Descriptive Statistics MT

for GAUSS™

Version 1.0

Aptech Systems, Inc.

Documentation Version: June 1, 2004

Part Number: 004379

Contents

Chapter 1

Installation

1.1 UNIX

If you are unfamiliar with UNIX, see your system administrator or system documentation for information on the system commands referred to below. The device names given are probably correct for your system.

1.1.1 Download

1. Copy the `.tar.gz` file to `/tmp`.
2. Unzip the file.

   `gunzip` *appxxx*`.tar.gz`
3. `cd` to the **GAUSS** or **GAUSS Engine** installation directory. We are assuming `/usr/local/gauss` in this case.

   ```
   cd /usr/local/gauss
   ```
4. Untar the file.

   `tar xvf /tmp/`*appxxx*`.tar`

1.1.2 Floppy

1. Make a temporary directory.

   ```
   mkdir /tmp/workdir
   ```

2. `cd` to the temporary directory.

   ```
   cd /tmp/workdir
   ```

3. Use `tar` to extract the files.

   `tar xvf` *device_name*

   If this software came on diskettes, repeat the `tar` command for each diskette.

4. Read the README file.

   ```
   more README
   ```

5. Run the `install.sh` script in the work directory.

   ```
   ./install.sh
   ```

   The directory the files are installed to should be the same as the install directory of **GAUSS** or the **GAUSS Engine**.

6. Remove the temporary directory (optional).

The following device names are suggestions. See your system administrator. If you are using Solaris 2.x, see Section 1.1.3.

Operating System	3.5-inch diskette	1/4-inch tape	DAT tape
Solaris 1.x SPARC	`/dev/rfd0`	`/dev/rst8`	
Solaris 2.x SPARC	`/dev/rfd0a` (vol. mgt. off)	`/dev/rst12`	`/dev/rmt/1l`
Solaris 2.x SPARC	`/vol/dev/aliases/floppy0`	`/dev/rst12`	`/dev/rmt/1l`
Solaris 2.x x86	`/dev/rfd0c` (vol. mgt. off)		`/dev/rmt/1l`
Solaris 2.x x86	`/vol/dev/aliases/floppy0`		`/dev/rmt/1l`
HP-UX	`/dev/rfloppy/c20Ad1s0`		`/dev/rmt/0m`
IBM AIX	`/dev/rfd0`	`/dev/rmt.0`	
SGI IRIX	`/dev/rdsk/fds0d2.3.5hi`		

1.1.3 Solaris 2.x Volume Management

If Solaris 2.x volume management is running, insert the floppy disk and type

```
volcheck
```

to signal the system to mount the floppy.

The floppy device names for Solaris 2.x change when the volume manager is turned off and on. To turn off volume management, become the superuser and type

```
/etc/init.d/volmgt off
```

To turn on volume management, become the superuser and type

```
/etc/init.d/volmgt on
```

1.2 Windows/NT/2000

1.2.1 Download

Unzip the `.zip` file into the **GAUSS** or **GAUSS Engine** installation directory.

1.2.2 Floppy

1. Place the diskette in a floppy drive.
2. Call up a DOS window
3. In the DOS window log onto the root directory of the diskette drive. For example:

```
A:<enter>
cd\<enter>
```

4. Type: **ginstall** *source_drive target_path*

source_drive	Drive containing files to install with colon included For example: **A:**
target_path	Main drive and subdirectory to install to without a final \ For example: **C:\GAUSS**

A directory structure will be created if it does not already exist and the files will be copied over.

*target_path***src**	source code files
*target_path***lib**	library files
*target_path***examples**	example files

1.3 Differences Between the UNIX and Windows/NT/2000 Versions

- If the functions can be controlled during execution by entering keystrokes from the keyboard, it may be necessary to press *Enter* after the keystroke in the UNIX version.

- On the Intel math coprocessors used by the Windows/NT/2000 machines, intermediate calculations have 80-bit precision, while on the current UNIX machines, all calculations are in 64-bit precision. For this reason, **GAUSS** programs executed under UNIX may produce slightly different results, due to differences in roundoff, from those executed under Windows/NT/2000.

Chapter 2

Descriptive Statistics MT

The *DESCRIPTIVE STATISTICS MT* module is a set of procedures which generates basic sample statistics of the variables in **GAUSS** data sets. These statistics describe the numerical characteristics of the random variables, and provide information for further statistical analysis.

DESCRIPTIVE STATISTICS MT uses structures instead of globals to hold program defaults, making it safe to use in multi-threaded applications.

2.1 Getting Started

GAUSS 6.0.25+ and the **GAUSS** Run-Time Library 6.0.20+ are required to use these routines.

2.1.1 README Files

The file `README.dsmt` contains any last minute information on this module. Please read it before using the procedures in this module.

2.2 Setup

In order to use the procedures in the *DESCRIPTIVE STATISTICS MT* module, the DSTATMT library must be active. This is done by including `dstatmt` in the **LIBRARY** statement at the top of your program:

```
library dstatmt,quantal,pgraph;
```

This enables **GAUSS** to find the *DESCRIPTIVE STATISTICS MT* procedures. If you are calling a procedure that takes a **dstatControl** structure as an argument, you also need the statements:

```
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;
```

The call to **dstatControlCreate** sets the members of the **dstatControl** structure to their default values. **dstatControlCreate** can also be used to reset the members of the structure in succeeding executions of the program.

2.3 Data Sets

A **GAUSS** data set is a binary disk file, which is saved with a `.dat` extension. The `.dat` file is comprised of the data and a header which contains the names and types of the variables associated with each column of the data set.

2.3.1 Data Transformations

It is assumed that the data set for analysis is ready before you call the procedures. If you need to modify your data, **GAUSS Data Tool** is available for fast and simple manipulation of data sets. **GAUSS Data Tool** provides users with a powerful and flexible environment for viewing and modifying data. It includes commands for keeping and dropping variables, selecting observations, sorting, merging on a key variable or set of variables, imputing missing data, and transforming data using **GAUSS** functions.

