An Electronic Digital Polynomial Root Extractor

R. R. JOHNSON†

Summary—**Many mathematical techniques exist for factoring algebraic polynomials. Most require much computation and programming and are practical only for large machine computers. A different approach to the general problem is described. The mathematical method is an adaptation of a Taylor series approximation used to connect the problem and its formulation with a special machine implementation. The result is a simple digital computer capable of extracting the complex roots of an *n*th degree polynomial.**

Introduction

THE ADVENT OF the large-scale digital computer has resulted in a general emphasis on numerical methods. This emphasis has led to many applications of digital techniques to problems having special characteristics; computers designed to capitalize on these characteristics can obtain solutions rapidly and with considerable savings in computing equipment. Reductions in preparation and programming time can result from machines designed to handle problems appearing repeatedly in practice.

In scientific computations, one universally belabored problem is that of factoring algebraic polynomials.[1–3] Many analog devices[4–10] and many mathematical methods[11–18] have been developed to solve this problem. A special-purpose computer has been designed and constructed at the California Institute of Technology to reduce the programming and scheduling delays involved in placing polynomials in large-scale machines. Other reasons for its construction were the development of new computer techniques and the desire for a machine having more versatility and accuracy than any of the devices now available.

This computer is designed to handle polynomials up to the sixteenth degree. Operating in the binary number system, it requires 20 flip-flops and approximately 200 germanium diodes. The operating memory is one circulating register with one clock channel on a small magnetic drum. Input is bit by bit, using a pair of switches, and output is visual on an oscilloscope. The only modifications necessary to extend this to higher degree polynomials would be a larger drum or a higher pulse density. The computer obtains the complex roots of polynomials having real or complex coefficients with an accuracy of approximately six decimal digits. Solution times average about 16 seconds per root.

Principle of Operation

The general method is that of evaluating a polynomial in the complex domain. The computer is designed to: (1) Evaluate the polynomial for successive increments in its complex argument, and (2) select the complex increment that always decreases the absolute value of the polynomial. A brief description is given of the problem preparation requirements and of the numerical accuracies attainable.

Mathematical Techniques

The computer generates values of the polynomial by repeated steps of Δ in its complex argument. Before each step the value

$$\pm\delta = \Delta$$

$$\pm j\delta = \Delta$$

is chosen such that the step will diminish the absolute value of the polynomial. In this fashion the argument is modified as the computer assumes the direct path to the

† California Institute of Technology, Pasadena, Calif.

