Loglinear Analysis

Documentation Version: January 15, 2001

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. Use `cd` to make the directory containing **GAUSS** the current working directory.
2. Use `tar` to extract the files.

   `tar xvf` *device_name*

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

The following device names are suggestions. See your system administrator. If you are using Solaris 2.x, see Section 1.1.1.

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`		

1.1.1 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 DOS

1. Place the diskette in a floppy drive.
2. Log onto the root directory of the diskette drive. For example:

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

3. 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

4. The screen output option used may require that the DOS screen driver ANSI.SYS be installed on your system. If ANSI.SYS is not already installed on your system, you can put the command like this one in your CONFIG.SYS file:

   DEVICE=C:\DOS\ANSI.SYS

   (This particular statement assumes that the file ANSI.SYS is on the subdirectory DOS; modify as necessary to indicate the location of your copy of ANSI.SYS.)

1.3 Differences Between the UNIX and DOS Versions

- In the DOS version, when the global **__output** = 2, information may be written to the screen using commands requiring the ANSI.SYS screen driver. These are not available in the current UNIX version, and therefore setting **__output** = 2 may have the same effect as setting **__output** = 1.

- 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 DOS 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 DOS.

Chapter 2

Loglinear Analysis

The *LOGLINEAR ANALYSIS* module contains a set of routines for the analysis of categorical data using loglinear analysis. These routines can estimate most of the models considered in such texts as Y.M.M. Bishop, S.E. Fienberg and P.W. Holland (1975) *Discrete Multivariate Analysis*, A. Agresti (1984) *Analysis of Ordinal Categorical Data*, and S. Haberman (1979) *Analysis of Qualitative Data*, Volumes 1 and 2.

Estimation is performed in this module with the procedure **llest** which computes maximum likelihood estimates by the Newton-Raphson method. The other routines are for entering data, constructing design matrices, easily passing information to **llest**, and printing the results. Input tables can be entered with the **GAUSS** editor, from the keyboard using the program **lltable**, or with a table constructed from raw data using the **GAUSS** procedure **crosstab**.

2.1 Getting Started

GAUSS 3.1.0+ is required to use these routines.

2.1.1 README Files

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

2.1.2 Setup

In order to use the procedures in the *LOGLINEAR ANALYSIS* module, the module library must be active. This is done by including `loglin` in the **LIBRARY** statement at

the top of your program:

```
library loglin,quantal,pgraph;
```

This enables **GAUSS** to find the *LOGLINEAR ANALYSIS* procedures. If you plan to make any right-hand references to the global variables (described in section 2.4), you will also need the statement:

```
#include loglin.ext;
```

Finally, to reset global variables in succeeding executions of the program the following instruction can be used:

```
llset;
```

This could be included with the above statements without harm and would insure the proper definition of the global variables for all executions of the program.

The version number of each module is stored in a global variable. For *LOGLINEAR ANALYSIS*, this global variable is:

__ll_ver 3×1 matrix, the first element contains the major version number, the second element the minor version number, and the third element the revision number.

If you call for technical support, you may be asked for the version of your copy of this module.

2.2 The Loglinear Model

This section presents the basic ideas related to the loglinear model. This is done primarily to clarify the notation that will be used below. The reader who is not familiar with these models should consult one of the texts given in the references.

Consider a K-way table constructed from variables $V_i(i = 1, K)$, with variable V_i having K_i categories. The resulting table contains $N = K_1 * K_2 * \ldots K_K$ cells. The observed variables for this table are contained in the Nx1 vector $\boldsymbol{n}$. Let the N cell indices associated with n be contained in the KxN matrix T where element t_{ij} is the category level of variable V_j for n_i. For example, for a 2x2x2 table ($K = 3$):

