Accuracy of Mathematical Functions in Single, Double, and Quadruple Precision

Paul Zimmermann

December 4, 2020

This document compares the accuracy of several mathematical libraries for the evaluation of mathematical functions, in single, double, and quadruple precision (respectively `binary32`, `binary64`, and `binary128` in the IEEE 754 standard). For single precision, an exhaustive search is possible for univariate functions, thus the given values are upper bounds. For larger precisions or bivariate functions, since an exhaustive search is not possible with academic resources, we use a black-box algorithm that tries to locate the values with the largest error; the given values are only lower bounds, but comparing them can give an idea of the relative accuracy of different libraries. An interesting fact is that, for several functions, different libraries yield the same largest error, for the exact same input value, which probably means they use the same code base.

1 Introduction

In this document we compare the accuracy of the six following mathematical libraries (in the rounding to nearest mode): GNU libc 2.32 [3], the Intel Math Library shipped with the Intel compiler (icc) 19.1.3.304 [4], AMD LibM 3.5 [1], RedHat Newlib 3.3.0 [7], OpenLibm 0.7.0 [8], and Musl 1.2.1 [6].

For each function, assuming y is the value returned by the library, and z is the exact result (as with infinite precision), we denote by e the absolute difference between y and z in terms of units-in-last-place of z. The value z is approximated with the GNU MPFR library [2], using a larger precision. We also denote by E the absolute difference between y and $Z = \mathrm{RN}(z)$, in terms of units-in-last-place of Z, where Z is also obtained with MPFR, which guarantees correct rounding. Thus e is a real, while E is an integer (except in some corner cases). Our definition of ulp (unit-in-last-place) is the following: for $2^{e-1} \leq |x| < e$, and precision p, we define $\mathrm{ulp}(x) = 2^{e-p}$. i.e., the distance between two consecutive p-bit floating-point numbers in the binade $[2^{e-1}, 2^e]$. Some other definitions exist, see [5].

The results for GNU libc, AMD LibM, RedHat Newlib, OpenLibm and Musl were obtained on an Intel Core i5-4590, with GCC 10.2.0 under Debian (note that the GNU libc results might differ sligthly from one `x86_64` processor to another one, due for example to the use of fused-multiply add or not). Those for the Intel Math Library were obtained on an Intel Xeon E5-2680, with icc version 19.1.3.304 (gcc version 9.2.0 compatibility). We used the options `-no-ftz` of icc to disable the default "flush-to-zero" mode and thus to get full IEEE-754 conformance with subnormals numbers. Newlib was configured with default flags (in particular, without use of hardware FMA).

In all tables, values of e are given with 3 decimal digits, rounded up; thus for example $e = 2.17$ for a univariate single-precision function means that the relative error is bounded by $2.17\mathrm{ulp}(z)$

for all `binary32` inputs, and in all other cases it means the largest known error is bounded by 2.17ulp(z), with at least one case giving an error of more than 2.16ulp(z).

2 Single Precision

2.1 Univariate Functions

The IEEE 754 single-precision (`binary32`) format has $2^{32} - 2^{24} = 4278190080$ values, not counting +Inf, -Inf, and NaN. For a function with a single input—i.e., excluding the `pow` function for example—it is possible to check all values by exhaustive search.

Table 1 summarizes the maximal value of e for each function and each library. In detailed tables (Tables 2, 3, 4), we indicate the number of inputs with $E \geq 1$ (thus incorrectly rounded), with $E \geq 2$, and the maximum value of e.

We see that for all libraries, the sqrt function is correctly rounded for all binary32 inputs, as required by IEEE 754. The single-precision cubic root function (cbrt) is also correctly rounded in OpenLibm and Musl, as well as the cosh function in AMD LibM. The Intel Math Library gives in general more accurate results.

The j0, j1, y0, and y1 functions give large errors for all libraries except the Intel Math Library.

For AMD LibM, the maximal error for exp is 1.00 since for $x =$ `-0x6.747688p+4`, it yields 0 instead of the smallest subnormal 2^{-149}, where $\exp(x)$ is slightly smaller than the smallest subnormal. The same issue arises with exp2 and $x =$ `-0x9.50001p+4` and with exp10 and $x =$ `-0x2.cda7d4p+4`.

Notes about RedHat Newlib: for $x = -0$, and $-2^{-128} \leq x < 0$, tgamma returns ∞ instead of $-\infty$, this case was not taken into account in the maximal error. Still for Newlib, we used the `lgammaf_r` function, since we were unable to compile the `lgammaf` function (with lgamma Newlib says `undefined reference to '_impure_ptr'`).

The notation NA means "Not Available" (exp10 is not available in OpenLibm).

2.2 Bivariate Functions

For bivariate functions, it is not possible to perform an exhaustive search with academic resources, since there are up to 2^{64} possible pairs of inputs. For example, for the power function x^y, there are about 2^{61} input pairs $x, y > 0$ that do not yield underflow nor overflow. We thus used the algorithm described in §3.1 to obtain the values at the end Tables 2, 3, and 4, which are only lower bounds for the maximal error.

