SpringerBriefs in Statistics

For further volumes:
http://www.springer.com/series/8921

Tejas Desai

# A Multiple-Testing Approach to the Multivariate Behrens-Fisher Problem

## with Simulations and Examples in SAS®

Springer

Tejas Desai
Adani Institute of Infrastructure
Management
Ahmedabad, India

ISSN 2191-544X
ISBN 978-1-4614-6442-6
DOI 10.1007/978-1-4614-6443-3
Springer New York Heidelberg Dordrecht London

ISSN 2191-5458 (electronic)
ISBN 978-1-4614-6443-3 (eBook)

Library of Congress Control Number: 2013931259



Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

# Contents

# Chapter 1
# Introduction

**Abstract** In this chapter, we give a brief introduction to the Behrens–Fisher problem. An outline of the rest of the chapters is also provided. Since randomly incomplete data is considered in the rest of the chapters, we thereafter clarify the idea of "missing at random (MAR)" and "missing completely at random (MCAR)." In particular, we demonstrate that if variables in a data set are all mutually dependent, then an assumption of MAR is equivalent to the assumption of MCAR.

**Keywords** Behrens–Fisher problem • Missing data • MAR • MCAR • Multivariate • Normality

## 1.1 The Behrens–Fisher Problem

The Behrens–Fisher problem constitutes the problem of testing two or more univariate normal or multivariate normal means for equality when there is evidence to believe that the underlying variances or covariance matrices, respectively, are not equal. When univariate normal means are compared, the problem is usually known as ANOVA (analysis of variance), and when multivariate normal means are compared, the problem is usually known as MANOVA (multivariate analysis of variance). There is a vast amount of literature on the subject and it is not our aim to survey all of it. We restrict ourself to fiducial approaches. There is only one explicit fiducial approach in the literature and that is the approach of Li et al. (2011). Moreover, this approach applies only to $k$-sample ANOVA, $k \geq 2$. Besides the method of Li et al. a fiducial approach is implicit in Behrens (1929) and Fisher (1935). The approaches of Behrens, Fisher, and Li et al. were all proposed for the univariate case. It is our purpose to demonstrate how these three approaches can be generalized to the multivariate case when the underlying data is complete or randomly incomplete.

Since this is a monograph on *parametric* approaches to the Behrens–Fisher problem, the rest of the monograph will proceed as follows:

T. Desai, *A Multiple-Testing Approach to the Multivariate Behrens-Fisher Problem: with Simulations and Examples in SAS®*, SpringerBriefs in Statistics,
DOI 10.1007/978-1-4614-6443-3_1, © The Author 2013

Chapter 2: This chapter will compare three existing approaches to testing multivariate normality. Rejection resulting from these or any other test means that we cannot proceed further with a parametric approach, and have to look at nonparametric alternatives. The novelty in this chapter comes when one is testing for multivariate normality in presence of randomly incomplete data; wherefore, a multiple-testing approach is proposed.

Chapter 3: This chapter presents tests of equality of variances or covariance matrices. Such a test is important, because if it doesn't reject equality of variances/covariance matrices, then the variances/covariance matrices may be pooled to yield a better test. The novelty in this chapter comes when comparing covariance matrices in presence of randomly incomplete data. A new test based on multiple testing is proposed for this purpose. It is noteworthy that this test may be used on complete data as well.

Chapter 4: This chapter generalizes the fiducial approaches of Behrens, Fisher, and Li et al. to the multivariate case. These approaches are recommended when a suitable test of multivariate normality doesn't lead to rejection and when a suitable test of equality of variances/covariance matrices leads to rejection. Once again, the novelty here is the use of multiple testing both in the complete and the randomly-incomplete-data case.

## 1.2 A Note on Missingness at Random

Randomly incomplete data sets will be considered in this monograph. Here "randomly incomplete" means in the sense of Rubin (1976, 1987). This notion of MAR (missing at random) needs to be examined closely. Suppose we have a data set $\mathbf{W}_n = (\mathbf{W}_{n,obs}, \mathbf{W}_{n,miss})$ where $\mathbf{W}_{n,obs}$ and $\mathbf{W}_{n,miss}$ denote the observed and missing parts, respectively. Then we say that data is MAR if $\Pr(\text{missingness}|\mathbf{W}_{n,obs}, \mathbf{W}_{n,miss}) = \Pr(\text{missingness}|\mathbf{W}_{n,obs})$. Now consider a bivariate normal random vector $\mathbf{W} = (X, Y)$ such that $E(\mathbf{W}) = (1, 2)$ and $Covar(\mathbf{W}) = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 2 \end{bmatrix}$. Suppose that $Y$ is missing whenever $X \geq 1$. Clearly, the probability of missingness depends on $X$. Consider all pairs $(x, y_{miss})$ where $X = x$ is observed and $Y = y_{miss}$ is not observed. Let

$$mu = 1 + (0.5 \times (1/2) \times (y_{miss} - 2)$$

and

$$sigmasq = 1 - (0.5 * (1/2) * 0.5)$$

Then notice that for any pair $(x, y_{miss})$,

$$x = mu + \sqrt{sigmasq} \times z \text{ where } z \text{ is some standard normal variate.}$$

Then, since *mu* depends on $y_{miss}$, $x$ depends on $y_{miss}$, and so Pr(missingness $|\mathbf{X}_{n,obs}, \mathbf{X}_{n,miss}) \neq$ Pr(missingness$|\mathbf{X}_{n,obs})$. That is, the data is not MAR by virtue of the fact that $X$ and $Y$ are dependent on each other. So if the components of a random vector are all mutually dependent, an assumption of MAR, if true, implies MCAR (missingness completely at random). In the rest of this monograph, any assumption of MAR will be equivalent to the assumption of MCAR. This claim becomes clearer if we examine a simulation example. We generated 1000000 two-dimensional vectors from the above distribution. Before generating any missing values, we generated a variable *Y_miss* such that $Y\_miss = Y$. Then, if $X \geq 1$, we set $Y$ to be missing. We also define an indicator variable, $R$, such that $R = 0$ if $Y$ is missing and $R = 1$ otherwise. We then found that when $X \geq 1$, *Y_miss* ranges from $-3.95026$ to 8.78385. Then consider the following table:

**Table 1.1**

| $x \in A$ *and* $y \in B$ | $\Pr(R = 0), \Pr(R = 1)$ |
|---|---|
| $A = \{x : 0 \leq x \leq 2\}$, $B = \{y\_miss : -3.95026 \leq y\_miss \leq 2\}$ | 0.4322, 0.5678 |
| $A = \{x : 0 \leq x \leq 2\}$, $B = \{y\_miss : 2 \leq y\_miss <= 8.78385\}$ | 0.5693, 0.4307 |

The above table demonstrates that the probability of missingness in $Y$ depends on both $X$ and *Y_miss* and not just on $X$. The SAS® syntax that generates the output in Table 1.1 is as follows:

```
data test;
do i = 1 to 1000000;
x = 1 + (sqrt(1) * normal(0));
mu = 2 + (0.5 * (1/1) * (x - 1));
sigma = 2 - (0.5 * (1/1) * 0.5);
y = mu + (sqrt(sigma) * normal(0));
y_miss = y;
if x >= 1 then y = .;
if y = . then r = 0; else r = 1;
output;
end;
run;
proc univariate data=test;
var y_miss;
where x >= 1;
run;
data one;
set test;
if ^((0 <= x <= 2)&(-3.95026 <= y_miss <= 2)) then delete;
run;
data two;
```

*set test*;
*if* ^((0 <= $x$ <= 2)&(2 < $y\_miss$ <= 8.78385)) *then delete*;
*run*;
*proc freq data=one*;
*tables r*;
*run*;
*proc freq data=two*;
*tables r*;
*run*;

Now suppose that, in the above example, we let $E(\mathbf{W}) = (1, 2)$ and $Covar(\mathbf{W}) = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$. Again, let $Y\_miss$ be such that $Y\_miss = Y$. Then, if $X \geq 1$, we set $Y$ to be missing. We then found that when $X \geq 1$, $Y\_miss$ ranges from $-4.77984$ to 8.83698. Then consider the following table:

**Table 1.2**

| $x \in A$ *and* $y \in B$ | $\Pr(R = 0), \Pr(R = 1)$ |
|---|---|
| $A = \{x : 0 \leq x \leq 2\}$, $B = \{y\_miss : -4.77984 \leq y\_miss \leq 2\}$ | 0.4994, 0.5006 |
| $A = \{x : 0 \leq x \leq 2\}$, $B = \{y\_miss : 2 \leq y\_miss <= 8.83698\}$ | 0.4997, 0.5003 |

Clearly, now the missingness doesn't seem to depend on the missing data and, indeed, the data is incomplete at random. In the rest of the monograph, MAR and MCAR are treated as synonymous, unless otherwise noted. The code that generated Table 1.2 is similar to the code that generated Table 1.1 and is left as an exercise to the reader.

We end this chapter with a note about the number of simulations and the significance level of hypothesis tests performed in the chapters that follow. Unless otherwise indicated, a 1000 simulations are used to investigate Type I errors and power of hypothesis tests. Furthermore, the significance level used is 5 % unless otherwise indicated.

# Chapter 2
# On Testing for Multivariate Normality

**Abstract** In this chapter we compare and contrast three approaches for testing multivariate normality. These are, namely, Mardia's skewness and kurtosis statistics and the Henze–Zirkler statistic. Type I errors and power are demonstrated using simulations in both the complete-data and the randomly-incomplete-data cases. In the randomly-incomplete-data case, we use Sidak's method for multiple testing. Examples are also provided.

**Keywords** Alternative hypothesis • Henze–Zirkler statistic • Mardia's skewness statistic • Mardia's kurtosis statistic • Multivariate • Normality • Null hypothesis • Power • Sidak's method • Type I error

## 2.1 The Complete-Data Case

Suppose $\mathbf{X} = \left(X_1, \ldots, X_p\right)$ is a $p$-dimensional ($p > 1$) random vector that has an unknown mean vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_p)$ and a positive-definite, symmetric covariance matrix

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & \sigma_{21} & \cdots & \sigma_{p1} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{p2} \\ \vdots & \ldots & \ddots & \vdots \\ \sigma_{p1} & \sigma_{p2} & \cdots & \sigma_p^2 \end{bmatrix}$$

Suppose we collect $n$ data $\mathbf{X}_i = (X_{i1}, \ldots, X_{ip}), i = 1, \ldots, n$, from the distribution of $\mathbf{X}$ above. Consider the five-variate normal distribution with mean row vector {1 2 3 4 5} and covariance matrix

T. Desai, *A Multiple-Testing Approach to the Multivariate Behrens-Fisher Problem: with Simulations and Examples in SAS®*, SpringerBriefs in Statistics,
DOI 10.1007/978-1-4614-6443-3_2, © The Author 2013

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0.5 & 0.4 & 0.3 & 0.2 \\ 0.5 & 1 & 0.5 & 0.4 & 0.3 \\ 0.4 & 0.5 & 1 & 0.5 & 0.4 \\ 0.3 & 0.4 & 0.5 & 1 & 0.5 \\ 0.2 & 0.3 & 0.4 & 0.5 & 1 \end{bmatrix}$$

Table 2.1 below gives Type 1 error rates under the null hypothesis that a multivariate random vector is from a multinormal distribution. For each sample size considered, a 1000 simulations were run. The Type I error rates were obtained using Mardia's skewness and kurtosis statistics (Mardia 1974) and the Henze–Zirkler test (Henze and Zirkler 1990). We will denote Mardia's skewness and kurtosis statistics as S and K, respectively. The Henze–Zirkler statistic will be denoted as HZ.

**Table 2.1** Type 1 error rates for complete data under the null

| Sample size | S | K | HZ |
|---|---|---|---|
| 6 | 0.000 | 0.000 | 0.000 |
| 10 | 0.013 | 0.000 | 0.012 |
| 20 | 0.061 | 0.008 | 0.022 |
| 30 | 0.077 | 0.018 | 0.018 |
| 40 | 0.073 | 0.021 | 0.027 |
| 50 | 0.077 | 0.039 | 0.030 |
| 75 | 0.060 | 0.027 | 0.038 |
| 100 | 0.063 | 0.033 | 0.039 |

The above table suggests that, when the null hypothesis of normality is true, K and HZ tend to be conservative, while S tends to be a bit anti-conservative. The SAS® code that generated the null distribution and Table 2.1 is given as follows. Note that the simulation uses a sample size of 100. To use a different sample size, modification of the code is simple and is left as an exercise for the reader.

```
proc iml;
sim = j(1000000, 5, 0);
sigma1 = {1 0.5 0.4 0.3 0.2,
          0.5 1 0.5 0.4 0.3,
          0.4 0.5 1 0.5 0.4,
          0.3 0.4 0.5 1 0.5,
          0.2 0.3 0.4 0.5 1};
mu1 = {1 2 3 4 5};
do i = 1 to 1000000;
   sim[i, 1] = mu1[1, 1] + (sqrt(1) * normal(0));
   mu = mu1[1, 2] + (sigma1[2, 1] * inv(sigma1[1, 1]) * (sim[i, 1] - mu1[1, 1]));
   s_sq = sigma1[2, 2] - (sigma1[2, 1] * inv(sigma[1, 1]) * sigma1[1, 2]);
   sim[i, 2] = mu + (sqrt(s_sq) * normal(0));
   mu = mu1[1, 3] + (sigma1[3, 1 : 2] * inv(sigma1[1 : 2, 1 : 2]) * (sim[i, 1 : 2]^t - mu1[1, 1 : 2]^t));
   s_sq = sigma1[3, 3] - (sigma1[3, 1 : 2] * inv(sigma1[1 : 2, 1 : 2]) * sigma1[1 : 2, 3]);
   sim[i, 3] = mu + (sqrt(s_sq) * normal(0));
```

```
    mu = mu1[1,4] + (sigma1[4,1:3] * inv(sigma1[1:3,1:3]) * (sim[i,1:3]^t - mu1[1,1:3]^t));
    s_sq = sigma1[4,4] - (sigma1[4,1:3] * inv(sigma1[1:3,1:3]) * sigma1[1:3,4]);
    sim[i,4] = mu + (sqrt(s_sq) * normal(0));
    mu = mu1[1,5] + (sigma1[5,1:4] * inv(sigma1[1:4,1:4]) * (sim[i,1:4]^t - mu1[1,1:4]^t));
    s_sq = sigma1[5,5] - (sigma1[5,1:4] * inv(sigma1[1:4,1:4]) * sigma1[1:4,5]);
    sim[i,5] = mu + (sqrt(s_sq) * normal(0));

end;
create complete_type1 from sim [col name = {'x1' 'x2' 'x3' 'x4' 'x5'}];
append from sim;
run;
quit;


%macro test;


%do t = 1 %to 1000;
ods listing close;


proc iml;
index = %sys eval f(((&t - 1) * 100) + 1) : %sys eval f(&t * 100);
use complete_type1;
read point index var {x1 x2 x3 x4 x5} into sim;
create sim from sim [col name = {'x1' 'x2' 'x3' 'x4' 'x5'}];

append from sim;
run;
quit;


proc model data = sim;
x1 = parm1;
x2 = parm2;
x3 = parm3;
x4 = parm4;
x5 = parm5;
fit x1 x2 x3 x4 x5 / normal;
ods output NormalityTest = nt
run;
quit;


proc iml;
use nt;
read all var {Prob} into p;
est = p[6,]||p[7,]||p[8,];
create est&t from est [col name = {'s' 'k' 'hz'}];
append from est;
run;
quit;


%end;
```

```
%mend test;


%test;


%macro cat;
    %do i = 1 %to 1000;
      est&i
     %end;
%mend cat;


data out_comp_tp1;
set %cat;
i = _n_;
run;


data _null_;
set out_comp_tp1 end = last;
if s < 0.05 then c1 + 1;
if k < 0.05 then c2 + 1;
if hz < 0.05 then c3 + 1;
if last then do; put c1 c2 c3; end;
run;
```