	A=1			A=2	
	C=1	C=2		C=1	C=2
B=1	1	2	B=1	5	6
B=2	3	4	B=2	7	8

$\boldsymbol{n}$ and $\boldsymbol{T}$ would be:

$$\boldsymbol{n} = \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{matrix} \qquad \boldsymbol{T} = \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 1 \\ 1 & 2 & 2 \\ 2 & 1 & 1 \\ 2 & 1 & 2 \\ 2 & 2 & 1 \\ 2 & 2 & 2 \end{matrix}$$

Let $\boldsymbol{m}$ be a Nx1 vector of expected cell frequencies for the loglinear model. The loglinear model is defined as:

$$\log \boldsymbol{m} = \boldsymbol{D}\boldsymbol{\beta}$$

for a NxP design matrix $\boldsymbol{D}$ with P parameters in the Px1 vector $\boldsymbol{\beta}$.

Estimation is based on the assumption that the n_i are independent Poisson random variables with expected values equal to m_i. $\boldsymbol{\beta}$ is estimated by the Newton-Raphson method, based on an algorithm presented in Agresti (1984).

The loglinear model is overparameterized, which means that constraints must be imposed on the parameters. Different programs use different constraints. Most commonly, parameters are constrained to sum to zero over the level of an individual variable. Alternatively, the parameter associated with the first level of any variable can be fixed to zero. **llest** allows either set of constraints.

The specific loglinear model is defined by its design matrix. The *LOGLINEAR ANALYSIS* module contains a number of procedures for automatically constructing common kinds of design matrices. Alternatively, the user can construct his or her own design matrix to pass to **llest**.

An important class of models is called hierarchical models. These models are uniquely specified by the configurations that are fit by the model. A configuration is defined as a table formed from the original table by collapsing over dimensions of the table. For example, V_1V_3 would be the table formed from $V_1V_2 \ldots V_K$ by collapsing over V_2, V_4, ..., V_K. For purposes of describing hierarchical models in the procedures described below, variables will be referred to by number (e.g., 1, 2) and configurations will be defined by integers containing the variables in the table (e.g., the configuration for variables 1 and 3 is indicated as 13).

Tables with cell weights can also be analyzed. Thus, the model being estimated is somewhat more general than indicated above. Specifically, the model with weights is:

$$\log(m_i/z_i) = \beta_k X_{ik}$$

where z_i is the cell weight associated with cell i. See Haberman (1979) for details on this form of the model. Clogg and Eliason (1988) provide a number of examples of using cell weights.

In its simplest form, weighting allows the analysis of incomplete tables. Incomplete tables are tables in which some of the cells are to be excluded from analysis. These cells are considered to have structural zeros. To indicate which cells have structural zeros, an Nx1 vector Z is constructed with a 1 for cells to be analyzed and a 0 for cells to be dropped from analysis. The example files `llquasi.e` and `llrate.e` provide examples of weighting.

2.3 General Design

The *LOGLINEAR ANALYSIS* module consists of a set of main procedures, auxiliary procedures called by the main procedures, and global variables controlling aspects of estimation and output. The main procedures are:

lletters	Set global variable **_lllettrs** to a character vector of labels.
ll3way	Estimate all 3-way hierarchical models for a table.
lldesign	Input a design matrix for use with **llest**.
llest	Compute maximum likelihood estimates for a loglinear model.
llhier	Estimate a hierarchical model given its fit configurations.
llout	Print the estimates for a loglinear model.
lltable	Input a table for use with **llest**.
lluser	Estimate a model with a user-supplied design.

llest is the central routine that estimates a loglinear model given a specific table, a set of weights, and a design matrix. **llest** will make efforts to fix an improper design matrix. For example, columns with all zeros are deleted, and a warning is printed. Further, if singularities are encountered that cannot be eliminated, the offending design matrix is printed along with an explanation. The degrees of freedom for the model are computed with methods suggested by Clogg and Eliason (1988). Essentially, the rank of the design matrix is subtracted from the number of cells without either structural or fitted zeros.

