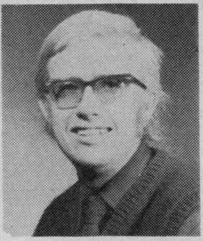
generation of uniform random numbers," in *Proc. 22nd Nat. Conf. ACM*, 1967, pp. 485–501.
[6] J. R. B. Whittlesey, "A comparison of the correlation behavior of random number generators for the IBM 360," *Commun. Ass. Comput. Math.*, vol. 9, pp. 641–644, 1968.
[7] N. Zierler and J. Brillhart, "On primitive trinomials (mod 2)," *Information and Control*, vol. 13, pp. 541–554, 1968.
[8] F. Neuman, R. Merrick, and C. F. Martin, "The correlation structure of several popular pseudorandom number generators," NASA TM-X62,275, July 1973.

Frank Neuman, for a photograph and biography see page 460 of this issue.



Clyde F. Martin received the B.S. degree in mathematics from Kansas State College, Pittsburg, and the M.A. and Ph.D. degrees in mathematics from the University of Wyoming, Laramie, in 1965, 1967, and 1971, respectively.

From 1971 to 1973 he was a National Research Council Research Associate at Ames Research Center (NASA), Moffett Field, CA, where he worked in the areas of vector valued performance indices and linear-quadratic control theory. From 1973 to September 1975 he was in the Department of Mathematics at Utah State University, Logan. His current research interest is in the application of modern mathematics to control theory in the control of aircraft.

Rounding and Truncation in Radix (−2) Systems

SHALHAV ZOHAR, MEMBER, IEEE

***Abstract*—Examination of rounding and truncation errors in radix (−2) systems establishes the following facts: 1) truncation error is 33 percent lower than in the positive radix case. 2) There is no 1-bit rounding algorithm. 3) The error bound of the standard (1-bit) positive radix rounding algorithm can be approached asymptotically by using more than one of the discarded bits. For N bits, this error bound is multiplied by $1 + (3\cdot 2^{N-1})^{-1}$.**

Realizations of the algorithms for $N = 2$–5 are presented.

***Index Terms*—N-bit rounding algorithms, radix (−2), rounding error bounds, truncation errors.**

I. Introduction

Recent increased interest in systems employing representation of numbers in radix (−2), [1]–[9] has lent importance to the problem of rounding in such systems.

It turns out that a negative radix has a peculiar effect on the rounding algorithm. In contrast with the single algorithm of the positive radix case, one finds a family of algorithms of varying precision in the present case. Our aim here is to derive these algorithms and outline their hardware realizations.

Our approach is to start with the familiar standard rounding algorithm but present it in a way which facilitates the transition to the negative radix case.

Manuscript received March 21, 1974; revised October 3, 1975. This paper presents the results of one phase of research carried out at the Jet Propulsion Laboratory, California Institute of Technology, under Contract NAS 7-100, sponsored by the National Aeronautics and Space Administration.

The author is with the Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91103

II. Rounding in Radix (+2)

Let us review the treatment of rounding in radix (+2). Assume that we are given the binary representation of the number x as follows:[1]

$$x = \sum_{k=-\infty}^{h} \xi_k 2^k \qquad (\xi_k = 0,1). \tag{2.1}$$

Suppose now that we have to represent this number in a digital device which allows the storage of only m bits to the right of the binary point. If we adopt simple truncation, we get the truncated value

$$x_m = \sum_{k=-m}^{h} \xi_k 2^k. \tag{2.2}$$

This introduces an error ϵ_m defined as

$$\epsilon_m = x - x_m \tag{2.3}$$

and satisfying

$$\epsilon_m = \sum_{k=m+1}^{\infty} \xi_{-k} 2^{-k}. \tag{2.4}$$

The bounds of this error are obtained by setting all ξ_{-k} in (2.4) to either zero or one. Hence,

$$0 \le \epsilon_m \le 2^{-m} \tag{2.5}$$

[1] x is restricted in this section to positive values as this is sufficient for our introductory purposes.

and

$$|\epsilon_m|_{\max} = 2^{-m}. \tag{2.6}$$

We introduce now the normalized error α defined by

$$\alpha = 2^m \epsilon_m. \tag{2.7}$$

Applying this to (2.5) and (2.6) we get

$$0 \le \alpha \le 1, \tag{2.8}$$

$$|\alpha|_{\max} = 1. \tag{2.9}$$

Note that while α is a function of m, its bounds are not. Hence, the convenience of basing the analysis on α.

