グラフィクスとCAD 54－3
(1991. 12. 20)

# 対数利用の演算法による円描画の誤差

黒河富夫

愛知工業大学　工学部　経営工学科

〒４７０－０３ 豊田市八草町八千草１２４７

あらまし

対数表現数値演算法（対数利用の演算法）は高速度の演算法である。その高速性は信号処理や画像生成で実証されている。しかし、画像生成における演算精度についての研究は殆どされていない。本論文では対数表現数値演算法による円画像の生成における誤差について解析した。すなわち、誤差の大きさ表現として相対誤差分散を用い、その表現式を導出し、シミュレーション実験により確認した。さらに、浮動小数点法によるものと理論的にまた実験的に比較した。対数表現数値演算法によるものは同語長、同ダイナミックレンジの条件で、同等以上の精度があることを示した。また、それらの結果は実際に描いた円によっても確認された。また対数表現数値演算法による円描画における誤差源の最大のものは整数から対数表現数値への変換誤差ではなく平方根演算による誤差であることを示した。

## Error Analysis of Circle Drawing with Logarithmic Arithmetic

Tomio Kurokawa

Aichi Institute of Technology, Department of Industrial Engineering
1247 Yachigusa, Yagusa-cho, Toyota 470-03 Japan

**Abstract** Logarithmic Arithmetic (LA) provides a very fast computational method. Its exceptional speed has been demonstrated in signal processing and then in computer graphics. But the precision problem of LA in computer graphics has not been fully examined. In this paper analysis is made for the problem in picture generation, in particular for circle drawing. Theoretical and experimental error analysis is made for the circle drawing. That is, some expressions are developed for the relative error variances, then they are examined by simulation experiments. Some comparisons are also done with floating point arithmetic with equivalent word length and dynamic range. The results show that the theory and the experiments agree reasonably well and that the logarithmic arithmetic is superior to or at least comparable to the corresponding floating point arithmetic with equivalent word length and dynamic range. Those results are also verified by visual inspections of actual drawn circles. It also shows that the conversion error from integer to LNS (logarithmic number system, used in LA), which is inherent in computer graphics with LA, does not make too much influence on the total computational error for circle drawing. But it shows that the square-rooting makes the larger influence.

## 1. Introduction

LA is a third type of arithmetic which uses a number system called logarithmic number system (LNS)[(1),(2)]. So far it has been studied mostly as an alternative means of computation for signal processing[(2),(3)], and then for computer graphics and image processing[(4)-(6)]. It has been verified to be very fast. The quality of LNS computation applied for picture generation, however, has not been examined fully except for some experiments[(6)]. The effect of the integer-to-LNS conversion and the square-rooting errors is also an important concern in this paper. In the following, detailed analyses and discussions are given about the computational quality of LNS applied for circle drawing.

**Definition:** In LNS, a number is expressed by a binary sequence[(2)]:

$$sd_0d_1...d_m.f_1f_2...f_n, \tag{1}$$

where s, $d_i$, and $f_j$ are 0 or 1; s and $d_0$ are the sign of the number and the exponent, respectively; d-part and f-part combined represents the exponent of the number; the base is assumed to be a constant a (greater than 1); [.] is the assumed binary point of the exponent. Then, the sequence (1) indicates the number,

$$\pm a^{\text{d-part.f-part}}. \tag{2}$$

**Method of Computation:** Due to the number expression form, many computations become simple. Let $a^x$ and $a^y$ be two numbers expressed as in the form of (2). In LA, multiplication/division, squaring/square-rooting, addition and subtraction can be done as in the expressions: (3), (4), (5) and (6), respectively.

$$a^x * a^y = a^{x+y}; \qquad a^x / a^y = a^{x-y}. \tag{3}$$
$$(a^x)^2 = a^{2x}; \qquad (a^x)^{1/2} = a^{x/2}. \tag{4}$$

$$\text{If } a^z = a^x + a^y, \text{ then } z = x + \log_a(1 + a^{y-x}), (x \geq y); \tag{5}$$
$$\text{If } a^z = a^x - a^y, \text{ then } z = x + \log_a(1 - a^{y-x}), (x \geq y). \tag{6}$$

As shown in (3), LA multiplication/division is equivalent to fixed point number addition/subtraction; squaring/square-rooting, as Eqs.(4) show, becomes equivalent to a one bit shift operation. As for Addition, if $\log_a(1+a^{y-x})$ is pre-computed as a look-up table with the table address y-x, then z can be obtained quickly as the expression (5) shows. Subtraction can be done in a similar fashion with $\log_a(1-a^{y-x})$, the additional lookup table.

## 2. Previous Work

One way to use LNS in picture generation is to make conversions before and after the necessary computations in LNS. This method is very simple but effective. In computing fixed point numbers, the numbers are first converted to logarithmic numbers (LN); then computations are done in LNS; and whenever necessary, the results are converted back to fixed point numbers. Those conversions of integer-to-LNS and vice versa can be done by look-up tables (LUT).