To see how **llest** is used, consider the hierarchical model 12,3 for the 2x2x2 table 123. Assume that all cells are to be analyzed. The loglinear model could be estimated as:

```
n = { 32, 86, 11, 35, 6, 73, 41, 70 };
d = { 1   1   1   1   1   1,
      1   1   1  -1   1  -1,
      1   1  -1   1  -1   1,
```

```
        1   1  -1  -1  -1  -1,
        1  -1   1   1  -1  -1,
        1  -1   1  -1  -1   1,
        1  -1  -1   1   1  -1,
        1  -1  -1  -1   1   1 };

{ m,b,v,gsq,df,tol } = llest(n,0,d);
```

llest returns various useful pieces of information that are discussed below. However, the procedure **llout** translates this information into a convenient form. Thus, readable results could be obtained by adding the matrix of cell indices and calling **llout**:

```
t = { 1  1  1,
      1  1  2,
      1  2  1,
      1  2  2,
      2  1  1,
      2  1  2,
      2  2  1,
      2  2  2 };
llout(t,m,b,vc,gsq,df,tol,n,0,d,0,0,0);
```

In general, it is tedious and error-prone to enter design matrices and tables. The procedure **lltable** prompts the user for the number of variables, the number of categories per variable, and the value for each cell. The user is given the option of specifying cell weights. The user can provide a name of up to seven letters (e.g., "myname") which the program will use to place the table indices, cell counts and weights in global matrices (here, **mynamet**, **mynamen** and **mynamez**). If desired, these matrices will be saved to disk.

The procedure **lldesign** prompts the user for the number of rows and columns in the design matrix and prompts for each element of the design matrix. The rows correspond to cells in the table and the columns correspond to parameters. The user can provide a name of up to seven letters (e.g., "myname") which the program will use to place the design matrix in a global matrix (here, **mynamed**). If desired, the matrix will be saved to disk.

If the design matrix, table, and cell indices have been entered, the procedure **lluser** can be used to pass the table indices, cell counts and user-defined design matrix to **llest** and print the results with **llout**. For example:

```
library loglin;
llset;
loadm myn, myt, myd;
call lluser(myt,myn,0,myd,0,0);
```

Having complete control over the design matrix is useful for many specialized models. Consider the row effects model presented on page 87 of Agresti (1985). Assume that the cell counts and table indices are saved in *agr87t* and *agr87n*. Then the row effects model could be estimated as:

```
/* 1 */    /* lluser.e:  Agresti, page 87 */
           library loglin;
           llset;
/* 2 */    loadm agr87t,agr87n;
/* 3 */    agr87d =
            { 1  1  0  1  0 -1  0,
              1  1  0  0  1  0  0,
              1  1  0 -1 -1  1  0,
              1  0  1  1  0  0 -1,
              1  0  1  0  1  0  0,
              1  0  1 -1 -1  0  1,
              1 -1 -1  1  0  1  1,
              1 -1 -1  0  1  0  0,
              1 -1 -1 -1 -1 -1 -1 };
/* 4 */     save agr87d;
/* 5 */     cls;
/* 6 */     output file = "lluser1.out" reset;
/* 7 */     print "lluser.e:  Row Effects Model from Agresti, page 87.";
            print;
/* 8 */     call lluser(agr87t,agr87n,0,agr87d,0,0);
/* 9 */     output off;
```

The numbers to the left are for reference purposes only.

Line (1) labels the program.

Line (2) loads the cell frequencies and indices.

Line (3) inputs the design matrix.

Line (4) saves the design matrix for later use.

Line (5) clears the screen.

Line (6) creates the output file.

Line (7) labels the output.

Line (8) calls **llest** and **llout** and thus estimates the model.

Line (9) turns off the output.

This program is saved as `lluser.e` in the `examples` subdirectory.

