Electronic Desk Top Calculator Oriented Arithmetical Method of Transcendental Functions

Atsushi Asada and Yukihiro Yoshida

1. Introduction

Usually, the Maclaurin's expansion (1) or the best approximation (2) are well known as a programming algorithm for programmable calculators. In the scientific calculator whose transcendental functions were pre-stored, Maclaurin's expansion, Pseudo Division and Pseudo Multiplication Process (3), CORDIC method (4) (5) or STL (Sequential Table Look Up) method (6) are frequently utilized to execute the microprogram. From the view point of the execution time, Oyanagi, Watanabe and Hagiwara have reviewed the above algorithms listed in the reference. These methods can attain higher accuracy and faster execution of the computations. However, when these methods are implemented with the BCD code, more memory area should be required to store the tables of pre-computed constants than with the pure binary design.

In the recent situation in which such electronic component as Large Scale Integration can be utilized for the calculator design, the algorithm which the calculator can be designed with a few chips of LSI is being required except for the high class programmable calculators.

The algorithm described in this paper is proposed for computing the transcendental functions (exponential function, trigonometric function, etc.) practically. Although the proposed algorithm basically depends upon the Euler's method, it has a feature that the numerical integration is incrementally conducted with an increment h by exchanging the approximate equations at the odd and the even step respectively.

In order to get the inverse functions, the accumulation of increment is used as the first approximate value required for the Newton's method. Prior to the calculator's design the proposed algorithm was simulated by IBM computer to determine the register length and to investigate the error trend of each function. The feasibility of this algorithm has been confirmed by designing the calculator.

2. Concept of Algorithm

When the function values are required for the set of $x=x_0, x_0+h, \text{-----}, x_0+nh$ (h: increment), the function f(x) is expanded for $x_{n+1}=x_n+h$ as follows.

$$f(x_n+h) = \sum_{m=0}^{\infty} \frac{f^{m}(x_n)}{m!} h^m \qquad (1)$$

where $x_n \triangleq x_0+nh$ $(n=0,1,2,\text{-----},N)$

This paper first appeared in Japanese in Joho-Shori (Journal of the Information Processing Society of Japan), Vol. 15, No. 11, (1974), pp. 850~856.

* Business Machine Group, Industrial Instruments Division, Sharp Corporation

If $f_n(x)$ is placed by f_n, f_{n+1} is given by the equation (2)

$$f_{n+1} = \sum_{m=0}^{\infty} \frac{f_n^m h^m}{m!} \tag{2}$$

When the equation (2) is approximated by the 1st term through 3rd term, each f_{n+1} becomes

$$f_{n+1} = f_n + hf_n' + \frac{1}{2}h^2 f_n'' \tag{3}$$

The equation (3) is one of form of the numerical integration.

Instead of the equation (3) this paper proposes the approximate form modified from Fig. 1.

because of

$$\int_{x_{2n-2}}^{x_{2n-1}} f'(x)\,dx \cong h(f_{2n-2}' + hf_{2n-2}'') \tag{4}$$

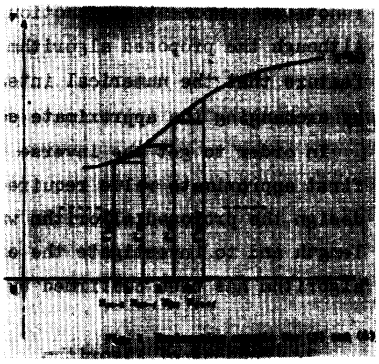
$$\int_{x_{2n-1}}^{x_{2n}} f'(x)\,dx \cong hf_{2n-1}' \tag{5}$$

$$\begin{aligned} f_{2n-1} &= f_{2n-2} + h(f_{2n-2}' + hf_{2n-2}'') \\ f_{2n} &= f_{2n-1} + hf_{2n-1}' \end{aligned} \tag{6}$$

$$\begin{aligned} f_{2n-1} &= f_{2n-2} + hf_{2n-2} \\ f_{2n} &= f_{2n-1} + h(f_{2n-1}' + hf_{2n-1}'') \end{aligned} \tag{7}$$

are realized.