Digital pictures are usually defined as a two variable function f(x,y), where f is the pixel intensity and x and y are coordinate addresses. In many cases, all of f, x, and y are fixed point numbers (usually integers). Then f(x,y) can be generated using the above method. For curve drawing, the dot generation can be done by computing y=g(x) or x=h(y), where g and h are curve functions.

Using the above LNS computation method, almost any kind of picture generation is possible[(4)-(6)], as long as the picture is expressed in the computational form: f, g or h. Circle drawing is a typical example. Figure 1 is the circles generated by the method. They are generated by just computing the expression,

$$y = \sqrt{R^2 - x^2}. \tag{7}$$

## 3. Error Analysis of Circle Drawing

**Error analysis in LNS:** Since the word size to represent a number is limited, the conversion or computation error cannot be avoided. LNS is no exception. As Eq.(7) indicates, there are three kinds of error sources for the computation, which are:

(1) the conversion error from integers (x and R) to LNS,
(2) the subtraction of $R^2 - x^2$,
(3) the square-rooting of $\sqrt{R^2 - x^2}$.

In this connection, multiplication/division does not have errors in LNS, neither does the square-rooting. Including the above sources of errors, the computational model for y becomes as

$$y_l = \sqrt{[\{R(1+e_{lcR})\}^2 - \{x(1+e_{lcx})\}^2](1+e_{ls})}(1+e_{lr}), \tag{8}$$

where $e_{lcR}$, $e_{lcx}$, $e_{ls}$, and $e_{lr}$ are

(1) $e_{lcR}$ : the relative error of the conversion of R, from an integer to LNS.
(2) $e_{lcx}$ : that of x.
(3) $e_{ls}$ : the relative error of the subtraction in LNS,
(4) $e_{lr}$ : that of the square-rooting.

$y_l$ is the value just before the final conversion to integers. Then the relative error ($e_l$) of y is expressed by

$$e_l = \frac{y_l - y}{y}. \tag{9}$$

By expanding Eq.(9) and discarding the terms of the second order of $e_i$ (i=lcR, lcx, or lr) and the higher, we obtain

$$e_l \approx \frac{R^2}{(R^2-x^2)} e_{lcR} - \frac{x^2}{(R^2-x^2)} e_{lcx} + \frac{1}{2} e_{ls} + e_{lr}. \tag{10}$$

Since R can be considered a constant for a single circle, the variance of $e_{lcR}$ should be zero. If the relative errors ($e_i$) are independent of the variable x and also independent each other (naturally expected) and the sample number of x is large enough, the variance of $e_l$ can be expressed as (see also Appendix A)

$$\mathrm{VAR}[e_l] \approx E[\frac{x^4}{(R^2-x^2)^2}]E[e^2_{lcx}]+\frac{1}{4}\mathrm{VAR}[e_{ls}]+\mathrm{VAR}[e_{lr}], \quad (11)$$

where the distributions of $e_{lcx}$ and $e_{ls}$ are expected to be near uniform ; and $e_{lr}$ is with a special distribution (see Appendix B) and the expected values and variances of $e_{lcx}$, $e_{ls}$ and $e_{lr}$ are supposed to be[2] as follows:

$$E[e_{lcx}] = \frac{a^{2^{-n-1}} + a^{-2^{-n-1}} - 2}{2}, \quad (12)$$

$$\mathrm{VAR}[e_{lcx}] = \mathrm{VAR}[e_{ls}] = \frac{(a^{2^{-n-1}} - a^{-2^{-n-1}})^2}{12}, \quad (13)$$

$$\mathrm{VAR}[e_{lr}] = \frac{(a^{-2^{-n-1}} - 1)^2}{4}, \quad (14)$$

$$\mathrm{VAR}[e_i] \approx E[e^2_i], \text{ where } i=lcx, ls \text{ or } lr. \quad (15)$$

If the square–rooting is built in the conversion table (LUT) from LNS to integer, $\mathrm{VAR}[e_{lr}]$ can be deleted from Eq.(11).

**Error Analysis in Floating Point Number System:** One way to evaluate the above LNS analysis is to make comparison with floating point number systems. Let us define the specific floating number systems (FPn) for the comparison with LNS[6]. The word length is defined to be m+n+2 (same as LNS), where n is the bit length of the fraction part and m+1 is that of the exponent part. Thus it represents the value:

$$f \times a^e, \quad (16)$$

where a is 2, the fraction part is normalized so as to be between 1/2 and 1 and both parts of the fraction and the exponent have two's complement form for negative numbers. Then the floating point number system has almost equal dynamic range as LNS if m and n are assigned equal numbers, respectively.

