SPECIAL PURPOSE DIGITAL COMPUTER WITH A NONERASABLE MEMORY UNIT

by

BEAT ALBERT GIMMEL

Dipl. El. Ing., Eidgenoessische Technische Hochschule, Zurich, Switzerland, 1964

A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1966

Approved by:

Charles A. Halijak

Major Professor

TABLE OF CONTENTS

INTRODUCTION

Scope

This thesis discusses a special purpose digital computer for the arctangent function. Reasons for this choice will be explained, and the computer design will be based on an upper polygonal approximation with uniform Chebyshev norm for the error.

A special purpose digital computer (SPC) executes a small, special set of instructions and solves only a restricted number of problems, perhaps one problem. On the other hand, a general purpose digital computer (GPC) solves a large set of problems. In past years, both types of digital computers have been constructed. The GPC's are almost exclusively employed in computing centers of industry, universities, and business corporations. However, SPC's came into existence long before GPC's. Familiar examples of SPC's are:

1. The switching system in an automatic telephone exchange.
2. Jacquard's punched card control for a loom.
3. Numerical control of drilling and milling machines.

SPC's are especially important in guidance systems of missiles and satellites. Economical and technical considerations demand SPC's for many applications.

The Computer as an Abstract Notion

The term, "computer", does not connote the metal parts, components, circuits, functional units, and so on. The underlying concept of a digital computer is the proven algorithm. The distinction between a GPC and an SPC may be narrowed by the following observation: A general purpose computer is a collection of special purpose computers (Figs. 1 and 2).

Nonerasable and Erasable Memories

There are two types of memories:

1. The nonerasable memory has memory cells whose contents remain unchanged during the computation process. There is no read-in of new information, and the read-out is nondestructive. Everyday examples are: dictionary, photography album, phonograph record, papercard with punched holes. Nonerasable memories are likewise denoted as permanent memories or as read-only memories.

   About ten years ago, the capacitive and the magnetic memory types aroused interest. Their features and operation are described in (2, 3, 4). A general survey of nonerasable memories is given in (1). They are used to store subroutines and other kinds of information.
2. The erasable memory has memory cells with the property that their contents can be replaced any time by new entries.

Fig. 1. Symbol for an SPC.

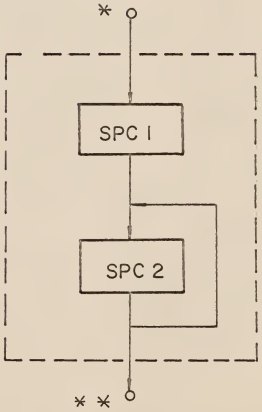

Fig. 2. GPC as a collection of interlinked SPC's.

Figures 3 and 4 show schematically these two types of memory cells.

AN ARCTANGENT SPECIAL PURPOSE COMPUTER

Problem

Our objective is to design an SPC that calculates the arctangent function with a preassigned accuracy. After a brief discussion of the power series method, some properties of the arctangent function will be indicated. As it is a transcendental function, the value for a certain argument is obtained in an infinite number of steps by means of power series such as:

$$\arctan(x) = x - \frac{1}{3}x^3 + \frac{1}{5}x^5 - \ldots, \quad x^2 \leq 1 \qquad (1)$$

$$\arctan(x) = \frac{\pi}{2} - x^{-1} + \frac{1}{3}x^{-3} - \frac{1}{5}x^{-5} + \ldots, \quad x \geq 1 \qquad (2)$$

Equation (1) is derived in (6) by means of Taylor series. Equation (2) is obtained from equation (1). This is done in the introduction of reference (7). Reference (2) contains a 12-digit tabulation of the arctangent function. The limitation to execute only a finite number of steps gives rise to questions of approximation, speed of convergence, and error measure.

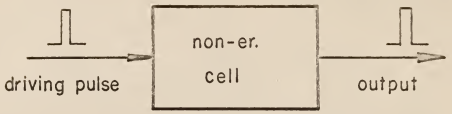

Fig. 3. Non-erasable memory cell.

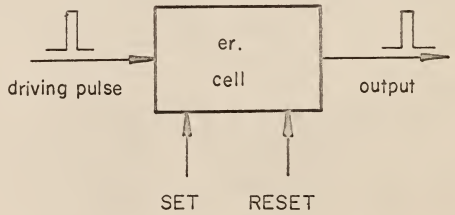

Fig. 4. Erasable memory cell.

Convexity, Symmetry, and Monotonicity

Let $f(x)$ be a function defined in the open interval $I = (a, b)$, and let $I_1 = (x_1, x_2)$ be an open interval such that $I_1 \subset I$. $g(x)$ is a line through the points $\{x_1, f(x_1)\}$ and $\{x_2, f(x_2)\}$. $f(x)$ is convex in I, if for all $x \in I_1$ holds $f(x) - g(x) < 0$, whichever I_1 is chosen (Fig. 5). Dually, $f(x)$ is concave in I, if for all $x \in I_1$ holds $f(x) - g(x) > 0$, whichever I_1 is chosen (Fig. 6).

Convex functions are also called sublinear functions, and concave functions are called superlinear functions. If a function is twice differentiable, convexity corresponds to positive curvature and concavity to negative curvature. A short and elegant theory on convex functions is given in Emil Artin's monograph (5).

Analyticity and concavity of the arctangent function in the open interval $(0, \infty)$ imply that it has negative curvature throughout $(0, \infty)$. By Rolle's Theorem, the minimum of curvature lies in $(0, \infty)$. The calculation of Appendix A shows that the minimum occurs at $x = 0.831$.

The arctangent function is skewsymmetric with respect to the origin; it is an odd function. Therefore only the open interval $(0, \infty)$ is relevant to us.

With the argument x increasing throughout the open interval $(-\infty, \infty)$, the arctangent function is increasing too, and therefore monotonic in $(-\infty, \infty)$. The arctangent function, of course, is considered as single valued, defined in the value range

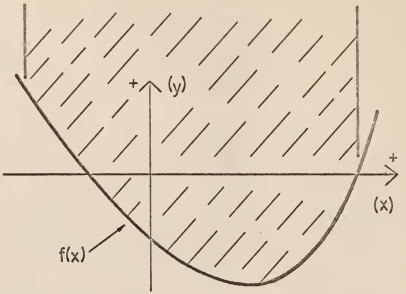

Fig. 5. A convex function f(x) .

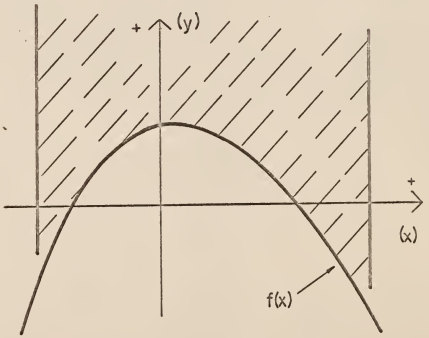

Fig. 6. A concave function f(x).

$$-\frac{\pi}{2} = y = +\frac{\pi}{2} .$$

Error Measures

Different ways to measure the quality of approximation are:

(i) error area $\int_X |e(x)|\, dx$

(ii) least square area $\int_X e(x)^2\, dx$

(iii) maximum of absolute error value, $\max_X |e(x)|$

where $e(x)$ is the error function. Error criteria (i) and (ii) have a great disadvantage. They are integral error measures and do not take account of local errors. Gibbs phenomenon in Fourier series is a case in point of this disadvantage. Error criterion (iii) notes local errors. They are bounded by the preassigned maximal error e_{max}. This error measure, the Chebyshev-norm, will be employed.

POLYGONAL APPROXIMATION

Definition

A way to approximate a continuous function f(x) (Fig. 7) is to use an approximating polygon ($P_0\ P_1\ P_2\ .\ .\ .\ P_N\ P_{N+1}$). The vertices of this polygon coincide with points of the function such that

$$P_0 = \{x_0,\ f(x_0)\}\ ,\quad P_1 = \{x_1,\ f(x_1)\}\ ,\ .\ .\ .,$$
$$P_N = \{x_N,\ f(x_N)\}\ ,\quad P_{N+1} = \{x_{N+1},\ f(x_{N+1})\}\ .$$

The purpose of this thesis is to approximate f(x) = arctan(x) in the interval $(0,\infty)$ by a polygon (Fig. 8). As this function is concave, the approximating polygon is also concave.

The arctangent function has two inflection points, one at zero, coincident with P_0, and one at infinity, coincident with P_∞. Hence the polygon is denoted by ($P_0\ P_1\ P_2\ .\ .\ .\ P_N\ P_{N+1}\ P_\infty$). In functional notation, the polygon ($P_0\ P_1\ P_2\ .\ .\ .\ P_N\ P_{N+1}\ P_\infty$) is denoted by g(x). g(x) is a continuous, monotonic, and single-valued function, defined in the domain $0 \leq x \leq \infty$ with the range $0 \leq y \leq +\pi/2$.

The orthogonal projection of polygon vertices P_0, P_1, . . ., P_N, P_{N+1}, P_∞ yields the end points A_0, A_1, . . ., A_N, A_{N+1}, A_∞ of the intervals I_0, I_1, . . . I_N, I_{N+1}, I_∞ on the x-axis such that $I_0 = (A_0\ A_1)$, $I_1 = (A_1\ A_2)$, . . ., $I_N = (A_N\ A_{N+1})$ and $I_\infty = (A_{N+1}\ A_\infty)$ (Fig. 8).

In order to calculate the value for the argument x, the computer has first to decide in which interval I_n the argument

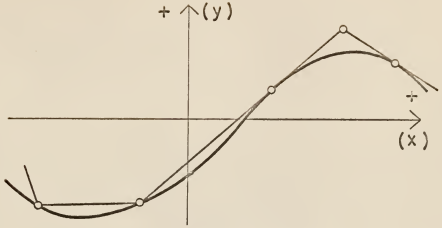

Fig. 7. Polygonal approximation g(x) of a continuous function f(x).

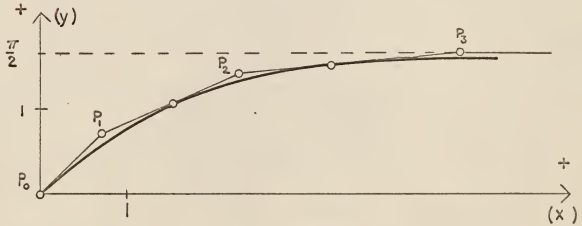

Fig. 8. A polygonal approximation of the arctangent function.

x lies. If $x \in I_n$, the value at x is

$$x \cdot m_n + b_n = y$$

where m_n, b_n are stored as nonerasable information in the computer memory.

The following subchapters treat in detail the generation of various polygonal approximations for the arctangent function. All of them are upper polygonal approximations, i.e., polygonal approximations that satisfy the inequality $g(x) - f(x) \geq 0$. As the arctangent function $f(x) = \arctan(x)$ is concave in interval $(0, \infty)$, these upper polygonal approximations consist exclusively of tangent lines.

Tangent Line Intersection Method

This method begins with two tangent lines of arctan(x) with points of tangency at $x = x_{t0} = 0$ and $x = x_{t\infty} = \infty$. Subscript "t" denotes the tangent line of arctan(x) at the corresponding arguments x. These two lines intersect in the point ${}_1P_{e0}$ (Fig. 10) with the argument ${}_1x_{e0}$. Subscript "1" denotes the first approximation stage, and "e" tells that at argument x, the maximum error of this polygon approximation occurs.

This first approximation divides $(0, \infty)$ in two intervals, ${}_1I_0$ and ${}_1I_\infty$, adjoining at the dichotomy point ${}_1x_{e0}$. The tangent lines of arctan(x) at $x = 0$ and $x = \infty$ are called <u>half tangent lines</u>, since only one side of their point of tangency is considered; if both sides are considered, it is called a <u>full tangent line</u>. The number of full tangent lines employed in a

polygonal arctangent approximation is signified by N and the total number of tangent lines by M. This terminology holds throughout this thesis.

Two consecutive tangent lines of arctan(x) of the k^{th} approximation stage with arguments ${}^{k}x_{t,j}$ and ${}^{k}x_{t,j+1}$ are called adjacent. Hence tangent lines for $x = x_{t0}$ and $x = x_{t\infty}$ are adjacent in the first approximation stage, but not in higher stages.

It was discussed how to arrive at the first approximation stage. In order to obtain from the k^{th} stage the $(k+1)^{th}$ stage, the points

$$\left\{{}^{k}x_{e0},\ {}^{k}y_{e0}\right\},\ \cdots,\ \left\{{}^{k}x_{en},\ {}^{k}y_{en}\right\},\ \cdots,\ \left\{{}^{k}x_{eN(k)},\ {}^{k}y_{eN(k)}\right\}$$

are constructed by intersecting adjacent tangent lines of the k^{th} stage. The maximal error arguments

$${}^{k}x_{e0},\ {}^{k}x_{e1},\ \cdots,\ {}^{k}x_{en},\ \cdots,\ {}^{k}x_{eN(k)}$$

become tangent arguments for the $(k+1)^{th}$ stage such that

$${}^{k}x_{e0} = {}^{k+1}x_{t1},\ {}^{k}x_{e1} = {}^{k+1}x_{t3},\ \cdots,\ {}^{k}x_{en} = {}^{(k+1)}x_{e(2n+1)},\ \cdots \tag{1}$$

Now the tangent lines of arctan(x) for the arguments

$${}^{k+1}x_{t1},\ {}^{k+1}x_{t3},\ \cdots,\ {}^{k+1}x_{e(2n+1)},\ \cdots$$

are constructed.

The tangent lines of the k_{th} stage become tangent lines of the $(k+1)^{th}$ stage such that

$${}^{k}x_{t0} = {}^{k+1}x_{t0},\ {}^{k}x_{t1} = {}^{k+1}x_{t2},\ \cdots,\ {}^{k}x_{tn} = {}^{(k+1)}x_{e(2n)},\ \cdots \tag{2}$$

Equations (1) and (2) form the set of tangent arguments of the points of tangency for the $(k+1)^{th}$ stage. The relations between tangent arguments and maximal error arguments are represented in Fig. 9.

Since each maximal error argument ${}_{k}x_{en}$ generates two new error arguments ${}_{k+1}x_{e(2n)}$ and ${}_{k+1}x_{e(2n+1)}$ in the next stage, the number of maximal error arguments of the k^{th} stage is

$$N_e = 2^{k-1} .$$

(For the first stage holds $N_e = 1$.) As tangent arguments and error arguments follow alternately, there are

$$N_t = 2^{k-1} + 1$$

tangent arguments in the k^{th} stage. After having dropped initial half tangent line at ${}_{k}x_{t0} = 0$ and ended half tangent line at ${}_{k}x_{t\infty} = 0$, there are

$$N = N(k) = N_t - 2 = 2^{k-1} - 1$$

full tangent lines for the k^{th} stage. This shows that the number of tangents employed in this type of polygonal approximations grows exponentially with the number of stages.

An important problem of polygonal approximation is to reduce the number of tangent lines for a required accuracy in order to get a fast computing SPC and to economize on memory. This approximation method, however, does not exhaust sufficiently the range of possible polygonal approximations. Also, the maximal errors are not uniform. The only advantage of this approximation is that it is easy to calculate.

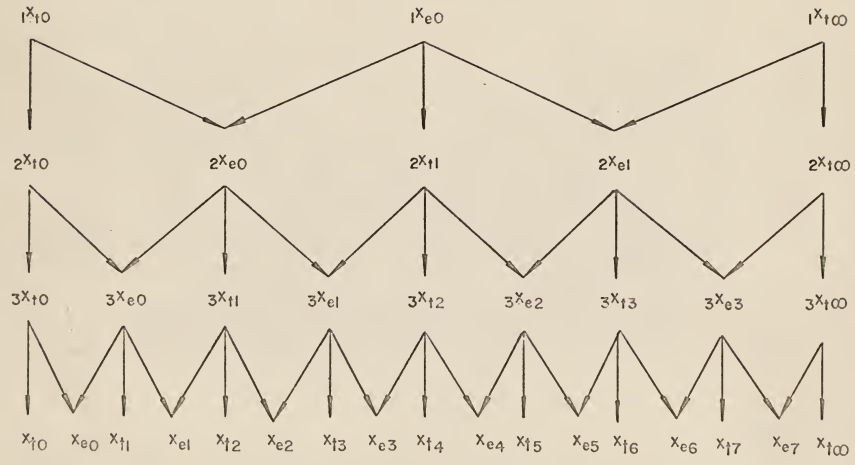

Fig.9. List of arguments of maximum errors and points of tangency for the first four approximation stages.

Quasi-equal Error Method

The initial stage of approximation is the same as the preceding one (Fig. 11).

The second stage is obtained by drawing a tangent line ${}_2t_1$ that contacts the arctangent curve at the point $\{{}_2x_{t1}, {}_2y_{t1}\}$. ${}_2x_{t1}$ lies in the interval $(0, \infty)$. This tangent intersects the two tangent lines of the initial stage in the points $\{{}_2x_{e0}, {}_2y_{e0}\}$ and $\{{}_2x_{e1}, {}_2x_{e1}\}$. Therefore the approximating part of the tangent line ${}_2t_1$ is determined by the interval $({}_2x_{e0}, {}_2x_{e1})$. The argument ${}_2x_{t1}$ is chosen such that the maximal errors at the endpoints of interval $({}_2x_{e0}, {}_2x_{e1})$ are equal. Thus the second stage yields an approximation with uniform maximal errors, i.e., in the uniform Chebyshev norm.

This procedure, when applied to all pairs of adjacent tangent lines of the second stage, leads to the third stage. The entire procedure of the second stage can be repeated arbitrarily. It generates new intervals that are always shorter than the preceding ones. As numerical computation demonstrates, the stages of higher order than two do not yield uniform maximum errors at the endpoints of all intervals.

