# An Equation Library Using the Formula Evaluation ROM


The Formula Evaluation ROM from Greg McClure and Ángel Martin gives the HP-41CX/CL owner the ability to evaluate algebraic expressions stored as text strings in the Alpha register or in extended memory text files. The ROM functionality is quite extensive, as shown by the examples in the manual. The examples included the use of a solver program to find the root(s) of a given expression.

The first application of the ROM that came to my mind was the Equation Library from the HP48 series calculators. Wouldn't it be nice to have a set of equations to choose from, equations that you could then establish values and solve for unknowns. Even better would be the ability to add your own equations to the library. The EQNLIB program and its associated Extended Memory[1] text file EQNS provides that capability.

[1] Although written for the HP-41CX/CL extended memory, EQNLIB can be adapted for use with HEPAX memory by changing the file operation commands.

## Limitations

Any formula you use in the Formula Evaluation ROM must fit within the 24 character size limit of the Alpha register. Additionally, the Equation Library uses the five formula variables ‘a’ through ‘e’ for the top row of user keys. This limits any equation you use to only five independent variables. As we will see in a later section, more variables and longer equations can be accommodated through equation chaining.

New in revision B, the Equation Library program allows you to use two formulas for more complicated expressions. The result of a first formula can be used in a second formula to complete the calculation. This does require a larger expression to be broken into two partial expressions, the details for which are explained in a later section.

Throughout this manual I will use *equation* to mean an entry in the Equation Library, *formula* to mean the alpha string accepted and processed by the Formula Evaluation ROM, and *expression* to mean the common or formal representation of an equation, often with implied multiplication, as in E = IR.

## *Program Setup*

The ROM & MOD files contain a 4K page of mostly FOCAL programs. The Equation Library is dependent on the Formula Evaluation ROM version 1F, which itself is dependent on Library#4 and the HP-41CX ROM. Two emulators have been tested as well as a calculator with the 41CL CPU board.

On the V41 emulator add the Equation Library MOD file to your 41cx setup file. For go41cx or go41cxt, copy the MOD file to the files/MOD directory, perform an import, and then plug the module into any empty port. If you have a 41CL calculator, then transfer the ROM file image to a suitable RAM page (830 for instance) and plug the RAM page into an empty port page.

Once configured, the program INIEQN, will create an initial Equation Library text file in extended memory called EQNS. Other library files can be created containing your own equations. The format of a library file will be described in detail in a later section. For now, run INIEQN to get started with Equation Library.

## *Program Usage*


Program operation is straightforward. Execute EQNLIB and it will display the name of the first equation in the default Equation Library (EQNS). If you create your own library file enter the name of the file in Alpha and then execute EQN$.

Use the F key (NEXT) to scroll forward through the list of equations and the G key (PREV) to scroll backwards. The H key (VIEW) lets you switch between the name of an equation and its formal algebraic expression as shown below.



You can pick an equation and start solving by pressing the J key (MENU) to display the solver menu. To set a value for a variable, key in the value then press the user key beneath the variable. The menu is redisplayed each time after you set the value of a variable and you can press the J key to redisplay the menu after solving for a variable. For the Quadratic equation, we would first need to establish the values of A, B and C.


Using the example from the Formula Evaluation ROM manual, let's use the values A=1, B=4 and C=1. If we set X to 5 then press Y, we get the value 46. Specify 41 as the value of Y and press X to compute a value of 4.6332 for X. Naturally, if we'd like to compute the root of this quadratic, we would set Y to 0. We end up with a result of -0.2679, but this is only one of two real roots. How do we find the other?

## Establishing Initial Guesses

By default the initial guesses for the solver are 0 and 1, which will work fine for many equations. For equations with multiple roots though we need a way to supply our own initial guesses. EQNLIB provides this ability. Just enter your lower and upper guess and press the shifted user key beneath the variable to solve for. Since the initial guesses 0 and 1 returned a negative root for out example quadratic equation, the second root must be less than -0.27, so type

```
-1 ENTER -5 SHIFT A
```

The result of the calculation is -3.7321 for the second root.