Considering the Eq.(7), the computational model with errors becomes as

$$y_f = \sqrt{[R^2(1+e_{fmR})-x^2(1+e_{fmx})](1+e_{fs})}(1+e_{fr}), \quad (17)$$

where $e_{fmR}$, $e_{fmx}$, $e_{fs}$, and $e_{fr}$ are

(1) $e_{fmR}$ : the round off (rounding) error (relative) of the square–rooting of R,

(2) $e_{fmx}$ : that of x,

(3) $e_{fs}$ : the relative error of the subtraction in LNS,

(4) $e_{fr}$ : that of the square–rooting.

If m and n (especially n) are large enough, there will be no conversion errors. $y_f$ is the computational results just before the final conversion to integers.

The relative error of y for FPn is

$$e_f = \frac{y_f - y}{y}. \quad (18)$$

Thus, by expanding Eq.(18), and discarding the terms of second order of $e_j$ (i=fmx, fs and fr) and the higher (same procedure applied to LNS), we obtain

$$e_f \approx \frac{R^2}{2(R^2-x^2)}e_{fmR} - \frac{x^2}{2(R^2-x^2)}e_{fmx} + \frac{1}{2}e_{fs} + e_{fr}. \quad (19)$$

With Eq.(19) and the same procedure applied to LNS, the variance for FPn: $\mathrm{VAR}[e_f]$ becomes (see also Appendix C)

$$\mathrm{VAR}[e_f] \approx E[\frac{x^4}{4(R^2-x^2)^2}]E[e^2_{fmx}]+\frac{1}{4}\mathrm{VAR}[e_{fs}]+\mathrm{VAR}[e_{fr}], \quad (20)$$

where the distributions of $e_{fmx}$, $e_{fs}$ and $e_{fr}$ are not actually uniform (see Appendix B). If the above errors ($e_{fmx}$, $e_{fs}$, $e_{fr}$) are uniformly distributed, the variance is said to be

$$\mathrm{VAR}[e_{fmx}] = \mathrm{VAR}[e_{fs}] = \mathrm{VAR}[e_{fr}] = \frac{2^{-2n}}{3}. \quad (21)$$

As reported[2], Eq.(21) is about 8.35 times larger than Eq.(13). Actually, however, the distribution of $e_{fmx}$, $e_{fs}$ and $e_{fr}$ are not uniform (see Appendix B). They are considered to be nearly trapezoidal or of no errors especially depending on the size and the number R. For large n the errors $e_{fmx}$ and $e_{fs}$ are always zero.

## 4. Experimental Evaluation of the Theoretical Analysis

First, the theoretical results of LNS were evaluated. That is, Eq.(11) was numerically computed for a number of m and n. The results are shown on column A (without square–rooting) and E (with square–rooting) of Table 1. That is, A is without $\mathrm{VAR}[e_{lr}]$ in Eq.(11); and E is with it. In order to examine the results, some experiments were done to compute

$$\mathrm{VAR}[e_l] = \mathrm{VAR}[(y_l-y)/y], \quad (22)$$

where y is supposed to be the true output of Eq.(7), which is computed by 64–bit FP (considered to be error free). Eq.(22) is considered to be the relative error variance of the actual computational result of Eq.(7). The results are shown on Column B (R=180 without square–rooting), F (R=180 with square–rooting) and G (R=1500 with square–rooting) of Table 1, all of which agree fairly well with the theoretical errors on Columns A and E, correspondingly. For more analysis of LNS, see Appendix B.

As for FPn, the theoretical variance, Eq.(20) with Eq.(21) was numerically evaluated as before. The results are on Column C of Table 1. This theory is based on the assumption that the relative error is distributed uniformly. For FPn, the square–rooting error always exists. Like Eq.(22) of LNS, the experimental version of the variance for FPn can be expressed as

$$\mathrm{VAR}[e_f] = \mathrm{VAR}[(y_f-y)/y], \qquad (23)$$

where y is supposed to be the true result. With Eq.(23), the experimental relative error variances are computed and shown on Columns D (R=180) and H (for larger R=1500) of Table 1. The results on Column C (theoretical) and the results on Column D or H (experimental) do not agree very well. Those on C are about twice bigger than those on D or H. It is probably due to the fact that the error distributions ($e_{fmx}$, $e_{fs}$, $e_{fr}$) of FPn computation is not uniform. As a matter of fact, it is reported so[7], although conditions are different. It is said that they have trapezoidal distributions or the distributions which are concentrated near zero (see also Appendix B). It is also true that the shape of the distribution varies depending on R. The experimental variances, Eq.(23) on Column D and H, are smaller, about 50% smaller than the theoretical variances of the uniform model, Eq.(20) with Eq.(21), which are on Column C. The data with n=10 on Column H is omitted because it includes the conversion error. Further investigations for the analysis of FPn error distribution were made. See Appendix B, C and D.