The number of tangent lines for the k^{th} stage is

$$N_t(k) = \frac{k(k+1)}{2} + 2$$

$N_t(k)$ grows exponentially as k increases. Also, the calculation to these approximations is not direct; it involves iteration.

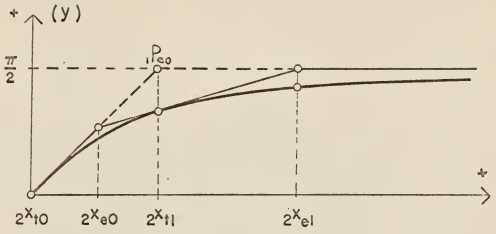

Fig. 10 : The tangent line intersection method.

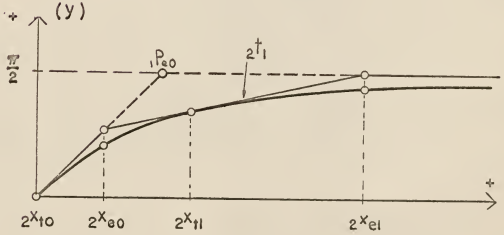

Fig. 11 : The quasi-equal error method.

The "Quasi-equal Error Method" yields by transition to higher stages smaller and smaller errors. In the following chapter, the "Polygonal Approximation with Uniform Chebyshev Norm" has not only this quality; it has uniform maximum error at the end-points of the intervals I_0, I_1, . . ., I_N, I_{N+1}, I_∞ . In addition, polygonal approximations are for any number of tangent lines available.

It should be noted that the polygonal approximations obtained by using either the tangent line intersection method or quasi-equal error method are all upper polygonal approximations.

POLYGONAL APPROXIMATION WITH UNIFORM CHEBYSHEV NORM

Definition

If an approximation is subject to the uniform Chebyshev norm, all relative maxima of $e(x)$ have the same magnitude (Figs. 12 and 13). This requirement automatically reduces the number of tangents used for polygonal approximation. The third attempt to approximate a polygonal approximation can be described as follows.

Design a special purpose digital computer that calculates the arctangent function with a preassigned accuracy, using an upper polygonal approximation with uniform Chebyshev norm.

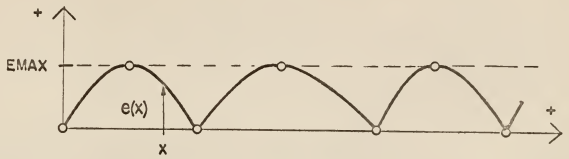

Fig. 12. Error distribution e(x) of a polygonal approximation with uniform Chebyshev norm .

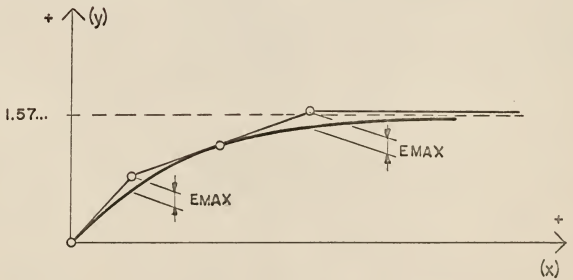

Fig. 13. Polygonal approximation with uniform Chebyshev norm.

The Chebyshev Polygonal Approximation with Zero Boundary Error

Zero boundary error means that the error of this approximation is zero at the inflection points of the arctangent function. The procedure is:

(i) The half tangent line t_0 of the arctangent graph at $x = 0$ is used as initial approximating line. Therefore the error at point $T_0 = \{x,y\} = \{0,0\}$ is zero. From here the point x,y travels on tangent line upwards to the right. The error $e(x) = x - \arctan(x)$ is measured simultaneously. $e(x)$ increases monotonically with x. Finally, point $\{x,y\}$ reaches point $P_1 = \{x_{el}, y_{el}\}$ with argument $x = x_{el}$. At this position, $e(x) = e(x_{el})$ is equal to the preassigned error e_{max}. The search for point P_1 is called the E-Start Problem.

(ii) Another point $\{x,y\} = \{x_{el}, y_{eel}\}$ lies on the curve of the arctangent function such that $x = x_{el}$. From here point $\{x,y\}$ starts traveling upwards to the right. At the same time, the position of the line L, determined by the points $\{x_{el}, y_{el}\}$ and $\{x,y\}$, is compared with the tangent line at $\{x,y\}$ of the arctangent function. The angle between these two lines decreases monotonically with x. Point $\{x,y\}$ stops as soon as the angle ϕ approaches zero, where line L coincides with the full tangent line t_1. The point of

contact, t_1, is denoted by $T_1 = \{x_{t1}, y_{t1}\}$. The search for point T_1 is called the T-Problem (Fig. 14).

(iii) The third step of this procedure is called the E-Problem. While the E-Start Problem began at point $\{0,0\}$, the E-Problem starts at the point of tangency, $T_1 = \{x_{t1}, y_{t1}\}$, calculated now. The point $\{x,y\}$ stops at point $P_2 = \{x_{e2}, y_{e2}\}$. If the procedure is terminated here, it is called a 1-Trip, because a polygonal approximation uses one full tangent line. The horizontal line L with the equation $y = y_{e2}$ is parallel to the asymptotic line t_∞ and approximates $\arctan(x)$ for $x > x_{e2}$.

If a polygonal approximation with two full tangent lines is desired, the procedure of the T-Problem (ii) for $\{x_{e2}, y_{e2}\}$ has to be iterated in order to get to the tangent point $\{x_{t2}, y_{t2}\}$. From here, the procedure of the E-Problem (iii) yields point $\{x_{e3}, y_{e3}\}$. This polygonal approximation is called a 2-Trip. For arguments $x > x_{e3}$, the line L with the equation $y = y_{e3}$ is used.

If E-Start Problem, T-Problem, and E-Problem are executed in the sequence

E-Start, T, E, T, . . ., T, E, . . .

where E- and T-Problem occur N-times, the resulting polygonal approximation is called an N-Trip.

The same train of ideas holds for all N-Trips. Hence the "Upper Chebyshev Polygonal Approximation with Zero Boundary Error" could be illustrated by the 2-Trip. To describe N-Trips,

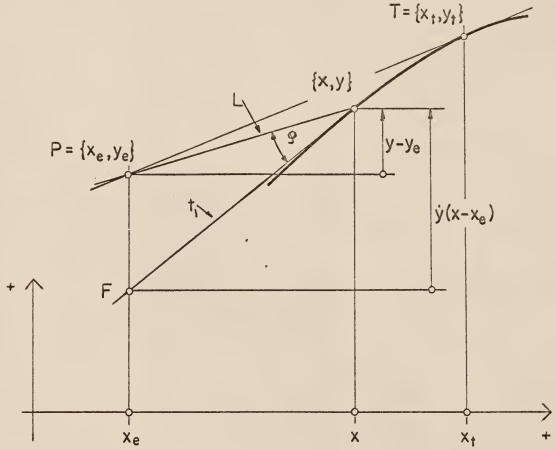

Fig.14. The T-Problem.

the following symbols are used for:

Tangent lines

$t_0, t_1, t_2, \ldots, t_N, t_\infty$;

corresponding points of tangency

$T_0, T_1, T_2, \ldots, T_N, T_\infty$;

x-coordinates of points $T_0, \ldots, T_\infty$

$x_{t0}, x_{t1}, x_{t2}, \ldots, x_{tN}, x_t$;

y-coordinates of points $T_0, \ldots, T_\infty$

$y_{t0}, y_{t1}, y_{t2}, \ldots, y_{tN}, y_t$;

end points of approximating tangent line pieces are

$P_0, P_1, P_2, \ldots, P_N, P_{N+1}, P_\infty$, where $P_0 = T_0$ and $P_\infty = T_\infty$ as tangent lines t_0 and t_∞ are half tangents;

x-coordinates of points $P_1, \ldots, P_\infty$

$x_{e1}, x_{e2}, x_{e3}, \ldots, x_{eN}, x_{e(N+1)}$;

y-coordinates of points $P_1, \ldots, P_\infty$

$y_{e1}, y_{e2}, y_{e3}, \ldots, y_{eN}, y_{e(N+1)}$;

approximation intervals on x-axis

$I_0, I_1, I_2, \ldots, I_N, I_\infty$.

Symbols with the same subscript are related together, e.g., tangent line t_3 contacts the graph of the arctangent function at T_3 which has the coordinates x_{t3} and y_{t3}. The approximating line piece of t_3 has the end points P_3 and P_4 whose coordinates are $\{x_{e3}, y_{e3}\}$ and $\{x_{e4}, y_{e4}\}$. x_{e3} and x_{e4} are the end points of the interval I_3. As different N-Trips are independent from each other, there is no need for left subscripts. N has a similar function as the k of the k^{th} approximation stage.

The 2-Trip mentioned above uses the tangent line t_0 at $x = 0$, but not necessarily the asymptotic line $y = \pi/2$ as approximating tangent line. The eventual result of the 2-Trip for $e_{max} = 0.01$ is $y_{e3} = 0.819 \ldots$. As $y_{e3} < \pi/2$, the value of e_{max} has to be increased. If $e_{max} = 0.02$, $y_{e2} = 1.015 \ldots$ e_{max} is increased until $y_{e3} = \pi/2$. This particular problem is called Pi-half problem. The ensuing approximation is called Asymptotic 2-Trip. In order to obtain the Asymptotic N-Trip, the same reasoning holds for any other N-Trip, provided N is a nonnegative integer.

Upper, Lower, and Intermediate Approximation

As the arctangent function is a concave function in the interval $(0, \infty)$, the upper polygonal approximation $g_u(x)$ is a tangent approximation. By the same reason, a lower approximation $g_\ell(x)$ of arctan (x) is a secant approximation in $(0, \infty)$. If a lower and an upper polygonal approximation have the same approximation intervals (Fig. 15) and if the maximum errors of $g_\ell(x)$ and $g_u(x)$ are uniform, their maximum errors are equal. The intermediate approximation of a lower and an upper approximation with common approximation intervals is defined as

$$g_i(x) = \frac{1}{2} (g_u(x) + g_\ell(x)) .$$

The maximum error of $g_i(x)$ is $\frac{1}{2} \cdot e_{max}$. Therefore the intermediate approximation allows to reduce the number of tangent lines for a uniform polygonal approximation with a preassigned

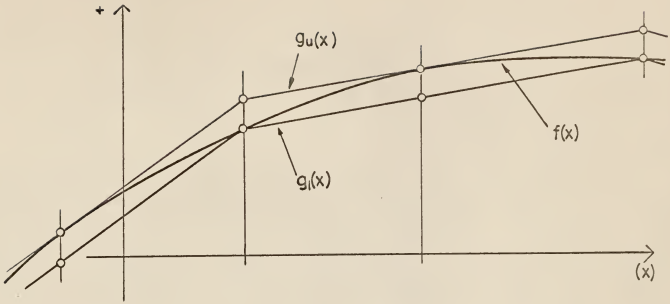

Fig. 15. Upper and lower approximations $g_u(x)$ and $g_l(x)$ of function f(x).

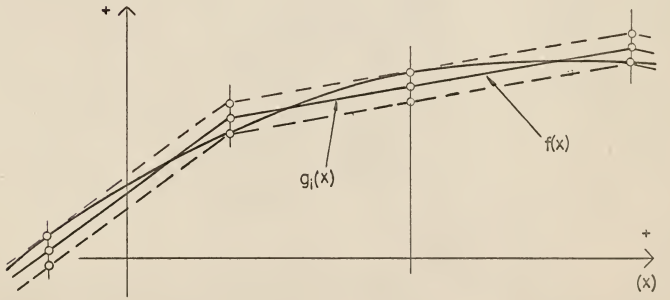

Fig. 16. Intermediate polygonal approximation $g_i(x)$ of function f(x).

accuracy (Fig. 16).

The Step-division Process

The E-, E-Start, and T-Problems still to be solved in the next subchapter have a common feature. They can be reduced to finding the zero crossing of a test function h(x). This can be done by establishing an appropriate algorithm. Subsequently, the Step-division Process is treated in detail.

Let h(x) be a monotonic and continuous function defined in a closed interval (a,b). It is assumed that there are two arguments, x_a and x_b in (a,b) whose values $h(x_a)$ and $h(x_b)$ have opposite signs. These assumptions necessarily entail a zero crossing for h(x) in the interval (x_a,x_b). The Step-division Process (SDP) whose purpose is to locate this zero crossing can be started at x_a. There are two types of Step-division Processes.

In the Unilateral SDP, x approaches the zero crossing always from one direction. The process starts at $x_0 = x_a$, then x_0 increases by a_0 such that $n \cdot a_0 = x_b - x_a$, where n is a positive integer. After computation of $h(x_1)$, where $x_1 = x_a + a_0$, there are two cases to distinguish.

Case (a). $\text{sign}\,(h(x_1)) = \text{sign}\,(h(x_0))$

This means that there is no zero crossing in the interval (x_0,x_1) and that the argument increases again such that $x_2 = x_1 + a_0$. This process is continued until Case (b) occurs.

Case (b). sign $(h(x_{i+1})) \neq$ sign $(h(x_0))$

at the $(i+1)^{th}$ iteration. Hence the zero crossing lies in the interval (x_i, x_{i+1}). In order to bring the argument back into the new starting position x_0, x_{i+1} is decreased by a_0, $(x_0 = x_{i+1} - a_0)$. Here the length of a_0 is divided by a positive integer t, $(a_1 = \frac{a_0}{t})$ such that $t > 1$. The same procedure, described so far, is repeated for a_1 and the new value of x_0.

Figure 17 shows the flow chart of the "Unilateral" algorithm. The substitution operation that is denoted by a reverted arrow " ← " avoids subscripting of argument x and step a. a ← b is read: The value of a is replaced by the value of b. The (a - δ)-decision-box terminates computation as soon as the desired accuracy, indicated by δ, is attained.

In the Oscillatory SDP, x approaches the zero crossing alternately from either direction. As in the "Unilateral SDP", the process starts at $x_0 = x_a$, too, and x_0 is increased likewise. The value $h(x_i)$ is calculated for each argument x_i. Again, there are two cases to distinguish.

Case (a). sign $(h(x_i))$ = sign $(h(x_{i-1}))$

Case (b). sign $(h(x_i)) \neq$ sign $(h(x_{i-1}))$.

It should be obvious that the signs of two succeeding arguments x_{i-1} and x_i are compared here. In Case (a), x_i is again increased by a_0. In Case (b), however, there exists a zero crossing in the interval (x_i, x_{i+1}). $x_{i+1} = x_0$ is the new initial argument. Step (a) is divided by t, $(a_1 = \frac{a_0}{t})$, t being a positive integer, greater than 1. The new step a_2 is subtracted from the new initial argument x_0, and the same process is

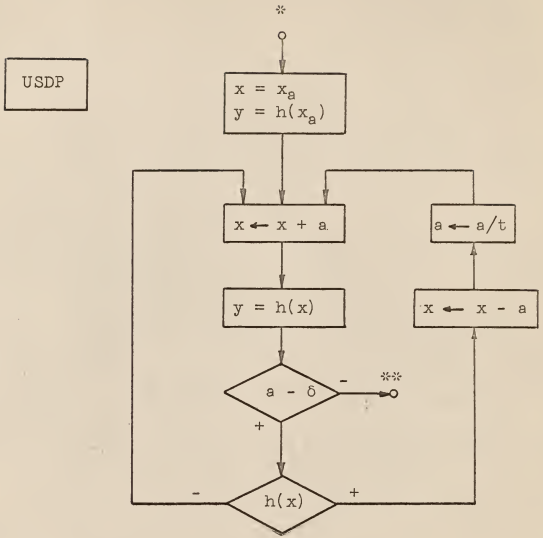

Fig. 17. Flow chart of the "Unilateral step division process" (USDP).

repeated over and over, resulting in

$$a_2 = \frac{a_1}{t} = \frac{a_0}{t^2} \qquad a_3 = \frac{a_2}{t} = \frac{a_0}{t^3} \qquad \ldots$$

Figure 18 shows the flow chart of the Oscillatory SDP.

The Bisecting Step Process

The most important assignment for t is t = 2. In this case, the interval with the zero crossing is bisected. The features of both, Unilateral and Oscillatory SDP's, can be combined. Figure 19 shows the flow chart of this process, called Bisecting Step Process. The advantage of this SPD over processes with $t > 2$ is due to the fact that after each calculation of $y = f(x)$, the step of length a is halved. After 100 iterations, if the process starts with an initial step $a = 1.0$, the error of the approximation is at most $\frac{1}{2^{100}}$. For the processes with $t > 2$, the upper bound for the error is

$$t^{-100/t} .$$

This holds for the worst case where t iterations are done between two step divisions. For the specific case t = 10, the error decreases after M iterations to

$$t = 10: \quad \delta_{max} = \frac{1}{10^{0.1 \cdot M}} = \frac{1}{(1.259\ldots)^M}$$

where $10^{0.1 \cdot M} = 10^{0.1 \cdot M} = (1.259\ldots)^M$

The upper bound of the error decreases much slower than for t = 2:

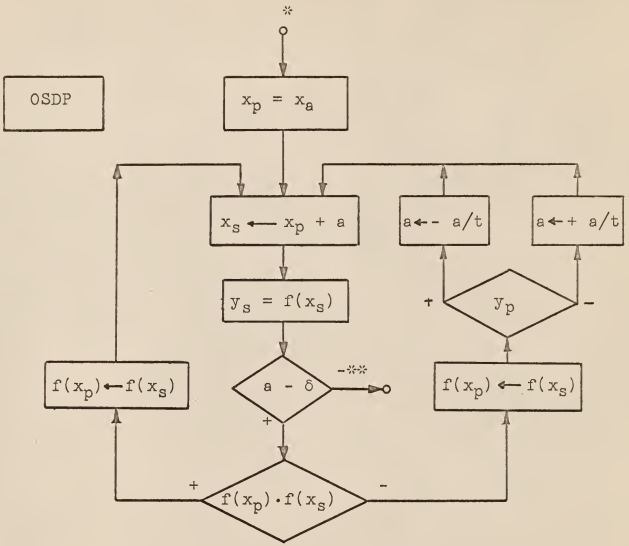

Fig. 18. Flow chart of the oscillatory step division process (OSDP).

