NAG Toolbox

nag_rand_field_2d_user_setup (g05zq)

1 Purpose

nag_rand_field_2d_user_setup (g05zq) performs the setup required in order to simulate stationary Gaussian random fields in two dimensions, for a user-defined variogram, using the *circulant embedding method*. Specifically, the eigenvalues of the extended covariance matrix (or embedding matrix) are calculated, and their square roots output, for use by nag_rand_field_2d_generate (g05zs), which simulates the random field.

2 Syntax

```
[lam, xx, yy, m, approx, rho, icount, eig, user, ifail] =
nag_rand_field_2d_user_setup(ns, xmin, xmax, ymin, ymax, maxm, var, cov2, even,
'pad', pad, 'icorr', icorr, 'user', user)
[lam, xx, yy, m, approx, rho, icount, eig, user, ifail] = g05zq(ns, xmin, xmax,
ymin, ymax, maxm, var, cov2, even, 'pad', pad, 'icorr', icorr, 'user', user)
```

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 $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^2$, the random vector $[Z(\mathbf{x}_1), \ldots, Z(\mathbf{x}_n)]^{\mathrm{T}}$ follows a multivariate Normal distribution, which would have a mean vector $\tilde{\mu}$ 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 functions nag_rand_field_2d_user_setup (g05zq) and nag_rand_field_2d_generate (g05zs) 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 Normal 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 blocks 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 variogram γ is even or not. γ is even in its first coordinate if $\gamma\left([-x_1, x_2]^{\mathrm{T}}\right) = \gamma\left([x_1, x_2]^{\mathrm{T}}\right)$, even in its second coordinate if $\gamma\left([x_1, -x_2]^{\mathrm{T}}\right) = \gamma\left([x_1, x_2]^{\mathrm{T}}\right)$, and even if it is even in both coordinates (in two dimensions it is impossible for γ to be even in one coordinate and uneven in the other). 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 $\mathbf{Y}$ – 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. We write $\Lambda = \Lambda_+ + \Lambda_-$, where Λ_+ and Λ_- contain the non-negative and negative eigenvalues of B respectively. Then B is replaced by ρB_+ where $B_+ = W\Lambda_+ W^*$ and $\rho \in (0,1]$ is a scaling factor. The error ϵ in approximating the distribution of the random field is given by

$$\epsilon = \sqrt{\frac{(1-\rho)^2 \operatorname{trace}\Lambda + \rho^2 \operatorname{trace}\Lambda_-}{M}}.$$

Three choices for ρ are available, and are determined by the input argument **icorr**:

setting **icorr** $= 0$ sets

$$\rho = \frac{\operatorname{trace}\Lambda}{\operatorname{trace}\Lambda_+},$$

setting **icorr** $= 1$ sets

$$\rho = \sqrt{\frac{\operatorname{trace}\Lambda}{\operatorname{trace}\Lambda_+}},$$

setting **icorr** $= 2$ sets $\rho = 1$.

nag_rand_field_2d_user_setup (g05zq) finds a suitable positive semidefinite embedding matrix B and outputs its sizes in the vector **m** and the square roots of its eigenvalues in **lam**. If approximation is used, information regarding the accuracy of the approximation is output. Note that only the first row (or column) of B is actually formed and stored.

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

5.1 Compulsory Input Parameters

1: **ns**(**2**) – INTEGER array

The number of sample points to use in each direction, with **ns**(1) sample points in the x-direction, N_1 and **ns**(2) sample points in the y-direction, N_2. The total number of sample points on the grid is therefore **ns**(1) × **ns**(2).

Constraints:

ns$(1) \geq 1$;
ns$(2) \geq 1$.

2: **xmin** – REAL (KIND=nag_wp)

The lower bound for the x-coordinate, for the region in which the random field is to be simulated.

Constraint: **xmin** $<$ **xmax**.

3: **xmax** – REAL (KIND=nag_wp)

The upper bound for the x-coordinate, for the region in which the random field is to be simulated.

Constraint: **xmin** $<$ **xmax**.

4: **ymin** – REAL (KIND=nag_wp)

The lower bound for the y-coordinate, for the region in which the random field is to be simulated.

Constraint: **ymin** $<$ **ymax**.

5: **ymax** – REAL (KIND=nag_wp)

The upper bound for the y-coordinate, for the region in which the random field is to be simulated.

Constraint: **ymin** $<$ **ymax**.

6: **maxm**(**2**) – INTEGER array

Determines the maximum size of the circulant matrix to use – a maximum of **maxm**(1) elements in the x-direction, and a maximum of **maxm**(2) elements in the y-direction. The maximum size of the circulant matrix is thus **maxm**$(1)\times$**maxm**(2).

