Chapter 107
The SIMNORMAL Procedure

Overview: SIMNORMAL Procedure

The SIMNORMAL procedure can perform conditional and unconditional simulation for a set of correlated normal or Gaussian random variables.

The means, variances, and covariances (or correlations) are read from an input TYPE=CORR or TYPE=COV data set. This data set is typically produced by the CORR procedure. Conditional simulations are performed by appending a special observation, identified by the value of `COND` for the _TYPE_ variable, which contains the conditioning value.

The output data set from PROC SIMNORMAL contains simulated values for each of the analysis variables. Optionally, the output data set also contains the seed stream and the values of the conditioning variables. PROC SIMNORMAL produces no printed output.

Getting Started: SIMNORMAL Procedure

The following example illustrates the use of PROC SIMNORMAL to generate two normal random variates that have specified means and covariance.
In this example, the means and covariances are given; these might have come from previous experiments, observational studies, or other considerations.

First you create a _TYPE_=COV data set as the input data set, and then you run PROC SIMNORM with NUMREAL=5000, creating a sample that contains 5,000 observations. The simple statistics of this sample are checked using PROC CORR. The results are shown in Figure 107.1.

```
data scov(type=COV) ;	   input _TYPE_ $ 1-4 _NAME_ $ 9-10 S1 S2 ;
   datalines ;
   COV  S1  1.915  0.3873
   COV  S2  0.3873  4.321
   MEAN 1.305  2.003
   run;

proc simnorm data=scov outsim=ssim
   numreal = 5000
   seed = 54321 ;
   var s1 s2 ;
   run;

proc corr data=ssim cov ;
   var s1 s2 ;
   title "Statistics for PROC SIMNORM Sample Using NUMREAL=5000" ;
   run;
```

**Figure 107.1** Statistics for PROC SIMNORM Sample Using NUMREAL=5000

**Statistics for PROC SIMNORM Sample Using NUMREAL=5000**

<table>
<thead>
<tr>
<th>The CORR Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2 Variables:</strong> S1 S2</td>
</tr>
<tr>
<td><strong>Covariance Matrix, DF = 4999</strong></td>
</tr>
<tr>
<td>S1</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>S1</td>
</tr>
<tr>
<td>S2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Simple Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Variable</strong></td>
</tr>
<tr>
<td>S1</td>
</tr>
<tr>
<td>S2</td>
</tr>
</tbody>
</table>
Syntax: SIMNORMAL Procedure

The following statements are available in the SIMNORMAL procedure:

```
PROC SIMNORMAL DATA=SAS-data-set < options > ;
   VAR variables ;
   BY variables ;
   CONDITION variables ;
```

Both the PROC SIMNORMAL and VAR statements are required. The following sections describe the PROC SIMNORMAL statement and then describe the other statements in alphabetical order.

### PROC SIMNORMAL Statement

```
PROC SIMNORMAL DATA=SAS-data-set < options > ;
```

The PROC SIMNORMAL statement invokes the SIMNORMAL procedure. Table 107.1 summarizes the options available in the PROC SIMNORMAL statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify Input and Output Data Sets</td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies input data set (TYPE=CORR, COV, and so on)</td>
</tr>
<tr>
<td>OUT=</td>
<td>Creates output data set that contains simulated values</td>
</tr>
<tr>
<td>Seed Values</td>
<td></td>
</tr>
<tr>
<td>SEED=</td>
<td>Specifies seed value (integer)</td>
</tr>
<tr>
<td>SEEDBY</td>
<td>Requests reinitialization of seed for each BY group</td>
</tr>
<tr>
<td>Control Contents of OUT= Data Set</td>
<td></td>
</tr>
<tr>
<td>OUTSEED</td>
<td>Requests seed values written to OUT= data set</td>
</tr>
<tr>
<td>OUTCOND</td>
<td>Requests conditioning variable values written to OUT=data set</td>
</tr>
<tr>
<td>Control Number of Simulated Values</td>
<td></td>
</tr>
<tr>
<td>NUMREAL=</td>
<td>Specifies the number of realizations for each BY group written to the OUT= data set</td>
</tr>
<tr>
<td>Singularity Criteria</td>
<td></td>
</tr>
<tr>
<td>SINGULAR1=</td>
<td>Sets the singularity criterion for Cholesky decomposition</td>
</tr>
<tr>
<td>SINGULAR2=</td>
<td>Sets the singularity criterion for covariance matrix sweeping</td>
</tr>
</tbody>
</table>

