

# HEWLETT-PACKARD

Step-by-Step Solutions  
For Your HP Calculator

## Mathematical Applications

$$\frac{b^2 - 4ac}{2a} = Pr \frac{2 - 2.1}{1.085} = \int_{t_1}^{t_2} \text{Re}(G(t)) dt + i \int_{t_1}^{t_2} \text{Im}(G(t)) dt$$

$$\leq 3] = Pr \left[ \frac{2 - 2.151}{1.085} < \frac{X - \mu}{f(x\sigma\Delta) - 1.085\Delta} \leq \frac{3 - 2.151}{1.085\Delta} \right]$$

$$U \frac{e^{iz}}{z + 1/z} dz \cdot \frac{dz}{s} = \frac{f(x + \Delta) - f(x - \Delta)}{30i}$$

$$\frac{ac}{b^2} = P \left( \frac{3 - 2.151}{1.085} \right) - P \left( \frac{2 - 2.151}{1.085} \right) J_1(x) =$$

$$\int_{t_1}^{t_2} G(t) dt \quad B = \begin{bmatrix} 3 & 3 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{3} & 1 \end{bmatrix} \begin{bmatrix} 3 & 3 \\ 0 & 0 \end{bmatrix}$$

$$z^n = r^n e^{in\theta}$$

HP-28S



HEWLETT  
PACKARD



# **Mathematical Applications**

---

## **Step-by-Step Solutions for Your HP-28S Calculator**



Edition 4 July 1990  
Reorder Number 00028-90111

---

# Notice

This manual and any keystroke programs contained herein are provided " as is" and are subject to change without notice. **Hewlett-Packard Company makes no warranty of any kind with regard to this manual or the keystroke programs contained herein, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.** Hewlett-Packard Co. shall not be liable for any errors or for incidental or consequential damages in connection with the furnishing, performance, or use of this manual or the keystroke programs contained herein.

© Hewlett-Packard Co. 1988. All rights reserved. Reproduction, adaptation, or translation of this manual, including any programs, is prohibited without prior written permission of Hewlett-Packard Company, except as allowed under the copyright laws. Hewlett-Packard Company grants you the right to use any program contained in this manual in this Hewlett-Packard calculator.

The programs that control your calculator are copyrighted and all rights are reserved. Reproduction, adaptation, or translation of those programs without prior written permission of Hewlett-Packard Company is also prohibited.

**Corvallis Division**  
**1000 N.E. Circle Blvd.**  
**Corvallis, OR 97330, U.S.A.**

---

# Printing History

<b>Edition 1</b>	February 1988	Mfg. No. 00028-90162
<b>Edition 2</b>	April 1988	Mfg. No. 00028-90126
<b>Edition 3</b>	June 1988	Mfg. No. 00028-90132
<b>Edition 4</b>	July 1990	Mfg. No. 00028-90162



# Welcome...

---

... to the HP-28S Solutions Books. These books are designed to help you get the most from your HP-28S calculator.

This book, *Mathematical Applications*, provides examples and programs for solving problems on your calculator. A variety of polynomial, differential equation, coordinate transformation, curve fitting, and triangle solutions programs are designed to familiarize you with the many functions built into your calculator.

Before you try the examples in this book, you should be familiar with certain basic concepts from the owner's documentation:

- The basics of your calculator: how to move from menu to menu, how to use the graphics capabilities, how to edit expressions and equations, and how to use the menu to assign values to, and solve for, user variables.
- Entering numbers, programs, and algebraic expressions into the calculator.

Please review the section, "How To Use This Book." It contains important information on the examples in this book.

For more information about the topics in the *Mathematical Applications* book, refer to a basic textbook on the subject. Many references are available in university libraries and in technical and college bookstores. The examples in the book demonstrate approaches to solving certain problems, but they do not cover the many ways to approach solutions to mathematical problems.



# Contents

---

<b>11</b>	<b>How To Use This Book</b>
<b>14</b>	How This Book Is Organized
<b>14</b>	Program Entry and Listings
<b>14</b>	PSUB (Polynomial Subtract) Main Program
<b>16</b>	Entering Programs
<b>18</b>	Using SYSEVAL

---

<b>1</b>	<b>19</b>	<b>Polynomial Value and Coefficients</b>
	<b>19</b>	Instructions
	<b>19</b>	Conversion to List Form
	<b>20</b>	Conversion to Polynomial Value
	<b>21</b>	Entering Coefficients
	<b>22</b>	Examples
	<b>24</b>	Programs
	<b>24</b>	PCOEF (Polynomial Coefficients) Main Program
	<b>25</b>	PVAL (Polynomial Value) Main Program
	<b>25</b>	PSERS (Polynomial Series) Main Program

---

<b>2</b>	<b>26</b>	<b>Polynomial Arithmetic</b>
	<b>27</b>	Instructions
	<b>27</b>	Conversion to List Form
	<b>28</b>	Conversion to Polynomial Value
	<b>29</b>	Entering Coefficients
	<b>30</b>	Examples

<b>31</b>	Programs
<b>31</b>	PADD (Polynomial Add) Main Program
<b>32</b>	PSUB (Polynomial Subtract) Main Program
<b>33</b>	PMULT (Polynomial Multiply) Main Program
<b>34</b>	PDIV (Polynomial Divide) Main Program

---

<b>3</b>	<b>36</b>	<b>Polynomial Root Finding</b>
	<b>36</b>	Algorithms
	<b>37</b>	Linear Expressions
	<b>37</b>	Quadratic Polynomials
	<b>37</b>	Cubic Polynomials
	<b>38</b>	Fourth-Order Polynomials
	<b>40</b>	Higher Than Fourth-Order Polynomials
	<b>41</b>	Instructions
	<b>41</b>	Conversion to List Form
	<b>42</b>	Conversion to Polynomial Value
	<b>43</b>	Entering Coefficients
	<b>43</b>	Examples
	<b>45</b>	Programs
	<b>45</b>	PROOT (Polynomial Root Finder) Main Program
	<b>46</b>	QUD (Quadratic) Subroutine
	<b>47</b>	CUBIC (Cubic Solve) Subroutine
	<b>48</b>	QUAR (Quartic Solver) Subroutine

---

<b>4</b>	<b>50</b>	<b>Symbolic Solutions to Differential Equations</b>
	<b>50</b>	Algorithms
	<b>52</b>	Instructions
	<b>53</b>	Examples
	<b>56</b>	Programs
	<b>56</b>	SDEQ (Symbolic Differential Equation) Program
	<b>57</b>	COLX (Simplify a Polynomial in X) Utility
	<b>58</b>	PART1 (Particular Solution 1) Utility
	<b>59</b>	PART2 (Particular Solution 2) Utility
	<b>59</b>	HOMOG (Homogeneous Solution) Utility

---

<b>5</b>	<b>60</b>	<b>Numerical Solutions to First-Order Differential Equations</b>
	<b>60</b>	Algorithms
	<b>61</b>	Instructions
	<b>62</b>	Example
	<b>63</b>	Programs
	<b>63</b>	DE1 (First-Order Differential Equation) Program
	<b>64</b>	YI1 (First-Order Y Increment) Subroutine

---

<b>6</b>	<b>65</b>	<b>Numerical Solutions to Second-Order Differential Equations</b>
	<b>65</b>	Algorithms
	<b>66</b>	Instructions
	<b>67</b>	Example
	<b>68</b>	Programs
	<b>68</b>	DE2 (Second-Order Differential Equation) Program
	<b>69</b>	YI2 (Second-Order Y Increment) Subroutine

---

<b>7</b>	<b>70</b>	<b>Coordinate Transformations</b>
	<b>71</b>	Algorithms
	<b>71</b>	Instructions
	<b>73</b>	Examples
	<b>76</b>	Programs
	<b>76</b>	SETX (Set up Transform) Main Program
	<b>77</b>	→NEW (Compute New Coordinates) Main Program
	<b>77</b>	→OLD (Compute Old Coordinates) Main Program
	<b>78</b>	XFMS (Transformation) Subroutine

---

<b>8</b>	<b>79</b>	<b>Curve Fitting</b>
	<b>79</b>	Algorithms
	<b>80</b>	Instructions
	<b>82</b>	Examples

---

<b>84</b>	Programs
<b>84</b>	CFU1 (Curve Fit 1) Utility
<b>85</b>	CFU2 (Curve Fit 2) Utility
<b>85</b>	LINC (Linear Curve Fit) Subroutine
<b>86</b>	EXPC (Exponential Curve Fit) Subroutine
<b>86</b>	PWRC (Power Curve Fit) Subroutine
<b>87</b>	LOGC (Logarithmic Curve Fit) Subroutine
<b>87</b>	PLOT (Curve Plot) Subroutine
<b>88</b>	POINT (Accumulate Point) Subroutine
<b>88</b>	DEL (Delete Point) Subroutine
<b>89</b>	BEST (Best Fit) Subroutine

---

<b>9</b>	<b>90</b>	<b>Triangle Solutions</b>
	<b>91</b>	Algorithms
	<b>91</b>	SSS (3 Sides)
	<b>91</b>	ASA (2 Angles and Included Side)
	<b>91</b>	SAA (Side, Adjacent Angle, Opposite Angle)
	<b>92</b>	SAS (2 Sides and Included Angle)
	<b>92</b>	SSA (2 Sides and Adjacent Angle)
	<b>92</b>	Area
	<b>92</b>	Instructions
	<b>94</b>	Examples
	<b>97</b>	Programs
	<b>97</b>	TU1 (Triangle 1) Utility
	<b>98</b>	TU2 (Triangle 2) Utility
	<b>98</b>	TU3 (Triangle 3) Utility
	<b>99</b>	Computation Subroutines
	<b>99</b>	→SSS (Compute SSS) Subroutine
	<b>100</b>	→ASA (Compute ASA) Subroutine
	<b>100</b>	→SAA (Compute SAA) Subroutine
	<b>101</b>	→SAS (Compute SAS) Subroutine
	<b>101</b>	→SSA1 (Compute SSA) Subroutine 1
	<b>102</b>	→SSA2 (Compute SSA) Subroutine 2
	<b>102</b>	Main Programs
	<b>102</b>	SSS (Side-Side-Side) Main Program
	<b>103</b>	ASA (Angle-Side-Angle) Main Program
	<b>103</b>	SAA (Side-Angle-Angle) Main Program

<b>104</b>	SAS (Side-Angle-Side) Main Program
<b>104</b>	SSA1 (Side-Side-Angle) Main Program 1
<b>105</b>	SSA2 (Side-Angle-Side) Main Program 2
<b>105</b>	AREA (Triangle Area)





# How To Use This Book

---

Please take a moment to familiarize yourself with the formats used in this book.

**Keys and Menu Selection.** A box represents a key on the calculator keyboard.

ENTER  
1/x  
STO  
  
ARRAY  
PLOT  
ALGBRA

In many cases, a box represents a shifted key on the calculator. In the example problems, the shift key is NOT explicitly shown. (For example, ARRAY requires the press of the shift key, followed by the ARRAY key, found above the "A" on the left keyboard.)

The "inverse" highlight represents a menu label:

## Key:

DRAW  
ISOL  
ABCD

## Description:

Found on the PLOT menu.

Found in the SOLV menu.

A user-created name. If you created a variable by this name, it could be found in either the

USER menu or the SOLVR menu.

menu. If you created a program by this name, it would be found in the **USER** menu.

Menus typically include more menu labels than can be displayed above the six, redefinable menu keys. Press **NEXT** and **PREV** to roll through the menu options. For simplicity, **NEXT** and **PREV** are not shown in the examples.

To solve for a user variable within **SOLVR**, press the shift key, followed by the appropriate user-defined menu key.

The symbol **<>** indicates the cursor-menu key.

Some keys cause the display to show different characters than are on the key.

Key:	Display:
<b>x</b>	*
<b>÷</b>	/
<b>1/x</b>	INV
<b>x<sup>2</sup></b>	SQ
<b>d/dx</b>	∂
<b>√x</b>	√

**Interactive Plots and the Graphics Cursor.** Coordinate values you obtain from plots using the **INS** and **DEL** digitizing keys may differ from those shown, due to small differences in the positions of the graphics cursor. The values you obtain should be satisfactory for the Solver root-finding that follows.

**Display Formats and Numeric Input.** Negative numbers, displayed as

```
-5
-12345.678
[ [-1, -2, -3 [-4, -5, -6 [ ...
```

are created using the `CHS` key.

```
5 CHS  
1 2 3 4 5 . 6 7 8 CHS  
[ [ 1 CHS , 2 CHS , ...
```

The examples in this book typically display a format for the number of decimal places. If your display is set such that numeric displays do not match exactly, you can modify your display format with the `MODE` menu and the `FIX` key within that menu. For example, to set the calculator to display two decimal places, set it to FIX2 mode by pressing `MODE` 2 `FIX`.

**Programming Reminders.** Before you key in the programs in this book, familiarize yourself with the locations of programming commands that appear as menu labels. By using the menu labels to enter commands, you can speed keying in programs and avoid errors that might arise from extra spaces appearing in the programs. Remember, the calculator recognizes commands that are set off by spaces. Therefore, the arrow ( $\rightarrow$ ) in the command `R $\rightarrow$ C` (the real to complex conversion function) is interpreted differently than the arrow in the command  `$\rightarrow$  C` (create the local variable "C").

The HP-28S automatically inserts spaces around each operator as you key it in. Therefore, using the `R`,  `$\rightarrow$` , and `C` keys to enter the `R $\rightarrow$ C` command will result in the expression `R  $\rightarrow$  C` and, ultimately, in an error in your program. As you key in programs on the HP-28S, take particular care to avoid spaces inside commands, especially in commands that include an  `$\rightarrow$` .

**User Menus.** If you do not purge all the programs and variables after working each example, or if your `USER` menu contains your own user-defined variables or programs, the `USER` menu on your calculator may differ from the displays shown in this book. Do not be concerned if the variables and programs appear in a slightly different order on your `USER` menu; this will not affect the calculator's performance.

---

# How This Book Is Organized

This solutions book presents several sets of programs to perform symbolic and numeric mathematical applications with the HP-28S calculator. In addition to performing useful calculations, the programs also serve as examples of HP-28S programming techniques.

Each set of programs is described in a separate chapter, with each chapter containing the following subsections:

- A brief description of the application.
- *Algorithms*. A listing of the principles and equations that form the mathematical basis of the programs. If the discussion is very brief, it is simply included in the application description.
- *Instructions*. Instructions for the use of the programs, including entering initial data and obtaining calculated results.
- *Examples*. Keystroke examples of the use of each of the programs.
- *Programs*. Listings of the programs.

## Program Entry and Listings

In the *Programs* sections, you will find several tabular program listings. A typical listing looks like this:

### PSUB (Polynomial Subtract) Main Program

For arguments  $\{ a_n \cdots a_0 \}$  and  $\{ b_n \cdots b_0 \}$ , PSUB returns  $\{ a_n - b_n \cdots a_0 - b_0 \}$ .