As is well known, the situation described above can be improved upon if $\xi_{-(m+1)}$, the most significant of the discarded bits, is available to us. Let us examine how this information is utilized in the process of rounding.

Consider Fig. 1 which shows the range of α as a function of $\xi_{-(m+1)}$. The upper line represents the interval (2.8). As we have seen above, this assumes that $\xi_{-(m+1)}$ is unknown so that both edges of the interval are based on worst case assumptions. When, however, it is known that $\xi_{-(m+1)} = 0$, we get the smaller interval

$$0 \le \alpha < \frac{1}{2} \qquad (\xi_{-(m+1)} = 0), \tag{2.10}$$

represented by the middle line. Finally, the lower line is simply the interval remaining when range (2.10) is removed from range (2.8).

Note now that (2.3) implies that if we modify x_m by adding an arbitrary number to it, the effect on the ϵ_m interval would be a shift to the left corresponding to this added number. Furthermore, the practically simple modification of adding a least significance bit (2^{-m}) to x_m will result in a one-unit left shift of the α interval described in Fig. 1. Such a shift, applied to the overall interval, will change only the sign of α. However, if we can selectively shift only the lower line, the maximal error magnitude will decrease. This can be simply achieved by adopting the rounded estimate

$$\hat{x}_m = x_m + 2^{-m}\xi_{-(m+1)}. \tag{2.11}$$

Indeed, with

$$\hat{\epsilon}_m = x - \hat{x}_m, \tag{2.12}$$

$$\hat{\alpha} = 2^m \hat{\epsilon}_m, \tag{2.13}$$

we have

$$\hat{\epsilon}_m = \epsilon_m - 2^{-m}\xi_{-(m+1)}, \tag{2.14}$$

$$\hat{\alpha} = \alpha - \xi_{-(m+1)}. \tag{2.15}$$

The new situation is described in Fig. 2 and from it we see that

$$|\hat{\alpha}|_{\max} = \frac{1}{2}\,\cdot \tag{2.16}$$

In the next section we apply the above interval shifting approach to the radix (-2) case.

III. Truncation Error in Radix (-2)

The initial stages of the development in this section are exactly analogous to the positive radix case and we state here the corresponding equations without comment.

$$x = \sum_{k=-\infty}^{g} \zeta_k (-2)^k, \qquad (\zeta_k = 0,1), \tag{3.1}$$

$$x'_m = \sum_{k=-m}^{g} \zeta_k (-2)^k, \tag{3.2}$$

$$\epsilon'_m = x - x'_m = \sum_{k=m+1}^{\infty} \zeta_{-k}(-2)^{-k}, \tag{3.3}$$

$$\alpha' = 2^m \epsilon'_m. \tag{3.4}$$

In trying to evaluate now the interval of α', we notice a significant difference. Due to the alternating signs of the terms summed in (3.3), the extreme values of α' are obtained when the values of ζ_{-k} are assigned in the sequence 0,1,0,1,⋯.

For m even we get

$$\{(-2)^{-(m+1)} + (-2)^{-(m+3)} + \cdots\}$$
$$\le \epsilon'_m \le \{(-2)^{-(m+2)} + (-2)^{-(m+4)} + \cdots\}$$
$$-\frac{2^{-(m+1)}}{1 - 1/4} \le \epsilon'_m \le \frac{2^{-(m+2)}}{1 - 1/4}$$
$$-\frac{2}{3}2^{-m} \le \epsilon'_m \le \frac{1}{3}2^{-m}, \qquad (m \text{ even}), \tag{3.5}$$
$$\therefore -\frac{2}{3} \le \alpha' \le \frac{1}{3}, \qquad (m \text{ even}). \tag{3.6}$$