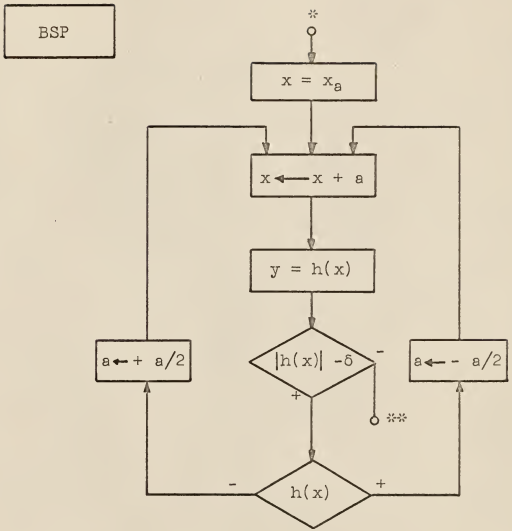

Fig. 19. Flow chart of the bisecting step process (BSP).

$$\delta_{max} = \frac{1}{2^M}$$

as the number M of iterations grows. Although in case t = 10 only five iterations between two step divisions are executed on the average

$$t = 10: \quad \delta_{max} \approx \frac{1}{10^{0.2 \cdot M}} = \frac{1}{(1.585\ldots)^M}$$

the SDP for t = 10 is still slower than the SDP for t = 2.

The SDP for t = 2 is called "BIT-BY-BIT METHOD", if the numbers used in the process are represented in binary notation. Likewise, the SDP for t = 10 is called "DIT-BY-DIT METHOD", if the numbers used in the process are represented in decimal notation. "DIT" is a contraction of the term "decimal digit". In the "BIT-BY-BIT METHOD" as well as in the "DIT-BY-DIT METHOD", the division $\frac{a}{t} = \frac{1}{t}$ at each step change takes very little time to be performed. Suppose "BIT-BY-BIT" (BBB) and "DIT-BY-DIT" (DBD) are started with unit step length, the following sequences of steps occur.

BBB: $a = 1._{(2)}, \ 0.1_{(2)}, \ 0.01_{(2)}, \ 0.001_{(2)}, \ \ldots$

DBD: $a = 1. \quad , \ 0.1 \quad , \ 0.01 \quad , \ 0.001 \quad , \ \ldots$

The subscript "(2)" indicates binary notation. For all step lengths, the mantissa is one digit long, whereas in other SDP's the length of the mantissa of the number representing step length is increased by each division. Thus computation time is larger, and the mantissas of the step lengths are truncated.

The Newton Process

In the "Newton Process" (NP), the conditions to be satisfied by h(x) are the same as those for the "Step-division Process", with the exception that there exists for h(x) a continuous monotonic first derivative h'(x) in the interval (x_a, x_b). These assumptions are sufficient for the convergence of the NP.

There are functions h(x) with non-monotonic continuous first derivative. h(x) with an inflection point at the zero crossing shows special interest. There are two cases to distinguish.

(a) If h(x) is concave for arguments less than the argument of zero crossing and convex for arguments greater than the argument of zero crossing, the NP is convergent.

(b) If h(x) is convex for arguments less than the argument of zero crossing and concave for arguments greater than the argument of zero crossing, the NP is either convergent or divergent, depending on the local properties of h(x).

The assertion of case (a) is thoroughly treated in Appendix B.

The flow chart of the "Newton Process" is shown in Fig. 20.

The E-Start Algorithm

The N-TRIP problem starts with finding the argument x_{el} where the maximal error e_{max} of the half tangent line t_0 occurs. The input data for this algorithm are:

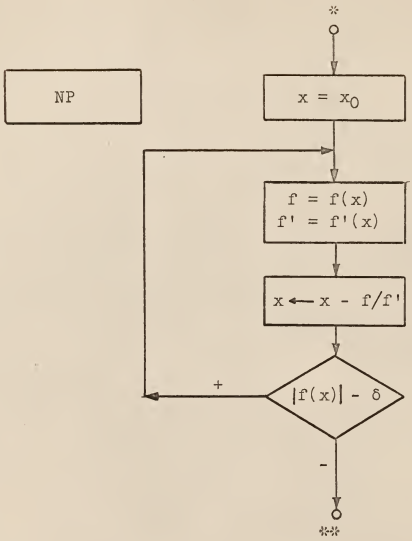

Fig. 20. Flow chart of the Newton process (NP).

$$x_{t0} = 0; \quad y_{t0} = 0; \quad \dot{y}_{t0} = 1 .$$

The "E-Start Algorithm" (E-START) searches for the zero crossing of the test function

$$h(x) = (x - \arctan(x)) - e_{max} .$$

As initial step, a = 1.0 is appropriate. Figure 21 shows the flow chart of the E-Start Algorithm that uses a DBD method. Expressions in the different blocks are written in FORTRAN language. The "equals" signs in the rectangular assertion boxes refer to "substitution", that were denoted by " ← " in non-FORTRAN flow charts. Table 1 explains symbols used in FORTRAN language. The output data of the E-START are the approximate values for x_{el} and y_{el}.

Experience with the IBM 1620 electronic digital computer has shown that the E-START computation time can be reduced by a factor 3 when using a Newton Process. The test function

$$h(x) = (x - \arctan(x)) - e_{max}$$

satisfies all the requirements for a NP. The calculation

$$h'(x) = 1 - \frac{1}{1 + x^2}$$

leads to the recurrence relation

$$x_{n+1} = x_n - \frac{h(x)}{h'(x)}$$

$$x_{n+1} = x_n - \frac{(x - \arctan(x)) - e_{max}}{1 - \frac{1}{1 + x^2}}$$

Figure 22 shows the FORTRAN flow chart of the E-START that uses an NP.

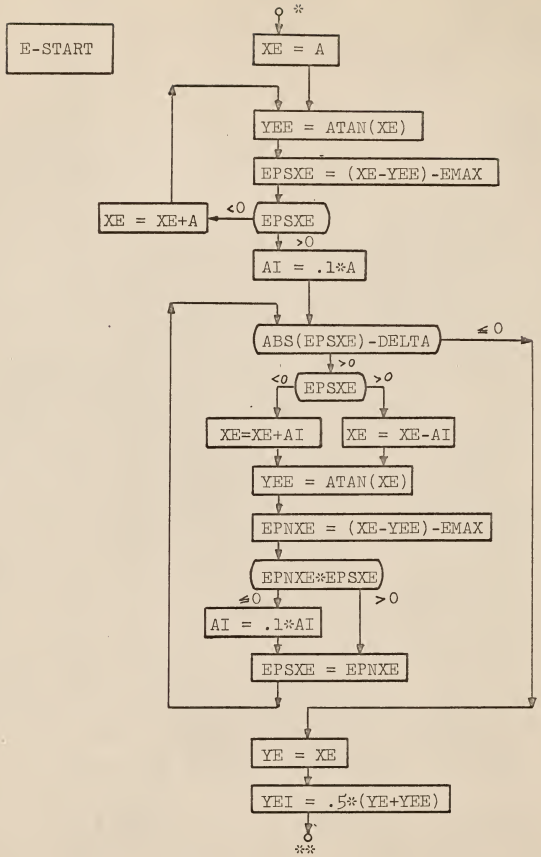

Fig. 21. FORTRAN flow chart of the E-START algorithm with DBD method.

Table 1. Explanation of FORTRAN symbols.

FORTRAN variable symbol	Standard notation	Name of variable that approaches ...
XE	$x \rightarrow x_e$	... argument x_e
YE	$y \rightarrow y_e$	... y_e
YEE	$y \rightarrow y_{ee}$	... y_{ee}
YEI	$y \rightarrow y_{el}$	... y_{ei}
EPSXE		... DELTA
EPNXE		... DELTA
EPS	$e(x)$	error
XT	$x \rightarrow x_t$	... x_t) coordinates of
YT	$y \rightarrow y_t$	... y_t) point of tangency T_n
YDT	$\dot{y} \rightarrow \dot{y}_t$	... $\dot{y}_t$ slope of tangent line
B	$b \rightarrow b_t$	... b_t y-intercept of " "
EPSXT	$h(x_p)$	error at argument x
EPNXT	$h(x_s)$	error at argument x + h
A	a_0	constant step
A1	a_1, a_2, ...	variable step
DELTA		constant reference error that determines arithmetic accuracy
EMAX	e_{max}	maximal error
ATAN	arctan	arctangent function
ABS	...	absolute value

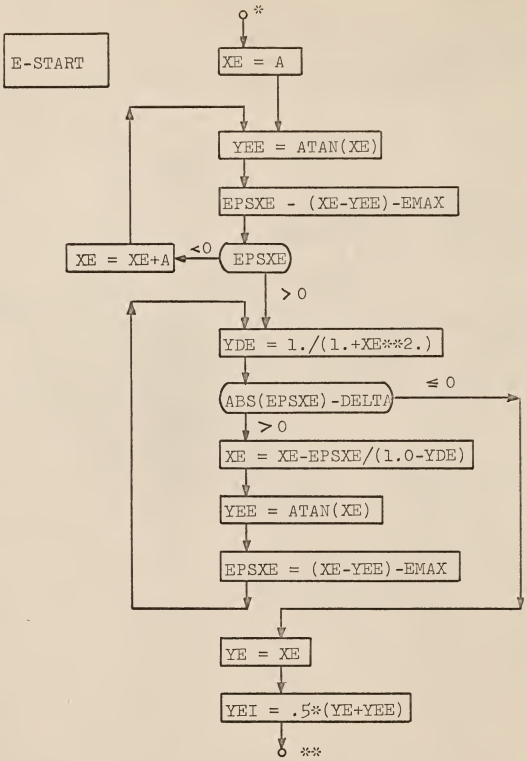

Fig. 22. FORTRAN flow chart of the E-START algorithm with Newton process.

The T-Algorithm

In the subchapter on "Upper Chebyshev Polygonal Approximation with Zero Boundary Error" it was explained how, at the point of tangency, the angle $\emptyset$ becomes zero. There, Fig. 14 makes plausible that the length of the directed segment $\overline{FP}_n$ tends toward zero, as x tends toward x_t. This yields the test function

$$h(x) = \overline{FP}_n = (y - y_e) - \dot{y}(x - x_e)$$

or

$$h(x) = (\arctan(x) - y_e) - ((x-x_e)/(1+x^2)) .$$

Input data are x_e and y_e, obtained from a previously computed "E-START Algorithm" or an "E-Algorithm". Output data are x_t and y_t. Figure 23 shows the "T-Algorithm" solved with the DBD Method.

The function h(x) meets requirements for the "Newton Process". The first derivative of h(x) is:

$$h'(x) = \frac{(x - x_e)2x}{(1 + x^2)^2} = (x - x_e)2x\dot{y}^2 \quad .$$

Functions h(x) and h'(x) yield the following recurrence relation"

$$x_{n+1} = x_n - \frac{(y - y_e) - \dot{y}(x - x_e)}{2x(x - x_e)\dot{y}^2}$$

where $x_0 = x_e$. Figure 24 shows the FORTRAN flow chart of the T-Algorithm with Newton Process.

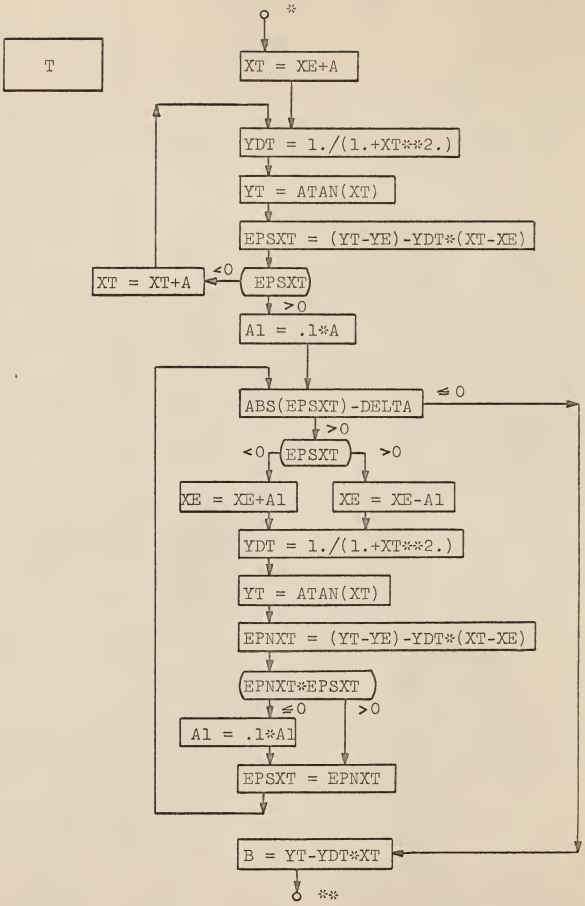

Fig. 23. FORTRAN flow chart of the T-Problem with DBD method.

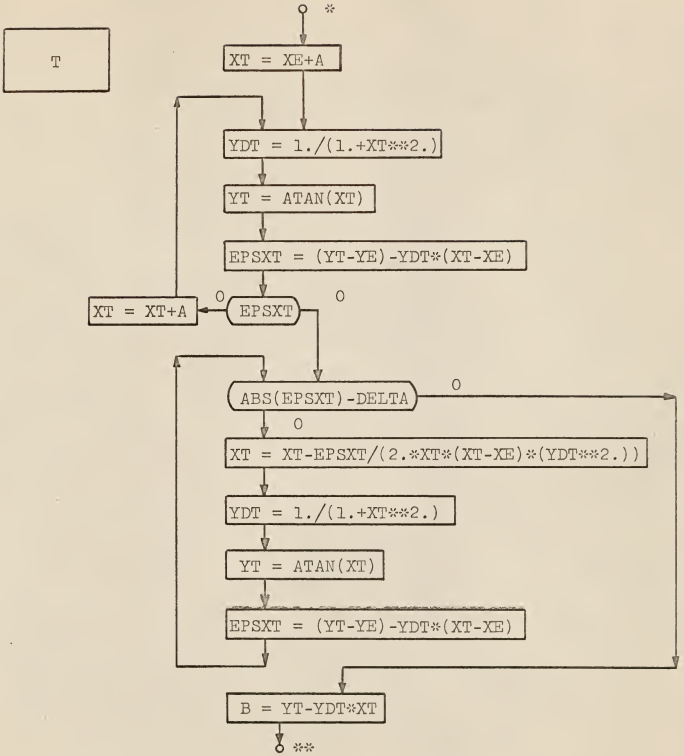

Fig. 24. FORTRAN flow chart of the T-Algorithm with Newton process.

The E-Algorithm

For an argument x, the error between the approximating tangent line and arctan(x) is:

$$e(x) = (mx + b) - \arctan(x).$$

The aim of the E-problem is to find the argument $x = x_e$ for which $e(x_e) = e_{max}$ holds, and therefore the E-problem reduces to finding a zero of the test function

$$h(x) = ((mx + b) - \arctan(x)) - e_{max} .$$

Input data for the "E-Algorithm" are x_t and y_t, obtained from the previously computed T-Algorithm. Output data of the "T-Algorithm" are x_e and y_e. Figure 25 shows the FORTRAN flow chart of the "E-Algorithm" with DBD Method.

Likewise, h(x) meets requirements for use of the "Newton Process". The first derivative of h(x) is

$$h'(x) = m - (1/(1 + x^2)).$$

The recurrence relation reads:

$$x_{n+1} = x_n - \frac{(mx_n + b) - \arctan(x_n) - e_{max}}{m - 1/(1 + x_n^2)}$$

where $x_0 = x_t$. Figure 26 shows the FORTRAN flow chart of the E-Algorithm with Newton Process.

The N-TRIP

The E-START, the E- and the T-algorithms are the building blocks for an algorithm to solve the N-TRIP problem. The 5-TRIP

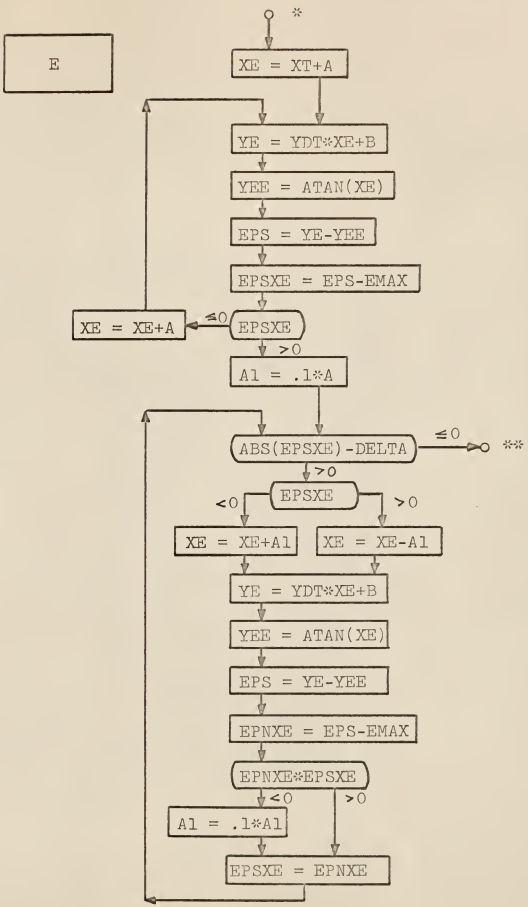

Fig. 25. FORTRAN flow chart of the E-Algorithm with DBD Method.