Arguments	Results
2: { coefficients 1 }	2:
1: { coefficients 2 }	1: { difference }

## Required Program:

- PADD

### Program:

```
<<
LIST→ → n
<< 1 n
START NEG n ROLL
NEXT
n →LIST PADD
>>
>>
```

### Comments:

n = number of terms.

Negate each term.

ENTER

'PSUB' STO

Store program.

The program listings are organized as follows:

- The first line contains the program name and the type of program. On the left is the name of the variable that you should use to store the program; in the parentheses is a spelled-out version of the name that suggests the use of the program. You can, of course, use any name that you choose to store the program. If you do use another name, remember to use the new name in any other programs which call it.

The program type is given after the program name and description and will be one of the following:

#### Main

A main program is intended for direct keyboard use, initiated by pressing the corresponding menu key. Many main programs use special input/output methods that make them unsuitable as subroutines for other programs.

#### Subroutine

A subroutine is a subprogram that performs a specific calculation. You may wish to use the subroutines for your own programming purposes.

#### Utility

A utility is a subroutine that is so specialized that it has little or no use other than within the programs that explicitly refer to it. Although they will appear in the USER menu, you will never have occasion to

execute them from the menu, so it is recommended that you order the menu so that they appear last. The utilities all have names ending in "U" and a number.

- After the program name may be some text describing the program.
- Next is a *stack diagram*. The diagram shows the arguments required on the left and the results returned by the program on the right. For a main program or a subroutine, the stack diagram is omitted if the program uses no stack arguments and returns nothing to the stack. For utilities, the stack diagrams are usually omitted because the programs are not intended for general use.
- If the program calls other programs, they are listed in a section called "Required Programs." These other programs must be keyed into the calculator before you can use the program being described. This section appears only if the program calls other programs.
- The remainder of the program table is a commented listing of the program steps. Each listing begins with the program delimiter « and ends with ». The comments at the right tell you what operations have been performed by the lines to the left.

## Entering Programs

The programs for each of the applications are listed at the end of each chapter. You may enter all of the programs into the HOME directory. However, we recommend that you take advantage of the HP-28S directory system to organize USER memory to provide easy access to the groups of programs. For example, in the HOME directory, create a directory named MATH to hold all of the programs listed in this book. Then, switch to the MATH directory, and create individual subdirectories for each application. Suggested names: TRI (triangle solutions), POLY (polynomial operations), DEQ (differential equations), XFM (coordinate transformations), and FIT (curve fitting). Enter all of the triangle programs into the TRI directory, the polynomial programs into the POLY directory, and so forth.

To enter a single program, enter the program commands and objects in the order shown in the program listing, left-to-right, top-to-bottom. The first item in each program is always the « program delimiter. You do



not have to use `NEWLINE` to break the program into lines; separate lines are used in the listings only for readability. When you have entered the complete program into the command line, press `ENTER`. If the program enters correctly, you will see the program in level 1. Then store the program in a variable *name*, using the syntax



`' name STO`

When you are entering a set of programs corresponding to one of the applications, you should enter the subroutines and utility programs first. Although this is not required, it has two advantages:

- The main programs will appear first in the final USER menu, followed by the subroutines and the utilities.
- You will be able to use the USER menu keys as typing aids when one program name is used in the definition of another program.

Most of the main programs are independent, so you can save memory by only entering the main programs you need for a specific purpose. For example, in the Curve Fitting application, you may only wish to use exponential curves, so you don't need to enter the programs for linear, power, and logarithmic curves.

## Using SYSEVAL

SYSEVAL is a special HP-28S command that enables the use of HP-28S system programs that are not accessible through ordinary commands. Two of the programs in this book (PCOEF and BEST) include the use of SYSEVAL. In all cases, the command SYSEVAL is preceded in a program by a binary integer (which specifies the address of the system program). These addresses are correct only for the HP-28S. Do not use them with any other calculator. All binary integers in this book are listed in hexadecimal. Be sure to include the "h" at the end of each digit string. As insurance, you can put the HP-28S in hexadecimal mode before you enter one of these programs. Just press **BINARY**  **HEX**  , then start entering the program.



### Warning

**When you key in a program, be very careful to enter the binary integer exactly as it is written. An incorrect number could result in your losing programs and data entered into the calculator's memory.**

---

# Polynomial Value and Coefficients

---

PCOEF converts an algebraic expression into a list of coefficients using an internal HP-28S routine that differentiates the polynomial.

PSERS converts a list of coefficients into a power series polynomial in a variable  $x$  as

$$a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0$$

PVAL converts a list of coefficients into a polynomial expression in a specified variable  $x$ , in Horner form

$$(\cdots (a_n x + a_{n-1}) x + a_{n-2}) x + \cdots) + a_0$$

PVAL and PSERS produce formally equivalent expressions, but the Horner form returned by PVAL is more efficient for continued evaluation.

---

## Instructions

### Conversion to List Form

Polynomials must be represented in a list form for most of the following operations. The list representing a polynomial has the following form:

$$\{ a_n \ a_{n-1} \ \cdots \ a_1 \ a_0 \}$$

where  $a_n$  is the coefficient of the  $n$ th-order term. Missing terms of order

less than  $n$  must be represented by zeros. For polynomial arithmetic,  $a_n$  can be real or complex numbers, or symbolics. For polynomial root finding,  $a_n$  must be real or complex numbers with  $n \leq 4$ .

To convert a polynomial from an HP-28S algebraic expression to a list, you can use one of the following:

- Inspection--enter by hand a list of coefficients derived from visual inspection of the algebraic.
- PCOEF. The stack should be configured as follows:

3: 'algebraic'  
 2: 'name'  
 1: order

where *algebraic* is to be treated as a polynomial in the variable *name*, and *order* is the highest power of *name* present. The result list is returned to the stack. If *order* is greater than the actual order of the polynomial, the list will contain leading zeros; if it is less, only coefficients of the terms up to *order* will be returned in the list.



Names in the original object *algebraic* (except for the polynomial variable *name*) are evaluated during execution of PCOEF. To prevent substitution of current values for those names, you should purge the corresponding variables.

## Conversion to Polynomial Value

After an operation is complete, you can convert the list representation of the result into an algebraic expression or a numerical value.

1. Place the coefficient list in level 2 and the variable name or value in level 1:

2: {  $a_n \cdots a_0$  }  
 1:  $x$

2. Execute PVAL. PVAL returns the value of the polynomial

$$(\cdots (a_n x + a_{n-1}) x + a_{n-2}) x + \cdots + a_0$$

If  $x$  is symbolic, the result is an algebraic expression; if  $x$  is numeric, and the coefficients  $a_n$  are numeric, the result will be a real or complex number.

## Entering Coefficients

Enter the coefficients of the polynomial as a list containing from two to five real or complex numbers. The number of elements in the list determines the order of the polynomial:

linear {  $a_1$   $a_0$  }

quadratic {  $a_2$   $a_1$   $a_0$  }

cubic {  $a_3$   $a_2$   $a_1$   $a_0$  }

quartic {  $a_4$   $a_3$   $a_2$   $a_1$   $a_0$  }

- If you wish to save the polynomial coefficients for later use, store the list in a variable, for example,  $\boxed{\boxed{\boxed{\text{DUP}}}} \text{ 'COEF' } \boxed{\boxed{\boxed{\text{STO}}}}$ .
- Press  $\boxed{\boxed{\boxed{\text{PROOT}}}}$ . PROOT returns one, two, three, or four real or complex roots to the stack, depending on the order of the polynomial.
- To evaluate the polynomial at a particular value of the independent variable, enter the coefficient list then the variable value (list in level 2, variable value in level 1), then press  $\boxed{\boxed{\boxed{\text{PVAL}}}}$  or  $\boxed{\boxed{\boxed{\text{PSERS}}}}$ . PVAL and PSERS will accept symbolic coefficients and variable values, as well as real and complex numbers.

## Examples

**Example 1.** Convert the algebraic expression

' $A \cdot X^4 + B \cdot X^3 + C \cdot X^2 - 10$ ' to a list of coefficients.

This optional step prevents substitutions for  $A$ ,  $B$ , and  $C$ , so your results will match this example.

{ A B C } PURGE

Enter the algebraic expression.

CLEAR

' $A \cdot X^4 + B \cdot X^3 + C \cdot X^2 - 10$ '

ENTER

2:	
1:	'A*X^4+B*X^3+C*X^2-10'
	<span style="border: 1px solid black; padding: 2px;">PCOEF</span> <span style="border: 1px solid black; padding: 2px;">PDIV</span> <span style="border: 1px solid black; padding: 2px;">PMUL</span> <span style="border: 1px solid black; padding: 2px;">PSUB</span> <span style="border: 1px solid black; padding: 2px;">PADD</span>

Enter the variable name and order of the expression.

'X' 4 ENTER

3:	'A*X^4+B*X^3+C*X^2-10'
2:	X
1:	4
	<span style="border: 1px solid black; padding: 2px;">PCOEF</span> <span style="border: 1px solid black; padding: 2px;">PDIV</span> <span style="border: 1px solid black; padding: 2px;">PMUL</span> <span style="border: 1px solid black; padding: 2px;">PSUB</span> <span style="border: 1px solid black; padding: 2px;">PADD</span>

Convert the expression.

USER PCOEF

3:	
2:	
1:	( A B C 0 -10 )
	<span style="border: 1px solid black; padding: 2px;">PROOT</span> <span style="border: 1px solid black; padding: 2px;">PSERS</span> <span style="border: 1px solid black; padding: 2px;">PV</span> <span style="border: 1px solid black; padding: 2px;">PCOEF</span> <span style="border: 1px solid black; padding: 2px;">PSUB</span> <span style="border: 1px solid black; padding: 2px;">PDIV</span>

**Example 2.** Convert the result of the previous example into a Horner form polynomial expression in 'Z'.

If { A B C 0 -10 } is not already on the stack:

{ A B C 0 -10 }

ENTER

3:	
2:	
1:	( A B C 0 -10 )
	<span style="border: 1px solid black; padding: 2px;">QUAR</span> <span style="border: 1px solid black; padding: 2px;">CUBIC</span> <span style="border: 1px solid black; padding: 2px;">QUD</span> <span style="border: 1px solid black; padding: 2px;">PROOT</span> <span style="border: 1px solid black; padding: 2px;">PSERS</span> <span style="border: 1px solid black; padding: 2px;">PVAL</span>

Perform the conversion.

'Z' [USER] [PVAL]

```
2:
1: '(A*Z+B)*Z+C)*Z*Z-
10'
[QUAR] [CUBIC] [QUO] [PROOT] [PSERS] [PVAL]
```

**Example 3.** Convert the result of example 1 into a power series in Z.

Enter { A B C 0 -10 } if it is not already on the stack:

{ A B C 0 -10 }

[ENTER]

```
3:
2:
1: { A B C 0 -10 }
[QUAR] [CUBIC] [QUO] [PROOT] [PSERS] [PVAL]
```

Perform the conversion.

'Z' [USER] [PSERS]

```
2:
1: 'A*Z^4+B*Z^3+C*Z^2-
10'
[QUAR] [CUBIC] [QUO] [PROOT] [PSERS] [PVAL]
```



---

# Programs

## PCOEF (Polynomial Coefficients) Main Program

Arguments	Results
3: <i>'expression'</i>	3:
2: <i>'variable name'</i>	2:
1: <i>n</i>	1: $\{a_n \cdots a_0\}$

### Program:

BINARY HEX

```
<<
3 DUPN TYPE SWAP
TYPE ROT
TYPE 3 →LIST { 0 6 9 }
IF ==
THEN DUP 1 + → n
<< #C53Bh SYSEVAL
SWAP
#33017h SYSEVAL
#59B5h SYSEVAL
#38104h SYSEVAL
DROP #5973h SYSEVAL
1 n
FOR m m ROLL COLCT
NEXT
n →LIST
>>
END
>>
```

ENTER

'PCOEF' STO

### Comments:

Check argument types.

Save  $n+1$  as  $n$ .

Put coefficients on stack.

Collect terms and reverse coefficients.

Make coefficient list.

Store the program.

PVAL (Polynomial Value) Main Program

Arguments	Results
2: { coefficients } 1: value name	2: 1: polynomial value

Program:

```
<<
→ st x
<< st 1 GET 2 st SIZE
FOR n x * st n GET +
NEXT
>> >>
```

ENTER

' PVAL'    STO

Comments:

Save list and value as temporary variables.  
Set up iterations.  
Multiply each term by  $x$  and add  $a_i$ .

Store the program.

PSERS (Polynomial Series) Main Program

Arguments	Results
2: { coefficients } 1: value name	2: 1: polynomial value

Program:

```
<<
→ x
<< LIST→ 0 SWAP 1
FOR n n 1 + ROLL
x n 1 - ^ *
+
-1 STEP
>> >>
```

ENTER

' PSERS'    STO

Comments:

Save value as a temporary variable.  
Get the  $n$ th coefficient.  
 $a_n x$   
Add to the expression.  
Iterate.

Store the program.

## Polynomial Arithmetic

---

This set of programs provides simple arithmetic (add, subtract, multiply, divide) and root finding for polynomial expressions. For speed of operation, polynomials are represented by lists of the form

$$\{ a_n \ a_{n-1} \ \dots \ a_1 \ a_0 \}$$

where  $a_n$  is the coefficient of the  $n$ th-order term. Missing terms of order less than  $n$  must be represented by zeros.

- PADD adds polynomials of arbitrary order.
- PSUB subtracts polynomials of arbitrary order.
- PMUL multiplies polynomials of arbitrary order.
- PDIV performs synthetic division on polynomials of arbitrary order, returning the quotient and the remainder.

To add, subtract, multiply, or divide two polynomials, place the two polynomial lists in levels 1 and 2, and execute PADD, PSUB, PMUL, or PDIV, respectively. For the first three programs, the result list is returned to level 1. For PDIV, three lists are returned:

$$\begin{array}{l} 3: \{ \textit{quotient} \} \\ 2: \{ \textit{remainder} \} \\ 1: \{ \textit{divisor} \} \end{array}$$

The *divisor* list is the original divisor from level 1.

---

## Instructions

### Conversion to List Form

Polynomials must be represented in a list form for most of the following operations. The list representing a polynomial has the following form:

$$\{ a_n \ a_{n-1} \ \dots \ a_1 \ a_0 \}$$

where  $a_n$  is the coefficient of the  $n$ th-order term. Missing terms of order less than  $n$  must be represented by zeros. For polynomial arithmetic, the  $a_n$  can be real or complex numbers, or symbolics.

To convert a polynomial from an HP-28S algebraic expression to a list, you can use one of the following:

- Inspection--enter by hand a list of coefficients derived from visual inspection of the algebraic.
- PCOEF. (The PCOEF program is defined in chapter 1.) The stack should be configured as follows:

3: '*algebraic*'  
2: '*name*'  
1: *order*

where *algebraic* is to be treated as a polynomial in the variable *name*, and *order* is the highest power of *name* present. The result list is returned to the stack. If *order* is greater than the actual order of the polynomial, the list will contain leading zeros; if it is less, only coefficients of the terms up to *order* will be returned in the list.



