1 An Introduction to SAS Procedures for the Analysis of Categorical Data

1.1 proc freq

The `freq` procedure is the basic procedure for the analysis of count data. We use it to construct and analyze contingency tables.

1.2 Analysis of One-Way Tables

Consider the following SAS program for testing goodness of fit for a multinomial distribution with specified cell probabilities:

```
data opinion;
input opinion $ count;
datalines;
favor 42
neutral 61
oppose 33
unaware 14
proc freq data=opinion order=data;
weight count;
tables opinion/nocum testp=(40 30 20 10);
run;
```

The categorical variable is `opinion`. The `weight` statement sets up `count` as the frequency variable for the various categories. The `nocum` option tells SAS not to report the cumulative frequencies. The `testp` option provides the specified cell percentages under the null hypothesis.

The following program will provide an analysis of binomial data:

```
data new;
input request $ count;
cards;
yes 69
no 208
proc freq order=data;
weight count;
table request/binomial;
run;
```

This program will carry out a basic analysis for binomial data (one-way data with two categories). The `binomial` options will result in Wald and exact confidence intervals for a binomial proportion.

1.2.1 Basic Analysis of Two-Way Tables

Consider the following SAS program:

```
data uremic;
input group $ react count;
cards;
uremic 1 38
uremic 2 62
normal 1 21
normal 2 79
proc freq order=data; weight count;
tables group*react/chisq;
run;
```

This program will form a 2×2 table of the frequencies and compute Pearson's chi-squared statistic for independence.

- `order=data` will keep the values of the variables in the table in the same order as in the data set. Otherwise, SAS places everything

in numerical or alphabetical order.

- `weight count` tells SAS that the frequencies for each cell are given by the variable `count`.
- `tables group*react/chisq;` will form a two-way table with rows given by `group` and columns given by `react`. The `chisq` option causes various chi-square statistics to be computed.

1.2.2 Analysis of Stratified 2×2 Tables

Suppose one has a three-way table where one wishes to test for association between treatment and response controlling for center. One can use the following SAS statements for the analysis:

```
proc freq order=data;
   weight count;
   tables center*treatment*response/ nocol nopct chisq cmh relrisk;
```

- The first variable, `center`, determines the stratification variable.
- The second variable, `treatment`, determines the rows.
- The third variable, `response`, determines the columns.
- Two-way tables relating `response` to `treatment` will be formed for each level of `center`.
- The `chisq` option causes chi-squared statistics for each of the two-way tables to be printed. The `cmh` option causes Cochran-Mantel-Haenszel statistics, estimates of the common odds ratio, and the Breslow-Day statistic to be computed. The `relrisk`

options cause estimates of the odds ratio to be computed for each two-way table.

1.2.3 Options for the tables statement

There are numerous options that can be used with the `tables` statement.

- `nopercent`: omit cell percent
- `nocol`: omit column percent
- `norow`: omit row percent
- `expected`: compute expected cell frequencies
- `cellchisq`: computes each cell's contribution to Pearson's statistic, $(O - E)^2/E$
- `chisq`: computes tests of independence; for 2×2 tables, computes Fisher's exact test
- `exact`: computes Fisher's exact test; also invokes `chisq` option
- `fisher`: computes Fisher's exact test; also invokes `chisq` option
- `measures`: compute several measures of association and odds ratios, relative risk, and confidence intervals
- `cl`: provides confidence intervals for the statistics in the `measures` option
- `noprint`: suppresses printing of tables

- `riskdiff`: provides estimates of risk (binomial proportions) and their differences for 2×2 tables
- `relrisk`: provides relative risk measures for 2×2 tables
- `cmh`: computes Cochran-Mantel-Haenszel statistics
- `cmh1`:
- `cmh2`:
- `scores=`*type*
  - table: numerical values if variables are numerical, otherwise, 1,2,3,...
  - rank: the row (column) ranks. Midranks are used for ties.
  - ridit: rank scores divided by sample size.
  - modridit: rank scores divided by sample size plus 1.
- `agree`: computes McNemar's test for 2×2 tables. Also, carries out a test of symmetry for an $I \times I$ table.

