NAG Library Function Document

nag_5d_shep_interp (e01tmc)

1 Purpose

nag_5d_shep_interp (e01tmc) generates a five-dimensional interpolant to a set of scattered data points, using a modified Shepard method.

2 Specification

```c
#include <nag.h>
#include <nage01.h>

void nag_5d_shep_interp (Integer m, const double x[], const double f[],
                          Integer nw, Integer nq, Integer iq[], double rq[], NagError *fail)
```

3 Description

nag_5d_shep_interp (e01tmc) constructs a smooth function \( Q(x) \), \( x \in \mathbb{R}^5 \) which interpolates a set of \( m \) scattered data points \( (x_r, f_r) \), for \( r = 1, 2, \ldots, m \), using a modification of Shepard’s method. The surface is continuous and has continuous first partial derivatives.

The basic Shepard method, which is a generalization of the two-dimensional method described in Shepard (1968), interpolates the input data with the weighted mean

\[
Q(x) = \frac{\sum_{r=1}^{m} w_r(x)q_r}{\sum_{r=1}^{m} w_r(x)},
\]

where \( q_r = f_r \), \( w_r(x) = \frac{1}{d_r^2} \) and \( d_r^2 = \|x - x_r\|^2 \).

The basic method is global in that the interpolated value at any point depends on all the data, but nag_5d_shep_interp (e01tmc) uses a modification (see Franke and Nielson (1980) and Renka (1988a)), whereby the method becomes local by adjusting each \( w_r(x) \) to be zero outside a hypersphere with centre \( x_r \) and some radius \( R_w \). Also, to improve the performance of the basic method, each \( q_r \) above is replaced by a function \( q_r(x) \), which is a quadratic fitted by weighted least squares to data local to \( x_r \) and forced to interpolate \( (x_r, f_r) \). In this context, a point \( x \) is defined to be local to another point if it lies within some distance \( R_q \) of it.

The efficiency of nag_5d_shep_interp (e01tmc) is enhanced by using a cell method for nearest neighbour searching due to Bentley and Friedman (1979) with a cell density of 3.

The radii \( R_w \) and \( R_q \) are chosen to be just large enough to include \( N_w \) and \( N_q \) data points, respectively, for user-supplied constants \( N_w \) and \( N_q \). Default values of these arguments are provided, and advice on alternatives is given in Section 9.2.

nag_5d_shep_interp (e01tmc) is derived from the new implementation of QSHEP3 described by Renka (1988b). It uses the modification for five-dimensional interpolation described by Berry and Minser (1999).

Values of the interpolant \( Q(x) \) generated by nag_5d_shep_interp (e01tmc), and its first partial derivatives, can subsequently be evaluated for points in the domain of the data by a call to nag_5d_shep_eval (e01tnc).
4 References


5 Arguments

1:  \( m \) – Integer
    
    On entry: \( m \), the number of data points.
    
    Note: on the basis of experimental results reported in Berry and Minser (1999), it is recommended to use \( m \geq 4000 \).
    
    Constraint: \( m \geq 23 \).

2:  \( x[5 \times m] \) – const double
    
    Note: the \((i, j)\)th element of the matrix \( X \) is stored in \( x[(j-1) \times 5 + i - 1] \).
    
    On entry: \( x[(r-1) \times 5], \ldots, x[(r-1) \times 5 + 4] \) must be set to the Cartesian coordinates of the data point \( x_r \), for \( r = 1, 2, \ldots, m \).
    
    Constraint: these coordinates must be distinct, and must not all lie on the same four-dimensional hypersurface.

3:  \( f[m] \) – const double
    
    On entry: \( f[r-1] \) must be set to the data value \( f_r \), for \( r = 1, 2, \ldots, m \).

4:  \( nw \) – Integer
    
    On entry: the number \( N_w \) of data points that determines each radius of influence \( R_w \), appearing in the definition of each of the weights \( w_r \), for \( r = 1, 2, \ldots, m \) (see Section 3). Note that \( R_w \) is different for each weight. If \( nw \leq 0 \) the default value \( nw = \min(32, m - 1) \) is used instead.
    