2.3.2 Creating Data Sets

There are three ways to create a **GAUSS** data set.

1. If you have an ASCII format data file, use the **ATOG** utility to convert it into a **GAUSS** data set. For details, see **ATOG** in the *UTILITIES* section of the **GAUSS** manual.

2. If you have a matrix in memory, use the command **create** or **saved** to create a data set. See the *COMMAND REFERENCE* section of the **GAUSS** manual.

3. **GAUSS Data Tool** has commands for creating new data sets and translating ASCII and Excel files to **GAUSS** data sets.

To look at a **GAUSS** data set, use the keyword **datalist**. The syntax is:

```
datalist filename [variables];
```

For details, see **datalist** in the **GAUSS** manual.

2.3.3 Variable Types

GAUSS currently supports three types of variables: numeric, character, and date. Since the types of the variables in a data set are kept in the associated `.dat` file, there is no need for the user to supply the variable types for *DESCRIPTIVE STATISTICS MT* routines that operate on data sets.

2.4 Error Codes

If problems are encountered in data being analyzed, the procedures attempt to trap the errors. Errors are handled with the low order bit of the trap flag. Depending on the value of the trap flag, the procedure either sends an error message indicating the nature of the problem and terminates the program, or returns an error code without termination.

TRAP 0 terminate with error message

TRAP 1 return scalar error code

Error codes are particularly helpful if you are running a large program and need to obtain values to pass to other programs.

2.4.1 Tests for Error Codes

If an error is encountered and a procedure returns an error code, it will appear as a missing value. Use the **SCALERR** procedure to return the value of the error code. For example:

```
{ t, n } = crosstab(ds,dataset,varnm);
errcode = scalerr(t);
if errcode /= 0;
    print "Error " errcode " was encountered.";
    end;
endif;
```

The error code returned by **SCALERR** is an integer.

2.4.2 List of Error Codes

Following is a list of error code definitions:

1	Data file not found.
2	Undefined variables in input argument.
3	Index out of range.
5	Type mismatch.
7	Non-numeric variables included.
8	This function does not compute one or more of the statistics specified.
9	Weight variable must be numeric.
32	Too many cells in procedure.
33	Cannot create crosstable with only one variable.
41	No missing values specified, but missing values were encountered.
77	No cases left after deleting missing observations.

2.5 Getting Help on Procedures

All of the procedures in the *DESCRIPTIVE STATISTICS MT* module are automatically accessible in **GAUSS** if the DSTATMT library is active. You can find the definition of a *DESCRIPTIVE STATISTICS MT* procedure and information about its syntax and arguments as follows:

If you are running Windows, place the cursor on the name of the procedure and press Ctrl-F1.

If you are running UNIX, type browse followed by the name of the procedure.

2.6 Compatibility with Previous Versions

This new version of the *DESCRIPTIVE STATISTICS MT* module requires **GAUSS 6.0.25+** and the **GAUSS** Run-Time Library 6.0.20+. Any programs that you had running under the previous modules may require minor changes before they run successfully under this new version.

Chapter 3

Descriptive Statistics MT Reference

A summary table listing the main procedures is displayed below.

Purpose

Creates a **dstatControl** structure and sets its members to default values.

Library

`dstatmt`

Format

ds = **dstatControlCreate;**

Output

ds	**dstatControl** structure.

Remarks

dstatControlCreate should be called before any procedure in the *DESCRIPTIVE STATISTICS MT* module that takes a **dstatControl** structure as an argument. It may also be called to reset the members of a **dstatControl** structure in succeeding executions of a program that invokes *DESCRIPTIVE STATISTICS MT* procedures.

Source

`dstatsetmt.src`

Purpose

Computes the correlations for variables in a **GAUSS** data set.

Library

`dstatmt`

Format

`{` *cor*,*vc*,*n*,*nms*,*des* `}` **= corr(***ds*,*dataset*,*vars***);**

Input

ds **dstatControl** structure. The following elements of *ds* are referenced within the **corr** routine:

ds.**cor** scalar, if 1, print correlation matrix. Default = 1.

ds.**header** string, specifies the format for the output header. *ds*.**header** can contain zero or more of the following characters:

t	print title (see *ds*.**title**)
l	bracket title with lines
d	print date and time
v	print procedure name and version number
f	print file name being analyzed

Example:

```
ds.header = "tld";
```

If *ds*.**header** == "", no header is printed. Default = "tldvf".

ds.**miss** scalar, determines how missing data is handled.

0	Missing values are not checked for, and so the data set must not have any missing observations. This is the fastest option.
1	Listwise deletion. Removes from computation any observation containing a missing value for any variable included in the analysis.
2	Pairwise deletion. **corr** does not require a complete set of data for each observation. This procedure deals separately with each pair of variables in the matrix, computing the covariance and correlation between that pair on the basis of all cases for which there is data. With pairwise deletion, any pair of variables containing missing values is excluded from the computation of their covariance.

Default = 0.

ds.**mmt** scalar, if 1, print moment matrix. Default = 0.

ds.**output** scalar, determines printing of intermediate results.

0 nothing is written.

1 serial ASCII output format suitable for disk files or printers.

Default = 1.

ds.**range** 2×1 vector, the range of records in the data set used for analysis. The first element is the starting row index, the second element is the ending row index. Default is *ds*.**range** = { 0, 0 }, the whole data set. For example, if one wants the range of data from row 100 to the end of data, then *ds*.**range** should be set as:

```
ds.range = { 100, 0 };
```

ds.**rowfac** scalar, "row factor". If a *DESCRIPTIVE STATISTICS MT* procedure fails due to insufficient memory while attempting to read a **GAUSS** data set, then *ds*.**rowfac** may be set to some value between 0 and 1 to read a *proportion* of the original number of rows of the **GAUSS** data set. For example, setting

```
ds.rowfac = 0.8;
```

causes **GAUSS** to read in only 80% of the rows that were originally calculated.