## Equation Chaining

The Equation Library uses the Formula Evaluation ROM variables ‘a’ through ‘e’, and these variables retain their values between expression evaluations. This allows one equation to establish values and another equation to use them, much the same way as using the same variable name in multiple equations with the HP Solver. Let’s continue with the quadratic equation to illustrate the idea of equation chaining.

Consider the expression for calculating the two real roots of a quadratic equation:

$$x_1, x_2 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Let us now have two equations adjacent to one another in the Equation Library and call them POSROOT and NEGROOT. Each would have a menu with the constants A, B and C as the first three user keys and with X1 or X2 following. Each equation would show a different formula and use a different formula string. A hint of the structure of an equation in the Equation Library can be gleaned from the table below.

| **Equation Name** | POSROOT | NEGROOT |
|---|---|---|
| **Equation Expression** | X1=-B+SQRT(B^2-4AC)/2A | X2=-B-SQRT(B^2-4AC)/2A |
| **Equation Formula** | (#b+Q(b^2-4*a*c))/2/a-d | (#b-Q(b^2-4*a*c))/2/a-d |
| **Equation Menu** | A B C X1 | A B C X2 |

Select the POSROOT equation and hit the MENU key (“J”) to display its user key menu. Enter the values for A, B and C and then press the X1 user menu key (“D”) to solve for the first root. Press the NEXT button (“F”) then press the “D” key again to solve for the second root. No need to press the MENU key before solving for the second root!

Obviously this is a simple example of what can be a powerful technique to extend the capabilities of the Equation Library and is described in the Formula Evaluation ROM manual itself.

## *Summing It Up*

A summary of the keys and flags used by the EQNLIB program is shown below.

| **Key** | **Name** | **Function** | **Description** |
|---|---|---|---|
| F | NEXT | Next equation | |
| G | PREV | Previous equation | |
| H | VIEW | Toggle display | Switch between name & expression |
| J | MENU | Initiate Solver | |
| A-E | Variable | Set or Solve Variable | Enter data to set variable or none to solve |
| SHIFT A-E | Initial Guess | Solve Variable | Y=lower guess, X=higher guess |
| **Flag** | **Clear** | **Set** | **Usage** |
| Flag 10 | No trace | Trace | Show intermediate solver results |

Execute EQNLIB and it will display the name of the first equation in the library. From there you can scroll through the set of equations using the NEXT and PREV key. If you use VIEW to switch to the equation expression then scrolling will show all of the available expressions in the library.

The internal solver uses flag 10 to indicate whether or not to display intermediate results. If the solver doesn't seem to be converging on an answer, try pressing R/S and then set flag 10 with SF 10. When you resume the program with R/S you should see the intermediate values flash by.

Finally, many equations need a good set of guesses to even return a valid result. The equation for the equivalent resistance of two resistors in parallel is a good example. Unless your guesses bracket the correct answer, the result displayed by the solver will be incorrect or will not converge. Equations with multiple closely-spaced roots can also be a challenge. Try the cubic equation with constants A=6, B=11 and C=6. The roots are -1, -2 and -3. The default initial guesses will find the first, but the others require a bit of hunting around with your guesses.

Now that you know how to use the Equation Library, let's see how to add your own equations.

## *Roll Your Own Equation*

Since we used a general form of the quadratic equation, let's try to create our own root finder for a fourth-order equation. We can write out a normalized fourth order power equation as

$$y = x^4 + ax^3 + bx^2 + cx + d$$

This equation has six unknowns which exceed our limit of five. Applying Horner's Method, with y set to zero, we get

$$d + x(c + x\big(b + x(a + x)\big))$$

Now we can construct an entry in the EQNS file to find roots of this expression. First, we need a name for the Equation Library entry. Names should be twelve characters or less to avoid display scroll. Let's choose "4TH ORDER" for a name. Next, we need an expression in the HP-41 character set for the above. The only change needed to use the above expression would be to change all characters to uppercase.

