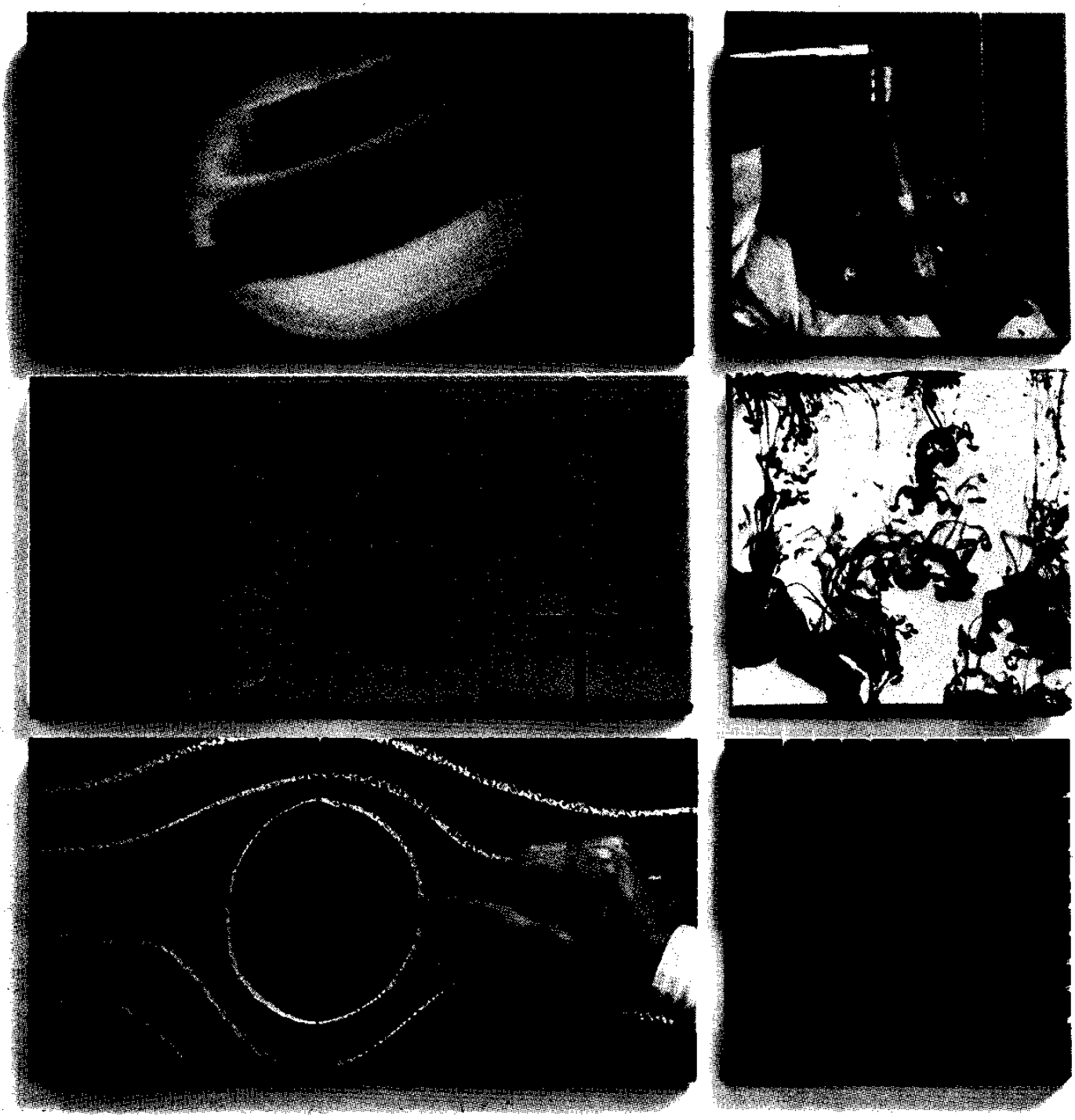
HEWLETT-PACKARD

HP-15C

ADVANCED FUNCTIONS HANDBOOK



NOTICE

Hewlett-Packard Company makes no express or implied warranty with regard to the keystroke procedures and program material offered or their merchantability or their fitness for any particular purpose. The keystroke procedures and program material are made available solely on an "as is" basis, and the entire risk as to their quality and performance is with the user. Should the keystroke procedures or program material prove defective, the user (and not Hewlett-Packard Company nor any other party) shall bear the entire cost of all necessary correction and all incidental or consequential damages. Hewlett-Packard Company shall not be liable for any incidental or consequential damages in connection with or arising out of the furnishing, use, or performance of the keystroke procedures or program material.

HP-15C

Advanced Functions Handbook

August 1982

00015-90011

Printed in U.S.A.

Contents

Introduction

The HP-15C provides several advanced capabilities never before combined so conveniently in a handheld calculator:

- Finding the roots of equations.
- Evaluating definite integrals.
- Calculating with complex numbers.
- Calculating with matrices.

The *HP-15C Owner's Handbook* gives the basic information about performing these advanced operations. It also includes numerous examples that show how to use these features. The owner's handbook is your primary reference for information about the advanced functions.

This *HP-15C Advanced Functions Handbook* continues where the owner's handbook leaves off. In this handbook you will find information about how the HP-15C performs the advanced computations and information that explains how to interpret the results that you get.

This handbook also contains numerous programs, or applications. These programs serve two purposes. First, they suggest ways of using the advanced functions, so that you might use these capabilities more effectively in your own applications. Second, the programs cover a wide range of applications—they may be useful to you in the form presented in this handbook.

> Note: The discussions of most topics in this handbook presume that you already understand the basic information about using the advanced functions and that you are generally familiar with the subject matter being discussed.

Section 1

Using [SOLVE] Effectively

The [SOLVE] algorithm provides an effective method for finding a root of an equation. This section describes the numerical method used by [SOLVE] and gives practical information about using [SOLVE] in various situations.

Finding Roots

In general, no numerical technique can be guaranteed to find a root of every equation that has one. Because a finite number of digits are used, the calculated function may differ from the theoretical function in certain intervals of x, it may not be possible to represent the roots exactly, or it may be impossible to distinguish between zeros and discontinuities of the function being used. Because the function can be sampled at only a finite number of places, it's also possible to conclude falsely that the equation has no roots.

Despite these inherent limitations on any numerical method for finding roots, an effective method—like that used by [SOLVE]—should strive to meet each of the following objectives:

- If a real root exists and can be exactly represented by the calculator, it should be returned. Note that the calculated function may underflow (and be set to zero) for some values of x other than the true roots.
- If a real root exists, but it can't be exactly represented by the calculator, the value returned should differ from the true root only in the last significant digit.
- If no real root exists, an error message should be displayed.

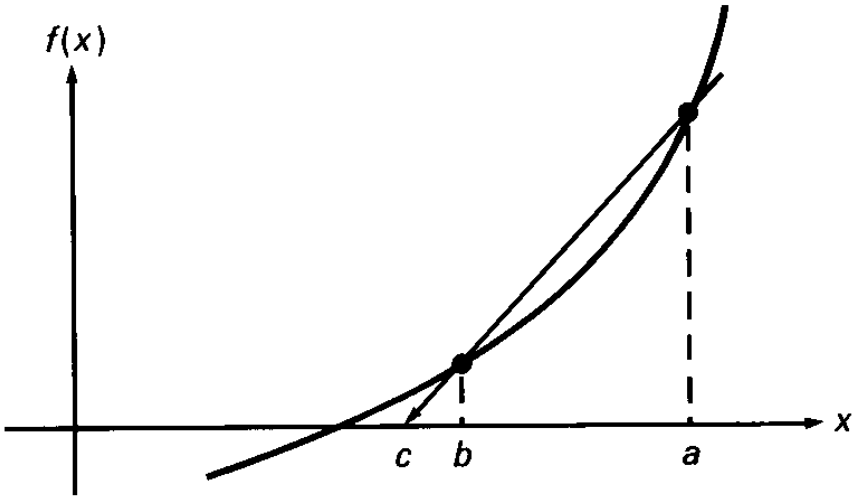
The [SOLVE] algorithm was designed with these objectives in mind. It is also easy to use and requires little of the calculator's memory. And because [SOLVE] in a program can detect the situation of not finding a root, your programs can remain entirely automatic regardless of whether [SOLVE] finds a root.

How [SOLVE] Samples

The [SOLVE] routine uses only five registers of allocatable memory in the HP-15C. The five registers hold three sample values (a, b, and c) and two previous function values ($f(a)$ and $f(b)$) while your function subroutine calculates $f(c)$.

The key to the effectiveness of [SOLVE] is how the next sample value c is found.

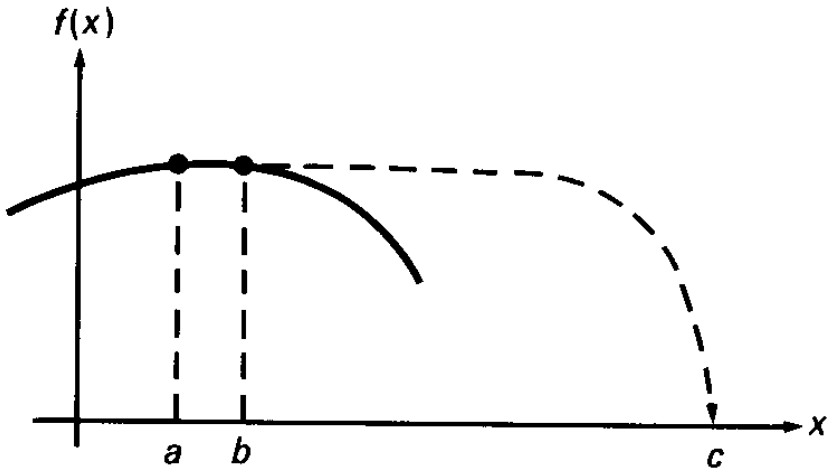
Normally, [SOLVE] uses the secant method to select the next value. This method uses the values of a, b, $f(a)$, and $f(b)$ to predict a value c where $f(c)$ might be close to zero.



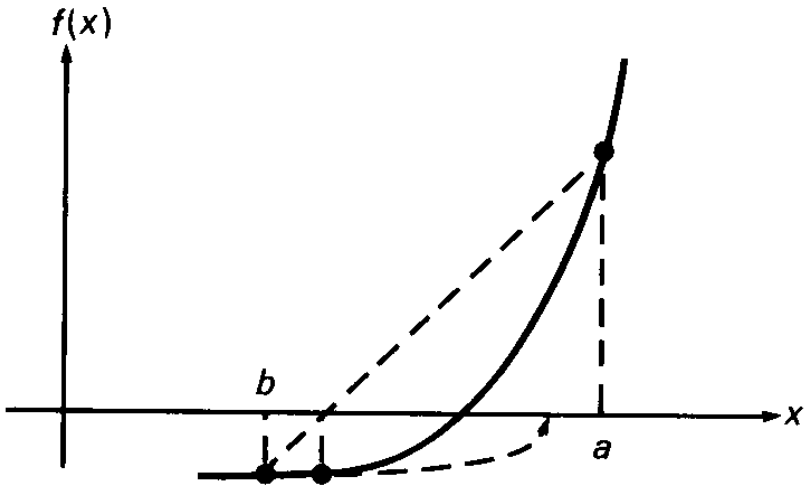
If c isn't a root, but $f(c)$ is closer to zero than $f(b)$, then b is relabeled as a, c is relabeled as b, and the prediction process is repeated. Provided the graph of $f(x)$ is smooth and provided the initial values of a and b are close to a simple root, the secant method rapidly converges to a root.

However, under certain conditions the secant method doesn't suggest a next value that will bound the search or move the search closer to a root, such as finding a sign change or a smaller function magnitude. In such cases, [SOLVE] uses a different approach.

If the calculated secant is nearly horizontal, [SOLVE] modifies the secant method to ensure that $|c - b| \leq 100|a - b|$. This is especially important because it also reduces the tendancy for the secant method to go astray when rounding error becomes significant near a root.



If [SOLVE] has already found values a and b such that $f(a)$ and $f(b)$ have opposite signs, it modifies the secant method to ensure that c always lies within the interval containing the sign change. This guarantees that the search interval decreases with each iteration, eventually finding a root.



If [SOLVE] hasn't found a sign change and a sample value c doesn't yield a function value with diminished magnitude, then [SOLVE] fits a parabola through the function values at a, b, and c. [SOLVE] finds the value d at which the parabola has its maximum or minimum, relabels d as a, and then continues the search using the secant method.

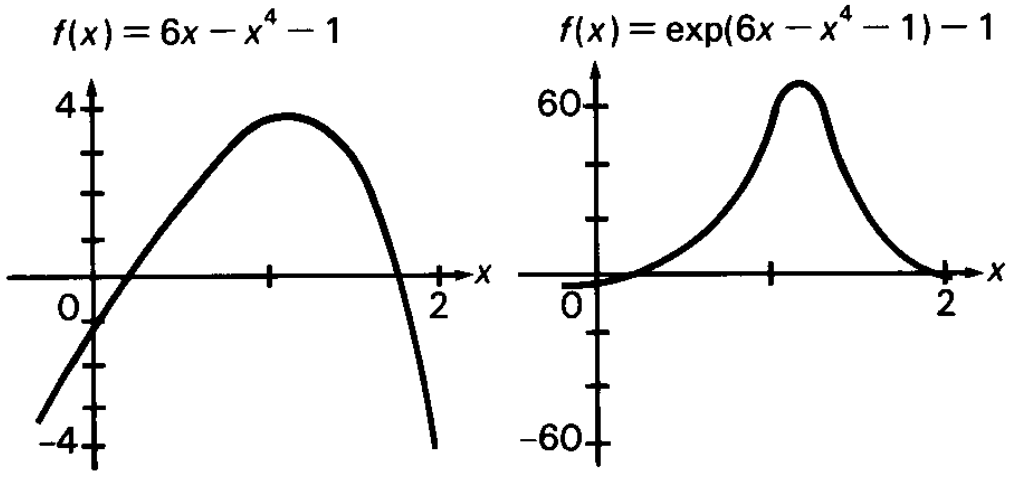
[SOLVE] abandons the search for a root only when three successive parabolic fits yield no decrease in the function magnitude or when $d = b$. Under these conditions, the calculator displays **Error 8**. Because b represents the point with the smallest sampled function magnitude, b and $f(b)$ are returned in the X- and Z-registers, respectively. The Y-register contains the value of a or c. With this information, you can decide what to do next. You might resume the search where it left off, or direct the search elsewhere, or decide that $f(b)$ is negligible so that $x = b$ is a root, or transform the equation into another equation easier to solve, or conclude that no root exists.

Handling Troublesome Situations

The following information is useful for working with problems that could yield misleading results. Inaccurate roots are caused by calculated function values that differ from the intended function values. You can frequently avoid trouble by knowing how to diagnose inaccuracy and reduce it.

Easy Versus Hard Equations

The two equations $f(x) = 0$ and $e^{f(x)} - 1 = 0$ have the same real roots, yet one is almost always much easier to solve numerically than the other. For instance, when $f(x) = 6x - x^4 - 1$, the first equation is easier. When $f(x) = \ln(6x - x^4)$, the second is easier. The difference lies in how the function's graph behaves, particularly in the vicinity of a root.



In general, every equation is one of an infinite family of equivalent equations with the same real roots. And some of those equations must be easier to solve than others. While [SOLVE] may fail to find a root for one of those equations, it may succeed with another.

Inaccurate Equations

[SOLVE] can't calculate an equation's root incorrectly *unless the function is incorrectly calculated.* The accuracy of your function subroutine affects the accuracy of the root that you find.

You should be aware of conditions that might cause your calculated function value to differ from the theoretical value you want it to have. [SOLVE] can't infer intended values of your function. Frequently, you can minimize calculation error by carefully writing your function subroutine.

Equations With Several Roots

The task of finding all roots of an equation becomes more difficult as the number of roots increases. And any roots that cluster closely will usually defy attempts at accurate resolution. You can use *deflation* to eliminate roots, as described in the *HP-15C Owner's Handbook.*

An equation with a multiple root is characterized by the function and its first few higher-order derivatives being zero at the multiple root. When [SOLVE] finds a double root, the last half of its digits may be inaccurate. For a triple root, two-thirds of the root's digits tend to be obscured. A quadruple root tends to lose about three-fourths of its digits.

Using [SOLVE] With Polynomials

Polynomials are among the easiest functions to evaluate. That is why they are traditionally used to approximate functions that model physical processes or more complex mathematical functions.

A polynomial of degree n can be represented as

$$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 \ .$$

This function equals zero at no more than n real values of x, called zeros of the polynomial. A limit to the number of *positive* zeros of this function can be determined by counting the number of times

the signs of the coefficients change as you scan the polynomial from left to right. Similarly, a limit to the number of *negative* zeros can be determined by scanning a new function obtained by substituting $-x$ in place of x in the original polynomial. If the actual number of real positive or negative zeros is less than its limit, it will differ by an even number. (These relationships are known as Descartes' Rule of Signs.)

As an example, consider the third-degree polynomial function

$$f(x) = x^3 - 3x^2 - 6x + 8 .$$

It can have no more than three real zeros. It has at most two positive real zeros (observe the sign changes from the first to second and third to fourth terms) and at most one negative real zero (obtained from $f(-x) = -x^3 - 3x^2 + 6x + 8$).

Polynomial functions are usually evaluated most compactly using nested multiplication. (This is sometimes referred to as Horner's method.) As an illustration, the function from the previous example can be rewritten as

$$f(x) = [(x - 3)x - 6]x + 8 .$$

This representation is more easily programmed and more efficiently executed than the original form, especially since [SOLVE] fills the stack with the value of x.

Example: During the winter of '78, Arctic explorer Jean-Claude Coulerre, isolated at his frozen camp in the far north, began scanning the southern horizon in anticipation of the sun's reappearance. Coulerre knew that the sun would not be visible to him until early March, when it reached a declination of 5°18′S. On what day and time in March was the chilly explorer's vigil rewarded?

The time in March when the sun reached 5°18′S declination can be computed by solving the following equation for t:

$$D = a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0$$

where D is the declination in degrees, t is the time in days from the beginning of the month, and

$$a_4 = \ 4.2725 \times 10^{-8}$$
$$a_3 = -1.9931 \times 10^{-5}$$
$$a_2 = \ 1.0229 \times 10^{-3}$$
$$a_1 = \ 3.7680 \times 10^{-1}$$
$$a_0 = -8.1806 \ .$$

This equation is valid for $1 \leqslant t < 32$, representing March, 1978.

First convert 5°18′S to decimal degrees (press 5.18 [CHS] [g] [→H]), obtaining −5.3000 (using [FIX] 4 display mode). (Southern latitudes are expressed as negative numbers for calculation purposes.)

The solution to Coulerre's problems is the value of t satisfying

$$-5.3000 = a_4t^4 + a_3t^3 + a_2t^2 + a_1t + a_0.$$

Expressed in the form required by [SOLVE], the equation is

$$0 = a_4t^4 + a_3t^3 + a_2t^2 + a_1t - 2.8806$$

where the last, constant term now incorporates the value of the declination.

Using Horner's method, the function to be set equal to zero is

$$f(t) = (((a_4t + a_3)t + a_2)t + a_1)t - 2.8806 \ .$$

To shorten the subroutine, store and recall the constants using the registers corresponding to the exponent of t.

Keystrokes	Display	
[ON] / [−]	**Pr Error**	Clears calculator's memory.*
[←]	**0.0000**	
[g] [P/R]	**000–**	Program mode.

*This step is included here only to ensure that sufficient memory is available for the examples that follow in this handbook.

Keystrokes	Display
[f] [LBL] [A]	001-42,21,11
[RCL] 4	002- 45 4
[×]	003- 20
[RCL] 3	004- 45 3
[+]	005- 40
[×]	006- 20
[RCL] 2	007- 45 2
[+]	008- 40
[×]	009- 20
[RCL] 1	010- 45 1
[+]	011- 40
[×]	012- 20
[RCL] 0	013- 45 0
[+]	014- 40
[g] [RTN]	015- 43 32

In Run mode, key in the five coefficients:

Keystrokes	Display		
[g] [P/R]			Run mode.
4.2725 [EEX] 8 [CHS]	4.2725	-08	
[STO] 4	4.2725	-08	Coefficient of t^4.
1.9931 [CHS] [EEX] 5 [CHS] [STO] 3	-1.9931	-05	Coefficient of t^3.
1.0229 [EEX] 3 [CHS]	1.0229	-03	
[STO] 2	0.0010		Coefficient of t^2.
3.7680 [EEX] 1 [CHS]	3.7680	-01	
[STO] 1	0.3768		Coefficient of t.
2.8806 [CHS] [STO] 0	-2.8806		Constant term.

Because the desired solution should be between 1 and 32, key in these two values for initial estimates. Then use [SOLVE] to find the roots.