Constraints:

if **even** $= 1$, **maxm**$(i) \geq 2^k$, where k is the smallest integer satisfying $2^k \geq 2($**ns**$(i) - 1)$, for $i = 1, 2$;
if **even** $= 0$, **maxm**$(i) \geq 3^k$, where k is the smallest integer satisfying $3^k \geq 2($**ns**$(i) - 1)$, for $i = 1, 2$.

7: **var** – REAL (KIND=nag_wp)

The multiplicative factor σ^2 of the variogram $\gamma(\mathbf{x})$.

Constraint: **var** ≥ 0.0.

8: **cov2** – SUBROUTINE, supplied by the user.

cov2 must evaluate the variogram $\gamma(\mathbf{x})$ for all **x** if **even** $= 0$, and for all **x** with non-negative entries if **even** $= 1$. The value returned in **gamma** is multiplied internally by **var**.

```
[gamma, user] = cov2(x, y, user)
```

Input Parameters

1: **x** – REAL (KIND=nag_wp)

The coordinate x at which the variogram $\gamma(\mathbf{x})$ is to be evaluated.

2: **y** – REAL (KIND=nag_wp)

The coordinate y at which the variogram $\gamma(\mathbf{x})$ is to be evaluated.

3: **user** – INTEGER array

cov2 is called from nag_rand_field_2d_user_setup (g05zq) with the object supplied to nag_rand_field_2d_user_setup (g05zq).

Output Parameters

1: **gamma** – REAL (KIND=nag_wp)

The value of the variogram $\gamma(\mathbf{x})$.

2: **user** – INTEGER array

9: **even** – INTEGER

Indicates whether the covariance function supplied is even or uneven.

even $= 0$
The covariance function is uneven.

even $= 1$
The covariance function is even.

Constraint: **even** $= 0$ or 1.

5.2 Optional Input Parameters

1: **pad** – INTEGER

Default: **pad** $= 1$

Determines whether the embedding matrix is padded with zeros, or padded with values of the variogram. The choice of padding may affect how big the embedding matrix must be in order to be positive semidefinite.

pad $= 0$
The embedding matrix is padded with zeros.

pad $= 1$
The embedding matrix is padded with values of the variogram.

Constraint: **pad** $= 0$ or 1.

2: **icorr** – INTEGER

Default: **icorr** $= 0$

Determines which approximation to implement if required, as described in Section 3.

Constraint: **icorr** $= 0$, 1 or 2.

3: **user** – INTEGER array

user is not used by nag_rand_field_2d_user_setup (g05zq), but is passed to **cov2**. Note that for large objects it may be more efficient to use a global variable which is accessible from the m-files than to use **user**.

5.3 Output Parameters

1: **lam**(**maxm**(**1**) $\times$ **maxm**(**2**)) – REAL (KIND=nag_wp) array

Contains the square roots of the eigenvalues of the embedding matrix.

2: **xx**(**ns**(**1**)) – REAL (KIND=nag_wp) array

The points of the x-coordinates at which values of the random field will be output.

3: **yy**(**ns**(**2**)) – REAL (KIND=nag_wp) array

The points of the y-coordinates at which values of the random field will be output.

4: **m**(**2**) – INTEGER array

m(1) contains M_1, the size of the circulant blocks and **m**(2) contains M_2, the number of blocks, resulting in a final square matrix of size $M_1 \times M_2$.

5: **approx** – INTEGER

Indicates whether approximation was used.

approx $= 0$
No approximation was used.

approx $= 1$
Approximation was used.

6: **rho** – REAL (KIND=nag_wp)

Indicates the scaling of the covariance matrix. **rho** $= 1.0$ unless approximation was used with **icorr** $= 0$ or 1.

7: **icount** – INTEGER

Indicates the number of negative eigenvalues in the embedding matrix which have had to be set to zero.

8: **eig**(**3**) – REAL (KIND=nag_wp) array

Indicates information about the negative eigenvalues in the embedding matrix which have had to be set to zero. **eig**(1) contains the smallest eigenvalue, **eig**(2) contains the sum of the squares of the negative eigenvalues, and **eig**(3) contains the sum of the absolute values of the negative eigenvalues.