Names in the original object *algebraic* (except for the polynomial variable *name*) are evaluated during execution of PCOEF. To prevent substitution of current values for those names, you should purge the corresponding variables.

---

## Conversion to Polynomial Value

After an operation is complete, you can convert the list representation of the result into an algebraic expression or a numerical value.

1. Place the coefficient list in level 2 and the variable name or value in level 1:

$$\begin{array}{l} 2: \{ a_n \cdots a_0 \} \\ 1: x \end{array}$$

2. Execute PVAL or PSERS. (Refer to chapter 1.) PVAL returns the value of the polynomial

$$(\cdots (a_n x + a_{n-1}) x + a_{n-2}) x + \cdots) + a_0$$

PSERS returns the value of the polynomial

$$(\cdots (a_n x^n + a_{n-1} x^{n-1} + \cdots) + a_0$$

If  $x$  is symbolic or if the coefficients are symbolic, the result is an algebraic expression; if  $x$  is numeric, and the coefficients  $a_n$  are numeric, the result will be a real or complex number.

## Entering Coefficients

Enter the coefficients of the polynomial as a list containing from two to five real or complex numbers. The number of elements in the list determines the order of the polynomial:

linear {  $a_1$   $a_0$  }

quadratic {  $a_2$   $a_1$   $a_0$  }

cubic {  $a_3$   $a_2$   $a_1$   $a_0$  }

quartic {  $a_4$   $a_3$   $a_2$   $a_1$   $a_0$  }

- If you wish to save the polynomial coefficients for later use, store the list in a variable, for example,  $\boxed{\boxed{\boxed{\text{DUP}}}} \text{ 'COEF' } \boxed{\boxed{\boxed{\text{STO}}}}$ .
- Press  $\boxed{\boxed{\boxed{\text{PROOT}}}}$ . PROOT returns one, two, three, or four real or complex roots to the stack, depending on the order of the polynomial.
- To evaluate the polynomial at a particular value of the independent variable, enter the coefficient list then the variable value (list in level 2, variable value in level 1), then press  $\boxed{\boxed{\boxed{\text{PVAL}}}}$  or  $\boxed{\boxed{\boxed{\text{PSERS}}}}$ . PVAL and PSERS will accept symbolic coefficients and variable values, as well as real and complex numbers.

## Examples

**Example 1.** Add the polynomials  $x^2 - 1$  and  $2x^3 + 4x + 3$ .

Enter the coefficients.

MODE STD  
 { 1 0 -1 ENTER  
 { 2 0 4 3 ENTER

```

3:
2:      ( 1 0 -1 )
1:      ( 2 0 4 3 )
PCOEF PDIV PMUL PSUB PADD
  
```

Add the polynomials.

USER PADD

```

3:
2:      ( 2 1 4 2 )
1:      ( 2 1 4 2 )
PCOEF PDIV PMUL PSUB PADD
  
```

The result is  $2x^3 + x^2 + 4x + 2$ .

**Example 2.** Subtract  $3x^3 + 1$  from  $2x^3 + x^2 + 4x + 2$ .

Enter the coefficients.

CLEAR  
 { 2 1 4 2 ENTER  
 { 3 0 0 1 ENTER

```

3:
2:      ( 2 1 4 2 )
1:      ( 3 0 0 1 )
PCOEF PDIV PMUL PSUB PADD
  
```

Subtract the polynomials.

USER PSUB

```

3:
2:      ( -1 1 4 1 )
1:      ( -1 1 4 1 )
PCOEF PDIV PMUL PSUB PADD
  
```

The result is  $-x^3 + x + 4x + 1$ .

**Example 3.** Multiply the polynomials  $x^2 - 1$  and  $2x^3 + 4x + 2$ .

Enter the multiplicand and multiplier.

CLEAR  
 { 1 0 -1 ENTER  
 { 2 0 4 2 ENTER

```

3:
2:      ( 1 0 -1 )
1:      ( 2 0 4 2 )
PCOEF PDIV PMUL PSUB PADD
  
```



Multiply the polynomials.

USER    PMUL

```
3:
2:
1:      { 2 0 2 2 -4 -2 }
PCOEF PDIV PMUL PSUB PADD
```

The result is  $2x^5 + 2x^3 + 2x^2 - 4x - 2$ .

**Example 4.** Compute  $x^2 + x + 1$  divided by  $x - 1$ .

Enter the dividend and divisor coefficients.

CLEAR  
{ 1 1 1 ENTER  
{ 1 -1 ENTER

```
3:
2:      { 1 1 1 }
1:      { 1 -1 }
PCOEF PDIV PMUL PSUB PADD
```

Divide the polynomials.

USER    PDIV

```
3:      { 1 2 }
2:      { 3 }
1:      { 1 -1 }
PCOEF PDIV PMUL PSUB PADD
```

The solution is  $x + 2$  with a remainder of  $3/(x - 1)$ .

---

# Programs

## PADD (Polynomial Add) Main Program

For arguments  $\{ a_n \cdots a_0 \}$  and  $\{ b_n \cdots b_0 \}$ , PADD returns the polynomial  $\{ a_n + b_n \cdots a_0 + b_0 \}$ .

Arguments	Results
2: { coefficients 1 }	2:
1: { coefficients 2 }	1: { sum-coefficients }

**Program:**

```
<<
DUP2 SIZE SWAP SIZE
IF > THEN SWAP END
LIST→ DUP 2 + ROLL
LIST→ DUP 2 + ROLL
→ g s

<< 1 g
FOR n n s
IF ≤
THEN g 1 + ROLL +

COLCT
END
g ROLL
NEXT
g →LIST
>>
>>
```

ENTER

' PADD'    STO

**Comments:**

Put longer list in level 2.

Dump lists on stack.  
Save lengths of lists as g (long) and s (short).

Add each pair of terms until short list is exhausted.

Store the program.

**PSUB (Polynomial Subtract) Main Program**

For arguments  $\{ a_n \cdots a_0 \}$  and  $\{ b_n \cdots b_0 \}$ , PSUB returns  $\{ a_n - b_n \cdots a_0 - b_0 \}$ .

Arguments	Results
2 : { <i>coefficients 1</i> }	2 :
1 : { <i>coefficients 2</i> }	1 : { <i>difference</i> }

**Required Program:**

- PADD

**Program:**

```

«
LIST→ → n
« 1 n
START NEG n ROLL
NEXT
n →LIST PADD
»
»

```

**Comments:**

n = number of terms.

Negate each term.

ENTER

'PSUB' STO

Store the program.

**PMULT (Polynomial Multiply) Main Program**

For arguments  $\{a_n \cdots a_0\}$  and  $\{b_m \cdots b_0\}$ , PMUL returns the polynomial  $\{c_k \cdots c_0\}$ , where

$$k = n + m$$

$$c_k = \sum_{i=0}^k a_i b_{k-i}.$$

Arguments	Results
2: { <i>multiplicand</i> }	2:
1: { <i>multiplier</i> }	1: { <i>product</i> }

**Required Program:**

- PADD

**Program:**

```

«
DUP SIZE ROT DUP
SIZE DUP 4 PICK
IF >

```

**Comments:**

```
THEN 4 ROLL 4 ROLL
END
```

Order as 4: long list; 3: long size; 2: short list; 1: short size.

```
→ g gs sh s
```

```
« { } 1 s
```

Start empty result list.

```
FOR ss g LIST→ 1
```

```
SWAP
```

```
START sh ss GET * gs
```

```
ROLL
```

```
NEXT
```

Multiply each element of L by the ss element of S.

```
1 s ss - DUP2
```

```
IF ≤
```

Justify the list.

```
THEN START 0
```

```
NEXT
```

```
ELSE DROP2
```

```
END
```

```
gs s ss - + →LIST
```

Add the list to the result list.

```
PADD
```

```
NEXT
```

```
»
```

```
»
```

ENTER

'PMULT' STO

Store the program.

## PDIV (Polynomial Divide) Main Program

For arguments  $A = a_n \cdots a_0$  and  $B = b_m \cdots b_0$ , PDIV returns a quotient polynomial  $C = c_{n-m} \cdots c_0$ , the remainder polynomial  $D = d_k \cdots d_0$ , and the original divisor  $B$ , such that  $B \cdot C + D = A$ .

The first term  $c_{n-m}$  is  $a_n/b_m$ . After this term is computed, a new dividend  $C'$  is computed, where  $C' = A - c_{n-m}B$ . The next quotient term  $c_{n-m-1}$  is then computed in the same manner from  $C'$ , and the process is repeated for each of the  $n$  terms of  $A$ . The remainder is the dividend that results at the end of the process.

Arguments	Results
3:	3: { <i>quotient</i> }
2: { <i>dividend</i> }	2: { <i>remainder</i> }
1: { <i>divisor</i> }	1: { <i>divisor</i> }

### Program:

```
«
DUP 1 GET OVER SIZE
→ d t n
```

```
« { } SWAP DUP SIZE
n - 1 +
1 SWAP
START
```

```
DUP 1 GET t / COLCT
ROT OVER 1 →LIST +
3 ROLL 1 n
FOR m
```

```
OVER m GET d m GET
3 PICK * - ROT
m ROT PUT SWAP
NEXT
DROP 2 100 SUB
NEXT d
»
»
```

ENTER

'PDIV' STO

### Comments:

Save divisor, top-order term of divisor, and number of divisor terms.

From 1 to  $n - m + 1$  ( $m$  = number of dividend terms) ...

Divide leading term by  $t$ .  
Append result to quotient list.

Subtract the result \* divisor from the dividend ...

Throw away the leading term.  
Repeat.

Store the program.

## Polynomial Root Finding

---

This set of programs provides root finding for polynomial expressions. For speed of operation, polynomials are represented by lists of the form

$$\{ a_n \ a_{n-1} \dots a_1 \ a_0 \}$$

where  $a_n$  is the coefficient of the  $n$ th-order term. Missing terms of order less than  $n$  must be represented by zeros.

PROOT computes all of the roots of a first-, second-, third-, or fourth-degree polynomial, of the form

$$a_4 z^4 + a_3 z^3 + a_2 z^2 + a_1 z + a_0$$

where  $z$  is the variable, and  $a_i$  are constant coefficients (real or complex). If  $a_4$  is zero, the polynomial is cubic, or third order; if  $a_4$  and  $a_3$  are zero, the polynomial is quadratic or second order.

---

## Algorithms

In the following discussions, the polynomials are assumed to have 1 as the coefficient of the highest order term. Any polynomial can be reduced to this form by dividing it by the highest order coefficient without changing the value of the roots.

## Linear Expressions

The form for a linear expression is  $z + a_0$ . The single root of a linear expressions is  $z = -a_0$ .

## Quadratic Polynomials

The form for a quadratic polynomial is  $z^2 + a_1 z + a_0$ . Quadratic polynomials can be solved for two roots by application of the quadratic formula:

$$z_{\pm} = -\frac{a_1}{2} \pm \left[ \left( \frac{a_1}{2} \right)^2 - a_0 \right]^{1/2}$$

## Cubic Polynomials

The form for cubic polynomials is  $z^3 + a_2 z^2 + a_1 z + a_0$ . The algorithm used for cubic polynomials is as follows:

1. Rewrite the polynomial as  $w^3 + aw + b$ , where

$$\begin{aligned} w &= z - \frac{a_2}{3} \\ a &= a_1 - \frac{a_2^2}{3} \\ b &= \frac{(2a_2^3 - 9a_2a_1 + 27a_0)}{27} \end{aligned}$$

2. Define

$$A = \left[ -\frac{b}{2} - \left( \frac{b^2}{4} - \frac{a^3}{27} \right)^{1/2} \right]^{1/3}$$

Then the solutions are

$$z_n = -\frac{a_2}{3} + Ae^{2\pi i n/3} - \frac{a}{3A} e^{-2\pi i n/3}$$

where  $n = 0, 1$ , and  $2$ . This can also be rewritten by defining

$$f = e^{2\pi i/3} = (-1, \sqrt{3})/2$$

$$B = -\frac{a}{3A}$$

Then the solutions become

$$z_1 = -\frac{a_2}{3} + A + B$$

$$z_2 = -\frac{a_2}{3} + fA + \frac{B}{f}$$

$$z_3 = -\frac{a_2}{3} + \frac{A}{f} + fB$$

For the case  $a = 0$ , the above solutions are valid with  $A = 0$  and  $B = (-b)^{\frac{1}{3}}$ .

## Fourth-Order Polynomials

The form for a fourth-order polynomial is  $z^4 + a_3 z^3 + a_2 z^2 + a_1 z + a_0$ . A fourth-order polynomial can be factored into

$$[z^2 + (A + C)z + (B + D)][z^2 + (A - C)z + (B - D)]$$



By expanding and comparing coefficients with the original expression, then

$$\begin{aligned}2A &= a_3 \\2B + A^2 - C^2 &= a_2 \\2AB - 2CD &= a_1 \\B^2 - D^2 &= a_0\end{aligned}$$

Solving these equations for the parameters  $A$ ,  $B$ ,  $C$ , and  $D$ , then

$$\begin{aligned}A &= \frac{a_3}{2} \\B &= \frac{y}{2} \\C &= \frac{AB - \frac{a_1}{2}}{D} \quad (*) \\D &= (B^2 - a_0)^{1/2}\end{aligned}$$

where  $y$  is a root of the cubic  $y^3 + b_2y^2 + b_1y + b_0 = 0$ :

$$\begin{aligned}b_2 &= -a_2 \\b_1 &= a_3a_1 - 4a_0 \\b_0 &= a_0(4a_2 - a_3^2) - a_1^2\end{aligned}$$

$C$  can also be expressed as  $(A^2 - a_2 + y)^{1/2}$ , but this introduces a sign ambiguity. The sign of the square root must be chosen so that  $C$  satisfies the equation (\*) above. However, care must be taken when using (\*), since  $D$  can be zero, or close to zero in some cases. In the programs below, the three roots  $y$  are inspected to determine which gives the largest absolute value for  $D$ ; that value of  $y$  is chosen for the remainder of the calculation. This leaves the pathological case where all three roots are the same and all yield  $D = 0$ . In this case, the sign choice for  $C$  is unimportant, so the alternate definition of  $C$  (described at the beginning of this paragraph) is used. (Note that these programs do not detect cases where  $D$  should be zero but is not because of round-off error in the numerical computations.

You should always verify that the values returned are actually roots by evaluating the original polynomial.)

## Higher Than Fourth-Order Polynomials

No closed form solution exists for polynomials of order higher than four. For a fifth-order polynomial with real coefficients, at least one real root  $x_0$  must exist. Therefore, you can use the Solver to find  $x_0$ . By dividing the original fifth-order polynomial synthetically by  $x - x_0$ , you can obtain a fourth-order polynomial suitable for PROOT to find the remaining four roots.