Now we consider power under an alternative. Consider the mixture of normals which has mean {1 2 3 4 5} with probability 0.5 and {-1 -2 -3 -4 -5} with probability 0.5. Let the covariance matrices be the same in both cases. The SAS® code that generates this data set is as follows:

```
proc iml;
sim = j(1000000, 5, 0);
sigma1 = {1 0.5 0.4 0.3 0.2,
          0.5 1 0.5 0.4 0.3,
          0.4 0.5 1 0.5 0.4,
          0.3 0.4 0.5 1 0.5,
          0.2 0.3 0.4 0.5 1};
do i = 1 to 1000000;
    r = rantbl(i * 10, 0.5, 0.5);
    if r = 1 then mu1 = {1 2 3 4 5};
     else mu1 = {−1 −2 −3 −4 −5};
    sim[i, 1] = mu1[1, 1] + (sqrt(1) * normal(0));
    mu = mu1[1, 2] + (sigma1[2, 1] * inv(sigma1[1, 1]) * (sim[i, 1] − mu1[1, 1]));
    s_sq = sigma1[2, 2] − (sigma1[2, 1] * inv(sigma[1, 1]) * sigma1[1, 2]);
    sim[i, 2] = mu + (sqrt(s_sq) * normal(0));
    mu = mu1[1, 3] + (sigma1[3, 1 : 2] * inv(sigma1[1 : 2, 1 : 2]) * (sim[i, 1 : 2]^t − mu1[1, 1 : 2]^t));
    s_sq = sigma1[3, 3] − (sigma1[3, 1 : 2] * inv(sigma1[1 : 2, 1 : 2]) * sigma1[1 : 2, 3]);
    sim[i, 3] = mu + (sqrt(s_sq) * normal(0));
    mu = mu1[1, 4] + (sigma1[4, 1 : 3] * inv(sigma1[1 : 3, 1 : 3]) * (sim[i, 1 : 3]^t − mu1[1, 1 : 3]^t));
    s_sq = sigma1[4, 4] − (sigma1[4, 1 : 3] * inv(sigma1[1 : 3, 1 : 3]) * sigma1[1 : 3, 4]);
    sim[i, 4] = mu + (sqrt(s_sq) * normal(0));
```

$mu = mu1[1,5] + (sigma1[5,1:4] * inv(sigma1[1:4,1:4]) * (sim[i,1:4]^t - mu1[1,1:4]^t));$
$s\_sq = sigma1[5,5] - (sigma1[5,1:4] * inv(sigma1[1:4,1:4]) * sigma1[1:4,5]);$
$sim[i,5] = mu + (sqrt(s\_sq) * normal(0));$

$end;$
*create complete_power from sim* [col *name* = {'x1' 'x2' 'x3' 'x4' 'x5'}];
*append from sim*;

The macro *test* which we used for examining Type I errors can be again used here except that we replace the data set $complete\_type1$ with the data set generated above, namely, *complete_power*.

Then we have the following power table:

**Table 2.2** Power under an alternative for complete data

| Sample size | S | K | HZ |
|---|---|---|---|
| 6 | 0.000 | 0.000 | 0.000 |
| 10 | 0.012 | 0.000 | 0.028 |
| 20 | 0.034 | 0.015 | 0.092 |
| 30 | 0.037 | 0.041 | 0.233 |
| 40 | 0.031 | 0.071 | 0.420 |
| 50 | 0.030 | 0.102 | 0.658 |
| 75 | 0.027 | 0.166 | 0.982 |
| 100 | 0.036 | 0.217 | 1.000 |

Note that in Table 2.2, S doesn't seem to converge to any value, while K increases very slowly compared to the HZ statistic.

### 2.1.1 Example: Rao's Cork Data

This data set of (Rao, 1948; excerpted from Khattree and Naik, 1999) consists of weights of cork borings in four directions for 28 trees. According to Khattree and Naik, E.S. Pearson believed that this data is asymmetric. The data and the ensuing tests of normality are provided in the SAS® code below:

*data rao*;
*input n e s w*;
*cards*;
72 66 76 77
60 53 66 63
56 57 64 58
41 29 36 38
32 32 35 36
30 35 34 26
39 39 31 27

```
42 43 31 25
37 40 31 25
33 29 27 36
32 30 34 28
63 45 74 63
54 46 60 52
47 51 52 43
91 79 100 75
56 68 47 50
79 65 70 61
81 80 68 58
78 55 67 60
46 38 37 38
39 35 34 37
32 30 30 32
60 50 67 54
35 37 48 39
39 36 39 31
50 34 37 40
43 37 39 50
48 54 57 43
;
run;


proc model data = rao;
n = parm1;
e = parm2;
s = parm3;
w = parm4;
fit n e s w / normal;
run;
quit;
```

The p-values for Mardia's skewness and kurtosis statistics, along with the p-value for the Henze–Zirkler statistic, are, respectively, 0.2369, 0.6904, and 0.0222. Thus, at the 5 % significance level, the Henze–Zirkler statistic rejects four-variate normality, while Mardia's statistics fail to reject four-variate normality.

## 2.2 The Randomly-Incomplete-Data Case

To investigate this case, we generated the data set specified in the SAS® code below. Note that with probability 0.7, all components are observed; with probability 0.10, the fifth component is set to missing; and with probability 0.2, the fourth and

fifth components are set to missing. We approach this case using multiple imputation (Rubin 1987) and multiple testing. Five imputations were generated for each simulation. For each of the three statistics, since the five p-values corresponding to five imputations are not mutually independent, one cannot use the FDR method of Benjamini and Hochberg (1995). Rather, we chose to use Sidak's procedure (Sidak, 1967). The code for generating the null data set is as follows:

```
proc iml;
sim = j(1000000, 6, 0);
sigma1 = {1 0.5 0.4 0.3 0.2,
          0.5 1 0.5 0.4 0.3,
          0.4 0.5 1 0.5 0.4,
          0.3 0.4 0.5 1 0.5,
          0.2 0.3 0.4 0.5 1};
mu1 = {1 2 3 4 5};
do i = 1 to 1000000;
    sim[i, 1] = mu1[1, 1] + (sqrt(1) * normal(0));
    mu = mu1[1, 2] + (sigma1[2, 1] * inv(sigma1[1, 1]) * (sim[i, 1] - mu1[1, 1]));
    s_sq = sigma1[2, 2] - (sigma1[2, 1] * inv(sigma[1, 1]) * sigma1[1, 2]);
    sim[i, 2] = mu + (sqrt(s_sq) * normal(0));
    mu = mu1[1, 3] + (sigma1[3, 1 : 2] * inv(sigma1[1 : 2, 1 : 2]) * (sim[i, 1 : 2]^t - mu1[1, 1 : 2]^t));
    s_sq = sigma1[3, 3] - (sigma1[3, 1 : 2] * inv(sigma1[1 : 2, 1 : 2]) * sigma1[1 : 2, 3]);
    sim[i, 3] = mu + (sqrt(s_sq) * normal(0));
    mu = mu1[1, 4] + (sigma1[4, 1 : 3] * inv(sigma1[1 : 3, 1 : 3]) * (sim[i, 1 : 3]^t - mu1[1, 1 : 3]^t));
    s_sq = sigma1[4, 4] - (sigma1[4, 1 : 3] * inv(sigma1[1 : 3, 1 : 3]) * sigma1[1 : 3, 4]);
    sim[i, 4] = mu + (sqrt(s_sq) * normal(0));
    mu = mu1[1, 5] + (sigma1[5, 1 : 4] * inv(sigma1[1 : 4, 1 : 4]) * (sim[i, 1 : 4]^t - mu1[1, 1 : 4]^t));
    s_sq = sigma1[5, 5] - (sigma1[5, 1 : 4] * inv(sigma1[1 : 4, 1 : 4]) * sigma1[1 : 4, 5]);
    sim[i, 5] = mu + (sqrt(s_sq) * normal(0));
   r = rantbl(0, 0.7, 0.1, 0.2);
   if r = 2 then sim[i, 5] = .;
   if r = 3 then do; sim[i, 4] = .; sim[i, 5] = .; end;
   sim[i, 6] = r;

end;
create incomplete_type1 from sim [colname = {'x1' 'x2' 'x3' 'x4' 'x5'}];
append from sim;
run;
quit;
```

The estimated Type I errors using five imputations are given below in Table 2.3.

**Table 2.3** Type I error using five imputations

| Sample size | Statistic | | |
|---|---|---|---|
| n | S | K | HZ |
| 20 | 0.040 | 0.000 | 0.013 |
| 30 | 0.052 | 0.001 | 0.016 |
| 40 | 0.058 | 0.002 | 0.017 |
| 50 | 0.050 | 0.004 | 0.017 |
| 75 | 0.052 | 0.007 | 0.019 |
| 100 | 0.046 | 0.008 | 0.023 |

The above table illustrates that the S statistic tends to remain near the nominal level, whereas the K and HZ statistics tend to approach the nominal level extremely slowly from below. However, none of the three statistics is anti-conservative.

The SAS® code for the macro that generated the above table is as follows:

```
%macro incomplete;

%do t = 1 %to 1000;
ods listing close;

proc iml;
index = %sysevalf(((&t - 1) * 100) + 1) : %sysevalf(&t * 100);
use incomplete_type1;
read point index var {x1 x2 x3 x4 x5 r} into x;
create x from x [colname = {'x1' 'x2' 'x3' 'x4' 'x5' 'r'}];
append from x;
run;
quit;

proc mi data = x nimpute = 5 out = bayes;
var x1 x2 x3 x4 x5;
mcmc;
run;

proc sort data = bayes;
by _Imputation_;
run;

proc model data = bayes;
x1 = parm1;
x2 = parm2;
x3 = parm3;
x4 = parm4;
x5 = parm5;
fit x1 x2 x3 x4 x5 / normal;
by _Imputation_;
ods output NormalityTest = nt1;
run;
quit;

proc sort data = nt1;
by test;
run;

proc iml;
use nt1;
```

```
read all var {prob} into p;
est = p[1 : 15, ]';
create est&t from est [colname = {'hz1' 'hz2' 'hz3' 'hz4' 'hz5' 'mk1' 'mk2' 'mk3' 'mk4' 'mk5'
                                  'ms1' 'ms2' 'ms3' 'ms4' 'ms5'}];
append from est;
run;
quit;


%end;
%mend incomplete;


%incomplete;


data out_incomp_tp1;
set %cat;
i = _n_;
run;


data _null_;
set out_incomp_tp1 end = last;
k = 1 − (0.95 ∗ ∗(1/5));
if ms1 < k or ms2 < k or ms3 < k or ms4 < k or ms5 < k then c1 + 1;
if mk1 < k or mk2 < k or mk3 < k or mk4 < k or mk5 < k then c2 + 1;
if hz1 < k or hz2 < k or hz3 < k or hz4 < k or hz5 < k then c3 + 1;
if last then do; put c1 c2 c3; end;
run;
```

Table 2.4 below displays the power under the mixture alternative considered for the complete-data case. Missing data was simulated in exactly the same manner as for the null case considered above.

**Table 2.4** Power using five imputations

| Sample size | Statistic | | |
|---|---|---|---|
| n | S | K | HZ |
| 20 | 0.021 | 0.000 | 0.049 |
| 30 | 0.021 | 0.000 | 0.160 |
| 40 | 0.031 | 0.008 | 0.317 |
| 50 | 0.032 | 0.018 | 0.527 |
| 75 | 0.019 | 0.059 | 0.936 |
| 100 | 0.031 | 0.092 | 0.999 |

Table 2.4 shows that of the three statistics, HZ is the most powerful. It is not clear if the power of the S statistic increases with increase in sample size, whereas the power of the K statistic increases very slowly with increasing sample size.

### 2.2.1 Example: Audiology Growth Data

Nunez-Anton and Woodworth (1994) present and analyze data from the Iowa Cochlear Implant Project (Gantz et al. 1988). The data consist of percentages of correct scores on a sentence test administered to two groups of deaf patients fitted with two different cochlear implants. Measurements were made 1, 9, 18, and 30 months after the fitting of implants. The data is presented in the SAS® data step below. Note that Nunez-Anton and Woodworth use data for only those patients who reached at least 5 % understanding. Thus, of the total sample size of 44, data for 9 patients were deleted bringing the total sample size down to 35. Following the data step, the code for the tests of normality done using five Bayesian imputations is presented for group 1. The corresponding code for group 0 is similar and not presented. In the data step below, note that the variable $r$ indicates the type of missing-data pattern:

```
data audio;
input group x1 x2 x3 x4 r;
id = _n_;
cards;
1 28.57 53.00 57.83 59.22 1
1 . 13.00 21.00 26.50 2
1 60.37 86.41 . . 4
1 33.87 55.60 61.06 . 3
1 1.61 0.69 . . 4
1 26.04 61.98 67.28 . 3
1 . 59.00 66.80 83.20 2
1 11.29 38.02 . . 4
1 0.00 0.00 0.00 2.76 1
1 . 35.10 37.79 54.80 2
1 16.00 33.00 45.39 40.09 1
1 40.55 50.69 41.70 52.07 1
1 3.90 11.06 4.15 14.90 1
1 1.80 2.30 2.53 2.53 1
1 0.00 17.74 44.70 48.85 1
1 64.75 84.50 92.40 95.39 1
1 38.25 81.57 89.63 . 3
1 67.50 91.47 92.86 . 3
1 45.62 58.00 . . 4
1 0.00 0.00 37.00 . 3
1 51.15 66.13 . . 4
1 0.00 48.16 . . 4
1 0.00 0.92 . . 4
0 . 0.00 0.90 1.61 2
```

```
0 0.00 0.00 0.00 . 3
0 0.00 0.00 . . 4
0 8.76 24.42 . . 4
0 0.00 20.79 27.42 31.80 1
0 2.30 12.67 28.80 24.42 1
0 12.90 28.34 . . 4
0 . 45.50 43.32 36.80 2
0 68.00 96.08 97.47 99.00 1
0 20.28 41.01 51.15 61.98 1
0 65.90 81.30 71.20 70.00 1
0 0.00 8.76 16.59 14.75 1
0 0.00 0.00 0.00 0.00 1
0 9.22 14.98 9.68 . 3
0 11.29 44.47 62.90 68.20 1
0 30.88 29.72 . . 4
0 29.72 41.40 64.00 . 3
0 0.00 43.55 48.16 . 3
0 0.00 0.00 . . 4
0 8.76 60.00 . . 4
0 8.00 25.00 30.88 55.53 1
;
run;


data audio_;
set audio;
if _n_ in (5, 9, 14, 23, 24, 25, 26, 36, 42) then delete;
run;


data audio1;
set audio_;
if group = 0 then delete;
run;


proc mi data = audio1 nimpute = 5 out = bayes1;
var x1 x2 x3;
mcmc;
run;


proc sort data = bayes1;
by _Imputation_;
run;
```

```
proc model data = bayes1;
x1 = parm1;
x2 = parm2;
x3 = parm3;
fit x1 x2 x3 / normal;
by _Imputation_;
ods output NormalityTest = nt1;
run;
quit;


proc sort data = nt1;
by test;
run;
```

Five imputations were drawn. The decisions reached by the S, K, and HZ statistics using Sidak's procedure for both groups are presented in Table 2.5 below.