The more important comparisons should be between Columns of B and D (without square-rooting); or G and H (with square-rooting). They are the comparisons between experimental LNS and FPn. Those on D (FPn) are ten or more times larger than those on B (LNS); and those on H (FPn) are about 5 to 6 times larger than those on G (LNS). This means that the relative error variances of FPn are larger, to that extent, than those of LNS with the equivalent word length and the dynamic range for both cases with or without square-rooting.

Looking at n, we can find that those on B (LNS without square-rooting) with n=i are very close to those on D (FPn) with n=i+2 and that those on G (LNS with square-rooting) with n=i are between those with n=i+1 and n=i+2 on H (FPn). This means that LNS without square-rooting is about 2 bits more accurate than FPn; and that LNS with square-rooting is 1 to 2 bits more accurate.

Figures 2, 3 and 4 are the circles by FPn with 12, 11, and 10 bits word, that is, a=2, m=4 and n=6, 5, and 4, respectively. While Fig.5, 6, and 7 are with square-rooting and LNS of 12, 11, and 10 bits word, that is, a=2, m=4, and n=6, 5, and 4, respectively. For a look, those by LNS are more smooth than FPn with equal bit assignments. Those of Fig.4 (FPn with n=4) have serious errors. For closer comparisons, look at Fig.2 (FPn with n=6) and Fig.6 (LNS with n=5). Both have a lot of zigzags. But those on Fig 6 may be a little smoother. Counting the number of bars for 1/8 circle of R=180, we see 14 bars for FPn and 17 bars for LNS. Therefore, those on Fig.6 are considered to be smoother than those in Fig.2 The count of the bars of Fig.7 (LNS with n=4, R=180) is 8. The above means that FPn with n=6 is between those of LNS with n=5 and n=4 but closer to LNS with n=5. This analysis agrees very well with the discussion made on the relative error variances of Column G and H of Table 1. The same discussion holds for the comparison with FPn and LNS without square-rooting. One drawback of LNS may be the gap observed at the angle of 45°. It is probably caused by the fact that the mean of Eq.(12) is not zero.

As suggested in the discussion of IEEE format (implied leading 1 of fraction) of floating point number system (IEEE–FP) in Appendix B, there is a one-bit precision gain for IEEE–FP. Therefore, for an equal word length and dynamic range, there should be no significant precision difference for circle drawing between LNS and IEEE–FP.

Error to signal ratio is a more common measurement for precision. The experimental comparison data between LNS and FPn for the circle drawing by error to signal ratio are given by the previous work[6], which gives the same kind of results. The reason to have used the relative error variance instead of error to signal ratio is that it was easier to use and that each error should equally contribute to the final evaluation value according to its relative size. In the error to signal ratio analysis, the larger data contribute more to the resulting value. Since the relative variance method have been examined numerically and visually, it should be considered justified.

## 5. Concluding Comments

The logarithmic number systems has been known effective in computer graphics. However, the extent of the accuracy of LNS applied in computer graphics has not been examined much except for the experimental study[6].

This paper has presented the theoretical and experimental analysis for circle drawing using logarithmic number systems. It has developed the theoretical expressions for the relative error variance for LNS and for floating point number systems. Comparisons have been made between the theoretical evaluation and the simulation experiments and between LNS and floating point number systems. The results of the theoretical analysis have been verified by a number of methods, the experimental total relative error variances, the individual variances and visual inspections of actually drawn circles. It has showed that LNS is more accurate than or comparable with floating point numbers systems with equivalent word length and dynamic range for circle drawing. It also has showed that the conversion did not make too much influence to the total precision but the square-rooting.

In this paper, the analysis is made only about the circle drawing. But the method is expected to be applicable to other types of lines and pictures. In future, analysis should be made for those other problems, hopefully in more general fashion.

## References

(1) Kingsbury N. G. and Rayner P.J.W.: "Digital Filtering Using Logarithmic Arithmetic," Electron. Lett., 7, 2, pp.56–58 (1971).

(2) Kurokawa T., Payne J.A. and Lee S. C.: "Error Analysis of Recursive Digital Filters Implemented with Logarithmic Number Systems," IEEE Trans. on ASSP, ASSP–28, pp.706–715 (1980).

(3) Sicuranza G. L.:"On Effective Implementation of 2–D Digital Filters Using Logarithmic Number Systems", IEEE Trans. on ASSP, ASSP–31, 4, pp.877–

885 (1983).

(4) Kurokawa T. and Mizukoshi T.: "A Fast and Simple Method for Curve Drawing---A New Approach Using Logarithmic Number Systems---", Journal of Information Processing, 14, 2, pp.144-152 (1991).

(5) Kurokawa T. and Mizukoshi T.:"Fast Method of Geometrical Picture Transformation Using Logarithmic Number Systems and its Applications for Computer Graphics", Proc. of SPIE's Visual Communications and Image Processing '90, pp.1479-1490, (1990).

(6) Kurokawa T. and Mizukoshi T.: "Computer Graphics Using Logarithmic Number Systems", The IEICE Trans. (on Information Systems), E74, 2, pp.447-451 (1991).