The next step is to design the equation menu and write the formula. If we place the power equation constants 'a' to 'd' first in the menu, then we can use the expression exactly as it is but with the 'e' variable substituting for x in the expression above. But rather than take the easy approach (and to illustrate the more general way to map an expression to a formula) I'll place the x variable in the menu first. Let's create a menu like X? A B C D where the question mark following the variable name X reminds us that this is what we want to solve for[2]. Here's an illustration that will help with the mapping task.

| **Menu** | X? | A | B | C | D |
|---|---|---|---|---|---|
| **Key/Variable** | A/**a** | B/**b** | C/**c** | D/**d** | E/**e** |
| **Expression** | d+x(c+x(b+x(a+c))) | | | | |
| **Formula** | e+a*(d+a*(c+a*(b+d))) | | | | |

The formula is slightly confusing, but only because we're using an expression with our 'a' through 'e' variables names in it. Try doing the above mapping process with the expression for Ohm's Law. Easier to visualize now?

Adding a new equation means adding four more records to the end of the EQNS XM text file. A program called APPEQN (Append Equation) is listed in the Appendix that will do exactly that for you. Just modify the four text lines and run the program. Another approach is to use ED to enter the four lines. ED cannot produce the special characters that OS/X or CCD provides, so you can use temporary substitutes and then run the SAR (Search and Replace) program also listed in the appendix. I would suggest using '<', and '>' for parentheses and '#' (SHIFT-H) for

[2] Mind you, we can always set the X value and three of the constants and then solve for the fourth, can't we? Why not!

not-equal when entering the equation, then run the search and replace program where you can enter the proper character at the replace prompt.

So here then are the four records we need to add to the Equation Library file:

```
4TH ORDER
D+X(C+X(B+X(A+X)))
e+a*(d+a*(c+a*(b+a)))
X? A  B C  D
```

Once you enter these lines into EQNS, run EQNLIB and give the equation a try. Taking the example from the Formula Evaluation ROM manual, try using 2 for A, 3 for B and 4 for C, 5 for D and then solve for X. With the default initial guesses you get -0.8569 for the answer. Can you find the other three roots?

To support more complex expressions, you can break the expression into two parts, with the result of the first expression serving as input to the second. To do this you create one formula that computes the partial expression. The result is placed in stack register T by the EVALT function. The second formula uses register T in the formula to incorporate the partial result from the first expression. Let's use the following formula for converting between Nominal and Effective interest rate as illustration.

$$Effective\ Rate\ = \left[\left(1 + \frac{Nominal\ Rate}{100\ \times P}\right)^{P} - 1\right] \times 100$$

Rates are expressed as percentages such as 4.5% and P is the number of payments per year. Let the formulas be based on the following menu

```
EFF NOM P
```

We can use the power expression in parentheses as the first partial expression. The formula would then be

```
(1+b/100/c)^c
```

The formula that incorporates this result would then be

```
(T-1)*100-a
```

The result through substitution would be the complete formula

```
((1+b/100/c)^c-1)*100-a
```

## *Equation Library Format*

An Equation Library is an extended memory text file. Each equation in the library occupies four records, none of which should exceed the 24 character length of the Alpha register. For each equation, the first record is the name of the equation, the second record is the expression or hint text displayed to the programmer when switching between views, the third record is the formula itself and the fourth record is the menu displayed when solving the equation.

