CLEARINGHOUSE FOR FEDERAL SCIENTIFIC AND TECHNICAL INFORMATION CFSTI
DOCUMENT MANAGEMENT BRANCH 410.11

LIMITATIONS IN REPRODUCTION QUALITY

ACCESSION # AD 604 202

☒ 1. WE REGRET THAT LEGIBILITY OF THIS DOCUMENT IS IN PART UNSATISFACTORY. REPRODUCTION HAS BEEN MADE FROM BEST AVAILABLE COPY.

☐ 2. A PORTION OF THE ORIGINAL DOCUMENT CONTAINS FINE DETAIL WHICH MAY MAKE READING OF PHOTOCOPY DIFFICULT.

☐ 3. THE ORIGINAL DOCUMENT CONTAINS COLOR, BUT DISTRIBUTION COPIES ARE AVAILABLE IN BLACK-AND-WHITE REPRODUCTION ONLY.

☐ 4. THE INITIAL DISTRIBUTION COPIES CONTAIN COLOR WHICH WILL BE SHOWN IN BLACK-AND-WHITE WHEN IT IS NECESSARY TO REPRINT.

☐ 5. LIMITED SUPPLY ON HAND: WHEN EXHAUSTED, DOCUMENT WILL BE AVAILABLE IN MICROFICHE ONLY.

☐ 6. LIMITED SUPPLY ON HAND: WHEN EXHAUSTED DOCUMENT WILL NOT BE AVAILABLE.

☐ 7. DOCUMENT IS AVAILABLE IN MICROFICHE ONLY.

☐ 8. DOCUMENT AVAILABLE ON LOAN FROM CFSTI (TT DOCUMENTS ONLY).

☐ 9.

PROCESSOR: PM

TSL-107-10/64

PERMANENT SET-UPS FOR IBM CALCULATORS

William Orchard-Hays

P-374

17 February 1953

The RAND Corporation
SANTA MONICA • CALIFORNIA

PERMANENT SET-UPS FOR IBM CALCULATORS

Introduction

The purpose of this paper is to discuss the efficiency of different methods of computation wherein the elementary functions and others must be evaluated on the IBM Electronic Calculating Punch, models 604 and 605, and the IBM Card-Programmed Electronic Calculator, models I and II. These machines will be referred to hereafter simply as the 604, 605, CPC-I, and CPC-II, respectively. Some of the following is applicable to computation on any electronic computer, but those computers with a large memory capacity and high-speed internal programming largely remove the very restrictions which make efficient evaluation of functions a problem on the 604 or CPC.

The fact that, given enough time, patience, and cards, practically any computation can be performed on the 604 or CPC is trivial. What is of interest is how any job from a fairly large class can be most efficiently done in the day-by-day work of an IBM computing establishment. Assuming that a problem is already stated explicitly - i.e., that the computing group is required to evaluate certain items or obtain numerical solutions to certain equations, or both, for a given number of parameter combinations - it would still be necessary to make several decisions and to perform considerable preparatory work before putting the job on a machine. Hence it is apparent that standardized methods can be as much of an aid to computation as the actual machines, provided that such methods are general enough to embrace a wide variety of jobs and are not cumbersome. Also it is helpful to have set-ups available which allow decisions to be made on the basis of the problem presented without excessive consideration of how such and such a step of the problem can actually be calculated efficiently.

Floating Decimal (FD) Set-Ups for the CPC

For the CPC, general-purpose set-ups (or special-purpose set-ups for a long continuing problem) are practically a necessity if the time required for coding, wiring, and testing is to be kept within reasonable limits. The CPC may be looked upon, not as a computer, but as a neat package of parts which the set-up man assembles, by means of wires, into a computing system. The skill which the coders then acquire in using this system becomes one of the assets of the organization. If the system is complicated to use, fewer coders will use it or become proficient with it. If the system is elaborate enough to be truly general and still simple to use, it is almost bound to be complicated plug-board-wise; but if the speed of operation is thereby reduced, the advantage of simplicity in preparing a job is largely overcome by the time consumed in running it. This situation is aggravated by floating decimal (FD) operation, which many people prefer for the CPC and which certainly has advantages if it is properly used. The writer has spent considerable time in improving and developing general-purpose FD set-ups, first for the CPC-I and then for the CPC-II. Many of the techniques used are also applicable to fixed-point parallel work on the 604.