(7) Liu B. and Kaneko T.: "Error analysis of Digital Filters Realized with Floating-Point Arithmetic," Proc. IEEE, 57, 10, pp.1735-1747 (1969).

(8) A. D. Edgar and S. C. Lee: "Focus Microcomputer Number System, Comm. of ACM, 22, 3, pp.166-177 (1979).

## Appendix A: Development of VAR[$e_l$]

Since $e_i$ (i=lcx, ls or lr) is considered to be independent each other and also independent of x, and $(E[e_i])^2$ is very small in comparison with other terms, we have

$$VAR[e_l] \approx VAR[\frac{x^2}{R^2-x^2}e_{lcx}] + \frac{1}{4}VAR[e_{ls}] + VAR[e_{lr}]$$

$$\approx E[(\frac{x^2}{R^2-x^2}e_{lcx} - E[\frac{x^2}{R^2-x^2}e_{lcx}])^2] + \frac{1}{4}VAR[e_{ls}] + VAR[e_{lr}]$$

$$\approx E[\frac{x^4}{(R^2-x^2)^2}]E[e^2_{lcx}] - (E[\frac{x^2}{(R^2-x^2)}]E[e_{lcx}])^2 + \frac{1}{4}R[e_{ls}] + VAR[e_{lr}]$$

$$\approx E[\frac{x^4}{(R^2-x^2)^2}]E[e^2_{lcx}] + \frac{1}{4} VAR[e_{ls}] + VAR[e_{lr}], \qquad (11)$$

where

$$E[\frac{x^4}{(R^2-x^2)^2}] \approx 0.130, \text{ and}$$

$$E[e^2_i] = VAR[e_i] + (E[e_i])^2 \approx VAR[e_i].$$

## Appendix B: Individual Relative Error Distributions of LNS and FPn

According to the reference(2), the relative error distribution of LNS addition is near uniform. In the circle drawing, there are more types of computations with error. They are integer-to-LNS conversion, subtraction (equivalently addition), and square-rooting. Although the variable x in Eq.(7) is not random, but its distribution is uniform over one to $R/\sqrt{2}$. Therefore, those distributions of $e_{lcx}$ and $e_{ls}$ should be like that of the addition(2), which is near uniform. The distribution of $e_{lr}$ is different. Since the error is caused by the one-bit shift operation, there are only two cases of errors, no error or the error caused by the one-bit loss.

(a), (b) and (c) of Fig. 8 are the experimentally obtained histograms of $e_{lcx}$, $e_{ls}$ and $e_{lr}$, respectively, for R=1500, m=5 and n=10. The distributions of $e_{lcx}$ and $e_{ls}$ are considered to be near uniform in the range:

$$a^{-2^{-n-1}} - 1 \sim a^{2^{-n-1}} -1, \qquad (24)$$

which becomes -3.38E-4 ～ 3.39E-4 for a=2 and n=10, and $e_{lr}$ falls on at the two points, zero (no error) or the minus end of the range (24). If the probability is 1/2 each, which is naturally expected, the error variance becomes as Eq.(14).

(d), (e), (f) and (g) are those of $e_{fcx}$ (the relative error of integer to FPn conversion), $e_{fmx}$, $e_{fs}$ and $e_{fr}$ of FPn, respectively. The distributions are in the range:

$$-2^n \sim 2^n, \qquad (25)$$

which becomes -9.77E-4 ～ 9.77E-4 for n=10. Compared with LNS, the range of FPn is about 2.89 times larger as the reference(2) shows. $e_{fcx}$ is concentrated at zero as expected; if n>10 all fall on zero. $e_{fmx}$ looks peculiar. It is probably because of the limited number of bit patterns of square of x. Many fell on near zero (because some of x are very small compared with R); $e_{fs}$ falls on mostly on zero as reported by the reference(7).

(h) of Fig.8 is of $e_{fr}$ but by 32-bit floating point variable of MS-C program with compile option of /Op (floating point variable substitution enforced according to the source program), which is actually computed by 80387. Since it uses IEEE-FP format its precision is supposed to be 33 bits (nearly equivalent to FPn with m=7 and n=24 with rounding). It was computed for the purpose of the validity check of FPn. Its round-off method is some what different from FPn. The same types of error distributions are observed on (g) and (h). The data on Column H with n=24 were obtained by the above method of 32-bit floating point variable. There are no extraordinariness about this data for FPn.

