SAS/IML Function Modules for Multivariate Random Sampling

Overview

For certain kinds of statistical simulations and Bayesian analyses, it is necessary to generate random samples from multivariate distributions. SAS/IML software provides the RANDGEN function for generating random samples from univariate distributions. However, the only subroutine for sampling from multivariate distributions is SAS/IML's VNORMAL call, which samples from multivariate normal distributions.

The typical method of generating a multivariate sample is to transform a sample from a related univariate distribution. Thus SAS/IML is a natural choice for generating multivariate samples.

This document introduces modules that sample from common multivariate distributions.

The SAS/IML function modules and associated multivariate distributions are as follows:

RANDDIRICHLET generates a random sample from a Dirichlet distribution (a multivariate generalization of the beta distribution).

RANDMULTINOMIAL generates a random sample from a multinomial distribution (a multivariate generalization of the binomial distribution)

RANDMVT generates a random sample from a multivariate Student's t distribution.

RANDNORMAL generates a random sample from a multivariate normal distribution.

RANDWISHART generates a random sample from a Wishart distribution (a multivariate generalization of the gamma distribution).

All of the modules compute their results by using transformations of univariate random samples generated using the RANDGEN function. Thus you can use the RANDSEED subroutine to set the seed for the modules.

While you can currently sample from a multivariate normal distribution by using the built-in SAS/IML subroutine VNORMAL, VNORMAL does not use the random number seed set in RANDSEED. Thus, to ensure independence and reproducibility of random number streams, the RANDNORMAL function is provided in this package.

In the following sections, N is used to denote the number of observations sampled from a multivariate distribution in p variables.

For an overview of multivariate sampling, see Gentle (2003).

RANDDIRICHLET Function

generates a random sample from a Dirichlet distribution

RANDDIRICHLET(*N, Shape* **)**

The inputs are as follows:

N	is the number of desired observations sampled from the distribution.
Shape	is a $1 \times (p+1)$ vector of shape parameters for the distribution, $Shape[i] > 0$.

The Dirichlet distribution is a multivariate generalization of the beta distribution. The RANDDIRICHLET function returns an $N \times p$ matrix containing N random draws from the Dirichlet distribution.

If $X = \{X_1\ X_2\ \ldots\ X_p\}$ with $\sum_{i=1}^{p} X_i < 1$ and $X_i > 0$ follows a Dirichlet distribution with shape parameter $\alpha = \{\alpha_1\ \alpha_2\ \ldots\ \alpha_{p+1}\}$, then

- the probability density function for x is

$$f(x;\alpha) = \frac{\Gamma(\sum_{i=1}^{p+1}\alpha_i)}{\prod_{i=1}^{p+1}\Gamma(\alpha_i)} \prod_{i=1}^{p} {x_i}^{\alpha_i - 1}(1 - x_1 - x_2 - \ldots - x_p)^{\alpha_{p+1}-1}$$

- if $p = 1$, the probability distribution is a beta distribution.
- if $\alpha_0 = \Sigma_{i=1}^{p+1}\alpha_i$, then
  - the expected value of X_i is α_i/α_0.
  - the variance of X_i is $\alpha_i(\alpha_0 - \alpha_i)/(\alpha_0^2(\alpha_0 + 1))$.
  - the covariance of X_i and X_j is $-\alpha_i\alpha_j/(\alpha_0^2(\alpha_0 + 1))$.

The following example generates 1000 samples from a two-dimensional Dirichlet distribution. Each row of the returned matrix `x` is a row vector sampled from the Dirichlet distribution. The example then computes the sample mean and covariance and compares them with the expected values.

```
call randseed(1);
n = 1000;
Shape = {2, 1, 1};
x = RANDDIRICHLET(n,Shape);
Shape0 = sum(Shape);
d = nrow(Shape)-1;
s = Shape[1:d];
ExpectedValue = s`/Shape0;
Cov = -s*s` / (Shape0##2*(Shape0+1));
/* replace diagonal elements with variance */
Variance = s#(Shape0-s) / (Shape0##2*(Shape0+1));
do i = 1 to d;
   Cov[i,i] = Variance[i];
```

```
end;

SampleMean = x[:,];
n = nrow(x);
y = x - repeat( SampleMean, n );
SampleCov = y`*y / (n-1);
print SampleMean ExpectedValue, SampleCov Cov;
```

```
      SampleMean            ExpectedValue

0.4992449 0.2485677          0.5      0.25

       SampleCov                    Cov

0.0502652 -0.026085         0.05    -0.025
-0.026085 0.0393922       -0.025    0.0375
```

