COMPLEX ROOTS

PURPOSE

Compute the complex roots of a polynomial with complex (or real) coefficients.

DESCRIPTION

DATAPLOT stores all variables as reals. Complex variables are supported as a pair of real variables. That is, the pair Y1, Y2 of real variables can be thought of as the single complex variable Y1 + i*Y2 where i is the square root of -1.

By the fundamental thereom of algebra, every polynomial of degree n where n is greater than or equal to 1 can be written as:

$$p(z) = c(z-z_1)(z-z_2)...(z-z_n)$$

where c and z_k are complex constants. This means that an nth degree polynomial has exactly n complex roots.

SYNTAX 1

LET <v3> <v4> = COMPLEX ROOTS <v1> <v2> <SUBSET/EXCEPT/FOR qualification>

where <v1> and <v2> are the real and imaginary components of the ordered polynomial coefficients:
element 1 is the coefficients of the constant term;
element 2 is the coefficients of the linear term;
element 3 is the coefficients of the quadratic term;
element 4 is the coefficient of the cubic term;
etc.
<v3> and <v4> are the real and imaginary components of the output roots:
element 1 is the first root;
element 2 is the second root;
element 3 is the third root;
element 4 is the fourth root;
etc.
and where the <SUBSET/EXCEPT/FOR qualification> is optional and rarely used in this context.

SYNTAX 2

LET <v3> <v4> = COMPLEX ROOTS <v1> <SUBSET/EXCEPT/FOR qualification>

This is the same as syntax 1 except <v2> is omitted. This syntax allows one to compute the complex roots of a polynomial with real coefficients. In practice, syntax 2 is more common than syntax 1.

EXAMPLES

LET Y3 Y4 = COMPLEX ROOTS Y1 Y2
LET Y3 Y4 = COMPLEX ROOTS Y1 Y2 SUBSET Y1 > 10
LET Y3 Y4 = COMPLEX ROOTS Y1 Y2 FOR I = 1 1 20
LET Y3 Y4 = COMPLEX ROOTS Y1

NOTE

DATAPLOT uses the routine CPZERO, written by Dr. David Kahanner of the National Institute of Standards and Technology, from the SLATEC Common Mathematical Library to compute this function. SLATEC is a large set of high quality, portable public domain Fortran routines for various mathematical capabilities maintained by seven federal laboratories. Versions of DATAPLOT prior to 95/8 use the LAGUER routine from the Numerical Recipes book (see the REFERENCE section below).

DEFAULT

None

SYNONYMS

None

RELATED COMMANDS

ROOTS	=	Computes the real roots of a function.
COMPLEX ADDITION	=	Carries out complex addition.
COMPLEX SUBTRACTION	=	Carries out complex subtraction.

COMPLEX MULTIPLICATION	=	Carries out complex multiplication.
COMPLEX DIVISION	=	Carries out complex division.
COMPLEX EXPONENTIATION	=	Carries out complex exponentiation.
COMPLEX SQUARE ROOT	=	Computes the complex square root.
COMPLEX CONJUGATE	=	Computes the complex conjugate.
POLYNOMIAL EVALUATION	=	Carries out a polynomial evaluation.
MATRIX SOLUTION	=	Computes a matrix solution.
MATRIX EIGENVALUES	=	Computes the eigenvalues of a matrix.
MATRIX SIMPLEX SOLUTION	=	Computes a simplex solution.
FFT	=	Computes the Fast Fourier Transform.
INVERSE FFT	=	Computes the Inverse FFT.
PLOT	=	Plots data or functions.

REFERENCE

“Numerical Recipes: The Art of Scientific Computing (FORTRAN Version),” Press, Flannery, Teukolsky, and Vetterling. Cambridge University Press, 1989 (pages 263-266).