## Appendix C: Development of VAR[$e_f$]:

$$VAR[e_f] \sim VAR[\frac{x^2}{2(R^2-x^2)}e_{fmx}] + \frac{1}{4}VAR[e_{fs}] + VAR[e_{fr}]$$

$$\sim E[(\frac{x^2}{2(R^2-x^2)} e_{fmx} - E[\frac{x^2}{2(R^2-x^2)}e_{fmx}])^2] + \frac{1}{4}VAR[e_{fs}] + VAR[e_{fr}]$$

$$\sim E[\frac{x^4}{4(R^2-x^2)^2}]E[e^2_{fmx}] - (E[\frac{x^2}{2(R^2-x^2)}]E[e_{fmx}])^2 + \frac{1}{4}VAR[e_{fs}] + VAR[e_{fr}]$$

$$\sim E[\frac{x^4}{4(R^2-x^2)^2}]E[e^2_{fmx}] + \frac{1}{4} VAR[e_{fs}] + VAR[e_{fr}], \qquad (20)$$

where

$$E[\frac{x^4}{4(R^2-x^2)^2}] \sim 0.0326, \quad \text{and} \quad E[e_{fmx}] = 0.$$

## Appendix D: Error factor analysis for LNS and Fpn.

Further experiments were done to check the theoretical relative error variances: Eqs.(11) and (20). Table 2 is the results. It was made by first experimentally computing individual computational relative error variances: conversion to LNS, subtraction and square–rooting in LNS, and squaring, subtraction and square–rooting in FPn; then obtaining each term of Eqs.(11) and (20). Columns I, J, K and L are for Eq.(11) of LNS and Columns M, N, O, and P are for Eq.(20) of FPn. L is the sum of I, J and K corresponding to Eq.(11). P is the sum of M, N and O corresponding to Eq.(20). Column L is expected to agree with Column G of Table 1; and Column P with Column H of Table 1. Indeed, they agree fairly well, respectively. Raw of n=10 are omitted in Table 2 because the conversion error occurs for FPn with n=10, which is not included in Eq.(20).

As for LNS factors (I, J and K), the major error factor seems to be that of square–rooting (K) despite the fact that the square–rooting error in LNS is caused by only one–bit shift loss. Other factors (I and J) are also not small enough to be ignored. Considering Eq.(11) and the LNS histograms of Fig.8, those data of Table 2 look reasonable. In this connection, look at n. Error size ratios between the factors of LNS do not seem to change for different n. So individual error distribution are kind of constant for different n. As a matter of fact, histograms like those of Fig.8 with different n showed that the distribution were kind of stable (not shown in this paper). That is, the distribution is nearly uniform for the conversion and the subtraction; and the distribution for the square–rooting are like that of (c) of Fig.8, that is, of two bars.

For FPn factors (M, N and O), the biggest factor is again the square–rooting. Different from LNS case, squaring (M) and subtraction (N) variances get small very rapidly and become zero as n increases. That is, the shape of those distribution pattern changes with different n. As the table shows, with n=22 and 23 there were no squaring and subtraction errors. That is, for n=22 or 23, the only error source is square–rooting. For those n, the data on Column P of Table 2 completely agree with those on Column H of Table 1. Looking at the data for n=11 to 23 of Table 2, the square–rooting is the major or dominant factor of errors, anyway. That is understood from Eq.(20) and the histograms of Fig.8.

Considering Eq.(11) and Eq.(20) and the discussions so far, the ratio (=8.35) between Eq.(13) and Eq.(21) seems to be the major factor of the error variance difference between LNS and FPn. That is, the error range of the individual relative error (conversion, subtraction, squaring or square–rooting) for LNS is 2.89 times smaller than the corresponding error of FPn with the equivalent word length and dynamic range[(2)]. That is, if the error distributions are uniform for both number systems, the variances of LNS is supposed to be 8.35 times smaller.

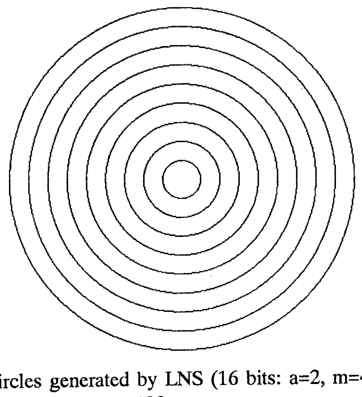
**Fig.1** Circles generated by LNS (16 bits: a=2, m=4, n=10).
R = 20, 40, 60, ..., 180.

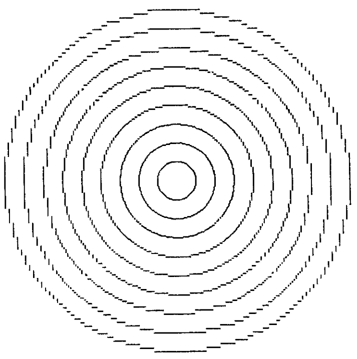
**Fig.2** Circles generated by FPn (12 bits: a=2, m=4, n=6).
R= 20, 40, 60, ..., 180.

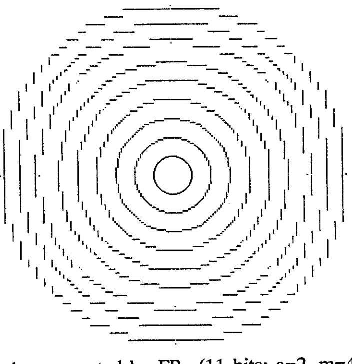
**Fig.3** Circles generated by FPn (11 bits: a=2, m=4, n=5).
R= 20, 40, 60, ..., 180.