    Constraint: \( nw \leq \min(50, m - 1) \).

5:  \( nq \) – Integer
    
    On entry: the number \( N_q \) of data points to be used in the least squares fit for coefficients defining the quadratic functions \( q_r(x) \) (see Section 3). If \( nq \leq 0 \) the default value \( nq = \min(50, m - 1) \) is used instead.
    
    Constraint: \( nq \leq 0 \) or \( 20 \leq nq \leq \min(70, m - 1) \).

6:  \( iq[2 \times m + 1] \) – Integer
    
    On exit: integer data defining the interpolant \( Q(x) \).
7: \( rq[21 \times m + 11] \) – double

\textit{On exit:} real data defining the interpolant \( Q(x) \).

8: \textit{fail} – NagError *

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 \( \langle \text{value} \rangle \) had an illegal value.

**NE_DATA_HYPERSURFACE**

On entry, all the data points lie on the same four-dimensional hypersurface. No unique solution exists.

**NE_DUPLICATE_NODE**

There are duplicate nodes in the dataset. \( x[(k - 1) \times 5 + i - 1] = x[(k - 1) \times 5 + j - 1] \), for \( i = \langle \text{value} \rangle, j = \langle \text{value} \rangle \) and \( k = 1, 2, \ldots, 5 \). The interpolant cannot be derived.

**NE_INT**

\begin{itemize}
  \item On entry, \( m = \langle \text{value} \rangle \).
  \item Constraint: \( m \geq 23 \).
\end{itemize}

\begin{itemize}
  \item On entry, \( nq = \langle \text{value} \rangle \).
  \item Constraint: \( nq \leq 0 \) or \( nq \geq 20 \).
\end{itemize}

**NE_INT_2**

\begin{itemize}
  \item On entry, \( nq = \langle \text{value} \rangle \) and \( m = \langle \text{value} \rangle \).
  \item Constraint: \( nq \leq \min(70, m - 1) \).
\end{itemize}

\begin{itemize}
  \item On entry, \( nw = \langle \text{value} \rangle \) and \( m = \langle \text{value} \rangle \).
  \item Constraint: \( nw \leq \min(50, m - 1) \).
\end{itemize}

**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.

7 Accuracy

On successful exit, the function generated interpolates the input data exactly and has quadratic precision. Overall accuracy of the interpolant is affected by the choice of arguments \( nw \) and \( nq \) as well as the smoothness of the function represented by the input data. Berry and Minser (1999) report on the results obtained for a set of test functions.
8 Parallelism and Performance

nag_5d_shep_interp (e01tmc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag_5d_shep_interp (e01tmc) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users’ Note for your implementation for any additional implementation-specific information.

9 Further Comments

9.1 Timing

The time taken for a call to nag_5d_shep_interp (e01tmc) will depend in general on the distribution of the data points and on the choice of \( N_w \) and \( N_q \) parameters. If the data points are uniformly randomly distributed, then the time taken should be \( O(m) \). At worst \( O(m^2) \) time will be required.

9.2 Choice of \( N_w \) and \( N_q \)

Default values of the arguments \( N_w \) and \( N_q \) may be selected by calling nag_5d_shep_interp (e01tmc) with \( n_w \leq 0 \) and \( n_q \leq 0 \). These default values may well be satisfactory for many applications.

If non-default values are required they must be supplied to nag_5d_shep_interp (e01tmc) through positive values of \( n_w \) and \( n_q \). Increasing these argument values makes the method less local. This may increase the accuracy of the resulting interpolant at the expense of increased computational cost. The default values \( n_w = \min(32, m - 1) \) and \( n_q = \min(50, m - 1) \) have been chosen on the basis of experimental results reported in Berry and Minser (1999). In these experiments the error norm was found to increase with the decrease of \( N_q \), but to be little affected by the choice of \( N_w \). The choice of both, directly affected the time taken by the function. For further advice on the choice of these arguments see Berry and Minser (1999).