---

# Instructions

## Conversion to List Form

Polynomials must be represented in a list form for most of the following operations. The list representing a polynomial has the following form:

$$\{ a_n \ a_{n-1} \ \dots \ a_1 \ a_0 \}$$

where  $a_n$  is the coefficient of the  $n$ th-order term. Missing terms of order less than  $n$  must be represented by zeros. For polynomial arithmetic,  $a_n$  can be real or complex numbers, or symbolics. For polynomial root finding,  $a_n$  must be real or complex numbers, with  $n \leq 4$ .

To convert a polynomial from an HP-28S algebraic expression to a list, you can use one of the following:

- Inspection--enter by hand a list of coefficients derived from visual inspection of the algebraic.
- PCOEF. (PCOEF is defined in chapter 1.) The stack should be configured as follows:

3: '*algebraic*'  
2: '*name*'  
1: *order*

where *algebraic* is to be treated as a polynomial in the variable *name*, and *order* is the highest power of *name* present. The result list is returned to the stack. If *order* is greater than the actual order of the polynomial, the list will contain leading zeros; if it is less, only coefficients of the terms up to *order* will be returned in the list.



### Note

Names in the original object *algebraic* (except for the polynomial variable *name*) are evaluated during execution of PCOEF. To prevent substitution of current values for those names, you should purge the corresponding variables.

---

## Conversion to Polynomial Value

After an operation is complete, you can convert the list representation of the result into an algebraic expression or a numerical value.

1. Place the coefficient list in level 2 and the variable name or value in level 1:

$$\begin{array}{l} 2: \{ a_n \cdots a_0 \} \\ 1: x \end{array}$$

2. Execute PVAL or PSERS. (Refer to chapter 1.) PVAL returns the value of the polynomial

$$(\cdots (a_n x + a_{n-1}) x + a_{n-2}) x + \cdots) + a_0$$

PSERS returns the value of the polynomial

$$(\cdots (a_n x^n + a_{n-1} x^{n-1}) + \cdots) + a_0$$

If  $x$  is symbolic, the result is an algebraic expression; if  $x$  is numeric, and the coefficients  $a_n$  are numeric, the result will be a real or complex number.

## Entering Coefficients

Enter the coefficients of the polynomial as a list containing from two to five real or complex numbers. The number of elements in the list determines the order of the polynomial:

linear  $\{ a_1 \ a_0 \}$

quadratic  $\{ a_2 \ a_1 \ a_0 \}$

cubic  $\{ a_3 \ a_2 \ a_1 \ a_0 \}$

quartic  $\{ a_4 \ a_3 \ a_2 \ a_1 \ a_0 \}$

- If you wish to save the polynomial coefficients for later use, store the list in a variable, for example,  $\boxed{\text{DUP}} \boxed{\text{' COEF '}} \boxed{\text{STO}}$ .
- Press  $\boxed{\text{PROOT}}$ . PROOT returns one, two, three, or four real or complex roots to the stack, depending on the order of the polynomial.
- To evaluate the polynomial at a particular value of the independent variable, enter the coefficient list then the variable value (list in level 2, variable value in level 1), then press  $\boxed{\text{PVAL}}$  or  $\boxed{\text{PSERS}}$ . PVAL and PSERS will accept symbolic coefficients and variable values, as well as real and complex numbers.

---

## Examples

**Example 1.** Find the roots of  $4x^4 - 8x^3 - 13x^2 - 10x + 22 = 0$ . Also, evaluate the polynomial at  $x = 2$ .

Set 4 FIX mode.

$\boxed{\text{MODE}} \boxed{4} \boxed{\text{FIX}}$

Save the coefficients in "COEF".

$\boxed{\{ 4 \ -8 \ -13 \ -10 \ 22 \}} \boxed{\text{ENTER}} \boxed{\text{' COEF '}} \boxed{\text{STO}}$

```
1: ( 4.0000 -8.0000
    -13.0000 -10.0000
    22.0000 )
COEF QUAR CUBIC QUD PROOT PSERS
```

Calculate roots.

USER  $\equiv$  PROOT  $\equiv$  <>

```
4: (0.8820,0.0000)
3: (-1.0000,-1.0000)
2: (3.1180,2.0000E-12)
1: (-1.0000,1.0000)
```

Real roots are in stack positions 4 and 2. Complex roots are in stack positions 3 and 1.

Return the list to the stack.

<>  $\equiv$  COEF  $\equiv$

```
1: ( 4.0000 -8.0000
   -13.0000 -10.0000
   22.0000 )
COEF QUAR CUBIC QUD PROOT PSERS
```

Evaluate the polynomial at  $x = 2$ .

2  $\equiv$  PVAL  $\equiv$

```
3: (3.1180,2.0000E-12)
2: (-1.0000,1.0000)
1: (-50.0000)
PVAL PCOEF PDIV PMUL PSUB PADD
```

The value at  $x = 2$  is  $-50$ .

Purge menu entries created by this example.

'COEF' PURGE

**Example 2.** Find the three cube roots of (0,8). The cube roots are the roots of the equation  $x^3 - (0,8) = 0$ .

Set 3 FIX mode.

MODE 3  $\equiv$  FIX  $\equiv$

Enter the coefficients and find the roots.

{ 1 0 0 (0 -8) }

USER  $\equiv$  PROOT  $\equiv$

```
3: (1.732,1.000)
2: (-1.732,1.000)
1: (-1.733E-12,-2.000)
QUAR CUBIC QUD PROOT PSERS PVAL
```

---

# Programs

## PROOT (Polynomial Root Finder) Main Program

Arguments	Results
4 :	4 : $z_4$
3 :	3 : $z_3$
2 :	2 : $z_2$
1 : { $a_n \cdots a_0$ }	1 : $z_1$

### Required Programs:

- QUD
- CUBIC
- QUAR

### Program:

```
<<
LIST→ →ARRY
DUP 1 GET /
ARRY→ LIST→ DROP 1
- DUP 2 + ROLL DROP
{NEG QUD CUBIC QUAR}
SWAP GET EVAL
>>
```

ENTER

'PROOT' STO

### Comments:

Put coefficients in a vector.  
Normalize coefficients.  
Polynomial order.  
Discard leading coefficient.  
Choices for various orders.  
Execute the appropriate choice.

Store the program.

The following programs perform the actual calculations of the polynomial roots after the polynomial is normalized and the order of the polynomial has been determined.

### QUD (Quadratic) Subroutine

Arguments	Results
2 : a <sub>1</sub> 1 : a <sub>0</sub>	2 : z <sub>2</sub> 1 : z <sub>1</sub>

**Program:**

```

«
SWAP 2 / NEG
DUP SQ ROT - √
DUP2 + 3 ROLL -
»

```

```

[ENTER]
'QUD' [STO]

```

**Comments:**

$$\begin{aligned}
 &-a_1/2 \\
 &[(a_1/2)^2 - a_0]^{1/2}
 \end{aligned}$$

Store the program.



## CUBIC (Cubic Solve) Subroutine

Arguments	Results
3: $a_2$	3: $z_1$
2: $a_1$	2: $z_2$
1: $a_0$	1: $z_3$

### Program:

### Comments:

```

«
3 PICK -3 /           - $a_2/3$ 
3 PICK 5 PICK SQ 3 /   $a$ 
-
5 ROLL DUP 3 ^ 2 *
SWAP 9 * 6 ROLL * -
27 / 4 ROLL + NEG       $b$ 
OVER ABS 0
IF ==                  If  $a = 0...$ 
THEN 3 INV ^           ...then  $B = (-b)^{1/3}$ 
SWAP DROP 0 SWAP       and  $A = 0$ .
ELSE 2 / DUP SQ 3
PICK 3 ^ 27 / +
 $\sqrt{-3}$  INV ^           $A$ 
SWAP OVER / 3 / NEG     $B$ 
END
-1 3  $\sqrt{\phantom{x}}$  R→C 2 /     $f$ 
4 ROLLD 3 DUPN 3        $z_1$ 
DUPN + +
8 ROLLD 7 PICK *
SWAP 7                  $z_2$ 
PICK / + +
5 ROLLD 4 PICK /
SWAP 4 ROLL * + +       $z_3$ 
»

```

ENTER

'CUBIC' STO

Store the program.

QUAR (Quartic Solver) Subroutine

Arguments	Results
4: a <sub>3</sub>	4: z <sub>1</sub>
3: a <sub>2</sub>	3: z <sub>2</sub>
2: a <sub>1</sub>	2: z <sub>3</sub>
1: a <sub>0</sub>	1: z <sub>4</sub>

Required Programs:

- QUD
- CUBIC

Program:

```
<<
3 PICK NEG DUP 6
ROLLD
5 PICK 4 PICK * 3
PICK 4 * -
5 ROLL 4 * 6 PICK
SQ - 4 PICK
* 5 PICK SQ -
CUBIC 3 →LIST
1 3

FOR n
DUP n GET 2 / SQ n
2 + PICK - ABS SWAP
NEXT 4 ROLLD DUP2
IF <
THEN SWAP DROP 3
ELSE DROP 2
END 3 ROLLD
IF >
THEN DROP 1
END GET
4 ROLL 2 /
SWAP 2 /
```

Comments:

$b_2$

$b_1$

$b_0$

Combine three roots in a list.  
Find the root  $y_0$  that maximizes  
 $| (y/2)^2 - a_0 |$ .

$y_0$

$A$

$B$

```

DUP SQ 4 ROLL -  $\sqrt{\quad}$   $D$ 
IF DUP ABS
THEN 3 DUPN 3 ROLLD
* 6 ROLL 2 / - SWAP  $C$ 
/ ELSE 3 PICK SQ 6
ROLL + 3 PICK 2 * +  $\sqrt{\quad}$ 
END 5 ROLL DROP
3 ROLLD 4 DUPN +
3 ROLLD + SWAP QUD  $z_1, z_2$ 
6 ROLLD 6 ROLLD -
3 ROLLD - SWAP QUD  $z_3, z_4$ 
»

```

ENTER

'QUAR' STO

Store the program.

## Symbolic Solutions to Differential Equations

---

The program SDEQ finds a symbolic solution to a second-order differential equation of the form

$$y'' + a y' + b y = f_0(x) + f_1(x) + \cdots + f_n(x)$$

where  $y$  is the unknown function of  $x$ ,  $a$  and  $b$  are constants,  $f_0$  is a polynomial in  $x$ , and the other  $f_i$  are the products of a polynomial with an exponential.

---

### Algorithms

To obtain a general solution to a second-order differential equation of the form listed above, use the following facts:

1. The left side of the equation can be represented as

$$(D + r_1)(D + r_2)y$$

where  $D$  is the differential operator  $d/dx$ , and  $r_1$  and  $r_2$  are the negatives of the roots of the quadratic equation

$$r^2 + ar + b = 0$$

2. The general solution of the equation

$$(D + r)y = f(x)$$

is given by

$$e^{-rx} \int_c^x f(u) e^{ru} du$$

where  $c$  is an arbitrary constant. Iterating this yields the general

solution of  $(D + r_1)(D + r_2)y = f(x)$ , for any  $f$ .

In the class of problems considered here, we can simplify the process of computing the solutions by making the following observations:

1. In the homogeneous case ( $f_i = 0$ ) the integral formula yields solutions

$$A e^{-r_1 x} + B e^{-r_2 x} \quad \text{for } r_1 \neq r_2$$

or

$$(Ax + B) e^{-r x} \quad \text{for } r_1 = r_2 = r$$

where  $A$  and  $B$  are arbitrary constants.

2. Acting on a polynomial of degree  $N$ , the differential operator  $(D + r)$  for nonzero  $r$  has an inverse given by

$$(D + r)^{-1} = \frac{1}{r} - \frac{1}{r^2} D + \frac{1}{r^3} D^2 - \dots + \frac{(-1)^N}{r^{N+1}} D^N$$

For  $r = 0$ , the inverse is just the integral of the polynomial. Thus we can effectively solve  $(D + r_1)(D + r_2)y = f$  when  $f$  is a polynomial by using the above inverse operation:

$$y = (D + r_2)^{-1}(D + r_1)^{-1}f$$

plus an arbitrary solution of the homogeneous equation.

3. The operators  $(D + r)$  almost commute with multiplication by  $e^{kx}$ . In particular, for any  $y$ ,

$$(D + r)(e^{kx} y) = e^{kx} (D + (r + k))y.$$

Solving

$$(D + r_1)(D + r_2)y = f(x) e^{kx}$$

amounts to solving

$$(D + (r_1 + k))(D + (r_2 + k))y = f(x)$$

for  $y$  and multiplying the result by  $e^{kx}$ .

The program SDEQ uses these techniques to solve differential equations of the form given in the introduction in a fairly efficient manner.

---

## Instructions

To solve the second-order differential equation

$$y'' + a y' + b y = f_0(x) + f_1(x) + \cdots + f_n(x)$$

execute SDEQ with two arguments, as follows:

1. The level 2 argument is an algebraic object that represents the left side of the differential equation. The object is to be a quadratic expression in the variable  $D$ , which represents the differential operator, for example  $'D^2 + A * D + B'$ .
2. The level 1 argument is a list containing successive groups of three objects ("triplets"), which together represent the right side of the equation. Each of the triplets consists of, in order:
  - A polynomial in the variable  $X$ , representing the polynomial part of the summand.
  - A number indicating the degree of the polynomial.
  - The characteristic exponent of the exponential factor of the summand. (This is the variable  $k$  in the algorithm section.)

The program returns the general solution of the equation as an expression in the variable  $X$  using the variables  $C0$  and  $C1$  to represent the arbitrary constants.

The only restriction on formal (non-numeric) variables used in any of the arguments is that the degree of the polynomial should be a number. If formal arguments are used, the program may not catch the special cases (when  $r_1 = r_2$ , or  $r_1 = 0$  or  $r_2 = 0$ ) since it may not be able to detect them. This should not be a problem in most cases, however.



### Note

Before running the next examples, make sure flag 34 is clear. Otherwise, you will not get the solutions as shown.

---

## Examples

**Example 1.** Solve  $y'' + 3y' + 2y = x^2 + x e^{-x}$ .

This equation is suitable for symbolic solution by SDEQ. The right side is a polynomial  $f_0 = x^2$  plus a second term  $f_1$  that is a polynomial  $x$  multiplied by the exponential  $e^{-x}$ .

Enter the left side of the equation.

CLEAR  
'D^2+3\*D+2'  
ENTER

```
3:
2:
1: 'D^2+3*D+2'
PART1 COL# SDEQ
```

Enter the triplet for the first term on the right.

{ 'X^2' 2 0 }  
ENTER

```
3:
2:
1: ( 'X^2' 2 0 )
PART1 COL# SDEQ
```

Enter the triplet for the second term on the right.