A general-purpose FL set-up for the CPC-I can include square root along with the four basic arithmetic and necessary manipulative operations. Single-card evaluation of other functions is virtually precluded for 8x8 arithmetic by the limitations of the machine. To fill this need, standardized subroutines are often provided, many using rational approximations to the functions. Mr. Cecil Hastings, Jr. has developed many valuable approximations of this kind.(1) Such approximations are equally valuable in 604 work, where they are incorporated

in the procedure rather than existing as sub-routines. An inherent advantage of these sub-routines is that the length of the computation can be readily fitted to the accuracy desired by inspecting the specifications of the approximation, whereas the number of terms required in a series, for example, will vary widely for a moderate interval for the argument.

The CPC-II makes it possible to consider FD set-ups of a much more elaborate kind. Its higher cost is worth while only if greater efficiency can be obtained. This is not as easy to do for a general-purpose set-up as first appears, since the internal storage of the 605 is no greater than that of the 604 and there is actually less "compute time" available per card cycle on the 605. At least three ways to increase the output of useful work are as follows:

(1) to perform more operations per card in the same time,

(2) to include functions as single-card operations instead of using sub-routines,* and

(3) to carry more digits for each number.

The last item mentioned is a subject of its own and will not be of concern here except in what immediately follows. The size of the storage units within the 604 or 605 limits the programmer to 8x8 FD arithmetic if any speed is to be maintained and if very many operations are to be built into the calculator board. Furthermore, since the main storage for the CPC is in the 941's, available only via 10-digit channels, and since it is necessary to carry an exponent of 10 for the scale factor (usually, although not very accurately, called a "power") with each number, 8x8 arithmetic with 2-digit powers seems to fit the CPC better than

---

*Increased speed is of course sought, but even without it, the elimination of sub-routines is an advantage by saving storage units in the tab or 941, coding and key-punching or else reproducing time, and card handling.

any other combination for general-purpose FD set-ups. To carry more digits, even 10 or 12, some severe restrictions must be placed on addresses of the operands and allowable operations. Double precision boards (18 or more digits per number) are of course very useful for some applications, and the CPC-II makes it possible to have much better ones than the CPC-I allows, but they prohibit single-card evaluation of functions, excepting perhaps square root.

A Four-Address FD Set-Up

On the other hand, the only possible advantage of FD operands with fewer than 8 digits is to reduce computing time, since the channels and the 941 units cannot be adapted to efficient handling of "½-words." But the 605 will go through 60 program steps every card cycle anyway, so that if 2 arbitrary FD operations* on 8-digit operands can be performed in these 60 steps** (without an excessive number of multiplications or divisions) there is no reason to reduce the word length. It is apparent that addition is the most difficult arithmetic operation with FD numbers since it involves actually four operations:

(1) a right shift to align the decimal points of the addends,

(2) the addition,

(3) the checking of the sum for a carry or cancellation of leading digits, with appropriate right or left shifts, and

(4) the correction of the power of the sum.

Although these different parts of the addition routine may be used as adjuncts to other operations, the complete routine, nevertheless, consumes over

---

*More than 2 such operations unduly complicate not only wiring but also coding.

**A single "program repeat" will always cause a delay cycle, even without using "repeat delay"; at least this is true on the three CPC-II's which the writer has used.

half a sweep - 43 program steps being the best the writer has been able to do to allow for all eventualities, although 2 or 3 of these steps may have been saved in combination operations. Thus if 2 arbitrary operations are to be performed in one sweep, two banks (chains) of programs on the 605 must be used just for the 3 basic arithmetic operations (subtraction can be accomplished by changing the sign of the subtrahend on the tab board), one bank performing addition first, followed by multiplication or division, and the other bank doing the addition last. The writer has wired such a set-up, which takes three 8-digit-plus-2-digit-power inputs (any two such inputs from the 941's) and performs any possible combination of 2 of the 4 basic arithmetic operations on each card cycle at 150 cards per minute, except for 2 additions (or subtractions). The latter combination is allowed, but a delay is required. (Occasionally, other operations require a delay, when there are many 8's or 9's in a multiplier or quotient.) This set-up is proving to be very useful, since it also allows computation of square root, 10^x, $\log_{10} x$, and cos x as single-card operations. All of these operations are quite fast except log x, which is sub-programmed (Special Programming" on the tabulator). The only interval restriction on arguments is for cos x, for which only $0 \leq x \leq \pi$ is allowed. Series are used for 10^x and cos x and an approximation(2) is used for log x. Since most computations consist of more basic arithmetic operations than anything else, and hence the faster they can be done the better, this set of functions and operations provides a good compromise for increasing the useful output of a CPC-II by the first two means stated previously.

