NAG Library Function Document
nag_dptcon (f07jgc)

1 Purpose
nag_dptcon (f07jgc) computes the reciprocal condition number of a real $n$ by $n$ symmetric positive definite tridiagonal matrix $A$, using the $LDL^T$ factorization returned by nag_dpttrf (f07jdc).

2 Specification

```c
#include <nag.h>
#include <nagf07.h>
void nag_dptcon (Integer n, const double d[], const double e[], double anorm,
                 double *rcond, NagError *fail)
```

3 Description

nag_dptcon (f07jgc) should be preceded by a call to nag_dpttrf (f07jdc), which computes a modified Cholesky factorization of the matrix $A$ as

$$A = LDL^T,$$

where $L$ is a unit lower bidiagonal matrix and $D$ is a diagonal matrix, with positive diagonal elements. nag_dptcon (f07jgc) then utilizes the factorization to compute $1/\kappa_1(A)$ by a direct method, from which the reciprocal of the condition number of $A$, $1/\kappa(A)$ is computed as

$$1/\kappa_1(A) = 1/\left(\|A\|_1\|A^{-1}\|_1\right).$$

$1/\kappa_1(A)$ is returned, rather than $\kappa(A)$, since when $A$ is singular $\kappa(A)$ is infinite.

4 References


5 Arguments

1:  **n** – Integer

   *On entry:* $n$, the order of the matrix $A$.

   *Constraint:* $n \geq 0$.

2:  **d[**$\dim$**]** – const double

   *On entry:* must contain the $n$ diagonal elements of the diagonal matrix $D$ from the $LDL^T$ factorization of $A$.

   *Note:* the dimension, $\dim$, of the array $d$ must be at least $\max(1, n)$.

3:  **e[**$\dim$**]** – const double

   *On entry:* must contain the $(n-1)$ subdiagonal elements of the unit lower bidiagonal matrix $L$. ($e$ can also be regarded as the superdiagonal of the unit upper bidiagonal matrix $U$ from the $U^TDU$ factorization of $A$.)

   *Note:* the dimension, $\dim$, of the array $e$ must be at least $\max(1, n-1)$.
4:   anorm – double

   Input

   On entry: the 1-norm of the original matrix A, which may be computed as shown in Section 10.
   anorm must be computed either before calling nag_dpttrf (f07jdc) or else from a copy of the
   original matrix A.

   Constraint: anorm ≥ 0.0.

5:   rcond – double *

   Output

   On exit: the reciprocal condition number, 1/κ₁(A) = 1/\left(\|A\|₁\|A⁻¹\|₁\right).

6:   fail – NagError *

   Input/Output

   The NAG error argument (see Section 3.6 in the Essential Introduction).

6   Error Indicators and Warnings

NE_ALLOC_FAIL

   Dynamic memory allocation failed.
   See Section 3.2.1.2 in the Essential Introduction for further information.

NE_BAD_PARAM

   On entry, argument value had an illegal value.

NE_INT

   On entry, n = \langle value\rangle.
   Constraint: n ≥ 0.

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 contact NAG for assistance.
   An unexpected error has been triggered by this function. Please contact NAG.
   See Section 3.6.6 in the Essential Introduction for further information.

NE_NO_LICENCE

   Your licence key may have expired or may not have been installed correctly.
   See Section 3.6.5 in the Essential Introduction for further information.

NE_REAL

   On entry, anorm = \langle value\rangle.
   Constraint: anorm ≥ 0.0.

7   Accuracy

   The computed condition number will be the exact condition number for a closely neighbouring matrix.

8   Parallelism and Performance

   Not applicable.

9   Further Comments

   The condition number estimation requires \(O(n)\) floating-point operations.
See Section 15.6 of Higham (2002) for further details on computing the condition number of tridiagonal matrices.

The complex analogue of this function is nag_zptcon (f07juc).

10 Example

This example computes the condition number of the symmetric positive definite tridiagonal matrix $A$ given by

$$A = \begin{pmatrix}
4.0 & -2.0 & 0 & 0 & 0 \\
-2.0 & 10.0 & -6.0 & 0 & 0 \\
0 & -6.0 & 29.0 & 15.0 & 0 \\
0 & 0 & 15.0 & 25.0 & 8.0 \\
0 & 0 & 0 & 8.0 & 5.0 \\
\end{pmatrix}$$

10.1 Program Text

/* nag_dptcon (f07jgc) Example Program. *
 * Copyright 2014 Numerical Algorithms Group. *
 * Mark 23, 2011. *
 * UNFINISHED - replace commented out climp calls */
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf07.h>
#include <nagx02.h>

int main(void)
{
    /* Scalars */
    double anorm, rcond;
    Integer exit_status = 0, i, n;
    /* Arrays */
    double *d = 0, *e = 0;
    /* Nit Types */
    NagError fail;

    INIT_FAIL(fail);

    printf("nag_dptcon (f07jgc) Example Program Results\n\n");
    /* Skip heading in data file */
    #ifdef _WIN32
    scanf_s("%*\[^
");
    #else
    scanf("%*\[^
");
    #endif
    #ifdef _WIN32
    scanf_s("%"NAG_IFMT"%*\[^
", &n);
    #else
    scanf("%"NAG_IFMT"%*\[^
", &n);
    #endif
    if (n < 0)
    {
        printf("Invalid n\n");
        exit_status = 1;
        goto END;
    }
    /* Allocate memory */
    if (!(d = NAG_ALLOC(n, double)) ||
      !(e = NAG_ALLOC(n-1, double)))

/* Read the lower bidiagonal part of the tridiagonal matrix A from */
/* data file */

#ifdef _WIN32
    for (i = 0; i < n; ++i) scanf_s("%lf", &d[i]);
#else
    for (i = 0; i < n; ++i) scanf("%lf", &d[i]);
#endif
#ifdef _WIN32
    scanf_s("%*[\n]");
#else
    scanf("%*[\n]");
#endif
#ifdef _WIN32
    for (i = 0; i < n - 1; ++i) scanf_s("%lf", &e[i]);
#else
    for (i = 0; i < n - 1; ++i) scanf("%lf", &e[i]);
#endif
#ifdef _WIN32
    scanf_s("%*[\n]");
#else
    scanf("%*[\n]");
#endif

/* Compute the 1-norm of A */
anorm = MAX(ABS(d[0])+ABS(e[0]), ABS(e[n-2])+ABS(d[n-1]));
for (i = 1; i < n-1; ++i)
    anorm = MAX(anorm, ABS(d[i])+ABS(e[i])+ABS(e[i-1]));

/* Factorize the tridiagonal matrix A using nag_dgbsv (f07bac). */
*nag_dpttrf(n, d, e, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_dgbsv (f07bac).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Estimate the condition number of A using nag_dptcon (f07jgc). */
nag_dptcon(n, d, e, anorm, &rcond, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_dptcon (f07jgc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Print the estimated condition number */
if (rcond >= nag_machine_precision)
    printf("Estimate of condition number = %11.2e\n\n", 1.0/rcond);
else
    printf("A is singular to working precision. RCOND = %11.2e\n\n", rcond);

END:
NAG_FREE(d);
NAG_FREE(e);
return exit_status;
}
10.2 Program Data

nag_dptcon (f07jgc) Example Program Data

\[
\begin{array}{c}
5 \\
4.0 & 10.0 & 29.0 & 25.0 & 5.0 : \text{diagonal } d \\
-2.0 & -6.0 & 15.0 & 8.0 : \text{sub-diagonal } e
\end{array}
\]

10.3 Program Results

nag_dptcon (f07jgc) Example Program Results

Estimate of condition number = 1.05e+02