{ X 1 -1 }  
ENTER

```
3:
2:
1: ( X 1 -1 )
PART1 COL# SDEQ
```

Combine the lists.

+

```
3:
2:
1: ( 'X^2' 2 0 X 1 -1 )
PART1 COL# SDEQ
```

Now solve the equation.

USER SDEQ <>

```
1: 'C0*EXP(-X)+C1*EXP(-
(2*X))+(1.75-1.5*X+
.5*X^2)+(1-X+.5*X^2)
*EXP(-X)'
```

It is equivalent to

$$c_0 e^{-x} + c_1 e^{-2x} + (1.75 - 1.5x + .5x^2) + (1 - x + .5x^2) e^{-x}$$

You can check this result by constructing the differential operator and applying it to the result. The differential operator corresponding to 'D^2+3\*D+2' can be created as the program DIFOP:

Enter the program DIFOP:

```
« → Y
« Y 'X' ∂ 'X' ∂ 3 Y 'X'
∂ * + 2 Y * + » »
[ENTER]
```

```
1: « → Y « Y 'X' ∂ 'X'
   ∂ 3 Y 'X' ∂ * + 2 Y
   * + » »
S1 Y12 DE2 Y11 DE1 HOM0
```

Store the program.

'DIFOP' [STO]

Although you could apply this to the entire expression above, it is easier and faster to check each piece separately.

Enter the first term.

'C0\*EXP(-X)'

[USER] [DIFOP]  
[ALGBRA] [COLCT]

```
3:
2: 'C0*EXP(X)+C1*EXP(-...
1: 0
COLCT EXPAN SIZE FORM ODSUB ENSUB
```

Enter the second term.

'C1\*EXP(-(2\*X))'

[USER] [DIFOP]  
[ALGBRA] [COLCT]

```
3: 'C0*EXP(X)+C1*EXP(-...
2: 0
1: 0
COLCT EXPAN SIZE FORM ODSUB ENSUB
```

Enter the third term.

'1.75-1.5\*X+.5\*X^2'

[USER] [DIFOP]  
[ALGBRA]  
[EXPAN] [EXPAN] [COLCT]

```
3: 0
2: 0
1: 'X^2'
COLCT EXPAN SIZE FORM ODSUB ENSUB
```



Enter the fourth term.

$$'(1-X+.5*X^2)*EXP(-X)'$$

USER DIFOP

ALGBRA

EXPAN

EXPAN

EXPAN

EXPAN

EXPAN

COLCT

```
3:
2:
1: 'X^2'
COLCT EXPAN SIZE FORM OBSUB EXSUB
```

Sum all the terms.

+ + +

```
3:
2:
1: 'X^2+EXP(-X)*X'
COLCT EXPAN SIZE FORM OBSUB EXSUB
```

The sum matches the original right side of the differential equation, demonstrating the validity of the solution.

**Example 2.** Solve  $y'' - y = 2 \sin 2x$ .

To cast this equation into a form suitable for SDEQ, you must rewrite it in terms of exponentials:

$$y'' - y = -i e^{2ix} + i e^{-2ix}.$$

Enter the left side.

CLEAR

'D^2-1'

ENTER

```
3:
2:
1: 'D^2-1'
DIFOP S1 YI2 DE2 YI1 DE1
```

Enter the triplets.

{ (0, -1) 0 (0, 2)

(0, 1) 0 (0, -2) }

ENTER

```
2:
1: { (0, -1) 0 (0, 2)
(0, 1) 0 (0, -2) }
DIFOP S1 YI2 DE2 YI1 DE1
```

Finally, solve the equation.

USER SDEQ <>

```
1: 'C0*EXP(X)+C1*EXP(-X)
+(0,.2)*EXP((0,2)*X)
+(0,-.2)*EXP((0,-2)
*X)'
```

This solution is equivalent to  $c_0 e^x + c_1 e^{-x} - \frac{2}{5} \sin 2x$ .

Purge the menu entries created by this example.

```
{ 's1' 'DIFOP' PURGE
```

---

## Programs

### SDEQ (Symbolic Differential Equation) Program

The program uses global variables  $X$ ,  $D$ ,  $C0$ ,  $C1$ , and  $s1$ .

Arguments	Results
2: <i>'quadratic'</i>	2:
1: <i>{ list }</i>	1: <i>'expression'</i>

#### Required Programs:

- HOMOG
- PART1
- PART2
- COLX

#### Program:

```
« → list
« { D X C0 C1 s1 }
PURGE
'D' QUAD
DUP 1 's1' STO EVAL
NEG
SWAP 's1' SNEG EVAL
NEG
→ r1 r2
« r1 r2 HOMOG
1 list SIZE
FOR I
list I I 2 + SUB
```

#### Comments:

- Save the arguments.
- Initialize the variables.
- Solve the quadratic.
- $r_1$
- $r_2$
- Save the roots.
- Compute the homogeneous solution.
- Repeat for each triplet.

```
LIST→ DROP
r1 r2 PART2
+
3 STEP
» » »
```

Put the triplet elements on the stack.  
 Compute the particular solution.  
 Add to the general solution.  
 Step to the next triplet.

```
[ENTER]
'SDEQ' [STO]
```

Store the program.

## COLX (Simplify a Polynomial in X) Utility

### Program:

```
« IF OVER TYPE 9 ==
THEN 2 + X SWAP
TAYLR
ELSE DROP
END »
```

```
[ENTER]
'COLX' [STO]
```

### Comments:

Make sure it's an algebraic  
 expression.  
 Simplify.

Store the program.

## PART1 (Particular Solution 1) Utility

### Program:

```
« → f n r
« IF r 0 SAME r
(0,0) SAME OR THEN
f '0 * X' + X n f
ELSE
```

```
f r / 1 n
FOR I
f 1 I
START
X ∂
NEXT
r I 1 + NEG ^
-1 I ^ * * +
NEXT
END
» »
```

ENTER

'PART1' STO

### Comments:

Save parameters.

Check for  $r=0$ .

Integrate the polynomial.

Otherwise, apply the inverse differential:

Store the program.

## PART2 (Particular Solution 2) Utility

### Required Programs:

- COLX
- PART1

### Program:

```

« → n k r1 r2
« n r1 k + PART1
n 1 + r2 k + PART1
n COLX
'EXP(k * X) ' EVAL *
» »

```

### Comments:

Save the parameters.  
 "Integrate" once.  
 Again.  
 Simplify.  
 Multiply by  $e^k x$ .

ENTER

'PART2' STO

Store the program.

## HOMOG (Homogeneous Solution) Utility

Arguments	Results
2: $r_1$ 1: $r_2$	2: 1: { solution }

### Program:

```

« → r1 r2
« IF r1 r2 SAME
THEN '(C0 * X + C1)
* EXP(-r1 * X) '
ELSE 'C0 * EXP(-r1 *
X) + C1 * EXP(-r2 *
X) '
END EVAL
» »

```

### Comments:

Save the roots.  
 Case for identical roots.  
 Different roots.

ENTER

'HOMOG' STO

Store the program.

## Numerical Solutions to First-Order Differential Equations

---

The program DE1 solves first-order differential equations by the fourth-order Runge-Kutta method.

---

### Algorithms

For a differential equation of the form  $y' = f(x, y)$ , the value of  $y$  at a nearby point  $x+h$  can be approximated from the value of  $y$  at  $x$ :

$$y(x+h) \approx y(x) + h(c_1 + 2c_2 + 2c_3 + c_4),$$

where

$$c_1 = hf(x, y)$$

$$c_2 = hf\left(x + \frac{h}{2}, y + \frac{c_1}{2}\right)$$

$$c_3 = hf\left(x + \frac{h}{2}, y + \frac{c_2}{2}\right)$$

$$c_4 = hf(x+h, y+c_3)$$

Values of  $y$  at successive points  $x, x+h, x+2h$ , etc., can be obtained by applying the above formulae iteratively, where after each approximation  $y(x+h)$  replaces  $y$ , and  $x+h$  replaces  $x$ . Choosing a small  $h$  minimizes the accumulated error.

---

## Instructions

1. Define  $f(x, y)$ .  $f$  must be entered as a procedure that takes two arguments,  $x$  and  $y$ , in that order (level 2 and level 1, respectively, if the procedure is a program). The procedure must be stored in the variable  $F$ .

The most straightforward method of defining  $F$  is to use the user-defined function syntax

$$\ll \rightarrow x \ y \ 'f(x, y)' \gg,$$

where  $f$  is entered as an algebraic expression.

2. Press  $\equiv$  DE1  $\equiv$ . The HP-28S will halt and display an input menu showing menu keys for  $X$ ,  $Y$ , and  $H$ .
3. Enter initial values for  $x$ ,  $y$ , and the step size  $h$  by placing each value in level 1 and then pressing the corresponding menu key.
  - You can enter the values in any order.
  - You do not have to enter a value for any of the prompted quantities if you want to preserve the current value.
  - To recall the current value of any of the variables, press  $\square$ , followed by the appropriate menu key, then  $\square$  EVAL  $\square$ .
  - If you leave this menu, you can return by pressing  $\square$  CUSTOM  $\square$ .
4. Press  $\square$  CONT  $\square$ . This displays the user menu and returns the point  $(x+h, y(x+h))$ , expressed as a complex number object. The menu now contains a key for the program INCR.
5. Press  $\equiv$  INCR  $\equiv$  to compute the next point  $(x+h, y(x+h))$ . Each successive press of  $\equiv$  INCR  $\equiv$  increments  $x$  by  $h$  and returns the next point  $(x, y)$ .

## Example

Solve numerically the first-order differential equation

$$y' = \frac{\sin x + \tan^{-1}(y/x)}{y - \ln(\sqrt{x^2 + y^2})},$$

where  $x_0 = y_0 = 1$ .

The angular mode must be radians. Use a step size  $h = 0.5$ .

Define the function:

```
« → x y ' (SIN(x) +
ATAN(Y/X)) /
(Y-LN(√(SQ(X)+SQ(Y)))) '
»
```

ENTER

```
1: « → x y ' (SIN(x) +
  ATAN(y/x)) / (y-LN(√(
  SQ(x)+SQ(y)))) ' »
Y12 DE2 Y11 DE1 HOME PAGE
```

Store the function in  $F$ .

' F ' STO

Solve the equation for  $(x_1, y_1)$ .

```
MODE RAD
USER DE1
1 X
1 Y
.5 H
CONT
```

```
0:
1: (1.5000, 2.0553)
INCR H Y N F Y12
```

Solve for  $(x_2, y_2)$ .

INCR

```
0:
1: (1.5000, 2.0553)
2: (2.0000, 2.7780)
INCR H Y N F Y12
```



Solve for  $(x_3, y_3)$ .

≡ INCR ≡

3:	(1.5000, 2.0553)				
2:	(2.0000, 2.7780)				
1:	(2.5000, 3.2781)				
INCR	H	Y	X	F	YI2

and so on.

Purge the menu entries created by this example.

{ ' INCR ' ' H ' ' Y ' ' X ' ' F ' PURGE

---

## Programs

### DE1 (First-Order Differential Equation) Program

#### Required Program:

- YI1

#### Program:

```
<<
{STO X Y H} MENU
HALT
'YI1' ' INCR ' STO
YI1
23 MENU
>>
```

ENTER  
'DE1' STO

#### Comments:

Display the input menu.  
Set up INCR key for first order.  
Do the first step.  
Activate the USER menu.

Store the program.

## YI1 (First-Order Y Increment) Subroutine

Argument	Result
1:	1: $(x+h, y(x+h))$

### Program:

```

«
« DUP 2 / Y + H
2 / X + SWAP F H *
»
H X Y F *
OVER EVAL
ROT EVAL
DUP Y + X H + SWAP F
H *
OVER + + OVER + + +
6 /
'Y' STO+ H 'X' STO+
X Y R→C
»

```

### Comments:

This sequence will be executed twice.

$c_1$

$c_2$

$c_3$

$c_4$

$\Delta y$

The  $R \rightarrow C$  may be omitted if you prefer to have  $x$  and  $y$  returned as separate real numbers.

ENTER

'YI1' STO

Store the program.

## Numerical Solutions to Second-Order Differential Equations

---

The program DE2 solves second-order differential equations by the fourth-order Runge-Kutta method.

---

### Algorithms

The approach for a second-order differential equation, of the form  $y'' = f(x, y, y')$ , is similar to that for first-order equations. Here both  $y$  and  $y'$  are approximated incrementally:

$$y(x+h) = y(x) + h[y'(x) + (k_1 + k_2 + k_3)/6]$$

$$y'(x+h) = y'(x) + (k_1 + 2k_2 + 2k_3 + k_4)/6$$

where

$$k_1 = hf(x, y, y')$$

$$k_2 = hf\left(x + \frac{h}{2}, y + \frac{h}{2}y' + \frac{h}{8}k_1, y' + \frac{k_1}{2}\right)$$

$$k_3 = hf\left(x + \frac{h}{2}, y + \frac{h}{2}y' + \frac{h}{8}k_2, y' + \frac{k_2}{2}\right)$$

$$k_4 = hf\left(x + h, y + hy' + \frac{h}{2}k_3, y' + k_3\right)$$

## Instructions

1. Define  $f(x, y, y')$ .  $f$  must be entered as a procedure that takes three arguments,  $x$ ,  $y$ , and  $y'$ , in that order (level 3, level 2 and level 1, respectively, if the procedure is a program). The procedure must be stored in the variable  $F$ .

The most straightforward method of defining  $F$  is to use the user-defined function syntax

$$\ll \rightarrow x \ y \ Dy \ 'f(x, y, Dy)' \gg,$$

where  $f$  is entered as an algebraic expression. Here  $Dy$  is a local variable representing  $y'$ .

2. Press  $\boxed{\boxed{\boxed{DE2}}}$ . The HP-28S will halt and display an input menu showing menu keys for  $X$ ,  $Y$ ,  $DY$ , and  $H$ .
3. Enter initial values for  $x$ ,  $y$ ,  $y'$ , and the step size  $h$  by placing each value in level 1 and then pressing the corresponding menu key.
  - You can enter the values in any order.
  - You do not have to enter a value for any of the prompted quantities if you want to preserve the current value.
  - To recall the current value of any of the variables, press  $\boxed{'}\boxed{}$ , followed by the appropriate menu key, then  $\boxed{EVAL}$ .
  - If you leave this menu, you can return by pressing  $\boxed{CUSTOM}$ .
4. Press  $\boxed{CONT}$ . This displays the user menu and returns the point  $(x+h, y(x+h))$ , expressed as a complex number object. The menu now contains a key for the program INCR.
5. Press  $\boxed{\boxed{\boxed{INCR}}}$  to compute the next point  $(x+h, y(x+h))$ . Each successive press of  $\boxed{\boxed{\boxed{INCR}}}$  increments  $x$  by  $h$  and returns the next point  $(x, y)$ .

## Example

Solve the second-order equation  $(1-x^2)y'' + xy' = x$ , where  $x_0 = y_0 = y'_0 = 0$ , using a step size  $h = 0.1$ .