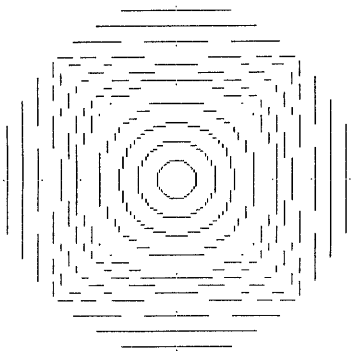

**Fig.4** Circles generated by FPn (10 bits: a=2, m=4, n=4).
R= 20, 40, 60, ..., 180.

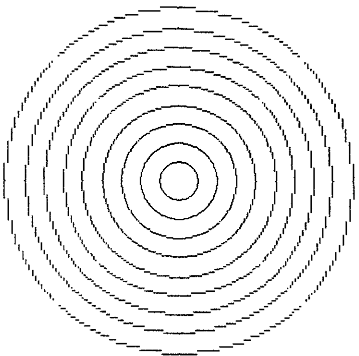

**Fig.6** Circles generated by LNS (11 bits: a=2, m=4, n=5).
R= 20, 40, 60, ..., 180.

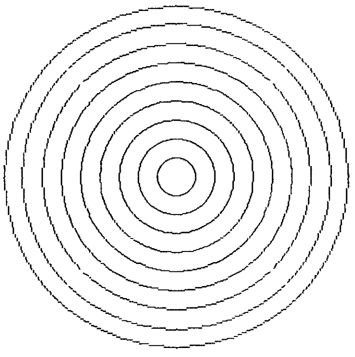

**Fig.5** Circles generated by LNS (12 bits: a=2, m=4, n=6).
R= 20, 40, 60, ..., 180.

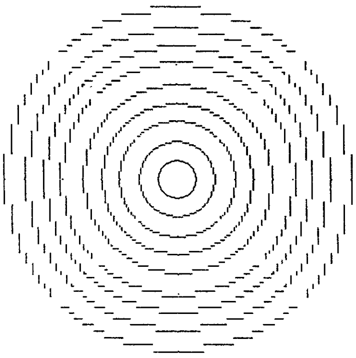

**Fig.7** Circles generated by LNS (10 bits: a=2, m=4, n=4).
R= 20, 40, 60, ..., 180.

```
-3.38e-004                                    -9.77e-004                                               -5.96e-008
↳  0          0          0                        ↳0         0        0                 0            ↳0
  73 *****   69 *****  533 ********                0         0       19 *               0             3
  38 ***     55 ****     0                         0         8       98 ***             6             7
  60 ****    51 ***      0                         0        16 *     85 ***            28 *          29 *
  57 ****    66 ****     0                         0        20 *    111 ****           61 ***        61 ***
  46 ***     44 ***      0                         0        37 *     89 ***            75 ****       75 ****
  27 **      59 ****     0                         0        67 **    16 *              78 ****       71 ****
  37 **      44 ***      0                         0        74 **    40 *              62 ***        73 ****
 102 ******* 52 ***      0                         0        83 ***   81 ***            65 ***        77 ****
  53 ****    48 ***      0                         0        92 ***   33 *              97 *****      62 ***
  88 ******  54 ****   527 ********                0       119 ****  11                51 ***        71 ****
  48 ***     62 ****     0                      1042 ******* 143 ***** 289 **********   70 ****       85 ****
  51 ***     53 ****     0                         0        48 **    29 *              60 ***        71 ****
  53 ****    51 ***      0                         0        59 **    21 *              95 *****      63 ***
  43 ***     53 ****     0                         0        50 **    22 *              80 ****       67 ***
  46 ***     57 ****     0                         0        51 **    23 *              67 ***        74 ****
  49 ***     57 ****     0                         0        74 **    19 *              63 ***        71 ****
  62 ****    46 ***      0                         0        55 **    17 *              66 ***        57 ***
  37 **      49 ***      0                         0        27 *     24 *              19 *          28 *
  66 ****    47 ***      0                         0        30 *     17 *              15 *          12 *
  24 **      43 ***      0                        18         7       16 *               2             3
↱  0          0          0                        ↱0         0        0                 0            ↱0
3.39e-004                                      9.77e-004                                                5.96e-008

  (a) elcx   (b) els   (c) elr              (d) efcx    (e) efmx  (f) efs           (g) efr    (h) ef32r
```

Fig. 8 Histograms of experimental individual relative error distributions
(R=1500, a=2, m=5, n=10).