While manually entering the design matrix provides a great deal of flexibility, it is tedious and error-prone. For hierarchical models, there are two procedures that do most of the work:

ll3way computes all possible hierarchical models for a three-way table. All the user must provide is the cell indices and the table. For example, assuming that t and n are in memory:

```
ll3way(t,n,z);
```

would estimate all models for the table with indices t, counts n, and weights contained in z.

llhier estimates the hierarchical model described by the configurations that are to be fit. For example,

```
/* llhier1.e:  Fienberg, page 27. */
library loglin;
llset;
loadm fien27t,fien27n;
print;
output file = "llhier1.out" reset;
print "llhier1.e:  Fienberg, page 27.";
print;
print "Model 1:  Conditional independence.";
print;
cfg = { 13, 12 };
nm = { Fien3, Fien2, Fien1 };
call llhier(fien27t,fien27n,0,cfg,nm);
print;
print "Model 2:  Complete independence.";
print;
cfg = { 1, 2, 3 };
call llhier(fien27t,fien27n,0,cfg,nm);
output off;
```

would estimate two hierarchical models for the table stored in *fien27t* and *fien27n*. This program is saved as `llhier1.e` in the `examples` subdirectory.

letters Sets global variable **_llettrs** to a character vector of labels. For example:

```
letters("S X U V W")
```

would set **_llettrs** to a vector with S, X, U, V and W in it. With **letters** it is possible to specify models with letters rather than numbers. For example: SU, VW rather than 13, 45.

The procedures **ll3way**, **llhier** and **lluser** illustrate the ways in which **llest** and **llout** can be used to construct customized applications for loglinear modeling.

llhier constructs a design matrix for hierarchical models, passes it to **llest** for estimation, and passes the results to **llout** for printing.

ll3way loops through all hierarchical models for three-way tables, passes the design matrix to **llest**, and prints the goodness of fit of each model.

lluser is a very simple procedure that takes information on a table and design matrix and passes them to **llest**, and then passes the results to **llout**.

Taking these three procedures as models, one could easily create a specialized procedure for logit type models, models for ordered data, and so on.

2.4 Global Control Variables

The following global variables are used in *LOGLINEAR ANALYSIS* to control various options. Chapter 3 provides details on which globals affect which procedures. The defaults can be changed by modifying the **DECLARE** statements in `loglinear.dec` . Alternatively, the defaults can be changed by setting new values immediately before using the *LOGLINEAR ANALYSIS* procedures.

_llmaxit scalar, indicates the maximum number of iterations. If convergence is not reached within 25 iterations, it generally means that there is a problem with the model. When **_llmaxit** is exceeded, a warning is printed along with the results. These results are to help you assess the problem; they are NOT correct. Default = 25.

_llettrs Kx1 character vector, contains the characters that will be used as mnemonics for labeling the output. The first element is associated with the first variable, the second with the second, etc. The abbreviations supplied in **_llettrs** are used to label the output in **llout**. By default, A, B, ... are used as abbreviations for variables.

_llmult scalar. Values are:

- **0** print coefficients for the loglinear model.
- **1** print coefficients for the multiplicative model. The parameters in the multiplicative models are simply the exponential of the parameters in the loglinear model.

Default = 0.

_llprtd scalar. If 1, print the design matrix. Default = 1.

_llprtit scalar. If 1, print the information on iterations. Default = 1.

_llprtt scalar. If 1, print the table of cell counts, expected values, residuals and weights. Default = 1.

_llprte scalar. If 0, the estimates of the parameters are not printed. This is useful if you just want the chi-squares or expected cell frequencies. Default = 1.

_llsum0 scalar. Values are (default = 1):

- **0** parameters for first level of any variable are constrained to zero.
- **1** parameters in the hierarchical model are constrained to sum to 0 across the level of any variable.

_lltol scalar. Sets tolerance for convergence. Tolerance is defined as the maximum difference between the coefficients of the current iteration and the previous one. Default = 1e−5.

_lltwo scalar. Values are (default = 0):

- **0** do not double or square the coefficients. This is useful if logit models are being estimated.
- **1** print two times the additive coefficients or the squared multiplicative coefficients.

2.5 References

Agresti, A. 1984. *Analysis of Ordinal Categorical Data.* New York:Wiley.