3 Double Precision

For double precision it is not possible to perfom an exhaustive search with academic resources. We thus designed a black-box algorithm that tries to find large errors. (We did not want to analyze the code of each library, since this approach would need more human work, and requires to start again from scratch for each new version of the library.) Therefore, the values in the double-precision tables are not upper bounds, only lower bounds.

library	GNU libc	Intel Math Library	AMD LibM	RedHat Newlib	OpenLibm	Musl
version	2.32	icc 19.1.3.304	3.5	3.3.0	0.7.0	1.2.1
acos	0.899	**0.528**	0.669	0.899	0.918	0.918
acosh	2.01	**0.501**	0.504	2.01	2.01	2.01
asin	0.898	**0.528**	0.861	0.926	0.743	0.743
asinh	1.78	0.527	**0.518**	1.78	1.78	1.78
atan	0.853	0.541	**0.501**	0.853	0.853	0.853
atanh	1.73	0.507	**0.506**	1.73	1.73	1.73
cbrt	0.969	0.520	0.548	3.56	**0.500**	**0.500**
cos	0.561	0.548	0.530	2.91	**0.501**	**0.501**
cosh	1.89	0.506	**0.500**	2.51	1.36	1.03
erf	0.968	**0.507**	0.968	0.968	0.943	0.968
erfc	3.13	**0.502**	3.13	63.9	3.17	3.13
exp	**0.502**	0.506	1.00	0.911	0.911	**0.502**
exp10	**0.502**	0.507	1.00	1.06	NA	3.88
exp2	0.502	0.519	1.00	1.02	**0.501**	0.502
expm1	0.813	0.544	**0.537**	0.813	0.813	0.813
j0	6.18e6	**0.678**	4.77e9	6.18e6	3.66e6	3.66e6
j1	2.25e6	**1.69**	7.15e8	1.68e7	2.25e6	2.25e6
lgamma	6.78	**0.510**	6.78	7.50e6	7.50e6	7.50e6
log	0.818	**0.524**	0.940	0.888	0.888	0.818
log10	2.07	**0.516**	0.626	2.10	0.832	0.832
log1p	1.30	0.525	**0.504**	1.30	0.839	0.835
log2	0.752	**0.508**	0.586	1.65	0.865	0.752
sin	0.561	0.546	0.530	1.37	**0.501**	**0.501**
sinh	1.89	0.538	**0.501**	2.51	1.83	1.83
sqrt	**0.500**	**0.500**	**0.500**	**0.500**	**0.500**	**0.500**
tan	1.48	0.520	**0.509**	3.48	0.800	0.800
tanh	2.19	**0.514**	1.27	2.19	2.19	2.19
tgamma	7.91	0.510	7.91	239.	**0.501**	**0.501**
y0	4.86e6	**3.40**	1.52e10	4.84e6	4.84e6	4.84e6
y1	6.18e6	**2.07**	4.65e8	6.18e6	4.17e6	3.66e6
atan2	1.52	**0.550**	0.584	1.52	1.55	1.55
hypot	**0.500**	**0.500**	**0.500**	1.21	1.21	0.927
pow	0.817	**0.515**	1.18	169	1.91	0.817

Table 1: Single precision: maximal value of e (for univariate functions), and largest known value of e (for bivariate functions atan2, hypot, pow).

	GNU libc 2.32			icc 19.1.3.304		
function	$E \geq 1$	$E \geq 2$	max e	$E \geq 1$	$E \geq 2$	max e
acos	5422146	0	0.899	66001	0	0.528
acosh	243413455	2698	2.01	283	0	0.501
asin	4581700	0	0.898	470024	0	0.528
asinh	619608176	2748	1.78	963558	0	0.527
atan	21089464	0	0.853	505178	0	0.541
atanh	52062348	5790	1.73	240154	0	0.507
cbrt	453492162	0	0.969	15275450	0	0.520
cos	28209642	0	0.561	15732888	0	0.548
cosh	17868534	3558	1.89	139708	0	0.506
erf	126805016	0	0.968	908674	0	0.507
erfc	20494449	302363	3.13	1761	0	0.502
exp	170648	0	0.502	250299	0	0.506
exp10	169838	0	0.502	386017	0	0.507
exp2	168362	0	0.502	717434	0	0.519
expm1	12920601	0	0.813	539655	0	0.544
j0	1334176546	269351612	6.18e6	11960	0	0.678
j1	1340594104	274741908	2.25e6	8400236	2	1.69
lgamma	500354453	8246657	6.78	100287	0	0.510
log	416908	0	0.818	1060	0	0.524
log10	29787060	62225	2.07	151499	0	0.516
log1p	11534111	0	1.30	254793	0	0.525
log2	313550	0	0.752	276	0	0.508
sin	29362812	0	0.561	12374252	0	0.546
sinh	71328448	34776	1.89	247226	0	0.538
sqrt	0	0	0.500	0	0	0.500
tan	83411250	0	1.48	694770	0	0.520
tanh	118674314	729782	2.19	164068	0	0.514
tgamma	209259574	20924067	7.91	3971282	0	0.510
y0	1304302535	187031173	4.86e6	6785	1	3.40
y1	1199498354	146032505	6.18e6	10384	1	2.07
	x	y	e	x	y	e
atan2	`1.dba63ep-93`	`1.d8166p-91`	1.52	`1.58a7ecp-91`	`1.58a7bep-96`	0.550
hypot	`1.3ac98p+67`	`-1.ba5ec2p+77`	0.500	`1.3ac98p+67`	`-1.ba5ec2p+77`	0.500
pow	`1.025736p+0`	`1.309f94p+13`	0.817	`0x1.000002p+0`	`-0x1.5aa1cp+29`	0.515

Table 2: Single precision: GNU libc and Intel Math Library.