An FD Set-Up With Several Functions

For jobs requiring frequent use of functions, one may well be satisfied with single arithmetic operations if a greater variety of functions can be evaluated at electronic speed as single-card operations. The RAND Computing

Group has one such set-up (FD) for the CPC-II on which the following single-card operations are allowed, all of which are evaluated completely within the 605:

1.	A + B	1.'	-A-B
2.	A - B	2.'	B - A
3.	AB	3.'	-AB
4.	A/B	4.'	-A/B
5.	sin B, $-\pi \leq x \leq \pi$	5.'	sinh B, $-2.9 \leq x \leq 2.9$, approx.
6.	cos B, $0 \leq x \leq \pi$	6.'	cosh B, $0 \leq x \leq 2.9$, approx.
7.	10^x, $-51 \leq x < 49$	8.	$\log_{10} x$, $10^{-51} \leq x < 10^{49}$
9.	$\sqrt{x}$, $10^{-51} \leq x < 10^{49}$	10.	arcsin x, $-1 \leq x \leq 1$ ($\|x\| > 1$ treated as $\|x\| = 1$)
11.	AB + P	12.	AP + B

A and B are arbitrary operands and P means the previous result, although as far as the 605 is concerned it could be arbitrary. Series are used for all the functions except $\sqrt{x}$, for which the usual Newton's method is used, 2 iterations being performed on one sweep. Speed and accuracy are both fairly good for the functions. Thirty arguments between 0.1 and 1.0 for arcsin x were run in one minute with a maximum error of 3×10^{-7}. An interesting sidenote is that it took some hunting to find the proper argument which would give this maximum error. It depends on the relative size of the rounding error in taking x^2 to 8 decimals as compared to the size of $\sqrt{1-x^2}$, since for arguments greater than 0.7, the relationship arcsin $x = \pi/2 - \arcsin \sqrt{1-x^2}$ is used. (Only non-negative arguments are used in the series, the sign being handled separately.) The argument 0.99993 gave the maximum error.

FD Techniques in Fixed-Decimal Computation

All the above functions except log x and $\sqrt{x}$ require that the argument be put in essentially fixed-decimal form within the 605; thus they can be evaluated just as well on a 604 for parallel work or on a CPC-I for fixed-point arithmetic. However, FD notation, at least within the 605, is essential for log x and a decided advantage for $\sqrt{x}$. To evaluate them efficiently on a 604, a $\frac{1}{2}$-time emitter is needed on the 521 so that the argument can be left-shifted on read-in until there is a leading non-zero digit for the log operation or a leading pair of digits, not both zero, for the square root operation, the shifted argument then being considered as a decimal number.

The advantage of left-shifting the argument before applying Newton's method to square root is due to the fact that if the first guess is much larger than the true root, many iterations are required before the first (non-zero) significant figure is obtained. The most convenient starting value for the whole interval $.01 \leq x < 1.0$ (which allows for the fact that x can only be shifted an even number of places) is $y_0 = \frac{1}{2}x + \frac{1}{2}$. In fact, for $x \geq .00090$, $\frac{1}{2}x + \frac{1}{2} = \sqrt{x}$ to 8 places, a fact useful in handling the troublesome special case of x = .99999999, which causes an overflow on $y_1 = \frac{1}{2}(y_0 + x/y_0)$. In all other cases, y_0 will be larger than either x or $\sqrt{x}$, and convergence will proceed with no complications. Of course, the root thus found must be right-shifted back half the number of places the argument was left-shifted. In FD set-ups the power merely has to be corrected, an operation easily performed by dividing the original power plus 50 by two and adding the remainder to the quotient.* This is done first and if the remainder is non-zero (i.e., 1). then the argument itself is right-shifted one place. The root found will always be

---

*This assumes unity is represented by 10000000 51.

in FP form if the argument was originally and will be correct in any case.

Series for the Elementary Functions

It is convenient to divide the power series for the elementary functions, for purposes of machine computation, into three classes. The expansion $e^x = \sum_0^\infty x^n/n!$ converges quickly to a specified accuracy because of the n! in the denominator of the term t_n. Thus the ratio $r(n) = \left| t_{n+1} / t_n \right|$ is $\left| x \right| / n+1$, and as soon as $n \geq k \left| x \right|$, $r(n) < 1/k$ and the succeeding terms quickly decrease in magnitude if, say, $k \geq 2$. For small intervals, say $\left| x \right| < \log_e 10 < 2.5$, $n = 5 > 2\left| x \right|$. Also $\left| t_5 \right| < (2.5)^5/5! < 1$ and $\left| t_{5+m} \right| < 2^{-m}$. Any series of the form $\sum c_n x^n$, where c_n is of the order of 1/n!, will be called, arbitrarily, exponential-type series, for want of a better name. They converge uniformly over any interval. They include e^x, sin x, cos x, sinh x, and cosh x.

If c_n in the above series is of the order of 1/n, the series will be called an harmonic type. This type includes the series for log x, $\tan^{-1}x$, and $\tanh^{-1}x$.

