Chapter 3

The SAS macro for the score test

3.1 Introduction to SAS macro

A SAS macro is a set of stored SAS code that is identified by a specified name to do a particular common task. It is defined in SAS by the %macro and %mend statements, such as

```
%MACRO name;
  macro expressions and statements;
%MEND;
```

After a macro is defined, it is invoked by its name after a percent sign

```
%name
```

Macros can have parameters, which appear in parentheses after the macro name. The use of parameters allows greater flexibility of the macro. The syntax of the macro definition and invocation becomes

```
%MACRO name(parameter list);
  macro expressions and statements;
%MEND;
```

and

```
%name(parameter list);
```

There are two kinds of macro parameters: positional parameters and keyword parameters. Positional parameters appear in a fixed position in the parameter list. Keyword parameters follow the last positional parameter. A keyword parameter is written as the parameter name followed by an equal sign and a value. Note that values for keyword parameters can be given in any order once positional parameters come first.

In the %macro definition statement, the names of the parameters are given, and default values for keyword parameters can be given. In the macro invocation, values for a keyword parameter is preceded by the parameter name and an equal sign. Parameters do not have to appear in a macro invocation; ones omitted are given default values. The default value of a positional parameter is a null value, a constant text value of length 0.

For example, the macro presented here is called "homotest". It defines parameters as follows:

```
%macro homotest(data,strata,group,time,censor,factors,outdata=);
  definition text of the macro;
%mend;
```

This macro has 6 positional parameters: `data, strata, group, time, censor, factors`. They are positional parameters because they appear at the beginning of the parameter list and are not followed by an equal sign. `outdata` is a keyword parameter with null default values.

If the macro is invoked by

```
%homotest(mydata, mystr, mygrp, mytime, mycensor, myfactor);
```

then the parameter `data` has the value of `mydata`, parameter `strata` has the value of `mystr`, parameter `group` has the value of `mygrp`, parameter `time` has the value of `mytime`, parameter `mycensor` has the value of `mycensor`, parameter of `factors` has the value of `myfactor`.

if the macro is invoked by

```
%homotest(mydata,mystr,mygrp,mytime,mycensor,myfactor,outdata=myout);
```

then parameter `outdata` has the value of `myout` and other parameters have the same value as the previous example.

In the SAS macro processing, there are two new objects: macro variables(identified by &) and macros(identified by %). Macro parameters are macro variables.

Macro processing is triggered by the characters & and %. When a & is encounter, the macro processor will search its macro variable tables for the value of the variable in the string immediately following. That value is substituted for the reference to macro variable. When a % is encountered, the macro processor will interpret the following characters as a macro you defined or a macro function like `%let` or `%eval`. When a reference to a macro is found, the macro will be run and the result will be

inserted in the place of the reference. When a macro function is called, the function will be executed, and the result substituted for the original reference.

After the macro processor has interpreted the value of a macro variable, a macro, or a macro function, the result will be scanned again until no more references & or % exist. the final result is treated as if they were a part of your original SAS program.

3.2 Parameters of the macro

The definition syntax of the SAS macro in this thesis is

```
%macro homotest(data,strata,group,time,censor,factors,outdata=);
```

Parameter `data` is a SAS dataset name on which the test will be based. This dataset must have at least 4 variables–`time`, `censor`, `covariate`, `factor`. These variables can be in any column order in the dataset.

Parameter `strata` is a variable in the dataset `data`. This variable can be numeric value or alphabetic value. It suggests its effect is not proportional and the Cox regression should be stratified by this variable when doing homogeneity test. However, if you do not have this kind of factor, you just replace `strata` by 0.

Parameter `group` is a variable in the data `data`. This variable can be numeric value or alphabetic value. It classifies subjects into groups and the test is going to test if the random effect of Cox model in these groups are homogeneous. If you want to test overdispersion model against ordinary Cox model, you can just replace `group` by 0. That means every individual subject is in its own one-member group.

Parameter `time` is the variable name of the time to the event in the dataset. It has to be a numeric value.

Parameter `censor` is the variable name of the censoring indicator in the dataset `data`. `censor` can only be 1 or 0, 1 means the event happened, 0 means the subject is censored.

Parameter `factors` are variable names in the dataset `data`. They are covariates in the Cox regression. If you have more than one covariate, you just separate them by

space such as `factor1 factor2 factor3` to replace `factors` in the definition syntax of the macro. Notice that if a covariate is a classification, it may need to be re-coded before using this macro.

Suppose you have a dataset like this

```
data example; input site level time censor factor1 factor2;
   A L 12   0  1 34
   A H 9.5  0  1 45
   B L 10   1  0 50
   B L 12   1  0 38
   ;
```

You can apply the homogeneous test just like:

```
%include 'homotest.macro';
%homotest(example, level, site, time, censor, factor1 factor2);
```

Here, based on dataset `example`, in Cox regression model stratified by `level`, we are testing homogeneity between different `site` of the effect of `factor1 factor2` on time to interested event (`time, censor`).

3.3 Macro description.

The macro presented here can be divided into the following 8 blocks.

Step 1: *line 2-6*

Adjust the original input dataset. If there is no strata variable specified, put all observations into one strata. If there is no group variable specified in the dataset, put each observation in one group. Move `time censor` variables to the last columns of the dataset. Keep only useful variables in the dataset.

Step 1: *line 8-12*

Use SAS procedure for Cox regression (`proc phreg`) to compute $\hat{M}_{ij}$, $\hat{\beta}^t\mathbf{Z}_{ij}$, $\hat{\sigma}_{hk}$, $\hat{H}_{ij}$;

Step 2: *line 14-20*

Merge the output of the Cox regression procedure and original dataset, compute the $\exp(\hat{\beta}^t\mathbf{Z}_{ij})$ and $\hat{H}_0$;

Step 3: *line 24-34*

Enter SAS proc IML, read the dataset prepared in the last step into matrix `x`. Delete missing records and construct strata index;

Step 4: *line 47-87*

Construct group index, death index for the matrix `x`;

Step 5: *line 90-117*

Define module for Y_{ij}, p_{ij}, $I(ij \in h)$, compute $\hat{M}_i$;

Step 6: *line 119-150*

For each strata h, calculate C^h, p_i^h, Q_i^h, $\hat{\theta}_l^h$;

Step 7: *line 158-170*

Sum C^h over strata, get C. Compute score statistic from C and martingale residuals in the matrix **x**. Compute V from p_i^h, Q_i^h, $\hat{\theta}_l^h$;

Step 8: *line 171-186*

Put computation result into dataset and quit IML. Use data step get the test statistic and the p-value.

Appendix

.1 Quick reference

```
%macro(dataset,strata, group,censor,time, factors);
```

`dataset` is the dataset name;

`strata` is the strata variable name in the dataset. If the data is not stratified, put 0;

`group` is the group variable name in the dataset. If there is no group variable in the data, put 0. That means the test will test overdispersion model against ordinary Cox model;

`censor` is the censor variable name in the dataset. 0 means censored and 1 means death;

`time` is the observation time variable name in the dataset;

`factors` is the covariate variable names. For example, if there are age and sex two covariate, put `age sex` there.