Keystrokes	Display	
1 [ENTER]	1.0000	
32	32	Initial estimates.
[f] [SOLVE] [A]	7.5137	Root found.
[R↓]	7.5137	Same previous estimate.

Keystrokes	Display	
[R↓]	0.0000	Function value.
[g] [R↑] [g] [R↑]	7.5137	Restores stack.

The day was March 7th. Convert the fractional portion of the number to decimal hours and then to hours, minutes, and seconds.

Keystrokes	Display	
[f] [FRAC]	0.5137	Fractional portion of day.
24 [×]	12.3293	Decimal hours.
[f] [→H.MS]	12.1945	Hours, minutes, seconds.

Explorer Coulerre should expect to see the sun on March 7th at $12^h\ 19^m\ 45^s$ (Coordinated Universal Time).

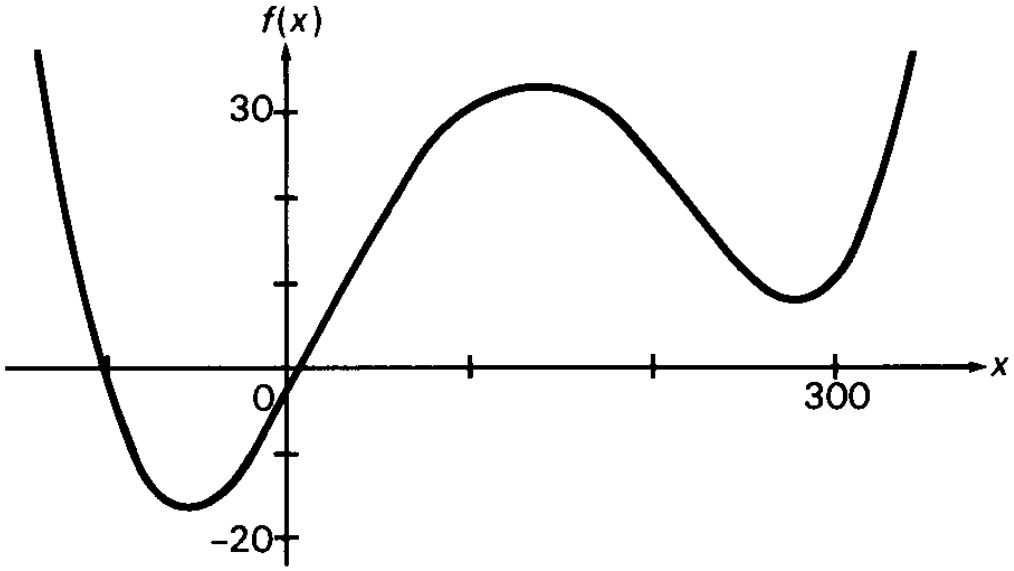
By examining Coulerre's function $f(t)$, you realize that it can have as many as four real roots—three positive and one negative. Try to find additional positive roots by using [SOLVE] with larger positive estimates.

Keystrokes	Display	
1000 [ENTER] 1100	1,100	Two larger, positive estimates.
[f] [SOLVE] [A]	Error 8	No root found.
[←]	278.4497	Last estimate tried.
[R↓]	276.7942	A previous estimate.
[R↓]	7.8948	Nonzero value of function.
[g] [R↑] [g] [R↑]	278.4497	Restores stack to original state.
[f] [SOLVE] [A]	Error 8	Again, no root found.
[←]	278.4398	Approximately same estimate.
[R↓]	278.4497	A previous estimate.
[R↓]	7.8948	Same function value.

You have found a positive local minimum rather than a root. Now try to find the negative root.

Keystrokes	Display	
1000 [CHS] [ENTER]	-1,000.0000	
1100 [CHS]	-1,100	Two larger, negative estimates.
[f] [SOLVE] [A]	-108.9441	Negative root.
[R↓]	-108.9441	Same previous estimate.
[R↓]	1.6000 -08	Function value.

There is no need to search further—you have found all possible roots. The negative root has no meaning since it is outside of the range for which the declination approximation is valid. The graph of the function confirms the results you have found.



Solving a System of Equations

[SOLVE] is designed to find a single variable value that satisfies a single equation. If a problem involves a system of equations with several variables, you may still be able to [SOLVE] to find a solution.

For some systems of equations, expressed as

$$f_1(x_1, \ldots, x_n) = 0$$
$$\vdots$$
$$f_n(x_1, \ldots, x_n) = 0$$

it is possible through algebraic manipulation to eliminate all but one variable. That is, you can use the equations to derive

expressions for all but one variable in terms of the remaining variable. By using these expressions, you can reduce the problem to using [SOLVE] to find the root of a single equation. The values of the other variables at the solution can then be calculated using the derived expressions.

This is often useful for solving a complex equation for a complex root. For such a problem, the complex equation can be expressed as two real-valued equations—one for the real component and one for the imaginary component—with two real variables—representing the real and imaginary parts of the complex root.

For example, the complex equation $z + 9 + 8e^{-z} = 0$ has no real roots z, but it has infinitely many complex roots $z = x + iy$. This equation can be expressed as two real equations

$$\begin{aligned} x + 9 + 8e^{-x}\cos y &= 0 \\ y \qquad - 8e^{-x}\sin y &= 0\ . \end{aligned}$$

The following manipulations can be used to eliminate y from the equations. Because the sign of y doesn't matter in the equations, assume $y > 0$, so that any solution (x,y) gives another solution $(x,-y)$. Rewrite the second equation as

$$x = \ln(8(\sin y)/y),$$

which requires that $\sin y > 0$, so that $2n\pi < y < (2n + 1)\pi$ for integer $n = 0, 1, \ldots$.

From the first equation

$$\begin{aligned} y &= \cos^{-1}(-e^x(x + 9)/8) + 2n\pi \\ &= (2n + 1)\pi - \cos^{-1}(e^x(x + 9)/8) \end{aligned}$$

for $n = 0, 1, \ldots$ Substitute this expression into the second equation,

$$x + \ln\left(\frac{(2n + 1)\pi - \cos^{-1}(e^x(x + 9)/8)}{\sqrt{64 - (e^x(x + 9))^2}}\right) = 0.$$

You can then use SOLVE to find the root x of this equation (for any given value of n, the number of the root). Knowing x, you can calculate the corresponding value of y.

A final consideration for this example is to choose the initial estimates that would be appropriate. Because the argument of the inverse cosine must be between -1 and 1, x must be more negative than about -0.1059 (found by trial and error or by using SOLVE). The initial guesses might be near but more negative than this value, -0.11 and -0.2 for example.

(The complex equation used in this example is solved using an iterative procedure in the example on page 81. Another method for solving a system of nonlinear equations is described on page 122.)

Finding Local Extremes of a Function

Using the Derivative

The traditional way to find local maximums and minimums of a function's graph uses the *derivative* of the function. The derivative is a function that describes the slope of the graph. Values of x at which the derivative is zero represent potential local extremes of the function. (Although less common for well-behaved functions, values of x where the derivative is infinite or undefined are also possible extremes.) If you can express the derivative of a function in closed form, you can use SOLVE to find where the derivative is zero—showing where the function may be maximum or minimum.

Example: For the design of a vertical broadcasting tower, radio engineer Ann Tenor wants to find the angle from the tower at which the relative field intensity is most negative. The relative intensity created by the tower is given by

$$E = \frac{\cos(2\pi h \cos\theta) - \cos(2\pi h)}{[1 - \cos(2\pi h)]\sin\theta}$$

where E is the relative field intensity, h is the antenna height in wavelengths, and θ is the angle from vertical in radians. The height is 0.6 wavelengths for her design.

The desired angle is one at which the derivative of the intensity with respect to θ is zero.

To save program memory space and execution time, store the following constants in registers and recall them as needed:

$r_0 = 2\pi h$ and is stored in register R_0,

$r_1 = \cos(2\pi h)$ and is stored in register R_1,

$r_2 = 1/[1 - \cos(2\pi h)]$ and is stored in register R_2.

The derivative of the intensity E with respect to the angle θ is given by

$$\frac{dE}{d\theta} = r_2\left[r_0\sin(r_0\cos\theta) - \frac{\cos(r_0\cos\theta) - r_1}{\sin\theta\tan\theta}\right].$$

Key in a subroutine to calculate the derivative.

Keystrokes	Display	
[g] [P/R]		Program mode.
[f] CLEAR [PRGM]	000-	
[f] [LBL] 0	001-42,21, 0	
[COS]	002- 24	
[RCL] 0	003- 45 0	
[×]	004- 20	
[COS]	005- 24	
[RCL] 1	006- 45 1	
[−]	007- 30	
[x≷y]	008- 34	
[SIN]	009- 23	
[÷]	010- 10	
[x≷y]	011- 34	
[TAN]	012- 25	
[÷]	013- 10	
[CHS]	014- 16	
[x≷y]	015- 34	
[COS]	016- 24	
[RCL] 0	017- 45 0	

Keystrokes	Display
[×]	018- 20
[SIN]	019- 23
[RCL] 0	020- 45 0
[×]	021- 20
[+]	022- 40
[RCL] 2	023- 45 2
[×]	024- 20
[g] [RTN]	025- 43 32

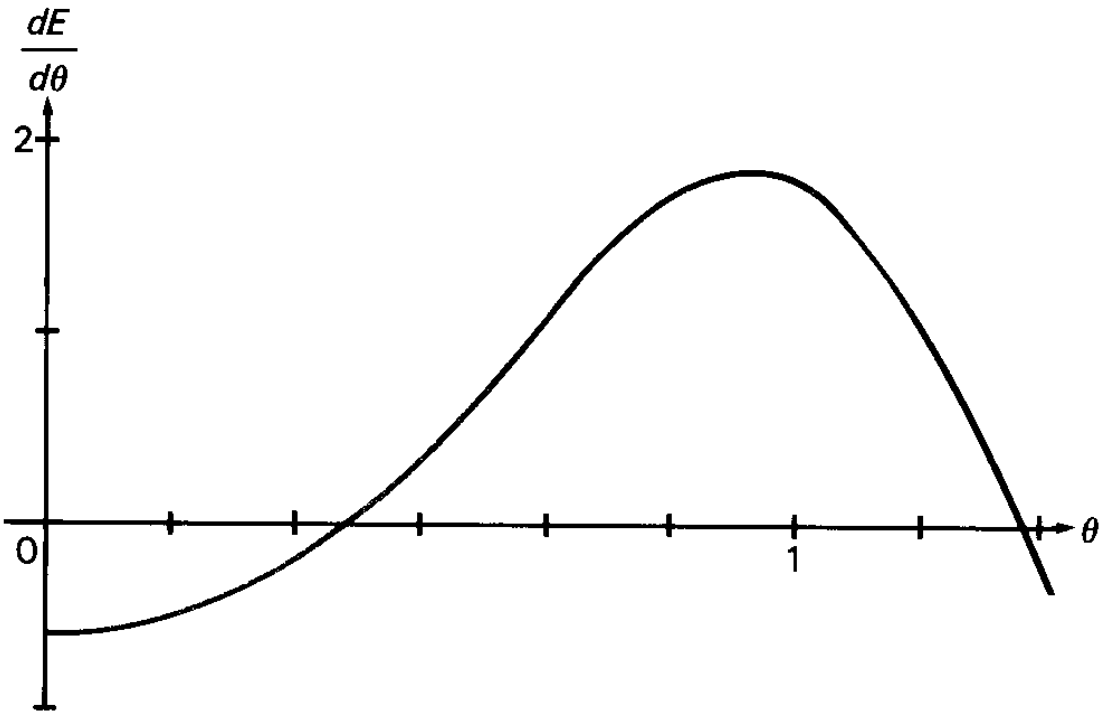
In Radians mode, calculate and store the three constants.

Keystrokes	Display	
[g] [P/R]		Run mode.
[g] [RAD]		Specifies Radians mode.
2 [g] [π] [×]	6.2832	
.6 [×] [STO] 0	3.7699	Constant r_0.
[COS] [STO] 1	-0.8090	Constant r_1.
[CHS] 1 [+]	1.8090	
[1/x] [STO] 2	0.5528	Constant r_2.

The relative field intensity is maximum at an angle of 90° (perpendicular to the tower). To find the minimum, use angles closer to zero as initial estimates, such as the radian equivalents of 10° and 60°.

Keystrokes	Display		
10 [f] [→RAD]	0.1745		
60 [f] [→RAD]	1.0472		Initial estimates.
[f] [SOLVE] 0	0.4899		Angle giving zero slope.
[R↓] [R↓]	-5.5279	-10	Slope at specified angle.
[g] [R↑] [g] [R↑]	0.4899		Restores the stack.
[g] [→DEG]	28.0680		Angle in degrees.

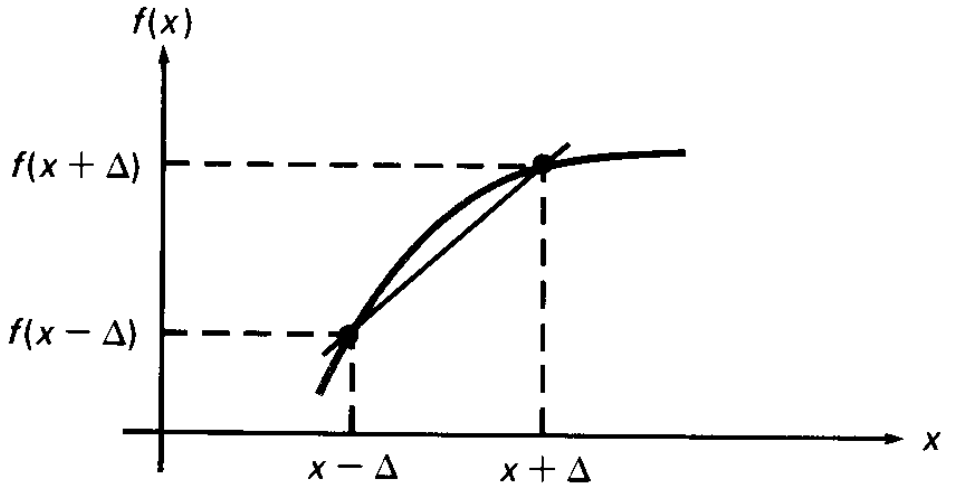
The relative field intensity is most negative at an angle of 28.0680° from vertical.



Using an Approximate Slope

The derivative of a function can also be approximated numerically. If you sample a function at two points relatively close to x (namely $x + \Delta$ and $x - \Delta$), you can use the slope of the secant as an approximation to the slope at x:

$$s = \frac{f(x + \Delta) - f(x - \Delta)}{2\Delta}$$



The accuracy of this approximation depends upon the increment Δ and the nature of the function. Smaller values of Δ give better approximations to the derivative, but excessively small values can cause round-off inaccuracy. A value of x at which the slope is zero is potentially a local extreme of the function.

Example: Solve the previous example without using the equation for the derivative $dE/d\theta$.

Find the angle at which the derivative (determined numerically) of the intensity E is zero.

In Program mode, key in two subroutines: one to estimate the derivative of the intensity and one to evaluate the intensity function E. In the following subroutine, the slope is calculated between $\theta + 0.001$ and $\theta - 0.001$ radians (a range equivalent to approximately 0.1°).

Keystrokes	Display	
[g] [P/R]	000-	Program Mode.
[f] [LBL] [A]	001-42,21,11	
[EEX]	002- 26	
[CHS]	003- 16	
3	004- 3	Evaluates E at $\theta + 0.001$.
[+]	005- 40	
[ENTER]	006- 36	
[GSB] [B]	007- 32 12	
[x≷y]	008- 34	
[EEX]	009- 26	
[CHS]	010- 16	
3	011- 3	Evaluates E at $\theta - 0.001$.
[-]	012- 30	
[ENTER]	013- 36	
[GSB] [B]	014- 32 12	
[-]	015- 30	
2	016- 2	
[EEX]	017- 26	
[CHS]	018- 16	
3	019- 3	

Keystrokes	Display	
[÷]	020- 10	
[g] [RTN]	021- 43 32	
[f] [LBL] [B]	022-42,21,12	Subroutine for $E(\theta)$.
[COS]	023- 24	
[RCL] 0	024- 45 0	
[×]	025- 20	
[COS]	026- 24	
[RCL] 1	027- 45 1	
[−]	028- 30	
[x≷y]	029- 34	
[SIN]	030- 23	
[÷]	031- 10	
[RCL] 2	032- 45 2	
[×]	033- 20	
[g] [RTN]	034- 43 32	

In the previous example, the calculator was set to Radians mode and the three constants were stored in registers R_0, R_1, and R_2. Key in the same initial estimates as before and execute [SOLVE].

Keystrokes	Display	
[g] [P/R]		Run mode.
10 [f] [→RAD]	0.1745	
60 [f] [→RAD]	1.0472	Initial estimates.
[f] [SOLVE] [A]	0.4899	Angle given zero slope.
[R↓] [R↓]	0.0000	Slope at specified angle.
[g] [R↑] [g] [R↑]	0.4899	Restores the stack.
[ENTER] [ENTER] [f] [B]	−0.2043	Uses function subroutine to calculate minimum intensity.
[x≷y]	0.4899	Recalls θ value.
[g] [→DEG]	28.0679	Angle in degrees.

This numerical approximation of the derivative indicates a minimum field intensity of −0.2043 at an angle of 28.0679°. (This angle differs from the previous solution by 0.0001°.)

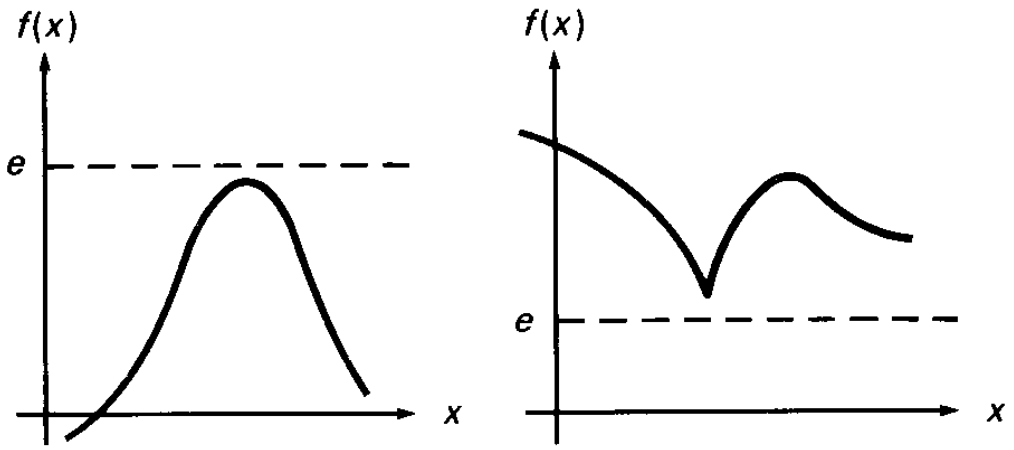
Using Repeated Estimation

A third technique is useful when it isn't practical to calculate the derivative. It is a slower method because it requires the repeated use of the [SOLVE] key. On the other hand, you don't have to find a good value for Δ of the previous method. To find a local extreme of the function $f(x)$, define a new function

$$g(x) = f(x) - e$$

where e is a number slightly beyond the estimated extreme value of $f(x)$. If e is properly chosen, $g(x)$ will approach zero near the extreme of $f(x)$ but will not equal zero. Use [SOLVE] to analyze $g(x)$ near the extreme. The desired result is **Error 8**.

- If **Error 8** is displayed, the number in the X-register is an x value near the extreme. The number in the Z-register tells roughly how far e is from the extreme value of $f(x)$. Revise e to bring it closer (but not equal) to the extreme value. Then use [SOLVE] to examine the revised $g(x)$ near the x value previously found. Repeat this procedure until successive x values do not differ significantly.
- If a root of $g(x)$ is found, either the number e is not beyond the extreme value of $f(x)$ or else [SOLVE] has found a different region where $f(x)$ equals e. Revise e so that it is close to—but beyond—the extreme value of $f(x)$ and try [SOLVE] again. It may also be possible to modify $g(x)$ in order to eliminate the distant root.



Example: Solve the previous example without calculating the derivative of the relative field intensity E.

The subroutine to calculate E and the required constants have been entered in the previous example.

In Program mode, key in a subroutine that subtracts an estimated extreme number from the field intensity E. The extreme number should be stored in a register so that it can be manually changed as needed.

Keystrokes	Display	
[g] [P/R]	000-	Program mode.
[f] [LBL] 1	001-42,21, 1	Begins with label.
[GSB] [B]	002- 32 12	Calculates E.
[RCL] 9	003- 45 9	
[-]	004- 30	Subtracts extreme estimate.
[g] [RTN]	005- 43 32	

In Run mode, estimate the minimum intensity value by manually sampling the function.

Keystrokes	Display	
[g] [P/R]		Run mode.
10 [f] [→RAD]	0.1745	Samples the function at 10°, 30°, 50°, ...
[ENTER] [f] [B]	-0.1029	
30 [f] [→RAD]	0.5236	
[ENTER] [f] [B]	-0.2028	
50 [f] [→RAD]	0.8727	
[ENTER] [f] [B]	0.0405	

Based on these samples, try using an extreme estimate of -0.25 and initial [SOLVE] estimates (in radians) near 10° and 30°.