Still a third type of series is met with in the elementary functions but, for practical purposes, may be considered along with the harmonic type. This type consists of the series whose n^{th} term is of the form $\frac{1\cdot3\cdot5\cdots(2n-1)}{2\cdot4\cdot6\cdots(2n)} \cdot \frac{x^{2n+1}}{2n+1}$. The first fraction may be written $\prod_{k=1}^{n} (1 - \frac{1}{2k})$, which is obviously less than 1 for all n. In fact, since $\sum_1^\infty \frac{1}{2k}$ diverges, the infinite product $\prod_1^\infty (1 - \frac{1}{2k})$ diverges also (in the stricter sense) and hence diverges to zero, decreasing monotonically as n increases. Therefore these product coefficients cannot hurt the convergence of the series in which they appear nor can they increase the size of the neglected terms when the series are truncated. On the other hand, they do not help convergence very much, at least for intervals practicable for computation. It is true that

$x + \sum_{1}^{\infty} \frac{1\cdot3\cdot5\cdots(2n-1)}{2\cdot4\cdot6\cdots(2n)} \cdot \frac{x^{2n+1}}{2n+1}$ = arcsin x converges for x = 1, whereas $\sum_{0}^{\infty} \frac{x^{2n+1}}{2n+1} = \tanh^{-1}x$ does not, but the convergence of the first series is too slow for x near 1. It converges to a specified accuracy in a reasonable number of terms for, say, $|x| \leq 0.7$, but so does the second series. The only other series similar to that for arcsine occurring among the elementary functions is the series for $\sinh^{-1}x$, which is the alternating series of the same form. Such series with even, instead of odd, terms are essentially polynomials in arcsin x or $\sinh^{-1}x$. However, some other functions can be expressed in similar forms and evaluated in the 604, e.g., complete elliptic integrals of the second kind, $E(k) = \int_{0}^{\pi/2} (1-k^2\sin^2\phi)^{\frac{1}{2}}d\phi$. Complete elliptic integrals of the first kind can be easily expressed as double infinite series of a similar nature, but the writer, at least, has not yet been able to adapt them to 604 computation as direct operations because of the difficulty in summing the coefficients of each power of k or, alternatively, summing the powers for each coefficient. The series for F(k) is discussed subsequently.

Evaluating The Log Function

The most useful harmonic-type series are those for log x. Here a leading non-zero digit is not merely a speed-up device but is essential for practical convergence and, in fact, to the use of approximations as well, since the latter are usually defined for $[.1,1.0]$ or some interval which may easily be transformed into this one. For fixed-point arithmetic, the number of places the argument is left-shifted determines the characteristic of the common log; in FD work the power amounts to the same thing. The natural-log series is easily adapted to taking common logs by using $\log_{10}x = M\cdot\log_e x$, where $M = \log_{10}e = .43429448$. The advantage of taking common logs is of course in the use of the characteristic,

since this is independent of the actual digits in the number, place zero's disregarded.

The series $S(x) = \sum_{1}^{\infty} (-1)^{n-1} x^n/n = \log (1+x) = \log t$ is convenient for evaluating logs. This series only converges for $-1 < x \leq 1$, but even this interval is too large to be of practical interest in computation. For $x = 1$, a hundred million terms would be needed to keep the error less than 10^{-8}! For $-1 < x < 0$, $S(x)$ is not even alternating, and even though the series converges and $r(n) < 1$ for all n, it cannot be guaranteed that the error committed by truncating the series is no greater than the magnitude of the first term omitted. Fortunately, however! for $|x| < .5$, $r(n) < .5$; in this case the error committed by truncating the series is less than $|t_m| \sum_{1}^{\infty} 2^{-n} = |t_m|$, where t_m is the last term included, and this is sufficient for the purpose at hand if m is not too large. If $|t_m| < 10^{-8}$ is required, it would mean that, for $|x| < .5$, $.5^m/m < 10^{-8}$. A simple calculation shows that this is satisfied for $m = 23$, which is a quite modest number of terms for electronic computation. Restricting x to (-.5, .5) does not render the series useless by any means, in fact it is quite adequate for evaluating log x for a very large interval if the FD notation is used and if proper but very simple transformations are made. In contrast, the interval (-.5, .5) is practically useless for the exponential-type series. Thus the disadvantage of the slow convergence of $S(x)$ is largely overcome by the small interval needed.