10 Example

This program reads in a set of 30 data points and calls nag_5d_shep_interp (e01tmc) to construct an interpolating function \( Q(x) \). It then calls nag_5d_shep_eval (e01tnc) to evaluate the interpolant at a set of points.

Note that this example is not typical of a realistic problem: the number of data points would normally be larger.

See also Section 10 in nag_5d_shep_eval (e01tnc).

10.1 Program Text

/* nag_5d_shep_interp (e01tmc) Example Program. */
/* Copyright 2014 Numerical Algorithms Group. */
/* Mark 23, 2010. */

#include <stdio.h>
#include <nag.h>
#include <nage01.h>

#define X(I, J) x[I *5 + J]
#define XE(I, J) xe[I *5 + J]

int main(void)
{ /* Scalars */
    Integer exit_status, i, j, m, nq, nw, liq, lrq;
    NagError fail;

    /* Arrays */
    double *f = 0, *q = 0, *qx = 0, *rq = 0, *xe = 0, *x = 0;
    Integer *iq = 0;
    exit_status = 0;
    INIT_FAIL(fail);

    printf("nag_5d_shep_interp (e01tmc) Example Program Results\\n");

    /* Skip heading in data file */
    #ifdef _WIN32
        scanf_s("%*[\n]");
    #else
        scanf("%*[\n]");
    #endif

    /* Input the number of nodes. */
    #ifdef _WIN32
        scanf("%"NAG_IFMT"%*[\n] ", &m);
    #else
        scanf("%"NAG_IFMT"%*[\n] ", &m);
    #endif

    /* Allocate memory */
    lrq = 21 * m + 11;
    liq = 2 * m + 1;
    if (!(f = NAG_ALLOC(m, double))
    || !(x = NAG_ALLOC(m*5, double))
    || !(rq = NAG_ALLOC(lrq, double))
    || !(iq = NAG_ALLOC(liq, Integer)))
    {
        printf("Allocation failure\\n");
        exit_status = -1;
        goto END;
    }

    /* Input the data points X and F. */
    for (i = 0; i < m; ++i) {
        for (j = 0; j < 5; ++j) {
            #ifdef _WIN32
                scanf_s("%lf", &X(i, j));
            #else
                scanf("%lf", &X(i, j));
            #endif
            #ifdef _WIN32
                scanf_s("%lf%*[\n]", &f[i]);
            #else
                scanf("%lf%*[\n]", &f[i]);
            #endif
        }
    }

    /* Generate the interpolant. */
    nq = 0;
    nw = 0;
    nag_5d_shep_interp (e01tmc).
    * Interpolating functions, modified Shepard’s method, five
    * variables
    *\n
    if (fail.code != NE_NOERROR) {
        printf("Error from nag_5d_shep_interp (e01tmc).
        \\
        exit_status = 1;
        goto END;
    }
/* Input the number of evaluation points. */
#ifdef _WIN32
    scanf_s("%NAG_IFMT%*[\n ] ", &n);
#else
    scanf("%NAG_IFMT%*[\n ] ", &n);
#endif
/* Allocate memory for nag_5d_shep_eval (e01tnc) */
if (!(q = NAG_ALLOC(n, double)) ||
    !(qx = NAG_ALLOC(n*5, double)) ||
    !(xe = NAG_ALLOC(n*5, double)))
{
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
}
/* Input the evaluation points. */
for (i = 0; i < n; ++i) {
    for (j = 0; j < 5; ++j) {
#ifdef _WIN32
        scanf_s("%lf", &XE(i, j));
#else
        scanf("%lf", &XE(i, j));
#endif
    }
#ifdef _WIN32
        scanf_s("%*[\n ] ");
#else
        scanf("%*[\n ] ");
#endif
    }
/* nag_5d_shep_eval (e01tnc). */
* Evaluate interpolant and first derivatives computed by
* nag_5d_shep_interp (e01tmc).
*/
fail.print = Nag_TRUE;
nag_5d_shep_eval(m, x, f, iq, rq, n, xe, q, qx, &fail);
if (fail.code != NE_NOERROR) {
    printf("Error from nag_5d_shep_eval (e01tnc).
%s
", fail.message);
    exit_status = 1;
    goto END;
}
printf("\n Evaluation of interpolant at various (5D) points\n");
printf("\n%6s%30s%17s\n"," pt.no.","point coordinates","value");
for (i = 0; i < n; ++i)
    printf("%5"NAG_IFMT%8.2f%8.2f%8.2f%8.2f%10.4f\n", i, XE(i,0),
        XE(i,1), XE(i,2), XE(i,3), XE(i,4), q[i]);
END:
NAG_FREE(f);
NAG_FREE(q);
NAG_FREE(qx);
NAG_FREE(rq);
NAG_FREE(xe);
NAG_FREE(iq);
return exit_status;
}
10.2 Program Data