Similarly, for m odd, we get

$$\{(-2)^{-(m+2)} + (-2)^{-(m+4)} + \cdots\}$$
$$\le \epsilon'_m \le \{(-2)^{-(m+1)} + (-2)^{-(m+3)} + \cdots\}$$
$$-\frac{1}{3}2^{-m} \le \epsilon'_m \le \frac{2}{3}2^{-m}, \qquad (m \text{ odd}) \tag{3.7}$$
$$\therefore -\frac{1}{3} \le \alpha' \le \frac{2}{3}, \qquad (m \text{ odd}). \tag{3.8}$$

In both cases, the length of the normalized error interval is 1, just as in the positive radix case. Now, however, since part of the error interval is negative, the worst error magnitude is smaller

$$|\alpha'|_{\max} = \frac{2}{3}\,\cdot \tag{3.9}$$

This normalized error bound, while 33 percent lower than (2.9), is still greater (by 33 percent) than the corresponding entity in the case of positive radix rounding (2.16). Can we introduce a similar procedure for a negative radix?

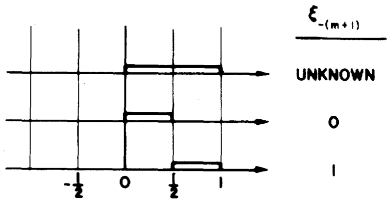


Fig. 1. Ranges of α.

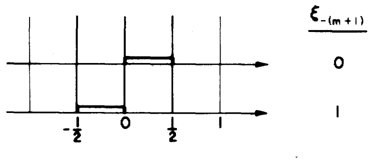


Fig. 2. Ranges of $\hat{\alpha}$.

To study this possibility, we construct now the range of α' as was done in Section II. This is shown in Fig. 3 for odd m. The upper line corresponds to interval (3.8). The next line ($\zeta_{-(m+1)} = 0$) is obtained by replacing m with $(m + 1)$ in (3.5). Note that the length of this interval is exactly half the length of the upper one. Finally, the lower line is simply the interval remaining when the ($\zeta_{-(m+1)} = 0$) subinterval is removed from the overall (upper) interval.

We try to see now whether a selective one-unit shift could bring the overall normalized error interval into the $(-\frac{1}{2},\frac{1}{2})$ range as was done in Fig. 2. It should be realized that both left and right one-unit shifts are now at our disposal since subtraction of a rounding bit is just as easy as its addition, in base (-2).

The left subinterval should not be disturbed as it lies totally inside the $(-\frac{1}{2},\frac{1}{2})$ limiting interval. On the other hand, the right subinterval extends $\frac{1}{6}$ beyond the limiting interval on the right but if we try to remedy this by applying a one-unit left shift, we get a worse extension ($\frac{1}{3}$) on the left of the limiting interval. Thus, unlike the positive radix case, knowledge of the value of $\zeta_{-(m+1)}$ does not lead to a lower error bound. This conclusion is independent of m since the even m case is just a mirror image of the present case (see Fig. 4).

We note in passing that if one blindly copies the positive radix algorithm, the "correction" yields a one-unit right shift of the right interval which increases the maximal normalized error magnitude to $\frac{5}{3}$. This is 2.5 times larger than (3.9) which applies to simple truncation.

We have established here two peculiarities of the radix (-2) system. First, the maximal truncation error magnitude is 33 percent lower than that of the positive radix case. Second, availability of the most significant bit of the discarded part of a number is of no help in reducing the error of truncation. For radix (-2) there is no simple rounding algorithm analogous to that of the positive radix system.

Yet, we know that whenever the truncation error is greater than $\frac{1}{2}(2^{-m})$, we can certainly get a better estimate by adding 2^{-m} to the truncated number.

This seeming impasse is resolved by noting that though corrective measures to reduce the error magnitude are possible, one bit ($\zeta_{-(m+1)}$) is not sufficient to determine when to apply them.

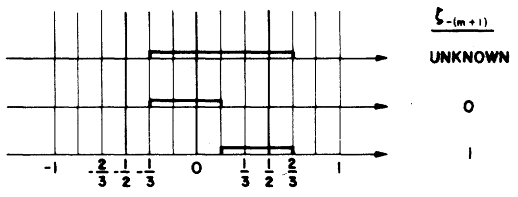


Fig. 3. Ranges of α' for odd m.