	AMD LibM 3.5			RedHat Newlib 3.3.0		
function	$E \geq 1$	$E \geq 2$	max e	$E \geq 1$	$E \geq 2$	max e
acos	707885	0	0.669	5422146	0	0.899
acosh	21852	0	0.504	244658623	2698	2.01
asin	2454466	0	0.861	2358230	0	0.926
asinh	185582	0	0.518	542122908	2748	1.78
atan	3534	0	0.501	6406812	0	0.853
atanh	50260	0	0.506	52062348	5790	1.73
cbrt	10626352	0	0.548	1799139486	116334632	3.56
cos	2876076	0	0.530	209833072	6	2.91
cosh	0	0	0.500	23905668	7706	2.51
erf	126805016	0	0.968	126741900	0	0.968
erfc	20494449	302363	3.13	21247299	1131209	63.9
exp	114132	0	1.00	17982847	0	0.911
exp10	102461	0	1.00	18423203	0	1.06
exp2	86902	0	1.00	18401203	0	1.02
expm1	102330	0	0.537	12920601	0	0.813
j0	1353797232	310998452	4.77e9	1338235574	279528826	6.18e6
j1	1369557306	337817680	7.15e8	1818091384	1376362116	1.68e7
lgamma	500354453	8246657	6.78	510903809	13277834	7.50e6
log	72371093	0	0.940	13363494	0	0.888
log10	2418509	0	0.626	30061115	91958	2.10
log1p	73898	0	0.504	11534111	0	1.30
log2	179825	0	0.586	602745869	258	1.65
sin	2866930	0	0.530	206155238	0	1.37
sinh	2	0	0.501	74587762	38924	2.51
sqrt	0	0	0.500	0	0	0.500
tan	529444	0	0.509	83455936	32	3.48
tanh	4314486	0	1.27	118674314	729782	2.19
tgamma	209259574	20924067	7.91	2028164923	1833526367	239.
y0	1314115311	207848357	1.52e10	1306144386	191859954	4.84e6
y1	1213975420	177569092	4.65e8	1201178797	153321647	6.18e6
	x	y	e	x	y	e
atan2	`1.fffe24p+59`	`1.000adcp+73`	0.584	`1.dba63ep-93`	`1.d8166p-91`	1.52
hypot	`1.3ac98p+67`	`-1.ba5ec2p+77`	0.500	`1.6a11aap-120`	`1.a9217p-128`	1.21
pow	`1.e00bbap-1`	`1.502508p+10`	1.18	`1.d55902p-1`	`-1.fe037ep+9`	169

Table 3: Single precision: AMD LibM and RedHat Newlib.

	OpenLibm 0.7.0			Musl 1.2.1		
function	$E \geq 1$	$E \geq 2$	max e	$E \geq 1$	$E \geq 2$	max e
acos	5717768	0	0.918	1700216587		0.918
acosh	244658828	2698	2.01	319260148	23345165	2.01
asin	4220748	0	0.743	4220748	0	0.743
asinh	542176908	2748	1.78	642880516	2730	1.78
atan	6483278	0	0.853	1717759310	0	0.853
atanh	52089660	5790	1.73	52062556	5740	1.73
cbrt	0	0	0.500	0	0	0.500
cos	647594	0	0.501	647594	0	0.501
cosh	23865830	0	1.36	16675588	0	1.03
erf	126619324	0	0.943	127569522	0	0.968
erfc	24416748	343931	3.17	19695704	302363	3.13
exp	19194854	0	0.911	170646	0	0.502
exp10	NA	NA	NA	41421106	3446689	3.88
exp2	102250	0	0.501	168362	0	0.502
expm1	12920593	0	0.813	12920592	0	0.813
j0	1332944944	268168176	3.66e6	1422932510	271739106	3.66e6
j1	1339381958	273573380	2.25e6	1320601392	117301706	2.25e6
lgamma	508702215	10980627	7.50e6	504159259	10758067	7.50e6
log	13361747	0	0.888	416908	0	0.818
log10	12305116	0	0.832	12305116	0	0.832
log1p	11588705	0	0.839	11678873	0	0.835
log2	11476491	0	0.865	313550	0	0.752
sin	625106	0	0.501	625106	0	0.501
sinh	72347778	31216	1.83	72812234	31516	1.83
sqrt	0	0	0.500	0	0	0.500
tan	303818252	0	0.800	303818252	0	0.800
tanh	70733480	729768	2.19	112377586	290564	2.19
tgamma	2	0	0.501	3	0	0.501
y0	1303826513	186525437	4.84e6	1309884649	190741595	4.84e6
y1	1198288693	144090005	4.17e6	1171308902	67527225	3.66e6
	x	y	e	x	y	e
atan2	`1.39308ap-74`	`1.33e304p-72`	1.55	`-1.39308ap-74`	`1.33e304p-72`	1.55
hypot	`1.6a11aap-120`	`1.a9217p-128`	1.21	`-1.69d8d4p-127`	`1.d04b8p-132`	0.927
pow	`d.65874p-4`	`4p+0`	1.91	`1.025736p+0`	`1.309f94p+13`	0.817

Table 4: Single precision: OpenLibm and Musl.

3.1 Search Algorithm

The idea of the algorithm is to subdivide recursively the set of values to search for. We describe it for a univariate double precision function, but it works for any IEEE format, as long as there is a corresponding integer type with the same bit-width, and it also works for bivariate functions. Assume $f(x)$ is a univariate double precision function. The number of possible inputs of f is less than 2^{64}, and can thus be mapped to a 64-bit integer. Assume we have a conversion function `to_uint64` from `uint64_t` to `double`. The algorithm takes as input a range $[a, b]$ of `uint64_t` values, and a threshold t. If $b - a \leq t$, it checks exhaustively all double precision values `x=to_uint64(i)` for $a \leq i \leq b$. This means for each x, we compute the ulp-error e between the value $y \approx f(x)$ returned by the corresponding library, and the exact result z (as with infinite precision), as described in §1.

