NAG C Library Function Document

nag_quartic_roots (c02alc)

1 Purpose

nag_quartic_roots (c02alc) determines the roots of a quartic equation with real coefficients.

2 Specification

    void nag_quartic_roots (double e, double a, double b, double c, double d,
                          double zeror[], double zeroi[], double errest[], NagError *fail)

3 Description

nag_quartic_roots attempts to find the roots of the quartic equation

\[ ez^4 + az^3 + bz^2 + cz + d = 0, \]

where \( e, a, b, c \) and \( d \) are real coefficients with \( e \neq 0 \). The roots are located by finding the eigenvalues of the associated 4 by 4 (upper Hessenberg) companion matrix \( H \) given by

\[
    H = \begin{pmatrix}
        0 & 0 & 0 & -d/e \\
        1 & 0 & 0 & -c/e \\
        0 & 1 & 0 & -b/e \\
        0 & 0 & 1 & -a/e \\
    \end{pmatrix}.
\]

Further details can be found in Section 6.

To obtain the roots of a cubic equation, nag_cubic_roots (c02akc) can be used.

4 Parameters

1: \( e \) – double

    On entry: \( e \), the coefficient of \( z^4 \).
    Constraint: \( e \neq 0 \).

2: \( a \) – double

    On entry: \( a \), the coefficient of \( z^3 \).

3: \( b \) – double

    On entry: \( b \), the coefficient of \( z^2 \).

4: \( c \) – double

    On entry: \( c \), the coefficient of \( z \).

5: \( d \) – double

    On entry: \( d \), the constant coefficient.

6: \( \text{zeror}[4] \) – double

    On exit: \( \text{zeror}[i-1] \) and \( \text{zeroi}[i-1] \) contain the real and imaginary parts, respectively, of the \( i \)th root.

7: \( \text{zeroi}[4] \) – double

    On exit: \( \text{zeror}[i-1] \) and \( \text{zeroi}[i-1] \) contain the real and imaginary parts, respectively, of the \( i \)th root.
8:   **errest[4]** – double  

   *Output*
   
   On exit: errest[i – 1] contains an approximate error estimate for the i-th root.

9:   **fail** – NagError *  

   *Input/Output*
   
   The NAG error parameter (see the Essential Introduction).

5   **Error Indicators and Warnings**

   **NE_REAL**
   
   On entry, e = 0.0.
   
   Constraint: e ≠ 0.0.

   **NE_C02_OVERFLOW**
   
   The companion matrix H cannot be formed without overflow.

   **NE_C02_NOT_CONV**
   
   The iterative procedure used to determine the eigenvalues has failed to converge.

   **NE_INTERNAL_ERROR**
   
   An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

6   **Further Comments**

   The method used by the routine consists of the following steps, which are performed by routines from LAPACK.

   (a) Form matrix H.

   (b) Apply a diagonal similarity transformation to H (to give H').

   (c) Calculate the eigenvalues and Schur factorization of H'.

   (d) Calculate the left and right eigenvectors of H'.

   (e) Estimate reciprocal condition numbers for all the eigenvalues of H'.

   (f) Calculate approximate error estimates for all the eigenvalues of H' (using the 1-norm).

6.1   **Accuracy**

   If **fail.code** = **NE_NOERROR** on exit, then the i-th computed root should have approximately \( |\log_{10}(\text{errest}[i - 1])| \) correct significant digits.

6.2   **References**


7   **See Also**

   nag_cubic_roots (c02akc)
8. Example

To find the roots of the quartic equation

\[ z^4 + 2z^3 + 6z^2 - 8z - 40 = 0. \]

8.1 Program Text

/* nag_quartic_roots (c02alc) Example Program. */
/* Copyright 2000 Numerical Algorithms Group. */
/* NAG C Library */
/* Mark 6, 2000. */

#include <nag.h>
#include <nag_stdlib.h>
#include <nagc02.h>

int main(void)
{
    double a, b, c, d, e;
    double *errest=0, *zeroi=0, *zeror=0;
    Integer i;
    Integer exit_status=0;
    NagError fail;

    INIT_FAIL(fail);
    Vprintf("c02alc Example Program Results\n\n");
    if (TRUE or errest == NAG_ALLOC(4,double) or zeroi == NAG_ALLOC(4,double) or zeror == NAG_ALLOC(4,double))
    {
        Vprintf("Allocation failure\n");
        exit_status=-1;
        goto END;
    }

    /* Skip heading in data file */
    Vscanf("%*[\n]");
    Vscanf("%lf %lf %lf %lf %lf", &e, &a, &b, &c, &d);

    c02alc (e, a, b, c, d, zeror, zeroi, errest, &fail);
    if (fail.code == NE_NOERROR)
    {
        Vprintf("Roots of quartic equation Error estimates\n");
        Vprintf("(machine-dependent)\n\n");
        for (i = 0; i <= 3; ++i)
        {
            Vprintf("%s %10.5f %10.5f%8g\n", " z =",
                     zeror[i], zeroi[i], "i", errest[i]);
        }
    }
}

[NP3491/6]
else
{
    Vprintf("Error from c02alc.\n\n", fail.message);
    exit_status = 1;
    goto END;
}
END:
if (errest) NAG_FREE (errest);
if (zeroi)  NAG_FREE (zeroi);
if (zeror)  NAG_FREE (zeror);
return exit_status;
}

8.2 Program Data

c02alc Example Program Data
 1.0  2.0  6.0  -8.0  -40.0 : Values of e, a, b, c and d

8.3 Program Results

c02alc Example Program Results

<table>
<thead>
<tr>
<th>Roots of quartic equation</th>
<th>Error estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(machine-dependent)</td>
</tr>
<tr>
<td>z = 2.00000 0.000000*i</td>
<td>3.38974e-15</td>
</tr>
<tr>
<td>z = -2.00000 0.000000*i</td>
<td>5.29396e-15</td>
</tr>
<tr>
<td>z = -1.00000 3.000000*i</td>
<td>4.54379e-15</td>
</tr>
<tr>
<td>z = -1.00000 -3.000000*i</td>
<td>4.54379e-15</td>
</tr>
</tbody>
</table>