This global has an effect only when *ds*.**rowsperiter** = 0.

Default = 1.

ds.**rowsperiter** scalar, specifies how many rows of the data set are read per iteration of the read loop. If *ds*.**rowsperiter** = 0, the number of rows to be read is calculated by **corr**. Default = 0.

ds.**stat** scalar, if 1, print univariate descriptive statistics. Default = 1.

ds.**statlist** M×1 vector, where $1 <= M <= 6$. If 0, all univariate descriptive statistics are included in report. Otherwise only the specified statistics are included, in the order in which they appear in *ds*.**statlist**.

Available statistics are:

1	Means
2	Standard Deviation
3	Minimum
4	Maximum
11	Number Valid
12	Number Missing

This global has an effect only when *ds*.**stat** = 1.

Default = 0.

ds.**t** scalar, if 1, print t-tests of hypothesis $H_0 : r = 0$. Based on the formula:

$$\sqrt{n-1}\frac{r}{\sqrt{1-r^2}} \sim t(n-2)$$

Default = 1.

ds.**title** string, this is the title used by *ds*.**header**. Default = "".

ds.**vc** scalar, if 1, print covariance matrix. Default = 0.

ds.**weight** string, name of weight variable. By default, unweighted correlations are calculated.

ds.**weightindex** scalar, index of weight variable. *ds*.**weightindex** overrides *ds*.**weight**. By default, unweighted correlations are calculated.

dataset string, name of data file.

vars K×1 string array, names of variables.
– or –
K×1 numeric vector, indices of variables.

If 0, all variables are included.

■ Output

cor K×K matrix, correlations in the order of *vars*.

vc K×K matrix, covariances in the order of *vars*.

n scalar, number of observations for listwise correlations.
– or –
K×K matrix of number of observations for each correlation.

A matrix is returned if and only if pairwise correlations are selected.

nms K×1 string array, variable names in the order of *vars*.

des K×7 matrix, descriptive statistics:

des[**.,1**] Means
des[**.,2**] Standard deviations
des[**.,3**] Variances
des[**.,4**] Minimum
des[**.,5**] Maximum
des[**.,6**] Number of valid cases
des[**.,7**] Number of missing cases

Error handling is controlled by the low order bit of the trap flag.

TRAP 0 terminate with error message.
TRAP 1 return scalar error code in all return arguments.

The function **SCALERR** can be used to return the value of the error code. The error codes returned are:

1 Data file not found

2 Undefined variables in input argument

3 Index out of range

8 This function does not compute one or more of the statistics specified

9 Weight variable must be numeric

41 No missing values specified, but missing values were encountered

77 No cases left after deleting missing observations

Example

```
library dstatmt;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;

string var = { pub1, pub3, pub6, job, enrol };
ds.weight = "pub1";
ds.miss = 2;
ds.header = "tl";
ds.title =  "corrmt.e:  WEIGHTED PAIRWISE - ALL OPTIONS";
output file = corrmt.out reset;
call corr(ds,"scigau",var);
output off;
```

Source

destatmt.src

Purpose

Creates contingency tables from raw or weighted data contained in a **GAUSS** data set.

Library

`dstatmt`

Format

`{ t,n }` = **crosstab(***ds*,*dataset*,*vars*);

Input

ds **dstatControl** structure. The following elements of *ds* are referenced within the **crosstab** routine:

ds.**case** scalar, if 1, case sensitivity turned on for character variables. Default = 0.

ds.**col** scalar, number of columns to print per section of a table. If a table is large, this allows printing it in sections. Default = 6.

ds.**colp** scalar, if 1, list column percentages. Default = 1.

ds.**header** string, specifies the format for the output header. *ds*.**header** can contain zero or more of the following characters:

t	print title (see *ds*.**title**)
l	bracket title with lines
d	print date and time
v	print procedure name and version number
f	print file name being analyzed

Example:

```
ds.header = "tld";
```

If *ds*.**header** == "", no header is printed. Default = "tldvf".

ds.**miss** scalar, determines how missing data are handled.

0 Missing values are included in the table as a separate category if there are missing observations in the data.

1 Listwise deletion. Removes from computation any observation with a missing value for any variable included in the analysis.

Default = 0.

ds.**output** scalar, determines printing of intermediate results.

0 nothing is written.

1 serial ASCII output format suitable for disk files or printers.

Default = 1.

ds.**range** 2×1 vector, the range of records in the data set used for analysis. The first element is the starting row index, the second element is the ending row index. Default is *ds*.**range** = { 0, 0 }, the whole data set. For example, if one wants the range of data from row 100 to the end of data, then *ds*.**range** should be set as:

```
ds.range = { 100, 0 };
```

ds.**row** scalar, number of rows to print per section of a table. If a table is large, this allows printing it in sections. Default = 3.

ds.**rowfac** scalar, "row factor". If **crosstab** fails due to insufficient memory while attempting to read a **GAUSS** data set, then *ds*.**rowfac** may be set to some value between 0 and 1 to read a *proportion* of the original number of rows of the **GAUSS** data set. For example, setting

```
ds.rowfac = 0.8;
```

causes **GAUSS** to read in only 80% of the rows that were originally calculated.

This global has an effect only when *ds*.**rowsperiter** = 0.