9: **user** – INTEGER array

10: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

Constraint: **ns**$(1) \geq 1$, **ns**$(2) \geq 1$.

ifail $= 2$

Constraint: **xmin** $<$ **xmax**.

ifail $= 4$

Constraint: **ymin** $<$ **ymax**.

ifail $= 6$

Constraint: the minima for **maxm** are $[\langle value \rangle, \langle value \rangle]$.
Where, if **even** $= 1$, the minimum calculated value of **maxm**(i) is given by 2^k, where k is the smallest integer satisfying $2^k \geq 2($**ns**$(i) - 1)$, and if **even** $= 0$, the minimum calculated value of **maxm**(i) is given by 3^k, where k is the smallest integer satisfying $3^k \geq 2($**ns**$(i) - 1)$, for $i = 1, 2$.

ifail $= 7$

Constraint: **var** ≥ 0.0.

ifail $= 9$

Constraint: **even** $= 0$ or 1.

ifail $= 10$

Constraint: **pad** $= 0$ or 1.

ifail $= 11$

Constraint: **icorr** $= 0$, 1 or 2.

ifail $= -99$

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

ifail $= -399$

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

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

If on exit **approx** $= 1$, see the comments in Section 3 regarding the quality of approximation; increase the values in **maxm** to attempt to avoid approximation.

8 Further Comments

None.

9 Example

This example calls nag_rand_field_2d_user_setup (g05zq) to calculate the eigenvalues of the embedding matrix for 25 sample points on a 5 by 5 grid of a two-dimensional random field characterized by the symmetric stable variogram:

$$\gamma(\mathbf{x}) = \sigma^2 \exp\left(-(x')^{\nu}\right),$$

where $x' = \left|\frac{x}{\ell_1} + \frac{y}{\ell_2}\right|$, and ℓ_1, ℓ_2 and ν are parameters.

It should be noted that the symmetric stable variogram is one of the pre-defined variograms available in nag_rand_field_2d_predef_setup (g05zr). It is used here purely for illustrative purposes.

9.1 Program Text

```
function g05zq_example

fprintf('g05zq example results\n\n');

% Random Field variance
var = 0.5;
% Domain endpoints
xmin = -1;
xmax =  1;
ymin = -0.5;
ymax =  0.5;
% Number of sample points in x and y
```

```
ns = [nag_int(5), 5];
% Maximum dimensions of circulant matrix
maxm = [nag_int(81), 81];
% Scaling factor rho=1.
icorr = nag_int(2);

% Put covariance parameters (for cov2) in user
norm_p = nag_int(2);
l1 = 0.1;
l2 = 0.15;
nu = 1.2;
user = {norm_p; l1; l2; nu};
% cov2 is even
even = nag_int(1);

% Get square roots of the eigenvalues of the embedding matrix
[lam, xx, yy, m, approx, rho, icount, eig, user, ifail] = ...
  g05zq( ...
         ns, xmin, xmax, ymin, ymax, maxm, var, ...
         @cov2, even, 'icorr', icorr, 'user', user);

fprintf('Size of embedding matrix = %d\n\n', m(1)*m(2));

% Display approximation information if approximation used
if approx == 1
  fprintf('Approximation required\n\n');
  fprintf('rho = %10.5f\n', rho);
  fprintf('eig = %10.5f%10.5f%10.5f\n', eig(1:3));
  fprintf('icount = %d\n', icount);
else
  fprintf('Approximation not required\n\n');
end

% Display square roots of the eigenvalues of the embedding matrix
fprintf('Square roots of eigenvalues of embedding matrix:\n');
mm = m(1)*m(2);
mlam = reshape(lam(1:mm), m(1), m(2));
for i = 1:m(1)
  fprintf('%8.4f',mlam(i,:));
  fprintf('\n');
end

function [gam, user] = cov2(x, y, user)
  norm_p = user{1};
  l1     = user{2};
  l2     = user{3};
  nu     = user{4};

  tl1 = abs(x)/l1;
  tl2 = abs(y)/l2;

  if norm_p == 1
    rnorm = tl1 +  tl2;
  else
    rnorm = sqrt(tl1^2+tl2^2);
  end

  gam = exp(-(rnorm^nu));
```

9.2 Program Results

```
     g05zq example results

Size of embedding matrix = 64

Approximation not required

Square roots of eigenvalues of embedding matrix:
  0.8966  0.8234  0.6810  0.5757  0.5391  0.5757  0.6810  0.8234
  0.8940  0.8217  0.6804  0.5756  0.5391  0.5756  0.6804  0.8217
```

```
  0.8877  0.8175  0.6792  0.5754  0.5391  0.5754  0.6792  0.8175
  0.8813  0.8133  0.6780  0.5751  0.5390  0.5751  0.6780  0.8133
  0.8787  0.8116  0.6774  0.5750  0.5390  0.5750  0.6774  0.8116
  0.8813  0.8133  0.6780  0.5751  0.5390  0.5751  0.6780  0.8133
  0.8877  0.8175  0.6792  0.5754  0.5391  0.5754  0.6792  0.8175
  0.8940  0.8217  0.6804  0.5756  0.5391  0.5756  0.6804  0.8217
```