1.2.4 The exact statement

The `exact` statement requests exact tests or confidence intervals. Options are used to specify the statistics.

- `chisq`: computes tests of independence
- `or`: odds ratio confidence intervals for 2×2 tables

1.3 proc genmod

The genmod procedure is a general procedure for fitting generalized linear models. We will use genmod to fit Poisson regression, logistic regression, and loglinear models.

1.3.1 Fitting Poisson Regression Models

Consider the following SAS statements:

```
proc genmod data=tornado;
     class month_number;
     model deaths = killer_tornadoes month_number / dist=p link=log;
```

This runs the genmod procedure for the tornado data set.

- The variable deaths is the response for the model.
- The variables killer_tornadoes and month_number are the explanatory variables.
- The class statement indicates that month_number is to be treated as a factor or nominal variable in the analysis.
- The random component is specified by the dist=p option as the Poisson distribution.
- The link function is specified by the link=log option as the log link.
- An offset variable can be specified by an offset=varname option in the model statement.

1.3.2 Fitting Logistic Regression Models

Consider the following SAS statements:

```
proc genmod;
model killed/total=dose/dist=bin link=logit;
```

For binomial data, use `events/trials` syntax to specify the response. The link for logistic regression is the log link. One could also specify the probit or identity link for probit analyis or a linear probability model, respectively.

When the response variable is 0 or 1 for each observation, use the following statements to model $P(Y = 1|x)$:

```
proc genmod desc;
model y=x/dist=bin link=logit;
```

1.3.3 Options for the Model Statement

As seen above, we can use options in the model statement to specify the distribution and link. We can also use options to obtain residuals and diagnostics.

- `obstats`: provides a table of statistics including the response, predicted, estimated linear predicator, standard error of linear predictor, confidence interval for predicted mean, residuals of various types.
- `residuals`: provides a table of residuals of various types.

- `type3`: carries out a likelihood ratio test for each effect specified in the model.
- `aggregate=(variable list)`: specifies the subpopulations for which the Pearson chi-squared statistic and deviance are calculated. The variables in the list can be any in the input data set.

1.3.4 Fitting Loglinear Models

To fit a loglinear model using `genmod`, we actually fit a Poisson regression model. The observed `count` is the response variable, and the loglinear effects appear on the right-hand side of the equal sign. Consider the following SAS statements to use `genmod` to fit a 2×2 table using a loglinear model.

```
proc genmod order=data; class group react;
title 'Fitting the Saturated Model with PROC GENMOD';
model count=group react group*react/dist=poi link=log;

proc genmod order=data; class group react;
ods select type3;
title 'Fitting the Saturated Model with PROC GENMOD with Less Output';
model count=group react group*react/dist=poi link=log type3;

proc genmod order=data; class group react;
title 'Fitting the Independence Model with PROC GENMOD';
model count=group react/dist=poi link=log;
```

The `order=data` phrase in the `proc genmod` statement orders the levels of the variables in the order in which they were input to the data set. The statement `ods select type3;` restricts the output to the table of likelihood ratio statistics.

1.4 proc logistic

The logistic procedure enables one to fit logistic regression models for data with binary outcomes or ordered categorical outcomes. A basic analysis can be performed with the following SAS commands:

```
proc logistic desc;
model y=x1 x2;

proc logistic;
model r/n=x1 x2;
```

The first logistic procedure is used when the response is binary to model $P(Y = 1|x_1, x_2)$. If `desc` is omitted, $P(Y = 0|x_1, x_2)$ is modeled. The second logistic procedure is used for the analysis of grouped data where `r` is the number of successes and `n` is the number of trials. In both cases, `x1` and `x2` are explanatory variables.