Keystrokes	Display	
.25 [CHS] [STO] 9	**-0.2500**	Stores extreme estimate.
.2 [ENTER]	**0.2000**	
.6	**0.6**	Initial estimates.
[f] [SOLVE] 1	**Error 8**	No root found.
[←] [STO] 4	**0.4849**	Stores θ estimate.
[R↓] [STO] 5	**0.4698**	Stores previous θ estimate.
[R↓]	**0.0457**	Distance from extreme.
.9 [×]	**0.0411**	Revises extreme estimate
[STO] [+] 9	**0.0411**	by 90 percent of the distance.
[RCL] 4	**0.4849**	Recalls θ estimate.
[ENTER] [ENTER] [f] [B]	**-0.2043**	Calculates intensity E.
[←]	**0.0000**	Recalls other θ estimate,
[RCL] 5	**0.4698**	keeping first estimate in Y-register.
[f] [SOLVE] 1	**Error 8**	No root found.
[←]	**0.4898**	θ estimate.
[x≷y]	**0.4893**	Previous θ estimate.
[x≷y]	**0.4898**	Recalls θ estimate.
[ENTER] [ENTER] [f] [B]	**-0.2043**	Calculates intensity E.
[x≷y]	**0.4898**	Recalls θ value.
[g] [→DEG]	**28.0660**	Angle in degrees.
[g] [DEG]	**28.0660**	Restores Degrees mode.

The second interation produces two θ estimates that differ in the fourth decimal place. The field intensities E for the two iterations are equal to four decimal places. Stopping at this point, a minimum field intensity of -0.2043 is indicated at an angle of 28.0660°. (This angle differs from the previous solutions by about 0.002°.)

Applications

The following applications illustrate how you can use [SOLVE] to simplify a calculation that would normally be difficult—finding an interest rate that can't be calculated directly. Other applications that use the [SOLVE] function are given in sections 3 and 4.

Annuities and Compound Amounts

This program solves a variety of financial problems involving money, time, and interest. For these problems, you normally know the values of three or four of the following variables and need to find the value of another:

n The *number* of compounding periods. (For example, a 30 year loan with monthly payments has $n = 12 \times 30 = 360$.)

i The *interest rate* per compounding period expressed as a percent. (To calculate *i*, divide the annual percentage rate by the number of compounding periods in a year. That is, 12% annual interest compounded monthly equals 1% periodic interest.)

PV The *present value* of a series of future cash flows or the initial cash flow.

PMT The periodic *payment* amount.

FV The *future value*. That is, the final cash flow (balloon payment or remaining balance) or the compounded value of a series of prior cash flows.

Possible Problems Involving Annuities and Compound Amounts

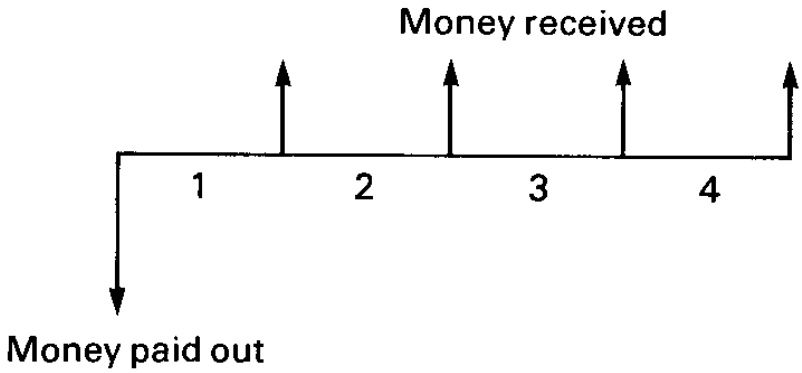
Allowable Combination of Variables	Typical Applications		Initial Procedure
	For Payments at End of Period	For Payments at Beginning of Period	
n, i, PV, PMT (Enter any three and calculate the fourth.)	Direct reduction loan. Discounted note. Mortgage.	Lease. Annuity due.	Use [f] CLEAR [REG] or set *FV* to zero.
n, i, PV, PMT, FV (Enter any four and calculate the fifth.)	Direct reduction loan with balloon payment. Discounted note.	Lease with residual value. Annuity due.	None.
n, i, PMT, FV (Enter any three and calculate the fourth.)	Sinking fund.	Periodic savings. Insurance.	Use [f] CLEAR [REG] or set *PV* to zero.
n, i, PV, FV (Enter any three and calculate the fourth.)	Compound growth. Savings.		Use [f] CLEAR [REG] or set *PMT* to zero.

The program accommodates payments that are made at the beginning or end of compounding periods. Payments made at the end of compounding periods (ordinary annuity) are common in direct reduction loans and mortgages. Payments made at the

beginning of compounding periods (annuity due) are common in leasing. For payments at the end of periods, clear flag 0. For payments at the beginning of periods, set flag 0. If the problem involves no payments, the status of flag 0 has no effect.

This program uses the convention that money paid out is entered and displayed as a negative number, and that money received is entered and displayed as a positive number.

A financial problem can usually be represented by a cash flow diagram. This is a pictorial representation of the timing and direction of financial transactions. The cash flow diagram has a horizontal time line that is divided into equal increments that correspond to the compounding period—months or years, for example. Vertical arrows represent exchanges of money, following the convention that an upward arrow (positive) represents money received and a downward arrow (negative) represents money paid out. (The examples that follow are illustrated using cash flow diagrams.)



Pressing [f] CLEAR [REG] provides a convenient way to set up the calculator for a new problem. However, it isn't necessary to press [f] CLEAR [REG] between problems. You need to reenter the values of only those variables that change from problem to problem. If a variable isn't applicable for a new problem, simply enter zero as its value. For example, if *PMT* is used in one problem but not used in the next, simply enter zero for the value of *PMT* in the second problem.

The basic equation used for the financial calculations is

$$PV + \frac{PMT\,A}{i/100}[1-(1+i/100)^{-n}] + FV(1+i/100)^{-n} = 0$$

where $i \neq 0$ and

$$A = \begin{cases} 1 & \text{for end-of-period payments} \\ 1+i/100 & \text{for beginning-of-period payments.} \end{cases}$$

The program has the following characteristics:

- [SOLVE] is used to find i. Because this is an iterative function, solving for i takes longer than finding other variables. It is possible to define problems which cannot be solved by this technique. If [SOLVE] can't find a root, **Error 4** is displayed.
- When finding any of the variables listed on the left below, certain conditions result in an **Error 4** display:

n	$PMT = -PV\,i/(100\,A)$
	$(PMT\,A - FV\,i/100)/(PMT\,A + PV\,i/100) \leqslant 0$
	$i \leqslant -100$
i	[SOLVE] can't find a root
PV	$i \leqslant -100$
PMT	$n = 0$
	$i = 0$
	$i \leqslant -100$
FV	$i \leqslant -100$

- If a problem has a specified interest rate of 0, the program generates an **Error 0** display (or **Error 4** when solving for PMT).
- Problems with extremely large (greater than 10^6) or extremely small (less than 10^{-6}) values for n and i may give invalid results.
- Interest problems with balloon payments of opposite signs to the periodic payments may have more than one mathematically correct answer (or no answer at all). This program may find one of the answers but has no way of finding or indicating other possibilities.

Keystrokes	Display	
[g] [P/R]		Program mode.
[f] CLEAR [PRGM]	000-	

Keystrokes	Display	
[f] [LBL] [A]	001-42,21,11	n routine.
[STO] 1	002- 44 1	Stores n.
[R/S]	003- 31	
[GSB] 1	004- 32 1	Calculates n.
[g] [LSTx]	005- 43 36	
[RCL] [×] 0	006-45,20, 0	
[RCL] 5	007- 45 5	
[x≷y]	008- 34	
[−]	009- 30	Calculates $FV - 100\ PMT\ A/i$.
[g] [LSTx]	010- 43 36	
[RCL] [+] 3	011-45,40, 3	Calculates $PV + 100\ PMT\ A/i$.
[g] [x=0]	012- 43 20	Tests $PMT = -PV\ i/(100\ A)$.
[GTO] 0	013- 22 0	
[÷]	014- 10	
[CHS]	015- 16	
[g] [TEST] 4	016-43,30, 4	Tests $x \leqslant 0$.
[GTO] 0	017- 22 0	
[g] [LN]	018- 43 12	
[RCL] 6	019- 45 6	
[g] [LN]	020- 43 12	
[÷]	021- 10	
[STO] 1	022- 44 1	
[g] [RTN]	023- 43 32	
[f] [LBL] [B]	024-42,21,12	i routine.
[STO] 2	025- 44 2	Stores i.
[R/S]	026- 31	
[·]	027- 48	
2	028- 2	
[ENTER]	029- 36	
[EEX]	030- 26	

Keystrokes	Display	
[CHS]	031- 16	
3	032- 3	
[g] [CF] 1	033-43, 5, 1	Clears flag 1 for [SOLVE] subroutine.
[f] [SOLVE] 3	034-42,10, 3	
[GTO] 4	035- 22 4	
[GTO] 0	036- 22 0	
[f] [LBL] 4	037-42,21, 4	
[EEX]	038- 26	
2	039- 2	
[×]	040- 20	Calculates *i*.
[STO] 2	041- 44 2	
[g] [RTN]	042- 43 32	
[f] [LBL] [C]	043-42,21,13	*PV* routine.
[STO] 3	044- 44 3	Stores *PV*.
[R/S]	045- 31	
[GSB] 1	046- 32 1	Calculates *PV*.
[GSB] 2	047- 32 2	
[CHS]	048- 16	
[STO] 3	049- 44 3	
[g] [RTN]	050- 43 32	
[f] [LBL] [D]	051-42,21,14	*PMT* routine.
[STO] 4	052- 44 4	Stores *PMT*.
[R/S]	053- 31	
1	054- 1	Calculates *PMT*.
[STO] 4	055- 44 4	
[GSB] 1	056- 32 1	
[RCL] 3	057- 45 3	
[GSB] 2	058- 32 2	
[x≷y]	059- 34	
[÷]	060- 10	
[CHS]	061- 16	
[STO] 4	062- 44 4	
[g] [RTN]	063- 43 32	

Keystrokes	Display	
[f] [LBL] [E]	064-42,21,15	*FV* routine.
[STO] 5	065- 44 5	Stores *FV*.
[R/S]	066- 31	
[GSB] 1	067- 32 1	Calculates *FV*.
[RCL] [+] 3	068-45,40, 3	
[RCL] [÷] 7	069-45,10, 7	
[CHS]	070- 16	
[STO] 5	071- 44 5	
[g] [RTN]	072- 43 32	
[f] [LBL] 1	073-42,21, 1	
[g] [SF] 1	074-43, 4, 1	Sets flag 1 for subroutine 3.
1	075- 1	
[RCL] 2	076- 45 2	
[g] [%]	077- 43 14	Calculates $i/100$.
[f] [LBL] 3	078-42,21, 3	[SOLVE] subroutine.
[STO] 8	079- 44 8	
1	080- 1	
[STO] 0	081- 44 0	
[+]	082- 40	
[g] [TEST] 4	083-43,30, 4	Tests $i \leqslant 100$.
[GTO] 0	084- 22 0	
[STO] 6	085- 44 6	
[g] [F?] 0	086-43, 6, 0	Tests for end-of-period payments.
[STO] 0	087- 44 0	
[RCL] 1	088- 45 1	
[CHS]	089- 16	
[y^x]	090- 14	Calculates $(1 + i/100)^{-n}$.
[STO] 7	091- 44 7	
1	092- 1	
[$x \gtrless y$]	093- 34	
[-]	094- 30	Calculates $1 - (1 + i/100)^{-n}$.

Keystrokes	Display	
[g] [x=0]	095- 43 20	Tests $i = 0$ or $n = 0$.
[GTO] 0	096- 22 0	
[RCL] [×] 0	097-45,20, 0	
[RCL] 4	098- 45 4	
[RCL] [÷] 8	099-45,10, 8	
[×]	100- 20	
[g] [F?] 1	101-43, 6, 1	Tests flag 1 set.
[g] [RTN]	102- 43 32	
[RCL] [+] 3	103-45,40, 3	[SOLVE] subroutine continues.
[f] [LBL] 2	104-42,21, 2	
[RCL] 5	105- 45 5	
[RCL] [×] 7	106-45,20, 7	Calculates $FV(1 + i/100)^{-n}$.
[+]	107- 40	
[g] [RTN]	108- 43 32	[SOLVE] subroutine ends.

Labels used: A, B, C, D, E, 0, 1, 2, 3, and 4.

Registers used: R_0 (A), R_1 (n), R_2 (i), R_3 (PV), R_4 (PMT), R_5 (FV), R_6, R_7, and R_8.

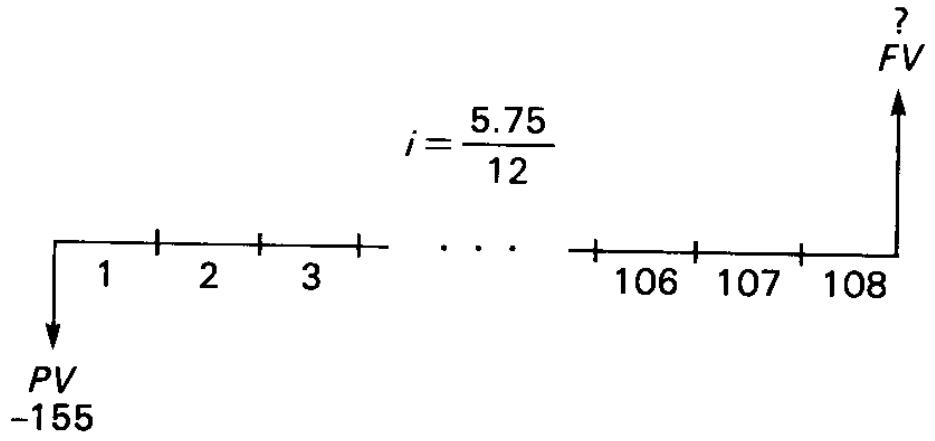
To use the program:

1. Press 8 [f] [DIM] [(i)] to reserve R_0 through R_8.
2. Press [f] [USER] to activate User mode.
3. If necessary, press [f] CLEAR [REG] to clear all of the financial variables. You don't need to clear the registers if you intend to specify all of the values.
4. Set flag 0 according to how payments are to be figured:
   - Press [g] [CF] 0 for payments at the end of the period.
   - Press [g] [SF] 0 for payments at the beginning of the period.
5. Enter the known values of the financial variables:
   - To enter n, key in the value and press [A].
   - To enter i, key in the value and press [B].

- To enter *PV*, key in the value and press [C].
- To enter *PMT*, key in the value and press [D].
- To enter *FV*, key in the value and press [E].

6. Calculate the unknown value:
   - To calculate *n*, press [A] [R/S].
   - To calculate *i*, press [B] [R/S].
   - To calculate *PV*, press [C] [R/S].
   - To calculate *PMT*, press [D] [R/S].
   - To calculate *FV*, press [E] [R/S].
7. To solve another problem, repeat steps 3 through 6 as needed. Be sure that any variable not to be used in the problem has a value of zero.

Example: You place $155 in a savings account paying 5¾% compounded monthly. What sum of money can you withdraw at the end of 9 years?



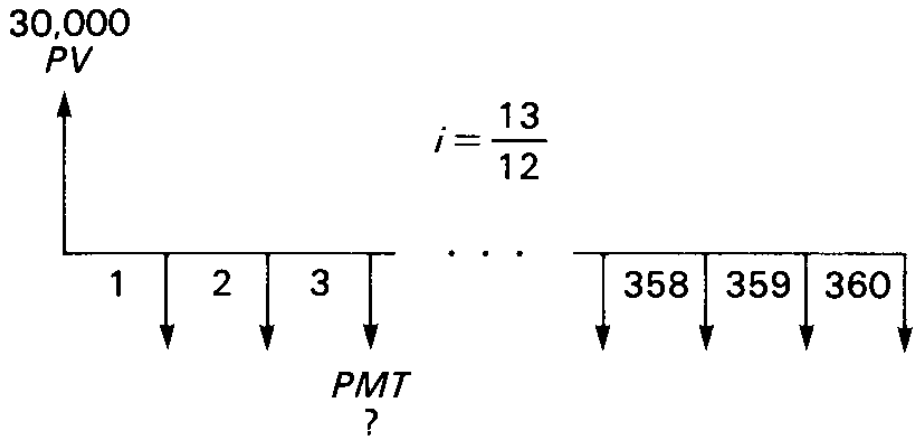
Keystrokes	Display	
[g] [P/R]		Run mode.
[f] CLEAR [REG]		Clears financial variables.
[f] [FIX] 2		
[f] [USER]		Activates User mode.
[g] [CF] 0		Ordinary annuity.
9 [ENTER] 12 [×] [A]	**108.00**	Enters $n = 9 \times 12$.

Keystrokes	Display	
5.75 [ENTER] 12 [÷] [B]	**0.48**	Enters $i = 5.75/12$.
155 [CHS] [C]	**-155.00**	Enters $PV = -155$ (money paid out).
[E] [R/S]	**259.74**	Calculates FV.

If you desire a sum of $275, what would be the required interest rate?

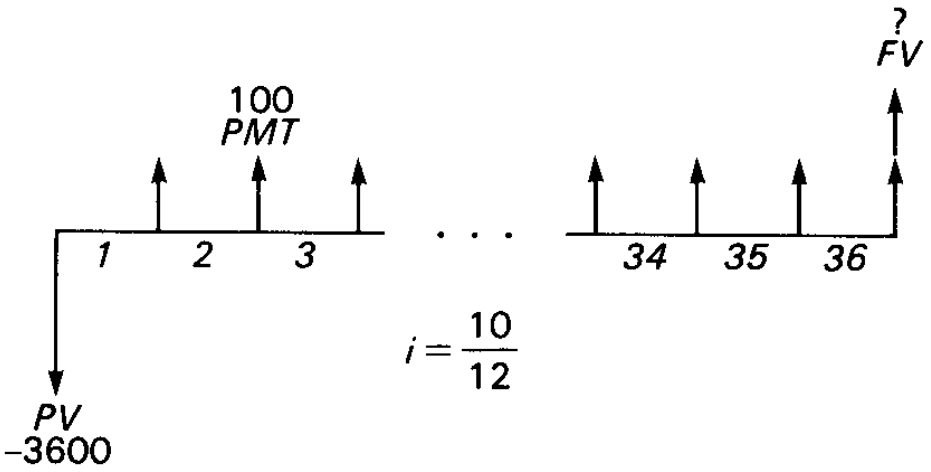
Keystrokes	Display	
275 [E]	**275.00**	Enters $FV = 275$.
[B] [R/S]	**0.53**	Calculates i.
12 [×]	**6.39**	Calculates annual interest rate.

Example: You receive $30,000 from the bank as a 30-year, 13% mortgage. What monthly payment must you make to the bank to fully amortize the mortgage?



Keystrokes	Display	
[f] CLEAR [REG]		Clears financial variables.
30 [ENTER] 12 [×] [A]	**360.00**	Enters $n = 30 \times 12$.
13 [ENTER] 12 [÷] [B]	**1.08**	Enters $i = 13/12$.
30000 [C]	**30,000.00**	Enters $PV = 30{,}000$.
[D] [R/S]	**-331.86**	Calculates PMT (money paid out).

Example: You offer a loan of $3,600 that is to be repaid in 36 monthly payments of $100 with an annual interest rate of 10%. What balloon payment amount, to be paid coincident with the 36th payment, is required to pay off the loan?



Keystrokes	Display	
[f] CLEAR [REG]		Clears financial variables.
36 [A]	**36.00**	Enters $n = 36$.
10 [ENTER] 12 [÷] [B]	**0.83**	Enters $i = 10/12$.
3600 [CHS] [C]	**-3600.00**	Enters $PV = -3600$ (money paid out).
100 [D]	**100.00**	Enters $PMT = 100$ (money received).
[E] [R/S]	**675.27**	Calculates FV.

The final payment is $675.27 + $100.00 = $775.27 because the final payment and balloon payment are due at end of the last period.

Example: You're collecting a $50,000 loan at 14% annual interest over 360 months. Find the remaining balance after the 24th payment and the interest accrued between the 12th and 24th payments.

You can use the program to calculate accumulated interest and the remaining balance for loans. The accumulated interest is equal to the total payments made during that time less the principal reduction during that time. The principal reduction is the difference between the remaining balances at the start and end of the period.

First, calculate the payment on the loan.

Keystrokes	Display	
[f] CLEAR [REG]		Clears financial variables.
360 [A]	**360.00**	Enter $n = 360$.
14 [ENTER] 12 [÷] [B]	**1.17**	Enters $i = 14/12$.
50000 [CHS] [C]	**-50,000.00**	Enters $PV = -50{,}000$.
[D] [R/S]	**592.44**	Calculates PMT.

