8004 Homework 1

Nooreen Dabbish
January 29, 2015

1 Write out the following models of elementary/intermediate statistical analysis in the matrix form

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon$$

(a) A one-variable quadratic polynomial regression model

$$y_i = \alpha_0 + \alpha_1 x_i + \alpha_2 x_i^2 + \epsilon_i \text{ for } (i = 1, 2, \ldots, 5).$$

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix}, \mathbf{X} = \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \\ 1 & x_4 & x_4^2 \\ 1 & x_5 & x_5^2 \end{pmatrix}, \beta = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{pmatrix}, \epsilon = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{pmatrix}$$

$\mathbf{y} = \mathbf{X}\beta + \epsilon$ in this model is therefore:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \\ 1 & x_4 & x_4^2 \\ 1 & x_5 & x_5^2 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{pmatrix}$$

(b) A two-factor ANCOVA model without interactions

$$y_{ijk} = \mu + \alpha_i + \beta_j + \gamma(x_{ijk} - \bar{x}) + \epsilon_{ijk} \text{ for } i = 1, 2,\ j = 1, 2, \text{ and } k = 1, 2.$$

This model describes an 8-dimensional data space, where the column of centered x values may be calculated as follows:

$$\mathbf{y} = \begin{pmatrix} y_{111} \\ y_{112} \\ y_{121} \\ y_{122} \\ y_{211} \\ y_{212} \\ y_{221} \\ y_{222} \end{pmatrix}, \epsilon = \begin{pmatrix} \epsilon_{111} \\ \epsilon_{112} \\ \epsilon_{121} \\ \epsilon_{122} \\ \epsilon_{211} \\ \epsilon_{212} \\ \epsilon_{221} \\ \epsilon_{222} \end{pmatrix}, \mathbf{x}_c = (\mathbf{I} - \frac{1}{n}\mathbf{J}) \begin{pmatrix} x_{111} \\ x_{112} \\ x_{121} \\ x_{122} \\ x_{211} \\ x_{212} \\ x_{221} \\ x_{222} \end{pmatrix} = \begin{pmatrix} x_{111} - \bar{x} \\ x_{112} - \bar{x} \\ x_{121} - \bar{x} \\ x_{122} - \bar{x} \\ x_{211} - \bar{x} \\ x_{212} - \bar{x} \\ x_{221} - \bar{x} \\ x_{222} - \bar{x} \end{pmatrix}$$

The design matrix $\mathbf{X}$ and regression coefficient vector β are given by:

$$\mathbf{X} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & x_{111} - \bar{x} \\ 1 & 1 & 0 & 1 & 0 & x_{112} - \bar{x} \\ 1 & 1 & 0 & 0 & 1 & x_{121} - \bar{x} \\ 1 & 1 & 0 & 0 & 1 & x_{122} - \bar{x} \\ 1 & 0 & 1 & 1 & 0 & x_{211} - \bar{x} \\ 1 & 0 & 1 & 1 & 0 & x_{212} - \bar{x} \\ 1 & 0 & 1 & 0 & 1 & x_{221} - \bar{x} \\ 1 & 0 & 1 & 0 & 1 & x_{222} - \bar{x} \end{pmatrix}, \beta = \begin{pmatrix} \mu \\ \alpha_1 \\ \alpha_2 \\ \beta_1 \\ \beta_2 \\ \gamma \end{pmatrix}$$

Putting these together gives the model:

$$\begin{pmatrix} y_{111} \\ y_{112} \\ y_{121} \\ y_{122} \\ y_{211} \\ y_{212} \\ y_{221} \\ y_{222} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & x_{111}-\bar{x} \\ 1 & 1 & 0 & 1 & 0 & x_{112}-\bar{x} \\ 1 & 1 & 0 & 0 & 1 & x_{121}-\bar{x} \\ 1 & 1 & 0 & 0 & 1 & x_{122}-\bar{x} \\ 1 & 0 & 1 & 1 & 0 & x_{211}-\bar{x} \\ 1 & 0 & 1 & 1 & 0 & x_{212}-\bar{x} \\ 1 & 0 & 1 & 0 & 1 & x_{221}-\bar{x} \\ 1 & 0 & 1 & 0 & 1 & x_{222}-\bar{x} \end{pmatrix} \begin{pmatrix} \mu \\ \alpha_1 \\ \alpha_2 \\ \beta_1 \\ \beta_2 \\ \gamma \end{pmatrix} + \begin{pmatrix} \epsilon_{111} \\ \epsilon_{112} \\ \epsilon_{121} \\ \epsilon_{122} \\ \epsilon_{211} \\ \epsilon_{212} \\ \epsilon_{221} \\ \epsilon_{222} \end{pmatrix}$$

*

2 Use eigen() function in R to compute the eigenvalues and eigenvectors of

$$\mathbf{V} = \begin{pmatrix} 3 & -1 & 1 \\ -1 & 5 & -1 \\ 1 & -1 & 3 \end{pmatrix}$$

Then use R to find and "inverse square root" of this matrix. That is, find a symmetric matrix $\mathbf{W}$ such that $\mathbf{WW} = \mathbf{V}^{-1}$.

(a) Eigenvalues and Eigenvectors

I ran the following R-code using `eigen()` eigen to calculate eigenvalues and eigenvectors. A small function `lvector()` calls `xtable()` and `print()` in order to generate latex output below (see appendix).

```
V <- matrix(c(3, -1, 1, -1, 5, -1, 1, -1, 3), 3,3, byrow=TRUE)
lvector(as.matrix(eigen(V)$values), dig=0)
for (i in 1:3){
lvector(as.matrix(eigen(V)$vectors[,i]))
}
```

The eigenvalues are

$$\lambda = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} = \begin{pmatrix} 6 \\ 3 \\ 2 \end{pmatrix}$$

With corresponding eigenvectors:

$$\mathbf{e_1} = \begin{pmatrix} 0.41 \\ -0.82 \\ 0.41 \end{pmatrix}, \mathbf{e_2} = \begin{pmatrix} 0.58 \\ 0.58 \\ 0.58 \end{pmatrix}, \mathbf{e_3} = \begin{pmatrix} 0.71 \\ 0.00 \\ -0.71 \end{pmatrix}.$$

(b) Inverse Square Root

In order to find the "inverse square root", spectral decomposition was performed, as outlined below. $\mathbf{U}$ is a matrix made up of the eigenvectors of $\mathbf{V}$ in columns, and $\mathbf{D}$ is a matrix with $\mathbf{V}$'s eigenvalues along the diagonal. $\mathbf{V}$ and $\mathbf{V}^{-1}$ have the same eigenvectors, denoted u_i below, and a square non-singular matrix with eigenvalues λ_i's will have an inverse with eigenvalues given by ${\lambda_i}^{-1}$.

$$V = UDU^T = \sum_{i=1}^{n} \lambda_i u_i u_i^T$$
$$V^{1/2} = UD^{1/2}U^T = \sum_{i=1}^{n} \lambda_i^{1/2} u_i u_i^T$$
$$V^{-1} = UD^{-1}U^T = \sum_{i=1}^{n} \lambda_i^{-1} u_i u_i^T$$
$$V^{-1/2} = UD^{-1/2}U^T = \sum_{i=1}^{n} \lambda_i^{-1/2} u_i u_i^T$$

```
V_inv = solve(V)
C = as.matrix(eigen(V_inv)$vectors)
D_sqrt = diag(lapply(eigen(V_inv)$values, sqrt))
W = C%*%D_sqrt%*%t(C)
lvector(W, dig=4)
```

$$\mathbf{W} = \begin{pmatrix} 0.6140 & 0.0564 & -0.0931 \\ 0.0564 & 0.4646 & 0.0564 \\ -0.0931 & 0.0564 & 0.6140 \end{pmatrix}$$