If $b - a > t$, we subdivide the interval $[a, b]$ into two equal intervals, in each interval we generate t random values and compute the corresponding errors. We then recurse in the interval where we found the largest error.

For example with $t = 10^6$, the initial interval has 2^{64} values, thus we compute $f(x)$ on $2t$ random inputs x (t in each sub-range of 2^{63} values), and so on... The recursion stops when the recursive algorithm would perform more function evaluations than trying all the values in the current interval.

The program also keeps track of the worst cases found for each library, and tries those input values for the other libraries. This helps determining the libraries using the same code base.

We have also used the worst cases found by Vincent Lefèvre, publicly available at `https://www.vinc17.net/research/testlibm/`.

3.2 Results

We used a threshold of $t = 10^6$ in most cases, but the program was first run with smaller thresholds, and it was run several times, cycling over all libraries, to detect common large errors.

Table 5 summarizes the maximal errors found, for example the 0.531 entry for acos and icc means that for all inputs tried by the above algorithm, the ulp-error e for the arc-cosine function with the Intel Math Library was bounded by 0.531 ulp. On each line, bold-face entries correspond to the smallest maximal value of e on the inputs tried by the algorithm (which should not be taken as an upper bound, since the search is not exhaustive here). Detailed tables (Tables 6, 7 and 8) give the input values (in hexadecimal) yielding the corresponding ulp-error e, which enables the reader to reproduce our results.

Like for single precision, the Intel Math Library gives the best results in most cases (for 22 of the 30 univariate functions, and for the `hypot` function). The following functions seem to be correctly rounded: the square root function (as required by IEEE 754), and the GNU libc `asin`, `atan`, `tan` and `atan2` functions. Large errors occur for the AMD `acosh`, `atanh` and `log1p` functions, for the `j0`, `j1`, `y0` and `y1` functions for all libraries except the Intel Math Library, for the `lgamma` function from Newlib, OpenLibm and Musl, for the `tgamma` function from Newlib and OpenLibm, and for the AMD power function.

4 Quadruple Precision

Only the GNU libc and the Intel Math Library support quadruple precision, through the `_Float128` type in GNU libc, and `_Quad` in the Intel Math Library (using the option of the Intel C compiler `-Qoption,cpp,--extended_float_types`). The results are summarized in Table 9, and detailed

library	GNU libc	Intel Math Library	AMD LibM	RedHat Newlib	OpenLibm	Musl
version	2.32	icc 19.1.3.304	3.5	3.3.0	0.7.0	1.2.1
acos	**0.501**	0.531	0.935	0.923	0.923	0.923
acosh	2.23	**0.509**	3.35e7	2.23	2.23	2.23
asin	**0.500**	0.531	1.05	0.972	0.972	0.972
asinh	1.92	**0.506**	1.22	1.92	1.92	1.92
atan	**0.500**	0.528	0.863	0.860	0.860	0.860
atanh	1.80	**0.507**	1.67e7	1.80	1.80	1.80
cbrt	3.63	0.523	**0.502**	0.670	0.668	0.668
cos	**0.516**	0.517	0.797	0.878	0.829	0.829
cosh	1.93	**0.516**	1.91	2.67	1.47	1.04
erf	1.41	**0.507**	1.41	1.02	1.02	1.02
erfc	4.82	**0.505**	4.82	3.94	3.94	3.72
exp	**0.511**	0.530	0.756	0.946	0.946	**0.511**
exp10	2.01	**0.536**	0.769	0.892	NA	4.14
exp2	**0.511**	0.535	0.775	0.895	0.751	**0.511**
expm1	0.908	**0.512**	0.722	0.898	0.898	0.898
j0	4.51e14	**0.561**	4.51e14	4.51e14	4.51e14	4.51e14
j1	4.47e14	**0.596**	4.47e14	2.20e15	2.20e15	2.20e15
lgamma	10.6	**0.515**	10.6	2.50e15	2.50e15	2.50e15
log	0.520	**0.518**	0.577	0.944	0.944	0.520
log10	1.62	**0.531**	0.632	2.07	0.805	0.805
log1p	0.892	**0.520**	2.52e8	0.891	0.891	0.892
log2	0.554	**0.504**	0.616	2.05	0.910	0.554
sin	**0.516**	**0.516**	0.799	0.879	0.828	0.828
sinh	1.93	**0.521**	1.50	2.67	1.88	1.88
sqrt	**0.500**	**0.500**	**0.500**	**0.500**	**0.500**	**0.500**
tan	**0.500**	0.547	1.36	1.01	1.01	1.01
tanh	2.22	**0.551**	1.45	2.22	2.22	2.22
tgamma	9.80	**0.516**	9.80	2.25e3	1.03e3	13.9
y0	2.97e15	**1.14**	2.97e15	1.01e15	1.01e15	1.01e15
y1	2.78e15	**1.25**	2.78e15	2.78e15	2.78e15	2.78e15
atan2	**0.500**	0.550	0.750	1.55	1.55	1.55
hypot	0.974	**0.751**	0.942	1.21	1.21	1.04
pow	**0.513**	0.596	700.	0.895	0.896	**0.513**