For further details on sampling from the Dirichlet distribution, see Kotz et al. (2000, p. 448), Gentle (2003, p. 205), or Devroye (1986, p. 593).

RANDMULTINOMIAL Function

generates a random sample from a multinomial distribution

RANDMULTINOMIAL(*N, NumTrials, Prob* **)**

The inputs are as follows:

N	is the number of desired observations sampled from the distribution.
$NumTrials$	is the number of trials for each observation. $NumTrials[j] \geq 0$, for $j = 1 \ldots p$.
Prob	is a $1 \times p$ vector of probabilities with $0 < Prob[j] \leq 1$ and $\Sigma_{j=1}^{p} Prob[j] = 1$.

The multinomial distribution is a multivariate generalization of the binomial distribution. For each trial, $Prob[j]$ is the probability of event E_j, where the E_j are mutually exclusive and $\Sigma_{j=1}^{p} Prob[j] = 1$.

The RANDMULTINOMIAL function returns an $N \times p$ matrix containing N observations of *NumTrials* random draws from the multinomial distribution. Each row of the resulting matrix is an integer vector $\{X_1\ X_2\ \ldots\ X_p\}$ with $\Sigma X_j = NumTrials$. That is, for each row, X_j indicates how many times event E_j occurred in *NumTrials* trials.

If $X = \{X_1\ X_2\ \ldots\ X_p\}$ follows a multinomial distribution with n trials and probabilities $\rho = \{\rho_1\ \rho_2\ \ldots\ \rho_p\}$, then

- the probability density function for x is

$$f(x; n, \rho) = \frac{n!}{\prod_{i=1}^{p} x_i!} \prod_{i=1}^{p} {\rho_i}^{x_i}$$

- the expected value of X_i is $n\rho_i$.
- the variance of X_i is $n\rho_i(1 - \rho_i)$.
- the covariance of X_i with X_j is $-n\rho_i\rho_j$.
- if $p = 1$ then X is constant.
- if $p = 2$ then X_1 is Binomial(n, ρ_1) and X_2 is Binomial(n, ρ_2).

The following example generates 1000 samples from a multinomial distribution with three mutually exclusive events. For each sample, ten events are generated. Each row of the returned matrix `x` represents the number of times each event was observed. The example then computes the sample mean and covariance and compares them with the expected values.

```
call randseed(1);
prob = {0.3,0.6,0.1};
NumTrials = 10;
N = 1000;
x = RANDMULTINOMIAL(N,NumTrials,prob);
ExpectedValue = NumTrials * prob`;
Cov = -NumTrials*prob*prob`;
/* replace diagonal elements of Cov with Variance */
Variance = -NumTrials*prob#(1-prob);
d = nrow(prob);
do i = 1 to d;
   Cov[i,i] = Variance[i];
end;

SampleMean = x[:,];
n = nrow(x);
y = x - repeat( SampleMean, n );
SampleCov = y`*y / (n-1);
print SampleMean, ExpectedValue, SampleCov, Cov;
```

```
        SampleMean                              ExpectedValue

  2.971     5.972     1.057                 3         6         1

        SampleCov                                      Cov

  2.0622212 -1.746559 -0.315663          -2.1      -1.8      -0.3
  -1.746559 2.3775936 -0.631035          -1.8      -2.4      -0.6
  -0.315663 -0.631035 0.9466977          -0.3      -0.6      -0.9
```

For further details on sampling from the multinominal distribution, see Gentle (2003, p. 198) or Fishman (1996, pp. 224–225).

