On Computing Sines and Cosines

John Kormylo

The standard method used to compute sines and cosines are the Taylor's series (polynomial) approximations

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

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} + \cdots + (-1)^n \frac{x^{2n}}{(2n)!} \quad (2)$$

where $(n+1)$ is the number of terms used. The advantage of the Taylor's series approximation is that it can generate an answer to any desired accuracy by adding more terms. It is not, however, particularly fast.

For Fast Fourier Transforms (FFT) one generally pre-computes an array of sines and cosines, since only certain values are used ($x = 0, 2\pi/N, 4\pi/N, 6\pi/N, \ldots, 2\pi(N-1)/N$ where N is a power of 2). This could be done using either the half-angle formulas

$$\sin(x) = \sqrt{\frac{1-\cos(2x)}{2}} \quad (3)$$

$$\cos(x) = \sqrt{\frac{1+\cos(2x)}{2}} \quad (4)$$

or the polynomial approximations for $x = \pi/8, \pi/16, \ldots, \pi/N$, then using the angle addition formulas

$$\sin(a+b) = \sin(a)\cos(b) + \cos(a)\sin(b) \quad (5)$$

$$\cos(a+b) = \cos(a)\cos(b) - \sin(a)\sin(b) \quad (6)$$

to fill in the array.

Specifically, one would start by filling in the known array values for $x = 0, \pi/4, \pi/2, \ldots, 7\pi/4$, computing $\sin(\pi/8)$ and $\cos(\pi/8)$, then filling in the array values for $x = \pi/8, (\pi/4+\pi/8), (\pi/2+\pi/8), \ldots, (7\pi/4+\pi/8)$. In the next step one would compute $\sin(\pi/16)$ and $\cos(\pi/16)$ and fill in the array values for $x = \pi/16, (\pi/8+\pi/16), (\pi/4+\pi/16), \ldots, (15\pi/8+\pi/16)$. This would continue for a total of $\log_2(N/8)$ steps to generate all N values.

Errors would accumulate at each step, but there are only a few steps.

DSP chips often include a table of sine and cosine value in ROM. For graphic applications it is generally adequate to grab the value for $\sin(x)$ in the table nearest to the desired x value. A linearly interpolated value can be obtained for the cost of a couple of multiplications and additions. The cubic interpolation solution is easy to obtain since the slopes are already known and stored in the table.

$$\frac{d}{dx}\sin(x) = \cos(x)$$

$$\frac{d}{dx}\cos(x) = -\sin(x)$$

Assume a solution of the form

$$\sin(x) \approx f_i(x) = a_i(x-x_i)^3 + b_i(x-x_i)^2 + c_i(x-x_i) + d_i \quad (7)$$

over the domain $x_i \le x \le x_{i+1}$. The coefficients are chosen to satisfy

$$f_i(x_i) = \sin(x_i)$$

$$f_i(x_{i+1}) = \sin(x_{i+1})$$

$$f_i'(x_i) = \cos(x_i)$$

$$f_i'(x_{i+1}) = \cos(x_{i+1})$$

where

$$f_i'(x) = 3a_i(x - x_i)^2 + 2b_i(x - x_i) + c_i$$

and the x_i correspond to the stored values in the table. One can easily show that

$$\begin{aligned} d_i &= \sin(x_i) && (8)\\ c_i &= \cos(x_i) && (9)\\ b_i &= 3\frac{\sin(x_{i+1}) - \sin(x_i)}{(x_{i+1} - x_i)^2} - \frac{\cos(x_{i+1}) + 2\cos(x_i)}{x_{i+1} - x_i} && (10)\\ a_i &= \frac{\cos(x_{i+1}) + \cos(x_i)}{(x_{i+1} - x_i)^2} - 2\frac{\sin(x_{i+1}) - \sin(x_i)}{(x_{i+1} - x_i)^3} && (11) \end{aligned}$$

satisfies the given conditions. It would be a good idea to pre-compute and store the a_i and b_i coefficients.

Repeating this for cosines, we now have

$$\cos(x) \approx g_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i \quad (12)$$

such that

$$\begin{aligned} g_i(x_i) &= \cos(x_i)\\ g_i(x_{i+1}) &= \cos(x_{i+1})\\ g_i'(x_i) &= -\sin(x_i)\\ g_i'(x_{i+1}) &= -\sin(x_{i+1}) \end{aligned}$$

which has the solution

$$\begin{aligned} d_i &= \cos(x_i) && (13)\\ c_i &= -\sin(x_i) && (14)\\ b_i &= 3\frac{\cos(x_{i+1}) - \cos(x_i)}{(x_{i+1} - x_i)^2} + \frac{\sin(x_{i+1}) + 2\sin(x_i)}{x_{i+1} - x_i} && (15)\\ a_i &= -2\frac{\cos(x_{i+1}) - \cos(x_i)}{(x_{i+1} - x_i)^3} - \frac{\sin(x_{i+1}) + \sin(x_i)}{(x_{i+1} - x_i)^2} \quad . && (16) \end{aligned}$$

Each calculation requires three multiplications, three additions and one subtraction (not counting the calculation to determine which table entry to use). The accuracy of any table depends on how closely the x_i are spaced. While cubic interpolation can produce the same degree of accuracy with a greater spacing, it also requires 3 times as much storage for the coefficients. It is neither the fastest nor the most accurate approach, but does possess a certain elegance.