



GALAHAD ROOTS

PACKAGE SPECIFICATION

GALAHAD Optimization Library version 2.1

SUMMARY

This package uses classical formulae together with Newton's method to find all the real roots of real polynomials of degree up to four.

ATTRIBUTES — Versions: GALAHAD_ROOTS_single, GALAHAD_ROOTS_double. Date: March 2006. Origin: N. I. M. Gould, Rutherford Appleton Laboratory. Language: Fortran 95 + TR 15581 or Fortran 2003.

2 HOW TO USE THE PACKAGE

Access to the package requires a USE statement such as

Single precision version

USE GALAHAD_ROOTS_single

Double precision version

USE GALAHAD_ROOTS_double

If it is required to use both modules at the same time, the subroutine ROOTS_solve, (Section 2.1) must be renamed on one of the USE statements.

2.1 Argument lists and calling sequences

There is a single procedure for user calls. Subroutine ROOTS_solve is called to find the real roots of the polynomial

$$\sum_{i=0}^{d} a_i x^i \tag{2.1}$$

of degree d, where the coefficients a_i , $0 \le i \le d$ are real.

2.1.1 The solution subroutine

The roots of the polynomial (2.1) are found as follows

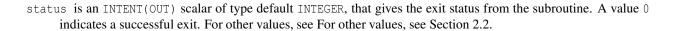
CALL ROOTS_solve(A, tol, nroots, ROOTS, status)

- is an INTENT(IN) rank-one array of type default REAL (double precision in GALAHAD_ROOTS_double), whose lower bound must be 0 and whose upper bound specifies the degree, d, of the polynomial. The entries A(i), i = 0, ..., UBOUND (A), must contain the values of the real coefficients a_i , $0 \le i \le d$.
- tol is an INTENT(IN) scalar of type default REAL (double precision in GALAHAD_ROOTS_double) that should be set to the required accuracy of the roots. Every effort will be taken to ensure that each computed root x_c lies within \pm tol x_e of its exact equivalent x_e , although sometimes the required accuracy will not be possible.

nroots is an INTENT(OUT) scalar of type default INTEGER, that gives the number of real roots of the polynomial.

ROOTS is an INTENT (OUT) rank-one array of length d and type default REAL (double precision in GALAHAD_ROOTS_ double). On exit, ROOTS(:nroots) give the values of the real roots of the polynomial in increasing order.

All use is subject to licence. See http://galahad.rl.ac.uk/galahad-www/cou.html. For any commercial application, a separate license must be signed.



2.2 Warning and error messages

A negative value of info%status on exit from ROOTS_solve indicates that an error has occurred. No further calls should be made until the error has been corrected. Possible values are:

- -3. The upper bound of the array A indicates that the degree of the polynomial is not in the range 0 to 4.
- -4. The upper bound of the array ROOTS is not the same as that of A.

3 GENERAL INFORMATION

Use of common: None.

Workspace: None.

Other routines called directly: None.

Other modules used directly: None.

Input/output: None.

Portability: ISO Fortran 95 + TR 15581 or Fortran 2003. The package is thread-safe.

4 METHOD

Littlewood and Ferrari's algorithms are used to find estimates of the real roots of cubic and quartic polynomials, respectively; a stabilized version of the well-known formula is used in the quadratic case. Newton's method is used to further refine the computed roots if necessary.

5 EXAMPLE OF USE

Suppose we wish to solve the quadratic, cubic and quartic equations

$$x^{2}-3x+2=0$$

$$x^{3}-6x^{2}+11x-6=0 \text{ and }$$

$$x^{4}-10x^{3}+35x^{2}-50x+24=0.$$

Then we may use the following code:

All use is subject to licence. See http://galahad.rl.ac.uk/galahad-www/cou.html. For any commercial application, a separate license must be signed.



```
A(1) = -3.0_{wp}
   A(2) = 1.0_{wp}
   WRITE( 6, "( ' Quadratic ' )" )
   CALL ROOTS_solve( A( : degree ), tol, nroots, ROOTS( : degree ), status )
  ELSE IF ( degree == 3 ) THEN
   A(0) = -6.0_{wp}
   A(1) = 11.0_{wp}
   A(2) = -6.0 \text{wp}
   A(3) = 1.0_{wp}
   WRITE( 6, "( /, ' Cubic ' )" )
   CALL ROOTS_solve( A( : degree ), tol, nroots, ROOTS( : degree ), status )
   A(0) = 24.0_{wp}
   A(1) = -50.0 \text{wp}
   A(2) = 35.0_wp
   A(3) = -10.0_{wp}
   A(4) = 1.0_wp
   WRITE( 6, "( /, ' Quartic ' )" )
   CALL ROOTS_solve( A( : degree ), tol, nroots, ROOTS( : degree ), status )
  END IF
  IF ( nroots == 0 ) THEN
   WRITE( 6, "( ' no real roots ' )" )
 ELSE IF ( nroots == 1 ) THEN
   WRITE( 6, "( ' 1 real root ' )" )
 ELSE IF ( nroots == 2 ) THEN
   WRITE( 6, "( ' 2 real roots ')")
 ELSE IF ( nroots == 3 ) THEN
   WRITE( 6, "( ' 3 real roots ' )" )
 ELSE IF ( nroots == 4 ) THEN
   WRITE( 6, "( ' 4 real roots ')" )
 END IF
 IF ( nroots /= 0 ) WRITE( 6, "( ' roots: ', 4ES10.2 )" ) ROOTS( : nroots )
END DO
END PROGRAM GALAHAD_ROOTS_EXAMPLE
```

This produces the following output:

```
Quadratic
2 real roots
roots: 1.00E+00 2.00E+00
Cubic
3 real roots
roots: 1.00E+00 2.00E+00 3.00E+00
Quartic
4 real roots
roots: 1.00E+00 2.00E+00 3.00E+00 4.00E+00
```

All use is subject to licence. See http://galahad.rl.ac.uk/galahad-www/cou.html. For any commercial application, a separate license must be signed.