**Table 2.5** Decisions for tests of multivariate normality of the audiology data

| | Statistic | | |
|---|---|---|---|
| Group | S | K | HZ |
| 1 | Reject = No | Reject = No | Reject = No |
| 0 | Reject = No | Reject = No | Reject = No |

# Chapter 3
# On Testing Equality of Covariance Matrices

**Abstract** In this chapter, we present two approaches for testing equality of covariance matrices. In the complete-data case, Box's M method is presented. The Type I errors and power of Box's M method are presented. In the randomly-incomplete-data case, a new method is proposed. This method uses the False Discovery Rate (FDR) algorithm of Benjamini and Hochberg (J. R. Stat. Soc. Series B. **57**, 1289–1300, 1995). The Type I errors and power in the randomly-incomplete-case are also presented. An example is also provided.

**Keywords** Alternative hypothesis • Box's M statistic • Covariance matrices • False discovery rate • Null hypothesis • Power • Type I error

## 3.1 The Complete-Data Case

To test the equality of covariance matrices, we start with Box's M test (Box 1949, 1950). A description of Box's M test is given in Rencher (2002). To investigate the Type I error rates under the null hypothesis that covariance matrices are the same, we generate data from three 5-dimensional normal distributions with covariance matrix used in Sect. 2.1. The mean vectors used are $\{1\ 2\ 3\ 4\ 5\}$, $\{-1\ -2\ -3\ -4\ -5\}$ and $\{0\ 0\ 0\ 0\ 0\}$. Table 3.1 below presents the Type I error rates for different combinations of sample sizes. The significance level is set at 5 %.

**Table 3.1** Type I error rates under the null

| $n1$ | $n2$ | $n3$ | Box's M Type I error rate |
|---|---|---|---|
| 10 | 15 | 20 | 0.053 |
| 20 | 30 | 40 | 0.051 |
| 40 | 60 | 80 | 0.054 |

T. Desai, *A Multiple-Testing Approach to the Multivariate Behrens-Fisher Problem: with Simulations and Examples in SAS®*, SpringerBriefs in Statistics,
DOI 10.1007/978-1-4614-6443-3_3, 

Table 3.1 suggests that Box's M tends to achieve the desired Type I error rate. The SAS® code that generated the above table is presented below:

```
%macro test;


%do t = 1 %to 1000;
ods listing close;
proc iml;
index1 = %sysevalf(((&t - 1) * 10) + 1) : %sysevalf(&t * 10);
use pop1;
read point index1 var {x1 x2 x3 x4 x5} into sim1;
index2 = %sysevalf(((&t - 1) * 15) + 1) : %sysevalf(&t * 15);
use pop2;
read point index2 var {x1 x2 x3 x4 x5} into sim2;
index3 = %sysevalf(((&t - 1) * 20) + 1) : %sysevalf(&t * 20);
use pop3;
read point index3 var {x1 x2 x3 x4 x5} into sim3;


n1 = nrow(sim1);
n2 = nrow(sim2);
n3 = nrow(sim3);


mu1_ = (sum(sim1[,1])/n1)||(sum(sim1[,2])/n1)||(sum(sim1[,3])/n1)||
       (sum(sim1[,4])/n1)||(sum(sim1[,5])/n1);
mu2_ = (sum(sim2[,1])/n2)||(sum(sim2[,2])/n2)||(sum(sim2[,3])/n2)||
       (sum(sim2[,4])/n2)||(sum(sim2[,5])/n2);
mu3_ = (sum(sim3[,1])/n3)||(sum(sim3[,2])/n3)||(sum(sim3[,3])/n3)||
       (sum(sim3[,4])/n3)||(sum(sim3[,5])/n3);


mu1 = j(n1,1,1)@mu1_;
mu2 = j(n2,1,1)@mu2_;
mu3 = j(n3,1,1)@mu3_;


sigma1 = (sim1 - mu1)` * (sim1 - mu1)/(n1 - 1);
sigma2 = (sim2 - mu2)` * (sim2 - mu2)/(n2 - 1);
sigma3 = (sim3 - mu3)` * (sim3 - mu3)/(n3 - 1);
nu1 = n1 - 1; nu2 = n2 - 1; nu3 = n3 - 1; p = n row(sigma1); k = 3
sigma_pool = ((nu1 * sigma1) + (nu2 * sigma2) + (nu3 * sigma3))/
             (n1 + n2 + n3 - 3);
```

$m = (((det(sigma1)/det(sigma\_pool)) * *(nu1/2)) * ((det(sigma2)/$
$det(sigma\_pool)) * *(nu2/2)) * ((det(sigma3)/$
$det(sigma\_pool)) * *(nu3/2)));$
$df = (0.5 * (k-1) * p * (p+1));$
$f1 = ((1/nu1) + (1/nu2) + (1/nu3)) - (1/(nu1 + nu2 + nu3));$
$f2 = ((2 * p * p) + (3 * p) - 1)/(6 * (p+1) * (k-1));$
$c1 = f1 * f2;$
$u = -2 * (1-c1) * log(m);$
$pval = 1 - probchi(u, df);$

*create est&t from pval* [*colname* = {′*p*′}];
*append from pval*;
*run*;
*quit*;

*%end*;

*%mend test*;

*%test*;

*data out_null*;
*set %cat*; $i = \_n\_$;
*run*;

*data* $\_null\_$;
*set out_null end* = *last*;
*if* $p < 0.05$ *then* $c1 + 1$;
*if last then do*; *put* $c1$; *end*;
*run*;

To test the power of Box's M statistic, we first generated data sets $pop1$, $pop2$, and $pop3$ in the same way as in the aforementioned case, except that the following covariance matrices were used, respectively:

$$\boldsymbol{\Sigma}_1 = \begin{bmatrix} 1 & 0.5 & 0.4 & 0.3 & 0.2 \\ 0.5 & 1 & 0.5 & 0.4 & 0.3 \\ 0.4 & 0.5 & 1 & 0.5 & 0.4 \\ 0.3 & 0.4 & 0.5 & 1 & 0.5 \\ 0.2 & 0.3 & 0.4 & 0.5 & 1 \end{bmatrix}, \boldsymbol{\Sigma}_2 = \begin{bmatrix} 1 & 0.7 & 0.4 & 0.3 & 0.2 \\ 0.7 & 1 & 0.5 & 0.4 & 0.3 \\ 0.4 & 0.5 & 1 & 0.5 & 0.4 \\ 0.3 & 0.4 & 0.5 & 1 & 0.5 \\ 0.2 & 0.3 & 0.4 & 0.5 & 1 \end{bmatrix}$$

$$\text{and } \boldsymbol{\Sigma}_3 = \begin{bmatrix} 1 & 0.5 & 0.4 & 0.3 & 0.9 \\ 0.5 & 1 & 0.5 & 0.4 & 0.3 \\ 0.4 & 0.5 & 1 & 0.5 & 0.4 \\ 0.3 & 0.4 & 0.5 & 1 & 0.5 \\ 0.9 & 0.3 & 0.4 & 0.5 & 1 \end{bmatrix}$$

The mean vectors were kept the same in the null case. The power achieved for different combinations of sample sizes is given in Table 3.2 below:

**Table 3.2** Power under the alternative

| $n1$ | $n2$ | $n3$ | Box's M Power |
|---|---|---|---|
| 10 | 15 | 20 | 0.962 |
| 20 | 30 | 40 | 1.000 |

Table 3.2 demonstrates that even for relatively small sample sizes, Box's M is quite powerful. The SAS® code that generated Table 3.2 is the same as that which generated Table 3.1, except that the multinormal observations used are from the three aforementioned distributions with three different covariance matrices.

### *3.1.1 Example: Wisconsin Nursing Home Study*

Johnson and Wichern (2002) present summary statistics of a study of nursing homes in Wisconsin. A purpose of this study was to examine the effects of ownership or certification (or both) on four types of costs: $X_1$ = cost of nursing labor, $X_2$ = cost of dietary labor, $X_3$ = cost of plant operation and maintenance labor, and $X_4$ = cost of housekeeping and laundry labor. The total sample size was 516 and the sample was divided into three groups based on type of ownership: private, nonprofit, or government. The sample sizes of these three groups were $n1 = 271$, $n2 = 138$, and $n3 = 107$, respectively. The three sample covariance matrices were the following:

$$\boldsymbol{\Sigma}_1 = \begin{bmatrix} 0.291 & -0.001 & 0.002 & 0.010 \\ -0.001 & 0.011 & 0.000 & 0.003 \\ 0.002 & 0.000 & 0.001 & 0.000 \\ 0.010 & 0.003 & 0.000 & 0.010 \end{bmatrix}, \boldsymbol{\Sigma}_2 = \begin{bmatrix} 0.561 & 0.011 & 0.001 & 0.037 \\ 0.011 & 0.025 & 0.004 & 0.007 \\ 0.001 & 0.004 & 0.005 & 0.002 \\ 0.037 & 0.007 & 0.002 & 0.019 \end{bmatrix},$$

$$\text{and } \boldsymbol{\Sigma}_3 = \begin{bmatrix} 0.261 & 0.030 & 0.003 & 0.018 \\ 0.030 & 0.017 & -0.000 & 0.006 \\ 0.003 & -0.000 & 0.004 & 0.001 \\ 0.018 & 0.006 & 0.001 & 0.013 \end{bmatrix}$$

Box's M test for the above summary statistics is performed using the following code:

```
proc iml;
n1 = 271;
n2 = 138;
n3 = 107;
sigma1 = {0.291 −0.001 0.002 0.010,
          −0.001 0.011 0.000 0.003,
           0.002 0.000 0.001 0.000,
           0.010 0.003 0.000 0.010};
sigma2 = {0.561 0.011 0.001 0.037,
          0.011 0.025 0.004 0.007,
          0.001 0.004 0.005 0.002,
          0.037 0.007 0.002 0.019};
sigma3 = {0.261 0.030 0.003 0.018,
          0.030 0.017 −0.000 0.006,
          0.003 −0.000 0.004 0.001,
          0.018 0.006 0.001 0.013};


nu1 = n1 − 1; nu2 = n2 − 1; nu3 = n3 − 1; p = nrow(sigma1); k = 3;
sigma_pool = ((nu1 ∗ sigma1) + (nu2 ∗ sigma2) + (nu3 ∗ sigma3))/
             (n1 + n2 + n3 − k);


m = (((det(sigma1)/det(sigma_pool)) ∗ ∗(nu1/2)) ∗ ((det(sigma2)/
    det(sigma_pool)) ∗ ∗(nu2/2)) ∗ ((det(sigma3)/
    det(sigma_pool)) ∗ ∗(nu3/2)));
df = (0.5 ∗ (k − 1) ∗ p ∗ (p + 1));
f1 = ((1/nu1) + (1/nu2) + (1/nu3)) − (1/(nu1 + nu2 + nu3));
f2 = ((2 ∗ p ∗ p) + (3 ∗ p) − 1)/(6 ∗ (p + 1) ∗ (k − 1));
c1 = f1 ∗ f2;
u = −2 ∗ (1 − c1) ∗ log(m);
pval = 1 − probchi(u, df);
```

*print pval*;
*run*;
*quit*;

The above code returns a p-value of 0, thus strongly rejecting homoscedasticity.

## 3.2 The Randomly-Incomplete-Data Case

Here we can try what we did in Sect. 2.2. Namely, we can create five copies of imputed data sets, perform Box's M test for each copy and then subject the 5 p-values to a multiple-testing procedure. However, the author's simulations show that this approach fails here. Box's M performed on imputed data sets returns highly inflated Type I errors. Thus, we have to try something else. Let $\mathbf{\Sigma}$ and $\boldsymbol{\rho}$ denote the unknown covariance and correlation matrices of a multivariate normal distribution. Let $\sigma_{11}, \ldots, \sigma_{pp}$ be the variances of the $p$ component variables. Let

$$\mathbf{D} = \mathbf{diag}(\sqrt{\sigma_{11}}, \sqrt{\sigma_{22}}, \ldots, \sqrt{\sigma_{pp}}).$$

Then it follows that (see Rencher 2002)

$$\boldsymbol{\rho} = D^{-1}\mathbf{\Sigma}\mathbf{D}^{-1} \text{ and } \mathbf{\Sigma} = \mathbf{D}\boldsymbol{\rho}\mathbf{D}.$$

From the above two relations between $\mathbf{\Sigma}$ and $\boldsymbol{\rho}$, it follows that two covariance matrices are equal *if and only if* the corresponding diagonal elements of $\mathbf{\Sigma}$ (individual variances) are equal *and* the corresponding off-diagonal elements of $\boldsymbol{\rho}$ (the correlations) are equal. This fact allows us to take a multiple-testing approach described shortly below. To investigate the Type I error, we consider three five-variate normal distributions with the common covariance matrix below:

$$\begin{bmatrix} 1 & 0.4 & 0.5 & 0.8 & 0.7 \\ 0.4 & 2 & 0.4 & 0.5 & 0.8 \\ 0.5 & 0.4 & 3 & 0.4 & 0.5 \\ 0.8 & 0.5 & 0.4 & 4 & 0.4 \\ 0.7 & 0.8 & 0.5 & 0.4 & 5 \end{bmatrix}$$

and mean vectors {1 2 3 4 5}, {−1 −2 −3 −4 −5}, and {0 0 0 0 0}. Next, we simulate missing values in the data sets. In the data with mean {1 2 3 4 5} the fifth component was set to missing with probability 0.1 and the fourth and fifth components were set to missing with probability 0.2. All components are observed with probability 0.7. In the data with mean $\{-1 - 2 - 3 - 4 - 5\}$, all components are observed with probability 0.75. The fifth component is set to missing with probability 0.1 and the fourth and fifth components are set to missing with probability 0.15. In the data with mean vector {0 0 0 0 0}, all components were observed with probability 0.8. The fifth component was set to missing with probability 0.1, and the fourth

and fifth components were set to missing with probability 0.1. If $(\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3)$ is a triple of data sets generated from the aforementioned three distributions, then we generate $m$ copies of imputed triples $(\mathbf{X}_{11}, \mathbf{X}_{12}, \mathbf{X}_{13})$, $(\mathbf{X}_{21}, \mathbf{X}_{22}, \mathbf{X}_{23})$, $(\mathbf{X}_{31}, \mathbf{X}_{32}, \mathbf{X}_{33})$, $(\mathbf{X}_{41}, \mathbf{X}_{42}, \mathbf{X}_{43})$, $\ldots$, $(\mathbf{X}_{m1}, \mathbf{X}_{m2}, \mathbf{X}_{m3})$. For each of the $m$ triples, we perform a test of equality of covariance matrices as follows. Note that since we are testing equality of 3 covariance matrices, we are making $p(p+1) = 30$ comparisons (comparing $\mathbf{\Sigma}_1$ to $\mathbf{\Sigma}_2$ and then comparing $\mathbf{\Sigma}_1$ to $\mathbf{\Sigma}_3$). Of the $p(p+1)$ comparisons, 10 are comparisons of component variances, and the rest are comparisons between corresponding correlations between components. To test equality of variances we use the well-known F test. To test equality of correlations between two corresponding components, we use the well-known Fisher's test. For each triple $(\mathbf{X}_{i1}, \mathbf{X}_{i2}, \mathbf{X}_{i3})$, $i = 1, \ldots, m$, we apply the FDR algorithm of Benjamini and Hochberg to the resulting 30 p-values. We reach a decision to either reject ($R_i = 1$) or do not reject the null ($R_i = 0$). Then we compute $R = (R_1 + R_2 + R_3 + R_4 + \ldots + R_m)/m$. If $R > 0.95$, then we reject the overall null hypothesis of equality of covariance matrices. Else, we do not reject.

