NAG Toolbox

nag_rand_field_1d_predef_setup (g05zn)

1 Purpose

nag_rand_field_1d_predef_setup (g05zn) performs the setup required in order to simulate stationary Gaussian random fields in one dimension, for a preset 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_1d_generate (g05zp), which simulates the random field.

2 Syntax

```
[lam, xx, m, approx, rho, icount, eig, ifail] = nag_rand_field_1d_predef_setup
(ns, xmin, xmax, var, icov1, params, 'maxm', maxm, 'np', np, 'pad', pad,
'icorr', icorr)
[lam, xx, m, approx, rho, icount, eig, ifail] = g05zn(ns, xmin, xmax, var,
icov1, params, 'maxm', maxm, 'np', np, 'pad', pad, 'icorr', icorr)
```

3 Description

A one-dimensional random field $Z(x)$ in $\mathbb{R}$ is a function which is random at every point $x \in \mathbb{R}$, so $Z(x)$ is a random variable for each x. The random field has a mean function $\mu(x) = \mathbb{E}[Z(x)]$ and a symmetric positive semidefinite covariance function $C(x,y) = \mathbb{E}[(Z(x) - \mu(x))(Z(y) - \mu(y))]$. $Z(x)$ is a Gaussian random field if for any choice of $n \in \mathbb{N}$ and $x_1, \ldots, x_n \in \mathbb{R}$, the random vector $[Z(x_1), \ldots, Z(x_n)]^{\mathrm{T}}$ follows a multivariate Normal distribution, which would have a mean vector $\tilde{\boldsymbol{\mu}}$ with entries $\tilde{\mu}_i = \mu(x_i)$ and a covariance matrix $\tilde{C}$ with entries $\tilde{C}_{ij} = C(x_i, x_j)$. A Gaussian random field $Z(x)$ is stationary if $\mu(x)$ is constant for all $x \in \mathbb{R}$ and $C(x,y) = C(x+a, y+a)$ for all $x, y, a \in \mathbb{R}$ and hence we can express the covariance function $C(x,y)$ as a function γ of one variable: $C(x,y) = \gamma(x-y)$. 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_1d_predef_setup (g05zn) and nag_rand_field_1d_generate (g05zp) are used to simulate a one-dimensional stationary Gaussian random field, with mean function zero and variogram $\gamma(x)$, over an interval $[x_{\min}, x_{\max}]$, using an equally spaced set of N points. The problem reduces to sampling a Normal random vector $\mathbf{X}$ of size N, with mean vector zero and a symmetric Toeplitz covariance matrix A. 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 circulant matrix B of size $M \geq 2(N-1)$, which can now be factorized as $B = W \Lambda W^* = R^* R$, where W is the 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, and so only the first row (or column) of B is needed – the whole matrix does not need to be formed.

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 $\mathbf{Y}$, two samples of which 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 elements 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_1d_predef_setup (g05zn) finds a suitable positive semidefinite embedding matrix B and outputs its size, **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 (1997) Algorithm AS 312: An Algorithm for Simulating Stationary Gaussian Random Fields *Journal of the Royal Statistical Society, Series C (Applied Statistics) (Volume 46)* **1** 171–181

5 Parameters

5.1 Compulsory Input Parameters

1: **ns** – INTEGER

The number of sample points to be generated in realizations of the random field.

Constraint: **ns** ≥ 1.

2: **xmin** – REAL (KIND=nag_wp)

The lower bound for the interval over which the random field is to be simulated. Note that if **icov1** $= 14$ (for simulating fractional Brownian motion), **xmin** is not referenced and the lower bound for the interval is set to zero.

Constraint: if **icov1** $\neq 14$, **xmin** $<$ **xmax**.

3: **xmax** – REAL (KIND=nag_wp)

The upper bound for the interval over which the random field is to be simulated. Note that if **icov1** $= 14$ (for simulating fractional Brownian motion), the lower bound for the interval is set to zero and so **xmax** is required to be greater than zero.

Constraints:

if **icov1** $\neq 14$, **xmin** $<$ **xmax**;
if **icov1** $= 14$, **xmax** > 0.0.

4: **var** – REAL (KIND=nag_wp)

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

Constraint: **var** ≥ 0.0.