Table 5: Double precision: Maximal known value of e.

function	GNU libc 2.32 x	max e	icc 19.1.3.304 x	max e
acos	0x1.fffff3634acd6p-1	0.501	0x1.6c32cd815db72p-1	0.531
acosh	0x1.001fb84a1c57dp+0	2.23	0x1.018f48a4ac8c3p+0	0.509
asin	0x1.7137449123ef6p-26	0.500	-0x1.6c03b1cf34b99p-1	0.531
asinh	-0x1.0240f2bdb3f25p-2	1.92	0x1.00129ddbf1615p-4	0.506
atan	0x1.20538781986c3p+51	0.500	-0x1.fff6ad41cf424p-7	0.528
atanh	-0x1.d919b473870d1p-2	1.80	0x1.edf923025f2f7p-9	0.507
cbrt	-0x1.79f03a86293e2p-668	3.63	0x1.f7be4e7cf1d0fp+254	0.523
cos	0x1.61d4bed9732fbp+363	0.516	-0x1.51b29c1621609p+22	0.517
cosh	-0x1.633c654fee2bap+9	1.93	0x1.52ef2077f39f4p+9	0.516
erf	0x1.c3326e3fcf749p-5	1.41	-0x1.0aa9a18e709ffp+2	0.507
erfc	0x1.3ffa8a9835079p+0	4.82	0x1.5d2eff37a006ep-1	0.505
exp	-0x1.29068fb78da57p+9	0.511	0x1.fce665fab54bap+5	0.530
exp10	0x1.33449e2bf973p-2	2.01	0x1.baaf85104c8d9p+2	0.536
exp2	0x1.55ee43d021e9bp+7	0.511	0x1.47f5ffe7ce088p+8	0.535
expm1	0x1.658c2133a3213p-2	0.908	0x1.62e38c83be401p+9	0.512
j0	0x1.33d152e971b4p+1	4.51e14	0x1.5027fc003f7b1p+6	0.561
j1	-0x1.ea75575af6f09p+1	4.47e14	-0x1.616cd076936ecp+7	0.596
lgamma	-0x1.f614148d313dep+1	10.6	-0x1.3f62ca8cf3849p+2	0.515
log	0x1.d411cc69364a8p-1	0.520	0x1.0080154dc05a6p+0	0.518
log10	0x1.de06fd0ac2b0ep-1	1.62	0x1.fedce8f7e0d05p-1	0.531
log1p	-0x1.2c3915f46c8b2p-2	0.892	0x1.00305109e757bp-9	0.520
log2	0x1.0b548b52d2c46p+0	0.554	0x1.fea3e73b57f62p-1	0.504
sin	0x1.7eba58c6403aep+814	0.516	0x1.95ec26bc4cfa9p+535	0.516
sinh	-0x1.633c654fee2bap+9	1.93	-0x1.adc43d7c21246p-2	0.521
sqrt	0x1.fffffffffffp-1	0.500	0x1.fffffffffffp-1	0.500
tan	0x1.50486b2f87014p-5	0.500	0x1.6a1b13c7010fp+519	0.547
tanh	0x1.e0becba6141f5p-3	2.22	0x1.00976168d1006p+0	0.551
tgamma	-0x1.62c4d519e8677p+3	9.80	-0x1.3dff641026917p+6	0.516
y0	0x1.c982eb8d417eap-1	2.97e15	0x1.404ac5c130173p-32	1.14
y1	0x1.193bed4dff243p+1	2.78e15	0x1.c30f8649d928p+0	1.25
atan2	-1.b48c630109d7ep+361 -1.878e5a0eb857dp+307	0.500	1.e4b06dfc32306p-611 1.fc94c7b453f03p-606	0.550
hypot	-1.6a1a3b51f82a7p+879 1.6a3431bdbf85bp+880	0.974	0.a1476350a454ep-1022 -0.4740305adceb5p-1022	0.751
pow	1.f02800d31fc92p-663 1.68c945f8f8263p+0	0.513	1.7efd837c9308bp-1 1.f7c8b2c39ee94p+10	0.596

Table 6: Double precision: GNU libc and Intel Math Library.