Define the function. Rewrite the equation as

$$y'' = \frac{x(y' - 1)}{x^2 - 1}.$$

Enter the function.

« → x y Dy 'x\*(Dy-1) /  
(SQ(x)-1)' »

ENTER

```
2:
1: « → x y Dy 'x*(Dy-1)
  / (SQ(x)-1)' »
VIB DE2 Y11 DE1 HOME PART2
```

Store the function in F.

'F' STO

Enter  $x_0, y_0, y'_0$ , and  $h$ . Solve the equation for  $(x_1, y_1)$ .

USER DE2

0 X

0 Y

0 DY

.1 H

CONT

```
3:
2:
1: (0.1000,0.0002)
INCR H DY Y X F
```

Solve for  $(x_2, y_2)$ .

INCR

```
3:
2: (0.1000,0.0002)
1: (0.2000,0.0013)
INCR H DY Y X F
```

Solve for  $(x_3, y_3)$ .

INCR

```
3: (0.1000,0.0002)
2: (0.2000,0.0013)
1: (0.3000,0.0046)
INCR H DY Y X F
```

Solve for  $(x_4, y_4)$ .

INCR

3:	(0.2000,0.0013)				
2:	(0.3000,0.0046)				
1:	(0.4000,0.0109)				
INCR	H	DY	Y	X	F

and so on.

Purge menu entries created by this example.

{ 'DY' 'INCR' 'H' 'Y' 'X' 'F' PURGE

---

# Programs

## DE2 (Second-Order Differential Equation) Program

### Required Program:

- YI2

#### Program:

<<  
{STO X Y DY H} MENU  
HALT  
'YI2' 'INCR' STO  
YI2  
23 MENU  
>>

ENTER  
'DE2' STO

#### Comments:

Display the input menu.  
Set up INCR key for second order.  
Do the first step.  
Activate the USER menu.

Store the program.

## YI2 (Second-Order Y Increment) Subroutine

Argument	Result
1:	1: $(x+h, y(x+h))$

### Program:

```

«
X Y DY F H *
H 2 / X OVER + Y 3
PICK
DY * + H 8 / DUP2
7 PICK * + 6 PICK 2
/
DY + 5 PICK 3 ROLL D k2
F H *
DUP 7 ROLL D * + 5
PICK 2 /
DY + F H * k3
SWAP OVER * DY H * +
Y + OVER DY +
X H + 3 ROLL D F H * Δy'
4 DUPN OVER + + +
OVER + + 6 / Δy
DY SWAP 'DY' STO+ Increment y.
5 ROLL D DROP + + Increment x.
6 / + H * (x + h, y(x + h))
'Y' STO+
H 'X' STO+
X Y R→C
»

```

ENTER

'YI2' STO

### Comments:

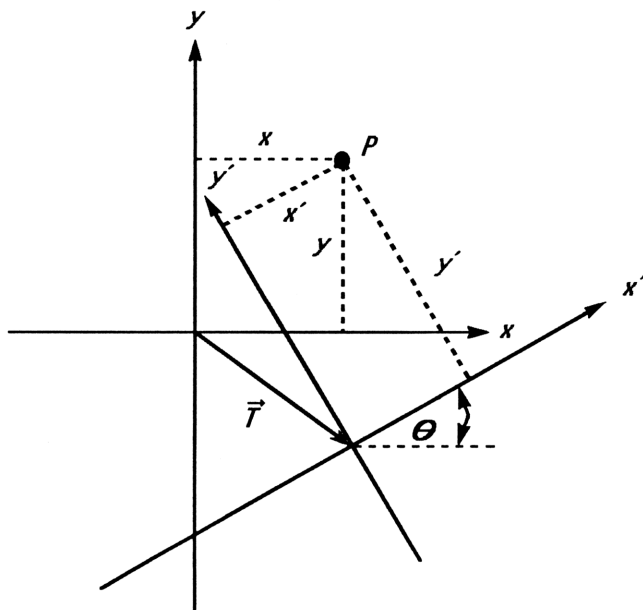
The R→C may be omitted if you prefer to have  $x$  and  $y$  returned as separate real numbers.

Store the program.

## Coordinate Transformations

These programs perform three-dimensional translation of coordinates, with or without rotation. A new coordinate system is displaced by the vector  $\vec{T}$  from the original system and rotated an angle  $\theta$  about the direction  $\vec{N}$ .  $\vec{T}$  and  $\vec{N}$  are specified in the original system. You can specify a vector  $\vec{P}$  in the original system and determine its value  $\vec{P}'$  in the new system, or vice-versa.

The following diagram illustrates the transformation for a two-dimensional system, where  $\vec{T}$  lies in the x-y plane, and  $\vec{N}$  is parallel to the z-axis (out of the paper):





---

## Algorithms

The programs implement the following expressions:

$$\vec{P}' = \left[ (\vec{P} - \vec{T}) \cdot \vec{N} \right] (1 - \cos \theta) \vec{N} + (\vec{P} - \vec{T}) \cos \theta + \left[ (\vec{P} - \vec{T}) \times \vec{N} \right] \sin \theta$$

The inverse relation is

$$\vec{P} = \left[ \vec{P}' \cdot \vec{N} \right] (1 - \cos \theta) \vec{N} + \vec{P}' \cos \theta - (\vec{P}' \times \vec{N}) \sin \theta + \vec{T}$$

If a flag  $\lambda$  is defined that has the value 1 (nonzero) for the forward transformation and the value 0 for the reverse, then these can be combined into a single equation

$$\begin{aligned} \vec{P}' = & \left[ (\vec{P} - \lambda \vec{T}) \cdot \vec{N} \right] (1 - \cos \theta) \vec{N} + (\vec{P} - \lambda \vec{T}) \cos \theta \\ & + (2\lambda - 1) \left[ (\vec{P} - \lambda \vec{T}) \times \vec{N} \right] \sin \theta + \vec{T} \tilde{\lambda} \end{aligned}$$

where  $\vec{P}$  ( $\vec{P}'$ ) represents the vector in the original (transformed) system or the transformed (original) system, according to whether  $\lambda$  has the value 1 or 0, respectively.  $\tilde{\lambda}$  means NOT  $\lambda$ .

---

## Instructions

Three main programs are provided:

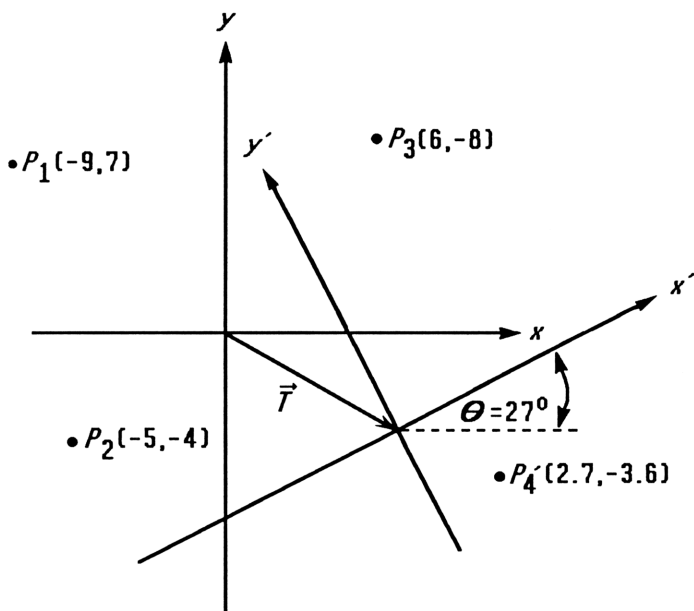
- SETX (Set Up Transform) prompts you for the values of  $\vec{T}$ ,  $\vec{N}$ , and  $\theta$ .
- $\rightarrow$ NEW converts a vector from the old (unprimed) system to the new (primed) system.
- $\rightarrow$ OLD converts a vector from the new system to the old.

To transform vectors:

1. Press  $\boxed{\boxed{\boxed{\text{SETX}}}}$ . This activates an input menu for  $T$ ,  $N$ , and  $\theta$ .
2. Enter initial values for  $\vec{T}$ ,  $\vec{N}$ , and  $\theta$  by placing each value in level 1 and pressing the corresponding menu key.  $\vec{T}$  and  $\vec{N}$  should be 3-element vectors;  $\theta$  a real number.
  - You can enter the values in any order.
  - You do not have to enter a value for any of the prompted quantities if you want to preserve the current value. If there is no current value, SETX will supply defaults of  $[0\ 0\ 0]$  for  $\vec{T}$  (no translation);  $[0\ 0\ 1]$  for  $\vec{N}$  (rotation about the third axis); and 0 for  $\theta$  (no rotation).
  - To recall the current value of any of the variables, press  $\boxed{[ ]}$ , followed by the appropriate menu key, then  $\boxed{\boxed{\boxed{\text{EVAL}}}}$ .
  - If you leave this menu, you can return by pressing  $\boxed{\boxed{\boxed{\text{CUSTOM}}}}$ .
3. Press  $\boxed{\boxed{\boxed{\text{CONT}}}}$ . This displays the user menu. ( $\vec{N}$  is normalized to a unit vector at this point.)
4. To transform a vector, enter the vector into stack level 1 as either a 2- or 3-element vector. Then perform one of the following:
  - If the vector is defined in the original (unprimed) system, press  $\boxed{\boxed{\boxed{\rightarrow\text{NEW}}}}$  to return its value in the new (primed) system.
  - If the vector is defined in the new (primed) system, press  $\boxed{\boxed{\boxed{\rightarrow\text{OLD}}}}$  to return its value in the old (unprimed) system.
5. For additional vectors in the same systems, you can use  $\boxed{\boxed{\boxed{\rightarrow\text{NEW}}}}$  and  $\boxed{\boxed{\boxed{\rightarrow\text{OLD}}}}$  any number of times without reentering the transformation parameters.
6. To change  $\vec{T}$  or  $\theta$ , press  $\boxed{\boxed{\boxed{\text{CUSTOM}}}}$  and use the input menu to enter the new values. You can also use the custom menu to enter a new  $\vec{N}$ , but be sure to normalize any new  $\vec{N}$  to a unit vector before storing it in  $N$ . ( $\boxed{\boxed{\boxed{\text{DUP}}}} \boxed{\boxed{\boxed{\text{ABS}}}} \boxed{\boxed{\boxed{[ ]}}}$  will normalize any vector.) SETX does this for you automatically.

# Examples

**Example 1.** The coordinate systems  $(x,y)$  and  $(x',y')$  are shown below:



Convert the points  $P_1$ ,  $P_2$ , and  $P_3$  to the  $(x',y')$  system. Convert the point  $P'_4$  to the  $(x,y)$  system.

Enter values for  $N$ ,  $THETA$ , and  $T$ .

```
MODE DEG
USER SETX
[ 0 0 1 ] N
27 THETA
[ 7 -4 0 ] T
CONT
```

```
3:
2:
1:
T THETA N XFMS OLD NEW
```

Convert  $P_1$  to  $P'_1$ .

```
[-9 7 ] ->NEW
```

```
3:
2:
1: [ -9.2622 17.0649 ]
T THETA N XFMS OLD NEW
```

Convert  $P_2$  to  $P_2'$ .

[ -5 -4 ] →NEW

3:	
2:	[ -9.2622 17.0649 ]
1:	[ -10.6921 5.4479 ]
T	THETA N SFMS →OLD →NEW

Convert  $P_3$  to  $P_3'$ .

[ 6 8 ] →NEW

3:	[ -9.2622 17.0649 ]
2:	[ -10.6921 5.4479 ]
1:	[ 4.5569 11.1461 ]
T	THETA N SFMS →OLD →NEW

Convert  $P_4'$  to  $P_4$ .

[ 2.7 -3.6 ] →OLD

3:	[ -10.6921 5.4479 ]
2:	[ 4.5569 11.1461 ]
1:	[ 11.0401 -5.9818 ]
T	THETA N SFMS →OLD →NEW

**Example 2.** A three-dimensional coordinate system is translated to (2.45, 4.00, 4.25). After translation, a  $62.5^\circ$  rotation occurs about the (0, -1, -1) axis. In the original system, a point had the coordinates (3.9, 2.1, 7.0). What are the coordinates of the point in the translated, rotated system?

Enter  $N$ ,  $\theta$ , and  $T$ .

CLEAR  
 MODE 3 FIX  
 USER SETX  
 [ 0 -1 -1 ] N  
 62.5 THETA  
 [ 2.45 4 4.25 ] T  
 CONT

3:	
2:	
1:	
T	THETA N SFMS →OLD →NEW



Convert the point.

[ 3.9 2.1 7 ] →NEW

3:	
2:	
1:	[ 3.586 0.261 0.589...
T	THETA N SFMS →OLD →NEW


In the translated, rotated system above, a point has the coordinates (1,1,1). What are the corresponding coordinates in the original system?

Convert the point.

[ 1 1 1  →OLD 

3:				
2:	[	3.586	0.261	0.589...
1:	[	2.912	4.373	5.877...
T	THETA	N	RFMS	→OLD →NEW

Purge the menu entries created by these examples.

{ 'T' 'THETA' 'N' 

---

## Programs

### SETX (Set up Transform) Main Program

#### Program:

```
<<
{ STO T N THETA }
MENU HALT
RCLF 31 SF
THETA
IF TYPE
THEN 0 'THETA' STO
END T
IF DUP TYPE 3 ≠
THEN DROP [0]
END
{3} RDM 'T' STO
N
IF DUP TYPE 3 ≠
THEN DROP [0 0 1]
ELSE DUP ABS INV *
END
'N' STO
STOF
23 MENU
>>
```

ENTER

'SETX' STO

#### Comments:

Prompt for  $\vec{T}, \vec{N}, \theta$ .  
Save flags, enable LAST.  
See if  $\theta$  has a value.  
If it's not a real number...  
...store default 0.

See if  $\vec{T}$  has a value.  
If it's not a vector...  
...replace with default.  
Expand to 3 dimensions if necessary.  
See if  $\vec{N}$  has a value.  
If it's not a vector...  
...replace with default.  
Normalize  $\vec{N}$ .

Restore flags.  
Restore USER menu.

Store the program.

## →NEW (Compute New Coordinates) Main Program

Argument	Result
1: $[x\ y\ z]$	1: $[x' \ y' \ z']$

### Required Program:

- XFMS

### Program:

« 1 SF XFMS »

ENTER  
'→NEW' STO

### Comments:

Set flag 1 for forward transform.

Store the program.

## →OLD (Compute Old Coordinates) Main Program

Argument	Result
1: $[x' \ y' \ z']$	1: $[x\ y\ z]$

### Required Program:

- XFMS

### Program:

« 1 CF XFMS »

ENTER  
'→OLD' STO

### Comments:

Clear flag 1 for inverse transform.

Store the program.