The obvious answer to our problem then is to use more input bits in the rounding algorithm. Section IV handles the practically important 2-bit case while Section V examines the advantages of using even more bits.

IV. The 2-Bit Rounding Algorithm

We have seen in Fig. 3 (m odd) how the total range of the truncation error can be subdivided into 2 equal parts, one for $\zeta_{-(m+1)} = 0$ and the other for $\zeta_{-(m+1)} = 1$. Each of these subintervals can be further subdivided into 2 equal parts according to the value of bit $\zeta_{-(m+2)}$ as shown in Fig. 5. (Ignore, for now, the broken line subinterval.)

Anticipating further developments, we introduce now the following nomenclature. The lower line subinterval in Fig. 5 is referred to as the {11} subinterval ($\zeta_{-(m+1)} = 1$; $\zeta_{-(m+2)} = 1$). In general, an α' subinterval will be denoted by the sequence of bit values $\{\zeta_{-(m+1)}\ \zeta_{-(m+2)} \cdots\}$ to which it corresponds. Thus, interval {1} of Fig. 3 is shown subdivided in Fig. 5 into intervals {10}, {11}. We already know that each one of these is half of interval {1}. The only remaining question is which half is, say, interval {11}. In order to answer this, note that the rightmost bit in the above two subintervals is $\zeta_{-(m+2)}$ and represents the number $(-2)^{-(m+2)}$. Since Fig. 5 is drawn for odd m, the above number is negative and therefore interval {11} must lie to the left of interval {10} as shown.

Examination of Fig. 5 reveals that subinterval {10} is the only one extending outside the limiting $(-\frac{1}{2},\frac{1}{2})$ interval. It protrudes $\frac{1}{6}$ on the right. Shifting it left one unit (broken line range) leads to a smaller protrusion ($\frac{1}{12}$) on the left. Hence, we do have here a mechanism for reducing the error magnitude.

The mathematical formulation of the above selective interval shift calls for a Boolean variable which has the value 1 in subinterval {10} and vanishes elsewhere. The Boolean variable

$$\rho = \zeta_{-(m+1)} \cdot \bar{\zeta}_{-(m+2)} \tag{4.1}$$

satisfies these conditions.

Utilizing ρ, denoting the reduced error $\hat{\epsilon}'_m$, and correspondingly

$$\hat{\alpha}' = 2^m \hat{\epsilon}'_m, \tag{4.2}$$

we get the following formulation of the above scheme:

$$\hat{\alpha}' = \alpha' - \rho, \qquad (m \text{ odd}). \tag{4.3}$$

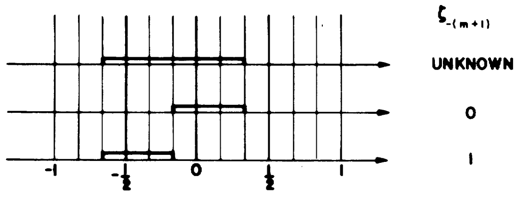


Fig. 4. Ranges of α' for even m.

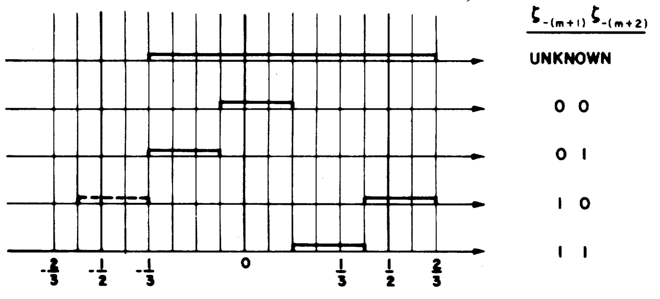


Fig. 5. The 2-bit ranges of α' for odd m.