1.4.1 Class Statement

A `class` statement can be used for categorical predictors. The `class` statement precedes the `model` statement. The form of the class statement follows:

```
class variable1 <(v-options)> variable2 <(v-options)>/<global options>;
```

Some commonly used options include the following:

- `desc`: reverses the sorting order of the class variable.
- `order=option`: specifies the sorting order for the levels of the class variable. This determines the level to which each parameter corresponds. The values for `option` and the corresponding manner of sorting follow:
  - `data`: order of appearance in data set.
  - `formatted`: external formatted value.
  - `freq`: descending frequency counts.
  - `internal`: unformatted value.

- `param=keyword`: specifies the parameterization method for the class variable. The default is `effect`. The `ref=` command sets the reference category. Some useful keywords follow:
  - `effect`: specifies effect coding.
  - `ref`: specifies reference cell coding.
- `ref=option`: The default is `ref=last`. To specify the first category as the reference category, use `ref=first`. You can also specify a level of the class variable by using `ref='level'`.

1.4.2 freq statement

The variable in the FREQ statement identifies a variable that contains the frequency of occurrence of each observation. Suppose that `count`

is a variable that provides the number of observations. Following is the logistic regression code when there `count` is the frequency associated with the response `malform` and predictor `alcohol`.

```
proc logistic des; freq count;
model malform=alcohol;
```

1.4.3 Options for the Model Statement

- `aggregate=(variable list)` or `aggregate`: specifies the subpopulations for which the Pearson chi-squared statistic and deviance are calculated. Observations with common values in the given list of variables are assumed to come from the same population. The variables in the list can be any in the input data set. If there is no variable list, all the explanatory variables in the `model` statement are assumed to be in the list. The `scale=` option must be specified for this option to have an effect.
- `scale=`: specifies the method of estimating the dispersion parameter. We will typically use `scale=none` which enables calculation of the deviance and Pearson chi-squared statistic without adjusting for overdispersion.
- `corrb`: provides estimated correlation matrix of estimated coefficients.
- `covb`: provides estimated covariance matrix of estimated coefficients.

- `lackfit`: carries out the Hosmer-Lemeshow goodness-of-fit test.
- `influence`: displays diagnostic measures for identifying influential observations.
- `iplots`: produces an index plot for each regression diagnostic statistic.

1.4.4 Output Statement

The `output` statement is used to create a new data set containing all the variables in the original data set plus other variables created by `proc logistic`. The form of the `output` statement follows:

```
output out=dataname <list of options>;
```

Some statistics options that are useful for logistic regression include the following:

- `lower=name`: lower confidence limit for the probability of an event response when `r/n` syntax is used.
- `predicted=name`: predicted probability of an event response when `r/n` syntax is used.
- `upper=name`: upper confidence limit for the probability of an event response when `r/n` syntax is used.
- `reschi=name`: Pearson residual.
- `resdev=name`: deviance residual.

1.5 proc catmod

The following statements will produce an analysis of a 2×2 table using `catmod`.

```
proc catmod; weight count;
title 'Fitting the Saturated Model with PROC CATMOD';
model group*react=_response_/pred=freq; loglin group|react;

proc catmod; weight count;
title 'Fitting the Independence Model with PROC CATMOD';
model group*react=_response_/noresponse noiter noparm noprofile pred=freq
loglin group react;
```

1.5.1 Model Statement

The `model` statement in `catmod` does not directly have a response and require the keywork `_response_` on the right hand side of the equal sign. The response variables `group` and `react` appear on the left-hand-side of the equal sign separated by an asterisk (*). The options `noresponse noiter noparm noprofile` suppress extra printed output that is not always needed for loglinear models.

1.5.2 Loglin Statement

The effects to be included in the loglinear model are specified by the `loglin` statement. The saturated two-variable model will have the statement, `loglin group|react;`, whereas, the independence model will have `loglin group react;`.