The equations denoted by the above equation (6) or (7) are approximate equation f_i for odd number i and for even number i. These equations are more effective to reduce repetition of the numerical integration than the case of equation (3). It is estimated that the accuracy is about the same as the equation (3).



When the inverse function $x=f^{-1}(y)$ for given h is computed by equation (6) or (7), the following procedures are considered in case of monotone function.

Procedure 1.

The first step is to choose $x_i (0 \leq i \leq N)$ so as to satisfy the condition $f(x_i) < y$ (f: monotone increasing function) or the condition $f(x_i) > y$ (f: monotone decreasing function), and then goes to procedure 2.

At this time, makes $\xi \leftarrow x_i + h$ $(=x_{i+1})$

Procedure 2.

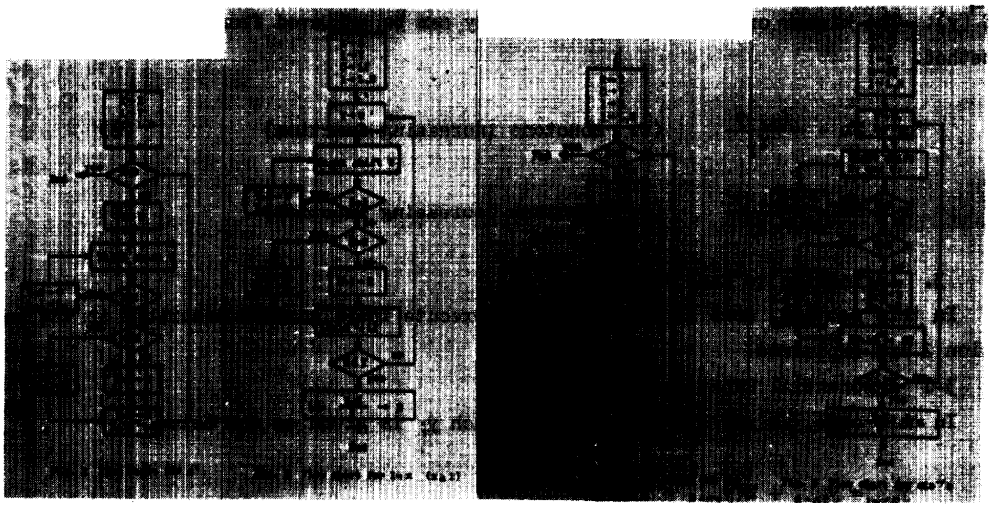
If the condition $f(\xi) > y$ (f: monotone increasing function) or the condition $f(\xi) < y$ (f: monotone decreasing function) is realized, obtained $\xi (\doteq x)$ is approximate solution for $f^{-1}(y)$. If the condition $f(\xi) < y$ (f: monotone increasing function) or the condition $f(\xi) > y$ (f: monotone decreasing function) is realized, the procedure 2 is repeated with new ξ $(\xi \leftarrow \xi + h)$.

The accuracy obtained by the above procedure 1 and 2 is obviously the same order

3-4 Execution Sequence

Hereunder, the execution which was based upon the concept explained in the above section is represented with Fig. 2 through Fig. 5. In each flow chart, the increment h is equal to 10^{-m} and K is a constant number for x. X,Y,Z,W_1 and W_2 are set of the register to be used for execution.

The L and M are the conditional memory to change the arithmetic equations at the odd step or the even step.



4. Simulation

Prior to the application of this algorithm, it is necessary to determine register length to be prepared for calculator design. At the same time, important point is to evaluate the accuracy of the each transcendental function distinguishly. For this purpose, each function has been simulated as outlined in the following.

4-1 Outline of Simulation

To investigate the relation between accuracy and register length, the behaviour of each register which stores function value should be properly simulated.

The simulation has been conducted to know what the length of register is optimum.

One of the factors related to accuracy is register length. It is considered that if the increment of $K=10^{-m}$ were selected to get h order accuracy, the register length would be 4m. Three kinds of register lengths 4m-1, 4m and 4m+1 were assumed to simulate that the length 4m is appropriate. The simulator has been constructed with pseudo variable length register to input data of the length. The set of variable number with suffix, X(M),-----,X(1) constructs one of the each necessary register, and the contents of the register is operated in this simulator as though each variable number with suffix were one decimal number.