nag_5d_shep_interp (e01tmc) Example Program Data
30 : number of data points
0.81 0.15 0.44 0.83 0.21 6.39 : x and f(x)
0.91 0.96 0.00 0.09 0.98 2.50
0.13 0.88 0.22 0.21 0.73 9.34
0.91 0.49 0.39 0.79 0.47 7.52
0.63 0.41 0.72 0.68 0.65 6.91
0.10 0.13 0.77 0.47 0.22 4.68
0.28 0.93 0.24 0.90 0.96 45.40
0.55 0.01 0.04 0.41 0.26 5.48
0.96 0.19 0.95 0.66 0.99 2.75
0.96 0.32 0.53 0.96 0.84 7.43
0.16 0.05 0.16 0.30 0.58 6.05
0.97 0.14 0.36 0.72 0.78 5.77
0.96 0.73 0.28 0.75 0.28 8.68
0.49 0.48 0.58 0.19 0.25 2.38
0.80 0.34 0.64 0.57 0.08 3.70
0.14 0.24 0.12 0.06 0.63 1.34
0.42 0.45 0.03 0.68 0.66 15.18
0.92 0.19 0.48 0.67 0.28 4.35
0.79 0.32 0.15 0.13 0.40 1.50
0.96 0.26 0.93 0.89 0.61 3.43
0.66 0.83 0.41 0.17 0.09 3.10
0.04 0.70 0.40 0.54 0.37 14.33
0.85 0.33 0.15 0.03 0.26 0.25
0.93 0.58 0.88 0.81 0.40 4.30
0.68 0.29 0.88 0.60 0.47 3.77
0.76 0.26 0.09 0.41 0.14 4.16
0.74 0.26 0.33 0.64 0.36 6.75
0.39 0.68 0.69 0.37 0.12 5.22
0.66 0.52 0.17 1.00 0.43 16.23
0.17 0.08 0.35 0.71 0.17 10.62 : End of data points
6 : number of evaluation points
0.10 0.10 0.10 0.10 0.10 : evaluation point ordinates
0.20 0.20 0.20 0.20 0.20
0.30 0.30 0.30 0.30 0.30
0.40 0.40 0.40 0.40 0.40
0.50 0.50 0.50 0.50 0.50
0.60 0.60 0.60 0.60 0.60 : End of evaluation points

10.3 Program Results

nag_5d_shep_interp (e01tmc) Example Program Results

Evaluation of interpolant at various (5D) points

<table>
<thead>
<tr>
<th>pt.no.</th>
<th>point coordinates</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.10 0.10 0.10</td>
<td>3.2313</td>
</tr>
<tr>
<td>1</td>
<td>0.20 0.20 0.20</td>
<td>4.2476</td>
</tr>
<tr>
<td>2</td>
<td>0.30 0.30 0.30</td>
<td>5.2695</td>
</tr>
<tr>
<td>3</td>
<td>0.40 0.40 0.40</td>
<td>6.3838</td>
</tr>
<tr>
<td>4</td>
<td>0.50 0.50 0.50</td>
<td>7.6837</td>
</tr>
<tr>
<td>5</td>
<td>0.60 0.60 0.60</td>
<td>9.3885</td>
</tr>
</tbody>
</table>