Now calculate the remaining balance at month 24.

Keystrokes	Display	
24 [A]	**24.00**	Enters $n = 24$.
[E] [R/S]	**49,749.56**	Calculates FV at month 24.

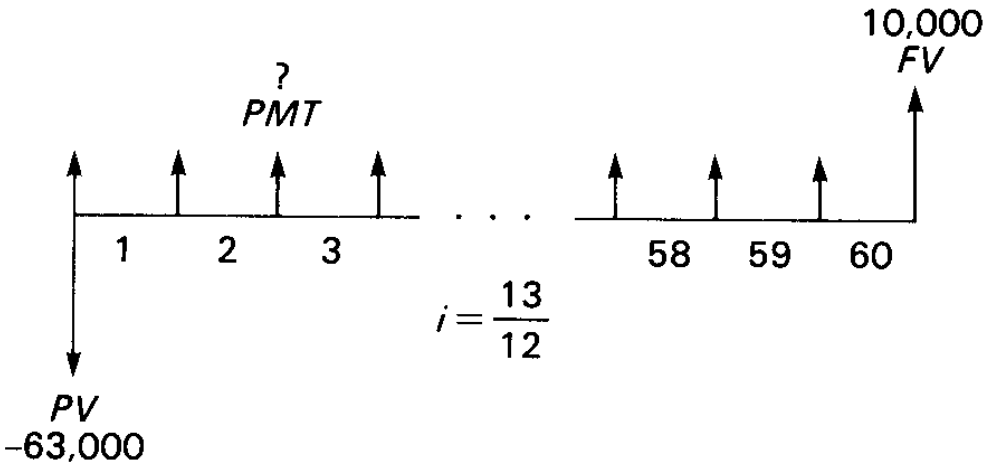
Store this remaining balance, then calculate the remaining balance at month 12 and the principal reduction between payments 12 and 24.

Keystrokes	Display	
[STO] [I]	**49,749.56**	
12 [A]	**12.00**	Enters $n = 12$.
[E] [R/S]	**49,883.48**	Calculates FV at month 12.
[RCL] [I]	**49,749.56**	Recalls FV at month 24.
[-]	**133.92**	Calculates principal reduction.

The accrued interest is the value of 12 payments less the principal reduction.

Keystrokes	Display	
[RCL] 4	**592.44**	Recalls PMT.
12 [×]	**7,109.23**	Calculates value of payments.
[x≷y] [-]	**6,975.31**	Calculates accrued interest.

Example: A leasing firm is considering the purchase of a minicomputer for $63,000 and wants to achieve a 13% annual yield by leasing the computer for a 5-year period. At the end of the lease the firm expects to sell the computer for at least $10,000. What monthly payment should the firm charge in order to achieve a 13% yield? (Because the lease payments are due at the beginning of each month, be sure to set flag 0 to specify beginning-of-period payments.)



Keystrokes	Display	
[f] CLEAR [REG]		Clears financial variables.
[g] [SF] 0		Specifies beginning-of-period payments.
5 [ENTER] 12 [×] [A]	**60.00**	Enters $n = 5 \times 12$.
13 [ENTER] 12 [÷] [B]	**1.08**	Enters $i = 13/12$.
63000 [CHS] [C]	**-63,000.00**	Enters $PV = -63{,}000$.
10000 [E]	**10,000.00**	Enters $FV = 10{,}000$.
[D] [R/S]	**1,300.16**	Calculates PMT.

If the price of the computer increases to $70,000, what should the payments be?

Keystrokes	Display	
70000 [CHS] [C]	**-70,000.00**	Enters $PV = -70{,}000$.
[D] [R/S]	**1,457.73**	Calculates PMT.

If the payments were increased to $1,500, what would the yield be?

Keystrokes	Display	
1500 [D]	**1,500.00**	Enters $PMT = 1500$.
[B] [R/S]	**1.18**	Calculates i (monthly).
12 [×]	**14.12**	Calculates annual yield.
[f] [USER]	**14.12**	Deactivates User mode.

Discounted Cash Flow Analysis

This program performs two kinds of discounted cash flow analysis: net present value (*NPV*) and internal rate of return (*IRR*). It calculates *NPV* or *IRR* for up to 24 groups of cash flows.

The cash flows are stored in the two-column matrix **C**. Matrix **C** has one row for each group of cash flows. In each row of **C**, the first element is the cash flow amount; the second element is the number of consecutive cash flows having that amount (the number of flows in that group.) The first element of **C** must be the amount of the initial investment. The cash flows must occur at equal intervals; if no cash flow occurs for several time periods, enter 0 for the cash flow amount and the number of zero cash flows in that group.

After all the cash flows have been stored in matrix **C**, you can enter an assumed interest rate and calculate the net present value (*NPV*) of the investment. Alternatively, you can calculate the internal rate of return (*IRR*). The *IRR* is the interest rate that makes the present value of a series of cash flows equal to the initial investment. It's the interest rate that makes the *NPV* equal zero. *IRR* is also called the *yield* or *discounted rate of return.*

The fundamental equation for *NPV* is

$$NPV = \begin{cases} \displaystyle\sum_{j=1}^{k} CF_j \left(\frac{1-(1+i/100)^{-n_j}}{i/100} \right) (1+i/100)^{-\sum_{l<j} n_l} & \text{for } i > -100,\ i \neq 0 \\ \displaystyle\sum_{j=1}^{k} CF_j n_j & \text{for } i = 0 \end{cases}$$

where $\sum_{l<1} n_l$ is defined as -1.

The program uses the convention that money received is entered and displayed as a positive number, and that money paid out is entered and displayed as a negative number.

The program has the following characteristics:

- The cash flow sequence (including the initial investment) must contain both a positive flow and a negative flow. That is, there must be at least one sign change.
- Cash flows with multiple sign changes may have more than one solution. This program may find one solution, but it has no way of indicating other possibilities.
- The *IRR* calculation may take several minutes (5 or more) depending of the number of cash flow entries.
- The program displays **Error 4** if it is unable to find a solution for *IRR* or if the yield $i \leqslant -100\%$ in the *NPV* calculation.

Keystrokes	Display	
[g] [P/R]		Program mode.
[f] CLEAR [PRGM]	000-	
[f] [LBL] [A]	001-42,21,11	*NPV* routine.
[EEX]	002- 26	
2	003- 2	
[÷]	004- 10	Calculates *IRR*/100.
[GSB] 2	005- 32 2	
[R/S]	006- 31	
[f] [LBL] [B]	007-42,21,12	*IRR* routine.
1	008- 1	
[ENTER]	009- 36	
[EEX]	010- 26	
[CHS]	011- 16	
3	012- 3	
[f] [SOLVE] 2	013-42,10, 2	
[GTO] 1	014- 22 1	
[GTO] 0	015- 22 0	Branch for no *IRR* solution.
[f] [LBL] 1	016-42,21, 1	
[EEX]	017- 26	

Keystrokes	Display	
2	018- 2	
[x]	019- 20	
[R/S]	020- 31	
[f] [LBL] 2	021-42,21, 2	Calculates NPV.
[g] [CF] 0	022-43, 5, 0	
[STO] 2	023- 44 2	
1	024- 1	
[STO] 4	025- 44 4	
[+]	026- 40	Calculates $1 + IRR/100$.
[g] [TEST] 4	027-43,30, 4	Tests $IRR \leqslant -100$.
[GTO] 0	028- 22 0	Branch for $IRR \leqslant -100$.
[STO] 3	029- 44 3	
0	030- 0	
[STO] 5	031- 44 5	
[f] [MATRIX] 1	032-42,16, 1	
[f] [LBL] 3	033-42,21, 3	
[g] [F?] 0	034-43, 6, 0	Tests if all flows used.
[GTO] 7	035- 22 7	Branch for all flows used.
[GSB] 6	036- 32 6	
[RCL] 2	037- 45 2	
[g] [x=0]	038- 43 20	Tests $IRR = 0$.
[GTO] 4	039- 22 4	Branch for $IRR = 0$.
1	040- 1	
[+]	041- 40	
[GSB] 6	042- 32 6	
[CHS]	043- 16	
[y^x]	044- 14	
[STO] 4	045- 44 4	
1	046- 1	
[$x \gtrless y$]	047- 34	
[-]	048- 30	
[RCL] [÷] 2	049-45,10, 2	
[RCL] [×] 3	050-45,20, 3	
[GTO] 5	051- 22 5	
[f] [LBL] 4	052-42,21, 4	
[$x \gtrless y$]	053- 34	
[GSB] 6	054- 32 6	
[f] [LBL] 5	055-42,21, 5	

Keystrokes	Display	
[×]	056- 20	
[STO] [+] 5	057-44,40, 5	
[RCL] 4	058- 45 4	
[STO] [×] 3	059-44,20, 3	
[GTO] 3	060- 22 3	
[f] [LBL] 6	061-42,21, 6	Recalls cash flow element.
[f] [USER] [RCL] [C]	062u 45 13	
[f] [USER]		
[g] [RTN]	063- 43 32	
[g] [SF] 0	064-43, 4, 0	Sets flag 0 if last element.
[g] [RTN]	065- 43 32	
[f] [LBL] 7	066-42,21, 7	
[RCL] 5	067- 45 5	Recalls *NPV*.
[g] [RTN]	068- 43 32	

Labels used: A, B, and 0 through 7.

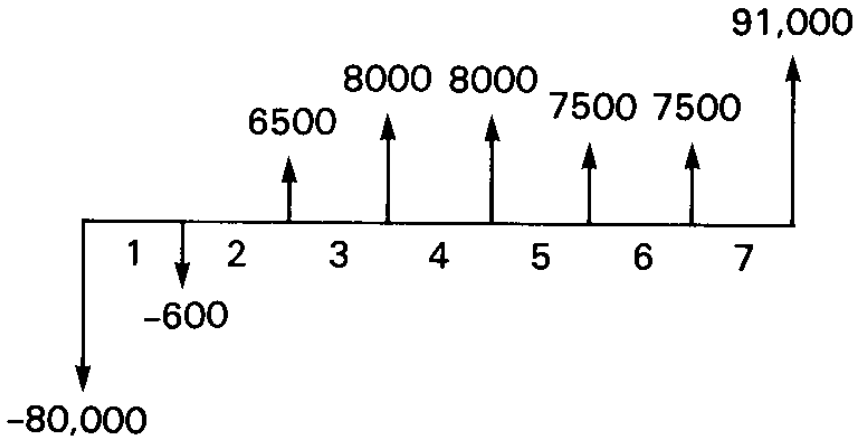
Registers used: R_0 through R_5.

Matrix used: **C**.

To use the discounted cash flow analysis program:

1. Press 5 [f] [DIM] [(i)] to allocate registers R_0 through R_5.
2. Press [f] [USER] to activate User mode (unless it's already active).
3. Key in the number of cash flow groups, then press [ENTER] 2 [f] [DIM] [C] to dimension matrix **C**.
4. Press [f] [MATRIX] 1 to set the row and column numbers to 1.
5. For each cash flow group:
   a. Key in the amount and press [STO] [C], then
   b. Key in the number of occurrences and press [STO] [C].
6. Calculate the desired parameter:
   - To calculate *IRR*, press [B].
   - To calculate *NPV*, enter periodic interest rate *i* in percent and press [A]. Repeat for as many interest rates as needed.
7. Repeat steps 3 through 6 for other sets of cash flows.

Example: An investor pays $80,000 for a duplex that he intends to sell after 7 years. He must spend some money the first year for repairs. At the end of the seventh year the duplex is sold for $91,000. Will he achieve a desired 9% after-tax yield with the following after-tax cash flows?



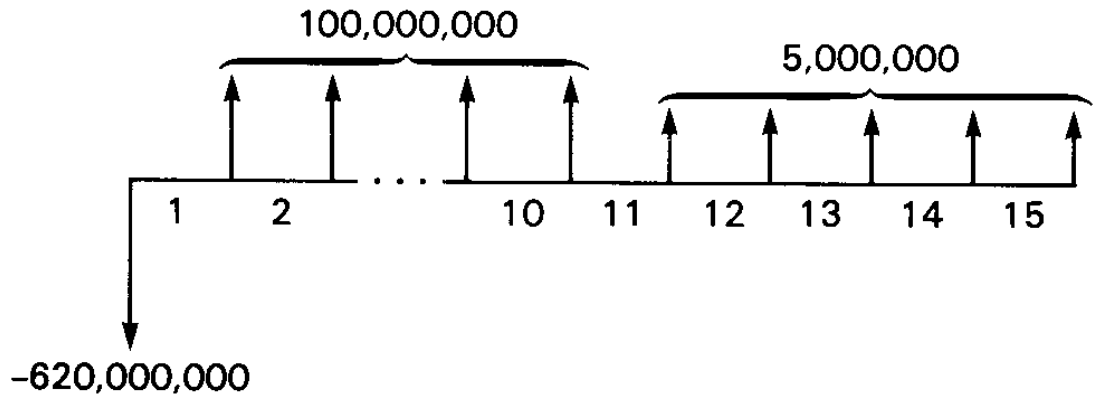
Keystrokes	Display	
[g] [P/R]		Run mode.
[f] [FIX] 2		
5 [f] [DIM] [(i)]	**5.00**	Reserve registers R_0 through R_5.
6 [ENTER] 2	**2**	
[f] [DIM] [C]	**2.00**	
[f] [MATRIX] 1	**2.00**	
[f] [USER]	**2.00**	
80000 [CHS] [STO] [C]	**-80,000.00**	Initial investment.
1 [STO] [C]	**1.00**	
600 [CHS] [STO] [C]	**-600.00**	
1 [STO] [C]	**1.00**	
6500 [STO] [C]	**6,500.00**	
1 [STO] [C]	**1.00**	
8000 [STO] [C]	**8,000.00**	
2 [STO] [C]	**2.00**	
7500 [STO] [C]	**7,500.00**	
2 [STO] [C]	**2.00**	
91000 [STO] [C]	**91,000.00**	
1 [STO] [C]	**1.00**	
9	**9**	Enters assumed yield.
[A]	**-4,108.06**	*NPV*.

Since the *NPV* is negative, the investment does not achieve the desired 9% yield. Calculate the *IRR*.

Keystrokes	Display	
[B]	**8.04**	*IRR* (after about 8 minutes).

The *IRR* is less than the desired 9% yield.

Example: An investment of $620,000,000 is expected to have an annual income stream for the next 15 years as shown in the diagram.



What is the expected rate of return?

Keystrokes	Display	
3 [ENTER] 2	**2**	
[f] [DIM] [C]	**2.00**	
[f] [MATRIX] 1	**2.00**	
620000000 [CHS]	**-620,000,000**	
[STO] [C]	**-620,000,000.0**	
1 [STO] [C]	**1.00**	
100000000 [STO] [C]	**100,000,000.0**	
10 [STO] [C]	**10.00**	
5000000 [STO] [C]	**5,000,000.00**	
5 [STO] [C]	**5.00**	
[B]	**10.06**	*IRR*.
[f] [FIX] 4	**10.0649**	
[f] [USER]	**10.0649**	Deactivates User mode.

Section 2

Working With $\int_y^x$

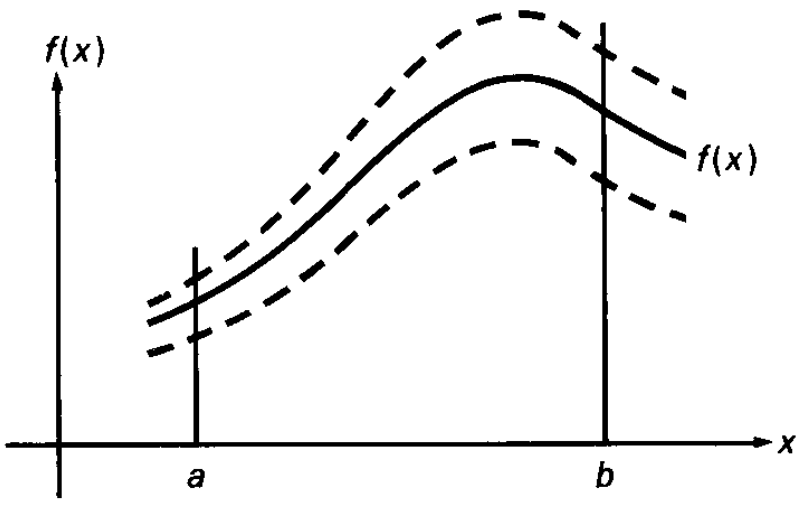
The HP-15C gives you the ability to perform numerical integration using $\int_y^x$. This section shows you how to use $\int_y^x$ effectively and describes techniques that enable you to handle difficult integrals.

Numerical Integration Using $\int_y^x$

A calculator using numerical integration can almost never calculate an integral precisely. But the $\int_y^x$ function asks you in a convenient way to specify how much error is tolerable. It asks you to set the display format according to how many figures are accurate in the integrand $f(x)$. In effect, you specify the width of a ribbon drawn around the graph of $f(x)$. The integral estimated by $\int_y^x$ corresponds to the area under some unspecified graph lying entirely within the ribbon. Of course, this estimate could vary by as much as the area of the ribbon, so $\int_y^x$ estimates this area too. If I is the desired integral, then

$$I = \begin{pmatrix}\text{area under a graph}\\ \text{drawn in the ribbon}\end{pmatrix} \pm \begin{pmatrix}\text{½ area of}\\ \text{the ribbon}\end{pmatrix}$$

The HP-15C places the first area estimate in the X-register and the second—the uncertainty—in the Y-register.



For example, $f(x)$ might represent a physical effect whose magnitude can be determined only to within $\pm$ 0.005. Then the value calculated as $f(x)$ has an uncertainty of 0.005. A display setting of [FIX] 2 tells the calculator that decimal digits beyond the second can't matter. The calculator need not waste time estimating the integral with unwarranted precision. Instead, the calculator can more quickly give you a fair idea of the range of values within which the integral must lie.

The HP-15C doesn't prevent you from declaring that $f(x)$ is far more accurate than it really is. You can specify the display setting after a careful error analysis, or you can just offer a guess. You may leave the display set to [SCI] 4 or [FIX] 4 without much further thought. You will get an estimate of the integral and its uncertainty, enabling you to interpret the result more intelligently than if you got the answer with no idea of its accuracy or inaccuracy.

The [∫] algorithm uses a Romberg method for accumulating the value of the integral. Several refinements make it more effective.

Instead of using uniformly spaced samples, which can induce a kind of resonance or aliasing that produces misleading results when the integrand is periodic, [∫] uses samples that are spaced nonuniformly. Their spacing can be demonstrated by substituting, say,

$$x = \frac{3}{2}u - \frac{1}{2}u^3$$

into

$$I = \int_{-1}^{1} f(x)dx = \int_{-1}^{1} f\left(\frac{3}{2}u - \frac{1}{2}u^3\right)\frac{3}{2}\ (1-u^2)\ du$$

and sampling u uniformly. Besides suppressing resonance, the substitution has two more benefits. First, no sample need be drawn from either end of the interval of integration (except when the interval is so narrow that no other possibilities are available). As a result, an integral like

$$\int_0^3 \frac{\sin x}{x}\,dx$$

won't be interrupted by division by zero at an endpoint. Second, ∫f can integrate functions that behave like $\sqrt{|x-a|}$, whose slope is infinite at an endpoint. Such functions are encountered when calculating the area enclosed by a smooth, closed curve.

Another refinement is that ∫f uses extended precision, 13 significant digits, to accumulate the internal sums. This allows thousands of samples to be accumulated, if necessary, without losing to roundoff any more information than is lost within your function subroutine.

Accuracy of the Function to be Integrated

The accuracy of an integral calculated using ∫f depends on the accuracy of the function calculated by your subroutine. This accuracy, which you specify using the display format, depends primarily on three considerations:

- The accuracy of empirical constants in the function.
- The degree to which the function may accurately describe a physical situation.
- The extent of round-off error in the internal calculations of the calculator.

Functions Related to Physical Situations

Functions like $\cos(4\theta - \sin\theta)$ are *pure mathematical functions*. In this context, this means that the functions do not contain any empirical constants, and neither the variables nor the limits of integration represent actual physical quantities. For such functions, you can specify as many digits as you want in the display format (up to nine) to achieve the desired degree of accuracy in the integral.* All you need to consider is the trade-off between the accuracy and calculation time.

---

* Provided that $f(x)$ is still calculated accurately, despite round-off error, to the number of digits shown in the display.

There are additional considerations, however, when you're integrating functions relating to an actual physical situation. Basically, with such functions you should ask yourself *whether the accuracy you would like in the integral is justified by the accuracy in the function.* For example, if the function contains empirical constants that are specified to only, say, three significant digits, it might not make sense to specify more than three digits in the display format.