Bishop, Y.M.M, S.E. Fienberg and P.W. Holland (1975) *Discrete Multivariate Analysis.* Boston:MIT Press.

Clogg, C.C. and S.R. Eliason. 1988. "Some common problems in log-linear analysis." Pp. 226-257 in Long, *Common Problems/Proper Solutions.* Beverly Hills:Sage University Press.

Fienberg, S.E. 1980. *The Analysis of Cross-Classified Categorical Data.* Boston:MIT Press.

Chapter 3

Loglinear Analysis Reference

Library

`loglin`

Purpose

Sets global variable **_llettrs** to a character vector of labels.

Format

letters(*str***);**

Input

str	string, containing a list of one character variables.

Output

None.

Global Output

_llettrs	Kx1 character vector, contains the characters that will be used as mnemonics for labeling the output. The first element is associated with the first variable, the second with the second, etc. The abbreviations supplied in **_llettrs** are used to label the output in **llout**. By default, A, B, ... are used as abbreviations for variables.

Remarks

For example:

```
letters("S X U V W")
```

would set **_llettrs** to a vector with S, X, U, V and W in it. With **letters** it is possible to specify models with letters rather than numbers. For example: SU, VW rather than 13, 45.

Example

```
library loglin;
#include loglin.ext;
llset;
loadm fien27t,fien27n;
letters("S D H");
cfg = { SH, SD };
nm = { SPECIES, DIAMETER, HEIGHT };
call llhier(fien27t,fien27n,0,cfg,nm);
```

Source

`llappl.src`

See also

llhier

Library

`loglin`

Purpose

Estimates all hierarchical models for a three-way table.

Format

ll3way(t,n,z**);**

Input

t	NxP matrix of cell indices, where N equals **ROWS(**n**)** and P equals the rows of the coefficient matrix.
n	Nx1 matrix of cell counts of table.
z	Nx1 matrix with cell weights.

Output

None.

Globals

_llmaxit	scalar, indicates the maximum number of iterations. If convergence is not reached within 25 iterations, it generally means that there is a problem with the model. When **_llmaxit** is exceeded, a warning is printed along with the results. These results are to help you assess the problem; they are NOT correct. Default = 25.
_llprtit	scalar. If 1, print information on iterations. For this procedure, it is best to set **_llprtit** to 0. Default = 1.
_lltol	scalar. Sets tolerance for convergence. Tolerance is defined as the maximum difference between the coefficients of the current iteration and the previous one. Default = 1e−5.
__output	scalar. If nonzero, intermediate results are printed. Under UNIX, default = 1; under DOS, default = 2.

Remarks

Likelihood ratio and Pearson chi-squares for all seventeen possible loglinear models for three-way tables are printed.

The procedures **_llhdsgn**, **_lldigit** and **_llsubcn** are called by this procedure. These are documented in the file `llappl.src`.

Example

```
library loglin;
llset;
loadm fien27t,fien27n;
_llprtit = 0;
print "ll3way.e:  See Fienberg, page 42.";
ll3way(fien27t,fien27n,0);
```

Source

`llappl.src`

See also

llest, **lltable**, **lldesign**, **llout**, **llhier**, **lluser**

Library

`loglin`

Purpose

Interactively input a design matrix for analysis with **llest**.

Format

lldesign;

Input

The user is prompted for the number of rows and columns in the design matrix. The rows correspond to the cells of the table, the columns to the parameters to be estimated. The user is then prompted to enter elements of the design matrix. Finally, a name of up to seven characters may be given to name the matrix.

Output

None.

Remarks

Assume that *name* is the user-supplied name to specify the matrices. **lldesign** then creates the global design matrix *name***d**.

The user is given the option to save the outputs to disk.

Example

```
library loglin;
llset;
lltable;
/* respond to prompts from lltable to create myt, myz and myn */
lldesign;
/* respond to prompts from lldesign to create myd */
call lluser(myt,myn,myz,myd,0,0);
```

Source

`llinput.src`

See also

llest, **lltable**, **llout**, **llhier**, **ll3way**, **lluser**

Library

`loglin`

Purpose

Estimates a loglinear model by maximum likelihood.