It is convenient to describe the performance of this algorithm in comparison with the positive radix algorithm for which we have $|\hat{\alpha}|_{\max} = \frac{1}{2}$. This suggests the introduction of the normalized error excess e defined as follows:

$$e = |\hat{\alpha}'|_{\max} - |\hat{\alpha}|_{\max} = |\hat{\alpha}'|_{\max} - \tfrac{1}{2}. \tag{4.4}$$

The preceding discussion indicates that for the 2-bit algorithm

$$e = \frac{1}{12}. \tag{4.5}$$

So far, we have considered the odd m case (Fig. 5). To get the situation corresponding to m even, all we have to do is change the signs of the coordinates in Fig. 5. This leaves (4.5) intact but modifies (4.3) as follows:

$$\hat{\alpha}' = \alpha' + \rho \qquad (m \text{ even}). \tag{4.6}$$

Combining (4.3) and (4.6) and recalling (3.4) and (4.2), we get

$$\hat{\epsilon}'_m = \epsilon'_m + (-2)^{-m}\rho. \tag{4.7}$$

The corresponding estimate $\hat{x}'_m$ is given as

$$\hat{x}'_m = x - \hat{\epsilon}'_m$$
$$= x'_m - (-2)^{-m}\rho = x'_m - (-2)^{-m}(\zeta_{-(m+1)} \cdot \bar{\zeta}_{-(m+2)}). \tag{4.8}$$

Thus, the 2-bit rounding algorithm calls for a subtraction of $(-2)^{-m}$ whenever bits $\zeta_{-(m+1)}, \zeta_{-(m+2)}$ yield $\rho = 1$. Note that just as in the standard rounding algorithm case, the rounding process can be implemented simultaneously with further processing of $\hat{x}'_m$ (say, addition of another number to it). This is based on the fact that the building blocks comprising a radix (−2) adder have in addition to the "carry" input, a "borrow" input [10]. Usually, these inputs will be set to zero for the least significant bit. In our case, initializing the "borrow" input to 1, implements the subtraction required by (4.8).

An outline of an implementation of the 2-bit rounding algorithm is shown in Fig. 6. B stands for the "borrow" input referred to above. The input on the left symbolizes the possible simultaneous further processing. As indicated before, this device is characterized by $e = 1/12$. Hence,

$$|\hat{\epsilon}'_m|_{\max} = \left(\frac{1}{2} + \frac{1}{12}\right) 2^{-m} = 0.58\bar{3} \cdot 2^{-m}. \tag{4.9}$$

If we want to approach still closer to the positive radix error bound of $0.5 \cdot 2^{-m}$, we have to use more bits. This is considered in the next section.

V. Higher Order Rounding Algorithms

We have seen in Fig. 5 that the excess of the error bound over the positive radix case is caused by interval {10}. Hence, all further improvements will be associated with its subintervals. Bearing this in mind, we present this interval together with some of its subintervals in Fig. 7 (m odd).

Note that the extension of interval {10} to the left of the $\frac{1}{2}$ boundary is identical with its extension to the left of the $-\frac{1}{2}$ boundary when shifted left one unit. The 2-bit rounding algorithm works by substituting (through shifting) the shaded protrusion for the doubly longer unshaded one.

Consider now what happens when 3 bits are available. Interval {10} is now subdivided into intervals {100}, {101} and we see that a smaller excursion out of the limiting $(-\frac{1}{2}, \frac{1}{2})$ interval will result if we leave subinterval {100} in place and shift left only subinterval {101}. As is clearly evident in Fig. 7, the normalized error excess in this case (shaded region) will be only half that of the 2-bit case.

With 4 bits we find that the following subintervals should be shifted {1000}, {1011}, {1010}. However, the combination of the last two is identical with {101}. Hence, the shifting prescription: {101},{1000}. The 5-bit case should be obvious by now.

Noting that each additional bit halves the normalized error excess, we conclude that an N-bit algorithm will yield

$$e = \frac{1}{3 \cdot 2^N}, \tag{5.1}$$

that is,

$$|\hat{\epsilon}'_m|_{\max} = \left(\frac{1}{2} + \frac{1}{3 \cdot 2^N}\right) 2^{-m}. \tag{5.2}$$

The even m case is handled simply by changing the signs of the coordinates. All the above conclusions are not affected by that.