Another important consideration—and one which is more subtle and therefore more easily overlooked—is that nearly every function relating to a physical situation *is inherently inaccurate to a certain degree*, because it is only a mathematical *model* of an actual process or event. A mathematical model is itself an *approximation* that ignores the effects of known or unknown factors which are insignificant to the degree that the results are still useful.

An example of a mathematical model is the *normal distribution function*

$$\int_{-\infty}^{t} \frac{e^{-(x-\mu)^2/2\sigma^2}}{\sigma\sqrt{2\pi}}\,dx$$

which has been found to be useful in deriving information concerning physical measurements on living organisms, product dimensions, average temperatures, etc. Such mathematical descriptions typically are either derived from theoretical considerations or inferred from experimental data. To be practially useful, they are constructed with certain assumptions, such as ignoring the effects of relatively insignificant factors. For example, the accuracy of results obtained using the normal distribution function as a model of the distribution of certain quantities depends on the size of the population being studied. And the accuracy of results obtained from the equation $s = s_0 - \frac{1}{2}gt^2$, which gives the height of a falling body, ignores the variation with altitude of g, the acceleration of gravity.

Thus, mathematical descriptions of the physical world can provide results of only limited accuracy. If you calculated an integral with an apparent accuracy beyond that with which the model describes

the actual behavior of the process or event, you would not be justified in using the calculated value to the full apparent accuracy.

Round-Off Error in Internal Calculations

With any computational device—including the HP-15C—calculated results must be "rounded off" to a finite number of digits (10 digits in the HP-15C). Because of this *round-off error*, calculated results—especially results of evaluating a function that contains several mathematical operations—may not be accurate to all 10 digits that can be displayed. Note that round-off error affects the evaluation of *any* mathematical expression, not just the evaluation of a function to be integrated using ∫f. (Refer to the appendix for additional information.)

If $f(x)$ is a function relating to a physical situation, its inaccuracy due to round-off typically is insignificant compared to the inaccuracy due to empirical constants, etc. If $f(x)$ is what we have called a pure mathematical function, its accuracy is limited only by round-off error. Generally, it would require a complicated analysis to determine precisely how many digits of a calculated function might be affected by round-off. In practice, its effects are typically (and adequately) determined through experience rather than analysis.

In certain situations, round-off error can cause peculiar results, particularly if you should compare the results of calculating integrals that are equivalent mathematically but differ by a transformation of variables. However, you are unlikely to encounter such situations in typical applications.

Shortening Calculation Time

The time required for ∫f to calculate an integral depends on how soon a certain density of sample points is achieved in the region where the function is interesting. The calculation of the integral of any function will be prolonged if the interval of integration includes mostly regions where the function is not interesting. Fortunately, if you must calculate such an integral, you can modify the problem so that the calculation time is reduced. Two such techniques are subdividing the interval of integration and transformation of variables.

Subdividing the Interval of Integration

In regions where the slope of $f(x)$ is varying appreciably, a high density of sample points is necessary to provide an approximation that changes insignificantly from one iteration to the next. However, in regions where the slope of the function stays nearly constant, a high density of sample points is not necessary. This is because evaluating the function at additional sample points would not yield much new information about the function, so it would not dramatically affect the disparity between successive approximations. Consequently, in such regions an approximation of comparable accuracy could be achieved with substantially fewer sample points: so much of the time spent evaluating the function in these regions is wasted. When integrating such functions, you can save time by using the following procedure:

1. Divide the interval of integration into subintervals over which the function is interesting and subintervals over which the function is uninteresting.
2. Over the subintervals where the function is interesting, calculate the integral in the display format corresponding to the accuracy you would like overall.
3. Over the subintervals where the function either is not interesting or contributes negligibly to the integral, calculate the integral with less accuracy, that is, in a display format specifying fewer digits.
4. To get the integral over the entire interval of integration, add together the approximations and their uncertainties from the integrals calculated over each subinterval. You can do this easily using the [Σ+] key.

Before subdividing the integration, check whether the calculator underflows when evaluating the function around the upper (or lower) limit of integration.* Since there is no reason to evaluate the function at values of x for which the calculator underflows, in some cases the upper limit of integration can be reduced, saving considerable calculation time.

---

*When the calculation of any quantity would result in a number less than 10^{-99}, the result is replaced by zero. This condition is known as underflow.

Remember that once you have keyed in the subroutine that evaluates $f(x)$, you can calculate $f(x)$ for any value of x by keying that value into the X-register and pressing [ENTER] [ENTER] [ENTER] [GSB] followed by the label of the subroutine.

If the calculator underflows at the upper limit of integration, try smaller numbers until you get closer to the point where the calculator no longer underflows.

For example, consider the approximation of

$$\int_0^{\infty} xe^{-x}dx \ .$$

Key in a subroutine that evaluates the function $f(x) = xe^{-x}$.

Keystrokes	Display	
[g] [P/R]		Program mode.
[f] CLEAR [PRGM]	000–	Clears program memory.
[f] [LBL] 1	001–42,21, 1	
[CHS]	002– 16	
[e^x]	003– 12	
[×]	004– 20	
[g] [RTN]	005– 43 32	

Set the calculator to Run mode and set the display format to [SCI] 3. They try several values of x to find where the calculator underflows for your function.

Keystrokes	Display		
[g] [P/R]			Run mode.
[f] [SCI] 3			Sets format to [SCI] 3.
[EEX] 3	1	03	Keys 1000 into X-register.
[ENTER] [ENTER] [ENTER]	1.000	03	Fills the stack with x.
[GSB] 1	0.000	00	Calculator underflows at $x = 1000$.
300 [ENTER]	3.000	02	Tries a smaller value of x.
[ENTER] [ENTER]	3.000	02	
[GSB] 1	0.000	00	Calculator still underflows.
200 [ENTER]	2.000	02	Try a smaller value of x.

Keystrokes	Display		
[ENTER] [ENTER]	2.000	02	
[GSB] 1	2.768	-85	Calculator doesn't underflow at $x = 200$; try a number between 200 and 250.
225 [ENTER]	2.250	02	
[ENTER] [ENTER]	2.250	02	
[GSB] 1	4.324	-96	Calculator is close to underflow.

At this point, you can use [SOLVE] to pinpoint the smallest value of x at which the calculator underflows.

Keystrokes	Display		
[R↓]	2.250	02	Roll down stack until the last value tried is in the X- and Y-registers.
[f] [SOLVE] 1	2.280	02	The minimum value of x at which the calculator underflows is about 228.

You've now determined that you need integrate only from 0 to 228. Since the integrand is interesting only for values of x less than 10, divide the interval of integration there. The problem has now become:

$$\int_0^{\infty} xe^{-x}dx \approx \int_0^{228} xe^{-x}dx = \int_0^{10} xe^{-x}dx + \int_{10}^{228} xe^{-x}dx\,.$$

Keystrokes	Display		
7 [f] [DIM] [(i)]	7.000	00	Allocates statistical storage registers.
[f] CLEAR [Σ+]	0.000	00	Clears statistical storage registers.
0 [ENTER]	0.000	00	Keys in lower limit of integration over first subinterval.
10	10		Keys in upper limit of integration over first subinterval.

Keystrokes	Display		
f ∫ 1	9.995	-01	Integral over (0, 10) calculated in SCI 3.
Σ+	1.000	00	Sum approximation and its uncertainty in registers R_3 and R_5.
x≷y	1.841	-04	Uncertainty of approximation.
R↓ R↓	1.000	01	Roll down stack until upper limit of first integral appears in X-register.
228	228		Keys upper limit of second integral into X-register. Upper limit of first integral is lifted into Y-register, becoming lower limit of second integral.
f SCI 0	2.	02	Specifies SCI 0 display format for a quick calculation over (10, 228). If the uncertainty of the approximation turns out not to be accurate enough, you can repeat the approximation in a display format specifying more digits.
f ∫ 1	5.	-04	Integral over (10, 228) calculated in SCI 0.
f SCI 3	5.328	-04	Changes display format back to SCI 3.
x≷y	7.568	-05	Checks uncertainty of approximation. Since it is less than the uncertainty of the approximation over the first subinterval, SCI 0 yielded an approximation of sufficient accuracy.

Keystrokes	Display		
[x≷y]	5.328	-04	Returns approximation and its uncertainty to the X- and Y-registers, respectively, before summing them in statistical storage registers.
[Σ+]	2.000	00	Sums approximation and its uncertainty.
[RCL] [Σ+]	1.000	00	Integral over total interval (0, 228) (recalled from R_3).
[x≷y]	2.598	-04	Uncertainty of integral (from R_5).

Transformation of Variables

In many problems where the function changes very slowly over most of a very wide interval of integration, a suitable transformation of variables may decrease the time required to calculate the integral.

For example, consider again the integral

$$\int_0^{\infty} xe^{-x}dx\,.$$

Let $e^{-x} = u^3.$

Then $x = -3 \ln u$

and $dx = -3\frac{du}{u}.$

Substituting,

$$\int_0^{\infty} xe^{-x}dx = \int_{e^{-0}}^{e^{-\infty}} (-3 \ln u)(u^3)\left(-3\frac{du}{u}\right)$$

$$= \int_1^0 9u^2 \ln u\, du.$$

Key in a subroutine that evaluates the function $f(u) = 9u^2 \ln u$.

Keystrokes	Display	
g P/R	000-	Program mode.
f LBL 3	001-42,21, 3	
g LN	002- 43 12	
x≷y	003- 34	
g x²	004- 43 11	
×	005- 20	
9	006- 9	
×	007- 20	
g RTN	008- 43 32	

Key in the limits of integration, then press f ∫ 3 to calculate the integral.

Keystrokes	Display		
g P/R			Run mode.
1 ENTER	1.000	00	Keys in lower limit of integration.
0	0		Keys in upper limit of integration.
f ∫ 3	1.000	00	Approximation to equivalent integral.
x≷y	3.020	-04	Uncertainty of approximation.

The approximation agrees with the value calculated in the previous problem for the same integral.

Evaluating Difficult Integrals

Certain conditions can prolong the time required to evaluate an integral or can cause inaccurate results. As discussed in the *HP-15C Owner's Handbook*, these conditions are related to the nature of the integrand over the interval of integration.

One class of integrals that are difficult to calculate is improper integrals. An improper integral is one that involves ∞ in at least one of the following ways:

- One or both limits of integration are $\pm\infty$, such as

$$\int_{-\infty}^{\infty} e^{-u^2} du = \sqrt{\pi}\ .$$

- The integrand tends to $\pm\infty$ someplace in the range of integration, such as

$$\int_0^1 \ln(u)\, du = 1.$$

- The integrand oscillates infinitely rapidly somewhere in the range of integration, such as

$$\int_0^1 \cos(\ln u)\, du = \tfrac{1}{2}.$$

Equally troublesome are nearly improper integrals, which are characterized by

- The integrand or its first derivative changes wildly within a relatively narrow subinterval of the range of integration, or oscillates frequently across that range.

The HP-15C attempts to deal with certain of the second type of improper integral by usually not sampling the integrand at the limits of integration.

Because improper and nearly improper integrals are not uncommon in practice, you should recognize them and take measures to evaluate them accurately. The following examples illustrate techniques that are helpful.

Consider the integrand

$$f(x) = \frac{\sqrt{-2 \ln \cos(x^2)}}{x^2}.$$

This function loses its accuracy when x becomes small. This is caused by rounding $\cos(x^2)$ to 1, which drops information about how small x is. But by using $u = \cos(x^2)$, you can evaluate the integrand as

$$f(x) = \begin{cases} 1 & \text{if } u = 1 \\ \dfrac{\sqrt{-2 \ln u}}{\cos^{-1} u} & \text{if } u \neq 1. \end{cases}$$

Although the branch for $u = 1$ adds four steps to your subroutine, integration near $x = 0$ becomes more accurate.

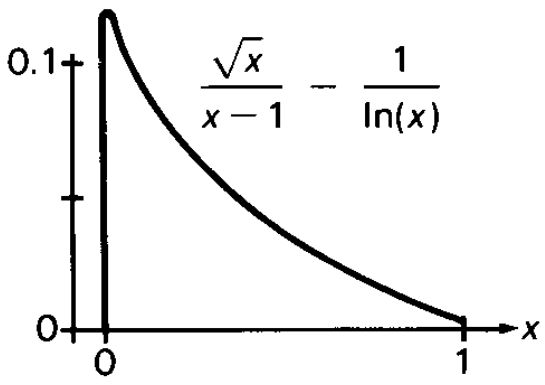
As a second example, consider the integral

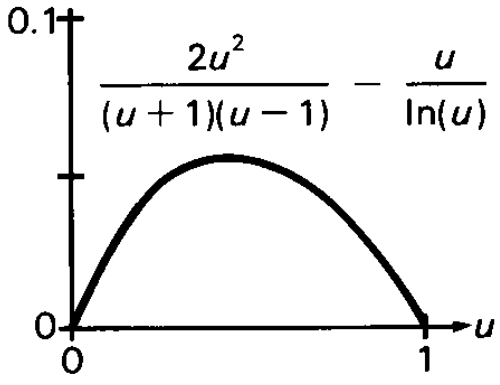
$$\int_0^1 \left(\frac{\sqrt{x}}{x-1} - \frac{1}{\ln x} \right) dx.$$

The derivative of the integrand approaches ∞ as x approaches 0, as shown in the illustration below. By substituting $x = u^2$, the function becomes more well behaved, as shown in the second illustration. This integral is easily evaluated:

$$\int_0^1 \left(\frac{2u^2}{(u+1)(u-1)} - \frac{u}{\ln u} \right) du.$$

Don't replace $(u + 1)(u - 1)$ by $(u^2 - 1)$ because as u approaches 1, the second expression loses to roundoff half of its significant digits and introduces to the integrand's graph a spike near $u = 1$.





As another example, consider a function whose graph has a long tail that stretches out many, many times as far as the main "body" (where the graph is interesting)—a function like

$$f(x) = e^{-x^2} \quad \text{or} \quad g(x) = \frac{1}{x^2 + 10^{-10}}.$$

Thin tails, like that of $f(x)$, can be truncated without greatly degrading the accuracy or speed of integration. But $g(x)$ has too wide a tail to ignore when calculating

$$\int_{-t}^{t} g(x)\,dx$$

if t is large.

For such functions, a substitution like $x = a + b \tan u$ works well, where a lies within the graph's main "body" and b is roughly its width. Doing this for $f(x)$ from above with $a = 0$ and $b = 1$ gives

$$\int_0^t f(x)\,dx = \int_0^{\tan^{-1}t} e^{-\tan^2 u}(1 + \tan^2 u)\,du,$$

which is calculated readily even with t as large as 10^{10}. Using the same substitution with $g(x)$, values near $a = 0$ and $b = 10^{-5}$ provide good results.

This example involves subdividing the interval of integration. Although a function may have features that look extreme over the entire interval of integration, over portions of that interval the function may look more well-behaved. Subdividing the interval of integration works best when combined with appropriate substitutions. Consider the integral

$$\begin{aligned}
\int_0^\infty dx/(1+x^{64}) &= \int_0^1 dx/(1+x^{64}) + \int_1^\infty dx/(1+x^{64}) \\
&= \int_0^1 dx/(1+x^{64}) + \int_0^1 u^{62}du/(u^{64}+1) \\
&= \int_0^1 (1+x^{62})dx/(1+x^{64}) \\
&= 1 + \int_0^1 (x^{62}-x^{64})dx/(1+x^{64}) \\
&= 1 + \tfrac{1}{8}\int_0^1 (1-v^{1/4})v^{55/8}\,dv/(1+v^8).
\end{aligned}$$

These steps use the substitutions $x = 1/u$ and $x = v^{1/8}$ and some algebraic manipulation. Although the original integral is improper, the last integral is easily handled by [∫]. In fact, by separating the constant term from the integral, you obtain (using [SCI] 8) an answer with 13 significant digits:

$$1.000401708155 \pm 1.2 \times 10^{-12}.$$

A final example drawn from real life involves the electrostatic field about an ellipsoidal probe with principal semiaxes a, b, and c:

$$V = \int_0^\infty \frac{dx}{(a^2+x)\sqrt{(a^2+x)(b^2+x)(c^2+x)}}$$

for $a = 100$, $b = 2$, and $c = 1$.*

Transform this improper integral to a proper one by substituting $x = (a^2 - c^2)/(1 - u^2) - a^2$:

$$V = p\int_r^1 \sqrt{(1-u^2)/(u^2+q)}\, du$$

where

$$p = 2/((a^2-c^2)\sqrt{a^2-b^2}) = 2.00060018 \times 10^{-6}$$

$$q = (b^2-c^2)/(a^2-b^2) = 3.001200480 \times 10^{-3}$$

$$r = c/a = 0.01\,.$$

However, this integral is nearly improper because q and r are both so nearly zero. But by using an integral in closed form that sufficiently resembles the troublesome part of V, the difficulty can be avoided. Try

$$W = p\int_r^1 du/\sqrt{u^2+q} = p\ln(u+\sqrt{u^2+q}\,)\Big|_r^1$$

$$= p\ln((1+\sqrt{1+q})/(r+\sqrt{r^2+q}\,))$$

$$= 8.40181880708 \times 10^{-6}.$$

Then

$$V = W + p\int_r^1 (\sqrt{(1-u^2)/(u^2+q)} - 1/\sqrt{u^2+q}\,)du$$

$$= p\int_r^1 \left(\frac{W/p}{1-r} - \frac{u^2}{(1+\sqrt{1-u^2}\,)\sqrt{u^2+q}}\right)du.$$

---

*From Stratton, J.A., *Electromagnetic Theory*, McGraw-Hill, New York, 1941, pp. 201-217.

The HP-15C readily handles this integral. Don't worry about $\sqrt{1-u^2}$ *as* u approaches 1 because the figures lost to roundoff aren't needed.

Application

The following program calculates the values of four special functions for any argument x:

$$P(x) = \frac{1}{2\pi}\int_{-\infty}^{x} e^{-t^2/2}dt$$ (normal distribution function)

$$Q(x) = 1 - P(x) = \frac{1}{2\pi}\int_{x}^{\infty} e^{-t^2/2}dt$$ (complementary normal distribution function)

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_{0}^{x} e^{-t^2}dt$$ (error function)

$$\mathrm{erfc}(x) = 1 - \mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_{x}^{\infty} e^{-t^2}dt$$ (complementary error function)

The program calculates these functions using the transformation $u = e^{-t^2}$ whenever $|x| > 1.6$.

The function value is returned in the X-register, and the uncertainty of the *integral* is returned in the Y-register. (The uncertainty of the *function* value is approximately the same order of magnitude as the number in the Y-register.) The original argument is available in register R_0.

The program has the following characteristics:

- The display format specifies the accuracy of the integrand in the same way as it does for ∫ itself. However, if you specify an unnecessarily large number of display digits, the calculation will be prolonged.
- Small function values, such as Q(20), P(-20), and erfc(10), are accurately computed as quickly as moderate values.

Keystrokes	Display	
g P/R		Program mode.
f CLEAR PRGM	000-	
f LBL A	001-42,21,11	Program for $P(x)$.
STO 2	002- 44 2	Stores x in R_2.
CHS	003- 16	Calculates $-x$.
GTO 2	004- 22 2	Branches to calculate $P(x) = Q(-x)$.
f LBL B	005-42,21,12	Program for $Q(x)$.
STO 2	006- 44 2	Stores x in R_2.
f LBL 2	007-42,21, 2	
2	008- 2	
$\sqrt{x}$	009- 11	
÷	010- 10	
GSB C	011- 32 13	Calculates $\text{erfc}(x/\sqrt{2})$.
2	012- 2	
÷	013- 10	Calculates $Q(x) = \frac{1}{2}\,\text{erfc}(x/\sqrt{2})$.
RCL 2	014- 45 2	
STO 0	015- 44 0	Stores x in R_0.
R↓	016- 33	
g RTN	017- 43 32	Returns function value.
f LBL C	018-42,21,13	Program for $\text{erfc}(x)$.
1	019- 1	
GSB 4	020- 32 4	
g F? 1	021-43, 6, 1	Tests flag 1 set.
GTO 5	022- 22 5	Branches for flag 1 set.
1	023- 1	
−	024- 30	Calculates $\text{erf}(x) - 1$ for flag 1 clear.
f LBL 5	025-42,21, 5	
CHS	026- 16	Calculates $\text{erfc}(x)$.
g RTN	027- 43 32	Returns function value.
f LBL E	028-42,21,15	Program for $\text{erf}(x)$.
0	029- 0	
f LBL 4	030-42,21, 4	Subroutine for $\text{erf}(x)$ or $\text{erfc}(x)$.
g CF 1	031-43, 5, 1	Clears flag 1.