In order to properly execute the flow chart mentioned-above, the simulator involves the equivalent arithmetic instructions which are such hardwares as trasfer, right shift, judgement, add/sub and division. The arithmetic procedure is controlled by these basic instructions. After the completion of simulation flow which was finalized by changing the register length and the repetition time, the contents of register, which store the function value and the position of decimal point, were printed out.

Thus, it is possible to correct the first approximate value by Newton's method. In other words, for approximate value obtained by the above mentioned procedure,

$$J \triangleq \begin{cases} 2N-2 \text{ (Approximate value is finally computed from } f_{2n} \\ 2N-1 \text{ (Approximate value is computed from } f_{2n-1} \end{cases}$$

are defined.

The equation (8) or (9) becomes approximately the solution of the inverse function $f^{-1}(y)$. The higher order of h in the accuracy can be achieved from the Newton's method.

$$x = Nh - \frac{f_{Nj}-y}{f'_j} \qquad (f: \text{ monotone increasing function}) \tag{8}$$

$$x = Nh + \frac{f_{Nj}-y}{f'_j} \qquad (f: \text{ monotone decreasing function}) \tag{9}$$

3. Aplication to Each Function.

In this section, some of examples which execute the algorithm described in the section 2 are explained.

3-1 Exponential Function ($y=e^x$)

In this case, as $y=y'=y''$ is concluded, each y_j is given as the below.

$$\begin{aligned} y_{2n-1} &= y_{2n-2} + h(y_{2n-2} + hy_{2n-2}) \\ y_{2n} &= y_{2n-1} + hy_{2n-1} \\ y_J &\cong e^x \end{aligned} \tag{10}$$

3-2 Trigonometric Function ($y=\sin x$, $z=\cos x$)

In this case, as it is similar to the exponential function, each y and z_j are given as the below.

$$\begin{aligned} y_{2n-1} &= y_{2n-2} + h(z_{2n-2} - hy_{2n-2}) \\ y_{2n} &= y_{2n-1} + hz_{2n-1} \\ z_{2n-1} &= z_{2n-2} - h(y_{2n-2} + hz_{2n-2}) \\ z_{2n} &= z_{2n-1} - hy_{2n-1} \\ y_J &\cong \sin x \\ z_J &\cong \cos x \end{aligned} \tag{11}$$

3-3 Hyperboric Function ($y=\sinh x$, $z=\cosh x$)

In this case, the following equation is obtained.

$$\begin{aligned} y_{2n-1} &= y_{2n-2} + h(z_{2n-2} + hy_{2n-2}) \\ y_{2n} &= y_{2n-1} + hz_{2n-1} \\ z_{2n-1} &= z_{2n-2} + h(y_{2n-2} + hz_{2n-2}) \\ z_{2n} &= z_{2n-1} + hy_{2n-1} \\ y_J &\cong \sinh x \\ z_J &\cong \cosh x \end{aligned} \tag{12}$$

The simulated result was compared with the numerical value generated by the IBM-360 computer, and it was confirmed that the result is proper to apply to calculator's design taking the practical use for design into account.

The m=3 was chosen as one of value m. All of the functions described in the section 2 have been simulated in accordance with the above method. It was made sure that the register length of 4m was proper to obtain the h order accuracy.