The implementations of all these algorithms differ from the one shown in Fig. 6 only in the gate structures required to realize the Boolean variable ρ. These are shown in Fig. 8 for $N = 2$–5. The logic equations on the right are simple translations of the statements on the right of Fig. 7 regarding the intervals to be shifted, that is, intervals for which we must have $\rho = 1$ (note the new symbol $\gamma_i \equiv \zeta_{-(m+i)}$).

When speed considerations are paramount, the realization of ρ for $N = 5$ should be based on the initial (unmanipulated) logic expression yielding a lower propagation delay.

VI. Statistical Properties

We consider now the mean and variance of the normalized error $\hat{\alpha}'$. Our starting point is Fig. 5 which shows the range of α' for odd m. With no rounding, we have α' uniformly distributed over the interval

$$-\frac{1}{3} \leq \alpha' \leq \frac{2}{3}. \tag{6.1}$$

Hence, its mean value which is the center of this interval is given by

$$E(\alpha') = \frac{1}{6}. \tag{6.2}$$

The application of 2-bit rounding eliminates the rightmost quarter of interval (6.1) and extends it to the left by the same amount. Thus, we have $\hat{\alpha}'$ uniformly distributed over the interval

$$-\frac{7}{12} \leq \hat{\alpha}' \leq \frac{5}{12}, \tag{6.3}$$

so that its mean value [the center of interval (6.3)] is given by

$$E(\hat{\alpha}') = -\frac{1}{12} \qquad (m \text{ odd; 2-bit rounding}). \tag{6.4}$$

A little reflection will reveal that the magnitude of the mean equals the error excess e defined in (4.4). Using the N-bit rounding error excess (5.1) and studying Fig. 7, we arrive at the final formula for the mean of $\hat{\alpha}'$ for N-bit rounding:

$$E(\hat{\alpha}') = \begin{cases} \dfrac{1}{3 \cdot (-2)^N} & (m \text{ even}) \\ \dfrac{1}{3 \cdot (-2)^N} & (m \text{ odd}). \end{cases} \tag{6.5}$$

We note that unlike positive radix rounding, the mean of the error is not zero. However, it can be made to approach zero by adopting a high-order rounding algorithm. On the other hand, simple truncation yields $E(\alpha')$ which is only ⅓ of the corresponding value for positive radix ($E(\alpha) = ½ = 3E(\alpha')$). These facts are important when error accumulation is considered. (Note that the rate of accumulation increases with the mean error.)

We turn now to the variance of $\hat{\alpha}'$. To simplify the notation we adopt here

$$\beta = \hat{\alpha}'; \bar{\beta} = E(\hat{\alpha}'). \tag{6.6}$$

For all orders of rounding algorithms, β is uniformly distributed over the interval

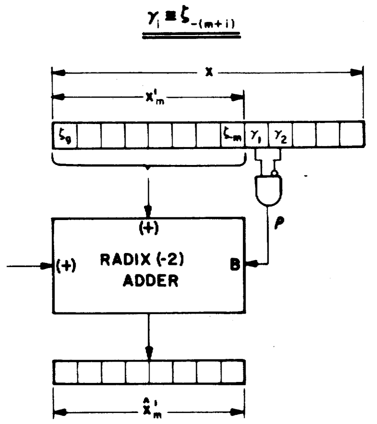


Fig. 6. Realization of the 2-bit rounding algorithm.

$$-\frac{1}{2} + \bar{\beta} \leq \beta \leq \frac{1}{2} + \bar{\beta}. \tag{6.7}$$

Hence, its probability distribution function has the value one in interval (6.7) and zero elsewhere. Therefore, σ^2 the variance of $\hat{\alpha}'$ is given by

$$\sigma^2 = \int_{-1/2+\bar{\beta}}^{1/2+\bar{\beta}} (\beta - \bar{\beta})^2 d\beta = \int_{-1/2}^{1/2} u^2 du \tag{6.8}$$

$$\sigma^2 = \frac{1}{12}. \tag{6.8}$$

This result is valid for all orders of rounding algorithms and is identical with the variance of the positive radix case.