## XFMS (Transformation) Subroutine

Argument	Result
1: [xyz]	1: [x' y' z']

### Program:

```

«
1 FC?
SWAP DUP SIZE 3
ROLLD
{ 3 } RDM
OVER NOT T * -
DUP DUP THETA COS 1
OVER -
ROT * N DOT N * 4
ROLLD
* SWAP N CROSS THETA
5 PICK
-2 * 1 + * SIN
* + + SWAP T * +
SWAP RDM
»

```

### Comments:

$\tilde{\lambda}$   
 Make  $\vec{P}$  a three-element vector.  
 $\vec{P} - \lambda \vec{T}$   
 $\vec{P}$  or  $\vec{P}'$   
 Restore original dimensions.

ENTER

'XFMS' STO

Store the program.



## Curve Fitting

---

These programs collect statistical data points  $(x, y)$  and fit any or all of four curves to the data. The programs can also choose the curve of "best fit," defined as that curve fit with the highest coefficient of determination for the data.

The four curves are:

$$\text{Linear } y = a + bx$$

$$\text{Exponential } y = a e^{bx}$$

$$\text{Logarithmic } y = a + b (\ln x)$$

$$\text{Power } y = a x^b \quad (a > 0)$$

For a given curve, the program also computes predicted values of  $y$  for specified values of  $x$ .

The program requires all data points be in the domain  $x > 0$ . To fit a linear curve with points with  $x \leq 0$ , use LR.

---

## Algorithms

The regression coefficients  $a$  and  $b$  are found by solving the following linear equations:

$$A n + b \sum_{i=1}^n Y_i = \sum_{i=1}^n Y_i$$

$$A \sum_{i=1}^n x_i + b \sum_{i=1}^n X_i^2 = \sum_{i=1}^n X_i Y_i$$

where  $n$  is the number of points, and  $A$ ,  $X$ , and  $Y$  are defined by

Regression	A	$X_i$	$Y_i$
Linear	$a$	$x_i$	$y_i$
Exponential	$\ln a$	$x_i$	$\ln y_i$
Logarithmic	$a$	$\ln x_i$	$y_i$
Power	$\ln a$	$\ln x_i$	$\ln y_i$

The coefficient of determination  $r^2$  is defined by:

$$r^2 = \frac{A \sum Y_i + b \sum (X_i Y_i) - \frac{1}{n} (\sum Y_i)^2}{\sum Y_i^2 - \frac{1}{n} (\sum Y_i)^2}$$

---

## Instructions

1. Accumulate the data. Enter each data point as one of the following:
  - two real numbers ( $x$  in level 2 and  $y$  in level 1).
  - a vector of the form  $[x \ y]$ .
  - a complex number of the form  $(x \ y)$ .

Then press  $\equiv$  POINT  $\equiv$ . POINT accumulates the vectors  $[x \ y \ \ln(x)]$  in the statistics matrix  $\Sigma$  DAT.

2. To remove the most recent data point, press  $\equiv$  DEL  $\equiv$ , which returns a vector  $[x \ y]$  representing the most recent entry, while removing it from the statistics matrix.

**3. Fit a particular curve using either of the following methods:**

**a. Press**

- $\equiv \text{LINC} \equiv$   
for a linear curve fit,
- $\equiv \text{PWRC} \equiv$   
for a power curve fit,
- $\equiv \text{EXPC} \equiv$   
for an exponential curve fit, or
- $\equiv \text{LOGC} \equiv$   
for a logarithmic curve fit.

Each of these computes the regression coefficients  $a$  (variable  $A$ ) and  $b$  ( $B$ ) and the coefficient of determination  $r^2$  ( $RSQ$ ), and returns them (in that order) to the stack. In addition, the curve formula is encoded in the variable  $X \rightarrow Y$ . After calculation, the program displays labeled values of  $a$ ,  $b$ , and  $r^2$  in display lines 1, 2, and 3 respectively. The curve formula is displayed in line 4. The next keystroke will clear the output display, leaving the computed values on the stack.

**b. Press  $\equiv \text{BEST} \equiv$  to find the best fit curve.**

BEST executes each of the individual curve fit programs, and returns the coefficients for the curve that returns the highest coefficient of determination. The name of the best curve fit (LINC, EXPC, PWRC, or LOGC) is also displayed, temporarily overwriting the menu key display in line 4.

- 4. Once one or more curves have been fit, compute predicted values  $y = f(x)$ :** Enter  $x$  and press  $\equiv X \rightarrow Y \equiv$ . The predicted value of  $y$  is returned to level 1.
- 5. You can plot the data and the most recently fit curve:** Press  $\equiv \text{PLOT} \equiv$ .

# Examples

**Example 1.** Fit a straight line to the following set of data and compute predicted values of  $y$  for  $x = 37$  and  $x = 35$ .

x	y
40.5	104.5
38.6	102
37.9	100
36.2	97.5
35.1	95.5
34.6	94

Purge old  $\Sigma$ DAT.

STAT CLΣ

MODE 4 FIX

Accumulate data points and run curve fit program.

40.5 104.5 USER POINT  
38.6 102 POINT  
37.9 100 POINT  
36.2 97.5 POINT  
35.1 95.5 POINT  
34.6 94 POINT  
LINC

3:	33.5271				
2:	1.7601				
1:	0.9909				
EXPC	LINC	CFU2	CFU1		

Predicted value of  $x = 37$ .

37 X→Y

3:	1.7601				
2:	0.9909				
1:	98.6526				
A	E	RSC	ΣPWR	X→Y	ΣDAT

Predicted value of  $x = 35$ .

35 X→Y

3:	0.9909				
2:	98.6526				
1:	95.1323				
A	E	RSC	ΣPWR	X→Y	ΣDAT

**Example 2.** Find the curve that best fits the following set of data. Then compute predicted values of  $y$  for  $x = 1.5$  and  $x = 2$ .

$x$	$y$
0.72	2.16
1.31	1.61
1.95	1.16
2.58	0.85
3.14	0.50

Enter the data points and find the best curve.

```

STAT  CLΣ
.72 2.16  USER  POINT
1.31 1.61  POINT
1.95 1.16  POINT
2.58 .85  POINT
3.14 .5  POINT
BEST

```

```

3: 1.8515
2: -1.1021
1: 0.9893
'LOGC'

```

Compute the predicted value for  $x = 1.5$ .

```

1.5  X→Y

```

```

3: -1.1021
2: 0.9893
1: 1.4046
ΣDAT  A  B  RSD  ΣPAR  X→Y

```

Compute the predicted value for  $x = 2$ .

```

2  X→Y

```

```

3: 0.9893
2: 1.4046
1: 1.0875
ΣDAT  A  B  RSD  ΣPAR  X→Y

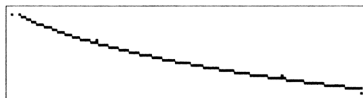
```

Plot the curve.

```

PLOT

```



Purge the menu entries created by this example.

```
{ 'PPAR' 'ΣDAT' 'A' 'B' 'RSQ' PURGE
```

---

## Programs

### CFU1 (Curve Fit 1) Utility

Arguments	Results
3: <i>m</i>	3: <i>a</i>
2: <i>n</i>	2: <i>b</i>
1: 'name'	1: $r^2$

**Program:**

```
<<  
'X→Y' STO OVER COLΣ  
LR  
SWAP ROT  
IF 4 ==  
THEN EXP  
END SWAP CORR SQ  
3 DUPN 'RSQ' STO 'B'  
STO 'A' STO  
>>
```

**Comments:**

```
b, A  
m  
  
  
a  
r2  
Store parameters.
```

```
ENTER  
'CFU1' STO
```

Store the program.

## CFU2 (Curve Fit 2) Utility

### Program:

```
<<
DUP EVAL DUP 6 PICK
IF >
THEN 4 →LIST 5 ROLL
4 DROPN LIST→ DROP
ELSE 4 DROPN
END
>>
```

### Comments:

Compare output from two  
curve fits and discards  
the one with the smaller  $r^2$ .

ENTER

'CFU2' STO

Store the program.

## LINC (Linear Curve Fit) Subroutine

Arguments	Results
3:	3: $a$
2:	2: $b$
1:	1: $r^2$

### Required Program:

- CFU1

### Program:

```
<< 2 1 << PREDV >> CFU1
>>
```

### Comments:

ENTER

'LINC' STO

Store the program.

## EXPC (Exponential Curve Fit) Subroutine

Arguments	Results
3 :	3 : $a$
2 :	2 : $b$
1 :	1 : $r^2$

### Required Program:

- CFU1

### Program:

```
<< 4 1 << B * EXP A *  
>> CFU1 >>
```

'EXPC'

### Comments:

Store the program.

## PWRC (Power Curve Fit) Subroutine

Arguments	Results
3 :	3 : $a$
2 :	2 : $b$
1 :	1 : $r^2$

### Required Program:

- CFU1

### Program:

```
<<  
4 3 << B ^ A * >> CFU1  
>>
```

'PWRC'

### Comments:

Store the program.



## LOGC (Logarithmic Curve Fit) Subroutine

Arguments	Results
3:	3: $a$
2:	2: $b$
1:	1: $r^2$

### Required Program:

- CFU1

### Program:

```
«
2 3 « LN PREDV »
CFU1
»
```

### Comments:

ENTER

'LOGC' STO

Store the program.

## PLOT (Curve Plot) Subroutine

### Program:

```
«
'ΣPAR' RCL
1 2 COLΣ SCLΣ
CLLCD DRWΣ
'ΣPAR' STO
« X X→Y » STEQ
DRAW
'EQ' PURGE
»
```

### Comments:

Save current ΣPAR.

Set up plot parameters.

Plot the data.

Restore the old ΣPAR.

Make EQ compute Y.

Plot the curve.

Discard EQ.

ENTER

'PLOT' STO

Store the program.

**POINT (Accumulate Point) Subroutine**

Arguments	Results
2: <i>x</i> 1: <i>y</i>	2: 1:

Program:	Comments:
<< DUP TYPE IF DUP 3 == THEN DROP ARRAY→ DROP ELSE IF 1 == THEN C→R END END DUP2 LN SWAP LN SWAP { 4 } →ARRAY Σ+ >>	Determine entry type. Unpack a vector. Unpack a complex number.
ENTER 'POINT' STO	Store the program.

**DEL (Delete Point) Subroutine**

Argument	Result
1:	1: [ <i>x y</i> ]

Program:	Comments:
<< Σ- { 2 } RDM >>	Return <i>x</i> and <i>y</i> .
ENTER 'DEL' STO	Store the program.

# BEST (Best Fit) Subroutine

Arguments	Results
3 :	3 : <i>a</i>
2 :	2 : <i>b</i>
1 :	1 : $r^2$

## Required Programs:

- LOGC
- PWRC
- LINC
- EXPC
- CFU2

## Program:

```
<<
'LOGC' DUP EVAL
'PWRC' CFU2
'LINC' CFU2
'EXPC' CFU2
3 DROPN DUP EVAL
#25AFCh SYSEVAL
4 ROLL 4 DISP
>>
```

```
[ENTER]
'BEST' [STO]
```

## Comments:

Logarithmic fit.  
Power fit.  
Linear fit.  
Exponential fit.  
Repeat the best fit.  
Display the stack.\*  
Display the best curve name.

Store the program.

\* This program shows the use of the SYSEVAL function to programmatically display an updated stack.

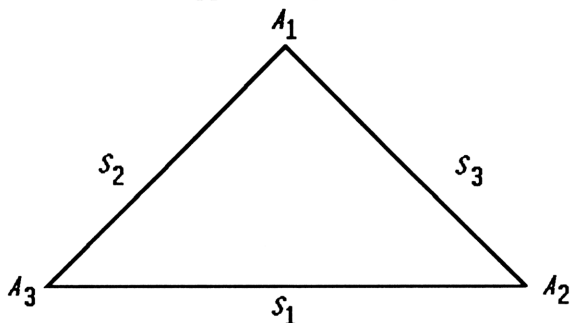
## Triangle Solutions

---

These programs can be used to find the area, the lengths of all of the sides, and all of the angles of a triangle. The triangle is specified in terms of one of the following:

- Three sides (SSS).
- Two sides and the included angle (SAS).
- Two sides and an adjacent angle (SSA).
- Two angles and the included side (ASA).

In the descriptions that follow, we will use the symbols  $S_n$  and  $A_n$  to represent the  $n$ th side and opposite angle, respectively:



The algorithms and programs work equally well when the sides and angles are numbered clockwise or counterclockwise around the triangle.

The triangle programs will work with numeric or symbolic values for the sides and angles.

---

## Algorithms

### SSS (3 Sides)

$S_1$ ,  $S_2$ , and  $S_3$  are known:

$$\begin{aligned}P &= (S_1 + S_2 + S_3)/2 \\A_1 &= 2\cos^{-1} \left[ \frac{P(P - S_1)}{S_2 S_3} \right]^{1/2} \\A_2 &= 2\cos^{-1} \left[ \frac{P(P - S_2)}{S_1 S_3} \right]^{1/2} \\A_3 &= \cos^{-1}(-\cos(A_2 + A_1))\end{aligned}$$

### ASA (2 Angles and Included Side)

$A_2$ ,  $S_1$ , and  $A_3$  are known:

$$\begin{aligned}A_1 &= \cos^{-1}(-\cos(A_2 + A_3)) \\S_2 &= S_1 \frac{\sin A_2}{\sin A_1} \\S_3 &= S_1 \cos A_2 + S_2 \cos A_1\end{aligned}$$

### SAA (Side, Adjacent Angle, Opposite Angle)

$S_1$ ,  $A_3$ , and  $A_1$  are known:

$$A_2 = \cos^{-1}(-\cos(A_3 + A_1))$$

The problem is now reduced to the ASA configuration.

## SAS (2 Sides and Included Angle)

$S_1, A_3$ , and  $S_2$  are known:

$$S_3 = \sqrt{S_1^2 + S_2^2 - 2S_1S_2 \cos A_3}$$

The problem is now reduced to the SSS configuration.

## SSA (2 Sides and Adjacent Angle)

$S_1, S_2$ , and  $A_1$  are known:

$$A_2 = \sin^{-1}\left[\frac{S_2}{S_1} \sin A_1\right]$$

$$A_3 = \cos^{-1}[-\cos(A_1 + A_2)]$$

The problem is now reduced to the ASA configuration. But note that there are two possible solutions if  $S_2 > S_1$  and  $A_3 \neq 90^\circ$ , corresponding to  $A_3$  determined by the above equation and  $A_3' = 180^\circ - A_3$ . In the program listing, the first solution corresponds to the program SSA1 and the second to SSA2.

## Area

$$Area = \frac{S_1 S_2 \sin A_3}{2}$$

---

## Instructions

1. Select radians mode or degrees mode as appropriate. In degrees mode, angles are entered and returned in degrees; in radians mode, angles are entered and returned in radians.

Press **MODE**, then either **DEG** or **RAD**.