And a comparison between $\mathbf{WW}$ and $\mathbf{V}^{-1}$:

```
Prod = W%*%W
lvector(Prod)
lvector(solve(V))
```

$$\mathbf{WW} = \begin{pmatrix} 0.39 & 0.06 & -0.11 \\ 0.06 & 0.22 & 0.06 \\ -0.11 & 0.06 & 0.39 \end{pmatrix}, \mathbf{V^{-1}} = \begin{pmatrix} 0.39 & 0.06 & -0.11 \\ 0.06 & 0.22 & 0.06 \\ -0.11 & 0.06 & 0.39 \end{pmatrix}$$

3 Consider the matrices

$$\mathbf{A} = \begin{pmatrix} 4 & 4.001 \\ 4.001 & 4.002 \end{pmatrix} \text{ and } \mathbf{B} = \begin{pmatrix} 4 & 4.001 \\ 4.001 & 4.002001 \end{pmatrix}.$$

Obviously, these matrices are nearly identical. Use R and compute the determinants and inverses of these matrices. (Even though the original two matrices are nearly the same, $\mathbf{A}^{-1} \approx -3\mathbf{B}^{-1}$. This shows that small changes in the elements of nearly singular matrices can have big effects on some matrix operations.)

(a) Determinants and Inverses.

Both determinants were determined using `det()` in R and are nearly zero. The determinant of A is `-1.00000000513756e-06` and the determinant of B is `2.99999999764467e-06`.

$$\mathbf{A}^{-1} = \begin{pmatrix} -4001999.98 & 4000999.98 \\ 4000999.98 & -3999999.98 \end{pmatrix} \text{ and } \mathbf{B}^{-1} = \begin{pmatrix} 1334000.33 & -1333666.67 \\ -1333666.67 & 1333333.33 \end{pmatrix}$$

#$$

$$-3\mathbf{B}^{-1} = \begin{pmatrix} -4002001.00 & 4001000.00 \\ 4001000.00 & -4000000.00 \end{pmatrix}$$

4 Write an R function to conduct projection, e.g. with the name project()

The input is the given design matrix $\mathbf{X}$, and the output is the projection matrix $\mathbf{P_X}$ for projecting a vector onto the column space of $\mathbf{X}$.

In the following code, I define a function `project()` which accepts a matrix as an input and uses the `t()` and `ginv()` functions to calculate the transpose and inverse. $\mathbf{P_X} = \mathbf{X}(\mathbf{X^T X})^{-}\mathbf{X^T}$ is returned. I also ran a few tests, first using the matrix $\mathbf{V}$ defined in problem 2 to make sure the projection is symmetric and idempotent. I also tested the results of scalar and $\mathbf{0}$ (matrix of all zero) inputs.

```
library(MASS)
project <- function (X) {X%*%(ginv(t(X)%*%X))%*%t(X)}

lvector(project(V))
```

$$\begin{pmatrix} 1.00 & 0.00 & 0.00 \\ -0.00 & 1.00 & -0.00 \\ 0.00 & 0.00 & 1.00 \end{pmatrix}$$

```
lvector(project(V)%*%project(V))
```

$$\begin{pmatrix} 1.00 & 0.00 & 0.00 \\ -0.00 & 1.00 & -0.00 \\ 0.00 & 0.00 & 1.00 \end{pmatrix}$$

```
lvector(t(project(V)))
```

$$\begin{pmatrix} 1.00 & -0.00 & 0.00 \\ 0.00 & 1.00 & 0.00 \\ 0.00 & -0.00 & 1.00 \end{pmatrix}$$

```
lvector(project(3))
```

$$\begin{pmatrix} 1.00 \end{pmatrix}$$

```
lvector(project(matrix(c(0,0,0,0),2,2)))
```

$$\begin{pmatrix} 0.00 & 0.00 \\ 0.00 & 0.00 \end{pmatrix}$$

5 Consider the (non-full-rank) two-way "effect model" with interactions in the Example (d) in lecture.

(a) Determine which of the parametric functions below are estimable:

$$\alpha_1, \alpha_2 - \alpha_a, \mu + \alpha_1 + \beta_1 + \delta_{11}, \delta_{12}, \delta_{12} - \delta_{11} - (\delta_{22} - \delta_{21})$$

For those that are estimable, find $\mathbf{c}^T(\mathbf{X}^T\mathbf{X})^{-}\mathbf{X}^T$, such that $\mathbf{c}^T(\mathbf{X}^T\mathbf{X})^{-}\mathbf{X}^T\mathbf{Y}$ produces the estimate of $\mathbf{C}^T\beta$.