[1] E. T. Whittaker and G. Robinson, "The Calculus of Observations," Blackie and Son, London, ch. VI; 1940.
[2] Fr. A. Willers, "Practical Analysis," Dover Press, New York, ch. IV; 1948.
[3] A. S. Householder, "Principles of Numerical Analysis," McGraw-Hill Book Co., Inc., New York, ch. 3; 1953.
[4] S. L. Brown and L. L. Wheeler, "Mechanical methods for graphical solutions of polynomials," *Jour. Franklin Inst.*, vol. 231, pp. 223–243; March, 1941.
[5] B. O. Marshall, Jr., "Electronic isograph for roots of polynomials," *Jour. Appl. Phys.*, vol. 21, pp. 307–312; April, 1950.
[6] E. O. Willoghby, G. A. Rose, and W. G. Forte, "Analog Computers to Solve Polynomial Equations with Real Coefficients," Proc. Conference on Automatic Computing Machinery, Commonwealth Scientific and Industrial Research Organization August, 1951.
[7] D. Herr, "Two new economical computers for design and analysis of servomechanisms, networks, amplifiers, and analogous dynamical systems," *Amer. Soc. Nav. Eng. Jour.*, vol. 63, pp. 950–971; November, 1951.
[8] F. W. Bubb, Jr., "Circuit for generating polynomials and finding their zeros," Proc. IRE, vol. 39, pp. 1556–1561; December, 1951.
[9] L. Löfgren, "Analog computer for roots of algebraic equations," Proc. IRE, vol. 41, pp. 907–913; July, 1953.
[10] M. L. Morgan, "A Computer for Algebraic Functions of a Complex Variable," Ph.D. thesis, Calif. Inst. Tech.; 1954.
[11] S. N. Lin, "A method of successive approximations of evaluating the real and complex roots of cubic and higher order equations," *Jour. Math. Phys.*, vol. 20, pp. 231–242; January, 1941.
[12] P. A. Samuelson, "Iterative computation of complex roots," *Jour. Math. Phys.*, vol. 28, pp. 259–267; January, 1949.
[13] Y. L. Luke and D. Ufford, "On the roots of algebraic equations," *Jour. Math. Phys.*, vol. 30, pp. 94–101; January, 1951.
[14] D. B. Steinman, "Simple formula solves all higher degree equations," *Civil Engrg.* (N. Y.), vol. 21, pp. 44–45; February, 1951.
[15] F. W. J. Oliver, "Evaluation of zeros of high degree polynomials," *Phil. Trans. Roy. Soc. London*, vol. 244, pp. 385–415; April, 1952.
[16] H. E. Salzer, "Calculating zeros of polynomials by method of Lucas," *Res. Jour.*, U. S. Bureau of Standards (RP 2348), vol. 49, pp. 133–134; August, 1952.
[17] J. F. Koenig, "On zeros of polynomials and degree of stability of linear systems," *Jour. Appl. Phys.*, vol. 24, pp. 476–482; April, 1953.
[18] L. Tasny-Tschiassny, "Location of roots of polynomial equations by repeated evaluation of linear forms," *Quart. Appl. Math.*, vol. II, pp. 319–326; October, 1953

nearest root. When it reaches a minimum value for the polynomial, the normal Δ selection causes the computer to encircle the point of minimum absolute value.

The single memory channel contains 19 word positions. The real and imaginary components of the polynomial are in the first word; in each of the following 16 words is located the corresponding derivative of the polynomial (see Fig. 1).

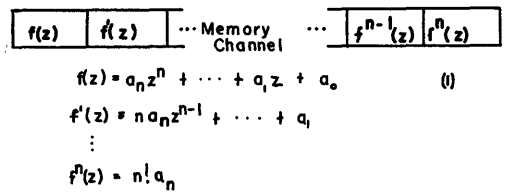

Fig. 1—The polynomial and its computer representation.

These derivatives, evaluated at some convenient point such as the origin, comprise the initial input data. The computational principle is to evaluate each derivative at an incremental distance Δ away from the initial co-ordinate (see Fig. 2). This approximation is the first term of a Taylor series expansion of the polynomial. All derivatives are recomputed for each step Δ using this principle.

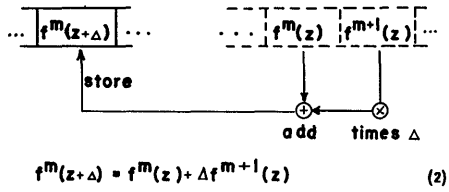

Fig. 2—The computation principle.

For the initial root location an improved approximation formula is used with a larger value of δ. Higher-order terms could be taken from the Taylor series, but a more elegant technique is to use part of each newly computed derivative:

$$f(z + \Delta) = f(z) + \tfrac{1}{2}\Delta f'(z) + \tfrac{1}{2}\Delta f'(z + \Delta). \quad (3)$$

The $(m-1)^{st}$ derivative is computed:

$$f^{(m-1)}(z + \Delta) = f^{(m-1)}(z) + \tfrac{1}{2}\Delta f^{(m)}(z) + \tfrac{1}{2}\Delta f^{(m)}(z + \Delta). \quad (4)$$

Repeated approximations permit the computer to evaluate its polynomial for any complex argument. Coupled with the direction decision elements, the computer follows the shortest path to the nearest zero of the polynomial.

