A Piecewise Linear Approximation of $\log_2 x$ with Equal Maximum Errors in All Intervals

JOHN B. KIOUSTELIDIS AND JOHN K. PETROU

***Abstract*—In this paper it is shown how to divide the interval [1,2] into n parts so that the uniform linear approximation of $\log_2 x$ in each subinterval has the same maximum error. This error is, in the case $n = 4$, smaller by a factor of 2.3 than the error of the linear mean-square approximation given by Hall *et al.* [1]. The final products of the mathematical analysis are explicit formulas which allow the direct determination of all parameters and the maximum error for any desired number n of subdivisions of [1,2].**

***Index Terms*—Approximate computation, approximate evaluation of elementary functions, binary logarithm, linear Chebyshev approximation.**

INTRODUCTION

THE construction of a formant vocoder (see Shafer and Rabiner [2]), as well as many other systems for the processing of analog signals, requires a fast and very accurate method for the determination of the logarithm. This problem has been already treated in the literature (see Hall *et al.* [1]) by use of a linear approximation which is the fastest computable approximation. But the approximations which have been proposed do not exhaust the limits of accuracy which a linear approximation can yield.

Therefore, the best possible uniform linear approximation will be determined in this paper, and it will be shown that it has considerably better error bounds. In order to make the comparison easier in the further considerations, the special case of the binary logarithm will be discussed. This does not in any way restrict the applicability of the proposed method because the logarithm with respect to any other basis is proportional to the binary logarithm ($\log_b x = k \cdot \log_2 x$ with $k = \log_2 b = 1/\log_b 2$).

To begin with, it can be easily seen how an approximation, which is valid only for a finite interval, can be used for the determination of the logarithm of an arbitrary number, and why the linear approximation is, as above claimed, the fastest computable approximation.

For an arbitrary positive number y, there is an integer power k of 2 such that

$$2^k \leq y < 2^{k+1}; \tag{1}$$

consequently,

$$0 \leq \log_2 y - k \leq 1$$

and we conclude that any binary logarithm is given by the formula

$$\log_2 y = k + \log_2 x \tag{2}$$

where k is the first significant binary digit of y and

$$x = y/2^k \in [1,2]. \tag{3}$$

The above division, as well as the determination of k, can be achieved by simple shifting if the numbers are expressed in binary.

For the above applications, it is therefore sufficient to find an appropriate approximation of $\log_2 x$ for [1,2]. In order that this approximation may be quickly computed, it must involve as few multiplications as possible. Consequently, we use only one multiplication and one addition. On the other hand, this is not sufficiently accurate unless we divide the interval [1,2] in smaller parts $[x_0,x_1]$, $[x_1,x_2],\cdots,[x_{n-1},x_n]$ with $x_0 = 1, x_n = 2$ and determine a separate linear approximation for each subinterval.

Besides that, for the sake of uniformity of the approximation, we try to make the maximum errors in all intervals equal. The most suitable approximation method in this case is that of the minimization of the maximum error, that is, the Chebyshev approximation (see Cheney [3]).

DETERMINATION OF THE BEST UNIFORM LINEAR APPROXIMATION OF $\log_2 x$ IN THE INTERVAL $I_{m+1} = [x_m,x_{m+1}]$

It is known that

$$\log_2 x = c \cdot \ln x, \qquad c = 1/\ln 2. \tag{4}$$

We therefore want to determine

$$E_{m+1} = \min_{a,b \in R} \left(\max_{x \in I_{m+1}} | c \cdot \ln x - ax - b | \right) \tag{5}$$

and the corresponding a,b.

Let

$$g(x) := c \cdot \ln x - ax - b. \tag{6}$$

$\max_{x \in I_{m+1}} | g(x) |$ appears either at the endpoints x_m,x_{m+1} of the interval or at the point

$$\bar{x} = c/a \tag{7}$$

where the derivative of $g(x)$ equals zero.

Thus we have

Manuscript received February 5, 1974; revised December 15, 1974.
The authors are with the Research Center for National Defense, Galatsi, Athens, Greece.

