximation is correct to $M + \kappa_3$ bits. We therefore have, from equation (11):

$$x \equiv \tilde{x} + \delta\tilde{x} + \zeta_0 \tilde{x} \pmod{\frac{\pi}{4}} \quad |\zeta_0| \leq 2^{-\kappa_3 - M}$$

For simplicity of the error analysis we can assume $0 \leq x \leq \frac{\pi}{4}$; even though we do not do argument reduction when x is less than $\frac{\pi}{4}$, the error analysis would work exactly the same if x was simply provided as a high-accuracy value (e.g., a double-double) and reduced using the techniques above.

Sin Near Zero

If $|\tilde{x}| \leq \Delta = 2^{-10}$ the steps of the computation are as follows:

$$\begin{cases} t_1 &= \llbracket p_{s0}(\tilde{x}^2) \rrbracket \dots \\ t_2 &= \llbracket \tilde{x}^3 \rrbracket \dots \\ t_3 &= \llbracket t_1 t_2 + \delta\tilde{x} \rrbracket \\ y &= \tilde{x} \\ \delta y &= t_3 \end{cases}$$

First note that, in the computation, the terms $\tilde{x}^n \delta\tilde{x}$ for $n > 0$ are irrelevant. The reason is that $|\tilde{x} \delta\tilde{x}| < \Delta^2/2^M = 2^{-M-20}$. But we chose $\kappa_3 = 18$ so we are already willing to tolerate errors significantly larger than $\tilde{x} \delta\tilde{x}$. This is why we do not involve $\delta\tilde{x}$ in steps t_1 and t_2 above: specifically the most significant term in $t_1 t_2$ would be $\mathcal{O}(\tilde{x}^2 \delta\tilde{x})$ which is negligible compared to \tilde{x} , the main component of the result.

The relative error on the minimax polynomial may be rewritten as:

$$x^3 p_{s0}(x^2) = (1 + \zeta_1) \sin x - x \quad |\zeta_1| < 2^{-75.538}$$

for $x \in [0, x_{max}]$.

TODO(phl): Document x_{max} .

The errors committed at each step are as follows:

$$\begin{cases} t_1 = p_{s0}(\tilde{x}^2)(1 + \zeta_2) & \zeta_2 \in]-2^{-52.415}, 2^{-53.999}[\\ t_2 = \tilde{x}^3(1 + \zeta_3) & \zeta_3 \in]-2^{-52.000}, 2^{-51.999}[\\ t_3 = (t_1 t_2 + \delta \tilde{x})(1 + \delta_1) \\ \quad = ((1 + \zeta_2)(1 + \zeta_3)((1 + \zeta_1) \sin \tilde{x} - \tilde{x}) + \delta \tilde{x})(1 + \delta_1) \end{cases}$$

The entire computation is:

$$r(\tilde{x}, \delta \tilde{x}) := \frac{y + \delta y}{\sin(\tilde{x} + \delta \tilde{x} + \zeta_0 \tilde{x})} - 1$$

The function r can be computed using *Mathematica* interval arithmetic over the triangular domain $|\tilde{x}| \leq 2^{-10}$, $|\delta \tilde{x}| \leq 2^{-M}|\tilde{x}|$. Plotting shows that it is monotonic and its extrema are therefore reached at the corners of the domain. In practice we find that $|r(\tilde{x}, \delta \tilde{x})| < 2^{-70.561}$ (as expected, this is a bit worse than the error originating from the angle reduction).

TODO(phl): $\kappa_3 = 19$ would gain
o.7 bits. Worthwhile?

Sin Around Table Entries

If $x \in [(2k - 1)\Delta, (2k + 1)\Delta]$ and $h = x - x_k$ we first compute exactly $hc_k + s_k$ using an FMA as explained in section 2.1 of [SZo5]:

$$z + \delta z = hc_k + s_k$$

TODO(phl): Align the notation in
the notebook.

