52-NOTES

Volume 3 Number 12 48/39/38 December 1978

Newsletter of the SR-52 Users Club
published at
9459 Taylorsville Road
Dayton, OH 45424

ROOTS OF POLYNOMIALS (V3N9p1)

Dix Fulton (83) and Frank Stallings (389) join Bill in addressing this topic. Some time ago Dix had submitted a Quartic, Cubic, Quadratic program to PPX-59 (#338005B), and has sent me an improved version, listed below. This is shorter than Bill's V3N9 program, primarily because the same processing is used for all 3 degrees: All 5 coefficients are entered, regardless of the degree. For a quadratic, D and E are zero, and for a cubic, E is zero, since the "quartic" $Ax^4+Bx^3+Cx^2+0x+0=0$ can be divided by x^2, making it the quadratic $Ax^2+Bx+C=0$ which it really is, and similarly the "quartic" $Ax^4+Bx^3+Cx^2+Dx+0=0$ reduces to a cubic. But quadratic and cubic processing are no faster than quartic, and the correct 2 or 3 roots must be identified among the always output 4.

Dix spotted a flaw in Bill's V3N9 program: In some cases precision is unnecessarily lost following a rounding procedure. Bill has corrected that, and reworked quadratic processing to improve accuracy when the 2 discriminant terms have a large difference in magnitude (V3N9p2). So for some cases, Bill's new program, listed below, is more accurate than Dix's.

Maurice Swinnen (779) has translated for English-speaking 59/PC users a Quartic, Cubic, Quadratic program written for German-speaking users, appearing in DISPLAY (V3N4/5S91). However, it requires 4 card sides of memory, and suffers the quadratic accuracy loss with critical discriminants.

Frank Stallings has been experimenting with iterative methods for finding the roots of higher order polynomials, and wrote the program listed below following the so-called Bairstow method. Frank's program appears to work well on a few sample problems, producing both real and complex roots for polynomials up to 61st degree, at a rate of about a minute per degree. However, there are no known iterative methods which work for all polynomials, and the Bairstow method (like the Newton method which it uses) is known to be bad for multiple roots solutions. So in cases where convergence is slow, Frank has provided a flag option allowing the monitoring of successive root approximations.

TI-59/PC Program: Polynomial Roots L Frank Stallings (389)

User Instructions: Press RST, R/S; see the prompter: DEGRE printed. Key degree (2-61), press R/S; key the coefficients, beginning with the highest, and follow each with R/S. Proc essing begins following input

---

of the constant. Monitor convergence (or lack thereof) by setting flag 0. With good convergence, roots are printed in pairs about every 2 minutes.

<u>Program Listing</u>:

000: CMs Adv Adv Op00 1617223517 Op2 CLR Op05 7 Op17 R/S Prt Adv GTO
027: 053 LE EE INV EE rtn LB 1) (R*69 ÷ R0 rtn Prt S*69 1 SUM69 ±
053: SUM68 R68 CP xGEt 045 LC Adv Adv 3532323736 Op2 Op5 GTO 084 Prt
083: Adv Dsz69 p51 R*69 x=t 082 2 ± ÷ R69 ÷ R0 X (CP X R1) S63 R2 ÷
112: (1 ± + R69 = S64 R64 X R68 + R63 X R67 + R*62) (Exc67 S68 R64
143: X R66 + R63 X R65 + (1 SUM62 INV SUM69 R69 CP x=t 121) INV xGEt
169: 181 R67) Exc65 S66 GTO 121 SUM62) 0 = X S69 R66 ± + R65 x^2 =
197: Exc66 X R67 - 0 Exc68 X R65 = ÷ 0 Exc66 INV x=t 219 B = SUM63
222: Prd69 Abs + ((0 Exc62 Exc69 + 0 Exc67) ÷ 0 Exc65 INV x=t 247 B)
248: INV SUM64 Abs = X 1 EE 8 = Op0 INV Ifflg0 266 Pause INV x=t 121
270: CLR 2 INV SUM69 1/x X (R63 x^2 + 4 X R64) INV xGEt 308 ($\sqrt{x}$ + R63)
297: - E Prt Adv R63 = E GTO 331 ± $\sqrt{x}$ = E xXt 24 Op4 R63 ÷ 2 = E Prt
324: xXt Op6 Adv xXt Prt xXt ± Op6 Adv R*62 X (CE X 1 SUM62 R62 xXt
347: R69 INV xGEt 370 R63) SUM*62 1 SUM62 R64 = SUM*62 Dsz62 335 CLR
371: S62 2 xXt R69 xGEt 121 xXt CLR x=t 395 R1 ÷ R0) ± E Prt Adv RST