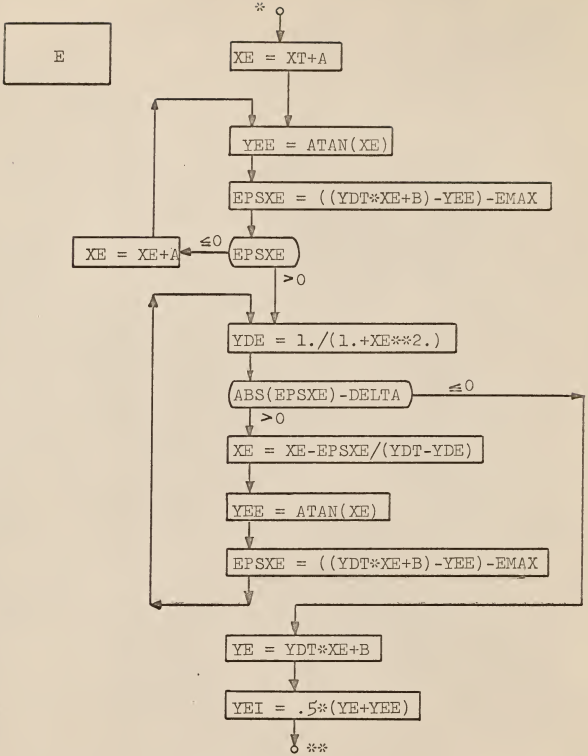

Fig. 26. FORTRAN flow chart of the E-Algorithm with Newton process.

with $e_{max} = 0.01$ serves as an example. First, the E-START problem is solved, yielding $\{x_{e1}, y_{e1}\}$, followed by the T problem, yielding $\{x_{t1}, y_{t1}\}$. Recurrent solving of the last two problems leads to the sequence:

$$\{x_{t2}, y_{t2}\},\ \{x_{e2}, y_{e2}\},\ \{x_{t3}, y_{t3}\},\ \{x_{t4}, y_{t4}\},$$

$$\{x_{e4}, y_{e4}\},\ \{x_{t5}, y_{t5}\},\ \{x_{e5}, y_{e5}\}.$$

Figure 27 shows the required FORTRAN flow chart.

The Asymptotic N-TRIP Algorithm

When $y_{eN+1} = \pi/2$, the N-TRIP becomes an Asymptotic N-TRIP. For the N-TRIP, y_{eN} is a function of e_{max}. Henceforth the test function reads:

$$h(x) = y_{eN+1}(e_{max}) = (\pi/2)$$

This test function is transcendental and satisfies the requirements for the Newton Process. Being a composite transcendental function, it is awkward to transform into a closed expression that would be needed for the speedy Newton Process. In this case the DBD Method proves to be more workable. It starts with a sufficiently small value for e_{max} for which h(x) is negative. In case of a 2-TRIP, $e_{max} = 0.01$ is sufficiently small. e_{max} grows gradually by increments of 0.01 until h(x) becomes negative. The procedure is then continued in the usual way (Figs. 28 and 29). The algorithm to find the zero of the above test function is called pi-half algorithm.

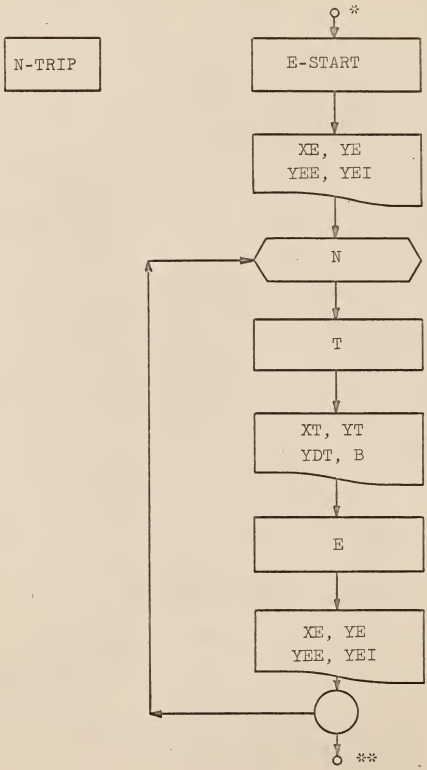

Fig. 27. FORTRAN flow chart of an N-TRIP.

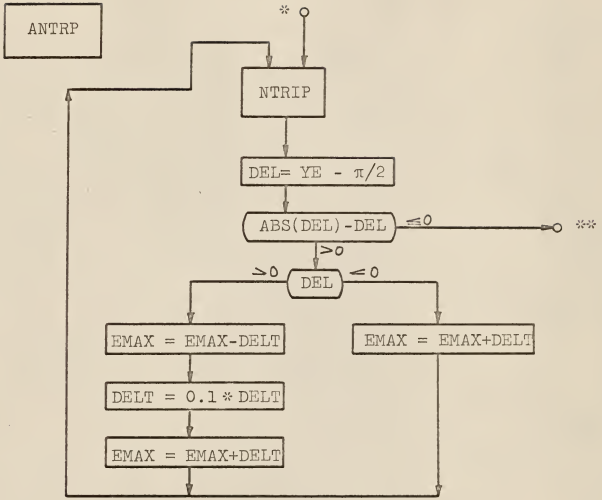

Fig. 28. FORTRAN flow chart of the asymptotic N-TRIP with unilateral SDP.

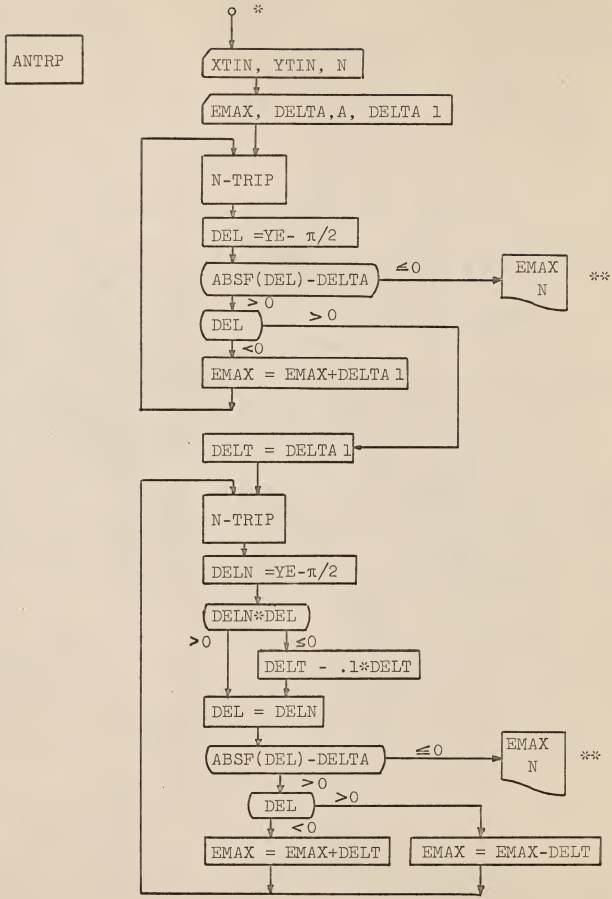

Fig. 29. Flow chart of asymptotic N-TRIP with oscillatory SDP.

Tables 2 through 11 show the results of the Asymptotic 1-, 2-, . . ., 10-TRIP. The way to use these tables is indicated for the Asymptotic 4-TRIP of Table 12. e_{max} = .401706E-01 is the value of the maximal error of the Asymptotic 4-TRIP, and "E-01" means 10^{-1} in standard mathematical notation. The first quadruple of the numbers XE, YE, YEE, YEI, called E-START values, reads in standard notation x_{e1}, y_{e1}, y_{ee1}, y_{ei1}. They are the result (output data) of the E-START algorithm and are used as input data for the ensuing T algorithm, whose output data are presented in the next quadruple of the numbers XT, YT, YDT, B. In standard notation, they are indicated by x_{t1}, y_{t1}, y_{t1}, m_{t1}. The next quadruple of E-values has subscript 2: x_{e2}, y_{e2}, y_{ee2}, y_{ei2}. The succeeding quadruple of E-values carries again subscript 2. With each new quadruple of E-values, the subscript increases by one. The last quadruple of numbers is x_{e5}, y_{e5}, y_{ee5}, y_{ei5}.

Tables 2 through 12 show the values of e_{max} for different Asymptotic N-TRIPS. M denotes the total number of tangent lines of the polygonal approximation for the interval $(0, \infty)$. N is the number of full tangents in $(0, \infty)$ (Table 13).

In the chapter on Quasi-equal Error Method it was recognized that the upper polygonal approximation for the second stage has maximum errors with uniform Cheybshev norm. This approximation is identical with the upper polygonal approximation of the Asymptotic 1-Trip. This assertion is inferred from the observation that both approximations employ the half tangent lines at $x = 0$ and $x = \infty$ and use the same number of full tangents. If the uniqueness of the Asymptotic 1-Trip approximation

is taken for granted, this fact implies the above identity.

The fact that the upper polygonal approximation of the second stage of the Quasi-equal Error Method has uniform Chebyshev norm, suggests to ask if the upper polygonal approximation of the third stage has maximum errors with uniform Chebyshev norm, too. This approximation corresponds to the Asymptotic 3-Trip. Similar reasoning as above leads to the assertion, "If the upper polygonal approximation of the third stage of the Quasi-equal Error Method has maximum error with uniform Chebyshev norm, then it is identical with the upper polygonal approximation of the Asymptotic 3-Trip." Since the full tangent line ${}_2t_1$ of the first stage approximation of the Quasi-equal Error Method is used as approximant in all other higher stages, the argument ${}_2x_{t1}$ of its point of tangency is identical with ${}_3x_{t2}$ (Fig. 9). This observation together with the two assertions mentioned above lead to the conclusion that argument x_{t1} of the full tangent line belonging to the Asymptotic 1-Trip, is equal to the argument x_{t2} of the middle tangent of the three full tangent lines of the Asymptotic 3-Trip. Approximate values for these arguments can be found in Tables 2 and 4, where

$$x_{t1} = 2.3878...$$

and

$$x_{t2} = 2.6311...$$

These two real numbers are different. Therefore it is proven that the upper polygonal approximation of the third stage of the Quasi-equal Error Method has maximum error with uniform Chevyshev norm must be negated.

Similar reasoning for the upper polygonal approximation at

the fourth stage of the Quasi-equal Error Method leads to the conclusion that this approximation corresponds to the Asymptotic 7-Trip and that x_{t1} must be equal to argument x_{t4} of the Asymptotic 7-Trip. Table 8 supplies the value of $x_{t4} = 2.82563$. As $x_{t1} \neq x_{t4}$, it is concluded that the above mentioned conjecture has to be negated for the fourth stage polygonal approximation of the Quasi-equal Error Method, too. A general negation proof is not given here. I conjecture such a proof would show that the progression $x_{t1(1\text{-Trip})}$, $x_{t2(3\text{-Trip})}$, $x_{t4(7\text{-Trip})}$, . . ., is monotonic.

Table 2. Asymptotic 1-TRIP.

EMAX =	.195652E-00		
XE =	.961343E 00	YE =	.961343E 00
YEE =	.765691E 00	YEI =	.863517E 00
XT =	.238783E 01	YT =	.117420E 01
YDT =	.149215E 00	B =	.817896E 00
XE =	.504573E 01	YE =	.157080E 01
YEE =	.137514E 01	YEI =	.147297E 01
XEN =	.504573E 01	YEN =	.157080E 01

Table 3. Asymptotic 2-TRIP.

EMAX =	.991702E-01		
XE =	.729274E 00	YE =	.729274E 00
YEE =	.630104E 00	YEI =	.679689E 00
XT =	.142475E 01	YT =	.958811E 00
YDT =	.330043E 00	B =	.488582E 00
XE =	.236947E 01	YE =	.127061E 01
YEE =	.117144E 01	YEI =	.122102E 01
XT =	.495861E 01	YT =	.137180E 01
YDT =	.390811E-01	B =	.117801E 01
XE =	.100506E 02	YE =	.157080E 01
YEE =	.147163E 01	YEI =	.152121E 01
XEN =	.100506E 02	YEN =	.157080E 01

Table 4. Asymptotic 3-TRIP.

EMAX =	.599589E-01		
XE =	.601427E 00	YE =	.601427E 00
YEE =	.541468E 00	YEI =	.[illegible]E 00
XT =	.107313E 01	YT =	.820661E 00
YDT =	.464766E 00	B =	.321904E 00
XE =	.163520E 01	YE =	.108189E 01
YEE =	.102193E 01	YEI =	.105191E 01
XT =	.263113E 01	YT =	.120759E 01
YDT =	.126217E 00	B =	.875498E 00
XE =	.407899E 01	YE =	.139034E 01
YEE =	.133038E 01	YEI =	.136036E 01
XT =	.828894E 01	YT =	.145073E 01
YDT =	.143459E-01	B =	.133182E 01
XE =	.166581E 02	YE =	.157080E 01
YEE =	.151084E 01	YEI =	.154082E 01
XEN =	.166581E 02	YEN =	.157080E 01

Table 5. Asymptotic 4-TRIP.

EMAX =	.401706E-01		
XE =	.518540E 00	YE =	.518540E 00
YEE =	.478370E 00	YEI =	.498455E 00
XT =	.882930E 00	YT =	.723304E 00
YDT =	.561935E 00	B =	.227154E 00
XE =	.128723E 01	YE =	.950496E 00
YEE =	.910325E 00	YEI =	.930411E 00
XT =	.187333E 01	YT =	.108047E 01
YDT =	.221760E 00	B =	.665039E 00
XE =	.262319E 01	YE =	.124676E 01
YEE =	.120659E 01	YEI =	.122667E 01
XT =	.405060E 01	YT =	.132876E 01
YDT =	.574468E-01	B =	.109606E 01
XE =	.616301E 01	YE =	.145011E 01
YEE =	.140994E 01	YEI =	.143002E 01
XT =	.124134E 02	YT =	.149041E 01
YDT =	.644780E-02	B =	.141037E 01
XE =	.248803E 02	YE =	.157080E 01
YEE =	.153062E 01	YEI =	.155071E 01
XEN =	.248803E 02	YEN =	.157080E 01

Table 6. Asymptotic 5-TRIP.

EMAX =	.287930E-01		
XE =	.459609E 00	YE =	.459609E 00
YEE =	.430816E 00	YEI =	.445212E 00
XT =	.760809E 00	YT =	.650383E 00
YDT =	.633380E 00	B =	.168502E 00
XE =	.108047E 01	YE =	.852853E 00
YEE =	.824060E 00	YEI =	.838456E 00
XT =	.149278E 01	YT =	.980566E 00
YDT =	.309751E 00	B =	.518175E 00
XE =	.198400E 01	YE =	.113272E 01
YEE =	.110393E 01	YEI =	.111832E 01
XT =	.275271E 01	YT =	.122234E 01
YDT =	.116585E 00	B =	.901417E 00
XE =	.375748E 01	YE =	.133948E 01
YEE =	.131069E 01	YEI =	.132509E 01
XT =	.571826E 01	YT =	.139767E 01
YDT =	.296749E-01	B =	.122798E 01
XE =	.863941E 01	YE =	.148435E 01
YEE =	.145556E 01	YEI =	.146996E 01
XT =	.173413E 02	YT =	.151319E 01
YDT =	.331432E-02	B =	.145572E 01
XE =	.347210E 02	YE =	.157080E 01
YEE =	.154200E 01	YEI =	.155640E 01
XEN =	.347210E 02	YEN =	.157080E 01

Table 7. Asymptotic 6-TRIP.

EMAX =	.216503E-01		
XE =	.415139E 00	YE =	.415139E 00
YEE =	.393489E 00	YEI =	.404314E 00
XT =	.674464E 00	YT =	.593382E 00
YDT =	.687331E 00	B =	.129801E 00
XE =	.941517E 00	YE =	.776935E 00
YEE =	.755285E 00	YEI =	.766110E 00
XT =	.126064E 01	YT =	.900186E 00
YDT =	.386217E 00	B =	.413305E 00
XE =	.162395E 01	YE =	.104050E 01
YEE =	.101885E 01	YEI =	.102968E 01
XT =	.213058E 01	YT =	.113196E 01
YDT =	.180526E 00	B =	.747339E 00
XE =	.274829E 01	YE =	.124348E 01
YEE =	.122182E 01	YEI =	.123265E 01
XT =	.374352E 01	YT =	.130976E 01
YDT =	.666047E-01	B =	.106043E 01
XE =	.505605E 01	YE =	.139718E 01
YEE =	.137553E 01	YEI =	.138636E 01
XT =	.764544E 01	YT =	.144074E 01
YDT =	.168201E-01	B =	.131214E 01
XE =	.115146E 02	YE =	.150582E 01
YEE =	.148417E 01	YEI =	.149499E 01
XT =	.230762E 02	YT =	.152749E 01
YDT =	.187437E-02	B =	.148423E 01
XE =	.461813E 02	YE =	.157080E 01
YEE =	.154914E 01	YEI =	.155997E 01
XEN =	.461813E 02	YEN =	.157080E 01

Table 8. Asymptotic 7-TRIP.

EMAX =	.168726E-01		
XE =	.380155E 00	YE =	.380155E 00
YEE =	.363283E 00	YEI =	.371719E 00
XT =	.609519E 00	YT =	.547389E 00
YDT =	.729122E 00	B =	.102976E 00
XE =	.840659E 00	YE =	.715919E 00
YEE =	.699046E 00	YEI =	.707483E 00
XT =	.110244E 01	YT =	.834085E 00
YDT =	.451390E 00	B =	.336454E 00
XE =	.139126E 01	YE =	.964455E 00
YEE =	.947582E 00	YEI =	.956018E 00
XT =	.176442E 01	YT =	.105518E 01
YDT =	.243122E 00	B =	.626209E 00
XE =	.219951E 01	YE =	.116096E 01
YEE =	.114408E 01	YEI =	.115252E 01
XT =	.282564E 01	YT =	.123065E 01
YDT =	.111306E 00	B =	.916138E 00
XE =	.359701E 01	YE =	.131651E 01
YEE =	.129963E 01	YEI =	.130807E 01
XT =	.485707E 01	YT =	.136775E 01
YDT =	.406650E-01	B =	.117023E 01
XE =	.652618E 01	YE =	.143562E 01
YEE =	.141875E 01	YEI =	.142719E 01
XT =	.983695E 01	YT =	.146949E 01
YDT =	.102285E-01	B =	.136887E 01
XE =	.147916E 02	YE =	.152016E 01
YEE =	.150329E 01	YEI =	.151173E 01
XT =	.296198E 02	YT =	.153705E 01
YDT =	.113852E-02	B =	.150332E 01
XE =	.592620E 02	YE =	.157080E 01
YEE =	.155392E 01	YEI =	.156236E 01
XEN =	.592620E 02	YEN =	.157080E 01

Table 9. Asymptotic 8-TRIP.