The following options can be used with the PROC SIMNORMAL statement.
**DATA=SAS-data-set**

specifies the input data set that must be a specially structured TYPE=CORR, COV, UCORR, UCOV, or SSCP SAS data set. If the DATA= option is omitted, the most recently created SAS data set is used.

**SEED=seed-value**

specifies the seed to use for the random number generator. If the SEED= value is omitted, the system clock is used. If the system clock is used, a note is written to the log; the note gives the seed value based on the system clock. In addition, the random seed stream is copied to the OUT= data set if the OUTSEED option is specified.

**SEEDBY**

specifies that the seed stream be reinitialized for each BY group. By default, a single random stream is used over all BY groups. If you specify SEEDBY, the random stream starts again at the initial seed value. This initial value is from the SEED= value that you specify. If you do not specify a SEED=value, the system clock generates this initial seed.

For example, suppose you had a TYPE=CORR data set with BY groups, and the mean, variances, and covariance or correlation values were identical for each BY group. Then if you specified SEEDBY, the simulated values in each BY group in the OUT= data set would be identical.

**OUT=SAS-data-set**

specifies a SAS data set in which to store the simulated values for the VAR variables. If you omit the OUT=option, the output data set is created and given a default name by using the DATA convention.

See the section “OUT= Output Data Set” on page 8917 for details.

**NUMREAL=n**

specifies the number of realizations to generate. A value of NUMREAL=500 generates 500 observations in the OUT=dataset, or 500 observations within each BY group if a BY statement is given.

NUMREAL can be abbreviated as NUMR or NR.

**OUTSEED**

requests that the seed values be included in the OUT= data set. The variable Seed is added to the OUT= data set. The first value of Seed is the SEED= value specified in the PROC SIMNORMAL statement (or obtained from the system clock); subsequent values are produced by the random number generator.

**OUTCOND**

requests that the values of the conditioning variables be included in the OUT= data set. These values are constant for the data set or within a BY group. Note that specifying OUTCOND can greatly increase the size of the OUT= data set. This increase depends on the number of conditioning variables.

**SINGULAR1=number**

specifies the first singularity criterion, which is applied to the Cholesky decomposition of the covariance matrix. The SINGULAR1= value must be in the range \((0, 1]\). The default value is \(10^{-8}\). SINGULAR1 can be abbreviated SING1.

**SINGULAR2=number**

specifies the second singularity criterion, which is applied to the sweeping of the covariance or correlation matrix to obtain the conditional covariance. The SINGULAR2=option is applicable only when a CONDITION statement is given. The SINGULAR2= value must be in the range \((0, 1]\). The default value is \(10^{-8}\). SINGULAR2 can be abbreviated SING2.
BY Statement

BY variables;

A BY statement can be used with the SIMNORMAL procedure to obtain separate simulations for each covariance structure defined by the BY variables. When a BY statement appears, the procedure expects the input DATA= data set to be sorted in the order of the BY variables. If a CONDITION statement is used along with a BY statement, there must be a _TYPE_='COND’ observation within each BY group. Note that if a BY statement is specified, the number of realizations specified by the NUMREAL= option are produced for each BY group.

CONDITION Statement

CONDITION | COND variables;

A CONDITION statement specifies the conditioning variables. The presence of a CONDITION statement requests that a conditional simulation be performed.

The lack of a CONDITIONAL statement simply means that an unconditional simulation for the VAR variables is to be performed.

If a CONDITION statement is given, the variables listed must be numeric variables in the DATA= data set. This requires a conditioning value for each of the CONDITION variables. This value is supplied by adding a _TYPE_='COND’ observation for each CONDITION variable. Such observations are added to the DATA= data set by a DATA step.

