NAG Library Routine Document

G05ZSF

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

G05ZSF produces realizations of a stationary Gaussian random field in two dimensions, using the circulant embedding method. The square roots of the eigenvalues of the extended covariance matrix (or embedding matrix) need to be input, and can be calculated using G05ZQF or G05ZRF.

2 Specification

```
SUBROUTINE G05ZSF (NS, S, M, LAM, RHO, STATE, Z, IFAIL)
INTEGER             NS(2), S, M(2), STATE(*), IFAIL
REAL (KIND=nag_wp) LAM(M(1)*M(2)), RHO, Z(NS(1)*NS(2),S)
```

3 Description

A two-dimensional random field $Z(\mathbf{x})$ in $\mathbb{R}^2$ is a function which is random at every point $\mathbf{x} \in \mathbb{R}^2$, so $Z(\mathbf{x})$ is a random variable for each $\mathbf{x}$. The random field has a mean function $\mu(\mathbf{x}) = \mathbb{E}[Z(\mathbf{x})]$ and a symmetric positive semidefinite covariance function $C(\mathbf{x}, \mathbf{y}) = \mathbb{E}[(Z(\mathbf{x}) - \mu(\mathbf{x}))(Z(\mathbf{y}) - \mu(\mathbf{y}))]$. $Z(\mathbf{x})$ is a Gaussian random field if for any choice of $n \in \mathbb{N}$ and $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^2$, the random vector $[Z(\mathbf{x}_1), \dots, Z(\mathbf{x}_n)]^{\mathrm{T}}$ follows a multivariate Normal distribution, which would have a mean vector $\tilde{\boldsymbol{\mu}}$ with entries $\tilde{\mu}_i = \mu(\mathbf{x}_i)$ and a covariance matrix $\tilde{C}$ with entries $\tilde{C}_{ij} = C(\mathbf{x}_i, \mathbf{x}_j)$. A Gaussian random field $Z(\mathbf{x})$ is stationary if $\mu(\mathbf{x})$ is constant for all $\mathbf{x} \in \mathbb{R}^2$ and $C(\mathbf{x}, \mathbf{y}) = C(\mathbf{x} + \mathbf{a}, \mathbf{y} + \mathbf{a})$ for all $\mathbf{x}, \mathbf{y}, \mathbf{a} \in \mathbb{R}^2$ and hence we can express the covariance function $C(\mathbf{x}, \mathbf{y})$ as a function γ of one variable: $C(\mathbf{x}, \mathbf{y}) = \gamma(\mathbf{x} - \mathbf{y})$. γ is known as a variogram (or more correctly, a semivariogram) and includes the multiplicative factor σ^2 representing the variance such that $\gamma(0) = \sigma^2$.

The routines G05ZQF or G05ZRF along with G05ZSF are used to simulate a two-dimensional stationary Gaussian random field, with mean function zero and variogram $\gamma(\mathbf{x})$, over a domain $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$, using an equally spaced set of $N_1 \times N_2$ points; N_1 points in the x-direction and N_2 points in the y-direction. The problem reduces to sampling a Gaussian random vector $\mathbf{X}$ of size $N_1 \times N_2$, with mean vector zero and a symmetric covariance matrix A, which is an N_2 by N_2 block Toeplitz matrix with Toeplitz blocks of size N_1 by N_1. Since A is in general expensive to factorize, a technique known as the *circulant embedding method* is used. A is embedded into a larger, symmetric matrix B, which is an M_2 by M_2 block circulant matrix with circulant bocks of size M_1 by M_1, where $M_1 \geq 2(N_1 - 1)$ and $M_2 \geq 2(N_2 - 1)$. B can now be factorized as $B = W \Lambda W^* = R^* R$, where W is the two-dimensional Fourier matrix (W^* is the complex conjugate of W), Λ is the diagonal matrix containing the eigenvalues of B and $R = \Lambda^{\frac{1}{2}} W^*$. B is known as the embedding matrix. The eigenvalues can be calculated by performing a discrete Fourier transform of the first row (or column) of B and multiplying by $M_1 \times M_2$, and so only the first row (or column) of B is needed – the whole matrix does not need to be formed.