Several alternatives exist for finding all n roots. The method used is to continue the normal computation cycle but to force a desired Δ selection. The operator chooses the direction in which he wishes to look for roots and forces the computer to move in that direction. Released, the computer seeks the nearest root. Complex conjugate roots can be eliminated immediately.

Direction Decision

Consider the simplified approximation formula; the real and imaginary components for each step are found:

$$f(z) = u(z) + jv(z).$$

For $\Delta = \pm\delta$,

$$\begin{aligned} u(z + \Delta) &= u(z) \pm \delta u'(z), \\ v(z + \Delta) &= v(z) \pm \delta v'(z). \end{aligned} \quad (5a)$$

For $\Delta = \pm j\delta$,

$$\begin{aligned} u(z + \Delta) &= u(z) \mp \delta v'(z), \\ v(z + \Delta) &= v(z) \pm \delta u'(z). \end{aligned} \quad (5b)$$

The selection of Δ is that which will assure

$$|f(z + \Delta)| \leqq |f(z)|.$$

For example, if both u and v are positive and their first derivatives are positive, $\Delta = -\delta$. However, if u and v are positive and their first derivatives differ in sign, $\Delta = \pm j\delta$ is the proper selection.

A simple case illustrates the behavior of the direction control when z approximates a root to within an amount δ. Consider all signs initially plus. The sequence of decisions prescribed by the rules above forces the computer to encircle the root. The computer remains in this mode until forced in another direction.

Problem Preparation

There are two restrictions placed on the polynomial. It is necessary to convert the coefficients to binary numbers and to limit all numbers to absolute values less than 1.0. There are 30 significant binary places available for the real and imaginary components of each derivative. Numbers are absolute value with sign when positive and zero's complements with sign when negative (see Fig. 3).

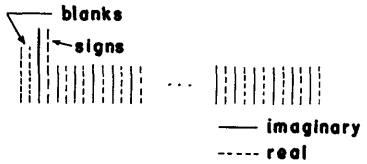

Fig. 3—The biplexed word structure for each derivative.

To insure that $|z| \leqq 1$ for all roots, it is necessary to compute the radius of the contour in the z plane which encloses all n zeros of the polynomial

$$R > \left|\frac{a_k}{a_n}\right| - 1, \quad (6)$$

where $|a_k|$ is the largest coefficient in the polynomial. A new argument is defined:

$$w = \frac{z}{R}, \quad (7)$$

which transforms the polynomial into

$$f(w) = a_n(R)^n w^n + \cdots + a_1(R)w + a_0. \quad (8)$$

The computer obtains the roots of (8). The roots of the original polynomial are found using (7). To insure that the absolute value of the n derivatives never exceed 1.0, it is necessary to divide $f(w)$ by the largest coefficient:

$$F(w) = \frac{f(w)}{|\, 2n!R^i a_i \,|}, \tag{9}$$

$$= A_n w^n + \cdots + A_1 w + A_0, \tag{10}$$

where i is the index of the largest coefficient in (8).

These computations must be done manually before inserting the derivatives into the computer. This is done to retain the computational simplicity of the computer. No multiplications or divisions are provided internally.

D. Accuracy

Each step in the approximation of the function and its derivatives has a truncation error

$$\epsilon = \sum_{i=m+3}^{n} \frac{\Delta^i}{2i!}(i-2)f^{(i)}(z). \tag{11}$$

This is the error in the approximation to the mth derivative. The error in the function itself is of the order of

$$\frac{\Delta^3}{12} f^{(3)}(z).$$

Two values of δ are used: 2^{-10} and 2^{-20}. The computation sequence is to locate the root with the larger and refine it with the smaller δ. With the coarse δ, the truncation error at each step is of the order of 2^{-34}. The total error in 2^{10} steps is 2^{-24}. The root is refined using approximation (2) with $\delta = 2^{-20}$. In 2^{10} steps this produces an error of 2^{-32}.