Note that a data set created by the CORR procedure is automatically given the TYPE=COV, UCOV, CORR, or UCORR attribute, so you do not have to specify the TYPE= option in the DATA= option in the PROC SIMNORMAL statement. However, when adding the conditioning values by using a DATA step with a SET statement, you must use the TYPE=COV, UCOV, CORR, or UCORR attribute in the new data set. See the section “Getting Started: SIMNORMAL Procedure” on page 8913 for an example in which the TYPE is set.

VAR Statement

VAR variables;

Use the VAR statement to specify the analysis variables. Only numeric variables can be specified. If a VAR statement is not given, all numeric variables in the DATA= data set that are not in the CONDITION or BY statement are used.

OUT= Output Data Set


The OUT= data set contains the following variables:

- all variables listed in the VAR statement
all variables listed in the BY statement, if one is given

- Rnum, which is the realization number within the current BY group

- Seed, which is current seed value, if the OUTSEED option is specified

- all variables listed in the CONDITION statement, if a CONDITION statement is given and the OUTCOND option is specified

The number of observations is determined by the value of the NUMREAL= option. If there are no BY groups, the number of observations in the OUT= data set is equal to the value of the NUMREAL= option. If there are BY groups, there are number of observations equals the value of the NUMREAL= option for each BY group.

---

Details: SIMNORMAL Procedure

Introduction

There are a number of approaches to simulating a set of dependent random variables. In the context of spatial random fields, these include sequential indicator methods, turning bands, and the Karhunen-Loève expansion. See Christakos (1992, Chapter 8) and Deutsch and Journel (1992, Chapter 5) for details.

In addition, there is the LU decomposition method, a particularly simple and computationally efficient for normal or Gaussian variates. For a given covariance matrix, the \( LU = LL' \) decomposition is computed once, and the simulation proceeds by repeatedly generating a vector of independent \( N(0,1) \) random variables and multiplying by the \( L \) matrix.

One problem with this technique is that memory is required to hold the covariance matrix of all the analysis and conditioning variables in core.

Unconditional Simulation

It is a simple matter to produce an \( N(0,1) \) random number, and by stacking \( k \) such numbers in a column vector you obtain a vector with independent standard normal components \( W \sim N_k(0, I) \). The meaning of the terms independence and randomness in the context of a deterministic algorithm required for the generation of these numbers is somewhat subtle; see Knuth (1973, Vol. 2, Chapter 3) for a discussion of these issues.

Rather than \( W \sim N_k(0, I) \), what is required is the generation of a vector \( Z \sim N_k(0, V) \)—that is,

\[
Z = \begin{bmatrix}
Z_1 \\
Z_2 \\
\vdots \\
Z_k
\end{bmatrix}
\]

with covariance matrix
Conditional Simulation

\[ \mathbf{V} = \begin{pmatrix}
\sigma_{11} & \sigma_{12} & \cdots & \sigma_{1k} \\
\sigma_{21} & \sigma_{22} & \cdots & \sigma_{2k} \\
\vdots & \vdots & \ddots & \vdots \\
\sigma_{k1} & \sigma_{k2} & \cdots & \sigma_{kk}
\end{pmatrix} \]

where

\[ \sigma_{ij} = \text{Cov}(Z_i, Z_j) \]

If the covariance matrix is symmetric and positive definite, it has a Cholesky root \( \mathbf{L} \) such that \( \mathbf{V} \) can be factored as

\[ \mathbf{V} = \mathbf{L} \mathbf{L}' \]

where \( \mathbf{L} \) is lower triangular. See Ralston and Rabinowitz (1978, Chapter 9, Section 3-3) for details. This vector \( \mathbf{Z} \) can be generated by the transformation \( \mathbf{Z} = \mathbf{LW} \). Note that this is where the assumption of multivariate normality is crucial. If \( \mathbf{W} \sim \mathcal{N}_k(0, \mathbf{I}_k) \), then \( \mathbf{Z} = \mathbf{LW} \) is also normal or Gaussian. The mean of \( \mathbf{Z} \) is

\[ \mathbf{E}(\mathbf{Z}) = \mathbf{L}(\mathbf{E(\mathbf{W}))} = \mathbf{0} \]

and the variance is

\[ \text{Var} (\mathbf{Z}) = \text{Var}(\mathbf{LW}) = \mathbf{E} (\mathbf{LWW'}L') = \mathbf{L} \mathbf{E(\mathbf{WW'})} \mathbf{L}' = \mathbf{LL'} = \mathbf{V} \]