**Table 3.3** Type I error rates under the null using Bayesian imputations and the FDR method

| Sample sizes | | | Number of imputations | |
|---|---|---|---|---|
| $n1$ | $n2$ | $n3$ | $m = 5$ | $m = 10$ |
| 20 | 25 | 30 | 0.085 | 0.048 |
| 40 | 50 | 60 | 0.069 | 0.037 |
| 50 | 75 | 100 | 0.053 | 0.048 |

Table 3.3 demonstrates that for smaller sample sizes, five imputations yield somewhat anti-conservative Type I error rates, but as the sample sizes increase, five imputations yield error rates that are not significantly different from the nominal level of 0.05. The table also illustrates that ten imputations yield error rates that are close to 0.05 even for smaller sample sizes. The SAS® code that generated the above table is as follows:

```
%macro null;

%do t = 1 %to 1000;
ods listing close;
proc iml;
index1 = %sysevalf(((&t - 1) * 20) + 1) : %sysevalf(&t * 20);
use pop1;
read point index1 var{x1 x2 x3 x4 x5 r} into sim1;
index2 = %sysevalf(((&t - 1) * 25) + 1) : %sysevalf(&t * 25);
use pop2;
read point index2 var {x1 x2 x3 x4 x5 r} into sim2;
index3 = %sysevalf(((&t - 1) * 30) + 1) : %sysevalf(&t * 30);
use pop3;
```

```
read point index3 var {x1 x2 x3 x4 x5 r} into sim3;
create x1 from sim1 [colname = {'x1' 'x2' 'x3' 'x4' 'x5' 'r'}];
append from sim1;
create x2 from sim2 [colname = {'x1' 'x2' 'x3' 'x4' 'x5' 'r'}];
append from sim2;
create x3 from sim3 [colname = {'x1' 'x2' 'x3' 'x4' 'x5' 'r'}];
append from sim3;
run;
quit;

proc mi data = x1 out = b1 nimpute = 5;
var x1 x2 x3 x4 x5;
run;

proc mi data = x2 out = b2 nimpute = 5;
var x1 x2 x3 x4 x5;
run;

proc mi data = x3 out = b3 nimpute = 5;
var x1 x2 x3 x4 x5;
run;

%do j = 1 %to 5;

proc iml;

index1 = %sysevalf(((&j - 1) * 20) + 1) : %sysevalf(&j * 20);
use b1;
read point index1 var {x1 x2 x3 x4 x5 r} into imp1;
index2 = %sysevalf(((&j - 1) * 25) + 1) : %sysevalf(&j * 25);
use b2;
read point index2 var {x1 x2 x3 x4 x5 r} into imp2;
index3 = %sysevalf(((&j - 1) * 30) + 1) : %sysevalf(&j * 30);
use b3;
read point index3 var {x1 x2 x3 x4 x5 r} into imp3;

n1 = nrow(imp1); n2 = nrow(imp2); ; n3 = nrow(imp3);
mu1_ = (sum(imp1[, 1])/n1)||(sum(imp1[, 2])/n1)||(sum(imp1[, 3])/n1)||
       (sum(imp1[, 4])/n1)||(sum(imp1[, 5])/n1);
mu2_ = (sum(imp2[, 1])/n2)||(sum(imp2[, 2])/n2)||(sum(imp2[, 3])/n2)||
       (sum(imp2[, 4])/n2)||(sum(imp2[, 5])/n2);
mu3_ = (sum(imp3[, 1])/n3)||(sum(imp3[, 2])/n3)||(sum(imp3[, 3])/n3)||
       (sum(imp3[, 4])/n3)||(sum(imp3[, 5])/n3);
mu1 = j(n1, 1, 1)@mu1_;
mu2 = j(n2, 1, 1)@mu2_;
```

$$
\begin{aligned}
&mu3 = j(n3, 1, 1)@mu3\_;\\
&sigma1 = (imp1[, 1:5] - mu1)' * (imp1[, 1:5] - mu1)/n1;\\
&d1 = diag(sqrt(sigma1[1, 1])//sqrt(sigma1[2, 2])//sqrt(sigma1[3, 3])//\\
&\quad sqrt(sigma1[4, 4])//sqrt(sigma1[5, 5]));\\
&corr1 = inv(d1) * sigma1 * inv(d1);\\
&sigma2 = (imp2[, 1:5] - mu2)' * (imp2[, 1:5] - mu2)/n2;\\
&d2 = diag(sqrt(sigma2[1, 1])//sqrt(sigma2[2, 2])//sqrt(sigma2[3, 3])//\\
&\quad sqrt(sigma2[4, 4])//sqrt(sigma2[5, 5]));\\
&corr2 = inv(d2) * sigma2 * inv(d2);\\
&sigma3 = (imp3[, 1:5] - mu3)' * (imp3[, 1:5] - mu3)/n3;\\
&d3 = diag(sqrt(sigma3[1, 1])//sqrt(sigma3[2, 2])//sqrt(sigma3[3, 3])//\\
&\quad sqrt(sigma3[4, 4])//sqrt(sigma3[5, 5]));\\
&corr3 = inv(d3) * sigma3 * inv(d3);\\
&p = j(30, 1, 0);\\
&f = sigma1[1, 1]/sigma2[1, 1];\\
&p[1, 1] = 1 - probf(f, n1 - 1, n2 - 1);\\
&tr1 = 0.5 * log((1 + corr1[2, 1])/(1 - corr1[2, 1]));\\
&tr2 = 0.5 * log((1 + corr2[2, 1])/(1 - corr2[2, 1]));\\
&z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));\\
&p[2, 1] = 2 * (1 - probnorm(abs(z)));\\
&tr1 = 0.5 * log((1 + corr1[3, 1])/(1 - corr1[3, 1]));\\
&tr2 = 0.5 * log((1 + corr2[3, 1])/(1 - corr2[3, 1]));\\
&z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));\\
&p[3, 1] = 2 * (1 - probnorm(abs(z)));\\
&tr1 = 0.5 * log((1 + corr1[4, 1])/(1 - corr1[4, 1]));\\
&tr2 = 0.5 * log((1 + corr2[4, 1])/(1 - corr2[4, 1]));\\
&z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));\\
&p[4, 1] = 2 * (1 - probnorm(abs(z)));\\
&tr1 = 0.5 * log((1 + corr1[5, 1])/(1 - corr1[5, 1]));\\
&tr2 = 0.5 * log((1 + corr2[5, 1])/(1 - corr2[5, 1]));\\
&z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));\\
&p[5, 1] = 2 * (1 - probnorm(abs(z)));\\
&f = sigma1[2, 2]/sigma2[2, 2];\\
&p[6, 1] = 1 - probf(f, n1 - 1, n2 - 1);\\
&tr1 = 0.5 * log((1 + corr1[3, 2])/(1 - corr1[3, 2]));\\
&tr2 = 0.5 * log((1 + corr2[3, 2])/(1 - corr2[3, 2]));\\
&z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));\\
&p[7, 1] = 2 * (1 - probnorm(abs(z)));\\
&tr1 = 0.5 * log((1 + corr1[4, 2])/(1 - corr1[4, 2]));\\
&tr2 = 0.5 * log((1 + corr2[4, 2])/(1 - corr2[4, 2]));\\
&z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));\\
&p[8, 1] = 2 * (1 - probnorm(abs(z)));\\
&tr1 = 0.5 * log((1 + corr1[5, 2])/(1 - corr1[5, 2]));\\
&tr2 = 0.5 * log((1 + corr2[5, 2])/(1 - corr2[5, 2]));\\
&z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));
\end{aligned}
$$

$p[9,1] = 2 * (1 - probnorm(abs(z)));$
$f = sigma1[3,3]/sigma2[3,3];$
$p[10,1] = 1 - probf(f, n1 - 1, n2 - 1);$
$tr1 = 0.5 * log((1 + corr1[4,3])/(1 - corr1[4,3]));$
$tr2 = 0.5 * log((1 + corr2[4,3])/(1 - corr2[4,3]));$
$z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));$
$p[11,1] = 2 * (1 - probnorm(abs(z)));$
$tr1 = 0.5 * log((1 + corr1[5,3])/(1 - corr1[5,3]));$
$tr2 = 0.5 * log((1 + corr2[5,3])/(1 - corr2[5,3]));$
$z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));$
$p[12,1] = 2 * (1 - probnorm(abs(z)));$
$f = sigma1[4,4]/sigma2[4,4];$
$p[13,1] = 1 - probf(f, n1 - 1, n2 - 1);$
$tr1 = 0.5 * log((1 + corr1[5,4])/(1 - corr1[5,4]));$
$tr2 = 0.5 * log((1 + corr2[5,4])/(1 - corr2[5,4]));$
$z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n2 - 3)));$
$p[14,1] = 2 * (1 - probnorm(abs(z)));$
$f = sigma1[5,5]/sigma2[5,5];$
$p[15,1] = 1 - probf(f, n1 - 1, n2 - 1);$
**********;
$f = sigma1[1,1]/sigma3[1,1];$
$p[16,1] = 1 - probf(f, n1 - 1, n3 - 1);$
$tr1 = 0.5 * log((1 + corr1[2,1])/(1 - corr1[2,1]));$
$tr2 = 0.5 * log((1 + corr3[2,1])/(1 - corr3[2,1]));$
$z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n3 - 3)));$
$p[17,1] = 2 * (1 - probnorm(abs(z)));$
$tr1 = 0.5 * log((1 + corr1[3,1])/(1 - corr1[3,1]));$
$tr2 = 0.5 * log((1 + corr3[3,1])/(1 - corr3[3,1]));$
$z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n3 - 3)));$
$p[18,1] = 2 * (1 - probnorm(abs(z)));$
$tr1 = 0.5 * log((1 + corr1[4,1])/(1 - corr1[4,1]));$
$tr2 = 0.5 * log((1 + corr3[4,1])/(1 - corr3[4,1]));$
$z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n3 - 3)));$
$p[19,1] = 2 * (1 - probnorm(abs(z)));$
$tr1 = 0.5 * log((1 + corr1[5,1])/(1 - corr1[5,1]));$
$tr2 = 0.5 * log((1 + corr3[5,1])/(1 - corr3[5,1]));$
$z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n3 - 3)));$
$p[20,1] = 2 * (1 - probnorm(abs(z)));$
$f = sigma1[2,2]/sigma3[2,2];$
$p[21,1] = 1 - probf(f, n1 - 1, n3 - 1);$
$tr1 = 0.5 * log((1 + corr1[3,2])/(1 - corr1[3,2]));$
$tr2 = 0.5 * log((1 + corr3[3,2])/(1 - corr3[3,2]));$
$z = (tr1 - tr2)/sqrt((1/(n1 - 3)) + (1/(n3 - 3)));$
$p[22,1] = 2 * (1 - probnorm(abs(z)));$
$tr1 = 0.5 * log((1 + corr1[4,2])/(1 - corr1[4,2]));$

$tr2 = 0.5 * log((1 + corr3[4,2])/(1 - corr3[4,2]));$
$z = (tr1 - tr2)/sqrt((1/(n1-3)) + (1/(n3-3)));$
$p[23,1] = 2 * (1 - probnorm(abs(z)));$
$tr1 = 0.5 * log((1 + corr1[5,2])/(1 - corr1[5,2]));$
$tr2 = 0.5 * log((1 + corr3[5,2])/(1 - corr3[5,2]));$
$z = (tr1 - tr2)/sqrt((1/(n1-3)) + (1/(n3-3)));$
$p[24,1] = 2 * (1 - probnorm(abs(z)));$
$f = sigma1[3,3]/sigma3[3,3];$
$p[25,1] = 1 - probf(f, n1-1, n3-1);$
$tr1 = 0.5 * log((1 + corr1[4,3])/(1 - corr1[4,3]));$
$tr2 = 0.5 * log((1 + corr3[4,3])/(1 - corr3[4,3]));$
$z = (tr1 - tr2)/sqrt((1/(n1-3)) + (1/(n3-3)));$
$p[26,1] = 2 * (1 - probnorm(abs(z)));$
$tr1 = 0.5 * log((1 + corr1[5,3])/(1 - corr1[5,3]));$
$tr2 = 0.5 * log((1 + corr3[5,3])/(1 - corr3[5,3]));$
$z = (tr1 - tr2)/sqrt((1/(n1-3)) + (1/(n3-3)));$
$p[27,1] = 2 * (1 - probnorm(abs(z)));$
$f = sigma1[4,4]/sigma3[4,4];$
$p[28,1] = 1 - probf(f, n1-1, n3-1);$
$tr1 = 0.5 * log((1 + corr1[5,4])/(1 - corr1[5,4]));$
$tr2 = 0.5 * log((1 + corr3[5,4])/(1 - corr3[5,4]));$
$z = (tr1 - tr2)/sqrt((1/(n1-3)) + (1/(n3-3)));$
$p[29,1] = 2 * (1 - probnorm(abs(z)));$
$f = sigma1[5,5]/sigma3[5,5];$
$p[30,1] = 1 - probf(f, n1-1, n3-1);$
*create p from p [colname = {'p'}];*
*append from p;*
*run;*
*quit;*
*proc sort data = p;*
*by p;*
*run;*
*proc iml;*
*use p;*
*read all var {p} into p;*
$n = nrow(p);$
$imax1 = 0;$
*do i = 1 to n;*
*if p[i,1] <= ((i/30) * 0.05) then imax1 = i;*
*end;*
*if imax1 = 0 then reject&j = 0; else reject&j = 1;*
*create r&j from reject&j [colname = {'r'}];*
*append from reject&j;*
*run;*
*quit;*

```
%end;
proc iml;
use r1;
read all var {r} into r1;
use r2;
read all var {r} into r2;
use r3;
read all var {r} into r3;
use r4;
read all var {r} into r4;
use r5;
read all var {r} into r5;
reject = r1||r2||r3||r4||r5;
create est&t from reject [colname = {'r1' 'r2' 'r3' 'r4' 'r5'}];
append from reject;
run;
quit;
%end;

%mend null;

%null;

%macro cat;
%do i = 1 %to 1000;
est&i
%end;
%mend cat;
data out_null;
set %cat;
i = _n_;
run;
data _null_;
set out_null end = last;
fa = (r1 + r2 + r3 + r4 + r5)/5;
if fa > 0.95 then f + 1;
if last then do; put f; end;
run;
```

To investigate the power of the above method, we generated observations from three multivariate normal distributions with the following three distinct matrices:

$$\boldsymbol{\Sigma}_1 = \begin{bmatrix} 1 & 0.4 & 0.5 & 0.8 & 0.7 \\ 0.4 & 2 & 0.4 & 0.5 & 0.8 \\ 0.5 & 0.4 & 3 & 0.4 & 0.5 \\ 0.8 & 0.5 & 0.4 & 4 & 0.4 \\ 0.7 & 0.8 & 0.5 & 0.4 & 5 \end{bmatrix}, \boldsymbol{\Sigma}_2 = \begin{bmatrix} 1 & 0.4 & 0.5 & 0.8 & 1.7 \\ 0.4 & 2 & 0.4 & 0.5 & 0.8 \\ 0.5 & 0.4 & 3 & 0.4 & 0.5 \\ 0.8 & 0.5 & 0.4 & 4 & 0.4 \\ 1.7 & 0.8 & 0.5 & 0.4 & 5 \end{bmatrix},$$

$$\text{and } \boldsymbol{\Sigma}_3 = \begin{bmatrix} 1 & 0.4 & 0.5 & 0.8 & 0.7 \\ 0.4 & 2 & 0.4 & 0.5 & 2.8 \\ 0.5 & 0.4 & 3 & 0.4 & 0.5 \\ 0.8 & 0.5 & 0.4 & 4 & 0.4 \\ 0.7 & 2.8 & 0.5 & 0.4 & 5 \end{bmatrix}.$$

The respective mean vectors were {1 2 3 4 5}, {−1 −2 −3 −4 −5}, and {0 0 0 0 0}. The power achieved for different sample sizes using Bayesian imputations is given in Table 3.4 below:

**Table 3.4** Power using Bayesian imputations and the FDR method

| | | | Number of imputations | |
|---|---|---|---|---|
| $n1$ | $n2$ | $n3$ | $m = 5$ | $m = 10$ |
| 20 | 25 | 30 | 0.622 | 0.548 |
| 40 | 50 | 60 | 0.965 | 0.954 |
| 50 | 75 | 100 | 0.999 | 0.989 |

The above table demonstrates that while using ten imputations is less powerful than using five imputations, the difference in power decreases as the sample sizes increase. This suggests the following recommendation: If sample sizes are relatively small, then it is better to use a number of imputations larger than 5, say 10, as this will lessen the inflation in the Type I error rate; however, if the sample sizes are relatively large, then five imputations should suffice as this will protect the Type I error rate and will also yield good power. The code that generated Table 3.4 is the same as that which generated Table 3.3, except that the multivariate observations used are from the above 3 alternative distributions.