function	AMD LibM 3.5 x	max e	RedHat Newlib 3.3.0 x	max e
acos	-0x1.0142eea86c564p-1	0.935	-0x1.00ab8f608b178p-1	0.923
acosh	0x1.40c35b828309cp+0	3.35e7	0x1.001fb84a1c57dp+0	2.23
asin	0x1.017864c40e986p-1	1.05	0x1.014f1c8b2002dp-1	0.972
asinh	0x1.0124ee1fd4b84p+0	1.22	-0x1.0240f2bdb3f25p-2	1.92
atan	-0x1.604e071f58b6bp-1	0.863	-0x1.607b26abaf75dp-1	0.860
atanh	-0x1.3438ca141da6bp-1	1.67e7	-0x1.d919b473870d1p-2	1.80
cbrt	-0x1.09804d4a074ddp+239	0.502	0x1.0463dd38db734p-966	0.670
cos	0x1.8c53a5c0e4e3fp+20	0.797	0x1.f7ea1d82d650bp+21	0.878
cosh	0x1.fe7d9935cba7bp+0	1.91	-0x1.633cae1335f26p+9	2.67
erf	0x1.c3326e3fcf749p-5	1.41	-0x1.c537f91fbefb6p-15	1.02
erfc	0x1.3ffa8a9835079p+0	4.82	0x1.54b20a16dedd3p+0	3.94
exp	-0x1.6237c4a78d506p+9	0.756	0x1.6f5e981917f38p+6	0.946
exp10	-0x1.33b58776304ebp+8	0.769	-0x1.cac2cdbd51bcdp+5	0.892
exp2	-0x1.ff03ffe8ed867p+9	0.775	-0x1.cff5ac13b60dfp+3	0.895
expm1	0x1.9a57707271a56p-2	0.722	0x1.636c0ba1d8ba5p-2	0.898
j0	0x1.33d152e971b4p+1	4.51e14	0x1.33d152e971b4p+1	4.51e14
j1	-0x1.ea75575af6f09p+1	4.47e14	0x1.ea75575af6f09p+1	2.20e15
lgamma	-0x1.f614148d313dep+1	10.6	-0x1.5fb410a1bd902p+1	2.50e15
log	0x1.0fff65cc00ea7p+0	0.577	0x1.4842fc5cb73a5p+0	0.944
log10	0x1.e016c9316f45fp-1	0.632	0x1.55267ad1a87d5p+0	2.07
log1p	0x1.107728fffffffp-4	2.52e8	-0x1.2c2e475684faap-2	0.891
log2	0x1.e001d7e25ae77p-1	0.616	0x1.67753545c6171p+0	2.05
sin	0x1.59ca6f45120d2p+5	0.799	0x1.5da0fa250a973p+501	0.879
sinh	-0x1.aff46eca66ba7p+4	1.50	-0x1.633cae1335f26p+9	2.67
sqrt	0x1.fffffffffffffp-1	0.500	0x1.fffffffffffffp-1	0.500
tan	0x1.3b1b7b8074c98p+1	1.36	0x1.e2553c2029edp+21	1.01
tanh	0x1.fd9e7edc1868dp-1	1.45	0x1.e0becba6141f5p-3	2.22
tgamma	-0x1.62c4d519e8677p+3	9.80	-0x1.53f198fe3b278p+7	2.25e3
y0	0x1.c982eb8d417eap-1	2.97e15	0x1.c982eb8d417e9p-1	1.01e15
y1	0x1.193bed4dff243p+1	2.78e15	0x1.193bed4dff243p+1	2.78e15
atan2	1.c3cf7fac80334p-320 1.08b0b09ed6889p+703	0.750	-1.cb5abdea8a1a3p-407 1.c24578957c633p-405	1.55
hypot	-1.9b2a45a7406c9p-726 -1.2a30805030bc6p-725	0.942	-1.6a23f1a7f1f0ap-1010 -0.d4fd0d032fb9p-1022	1.21
pow	1.ffeffa651a733p-1 1.5d318bcd8f1f1p+22	700.	1.167d6ebc0a0ecp+474 -1.13b6a2be08fa9p-10	0.895

Table 7: Double precision: AMD LibM and RedHat Newlib

	OpenLibm 0.7.0		Musl 1.2.1	
function	x	max e	x	max e
acos	-0x1.00ab8f608b178p-1	0.923	-0x1.00ab8f608b178p-1	0.923
acosh	0x1.001fb84a1c57dp+0	2.23	0x1.001fb84a1c57dp+0	2.23
asin	0x1.014f1c8b2002dp-1	0.972	0x1.014f1c8b2002dp-1	0.972
asinh	-0x1.0240f2bdb3f25p-2	1.92	-0x1.0240f2bdb3f25p-2	1.92
atan	-0x1.607b26abaf75dp-1	0.860	-0x1.607b26abaf75dp-1	0.860
atanh	-0x1.d919b473870d1p-2	1.80	-0x1.d919b473870d1p-2	1.80
cbrt	-0x1.2bf9d2510bed4p+798	0.668	-0x1.2bf9d2510bed4p+798	0.668
cos	-0x1.922852ab89202p+21	0.829	-0x1.922852ab89202p+21	0.829
cosh	-0x1.63109fbed1e86p+9	1.47	-0x1.502cd557517a3p+0	1.04
erf	-0x1.c537f91fbefb6p-15	1.02	-0x1.c537f91fbefb6p-15	1.02
erfc	0x1.54b20a16dedd3p+0	3.94	0x1.527f4fb0d9331p+0	3.72
exp	0x1.6f5e981917f38p+6	0.946	-0x1.1820bb07ae6fcp+6	0.511
exp10	NA	NA	-0x1.fe8c98781d816p+3	4.14
exp2	-0x1.ff2a39382f61ap+9	0.751	0x1.55ee43d021e9bp+7	0.511
expm1	0x1.636c0ba1d8ba5p-2	0.898	0x1.636c0ba1d8ba5p-2	0.898
j0	0x1.33d152e971b4p+1	4.51e14	-0x1.33d152e971b4p+1	4.51e14
j1	-0x1.ea75575af6f09p+1	2.20e15	0x1.ea75575af6f09p+1	2.20e15
lgamma	-0x1.5fb410a1bd902p+1	2.50e15	-0x1.5fb410a1bd902p+1	2.50e15
log	0x1.4842fc5cb73a5p+0	0.944	0x1.dc0b7b171df46p-1	0.520
log10	0x1.5530c2ed2cd09p+0	0.805	0x1.5530c2ed2cd09p+0	0.805
log1p	-0x1.2c2e475684faap-2	0.891	-0x1.2c10cda877f0ap-2	0.892
log2	0x1.671ea3b7b37a9p+0	0.910	0x1.0b548b52d2c46p+0	0.554
sin	0x1.c8f2d8c965cb4p+21	0.828	0x1.c8f2d8c965cb4p+21	0.828
sinh	0x1.6320943636f24p-1	1.88	0x1.6320943636f24p-1	1.88
sqrt	0x1.fffffffffffffp-1	0.500	0x1.fffffffffffffp-1	0.500
tan	0x1.e2553c2029edp+21	1.01	0x1.e2553c2029edp+21	1.01
tanh	0x1.e0becba6141f5p-3	2.22	0x1.e0becba6141f5p-3	2.22
tgamma	-0x1.540b16dda79bp+7	1.03e3	-0x1.ff2954359aec2p+2	13.9
y0	0x1.c982eb8d417e9p-1	1.01e15	0x1.c982eb8d417e9p-1	1.01e15
y1	0x1.193bed4dff243p+1	2.78e15	0x1.193bed4dff243p+1	2.78e15
atan2	-1.cb5abdea8a1a3p-407 1.c24578957c633p-405	1.55	-1.cb5abdea8a1a3p-407 1.c24578957c633p-405	1.55
hypot	-1.6a23f1a7f1f0ap-1010 -0.d4fd0d032fb9p-1022	1.21	-1.0000783a13589p+153 -1.00211a3a1d4bdp+153	1.04
pow	1.15cf1658fc1bfp+474 1.148d9f780e926p-10	0.896	1.f02800be5acf5p-663 1.68c945fb57d8dp+0	0.513