The symmetry of A as a block matrix, and the symmetry of each block of A, depends on whether the covariance function γ is even or not. γ is even if $\gamma(\mathbf{x}) = \gamma(-\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^2$, and uneven otherwise (in higher dimensions, γ can be even in some coordinates and uneven in others, but in two dimensions γ is either even in both coordinates or uneven in both coordinates). If γ is even then A is a symmetric block matrix and has symmetric blocks; if γ is uneven then A is not a symmetric block matrix and has non-symmetric blocks. In the uneven case, M_1 and M_2 are set to be odd in order to guarantee symmetry in B.

As long as all of the values of Λ are non-negative (i.e., B is positive semidefinite), B is a covariance matrix for a random vector $\mathbf{Y}$ which has M_2 'blocks' of size M_1. Two samples of $\mathbf{Y}$ can now be simulated from the real and imaginary parts of $R^*(\mathbf{U}+i\mathbf{V})$, where $\mathbf{U}$ and $\mathbf{V}$ have elements from the standard Normal distribution. Since $R^*(\mathbf{U}+i\mathbf{V}) = W\Lambda^{\frac{1}{2}}(\mathbf{U}+i\mathbf{V})$, this calculation can be done using a discrete Fourier transform of the vector $\Lambda^{\frac{1}{2}}(\mathbf{U}+i\mathbf{V})$. Two samples of the random vector $\mathbf{X}$ can now be recovered by taking the first N_1 elements of the first N_2 blocks of each sample of Y – because the original covariance matrix A is embedded in B, $\mathbf{X}$ will have the correct distribution.

If B is not positive semidefinite, larger embedding matrices B can be tried; however if the size of the matrix would have to be larger than MAXM, an approximation procedure is used. See the documentation of G05ZQF or G05ZRF for details of the approximation procedure.

G05ZSF takes the square roots of the eigenvalues of the embedding matrix B, and its size vector M, as input and outputs S realizations of the random field in Z.

One of the initialization routines G05KFF (for a repeatable sequence if computed sequentially) or G05KGF (for a non-repeatable sequence) must be called prior to the first call to G05ZSF.

4 References

Dietrich C R and Newsam G N (1997) Fast and exact simulation of stationary Gaussian processes through circulant embedding of the covariance matrix *SIAM J. Sci. Comput.* **18** 1088–1107

Schlather M (1999) Introduction to positive definite functions and to unconditional simulation of random fields *Technical Report ST 99–10* Lancaster University

Wood A T A and Chan G (1994) Simulation of stationary Gaussian processes in $[0,1]^d$ *Journal of Computational and Graphical Statistics* **3(4)** 409–432

5 Parameters

1: NS(2) – INTEGER array *Input*

On entry: the number of sample points to use in each direction, with NS(1) sample points in the x-direction and NS(2) sample points in the y-direction. The total number of sample points on the grid is therefore NS(1) × NS(2). This must be the same value as supplied to G05ZQF or G05ZRF when calculating the eigenvalues of the embedding matrix.

Constraints:

$\text{NS}(1) \geq 1$;
$\text{NS}(2) \geq 1$.

2: S – INTEGER *Input*

On entry: S, the number of realizations of the random field to simulate.

Constraint: $\text{S} \geq 1$.

3: M(2) – INTEGER array *Input*

On entry: indicates the size, M, of the embedding matrix as returned by G05ZQF or G05ZRF. The embedding matrix is a block circulant matrix with circulant blocks. M(1) is the size of each block, and M(2) is the number of blocks.

Constraints:

$\text{M}(1) \geq \max(1, 2(\text{NS}(1) - 1))$;
$\text{M}(2) \geq \max(1, 2(\text{NS}(2) - 1))$.

4: LAM(M(1) × M(2)) – REAL (KIND=nag_wp) array *Input*

On entry: contains the square roots of the eigenvalues of the embedding matrix, as returned by G05ZQF or G05ZRF.

Constraint: $\text{LAM}(i) \geq 0$, $i = 1, 2, \ldots, \text{M}(1) \times \text{M}(2)$.

5: RHO – REAL (KIND=nag_wp) *Input*

On entry: indicates the scaling of the covariance matrix, as returned by G05ZQF or G05ZRF.

Constraint: $0.0 < \text{RHO} \leq 1.0$.

6: STATE($*$) – INTEGER array *Communication Array*

Note: the actual argument supplied **must** be the array STATE supplied to the initialization routines G05KFF or G05KGF.