EMAX =	.135193E-01		
XE =	.351771E 00	YE =	.351771E 00
YEE =	.338252E 00	YEI =	.345012E 00
XT =	.558519E 00	YT =	.509360E 00
YDT =	.762228E 00	B =	.836414E-01
XE =	.763517E 00	YE =	.665615E 00
YEE =	.652096E 00	YEI =	.658856E 00
XT =	.986675E 00	YT =	.778691E 00
YDT =	.506707E 00	B =	.278736E 00
XE =	.122730E 01	YE =	.900616E 00
YEE =	.887097E 00	YEI =	.893857E 00
XT =	.152177E 01	YT =	.989426E 00
YDT =	.301587E 00	B =	.530479E 00
XE =	.185459E 01	YE =	.108980E 01
YEE =	.107628E 01	YEI =	.108304E 01
XT =	.229891E 01	YT =	.116049E 01
YDT =	.159109E 00	B =	.794717E 00
XE =	.282282E 01	YE =	.124385E 01
YEE =	.123033E 01	YEI =	.123709E 01
XT =	.358872E 01	YT =	.129904E 01
YDT =	.720517E-01	B =	.104046E 01
XE =	.453739E 01	YE =	.136739E 01
YEE =	.135387E 01	YEI =	.136063E 01
XT =	.609852E 01	YT =	.140827E 01
YDT =	.261835E-01	B =	.124859E 01
XE =	.817134E 01	YE =	.146254E 01
YEE =	.144902E 01	YEI =	.145578E 01
XT =	.122951E 02	YT =	.148964E 01
YDT =	.657156E-02	B =	.140884E 01

Table 9 (Concl.).

XE = .184717E 02	YE = .153023E 01
YEE = .151671E 01	YEI = .152347E 01
XT = .369729E 02	YT = .154376E 01
YDT = .730998E-03	B = .151673E 01
XE = .739638E 02	YE = .157080E 01
YEE = .155728E 01	YEI = .156404E 01
XEN = .739638E 02	YEN = .157080E 01

Table 10. Asymptotic 9-TRIP.

EXAM =	.110755E-01		
XE =	.328190E 00	YE =	.328190E 00
YEE =	.317114E 00	YEI =	.322652E 00
XT =	.517184E 00	YT =	.477300E 00
YDT =	.788967E 00	B =	.692587E-01
XE =	.702240E 00	YE =	.623303E 00
YEE =	.612227E 00	YEI =	.617765E 00
XT =	.897667E 00	YT =	.731525E 00
YDT =	.553769E 00	B =	.234424E 00
XE =	.110475E 01	YE =	.846202E 00
YEE =	.835126E 00	YEI =	.840664E 00
XT =	.134813E 01	YT =	.932583E 00
YDT =	.354932E 00	B =	.454090E 00
XE =	.161696E 01	YE =	.102800E 01
YEE =	.101692E 01	YEI =	.102246E 01
XT =	.195710E 01	YT =	.109842E 01
YDT =	.207029E 00	B =	.693242E 00
XE =	.234607E 01	YE =	.117895E 01
YEE =	.116787E 01	YEI =	.117341E 01
XT =	.287424E 01	YT =	.123598E 01
YDT =	.107977E 00	B =	.925627E 00
XE =	.350084E 01	YE =	.130363E 01
YEE =	.129256E 01	YEI =	.129810E 01
XT =	.442494E 01	YT =	.134854E 01
YDT =	.485906E-01	B =	.113353E 01
XE =	.557310E 01	YE =	.140433E 01
YEE =	.139325E 01	YEI =	.139879E 01
XT =	.747054E 01	YT =	.143773E 01
YDT =	.176028E-01	B =	.130622E 01

Table 10 (Concl.).

XE =	.999340E 01	YE =	.148214E 01
YEE =	.147106E 01	YEI =	.147660E 01
XT =	.150213E 02	YT =	.150432E 01
YDT =	.441228E-02	B =	.143804E 01
XE =	.225557E 02	YE =	.153757E 01
YEE =	.152649E 01	YEI =	.153203E 01
XT =	.451353E 02	YT =	.154864E 01
YDT =	.490629E-03	B =	.152650E 01
XE =	.902853E 02	YE =	.157080E 01
YEE =	.155972E 01	YEI =	.156526E 01
XEN =	.902853E 02	YEN =	.157080E 01

Table 11. Asymptotic 10-TRIP.

EMAX = .923941E-02

XE =	.308224E 00	YE =	.308224E 00
YEE =	.298984E 00	YEI =	.303604E 00
XT =	.482855E 00	YT =	.449838E 00
YDT =	.810932E 00	B =	.582753E-01
XE =	.652152E 00	YE =	.587126E 00
YEE =	.577886E 00	YEI =	.582506E 00
XT =	.826711E 00	YT =	.690817E 00
YDT =	.594018E 00	B =	.199736E 00
XE =	.100917E 01	YE =	.799204E 00
YEE =	.789964E 00	YEI =	.794584E 00
XT =	.121700E 01	YT =	.882969E 00
YDT =	.403047E 00	B =	.392459E 00
XE =	.144257E 01	YE =	.973885E 00
YEE =	.964645E 00	YEI =	.969265E 00
XT =	.171672E 01	YT =	.104334E 01
YDT =	.253349E 00	B =	.608410E 00
XE =	.202320E 01	YE =	.112098E 01
YEE =	.111174E 01	YEI =	.111636E 01
XT =	.241792E 01	YT =	.117864E 01
YDT =	.146063E 00	B =	.825470E 00
XE =	.287229E 01	YE =	.124501E 01
YEE =	.123577E 01	YEI =	.124039E 01
XT =	.349535E 01	YT =	.129214E 01
YDT =	.756572E-01	B =	.102770E 01
XE =	.423720E 01	YE =	.134827E 01
YEE =	.133903E 01	YEI =	.134365E 01
XT =	.533708E 01	YT =	.138557E 01
YDT =	.339162E-01	B =	.120456E 01

Table 11 (Concl.).

XE =	.670618E 01	YE =	.143201E 01
YEE =	.142277E 01	YEI =	.142739E 01
XT =	.897467E 01	YT =	.145983E 01
YDT =	.122632E-01	B =	.134977E 01
XE =	.119934E 02	YE =	.149685E 01
YEE =	.148761E 01	YEI =	.149223E 01
XT =	.180162E 02	YT =	.151535E 01
YDT =	.307141E-02	B =	.146001E 01
XE =	.270441E 02	YE =	.154307E 01
YEE =	.153384E 01	YEI =	.153846E 01
XT =	.541082E 02	YT =	.155232E 01
YDT =	.341448E-03	B =	.153384E 01
XE =	.108229E 03	YE =	.157080E 01
YEE =	.156156E 01	YEI =	.156618E 01
XEN =	.108229E 03	YEN =	.157080E 01

Table 12. Asymptotic 14-TRIP.

EMAX =	.509649E-02		
XE =	.251271E 00	YE =	.251271E 00
YEE =	.246175E 00	YEI =	.248723E 00
XT =	.387884E 00	YT =	.370019E 00
YDT =	.869222E 00	B =	.328609E-01
XE =	.517036E 00	YE =	.482280E 00
YEE =	.477183E 00	YEI =	.479731E 00
XT =	.642499E 00	YT =	.571083E 00
YDT =	.707812E 00	B =	.116315E 00
XE =	.769304E 00	YE =	.660838E 00
YEE =	.655742E 00	YEI =	.658290E 00
XT =	.903179E 00	YT =	.734569E 00
YDT =	.550742E 00	B =	.237150E 00
XE =	.104251E 01	YE =	.811306E 00
YEE =	.806210E 00	YEI =	.808758E 00
XT =	.119669E 01	YT =	.874699E 00
YDT =	.411173E 00	B =	.382653E 00
XE =	.136037E 01	YE =	.941999E 00
YEE =	.936902E 00	YEI =	.939451E 00
XT =	.154806E 01	YT =	.997261E 00
YDT =	.294420E 00	B =	.541480E 00
XE =	.175066E 01	YE =	.105691E 01
YEE =	.105181E 01	YEI =	.105436E 01
XT =	.199056E 01	YT =	.110525E 01
YDT =	.201519E 00	B =	.704118E 00
XE =	.225361E 01	YE =	.115826E 01
YEE =	.115317E 01	YEI =	.115571E 01
XT =	.257528E 01	YT =	.120041E 01
YDT =	.131026E 00	B =	.862982E 00
XE =	.293389E 01	YE =	.124740E 01
YEE =	.124230E 01	YEI =	.124485E 01
XT =	.338813E 01	YT =	.128380E 01
YDT =	.801319E-01	B =	.101230E 01

Table 12 (Concl.).

XE =	.390412E 01	YE =	.132514E 01
YEE =	.132005E 01	YEI =	.132259E 01
XT =	.458541E 01	YT =	.135607E 01
YDT =	.454009E-01	B =	.114789E 01
XE =	.537717E 01	YE =	.139202E 01
YEE =	.138692E 01	YEI =	.138947E 01
XT =	.647893E 01	YT =	.141766E 01
YDT =	.232685E-01	B =	.126690E 01
XE =	.779798E 01	YE =	.144835E 01
YEE =	.144325E 01	YEI =	.144580E 01
XT =	.976967E 01	YT =	.146879E 01
YDT =	.103684E-01	B =	.136750E 01
XE =	.122312E 02	YE =	.149431E 01
YEE =	.148922E 01	YEI =	.149177E 01
XT =	.163264E 02	YT =	.150962E 01
YDT =	.373759E-02	B =	.144860E 01
XE =	.217835E 02	YE =	.153002E 01
YEE =	.152492E 01	YEI =	.152747E 01
XT =	.326896E 02	YT =	.154021E 01
YDT =	.934918E-03	B =	.150965E 01
XE =	.490453E 02	YE =	.155551E 01
YEE =	.155041E 01	YEI =	.155296E 01
XY =	.981012E 02	YT =	.156060E 01
YDT =	.103898E-03	B =	.155041E 01
XE =	.196208E 03	YE =	.157080E 01
YEE =	.156570E 01	YEI =	.156825E 01
XEN =	.196208E 03	YEN =	.157080E 01

Table 13. Maximal errors for polygonal approximations with Chebyshev norm and zero boundary error.

M	:	N	:	EMAX
1				1.570796
2		0		0.566912
3		1		0.195652
4		2		0.0991702
5		3		0.0599589
6		4		0.0401706
7		5		0.0287930
8		6		0.0216503
9		7		0.01687258
10		8		0.01351934
11		9		0.01107548
12		10		0.00923948
13		11		
14		12		
15		13		
16		14		0.00509649

REALIZATION OF THE ARCTANGENT SPC

Fundamental Idea

This chapter conveys the basic idea for constructing an SPC, but does not give a hardware design of an Arctangent SPC. It will lead to a better understanding of the simulation of an Arctangent SPC on a GPC.

Figure 30 shows a schematic diagram of an SPC with upper polygonal approximation. Next to the input, which is assumed to have pulse form, is an encoder (enc.) unit translating the input signal x into machine language. There the signal path transmitting x splits up in two paths.

The first path leads to the decision network (dec. netw.). There the signal x is compared at different decision stations with precomputed signal levels. The number of outputs equals the number of pairs (m_n, b_n) that are stored in the nonerasable memory (non-er.mem.). After having passed several decision stages, the input signal reaches one of the outputs of the decision network. Here a driving pulse is generated and reads out the information, which thereupon is transferred to the arithmetic control (arith. con.) unit.

The second path transmits x through a delay (del.) unit to the accumulator (acc.), where the arithmetic operations specified by the expression

$$y = m_n \cdot x + b_n$$

are executed. The final result y passes through a decoding (dec.) unit, where y becomes the desired output form.

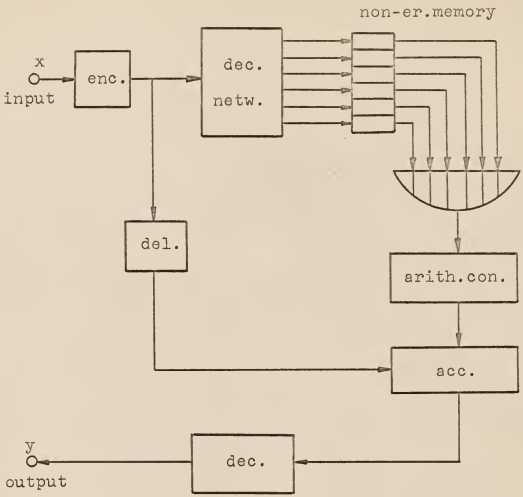

Fig. 30. Schematic diagram of an SPC, using polygonal approximation.

Thus the main parts of this computer are the accumulator, the nonerasable memory, and the decision network. The preceding chapter included all the numerical work used for the realization (hardware design) or the simulation (software programming). The following chapter takes up some details about the decision network.

The Decision Network

As argument x enters the decision network, it is compared to the different precomputed levels x_{e1}, x_{e2}, . . ., x_{eN}, x_{eN+1}. Thus the appropriate parameter pair (m_n, b_n) is read out from the immediately following nonerasable memory. The comparisons can be executed in many ways. Here, two extreme cases, case (a) and case (b), are considered.

Before, it should be recalled that the maximal error arguments x_{e1}, x_{e2}, . . ., x_{eN}, x_{eN+1} are the boundaries of the intervals $I_0 = (0, x_{e1})$, $I_1 = (x_{e1}, x_{e2})$, . . ., $I_N = x_{eN}, x_{eN+1}$, $I_\infty = (x_{eN+1}, x_{e\infty})$, where $I = I_0 \cup I_1 \cup \ldots \cup I_N \cup I_\infty = (0,\infty)$.

Case (a)--The Search Method. This method takes the absolute value of the argument x and saves its sign. Then it continues with the left most interval $I_0 = (0, x_{e1})$ of $I = (0, \infty)$. Here at the first decision station, it is decided whether $x \in I_0$ or $x \in I_1 \cup I_2 \cup \ldots \cup I_N \cup I_\infty$. In case $x \in I_0$, argument x generates a pulse that is transmitted to the corresponding place in the nonerasable memory for read-out purpose.

In case $x \in I_1 \cup I_2 \cup \ldots \cup I_N \cup I_\infty$, x is compared at the next decision station with $x_{\theta 2}$, where either $x \in I_1$ or $x \in I_2 \cup \ldots \cup I_N \cup I_\infty$. From here the procedure, as described for the first decision station, starts afresh.

If x lies in the right most interval I_∞ , x passes through N+1 decision levels. As it takes time to execute these decisions, the processing time for $x \in I_N$ is much longer than for $x \in I_0$, where only one decision has to be made. Thus the decision time depends largely on the magnitude of the argument x. This is not an expedient feature if equal overall computation time is required regardless of the size of the argument x. The advantage of this method is that its decision process can be programmed as a loop, called "Search Loop". Figure 31 shows a flow chart of the search method and Fig. 32 its representation as search loop.

Case (b)--The Tree Method. Real-time control systems often demand equal overall computation time for their SPC's, regardless of the magnitude of the input argument. The computation time (CT) of the ARCTANGENT SPC consists mainly of decision CT and arithmetic CT. A workable way to satisfy the demand of equal overall CT is to make decision CT and arithmetic CT constant for all possible arguments x.

For the arithmetic unit, this is done by not taking advantage of special situations, as for $x \doteq 0$ or $x \doteq \infty$, where multiplication by 1 and addition of 0 become trivial. These special cases

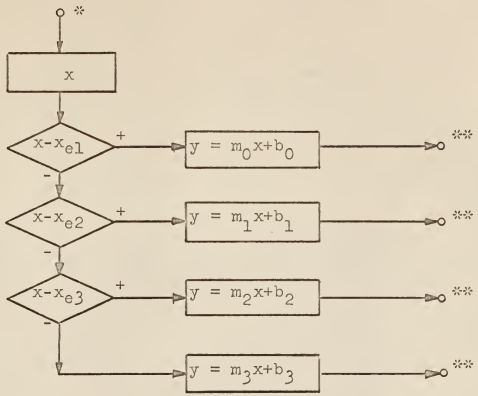

Fig. 31. Flow chart of the search method with tree decision levels.

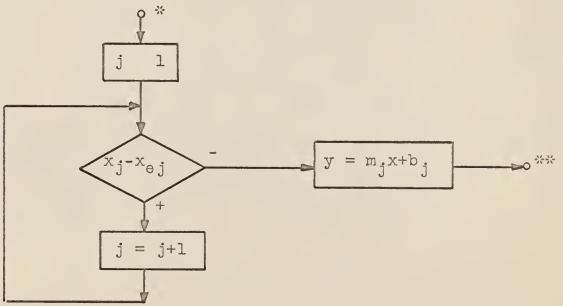

Fig. 32. Flow chart for the search loop method ($x_{eN+1} = x_e = .999999E+99$).

$x \doteq 0$: $y = 1.00000 \cdot x + 0.00000$

$x \doteq \infty$: $y = 0.00000 \cdot x + 1.57080$

are treated as the others with no such trivial values.

The decision CT can be equalized by using the tree method. Argument x passes through the same number of decision levels, regardless of the magnitude of x.