Default = 1.

ds.**rowp** scalar, if 1, list row percentages. Default = 1.

ds.**rowsperiter** scalar, specifies how many rows of the data set are read per iteration of the read loop. If *ds*.**rowsperiter** = 0, the number of rows to be read is calculated by **crosstab**. Default = 0.

ds.**stat** scalar, if 1, print statistics. See documentation of **tblstat** for details on statistics printed. Default = 1.

ds.**title** string, this is the title used by *ds*.**header**. Default = "".

ds.**totp** scalar, if 1, list total percentages. Default = 1.

ds.**weight** string, name of weight variable. By default, unweighted correlations are calculated.

ds.**weightindex** scalar, index of weight variable. *ds*.**weightindex** overrides *ds*.**weight**. By default, unweighted correlations are calculated.

dataset string, name of data set.

vars K×1 string array of variable names for the table.
– or –
K×1 numeric vector of column indices of variables for the table.

K must be at least 2. The first variable in *vars* is the row variable, the second variable is the column variable, the remaining variables are levels of the control variables.

Output

t N×K matrix, table indices.

n N×1 vector, table counts.

Error handling is controlled by the low order bit of the trap flag.

TRAP 0 terminate with error message.

TRAP 1 return scalar error code in all return arguments.

The function **SCALERR** can be used to return the value of the error code. The error codes returned are:

1 Data file not found

2 Undefined variables in input argument

3 Index out of range

9 Weight variable must be numeric

32 Too many cells in crosstable

33 Cannot create crosstable with only one variable

Remarks

This procedure handles both character and numeric data. Date variables are skipped.

crosstab constructs a K-way table from variables $V_i(i = 1, K)$ in *vars*, with variable V_i having K_i categories. The resulting table contains $N = K_1 \times K_2 \times \ldots K_K$ cells. The observed variables for this table are contained in the N×1 vector n. The N cell indices associated with n are contained in the K×N matrix t, where element t_{ij} is the category level of variable V_j for n_i.

The matrices t and n that are returned by **crosstab** are in the appropriate form for input to the programs in the *LOGLINEAR ANALYSIS* module.

A table can contain at most **MAXVEC** cells.

Example

Print a two-way table.

```
library dstatmt;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;

dataset = "mydata";
call crosstab(ds,dataset,2|5); /* crosstab 2nd variable by 5th */
```

Source

`crosstabmt.src`

See also

tblstat, **freq**

Purpose

Computes frequency distributions for variables contained in a **GAUSS** data set.

Library

`dstatmt`

Format

`{` *cats*,*ncats*,*freqn* `}` **= freq(**_ds_,_dataset_,_vars_**);**

Input

ds **dstatControl** structure. The following elements of *ds* are referenced within the **freq** routine:

ds.**case** scalar, if 1, case sensitivity turned on for character variables. Default = 0.

ds.**header** string, specifies the format for the output header. *ds*.**header** can contain zero or more of the following characters:

t	print title (see *ds*.**title**)
l	bracket title with lines
d	print date and time
v	print procedure name and version number
f	print file name being analyzed

Example:

```
ds.header = "tld";
```

If *ds*.**header** == "", no header is printed. Default = "tldvf".

ds.**miss** scalar, determines how missing data are handled.

0	Missing values are included in the table as a separate category if there are missings in the data.
1	Listwise deletion. Removes from computation any observation with a missing value for any variable included in the analysis.

Default = 0.

ds.**output** scalar, determines printing of intermediate results.

0	nothing is written.
1	serial ASCII output format suitable for disk files or printers.

Default = 1.

ds.**range** 2×1 vector, the range of records in data set used for analysis. The first element is the starting row index, the second element is the ending row index. Default is *ds*.**range** = { 0, 0 }, the whole data set. For example, if one wants the range of data from row 100 to the end of data, then *ds*.**range** should be set as:

```
ds.range = { 100, 0 };
```

ds.**rowfac** scalar, "row factor". If **freq** fails due to insufficient memory while attempting to read a **GAUSS** data set, then *ds*.**rowfac** may be set to some value between 0 and 1 to read a *proportion* of the original number of rows of the **GAUSS** data set. For example, setting

```
ds.rowfac = 0.8;
```

causes **GAUSS** to read in only 80% of the rows that were originally calculated.

This global has an effect only when *ds*.**rowsperiter** = 0.

Default = 1.

ds.**rowsperiter** scalar, specifies how many rows of the data set are read per iteration of the read loop. If *ds*.**rowsperiter** = 0, the number of rows to be read is calculated by **freq**. Default = 0.

ds.**sort** scalar, if 1, output is sorted by the names of the variables in *vars*. Default = 0.

ds.**stat** scalar, if 1, print descriptive statistics. Default = 1.

ds.**title** string, this is the title used by *ds*.**header**. Default = "".

ds.**weight** string, name of weight variable. By default, unweighted correlations are calculated.

ds.**weightindex** scalar, index of weight variable. *ds*.**weightindex** overrides *ds*.**weight**. By default, unweighted correlations are calculated.

dataset string, name of data set.

vars K×1 string array, names of variables
– or –
K×1 numeric vector, indices of variables

for which frequency distributions are requested. If *vars* = 0, all the variables in the data set are used.

Output

cats L×1 vector of categories for each variable, where $L = \sum_{i=1}^{K} cat_i$, and cat_i is the number of categories of the i^{th} variable.

ncats K×1 vector of $(cat_1, cat_2, \ldots, cat_K)$, where each element is the number of categories for the corresponding variable.

freqn L×1 vector of frequencies for each variable.

Error handling is controlled by the low order bit of the trap flag.

TRAP 0 terminate with error message.

TRAP 1 return scalar error code in all return arguments.

The function **SCALERR** can be used to return the value of the error code. The error codes returned are:

1 Data file not found

2 Undefined variables in input argument

3 Index out of range

9 Weight variable must be numeric

32 Too many cells in frequency

77 No cases left after deleting missing observations

Remarks

This procedure handles both character and numeric data. Date variables are skipped.

If there are missing data for a variable, counts excluding the missing category are also made.

For each variable in *vars*, **freq** returns a sorted list of all categories of that variable, the number of categories for that variable, and the number of cases in each category.

The procedure **getfreq** gets the categories and frequencies for a specified variable.