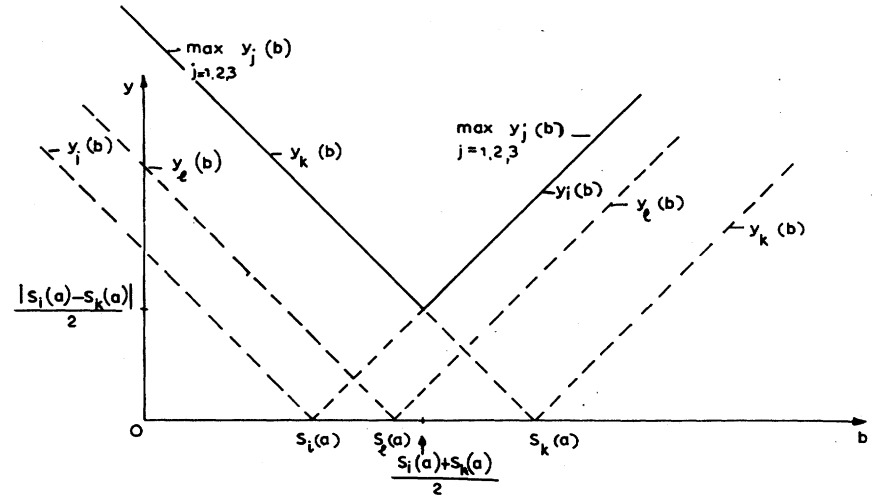

Fig. 1. Functions $y_j(b)$ and $\max_{j=1,2,3} y_j(b)$.

$$\max_{x \in I_{m+1}} | c \cdot \ln x - ax - b | = \max_{x = x_m, x_{m+1}, c/a} | c \cdot \ln x - ax - b | = \max_{j=1,2,3} | b - s_j(a) | \tag{8}$$

with

$$s_1(a) = c \cdot \ln x_m - ax_m \tag{9}$$

$$s_2(a) = c \cdot \ln x_{m+1} - ax_{m+1} \tag{10}$$

$$s_3(a) = c \cdot \ln (c/a) - c. \tag{11}$$

The functions

$$y_j(b) := | b - s_j(a) |, \qquad j = 1,2,3 \tag{12}$$

are shown in Fig. 1 where

$$s_i(a) = \min_{j=1,2,3} \{s_j(a)\} \tag{13}$$

$$s_k(a) = \max_{j=1,2,3} \{s_j(a)\}. \tag{14}$$

In Fig. 1, the plot of the function $\max_{j=1,2,3} \{y_j(b)\}$ is given by the full line, and the point $\min_b (\max_j (y_j(b)))$ is the intersection point of $y_i(b)$ and $y_k(b)$. The corresponding b value is

$$b = \frac{s_i + s_k}{2}. \tag{15}$$

Therefore,

$$y(a) := \min_b (\max_j (y_j(b))) = y_i\left(\frac{s_i + s_k}{2}\right) = \tfrac{1}{2} | s_i(a) - s_k(a) |. \tag{16}$$

For the further determination of $y(a)$, we must find the indices i and k, that is, the relative position of the $s_j(a)$ to each other.

At first we note that it is

$$s_1(a) \leq s_2(a) \tag{17}$$

for

$$a \leq c/x_{m+1/2} \tag{18}$$

where

$$x_{m+1/2} = \frac{x_{m+1} - x_m}{\ln (x_{m+1}/x_m)}. \tag{19}$$

Further, we use the well-known inequality

$$\ln t \leq t - 1, \qquad \text{for } t > 0. \tag{20}$$

If we successively substitute in it the values x_{m+1}/x_m, x_m/x_{m+1}, ax_m/c, and ax_{m+1}/c for t, we get the inequalities

$$x_m \leq x_{m+1/2} \leq x_{m+1} \tag{21}$$

$$s_1(a) \leq s_3(a) \tag{22}$$

$$s_2(a) \leq s_3(a). \tag{23}$$

The relative positions of the $s_j(a)$ and the value of $y(a)$ are therefore as shown in Fig. 2.

Considering now $y(a)$ in each of the subintervals $[c/x_{m+1}, c/x_{m+1/2}]$ and $[c/x_{m+1/2}, c/x_{m+1}]$, we can show by means of inequality (20) that it is monotonically decreasing in the first subinterval and monotonically increasing in the second. We therefore have

$$E_{m+1} = \min_{a \in [c/x_{m+1}, c/x_m]} y(a) = y(c/x_{m+1/2}) = \frac{c}{2}\left(\frac{x_m}{x_{m+1/2}} - 1 - \ln\left(\frac{x_m}{x_{m+1/2}}\right)\right). \tag{24}$$

The corresponding values of a and b, which give the optimal approximation, are

$$a_{m+1} = c/x_{m+1/2} \tag{25}$$

$$b_{m+1} = \frac{c}{2}\left\{\ln (x_m x_{m+1/2}) - 1 - \frac{x_m}{x_{m+1/2}}\right\}. \tag{26}$$

