Static and Dynamic Numerical Characteristics of Floating-Point Arithmetic

WILLIAM J. CODY, JR.

Abstract—The appearance of hexadecimal floating-point arithmetic systems has prompted a continuing discourse on the relative numerical merits of various choices of base. Until lately this discourse has centered around the static properties of the floating-point representation of numbers, and has primarily concerned only binary and hexadecimal representations. Recent events may change this discourse considerably. A third numerically attractive alternative for the choice of base has been proposed, and a comparison of the dynamic numerical properties of floating-point arithmetic systems has been completed. This paper surveys these recent events and summarizes our current knowledge of the numerical characteristics of floating-point arithmetic systems.

Index Terms—Floating-point arithmetic, representational errors, rounding errors.

Introduction

Until the advent of the IBM S/360 with its hexadecimal base for the floating-point number system, most computers were designed with a binary floating-point number system, although Burroughs produced machines with an octal system. These latter machines were used for scientific computation with very little complaint about their numerical properties. The hexadecimal machines, however, have drawn many unfavorable comments from users who have applied them to scientific computation.

Until recently, the arguments and discussions of the relative merits of hexadecimal, binary, and other floating-point systems have primarily centered upon such static aspects of the representation of numbers as the exponent range, density of numbers, and the maximum relative error of representation. Within the last year, however, the discussion has broadened to include the dynamic characteristics of the whole floating-point arithmetic system, of which the number representation scheme is only a part. In addition, a base 4, or quaternary, number representation scheme has been proposed that has interesting properties.

This paper surveys and summarizes what we now know about the static and dynamic numerical characteristics of floating-point arithmetic systems.

Preliminaries

We assume that we have a normalized sign-magnitude floating-point representation scheme with d bits per word. We will let β denote the base for the number system, e denote the biased exponent, and f the fraction. We will assume that f contains t bits, equivalent to N β-ary digits (where N need not be integral), hence $p = d - t - 1$ bits for the exponent e.

When we particularize our discussions we will consider only three specific floating-point representations. These can be characterized as $(\beta, p, t) = (2, 9, 22)$, $(4, 8, 23)$, and $(16, 7, 24)$, respectively. They have been chosen to give essentially the same range of number representation for a 32-bit word as is found in the short-precision arithmetic on current hexadecimal machines. This is a convenience for comparison purposes only. Other word lengths are more appropriate for scientific computation, but the results we will obtain can be easily modified for other values of d and t.

Static Characteristics

At this point we can already say a few things about the effect of various choices of β. McKeeman [7] takes the view that f represents an equivalence class of real numbers, and that an error is made in representing an element of the equivalence class of f. Let us call this error the ***representation error.*** Assuming the logarithmic probability distribution for floating-point numbers (Hamming [4])

$$P(f) = 1/(f \ln \beta), \qquad 1/\beta \leqslant f < 1$$

the average relative representation error (ARRE) is then found to be

$$\mathrm{ARRE}\,(t,\beta) = \int_{1/\beta}^{1} \frac{2^{-t}\, df}{f \ln \beta\, 4f} = \frac{\beta - 1}{4 \cdot 2^{-t} \ln \beta}$$

and the maximum (over all f) relative representation error (MRRE) is

$$\mathrm{MRRE}\,(t,\beta) = 2^{-t-1}\,\beta.$$

The values of ARRE and MRRE for the three representation schemes under consideration are given in Table I. The fact that the MRRE for binary representations is half of that for the corresponding hexadecimal representations is frequently used as an argument for the superiority of binary representations over hexadecimal representations. At best this superiority is marginal, especially if the values of ARRE are also considered. The interesting feature of our comparison is the showing for $\beta = 4$. As Brent [1] pointed out in suggesting the consideration of the quaternary representation system, the quaternary representation of a number is never less accurate than the corresponding binary representation, and the value of ARRE is more than 20 percent smaller than that for the corresponding binary scheme.

A second possible comparison involves the range of repre-

Manuscript received March 15, 1972; revised December 15, 1972. This work was performed under the auspices of the U.S. Atomic Energy Commission.

The author is with the Applied Mathematics Division, Argonne National Laboratory, Argonne, Ill. 60439.