The total number of cells in the frequency distributions for all variables in *vars* cannot exceed **MAXVEC**.

Example

Using **freq**:

```
library dstatmt;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;

dataset = "mydata";
call freq(ds,dataset,2|5);  /* <=== Frequencies for the   */
                           /*      2nd and 5th variables */
```

Using **freq**, **getfreq** and **HISTF**:

```
library dstatmt,pgraph;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;
graphset;

print "FR4.E:   Using getfreq and HISTF";
print;
dataset = "freq";
output file = fr1.out reset;
ds.miss = 1;      /* get rid of missing category */
{ cats,ncats,freqs } = freq(ds,dataset,1|2|3);
output off;
/* get frequencies for var 1 */
histf(getfreq(1,cats,ncats,freqs));
```

Source

freqmt.src

See also

getfreq, **freqstat**

Purpose

Prints percentages, descriptive statistics and simple histogram given category values and number of cases in each category.

Library

`dstatmt`

Format

freqstat(*ds*,*nm*,*freq*,*val*,*c***);**

Input

ds **dstatControl** structure. The following elements of *ds* are referenced within the **freqstat** routine:

ds.**freq** scalar, if 1, print frequencies. Default = 1.

ds.**miss** scalar, determines how missing data are handled.

0 Missing values are included in the table as a separate category if there are missings in the data.

1 Missing values are excluded from the table.

Default = 0.

ds.**stat** scalar, if 1, print descriptive statistics. Default = 1.

nm string, name of variable.

freq L×1 vector, number of observations in each category.

val L×1 character vector, label of each category.
– or –
L×1 numeric vector, value of each category.

c scalar, 1 if numeric variable, 0 if character variable.

Output

None.

Remarks

freqstat prints frequencies, percentages, descriptive statistics and a histogram.

freqstat works with matrices in memory; if your data is in a **GAUSS** data set, you should use the procedure **freq**, which works with data sets.

Example

```
library dstatmt;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;

output file = frst.out reset;
freq = { 10, 3, 4 };
cats = { 1, 2, 3 };
freqstat(ds, "var", freq, cats, 1);
output off;
```

Source

freqstatmt.src

See also

freq

Purpose

Gets frequencies and categories for a particular variable from the results returned by **freq**.

Library

`dstatmt`

Format

{ *vfreq*,*vcat* } = **getfreq(***var*,*cats*,*ncats*,*freqs***);**

Input

var	scalar, the index of the variable for which frequencies are desired.
cats	K×1 character vector, labels – or – K×1 numeric vector, category values for all counts in the *freqs* vector.
ncats	L×1 vector, number of categories associated with each variable in the *freqs* vector.
freqs	M×1 vector, frequencies associated with *cats* and *ncats*.

Output

vfreq	Q×1 vector, frequencies associated with specified variable.
vcat	P×1 vector, categories associated with specified variable.

Remarks

The variables *cats*, *ncats* and *freqs* are the results returned from **freq**. **freq** returns vectors for all variables specified. **getfreq** returns the frequencies for a specified variable.

Example

```
library dstatmt,pgraph;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;

{ cats,ncats,freqs } =  freq(ds,dataset,1|2|3);
{ f,c } =
   getfreq(1,cats,ncats,freqs);   /* <=== Get frequencies */
                                  /*      for VAR 1       */
histf(f,c);   /* <=== Plot histogram for VAR 1 */
```

Source

getfreqmt.src

See also

freq

Purpose

Computes multivariate skew and kurtosis on a **GAUSS** data set.

Library

`dstatmt`

Format

{ *skewc*,*skewp*,*kurtc*,*kurtp*,*combc*,*combp* } = **mardia**(*ds*,*dataset*,*vars*);

Input

ds **dstatControl** structure. The following elements of *ds* are referenced within the **mardia** routine:

ds.**header** string, specifies the format for the output header. *ds*.**header** can contain zero or more of the following characters:

t	print title (see *ds*.**title**)
l	bracket title with lines
d	print date and time
v	print procedure name and version number
f	print file name being analyzed

Example:

```
ds.header = "tld";
```

If *ds*.**header** == "", no header is printed. Default = "tldvf".

ds.**output** scalar, determines printing of intermediate results.

0	nothing is written.
1	serial ASCII output format suitable for disk files or printers.

Default = 1.

ds.**range** 2×1 vector, the range of records in data set used for analysis. The first element is the starting row index, the second element is the ending row index. Default is *ds*.**range** = { 0, 0 }, the whole data set. For example, if one wants the range of data from row 100 to the end of data, then *ds*.**range** should be set as:

```
ds.range = { 100, 0 };
```

ds.**title** string, this is the title used by *ds*.**header**. Default = "".

dataset string, name of data file.

vars K×1 string array, names of variables.
– or –
K×1 numeric vector, indices of variables.

If *vars* = 0, all variables are included.

Output

skewc scalar, skew coefficient (unit normal distribution).

skewp scalar, probability of *skewc* being that large or larger if zero in pop.

kurtc scalar, kurtosis coefficient (unit normal distribution).

kurtp scalar, probability of *kurtc* being that large or larger if zero in pop.

combc scalar, summary statistic (chi-squared dist. with 2 df)

skewp scalar, probability of *combc*

Error handling is controlled by the low order bit of the trap flag.

TRAP 0 terminate with error message.

TRAP 1 return scalar error code in all return arguments.

The function **SCALERR** can be used to return the value of the error code. The error codes returned are:

1 Data file not found

2 Undefined variables in input argument

3 Index out of range

7 Non-numeric variables included

77 No cases left after deleting missing observations

Remarks

This procedure handles only numeric data.

mardia uses listwise deletion to handle missings. If any observations contain missings, those observations are removed from computation.