Figure 33 shows a decision tree with three decision levels. This is appropriate if $M = 2^3 = 8$ outputs are required, or in general $M = 2^k$ ($k = 1, 2, 3, \ldots$). In case $M \neq 2^k$ ($k = 1, 2, 3, \ldots$), degenerated decisions can be introduced. A degenerated decision element has outputs which lead to the same destination. (Fig. 34.)

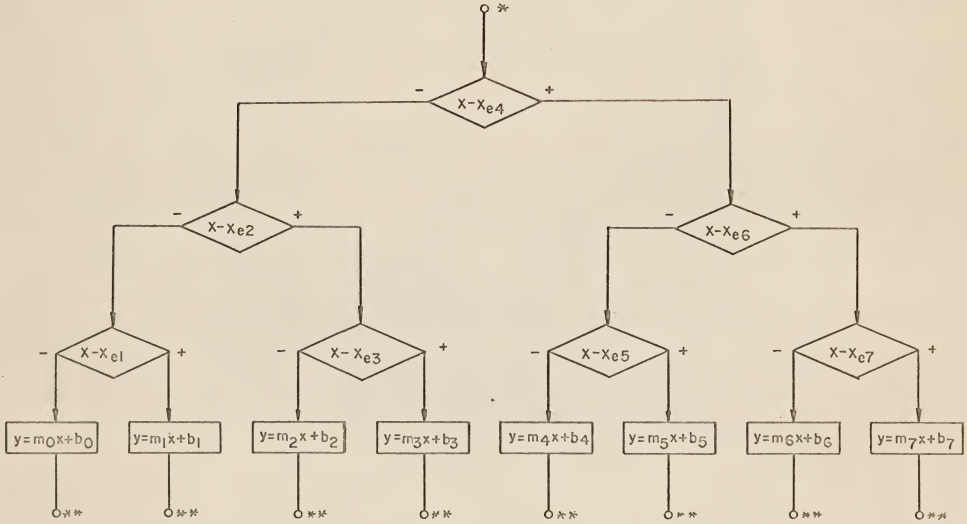

Fig. 33. Decision Tree with three decision levels.

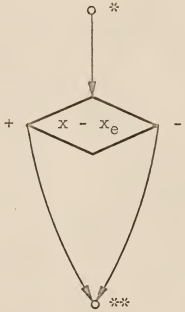

Fig. 34. Degenerate decision.

ACKNOWLEDGMENTS

The author wishes to express his sincere appreciation to his major professor, Dr. C. A. Halijak, for helpful assistance throughout the author's graduate study period, for suggesting the topic of this dissertation, and for his assistance in the preparation of this thesis.

The careful typing of the manuscript by Mrs. F. M. Crawford is gratefully acknowledged.

REFERENCES

1. Taub, D. M.
A short review of read-only memories. IEE Proceedings, Vol. 110, No. 1, January, 1963, pp. 157-166.

2. Taub, D. M.
The design of transformer (Dimond ring) read-only stores. IBM Journal of Research and Development, September, 1964, pp. 443-459.

3. Foglia, H. R., W. L. McDermid, H. E. Petersen.
Card capacitor--a semipermanent read-only memory. IBM Journal of Research and Development, January, 1961, pp. 67-68.

4. Haskell, J. W.
Printed cards for the card capacitor memory. IBM Journal of Research and Development, October, 1962, pp. 462-463.

5. Artin, E.
The gamma function. Holt, Rinehart, and Winston, Inc., pp. 1-10.

6. V. Mangoldt-Knopp.
Einfuhrung in die hohere mathematik, Bd. II, S. Hirzel Verlag, Stuttgart, pp. 137-139.

7. United States Department of Commerce.
Table of arctan x, The National Bureau of Standards, Applied Mathematics, Series 26.

8. Leeson, D. N., and D. L. Dimitry.
The IBM 1620 Computer. Holt, Rinehart, and Winston, Inc., 1962.

APPENDICES

APPENDIX A

Calculation of the Argument with Minimum Curvature

The derivatives of $f(x) = \arctan(x)$ are:

$$f'(x) = 1/(1 + x^2) \tag{1}$$

$$f''(x) = (-2x)/(1 + x^2)^2 \tag{2}$$

$$f'''(x) = (6x^2 - 2)/(1 + x^2)^3 \tag{3}$$

The curvature of an analytic function $f(x)$ is given by the equation:

$$\text{curv}(x) = \frac{f''(x)}{(1 + (f'(x))^2)^{3/2}}$$

This equation derived with respect to x yields:

$$\frac{d(\text{curv}(x))}{dx} = \frac{f'''(x) \cdot (1 + (f'(x))^2)^{3/2} - 3f'(x)(f''(x))^2 (1 + (f(x))^2)^{1/2}}{(1 + (f'(x))^2)^3}$$

The right member of this equation vanishes at the minimum of curvature. The inequality

$$1 + (f'(x))^2 \neq 0$$

should be noticed. By taking into account both remarks, one obtains:

$$f'''(x) \cdot (1 + (f'(x))^2) - 3f'(x)(f''(x))^2 = 0 \tag{4}$$

Substitution of $f'(x)$, $f''(x)$, $f'''(x)$ in equation (4) by equations (1), (2), and (3) yields:

$$\frac{6x^2 - 2}{(1 + x^2)^3}\left(1 + \frac{1}{(1 + x^2)^2}\right) - 3\,\frac{1}{1 + x^2} \cdot \frac{4x^2}{(1 + x^2)^4} = 0$$

or

$$(6x^2 - 2)((1 + x^2)^2 + 1) - 12x^2 = 0 .$$

Further reduction yields the equation

$$x^2(6x^2 - 2)(2 + x^2) - 4 = 0 .$$

The substitution $x^2 = t$ yields

$$t(6t - 2)(2 + t) - 4 = 0$$

or

$$6t^3 + 10t^2 - 4t - 4 = 0 .$$

The next chore is to find the zero crossing of

$$h(t) = 6t^3 + 10t^2 - 4t - 4 . \qquad (5)$$

As $h(0) = -4$ and $h(1) = 8$, the zero crossing of (5) lies in the interval (0, 1). Further calculation yields an approximate value for the zero crossing:

$$t = 0.691$$

and

$$x = \sqrt{0.691} = 0.831$$

is the abscissa value, where arctan(x) has minimum curvature.

APPENDIX B

Sufficient Conditions for Convergence of the Newton Process

Let "$\square$" be a general symbol for either one of the linear order relations "$<$" or "$>$", and let "$\sqsubseteq$" be a general symbol for either one of the partial order relations "$\leq$" or "$\geq$". If "$\square$" is identified with "$<$", then "$\sqsubseteq$" is to be identified with "$\leq$"; and if "$\square$" is identified with "$>$", then "$\sqsubseteq$" is to be identified with "$\geq$".

Definition 1. A function $h(x)$ defined in an interval (x_a, x_b) is monotonic, if for all $x_1, x_2 \in (x_a, x_b)$ the relation $x_1 \square x_2$ implies $h(x_1) \sqsubseteq h(x_2)$.

Definition 2. A function $h(x)$ defined in an interval (x_a, x_b) is strictly monotonic, if for all $x_1, x_2 \in (x_a, x_b)$ the relation $x_1 \square x_2$ implies $h(x_1) \square h(x_2)$.

For the next two definitions, the general linear order relation "$\square$" is identified with the specific linear order relation "$<$", and the general partial order relation "$\sqsubseteq$" is identified with the specific partial order relation "$\leq$".

Definition 3. A function $h(x)$ defined in an interval (x_a, x_b) is increasing if for all $x_1, x_2 \in (x_a, x_b)$ the relation $x_1 < x_2$ implies $h(x_1) \leq h(x_2)$.

Definition 4. A function $h(x)$ defined in an interval (x_a, x_b) is strictly increasing, if for all $x_1, x_2 \in (x_a, x_b)$ the relation $x_1 < x_2$ implies $h(x_1) < h(x_2)$. Similarly, the dual

notions decreasing function and strictly decreasing function are defined.

Definition 5. Let $h(x)$ be a continuous function defined in the interval $I = (x_a, x_b)$, and let $I_1 = (x_1, x_2)$ be an open interval such that $I_1 \subseteq I$. The points $\{x_1, h(x_1)\}$ and $\{x_2, h(x_2)\}$ determine a secant line $s_{x_1x_2}(x)$. The function $h(x)$ is convex in $I = (x_a, x_b)$, if for all intervals $I_1 \subseteq I$ and for all $x \in I_1$ the inequality $s_{x_1x_2}(x) - h(x) \geqslant 0$ holds.

Definition 6. Let $h(x)$ be a continuous function defined in the closed interval $I = (x_a, x_b)$, and let $I_1 = (x_1, x_2)$ be an open interval such that $I_1 \subseteq I$. The points $\{x_1, h(x_1)\}$ and $\{x_2, h(x_2)\}$ determine a secant line $s_{x_1x_2}(x)$. The function $h(x)$ is strictly convex in $I = (x_a, x_b)$, if for all intervals $I \supseteq I_1$ and for all $x \in I_1$ the inequality $s_{x_1x_2}(x) - h(x) > 0$ holds.

Strict monotonicity excludes horizontal line pieces in $h(x)$; however, it does not exclude horizontal tangent lines. Strict convexity excludes the equation of a straight line as a convex function. However, strict convexity is not required for the Newton process, as will be seen later. All dual definitions of terms like monotonically decreasing function, concave function, etc., as well as the corresponding dual theorems are omitted, as both kinds are easily obtained by applying duality rules. For the same reason, the convergence of the Newton process is only proved for one out of four cases where $h(x)$ is convex and increasing. In the subsequent theorem, the notion "continuous differentiability" will appear. This is not a

pleonasm. The attribute "continuous" excludes infinite derivatives, i.e., tangent lines that are perpendicular to the x-axis.

Theorem 1. Let $h(x)$ be a convex, continuous, continuously differentiable, and strictly monotonic function in the closed interval (x_a, x_b). If $h(x_a) < 0$ and $h(x_b) > 0$, there is a unique real number x_0 in the interval (x_a, x_b) for which $h(x_0)$ holds. If the Newton process is started at $x = x_b$, then it converges to x_0.

Proof. Part A (Existence of x_0). Continuity of $h(x)$ and the satisfied condition sign $(x_a) \neq$ sign (x_b) imply, by means of a theorem of Bolzano that can be found in reference (6, Bd. 1, page 538), the existence of an argument x_0 in an interval (x_a, x_b) where $h(x_0) = 0$. Part A Q.E.D.

Proof. Part B (Uniqueness of x_0). If it is assumed that $h(x)$ has two zero crossings, x_{01} and x_{02}, then by Rolle's theorem there exists an argument x_m such that $x_{01} < x_m < x_{02}$ and $h'(x_m) = 0$. $h(x_m)$ is the maximum of $h(x)$ in interval (x_{01}, x_{02}). By definition of such a maximum $h(x_m)$, the function $h(x)$ decreases for some $x > x_m$. This contradicts that $h(x)$ is a strictly increasing function. Hence $h(x)$ has exactly one zero. Part B Q.E.D.

Lemma 1. Under the assumptions of Theorem 1, $h(x) > 0$, if x lies in the interval (x_0, x_b), and $h(x) > 0$ if x lies in the interval (x_a, x_0).

Proof. As $f(x_0) = 0$, strict monotonicity does not tolerate any sign change inside the intervals (x_a, x_0) and (x_0, x_b). Q.E.D.

Lemma 2. Under the assumptions of Theorem 1, the slope $m(x_1, x_2)$ of $s_{x_1x_2}(x)$ is a monotonically increasing function, if $s_{x_1x_2}(x)$ is a secant line through the points $\{x_1, h(x_1)\}$ and $\{x_2, h(x_2)\}$, where $x_1 < x_2$. It does not matter whether x_1 increases and x_2 is constant or x_2 increases and x_1 is constant.

Proof. Let us consider the case where x_2 is constant and x_1 increases in the interval (x_a, x_2). Lemma 2 is proved by showing that for two positions x_{11} and x_{12} of argument x_1 where $x_{11} < x_{12}$, the slope $m(x_{11}, x_2)$ is less than the slope $m(x_{12}, x_2)$, where $x_{11} < x_{12}$. The definition of convexity, symmetry, and monotonicity is applied to the secant line $s_{x_{11}x_2}(x)$. Accordingly, at argument x_{12} holds

$$d = s_{x_{11}x_2}(x_{12}) - h(x_{12}) > 0 . \tag{1}$$

The slope of secant line $s_{x_{12}x_2}(x)$ is:

$$m(x_{12}, x_2) = \frac{h(x_2) - h(x_{12})}{x_2 - x_{12}} . \tag{2}$$

The slope of secant line $s_{x_{11}x_2}(x)$ is:

$$m(x_{11}, x_2) = \frac{(h(x_2) - h(x_{12})) - d}{x_2 - x_{12}} . \tag{3}$$

Since $d > 0$, and the other differences are by definition positive, it follows that

$$m(x_{11}, x_2) < m(x_{12}, x_2). \tag{4}$$

Therefore in case x_2 is constant, and x_1 increases. Lemma 3 is valid. In the other case where x_1 is constant and x_2 increases,

similar reasoning leads to the inequality

$$m(x_1, x_{21}) < m(x_1, x_{22}) , \qquad (5)$$

where $x_1 < x_{21} < x_{22} < x_b$. Q.E.D.

Lemma 3. Under the assumptions of Theorem 1, the only points of intersection in the interval (x_a, x_b) between the function $h(x)$ and the secant line $s_{x_1x_2}(x)$ are the points $\{x_1, h(x_1)\}$ and $\{x_2, h(x_2)\}$.

Proof. It is supposed that there is a third intersection point with an argument x_{in} that lies in the interval (x_a, x_1) such that $x_a < x_{in} < x_1$. An argument x_{11} begins to increase at $x_{11} = x_{in}$, where the slope of the secant line $m(x_{11}, x_1) = m(x_{in}, x_1) = m(x_1, x_2)$. As x_{11} increases, $m(x_{11}, x_1)$ increases according to Lemma 2. x_{11} increases until it reaches x_1. There secant line $s_{x_{11}x_1}(x)$ becomes tangent line ${}_0t_{x1}(x)$ with the slope ${}_0m(x_1)$. By Lemma 2 holds:

$${}_0m(x_1) > m(x_1, x_2) . \qquad (1)$$

The existence of this tangent line is guaranteed by the assumed continuous differentiability of $h(x)$. The proof goes on with an argument x_{12} beginning to decrease at $x_{12} = x_2$, where the slope of the secant line $m(x_1, x_{12}) = m(x_1, x_2)$. x_{12} decreases until it reaches x_1, where the secant line $s_{x_{12}x_2}(x)$ becomes tangent line ${}_1t_{x_1}(x)$ with slope ${}_1m(x_1)$. From Lemma 2 can be inferred that

$${}_1m(x_1) < m(x_1, x_2) . \qquad (2)$$

Inequalities (1) and (2) yield inequality

$${}_0m(x_1) \neq {}_1m(x_1) .$$

This means that there exist two tangent lines for $h(x)$ at x_1. This contradicts the previously assumed continuous differentiability of $h(x)$. Q.E.D.

From Lemma 3 follows immediately an interesting property of continuously differentiable functions $h(x)$. By decreasing x_2 toward x_1, such that after having taken the limit $x_2 \longrightarrow x_1$, secant line $s_{x_1x_2}(x)$ becomes tangent line $t_{x_1}(x)$. This leads to Lemma 4, a modified form of Lemma 3.

Lemma 4. Under the assumptions of Theorem 1 and if $t_{x_1}(x)$ is a tangent line of $h(x)$ with point of tangency at $\{x_1, h(x_1)\}$, the only common point of $h(x)$ and $t_{x_1}(x)$ is the point of tangency at $\{x_1, h(x_1)\}$ in the interval (x_a, x_b) such that for all $x \in (x_a, x_b) - \{x_1\}$ holds:

$$h(x) - t_{x_1}(x) > 0 .$$

Lemma 5. Under the assumptions of Theorem 1, the first derivative $h'(x)$ is a monotonically increasing function.

Proof. Let us consider an interval (x_1, x_2) that is contained in interval (x_a, x_b). The application of Lemma 2 to tangent line $t_{x_1}(x)$ and secant line $s_{x_1x_2}(x)$, and to secant line $s_{x_1x_2}(x)$ and tangent line $t_{x_2}(x)$ yields:

$$m(t_{x_1}) < m(s_{x_1x_2}) < m(t_{x_2}) .$$

This implies

$$m(t_{x_1}) < m(t_{x_2}) .$$

As $m(t_{x_1}) = h'(x_3)$ and $m(t_{x_2}) = h'(x_2)$, the above inequality yields:

$$h'(x_1) < h'(x_2) .$$

In other words, $x_1 < x_2$ implies $h'(x_1) < h'(x_2)$. Q.E.D.

Lemma 6. Under the assumptions of Theorem 1, zero x_0 of the function $h(x)$ is a zero crossing.

Proof. By Lemma 5, x, the derivative of $h(x)$ at x_0, is different from zero. Q.E.D.

Lemma 7. Under the assumptions of Theorem 1, the only zero derivative $h'(x) = 0$ occurs at argument x_a, and the inequality $h'(x) > 0$ holds in the left half interval (x_a, x_b).

Proof. Let us consider the secant line $s_{x_a x_1}(x)$, where $x_a < x_1 < x_b$. The slope of this secant line is:

$$m(x_a, x_1) = \frac{h(x_a) - h(x_1)}{x_a - x_1}$$

The strictly increase of $h(x)$ implies that the above quotient is positive for all x_1's in left open interval (x_a, x_b). As x_1 tends to x_a, one obtains

$$\lim_{x_1 \to x_a} \frac{h(x_a) - h(x_1)}{x_a - x_1} = h'(x_a) \geq 0$$

The limit has the possibility to take to the value zero. This proves the first assertion. As $h'(x)$ is strictly increasing by Lemma 5 in closed interval (x_a, x_b) and as $h'(x_a) \geq 0$, $h(x)$ is positive throughout interval (x_a, x_b). Q.E.D.