Keystrokes	Display	
STO 1	032- 44 1	Stores 0 for erf(x), 1 for erfc(x).
$x \gtrless y$	033- 34	
STO 0	034- 44 0	
g ABS	035- 43 16	Calculates $\|x\|$.
1	036- 1	
.	037- 48	
6	038- 6	
g TEST 8	039-43,30, 8	Tests $\|x\| > 1.6$.
GTO 6	040- 22 6	Branch for $\|x\| > 1.6$.
0	041- 0	
RCL 0	042- 45 0	Recalls x.
f ∫ 0	043-42,20, 0	Integrates e^{-t^2} from 0 to x.
2	044- 2	
×	045- 20	
f LBL 3	046-42,21, 3	Subroutine to divide by $\sqrt{\pi}$.
g π	047- 43 26	
√x	048- 11	
÷	049- 10	
g RTN	050- 43 32	
f LBL 6	051-42,21, 6	Subroutine to integrate when $\|x\| > 1.6$.
g SF 1	052-43, 4, 1	Sets flag 1.
0	053- 0	
RCL 0	054- 45 0	
g x^2	055- 43 11	
CHS	056- 16	
e^x	057- 12	Calculates e^{-x^2}.
f ∫ 1	058-42,20, 1	Integrates $(-\ln u)^{-1/2}$ from 0 to e^{-x^2}.
GSB 3	059- 32 3	Divides integral by $\sqrt{\pi}$.
RCL 0	060- 45 0	
ENTER	061- 36	
g ABS	062- 43 16	
÷	063- 10	Calculates sign of x.
×	064- 20	

Keystrokes	Display	
[RCL] 1	065- 45 1	Recalls 1 for erfc(x), 0 for erf(x).
[g] [LSTx]	066- 43 36	
[-]	067- 30	
[+]	068- 40	Adjusts integral for sign of x and function.
[CHS]	069- 16	
[g] [RTN]	070- 43 32	
[f] [LBL] 0	071-42,21, 0	Subroutine to calculate e^{-t^2}.
[g] [x^2]	072- 43 11	
[CHS]	073- 16	
[e^x]	074- 12	
[g] [RTN]	075- 43 32	
[f] [LBL] 1	076-42,21, 1	Subroutine to calculate $(-\ln u)^{-1/2}$.
[g] [x=0]	077- 43 20	
[g] [RTN]	078- 43 32	
[g] [LN]	079- 43 12	
[CHS]	080- 16	
[$\sqrt{x}$]	081- 11	
[1/x]	082- 15	
[g] [RTN]	083- 43 32	

Labels used: A, B, C, E, 0, 1, 2, 3, 4, 5, and 6.

Registers used: R_0 (x), R_1, R_2.

Flag used: 1.

To use this program:

1. Enter the argument x into the display.
2. Evaluate the desired function:
   - Press [f] [A] to evaluate P(x).
   - Press [f] [B] to evaluate Q(x).
   - Press [f] [E] to evaluate erf(x).
   - Press [f] [C] to evaluate erfc(x).

Example: Calculate Q(20), P(1.234), and erf(0.5) in [SCI] 3 display format.

Keystrokes	Display		
[g] [P/R]			Run mode.
[f] [SCI] 3			Specifies format.
20 [f] [B]	**2.754**	**-89**	Q(20).
1.234 [f] [A]	**8.914**	**-01**	P(1.234).
.5 [f] [E]	**5.205**	**-01**	erf(0.5).

Example: For a Normally distributed random variable X with mean 2.151 and standard deviation 1.085, calculate the probability $Pr[2 < X \leqslant 3]$.

$$Pr[2 < X \leqslant 3] = Pr\left[\frac{2-2.151}{1.085} < \frac{X-\mu}{\sigma} \leqslant \frac{3-2.151}{1.085}\right]$$

$$= \mathrm{P}\left(\frac{3-2.151}{1.085}\right) - \mathrm{P}\left(\frac{2-2.151}{1.085}\right)$$

Keystrokes	Display		
2 [ENTER]	**2.000**	**00**	
2.151 [-]	**-1.510**	**-01**	
1.085 [÷]	**-1.392**	**-01**	
[f] [A]	**4.447**	**-01**	Calculates $Pr[X \leqslant 2]$.
[STO] 3	**4.447**	**-01**	Stores value.
3 [ENTER]	**3.000**	**00**	
2.151 [-]	**8.490**	**-01**	
1.085 [÷]	**7.825**	**-01**	
[f] [A]	**7.830**	**-01**	Calculates $Pr[X \leqslant 3]$.
[RCL] 3	**4.447**	**-01**	Recalls $Pr[X \leqslant 2]$.
[-]	**3.384**	**-01**	Calculates $Pr[2 < X \leqslant 3]$.
[f] [FIX] 4	**0.3384**		

Section 3

Calculating in Complex Mode

Physically important problems involving real data are often solved by performing relatively simple calculations using complex numbers. This section gives important insights into complex computation and shows several examples of solving problems involving complex numbers.

Using Complex Mode

Complex mode in the HP-15C enables you to evaluate complex-valued expressions simply. Generally, in Complex mode a mathematical expression is entered in the same manner as in the normal "real" mode. For example, consider a program that evaluates the polynomial $P(x) = a_n x^n + \ldots + a_1 x + a_0$ for the value x in the X-register. By activating Complex mode, this same program can evaluate $P(z)$, where z is complex. Similarly, other expressions, such as the Gamma function $\Gamma(x)$ in the next example, can be evaluated for complex arguments in Complex mode.

Example: Write a program that evaluates the continued-fraction approximation

$$\ln(\Gamma(x)) = (x - \tfrac{1}{2})\ln x - x + a_0 + \cfrac{a_1}{x + \cfrac{a_2}{x + \cfrac{a_3}{x + \ldots}}}$$

for the first six values of a:

$$\begin{aligned} a_0 &= \tfrac{1}{2}\ln(2\pi) \\ a_1 &= 1/12 \\ a_2 &= 1/30 \\ a_3 &= 53/210 \\ a_4 &= 195/371 \\ a_5 &= 1.011523068 \\ a_6 &= 1.517473649. \end{aligned}$$

Because this approximation is valid for both real arguments and complex arguments with $\mathrm{Re}(z) > 0$, this program approximates $\ln(\Gamma(z))$ in Complex mode (for sufficiently large $|z|$). When $|z| > 4$ (and $\mathrm{Re}(z) > 0$), the approximation has about 9 or 10 accurate digits.

Enter the following program.

Keystrokes	Display	
g P/R		Program mode.
f CLEAR PRGM	000-	
f LBL A	001-42,21,11	
6	002- 6	
STO I	003- 44 25	Stores counter in Index register.
$x \gtrless y$	004- 34	
ENTER	005- 36	
ENTER	006- 36	
ENTER	007- 36	Fills stack with z.
RCL 6	008- 45 6	Recalls a_6.
f LBL 1	009-42,21, 1	Loop for continued fraction.
+	010- 40	
RCL (i)	011- 45 24	Recalls a_i.
$x \gtrless y$	012- 34	Restores z.
÷	013- 10	
f DSE I	014-42, 5,25	Decrements counter.
GTO 1	015- 22 1	
RCL 0	016- 45 0	Recalls a_0.
+	017- 40	
$x \gtrless y$	018- 34	Restores z.
−	019- 30	
g LSTx	020- 43 36	Recalls z.
g LN	021- 43 12	Calculates $\ln(z)$.
g LSTx	022- 43 36	Recalls z.
.	023- 48	
5	024- 5	
−	025- 30	Calculates $z - \frac{1}{2}$.

Keystrokes	Display	
[×]	026- 20	
[+]	027- 40	Calculates $\ln(\Gamma(z))$.
[g] [RTN]	028- 43 32	

Store the constants in registers R_0 through R_6 in order according to their subscripts.

Keystrokes	Display	
[g] [P/R]		Run mode.
2 [g] [π] [×]	6.2832	
[g] [LN] 2 [÷]	0.9189	
[STO] 0	0.9189	Stores a_0.
12 [1/x] [STO] 1	0.0833	Stores a_1.
30 [1/x] [STO] 2	0.0333	Stores a_2.
53 [ENTER] 210 [÷]	0.2524	
[STO] 3	0.2524	Stores a_3.
195 [ENTER] 371 [÷]	0.5256	
[STO] 4	0.5256	Stores a_4.
1.011523068 [STO] 5	1.0115	Stores a_5.
1.517473649 [STO] 6	1.5175	Stores a_6.

Use this program to calculate $\ln(\Gamma(4.2))$, then compare it with $\ln(3.2!)$ calculated with the [x!] function. Also calculate $\ln(\Gamma(1 + 5i))$.

Keystrokes	Display	
4.2 [f] [A]	2.0486	Calculates $\ln(\Gamma(4.2))$.
[f] [FIX] 9	2.048555637	Displays 10 digits.
3.2 [f] [x!]	7.756689536	Calculates $(3.2)! = \Gamma(3.2 + 1)$.
[g] [LN]	2.048555637	Calculates $\ln(3.2!)$.
1 [ENTER]	1.000000000	Enters real part of $1 + 5i$.
5 [f] [I]	1.000000000	Forms complex number $1 + 5i$.

Keystrokes	Display	
f A	**-6.130324145**	Real part of $\ln(\Gamma(1+5i))$.
f Re≷Im	**3.815898575**	Imaginary part of $\ln(\Gamma(1+5i))$.
f FIX 4	**3.8159**	

The complex result is calculated with no more effort than that needed to enter the imaginary part of the argument z. (The result $\ln(\Gamma(1+5i))$ has 10 correct digits in each component.)

Trigonometric Modes

Although the trignometric mode annunciator remains lit in Complex mode, complex functions are *always* computed using *radian* measure. The annunciator indicates the mode (Degrees, Radians, or Grads) for only the two complex conversions: →P and →R.

If you want to evaluate $re^{i\theta}$ where θ is in degrees, e^x can't be used directly because θ must be in radians. If you attempt to convert from degrees to radians, there is a slight loss of accuracy, especially at values like 180° for which the radian measure π can't be represented exactly with 10 digits.

However, in Complex mode the →R function computes $re^{i\theta}$ accurately for θ in *any* measure (indicated by the annunciator). Simply enter r and θ into the complex X-registers in the form $r + i\theta$, then execute →R to calculate the complex value

$$re^{i\theta} = r\cos\theta + ir\sin\theta.$$

(The program listed under Calculating the nth Roots of a Complex Number at the end of this section uses this function.)

Definitions of Math Functions

The lists that follow define the operation of the HP-15C in Complex mode. In these definitions, a complex number is denoted by $z = x + iy$ (rectangular form) or $z = re^{i\theta}$ (polar form). Also $|z| = \sqrt{x^2 + y^2}$.

Arithmetic Operations

$$(a+ib)\pm(c+id)=(a\pm c)+i(b\pm d)$$

$$(a+ib)(c+id)=(ac-bd)+i(ad+bc)$$

$$z^2=z\times z$$

$$1/z=x/|z|^2-iy/|z|^2$$

$$z_1\div z_2=z_1\times 1/z_2$$

Single-Valued Functions

$$e^z=e^x(\cos y+i\sin y)$$

$$10^z=e^{z\ln 10}$$

$$\sin z=\frac{1}{2i}(e^{iz}-e^{-iz})$$

$$\cos z=\tfrac{1}{2}(e^{iz}+e^{-iz})$$

$$\tan z=\sin z/\cos z$$

$$\sinh z=\tfrac{1}{2}(e^z-e^{-z})$$

$$\cosh z=\tfrac{1}{2}(e^z+e^{-z})$$

$$\tanh z=\sinh z/\cosh z$$

Multivalued Functions

In general, the inverse of a function $f(z)$—denoted by $f^{-1}(z)$—has more than one value for any argument z. For example, $\cos^{-1}(z)$ has infinitely many values for each argument. But the HP-15C calculates the single *principal value*, which lies in the part of the range defined as the principal branch of $f^{-1}(z)$. In the discussion that follows, the single-valued inverse function (restricted to the principal branch) is denoted by uppercase letters—such as $\mathrm{COS}^{-1}(z)$—to distinguish it from the multivalued inverse—$\cos^{-1}(z)$.

For example, consider the nth roots of a complex number z. Write z in polar form as $z=re^{i(\theta+2k\pi)}$ for $-\pi<\theta\leqslant\pi$ and $k=0,\pm1,\pm2,\ldots$. Then if n is a positive integer,

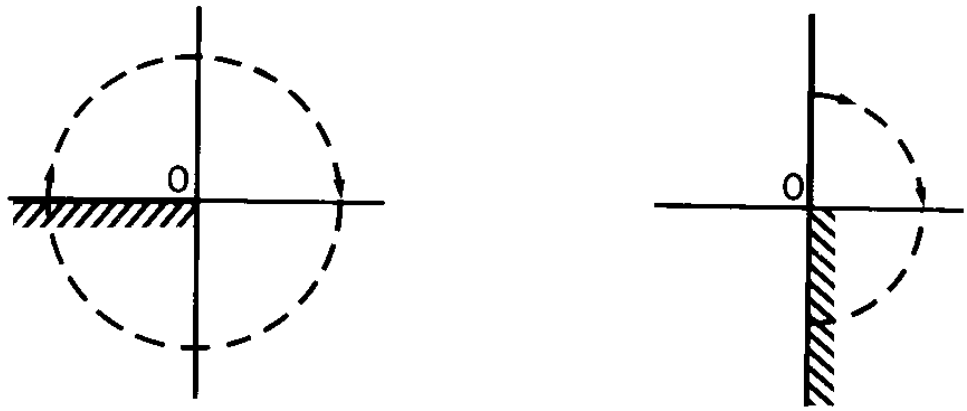
$$z^{1/n}=r^{1/n}e^{i(\theta/n+2k\pi/n)}=r^{1/n}e^{i\theta/n}e^{i2k\pi/n}.$$

Only $k=0,1,\ldots,n-1$ are necessary since $e^{i2k\pi/n}$ repeats its values in cycles of n. The equation defines the nth roots of z, and $r^{1/n}e^{i\theta/n}$ with $-\pi<\theta\leqslant\pi$ is the principal branch of $z^{1/n}$. (A program listed on page 78 computes the nth roots of z.)

The illustrations that follow show the principal branches of the inverse relations. The left-hand graph in each figure represents the cut domain of the inverse function; the right-hand graph shows the range of the principal branch.

For some inverse relations, the definitions of the principal branches are not universally agreed upon. The principal branches used by the HP-15C were carefully chosen. First, they are analytic in the regions where the arguments of the real-valued inverse functions are defined. That is, the branch cut occurs where its corresponding real-valued inverse function is undefined. Second, most of the important symmetries are preserved. For example, $\text{SIN}^{-1}(-z) = -\text{SIN}^{-1}(z)$ for all z.

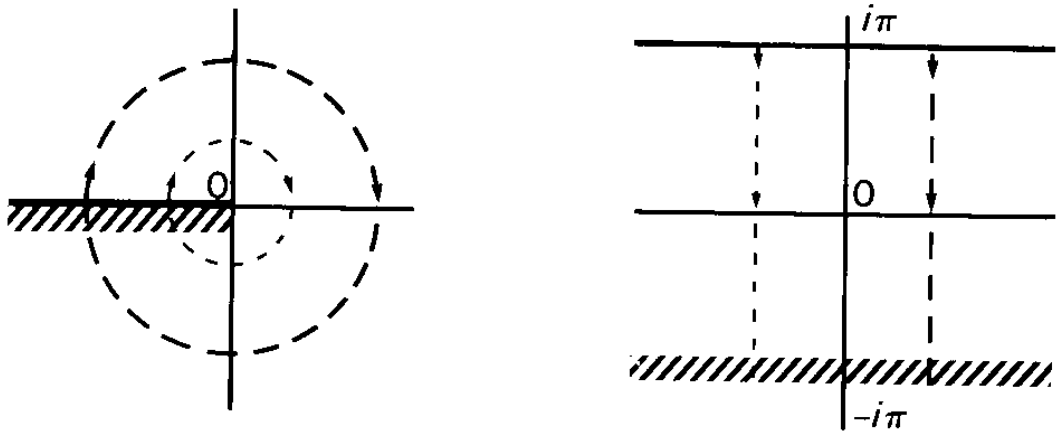
z $\qquad$ $w = \sqrt{z}$



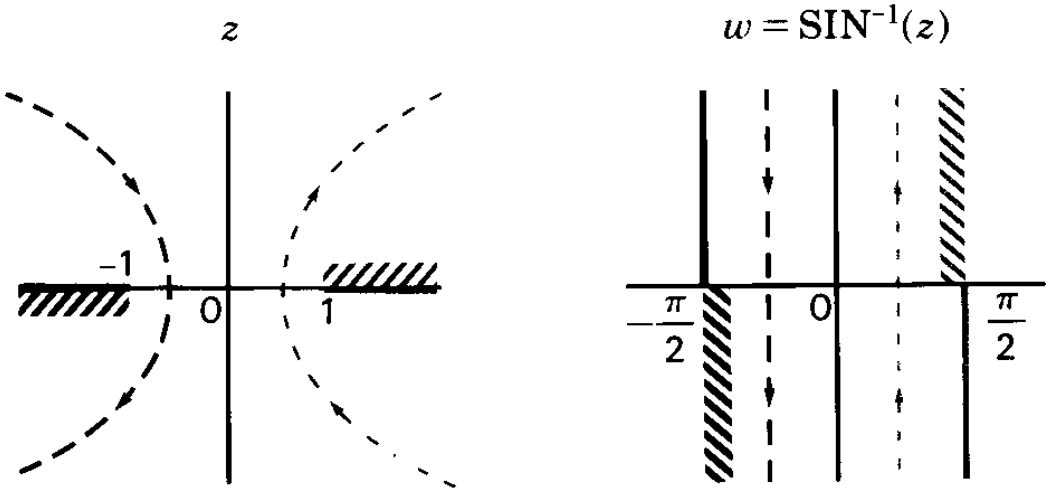
$\sqrt{z} = \sqrt{r}\, e^{i\theta/2}$ $\qquad$ for $-\pi < \theta \leq \pi$

---

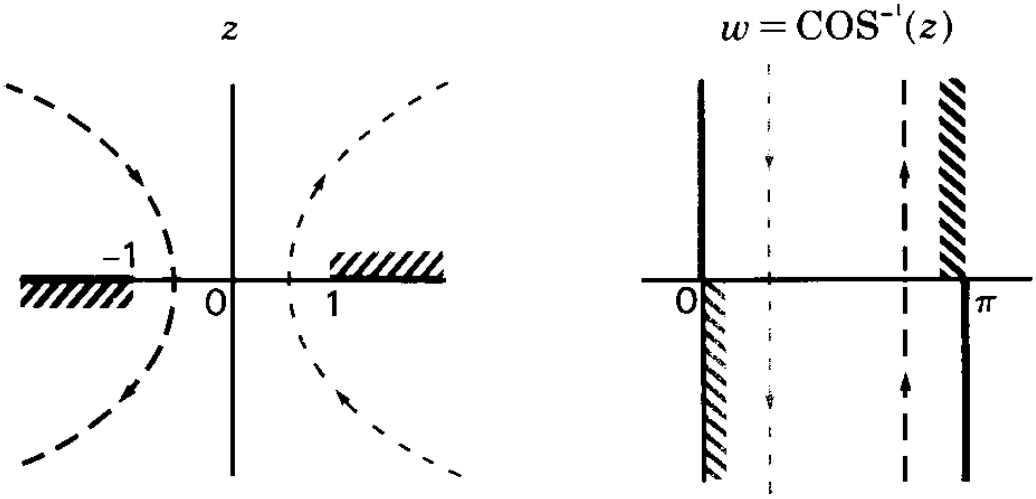
z $\qquad$ $w = \text{LN}(z)$



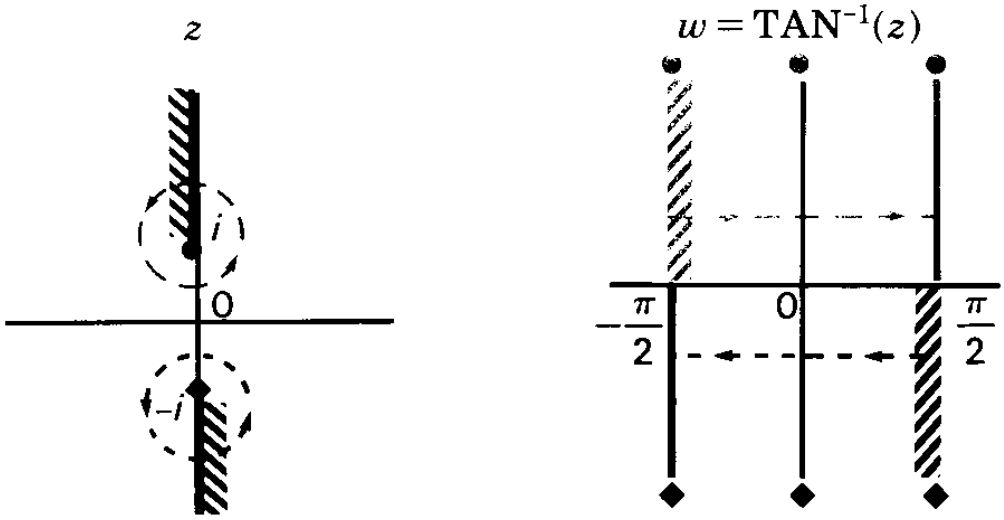
$\text{LN}(z) = \ln r + i\theta$ $\qquad$ for $-\pi < \theta \leq \pi$



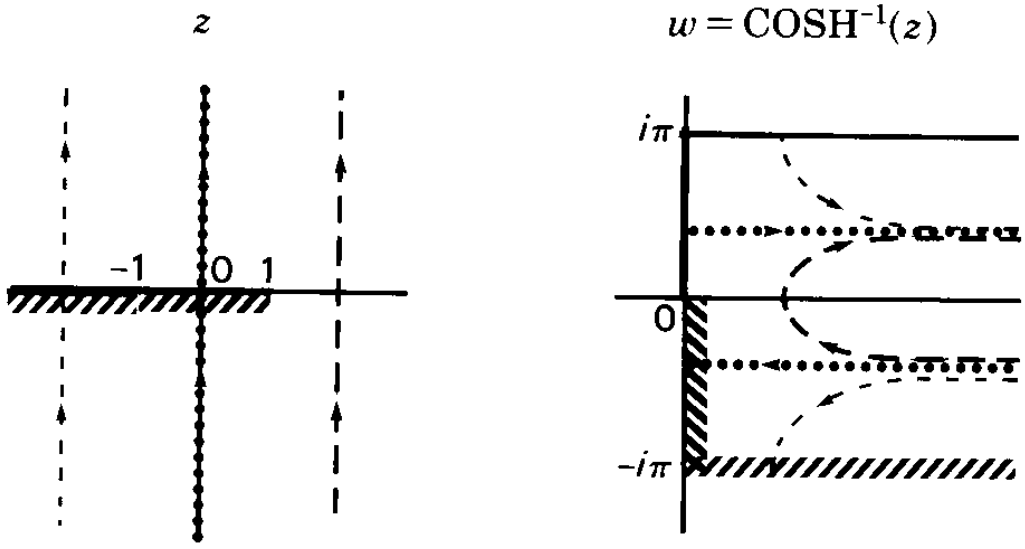
$$\sin^{-1}(z) = -i \ln[iz + (1 - z^2)^{1/2}]$$



$$\cos^{-1}(z) = -i \ln[z + (z^2 - 1)^{1/2}]$$



$$\tan^{-1}(z) = \frac{i}{2} \ln \frac{i + z}{i - z}$$



$$\cosh^{-1}(z) = \ln[z + (z^2 - 1)^{1/2}]$$

The principal branches in the last four graphs above are obtained from the equations shown, but don't necessarily use the principal branches of $\ln(z)$ and $\sqrt{z}$.