Keeping x in (-.5,.5) requires t in (.5,1.5). Multiplying the first term of the series, t_1, by M and developing t_{n+1} from t_n transforms the natural log to the common log, and this can only help convergence. The FD notation, or the equivalent left-shifting, automatically restricts t (ignoring the characteristic) to $[.1,1.0)$ unless t = 0; special provisions can be made to detect inadvertent zero arguments. To raise the lower bound on t, the relationship $\log t = \log kt - \log k$ can be used. It is convenient to use a power of 2 for k, since $\log_{10} 2 = .30103000$

and hence, even if this number must be emitted during electronic programming, it does not take many programs because there are only 3 non-zero digits. On the 605, both 3's can be emitted at once. By testing the leading digit of t with a digit selector at read-in, kt can be made to lie in $[.6,1.6)$ as follows:

Leading digit of t	k	kt in:
0	-	- - ERROR.
1	8	$[.8,1.6)$
2	4	$[.8,1.2)$
3,4, or 5	2	$[.6,1.2)$
7,8, or 9	1	$[.7,1.0)$

If t is slightly less than .2, kt = 1.6-; but then x = kt-1 = .6- and the series is alternating, which improves the accuracy, for now the truncation error is no greater than the first term omitted. For x = .6, if $|t_n| = (.6^n/n)\,M < 10^{-8}$ is required, n must be allowed to reach 29 - certainly not too many terms to take for the extreme case.

In the second FD set-up described above, an even simpler choice was made for k, namely, k = 4 if kt < 1.4, k = 1 otherwise. This only guarantees kt to lie in $[.35,1.4)$, but very good results are obtained nevertheless. Since $\log_{10}$kt is much less than 1 in this interval, 8 decimals are carried throughout the series, the correction -.60206 = log 1/4 being added, if necessary, at the end (to prevent overflow). Finally, the proper characteristic is added and the results rounded to 7 decimals, which are usually good all the way out.

A fixed-point 604 board was wired, using the left-shifting technique and the above table for k and evaluating 2 terms per sweep. Using calc selectors to move the (actual) decimal point, logs could be evaluated to 7 good decimals for arguments lying between 10^{-8} and $10^{7}-10^{-1}$. However, the speed was between

$33\frac{1}{3}$ and 50 cards per minute. A faster board, practically a constant 50 cards per minute, is now used which gives virtually the same accuracy. This 604 set-up uses a rational approximation[2] of the form

$$\log t \doteq f(x) = C_9x^9 + C_7x^7 + C_5x^5 + C_3x^3 + C_1x + .5,$$

where the C_i's are 9 digit decimals and $x = \frac{t - \sqrt{10}}{t + \sqrt{10}}$ for $1 \leq t \leq 10$. Using $x = \frac{\frac{1}{2}(t - \sqrt{.1})}{\frac{1}{2}(t + \sqrt{.1})}$ and $f(x)$ -1, instead, allows t to lie in $[.1,1.0]$. The C_i's are rounded to 8 digits; and t (8 digits), $\sqrt{.1}$ (to 8 digits), C_9, and C_7 are fed in. C_n, n = 5, 3, 1, is developed within the 604 by multiplying C_{n+2} and emitting proper corrections. This board was much more difficult to program than the series board, but it is faster. For CPC boards the approximation is not feasible within the 605, since it would tie up too many programs for one function; but it is easily sub-programmed as in the first CPC-II set-up described previously.

Evaluating Functions With Exponential-Type Series

The accuracy of the exponential-type functions is limited for general-purpose 8-digit set-ups by the fact that x must be allowed to exceed 1 if the evaluation is to be very useful. Although this does not impede convergence and only moderately increases the number of terms required, it does allow the individual terms to become greater than 1 even though the sum may be quite small. Thus fewer significant figures can be retained on a heavily programmed board.

To use e^x to evaluate 10^t in a manner somewhat the reverse to that used for $\log_e t$ to obtain $\log_{10} t$, x must be allowed to vary over (-2.3025851, 2.3025851), i.e., $(\log_e .1, \log_e 10)$, since the decimal part of t, d(t), will vary over (-1,1) and hence $10^{d(t)}$ over (.1,10). The whole-number part of the argument, $[t]$, can of course be handled separately to shift the decimal point. For the general term of the series for e^x, $|x^n/n!| < 10$ for $|x| \leq 3.9147$ and for all n, in particular n = 3, since the maximum term for $3 < x < 4$ is t_3. Hence it will be sufficient to allow only one

digit for $[t_n]$ for all n with $[x] < \log_e 10$. However, since the last place is bound to be inaccurate, more than 6 good decimals cannot be expected for e^x or 10^t if only 8 places are carried. Similar considerations hold for sin x and cos x. In fact, it would be desirable to allow x to vary over $[-2\pi, 2\pi]$ for sin x and cos x, but this interval is too large even when allowing a place for $[t_n]$. In the set-ups previously described, a reduction of the argument to $[-\pi, \pi]$ must be card-programmed. For a 604 board for only one or two functions, this reduction could no doubt be programmed internally. The argument for sinh x or cosh x is restricted by the range of the function rather than by the magnitude of the terms. Again, for special 604 boards, one could arrange to store the tens and higher-order digits of the sum separately, thus extending the interval up to about (-3.9, 3.9) while keeping 7 decimals. These extra devices impose too great a load on the programming of a general-purpose FD CPC set-up.