| | | | |
|---|---|---|---|
| **00** | **LINEAR** | **32** | **RLC FREQ.** |
| **01** | Y=AX+B | **33** | F0=1/SQRT(LC) |
| **02** | c*a+d-b | **34** | 1/Q(b*c)-a |
| **03** | X Y A B | **35** | F0 L C |
| **04** | **QUADRATIC** | **36** | **GAS EQUATION** |
| **05** | Y=AX^2+BX+C | **37** | PV=NRT |
| **06** | c*a^2+d*a+e-b | **38** | c*(16629/2000)*d-a*b |
| **07** | X Y A B C | **39** | P V N T |
| **08** | **CUBIC** | **40** | **LIN. MOTION** |
| **09** | Y=X^3+AX^2+BX+C | **41** | X=VT+1/2*AT^2 |
| **10** | a^3+c*a^2+d*a+e-b | **42** | c*b+1/2*d*b^2-a |
| **11** | X Y A B C | **43** | X T V A |
| **12** | **4TH ORDER** | **44** | **NEWTONS LAW[3]** |
| **13** | D+X(C+X(B+X(A+X))) | **45** | F=G*M1*M2/R^2 |
| **14** | e+a*(d+a*(c+a*(b+a))) | **46** | e*b*c/d^2-a |
| **15** | X? A B C D | **47** | F M1 M2 R G |
| **16** | **POSROOT** | **48** | **INTEREST** |
| **17** | X1=(-B+SQRT(B^2-4AC))/2A | **49** | P=PERIODS |
| **18** | (#b+Q(b^2-4*a*c))/2/a-d | **50** | ((1+b/100/c)^c-1)*100-a |
| **19** | A B C X1 | **51** | EFF NOM P |
| **20** | **NEGROOT** | **52** | **+INTEREST** |
| **21** | X2=(-B-SQRT(B^2-4AC))/2A | **53** | (T-1)*100-a |
| **22** | (#b-Q(b^2-4*a*c))/2/a-d | **54** | (1+b/100/c)^c |
| **23** | A B C X2 | **55** | EFF NOM P |
| **24** | **OHMS LAW** | **56** | **+TVM END MODE** |
| **25** | E=IR | **57** | a+T+e*(1+b)^#d |
| **26** | b*c-a | **58** | (1+b)*c*((1-(1+b)^#d)/b) |
| **27** | E I R | **59** | PV I PM N FV |
| **28** | **PARALLEL R** | **60** | **+TVM BEG MODE** |
| **29** | 1/R1=1/R2+1/R3 | **61** | a+T+e*(1+b)^#d |
| **30** | 1/b+1/c-1/a | **62** | c*((1-(1+b)^#d)/b) |
| **31** | R1 R2 R3 | **63** | PV I PM N FV |

[3] G is $6.67408\times10^{-11}$ in SI units.

The table above shows the set of equations for the EQNS extended memory text file. When a complex expression needs a formula that exceeds the 24 character limit for formula size, the formula can be broken into two partial formulas as outlined in the preceding section. Place the first partial formula in the third record and the second formula in the second record. To flag the equation as a two formula problem put a '+' sign at the beginning of the equation name.

Notice the two versions of the interest rate calculation example. The formula was short enough to fit the 24 character limit as shown in the **INTEREST** equation entry. The **+INTEREST** equation entry is our example of how to handle more complex formulas and the way they are represented in the library.

The last pair of TVM equations is a fair illustration of what can be accomplished with the Formula Evaluation ROM. The interest rate should be expressed as a percentage divided by the number of periods per year, e.g. 6% would be 0.06/12 for something like a car loan. P is the number of payments, as in 360 for a 30 year mortgage. If Present Value (PV) and Future Value (FV) are positive then Payment (PM) will be a negative value.

## *Creating Your Own Library*

Now that you know the format of an equation entry, how would you add new equations to the default EQNS library or create your own custom library? A new custom library is just a text file created by putting the filename in Alpha and a starting file size of say 4 registers in X and then executing the CRFLAS command.

Use the COPY command to copy the APPEQN program to user memory. The program has two entry points; the default APPEQN label that will append a new equation entry to the default EQNS library and the APP$ label that will append a new equation entry to the text file named in the Alpha register.