Table 8: Double precision: OpenLibm and Musl

in Table 10. Only the square root function is correctly rounded (or at least seems to be). The Intel Math Library gives better results than the GNU libc for all functions, except for `lgamma`, `tgamma`, `y0` and `y1`. Apart from those four functions, and for `j0`, `j1`, the observed error for the Intel Math Library is at most 0.7 ulps. The GNU libc has large errors for `j0`, `j1`, `y0` and `y1`.

Acknowledgements. The author thanks Claude-Pierre Jeannerod and Vincent Lefèvre who helped to improve that article, Alexei Sibidanov who managed to compile Newlib 3.3.0, and Eric Schneider for interesting discussions. Experiments presented in this article were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see `https://www.grid5000.fr`). This work was also supported by the French "Ministère de l'Enseignement Supérieur et de la Recherche", by the "Conseil Régional de Lorraine", and by the European Union, through the "Cyber-Entreprises" project. Access to the Intel C Compiler and thus to the Intel Math Library was possible thanks to the PlaFRIM experimental testbed, supported by Inria, CNRS (LABRI and IMB), Université de Bordeaux, Bordeaux INP and Conseil Régional d'Aquitaine (see `https://www.plafrim.fr/`).

References

[1] AMD LibM version 3.5. `https://developer.amd.com/amd-aocl/amd-math-library-libm/`, 2020.

[2] Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., and Zimmermann, P. MPFR: A multiple-precision binary floating-point library with correct rounding. *ACM Trans. Math. Softw. 33*, 2 (2007), article 13.

[3] GNU libc version 2.32. `https://www.gnu.org/software/libc/`, 2020.

[4] Intel Math Library. Distributed with the Intel C compiler 19.1.3.304, 2020.

[5] Muller, J.-M. On the definition of ulp(x). Research Report RR-5504, LIP RR-2005-09, INRIA, LIP, Feb. 2005.

[6] Musl version 1.2.1. `https://www.musl-libc.org/`, 2020.

[7] Redhat Newlib version 3.3.0. `https://sourceware.org/newlib/`, 2020.

[8] OpenLibm version 0.7.0. `https://openlibm.org/`, 2019.

library	GNU libc	Intel Math Library
version	2.32	icc 19.1.3.304
acos	1.26	**0.501**
acosh	3.80	**0.501**
asin	1.16	**0.501**
asinh	3.82	**0.501**
atan	1.41	**0.501**
atanh	3.80	**0.501**
cbrt	0.736	**0.501**
cos	1.50	**0.501**
cosh	1.92	**0.501**
erf	1.37	**0.501**
erfc	3.92	**0.503**
exp	0.751	**0.501**
exp10	2.00	**0.501**
exp2	1.08	**0.501**
expm1	1.63	**0.501**
j0	4.10e32	**7.07e27**
j1	1.23e33	**1.56e28**
lgamma	**11.9**	2.79e30
log	1.05	**0.501**
log10	1.92	**0.501**
log1p	3.36	**0.501**
log2	3.11	**0.501**
sin	1.50	**0.501**
sinh	2.05	**0.501**
sqrt	**0.500**	**0.500**
tan	0.980	**0.501**
tanh	2.38	**0.501**
tgamma	**9.19**	820e1
y0	1.95e33	**3.47e27**
y1	1.74e33	**7.75e29**
atan2	1.88	**0.501**
hypot	0.970	**0.501**
pow	30.0	**0.698**

Table 9: Quadruple precision: Maximal known value of e.