Example

```
library dstatmt;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;

dsn = "xdat";
output file = mardia.out reset;
{ sc,sp,kc,kp,cc,cp } = mardia(ds,dsn,0);
output off;
```

Source

mardiamt.src

Purpose

Computes the descriptive statistics for variables in a **GAUSS** data set.

Library

`dstatmt`

Format

{ *nms*,*mn*,*std*,*min*,*max*,*valid*,*missing* } = **means**(*ds*,*dataset*,*vars*);

Input

ds **dstatControl** structure. The following elements of *ds* are referenced within the **means** routine:

ds.**header** string, specifies the format for the output header. *ds*.**header** can contain zero or more of the following characters:

t	print title (see *ds*.**title**)
l	bracket title with lines
d	print date and time
v	print procedure name and version number
f	print file name being analyzed

Example:

```
ds.header = "tld";
```

If *ds*.**header** == "", no header is printed. Default = "tldvf".

ds.**miss** scalar, determines how missing data are handled.

0	Missing values are not checked for, so the data set must not have any missing observations. This is the fastest option.
1	Listwise deletion. Removes from computation any observation with a missing value for any variable included in the analysis.
2	Pairwise deletion. **means** does not require a complete set of data for each observation. This procedure deals separately with each variable in the matrix, computing descriptive statistics for that variable on the basis of all cases for which there is data. With pairwise deletion, missing values are excluded from computation, so the number of cases used in calculating descriptive statistics for each variable differs from variable to variable.

Default = 0.

ds.**output** scalar, determines printing of intermediate results.

0 nothing is written.

1 serial ASCII output format suitable for disk files or printers.

Default = 1.

ds.**range** 2×1 vector, the range of records in data set used for analysis. The first element is the starting row index, the second element is the ending row index. Default is *ds*.**range** = { 0, 0 }, the whole data set. For example, if one wants the range of data from row 100 to the end of data, then *ds*.**range** should be set as:

```
ds.range = { 100, 0 };
```

ds.**reporttyp** scalar, specifies type of report to print.

1 A single table is generated with the included statistics as column headings and the variable names as row headings.

2 An individual report is generated for each variable included.

Default = 1.

ds.**rowfac** scalar, "row factor". If **means** fails due to insufficient memory while attempting to read a **GAUSS** data set, then *ds*.**rowfac** may be set to some value between 0 and 1 to read a *proportion* of the original number of rows of the **GAUSS** data set. For example, setting

```
ds.rowfac = 0.8;
```

causes **GAUSS** to read in only 80% of the rows that were originally calculated.

This global has an effect only when *ds*.**rowsperiter** = 0.

Default = 1.

ds.**rowsperiter** scalar, specifies how many rows of the data set are read per iteration of the read loop. If *ds*.**rowsperiter** = 0, the number of rows to be read is calculated by **means**. Default = 0.

ds.**sort** scalar, if 1, output is sorted by the names of the variables in *vars*. Default = 0.

ds.**statlist** M×1 vector, where 1 <= *M* <= 6. If 0, all univariate descriptive statistics are included in report. Otherwise only the specified statistics are included, in the order in which they appear in *ds*.**statlist**.

Available statistics are:

1	Means
2	Standard Deviation
3	Minimum
4	Maximum
11	Number Valid
12	Number Missing

Default = 0.

ds.**title** string, this is the title used by *ds*.**header**. Default = "".

ds.**weight** string, name of weight variable. By default, unweighted correlations are calculated.

ds.**weightindex** scalar, index of weight variable. *ds*.**weightindex** overrides *ds*.**weight**. By default, unweighted correlations are calculated.

dataset string, name of data file.

vars K×1 string array, names of variables.
– or –
K×1 numeric vector, indices of variables.

If *vars* = 0, all variables are included.

Output

nms K×1 string array of variable names.

mn K×1 vector of means.

std K×1 vector of standard deviations.

min K×1 vector of minimum values.

max K×1 vector of maximum values.

valid K×1 vector of the number of valid cases for each selected variable.

missing K×1 vector of the number of missing cases in each selected variable.

Error handling is controlled by the low order bit of the trap flag.

TRAP 0 terminate with error message.

TRAP 1 return scalar error code in all return arguments.

The function **SCALERR** can be used to return the value of the error code. The error codes returned are:

1 Data file not found

2 Undefined variables in input argument

3 Index out of range

8 This function does not compute one or more of the statistics specified

9 Weight variable must be numeric

77 No cases left after deleting missing observations

Remarks

This procedure handles character, numeric, and date data.

Example

```
library dstatmt;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;

dsn = "cook";
output file = means.out reset;
ds.statlist = { 11,12,1,3,4,2 };
{ nms,mn,std,min,max,valid,missing } = means(ds,dsn,0);
output off;
```

Source

destatmt.src

Purpose

Computes the descriptive statistics for variables in a **GAUSS** data set, including skew and kurtosis.

Library

`dstatmt`

Format

`{` *nms*,*mn*,*std*,*min*,*max*,*skewc*,*skewp*,*kurtc*,*kurtp*,*combc*,*combp*,*valid*,*missing* `}` =
meanssk(*ds*,*dataset*,*vars***);**

Input

ds **dstatControl** structure. The following elements of *ds* are referenced within the **meanssk** routine:

ds.**header** string, specifies the format for the output header. *ds*.**header** can contain zero or more of the following characters:

t	print title (see *ds*.**title**)
l	bracket title with lines
d	print date and time
v	print procedure name and version number
f	print file name being analyzed

Example:

```
ds.header = "tld";
```

If *ds*.**header** == "", no header is printed. Default = "tldvf".

ds.**miss** scalar, determines how missing data are handled.

0	Missing values are not checked for, so the data set must not have any missing observations. This is the fastest option.
1	Listwise deletion. Removes from computation any observation with a missing value for any variable included in the analysis.
2	Pairwise deletion. **meanssk** does not require a complete set of data for each observation. This procedure deals separately with each variable in the matrix, computing descriptive statistics for that variable on the basis of all cases for which there is data. With pairwise deletion, missing values are excluded from computation, so the number of cases used in calculating the descriptive statistics for each variable differs from variable to variable.

