

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.

In this document we assume a base-2 floating-point number system with M significant 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 distance between 1 and the next larger floating-point number is:

$$\epsilon_M := 2^{1-M}$$

and the distance between 1 and the next smaller floating-point number is $\frac{\epsilon_M}{2}$. The *unit of the last place* of x is defined as:

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

In particular, $u(1) = \epsilon_M$ and:

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

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): unless otherwise indicated, δ_i is a roundoff factor such that $\delta_i < u = \epsilon_M/2 = 2^{-M}$ (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.

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

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

The definition of the floor function implies:

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

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

$$0 \leq z - \text{Tr}(\kappa, z) < 2^{\kappa'} u(z) \quad \text{with} \quad \kappa' = \kappa - k \quad (1)$$

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

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

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

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

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 (1) becomes:

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

where $\kappa'_1 \leq \kappa_1$ accounts for any leading zeroes in the bits of $\frac{\pi}{2}$ that are being truncated. Equation (2) becomes:

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

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 the absolute error on the two-term approximation:

$$\left| \frac{\pi}{2} - C_1 - \delta C_1 \right| \leq \frac{1}{2} u\left(\frac{\pi}{2} - C_1\right) = 2^{\kappa'_1 - M - 1} u\left(\frac{\pi}{2}\right) \quad (3)$$

and the following upper bound for δC_1 :

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

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:

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

Applying equation (2) to the definition of C_2 yields:

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

where $\kappa'_2 \leq \kappa_2$ accounts for any leading zeroes in the bits of $\frac{\pi}{2}$ that are being truncated. Similarly, applying equation (2) to the definition of C'_2 yields:

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

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 .

In other words, we have 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 $(\hat{x}, \delta\hat{x})$ such that:

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

Argument Reduction for Small Angles

If $|x| < \left\llbracket \frac{\pi}{4} \right\rrbracket$ then $\hat{x} = x$ and $\delta\hat{x} = 0$.

Argument Reduction Using the Two-Term Approximation

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

$$\begin{cases} n &= \left\llbracket \left\llbracket x \left\llbracket \frac{2}{\pi} \right\rrbracket \right\rrbracket \right\rrbracket \\ y &= x - n C_1 \\ \delta y &= \left\llbracket n \delta C_1 \right\rrbracket \\ \hat{x} &= \left\llbracket y - \delta y \right\rrbracket \\ \delta\hat{x} &= (y - \hat{x}) - \delta y \end{cases}$$

Let's first show that $|n| \leq 2^{\kappa_1}$. :

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

As long as $2^{\kappa_1} \gamma_3$ 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.

The product $n C_1$ is exact thanks to the κ_1 trailing zeroes of C_1 . The subtraction $x - n C_1$ is exact by Sterbenz's Lemma. Finally, the last two steps form a compensated summation so that $\hat{x} + \delta\hat{x} = y + \delta y$.

To compute the overall error on argument reduction, first remember that, from equation (3) we have:

$$C_1 + \delta C_1 = \frac{\pi}{2} + \delta_5 \quad \text{with} \quad |\delta_4| \leq 2^{\kappa'_1 - M - 1} u\left(\frac{\pi}{2}\right)$$

The error computation proceeds as follows:

$$\begin{aligned} y + \delta y &= x - n C_1 - n \delta C_1 (1 + \delta_5) \\ &= x - n (C_1 + \delta C_1) - n \delta C_1 \delta_5 \\ &= x - n \frac{\pi}{2} - n (\delta_5 + \delta C_1 \delta_5) \end{aligned}$$

from which we can 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 (4).

If we want $\hat{x} + \delta\hat{x}$ to have κ_3 extra bits of accuracy, we must have:

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

which leads to the following condition on the reduced angle:

$$|\hat{x}| \geq 2^{\kappa_1 + \kappa'_1 + \kappa_3} u\left(\frac{\pi}{2}\right)$$

The rest of the implementation assumes that $\kappa_3 = 18$ to achieve correct rounding most of the time and detect cases of dangerous rounding. 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 $|\hat{x}| \geq 2^{-21} \simeq 4.8 \times 10^{-7}$.

Argument Reduction Using the Three-Term Approximation

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

$$\begin{cases} n &= \left\lceil \left\lceil x \left\lceil \frac{2}{\pi} \right\rceil \right\rceil \right\rceil \\ y &= x - n C_2 \\ y' &= \llbracket y - n C'_2 \rrbracket \\ \delta y &= \llbracket n \delta C_2 \rrbracket \end{cases}$$

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 $2^{\kappa_2} \gamma_3 < 1/2$.

Accurate Tables and Their Generation

Computation of the Functions

Sin

Near Zero

For \hat{x} near zero we evaluate:

$$\widehat{x^2} = \llbracket \hat{x}^2 \rrbracket = \hat{x}^2(1 + \delta_1)$$

$$\widehat{x^3} = \llbracket \hat{x} \widehat{x^2} \rrbracket = \hat{x}^3(1 + \delta_1)(1 + \delta_2)$$

$$\hat{p} = \llbracket a\widehat{x^2} + b \rrbracket = (a\hat{x}^2(1 + \delta_1) + b)(1 + \delta_3)$$

$$\begin{aligned} s(x) &:= \hat{x} + \llbracket \llbracket \widehat{x^3} \hat{p} \rrbracket + \delta \hat{x} \rrbracket \\ &= \hat{x} + (\hat{x}^3(1 + \delta_1)(1 + \delta_2)(a\hat{x}^2(1 + \delta_1) + b)(1 + \delta_3)(1 + \delta_4) + \delta \hat{x})(1 + \delta_5) \\ &= \hat{x} + a\hat{x}^5(1 + \theta_5) + b\hat{x}^5(1 + \theta_4) + \delta \hat{x}(1 + \delta_5) \end{aligned}$$

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.
- [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, 27th–29th June 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.