The steps of the computation are then as follows:

$$\begin{cases} t_1 &= \llbracket h + 2\delta x \rrbracket \\ t_2 &= \llbracket h t_1 \rrbracket \\ t_3 &= \llbracket t_2 h \rrbracket \\ t_4 &= \llbracket p_s(t_2) \rrbracket \\ t_5 &= \llbracket t_3 t_4 \rrbracket \\ t_6 &= \llbracket p_c(t_2) \rrbracket \\ t_7 &= \llbracket t_2 t_6 \rrbracket \\ t_8 &= \llbracket s_k t_7 \rrbracket \\ t_9 &= \llbracket c_k t_5 + t_8 \rrbracket \\ t_{10} &= \llbracket t_9 + \delta z \rrbracket \\ y &= z \\ \delta y &= t_{10} \end{cases}$$

The errors entailed at each step are as follows:

$$\begin{cases} t_1 &= (h + 2\delta x)(1 + \delta_1) \\ t_2 &= (t_1 h)(1 + \delta_2) \\ t_3 &= (t_2 h)(1 + \delta_3) \\ t_4 &= p_s(t_2)(1 + \zeta_1) \end{cases}$$

Expression	Bounds on the absolute error
$t_1 = \llbracket h \llbracket h + 2\delta x \rrbracket \rrbracket$	$] -2^{-71.996}, 2^{-71.996}[$
$t_2 = \llbracket t_1 h \rrbracket$	$] -2^{-81.408}, 2^{-81.408}[$
$t_3 = \llbracket t_1 p_c(t_1) \rrbracket \dots$	$] -2^{-71.995}, 2^{-71.995}[$
$t_4 = \llbracket t_2 p_s(t_1) \rrbracket \dots$	$] -2^{-83.254}, 2^{-83.254}[$

Cos Around Table Entries

References

- [GB91] S. Gal and B. Bachelis. “An Accurate Elementary Mathematical Library for the IEEE Floating Point Standard”. In: *ACM Transactions on Mathematical Software* 17.1 (Mar. 1991), pp. 26–45.

- [Higo2] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, 2002.
- [HLBo8] Y. Hida, X. S. Li, and D. H. Bailey. “Library for Double-Double and Quad-Double Arithmetic”. Preprint at <https://www.davidhbailey.com/dhbpapers/qd.pdf>. May 8, 2008.
- [Lin81] S. Linnainmaa. “Software for Doubled-Precision Floating-Point Computations”. In: *ACM Transactions on Mathematical Software* 7.3 (Sept. 1981), pp. 272–283.
DOI: 10.1145/355958.355960.
- [Mul+10] J.-M. Muller, N. Brisebarre, F. De Dinechin, C.-P. Jeannerod, V. Lefèvre, G. Melquiond, N. Revol, D. Stehlé, and S. Torres. *Handbook of Floating-Point Arithmetic*. Birkhäuser, 2010.
- [SZ05] D. Stehlé and P. Zimmermann. “Gal’s accurate tables method revisited”. In: *17th IEEE Symposium on Computer Arithmetic (ARITH’05)* (Cape Cod, MA, USA, June 27–29, 2005). Ed. by P. Montuschi and E. Schwarz. IEEE Computer Society, June 2005, pp. 257–264.
DOI: 10.1109/ARITH.2005.24.
- [SZG22] A. Sibidanov, P. Zimmermann, and S. Glondou. “The CORE-MATH Project”. In: *2022 IEEE 29th Symposium on Computer Arithmetic (ARITH)*. IEEE, Sept. 2022, pp. 26–34.
DOI: 10.1109/ARITH54963.2022.00014.
eprint: <https://inria.hal.science/hal-03721525v3/file/core-math-final.pdf>.
- [ZSG+24] P. Zimmermann, A. Sibidanov, S. Glondou, et al. *The CORE-MATH Project*. Software. Apr. 2024.