TABLE I
STATIC CHARACTERISTICS OF FLOATING-POINT NUMBERS

β	p	t	MRRE	ARRE	Exponent Range	N.R.
2	9	22	$.5*2^{-21}$	$.18*2^{-21}$	2^{2^9-1}	$2^{2^9}(1-2^{-22})$
4	8	23	$.5*2^{-21}$	$.14*2^{-21}$	2^{2^9-2}	$2^{2^9}(1-2^{-23})$
16	7	24	2^{-21}	$.17*2^{-21}$	2^{2^9-4}	$2^{2^9}(1-2^{-24})$

TABLE II
MEAN AND VARIANCE OF ERROR

β	t	$\mu(\epsilon_c)$	$\sigma^2(\epsilon_r)$
2	22	$2^{-22.47}$	$2^{-46.47}$
4	23	$2^{-22.89}$	$2^{-47.15}$
16	24	$2^{-22.56}$	$2^{-46.06}$

sentable numbers. Specializing a theorem due to Brown and Richman [2] to the case where each machine word consists of d two-state devices, we see that for a given word length, the choice $\beta = 2$ gives at least as much accuracy as and a greater exponent range than the choice $\beta = 2^j$, $j > 1$. The measurement of accuracy used here is the MRRE.

Dunham [3] points out that this statement is somewhat misleading in the sense that the extra fractional bits for the $\beta = 2^j$, $j > 1$, case partially compensate for the decrease in exponent range. He feels that the proper measure is the ratio of the largest floating-point number to the smallest positive number. This *number range* (NR) is

$$\mathrm{NR} = \frac{\beta^{2^p-1}(1 - 2^{-t})}{\beta^0 \beta^{-1}} = \beta^{2^p}(1 - 2^{-t}).$$

Table I also gives the values of the exponent ranges and the NR's for the three representation schemes under consideration. The overall comparison presented in Table I shows that there is little difference between static characteristics of binary and hexadecimal representations, and only a slight edge in the ARRE for the quaternary representation.

DYNAMIC CHARACTERISTICS

We now turn our attention to the characteristics of the total arithmetic system in a dynamic situation. By an arithmetic system we mean a floating-point number representation coupled with a specific arithmetic. The particular arithmetic parameters that we will consider are the method of rounding and the number of available guard digits.

We will assume that the arithmetic has g β-ary guard digits. Thus $N + g$ β-ary digits, of which there may be some leading zero digits, are developed in the arithmetic registers at an intermediate stage of an arithmetic operation. The g guard digits participate in postnormalization of the result fraction and in rounding, but only N digits are retained at the end of the operation.

We will discuss three different modes of rounding. In the chop mode, hereafter abbreviated to C-mode, the intermediate result of an operation is fitted to the precision of the machine by ignoring any extra digits of the fraction after postnormalization. By the round mode, or R-mode, we mean a simple rounding up or down of the postnormalized fraction, depending upon whether the first binary guard digit of the postnormalized fraction is 1 or 0. (To the author's knowledge, most machines that round do so before postnormalization, giving a bastardized version of the R-mode arithmetic.) Both C-mode and R-mode arithmetics are biased. For the C-mode case the bias is obvious. For the R-mode, the bias is due to the case where the guard digits prior to rounding are equivalent to precisely half a unit in the last digit to be retained. This threshold case, in which the fraction is always increased in magnitude, occurs with a frequency that cannot be ignored in computer arithmetic. There are many ways of removing the bias, but perhaps the simplest is to force the last bit retained to a 1 in the threshold cases. We will call this the R^*-mode.

Some of the earliest investigations of arithmetic systems are those of Wilkinson [8]. He developed rigid error bounds for the results of basic arithmetic operations assuming N guard digits and perfect rounding. His bounds are rigid upper bounds on the error, but they are not always sharp, nor does he attempt a statistical error analysis.

Recently, Kaneko and Liu [5] extended Wilkinson's work to the case of summation with g guard characters. They showed that Wilkinson's original bounds are modified by a factor $1/(1 - \beta^{-g})$. For the important case $g = 1$, this reduces to $\beta/(\beta - 1)$. Thus, on a hexadecimal machine the error bounds are increased by the small factor $\frac{16}{15}$, prompting the observation that one guard character is almost as effective as a double-length accumulator for summation in hexadecimal systems.

To initiate the statistical study of error, let E_c denote the error due to chopping, and assume that it is uniformly distributed in $[-2^{-t}, 0]$. Similarly, assume E_r, the error due to rounding, is uniformly distributed in $[-2^{-t-1}, 2^{-t-1}]$. Assuming the logarithmic distribution for f, we can then determine the mean and variance of the relative error

$$\epsilon = E/f$$

directly, as is done by Kuki and Cody [6], or by first determining the probability distribution function for ϵ, as is done by Brent [1], and by Kaneko and Liu [5]. In our present terminology

$$\mu(\epsilon_c) = \frac{\beta - 1}{2^{t+1} \ln \beta}$$

$$\mu(\epsilon_r) = 0$$

$$\sigma^2(\epsilon_c) = \frac{\beta^2 - 1}{6 \cdot 2^{2t} \ln \beta}$$

$$\sigma^2(\epsilon_r) = \tfrac{1}{4}\, \sigma^2(\epsilon_c).$$

Table II lists some of these quantities for the arithmetic systems under consideration. Again note that there is little difference between binary and hexadecimal, but that quaternary consistently has a slight advantage.