5: **icov1** – INTEGER

Determines which of the preset variograms to use. The choices are given below. Note that $x' = \frac{|x|}{\ell}$, where ℓ is the correlation length and is a parameter for most of the variograms, and σ^2 is the variance specified by **var**.

icov1 $= 1$
Symmetric stable variogram

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

where

$\ell =$ **params**(1), $\ell > 0$,

$\nu =$ **params**(2), $0 \leq \nu \leq 2$.

icov1 $= 2$
Cauchy variogram

$$\gamma(x) = \sigma^2 \left(1 + (x')^2\right)^{-\nu},$$

where

$\ell =$ **params**(1), $\ell > 0$,

$\nu =$ **params**(2), $\nu > 0$.

icov1 $= 3$
Differential variogram with compact support

$$\gamma(x) = \begin{cases} \sigma^2\left(1 + 8x' + 25(x')^2 + 32(x')^3\right)(1 - x')^8, & x' < 1, \\ 0, & x' \geq 1, \end{cases}$$

where

$\ell =$ **params**(1), $\ell > 0$.

icov1 $= 4$
Exponential variogram

$$\gamma(x) = \sigma^2 \exp(-x'),$$

where

$\ell =$ **params**(1), $\ell > 0$.

icov1 $= 5$
Gaussian variogram

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

where

$\ell =$ **params**(1), $\ell > 0$.

icov1 $= 6$
Nugget variogram

$$\gamma(x) = \begin{cases} \sigma^2, & x = 0, \\ 0, & x \neq 0. \end{cases}$$

No parameters need be set for this value of **icov1**.

icov1 $= 7$
Spherical variogram

$$\gamma(x) = \begin{cases} \sigma^2\left(1 - 1.5x' + 0.5(x')^3\right), & x' < 1, \\ 0, & x' \geq 1, \end{cases}$$

where

$\ell =$ **params**(1), $\ell > 0$.

icov1 $= 8$
Bessel variogram

$$\gamma(x) = \sigma^2 \frac{2^\nu \Gamma(\nu+1) J_\nu(x')}{(x')^\nu},$$

where

$J_\nu(\cdot)$ is the Bessel function of the first kind,

$\ell =$ **params**(1), $\ell > 0$,

$\nu =$ **params**(2), $\nu \geq -0.5$.

icov1 $= 9$
Hole effect variogram

$$\gamma(x) = \sigma^2 \frac{\sin(x')}{x'},$$

where

$\ell =$ **params**(1), $\ell > 0$.

icov1 $= 10$
Whittle-Matérn variogram

$$\gamma(x) = \sigma^2 \frac{2^{1-\nu}(x')^\nu K_\nu(x')}{\Gamma(\nu)},$$

where

$K_\nu(\cdot)$ is the modified Bessel function of the second kind,

$\ell =$ **params**(1), $\ell > 0$,

$\nu =$ **params**(2), $\nu > 0$.

icov1 $= 11$
Continuously parameterised variogram with compact support

$$\gamma(x) = \begin{cases} \sigma^2 \frac{2^{1-\nu}(x')^\nu K_\nu(x')}{\Gamma(\nu)}\left(1 + 8x'' + 25(x'')^2 + 32(x'')^3\right)(1 - x'')^8, & x'' < 1, \\ 0, & x'' \geq 1, \end{cases}$$

where

$x'' = \frac{x'}{s}$,

$K_\nu(\cdot)$ is the modified Bessel function of the second kind,

$\ell =$ **params**(1), $\ell > 0$,

$s = \mathbf{params}(2)$, $s > 0$ (second correlation length),

$\nu = \mathbf{params}(3)$, $\nu > 0$.

icov1 $= 12$
Generalized hyperbolic distribution variogram

$$\gamma(x) = \sigma^2 \frac{\left(\delta^2 + (x')^2\right)^{\frac{\lambda}{2}}}{\delta^\lambda K_\lambda(\kappa\delta)} K_\lambda\left(\kappa\left(\delta^2 + (x')^2\right)^{\frac{1}{2}}\right),$$

where

$K_\lambda(\cdot)$ is the modified Bessel function of the second kind,

$\ell = \mathbf{params}(1)$, $\ell > 0$,

$\lambda = \mathbf{params}(2)$, no constraint on λ

$\delta = \mathbf{params}(3)$, $\delta > 0$,

$\kappa = \mathbf{params}(4)$, $\kappa > 0$.

icov1 $= 13$
Cosine variogram

$$\gamma(x) = \sigma^2 \cos(x'),$$

where

$\ell = \mathbf{params}(1)$, $\ell > 0$.

icov1 $= 14$
Used for simulating fractional Brownian motion $B^H(t)$. Fractional Brownian motion itself is not a stationary Gaussian random field, but its increments $\tilde{X}(i) = B^H(t_i) - B^H(t_{i-1})$ can be simulated in the same way as a stationary random field. The variogram for the so-called 'increment process' is

$$C\left(\tilde{X}(t_i), \tilde{X}\left(t_j\right)\right) = \tilde{\gamma}(x) = \frac{\delta^{2H}}{2}\left(\left|\frac{x}{\delta} - 1\right|^{2H} + \left|\frac{x}{\delta} + 1\right|^{2H} - 2\left|\frac{x}{\delta}\right|^{2H}\right),$$

where

$x = t_j - t_i$,

$H = \mathbf{params}(1)$, $0 < H < 1$, H is the Hurst parameter,

$\delta = \mathbf{params}(2)$, $\delta > 0$, normally $\delta = t_i - t_{i-1}$ is the (fixed) stepsize.