Finally, let \( \mathbf{Y}_k = Z_k + \mu_k \); that is, you add a mean term to each variable \( Z_k \). The covariance structure of the \( \mathbf{Y}_k \)'s remains the same. Unconditional simulation is done by simply repeatedly generating \( k \mathcal{N}(0, 1) \) random numbers, stacking them, and performing the transformation

\[ \mathbf{W} \rightarrow \mathbf{Z} = \mathbf{LW} \rightarrow \mathbf{Y} = \mathbf{Z} + \mu \]

---

**Conditional Simulation**

For a conditional simulation, this distribution of

\[ \mathbf{Y} = \begin{bmatrix}
Y_1 \\
Y_2 \\
\vdots \\
Y_k
\end{bmatrix} \]

must be conditioned on the values of the CONDITION variables. The relevant general result concerning conditional distributions of multivariate normal random variables is the following. Let \( \mathbf{X} \sim \mathcal{N}_m(\mu, \Sigma) \), where
\[ X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \]

\[ \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \]

\[ \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \]

and where \( bX_1 \) is \( k \times 1 \), \( X_2 \) is \( n \times 1 \), \( \Sigma_{11} \) is \( k \times k \), \( \Sigma_{22} \) is \( n \times n \), and \( \Sigma_{12} = \Sigma_{21}' \) is \( k \times n \), with \( k + n = m \). The full vector \( X \) has simply been partitioned into two subvectors, \( X_1 \) and \( X_2 \), and \( \Sigma \) has been similarly partitioned into covariances and cross covariances.

With this notation, the distribution of \( X_1 \) conditioned on \( X_2 = x_2 \) is \( N_k(\tilde{\mu}, \tilde{\Sigma}) \), with

\[
\tilde{\mu} = \mu_1 + \Sigma_{12} \Sigma_{22}^{-1} (x_2 - \mu_2)
\]

and

\[
\tilde{\Sigma} = \Sigma_{11} - \Sigma_{12} \Sigma_{22}^{-1} \Sigma_{21}
\]


Using the SIMNORMAL procedure corresponds with the conditional simulation as follows. Let \( Y_1, \ldots, Y_k \) be the VAR variables as before (\( k \) is the number of variables in the VAR list). Let the mean vector for \( Y \) be denoted by \( \mu_1 = E(Y) \). Let the CONDITION variables be denoted by \( C_1, \ldots, C_n \) (where \( n \) is the number of variables in the COND list). Let the mean vector for \( C \) be denoted by \( \mu_2 = E(C) \) and the conditioning values be denoted by

\[
c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}
\]

Then stacking

\[ X = \begin{bmatrix} Y \\ C \end{bmatrix} \]

the variance of \( X \) is

\[ V = \text{Var}(X) = \Sigma = \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix} \]

where \( V_{11} = \text{Var}(Y) \), \( V_{12} = \text{Cov}(Y, C) \), and \( V_{22} = \text{Var}(C) \). By using the preceding general result, the relevant covariance matrix is

\[ \tilde{V} = V_{11} - V_{12} V_{22}^{-1} V_{21} \]

and the mean is

\[ \tilde{\mu} = \mu_1 + V_{12} V_{22}^{-1} (c - \mu_2) \]

By using \( \tilde{V} \) and \( \tilde{\mu} \), simulating \( (Y|C = c) \sim N_k(\tilde{\mu}, \tilde{V}) \) now proceeds as in the unconditional case.
Example: SIMNORM Procedure

The following example illustrates the use of PROC SIMNORMAL to generate variable values conditioned on a set of related or correlated variables.