2. Choose the program corresponding to the known elements of the program:

Press **USER**, then one of the following: **SSS**, **ASA**, **SAA**, **SAS**, **SSA1**, or **SSA2**.

3. The HP-28S will halt and display an input menu showing menu keys for each of the three known sides or angles or both. For each variable, enter the desired value and press the corresponding menu key. You may enter the values in any order.

- If you leave this menu, you can return at any time by pressing **CUSTOM**.
- You do not have to enter a value for any of the prompted quantities if you want to preserve the current value.
- To recall the current value of any of the variables, press **'**, followed by the appropriate menu key, then **EVAL**.

One or more of the values that you enter may be symbolic, for which the programs will return symbolic results. However, if you do enter symbolic values, don't use the names S1, S2, S3, A1, A2, or A3, or any of the triangle program names for any of the sides or angles. This will cause circular references, which can result in endless program execution (from which you can recover by pressing **ON** to produce a system halt).

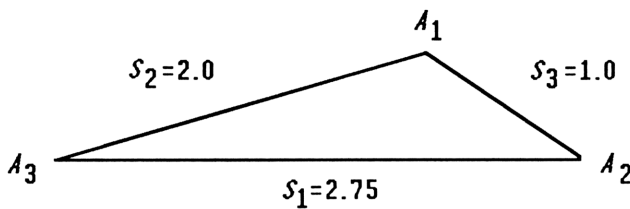
4. Press **CONT** to continue the program. When the triangle computation is complete, you will see a **CUSTOM** menu appear with a menu key for each of the six sides and angles of the triangle. Press any of the six keys to return the value of the corresponding variable to the stack. The triangle variables may also be used from the **USER** menu.

5. Press **NEXT** (repeatedly, if necessary), to find the menu key **AREA**. Press **AREA** to return the area of the triangle.

6. To perform another triangle computation, return to step 1. After the first time you use one of the triangle programs, you may wish to reorder the **USER** menu so that the program keys are moved to the front of the menu, ahead of the newly created side and angle variables.

# Examples

**Example 1.** Find the angles (in degrees) and the area for the following triangle:



Call the SSS program.

MODE DEG 4 FIX  
USER SSS

0:					
2:					
1:					
	S1	S2	S3		

Enter the three sides and solve for angle  $A_1$ .

2.75 S1  
2 S2  
1 S3  
CONT  
A1

0:					
2:					
1:					129.8384
	S1	A1	S2	A2	S3

Solve for angle  $A_2$ .

A2

0:					
2:					129.8384
1:					33.9479
	S3	A3	A2	A1	S2

Solve for angle  $A_3$ .

A3

0:					129.8384
2:					33.9479
1:					16.2136
	S3	A3	A2	A1	S2

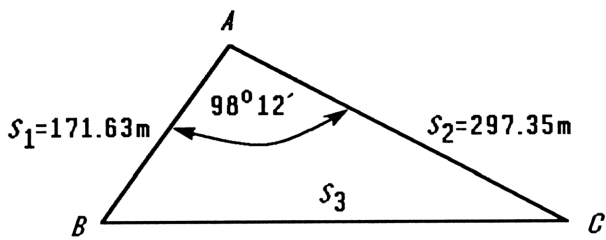
Find the area of the triangle.

AREA

0:					33.9479
2:					16.2136
1:					0.7679
	AREA	SSA2	SSA1	SAS	SAA



**Example 2.** A surveyor is to find the area and dimensions of a triangular land parcel. From point  $A$ , the distances to  $B$  and  $C$  are measured with an electronic distance meter. The angle between  $AB$  and  $AC$  is also measured. Find the area and the other dimensions of the triangle.



This is an SAS problem:

Call the SAS program.

CLEAR

USER

SAS

3:

2:

1:

S1

A1

S2

Enter the angle and two sides, then solve for side  $S_3$  (in meters).

171.63

S1

297.35

S2

98.12

TRIG

HMS→

CUSTOM

A3

CONT

S3

3:

2:

1:

363.9118

S1

A1

S2

A2

S3

A3

Solve for angle  $A_1$ .

A1

3:

2:

1:

363.9118

27.8270

A1

A2

S3

A3

S2

S1

Solve for angle  $A_2$ .

≡ A2 ≡

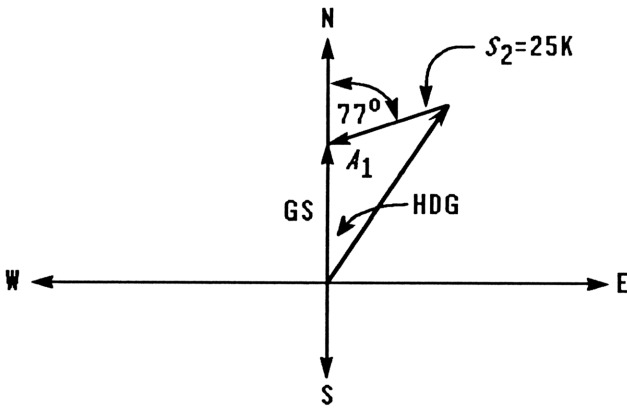
3:	363.9118				
2:	27.8270				
1:	53.9730				
A1	A2	S3	A3	S2	S1

Find the area of the triangle.

≡ AREA ≡

3:	27.8270				
2:	53.9730				
1:	25256.2094				
AREA	SSA2	SSA1	SAS	SAA	ASA

**Example 3.** A pilot wishes to fly due north. The wind is reported as 25 knots at  $77^\circ$ . Because winds are reported opposite to the direction of air motion, this is interpreted as  $77 + 180^\circ$  or  $257^\circ$ . The true airspeed (TAS) of the aircraft is 140 knots. What heading (HDG) should be flown? What is the ground speed (GS)?



By subtracting the wind direction from  $180^\circ$  (yielding an angle of  $103^\circ$ ), the problem reduces to an SSA triangle solution:

Call the SSA1 program.

USER ≡ SSA1 ≡

3:					
2:					
1:					
	S1	S2	A1		

Enter two sides and the adjacent angle and solve for HDG ( $A_2$ ).

140  $\overline{\text{S1}}$   
 25  $\overline{\text{S2}}$   
 103  $\overline{\text{A1}}$   
 CONT  
 $\overline{\text{A2}}$

3:					
2:					
1:					10.0202
	M2	M1	S2	S1	MREN S2M2

Solve for GS ( $S_3$ ).

$\overline{\text{S3}}$

3:					
2:					10.0202
1:					132.2407
	S3	M3	M2	M1	S2 S1

Purge the menu entries created by this example.

{ 'A1' 'A2' 'A3' 'S1' 'S2' 'S3' PURGE }

## Programs

### TU1 (Triangle 1) Utility

Arguments	Results
2: x 1: y	2: 1: $\cos^{-1}[-\cos(x+y)]$

#### Program:

« + COS NEG ACOS »

ENTER

'TU1' STO

#### Comments:

$\cos^{-1}(-\cos(z_1 + z_2))$

Store the program.

## TU2 (Triangle 2) Utility

Arguments	Results
2 : { list } 1 : 'name'	

### Program:

```
«  
{ S1 A1 S2 A2  
S3 A3 AREA } MENU  
»
```

'TU2'

### Comments:

Store the program.

## TU3 (Triangle 3) Utility

Argument	Result
1 : x	1 :

### Required Programs:

- TU1
- ASA

### Program:

```
«  
DUP 'A2' STO A1 TU1  
'A3' STO  
S2 A1  
→ASA 'A1' STO 'S2'  
STO  
»
```

'TU3'

### Comments:

Program common to →SSA1 and  
→SSA2.

Store the program.

## Computation Subroutines

The following subroutines perform the actual triangle computations and can be called by other routines. The name of each routine is "→" followed by three letters indicating which case (for example, SSS, SAS, and so on) the routine evaluates.

### →SSS (Compute SSS) Subroutine

#### Required Program:

- TU1

#### Program:

```
« S1 S2 S3
+ + 2 /
DUP DUP2 S2 - * S1 /
S3
/ √ ACOS 2 * 'A2'
STO
S1 - * S2 / S3 / √
ACOS 2 * DUP 'A1'
STO
A2 TU1 'A3' STO
»
```

#### Comments:

P  
Store  $A_2$ .  
Store  $A_1$ .  
Store  $A_3$ .

ENTER

'→SSS' STO

Store the program.

## →ASA (Compute ASA) Subroutine

### Required Program:

- TU1

### Program:

```
« S1 A3 A2
TU1 DUP 'A1' STO
SIN / A2 SIN * 'S2'
STO
S1 A2 COS * A1 COS
S2 * +
'S3' STO
»
```

### Comments:

Store  $A_1$ .

Store  $S_2$ .

$S_3$

ENTER

'→ASA' STO

Store the program.

## →SAA (Compute SAA) Subroutine

### Required Programs:

- TU1
- →ASA

### Program:

```
« A1 A3 OVER
TU1 'A2' STO
→ASA
'A1' STO
»
```

### Comments:

Save  $A_2$ .

$S_2, S_3$

Restore original  $A_1$ .

ENTER

'→SAA' STO

Store the program.

## →SAS (Compute SAS) Subroutine

### Required Programs:

- TU1
- →SSS

### Program:

```
« S1 S2 A3 3 DUPN
COS * * -2 *
ROT SQ + ROT SQ +
√ 'S3' STO
→SSS
'A3' STO
»
```

ENTER

'→SAS' STO

### Comments:

Store  $S_3$ .  
 $A_1, A_2$   
Restore original  $A_3$ .

Store the program.

## →SSA1 (Compute SSA) Subroutine 1

### Required Program:

- TU3

### Program:

```
«
A1 SIN S2 * S1 /
ASIN
TU3
»
```

ENTER

'→SSA1' STO

### Comments:

$A_2$

Store the program.

## →SSA2 (Compute SSA) Subroutine 2

### Required Program:

- TU3

### Program:

```
«  
A1 SIN S2 * S1 /  
ASIN  
-1 ACOS SWAP -  
TU3 »
```

### Comments:

$A'_2$

$A'_2$

ENTER

'→SSA2' STO

Store the program.

## Main Programs

The following programs are intended to be used directly from the USER menu. The name of each program describes which of the various triangle solutions is performed by the program.

## SSS (Side-Side-Side) Main Program

### Required Programs:

- TU2
- →SSS

### Program:

```
« { STO S1 S2 S3 }  
MENU HALT →SSS TU2 »
```

### Comments:

ENTER

'SSS' STO

Store the program.



## ASA (Angle-Side-Angle) Main Program

### Required Programs:

- TU2
- →ASA

### Program:

```
« { STO S1 A3 A2 }  
MENU HALT →ASA TU2 »
```

### Comments:

'ASA'

Store the program.

## SAA (Side-Angle-Angle) Main Program

### Required Programs:

- TU2
- →SAA

### Program:

```
« { STO S1 A3 A1 }  
MENU HALT →SAA TU2 »
```

### Comments:

'SAA'

Store the program.

## SAS (Side-Angle-Side) Main Program

### Required Programs:

- TU2
- →SAS

### Program:

### Comments:

```
« { STO S1 A3 S2 }  
MENU HALT →SAS TU2 »
```

'SAS'

Store the program.

## SSA1 (Side-Side-Angle) Main Program 1

### Required Programs:

- TU2
- →SSA1

### Program:

### Comments:

```
« { STO S1 S2 A1 }  
MENU HALT →SSA1 TU2  
»
```

'SSA1'

Store the program.

## SSA2 (Side-Angle-Side) Main Program 2

### Required Programs:

- TU2
- →SSA2

### Program:

```
<< { STO S1 S2 A1 }  
MENU HALT →SSA2 TU2  
>>
```

### Comments:

ENTER

'SSA2' STO

Store the program.

## AREA (Triangle Area)

### Program:

```
<< S1 S2 * A3 SIN * 2  
/ >>
```

### Comments:

ENTER

'AREA' STO

Store the program.



---

## More Step-by-Step Solutions for Your HP-28C and HP-28S Calculators

These additional books offer a variety of examples and keystroke procedures to help you set up your calculations the way *you* need them.

Practical routines show you how to use the built-in menus to solve problems more effectively, while easy-to-follow instructions help you create personalized menus.

### **Algebra and College Math** (00028-90101)

- Solve algebraic problems: polynomial long division, function evaluation, simultaneous linear equations, quadratic equations, logarithms, polynomial equations, matrix determinants, and infinite series.
- Perform trigonometric calculations: graphs of functions, relations and identities, inverse functions, equations, and complex numbers.
- Solve analytic geometry problems: rectangular and polar coordinates, straight line, circle, parabola, ellipse, hyperbola, and parametric equations.

### **Calculus** (00028-90102)

- Perform function operations: definition, composition, analysis, angles between lines, and angles between a line and a function.
- Solve problems of differential calculus: function minimization, computing tangent lines and implicit differentiation.
- Obtain symbolic and numerical solutions for integral calculus problems: polynomial integration, area between curves, arc length of a function, surface area, and volume of a solid of revolution.

### **Probability and Statistics** (00028-90104)

- Set up a statistical matrix.
- Calculate basic statistics: mean, standard deviation, variance, covariance, correlation coefficient, sums of products, normalization, delta percent of paired data, moments, skewness, and kurtosis.
- Perform regression techniques: curve fitting, multiple linear, and polynomial regression.

- Compute several test statistics.

## **Vectors and Matrices** (00028-90105)

- Perform general matrix operations: summation, multiplication, determinant, inverse, transpose, conjugate, and minor rank.
- Solve a system of linear equations.
- Calculate several important vector operations.
- Learn methods for calculating eigenvalues and eigenvectors.
- Perform the method of least squares and Markov Chain calculations.

---

## **How To Order...**

To order a book your dealer does not carry, call toll-free 1-800-538-8787. MasterCard, VISA, and American Express cards are welcome. For countries outside the U.S., contact your local Hewlett-Packard sales office.



# Step-by-Step Solutions for Your HP-28S Calculator

---

*Mathematical Applications* contains a variety of programs and examples to help you solve your technical problems more easily.

## ■ **Polynomials**

Addition • Subtraction • Multiplication • Division • Conversion  
Between Algebraic and Power Series Expressions • Roots of First-,  
Second-, Third-, and Fourth-Order Polynomials

## ■ **Differential Equations**

Symbolic Solution to Second-Order Differential Equations  
• Numerical Solutions to First-and Second-Order Differential  
Equations Using the Runge-Kutta Method

## ■ **Coordinate Transformation**

Convert Between Two Systems of Coordinates

## ■ **Curve Fitting**

Accumulate Data Points • Delete Data Points • Linear Curve Fit  
• Power Curve Fit • Exponential Curve Fit • Logarithmic Curve Fit  
• Determine Best Fit • Plot Data Points

## ■ **Triangle Solutions**

Find All Sides and Angles of a Plane Triangle Given Any  
Combination of Three Sides or Angles • Area



**HEWLETT  
PACKARD**

**Reorder Number**

**00028-90111**

00028-90162 English

Printed in Canada 7/90



0 88698 00026 7