Listwise deletion is used for computation of multivariate skew and kurtosis.
Default = 0.

ds.**output** scalar, determines printing of intermediate results.

0 nothing is written.

1 serial ASCII output format suitable for disk files or printers.

Default = 1.

ds.**range** 2×1 vector, the range of records in data set used for analysis. The first element is the starting row index, the second element is the ending row index. Default is *ds*.**range** = { 0, 0 }, the whole data set. For example, if one wants the range of data from row 100 to the end of data, then *ds*.**range** should be set as:

```
ds.range = { 100, 0 };
```

ds.**reporttyp** scalar, specifies type of report to print.

1 A single table is generated with the included statistics as column headings and the variable names as row headings.

2 An individual report is generated for each variable included.

Default = 1.

ds.**rowfac** scalar, "row factor". If **meanssk** fails due to insufficient memory while attempting to read a **GAUSS** data set, then *ds*.**rowfac** may be set to some value between 0 and 1 to read a *proportion* of the original number of rows of the **GAUSS** data set. For example, setting

```
ds.rowfac = 0.8;
```

causes **GAUSS** to read in only 80% of the rows that were originally calculated.

This global has an effect only when *ds*.**rowsperiter** = 0.

Default = 1.

ds.**rowsperiter** scalar, specifies how many rows of the data set are read per iteration of the read loop. If *ds*.**rowsperiter** = 0, the number of rows to be read is calculated by **meanssk**. Default = 0.

ds.**sort** scalar, if 1, output is sorted by the names of the variables in *vars*. Default = 0.

ds.**statlist** M×1 vector, where 1 <= *M* <= 12. If 0, all univariate descriptive statistics are included in report. Otherwise only the specified statistics are included, in the order in which they appear in *ds*.**statlist**.

Available statistics are:

1	Means
2	Standard Deviation
3	Minimum
4	Maximum
5	Skew Coefficient
6	Skew Probability
7	Kurtosis Coefficient
8	Kurtosis Probability
9	Combined Coefficient
10	Combined Probability
11	Number Valid
12	Number Missing

Default = 0.

ds.**title** string, this is the title used by *ds*.**header**. Default = "".

ds.**weight** string, name of weight variable. By default, unweighted correlations are calculated.

ds.**weightindex** scalar, index of weight variable. *ds*.**weightindex** overrides *ds*.**weight**. By default, unweighted correlations are calculated.

dataset string, name of data file.

vars K×1 string array, names of variables.
– or –
K×1 numeric vector, indices of variables.

If *vars* = 0, all variables are included.

Output

nms K×1 string array of variable names.

mn K×1 vector of means.

std K×1 vector of standard deviations.

min K×1 vector of minimum values.

max K×1 vector of maximum values.

skewc K×1 vector of skew coefficients (unit normal distribution).

skewp K×1 vector of probabilities of *skewc* being that large or larger if zero in pop.

kurtc K×1 vector of kurtosis coefficients (unit normal distribution).

kurtp K×1 vector of probabilities of *kurtc* being that large or larger if zero in pop.

combc K×1 vector of summary statistics (chi-squared dist. with 2 df)

skewp K×1 vector of probabilities of *combc*

valid K×1 vector of the number of valid cases for each selected variable.

missing K×1 vector of the number of missing cases in each selected variable.

Error handling is controlled by the low order bit of the trap flag.

TRAP 0 terminate with error message.
TRAP 1 return scalar error code in all return arguments.

The function **SCALERR** can be used to return the value of the error code. The error codes returned are:

1 Data file not found
2 Undefined variables in input argument
3 Index out of range
9 Weight variable must be numeric
77 No cases left after deleting missing observations

Remarks

This procedure handles character, numeric, and date data. Skew and kurtosis computations are performed only on numeric variables.

Multivariate skew and kurtosis computations are included in the report generated by **meanssk** if skew and kurtosis are specified in *ds*.**statlist**. However, **meanssk** returns only *univariate* skew and kurtosis coefficients and probabilities. Use **mardia** for multivariate skew and kurtosis.

Example

```
library dstatmt;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;

dsn = "cook";
ds.reporttyp = 2;
ds.statlist = { 3,4,1,2,5,6,7,8,11,12 };
output file = meanssk.out reset;
{ nms,mn,std,min,max,valid,missing } = meanssk(ds,dsn,0);
output off;
```

Source

destatmt.src

Purpose

Computes statistics and measures of association for an I×J contingency table.

Library

`dstatmt`

Format

tblstat(x**);**

Input

x	I×J matrix of cell frequencies.

Output

Measures of fit and association for table are sent to the output device.

Remarks

The following statistics are computed and printed.

Statistic	Reference
Pearson's Chi Square	BFH, 124
Likelihood Chi Square	BFH, 124
Yate's Corrected Chi Square (2×2)	BFH, 124
McNemar's Symmetry Chi-Square	
Phi	Agresti, 175
Cramer's V (not for 2×2)	BFH, 386
Contingency (Pearson's P)	BFH, 385
Spearman's Rho (Correlation)	BFH, 381
Cohen's Kappa (symmetric tables)	BFH, 395
Yule's Q (2×2)	BFH, 378
Yule's Y (2×2)	BFH, 378
Goodman-Kruskal Gamma	Agresti, 159
Kendall's Tau-B	Agresti, 161
Stuart's Tau-C	Agresti, 177
Somer's D	Agresti, 161
Lambda	BFH, 388
Uncertainty	

Agresti: Agresti, Alan. 1984. *Analysis of Ordinal Categorical Data*. New York: John Wiley and Sons.

BFH: Bishop, Yvonne, Stephen Fienberg and Paul Holland 1975. *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, Mass.: MIT Press.