Edit the four text line records for name, expression, formula and menu. Don't forget for a long text record you may need to add a second line starting with the append symbol (SHIFT-K) to complete the full text of an expression or formula record. The program, through either entry point, will go to the last record in the file, extend the file length by ten registers, and then append the text records you've entered. Repeat until you've added all of your equations. You can delete APPEQN when you are finished.

# *Appendix*

Listings for the EQNLIB program and various support programs are included.

## SAR: Search and Replace program

```
  1  LBL "SAR"
  2  "FNAME?"
  3  AON
  4  PMTK
  5  LBL "SAR$"
  6  ASTO 02
  7  0
  8  SEEKPTA
  9  "SERCH?"
 10  AON
 11  PMTK
 12  ASTO 00
 13  "RPLCE?"
 14  AON
 15  PMTK
 16  AOFF
 17  ASTO 01
 18  LBL 00
 19  CLA
 20  ARCL 00
 21  POSFL
 22  X<0?
 23  GTO 01
 24  ALENG
 25  DELCHR
 26  CLA
 27  ARCL 01
 28  INSCHR
 29  RDN
 30  GTO 00
 31  LBL 01
 32  CLA
 33  ARCL 02
 34  RDN
 35  END
```

This program will search for and replace strings in an extended memory text file. The search and replace strings are limited to a maximum of six characters. The program will save the name of the XM text file and restore it to the Alpha register so that you can invoke the ED editor to examine results. The last three text pointers are also left on the stack so that you can roll the stack to one of these pointers and do a SEEKPT before invoking the editor.

The internal entry SAR$ can be used to bypass the prompt for the XM text file name if it is already in the Alpha register. Note that because the filename is saved to a single register, the name is also limited to six characters.

If you need to insert the special CCD/OSX characters like '(' into a text file with ED then just use the unshifted (USER mode on) character like '<' instead. You can use the search and replace to correct the characters in your file.

## APPEQN: Append Equation program

```
 1 LBL “APPEQN”
 2 “EQNS”
 3 LBL “APP$”
 4 0
 5 SEEKPTA
 6 LBL 00
 7 1
 8 +
 9 SF 25
10 SEEKPT
11 FS?C 25
12 GTO 00
13 FLSIZE
14 10
15 +
16 RESZFL
17 “EQN NAME”
18 APPREC
19 “EQN EXPRESSION”
20 APPREC
21 “EQN FORMULA”
22 APPREC
23 “EQN MENU”
24 APPREC
25 END
```

The APPEQN program will seek to the end of the EQNS Equation Library file and append a new equation of your own design. Simply edit the four text lines (in bold) for your equation then run the program. If you want to append to an equation file other than the default EQNS, then put the name of the file in the Alpha register and execute APP$.

**Title**

| **Description** | **Note** |
|---|---|
| Equation Library. Equations stored in XM text file EQNS. Use F & G keys to move forward and back through equation list, H key to switch between Formula name and equation, and J key to display menu and solve equation. | "808-RAM" means **ALPHA** 808-RAM **ALPHA**<br>YPOKE means **XEQ** A**LPHA** YPOKE **ALPHA**<br><br>Dependencies:<br>Formula Evaluation ROM 1F, Library4 |