Note: The p51 at step 086 is the pseudo BST which must be created synthetically (R51 BST BST Del)

- - - - - - - - - - -

TI-59/(PC) Program: Quadratic, Cubic, Quartic Roots Bill Skillman (710)

<u>User Instructions</u>: Same as V3N9p2)

<u>Program Listing</u>:

000: GTO 620 LE' SBR623 rtn LD' 1 Exc1 1/x Prd2 Prd3 Prd4 rtn LB' D'
025: 35 Op4 R2 ÷ 3 = S8 x^2 ± + R3 ÷ xXt 3 = S10 R4 + R8 X (x^2 X 2 -
058: 0 xXt = S11 x^2 + R10 X x^2 X 4 = INV xGEt 147 $\sqrt{x}$ + xXt R11 = ÷
083: 2 ± + SBR126 xXt = SBR126 + xXt - R8 + Ifflg2 142 E' R8 = ÷ 2
110: ± + xXt = X 3 $\sqrt{x}$ = xXt - R8 = GTO 609 (S13 Op10 Exc13 Abs INV y^x
136: 3 X R13) rtn 0 = S21 rtn R10 ± $\sqrt{x}$ X xXt 2 = S12 R11 ÷ 2 ÷ xXt y^x
164: 3 = ± Rad INV cos ÷ 3 = S9 SBR196 S21 SBR187 S22 2 X 2 X π ÷ 3 +
193: R9 = cos X R12 - R8 = S23 Ifflg2 719 GTO E' LA xXt 3 Op17 CMs
219: xXt S1 xXt 13 RST LB S2 xXt 14 RST LC S3 xXt 15 RST LD S4 xXt
247: 16 RST LE S5 xXt 17 RST LA' D' 35 Op4 R2 ÷ 2 = ± S6 S7 x^2 - R3
279: = CP xGEt 296 ± $\sqrt{x}$ xXt R6 Ifflg4 516 GTO 609 $\sqrt{x}$ X R6 Op10 =
303: SUM6 R3 ÷ R6 GTO 628 LC' R2 CP x=t 469D' Prd5 R2 X R4 - 4 X R5
334: = Exc3 S15 X ± Exc2 S14 4 - R14 x^2 = X R5 - R4 x^2 = Exc4 S16
363: Stflg2 20 S0 SBR025 INV Stflg2 CP CLR Op20 R*0 + R14 x^2 ÷ 4 -
388: R15 = SBR581 INV xGEt 376 $\sqrt{x}$ S19 ± + R14 ÷ 2 INV Prd*0 = S2 R14
415: X R*0 - R16 = Op10 X (R*0 x^2 - R5) SBR581 INV xGEt 376 $\sqrt{x}$ = S20
444: ± + R*0 = S3 SBR261 R19 SUM2 SUM2 R20 SUM3 SUM3 GTO 261 R4 INV
472: x=t 321 R3 x=t 540 R5 Exc3 S2 Stflg4 GTO 260 SBR496 R7 CP xGEt
498: 511 ± $\sqrt{x}$xXt SBR 616 ± xXt GTO 616$\sqrt{x}$ E' ± GTO E' xXt INV P/R ÷ 2
521: = xXt $\sqrt{x}$ xXt P/R xXt S6 SBR 603 xXt 1 ± Prd6 GTO 603 R5 ÷ R1 =
546: xGEt 569 ± $\sqrt{x}$ $\sqrt{x}$ xXt 35 SBR620 ± E' ± xXt SBR616 ± GTO E' Nop ÷
570: 4 = $\sqrt{x}$ $\sqrt{x}$ X xXt 1 = GTO 527 Nop S18 Fix4 EE INV EE INV Fix INV
592: x=t 597 CLR rtn R18 rtn Nop Nop Nop 35 Op4 R6 E' 47632024 Stflg3
620: Op4 xXt Op6 GTO 705 = S7 R6 Ifflg4 491 E' R7 GTO 623
705: Op8 Ifflg3 717 Op69 INV Stflg3 R/S CE rtn

- - - - - - - - - -

TI-59 Program: Quartic, Cubic, Quadratic Solutions Dix Fulton (83)

User Instructions: Input coefficients per V3N9p2, following a manual CMs. Press E' and see 4 roots printed; or without printer press E', see 0, press A', see root 1, press B', see root 2, press C', see root 3, press D', see root 4. For each root, the imaginary part is pause-displayed, followed by an xXt displaying the real part.

