SAS Examples of Chapter 2

2.1 Normality Checking

2.1.1 Univariate Case

A. Qualitatively

1. Stem-and-Leaf Plot and Box-and-Whisker Plot (box plot)
2. Q-Q Plot (Quantile - Quantile Plot)

B. Quantitatively

1. Shapiro-Wilk W Test
2. Kolmogorov-Smirnov (KS) Test (for large sample)
3. Skewness and Kurtosis

To test for univariate normality (Q-Q plot and statistical tests), we can make use of `proc univariate` and the syntax is given below:

```
proc univariate data=dataset_name normal;
    var x1 x2;
    symbol1 value=dot color=blue;
    qqplot x1 x2 /normal(mu=est sigma=est color=red w=2) square;
    run;
```

In this example SAS code, the univariate normalities of variables `x1` and `x2` are specified by the statement `var`. The option `normal` specified in the `proc` statement tells SAS to perform the quantitative testing of univariate normality (Shapiro-Wilk W test and Kolmogorov-Smirnov test). For both quantitative tests, the normality is tested under the null of normality. Thus, when the p-value is greater than 0.05, we can conclude that the normality cannot be rejected at 5% significance level. To check the normality by the skewness and kurtosis, both statistics are shown in the output already. If the data series is close to the normal distribution, the skewness and kurtosis in the SAS output should be close to 0 since the kurtosis in the SAS output is the excess kurtosis indeed. On the other hand, the statement `qqplot` is used to detect the unvariate normality graphically even though the options used here are quite complicated.

2.1.2 Multivariate Case

A. Qualitatively

1. Scatter Plot

2. Construct Contours
3. Chi-square Plot

B. Quantitatively

1. Principal Components Method

To test the bivariate normality graphically (chi-square plot), we can make use of a SAS macro `chisq_plot`. The general usage of this macro is given as follows:

```
%include 'D:/chisq_plot.sas';
%chisq_plot(dataset_name,x1 x2);
```

Before invoking the macro, the `%include` statement should be placed at the beginning of the SAS program. This `%include` statement asks SAS to read the file `chisq_plot.sas` into the memory and the macro is not executed at this moment. The macro is then invoked by using statement `%chisq_plot` which has been in the file `chisq_plot.sas` already. Basically, this macro consists of two arguments: (1) The name of dataset and (2) the normality of a list of variables being tested. Note that each variable is separated by a space only. No comma '`,`' is inserted between the variable names.

To test the multivariate normality test quantatively, principal components method can be used here. This test consists of two parts (1) find the principal components of a given dataset (2) perform univariate normality test for each principal component.

In step (1), we need to use the following SAS procedure:

```
proc princomp data=dataset_name cov standard out=dataset_name;
   var variables;
   run;
```

Note that the option `cov` is essential here and this option will be discussed in the chapter of principal components analysis later. The option `out=` will store computed PC (principal components) scores in a dataset which will be used in next step. The option `standard` asks SAS to compute the standardized PC scores with unit variance. Suppose that we test the multivariate normality of variables `x1`, `x2` and `x3` in a dataset `test`. Then, we can use the following statements:

```
proc princomp data=test cov standard out=pcascore;
   var x1 x2 x3;
   run;
```

Then, the computed PC scores will be saved in the dataset `pcascore`. In step (2), we need to use the following SAS procedure:

```
proc univariate data=dataset_name normal;
   var variables;
   qqplot variables /normal(mu=est sigma=est color=red w=2)
         square;
   run;
```

The most important option in this procedure is `normal`. If it is omitted, SAS will not perform any univariate normality test. The `qqplot` statement is just used to produce the Q-Q plots for the variables only and it is not essential for the test.

Continue from the previous example, the corresponding SAS statements are

```
proc univariate data=pcascore normal;
   var prin1 prin2 prin3;
   qqplot prin1 prin2 prin3 /normal(mu=est sigma=est color=red
         w=2) square;
   run;
```

Note that the variables `prin1`, `prin2` and `prin3` are default output variable names by `proc princomp`. The maximum number of principal components should be equal to the number of variables. In this example, 3 variables are used to find the principal components, 3 principal components will be produced then. If the number of principal components increases, the number after the prefix `prin` will be increased as well.

2.2 Box-Cox Transformation

If some variables are not found to be univariate normal, one may consider that some transformations are made on these variables in order to yield the univariate normality. Box-Cox transformation could be one of the methods used for this purpose.