Example (d) described a two-way model to study trees of types A and B treated with either old or new fungicide. Each response variable, y_{ijk} represents the response of variety i to fungicide j of the tree with index k, where $i = 1,2$, $j = 1,2$, and $k = 1,2$.

The model, $\mathbf{Y} = \mathbf{X}\beta + \epsilon$ in example (d) is:

$$\begin{pmatrix} y_{111} \\ y_{112} \\ y_{121} \\ y_{122} \\ y_{211} \\ y_{212} \\ y_{221} \\ y_{222} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mu \\ \alpha_1 \\ \alpha_2 \\ \beta_1 \\ \beta_2 \\ \delta_1 1 \\ \delta_1 2 \\ \delta_2 1 \\ \delta_2 2 \end{pmatrix} + \begin{pmatrix} \epsilon_{111} \\ \epsilon_{112} \\ \epsilon_{121} \\ \epsilon_{122} \\ \epsilon_{211} \\ \epsilon_{212} \\ \epsilon_{221} \\ \epsilon_{222} \end{pmatrix}$$

The only regression coefficients or combinations that can be estimated can be written in the form $\mathbf{c^T}\beta$, where the vector $\mathbf{c}$ is in the row space of the design matrix $\mathbf{X}$ (equivalent to the column space of $\mathbf{X^T}$).

There are 5 parametric functions in this question:

1. α_1, $c^T = (0,1,0,0,0,0,0,0,0)$. This is not in the column space of $\mathbf{X^T}$, so it is not estimable.
2. $\alpha_2 - \alpha_1$, $c^T = (0,-1,1,0,0,0,0,0,0)$ This is not in the column space of $\mathbf{X^T}$, so it is not estimable.
3. $\mu + \alpha_1 + \beta_1 + \delta_{11}$, $c^T = (1,1,0,1,0,1,0,0,0)$ This is in the column space of $\mathbf{X^T}$, so it is estimable.
4. δ_{12}, $c^T = (0,0,0,0,0,0,1,0,0)$. Not in the column space of $\mathbf{X^T}$ and therefore not estimable.
5. $\delta_{12} - \delta_{11} - (\delta_{22} - \delta_{21})$, $c^T = (0,0,0,0,0,-1,1,1,-1)$. This is in the column space of $\mathbf{X^T}$, so it is estimable.

We can use R to find $\mathbf{c^T}(X^TX)^-X^T$ for $\mu + \alpha_1 + \beta_1 + \delta_1 1$ and $\delta_1 2 - \delta_1 1 - (\delta_2 2 - \delta_2 1)$.

```
X <- matrix(c(rep(1,8), rep(1,4), rep(0,8), rep(1,4),
              rep(c(1,1,0,0),2), rep(c(0,0,1,1),2),
              rep(c(1,1,rep(0,8)),3), 1,1),nrow=8,ncol=9, byrow=FALSE)
ct3 <- matrix(c(1,1,0,1,0,1,0,0,0),nrow=1)
ct5 <- matrix(c(0,0,0,0,0,-1,1,1,-1), nrow=1)
lvector(ct3%*%ginv(t(X)%*%X)%*%t(X))
lvector(ct5%*%ginv(t(X)%*%X)%*%t(X))
```

$$\mathbf{c^T_{\mu+\alpha_1+\beta_1+\delta_1 1}}(\mathbf{X^TX})^-\mathbf{X^T} = \begin{pmatrix} 0.50 & 0.50 & -0.00 & -0.00 & -0.00 & -0.00 & 0.00 & 0.00 \end{pmatrix}$$

and

$$\mathbf{c^T_{\delta_{12}-\delta_{11}-(\delta_{22}-\delta_{21})}}(\mathbf{X^TX})^-\mathbf{X^T} = \begin{pmatrix} -0.50 & -0.50 & 0.50 & 0.50 & 0.50 & 0.50 & -0.50 & -0.50 \end{pmatrix}$$