VII. Concluding Remarks

We have seen that, for radix (−2), the maximal error magnitude is $0.\bar{6}\cdot 2^{-m}$ for simple truncation and that this can be reduced to $0.58\bar{3}\cdot 2^{-m}$ with 2-bit rounding. Increasing the number of rounding bits (N), decreases this error bound in diminishing steps to the asymptotic value $0.5\cdot 2^{-m}$.

In contrast, the positive radix case yields the bound 2^{-m} for truncation and $0.5\cdot 2^{-m}$ for (1-bit) rounding. These numbers indicate that the incentive to round will be weaker in radix (−2). Most of the applications where rounding is implemented would probably utilize the 2-bit algorithm (Fig. 6). The inverter and gate required here represent an insignificant increase in the amount of hardware. The requirement of (temporarily) preserving the extra bit ($\zeta_{-(m+2)}$) is probably a more serious drawback. In applications where more extra bits are available anyhow, it would make sense to apply the higher order algorithms in spite of the diminishing improvements.

It should be pointed out that the different characteristics of rounding in the two systems follow directly from the positioning of the relative error intervals α, α' with respect to the origin (Figs. 1 and 3). Both intervals are of length 1 and are positioned asymmetrically with

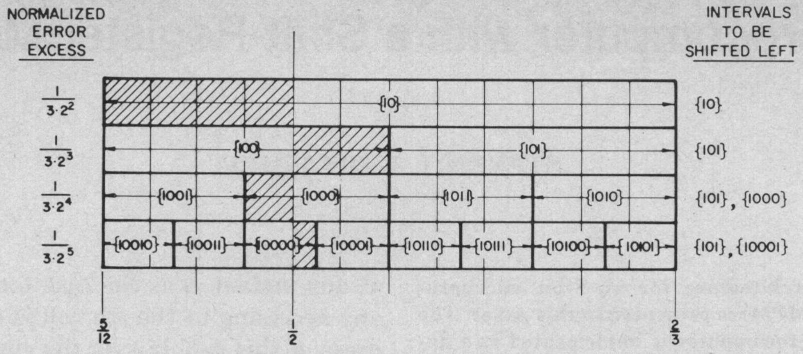


Fig. 7. Ranges of α' for higher order rounding algorithms (odd m).

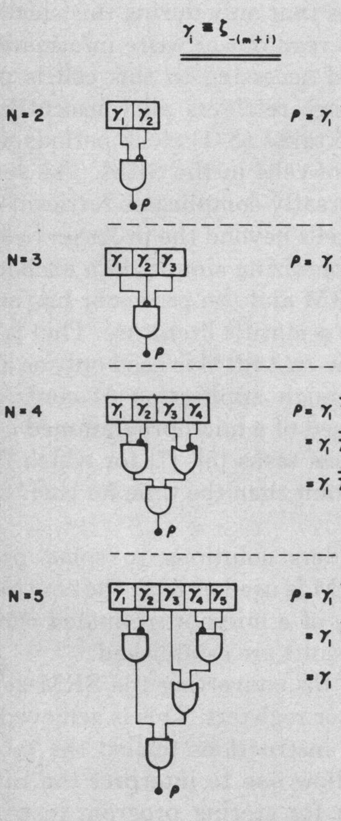

$\gamma_i \equiv \zeta_{-(m+i)}$
N = 2
$\rho = \gamma_1\bar{\gamma}_2$
N = 3
$\rho = \gamma_1\bar{\gamma}_2\gamma_3$
N = 4
$\rho = \gamma_1\bar{\gamma}_2\gamma_3 + \gamma_1\bar{\gamma}_2\bar{\gamma}_3\bar{\gamma}_4 = \gamma_1\bar{\gamma}_2(\gamma_3+\bar{\gamma}_3\bar{\gamma}_4) = \gamma_1\bar{\gamma}_2(\gamma_3+\bar{\gamma}_4)$
N = 5
$\rho = \gamma_1\bar{\gamma}_2\gamma_3 + \gamma_1\bar{\gamma}_2\bar{\gamma}_3\bar{\gamma}_4\gamma_5 = \gamma_1\bar{\gamma}_2(\gamma_3+\bar{\gamma}_3\bar{\gamma}_4\gamma_5) = \gamma_1\bar{\gamma}_2(\gamma_3+\bar{\gamma}_4\gamma_5)$

