Homework 7 - Data Manipulation

James Young

07/11/2019

There are six exercises below. You are required to provide solutions for at least four of the five. You are required to solve at least one exercise in R, and at least one in SAS. You are required to provide five solutions, each solution will be worth 10 points. Thus, you may choose to provide both R and SAS solutions for a single exercise, or you may solve five of the sixth problems, mixing the languages as you wish.

If you choose SAS for an exercise, you may use IML, DATA operations or PROC SQL at your discretion.

*Warning* I will continue restricting the use of external libraries in R, particularly tidyverse libraries. You may choose to use ggplot2, but take care that the plots you produce are at least as readable as the equivalent plots in base R. You will be allowed to use whatever libraries tickle your fancy in the midterm and final projects.

## Reuse

For many of these exercises, you may be able to reuse functions written in prior homework. Define those functions here.

required.replicates <- function (CV, Diff,alpha, beta) {  
 2 \* ((CV/Diff)\*\*2) \* (((1-(alpha/2))+(1-beta)))\*\*2  
}

# Exercise 1 Please Grade.

### Part a.

Repeat the table from Homework 5, Exercise 2. The table will contain 30 rows, each corresponding to a unique combination of CV from and Diff from . However, for this exercise you only need to calculate one column for required replicates ( and )

Define the table in the space below. *Do not print this table*.

A = seq(8, 28,by=4)  
B = seq(5, 25,by=5)  
  
  
d1 <- expand.grid(CV = B, Diff = A, KEEP.OUT.ATTRS = FALSE)  
d1$RequiredReps = required.replicates(d1$CV, d1$Diff, 0.05, 0.2)

### Part b.

Create two subset tables, one that contains the combinations of CV and Diff that require the five largest number of replicates and one the contains the combinations of CV and Diff the five smallest number of replicates. You can determine the subset by ranking or sorting by required replicates. You can add a rank column to your table if you wish. Call one table LargestFive and one table SmallestFive.

#rank order the required reps  
order.scores <- order(d1$RequiredReps)  
d1$rank[order.scores] <- 1:nrow(d1)  
#subset based on rank order  
SmallestFive <- subset(d1, rank < 6)  
LargestFive <-subset(d1, rank > 25)

### Part c.

Print LargestFive sorted by required replicates in descending order, and print SmallestFive in ascending order.

LargestFive <- LargestFive[order(-LargestFive$RequiredReps),]   
SmallestFive <- SmallestFive[order(SmallestFive$RequiredReps),]   
#First table  
LargestFive

## CV Diff RequiredReps rank  
## 5 25 8 61.53564 30  
## 4 20 8 39.38281 29  
## 10 25 12 27.34918 28  
## 3 15 8 22.15283 27  
## 9 20 12 17.50347 26

#second table  
SmallestFive

## CV Diff RequiredReps rank  
## 26 5 28 0.2009327 1  
## 21 5 24 0.2734918 2  
## 16 5 20 0.3938281 3  
## 11 5 16 0.6153564 4  
## 27 10 28 0.8037309 5

# Exercise 2 Please Grade

### Part a

Go to <http://www.itl.nist.gov/div898/strd/anova/SiRstv.html> and use the data listed under Data File in Table Format (<https://www.itl.nist.gov/div898/strd/anova/SiRstvt.dat>). You may reuse the file from Homework 6. Load the data into a table below.

NIST <- read.csv("C://data/NIST.csv", header=T, stringsAsFactors = F)

### Part b

Reshape or transpose this table from the wide format to the long format. Make sure the resulting table has two columns - Resistance and Instrument.

#manipulate data frame into long shape. Had to add special first column to csv whiche denotes reps in order to get desired results. CSV file included in upload.   
tmydf = setNames(data.frame(t(NIST[,-1])), NIST[,1])  
tmydf$Instrument <- rownames(tmydf)  
tmydf <- as.data.frame(tmydf)  
NIST\_Long <- reshape(tmydf,   
 direction="long",  
 varying=1:5,  
 idvar='Instrument',  
   
 v.names="Resistance")  
NIST\_Long$time<-NULL  
NIST\_Long[order(NIST\_Long$Instrument),]

