

# NAG Library Routine Document

## E01TGF

**Note:** before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

### 1 Purpose

E01TGF generates a three-dimensional interpolant to a set of scattered data points, using a modified Shepard method.

### 2 Specification

```
SUBROUTINE E01TGF (M, X, Y, Z, F, NW, NQ, IQ, LIQ, RQ, LRQ, IFAIL)
  INTEGER          M, NW, NQ, IQ(LIQ), LIQ, LRQ, IFAIL
  REAL (KIND=nag_wp) X(M), Y(M), Z(M), F(M), RQ(LRQ)
```

### 3 Description

E01TGF constructs a smooth function  $Q(x, y, z)$  which interpolates a set of  $m$  scattered data points  $(x_r, y_r, z_r, f_r)$ , for  $r = 1, 2, \dots, m$ , 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

$$Q(x, y, z) = \frac{\sum_{r=1}^m w_r(x, y, z) q_r}{\sum_{r=1}^m w_r(x, y, z)},$$

where

$$q_r = f_r \text{ and } w_r(x, y, z) = \frac{1}{d_r^2} \text{ and } d_r^2 = (x - x_r)^2 + (y - y_r)^2 + (z - z_r)^2.$$

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

The efficiency of the routine is further enhanced by using a cell method for nearest neighbour searching due to Bentley and Friedman (1979).

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 by the routine, and advice on alternatives is given in Section 9.2.

This routine is derived from the routine QSHEP3 described by Renka (1988b).

Values of the interpolant  $Q(x, y, z)$  generated by this routine, and its first partial derivatives, can subsequently be evaluated for points in the domain of the data by a call to E01THF.

## 4 References

- Bentley J L and Friedman J H (1979) Data structures for range searching *ACM Comput. Surv.* **11** 397–409
- Franke R and Nielson G (1980) Smooth interpolation of large sets of scattered data *Internat. J. Num. Methods Engrg.* **15** 1691–1704
- Renka R J (1988a) Multivariate interpolation of large sets of scattered data *ACM Trans. Math. Software* **14** 139–148
- Renka R J (1988b) Algorithm 661: QSHEP3D: Quadratic Shepard method for trivariate interpolation of scattered data *ACM Trans. Math. Software* **14** 151–152
- Shepard D (1968) A two-dimensional interpolation function for irregularly spaced data *Proc. 23rd Nat. Conf. ACM* 517–523 Brandon/Systems Press Inc., Princeton

## 5 Arguments

- 1: M – INTEGER *Input*  
*On entry:*  $m$ , the number of data points.  
*Constraint:*  $M \geq 10$ .
- 2: X(M) – REAL (KIND=nag\_wp) array *Input*
- 3: Y(M) – REAL (KIND=nag\_wp) array *Input*
- 4: Z(M) – REAL (KIND=nag\_wp) array *Input*  
*On entry:*  $X(r)$ ,  $Y(r)$ ,  $Z(r)$  must be set to the Cartesian coordinates of the data point  $(x_r, y_r, z_r)$ , for  $r = 1, 2, \dots, m$ .  
*Constraint:* these coordinates must be distinct, and must not all be coplanar.
- 5: F(M) – REAL (KIND=nag\_wp) array *Input*  
*On entry:*  $F(r)$  must be set to the data value  $f_r$ , for  $r = 1, 2, \dots, m$ .
- 6: NW – INTEGER *Input*  
*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  $r = 1, 2, \dots, m$  (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:*  $NW \leq \min(40, M - 1)$ .
- 7: NQ – INTEGER *Input*  
*On entry:* the number  $N_q$  of data points to be used in the least squares fit for coefficients defining the nodal functions  $q_r(x, y, z)$  (see Section 3). If  $NQ \leq 0$  the default value  $NQ = \min(17, M - 1)$  is used instead.  
*Constraint:*  $NQ \leq 0$  or  $9 \leq NQ \leq \min(40, M - 1)$ .
- 8: IQ(LIQ) – INTEGER array *Output*  
*On exit:* integer data defining the interpolant  $Q(x, y, z)$ .
- 9: LIQ – INTEGER *Input*  
*On entry:* the dimension of the array IQ as declared in the (sub)program from which E01TGF is called.  
*Constraint:*  $LIQ \geq 2 \times M + 1$ .

- 10: RQ(LRQ) – REAL (KIND=nag\_wp) array *Output*  
*On exit:* real data defining the interpolant  $Q(x, y, z)$ .
- 11: LRQ – INTEGER *Input*  
*On entry:* the dimension of the array RQ as declared in the (sub)program from which E01TGF is called.  
*Constraint:*  $LRQ \geq 10 \times M + 7$ .
- 12: IFAIL – INTEGER *Input/Output*  
*On entry:* IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.  
 For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**  
*On exit:* IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

## 6 Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry,  $M < 10$ ,  
 or  $0 < NQ < 9$ ,  
 or  $NQ > \min(40, M - 1)$ ,  
 or  $NW > \min(40, M - 1)$ ,  
 or  $LIQ < 2 \times M + 1$ ,  
 or  $LRQ < 10 \times M + 7$ .

IFAIL = 2

On entry,  $(X(i), Y(i), Z(i)) = (X(j), Y(j), Z(j))$  for some  $i \neq j$ .

IFAIL = 3

On entry, all the data points are coplanar. No unique solution exists.