Roundoff errors may propagate to the twenty-first binary position in 2^{10} steps so that the polynomial, its derivatives, and the value of z may be considered accurate to the twentieth binary place. The smallest increments in z are $\delta = 2^{-20}$.

To obtain full accuracy for all roots, it is necessary to normalize by (7) and locate the roots approximately. Full significance is obtained for the smaller roots by renormalizing the polynomial to an R just greater than the modulus of the desired root.

Machine Design

The computer has three modes of operation:

1. Input.
2. Computation with $\delta = 2^{-10}$ or $\delta = 2^{-20}$,
3. Direction decision.

The operational aspects of these modes have been described. This section gives a general description of the techniques by which that functional behavior is accomplished. The detailed logical design will be available from the California Institute of Technology.

Input

To describe the input routine, it is necessary to understand some of the internal features of the computer. Input itself is so simple that it will be convenient to present a more detailed picture of the computer in this section as well.

During the first moments of the input phase the computer orients itself with respect to its internal timing. This feature arises from a general consideration of the computer configuration (see Fig. 4).

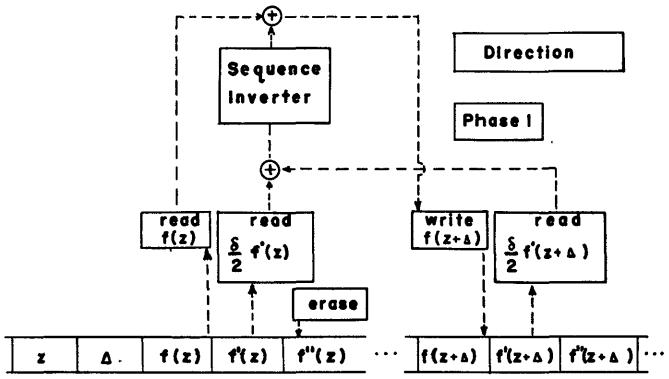

Fig. 4—The computer block diagram.

To minimize the number of reading and writing heads on the drum, it was necessary to use one circulating register. This implied a single clock channel with no reference to an origin pulse on the drum. Timing signals are required, however, to indicate certain word positions; without having the circulating register exactly one drum circumference in length it is necessary to generate certain timing markers inside the register itself.

In addition to the polynomial and its derivatives, the memory channel contains two words, "z" and "Δ." "Δ" holds a marker used to add the correct Δ into "z." "z" holds the value of the complex argument z of the polynomial. The timing signals are those necessary to distinguish:

1. The words $f(z)$, "Δ," "z," and $f^{(n)}(z)$,
2. The internal word structure: blank bits, sign bits, real or imaginary bits, and the position of the δth binary place.

Phase Control designates the first group of timing signals; the second group are general timing signals.

The blank bits between words may contain two markers. These markers are entered under the control of the general timing signals when the input mode switch is closed. One marker precedes the words holding $f(z)$ and "z"; it serves to synchronize the phase control. The other is a movable marker identifying any particular word.

The general timing signals originate in the clock channel. Originally the drum is uniformly divided into 1,344 positions. The clock channel is derived by occasionally omitting a single pulse (see Fig. 5). The clock pulses to operate the computer are obtained from a free-running phantastron synchronized by pulses from the clock channel. An omitted pulse is filled in for clock-pulse use by the phantastron. With one pulse omitted at a time, there is little drift. The missing pulse operates

the general timing flip-flops; they change state whenever a pulse is omitted from the clock channel. Synchronization of the general timing pattern with the drum pattern is self-controlled and may take several word times after the computer is turned on.

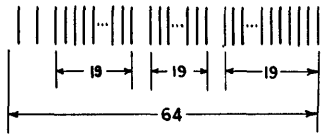

Fig. 5—One word of the pulse configuration on the clock channel.

The input bits are entered via two lever switches; each bit is inserted in the word identified by the movable marker. Input is most significant bit first. The bit is placed in a flip-flop and the desired word circulates back to the memory through that flip-flop.