For Box-Cox transformation by SAS procedures, the following syntax is used here:

```
data testdat;
   set testdat;
   inpt = 1;
   run;
proc transreg data=testdat;
   model boxcox(x1 / lambda = -2 to 2 by 0.01) = identity(inpt)
        /noint;
   run;
```

In the above data steps, a new variable `inpt` is created in the dataset `testdat` artificially. Then, `proc transreg` is executed. The `model` statement is essential because the dependent / response variable `x1` is transformed by Box-Cox transformation and we regress this transformed variable on a constant term. Note that the option `/noint` is used in the `model` statement. Otherwise, the problem of multicollinearity will occur. In this example, the value of λ in Box-Cox transformation is searched over the range of -2 and 2 with step size 0.01. The log-likelihood is maximized in the `proc transreg`. The best value of λ is indicated by the symbol "<" and its 95% confidence interval is indicated by the symbol "*".

Note that the step size should not be too small. Otherwise, the computational time will be increased dramatically.

2.3 Example

The data below were collected in an experiment on jute in Bishnupur village of West Bengal, India, in which the weights of green jute plants (x_1) and their dry jute fibers (x_2) were recorded for 20 randomly selected individual plants.

Plant No.	x_1	x_2	**Plant No.**	x_1	x_2
1	68	971	11	33	462
2	63	892	12	27	352
3	70	1125	13	21	305
4	6	82	14	5	84
5	65	931	15	14	229
6	9	112	16	27	332
7	10	162	17	17	185
8	12	321	18	53	703
9	20	315	19	62	872
10	30	375	20	65	740

a. Examine the univariate normality of x_1 and x_2.

b. Examine the bivariate normality of x_1 and x_2 by chi-square plot.

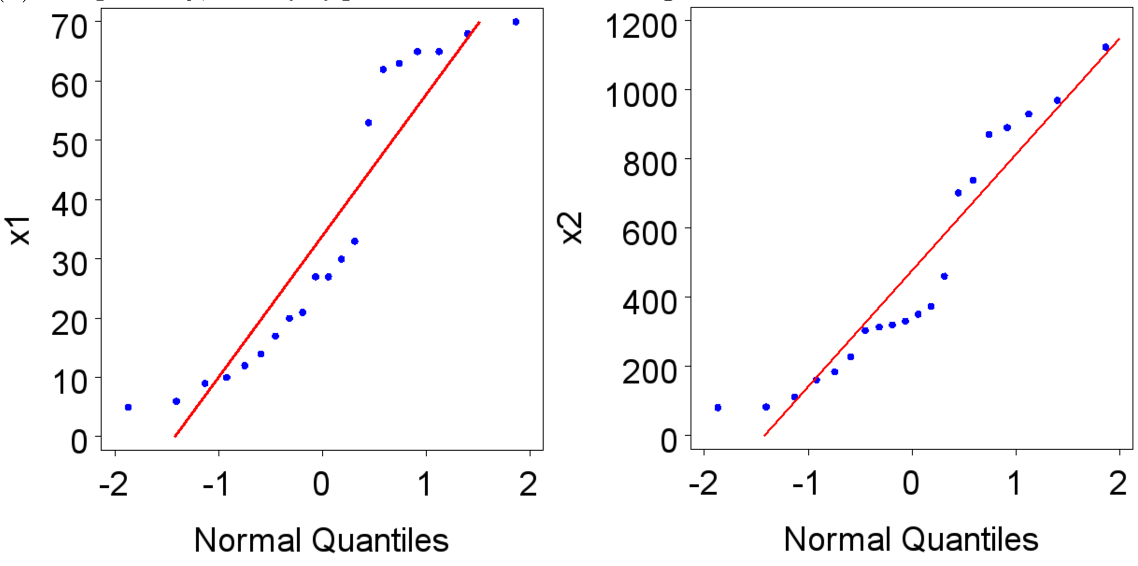
c. Assume bivariate normality on (x_1, x_2),

(i) Compute the Hotelling's $T^2 = n(\bar{\boldsymbol{x}} - \boldsymbol{\mu}_0)'\boldsymbol{S}^{-1}(\bar{\boldsymbol{x}} - \boldsymbol{\mu}_0)$ statistic where $\boldsymbol{\mu}_0$ is the hypothesized mean, for testing $H_0 : \mu_1 = 50, \mu_2 = 1000$.

(ii) Find the maximum likelihood estimate for $\mathrm{E}(x_1|x_2)$ in terms of x_2. What is the maximum likelihood estimate of $\mathrm{E}(x_1|x_2 = 200)$?

Solution

(a) Graphically, the Q-Q plots for x_1 and x_2 are given below:



We can see that both Q-Q plots are not close to the reference line. Therefore, we can say that both of them may not be normally distributed.