| **Step** | **Instruction** | **Comment** |
|---|---|---|
| 1 | LBL "EQNLIB" | Equation Library |
| 2 | "EQNS" | Default Equation Library Data |
| 3 | LBL "EQN$" | User specified library |
| 4 | 0 | |
| 5 | STO 03 | Initialize record pointer |
| 6 | SEEKPTA | Go to beginning |
| 7 | GETREC | Get equation name |
| 8 | AVIEW | and display it |
| 9 | RTN | |
| | | |
| 10 | LBL "F" | Advance to next equation |
| 11 | RCL 03 | |
| 12 | 4 | Records per equation |
| 13 | + | |
| 14 | SF 25 | Set Error flag |
| 15 | SEEKPT | Go to next equation |
| 16 | FC?C 25 | Did the seek succeed? |
| 17 | GTO 00 | No, don't update record pointer |
| 18 | STO 03 | |
| 19 | GETREC | Get equation name/formula |
| 20 | LBL 00 | |
| 21 | AVIEW | and display it |
| 22 | RTN | |
| | | |
| 23 | LBL "G" | Go back to previous equation |
| 24 | RCL 03 | |
| 25 | 4 | |
| 26 | - | |
| 27 | X<0? | Still within file? |
| 28 | GTO 00 | No, leave as-is |
| 29 | STO 03 | Update pointer |
| 30 | SEEKPT | Go to equation |
| 31 | GETREC | Get formula name or equation |
| 32 | LBL 00 | |
| 33 | AVIEW | and display it |
| 34 | RTN | |
| | | |
| | | |

**Author** Mark A. Fleming
**Date** September 5, 2017

| Step | Instruction | Comment |
|---|---|---|
| 35 | LBL "H" | Switch between equation name & formula views |
| 36 | RCL 03 | Record pointer |
| 37 | 4 | Record size |
| 38 | MOD | |
| 39 | X=0? | Are we pointing to an equation name |
| 40 | GTO 00 | Yes, need to bump record pointer by one |
| 41 | 1 | No, decrement record pointer by one |
| 42 | ST- 03 | |
| 43 | GTO 01 | |
| 44 | LBL 00 | |
| 45 | 1 | |
| 46 | ST+ 03 | |
| 47 | LBL 01 | |
| 48 | RCL 03 | |
| 49 | SEEKPT | Go to equation name or formula record |
| 50 | GETREC | Retrieve record |
| 51 | AVIEW | and display it |
| 52 | RTN | |
| | | |
| 53 | LBL "I" | Refresh view |
| 54 | AVIEW | |
| 55 | RTN | |
| | | |
| 56 | LBL "J" | Select equation and solve |
| 57 | RCL 03 | |
| 58 | RCL 03 | |
| 59 | 4 | |
| 60 | MOD | |
| 61 | - | |
| 62 | STO 03 | Reset pointer to start of the formula block |
| 63 | SEEKPT | Get Equation Name |
| 64 | GETREC | |
| 65 | SF 00 | Extended two-line formula |
| 66 | 43 | "+" character |
| 67 | POSA | |
| 68 | X<0? | Not present in equation name? |
| 69 | CF 00 | Normal one-line formula |
| 70 | RDN | Bring back equation pointer |
| 71 | 3 | |
| 72 | + | |
| 73 | SEEKPT | Menu record is 4th entry |
| 74 | GETREC | |
| 75 | AVIEW | |
| 76 | RTN | |
| | | |
| | | |
| | | |

| Step | Instruction | Comment |
|---|---|---|
| 77 | LBL "A" | First variable |
| 78 | FC?C 22 | Was a number entered? |
| 79 | GTO 00 | No, solve for variable |
| 80 | LET= | Set variable |
| 81 | 1 | "a" |
| 82 | GTO "J" | Redisplay menu |
| 83 | LBL 00 | |
| 84 | 97 | "a" |
| 85 | XEQ 10 | Replace character and invoke solver |
| 86 | LET= | Save result back to variable |
| 87 | 1 | "a" |
| 88 | RTN | |
| | | |
| 89 | LBL "B" | Second variable |
| 90 | FC?C 22 | Was a number entered? |
| 91 | GTO 00 | No, solve for variable |
| 92 | LET= | Set variable |
| 93 | 2 | "b" |
| 94 | GTO "J" | Redisplay menu |
| 95 | LBL 00 | |
| 96 | 98 | "b" |
| 97 | XEQ 10 | Replace character and invoke solver |
| 98 | LET= | Save result back to variable |
| 99 | 2 | "b" |
| 100 | RTN | |
| | | |
| 101 | LBL "C" | Third variable |
| 102 | FC?C 22 | Was a number entered? |
| 103 | GTO 00 | No, solve for variable |
| 104 | LET= | Set variable |
| 105 | 3 | "c" |
| 106 | GTO "J" | Redisplay menu |
| 107 | LBL 00 | |
| 108 | 99 | "c" |
| 109 | XEQ 10 | Replace character and invoke solver |
| 110 | LET= | Save result back to variable |
| 111 | 3 | "c" |
| 112 | RTN | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