Computation

There are three operations in the computation mode:
1. Multiplication by $\frac{1}{2}\delta$,
2. Inversion of the bit sequences as indicated by (5),
3. Addition of the respective components.

Multiplication by $\frac{1}{2}\delta$ is done by shifting the reading heads. This is indicated in Fig. 4; it is also the reason for limiting the absolute values to 1.0.

Inversion merely exchanges the biplexed positions of the real and imaginary components when Δ changes from real to imaginary. Two flip-flops are used for the inversion, and two separate adders are used to obtain a compromise between numbers of flip-flops and numbers of diodes.

Subtraction or addition is governed by Direction Control. $\frac{1}{2}\delta f'(z)$ and $\frac{1}{2}\delta f'(z+\Delta)$ are always added, but their sum is added to or subtracted from $f(z)$. The simplest technique is to combine the conversion to zero's complements of the subtrahend with the sequence inversion.

Direction Control

This is the most complicated logical operation in the computer. The decision is based on a truth table of all sign conditions in (5). The logic is derived from a similar table based on the information available internally in the computer. That is, due to sequence inversion and zero's complementation in D1 and D2 (Fig. 6), it is necessary to consider the form in which the sign information is available. One additional limitation on the decision logic is that the modified sign bits are not all available during any one clock time.

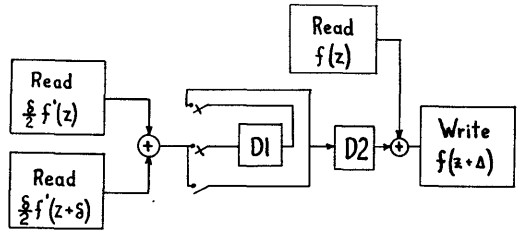

Fig. 6—Inversion of the bit sequence. (The logic is represented by a gang switch; the switches change state every clock time.)

Two flip-flops, G and H, are used to perform the decision. After each circulation of the memory channel a new value for Δ is determined from the new signs of $f(z)$ and $f'(z)$ and from the original states of G and H. Referring to Fig. 6 and using the additional notation

$B \equiv$ odd-even clock timer,
$M1 \equiv$ reading flip-flop for $f(z)$,
$T0 \equiv$ sign bit times (the real sign is in $M1$ during $\overline{B} \cdot T0 = 1$, and the imaginary sign is in $M1$ during $B \cdot T0 = 1$),

the input equations to G and H are written:

$$\begin{aligned} 1G &= (D1 \times D2 \times B) \cdot \overline{(D2 \times M1)} \cdot T0 \\ 0G &= 1G \\ 1H &= \overline{(D2 \times M1)} \cdot T0 \\ 0H &= 1H. \end{aligned} \tag{12}$$

These equations are given to illustrate the form in which the logical design is obtained. The bar denotes complementation and the $(\times)$ denotes the exclusive "or" operator

$$D1 \times D2 = D1 \cdot \overline{D2} + \overline{D1} \cdot D2.$$

The $+$ and the $\cdot$ symbols designate logical union and intersection, respectively. Eqs. (12) describe a flip-flop having the following behavior.

Input:		Output:
$1G_n$	$0G_n$	G_{n+1}
0	0	G_n
0	1	0
1	0	1
1	1	$\overline{G_n}$

(13)

The output is delayed one clock interval with respect to the input; n denotes the clock interval.

The decisions produced by (12) are not exactly correct. In a few cases G and H will select the correct real or imaginary Δ but will give it the wrong algebraic sign. The computer takes one step in the wrong direction and then recognizes its mistake and proceeds properly.

This is a situation where it is easiest to make an occasional wrong decision without hurting the functional operation of the device. Several additional diodes and an extra flip-flop would be required to make the correct decision at all times.