IFAIL = -99

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -399

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL = -999

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

## 7 Accuracy

On successful exit, the function generated interpolates the input data exactly and has quadratic accuracy.

## 8 Parallelism and Performance

E01TGF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

E01TGF 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 routine. 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 E01TGF will depend in general on the distribution of the data points. If  $X$ ,  $Y$  and  $Z$  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 E01TGF with  $NW \leq 0$  and  $NQ \leq 0$ . These default values may well be satisfactory for many applications.

If non-default values are required they must be supplied to E01TGF through positive values of  $NW$  and  $NQ$ . Increasing these arguments makes the method less local. This may increase the accuracy of the resulting interpolant at the expense of increased computational cost. The default values  $NW = \min(32, M - 1)$  and  $NQ = \min(17, M - 1)$  have been chosen on the basis of experimental results reported in Renka (1988a). In these experiments the error norm was found to vary smoothly with  $N_w$  and  $N_q$ , generally increasing monotonically and slowly with distance from the optimal pair. The method is not therefore thought to be particularly sensitive to the argument values. For further advice on the choice of these arguments see Renka (1988a).

## 10 Example

This program reads in a set of 30 data points and calls E01TGF to construct an interpolating function  $Q(x, y, z)$ . It then calls E01THF 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.

### 10.1 Program Text

```

Program e01tgfe

!      E01TGF Example Program Text

!      Mark 26 Release. NAG Copyright 2016.

!      .. Use Statements ..
      Use nag_library, Only: e01tgf, e01thf, nag_wp
!      .. Implicit None Statement ..
      Implicit None
!      .. Parameters ..
      Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..

```

```

Integer                                :: i, ifail, liq, lrq, m, n, nq, nw
! .. Local Arrays ..
Real (Kind=nag_wp), Allocatable        :: f(:), q(:), qx(:), qy(:), qz(:), &
                                         rq(:), u(:), v(:), w(:), x(:), y(:), &
                                         z(:)
Integer, Allocatable                   :: iq(:)
! .. Executable Statements ..
Write (nout,*) 'E01TGF Example Program Results'

! Skip heading in data file
Read (nin,*)

! Input the number of nodes.

Read (nin,*) m
liq = 2*m + 1
lrq = 10*m + 7
Allocate (x(m),y(m),z(m),f(m),iq(liq),rq(lrq))

Do i = 1, m
  Read (nin,*) x(i), y(i), z(i), f(i)
End Do

! Generate the interpolant.

nq = 0
nw = 0

ifail = 0
Call e01tgf(m,x,y,z,f,nw,nq,iq,liq,rq,lrq,ifail)

! Input the number of evaluation points.

Read (nin,*) n
Allocate (u(n),v(n),w(n),q(n),qx(n),qy(n),qz(n))

! Input the evaluation points.

Do i = 1, n
  Read (nin,*) u(i), v(i), w(i)
End Do

! Evaluate the interpolant using E01THF.

ifail = -0
Call e01thf(m,x,y,z,f,iq,liq,rq,lrq,n,u,v,w,q,qx,qy,qz,ifail)

Write (nout,*)
Write (nout,*) '      I      U(I)      V(I)      W(I)      Q(I)'

Do i = 1, n
  Write (nout,99999) i, u(i), v(i), w(i), q(i)
End Do

99999 Format (1X,I6,4F10.4)
End Program e01tgfe

```

## 10.2 Program Data

E01TGF Example Program Data

```

30      M, the number of data points
0.80 0.23 0.37 0.51 X, Y, Z, F data point definition
0.23 0.88 0.05 1.80
0.18 0.43 0.04 0.11
0.58 0.95 0.62 2.65
0.64 0.69 0.20 0.93
0.88 0.35 0.49 0.72
0.30 0.10 0.78 -0.11
0.87 0.09 0.05 0.67
0.04 0.02 0.40 0.00

```

```

0.62  0.90  0.43  2.20
0.87  0.96  0.24  3.17
0.62  0.64  0.45  0.74
0.86  0.13  0.47  0.64
0.87  0.60  0.46  1.07
0.49  0.43  0.13  0.22
0.12  0.61  0.00  0.41
0.02  0.71  0.82  0.58
0.62  0.93  0.44  2.48
0.49  0.54  0.04  0.37
0.36  0.56  0.39  0.35
0.62  0.42  0.97 -0.20
0.01  0.72  0.45  0.78
0.41  0.36  0.52  0.11
0.17  0.99  0.65  2.82
0.51  0.29  0.59  0.14
0.85  0.05  0.04  0.61
0.20  0.20  0.87 -0.25
0.04  0.67  0.04  0.59
0.31  0.63  0.18  0.50
0.88  0.27  0.07  0.71  End of data points
6      N, the number of evaluation points
0.10  0.10  0.10      U, V, W evaluation point definition
0.20  0.20  0.20
0.30  0.30  0.30
0.40  0.40  0.40
0.50  0.50  0.50
0.60  0.60  0.60      End of evaluation points

```

### 10.3 Program Results

E01TGF Example Program Results

I	U(I)	V(I)	W(I)	Q(I)
1	0.1000	0.1000	0.1000	0.2663
2	0.2000	0.2000	0.2000	0.1040
3	0.3000	0.3000	0.3000	0.0695
4	0.4000	0.4000	0.4000	0.1539
5	0.5000	0.5000	0.5000	0.3010
6	0.6000	0.6000	0.6000	0.5695

---