| Step | Instruction | Comment |
|---|---|---|
| 113 | LBL "D" | Forth variable |
| 114 | FC?C 22 | Was a number entered? |
| 115 | GTO 00 | No, solve for variable |
| 116 | LET= | Set variable |
| 117 | 4 | "d" |
| 118 | GTO "J" | Redisplay menu |
| 119 | LBL 00 | |
| 120 | 100 | "d" |
| 121 | XEQ 10 | Replace character and invoke solver |
| 122 | LET= | Save result back to variable |
| 123 | 4 | |
| 124 | RTN | |
| | | |
| 125 | LBL "E" | Fifth variable |
| 126 | FC?C 22 | Was a number entered? |
| 127 | GTO 00 | No, solve for variable |
| 128 | LET= | Set variable |
| 129 | 5 | "e" |
| 130 | GTO "J" | Redisplay menu |
| 131 | LBL 00 | |
| 132 | 101 | "e" |
| 133 | XEQ 10 | Replace character and invoke solver |
| 134 | LET= | Save result back to variable |
| 135 | 5 | |
| 136 | RTN | |

| Step | Instruction | Comment |
|---|---|---|
| 137 | LBL 10 | Solve for unknown variable, letter in X register |
| 138 | 1E-99 | Default solver guesses |
| 139 | STO 02 | |
| 140 | 1 | |
| 141 | STO 01 | |
| 142 | RDN | |
| 143 | RDN | |
| 144 | LBL 11 | Replace character in formula(s) |
| 145 | RCL 03 | |
| 146 | 2 | |
| 147 | + | |
| 148 | SEEKPT | |
| 149 | GETREC | First formula in third record |
| 150 | RDN | |
| 151 | XEQ "CH2X" | Replace selected character with "X" |
| 152 | FC? 00 | Regular one line formula? |
| 153 | GTO 00 | Yes |
| 154 | STO$ | Save first formula |
| 155 | 7 | |
| 156 | RCL 03 | |
| 157 | 1 | |
| 158 | + | |
| 159 | SEEKPT | |
| 160 | GETREC | Second formula in second record |
| 161 | RDN | |
| 162 | XEQ "CH2X" | Replace character for two formulas |
| 163 | STO$ | Save second formula |
| 164 | 11 | |
| 165 | LBL 00 | Invoke default or user supplied solver |
| 166 | RCL 02 | Recall default or user specified guesses |
| 167 | RCL 01 | |
| 168 | FC? 00 | Regular equation library? |
| 169 | GTO 08 | Yes |
| 170 | XEQ "ESLV+" | Internal solver, two line formula |
| 171 | RTN | |
| 172 | LBL 08 | |
| 173 | XEQ "ESLV" | Internal solver, regular equation library |
| 174 | RTN | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