RANDMVT Function

generates a random sample from a multivariate Student's t distribution

RANDMVT(*N, DF, Mean, Cov* **)**

The inputs are as follows:

N	is the number of desired observations sampled from the multivariate Student's t distribution.
DF	is a scalar value representing the degrees of freedom for the t distribution.
Mean	is a $1 \times p$ vector of means.
Cov	is a $p \times p$ symmetric positive definite variance-covariance matrix.

The RANDMVT function returns an $N \times p$ matrix containing N random draws from the Student's t distribution with *DF* degrees of freedom, mean vector *Mean* and covariance matrix *Cov*.

If X follows a multivariate t distribution with ν degrees of freedom, mean vector μ and variance-covariance matrix Σ, then

- the probability density function for x is

$$f(x;\nu,\mu,\Sigma) = \frac{\Gamma((\nu+p)/2)}{|\Sigma|^{1/2}(\pi\nu)^{p/2}\Gamma(\nu/2)}\left(1+\frac{(x-\mu)\Sigma^{-1}(x-\mu)^T}{\nu}\right)^{-(\nu+p)/2}$$

- if $p = 1$, the probability density function reduces to a univariate Student's t distribution.
- the expected value of X_i is μ_i.
- the covariance of X_i and X_j is $\frac{\nu}{\nu-2}\Sigma_{ij}$ when $\nu > 2$.

The following example generates 1000 samples from a two-dimensional t distribution with 7 degrees of freedom, mean vector $(1\,2)$, and covariance matrix S. Each row of the returned matrix x is a row vector sampled from the t distribution. The example then computes the sample mean and covariance and compares them with the expected values.

```
call randseed(1);
N=1000;
DF = 4;
Mean = {1 2};
S = {1 1, 1 5};
x = RandMVT( N, DF, Mean, S );
SampleMean = x[:,];
n = nrow(x);
y = x - repeat( SampleMean, n );
SampleCov = y`*y / (n-1);
```

```
Cov = (DF/(DF-2)) * S;
print SampleMean Mean, SampleCov Cov;
```

```
SampleMean                Mean

1.0768636 2.0893911       1         2

SampleCov                 Cov

1.8067811 1.8413406       2         2
1.8413406 9.7900638       2        10
```

In the preceding example, the columns (marginals) of `X` do *not* follow univariate t distributions. If you want a sample whose marginals are univariate t, then you need to scale each column of the output matrix:

```
x = RandMVT( N, DF, Mean, S );
StdX = x / sqrt(diag(S)); /* StdX columns are univariate t */
```

Equivalently, you can generate samples whose marginals are univariate t by passing in a correlation matrix instead of a general covariance matrix.

For further details on sampling from the multivariate t distribution, see Kotz and Nadarajah (2004, pp. 1–11).

RANDNORMAL Function

generates a random sample from a multivariate normal distribution

RANDNORMAL(*N, Mean, Cov* **)**

The inputs are as follows:

N	is the number of desired observations sampled from the multivariate normal distribution.
Mean	is a $1 \times p$ vector of means.
Cov	is a $p \times p$ symmetric positive definite variance-covariance matrix.

The RANDNORMAL function returns an $N \times p$ matrix containing N random draws from the multivariate normal distribution with mean vector *Mean* and covariance matrix *Cov*.

If X follows a multivariate normal distribution with mean vector μ and variance-covariance matrix Σ, then

- the probability density function for x is

$$f(x;\mu,\Sigma) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{(x-\mu)\Sigma^{-1}(x-\mu)^T}{2}\right)$$

- if $p = 1$, the probability density function reduces to a univariate normal distribution.
- the expected value of X_i is μ_i.
- the covariance of X_i and X_j is Σ_{ij}.