The remaining inverse functions may be determined from the illustrations above and the following equations:

$$\text{LOG}(z) = \text{LN}(z)/\text{LN}(10)$$

$$\text{SINH}^{-1}(z) = -i\,\text{SIN}^{-1}(iz)$$

$$\text{TANH}^{-1}(z) = -i\,\text{TAN}^{-1}(iz)$$

$$w^z = e^{z\,\text{LN}(w)}.$$

To determine *all* values of an inverse relation, use the following expressions to derive these values from the principal value calculated by the HP-15C. In these expressions, $k = 0, \pm1, \pm2, \ldots$.

$$z^{1/2} = \pm\sqrt{z}$$

$$\ln(z) = \text{LN}(z) + i2k\pi$$

$$\sin^{-1}(z) = (-1)^k\,\text{SIN}^{-1}(z) + k\pi$$

$$\cos^{-1}(z) = \pm\text{COS}^{-1}(z) + 2k\pi$$

$$\tan^{-1}(z) = \text{TAN}^{-1}(z) + k\pi$$

$$\sinh^{-1}(z) = (-1)^k\,\text{SINH}^{-1}(z) + ik\pi$$

$$\cosh^{-1}(z) = \pm\text{COSH}^{-1}(z) + i2k\pi$$

$$\tanh^{-1}(z) = \text{TANH}^{-1}(z) + ik\pi$$

$$w^z = w^z e^{i2\pi kz}$$

Using [SOLVE] and [∫] in Complex Mode

The [SOLVE] and [∫] functions use algorithms that sample your function at values along the real axis. In Complex mode, the [SOLVE] and [∫] functions operate with only the real stack, even though your function subroutine may have complex computations in it.

For example, [SOLVE] will not search for the roots of a complex function, but rather will sample the function on the real axis and search for a zero of the function's real part. Similarly, [∫] computes the integral of the function's real part along an interval on the real axis. These operations are useful in various applications, such as calculating contour integrals and complex potentials. (Refer to Applications at the end of this section.)

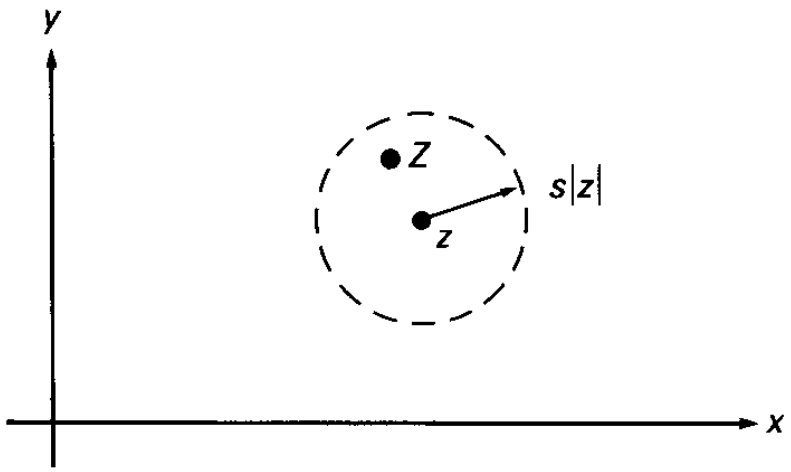
Accuracy in Complex Mode

Because complex numbers have both real components and imaginary components, the accuracy of complex calculations takes on another dimension compared to real-valued calculations.

When dealing with *real* numbers, an approximation X is close to x if the relative difference $E(X,x) = |(X - x)/x|$ is small. This relates directly to the number of correct significant digits of the approximation X. That is, if $E(X,x) < 5 \times 10^{-n}$, then there are at least n significant digits. For *complex* numbers, define $E(Z,z) = |(Z - z)/z|$. This does *not* relate directly to the number of correct digits in each *component* of Z, however.

For example, if $E(X,x)$ and $E(Y,y)$ are both small, then for $z = x + iy$, $E(Z,z)$ must also be small. That is, if $E(X,x) < s$ and $E(Y,y) < s$, then $E(Z,z) < s$. But consider $z = 10^{10} + i$ and $Z = 10^{10}$. The imaginary component of Z is far from accurate, and yet $E(Z,z) < 10^{-10}$. Even though the imaginary components of z and Z are completely different, in a relative sense z and Z are extremely close.

There is a simple, geometric interpretation of the complex relative error. Any approximation Z of z satisfies $E(Z,z) < s$ (where s is a positive real number) if and only if Z lies inside the circle of radius $s|z|$ centered at z in the complex plane.



To require approximations with accurate *components* is to demand more than keeping relative errors small. For example, consider this problem in which the calculations are done with four significant digits. It illustrates the limitations imposed on a complex calculation by finite accuracy.

$$z_1 = Z_1 = 37.1 + 37.3i$$
$$z_2 = Z_2 = 37.5 + 37.3i$$

and

$$\begin{aligned} &Z_1 \times Z_2 \\ &= (37.10 \times 37.50 - 37.30 \times 37.30) + i(37.10 \times 37.30 + 37.30 \times 37.50) \\ &= (1391. - 1391.) + i(1384. + 1399.) \\ &= 0 + i(2783.) \end{aligned}$$

The true value $z_1 z_2 = -0.04 + 2782.58i$. Even though Z_1 and Z_2 have no error, the real part of their four-digit product has no correct significant decimals, although the relative error of the complex product is less than 2×10^{-4}.

The example illustrates that complex multiplication doesn't propogate its errors componentwise. But even if complex multiplication produced accurate components, the rounding errors of a chain computation could quickly produce inaccurate components. On the other hand, the relative error, which corresponds to the precision of the calculation, grows only slowly.

For example, using four-digit accuracy as before

$$z_1 = (1 + 1/300) + i$$
$$Z_1 = 1.003 + i$$
$$z_2 = Z_2 = 1 + i$$

then

$$\begin{aligned} Z_1 \times Z_2 &= (1.003 + i) \times (1 + i) \\ &= 0.003 + 2.003i \\ &= 3.000 \times 10^{-3} + 2.003i \end{aligned}$$

The *correct* four-digit value is $3.333 \times 10^{-3} + 2.003i$. In this example, Z_1 and Z_2 are accurate in each component and the arithmetic is exact. But the product is inaccurate—that is, the real component has only one significant digit. One rounding error causes an inaccurate component, although the complex relative error of the product remains small.

For the HP-15C the results of any complex operation are designed to be accurate in the sense that the complex relative error $E(Z,z)$ is kept small. Generally, $E(Z,z) < 6 \times 10^{-10}$.

As shown earlier, this small relative error doesn't guarantee 10 accurate digits in each component. Because the error is relative to the size $|z|$, and because this is not greatly different from the size of the largest component of z, the smaller component can have fewer accurate digits. There is a quick way for you to see which digits are generally accurate. Express each component using the largest exponent. In this form, approximately the first 10 digits of each component are accurate. For example, if

$$Z = 1.234567890 \times 10^{-10} + i(2.222222222 \times 10^{-3}),$$

then think of Z as

$$0.0000001234567890 \times 10^{-3} + i(2.222222222 \times 10^{-3}).$$

The accurate digits are

$$0.000000123 \times 10^{-3} + i(2.222222222 \times 10^{-3}).$$

Applications

The capability of the HP-15C to work with complex numbers enables you to solve problems that extend beyond the realm of real-valued numbers. On the following pages are several programs that illustrate the usefulness of complex calculations—and the HP-15C.

Storing and Recalling Complex Numbers Using a Matrix

This program uses the stack and matrix **C** to store and recall complex numbers. It has the following characteristics:

- If you specify an index greater than the matrix's dimensions, the calculator displays **Error 3** and the stack is ready for another try.
- If the calculator isn't in Complex mode, the program activates Complex mode and the imaginary part of the number is set to zero.
- When you store a complex number, the index is lost, the stack drops, and the T-register is duplicated in the Z-register.
- The "Store" program uses the [D] key, which is above the [STO] key. The "Recall" program uses the [E] key, which is above the [RCL] key.

Keystrokes	Display	
[g] [P/R]		Program mode.
[f] CLEAR [PRGM]	000-	
[f] [LBL] [D]	001-42,21,14	"Store" program.
[f] [MATRIX] 1	002-42,16, 1	Sets $R_0 = R_1 = 1$.
[STO] 0	003- 44 0	$R_0 = k$.
[R↓]	004- 33	
0	005- 0	Enters 0 in real (and imaginary) X-registers.
[+]	006- 40	Drops stack and has $a + ib$ in X-register.
[f] [USER] [STO] [C]	007u 44 13	Stores a and increments indices (User mode).
[f] [USER]		
[f] [Re≷Im]	008- 42 30	

Keystrokes	Display		
[STO] [C]	009-	44 13	Stores b (*no* User mode here).
[f] [Re≷Im]	010-	42 30	Restores $a + ib$ in X-registers.
[g] [RTN]	011-	43 32	
[f] [LBL] [E]	012-42,21,15		"Recall" program.
[STO] 0	013-	44 0	$R_0 = k$.
[g] [CLx]	014-	43 35	Disables stack.
2	015-	2	
[STO] 1	016-	44 1	Sets $R_1 = 2$.
[R↓]	017-	33	
0	018-	0	
[+]	019-	40	Sets stack for another try if **Error 3** occurs next.
[RCL] [C]	020-	45 13	Recalls b (imaginary part).
[f] [Re≷Im]	021-	42 30	
[f] [DSE] 1	022-42, 5, 1		Decrements to $R_1 = 1$.
[g] [CLx]	023-	43 35	Disables stack and clears real X-register.
[RCL] [C]	024-	45 13	Recalls a (real part).
[g] [RTN]	025-	43 32	

Example: Store $2 + 3i$ and $7 + 4i$ in elements 1 and 2 using the previous program. Then recall and add them. Dimension matrix **C** to 5×2 so that it can store up to 5 complex numbers.

After entering the preceding program:

Keystrokes	Display	
[g] [P/R]		Run mode.
5 [ENTER] 2	2	Specifies 5 rows and 2 columns.
[f] [DIM] [C]	2.0000	Dimensions matrix **C**.
2 [ENTER] 3 [f] [I]	2.0000	Enters $2 + 3i$.
1 [f] [D]	2.0000	Stores number in **C** using index 1.

Keystrokes	Display	
7 ENTER 4 f I	7.0000	Enters $7 + 4i$.
2 f D	7.0000	Stores number in C using index 2.
1 f E	2.0000	Recalls first number.
2 f E	7.0000	Recalls second number.
+	9.0000	Real part of sum.
f Re≷Im	7.0000	Imaginary part of sum.

Calculating the *n*th Roots of a Complex Number

This program calculates the nth roots of a complex number. The roots are z_k for $k = 0, 1, 2, ..., n - 1$. You can also use the program to calculate $z^{1/r}$, where r isn't necessarily an integer. The program operates the same way except that there may be infinitely many roots z_k for $k = 0, \pm 1, \pm 2, ...$.

Keystrokes	Display	
g P/R		Program mode.
f CLEAR PRGM	000-	
f LBL A	001-42,21,11	
x≷y	002- 34	Places n in X-register, z in Y-registers.
1/x	003- 15	Calculates $1/n$.
g LSTx	004- 43 36	Retrieves n.
R↓	005- 33	
g SF 8	006-43, 4, 8	Activiates Complex mode.
y^x	007- 14	Calculates $z^{1/n}$.
STO 2	008- 44 2	Stores real part of z_0 in R_2.
f Re≷Im	009- 42 30	
STO 3	010- 44 3	Stores imaginary part of z_0 in R_3.
3	011- 3	
6	012- 6	
0	013- 0	
g R↑	014- 43 33	
÷	015- 10	Calculates $360/n$.
STO 4	016- 44 4	Stores $360/n$ in R_4.
0	017- 0	
STO I	018- 44 25	Stores 0 in Index register.

Keystrokes	Display	
f LBL 0	019–42,21, 0	
RCL 4	020– 45 4	Recalls $360/n$.
RCL × I	021–45,20,25	Calculates $360k/n$ using Index register.
f Re≷Im	022– 42 30	
g CLx	023– 43 35	
1	024– 1	Places $1 + i(k360/n)$ in the X-register.
g DEG	025– 43 7	Sets Degrees mode.
f →R	026– 42 1	Calculates $e^{ik360/n}$.
RCL 2	027– 45 2	Recalls real part of z_0.
RCL 3	028– 45 3	Recalls imaginary part of z_0.
f I	029– 42 25	Forms complex z_0.
×	030– 20	Calculates $z_0e^{ik360/n}$, root number k.
RCL I	031– 45 25	Recalls number k.
x≷y	032– 34	Places z_k in X-registers, k in Y-register.
1	033– 1	
STO + I	034–44,40,25	Increments number k in Index register.
R↓	035– 33	Restores z_k and k to X- and Y-registers.
R/S	036– 31	Halts execution.
GTO 0	037– 22 0	Branch for next root.

Labels used: A and 0.

Registers used: R_2, R_3, R_4, and Index register.

To use this program:

1. Enter the order n into the Y-register and the complex number z into the X-registers.
2. Press f A to calculate the principal root, z_0, which is placed in the real and imaginary X-registers. (Press and hold f (i) to view the imaginary part.)

3. To calculate higher number roots z_k:
   - Press [R/S] to calculate each successive higher-number root. Each root z_k is placed in the complex X-registers and its number k is placed in the Y-register. Between root calculations, you can perform other calculations without disturbing this program (if R_2, R_3, R_4, and the Index register aren't changed).
   - Store the number of the root k in the Index register (using [STO] [I]), then press [R/S] to calculate z_k. The complex root and its number are placed in the X- and Y-registers, respectively. (By pressing [R/S] again, you can continue calculating higher-number roots.)

Example: Use the previous program to compute $(1)^{1/100}$. Calculate z_0, z_1, and z_{50} for this expression.

Keystrokes	Display	
[g] [P/R]		Run mode.
100 [ENTER] 1	**1**	Enters $n = 100$ and $z = 1$ (purely real).
[f] [A]	**1.0000**	Calculates z_0 (real part).
[f] [(i)] (hold)	**0.0000**	Imaginary part of z_0.
[R/S]	**0.9980**	Calculates z_1 (real part).
[f] [(i)] (hold)	**0.0628**	Imaginary part of z_1.
50 [STO] [I]	**50.0000**	Stores root number in Index register.
[R/S]	**-1.0000**	Calculates z_{50} (real part).
[f] [(i)] (hold)	**0.0000**	Imaginary part of z_{50}.

Solving an Equation for Its Complex Roots

A common method for solving the complex equation $f(z) = 0$ numerically is Newton's iteration. This method starts with an approximation z_0 to a root and repeatedly calculates

$$z_{k+1} = z_k - f(z_k)/f'(z_k)$$

until z_k converges.

The following example shows how [SOLVE] can be used with Newton's iteration to estimate complex roots. (A different

technique that doesn't use Complex mode is mentioned on page 16.)

Example: The response of an automatically controlled system to small transient perturbations has been modeled by the differential-delay equation

$$\frac{d}{dt}w(t) + 9\,w(t) + 8\,w(t-1) = 0.$$

How stable is this system? In other words, how rapidly do solutions of this equation decay?

Every solution $w(t)$ is known to be expressible as a sum

$$w(t) = \sum_k c(z)e^{zt}$$

involving constant coefficients $c(z)$ chosen for each root z of the differential-delay equation's associated characteristic equation:

$$z + 9 + 8e^{-z} = 0.$$

Every root $z = x + iy$ contributes to $w(t)$ a component $e^{zt} = e^{xt}(\cos(yt) + i\,\sin(yt))$ whose rate of decay is faster as x, the real part of z, is more negative. Therefore, the answer to the question entails the calculation of all the roots z of the characteristic equation. Since that equation has infinitely many roots, none of them real, the calculation of *all* roots could be a large task.

However, the roots z are known to be approximated for large integers n by $z \approx A(n) = -\ln((2n + \frac{1}{2})\pi/8) \pm i(2n + \frac{1}{2})\pi$ for $n = 0, 1, 2, \ldots$. The bigger is n, the better is the approximation. Therefore you need calculate only the few roots not well approximated by $A(n)$—the roots with $|z|$ not very big.

When using Newton's iteration, what should $f(z)$ be for this problem? The obvious function $f(z) = z + 9 + 8e^{-z}$ isn't a good choice because the exponential grows rapidly for larger negative values of $\mathrm{Re}(z)$. This would slow convergence considerably unless the first guess z_0 were extremely close to a root. In addition, this $f(z)$ vanishes infinitely often, so it's difficult to determine when all desired roots have been calculated. But by rewriting this equation as

$$e^z = -8/(z + 9)$$

and taking logarithms, you obtain an equivalent equation

$$z = \ln(-8/(z+9)) \pm i2n\pi \qquad \text{for } n = 0, 1, 2, \ldots$$

This equation has only two complex conjugate roots z for each integer n. Therefore use the equivalent function

$$f(z) = z - \ln(-8/(z+9)) \pm i2n\pi \qquad \text{for } n = 0, 1, 2, \ldots.$$

and apply Newton's iteration

$$z_{k+1} = z_k - (z_k - \ln(-8/(z_k+9)) \pm i2n\pi)/(1 + 1/(z_k+9)).$$

As a first guess, choose z_0 as $A(n)$, the approximation given earlier. A bit of algebraic rearrangement using the fact that $\ln(\pm i) = \pm i\pi/2$ leads to this formula:

$$z_{k+1} = A(n) + ((z_k - A(n)) + (z_k+9)\ln(i\mathrm{Im}(A(n))/(z_k+9)))/(z_k+10).$$

In the program below, $\mathrm{Re}(A(n))$ is stored in R_0 and $\mathrm{Im}(A(n))$ is stored in R_1. Note that only one of each conjugate pair of roots is calculated for each n.

Keystrokes	Display		Comments
g P/R			Program mode.
f CLEAR PRGM	000-		
f LBL A	001-	42,21,11	Program for $A(n)$.
g CF 8	002-	43, 5, 8	Specifies real arithmetic.
ENTER	003-	36	
+	004-	40	
.	005-	48	
5	006-	5	
+	007-	40	
g π	008-	43 26	
×	009-	20	Calculates $(2n + ½)\pi$.
ENTER	010-	36	
STO 1	011-	44 1	
8	012-	8	
÷	013-	10	
g LN	014-	43 12	
CHS	015-	16	Calculates $-\ln((2n + ½)\pi/8)$.
STO 0	016-	44 0	
x≷y	017-	34	
f I	018-	42 25	Forms complex $A(n)$.