Format

{ *m*,*b*,*vc*,*gsq*,*df*,*tol* } = **llest**(*n*,*z*,*d*);

Input

n	Nx1 matrix of cell counts of table, where N is the number of cells.
z	Nx1 matrix with cell weights.
d	NxP design matrix, where P is the number of parameters to be estimated. The rows of *d* correspond to the cells in *n*.

Output

m	Nx1 vector of expected cell frequencies, in the same order as *n*.
b	Px1 vector of parameter estimates.
vc	PxP covariance matrix of *b*.
gsq	scalar, likelihood ratio chi-square.
df	scalar, degrees of freedom associated with the model. The degrees of freedom are computed as the difference between the number of nonzero fitted cells and the rank of the design matrix.
tol	scalar, the actual tolerance at convergence.

Globals

_llmaxit	scalar, indicates the maximum number of iterations. If convergence is not reached within 25 iterations, it generally means that there is a problem with the model. When **_llmaxit** is exceeded, a warning is printed along with the results. These results are to help you assess the problem; they are NOT correct. Default = 25.
_llprtit	scalar. If 1, print information on iterations. Default = 1.

_lltol scalar. Sets tolerance for convergence. Tolerance is defined as the maximum difference between the coefficients of the current iteration and the previous one. Default = 1e−5.

Remarks

Maximum likelihood estimation is performed by the Newton-Raphson method. The program is based on a SAS program written by Agresti (1984).

Source

`llest.src`

See also

lltable, **lldesign**, **llout**, **llhier**, **ll3way**, **lluser**

Library

`loglin`

Purpose

Estimates a loglinear hierarchical model defined by a set of configurations to be fit.

Format

{ *m*,*b*,*covb*,*gsq*,*df*,*tol*,*dsgn* } = **llhier**(*t*,*n*,*z*,*cfg*,*nm*);

Input

t	NxP matrix of cell indices.
n	Nx1 matrix of cell counts of table.
z	Nx1 matrix of cell weights.
cfg	Px1 character vector of character labels – or – Px1 numeric vector of variable indices of the configurations to be fit.
nm	Qx1 character vector of variable names, or 0 for no names.

Output

m	Nx1 vector of expected cell frequencies, in the same order as *n*.
b	Px1 vector of parameter estimates.
covb	PxP covariance matrix of *b*.
gsq	scalar, likelihood ratio chi-square.
df	scalar, degrees of freedom associated with the model. The degrees of freedom are computed as the difference between the number of nonzero fitted cells and the rank of the design matrix.
tol	scalar, the actual tolerance at convergence.
dsgn	scalar, the design matrix that was estimated.

Globals

_llmaxit scalar, indicates the maximum number of iterations. If convergence is not reached within 25 iterations, it generally means that there is a problem with the model. When **_llmaxit** is exceeded, a warning is printed along with the results. These results are to help you assess the problem; they are NOT correct. Default = 25.

_llettrs Kx1 character vector, contains the characters that will be used as mnemonics for labeling the output. The first element is associated with the first variable, the second with the second, etc. The abbreviations supplied in **_llettrs** are used to label the output in **llout**. By default, A, B, ... are used as abbreviations for variables.

_llmult scalar. Values are:

0 print coefficients for the loglinear model.

1 print coefficients for the multiplicative model. The parameters in the multiplicative models are simply the exponential of the parameters in the loglinear model.

Default = 0.

_llprtd scalar. If 1, print the design matrix. Default = 1.

_llprte scalar. If 0, the estimates of the parameters are not printed. Default = 1.

_llprtit scalar. If 1, print the information on iterations. Default = 1.

_llprtt scalar. If 1, print the table of cell counts, expected values, residuals and weights. Default = 1.

_llsum0 scalar. Values are (default = 1):

0 parameters for first level of any variable are constrained to zero.