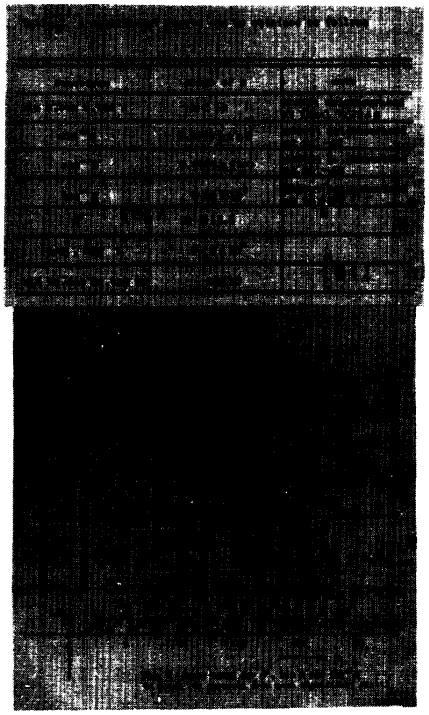
4-2 Error trend by Simulation

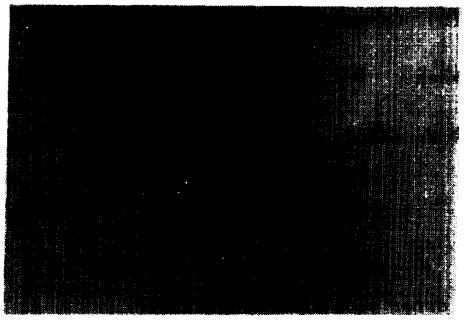
The relative error is meaningless in case that true value T for x is equal to zero. For example, when y=sin x

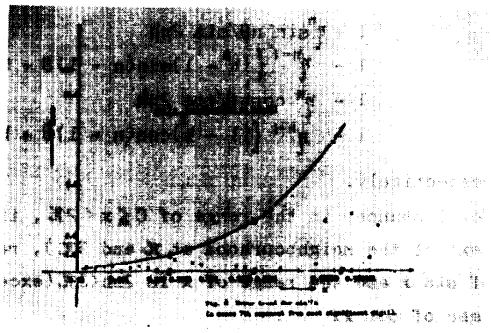
$$\left|\frac{\Delta y}{y}\right| = \left|\frac{\Delta x \cos x}{\sin x}\right| \qquad (13)$$

Since the function sin x is nearly equal to zero at the neighobourhood of $x=n\pi$, $|\Delta y/y|$ becomes too large number. So far as the register length is limited, it is very difficult to assure constant accuracy by the relative error. The same figure causes in the logarithmic function at the neighbourhood of x=1. It depends upon particular characteristics of these function. A calculator by the proposed algorithm was designed with twelve digits register and with fixed decimal point. The error trend was evaluated by the number of significant digits, which are similar to the case by the absolute error. The range of maximum error can be theoretically denoted. However, it is far more important to investigate actual error trend in the calculator design. Considering the application of this algorithm, it was simulated with m=3 from the standpoint of practical design.

The range of x, whose significant digits 6 is obtained, is summarized in the following table 1.







The error trend for e^x, sin x and $\tan^{-1}x$ are plotted at Fig. 6. In case of e^x, the trend of over correction or under correction by this algorithm is shown remarkably. The error trend for tan x which was plotted at Fig. 7 is improved in the neighbourhood of $x=\pi/2$.

It depends on the reason that 0.99999987 was selected as it becomes $\alpha \equiv 1$ experimentally.

That is, when $x=\alpha X$

$$\left|\frac{\Delta y}{y}\right| = \left|\frac{2\alpha\Delta x}{\sin 2\alpha x}\right| \tag{14}$$

The equation (14) is obtained from y=tan x. The error trend of $\sin^{-1}x$ is descontinuous as plotted at Fig. 8. It depends on the correction affect by the equation (8). The error trend of ln x is satisfactory in the range of $0 < x < 10^7$ as shown in the table 1.

The error equations for e^x, sin x and cos x are theoretically led as the below.

(a) Case of e^x :

Now, when $\gamma \triangleq (1 + h)(1 + h + h^2)$

each y_i are

$$\begin{aligned} y_{2n} &= e^{n\ln\gamma} \\ y_{2n-1} &= (1 + h + h^2)e^{(n-1)\ln\gamma} \end{aligned} \tag{15}$$

The error equations (relative) are

$$1 - e^{n(\ln\gamma - 2h)}, \quad 1 - (1 + h + h^2)e^{(n-1)\ln\gamma}$$

respectively.

For instance, if $h=10^{-3}$ when the range of x is $2nh < 8$, the relative error becomes less than 10^{-6}.