For some jobs, the relative error of sin x or cos x as compared with x is the important thing. In such cases, series as used above - i.e., with 7 fixed decimal places - are not of much value. Mr. Hastings' approximations for sin x, as well as some other functions, give a relative error rather than an absolute one. The computation can be carried out as an FD routine or on a double-precision set-up, and good accuracy could be obtained for very small arguments. This is an invaluable method for such jobs.

The Arctangent Function

Perhaps the most difficult elementary function, normally used, to handle by series over an adequate interval is $\tan^{-1}x$. The series for this miserable function has all the disadvantages of the harmonic-type series with none of the saving graces of the log function. The first complication is due to the fact that $T(x) = \sum_0^{\infty} (-1)^n \frac{x^{2n+1}}{2n+1} = \tan^{-1}x$ converges only for $|x| \leq 1$. For $|x| > 1$, the relationship $\tan^{-1}x = \frac{1}{2}\pi(\text{sgn } x) - \tan^{-1}(1/x)$ must be used,

where $\operatorname{sgn} x = \begin{cases} 1 \text{ if } x > 0 \\ -1 \text{ if } x < 0. \end{cases}$ Such a choice is difficult enough to program in itself, and there is the even greater difficulty that the convergence of T(x) for $|x|$ near 1 is theoretical only, since the order of magnitude of n to ensure an error of less than 10^{-p} for such values of x is 10^{p}. T(x) can be evaluated on the 605 so that $|t_n|$ becomes less than 10^{-8} for $2n+1 < 100$ if $|x| < .8709$. By letting $r(n) = mx^2/m+2$, $m = 2n+1$, only the odd-power terms need be evaluated, i.e., actually less than 50 terms. An earlier FD C P C-II set-up than those previously described included this function instead of arcsine. Every imaginable kind of reduction formula was tried, but all proved to be too involved to program internally. Resorting to sub-programming, the following procedure was finally used:

Taking the well-known formula, $\tan 2\phi = \frac{2 \tan \phi}{1-\tan^2\phi}$, let $x = \tan \theta$, $y = \frac{1}{2}\theta$, $|\theta| \leq \frac{1}{2}\pi$, whence $xy^2 + 2y - x = 0$, giving $y = \frac{-1 \pm \sqrt{1-x^2}}{x}$. Since $|v| \leq |x|$, the plus sign must be chosen. Applying this again, $t = \frac{1+\sqrt{1-y^2}}{v} = \theta/4$. Since $|t| \leq .4142 = \tan \pi/8$, the series T(t) will converge quickly. Finally, $\tan^{-1}x = 4 \tan^{-1}t$. This sub-programmed reduction introduced very little loss in accuracy, but it was slow. However, it would handle any argument up to about $\pm .316 \times 10^{25}$. A rational approximation would probably do as well with about the same speed.(3)

The arctangent function is one for which a continued fraction should prove very useful for special-purpose boards. Since $\tan^{-1}x$ is an odd function, the sign can be "remembered" with selectors on the tab or 521. The sgn x in the transformation previously mentioned may then be ignored, and all arguments reduced to $[0,1]$ by not more than one division. The continued fraction for $\tan^{-1}x$,

$$\tan^{-1}x = \cfrac{x}{1 + \cfrac{1\,x^2}{3 + \cfrac{4\,x^2}{5 + \cfrac{9\,x^2}{7 + \ddots}}}}$$

is valid for all real x but would be unwieldy for large x. It is unlikely that the magnitude of x could be tested, the continued fraction evaluated, and the correction of $\frac{1}{2}\pi - \tan^{-1}(1/x)$ made when necessary, all within a 604, but it might well be done in 2 operations. It is impossible to evaluate a continued fraction from the top down with 8-digit numbers in a 604 or 605 because of storage limitations. In fact it requires all the storage to evaluate them from the bottom up. This puts continued fractions at a disadvantage with respect to transformations as compared with series. The continued fraction has no sum to which a correction can be pre-added, and there is no "first term" which can be pre-multiplied. Furthermore, the index of the bottom quotient used must be prefixed at a large enough value to allow for the worst case in the interval allowed.* This is wasteful in time, since the worst case will seldom be encountered. However, for a function like $\tan^{-1}x$, which is so troublesome to handle with series, the advantage of the larger interval allowed by a continued fraction outweighs all its disadvantages.