## Instrument Resistance  
## Inst1.1 Inst1 196.3052  
## Inst1.2 Inst1 196.1240  
## Inst1.3 Inst1 196.1890  
## Inst1.4 Inst1 196.2569  
## Inst1.5 Inst1 196.3403  
## Inst2.1 Inst2 196.3042  
## Inst2.2 Inst2 196.3825  
## Inst2.3 Inst2 196.1669  
## Inst2.4 Inst2 196.3257  
## Inst2.5 Inst2 196.0422  
## Inst3.1 Inst3 196.1303  
## Inst3.2 Inst3 196.2005  
## Inst3.3 Inst3 196.2889  
## Inst3.4 Inst3 196.0343  
## Inst3.5 Inst3 196.1811  
## Inst4.1 Inst4 196.2795  
## Inst4.2 Inst4 196.1748  
## Inst4.3 Inst4 196.1494  
## Inst4.4 Inst4 196.1485  
## Inst4.5 Inst4 195.9885  
## Inst5.1 Inst5 196.2119  
## Inst5.2 Inst5 196.1051  
## Inst5.3 Inst5 196.1850  
## Inst5.4 Inst5 196.0052  
## Inst5.5 Inst5 196.2090

### Part c

To confirm that the table was reshaped correctly, use aggregate or tapply to calculate mean Resistance grouped by Instrument from the long table, and use apply or colMeans to calculate column means from the wide table. Print and compare the results.

aggregate(NIST\_Long$Resistance, by=NIST\_Long["Instrument"], FUN=mean, na.rm=TRUE)

## Instrument x  
## 1 Inst1 196.2431  
## 2 Inst2 196.2443  
## 3 Inst3 196.1670  
## 4 Inst4 196.1481  
## 5 Inst5 196.1432

colMeans(NIST[2:6])

## Inst1 Inst2 Inst3 Inst4 Inst5   
## 196.2431 196.2443 196.1670 196.1481 196.1432

The means taken from the original and the reshaped are the same.

Note that the reshaped table should be equivalent to the file linked under ‘Data File in Two-Column Format’.

# Exercise 5 Please Grade

### Background

I’m working on software that produces a repeated measures analysis. To test my code, I use published data and compare results. For one analysis, I used data from **Contemporary Statistical Models for the Plant and Soil Sciences**, Oliver Schabenberger and Francis J. Pierce, 2001. These data are measurements of the diameter of individual apples from selected apple trees.

## Part a.

Download the AppleData.csv if you choose R, the SAS data is included in the SAS template. Note the file include comments for the data; you may need to specify comment character in import. *Do not print this table*.

To simplify this exercise, create a subset of the AppleData including only trees number 3, 7 and 10.

AppleData <- read.csv("C://data/AppleData.csv", comment.char = "#")  
AppleData <- subset(AppleData, tree ==3 | tree == 7 | tree == 10)

## Part b.

Reshape or transpose this data from the long form to the wide form. Call this data AppleWide. This table should have one column for Tree, one column for Apple and six columns, diam.1 - diam.6. The values in the time columns come from diam in AppleData.

#Going from long data to wide data  
AppleWide = reshape(AppleData,  
 timevar = "time",  
 idvar = c("tree","apple"),  
 direction = "wide")  
print(AppleWide)

## tree apple diam.1 diam.2 diam.3 diam.4 diam.5 diam.6  
## 111 3 1 2.91 3.00 3.02 3.03 NA NA  
## 115 3 10 2.81 2.89 2.87 2.93 2.93 2.94  
## 121 3 16 2.95 3.00 3.03 3.03 3.06 3.08  
## 127 3 17 2.79 2.83 2.86 2.87 2.87 2.93  
## 133 3 18 2.98 3.03 3.06 3.09 3.09 3.09  
## 139 3 20 2.76 2.82 2.83 2.85 2.86 2.88  
## 145 3 22 2.76 2.82 2.85 2.87 2.90 2.90  
## 151 3 23 2.76 2.78 2.77 2.79 2.79 2.79  
## 157 3 24 2.80 2.85 2.87 2.87 2.89 2.92  
## 317 7 2 2.79 2.89 2.89 2.91 2.91 2.95  
## 323 7 4 2.80 2.81 2.85 2.91 2.92 2.96  
## 329 7 9 3.06 3.15 3.15 3.23 3.27 3.31  
## 335 7 25 2.84 2.86 2.88 2.93 2.93 2.96  
## 399 10 2 2.92 2.95 3.00 3.01 3.07 NA  
## 404 10 5 2.87 2.89 2.94 2.95 3.01 3.02  
## 410 10 8 2.76 2.81 2.86 2.90 NA NA  
## 414 10 9 2.91 3.01 3.07 3.09 3.11 NA  
## 419 10 10 2.88 2.88 2.92 2.97 2.97 2.99  
## 425 10 17 3.00 3.05 3.05 3.06 3.11 NA  
## 430 10 18 2.85 2.87 2.91 2.95 2.98 3.00  
## 436 10 21 2.76 2.83 2.84 2.87 2.88 2.91  
## 442 10 22 3.25 3.34 3.34 3.38 3.47 NA  
## 447 10 23 3.00 3.06 3.08 3.14 3.18 NA