Suppose you are given a sample of size 50 from ten normally distributed, correlated random variables, \( IN_{1,i}, \ldots, IN_{5,i}, OUT_{1,i}, \ldots, OUT_{5,i}, i = 1, \ldots, 50 \). The first five variables represent input variables for a chemical manufacturing process, and the last five are output variables.

First, the data are input and the correlation structure is determined by using PROC CORR, as in the following statements. The results are shown in Figure 107.2.

```plaintext
data a ;
  input in1-in5 out1-out5 ;
datalines ;
10.7443 9.9026 9.0144 11.7968
7.8599 10.4560 10.0075 8.5875 10.0014 10.3869

... more lines ...
run ;

proc corr data=a cov nocorr outp=outcov ;
  var in1-in5 out1-out5 ;
run ;
```

Figure 107.2 Correlation of Chemical Process Variables

Statistics for PROC SIMNORM Sample Using NUMREAL=5000

The CORR Procedure

<table>
<thead>
<tr>
<th>10 Variables:</th>
<th>in1</th>
<th>in2</th>
<th>in3</th>
<th>in4</th>
<th>in5</th>
<th>out1</th>
<th>out2</th>
<th>out3</th>
<th>out4</th>
<th>out5</th>
</tr>
</thead>
</table>

After the mean and correlation structure are determined, any subset of these variables can be simulated. Suppose you are interested in a particular function of the output variables for two sets of values of the input variables for the process. In particular, you are interested in the mean and variability of the following function over 500 runs of the process conditioned on each set of input values:

\[
    f(out_1, \ldots, out_5) = \frac{out_1 - out_3}{out_1 + out_2 + out_3 + out_4 + out_5}
\]
Although the distribution of these quantities could be determined theoretically, it is simpler to perform a conditional simulation by using PROC SIMNORMAL.

To do this, you first append a _TYPE_='COND' observation to the covariance data set produced by PROC CORR for each group of input values:

```r
data cond1 ;
  _TYPE_='COND' ;
  in1 = 8 ;
  in2 = 10.5 ;
  in3 = 12 ;
  in4 = 13.5 ;
  in5 = 14.4 ;
  output ;
run ;
```

```r
data cond2 ;
  _TYPE_='COND' ;
  in1 = 15.4 ;
  in2 = 13.7 ;
  in3 = 11 ;
  in4 = 7.9 ;
  in5 = 5.5 ;
  output ;
run ;
```

Next, each of these conditioning observations is appended to a copy of the OUTP=OUTCOV data from the CORR procedure, as in the following statements. A new variable, INPUT, is added to distinguish the sets of input values. This variable is used as a BY variable in subsequent steps.

```r
data outcov1 ;
  input=1 ;
  set outcov cond1 ;
run ;
```

```r
data outcov2 ;
  input=2 ;
  set outcov cond2 ;
run ;
```

Finally, these two data sets are concatenated:

```r
data outcov ;
  set outcov1 outcov2 ;
run ;
```

```r
proc print data=outcov ;
  where (_type_ ne 'COV') ;
run ;
```

Figure 107.3 shows the added observations.
You now run PROC SIMNORMAL, specifying the input data set and the VAR and COND variables. Note that you must specify a TYPE=COV or TYPE=CORR for the input data set. PROC CORR automatically assigns a TYPE=COV or TYPE=CORR attribute for the OUTP= data set. However, since the intermediate DATA steps that appended the _TYPE_='COND' observations turned off this attribute, an explicit TYPE=CORR in the DATA= option in the PROC SIMNORMAL statement is needed.