| Step | Instruction | Comment |
|---|---|---|
| 175 | LBL "a" | |
| 176 | 97 | "a" |
| 177 | XEQ 12 | Save guesses and replace character |
| 178 | LET= | Save result back to variable |
| 179 | 1 | "a" |
| 180 | RTN | |
| | | |
| 181 | LBL "b" | |
| 182 | 98 | "b" |
| 183 | XEQ 12 | Save guesses and replace character |
| 184 | LET= | Save result back to variable |
| 185 | 2 | "b" |
| 186 | RTN | |
| | | |
| 187 | LBL "c" | |
| 188 | 99 | "c" |
| 189 | XEQ 12 | Save guesses and replace character |
| 190 | LET= | Save result back to variable |
| 191 | 3 | "c" |
| 192 | RTN | |
| | | |
| 193 | LBL "d" | |
| 194 | 100 | "d" |
| 195 | XEQ 12 | Save guesses and replace character |
| 196 | LET= | Save result back to variable |
| 197 | 4 | "d" |
| 198 | RTN | |
| | | |
| 199 | LBL "e" | |
| 200 | 101 | "e" |
| 201 | XEQ 12 | Save guesses and replace character |
| 202 | LET= | Save result back to variable |
| 203 | 5 | "e" |
| 204 | RTN | |
| | | |
| 205 | LBL 12 | Save initial guesses and invoke solver |
| 206 | RDN | |
| 207 | STO 01 | |
| 208 | RDN | |
| 209 | STO 02 | |
| 210 | RDN | |
| 211 | RDN | X has character to replace in formula |
| 212 | GTO 11 | Invoke Solver |
| 213 | END | |
| | | |
| | | |
| | | |

| Step | Instruction | Comment |
|---|---|---|
| 1 | LBL "CH2X" | Replace character in Alpha string with "X" |
| 2 | ENTER^ | Duplicate search character |
| 3 | POSA | Position of character in X register in Alpha string |
| 4 | X<0? | Not found? |
| 5 | GTO 00 | Yes, exit |
| 6 | AROT | Bring character to start of Alpha |
| 7 | ATOX | Remove character |
| 8 | CLX | No stack lift |
| 9 | 88 | Character "X" |
| 10 | XTOA | Append to Alpha string |
| 11 | RDN | Drop "X", get position |
| 12 | -1 | |
| 13 | X<>Y | |
| 14 | - | |
| 15 | AROT | Rotate back to original string with replacement |
| 16 | RDN | Drop number, bring Search character back to X |
| 17 | GTO "CH2X" | |
| 18 | LBL 00 | |
| 19 | RDN | Return with search character still in X |
| 20 | END | |
| | | |
| 1 | LBL "ESLV" | Internal solver, same as APP module SV$ function |
| 2 | STO$ | Save modified equation |
| 3 | 7 | |
| 4 | LBL 00 | |
| 5 | EVALZ | |
| 6 | X<>Y | |
| 7 | EVALT | |
| 8 | Z=T? | Has the estimate converged (T==Z?) |
| 9 | RTN | |
| 10 | "Y-Z*(Y-X)/(Z-T)" | Compute new guess |
| 11 | EVAL$ | |
| 12 | FS? 10 | Show intermediate result? |
| 13 | VIEW X | Yes |
| 14 | RCL$ | |
| 15 | 7 | |
| 16 | X#Y? | Converged on answer? |
| 17 | GTO 00 | No |
| 18 | END | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

| Step | Instruction | Comment |
|---|---|---|
| 1 | LBL "ESLV+" | Extended solver |
| 2 | RCL$ | First partial formula |
| 3 | 7 | |
| 4 | EVALT | Result in T |
| 5 | RCL$ | Second partial formula |
| 6 | 11 | |
| 7 | EVALT | Final result in T |
| 8 | R^ | |
| 9 | STO T | |
| 10 | RDN | Final result now in Z |
| 11 | X<>Y | |
| 12 | RCL$ | First partial formula |
| 13 | 7 | |
| 14 | EVALT | Result in T |
| 15 | RCL$ | Second partial formula |
| 16 | 11 | |
| 17 | EVALT | Final result in T |
| 18 | Z=T? | Has the estimate converged (T==Z?) |
| 19 | RTN | |
| 20 | "Y-Z*(Y-X)/(Z-T)" | |
| 21 | EVAL$ | |
| 22 | FS? 10 | |
| 23 | VIEW X | |
| 24 | X#Y? | |
| 25 | GTO "ESLV+" | |
| 26 | END | |