Lemma 8. Under the assumptions of Theorem 1, $h'(x)$ is never infinite throughout (x_a, x_b).

Proof. This follows immediately from continuous differentiability, which excludes $h'(x) = \infty$.

Lemma 9. If argument x_1 lies in the left open interval (x_0, x_b), where x_0 is the zero crossing of $h(x)$, then the

tangent line $t_{x_1}(x)$ intersects the x-axis at x_2 such that $x_0 < x_2 < x_1$.

Proof. By Lemma 4 holds $h(x) - t_{x_1}(x) > 0$. Therefore $h(x_2) - t_{x_1}(x_2) > 0$. Here Part A and Part B of Theorem 1 are applied to interval (x_a, x_2). According to these proved assertions, there is a unique zero crossing x_0 in the interval (x_a, x_b). Continuous differentiability of $h(x)$ prohibits $x_0 = x_2$ by Lemma 8. As $h(x_2) > 0$ and $h'(x) > 0$, the relation $x_0 < x_2$ holds. The same reasoning applied to argument x_1 leads to the inequality $x_2 < x_1$. Q.E.D.

Proof. Part C (Convergence of Newton Process). Let us start the procedure known as Newton process at $x = x_b = x_1$. The intersection point x_2 of the tangent line $t_{x_1}(x)$ and the x-axis are given by

$$x_2 = x_1 - \frac{h(x_1)}{h'(x_1)} \quad .$$

x_2 lies in the open interval $x_0 < x_2 < x_1$ by Lemma 9. Inequality $x_0 < x_2$ and strict increase of $h(x)$ imply $h(x_2) > 0$. Therefore this procedure can be applied again to argument x_2. This iterative procedure, called a Newton Process, yields the recurrence relation

$$x_{n+1} = x_n - \frac{h(x_1)}{h'(x_1)} \qquad (n = 1, 2, 3, \ldots)$$

The repetitive execution of this relation yields the chain inequality

$$x_0 < \ldots x_3 < x_2 < x_1$$

where x_1 is the starting point of the Newton process. By

choosing a sufficiently high value for n, the above chain inequality enables to satisfy the relation

$$x_n - x_0 < \varepsilon$$

whichever small positive value ε has. In other words,

$$\lim_{n \to \infty} x_n = x_0 .$$

This completes the proof of Part C of Theorem 1.

Definition 7. A Newton process for a function f(x) defined in the interval (x_a, x_b) and with a zero crossing at x_0 such that $x_0 \in (x_a, x_b)$, is strictly convergent, if it converges to x_0 for any starting point $x_1 \in (e_0, x_b)$.

Assumptions of Theorem 1 are sufficient for strict convergence of the Newton process in interval (x_a, x_b). It should be observed that the above described Newton process does not make use of any assumption of Theorem 1 for the interval (x_a, x_0). Therefore Theorem 1 is restated with slightly weaker assumptions.

Theorem 2. If a function h(x) has a zero and a positive derivative at x_0 and if h(x) is continuous, convex, strictly increasing, and continuously differentiable in the interval (x_0, x_b), then the Newton process is strictly convergent for interval (x_a, x_b).

It should be noted that in the numerical computation of a Newton process, assumptions stated above might be satisfied and yet the calculation might become unstable. The reason for this instability is that the above theory does not take account of the finiteness of registers and computation time.

APPENDIX C

FORTRAN Program of Asymptotic N-Trip with Newton Process in E-Start, T-, and E-Algorithm and with Unilateral Dit-by-dit Method in Pi Half Algorithm

```
   MON$$      JOB  ASYMPTOTIC  N TRIP (NP) UNILAT APPR
   MON$$      COMT 10 MINUTES, 10 PAGES
   MON$$      ASGN MJB,12
   MON$$      ASGN MGO,16
   MON$$      MODE GO,TEST
   MON$$      EXEQ FORTRAN,,,10,03,,,ASYMPNTRIP
800 FORMAT(I3)
801 FORMAT(1H1,30H ASYMPTOTIC  N TRIP(NP)  N =  ,I3)
803 FORMAT(1HL,10H   EMAX = ,E20.14,10H   DELT = ,E20.14)
805 FORMAT(1HT,10H  DELTA = ,E20.14)
802 FORMAT(10X          ,E20.14,10X          ,E20.14)
804 FORMAT(10X          ,E20.14)
820 FORMAT(1HK,10H     XE = ,E20.14,10H     YE = ,E20.14)
821 FORMAT(1HT,10H    YEE = ,E20.14,10H    YEI = ,E20.14)
830 FORMAT(1H ,10H     XT = ,E20.14,10H     YT = ,E20.14)
831 FORMAT(1HS,10H    YDT = ,E20.14,10H      B = ,E20.14)
855 FORMAT(1HT,10H    XEN = ,E20.14,10H    YEN = ,E20.14)
860 FORMAT(1HL,10H   EMAX = ,E20.14)
861 FORMAT(1HL,10H EMAXEN = ,E20.14)
    READ  (1,800) N
    WRITE (3,801) N
    READ  (1,802) EMAX, DELT
    WRITE (3,803) EMAX, DELT
    READ  (1,804) DELTA
    WRITE (3,805) DELTA
          A = 1.0
  1 WRITE (3,860) EMAX
    CALLFLTPT(EMAX)
         XT = 0.0
         YT = 0.0
        YDT = 1.0
          B = 0.0
         XE = A
  2      YEE = ATAN(XE)
      EPSXE = (XE-YEE) - EMAX
    IF(EPSXE)                          3,   4,   4
  3        XE = XE + A
    GO TO   2
  4      YDE = 1./(1. + XE**2.)
    IF(ABS(EPSXE) - DELTA)            12,  12,   6
  6        XE = XE - EPSXE/(1.0-YDE)
         YEE = ATAN(XE)
      EPSXE = (XE-YEE) - EMAX
    GO TO   4
 12        YE = XE
         YEI = .5*(YE+YEE)
    WRITE(3,820)   XE,   YE
    WRITE(3,821)  YEE,  YEI
```

```
      DO  91 J=1,N
           XT = XE + A
   22      YDT = 1./(1. + XT**2.)
           YT = ATAN(XT)
        EPSXT = (YT-YE) - YDT*(XT-XE)
      IF(EPSXT)                          23,  25,  25
   23      XT = XT + A
      GO TO  22
   25 IF(ABS(EPSXT) - DELTA)             32,  32,  26
   26      XT = XT - EPSXT/(2.*XT*(XT-XE)*(YDT**2.))
           YDT = 1./(1. + XT**2.)
           YT = ATAN(XT)
        EPSXT = (YT-YE) - YDT*(XT-XE)
      GO TO  25
   32       B = YT - YDT*XT
      WRITE(3,830)   XT,    YT
      WRITE(3,831)  YDT,     B
           XE = XT + A
   42      YEE = ATAN(XE)
        EPSXE = ((YDT*XE + B) - YEE) - EMAX
      IF(EPSXE)                          43,  44,  44
   43      XE = XE + A
      GO TO  42
   44      YDE = 1./(1. + XE**2.)
      IF(ABS(EPSXE) - DELTA)             52,  52,  46
   46      XE = XE - EPSXE/(YDT-YDE)
           YEE = ATAN(XE)
        EPSXE = ((YDT*XE + B) - YEE) - EMAX
      GO TO  44
   52      YE = YDT*XE + B
           YEI = .5*(YE+YEE)
      WRITE(3,820)   XE,    YE
      WRITE(3,821)  YEE,   YEI
           DEL = YE - 1.570 796 326 794 9
      IF(DEL)                            91,  91,  63
   91 CONTINUE
      WRITE (3,855) XE, YE
      IF(ABS(DEL) - DELTA)              192, 192,  61
   61 IF(DEL)                            62,  63,  63
   62    EMAX = EMAX + DELT
      GO TO   1
   63    EMAX = EMAX - DELT
         DELT = .1*DELT
         EMAX = EMAX + DELT
      GO TO   1
  192 WRITE (3,861) EMAX
      END
```

```
  1 MON$$     EXEQ AUTOCODER,,,NOMAC
  2           HEADR***FLOATING POINT TYPE SUBROUTINE***
  3           TITLEFLTPT
  4           SBR  X13
  5           BCE  END,/MCI/
  6           MLCA 4+X13,*+6
  7           MLC  0,WORK
 71           MLC
  8           BZN  *-11,/CTB/
  9           BXPA /CNC/
 10           DCW  MESSAGE
 12 END       B    5+X13
 13  MESSAGE  DCW  =10
 14 WORK      DCW  =2
 16           DCW  -)-
 17           END
    MON$$     EXEQ LINKLOAD
              CALL ASYMPNTRIP
    MON$$     EXEQ ASYMPNTRIP,MJB
    MON$$     JOB  ACT$$   BEAT A. GIMMEL     EE       0154U40408
```

The flow chart of this program is given in Fig. 29. The name of the above AUTOCODER subprogram is FLTPT. Each time the statement FLTPT(EMAX) is encountered in the main program, the actual value of EMAX appears on the console typewriter of the IBM 1410 Data Processing System.

APPENDIX D

FORTRAN Program of Asymptotic N-Trip with Newton Process in E-Start, T-, and E-Algorithm and with Oscillatory Dit-by-dit Method in Pi Half Algorithm

```
   MON$$     JOB  ASYMPTOTIC  N TRIP (NP) OSCILL APPR
   MON$$     COMT 10 MINUTES, 10 PAGES
   MON$$     ASGN MJB,12
   MON$$     ASGN MGO,16
   MON$$     MODE GO,TEST
   MON$$     EXEQ FORTRAN,,,10,03,,,ASYMPNTRIP
800 FORMAT(I3)
801 FORMAT(1H1,30H ASYMPTOTIC  N TRIP(NP)  N =  ,I3)
803 FORMAT(1HL,10H   EMAX = ,E20.14,10H   DELT = ,E20.14)
805 FORMAT(1HT,10H  DELTA = ,E20.14)
802 FORMAT(10X          ,E20.14,10X          ,E20.14)
804 FORMAT(10X          ,E20.14)
820 FORMAT(1HK,10H     XE = ,E20.14,10H     YE = ,E20.14)
821 FORMAT(1HT,10H    YEE = ,E20.14,10H    YEI = ,E20.14)
830 FORMAT(1H ,10H     XT = ,E20.14,10H     YT = ,E20.14)
831 FORMAT(1HS,10H    YDT = ,E20.14,10H      B = ,E20.14)
855 FORMAT(1HT,10H     XEN = ,E20.14,10H    YEN = ,E20.14)
860 FORMAT(1HL,10H   EMAX = ,E20.14)
861 FORMAT(1HL,10H EMAXEN = ,E20.14)
    READ  (1,800) N
    WRITE (3,801) N
    READ  (1,802) EMAX, DELT
    WRITE (3,803) EMAX, DELT
    READ  (1,804) DELTA
    WRITE (3,805) DELTA
          A = 1.0
  1 WRITE (3,860) EMAX
         XT = 0.0
         YT = 0.0
        YDT = 1.0
          B = 0.0
         XE = A
  2      YEE = ATAN(XE)
      EPSXE = (XE-YEE) - EMAX
    IF(EPSXE)                          3,   4,   4
  3       XE = XE + A
    GO TO   2
  4      YDF = 1./(1. + XE**2.)
    IF(ABS(EPSXE) - DELTA)            12,  12,   6
  6       XE = XE - EPSXE/(1.0-YDE)
         YEE = ATAN(XE)
      EPSXE = (XE-YEE) - EMAX
    GO TO   4
 12       YE = XE
         YEI = .5*(YE+YEE)
    WRITE(3,820)   XE,   YE
    WRITE(3,821)  YEE,  YEI
    DO  91 J=1,N
          XT = XE + A
 22       YDT = 1./(1. + XT**2.)
```

```
        YT = ATAN(XT)
      EPSXT = (YT-YE) - YDT*(XT-XE)
   IF(EPSXT)                          23,  25,  25
23      XT = XT + A
   GO TO  22
25 IF(ABS(EPSXT) - DELTA)             32,  32,  26
26      XT = XT - EPSXT/(2.*XT*(XT-XE)*(YDT**2.))
        YDT = 1./(1. + XT**2.)
        YT = ATAN(XT)
      EPSXT = (YT-YE) - YDT*(XT-XE)
   GO TO  25
32        B = YT - YDT*XT
   WRITE(3,830)   XT,   YT
   WRITE(3,831)  YDT,    B
        XE = XT + A
42      YEE = ATAN(XE)
     EPSXE = ((YDT*XE + B) - YEE) - EMAX
   IF(EPSXE)                          43,  44,  44
43      XE = XE + A
   GO TO  42
44      YDE = 1./(1. + XE**2.)
   IF(ABS(EPSXE) - DELTA)             52,  52,  46
46      XE = XE - EPSXE/(YDT-YDE)
        YEE = ATAN(XE)
     EPSXE = ((YDT*XE + B) - YEE) - EMAX
   GO TO  44
52      YE = YDT*XE + B
        YEI = .5*(YE+YEE)
   WRITE(3,820)   XE,   YE
   WRITE(3,821)  YEE,  YEI
        DEL = YE - 1.570 796 326 794 9
   IF(DEL)                            91,  91,  63
91 CONTINUE
   WRITE (3,855) XE, YE
   IF(ABS(DEL) - DELTA)              192, 192,  61
61 IF(DEL)                            62,  63,  63
62    EMAX = EMAX + DELT
   GO TO   1
63    DELT = 0.1 * DELT
      EMAX = EMAX - DELT
101 WRITE (3,860) EMAX
   CALLFLTPT(EMAX)
        XT = 0.0
        YT = 0.0
       YDT = 1.0
         B = 0.0
        XE = A
102     YEE = ATAN(XE)
```

```
      EPSXE = (XE-YEE) - EMAX
      IF(EPSXE)                          103, 104, 104
103       XE = XE + A
      GO TO 102
104       YDE = 1./(1.+XE**2.)
      IF(ABS(EPSXE) - DELTA)             112, 112, 106
106       XE = XE - EPSXE/(1.0-YDE)
          YEE = ATAN(XE)
        EPSXE = (XE-YEE) - EMAX
      GO TO 104
112       YE = XE
          YEI = .5*(YE+YEE)
      WRITE(3,820)   XE,   YE
      WRITE(3,821)  YEE,  YEI
      DO 191 J=1,N
          XT = XE + A
122       YDT = 1./(1. + XT**2.)
          YT = ATAN(XT)
        EPSXT = (YT-YE) - YDT*(XT-XE)
      IF(EPSXT)                          123, 125, 125
123       XT = XT + A
      GO TO 122
125 IF(ABS(EPSXT) - DELTA)               132, 132, 126
126       XT = XT - EPSXT/(2.*XT*(XT-XE)*(YDT**2.))
          YDT = 1./(1. + XT**2.)
          YT = ATAN(XT)
        EPSXT = (YT-YE) - YDT*(XT-XE)
      GO TO 125
132         B = YT - YDT*XT
      WRITE(3,830)   XT,   YT
      WRITE(3,831)  YDT,    B
          XE = XT + A
142       YEE = ATAN(XE)
        EPSXE = ((YDT*XE + B) - YEE) - EMAX
      IF(EPSXE)                          143, 143, 144
143       XE = XE + A
      GO TO 142
144       YDE = 1./(1. + XE**2.)
      IF(ABS(EPSXE) - DELTA)             152, 152, 146
146       XE = XE - EPSXE/(YDT-YDE)
          YEE = ATAN(XE)
        EPSXE = ((YDT*XE + B) - YEE) - EMAX
      GO TO 144
152       YE = YDT*XE + B
          YEI = .5*(YE+YEE)
      WRITE(3,820)   XE,   YE
      WRITE(3,821)  YEE,  YEI
         DELN = YE - 1.570 796 326 794 9
```

```
    IF(DELN)                              191, 191, 161
191 CONTINUE
    WRITE (3,855) XE, YE
161 IF(DELN * DEL)                        162, 162, 163
162     DELT = .1 * DELT
163     DEL = DELN
    IF(ABS(DEL) - DELTA)                  192, 192, 164
164 IF(DEL)                               165, 166, 166
165     EMAX = EMAX + DELT
    GO TO 101
166     EMAX = EMAX - DELT
    GO TO 101
192 WRITE (3,861) EMAX
    END
 1 MON$$      EXEQ AUTOCODER,,,NOMAC
 2            HEADR***FLOATING POINT TYPE SUBROUTINE***
 3            TITLEFLTPT
 4            SBR  X13
 5            BCE  END,/MCI/
 6            MLCA 4+X13,*+6
 7            MLC  0,WORK
 71           MLC
 8            BZN  *-11,/CTB/
 9            BXPA /CNC/
10            DCW  MESSAGE
12 END        B    5+X13
13  MESSAGE   DCW  =10
14 WORK       DCW  =2
16            DCW  -)-
17            END
   MON$$      EXEQ LINKLOAD
              CALL ASYMPNTRIP
   MON$$      EXEQ ASYMPNTRIP,MJB
   MON$$      JOB  ACT$$   BEAT A. GIMMEL     EE       0154U40408
```

APPENDIX E

Simulation of an Arctangent SPC on IBM 1620

The computation program that is going to be described is written in SPS - language for electronic digital computer IBM 1620. The SPS - processor program that translates SPS - language into the machine language of IBM 1620 is an assembler program such that arithmetic, branch, and input-output SPS statements correspond in a unique manner to IBM 1620's machine instructions. This one-to-one correspondence and SPS's mnemonic features predestinate this symbolic program language to be used for simulating an arctangent SPC on IBM 1620.