APPLICATIONS

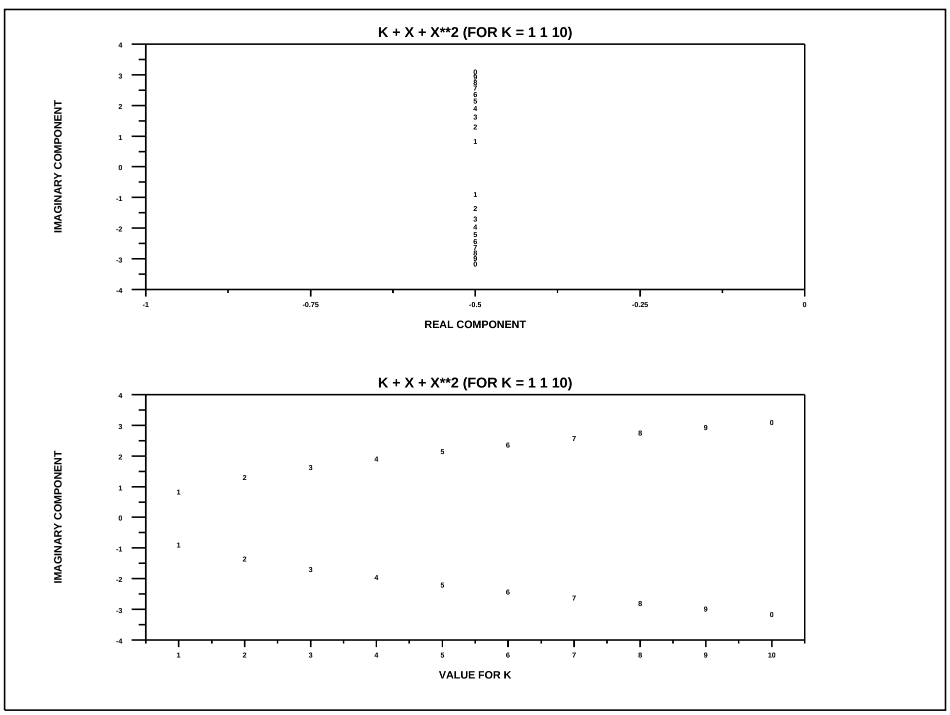
Mathematics, Digital Filter Design

IMPLEMENTATION DATE

87/10

PROGRAM 1

```
. PURPOSE--PLOT OUT THE COMPLEX ROOTS FROM THE FAMILY OF FUNCTIONS
.                   K + 1*X + 1*X**2
. ANALYSIS TECHNIQUE--COMPLEX ROOTS AND PLOT
DIMENSION 20 VARIABLES
. STEP 1--DEFINE THE BASE POLYNOMIAL
LET P = DATA 1 1 1
. STEP 2--DEFINE DUMMY VARIABLES (NEEDED LATER)
LET X2 = DATA -999 -999
LET Y2 = DATA -999 -999
LET D2 = DATA -999 -999
. STEP 3--EXECUTE A LOOP. FOR EACH ITERATION, CHANGE THE BASE
.           POLYNOMIAL TO K + 1*X + 1*X**2. COMPUTE AND STORE THE ROOTE
LOOP FOR K = 1 1 10
    LET P(1) = K
    LET X Y = COMPLEX ROOTS P
    LET D = K FOR I = 1 1 2
    APPEND X X2; APPEND Y Y2; APPEND D D2
END OF LOOP
. STEP 4--PLOT THE ROOTS
MULTIPLOT 2 1; MULTIPLOT CORNER COORDINATES 0 0 100 100
CHAR 1 2 3 4 5 6 7 8 9 0; LINES BLANK ALL
TITLE K + X + X**2 (FOR K = 1 1 10); TITLE SIZE 4
X1LABEL REAL COMPONENT; Y1LABEL IMAGINARY COMPONENT
PLOT Y2 X2 D2 EXCEPT D2 = -999
X1LABEL VALUE FOR K
XTIC OFFSET 0.5 0.5
PLOT Y2 D2 D2 EXCEPT D2 = -999
END OF MULTIPLOT
```



PROGRAM 2

```
. PURPOSE--ASSESS THE STABILITY OF A LINEAR RECURSIVE DIGITAL FILTER
.                    Y(I) = SUM(1,M) C(J)*X(I-J)   +   SUM(1,N) D(J)*Y(I-J)
. SOURCE (FOR PROBLEM)--PRESS, FLANNERY, TEUKOLSKY, AND VETTERLING
.                   NUMERICAL RECIPES--THE ART OF SCIENTIFIC
.                   COMPUTING, CAMBRIDGE UNIVERSITY PRESS, 1986, PAGES 439-440.
. NOTE--IN A STABLE FILTER, THE OUTPUT WILL EVENTUALLY STOP (AFTER M
.      STEPS) WHEN THE INPUT STOPS.  IN AN UNSTABLE FILTER,
.      THE OUTPUT MAY GROW EXPONENTIALLY EVEN AFTER STOPPING THE INPUT.
. NOTE--A FILTER IS STABLE IF AND ONLY IF THE N COMPLEX ROOTS OF
.                X**N - D1*X**(N-1) - D2*X**(N-2) - ... - DN = 0
.          ALL FALL INSIDE OR ON THE UNIT CIRCLE.
. NOTE--FOR TESTING PURPOSES, THE FILTER 16 8 4 2 1 WILL BE UNSTABLE.
.          AN EXAMPLE OF A STABLE FILTER IS 1 -1 .4
. STEP 1--DEFINE THE FILTER COEFFICIENTS
.           16 IS THE WEIGHT FOR Y(I-1), 8 IS THE WEIGHT FOR Y(I-2), ETC.
LET D = DATA 16 8 4 2 1
LET N = NUMBER D
. STEP 2--REARRANGE TO FORM A POLYNOMIAL WITH COEFFICIENTS -1 -2 -4 -8 -16 +1
LOOP FOR I = 1 1 N
     LET IREV = N-I+1; LET DIREV = D(IREV); LET P(I) = DIREV
END OF LOOP
LET P = -P; LET NP1 = N+1; LET P(NP1) = 1
. STEP 3--COMPUTE  AND PLOT THE COMPLEX ROOTS (AND A UNIT CIRCLE)
LET X Y = COMPLEX ROOTS P
LINES BLANK SOLID SOLID; CHAR X BLANK BLANK
X1LABEL REAL COMPONENT; Y1LABEL IMAGINARY COMPONENT
PLOT Y X AND
PLOT SQRT(1-X**2) FOR X = -1 .01 1 AND
PLOT -SQRT(1-X**2) FOR X = -1 .01 1
```