The specification of PROC SIMNORMAL now follows from the problem description. The condition variables are IN1–IN5, the analysis variables are OUT1–OUT5, and 500 realizations are required. A seed value can be chosen arbitrarily, or the system clock can be used. Note that in the following statements, the simulation is done for each of the values of the BY variable INPUT:

```
proc simnormal data=outcov(type=cov)
   out = osim
   numreal = 500
   seed = 33179
;
by input ;
var out1-out5 ;
cond in1-in5 ;
run;
```

The DATA step that follows the simulation computes the function $f(out_1, \ldots, out_5)$; in the following statements the UNIVARIATE procedure computes the simple statistics for this function for each set of conditioning input values. This is shown in Figure 107.4, and Figure 107.5 shows the distribution of the function values for each set of input values by using the SGPANEL procedure.

```
proc univariate data=b ;
   by input ;
   var ff ;
run ;
title ;
```
Example: SIMNORM Procedure

```sas
proc sgpanel data=b ;
    panelby input ;
    REFLINE 0 / axis= x ;
    density ff ;
run ;
```

Figure 107.4 Simple Statistics for ff for Each Set of Input Values

Statistics for PROC SIMNORM Sample Using NUMREAL=5000

The UNIVARIATE Procedure
Variable: ff

<table>
<thead>
<tr>
<th>input=1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Moments</strong></td>
</tr>
<tr>
<td>N</td>
</tr>
<tr>
<td>Mean</td>
</tr>
<tr>
<td>Std Deviation</td>
</tr>
<tr>
<td>Skewness</td>
</tr>
<tr>
<td>Uncorrected SS</td>
</tr>
<tr>
<td>Coeff Variation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>input=1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic Statistical Measures</strong></td>
</tr>
<tr>
<td>Location</td>
</tr>
<tr>
<td>Mean</td>
</tr>
<tr>
<td>Median</td>
</tr>
<tr>
<td>Mode</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>input=1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tests for Location: Mu0=0</strong></td>
</tr>
<tr>
<td>Test</td>
</tr>
<tr>
<td>Student's t</td>
</tr>
<tr>
<td>Sign</td>
</tr>
<tr>
<td>Signed Rank</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>input=1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Quantiles (Definition 5)</strong></td>
</tr>
<tr>
<td>Level</td>
</tr>
<tr>
<td>100% Max</td>
</tr>
<tr>
<td>99%</td>
</tr>
<tr>
<td>95%</td>
</tr>
<tr>
<td>90%</td>
</tr>
<tr>
<td>75% Q3</td>
</tr>
<tr>
<td>50% Median</td>
</tr>
<tr>
<td>25% Q1</td>
</tr>
<tr>
<td>10%</td>
</tr>
<tr>
<td>5%</td>
</tr>
<tr>
<td>1%</td>
</tr>
<tr>
<td>0% Min</td>
</tr>
</tbody>
</table>
Chapter 107: The SIMNORMAL Procedure

Figure 107.4 continued

input=1

<table>
<thead>
<tr>
<th>Extreme Observations</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Lowest</td>
<td>Highest</td>
<td></td>
</tr>
<tr>
<td>Value</td>
<td>Obs</td>
<td>Value</td>
</tr>
<tr>
<td>-0.0985835</td>
<td>471</td>
<td>0.0750538</td>
</tr>
<tr>
<td>-0.0908179</td>
<td>472</td>
<td>0.0794747</td>
</tr>
<tr>
<td>-0.0802423</td>
<td>90</td>
<td>0.0840160</td>
</tr>
<tr>
<td>-0.0760645</td>
<td>249</td>
<td>0.1004812</td>
</tr>
<tr>
<td>-0.0756070</td>
<td>226</td>
<td>0.1126860</td>
</tr>
</tbody>
</table>

Statistics for PROC SIMNORM Sample Using NUMREAL=5000

The UNIVARIATE Procedure
Variable: ff

input=2

Moments

<table>
<thead>
<tr>
<th></th>
<th>Sum Weights</th>
<th>Sum Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>Mean</td>
<td>-0.0405913</td>
<td>20.295631</td>
</tr>
<tr>
<td>Std Deviation</td>
<td>0.03027008</td>
<td>0.00091628</td>
</tr>
<tr>
<td>Skewness</td>
<td>0.1033062</td>
<td>-0.1458848</td>
</tr>
<tr>
<td>Uncorrected SS</td>
<td>1.28104777</td>
<td>Corrected SS 0.4572225</td>
</tr>
<tr>
<td>Coeff Variation</td>
<td>-74.57289</td>
<td>Std Error Mean 0.00135372</td>
</tr>
</tbody>
</table>