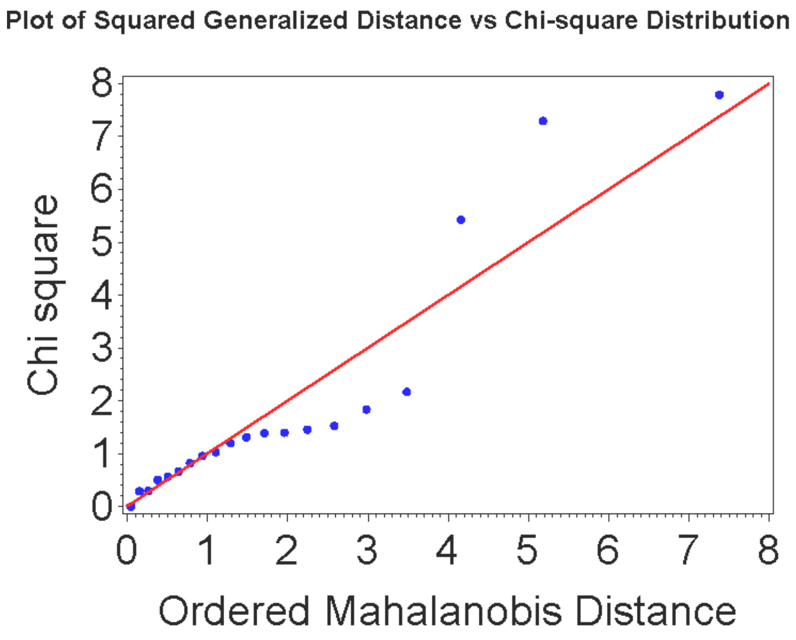
From the SAS output, we have the results of Kolmogorov-Smirnov test and Shapiro-Wilk test.

	Shapiro-Wilk	p-value	Kolmogorov-Smirnov	p-value
x_1	0.858536	0.0074	0.180723	0.0857
x_2	0.889258	0.0261	0.220093	0.0121

Here, only the results of Shapiro-Wilk test are considered because the sample size is small and the Kolmogorov-Smirnov test is valid for large-sample. From the Shapiro-Wilk test, both x_1 and x_2 are non-normal.

(b) There are two new SAS statements: (1) `%include "d:\chisq_plot.sas";` (2) `%chisq_plot(plant, x1 x2);` The first line can attach the SAS codes of an external SAS program file to current SAS program, but the statements in the file are not executed at this moment. Indeed, the file '`chisq_plot.sas`' stores SAS statements which define a macro called `%chisq_plot` to produce a chi-square plot. The second line executes the macro `%chisq_plot` which uses the name of dataset and variable names as input parameters. Nevertheless, all we need to know is how to use the macro `%chisq_plot` rather than understand it. So, the first input parameter is the name of dataset while the second input parameter is the pair of variables used to produce chi-square plot.

After running the macro, the chi-square plot is shown as follow:



Obviously, the chi-square plot is not close to the reference line. Therefore, x_1 and x_2 may not be bivariate normal.

(c) (i) By using the formula $T^2 = n(\overline{\boldsymbol{x}} - \boldsymbol{\mu})'\boldsymbol{S}^{-1}(\overline{\boldsymbol{x}} - \boldsymbol{\mu})$ where $\boldsymbol{\mu} = (50, 1000)'$, the computed T^2 is 408.8485.
(ii) The MLE estimate for $\mathrm{E}(x_1|x_2)$ is given by

$$\widehat{\mathrm{E}\,(X_1|X_2 = x_2)} = \bar{x}_1 + \frac{s_{12}}{s_{22}}(x_2 - \bar{x}_2),$$

where $s_{12} = \sum_{i=1}^n (x_{i1} - \overline{x}_1)(x_{i2} - \overline{x}_2)/(n-1)$ and $s_{22} = \sum_{i=1}^n (x_{i2} - \overline{x}_2)^2/(n-1)$. Note that the above formula is the same as that in the case of simple linear regression.

Given that $x_2 = 200$, we have

$$\widehat{\mathrm{E}\,(X_1|X_2 = 200)} = 14.4823.$$

SAS Codes for Example

```
data plant;
   infile 'd:\plant.txt' delimiter='09'x;
   /* delimiter='09'x means that the observations are separated by tab */
   input number $ x1 x2;
   run;

/* (a) */
proc univariate data=plant normal;
   var x1 x2;
   symbol1 value=dot color=blue;
   qqplot x1 x2 /normal(mu=est sigma=est color=red w=2) square;
   run;

/* (b) */
%include 'd:\chisq_plot.sas'; /* include the macro */
%chisq_plot(plant,x1 x2); /* run the macro */

/* (c) */
proc iml;
   use plant;
   read all var {x1 x2} into x;
   close plant;

   /* (i) */
   mu = {50,1000};
   n = nrow(x);
   xbar = x[:,]`;
   s = (x`*x-n*xbar*xbar`)/(n-1);
   tsq = n*(xbar-mu)`*inv(s)*(xbar-mu);
   print 'Hotelling T-square Statistic',, tsq;

   /* (ii) */
   x2 = 200;
   ex1x2 = xbar[1]+s[1,2]*inv(s[2,2])*(x2-xbar[2]);
   print 'MLE of E(x1|x2=200)',, ex1x2;

quit;
```