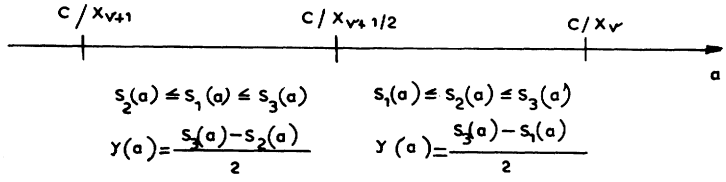

Fig. 2. Relative position of the functions $s_1(a), s_2(a), s_3(a)$.

DETERMINATION OF INTERVALS $[x_0,x_1],\cdots,[x_{n-1},x_n]$ $(x_0 = 1, x_n = 2)$ WITH EQUAL MAXIMUM ERRORS $E_1,\cdots,E_n$

Let $f_m(x)$ be any approximation of $\log_2 x = c\cdot\ln x$ in the interval $I_m = [x_{m-1},x_m]$ and $e_m(x)$ the corresponding error function:

$$c\cdot\ln x = f_m(x) + e_m(x), \qquad \text{for } x \in I_m. \tag{27}$$

Considering the next interval I_{m+1}, we may now observe that if to every $t \in I_{m+1}$ there corresponds an $x \in I_m$ by means of

$$t = d\cdot x, \tag{28}$$

then

$$c\cdot\ln t = c\cdot\ln (d\cdot x) = c\cdot\ln d + c\cdot\ln x \tag{29}$$

$$= c\cdot\ln d + f_m(x) + e_m(x) = f_{m+1}(t) + e_{m+1}(t). \tag{30}$$

Thus, in order to have

$$e_{m+1}(t) = e_m(x) = e_m(t/d), \tag{31}$$

it is sufficient to set

$$f_{m+1}(t) = c\cdot\ln d + f_m(x) = c\cdot\ln d + f_m(t/d). \tag{32}$$

Then we have also for the maximal errors

$$E_{m+1} = \max_{t\in I_{m+1}} | e_{m+1}(t) | = \max_{x\in I_m} | e_m(x) | = E_m. \tag{33}$$

Further, it should be noted that if $f_m(x)$ is the best uniform approximation of $\log_2 x$ in I_m, then $f_{m+1}(t)$ is the best uniform approximation of $\log_2 x$ in I_{m+1}. If a better uniform approximation $h_{m+1}(t)$ existed in I_{m+1}, that is, one with maximum error $E_{m+1}' < E_{m+1} = E_m$, then the function

$$h_m(x) = h_{m+1}(d\cdot x) - c\cdot\ln d \tag{34}$$

would be a better uniform approximation of $\log_2 x$ in I_m than $f_m(x)$.

Let us now divide $[1,2]$ in n subintervals of the above kind. From (28), we get for the initial points and the final points of these intervals (with $x_0 = 1, x_n = 2$)

$$x_{n-1} = dx_{n-2} = \cdots = d^{n-1}\cdot x_0 = d^{n-1} \tag{35}$$

$$2 = x_n = dx_{n-1} = \cdots = d^{n-1}\cdot x_1, \tag{36}$$

that is,

$$2 = d\cdot x_{n-1} = d^n \tag{37}$$

or

$$d = 2^{1/n}. \tag{38}$$

With this value for d, we get

$$f_{m+1}(t) = 1/n + f_m(t/d) = \cdots = m/n + f_1(t/d^m)$$

$$= m/n + f_1(2^{-m/n}t). \tag{39}$$

SUMMARY OF THE RESULTS AND COMPARISON TO ANOTHER APPROXIMATION

Summarizing the above results, we have for the best uniform linear approximation

$$f_{m+1}(x) = a_{m+1}x + b_{m+1}, \qquad \text{for } x \in [x_m,x_{m+1}] \tag{40}$$

with

$$x_m = 2^{m/n}, \qquad m = 1,2,3,\cdots,n \tag{41}$$

$$a_{m+1} = \frac{1}{n2^{m/n}(2^{1/n}-1)}, \qquad m = 0,1,2,3,\cdots,n-1 \tag{42}$$

$$b_{m+1} = \frac{2m+1}{2n} - \frac{1}{2\ln n} - \frac{1}{2n(2^{1/n}-1)},$$

$$m = 0,1,2,\cdots,n-1 \tag{43}$$

$$E_{m+1} = \frac{1}{2n(2^{1/n}-1)} - \ln\frac{\ln 2}{n(2^{1/n}-1)} - \frac{1}{2\ln 2},$$

$$m = 0,1,2,\cdots,n-1. \tag{44}$$

In the special case $n = 4$, by use of six decimals, we have the following results.