Kuki and Cody [6] have recently completed a number of experiments comparing various arithmetic systems for evaluation of sums, products, and inner products. For these experiments β was limited to either 2 or 16. For each value of β, C-mode, R-mode, and R^*-mode arithmetics were coupled with 0, 1, 2, and "many" guard characters, as appropriate. The experiments were carefully designed to neutralize the effect of the fluctuation of the number of significant bits in a hexadecimal fraction. Tabulated results include the minimum value, mean, and standard deviation of the binary significance

$$s = -\log_2 (|r|)$$

where r is the relative error of the computational result. In a number of cases the validity of the experimental results was established by reproducing the results using a statistical error analysis. The analytic results have since been extended to the quaternary system.

Table III summarizes the results for the summation experiments in which 500 sums of 1024 summands each were evaluated. The summands were drawn at random from an appropriate distribution over the interval $[\frac{1}{16}, 16]$, using the arithmetic under study to fit the data to the machine precision. Table IV compares the analytic and experimental results for the summation tests involving only positive summands and the corresponding results for the experiments involving products. For both experiments, the analytic results found in [6] have been extended to the quaternary case.

For these experiments, arithmetic systems differing only in their choice of binary or hexadecimal representation schemes appear to be statistically equivalent, but somewhat inferior to corresponding quaternary systems. R-mode arithmetic is uniformly superior to C-mode arithmetic, except in the case of C-mode without guard digits applied to mixed-sign sums (see Table III). In this case, C-mode outperformed R-mode with an arbitrary number of guard digits. This local superiority is because C-mode arithmetic is unbiased in this particular case, whereas R-mode is not. R^*-mode was superior to C-mode in all cases. For hexadecimal systems, C-mode coupled with one guard digit was essentially as effective as C-mode with an infinite number of guard digits. Similarly, R-mode and R^*-mode essentially achieved maximum effectiveness with two guard digits (one being required for postnormalization in multiplication, and the other for rounding). These latter trends were not nearly as striking for binary arithmetic, hence appear to depend upon the choice of β.

Conclusions

We can summarize our findings on the characteristics of floating-point arithmetic systems as follows.

1) Based solely on the MREE, binary representations are superior to hexadecimal.

2) Arithmetic systems differing only in the use of binary and hexadecimal representations appear statistically to give the same computational accuracy.

3) Quaternary number systems appear statistically to give better accuracy than binary or hexadecimal systems. At least they merit further consideration.

TABLE III
Experimental Results for 500 Sums of 1024 Summands Each

Mode	g	Hexadecimal		Binary	
		min. s	μ(s)	min. s	μ(s)
		All summands positive			
C	-	12.85	13.71±.69	13.34	13.47±.13
R	≥1	16.57	20.01±.65	17.37	19.88±1.57
		Mixed sign summands			
C	0	8.96	18.23±2.04	7.48	18.10±2.05
	1	5.86	13.55±1.93	5.68	14.17±1.84
	∞	5.85	13.52±1.94	4.80	13.23±1.83
R	1	9.35	17.53±2.06	6.11	14.20±1.77
	2	10.26	18.08±2.00	6.69	15.16±1.88
	∞	10.26	18.09±2.01	8.81	17.78±2.10
R*	∞	11.31	19.03±2.08	9.69	19.07±2.16

TABLE IV
Comparison of Analytic and Experimental Results

Experiment[1]	Mode	g	β	Statistic[2]	
				analytic	experimental
I	C	0	2	13.470	13.46
			4	13.885	---
			16	13.563	13.55
	R*	≥1	2	19.943	19.88
			4	20.282	---
			16	19.738	20.01
II	C	0	2	16.904	16.87
			4	16.953	---
			16	15.624	15.50
		1	2	17.186	17.18
			4	17.601	---
			16	17.279	17.28
	R*	∞	2	21.509	21.51
			4	21.848	---
			16	21.305	21.36

[1] Experiment I is 500 sums of 1024 positive summands each. Experiment II is 500 products of 20 factors each.

[2] The statistic for C-mode is $-\log_2(\mu|r|)$. That for R^*-mode is $\mu(s)$.

4) Properly implemented R^*-mode arithmetic is superior to C-mode.

5) Almost maximum accuracy can be achieved in R^*-mode with two guard digits, and in C-mode with one guard digit for larger values of β.