1 parameters in the hierarchical model are constrained to sum to 0 across the level of any variable.

_lltol scalar. Sets tolerance for convergence. Tolerance is defined as the maximum difference between the coefficients of the current iteration and the previous one. Default = 1e−5.

_lltwo scalar. Values are (default = 0):

1 print two times the additive coefficients or the squared multiplicative coefficients.

0 do not double or square the coefficients. This is useful if logit models are being estimated.

__output scalar. If nonzero, intermediate results are printed. Under UNIX, default = 1; under DOS, default = 2.

Remarks

Results of the estimation are also sent to the output device.

The procedures **_llhdsgn**, **_lldigit**, and **_llsubcn** are called by this procedure. These are documented in the file `llappl.src` .

Example

```
library loglin;
llset;
cfg = { 13, 12 };
nm = { VAR_A, VAR_B, VAR_C };
call llhier(t,n,0,cfg,nm);
```

Source

`llappl.src`

See also

lltable, **lldesign**, **llout**, **llhier**, **ll3way**, **lluser**

Library

`loglin`

Purpose

Prints the results obtained from **llest**.

Format

llout(t,m,b,vc,gsq,df,tol,n,z,d,$xcfg$,nm,lbl**);**

Input

t	NxP matrix of cell indices.
m	Nx1 vector of expected cell counts.
b	Px1 vector of parameter estimates.
vc	PxP covariance matrix of estimates in b.
gsq	scalar, likelihood ratio chi-square.
df	scalar, degrees of freedom associated with the model. The degrees of freedom are computed as the difference between the number of nonzero fitted cells and the rank of the design matrix.
tol	scalar, the actual tolerance at convergence. If this is not less than **_lltol**, the estimates did not converge.
n	Nx1 vector of cell counts of table.
z	Nx1 vector of cell weights.
d	NxP design matrix, where P is the number of parameters to be estimated. The rows of d correspond to the cells in n.
$xcfg$	scalar, configurations fit in hierarchical model. If the model is not hierarchical or you do not want the configurations printed, set $xcfg = 0$.
nm	Nx1 character vector of variable names, or 0 for no names.
lbl	Px1 character vector of labels to be associated with parameters being estimated. If 0, parameters will not be labeled.

Output

Results are sent to the output device.

Globals

_llmult scalar. Values are:

- **0** print coefficients for the loglinear model.
- **1** print coefficients for the multiplicative model. The parameters in the multiplicative models are simply the exponential of the parameters in the loglinear model.

Default = 0.

_llprtd scalar. If 1, print the design matrix. Default = 1.

_llprtt scalar. If 1, print the table of cell counts, expected values, residuals and weights. Default = 1.

_lltwo scalar. Values are (default = 0):

- **1** print two times the additive coefficients or the squared multiplicative coefficients.
- **0** do not double or square the coefficients. This is useful if logit models are being estimated.

__output scalar. If nonzero, intermediate results are printed. Under UNIX, default = 1; under DOS, default = 2.

Remarks

This routine may be used to output the results from any use of **llest**.

Example

This example illustrates how **llest** and **llout** can be combined to create routines to simplify the estimation of loglinear models. This procedure is documented as **lluser**.

```
proc (0) = lluser(t,n,z,d,nm,lbl);
   local m,b,vc,gsq,df,tol,xcfg;
   xcfg = 0;
   { m,b,vc,gsq,df,tol } = llest(n,z,d);
   llout(t,m,b,vc,gsq,df,tol,n,z,d,xcfg,nm,lbl);
endp; /* lluser */
```

Source

llest.src

See also

llest, **lltable**, **lldesign**, **llhier**, **ll3way**, **lluser**

Library

`loglin`

Purpose

Resets global variables.

Format

llset;

Input

None.

Output

None.

Remarks

Putting this instruction at the top of all programs that invoke *LOGLINEAR ANALYSIS* procedures is generally good practice. This will prevent globals from being inappropriately defined when a program is run either several times or after another program that also calls *LOGLINEAR ANALYSIS* procedures.

llset calls **gausset**.