**Table 1** Error comparison between LNS and FPn and between theory and experiment for circle drawing.

A : Theoretical relative error variance for LNS without square-rooting ;
B : Experimental relative error variance for LNS without square-rooting (R=180);
C : Theoretical relative error variance for FPn;
D : Experimental relative error variance for FPn (R=180);
E : Theoretical relative error variance for LNS with square-rooting;
F : Experimental relative error variance for LNS with square-rooting (R=180);
G : Experimental relative error variance for LNS with square-rooting (R=1500);
H : Experimental relative error variance for FPn (R=1500).
LNSR: With square-rooting for LNS; T: Theoretical evaluation; Ex: Experimental evaluation.

| | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** |
|---|---|---|---|---|---|---|---|---|
| | | **R=180** | | **R=180** | | **R=180** | **R=1500** | **R=1500** |
| **n** | **LNS:T** | **LNS:Ex** | **FPn:T** | **FPn:Ex** | **LNSR:T** | **LNSR:Ex** | **LNSR:Ex** | **FPn:Ex** |
| 10 | 1.46E−8 | 1.32E−8 | 4.08E−7 | 2.05E−7 | 4.32E−8 | 4.80E−8 | 4.50E−8 | |
| 11 | 3.65E−9 | 3.42E−9 | 1.02E−7 | 4.89E−8 | 1.08E−8 | 1.04E−8 | 1.08E−8 | 6.17E−8 |
| 12 | 9.12E−10 | 9.19E−10 | 2.55E−8 | 1.22E−8 | 2.70E−9 | 2.82E−9 | 2.76E−9 | 1.52E−8 |
| 13 | 2.28E−10 | 2.09E−10 | 6.37E−9 | 3.57E−9 | 6.76E−10 | 7.09E−10 | 6.40E−10 | 3.86E−9 |
| 14 | 5.70E−11 | 5.60E−11 | 1.59E−9 | 9.94E−10 | 1.69E−10 | 1.62E−10 | 1.70E−10 | 9.42E−10 |
| 15 | 1.43E−11 | 1.49E−11 | 3.98E−10 | 1.97E−10 | 4.22E−11 | 4.04E−11 | 4.51E−11 | 2.41E−10 |
| 22 | 8.70E−16 | 8.90E−16 | 2.43E−14 | 1.01E−14 | 2.58E−15 | 2.54E−15 | 2.54E−15 | 1.11E−14 |
| 23 | 2.18E−16 | 2.36E−16 | 6.08E−15 | 2.91E−15 | 6.44E−16 | 6.03E−16 | 6.48E−16 | 2.91E−15 |
| 24 | 5.44E−17 | | 1.52E−15 | | 1.61E−16 | | | 6.83E−16 |

**Table 2** Experimental error factor evaluation of Eq.(11) and Eq.(20).

I: Experimental evaluation of integer-to-LNS conversion: 0.13*VAR[$e_{lcx}$];
J: that of subtraction in LNS: VAR[$e_{ls}$]/4;
K: that of square-rooting in LNS: VAR[$e_{lr}$];
L: that of Eq.(11): I + J + L;
M: that of square in FPn: 0.0326*E[$e^2_{fmx}$];
N: that of subtraction in FPn: VAR[$e_{fs}$]/4;
O: that of square-rooting in FPn: VAR[$e_{fr}$];
P: that of Eq.(20): M + N + O.

| | **I** | **J** | **K** | **L** | **M** | **N** | **O** | **P** |
|---|---|---|---|---|---|---|---|---|
| | **LNS** | **LNS** | **LNS** | **LNS** | **FPn** | **FPn** | **FPn** | **FPn** |
| **n** | **conv.** | **sub.** | **s-root.** | **Eq.(11)** | **square** | **sub** | **s-root** | **Eq.(20)** |
| 11 | 1.14E−9 | 2.38E−9 | 7.16E−9 | 1.07E−8 | 1.19E−9 | 1.33E−8 | 4.88E−8 | 6.33E−8 |
| 12 | 3.41E−10 | 5.80E−10 | 1.79E−9 | 2.71E−9 | 2.90E−10 | 3.19E−9 | 1.12E−8 | 1.47E−8 |
| 13 | 6.65E−11 | 1.48E−10 | 4.45E−10 | 6.63E−10 | 7.02E−11 | 8.36E−10 | 2.96E−9 | 3.86E−9 |
| 14 | 1.77E−11 | 3.70E−11 | 1.12E−10 | 1.66E−10 | 1.63E−11 | 1.95E−10 | 7.05E−10 | 9.17E−10 |
| 15 | 4.24E−12 | 9.77E−12 | 2.80E−11 | 4.20E−11 | 3.73E−11 | 5.25E−11 | 1.78E−10 | 2.34E−10 |
| 22 | 2.80E−16 | 5.60E−16 | 1.71E−15 | 2.55E−15 | 0 | 0 | 1.11E−14 | 1.11E−14 |
| 23 | 6.96E−17 | 1.44E−16 | 4.26E−16 | 6.40E−16 | 0 | 0 | 2.91E−15 | 2.91E−15 |