On entry: contains information on the selected base generator and its current state.

On exit: contains updated information on the state of the generator.

7: Z($\text{NS}(1) \times \text{NS}(2), \text{S}$) – REAL (KIND=nag_wp) array *Output*

On exit: contains the realizations of the random field. The kth realization (where $k = 1, 2, \ldots, \text{S}$) of the random field on the two-dimensional grid (x_i, y_j) is stored in $\text{Z}((j-1) \times \text{NS}(1) + i, k)$, for $i = 1, 2, \ldots, \text{NS}(1)$ and for $j = 1, 2, \ldots, \text{NS}(2)$. The points are returned in XX and YY by G05ZQF or G05ZRF .

8: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction 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 parameter, 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: $\text{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

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

Errors or warnings detected by the routine:

$\text{IFAIL} = 1$

On entry, $\text{NS} = [\langle\textit{value}\rangle, \langle\textit{value}\rangle]$.
Constraint: $\text{NS}(1) \geq 1$, $\text{NS}(2) \geq 1$.

$\text{IFAIL} = 2$

On entry, $\text{S} = \langle\textit{value}\rangle$.
Constraint: $\text{S} \geq 1$.

$\text{IFAIL} = 3$

On entry, $\text{M} = [\langle\textit{value}\rangle, \langle\textit{value}\rangle]$, and $\text{NS} = [\langle\textit{value}\rangle, \langle\textit{value}\rangle]$.
Constraints: $\text{M}(i) \geq \max(1, 2(\text{NS}(i)) - 1)$, for $i = 1, 2$.

$\text{IFAIL} = 4$

On entry, at least one element of LAM was negative.
Constraint: all elements of LAM must be non-negative.

IFAIL $= 5$

On entry, RHO $= \langle value \rangle$.
Constraint: $0.0 <$ RHO ≤ 1.0.

IFAIL $= 6$

On entry, STATE vector has been corrupted or not initialized.

IFAIL $= -99$

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

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

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

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

G05ZSF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

G05ZSF 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

Because samples are generated in pairs, calling this routine k times, with S $= s$, say, will generate a different sequence of numbers than calling the routine once with S $= ks$, unless s is even.

10 Example

This example calls G05ZSF to generate 5 realizations of a two-dimensional random field on a 5 by 5 grid. This uses eigenvalues of the embedding covariance matrix for a symmetric stable variogram as calculated by G05ZRF with ICOV2 $= 1$.