Fig. 8. Realization of ρ for higher order rounding algorithms.

respect to the origin. Rounding does not affect the length of the intervals. The only effect it has is in replacing the original intervals with the more symmetric ones $\hat{\alpha}, \hat{\alpha}'$, thus decreasing both the maximal relative error and the magnitude of its mean. In base (+2), the availability of a single extra bit leads to a perfectly symmetric $\hat{\alpha}$ interval (Fig. 2) and thus the optimal rounding is realized with just one extra bit. On the other hand, in base (−2) we cannot completely remove the asymmetry, no matter how many extra bits are available. All we can do is progressively decrease the asymmetry as more and more bits are made available. It is this fact which leads to the high-order algorithms and their asymptotic approach to optimal rounding.

We conclude with a few words regarding the extension of the above results to other negative bases. It can be shown that the generalization of Fig. 3 to a base $(-q)$, $(q = 3,4,\cdots)$ yields the following results. The total interval[2] (of $q^m\epsilon'_m$) is $(-1/(q+1), q/(q+1))$ and has a length 1. This is subdivided into q equal subintervals for the q different values of digit $\zeta_{-(m+1)}$. From this point on, the development of N-digit rounding algorithms follows in a straightforward manner.

References

[1] L. B. Wadel, "Negative base number systems," *IRE Trans. Electron. Comput.* (Corresp.), vol. EC-6, p. 123, June 1957.
[2] M. P. DeRegt, "Negative radix arithmetic," *Computer Design,* vol. 6, pp. 52–63, May 1967.
[3] S. Zohar, "Negative radix conversion," *IEEE Trans. Comput.,* vol. C-19, pp. 222–226, Mar. 1970.
[4] ——, "New hardware realization of nonrecursive digital filters," *IEEE Trans. Comput.,* vol. C-22, pp. 328–338, Apr. 1973.
[5] ——, "The counting recursive digital filter," *IEEE Trans. Comput.,* vol. C-22, pp. 338–347, Apr. 1973.
[6] ——, "Fast hardware Fourier transformation through counting," *IEEE Trans. Comput.,* vol. C-22, pp. 433–441, May 1973.
[7] P. V. Sankar *et al.*, "Arithmetic algorithms in a negative base," *IEEE Trans. Comput.,* vol. C-22, pp. 120–124, Feb. 1973.
[8] ——, "Deterministic division algorithm in a negative base," *IEEE Trans. Comput.,* vol. C-22, pp. 125–128, Feb. 1973.
[9] W. D. Little, "A fast algorithm for digital filters," *IEEE Trans. Comput.,* vol. C-23, pp. 466–469, May 1974.
[10] H. C. Wilck, Jet Propulsion Laboratory, private communication.

Shalhav Zohar (A'54–M'60) was born in Tel Aviv, Israel, on November 16, 1927. He received the B.S. and Ingénieur degrees in electrical engineering from the Technion, Israel Institute of Technology, Haifa, Israel, in 1952 and 1953, respectively, the M.S. degree in electrical engineering from the University of Cincinnati, Cincinnati, OH, in 1958, and the Ph.D. degree in electrical engineering from the Polytechnic Institute of Brooklyn, Brooklyn, NY, in 1962.

Following his graduation in 1952, he joined the Israel Signal Corps as a Communication Engineer. In 1956 he was appointed a Research Fellow at the Applied Science Research Laboratory of the University of Cincinnati. In 1958 he joined the Networks and Waveguides Group at the Microwave Research Institute, Brooklyn, NY, where he was engaged in research in the field of distributed networks. Since 1962 he has been with the Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, where is is engaged in research in the field of planetary radar and associated areas in numerical analysis and data processing.

Dr. Zohar is a member of Sigma Xi.

[2] The interval given here is for odd m. The even m interval is a mirror image of the odd interval [see (3.6) and (3.8)].