function	GNU libc 2.32 x	max e	icc 19.1.3.304 x	max e
acos	0x9.fd3c51387b1f698381bc5d3b4af8p-4	1.26	0xe.217a11378bd0c6ccddc53b539b78p-4	0.501
acosh	0x1.0f9a6e1f4499e484d7f53bc170e7p+0	3.80	0x1.788a2041a667a201e7093d866375p+5628	0.501
asin	-0x7.7672f932b0c5f0c5666c8163555cp-4	1.16	0xf.1b9d12f4feb7cc1c9fdc8dcc54fp-8	0.501
asinh	0x5.bbcc54dde5dabba4a36c37747898p-4	3.82	0x7.1135edcbf336d544ae8e7f6045ccp+12704	0.501
atan	-0x3.6b2b37b59b1cd425685c2746a6cp-4	1.41	0x8.19bd98406e1bd58d545504f51ba8p-4	0.501
atanh	0x2.c2d2fd8fa42ab1d37b453fc78f92p-4	3.80	-0xc.695e3aa5d7e2b887925109fb9008p-8	0.501
cbrt	-0x2.d42980a1deea450d7ec2c4ae159cp-10368	0.736	-0x2.1de15f5104791b63ee07d1e082eep+7240	0.501
cos	0x2.3a6711cb055589aa34d0218a401cp+9532	1.50	0xb.05c12be250566b68401ead906f18p+1808	0.501
cosh	0x2.c5d375cf727efd4b92d0bf9886bp+12	1.92	0xc.ec4b880ab1fa8c878a1224d85b3p+8	0.501
erf	0xd.f28887263c09c318d53a958b5618p-4	1.37	-0x3.5f29e8794e234313fd4054aeb1eep-16012	0.501
erfc	0x1.5188f8154395cdf53061a2313f44p+0	3.92	0x5.302fafa0957e8b708c0598bf086p+0	0.503
exp	-0x2.c5b81ef37e1c803cfe8df1c48b76p+12	0.751	-0xa.6d595ca011b0cfbdd8af23101328p-8	0.501
exp10	0x1.3426ffa24cc1a0a3471ab64c223ap+0	2.00	-0x1.c4e7130bfa380e1fb95dd2a79c49p-8	0.501
exp2	0x9.4ffff7996aa526db92440861abd8p+4	1.08	0x6.0bcfa351613cc07b9ff633b8525cp+4	0.501
expm1	0x5.a3be875d8a17cbf85ac8dc4f1054p-4	1.63	0xc.ec4b880ab1fa8c8765f984544e18p+8	0.501
j0	-0x8.a75ab6666f64eae68f8eb383dad8p+0	4.10e32	0x1.b7e54a5fd5f1174acdc525caca0bp+4	7.06866e+27
j1	0x3.d4eaaeb5ede114ff552b1726d4ep+0	1.23e33	0x3.f9c81df622adc7f47ba0a18d6288p+4	1.56e28
lgamma	-0x3.ec1bd0fc6a9c6bf9b04c0bcd1a96p+0	11.9	-0x3.24c1b793cb35efb8be699ad3d9bap+0	2.79e30
log	0xf.d003c0a04f7a8cdf5781a2c6c6bp-4	1.05	0x7.f42a630d366fb759e164e66360a8p-11240	0.501
log10	0x1.6a67162f5d4c66be7c3acaae03adp+0	1.92	0x3.ffd8ce516b0ab868e40093fcb98ap-3112	0.501
log1p	0x6.a44f02701fd76e2e51619abff3e8p-4	3.36	0x7.2948b50872ca1286bcd77af9e9d4p-16	0.501
log2	0xb.5093d5e00665368875e3b3acb1bp-4	3.11	0x8.d84c20c719f4996d47c8e4a7781p-4	0.501
sin	0xd.f0946b5f80ab727eb20232446118p+16	1.50	0x5.0e5e595b3dc9a1955bf63b192a94p+11456	0.501
sinh	-0x6.8029c99aa94b517846e372e9546p-4	2.05	0x1.6645bfeb3445fb190fcc69bca122p+0	0.501
sqrt	0xf.ffffffffffffffffffffffffffp-4	0.500	0xf.ffffffffffffffffffffffffffp-4	0.500
tan	-0x7.b47f28fc82e6766f8e08db648358p+44	0.980	-0x6.3ebb44fab8cf5e8aa2e9ee07fbe8p+5252	0.501
tanh	-0x3.c0dd2fc8f76ac524817e9e048658p-4	2.38	0x3.6d01352ca09b98f141cd5e1c5e7ep-4	0.501
tgamma	-0x1.6d4874ee3d217e2d944e85ebac0ap+4	9.19	-0x8.000312d2ac527656bbb3962352dp-15208	820e1
y0	0xe.4c175c6a0bf51ea9d270347f83p-4	1.95e33	0x1.9ec46f3e80145efc1fad4263f512p+4	3.47e27
y1	0x2.3277da9bfe485c85c35e5bcc806p+0	1.74e33	0x2.4e74f66b4f5fe49207a1eb7f95cap+4	7.75e29
atan2	0xf.938844ba20022bcea059e0284cd8p+5552 0x4.90f37d2eb884ef22c644975011b4p+5556	1.88	0x1.ee775bceed12e2483de370ae7897p+2640 0x8.27ffd5e2ae10b37f8999dbcd7e4p+15300	0.501
hypot	-0x2.d8ae430f6bd03e67982daec79af8p-6212 -0x5.b2d848a3fcd3feb7ca53ea213ee4p-6212	0.970	0x5.dc9844d192a156e7591a6a7f41f8p+13692 0x3.0f21883c704c6702b80d3249c604p+13680	0.501
pow	0x1.3634b6a42d9274bdb866a0f13273p+0 0xe.6e4c4c99b8a46dd4faf0c6279b8p+12	30.0	0x4p-16496 0x3.fffe2a13f7d847ae8771a3816c6ep-128	0.698

Table 10: Quadruple precision: GNU libc.