# Chapter 4
# On Heteroscedastic MANOVA

**Abstract** In this chapter, we introduce three fiducial approaches to heteroscedastic ANOVA and MANOVA. The first approach is that of Li et al. (2011) which was proposed for ANOVA but can be easily generalized to MANOVA. The second approach is that implicit in Behrens (Landw. Jb. **68**, 807–837, 1929) paper. The third approach is that implicit in Fisher (Ann. Eugen. **6**, 391–398, 1935) paper. As a motivation, we begin with the two-sample ANOVA problem to which all the three approaches are applied. As a further motivation, the $k$-sample ANOVA problem is presented where $k > 2$. Finally, we present the heteroscedastic MANOVA problem to which all the three approaches are applied. For the $k$-sample ANOVA problem, $k > 2$, and for the heteroscedastic MANOVA problem, we use the FDR algorithm. Type I errors and power for each method are also presented. Finally, two examples are also presented.

**Keywords** ANOVA • Behrens–Fisher problem • False discovery rate • Heteroscedasticity • MANOVA • Power • Type I errors

Suppose a data analyst wants to test for equality of multivariate mean vectors when there is statistical evidence to believe that the underlying covariance matrices are not equal, but that there is evidence that the distributions generating the data are multivariate normal. This is the multivariate Behrens–Fisher problem. There is considerable literature on this problem, but we focus only on three fiducial approaches. The first one was hinted by Welch in his 1929 paper. The second one was suggested by Fisher in his 1935 paper. To the best of the author's knowledge, there is no literature on a fiducial approach to the multivariate Behrens–Fisher problem. However, in the univariate case (i.e., ANOVA), there is the fiducial approach proposed by Li et al. (2011). This approach can be extended to the multivariate version as we shall see a little later, and this will be our third approach. We present the univariate case below because that will serve as motivation for the multivariate approach.

T. Desai, *A Multiple-Testing Approach to the Multivariate Behrens-Fisher Problem: with Simulations and Examples in SAS®*, SpringerBriefs in Statistics,
DOI 10.1007/978-1-4614-6443-3_4, 

## 4.1 Motivation: $k$-Sample ANOVA, $k = 2$

Before we describe the univariate approach of Li et al. and that which were suggested by Welch and Fisher, we establish some notation. Suppose there are $k$ samples indexed by $i$, $i = 1, \ldots, k$. Let $n_i$ be the sample size, $\overline{x}_i$ be the sample mean, and $s_i^2$ be the unbiased version of the sample variance, $i = 1, \ldots, k$. The approach of Li et al. is as follows:

(a) Compute $R_0 = \sum_{i=1}^{k} \frac{n_i \overline{x}_i^2}{s_i^2} - \left[ \left( \sum_{i=1}^{k} \frac{n_i \overline{x}_i}{s_i^2} \right)^2 / \left( \sum_{i=1}^{k} \frac{n_i}{s_i^2} \right) \right]$.

(b) For some predecided $M$, perform the following operations for $j = 1, \ldots, M$:

- For $i = 1, \ldots, k$, generate $t_i$ from Student's $t$ distribution with $n_i - 1$ degrees of freedom.
- Compute $R_j = \sum_{i=1}^{k} t_i^2 - \left[ \left( \sum_{i=1}^{k} \frac{\sqrt{n_i} t_i}{s_i} \right)^2 / \left( \sum_{i=1}^{k} \frac{n_i}{s_i^2} \right) \right]$.
- If $R_j \geq R_0$, then $W_j = 1$; else $W_j = 0$.
- Let $\widehat{p} = \sum_{j=1}^{M} W_j \ / M$.

Then $\widehat{p}$ is a simulated value of the $p$-value of the above test. This simulation of $p$-values lie at the heart of fiducial approaches to ANOVA and MANOVA. We will refer to the method of Li et al. as method A. Next, we present the approach suggested by Welch for the two-sample case (it can be extended to the multisample case as we shall see later). Suppose we have two independent samples of data from two unknown but normal distributions. Suppose the unknown means are $\mu_1$ and $\mu_2$. Then it is a known fact that

$$T_1 = \frac{\mu_1 - \overline{x}_1}{s_1/\sqrt{n_1}} \backsim t_{n_1 - 1} \text{ and } T_2 = \frac{\mu_2 - \overline{x}_2}{s_2/\sqrt{n_2}} \backsim t_{n_2 - 1}$$

Now let

$$W = \frac{\mu_1 - \mu_2 - (\overline{x}_1 - \overline{x}_2)}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \text{ and } \tan(\theta) = \frac{s_1/\sqrt{n_1}}{s_2/\sqrt{n_2}}$$

It then follows that

$$W = T_1 \sin(\theta) - T_2 \cos(\theta)$$

Note that under the null hypothesis of no difference in means, it follows that

$$W^2 = \frac{(\overline{x}_1 - \overline{x}_2)^2}{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$$

Method B is as follows:

(a) Let $\theta = \left|\tan^{-1}\left(\frac{s_1/\sqrt{n_1}}{s_2/\sqrt{n_2}}\right)\right|$. Let $c = 0$. For some predecided, suitably large $M$, perform the following operations for $j = 1, \ldots, M$.
(b) Generate $t_1$ and $t_2$ randomly from Student's t distribution with $n_i - 1$ degrees of freedom, $i = 1, 2$, respectively.
(c) Compute

$$W_j^2 = (T_1 \sin(\theta) - T_2 \cos(\theta))^2.$$

If $W_j^2 > W^2$ then $c = c + 1$.
(d) At the end of $M$ iterations as described above, the simulated $p$-value is $c/M$.

Method C is conceptually even simpler. Note that

$$\mu_1 - \mu_2 - (\overline{x}_1 - \overline{x}_2) = \left(t_{n_1-1}\frac{s_1}{\sqrt{n_1}}\right) - \left(t_{n_2-1}\frac{s_2}{\sqrt{n_2}}\right)$$

Let

$$D = \mu_1 - \mu_2 - (\overline{x}_1 - \overline{x}_2)$$

Under the null hypothesis of no difference in means, it follows that

$$D^2 = (\overline{x}_1 - \overline{x}_2)^2$$

is the observed value of the $D^2$ statistic under the null of no difference in means. Then method C proceeds as follows:

(a) Let $c = 0$. For some predecided, suitably large $M$, perform the following operations for $j = 1, \ldots, M$.
(b) Generate $t_1$ and $t_2$ randomly from Student's t distribution with $n_i - 1$ degrees of freedom, $i = 1, 2$, respectively.
(c) Compute

$$D_j^2 = \left(\left(t_1\frac{s_1}{\sqrt{n_1}}\right) - \left(t_2\frac{s_2}{\sqrt{n_2}}\right)\right)^2$$

If $D_j^2 > D^2$, then $c = c + 1$.
(d) At the end of M iterations, the simulated p-value is $c/M$.

Table 4.1 below presents the Type I error rates under the null. The two samples used in each simulation were generated from $N(0, 1)$ and $N(0, 2)$.

Table 4.1 exhibits a quaint property: when the two samples are equal, the Type I error reported is 0.000 for all the three methods. When there is imbalance in the sample sizes, the reported Type I error is close to the nominal level of 0.05.

**Table 4.1** Type I error rates for the above three methods: two-sample complete-data case

| Sample size | | Method | | |
|---|---|---|---|---|
| $n_1$ | $n_2$ | A | B | C |
| 5 | 10 | 0.029 | 0.038 | 0.036 |
| 10 | 10 | 0.000 | 0.000 | 0.000 |
| 10 | 15 | 0.028 | 0.033 | 0.033 |
| 20 | 30 | 0.033 | 0.040 | 0.043 |
| 40 | 60 | 0.032 | 0.043 | 0.044 |
| 100 | 100 | 0.000 | 0.000 | 0.000 |
| 160 | 240 | 0.033 | 0.037 | 0.037 |
| 320 | 480 | 0.031 | 0.039 | 0.040 |
| 500 | 500 | 0.000 | 0.000 | 0.000 |
| 640 | 960 | 0.040 | 0.046 | 0.046 |
| 1000 | 1000 | 0.000 | 0.000 | 0.000 |

The SAS® program that generated Table 4.1 is as follows:

```
data one;
do i = 1 to 1000000;
x = 0 + (sqrt(1) * normal(i));
output;
end;
run;
data two;
do i = 1 to 1000000;
x = 0 + (sqrt(2) * normal(i));
output;
end;
run;
%macro test;
%do t = 1 %to 1000;
ods listing close;
proc iml;
index1 = %sysevalf(((&t - 1) * 60) + 1) : %sysevalf(&t * 60);
use one;
read point index1 var {x} into x1;
create x1 from x1 [colname = {'x'}];
append from x1;
index2 = %sysevalf(((&t - 1) * 50) + 1) : %sysevalf(&t * 50);
use two;
read point index2 var {x} into x2;
create x2 from x2 [colname = {'x'}];
append from x2;
run;
quit;
proc iml;
```

```
use x1;
read all var {x} into x1;
n1 = nrow(x1);
use x2;
read all var {x} into x2;
n2 = nrow(x2);
mu1_ = (sum(x1[, 1])/n1);
mu2_ = (sum(x2[, 1])/n2);

mu1 = j(n1, 1, 1)@mu1_;
mu2 = j(n2, 1, 1)@mu2_;
sigma1 = (x1 − mu1)‘ ∗ (x1 − mu1)/(n1 − 1);
sigma2 = (x2 − mu2)‘ ∗ (x2 − mu2)/(n2 − 1);

diff_ = ((mu1[1, 1] − mu2[1, 1]) ∗ ∗2);
r_ = ((n1 ∗ (mu1[1, 1] ∗ ∗2)/sigma1[1, 1]) + (n2 ∗ (mu2[1, 1] ∗ ∗2)/
     sigma2[1, 1]))
−((((n1 ∗ mu1[1, 1]/sigma1[1, 1]) + (n2 ∗ mu2[1, 1]/sigma2[1, 1])) ∗ ∗2)
/((n1/sigma1[1, 1]) + (n2/sigma2[1, 1])));
bf_ = (((mu1[1, 1] − mu2[1, 1]) ∗ ∗2)/((sigma1[1, 1]/n1)
      +(sigma2[1, 1]/n2)));
c1 = 0; c2 = 0; c3 = 0;
do j = 1 to 1000;


t1 = tinv(ranuni(7 ∗ j), n1 − 1);
t2 = tinv(ranuni(10 ∗ j), n2 − 1);
diff = (((sqrt(sigma1[1, 1]/n1) ∗ t1) − (sqrt(sigma2[1, 1]/n2) ∗ t2)) ∗ ∗2);
if diff > diff_then c1 = c1 + 1;
t1 = tinv(ranuni(10 ∗ j), n1 − 1);
t2 = tinv(ranuni(11 ∗ j), n2 − 1);
r = ((t1 ∗ ∗2) + (t2 ∗ ∗2))
−((((sqrt(n1/sigma1[1, 1]) ∗ t1) + (sqrt(n2/sigma2[1, 1]) ∗ t2)) ∗ ∗2)
/((n1/sigma1[1, 1]) + (n2/sigma2[1, 1])));
if r >= r_then c2 = c2 + 1;
t1 = tinv(ranuni(7 ∗ j), n1 − 1);
t2 = tinv(ranuni(10 ∗ j), n2 − 1);
theta = abs(atan(sqrt(sigma1[1, 1] ∗ n2/(sigma2[1, 1] ∗ n1))));
bf = (((t1 ∗ sin(theta)) − (t2 ∗ cos(theta))) ∗ ∗2);
ifbf > bf_thenc3 = c3 + 1;
end;
p1 = c1/1000; p2 = c2/1000; p3 = c3/1000;
est = p1||p2||p3;
create est&t from est [colname = {‘p1‘ ‘p2‘ ‘p3‘}];
append from est;
run;
quit;
```

```
%end;
%mend test;
%test;
%macro cat;
%do i = 1 %to 1000;
est&i
%end;
%mend cat;
data concat;
set %cat;
i = _n_;
run;
data _null_;
set concat end = last;
if p2 < 0.05 then c1 + 1;
if p3 < 0.05 then c2 + 1;
if p1 < 0.05 then c3 + 1;
if last then
do;
put c1 c2 c3;
end;
run;
```

To investigate power, two samples were generated from $N(0, 1)$ and $N(1, 2)$. Power of the three procedures is illustrated in Table 4.2 below:

**Table 4.2** Power under the alternative: two-sample complete-data case

| Sample size | | Method | | |
|---|---|---|---|---|
| $n_1$ | $n_2$ | A | B | C |
| 5 | 10 | 0.201 | 0.232 | 0.228 |
| 10 | 15 | 0.432 | 0.479 | 0.488 |
| 15 | 20 | 0.594 | 0.640 | 0.641 |
| 50 | 40 | 0.945 | 0.956 | 0.953 |
| 60 | 50 | 0.984 | 0.990 | 0.988 |

Table 4.2 suggests that methods B and C may be more powerful than method A, at least as far as two-sample comparisons are concerned. Also, in most cases, method C is only slightly less powerful than method B. The code that generated the above table is similar to the one that generated Table 4.1 except that we replace the second data step with:

```
data two;
do i = 1 to 1000000;
x = 1 + (sqrt(2) * normal(i));
output;
end;
run;
```

## 4.2 Further Motivation: $k$-Sample ANOVA, $k > 2$

### *4.2.1 The Complete-Data Case*

Consider the $k$ univariate normal means: $\mu_1, \ldots, \mu_k$. We want to test the following hypothesis:

$$H_0 : \mu_1 = \ldots = \mu_k$$

Method A is applicable to an arbitrary number of univariate samples, and so it doesn't require any modification. To use methods B or C, we conduct the $\binom{k}{2}$ distinct pairwise tests using methods B or C. This will yield us $\binom{k}{2}$ $p$-values which we then subject to the FDR algorithm. The FDR algorithm will in turn give us a decision as to whether to reject or do not reject the null hypothesis above.