Program Listing:

000: LA' INV LB' Stflgl 6 GTO STO LC' INV LD' Stflgl 8 Lbl STO S9
022: R*9 ÷ 2 = x^2 - Op39 R*9 Op29 = CP xGEt Rcl Abs $\sqrt{x}$ xXt Lbl Rcl
044: $\sqrt{x}$ Ifflgl SUM xXt ± xXt ± Lbl SUM - R*9 ÷ 35171327 Op04 2 = Adv
071: Op6 xXt S09 24301322 Op4 R9 Op6 Pause xXt rtn LE' R4 1/x Prd0
100: Prd1 Prd2 Prd3 R1 S6 x^2 + R3 Prd6 x^2 X R0 + R2 ± S7 X (4 X R0)
133: INV SUM6 = ± S5 R7 ÷ 3 X S7 R6 - R5 = ÷ 2 - R7 X x^2 = S8 x^2 +
165: (R6 ÷ 3 - R7 x^2) X x^2 = CP xGEt Cos ± $\sqrt{x}$ xXt R8 xXt inv P/R
189: xXt INV y^x 3 = xXt ÷ 3 = P/R X 3 $\sqrt{x}$ ÷ 2 = S6 ± S5 xXt INV SUM5
214: INV SUM6 X 2 = xXt R5 INV xGEt Sin xXt Lbl Sin R6 xGEt Tan xXt
234: GTO Tan Lbl Cos $\sqrt{x}$ SUM8 ± X 2 + R8 + xXt = Op10 X xXt Abs INV
256: y^x 3 + R8 Op10 X R8 Abs INV y^x 3 Lbl Tan - R7 = S6 ÷ 2 = S5 +
284: ((x^2 - R0 + Abs) ÷ 2) $\sqrt{x}$ INV SUM5 = S7 R3 ÷ 2 = S8 + (x^2 + R6
316: - R2) $\sqrt{x}$ INV SUM8 = S6 X R7 + R8 X R5 - R1 = Abs xXt R6 X R5 +
348: R8 X R7 - R1 = Abs xGEt Exc R5 Exc7 S5 Lbl Exc R4 Prd0 Prd1 Prd2
376: Prd3 0 H8 1 Op4 H18 CP x=t Prd INV Fix A' B' C' D' Adv Adv Adv
398: Lbl Prd R/S LA S4 xXt 1305 Op4 xXt Op6 R/S LB S3 xXt 1304 Op4
427: xXt Op6 R/S LC S2 xXt 1303 Op4 xXt Op6 R/S LD S1 xXt 1302 Op4
457: xXt Op6 R/S LE S0 xXt 1301 Op4 xXt Op6 R/S GTO E'

\- - - - - - - - - -

THE MATH/UTILITIES CROM (V3N8p5,6)

A few pre-production modules have now been made, and TI expects production to begin mid-December, with first deliveries expected by early January. Many of the programs in this new CROM do indeed reflect better programming quality than has gone into earlier ones, and the routines themselves will probably turn out to be the most universally useful. Fortunately, subroutine callability (from RAM) was one of the design criteria. Following is a brief run-down of the 21 programs:

MU-01 is the usual module check and register-clearer. (60 steps)

MU-02works only with the PC, providing a few common prompting messages, including multiple-card read-write directions with code which cleverly interrogates the first card-side read to determine how many more sides are to be read. Unfortunately, the prompter refers to "cards" instead of "sides", which may confuse some users.(329 steps)

MU-03 is similar to the LE-10 Memopad, but with more symbol-code keyboard-addressable, and with the words ENTER, PRESS, and PRESS SBR single-key specifiable. Users will need to make their own overlays for finding the alpha prefixes, as TI budget constraints did not allow for one to be included in the module package. (474 steps)

MU-04 is used with MU-03 to format both data and text in any desired configuration for each line. It's slow, but can save paper. (430 steps)

MU-05 is titled Superplotter, and makes it easy for the user to generate plots of up to ten different functions, with size and precision limited only by acceptable execution time and paper. The result is one long tape which the user cuts and reassembles with the aid of well-chosen alignment and coordinate symbols. This looks like a winner. (444 steps)

MU-06 is the fast sorter (V3N8p6) which orders one sequence of 99 random numbers in 9¼ minutes, but which takes 19½ minutes for the same code executed in RAM. The V3N2p5 program, which was TI's point of departure, takes 23 1/3 minutes to order the same sequence. The MU-06 interface with the user was well planned, giving the user a good choice of I/O options for either keyboard or RAM program call. (132 steps)

MU-07 is a data array processor, and at 650 steps is the longest MU program. Up to 93 (53 for the 58) elements of a 2-dimensional array can be arithmetically combined or changed by row or column, and the elements shifted right or left by row. There are a lot of business oriented manipulative options, probably best run via a RAM program with lots of prompting. MU-07 appears to be an extension of the Business Decisions Project Planning and Budgeting program (BD-05).