The following example generates 1000 samples from a two-dimensional multivariate normal distribution with mean vector $(1\,2)$, correlation matrix Corr, and variance vector Var. Each row of the returned matrix x is a row vector sampled from the multivariate normal distribution. The example then computes the sample mean and covariance and compares them with the expected values.

```
call randseed(1);
N=1000;
Mean = {1 2};
Corr = {0.6 0.5,0.5 0.9};
Var = {4 9};
/*create the covariance matrix*/
Cov = Corr # sqrt(Var` * Var);
x = RANDNORMAL( N, Mean, Cov );
SampleMean = x[:,];
n = nrow(x);
y = x - repeat( SampleMean, n );
SampleCov = y`*y / (n-1);
print SampleMean Mean, SampleCov Cov;
```

```
      SampleMean                Mean

1.0619604 2.1156084         1         2

       SampleCov                 Cov

2.5513518 3.2729559       2.4       3
3.2729559 8.7099585         3     8.1
```

For further details on sampling from the multivariate normal distribution, see Gentle (2003, p. 197).

RANDWISHART Function

generates a random sample from a Wishart distribution

RANDWISHART(*N, DF, Sigma* **)**

The inputs are as follows:

N	is the number of desired observations sampled from the distribution.
DF	is a scalar value representing the degrees of freedom, $DF \geq p$.

Sigma is a $p \times p$ symmetric positive definite matrix.

The RANDWISHART function returns an $N \times (p \times p)$ matrix containing N random draws from the Wishart distribution with *DF* degrees of freedom. Each row of the returned matrix represents a $p \times p$ matrix.

The Wishart distribution is a multivariate generalization of the gamma distribution. (Note, however, that Kotz et al. (2000) suggest that the term "multivariate gamma distribution" should be restricted to those distributions for which the marginal distributions are univariate gamma. This is not the case with the Wishart distribution.) A Wishart distribution is a probability distribution for nonnegative definite matrix-valued random variables. These distributions are often used to estimate covariance matrices.

If a $p \times p$ nonnegative definite matrix X follows a Wishart distribution with parameters ν degrees of freedom and a $p \times p$ symmetric positive definite matrix Σ, then

- the probability density function for x is

$$f(x;\nu,\Sigma) = \frac{|x|^{(\nu-p-1)/2}\exp(-\frac{1}{2}\,\text{trace}(x\,\Sigma^{-1}))}{2^{p\nu/2}|\Sigma|^{\nu/2}\pi^{p(p-1)/4}\prod_{i=1}^{p}\Gamma(\frac{\nu-i+1}{2})}$$

- if $p = 1$ and $\Sigma = 1$, then the Wishart distribution reduces to a chi-square distribution with ν degrees of freedom.
- the expected value of X is $\nu\Sigma$.

The following example generates 1000 samples from a Wishart distribution with 7 degrees of freedom and 2×2 matrix parameter S. Each row of the returned matrix x represents a 2×2 nonnegative definite matrix. (You can reshape the ith row of x with the SHAPE function.) The example then computes the sample mean and compares them with the expected value.

```
call randseed(1);
N=1000;
DF = 7;
S = {1 1, 1 5};
x = RandWishart( N, DF, S );
ExpectedValue = DF * S;
SampleMean = shape( x[:,], 2, 2);
print SampleMean ExpectedValue;
```

```
      SampleMean              ExpectedValue

7.0518633 14.103727            7        14
14.103727 28.207453           14        28
```

For further details on sampling from the Wishart distribution, see Johnson (1987, pp. 203–204).

Reference

Devroye, L. (1986), *Non-Uniform Random Variate Generation*, New York: Springer-Verlag, Inc., 593–596.

Fishman, G.S. (1996), *Monte Carlo: Concepts, Algorithms, and Applications*, New York: Springer-Verlag, Inc., 224–225.

Gentle, J.E. (2003), *Random Number Generation and Monte Carlo Methods*, New York: Springer-Verlag, Inc., 197–206.

Johnson, M.E. (1987), *Multivariate Statistical Simulation*, New York: John Wiley & Sons, Inc., 203–204.

Kotz, S., Balakrishnan, N., and Johnson, N.L. (2000), *Continuous Multivariate Distributions*, New York: John Wiley & Sons, Inc., 485–488.

Kotz, S. and Nadarajah, S. (2004), *Multivariate t Distributions and their Applications*, New York: Cambridge University Press, 1–12.