Physical Specifications

Two types of flip-flops are used. Those described by (13) are the first. The second kind are described by the middle two rows of (13). The second type introduces a delay and power amplification only (see Fig. 7).

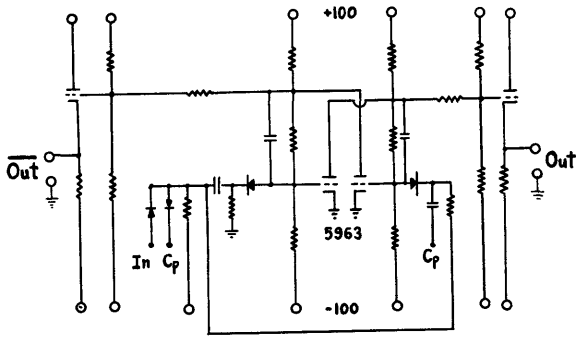

Fig. 7—Single-Input Flip-Flop. (The output swings from +5 to +20 volts; the clock pulse is referenced to +20 volts.)

Note that only two power supplies are required. These supply the flip-flops, diode matrices, and the read-write amplifiers. Approximately 100 watts is dissipated exclusive of heaters.

The basic clock frequency is 80.7 kc derived from 1,344 pulses on a 5-inch drum rotating at 3,600 rpm.

Mode selection is under the control of several switches on the control panel. They are used for starting, inserting the timing markers, input, Δ selection, forced direction control, locating the movable marker, and selecting any desired output (see Fig. 8).

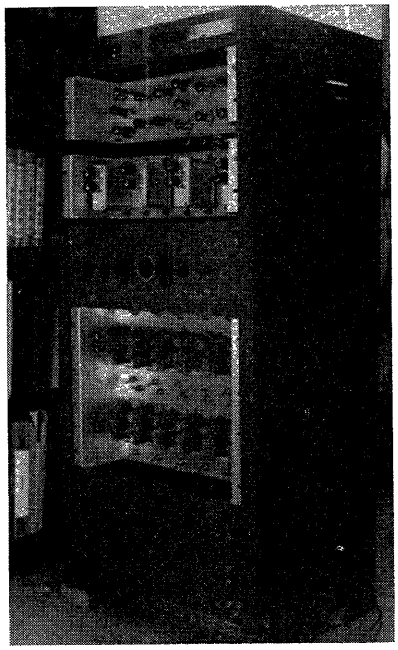

Fig. 8—The computer.

Conclusion

Several observations result from the design and construction of the machine. It requires about half the equipment necessary for a medium-speed general-purpose computer of moderate memory capacity. Where there is need for its special purpose, the ready availability of the computer and its ease of programming make it a valuable scientific tool. The binary coding and bit by bit input and output are limitations, but it is still felt that the equipment savings warrant them. Even more equipment could be eliminated if certain other features of the present computer were omitted.

While the logical techniques developed for the root extractor are conceptually useful in other digital devices, the computer itself has several other uses.

The machine can be considered as a function generator in two variables in the sense that it evaluates a polynomial of high degree. Inputs are any desired x and y. Internal programming is set to approach the x and y inputs rather than to diminish $|f(z+\Delta)|$. Discontinuous functions could be generated if there were some particular behavior characteristic of a higher derivative. Input could be to the derivative word position in this case.

In the field of servomechanisms the computer could be used to obtain root locus plots. A curve-plotter would be attached to read the "z" word in this case, and some form of internal gating would be used to indicate that a root had been located.

Acknowledgment

The material described above covers a portion of the work performed at the California Institute of Technology by the author in partial fulfillment of the requirements for the Ph.D. degree in Electrical Engineering. It was done during periods when the author was the holder of a Howard Hughes Fellowship and an International Business Machines Corporation Fellowship in Electrical Engineering. Particular acknowledgment is due Dr. Stanley P. Frankel for his invaluable ideas and suggestions contributed during the development of the computer, and to Dr. G. D. McCann for his helpful assistance throughout the project's duration.