MU-08 is a general-purpose data packer and unpacker of positive integers, allowing the user to specify mixed as well as constant data lengths for 13-digit per register packing into Registers 4 onward to the end of partitioning. The 3 functions: store, recall, and exchange each take about 4 seconds to execute, and may be invoked in any order any number of times. (226 steps)

MU-09 is a disappointing factor finder: It's short (TI claims it needed to cut down on memory requirements, although MU-09 is only 7 steps shorter than the V3N11p4 program), but slow (doesn't bypass trial divisors which are multiples of smaller ones already tried (V3N11p4), and does quite a bit of flag manipulating). (85 steps)

MU-10 is a fast, straightforward generator of hyperbolic trig functions, and their inverses. (100 steps)

MU-11 calculates the gamma function and factorials for both integers and positive reals up to 69 or 70, and the natural logs of these 2 functions for reals up to about 10^{10}. (147 steps)

MU-12 is a short random number generator which produces uniformly distributed numbers in the 0-1 range the same way ML-15 does, but a bit more efficiently. It uses MU-13 to generate normally distributed numbers (54 steps)

MU-13 does normal distribution processing, 2 of the functions producing the same results as 2 of the Applied Statistics Normal Distribution program (ST-19); Routines A and C of MU-13 correspond to routines C and A, respectively, of ST-19. (260 steps)

MU-14 mechanizes the Aitken interpolation algorithm, fitting up to a 26th order polynomial to 27 data points (13th order for 14 data points for the 58), which is then used to solve for intermediate points. (193 steps)

MU-15 is a primitive root finder, requiring the user to provide approximations to each (real only) root. It does provide an option to specify the max number of iterations for cases where convergence is expected to be slow or maybe nonexistent. (99 steps)

MU-16 finds the max and min values of user-supplied functions in specifiable intervals by looking for a change in sign of an approximation to the function's derivative. This approximation (also used by MU-15) imposes processing limitations of which the user needs to be aware. It would seem that a straight forward sample-and-save-largest-and-smallest approach (V2N8p4,5) would have been better. (211 steps)

MU-17 mechanizes Romberg integration of definite integrals, allowing the user to specify accuracy. (243 steps)

MU-18 uses a 4th-order Runge-Kutta method to solve differential equations of the y'=f(x,y) and y"=f(x,y,y') types. (293 steps)

MU-19 performs discrete Fourier series summations on as many input equally spaced data points as there are registers available from Reg 16 onward. If a user's RAM program needs a few registers from Reg 16 on, he can specify a higher start register for MU-19 to use for input storage. (115 steps)

MU-20 does a good job of determining, recording, and resetting the status (V3N2p2) of flags, partitioning, angle mode, and Fix. The number of open parentheses, and printer connection can be determined, but not recorded or reset. (307 steps)

MU-21 is designed to aid manual (keyboard) calculations by assigning 5 of the user-defined keys to specified variables and subroutines. I can see where single-key variable recall might help to speed things up for the non-programmer, but since the user has to write whatever subroutines he wants, anyway, he might as well call them directly. (101 steps)

I'll comment further in future articles, as I delve into the MU programs in greater detail, and/or start getting inputs from the membership.

- - - - - - - - - - - - - -

REVEALED FIRMWARE (V3N10p4, V3N11p1,2)

Steve Bepko (45), Maurice Swinnen (779), Dave Leising (890), and John Mickelsen (990) have all found more ROM code past step 487. Steve and John got to step 575 by single stepping without the printer; Maurice and Dave to step 583 with the printer in trace mode. Apparently no one else had bothered to SST past step 487, since what Steve and John did works with any of the established ROM-revelation procedures. For a shortcut confirmation, key at turn-on: GTO 479 9 Op17 Pgm 12 SBR 444 R/S P/R LRN, and SST to step 575, at which point the next SST causes a switch to RUN mode. If LRN is then pressed, the whole keyboard appears to be locked out with a displayed C.

Confirmation of Maurice's and Dave's approach can be made with a 59/PC at turn-on with: Stflg 9 9 Op17 Pgm12 SBR 444 R/S P/R LRN BST which lists steps 000-583 without mnemonics, and can only be stopped by turning the machine off. If instead of keying the Stflg 9 you press (latch) the TRACE key, printing will stop when the TRACE key is unlatched.