The subsequent program consists of two main programs A and B. Program A computes the arctangent (ATAN) function by means of an upper polygonal approximation, whereas program B computes the ATAN function by using Chebyshev power series. The upper polygonal approximation of program A employs 16-line segments whose upper boundaries XE, slopes YDT, and y-intercepts B were calculated in the Asymptotic 14-Trip (cf Table 12). In this program the triple (XE, YDT, B) is denoted by (XHIGH, MSLOPE, BINTCP). The Chebyshev power series approximation of program B calculates arctan(x) for the interval (0,1) in a straightforward manner, whereas for $(1, \infty)$ arctan(1/x) is computed first, then the transformation $\arctan(x) = \pi/2 - \arctan(1/x)$ is used. An exhaustive description of the IBM 1620 machine instructions and the SPS program language is given by (8).

```
01010       DORG402
01020AEFRTNDS  28
01030E14A/NSF  AEFRTN-25
01040       CM  AEFRTN-24,03,10,CHECK FOR DECIMAL
01050       BNZ A,,,ERROR BRANCH
01060       CM  AEFRTN-26,20,10,MANTISSA SIGN
01070       BNZ B
01080       SF  AEFRTN-8
01090B      SF  AEFRTN-22
01100       TD  NEFRTN-9,AEFRTN-22
01110       TD  NEFRTN-8,AEFRTN-20
01120       TD  NEFRTN-7,AEFRTN-18
01130       TD  NEFRTN-6,AEFRTN-16
01140       TD  NEFRTN-5,AEFRTN-14
01150       TD  NEFRTN-4,AEFRTN-12
01160       TD  NEFRTN-3,AEFRTN-10
01170       TD  NEFRTN-2,AEFRTN-8
01180       SF  AEFRTN-7
01190       CM  AEFRTN-6,45,10,CHECK FOR E
01200       BNZ C,,,NO E BRANCH
02010       SF  AEFRTN-5
02020       CM  AEFRTN-4,20,10,CHECK OF CHARACTERISTIC SIGN
02030       BNZ D
02040       SF  AEFRTN
02050D      SF  AEFRTN-2
02060       TD  NEFRTN-1,AEFRTN-2
02070       TD  NEFRTN,AEFRTN
02080       BB
02090       DORG*-9
02100A      RCTY
02110       WATYERROR
02120       H
02130ADRESAB   0
02140       DORG*-3
02150C      TFM NEFRTN,00,10
02160       BB
02170       DORG*-9
02180NEFRTNDS  10
02190E14N/ATF  AEFFTN,AFIELD
02200       BNF E,NEFRTN-2,,MANTISSA SIGN
03010       TDM AEFRTN-27,2,11
03020       CF  NEFRTN-2
03030E      CF  NEFRTN-9
03040       TD  AEFRTN-22,NEFRTN-9
03050       TD  AEFRTN-20,NEFRTN-8
03060       TD  AEFRTN-18,NEFRTN-7
03070       TD  AEFRTN-16,NEFRTN-6
03080       TD  AEFRTN-14,NEFRTN-5
03090       TD  AEFRTN-12,NEFRTN-4
03100       TD  AEFRTN-10,NEFRTN-3
```

```
03110         TD   AEFRTN-8,NEFRTN-2
03120         BNF  F,NEFRTN
03130         TDM  AEFRTN-5,2
03140         CF   NEFRTN
03150F        CF   NEFRTN-1
03160         TD   AEFRTN-2,NEFRTN-1
03170         TD   AEFRTN,NEFRTN
03180         BB
03190         DORG*-9
03200ERROR DAC  14,ERROR IN DATA-
04010AFIELDDC   28,0003707070707070707045107070
04020STARTSRCTY
04030         WATYCMNTA
04040G        H
04050         BC1  PROGMB
04060PROGMARCTY
04070         WATYCMNTB
04080         TFM  PROGMA+6,16,10
04090         TFM  AA+6,XHIGH,7
04100         TFM  AB+6,MSLOPE,7
04110         TFM  AC+6,BINTCP,7
04120         TFM  ADRESA+6,AD,7
04130AD       RACDINPUT
04140         SF   INPUT-1
04150         SF   INPUT+27
04160         SF   INPUT+55
04170         BT   E14A/N,INPUT+26
04180AA       TFL  0,NEFRTN
04190         BT   E14A/N,INPUT+54
04200AB       TFL  0,NEFRTN
05010         BT   E14A/N,INPUT+82
05020AC       TFL  0,NEFRTN
05030         AM   AA+6,10,10
05040         AM   AB+6,10,10
05050         AM   AC+6,10,10
05060         SM   PROGMA+6,01,10
05070         BH   AD
05080         RCTY
05090         WATYCMNTCA
05100H        RCTY
05110         RNTYSTARTS+2,,,TIMES LOOP NUMBER
05120         BC4  H
05130         SF   STARTS+2
05140         RCTY
05150         WATYCMNTC
05160         TFM  ADRESA+6,SWA,7
05170J        RACDINPUT
05180         SF   INPUT-1
05190         BT   E14A/N,INPUT+26
05200         TFL  XNUMBR,NEFRTN
```

```
06010       TDM H+7,0
06020       BNF AF,XNUMBR-2
06030       SF  H+7
06040       CF  XNUMBR-2
06050AF     TF  H+6,STARTS+6
06051       H
06060I      TFL G+11,XNUMBR
06070       FSUBG+11,XHIGH+150
06080       BNH XA
06090       BTFLE14M/A,XNUMBR
06100       TF  INPUT+26,AEFRTN
06110       TR  INPUT+27,CMNTD-1
06120       WACDINPUT
06130       RCTY
06140       WATYINPUT
06150       B   SWA
06160       DORG*-3
06170XA     TFL G+11,XNUMBR
06180       FSUBG+11,XHIGH+  70
06190       BH  XI
06200       TFL G+11,XNUMBR
07010       FSUBG+11,XHIGH+  30
07020       BH  XE
07030       TFL G+11,XNUMBR
07040       FSUBG+11,XHIGH+  10
07050       BH  XC
07060       TFL G+11,XNUMBR
07070       FSUBG+11,XHIGH
07080       BH  XB
07090       TFM AH+11,MSLOPE
07100       TFM AI+11,BINTCP
07110       B   AK
07120       DORG*-3
07130XB     TFM AH+11,MSLOPE+10
07140       TFM AI+11,BINTCP+10
07150       B   AK
07160       DORG*-3
07170XC     TFL G+11,XNUMBR
07180       FSUBG+11,XHIGH+ 20
07190       BH  XD
07200       TFM AH+11,MSLOPE+20
08010       TFM AI+11,BINTCP+20
08020       B   AK
08030       DORG*-3
08040XD     TFM AH+11,MSLOPE+30
08050       TFM AI+11,BINTCP+30
08060       B   AK
08070       DORG*-3
08080XE     TFL G+11,XNUMBR
08090       FSUBG+11,XHIGH+ 50
```

```
0A100       BH  XG
0A110       TFL G+11,XNUMBR
0A120       FSUBG+11,XHIGH+ 40
0A130       BH  XF
0A140       TFM AH+11,MSLOPE+40
0A150       TFM AI+11,BINTCP+40
0A160       B   AK
0A170       DORG*-3
0A180XF     TFM AH+11,MSLOPE+50
0A190       TFM AI+11,BINTCP+50
0A200       B   AK
00010       DORG*-3
00020XG     TFL G+11,XNUMBR
00030       FSUBG+11,XHIGH+ 60
00040       BH  XH
00050       TFM AH+11,MSLOPE+60
00060       TFM AI+11,BINTCP+60
00070       B   AK
00080       DORG*-3
00090XH     TFM AH+11,MSLOPE+70
00100       TFM AI+11,BINTCP+70
00110       B   AK
00120       DORG*-3
00130XI     TFL G+11,XNUMBR
00140       FSUBG+11,XHIGH+ 110
00150       BH  XM
00160       TFL G+11,XNUMBR
00170       FSUBG+11,XHIGH+ 90
00180       BH  XK
00190       TFL G+11,XNUMBR
00200       FSUBG+11,XHIGH+ 80
10010       BH  XJ
10020       TFM AH+11,MSLOPE+80
10030       TFM AI+11,BINTCP+80
10040       B   AK
10050       DORG*-3
10060XJ     TFM AH+11,MSLOPE+90
10070       TFM AI+11,BINTCP+90
10080       B   AK
10090       DORG*-3
10100XK     TFL G+11,XNUMBR
10110       FSUBG+11,XHIGH+ 100
10120       BH  XL
10130       TFM AH+11,MSLOPE+100
10140       TFM AI+11,BINTCP+100
10150       B   AK
10160       DORG*-3
10170XL     TFM AH+11,MSLOPE+110
10180       TFM AI+11,BINTCP+110
10190       B   AK
```

```
10200       DORG*-3
11010XM     TFL G+11,XNUMBR
11020       FSUBG+11,XHIGH+ 130
11030       BH  XO
11040       TFL G+11,XNUMBR
11050       FSUBG+11,XHIGH+ 120
11060       BH  XN
11070       TFM AH+11,MSLOPE+120
11080       TFM AI+11,BINTCP+120
11090       B   AK
11100       DORG*-3
11110XN     TFM AH+11,MSLOPE+130
11120       TFM AI+11,BINTCP+130
11130       B   AK
11140       DORG*-3
11150XO     TFL G+11,XNUMBR
11160       FSUBG+11,XHIGH+ 140
11170       BH  XP
11180       TFM AH+11,MSLOPE+140
11190       TFM AI+11,BINTCP+140
11200       B   AK
12010       DORG*-3
12020XP     TFM AH+11,MSLOPE+150
12030       TFM AI+11,BINTCP+150
12040       B   AK
12050       DORG*-3
12060AK     TFL YNUMBR,XNUMBR
12070AH     FMULYNUMBR,0
12080AI     FADDYNUMBR,0
12090       SM  H+6,01,10
12100       BH  I
12101       H
12110       BNF AL,H+7
12120       SF  XNUMBR-2
12130       SF  YNUMBR-2
12140AL     TR  INPUT-1,ZERO-1
12150       TR  INPUT+79,ZERO
12160       BTFLE14N/A,XNUMBR
12170       TF  INPUT+26,AEFRTN
12180       BTFLE14N/A,YNUMBR
12190       TF  INPUT+56,AEFRTN
12200       WACDINPUT
13010SWA    BNC1J
13020PROGMBRCTY
13030       WATYCMNTCA
13040TCERR RCTY
13050       RNTYSTARTS+2,,,TIMES LOOP NUMBER
13060       BC4 TCERR
13070       SF  STARTS+2
13080       RCTY
```

```
13090       WATYCMNTC
13100       TFM ADRESA+6,SWB,7
13110AM     RACDINPUT
13120       SF  INPUT-1
13130       BT  E14A/N,INPUT+26
13140       TFL XNUMBR,NEFRTN
13150       TF  H+6,STARTS+6
13151       H
13160K      FATNYNUMBR,XNUMBR
13170       SM  H+6,01,10
13180       BH  K
13181       H
13190       TR  INPUT-1,ZERO-1
13200       TR  INPUT+79,ZERO
14010       BTFLE14N/A,XNUMBR
14020       TF  INPUT+26,AEFRTN
14030       BTFLE14N/A,YNUMBR
14040       TF  INPUT+56,AEFRTN
14050       WACDINPUT
14060SWB    BC1 AM
14070       B   PROGMA
14080       DORG*-3
14090CMNTCADAC 33,ENTER NUMBER OF FUNCTION PASSES.-
14100INPUT  DAS 80
14110       DS  2
14120CMNTA  DAC 48,SW1 ON FOR ARCTAN,OFF FOR FUNCTION APROXIMATION-
14130CMNTB  DAC 33,ENTER FUNCTION APROXIMATION DATA-
14140CMNTC  DAC 8,ENTER X-
14150XHIGH  DSB 10,16
14160MSLOPEDSB 10,16
14170BINTCPDSB 10,16
14180XNUMBRDS  10
14190YNUMBRDS  10
14200CMNTD  DAC 34, =X IS NOT IN RANGE OF FUNCTION. -
15010ZERO   DAC 41,                                    -
15020NOISE  DAC 1,0
15030       DENDSTARTS
```

PRECOMPUTED FUNCTION CONSTANTS

+.25127153E+00+.10000000E+01+.00000000E-99
+.51703588E+00+.86922173E+00+.32860913E-01
+.76930414E+00+.70781189E+00+.11631560E+00
+.10425140E+01+.55074173E+00+.23715032E+00
+.13603676E+01+.41117293E+00+.38265275E+00
+.17506624E+01+.29442000E+00+.54147968E+00
+.22536126E+01+.20151870E+00+.70411849E+00
+.29338931E+01+.13102586E+00+.86298204E+00
+.39041213E+01+.80131869E-01+.10122996E+01
+.53771745E+01+.45400912E-01+.11478935E+01
+.77979787E+01+.23268503E-01+.12669033E+01
+.12231175E+02+.10368454E-01+.13674976E+01
+.21783537E+02+.37375932E-02+.14486008E+01
+.49045288E+02+.93491810E-03+.15096530E+01
+.19620f30E+03+.10389779E-03+.15504106E+01
+.99999999E+99+.000 00000E+00+.15707962E+01

The sequence of SPS statements from line 01010 to 02160 represents the input subroutine E14A/N that performs a part of the input translations by converting alphamerical floating point input data of the form +.50732777E-02 into numerical floating point data of the form $\overline{5}0732777\overline{02}$. The sequence of statements from line 02170 to 03180 performs an output translation opposite to the above described input translation. This output subroutine is labeled E14N/A. When the programs A and B are to be executed, execution starts by means of a branch instruction on line 04020. If program switch 1 is in on-position, program A is executed according to the statement of line 04050. It is supposed that program A (PROGMA) is executed first. The 5-digit data field PROGMA+6 stores initially the number of triples, (XHIGH, MSLOPE, BINTCP), that will be read-in as nonerasable information into core storage; it also serves as a counter. Each time a triple (XIGH, MSLOPE, BINTCP) has been read-in this count is decreased by 1. Statement 04080 resets this field to the initial value of 16 as the upper polygonal approximation consists of 16 line segments. The sequence of statements from line 04130 to line 05070 represents the loop for the triples (XHIGH, MSLOPE, BINTCP). At every pass each member of the triple undergoes the input translation E14A/N and thereafter the counter PROGMA+6 is decremented by 1. The triple (XHIGH, MSLOPE, BINTCP) is punched manually on an IBM card with XHIGH starting in column 1 and the format 3E14.8 (cf. end of program). Statement 05090 causes the computer to print out the message "ENTER NUMBER OF FUNCTION PASSES". This number also called TIMES

LOOP NUMBER is entered as a 5-digit number on the console typewriter. It is stored in core storage in field STARTS+2. This number determines how many times ATAN(X) has to be calculated by using upper polygonal approximation. The purpose of repetitive calculation of arctan(x), where x remains unchanged, is to facilitate computation time measurement. Statement 05150 causes the console typewriter to print out the message ENTER X-. Statement 05170 causes the card reader to read argument x. Argument x is punched manually on IBM card with format E14.8, e.g., +.11111111E+01; and the mantissa sign is entered in column 1. Argument x undergoes likewise the input translation E14A/N. Statement 06050 transfers the TIMES LOOP NUMBER from field STARTS+6 to count-down area H+6. The computer halts at statement 06051. Here the central part of program A starts with the decision tree. This tree has four levels and leads to 16 pairs (MSLOPE, BINTCP). It starts at statement 06170 and ends at statement 12040. The arithmetic unit of this arctangent SPC is simulated from 12060 to 12080. After statement 12080 is executed calculation of arctan(x) is completed and the times loop counter is decreased by 1. The computer branches back to statement 06060 to repeat the same calculation. This looping will continue until the times loop counter attains zero and the computer is caused to halt at instruction 12101. By choosing the TIMES LOOP NUMBER adequately high, it is possible to measure computation time between the two HALT instructions of statement 04040 and 12101 with a wrist watch. To cause the computer to execute program B Program Switch 1 is set in on-position. After

depressing the Start key the output translation E14N/A for argument x and its calculated value is performed. Both numbers are punched with format 2E14.6. Then the computer branches automatically to program B. The formats for this program are identical with those of program A. The central part of this program is located between the HALT statements 13151 and 13181. At statement 13160 the computer branches to the arctangent library subroutine FATN. This subroutine uses a Chebyshev power series approximation. Computation time is measured in the same way as for program A and the same holds for the output translation.

Time measurement yielded:

Polygonal approximation	28ms
Power series approximation	340ms

In this program polygonal approximation is twelve times faster than power series approximation. However, this ratio would be reduced if a polygonal approximation is employed that yields the same accuracy of 10^{-8} as the power series approximation.

Higher accuracy for polygonal approximation is obtained by employing a larger number of line segments. Such ATAN SPC's use more decision levels. The number of decision levels grows logarithmically as the number of line segments increases. Therefore decision computation time increases slowly, whereas arithmetic computation time remains constant. For power series approximation computation time increases exponentially as accuracy improves. Polygonal approximation, therefore, is a faster com-

putation method than power series approximation. On the other hand, polygonal approximation uses more storage space than power series approximation. This is due to the larger number of precomputed constants to be stored and the greater number of instructions involved.

SPECIAL PURPOSE DIGITAL COMPUTER WITH A NONERASABLE MEMORY UNIT

by

BEAT ALBERT GIMMEL

Dipl. El. Ing., Eidgenoessische Technische Hochschule, Zurich, Switzerland, 1964

ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1966

This thesis discusses a special purpose digital computer for the arctangent function. The arctangent function is approximated by upper polygonal approximation. Different kinds of polygonal approximation are discussed in detail and their features compared. The "Upper Polygonal Approximation with Uniform Chebyshev Norm" turns out the most favorable approximation. Problems induced by different approximations are thoroughly treated. Numerical results for the first ten approximations are given.

Finally, the basic ideas for the realization of an arctangent special purpose computer are summarized. They should convey an understanding of its simulation on a general purpose computer. Numerical calculation and simulation was done on an IBM 1620 digital computer.