Suppose we generate observations from $N(0, 1)$, $N(0, 1.5)$, $N(0, 2)$, $N(0, 2.5)$, and $N(0, 3)$. Then the Type I errors corresponding to various sample size combinations are given in Table 4.3 below:

**Table 4.3** Type I errors in the five-sample complete-data case

| Sample sizes | | | | | Method | | |
|---|---|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | $n_4$ | $n_5$ | A | B | C |
| 10 | 20 | 30 | 40 | 50 | 0.052 | 0.051 | 0.056 |
| 50 | 10 | 20 | 30 | 40 | 0.051 | 0.050 | 0.052 |
| 40 | 50 | 10 | 20 | 30 | 0.053 | 0.060 | 0.043 |
| 30 | 40 | 50 | 10 | 20 | 0.051 | 0.053 | 0.049 |
| 20 | 30 | 40 | 50 | 10 | 0.058 | 0.058 | 0.062 |
| 30 | 10 | 40 | 20 | 50 | 0.042 | 0.051 | 0.059 |

The SAS® code that generated the above table is as follows:

```
data one;
do i = 1 to 1000000;
x = 0 + (sqrt(1) * normal(i));
output;
end;
run;
data two;
do i = 1 to 1000000;
x = 0 + (sqrt(1.5) * normal(i));
output;
end;
run;
data three;
do i = 1 to 1000000;
```

```
x = 0 + (sqrt(2) * normal(i));
output;
end;
run;
data four;
do i = 1 to 1000000;
x = 0 + (sqrt(2.5) * normal(i));
output;
end;
run;
data five;
do i = 1 to 1000000;
x = 0 + (sqrt(3) * normal(i));
output;
end;
run;
%macro test;
%do t = 1 %to 1000;
ods listing close;
proc iml;
index1 = %sysevalf(((&t - 1) * 20) + 1) : %sysevalf(&t * 20);
use one;
read point index1 var {x} into x1;
create x1 from x1 [colname = {'x'}];
append from x1;
index2 = %sysevalf(((&t - 1) * 50) + 1) : %sysevalf(&t * 50);
use two;
read point index2 var {x} into x2;
create x2 from x2 [colname = {'x'}];
append from x2;
index3 = %sysevalf(((&t - 1) * 40) + 1) : %sysevalf(&t * 40);
use three;
read point index3 var {x} into x3;
create x3 from x3 [colname = {'x'}];
append from x3;
index4 = %sysevalf(((&t - 1) * 30) + 1) : %sysevalf(&t * 30);
use four;
read point index4 var {x} into x4;
create x4 from x4 [colname = {'x'}];
append from x4;
index5 = %sysevalf(((&t - 1) * 10) + 1) : %sysevalf(&t * 10);
use five;
read point index5 var {x} into x5;
create x5 from x5 [colname = {'x'}];
append from x5;
```

```
run;
quit;
proc iml;
use x1;
read all var {x} into x1;
n1 = nrow(x1);
use x2;
read all var {x} into x2;
n2 = nrow(x2);
use x3;
read all var {x} into x3;
n3 = nrow(x3);
use x4;
read all var {x} into x4;
n4 = nrow(x4);
use x5;
read all var {x} into x5;
n5 = nrow(x5);
mu1_ = (sum(x1[, 1])/n1);
mu2_ = (sum(x2[, 1])/n2);
mu3_ = (sum(x3[, 1])/n3);
mu4_ = (sum(x4[, 1])/n4);
mu5_ = (sum(x5[, 1])/n5);

mu1 = j(n1, 1, 1)@mu1_;
mu2 = j(n2, 1, 1)@mu2_;
mu3 = j(n3, 1, 1)@mu3_;
mu4 = j(n4, 1, 1)@mu4_;
mu5 = j(n5, 1, 1)@mu5_;
sigma1 = (x1 − mu1)‘ ∗ (x1 − mu1)/(n1 − 1);
sigma2 = (x2 − mu2)‘ ∗ (x2 − mu2)/(n2 − 1);
sigma3 = (x3 − mu3)‘ ∗ (x3 − mu3)/(n3 − 1);
sigma4 = (x4 − mu4)‘ ∗ (x4 − mu4)/(n4 − 1);
sigma5 = (x5 − mu5)‘ ∗ (x5 − mu5)/(n5 − 1);

diff1_ = (mu1_ − mu2_) ∗ ∗2;
diff2_ = (mu1_ − mu3_) ∗ ∗2;
diff3_ = (mu1_ − mu4_) ∗ ∗2;
diff4_ = (mu1_ − mu5_) ∗ ∗2;
diff5_ = (mu2_ − mu3_) ∗ ∗2;
diff6_ = (mu2_ − mu4_) ∗ ∗2;
diff7_ = (mu2_ − mu5_) ∗ ∗2;
diff8_ = (mu3_ − mu4_) ∗ ∗2;
diff9_ = (mu3_ − mu5_) ∗ ∗2;
diff10_ = (mu4_ − mu5_) ∗ ∗2;
```

$$
\begin{aligned}
&r\_ = ((n1 * (mu1[1,1] * *2)/sigma1[1,1]) + (n2 * (mu2[1,1] * *2)/\\
&\quad sigma2[1,1]) + (n3 * (mu3[1,1] * *2)/sigma3[1,1])\\
&\quad +(n4 * (mu4[1,1] * *2)/sigma4[1,1]) + (n5 * (mu5[1,1] * *2)/\\
&\quad sigma5[1,1]))\\
&-((((n1 * mu1[1,1]/sigma1[1,1]) + (n2 * mu2[1,1]/sigma2[1,1]) + (n3 *\\
&mu3[1,1]/sigma3[1,1]) + (n4 * mu4[1,1]/sigma4[1,1])\\
&+(n5 * mu5[1,1]/sigma5[1,1])) * *2)\\
&/((n1/sigma1[1,1]) + (n2/sigma2[1,1]) + (n3/sigma3[1,1])\\
&\quad +(n4/sigma4[1,1]) + (n5/sigma5[1,1])));\\
&bf1\_ = (((mu1\_ - mu2\_) * *2)/((sigma1/n1) + (sigma2/n2)));\\
&bf2\_ = (((mu1\_ - mu3\_) * *2)/((sigma1/n1) + (sigma3/n3)));\\
&bf3\_ = (((mu1\_ - mu4\_) * *2)/((sigma1/n1) + (sigma4/n4)));\\
&bf4\_ = (((mu1\_ - mu5\_) * *2)/((sigma1/n1) + (sigma5/n5)));\\
&bf5\_ = (((mu2\_ - mu3\_) * *2)/((sigma2/n2) + (sigma3/n3)));\\
&bf6\_ = (((mu2\_ - mu4\_) * *2)/((sigma2/n2) + (sigma4/n4)));\\
&bf7\_ = (((mu2\_ - mu5\_) * *2)/((sigma2/n2) + (sigma5/n5)));\\
&bf8\_ = (((mu3\_ - mu4\_) * *2)/((sigma3/n3) + (sigma4/n4)));\\
&bf9\_ = (((mu3\_ - mu5\_) * *2)/((sigma3/n3) + (sigma5/n5)));\\
&bf10\_ = (((mu4\_ - mu5\_) * *2)/((sigma4/n4) + (sigma5/n5)));\\
&c1\_1 = 0; c1\_2 = 0; c1\_3 = 0; c1\_4 = 0; c1\_5 = 0; c1\_6 = 0; c1\_7 = 0; c1\_8 =\\
&0; c1\_9 = 0; c1\_10 = 0;\\
&c2 = 0;\\
&c3\_1 = 0; c3\_2 = 0; c3\_3 = 0; c3\_4 = 0; c3\_5 = 0; c3\_6 = 0; c3\_7 = 0; c3\_8 =\\
&0; c3\_9 = 0; c3\_10 = 0;\\
&do\ j = 1\ to\ 1000;\\
&t1 = tinv(ranuni(7 * j), n1 - 1);\\
&t2 = tinv(ranuni(10 * j), n2 - 1);\\
&diff1 = (((sqrt(sigma1/n1) * t1) - (sqrt(sigma2/n2) * t2)) * *2);\\
&if\ diff1 > diff1\_\ then\ c1\_1 = c1\_1 + 1;\\
&t1 = tinv(ranuni(7 * j), n1 - 1);\\
&t3 = tinv(ranuni(10 * j), n3 - 1);\\
&diff2 = (((sqrt(sigma1/n1) * t1) - (sqrt(sigma3/n3) * t3)) * *2);\\
&if\ diff2 > diff2\_\ then\ c1\_2 = c1\_2 + 1;\\
&t1 = tinv(ranuni(7 * j), n1 - 1);\\
&t4 = tinv(ranuni(10 * j), n4 - 1);\\
&diff3 = (((sqrt(sigma1/n1) * t1) - (sqrt(sigma4/n4) * t4)) * *2);\\
&if\ diff3 > diff3\_\ then\ c1\_3 = c1\_3 + 1;\\
&t1 = tinv(ranuni(7 * j), n1 - 1);\\
&t5 = tinv(ranuni(10 * j), n5 - 1);\\
&diff4 = (((sqrt(sigma1/n1) * t1) - (sqrt(sigma5/n5) * t5)) * *2);\\
&if\ diff4 > diff4\_\ then\ c1\_4 = c1\_4 + 1;\\
&t2 = tinv(ranuni(7 * j), n2 - 1);\\
&t3 = tinv(ranuni(10 * j), n3 - 1);\\
&diff5 = (((sqrt(sigma2/n2) * t2) - (sqrt(sigma3/n3) * t3)) * *2);\\
&if\ diff5 > diff5\_\ then\ c1\_5 = c1\_5 + 1;
\end{aligned}
$$

$t2 = tinv(ranuni(7 * j), n2 - 1);$
$t4 = tinv(ranuni(10 * j), n4 - 1);$
$diff6 = (((sqrt(sigma2/n2) * t2) - (sqrt(sigma4/n4) * t4)) * *2);$
$if\ diff6 > diff6\_then\ c1\_6 = c1\_6 + 1;$
$t2 = tinv(ranuni(7 * j), n2 - 1);$
$t5 = tinv(ranuni(10 * j), n5 - 1);$
$diff7 = (((sqrt(sigma2/n2) * t2) - (sqrt(sigma5/n5) * t5)) * *2);$
$if\ diff7 > diff7\_then\ c1\_7 = c1\_7 + 1;$
$t3 = tinv(ranuni(7 * j), n3 - 1);$
$t4 = tinv(ranuni(10 * j), n4 - 1);$
$diff8 = (((sqrt(sigma3/n3) * t3) - (sqrt(sigma4/n4) * t4)) * *2);$
$if\ diff8 > diff8\_then\ c1\_8 = c1\_8 + 1;$
$t3 = tinv(ranuni(7 * j), n3 - 1);$
$t5 = tinv(ranuni(10 * j), n5 - 1);$
$diff9 = (((sqrt(sigma3/n3) * t3) - (sqrt(sigma5/n5) * t5)) * *2);$
$if\ diff9 > diff9\_then\ c1\_9 = c1\_9 + 1;$
$t4 = tinv(ranuni(7 * j), n4 - 1);$
$t5 = tinv(ranuni(10 * j), n5 - 1);$
$diff10 = (((sqrt(sigma4/n4) * t4) - (sqrt(sigma5/n5) * t5)) * *2);$
$if\ diff10 > diff10\_then\ c1\_10 = c1\_10 + 1;$

$t1 = tinv(ranuni(10 * j), n1 - 1);$
$t2 = tinv(ranuni(11 * j), n2 - 1);$
$t3 = tinv(ranuni(12 * j), n3 - 1);$
$t4 = tinv(ranuni(13 * j), n4 - 1);$
$t5 = tinv(ranuni(14 * j), n5 - 1);$
$r = ((t1 * *2) + (t2 * *2) + (t3 * *2) + (t4 * *2) + (t5 * *2))$
$-((((sqrt(n1/sigma1) * t1) + (sqrt(n2/sigma2) * t2)$
$+(sqrt(n3/sigma3) * t3)$
$+(sqrt(n4/sigma4) * t4) + (sqrt(n5/sigma5) * t5)) * *2)/$
$((n1/sigma1) + (n2/sigma2) + (n3/sigma3) + (n4/sigma4)$
$+(n5/sigma5)));$
$if\ r >= r\_then\ c2 = c2 + 1;$
$t1 = tinv(ranuni(7 * j), n1 - 1);$
$t2 = tinv(ranuni(10 * j), n2 - 1);$
$theta = abs(atan(sqrt(sigma1 * n2/(sigma2 * n1))));$
$bf1 = (((t1 * sin(theta)) - (t2 * cos(theta))) * *2);$
$if\ bf1 > bf1\_then\ c3\_1 = c3\_1 + 1;$
$t1 = tinv(ranuni(7 * j), n1 - 1);$
$t3 = tinv(ranuni(10 * j), n3 - 1);$
$theta = abs(atan(sqrt(sigma1 * n3/(sigma3 * n1))));$
$bf2 = (((t1 * sin(theta)) - (t3 * cos(theta))) * *2);$
$if\ bf2 > bf2\_then\ c3\_2 = c3\_2 + 1;$
$t1 = tinv(ranuni(7 * j), n1 - 1);$
$t4 = tinv(ranuni(10 * j), n4 - 1);$