Steve found that some regular (with mnemonics) listing could be made past step 487 by first SSTing past step 487, then keying LRN List. It turns out that step 489 is the lowest that will work, and in any case step 503 is as high as it goes, starting over at step 039 set to code 80 (Grd) this time, and continuing on listing RAM from step 040 to the end of the current partition.

Bill Skillman (710) identifies steps 384-511 as non-normalized constants used by the transcendental functions. (Steps 512-575 repeat steps 384-447). Bill finds that if the significant digits in the equivalent 16 data registers are allowed to be rescaled, they can be interpreted as: ln 10, ln 2, ln 1.1, ln 1.01, ln 1.001, ln 1.0001, ln 1.00001, ln 1.000001, $\pi/4$, arctan .1, arctan .01, arctan .001, arctan .0001, $\pi/2$, π, and $180/\pi$. If steps 384-447 are written in RAM, and recalled as data, these 16 constants appear as: -ln 10 X 10, ln 2 X 10^{-94}, ln 1.1 X 10^{-32}, ln 1.01 X 10^{102}, ln 1.001 X 10^{-8} , -ln 1.0001 X 10^{33}, ln 1.00001 (no change), ln 1.000001 X 10^{50}, $\pi/4$ X 10^{45}, arctan .1 X 10^{20}, arctan .01 X 10^{67}, -arctan .001 X 10^{-66}, -arctan .0001 X X 10^{-66}, $-\pi/2$ X 10^{2}, $-\pi$ X 10, and $-180/\pi$ X 10. While the revelation of these constants raises new questions as to how data are formatted during transcendental function processing, their identity and precision should help in the determination of the algorithms used (V2N9p6).

Dave reports that the ROM resides "... in the TMC 0571 chip located on the lower right corner of the PC board. This chip is essentially a 'CROM' intended for the storage of resident..." built-in functions.

- - - - - - - - - - -

TWO NEW PERIODICALS

Ken and Jon Mills, who used to write a Recreational Programmer column in 65-NOTES, have begun publishing an independent bimonthly magazine of the same title. V1N1 is dated Sept-Oct 78, and addresses an assortment of calculator-computer topics including business programming, transcendental functions, celestial navigation computations, a couple of games, PPC cryptography, and a book review. V1N2 continues the business programming column and another book review, adding several more games. So far, most of the material is HP PPC oriented, but the fairly generous flow charting and English descriptions accompanying program listings should facilitate translation to other machines. Contributing authors come across as enthusiastic, writing in an easy style, though at times somewhat rambling and short on rigor. Subscription is $12 per year in the US; $15 elsewhere. For more information write to: The Recreational Programmer Box 2571 Kalamazoo, MI 49003.

Didactic Programming, A Journal of Calculator-Demonstrated Math Instruction, began with a Fall 1978 issue aimed at math instructors who are integrating PPCs into their classrooms. Topics range over iterative equation solving, Fibonacci Search for relative minima, Gaussian Elimination, and a 3 simultaneous equations solver. Continuation and periodicity of this publication will be determined largely by the amount of contributed material received. For further information, write to Didactic Programming Box 974 Laguna Beach, CA 92652.

TIPS AND MISCELLANY

Membership Address Changes: 343: RR 1 Box 176 Blue Earth, MN 56013; 606: 1740 N Cherry St Mesa, AZ 85201; 1031: 2141 Steiger Ln Oceanside, CA 92054; 1072: 18-82 47th St Brooklyn, NY 11204.

The Washington (DC) Area Local Club: The only organized TI PPC local group I am aware of is the one Dave Johnston (5) and Maurice Swinnen (779) started (V3N4p5). Maurice reports a current regular membership of 15, and some Friendly Competition with a local HP PPC group. A current challenge is to use only the functions: = (or ENTER), xXt (or xXy, or Exc 00), y^x, $1/x$, $\sqrt{x}$, x^2, and INV lnx (or e^x) on a PPC at turn-on, and display a 3 with the fewest steps, where INV lnx and Exc 00 each count only as one step. No other key may be used, including the numerals and CLR. Maurice has a 14-step solution, which I've trimmed to 11. Send your best to Maurice or me. It should be interesting to see whether a TI or HP PPC is the ultimate winner. The Washington area members have been noteworthy contributors to 52-NOTES, both individually and collectively, and I know I speak for all of us when I hope they keep up their productive activities.