We scale the increments to set $\gamma(0) = 1$; let $X(i) = \frac{\tilde{X}(i)}{\delta^{-H}}$, then

$$C\left(X(t_i), X\left(t_j\right)\right) = \gamma(x) = \frac{1}{2}\left(\left|\frac{x}{\delta} - 1\right|^{2H} + \left|\frac{x}{\delta} + 1\right|^{2H} - 2\left|\frac{x}{\delta}\right|^{2H}\right).$$

The increments $X(i)$ can then be simulated using nag_rand_field_1d_generate (g05zp), then multiplied by δ^H to obtain the original increments $\tilde{X}(i)$ for the fractional Brownian motion.

Constraint: **icov1** $= 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13$ or 14.

6: **params**(**np**) – REAL (KIND=nag_wp) array

The parameters set for the variogram.

Constraint: see **icov1** for a description of the individual parameter constraints.

5.2 Optional Input Parameters

1: **maxm** – INTEGER

Suggested value: 2^{k+2} where $k = 1 + \lceil \log_2 (\mathbf{ns} - 1) \rceil$.

Default: $2^{3+\text{ceiling}\log_2 (\mathbf{ns}-1)}$

The maximum size of the circulant matrix to use. For example, if the embedding matrix is to be allowed to double in size three times before the approximation procedure is used, then choose $\mathbf{maxm} = 2^{k+2}$ where $k = 1 + \lceil \log_2 (\mathbf{ns} - 1) \rceil$.

Constraint: $\mathbf{maxm} \geq 2^k$, where k is the smallest integer satisfying $2^k \geq 2(\mathbf{ns} - 1)$.

2: **np** – INTEGER

Default: the dimension of the array **params**.

The number of parameters to be set. Different variograms need a different number of parameters.

icov1 $= 6$
np must be set to 0.

icov1 $= 3$, 4, 5, 7, 9 or 13
np must be set to 1.

icov1 $= 1$, 2, 8, 10 or 14
np must be set to 2.

icov1 $= 11$
np must be set to 3.

icov1 $= 12$
np must be set to 4.

3: **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.

4: **icorr** – INTEGER

Default: **icorr** $= 0$

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

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

5.3 Output Parameters

1: **lam**(**maxm**) – REAL (KIND=nag_wp) array

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

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

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

3: **m** – INTEGER

The size of the embedding matrix.

4: **approx** – INTEGER

Indicates whether approximation was used.

approx $= 0$
No approximation was used.

approx $= 1$
Approximation was used.

5: **rho** – REAL (KIND=nag_wp)

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

6: **icount** – INTEGER

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

7: **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.

8: **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.

ifail $= 2$

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

ifail $= 3$

Constraint: **xmax** > 0.0.

ifail $= 4$

Constraint: the minimum calculated value for **maxm** is $\langle value \rangle$.
Where the minimum calculated value is given by 2^k, where k is the smallest integer satisfying $2^k \geq 2(\mathbf{ns} - 1)$.

ifail $= 5$

Constraint: **var** ≥ 0.0.

ifail $= 6$

Constraint: **icov1** ≥ 1 and **icov1** ≤ 14.

ifail $= 7$

Constraint: for **icov1** $= \langle value \rangle$, **np** $= \langle value \rangle$.

ifail $= 8$

Constraint: dependent on **icov1**.

ifail $= 9$

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

ifail $= 10$

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 value of **maxm** to attempt to avoid approximation.

8 Further Comments

None.

9 Example

This example calls nag_rand_field_1d_predef_setup (g05zn) to calculate the eigenvalues of the embedding matrix for 8 sample points of a random field characterized by the symmetric stable variogram (**icov1** $= 1$).

9.1 Program Text

```
function g05zn_example

fprintf('g05zn example results\n\n');

% Choose the Symmetric stable variogram
icov1 = nag_int(1);
params = [0.1; 1.2];

% Random Field variance
var = 0.5;
% Domain endpoints
xmin = -1;
xmax = 1;
% Number of sample points
ns = nag_int(8);
% scaling factor, rho=1
icorr = nag_int(2);

% Get square roots of the eigenvalues of the embedding matrix
[lam, xx, m, approx, rho, icount, eig, ifail] = ...
```

```
  g05zn( ...
         ns, xmin, xmax, var, icov1, params, 'icorr', icorr);

fprintf('\nSize of embedding matrix = %d\n\n', m);

% 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');
fprintf('%9.5f%9.5f%9.5f%9.5f\n',lam(1:m));
```

9.2 Program Results

```
     g05zn example results

Size of embedding matrix = 16

Approximation not required

Square roots of eigenvalues of embedding matrix:
  0.74207  0.73932  0.73150  0.71991
  0.70639  0.69304  0.68184  0.67442
  0.67182  0.67442  0.68184  0.69304
  0.70639  0.71991  0.73150  0.73932
```