10.1 Program Text

```
!   G05ZSF Example Program Text

!   Mark 25 Release. NAG Copyright 2014.

    Program g05zsfe

!     G05ZSF Example Main Program
!     .. Use Statements ..
      Use nag_library, Only: g05zrf, g05zsf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                 :: lenst = 17, nin = 5, nout = 6,   &
                                            npmax = 4
!     .. Local Scalars ..
      Real (Kind=nag_wp)                 :: rho, var, xmax, xmin, ymax, ymin
      Integer                            :: approx, icorr, icount, icov2,    &
                                            ifail, norm, np, pad, s
!     .. Local Arrays ..
      Real (Kind=nag_wp)                 :: eig(3), params(npmax)
      Real (Kind=nag_wp), Allocatable    :: lam(:), xx(:), yy(:), z(:,:)
      Integer                            :: m(2), maxm(2), ns(2), state(lenst)
!     .. Executable Statements ..
      Write (nout,*) 'G05ZSF Example Program Results'
      Write (nout,*)
      Flush (nout)

!     Get problem specifications from data file
      Call read_input_data(icov2,np,params,norm,var,xmin,xmax,ymin,ymax,ns, &
        maxm,icorr,pad,s)

      Allocate (lam(maxm(1)*maxm(2)),xx(ns(1)),yy(ns(2)))

!     Get square roots of the eigenvalues of the embedding matrix
      ifail = 0
      Call g05zrf(ns,xmin,xmax,ymin,ymax,maxm,var,icov2,norm,np,params,pad, &
        icorr,lam,xx,yy,m,approx,rho,icount,eig,ifail)

      Call display_embedding_results(approx,m,rho,eig,icount)

!     Initialize state array
      Call initialize_state(state)

      Allocate (z(ns(1)*ns(2),s))

!     Compute s random field realisations
      ifail = 0
      Call g05zsf(ns,s,m,lam,rho,state,z,ifail)

      Call display_realizations(ns,s,xx,yy,z)

    Contains
      Subroutine read_input_data(icov2,np,params,norm,var,xmin,xmax,ymin,ymax, &
        ns,maxm,icorr,pad,s)

!       .. Implicit None Statement ..
        Implicit None
!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (Out)     :: var, xmax, xmin, ymax, ymin
        Integer, Intent (Out)                :: icorr, icov2, norm, np, pad, s
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (Out)     :: params(npmax)
        Integer, Intent (Out)                :: maxm(2), ns(2)
!       .. Executable Statements ..
!       Skip heading in data file
        Read (nin,*)

!       Read in covariance function number
        Read (nin,*) icov2

!       Read in number of parameters
        Read (nin,*) np

!       Read in parameters
        If (np>0) Then
          Read (nin,*) params(1:np)
        End If
```

```
!       Read in choice of norm to use
        Read (nin,*) norm

!       Read in variance of random field
        Read (nin,*) var

!       Read in domain endpoints
        Read (nin,*) xmin, xmax
        Read (nin,*) ymin, ymax

!       Read in number of sample points
        Read (nin,*) ns(1:2)

!       Read in maximum size of embedding matrix
        Read (nin,*) maxm(1:2)

!       Read in choice of scaling in case of approximation
        Read (nin,*) icorr

!       Read in choice of padding
        Read (nin,*) pad

!       Read in number of realization samples to be generated
        Read (nin,*) s

        Return

      End Subroutine read_input_data

      Subroutine display_embedding_results(approx,m,rho,eig,icount)

!       .. Implicit None Statement ..
        Implicit None
!       .. Scalar Arguments ..
        Real (Kind=nag_wp), Intent (In)      :: rho
        Integer, Intent (In)                 :: approx, icount
!       .. Array Arguments ..
        Real (Kind=nag_wp), Intent (In)      :: eig(3)
        Integer, Intent (In)                 :: m(2)
!       .. Executable Statements ..
!       Display size of embedding matrix
        Write (nout,*)
        Write (nout,99999) 'Size of embedding matrix = ', m(1)*m(2)

!       Display approximation information if approximation used
        Write (nout,*)
        If (approx==1) Then
          Write (nout,*) 'Approximation required'
          Write (nout,*)
          Write (nout,99998) 'RHO = ', rho
          Write (nout,99997) 'EIG = ', eig(1:3)
          Write (nout,99999) 'ICOUNT = ', icount
        Else
          Write (nout,*) 'Approximation not required'
        End If

        Return

99999   Format (1X,A,I7)
99998   Format (1X,A,F10.5)
99997   Format (1X,A,3(F10.5,1X))

      End Subroutine display_embedding_results

      Subroutine initialize_state(state)

!       .. Use Statements ..
        Use nag_library, Only: g05kff
!       .. Implicit None Statement ..
        Implicit None
```

```
!       .. Parameters ..
        Integer, Parameter                  :: genid = 1, inseed = 14965,     &
                                               lseed = 1, subid = 1
!       .. Array Arguments ..
        Integer, Intent (Out)               :: state(lenst)
!       .. Local Scalars ..
        Integer                             :: ifail, lstate
!       .. Local Arrays ..
        Integer                             :: seed(lseed)
!       .. Executable Statements ..
!       Initialize the generator to a repeatable sequence
        lstate = lenst
        seed(1) = inseed
        ifail = 0
        Call g05kff(genid,subid,seed,lseed,state,lstate,ifail)

      End Subroutine initialize_state

      Subroutine display_realizations(ns,s,xx,yy,z)

!       .. Use Statements ..
        Use nag_library, Only: x04cbf
!       .. Implicit None Statement ..
        Implicit None
!       .. Parameters ..
        Integer, Parameter                  :: indent = 0, ncols = 80
        Character (1), Parameter            :: charlab = 'C', intlab = 'I',   &
                                               matrix = 'G', unit = 'n'
        Character (5), Parameter            :: form = 'F10.5'
!       .. Scalar Arguments ..
        Integer, Intent (In)                :: s
!       .. Array Arguments ..
        Integer, Intent (In)                :: ns(2)
        Real (Kind=nag_wp), Intent (In)     :: xx(ns(1)), yy(ns(2)),          &
                                               z(ns(1)*ns(2),s)
!       .. Local Scalars ..
        Integer                             :: i, ifail, j, nn
        Character (61)                      :: title
!       .. Local Arrays ..
        Character (1)                       :: clabs(0)
        Character (12), Allocatable         :: rlabs(:)
!       .. Executable Statements ..
        nn = ns(1)*ns(2)
        Allocate (rlabs(nn))

!       Set row labels to grid points (column label is realization number).
        Do j = 1, ns(2)
          Do i = 1, ns(1)
            If (i==1) Then
              Write (rlabs((j-1)*ns(1)+i),99999) xx(i), yy(j)
            Else
              Write (rlabs((j-1)*ns(1)+i),99998) xx(i)
            End If
          End Do
        End Do

!       Display random field results
        title = 'Random field realisations (x,y coordinates first):'
        Write (nout,*)
        ifail = 0
        Call x04cbf(matrix,unit,nn,s,z,nn,form,title,charlab,rlabs,intlab, &
          clabs,ncols,indent,ifail)

99999   Format (2F6.1)
99998   Format (F6.1,5X,'.')

      End Subroutine display_realizations


    End Program g05zsfe
```