Keystrokes	Display	
g RTN	019– 43 32	
f LBL B	020–42,21,12	Program for z_{k+1}.
ENTER	021– 36	
ENTER	022– 36	
RCL 1	023– 45 1	
f Re≷Im	024– 42 30	Creates $i\text{Im}(A(n))$.
x≷y	025– 34	
9	026– 9	
+	027– 40	
÷	028– 10	
g LSTx	029– 43 36	
x≷y	030– 34	
g LN	031– 43 12	
×	032– 20	
x≷y	033– 34	
RCL 1	034– 45 1	
f Re≷Im	035– 42 30	
RCL + 0	036–45,40, 0	
−	037– 30	
g LSTx	038– 43 36	
R↓	039– 33	
+	040– 40	
x≷y	041– 34	
1	042– 1	
0	043– 0	
+	044– 40	
÷	045– 10	
+	046– 40	
g RTN	047– 43 32	
f LBL C	048–42,21,13	Program for residual, $\lvert e^z + 8/(z+9)\rvert$.
ENTER	049– 36	
e^x	050– 12	
9	051– 9	
g LSTx	052– 43 36	
+	053– 40	
8	054– 8	
x≷y	055– 34	
÷	056– 10	
+	057– 40	

Keystrokes	Display		
g ABS	058-	43 16	Calculates $\lvert e^z + 8/(z+9)\rvert$.
g RTN	059-	43 32	

Labels used: A, B, and C.

Registers used: R_0 and R_1.

Now run the program. For each root, press B until the displayed real part doesn't change. (You might also check that the imaginary part doesn't change.)

Keystrokes	Display		
g P/R			Run mode.
f USER			Activates User mode.
0 A	1.6279		Displays $\mathrm{Re}(A(0)) = \mathrm{Re}(z_0)$.
B	-0.1487		$\mathrm{Re}(z_1)$.
B	-0.1497		$\mathrm{Re}(z_2)$.
B	-0.1497		$\mathrm{Re}(z)$.
f (i) (hold)	2.8319		$\mathrm{Im}(z)$.
C	1.0000	-10	Calculates residual.
x≷y	-0.1497		Restores z to X-register.

By repeating the same process for $n = 1$ through 5, you will obtain the results listed below. (Only one of each pair of complex roots is listed.)

n	$A(n)$	Root z_k	Residual
0	$1.6279 + i1.5708$	$-0.1497 + i2.8319$	1×10^{-10}
1	$0.0184 + i7.8540$	$-0.4198 + i8.6361$	6×10^{-10}
2	$-0.5694 + i14.1372$	$-0.7430 + i14.6504$	2×10^{-9}
3	$-0.9371 + i20.4204$	$-1.0236 + i20.7868$	5×10^{-10}
4	$-1.2054 + i26.7035$	$-1.2553 + i26.9830$	9×10^{-10}
5	$-1.4167 + i32.9867$	$-1.4486 + i33.2103$	2×10^{-9}

As n increases, the first guess $A(n)$ comes ever closer to the desired root z. (When you're finished, press [f] [USER] to deactivate User mode.)

Since all roots have negative real parts, the system is stable, but the margin of stability (the smallest in magnitude among the real parts, namely -0.1497) is small enough to cause concern if the system must withstand much noise.

Contour Integrals

You can use [∫f] to evaluate the contour integral $\int_C f(z)dz$, where C is a curve in the complex plane.

First parameterize the curve C by $z(t) = x(t) + iy(t)$ for $t_1 \leqslant t \leqslant t_2$. Let $G(t) = f(z(t))z'(t)$. Then

$$\int_C f(z)dz = \int_{t_1}^{t_2} G(t)dt$$

$$= \int_{t_1}^{t_2} \mathrm{Re}(G(t))dt + i\int_{t_1}^{t_2} \mathrm{Im}(G(t))dt.$$

These integrals are precisely the type that [∫f] evaluates in Complex mode. Since $G(t)$ is a complex function of a real variable t, [∫f] will sample $G(t)$ on the interval $t_1 \leqslant t \leqslant t_2$ and integrate $\mathrm{Re}(G(t))$—the value that your function returns to the real X-register. For the imaginary part, integrate a function that evaluates $G(t)$ and uses [Re≷Im] to place $\mathrm{Im}(G(t))$ into the real X-register.

The general-purpose program listed below evaluates the complex integral

$$I = \int_a^b f(z)dz$$

along the straight line from a to b, where a and b are complex numbers. The program assumes that your complex function subroutine is labeled "B" and evaluates the complex function $f(z)$, and that the limits a and b are in the complex Y- and X-registers, respectively. The complex components of the integral I and the uncertainty ΔI are returned in the X- and Y-registers.

Keystrokes	Display	
[g] [P/R]		Program mode.
[f] CLEAR [PRGM]	000–	

Keystrokes	Display	
f LBL A	001-42,21,11	
x≷y	002- 34	
−	003- 30	Calculates $b - a$.
STO 4	004- 44 4	Stores Re($b - a$) in R_4.
f Re≷Im	005- 42 30	
STO 5	006- 44 5	Stores Im($b - a$) in R_5.
g LSTx	007- 43 36	Recalls a.
STO 6	008- 44 6	Stores Re(a) in R_6.
f Re≷Im	009- 42 30	
STO 7	010- 44 7	Stores Im(a) in R_7.
0	011- 0	
ENTER	012- 36	
1	013- 1	
f /: 0	014-42,20, 0	Calculates Im(I) and Im(ΔI).
STO 2	015- 44 2	Stores Im(I) in R_2.
R↓	016- 33	
STO 3	017- 44 3	Stores Im(ΔI) in R_3.
R↓	018- 33	
f /: 1	019-42,20, 1	Calculates Re(I) and Re(ΔI).
RCL 2	020- 45 2	Recalls Im(I).
f I	021- 42 25	Forms complex I.
x≷y	022- 34	
RCL 3	023- 45 3	Recalls Im(ΔI).
f I	024- 42 25	Forms complex ΔI.
x≷y	025- 34	Restores I to X-register.
g RTN	026- 43 32	
f LBL 0	027-42,21, 0	Subroutine for Im($f(z)z'(t)$).
GSB 1	028- 32 1	Calculates $f(z)z'(t)$.
f Re≷Im	029- 42 30	Swaps complex components.
g RTN	030- 43 32	
f LBL 1	031-42,21, 1	Subroutine to calculate $f(z)z'(t)$.

Keystrokes	Display		
RCL 4	032-	45 4	
RCL 5	033-	45 5	
f I	034-	42 25	Forms complex $b-a$.
×	035-	20	Calculates $(b-a)t$.
RCL 6	036-	45 6	
RCL 7	037-	45 7	
f I	038-	42 25	Forms complex a.
+	039-	40	Calculates $a+(b-a)t$.
GSB B	040-	32 12	Calculates $f(a+(b-a)t)$.
RCL 4	041-	45 4	
RCL 5	042-	45 5	
f I	043-	42 25	Forms complex $z'(t)=b-a$.
×	044-	20	Calculates $f(z)z'(t)$.
g RTN	045-	43 32	

Labels used: A, 0, and 1.

Registers used: R_2, R_3, R_4, R_5, R_6, and R_7.

To use this program:

1. Enter your function subroutine labeled "B" into program memory.
2. Press 7 f DIM (i) to reserve registers R_0 through R_7. (Your subroutine may require additional registers.)
3. Set the display format for $f\hat{x}$.
4. Enter the two complex points that define the ends of the straight line that your function will be integrated along. The lower limit should be in the Y-registers; the upper limit should be in the X-registers.
5. Press f A to calculate the complex line integral. The value of the integral is in the X-registers; the value of the uncertainty is in the Y-registers.

Because two integrals are being evaluated, the program will usually take longer than a real integral, although the $f\hat{x}$ routine doesn't have to use the same number of sample points for both integrals. The easier integral will use less calculations than the more difficult one.

Example: Approximate the integrals

$$I_1 = \int_1^{\infty} \frac{\cos x}{x + 1/x}\,dx \qquad \text{and} \qquad I_2 = \int_1^{\infty} \frac{\sin x}{x + 1/x}\,dx.$$

These integrands decay very slowly as x approaches infinity and therefore require a long interval of integration and a long execution time. You can expedite this calculation by deforming the path of integration from the real axis into the complex plane. According to complex variable theory, these integrals can be combined as

$$I_1 + iI_2 = \int_1^{1+i\infty} \frac{e^{iz}}{z + 1/z}\,dz.$$

This complex integrand, evaluated along the line $x = 1$ and $y \geq 0$, decays rapidly as y increases—like e^{-y}.

To use the previous program to calculate both integrals at the same time, write a subroutine to evaluate

$$f(z) = \frac{e^{iz}}{z + 1/z}.$$

Keystrokes	Display	
f LBL B	046-42,21,12	
1/x	047- 15	
g LSTx	048- 43 36	
+	049- 40	Calculates $z + 1/z$.
g LSTx	050- 43 36	
1	051- 1	
f Re≷Im	052- 42 30	Forms $0 + i$.
×	053- 20	
e^x	054- 12	Calculates e^{iz}.
x≷y	055- 34	
÷	056- 10	Calculates $f(z)$.
g RTN	057- 43 32	

Approximate the complex integral by integrating the function from $1 + 0i$ to $1 + 6i$ using a SCI 2 display format to obtain three significant digits. (The integral beyond $1 + 6i$ doesn't affect the first three digits.)

Keystrokes	Display		
[g] [P/R]			Run mode.
[f] [SCI] 2			Specifies [SCI] 2 format.
1 [ENTER]	1.00	00	Enters first limit of integration, $1 + 0i$.
1 [ENTER] 6	6		
[f] [I]	1.00	00	Enters second limit of integration, $1 + 6i$.
[f] [A]	-3.24	-01	Calculates I and displays $\mathrm{Re}(I) = I_1$ (after about 9 minutes).
[f] [(i)] (hold)	3.82	-01	Displays $\mathrm{Im}(I) = I_2$.
[x≷y]	7.87	-04	Displays $\mathrm{Re}(\Delta I) = \Delta I_1$.
[f] [(i)] (hold)	1.23	-03	Displays $\mathrm{Im}(\Delta I) = \Delta I_2$.
[f] [FIX] 4	0.0008		

This result I is calculated much more quickly than if I_1 and I_2 were calculated directly along the real axis.

Complex Potentials

Conformal mapping is useful in applications associated with a complex potential function. The discussion that follows deals with the problem of fluid flow, although problems in electrostatics and heat flow are analagous.

Consider the potential function $P(z)$. The equation $\mathrm{Im}(P(z)) = c$ defines a family of curves that are called *streamlines* of the flow. That is, for any value of c, all values of z that satisfy the equation lie on a streamline corresponding to that value of c. To calculate some points z_k on the streamline, specify some values for x_k and then use [SOLVE] to find the corresponding values of y_k using the equation

$$\mathrm{Im}(P(x_k + iy_k)) = c.$$

If the x_k values are not too far apart, you can use y_{k-1} as an initial estimate for y_k. In this way, you can work along the streamline and calculate the complex points $z_k = x_k + iy_k$. Using a similar procedure, you can define the equipotential lines, which are given by $\mathrm{Re}(P(z)) = c$.

The program listed below is set up to compute the values of y_k from evenly spaced values of x_k. You must provide a subroutine labeled "B" that places $\mathrm{Im}(P(z))$ in the real X-register. The program uses inputs that specify the step size h, the number of points n along the real axis, and $z_0 = x_0 + iy_0$, the initial point on the streamline. You must enter n, h, and z_0 into the Z-, Y-, and X-registers before running the program.

The program computes the values of z_k and stores them in matrix **A** in the form $a_{k1} = x_{k-1}$ and $a_{k2} = y_{k-1}$ for $k = 1, 2, ..., n$.

Keystrokes	Display	
g P/R		Program mode.
f CLEAR PRGM	000-	
f LBL A	001-42,21,11	
R↓	002- 33	
STO 4	003- 44 4	Stores h in R_4.
R↓	004- 33	
2	005- 2	
f DIM A	006-42,23,11	Dimensions matrix **A** to be $n \times 2$.
g CLx	007- 43 35	
STO MATRIX A	008-44,16,11	Makes all elements of **A** be zero.
STO I	009- 44 25	Stores zero in Index register.
f MATRIX 1	010-42,16, 1	Sets $R_0 = R_1 = 1$.
g R↑	011- 43 33	Recalls z_0 to X-registers.
STO 2	012- 44 2	Stores x_0 in R_2.
f USER STO A	013u 44 11	Sets $a_{11} = x_0$.
f USER		
f Re≷Im	014- 42 30	
STO 3	015- 44 3	Stores y_0 in R_3.
f USER STO A	016u 44 11	Sets $a_{12} = y_0$.
f USER		
GTO 1	017- 22 1	Branches if matrix **A** not full ($n > 1$).
f LBL 0	018-42,21, 0	
RCL MATRIX A	019-45,16,11	Recalls descriptor of matrix **A**.

Keystrokes	Display	
g RTN	020- 43 32	
f LBL 1	021-42,21, 1	
f Re≷Im	022- 42 30	Restores z_0.
GSB B	023- 32 12	Calculates $\mathrm{Im}(P(z_0))$ (or $\mathrm{Re}(P(z_0))$ for equipotential line.)
STO 5	024- 44 5	Stores c in R_5.
f LBL 2	025-42,21, 2	Loop for finding y_k.
1	026- 1	
STO + I	027-44,40,25	Increments counter k in Index register.
RCL 4	028- 45 4	Recalls h.
RCL I	029- 45 25	Recalls counter k.
×	030- 20	Calculates kh.
RCL 2	031- 45 2	Recalls x_0.
+	032- 40	Calculates $x_k = x_0 + kh$.
STO 6	033- 44 6	Stores x_k in R_6.
RCL 3	034- 45 3	Recalls y_{k-1} from R_3.
ENTER	035- 36	Duplicates y_{k-1} for second estimate.
f SOLVE 3	036-42,10, 3	Searches for y_k.
GTO 4	037- 22 4	Branches for valid y_k root.
1	038- 1	Starts decreasing step size.
STO - I	039-44,30,25	Decrements counter k.
4	040- 4	
STO ÷ 4	041-44,10, 4	Reduces h by factor of 4.
STO × I	042-44,20,25	Multiplies counter by 4.
GTO 2	043- 22 2	Loops back to find y_k again.
f LBL 4	044-42,21, 4	Continues finding y_k.
RCL 6	045- 45 6	
f PSE	046- 42 31	Displays x_k.
f USER STO A	047u 44 11	Sets $a_{k+1,1} = x_k$.
f USER		

Keystrokes	Display		
R↓	048-	33	
f PSE	049-	42 31	Displays y_k.
STO 3	050-	44 3	Stores y_k in R_3.
f USER STO A	051u	44 11	Sets $a_{k+1,2} = y_k$.
f USER			
GTO 2	052-	22 2	Branch for $k+1 < n$ (**A** isn't full).
GTO 0	053-	22 0	Branch for $k+1 = n$ (**A** is full).
f LBL 3	054-	42,21, 3	Function subroutine for SOLVE.
RCL 6	055-	45 6	Recalls x_k.
x≷y	056-	34	Restores current estimate for y_k.
f I	057-	42 25	Creates estimate $z_k = x_k + iy_k$.
GSB B	058-	32 12	Calculates $\mathrm{Im}(P(z_k))$ (or $\mathrm{Re}(P(z_k))$ for equipotential lines).
RCL 5	059-	45 5	Recalls c.
-	060-	30	Calculates $\mathrm{Im}(P(z_k)) - c$.
g RTN	061-	43 32	

Labels used: A, B, 0, 1, 2, 3, and 4.

Registers used: R_0, R_1, R_2 (x_0), R_3 (y_0), R_4 (h), R_5 (c), R_6 (x_k), and Index register (k).

Matrix used: **A**.

One special feature of this program is that if an x_k value lies beyond the domain of the streamline (so that there is no root for SOLVE to find), then the step size is decreased so that x_k approaches the boundary where the streamline turns back. This feature is useful for determining the nature of the streamline when y_k isn't a single-valued function of x_k. If h is small enough, the values of z_k will lie on one branch of the streamline and approach the boundary. (The second example below illustrates this feature.)

To use this program:

1. Enter your subroutine labeled "B" into program memory. It should place into the real X-register $\mathrm{Im}(P(z))$ when calculating streamlines or $\mathrm{Re}(P(z))$ when calculating equipotential lines.
2. Press 6 [f] [DIM] [(i)] to reserve registers R_0 through R_6 (and the Index register). (Your subroutine may require additional registers.)
3. Enter the values of n and h into the Z- and Y-registers by pressing n [ENTER] h [ENTER].
4. Enter the complex value of $z_0 = x_0 + iy_0$ into the X-registers by pressing x_0 [ENTER] y_0 [f] [I].
5. Press [f] [A] to display the successive values of x_k and y_k for $k = 1, \ldots, n$ and finally the descriptor of matrix **A**. The values for $k = 0, \ldots, n$ are stored in matrix **A**.
6. If desired, recall values from matrix **A**.

Example: Calculate the streamline of the potential $P(z) = 1/z + z$ passing through $z = -2 + 0.1i$.

First, enter subroutine "B" to compute $\mathrm{Im}(P(z))$.

Keystrokes	Display		
[f] [LBL] [B]	**062-42,21,12**		
[ENTER]	**063-**	**36**	Duplicates z.
[1/x]	**064-**	**15**	
[+]	**065-**	**40**	Calculates $1/z + z$.
[f] [Re≷Im]	**066-**	**42 30**	Places $\mathrm{Im}(P(z))$ in X-register.
[g] [RTN]	**067-**	**43 32**	

Determine the streamline using $z_0 = -2 + 0.1i$, step size $h = 0.5$, and number of points $n = 9$.

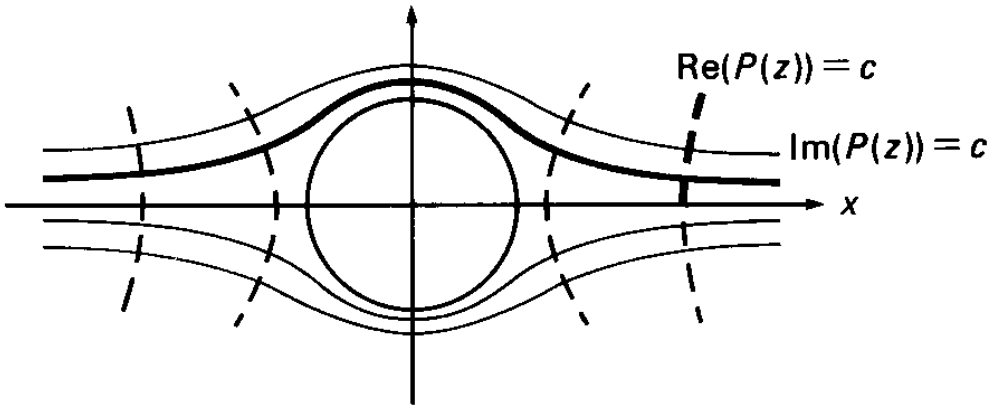
Keystrokes	Display	
[g] [P/R]		Run mode.
9 [ENTER]	**9.0000**	Enters n.
.5 [ENTER]	**0.5000**	Enters h.

Keystrokes	Display			
2 CHS ENTER	-2.0000			
.1 f I	-2.0000			Enters z_0.
f A	-1.5000			x_1.
	0.1343			y_1.
	⋮			⋮
	2.0000			x_9.
	0.1000			y_9.
	A	9	2	Descriptor for matrix **A**.
g CF 8	A	9	2	Deactivates Complex mode.

Matrix **A** contains the following values of x_k and y_k:

x_k	y_k
-2.0	0.1000
-1.5	0.1343
-1.0	0.4484
-0.5	0.9161
0.0	1.0382
0.5	0.9161
1.0	0.4484
1.5	0.1343
2.0	0.1000

The streamline and velocity equipotential lines are illustrated below. The derived streamline corresponds to the heavier solid line.



Example: For the same potential as the previous example, $P(z) = 1/z + z$, compute the velocity equipotential line starting at $z = 2 + i$ and proceeding to the left.

First change subroutine "B" so that it returns Re($P(z)$)—that is, remove the [Re≷Im] instruction from "B". Try $n = 6$ and $h = -0.5$. (Notice that h is negative, which specifies that x_k will be to the left of x_0.)

Although the keystrokes are not listed here, the results that would be calculated and stored in matrix **A** are shown below.

x_k	y_k
2.0000	1.0000
1.8750	0.2362
1.8672	0.1342
1.8652	0.0941
1.8647	0.0811
1.8646	0.0775

The results show the nature of the top branch of the curve (the heavier dashed line in the graph for the previous example). Note that the step size h is automatically decreased in order to follow the curve—rather than stop with an error—when no y-value is found for $x < 1.86$.