(b) For the parameter vector β written in the order used in class, consider the hypothesis $H_0 : \mathbf{C}\beta = \mathbf{0}$ for

$$\mathbf{C} = \begin{pmatrix} 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 \end{pmatrix}$$

Is this hypothesis testable? Explain.

No, this hypothesis is not fully testable. This hypothesis asks whether the parametric function $\alpha_1 - \alpha_2$ (represented by the first row of $\mathbf{C}$) is zero and the parametric function $\delta_1 1 - \delta_1 2 - \delta_2 1 + \delta_2 2$ is zero (second row). The second row of $\mathbf{C}$ is in the column space of $\mathbf{X^T}$ so it would be testable. However, the first row is not, the function $\alpha_1 - \alpha_2$ is not estimable and therefore not testable.

6 Appendix: Tangled R-code

```
library(xtable)
lvector <- function(x, dig = 2, dsply=rep("f",ncol(x)+1)) {
 x <- xtable(x, align=rep("",ncol(x)+1),display=dsply,digits=dig) # We repeat empty string 6 times
 print(x, floating=FALSE, tabular.environment="pmatrix",
   hline.after=NULL, include.rownames=FALSE, include.colnames=FALSE)
 }

lvector(as.matrix(lapply(1:5, function(x) paste("y_", x, sep="")), dsply=c("s","s")))
x <- as.matrix(lapply(1:5, function(x) paste("x_", x, sep="")))
x2 <- as.matrix(lapply(1:5, function(x) paste("x2_",x,sep="")))
X <- cbind(rep(1,5), x, x2)
lvector(X, dig=0)
lvector(as.matrix(lapply(1:5, function(x) paste("epsilon_", x, sep="")), dsply=c("s","s")))

V <- matrix(c(3, -1, 1, -1, 5, -1, 1, -1, 3), 3,3, byrow=TRUE)
lvector(as.matrix(eigen(V)$values), dig=0)
for (i in 1:3){
lvector(as.matrix(eigen(V)$vectors[,i]))
}

V_inv = solve(V)
C = as.matrix(eigen(V_inv)$vectors)
D_sqrt = diag(lapply(eigen(V_inv)$values, sqrt))
W = C%*%D_sqrt%*%t(C)
lvector(W, dig=4)

Prod = W%*%W
lvector(Prod)
lvector(solve(V))

A <- matrix(c(4, 4.001, 4.001, 4.002),2,2,byrow=T)
B <- matrix(c(4, 4.001, 4.001, 4.002001),2,2,byrow=T)

library(MASS)
project <- function (X) {X%*%(ginv(t(X)%*%X))%*%t(X)}

lvector(project(V))

lvector(project(V)%*%project(V))

lvector(t(project(V)))

lvector(project(3))

lvector(project(matrix(c(0,0,0,0),2,2)))

X <- matrix(c(rep(1,8), rep(1,4), rep(0,8), rep(1,4),
```

```
                 rep(c(1,1,0,0),2), rep(c(0,0,1,1),2),
                 rep(c(1,1,rep(0,8)),3), 1,1),nrow=8,ncol=9, byrow=FALSE)
ct3 <- matrix(c(1,1,0,1,0,1,0,0,0),nrow=1)
ct5 <- matrix(c(0,0,0,0,0,-1,1,1,-1), nrow=1)
lvector(ct3%*%ginv(t(X)%*%X)%*%t(X))
lvector(ct5%*%ginv(t(X)%*%X)%*%t(X))
```