Source

`llappl.src`

Library

`loglin`

Purpose

Interactively constructs a table for analysis by **llest**.

Format

lltable;

Input

The user is prompted for the number of variables and the number of categories per variable. The user is then prompted to enter cell counts. Optionally, the user may specify cell weights. Finally, a name of up to seven characters may be given to name the matrices.

Output

Assume that *name* is the user-supplied name to specify the matrices. **lltable** creates the following global matrices:

*name***n**	Nx1 matrix of cell counts of table, where N is the number of cells.
*name***z**	Nx1 matrix of cell weights.
*name***t**	NxP matrix of cell indices associated with *name***n**.

Globals

None.

Remarks

The user is given the option to save the outputs to disk.

Source

`llinput.src`

See also

llest, **lldesign**, **llout**, **llhier**, **ll3way**, **lluser**

Library

loglin

Purpose

Estimates and prints results for a loglinear model based on a user-supplied design matrix.

Format

{ *m*,*b*,*covb*,*gsq*,*df*,*tol* } = **lluser**(*t*,*n*,*z*,*d*,*nm*,*lbl*);

Input

t	NxP matrix of cell indices.
n	Nx1 matrix of cell counts of table.
z	Nx1 matrix of cell weights.
d	NxP design matrix, where P is the number of parameters to be estimated. The rows of *d* correspond to the cells in *n*.
nm	Nx1 vector of variable names, or 0 for no names.
lbl	Px1 vector of parameter names. If 0, parameters will not be given names.

Output

m	Nx1 vector of expected cell frequencies, in same order as *n*.
b	Px1 vector of parameter estimates.
covb	PxP covariance matrix of *b*.
gsq	scalar, likelihood ratio chi-square.
df	scalar, degrees of freedom associated with the model. The degrees of freedom are computed as the difference between the number of nonzero fitted cells and the rank of the design matrix.
tol	scalar, the actual tolerance at convergence.

Globals

_llmaxit scalar, indicates the maximum number of iterations. If convergence is not reached within 25 iterations, it generally means that there is a problem with the model. When **_llmaxit** is exceeded, a warning is printed along with the results. These results are to help you assess the problem; they are NOT correct. Default = 25.

_llmult scalar. Values are:

0 print coefficients for the loglinear model.

1 print coefficients for the multiplicative model. The parameters in the multiplicative models are simply the exponential of the parameters in the loglinear model.

Default = 0.

_llprtd scalar. If 1, print the design matrix. Default = 1.

_llprtit scalar. If 1, print the information on iterations. Default = 1.

_llprtt scalar. If 1, print the table of cell counts, expected values, residuals and weights. Default = 1.

_lltol scalar. Sets tolerance for convergence. Tolerance is defined as the maximum difference between the coefficients of the current iteration and the previous one. Default = 1e−5.

_lltwo scalar. Values are (default = 0):

1 print two times the additive coefficients or the squared multiplicative coefficients.

0 do not double or square the coefficients. This is useful if logit models are being estimated.

__output scalar. If nonzero, intermediate results are printed. Under UNIX, default = 1; under DOS, default = 2.

Remarks

Results of the estimation are also sent to the output device.

Example

This example will estimate the row effect model for ordinal data.

```
library loglin;
llset;
loadm agr87t,agr87n;
agr87d =
  { 1  1  0  1  0 -1  0,
    1  1  0  0  1  0  0,
    1  1  0 -1 -1  1  0,
    1  0  1  1  0  0 -1,
    1  0  1  0  1  0  0,
    1  0  1 -1 -1  0  1,
    1 -1 -1  1  0  1  1,
    1 -1 -1  0  1  0  0,
    1 -1 -1 -1 -1 -1 -1 };
output file = ll1.out reset;
call lluser(agr87t,agr87n,0,agr87d,0,0);
output off;
```

Source

llappl.src

See also

lltable, **lldesign**, **llout**, **llhier**, **ll3way**

Index