Necessary Conditions For Efficient Use of Series in the 604 or 605

What is required of a series in order to be able to program its complete evaluation within a 604 or 605, might be summed up a little more precisely as follows:

There must exist a useful interval (a,b) and a form of t_n such that, for x in (a,b),

(1) $\left| t_{n+1}/t_n \right| = r(n,x)$ is recursive and is both sufficiently simple and small enough for manipulation within the computer;

---

*Sometimes series are evaluated this way - i.e., as a polynomial of high degree - the "starting" (the highest) n being fed in with the argument and subject to the discretion of the coder. There appears to be little advantage to this method. It takes so little more time to attain maximum accuracy, once the series is started, that there seems to be no reason why a coder should have to compute or guess at the proper n, even assuming that he knows what the argument will be.

Those familiar with CPC-II timing, will realize that once a delay cycle is taken, all of it is then available for computation, which is several times the interval alloted to a card cycle as "compute time."

(2) an N can be chosen which will keep r(n,x) within necessary bounds for $n < N$ and which will allow, for desired p,

$$\left| \sum_{N}^{\infty} t_n(x) \right| < 10^{-p}; \text{ and}$$

(3) 10^q for which $\left| t_n(x) \right| < 10^q$ for all n, S for which $\left| \sum t_n(x) \right| \leq S$, and $\max \left\{ |a| , |b| \right\}$ are of comparable magnitude and compatible with p in (2).

It is best not to be too stringent with (a,b) or some useful series, such as the log series, are likely to be summarily dismissed. Likewise, whether or not r(n,x) is "sufficiently simple" depends partly on the ingenuity of the programmer. The interdependence of the demands makes more precise statements difficult.

Examples of Evaluating Non-Elementary Functions in the 604.

In conclusion, two examples are given to show how a series can sometimes be used for rapid evaluation, within restricted intervals, of non-elementary functions on the 604:

(A) the probability integral, $\phi(x) = (2\pi)^{-\frac{1}{2}} \int_{-x}^{x} e^{-\frac{1}{2}t^2} \, dt$, and

(B) the elliptic integral, $E(k) = \int_{0}^{\pi/2} (1-k^2 \sin^2 \phi)^{\frac{1}{2}} d\phi$.

(A) Since $\phi(x)$ is an even function and the series for e^x is uniformly convergent over any interval, we can write, where $k = \sqrt{2/\pi}$,

$$\phi(x) = k \int_{0}^{x} \left(1 - \frac{t^2}{2} + \frac{t^4}{2^2 2!} - \frac{t^6}{2^3 3!} + \ldots + (-1)^n \frac{t^{2n}}{2^n n!} + \ldots\right) dt$$

$$= k \left[x - \frac{x^3}{2 \cdot 3} + \frac{x^5}{2^2 \cdot 2! \cdot 5} - \frac{x^7}{2^3 \cdot 3! \cdot 7} + \ldots + (-1)^n \frac{x^{2n+1}}{2^n \cdot n! (2n+1)} + \ldots \right]$$

Letting $y = \frac{1}{2}x^2$ and factoring out an x, we have

$$\phi(x) = kx \sum_{0}^{\infty} \frac{y^n}{(2n+1)n!}$$

For x in $[0,b)$, if we wish to carry only one digit for $[y]$, we must have $b < 4.472$. To keep $|t_n| < 10$ also, b is further restricted. We can find this value by setting

$n = 1, 2, \ldots$ and solving $|t_n| = \frac{kx^{2n+1}}{2^n(2n+1)n!} = f_n(x) = 10$ for x, say $f_n(x_n) = 10$, since the initial kx must be carried through all the terms in the actual programming to save storage. Since $f_n'(x) > 0$, for all n and $x > 0$, $b = \min\{x_n\}$ is obviously necessary and sufficient to ensure $|t_n| < 10$ for x in $[0,b)$ and all n. Computation gives:

$$x_1 = 4.2209,\ x_2 = 3.4676,\ x_3 = 3.2922,\ x_4 = 3.2748,\ x_5 = 3.3139,\ x_6 = 3.3798.$$

Since $x_4 < x_5$, $f_4(x_4) > f_5(x_4)$, i.e., $r(4) < 1$ for $x = x_4$. Since the series is an exponential type, $r(4+p) < 1$ for $x = x_4$ and hence $f_4(x_4) > f_{4+p}(x_4)$, $p = 1, 2, \ldots$. Therefore, $x_4 = \min\{x_n\}$. If we similarly compute b' such that $|t_n| < 100$ for x in $[0,b')$, we find that b' is only 4.0326. If we are more interested in accuracy, it is better to be satisfied with b, since $x = b'$ is a pretty large argument for $\phi(x)$ anyway.