Example

```
library dstatmt;

x = { 10 23,
      34 47 };
tblstat(x);
```

Source

tblstatmt.src

See also

crosstab

Purpose

Tests the differences of means between two groups.

Library

`dstatmt`

Format

`{` *varl*,*descstat*,*mntest*,*vartest* `}` = **ttest(***ds*,*dataset*,*grpvar*,*varnm***);**

Input

ds **dstatControl** structure. The following elements of *ds* are referenced within the **meanssk** routine:

ds.**cut** scalar or 2×1 vector.
The groups are defined by *ds*.**cut** and the conditioning variable *grpvar* as follows:
scalar *ds*.**cut**, numeric *grpvar*:

first group	$grpvar < ds.\mathbf{cut}$
second group	$grpvar \geq ds.\mathbf{cut}$

scalar *ds*.**cut**, character *grpvar*:

first group	$grpvar = ds.\mathbf{cut}$
second group	$grpvar \neq ds.\mathbf{cut}$

vector (2×1) *ds*.**cut**, character or numeric *grpvar*:

first group	$grpvar = ds.\mathbf{cut}[1]$
second group	$grpvar = ds.\mathbf{cut}[2]$

Default = 0.

ds.**grpnm** 2×1 character vector of names of groups. By default, the names "Group 0" and "Group 1" are used.

ds.**header** string, specifies the format for the output header. *ds*.**header** can contain zero or more of the following characters:

t	print title (see *ds*.**title**)
l	bracket title with lines
d	print date and time
v	print procedure name and version number
f	print file name being analyzed

Example:

```
ds.header = "tld";
```

If *ds*.**header** == "", no header is printed. Default = "tldvf".

ds.**miss** scalar, determines how missing data are handled.

0 Missing values are not checked for, and so the data set must not have any missing observations. This is the fastest option.

1 Listwise deletion. Removes from computation any observation containing a missing value for any variable included in the analysis.

2 Pairwise deletion. **ttest** does not require a complete set of data for each observation. This procedure deals separately with each pair of variables in the matrix, computing the covariance and correlation between that pair on the basis of all cases for which there is data. With pairwise deletion, if any pair of variables contains missing values in an observation, that observation is excluded from the computation of their covariance.

Default = 0.

ds.**output** scalar, determines printing of intermediate results.

0 nothing is written.

1 serial ASCII output format suitable for disk files or printers.

Default = 1.

ds.**range** 2×1 vector, the range of records in data set used for analysis. The first element is the starting row index, the second element is the ending row index. Default is *ds*.**range** = { 0, 0 }, the whole data set. For example, if one wants the range of data from row 100 to the end of data, then *ds*.**range** should be set as:

```
ds.range = { 100, 0 };
```

ds.**rowfac** scalar, "row factor". If **ttest** fails due to insufficient memory while attempting to read a **GAUSS** data set, then *ds*.**rowfac** may be set to some value between 0 and 1 to read a *proportion* of the original number of rows of the **GAUSS** data set. For example, setting

```
ds.rowfac = 0.8;
```

causes **GAUSS** to read in only 80% of the rows that were originally calculated.

This global has an effect only when *ds*.**rowsperiter** = 0.

Default = 1.

ds.**rowsperiter** scalar, specifies how many rows of the data set are read per iteration of the read loop. If *ds*.**rowsperiter** = 0, the number of rows to be read is calculated by **ttest**. Default = 0.

ds.**title** string, this is the title used by *ds*.**header**. Default = "".

dataset string, name of data file.

grpvar string, name of the group variable.
– or –
scalar, index of the group variable.

grpvar may be the name of a variable that contains character data.

varnm K×1 string array, names of variables to be tested.
– or –
K×1 numeric vector, indices of variables to be tested.
– or –
scalar 0, all variables but *grpvar* are tested.

Output

varl (K+1)×1 string array, variable names.

descstat L×6 matrix of descriptive statistics, where

descstat[**.,1**] means of group 0
descstat[**.,2**] means of group 1
descstat[**.,3**] standard deviation of group 0
descstat[**.,4**] standard deviation of group 1
descstat[**.,5**] number of valid cases
descstat[**.,6**] number of missing cases

mntest L×6 matrix, results of test of the hypothesis that the true means are the same. Columns are:

mntest[**.,1**] t value for assumption that the two groups have equal variances
mntest[**.,2**] degrees of freedom for equal variances
mntest[**.,3**] probability for equal variances
mntest[**.,4**] t value for unequal variances
mntest[**.,5**] degrees of freedom for unequal variances
mntest[**.,6**] probability for unequal variances

vartest L×4 matrix of results from test of variances. Columns are:

vartest[**.,1**] F value for test of differences
vartest[**.,2**] degrees of freedom for equal variances
vartest[**.,3**] degrees of freedom for unequal variances
vartest[**.,4**] probability of the F statistic

Error handling is controlled by the low order bit of the trap flag.

TRAP 0 terminate with error message.

TRAP 1 return scalar error code in all return arguments.

The function **SCALERR** can be used to return the value of the error code. The error codes returned are:

1 Data file not found

2 Undefined variables in input argument

3 Index out of range

5 Type mismatch

41 No missing values specified, but missing values were encountered

77 No cases left after deleting missing observations

Remarks

This procedure handles both character and numeric data. Date variables are skipped.

Descriptive statistics for each group and tests of differences of means are sent to the output device.

Example

```
library dstatmt;
#include dstatmt.sdf
struct dstatControl ds;
ds = dstatControlCreate;

dataset = "scigau";
string vars = { cit1, pub1 };
grpvar = "job";
ds.grpnm = { Lo_Job, Hi_Job };
ds.miss = 2;
ds.cut = 2;
output file = ttest.out reset;
{ desc,mtest,vtest } = ttest(ds,dataset,grpvar,vars);
output off;
```

Source

testmt.src

Index