10.2 Program Data

```
G05ZSF Example Program Data
  1                : icov2  (icov2=1, symmetric stable)
  3                : np     (icov2=1, 3 parameters)
  0.1   0.15  1.2  : params (icov2=1, l1, l2  and nu)
  2                : norm
  0.5              : var
 -1 1              : xmin, xmax
 -0.5   0.5        : ymin, ymax
  5     5          : ns(1:2)
 64    64          : maxm(1:2)
  2                : icorr
  1                : pad
  5                : s
```

10.3 Program Results

```
 G05ZSF Example Program Results


 Size of embedding matrix =      64

 Approximation not required

 Random field realisations (x,y coordinates first):
                     1         2         3         4         5
 -0.8  -0.4   -0.61951  -0.93149  -0.32975  -0.51201   1.38877
 -0.4     .    0.74779   1.33518  -0.51237   0.26595   0.30051
  0.0     .   -0.30579   0.51819   0.50961   0.10379   0.36815
  0.4     .    0.53797  -0.53992  -0.86589  -0.37098   0.21571
  0.8     .   -0.61221  -1.04262   0.00007  -1.22614  -0.06650
 -0.8  -0.2    0.01853   0.64126  -0.42978  -0.79178  -0.55728
 -0.4     .   -0.77912   0.81079  -0.60613   0.07280   1.61511
  0.0     .   -0.23198   1.48744  -0.78145   0.10347   0.07053
  0.4     .    0.32356   0.58676   0.05846   0.34828   1.40522
  0.8     .   -1.24085  -0.92512   0.27247  -0.66965   0.67073
 -0.8   0.0   -1.18183  -0.99775   0.03888   0.01789  -0.65746
 -0.4     .    0.26155  -0.01734  -0.14924   0.28886   0.25940
  0.0     .    1.14960   0.48850  -0.59023   0.22795  -0.60773
  0.4     .   -0.32684  -0.09616  -0.63497  -1.06753  -0.64594
  0.8     .    0.10064   1.06148   0.15020  -0.53168  -0.29251
 -0.8   0.2   -1.30595  -0.03899  -0.35549  -0.20589  -0.35956
 -0.4     .   -0.01776   0.84501   0.20406   0.89039  -0.58338
  0.0     .    0.41898   0.93435  -1.10725   0.76913  -0.74579
  0.4     .   -1.37738   1.72404  -0.20558  -1.41877   1.21816
  0.8     .    0.77866   0.84922  -0.65055   0.83518  -0.26425
 -0.8   0.4   -0.65163   0.50492  -0.52463  -1.12816   1.12817
 -0.4     .    0.15437   0.20739  -0.12675   1.27782  -0.26157
  0.0     .    0.20324   0.54670  -1.73909   0.61580   0.17551
  0.4     .   -1.09470   0.83967   0.70226  -0.34259   0.29368
  0.8     .    1.08452   1.23097  -0.36003   1.06884   0.23594
```