$theta = abs(atan(sqrt(sigma1 * n4/(sigma4 * n1))));$
$bf3 = (((t1 * sin(theta)) - (t4 * cos(theta))) * *2);$
$if\ bf3 > bf3\_then\ c3\_3 = c3\_3 + 1;$
$t1 = tinv(ranuni(7 * j), n1 - 1);$
$t5 = tinv(ranuni(10 * j), n5 - 1);$
$theta = abs(atan(sqrt(sigma1 * n5/(sigma5 * n1))));$
$bf4 = (((t1 * sin(theta)) - (t5 * cos(theta))) * *2);$
$if\ bf4 > bf4\_then\ c3\_4 = c3\_4 + 1;$
$t2 = tinv(ranuni(7 * j), n2 - 1);$
$t3 = tinv(ranuni(10 * j), n3 - 1);$
$theta = abs(atan(sqrt(sigma2 * n3/(sigma3 * n2))));$
$bf5 = (((t2 * sin(theta)) - (t3 * cos(theta))) * *2);$
$if\ bf5 > bf5\_then\ c3\_5 = c3\_5 + 1;$
$t2 = tinv(ranuni(7 * j), n2 - 1);$
$t4 = tinv(ranuni(10 * j), n4 - 1);$
$theta = abs(atan(sqrt(sigma2 * n4/(sigma4 * n2))));$
$bf6 = (((t2 * sin(theta)) - (t4 * cos(theta))) * *2);$
$if\ bf6 > bf6\_then\ c3\_6 = c3\_6 + 1;$
$t2 = tinv(ranuni(7 * j), n2 - 1);$
$t5 = tinv(ranuni(10 * j), n5 - 1);$
$theta = abs(atan(sqrt(sigma2 * n5/(sigma5 * n2))));$
$bf7 = (((t2 * sin(theta)) - (t5 * cos(theta))) * *2);$
$if\ bf7 > bf7\_then\ c3\_7 = c3\_7 + 1;$
$t3 = tinv(ranuni(7 * j), n3 - 1);$
$t4 = tinv(ranuni(10 * j), n4 - 1);$
$theta = abs(atan(sqrt(sigma3 * n4/(sigma4 * n3))));$
$bf8 = (((t3 * sin(theta)) - (t4 * cos(theta))) * *2);$
$if\ bf8 > bf8\_then\ c3\_8 = c3\_8 + 1;$
$t3 = tinv(ranuni(7 * j), n3 - 1);$
$t5 = tinv(ranuni(10 * j), n5 - 1);$
$theta = abs(atan(sqrt(sigma3 * n5/(sigma5 * n3))));$
$bf9 = (((t3 * sin(theta)) - (t5 * cos(theta))) * *2);$
$if\ bf9 > bf9\_then\ c3\_9 = c3\_9 + 1;$
$t4 = tinv(ranuni(7 * j), n4 - 1);$
$t5 = tinv(ranuni(10 * j), n5 - 1);$
$theta = abs(atan(sqrt(sigma4 * n5/(sigma5 * n4))));$
$bf10 = (((t4 * sin(theta)) - (t5 * cos(theta))) * *2);$
$if\ bf10 > bf10\_then\ c3\_10 = c3\_10 + 1;$
$end;$
$p1 = j(10, 1, 0);$
$p1[1, 1] = c1\_1/1000;\ p1[2, 1] = c1\_2/1000;\ p1[3, 1] = c1\_3/1000;\ p1[4, 1] = c1\_4/1000;\ p1[5, 1] = c1\_5/1000;$
$p1[6, 1] = c1\_6/1000;\ p1[7, 1] = c1\_7/1000;\ p1[8, 1] = c1\_8/1000;\ p1[9, 1] = c1\_9/1000;\ p1[10, 1] = c1\_10/1000;$
$p2 = c2/1000;$

```
if p2 < 0.05 then r2 = 1; else r2 = 0;
p3 = j(10, 1, 0);
p3[1, 1] = c3_1/1000; p3[2, 1] = c3_2/1000; p3[3, 1]
  = c3_3/1000; p3[4, 1] = c3_4/1000; p3[5, 1] = c3_5/1000;
p3[6, 1] = c3_6/1000; p3[7, 1] = c3_7/1000; p3[8, 1] = c3_8/1000; p3[9, 1]
  = c3_9/1000; p3[10, 1] = c3_10/1000;
create p1 from p1 [colname = {'p'}];
append from p1;
create r2 from r2 [colname = {'r2'}];
append from r2;
create p3 from p3 [colname = {'p'}];
append from p3;
run;
quit;
proc sort data = p1;
by p;
run;
proc sort data = p3;
by p;
run;
proc iml;
use p1;
read all var {p} into p1;
use p3;
read all var {p} into p3;
n1 = nrow(p1); n3 = nrow(p3);
imax1 = 0;
do i = 1 to n1;
if p1[i, 1] <= ((i/10) * 0.05) then imax1 = i;
end;
if imax1 = 0 then r1 = 0; else r1 = 1;
imax3 = 0;
do i = 1 to n3;
if p3[i, 1] <= ((i/10) * 0.05) then imax3 = i;
end;
if imax3 = 0 then r3 = 0; else r3 = 1;
use r2;
read all var {r2} into r2;
est = r1||r2||r3;
create est&t from est [colname = {'r1' 'r2' 'r3'}];
append from est;
run;
quit;
%end;
%mend test;
```

```
%test;
%macro cat;
%do i = 1 %to 1000;
est&i
%end;
%mend cat;
data concat;
set %cat;
i = _n_;
run;
data _null_;
set concat end = last;
if r2 = 1 then c1 + 1;
if r3 = 1 then c2 + 1;
if r1 = 1 then c3 + 1;
if last then do;
put c1 c2 c3;
end;
run;
```

The power of the two methods is displayed in Table 4.4 below. The alternative considered is obtained by generating observations from $N(0, 1)$, $N(0, 1.5)$, $N(0, 2)$, $N(0, 2.5)$, and $N(1, 3)$.

**Table 4.4** Power in the five-sample complete-data case

| Sample sizes | | | | | Method | | |
|---|---|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | $n_4$ | $n_5$ | A | B | C |
| 10 | 20 | 30 | 40 | 50 | 0.742 | 0.763 | 0.775 |
| 50 | 10 | 20 | 30 | 40 | 0.678 | 0.716 | 0.723 |
| 40 | 50 | 10 | 20 | 30 | 0.545 | 0.628 | 0.581 |
| 30 | 40 | 50 | 10 | 20 | 0.382 | 0.437 | 0.457 |
| 20 | 30 | 40 | 50 | 10 | 0.225 | 0.204 | 0.217 |
| 30 | 10 | 40 | 20 | 50 | 0.730 | 0.772 | 0.795 |
| 50 | 20 | 40 | 10 | 30 | 0.537 | 0.604 | 0.596 |
| 40 | 10 | 30 | 50 | 20 | 0.374 | 0.436 | 0.460 |
| 10 | 30 | 20 | 50 | 40 | 0.673 | 0.712 | 0.746 |
| 20 | 50 | 40 | 30 | 10 | 0.230 | 0.198 | 0.212 |

The above table suggests that although methods B and C are not uniformly better than method A in terms of power, they are strong contenders against method A when it comes to power against an alternative. The code that generated the above table is the same as that which generated Table 4.3, except that we replace the fifth data step with:

```
data five;
do i = 1 to 1000000;
```

$x = 1 + (sqrt(3) * normal(i));$
$output;$
$end;$
$run;$

### 4.2.2 The Randomly-Incomplete-Data Case

Suppose we generate observations from $N(0, 1)$, $N(0, 1.5)$, $N(0, 2)$, $N(0, 2.5)$, and $N(0, 3)$, except that we set an observation to missing with probabilities 0.20, 0.25, 0.3, 0.15, and 0.10, respectively. We then perform our five-sample ANOVA using only the observed data. The type I errors are given in Table 4.5 below:

**Table 4.5** Type I errors in the five-sample randomly-incomplete-data case

| Sample sizes | | | | | Method | | |
|---|---|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | $n_4$ | $n_5$ | A | B | C |
| 10 | 20 | 30 | 40 | 50 | 0.030 | 0.028 | 0.030 |
| 50 | 10 | 20 | 30 | 40 | 0.028 | 0.032 | 0.034 |
| 40 | 50 | 10 | 20 | 30 | 0.029 | 0.031 | 0.035 |
| 30 | 40 | 50 | 10 | 20 | 0.032 | 0.035 | 0.037 |
| 20 | 30 | 40 | 50 | 10 | 0.036 | 0.032 | 0.036 |
| 30 | 10 | 40 | 20 | 50 | 0.030 | 0.026 | 0.029 |

The power of the two methods is displayed in Table 4.6 below. The alternative considered is the same as in the complete-data case. Fractions of missingness are the same as in the null case above.

**Table 4.6** Power in the five-sample randomly-incomplete-data case

| Sample sizes | | | | | Method | | |
|---|---|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | $n_4$ | $n_5$ | A | B | C |
| 10 | 20 | 30 | 40 | 50 | 0.629 | 0.658 | 0.685 |
| 50 | 10 | 20 | 30 | 40 | 0.519 | 0.619 | 0.632 |
| 40 | 50 | 10 | 20 | 30 | 0.399 | 0.524 | 0.481 |
| 30 | 40 | 50 | 10 | 20 | 0.298 | 0.357 | 0.372 |
| 20 | 30 | 40 | 50 | 10 | 0.201 | 0.172 | 0.183 |
| 30 | 10 | 40 | 20 | 50 | 0.597 | 0.682 | 0.718 |
| 50 | 20 | 40 | 10 | 30 | 0.427 | 0.498 | 0.512 |
| 40 | 10 | 30 | 50 | 20 | 0.279 | 0.342 | 0.358 |
| 10 | 30 | 20 | 50 | 40 | 0.547 | 0.617 | 0.639 |
| 20 | 50 | 40 | 30 | 10 | 0.199 | 0.165 | 0.187 |

Table 4.6 demonstrates that, just as in the complete-data case, methods B and C prove to be strong contenders in terms of power vis-a-vis method A. The code that generated Tables 4.5 and 4.6 is left to the reader as an exercise.

### 4.2.3 Example: Temperature Recovery Times

This example and accompanying data appears in Westfall et al. (1999). Following a surgical procedure, when anesthesia wears off, the temperature of a patient may dip. To maintain the body temperature at an acceptable level, a company manufactured specialized heating blankets. Four types of blankets were tried on surgical patients. One of the four blankets was a standard one which was already in use in various hospitals. The company's interest was to compare the recovery times of patients using the four different blankets. The data are as follows:

```
data blanket;
input blanket minutes @@;
cards;
1 15 1 13 1 12 1 16 1 16 1 17 1 13 1 13 1 16 1 17
1 17 1 19 1 17 1 15 1 13 1 12 1 16 1 10 1 17 1 12
2 13 2 16 2 9
3 5 3 8 3 9
4 14 4 16 4 16 4 12 4 7 4 12 4 13 4 13 4 9 4 16
4 13 4 18 4 13 4 12 4 13
;
run;
```

The analysis of the above data returns the values $r1 = 1$, $r2 = 0$, and $r3 = 1$. Note that $r1$ corresponds to method C, $r2$ to method A, and $r3$ to method B. Thus, method A fails to reject equality of means, whereas method B and C reject the equality of means.

## 4.3 Heteroscedastic MANOVA: The Multivariate Behrens–Fisher Problem

Consider the $k$ $p$-variate normal mean vectors: $\boldsymbol{\mu}_1 = (\mu_{11}, \ldots, \mu_{1p}), \ldots, \boldsymbol{\mu}_k = (\mu_{k1}, \ldots, \mu_{kp})$. We want to test the following hypothesis:

$$H_0 : \boldsymbol{\mu}_1 = \ldots = \boldsymbol{\mu}_k$$

Our approach is conceptually simple. To use approach A, we apply method A to the $k$ hypotheses:

$$H_{01} : \mu_{11} = \ldots = \mu_{k1}, \ldots H_{0k} : \mu_{1p} = \ldots = \mu_{kp}$$

To use method A, we apply it to each of the $p$ hypotheses above. This will yield us $p$ $p$-values, which we subject to the FDR algorithm. The FDR algorithm will then give us a decision as to whether to reject or not reject the equality of the $k$ $p$-variate

mean vectors. To use methods B or C, we conduct the $p \times \binom{k}{2}$ distinct pairwise tests using methods B or C. This will yield us $p \times \binom{k}{2}$ $p$-values which we then subject to the FDR algorithm. The FDR algorithm will in turn give us a decision as to whether to reject or do not reject the null hypothesis above.

### 4.3.1 The Complete-Data Case

To investigate the Type I errors of the above method, the following three $5 \times 5$ covariance matrices were used:

$$\boldsymbol{\Sigma}_1 = \begin{bmatrix} 1 & 0.4 & 0.5 & 0.8 & 0.7 \\ 0.4 & 2 & 0.4 & 0.5 & 0.8 \\ 0.5 & 0.4 & 3 & 0.4 & 0.5 \\ 0.8 & 0.5 & 0.4 & 4 & 0.4 \\ 0.7 & 0.8 & 0.5 & 0.4 & 5 \end{bmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{bmatrix} 1.5 & 0.4 & 0.5 & 0.8 & 1.7 \\ 0.4 & 2.5 & 0.4 & 0.5 & 0.8 \\ 0.5 & 0.4 & 3.5 & 0.4 & 0.5 \\ 0.8 & 0.5 & 0.4 & 4 & 0.4 \\ 1.7 & 0.8 & 0.5 & 0.4 & 5 \end{bmatrix},$$

$$\text{and } \boldsymbol{\Sigma}_3 = \begin{bmatrix} 2 & 0.4 & 0.5 & 0.8 & 0.7 \\ 0.4 & 3 & 0.4 & 0.5 & 2.8 \\ 0.5 & 0.4 & 4 & 0.4 & 0.5 \\ 0.8 & 0.5 & 0.4 & 4 & 0.4 \\ 0.7 & 2.8 & 0.5 & 0.4 & 5 \end{bmatrix}$$

The three mean vectors were all equal to {0 0 0 0 0}. Table 4.7 below gives the type I errors:

**Table 4.7** Type I errors in the three-multivariate-sample complete-data case

| Sample sizes | | | Method | | |
|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | A | B | C |
| 10 | 20 | 30 | 0.055 | 0.052 | 0.043 |
| 10 | 30 | 20 | 0.049 | 0.048 | 0.040 |
| 20 | 10 | 30 | 0.027 | 0.042 | 0.032 |
| 20 | 30 | 10 | 0.030 | 0.043 | 0.030 |
| 30 | 10 | 20 | 0.029 | 0.042 | 0.026 |
| 30 | 20 | 10 | 0.032 | 0.046 | 0.025 |