Current machine designs offer primarily hexadecimal C-mode or bastardized binary R-mode arithmetic. There may be valid reasons for preferring binary representation schemes over hexadecimal ones, but the inferior numerical performance of current hexadecimal machines appears to be due to other design considerations. At this point it appears that the best machine design would couple R^*-mode arithmetic with two guard characters, and possibly a quaternary representation scheme.

REFERENCES

[1] R. P. Brent, "On the best choice of a base for floating-point number representation," IBM Res. Rep. RC 3751, Feb. 1972.
[2] W. S. Brown and P. L. Richman, "The choice of base," *Commun. Ass. Comput. Mach.*, vol. 12, pp. 560-561, Oct. 1969.
[3] C. B. Dunham, "Choice of base for a binary machine," unpublished.
[4] R. W. Hamming, "On the distribution of numbers," *Bell Syst. Tech. J.*, vol. 49, pp. 1609-1626, Oct. 1970.
[5] T. Kaneko and B. Liu, "On local round-off errors in floating-point arithmetic," *J. Ass. Comput. Mach.*, to be published.
[6] H. Kuki and W. J. Cody, "A statistical study of the accuracy of floating point number systems," *Commun. Ass. Comput. Mach.*, to be published.
[7] W. M. McKeeman, "Representation error for real numbers in binary computer arithmetic," *IEEE Trans. Electron. Comput.* (Short Notes), vol. EC-16, pp. 682-683, Oct. 1967.
[8] J. H. Wilkinson, *Rounding Errors in Algebraic Processes.* Englewood Cliffs, N.J.: Prentice-Hall, 1963.

William J. Cody, Jr., was born in Melrose Park, Ill., on November 28, 1929. He received the B.S. degree in mathematics from Elmhurst College, Elmhurst, Ill., in 1951 and the M.A. degree in mathematics from the University of Oklahoma, Norman, in 1956.

During the summers of 1957 and 1958 he was employed at Los Alamos National Laboratory. In 1958 he joined Northwestern University, Evanston, Ill., as an Instructor in Mathematics. He has been at Argonne National Laboratory, Argonne, Ill., since 1959. His research interests include the approximation of special functions, the design and evaluation of mathematical software, and the interaction between computer arithmetic design and numerical algorithms.

Mr. Cody is a member of the Association for Computing Machinery, the Society for Industrial and Applied Mathematics, the American Mathematical Society, the Mathematical Association of America, and SIGNUM.

On the Precision Attainable with Various Floating-Point Number Systems

RICHARD P. BRENT

Abstract—For scientific computations on a digital computer the set of real numbers is usually approximated by a finite set F of "floating-point" numbers. We compare the numerical accuracy possible with different choices of F having approximately the same range and requiring the same word length. In particular, we compare different choices of base (or radix) in the usual floating-point systems. The emphasis is on the choice of F, not on the details of the number representation or the arithmetic, but both rounded and truncated arithmetic are considered. Theoretical results are given, and some simulations of typical floating-point computations (forming sums, solving systems of linear equations, finding eigenvalues) are described. If the leading fraction bit of a normalized base-2 number is not stored explicitly (saving a bit), and the criterion is to minimize the mean square roundoff error, then base 2 is best. If unnormalized numbers are allowed, so the first bit must be stored explicitly, then base 4 (or sometimes base 8) is the best of the usual systems.

Index Terms—Base, floating-point arithmetic, radix, representation error, rms error, rounding error, simulation.

Manuscript received May 15, 1972; revised May 31, 1972.

The author was with the Mathematical Sciences Department, IBM T. J. Watson Research Center, Yorktown Heights, N.Y. 10598. He is now with the Computer Centre, Australian National University, Canberra, A.C.T., Australia.

I. INTRODUCTION

A REAL number x is usually approximated in a digital computer by an element $\mathrm{fl}(x)$ of a finite set F of "floating-point" numbers. We regard the elements of F as exactly representable real numbers, and take $\mathrm{fl}(x)$ as the floating-point number closest to x. The definition of "closest," rules for breaking ties, and the possibility of truncating instead of rounding are discussed later.

We restrict our attention to binary computers in which floating-point numbers are represented in a word (or multiple word) of fixed length w bits, using some convenient (possibly redundant) code. Usually F is a set of numbers of the form

$$s \sum_{i=1}^{t} d_i \beta^{e-i} \tag{1.1}$$

where $\beta = 2^k > 1$ is the base (or radix), $t > 0$ is the number of digits, $s = \pm 1$ is a sign, e is an exponent in some fixed range

$$m < e \leqslant M, \tag{1.2}$$

and each d_i is a β-ary digit $0, 1, \cdots, \beta - 1$. Other possible