(b) Cases of sin x and cox x:

When $\theta \triangleq \tan^{-1}\frac{h(h^2-2)}{1-2h^2}$, $\eta \triangleq \sqrt{1 + h^6}$

each y and z are

$$\begin{aligned} y_{2n} &= -\eta^n \sin n\theta \\ y_{2n-1} &= \eta^{n-1}\left\{(h^2 - 1)\cos(n - 1)\theta + h\cos(n - 1)\theta\right\} \\ z_{2n} &= \eta^n \cos n\theta \\ z_{2n-1} &= \eta^{n-1}\left\{(1 - h^2)\cos(n - 1)\theta + h\sin(n - 1)\theta\right\} \end{aligned} \tag{16}$$

The error equations (relative)

are

$$\begin{aligned} &1 + \eta^n \sin n\theta/\sin 2nh \\ &1 - \eta^{n-1}\left\{(h^2 - 1)\sin(n - 1)\theta + h\cos(n - 1)\theta\right\}/\sin(2n - 1)h \\ &1 - \eta^n \cos n\theta/\cos 2nh \\ &1 - \eta^{n-1}\left\{(1 - h^2)\cos(n - 1)\theta + h\sin(n - 1)\theta\right\}/\cos(2n - 1)h \end{aligned}$$

respectively.

For instance, in the range of $0 \leq x < 2\pi$, if $h=10^{-3}$, when the range of x is $2nh \quad 2\pi$ (except the neighbourhood at π and 2π), relative error becomes less than 10 in case of sin x and the range of x is $2nh<2\pi$ (except the neighbourhood at $\pi/2$ and $3\pi/2$) in case of cos x.

5. Conclusion

The algorithm proposed in this paper is mainly considered for the pre-stored function of the calculator. Its objective is to get the function value by simple method.

As studied in this paper, it is difficult to theoretically analize the error on this method, because the numerical integrations are proceeded with the increment step by step. Therefore, in case that the calculator's design is performed, the simulation can be done to determine the register length and to investigate the error trend because the theoretical analysis is insufficient. Considering the application to calculator's design, the accuracy was investigated in detail with the value of m=3.

In the past, the study on the numerical analysis had been of importance for the software. For this reason a large number of electronic cimponents have been required for the design of the calculator with the transcendental functions. But if the calculator which can perform the basic instruction of each step shown in the flow chart of Fig. 2 through Fig. 5 is equipped with hardware, it is feasible to reduce electronic omponents required for the design.

As well known in general, it is necessary to use this method after the scaling was processed, but its points are eliminated, since the emphasis is mainly placed on the contents of the concept for the algorithm in this paper. Only the necessary data for explanation are attached, because the volume of data forosimulation results is huge. Please note that such method can be considered that $\ln x = \int_1^x \frac{dt}{t}$ can be obtained by the numerical integration (Simpson's formula) and then e can be obtained from ln x by Newton's method.

Acknowledgement

In closing this paper, authors deeply appreciate Prof. Dr. Ozaki, Assistant Prof. Dr. Shirakawa (they belong Osaka University) and Mr. I. Washizuka (he is a chief engineer of Sharp Corp.) who kindly advised us of study on this algorithm, and also appreciate Messrs. T. Izaki, T. Maegawa, Y. Takeda and H. Nakagawa (they are engineers of Sharp Corp.) who concerned with simulation and design.

References

1) A. Ralston & H. S. Wilf: Mathermatical Method for Digital Computers, Vol. 1, P.9, John Wiley & Sons, Inc. 1960

2) S. Hitotumatu: Suchi Kaiseki (Neumerical Analysis), P.97, Zeimu-Keiri-Kyokai, Tokyo (1971)

3) J. E. Meggitt: Pseudo Division and Pseudo Multiplication Process, P.210, IBM Journal April 1962

4) J. S. Walther: A nuified Algorithm for Elementary Functions, Vol.38, P.379, AFIPS. Conference Proceedings, SJCC 1971

5) J. E. Volder: The CORDIC Trigonometric computing Technique, P.330, IRE Trans. E. C. September, 1959

6) W. H. Specker: A Class of Algorithms for Ln x, Exp x, cos x, tan x and cot x P.85 IEEE Trans. E. C February, 1965

7) Oyanagi, Watanabe & Hagiwara: Algorithms for Elementary Functions by Micro-Program, P.171, 14th National Convention IPSJ, 1973