The code that generated the above table is as follows:

```
%macro test;
%do t = 1 %to 1000;
ods listing close;
%do w = 1 %to 5;
```

```
proc iml;
index1 = %sysevalf(((&t - 1) * 30) + 1) : %sysevalf(&t * 30);
use pop1;
read point index1 var {x&w} into x1;
createx1 from x1 [colname = {'x'}];
append from x1;
index2 = %sysevalf(((&t - 1) * 20) + 1) : %sysevalf(&t * 20);
use pop2;
read point index2 var {x&w} into x2;
createx2fromx2[colname = {'x'}];
append from x2;
index3 = %sysevalf(((&t - 1) * 10) + 1) : %sysevalf(&t * 10);
use pop3;
read point index3 var {x&w} into x3;
create x3 from x3 [colname = {'x'}];
append from x3;
run;
quit;
proc iml;
use x1;
read all var {x} into x1;
n1 = nrow(x1);
use x2;
read all var {x} into x2;
n2 = nrow(x2);
use x3;
read all var {x} into x3;
n3 = nrow(x3);
mu1_ = (sum(x1[, 1])/n1);
mu2_ = (sum(x2[, 1])/n2);
mu3_ = (sum(x3[, 1])/n3);
mu1 = j(n1, 1, 1)@mu1_;
mu2 = j(n2, 1, 1)@mu2_;
mu3 = j(n3, 1, 1)@mu3_;
sigma1 = (x1 - mu1)' * (x1 - mu1)/(n1 - 1);
sigma2 = (x2 - mu2)' * (x2 - mu2)/(n2 - 1);
sigma3 = (x3 - mu3)' * (x3 - mu3)/(n3 - 1);
diff1_ = (mu1_ - mu2_) * *2;
diff2_ = (mu1_ - mu3_) * *2;
diff3_ = (mu2_ - mu3_) * *2;
r_ = ((n1 * (mu1[1, 1] * *2)/sigma1[1, 1]) + (n2 * (mu2[1, 1] * *2)/
     sigma2[1, 1]) + (n3 * (mu3[1, 1] * *2)/sigma3[1, 1]))
-((((n1 * mu1[1, 1]/sigma1[1, 1]) + (n2 * mu2[1, 1]/sigma2[1, 1]) + (n3 *
mu3[1, 1]/sigma3[1, 1])) * *2)
/((n1/sigma1[1, 1]) + (n2/sigma2[1, 1]) + (n3/sigma3[1, 1])));
```

$$
\begin{aligned}
&bf1\_ = (((mu1\_ - mu2\_) * *2)/((sigma1/n1) + (sigma2/n2)));\\
&bf2\_ = (((mu1\_ - mu3\_) * *2)/((sigma1/n1) + (sigma3/n3)));\\
&bf3\_ = (((mu2\_ - mu3\_) * *2)/((sigma2/n2) + (sigma3/n3)));\\
&c1\_1 = 0; c1\_2 = 0; c1\_3 = 0;\\
&c2 = 0;\\
&c3\_1 = 0; c3\_2 = 0; c3\_3 = 0;\\
&do\ j = 1\ to\ 1000;\\
&t1 = tinv(ranuni(7 * j), n1 - 1);\\
&t2 = tinv(ranuni(10 * j), n2 - 1);\\
&diff1 = (((sqrt(sigma1/n1) * t1) - (sqrt(sigma2/n2) * t2)) * *2);\\
&if\ diff1 > diff1\_then\ c1\_1 = c1\_1 + 1;\\
&t1 = tinv(ranuni(7 * j), n1 - 1);\\
&t3 = tinv(ranuni(10 * j), n3 - 1);\\
&diff2 = (((sqrt(sigma1/n1) * t1) - (sqrt(sigma3/n3) * t3)) * *2);\\
&if\ diff2 > diff2\_then\ c1\_2 = c1\_2 + 1;\\
&t2 = tinv(ranuni(7 * j), n2 - 1);\\
&t3 = tinv(ranuni(10 * j), n3 - 1);\\
&diff3 = (((sqrt(sigma2/n2) * t2) - (sqrt(sigma3/n3) * t3)) * *2);\\
&if\ diff3 > diff3\_then\ c1\_3 = c1\_3 + 1;\\
&t1 = tinv(ranuni(10 * j), n1 - 1);\\
&t2 = tinv(ranuni(11 * j), n2 - 1);\\
&t3 = tinv(ranuni(12 * j), n3 - 1);\\
&r = ((t1 * *2) + (t2 * *2) + (t3 * *2))\\
&-((((sqrt(n1/sigma1) * t1) + (sqrt(n2/sigma2) * t2)\\
&\quad + (sqrt(n3/sigma3) * t3)) * *2)/\\
&((n1/sigma1) + (n2/sigma2) + (n3/sigma3)));\\
&if\ r >= r\_then\ c2 = c2 + 1;\\
&t1 = tinv(ranuni(7 * j), n1 - 1);\\
&t2 = tinv(ranuni(10 * j), n2 - 1);\\
&theta = abs(atan(sqrt(sigma1 * n2/(sigma2 * n1))));\\
&bf1 = (((t1 * sin(theta)) - (t2 * cos(theta))) * *2);\\
&if\ bf1 > bf1\_then\ c3\_1 = c3\_1 + 1;\\
&t1 = tinv(ranuni(7 * j), n1 - 1);\\
&t3 = tinv(ranuni(10 * j), n3 - 1);\\
&theta = abs(atan(sqrt(sigma1 * n3/(sigma3 * n1))));\\
&bf2 = (((t1 * sin(theta)) - (t3 * cos(theta))) * *2);\\
&if\ bf2 > bf2\_then\ c3\_2 = c3\_2 + 1;\\
&t2 = tinv(ranuni(7 * j), n2 - 1);\\
&t3 = tinv(ranuni(10 * j), n3 - 1);\\
&theta = abs(atan(sqrt(sigma2 * n3/(sigma3 * n2))));\\
&bf3 = (((t2 * sin(theta)) - (t3 * cos(theta))) * *2);\\
&if\ bf3 > bf3\_then\ c3\_3 = c3\_3 + 1;\\
&end;\\
&p1 = j(3, 1, 0);\\
&p1[1, 1] = c1\_1/1000; p1[2, 1] = c1\_2/1000; p1[3, 1] = c1\_3/1000;
\end{aligned}
$$

```
p2 = c2/1000;
p3 = j(3,1,0);
p3[1,1] = c3_1/1000; p3[2,1] = c3_2/1000; p3[3,1] = c3_3/1000;
create p1_&w from p1 [colname = {'p'}];
append from p1;
create p2_&w from p2 [colname = {'p'}];
append from p2;
create p3_&w from p3 [colname = {'p'}];
append from p3;
run;
quit;
%end;
data p1;
set p1_1 p1_2 p1_3 p1_4 p1_5;
run;
proc sort data = p1;
by p;
run;
data p2;
set p2_1 p2_2 p2_3 p2_4 p2_5;
run;
proc sort data = p2;
by p;
run;
data p3;
set p3_1 p3_2 p3_3 p3_4 p3_5;
run;
proc sort data = p3;
by p;
run;
proc iml;
use p1;
read all var {p} into p1;
n1 = nrow(p1);
imax1 = 0;
do i = 1 to n1;
if p1[i,1] <= ((i/15) * 0.05) then imax1 = i;
end;
if imax1 = 0 then r1 = 0; else r1 = 1;
use p2;
read all var {p} into p2;
n2 = nrow(p2);
imax2 = 0;
do i = 1 to n2;
if p2[i,1] <= ((i/5) * 0.05) then imax2 = i;
end;
```

```
if imax2 = 0 then r2 = 0; else r2 = 1;
use p3;
read all var {p} into p3;
n3 = nrow(p3);
imax3 = 0;
do i = 1 to n3;
if p3[i, 1] <= ((i/15) * 0.05) then imax3 = i;
end;
if imax3 = 0 then r3 = 0; else r3 = 1;
est = r1||r2||r3;
create est&t from est [colname = {'r1' 'r2' 'r3'}];
append from est;
run;
quit;
%end;
%mendtest;
%test;
%macro cat;
%do i = 1 %to 1000;
est&i
%end;
%mend cat;
data concat;
set %cat;
i = _n_;
run;
data _null_;
set concat end = last;
if r2 = 1 then c1 + 1;
if r3 = 1 then c2 + 1;
if r1 = 1 then c3 + 1;
if last then do;
put c1 c2 c3;
end;
run;
```

To investigate the power of methods A, B, and C in the complete-data multivariate case, we generate observations from the same three distributions that generated Table 4.7 except that instead of letting all mean vectors equal to {0 0 0 0 0}, we let the mean vectors to be {0 0 0 0 0}, {0 0 0 0 0}, and {1 1 1 0 0}, respectively. We call this Alternative 1. The power is displayed in Table 4.8 below:

Another alternative we can consider is the one where the three covariance matrices are the same as above, but the mean vectors are {1 0 0 0 0}, {0 1 0 0 0}, and {0 0 1 0 0}, respectively. We call this Alternative 2. The power generated is given in Table 4.10 below:

**Table 4.8** Power in the three-multivariate-sample complete-data case for Alternative 1

| Sample sizes | | | Method | | |
|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | A | B | C |
| 10 | 20 | 30 | 0.840 | 0.839 | 0.752 |
| 10 | 30 | 20 | 0.719 | 0.816 | 0.679 |
| 20 | 10 | 30 | 0.773 | 0.831 | 0.779 |
| 20 | 30 | 10 | 0.424 | 0.513 | 0.374 |
| 30 | 10 | 20 | 0.695 | 0.798 | 0.720 |
| 30 | 20 | 10 | 0.445 | 0.541 | 0.358 |

**Table 4.9** Power in the three-multivariate-sample complete-data case for Alternative 2

| Sample sizes | | | Method | | |
|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | A | B | C |
| 10 | 20 | 30 | 0.655 | 0.647 | 0.567 |
| 10 | 30 | 20 | 0.648 | 0.635 | 0.597 |
| 20 | 10 | 30 | 0.720 | 0.819 | 0.778 |
| 20 | 30 | 10 | 0.734 | 0.788 | 0.783 |
| 30 | 10 | 20 | 0.763 | 0.824 | 0.744 |
| 30 | 20 | 10 | 0.764 | 0.832 | 0.752 |

Tables 4.8 and 4.9 demonstrate that while methods B and C, particularly method B, are not uniformly better than method A in terms of power, method B can be a strong contender to method A when it comes to performing heteroscedastic MANOVA.

### *4.3.2 The Randomly-Incomplete-Data Case*

To investigate Type I errors in the randomly-incomplete-data case, we generate observations from the same three null distributions as considered in Sect. 4.3.1. There is no need to impute as we can just work with the observed data. Let $M$ be a five-variate missingness indicator such that a component is 1 if the corresponding value in the data set is observed and is 0 if the corresponding value in the data set is missing. Then we simulate missing data as given in Table 4.10 below:

**Table 4.10** Simulation of missing data to investigate Type I error

| Covariance matrix | Mean vector | Pr($M = \{1\,1\,1\,1\,1\}$) | Pr($M = \{1\,1\,1\,1\,0\}$) | Pr($M = \{1\,1\,1\,0\,0\}$) |
|---|---|---|---|---|
| $\mathbf{\Sigma}_1$ | {0 0 0 0 0} | 0.70 | 0.15 | 0.15 |
| $\mathbf{\Sigma}_2$ | {0 0 0 0 0} | 0.80 | 0.10 | 0.10 |
| $\mathbf{\Sigma}_3$ | {0 0 0 0 0} | 0.90 | 0.05 | 0.05 |

The type I errors are shown in Table 4.11 below:

**Table 4.11** Type I errors in the three-multivariate-sample randomly-incomplete-data case

| Sample sizes | | | Method | | |
|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | A | B | C |
| 10 | 20 | 30 | 0.052 | 0.049 | 0.039 |
| 10 | 30 | 20 | 0.046 | 0.057 | 0.041 |
| 20 | 10 | 30 | 0.029 | 0.036 | 0.026 |
| 20 | 30 | 10 | 0.021 | 0.035 | 0.020 |
| 30 | 10 | 20 | 0.031 | 0.041 | 0.028 |
| 30 | 20 | 10 | 0.022 | 0.034 | 0.024 |

To investigate power, we consider two alternatives. To consider the first alternative, we generate data with the same covariance matrices as in Sect. 4.3.1, but the mean vectors being {0 0 0 0 0}, {0 0 0 0 0}, and {0 0 1 1 1}. The power for different combinations of sample sizes is displayed in Table 4.12 below:

**Table 4.12** Power in the three-multivariate-sample randomly-incomplete-data case and Alternative 1

| Sample sizes | | | Method | | |
|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | A | B | C |
| 10 | 20 | 30 | 0.299 | 0.366 | 0.265 |
| 10 | 30 | 20 | 0.258 | 0.355 | 0.234 |
| 20 | 10 | 30 | 0.260 | 0.361 | 0.304 |
| 20 | 30 | 10 | 0.146 | 0.206 | 0.105 |
| 30 | 10 | 20 | 0.222 | 0.329 | 0.257 |
| 30 | 20 | 10 | 0.151 | 0.213 | 0.114 |

Table 4.12 demonstrates that method B can be a strong contender to method A in terms of power for the type of alternative considered. Let us now consider another alternative with mean vectors being {0 0 1 0 0}, {0 0 0 1 0}, and {0 0 0 0 1}. The power is displayed in Table 4.13 below:

**Table 4.13** Power in the three-multivariate-sample randomly-incomplete-data case and Alternative 2

| Sample sizes | | | Method | | |
|---|---|---|---|---|---|
| $n_1$ | $n_2$ | $n_3$ | A | B | C |
| 10 | 20 | 30 | 0.247 | 0.291 | 0.217 |
| 10 | 30 | 20 | 0.222 | 0.287 | 0.217 |
| 20 | 10 | 30 | 0.199 | 0.277 | 0.223 |
| 20 | 30 | 10 | 0.218 | 0.305 | 0.266 |
| 30 | 10 | 20 | 0.218 | 0.274 | 0.212 |
| 30 | 20 | 10 | 0.213 | 0.311 | 0.216 |

The code which generated Tables 4.11–4.13 is left as an exercise to the reader.

### *4.3.3 Example: Wisconsin Nursing Home Study Revisited*

In Sect. 3.1.1, we tested for equality of three covariance matrices, and the result was to reject equality. Therefore the method of Sect. 4.3.1 applies here. The three sample mean vectors are {2.066 0.480 0.082 0.360}, {2.167 0.596 0.124 0.418}, and {2.273 0.521 0.125 0.383}. Using the methodology of Sect. 4.3.1, methods A, B, and C all reject the null hypothesis of equality of the mean vectors at the 1 % significance level.

# References


Behrens, W.V.: Ein betrag zur fehlenberechnung bei wenigen beobachtungen. Landw. Jb. **68**, 807–837 (1929)

Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: a new and powerful approach to multiple testing. J. R. Stat. Soc. Ser. B. **57**, 1289–1300 (1995)

Box, G.E.P.: A general distribution theory for a class of likelihood criteria. Biometrika **36**, 317–346 (1949)

Box, G.E.P.: Problems in the analysis of growth and linear curves. Biometrics **6**, 362–389 (1950)

Fisher, R.A.: The fiducial argument in statistical inference. Ann. Eugen. **6**, 391–398 (1935)

Gantz, B.J., et. al.: Evaluation of five different cochlear implant designs: Audiologic assessment and predictors of performance. Laryngoscope **98**, 1100–1106 (1988)

Henze, N., Zirkler, B.: A class of invariant consistent tests for multivariate normality. Commun. Statist. Theor. Methods **19**(10), 3595–3618 (1990)

Johnson, R.A., Wichern, D.W.: Applied Multivariate Statistical Analysis, 5th edn. Prentice Hall, Upper Saddle River (2002)

Khattree, R., Naik, D.N.: Applied Multivariate Statistics With Sas® Software, 2nd edn. Sas Institute, Inc., Cary (1999)

Li, X., Wang, J., Liang, H.: Comparison of several means: A fiducial based approach. Computational Statistics and Data Analysis **55**, 1993–2002 (2011)

Mardia, K.V.: Applications of some measures of multivariate skewness and kurtosis for testing normality and robustness studies. Sankhya Ser. A. **36**, 115–128 (1974)

Nunez-Anton, V., Woodworth, G.G.: Analysis of longitudinal data with unequally spaced observations and time-dependent correlated errors. Biometrics **50**, 445–456 (1994)

Rao, C.R.: Tests of significance in multivariate analysis. Biometrika **35**, 58–79 (1948)

Rencher, A.C.: Methods of Multivariate Analysis, 2nd edn. Wiley, New York (2002)

Rubin, D.B.: Inference and missing data. Biometrika **63**, 581–592 (1976)

Rubin, D.B.: Multiple Imputation for Nonresponse in Surveys. Wiley, New York (1987)

Sidak, Z.: Rectangular confidence regions for the means of multivariate normal distributions. J. Am. Stat. Assoc. **62**, 626–633 (1967)

Westfall, P.H., Tobias, R.D., Rom, D., Wolfinger, R.D., Hochberg, Y.: Multiple Comparisons and Multiple Tests Using SAS®. SAS Institute, Inc., Cary (1999)




DOI 10.1007/978-1-4614-6443-3