m	x_m	a_m	b_m	E_m
0	1.000000			
1	1.189207	1.321303	−1.318597	0.002707
2	1.414214	1.111079	−1.068597	0.002707
3	1.681733	0.934303	−0.818597	0.002707
4	2.000000	0.785652	−0.568597	0.002707

The maximum error is the same for all intervals and equals $2.707\ 10^{-3}$. The maximum error in the approximation given by Hall *et al.* [1, p. 99, Table I] for $n = 4$ with six decimals is, on the contrary, $6.243\ 10^{-3}$, that is, greater by 2.3 times.

REFERENCES

[1] E. L. Hall, D. D. Lynch, and J. Dwyer, III, "Generation of products and quotients using approximate binary logarithms for digital filtering and applications," *IEEE Trans. Comput.*, vol. C-19, pp. 97–105, Feb. 1970.

[2] R. W. Schafer and R. Rabiner, "System for automatic formant analysis of voiced speech," *J. Acoust. Soc. Amer.*, vol. 47, pp. 634–648, 1970.
[3] E. W. Cheney, *Introduction to Approximation Theory*. New York: McGraw-Hill, 1966.

John B. Kioustelidis was born in Athens, Greece, on July 15, 1940. He received the Dipl. Ing. degree in electrical engineering from the Technical University of Aachen (Technische Hochschule Aachen), Aachen, West Germany, in 1965, and the Dipl. Math. and Dr. rer. nat. degrees in 1969 and 1972, respectively.

From 1965 to 1968 he was engaged in research on semiconductors at the Technical University of Aachen, and in 1972 he was a Lecturer on numerical analysis. Since October 1972 he has been doing military service in Greece, working at the Research Center for National Defense, Athens.

John K. Petrou was born in Athens, Greece, on February 3, 1945. He received the B.S. degree in mathematics from the University of Athens in 1970.

From 1970 to 1972 he specialized in systems analysis at the Nuclear Research Center "Democritos." Since 1972 he has been with the Research Center for National Defense, Athens, Greece.

Design Verification at the Register Transfer Language Level

HAROLD HOEHNE AND ROBERT PILOTY, MEMBER, IEEE

***Abstract*—Computer description languages can be used as input to software tools which aid in the design of digital hardware structures. One of the important phases in the design process is verification. A software system is described which aids the verification process of a computer description at the register transfer language (RTL) level. It is based on the concept of concurrent simulation and comparison of the functional and the structural description of a computer. Error types, in particular consistency and semantic errors, and algorithms for their detection are discussed. The tools necessary to implement these detection procedures are outlined.**

***Index Terms*—Compiler–interpreter system, computer description language, design verification, error detection, hardware design automation, register transfer language (RTL), simulation.**

I. INTRODUCTION

IN the past decade numerous languages for the formal description of sequentially operating computer hardware units and systems have been proposed. The majority of them are designed to describe the operation of functional units in terms of the state transition of all their actual data storage elements, some even including the control storage elements [1]–[4]. They are of the register transfer (RT) type in the sense that the transfer of bit strings between groups of storage elements, i.e., registers, is considered and notationally treated as a base operation.

Other more functionally oriented languages restrict themselves to the description of the input–output behavior of hardware units [5]–[7]. With this approach, only the input–output registers or terminals of the hardware unit and their state transitions are described, while the internal storage elements necessary to implement these transitions with a reasonable balance between speed and cost remain unspecified.

The motives behind the introduction of these languages has been the assumption that an unambiguous, concise, and still readable notational system to express what a hardware system should do and how it should be built would contribute to speed up and reduce the cost of the design process of increasingly complex computers. However, the notions of just how this can be achieved remain vague.

Manuscript received December 1, 1973; revised November 14, 1974. This work was presented at the IEEE-TCCA and ACM SIGARCH workshop on Computer Description Languages, Rutgers University, New Brunswick, N. J., Sept. 6–7, 1973.
H. Hoehne is with the Firm Computer Gesellschaft Konstanz mbH, Konstanz, Germany.
R. Piloty is with the Technische Hochschule Darmstadt, Darmstadt, Germany.