input=2

Basic Statistical Measures

<table>
<thead>
<tr>
<th>Location</th>
<th>Variability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>Std Deviation</td>
</tr>
<tr>
<td>Median</td>
<td>Variance</td>
</tr>
<tr>
<td>Mode</td>
<td>Range</td>
</tr>
<tr>
<td></td>
<td>Interquartile Range</td>
</tr>
</tbody>
</table>

input=2

Tests for Location: Mu0=0

<table>
<thead>
<tr>
<th>Test</th>
<th>Statistic</th>
<th>p Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student's t</td>
<td>-29.985</td>
<td>Pr &gt;</td>
</tr>
<tr>
<td>Sign</td>
<td>M</td>
<td>-203</td>
</tr>
<tr>
<td>Signed Rank</td>
<td>S</td>
<td>-58745</td>
</tr>
</tbody>
</table>
**Figure 107.4 continued**

input=2

<table>
<thead>
<tr>
<th>Quantiles (Definition 5)</th>
<th>Quantile</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% Max</td>
<td>0.06101208</td>
</tr>
<tr>
<td>99%</td>
<td>0.02693796</td>
</tr>
<tr>
<td>95%</td>
<td>0.010008202</td>
</tr>
<tr>
<td>90%</td>
<td>-0.00111776</td>
</tr>
<tr>
<td>75% Q3</td>
<td>-0.01847726</td>
</tr>
<tr>
<td>50% Median</td>
<td>-0.04169199</td>
</tr>
<tr>
<td>25% Q1</td>
<td>-0.06187039</td>
</tr>
<tr>
<td>10%</td>
<td>-0.07798499</td>
</tr>
<tr>
<td>5%</td>
<td>-0.08606522</td>
</tr>
<tr>
<td>1%</td>
<td>-0.11026564</td>
</tr>
<tr>
<td>0% Min</td>
<td>-0.12231183</td>
</tr>
</tbody>
</table>

input=2

<table>
<thead>
<tr>
<th>Extreme Observations</th>
<th>Lowest Value</th>
<th>Highest Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value</td>
<td>Obs</td>
</tr>
<tr>
<td>-0.122312</td>
<td>937</td>
<td>0.0272906</td>
</tr>
<tr>
<td>-0.119884</td>
<td>980</td>
<td>0.0291769</td>
</tr>
<tr>
<td>-0.113512</td>
<td>920</td>
<td>0.0388217</td>
</tr>
<tr>
<td>-0.112345</td>
<td>523</td>
<td>0.0477261</td>
</tr>
<tr>
<td>-0.110497</td>
<td>897</td>
<td>0.0610121</td>
</tr>
</tbody>
</table>
Chapter 107: The SIMNORMAL Procedure

Figure 107.5 Frequency Plot for \( \phi \) for Each Set of Input Values

![Frequency Plot](image)

References


Subject Index

SIMNORMAL procedure
  conditional simulation, 8919
  Gaussian random variables, 8913
  introductory example, 8913
  LU decomposition method, 8918
  normal random variables, 8913
  simulation, 8913
  unconditional simulation, 8918
Syntax Index

BY statement
   SIMNORMAL procedure, 8917

CONDITION statement
   CONDITION statement (SIMNORMAL), 8917

DATA= option
   PROC SIMNORMAL statement, 8916

NUMREAL= option
   PROC SIMNORMAL statement, 8916

OUT= option
   PROC SIMNORMAL statement, 8916
OUTCOND option
   PROC SIMNORMAL statement, 8916
OUTSEED option
   PROC SIMNORMAL statement, 8916

PROC SIMNORM statement
   SIMNORMAL procedure, 8915

SEED= option
   PROC SIMNORMAL statement, 8916
SEEDBY option
   PROC SIMNORMAL statement, 8916
SIMNORMAL procedure
   syntax, 8915
   SIMNORMAL procedure, BY statement, 8917
SINGULAR1= option
   PROC SIMNORMAL statement, 8916
SINGULAR2= option
   PROC SIMNORMAL statement, 8916

VAR statement
   VAR statement (SIMNORMAL), 8917