We can carry 8 decimals for the sum by ignoring the whole numbers in the leading terms, for, since $0 < \phi(x) < 1$ for all x, these must all cancel, and the worst error we can commit by ignoring them is to come out short by exactly 1. We can simply add 1 to the result and ignore whole numbers when reading out the answer. A difficulty arises in the comoutation, however, since $r(n-1) = \frac{(2n-1)|y|}{(2n+1)n}$, and we need to keep 9 decimals for $(2n-1)t_{n-1}y$ to maintain 7 good decimals for $\phi(x)$. But if $n \geq 23$, $2n^2 + n > 1000$, which is too large for the division in the 604 without using extra programs which are not available. Limiting n to 22 reduces b to about 3.15 if we are to require $|t_n|$ to become less than 10^{-7}. A board has been wired which carries 7 decimals for t_n if $0 \leq x < 3$, and which carries 6 if $3 \leq x < 4$. For $4 \leq x < 10$, the computation is suppressed and a constant .9999900 is emitted for $\phi(x)$; also a negative balance selector is picked up to be used as desired. For $0 \leq x < 3$, the maximum error in $\phi(x)$ is 10^{-7}; for $3 \leq x < 4$, it is 10^{-6}. Speed is somewhat better than $33\frac{1}{3}$ cards per minute on the average.

B. To express E(k) as a series, we first expand $(1-t^2)^{\frac{1}{2}}$ in a Taylor's series. This series is uniformly convergent for $t^2 \leq b < 1$; although setting $t^2 = k^2\sin^2\phi$, where $k = \sin\theta$, allows t^2 to equal 1, we can restrict k^2 to $[0,b)$, $b < 1$ to bound t^2 away from 1. This restriction is less severe than the one we shall eventually have to make anyway. We then have,

$$E(k) = \int_0^{\pi/2} (1-k^2\sin^2\phi)^{\frac{1}{2}} d\phi = \int_0^{\pi/2} \left[1 - \frac{k^2\sin^2\phi}{2} - \sum_2^{\infty} \frac{1\cdot3\cdot5\cdots(2n-3)}{2\cdot4\cdot6\cdots(2n-2)} \frac{k^{2n}\sin^{2n}\phi}{2n} \right] d\phi.$$

Integrating term by term and using the definite integral

$$\int_0^{\pi/2} \sin^{2n}\phi \, d\phi = \frac{1\cdot3\cdot5\cdots(2n-1)}{2\cdot4\cdot6\cdots(2n)} \frac{\pi}{2},$$

we have $E(k) = \frac{\pi}{2}\left\{ 1 - \frac{k^2}{4} - \sum_2^{\infty} \left[\frac{1\cdot3\cdot5\cdots(2n-1)}{2\cdot4\cdot6\cdots(2n)} \right]^2 \frac{k^{2n}}{2n-1} \right\} = \frac{\pi}{2} \sum_0^{\infty} t_n,$

where $t_o = 1$, $t_1 = -\frac{1}{4}k^2$, and $t_{n+1} = \frac{(4n^2-1)}{(2n+2)^2} k^2 \; t_n = \frac{-4n^2+1}{(n+1)^2} \; t_1 \; t_n$, $n \geq 1$.

This is not too difficult to program on the 604, but the interval is restricted. It will probably not be possible to allow more than 7 decimals for t because of the magnitude of the expressions $4n^2-1$ and $(n+1)^2$, which will have to be developed as a multiplier and a divisor, respectively. Seven decimals can be carried for the sum which allows $\pi/2$ to be inputted at read time and carried as a factor throughout all the terms.

For $k = .96296 = \sin 74^\circ21' + $, $|t_{100}| < 10^{-7}$, but this is not an upper bound to the error, since the series is non-alternating. Since $\frac{4n^2-1}{(n-1)^2} \longrightarrow 4$ as $n \longrightarrow \infty$, if $|t_1| = k^2/4 \leq 1/8$, we can say the error is no greater than the last term included. This requires $k \leq \sqrt{.5} = .707107$. Actually 6 good decimals can probably be obtained for considerably higher values, say .8 or even .85. In any event, as soon as t_n is less then the allowed tolerance, the computation will stop if it has been programmed properly.

References Superscripted in Text

1. "APPROXIMATIONS IN NUMERICAL ANALYSIS" (formerly "Cumulative Index of Approximations"), Cecil Hastings, Jr., The RAND Corporation

2. Ibid., Sheet #4.

3. Ibid., see sheet #12.