## Part c.

To confirm that you’ve reshaped correctly, print column means for the wide data set and use an aggregate or apply function to compute time means for the long format.

apply(AppleWide,2,mean)

## tree apple diam.1 diam.2 diam.3 diam.4 diam.5   
## 6.739130 14.173913 2.878696 2.931304 2.953913 2.983913 NA   
## diam.6   
## NA

aggregate(AppleData$diam, by=AppleData["time"], FUN=mean, na.rm=TRUE)

## time x  
## 1 1 2.878696  
## 2 2 2.931304  
## 3 3 2.953913  
## 4 4 2.983913  
## 5 5 3.009524  
## 6 6 2.976875

The means from the original and the reshaped are the same.

## Part d.

I choose this example for a test case because it shows a case where the best repeated measures model is an auto-regressive model - each measure is correlated with the preceding measure. We can estimate the degree of using the following R code. You don’t need to evaluate this code for this exercise; it’s provided as a motivation for reshaping the data.

#Following the code, a nice correlation table is output.  
mult.lm <- lm(cbind(diam.1, diam.2, diam.3, diam.4, diam.5, diam.6) ~ tree, data=AppleWide)  
mult.manova <- manova(mult.lm)  
print(cov2cor(estVar(mult.lm)))

# Exercise 6 Please Grade

This is an exercise in computing the Wilcoxon Signed Rank test. We will be using an example from NIST (NATR332.DAT). See <https://www.itl.nist.gov/div898/software/dataplot/refman1/auxillar/signrank.htm> .

The data are provided:

NATR332.DAT <- data.frame(  
 Y1 = c(146,141,135,142,140,143,138,137,142,136),  
 Y2 = c(141,143,139,139,140,141,138,140,142,138)  
)

## Part a.

Add a column Difference that is the difference between Y1 and Y2. For further analysis, exclude any rows where the difference is 0.

Next add add the column Rank, which will be the rank of the absolute value of Difference.

NATR332.DAT$Difference = NATR332.DAT$Y1 - NATR332.DAT$Y2  
#Drop 0's  
NATR332.DAT2 = subset(NATR332.DAT, 0 != Difference)  
  
#rank difference  
diffrank = rank(abs(NATR332.DAT2$Difference))  
NATR332.DAT2$Rank = diffrank  
  
  
  
print(NATR332.DAT2)

## Y1 Y2 Difference Rank  
## 1 146 141 5 7.0  
## 2 141 143 -2 2.0  
## 3 135 139 -4 6.0  
## 4 142 139 3 4.5  
## 6 143 141 2 2.0  
## 8 137 140 -3 4.5  
## 10 136 138 -2 2.0

## Part c.

Add the column SignedRank by applying the sign (+ or -) of Difference, to to Rank (that is, if Difference is < 0, then SignedRank is -Rank, otherwise SignedRank is Rank).

for(i in 1:length(NATR332.DAT2$Rank)){  
 if(NATR332.DAT2$Difference[i]<0){  
 NATR332.DAT2$SignedRank[i] = NATR332.DAT2$Rank[i]\*(-1)  
 }else{  
 NATR332.DAT2$SignedRank[i] = NATR332.DAT2$Rank[i]  
 }  
}  
  
  
NATR332.DAT2

## Y1 Y2 Difference Rank SignedRank  
## 1 146 141 5 7.0 7.0  
## 2 141 143 -2 2.0 -2.0  
## 3 135 139 -4 6.0 -6.0  
## 4 142 139 3 4.5 4.5  
## 6 143 141 2 2.0 2.0  
## 8 137 140 -3 4.5 -4.5  
## 10 136 138 -2 2.0 -2.0

## Part d.

Compute the sum of the positive ranks, and the absolute value of the sum of the negative ranks. Let be the minimum of these two sums. Print .

positive.sum = 0  
positive.val = 0  
negative.sum = 0  
negative.val = 0  
for(i in 1:length(NATR332.DAT2$SignedRank)){  
 if(NATR332.DAT2$SignedRank[i]>0){  
 positive.sum = positive.sum + NATR332.DAT2$SignedRank[i]  
 positive.val = positive.val + 1  
 }else{  
 negative.sum = negative.sum + abs(NATR332.DAT2$SignedRank[i])  
 negative.val = negative.val + 1  
 }  
}  
  
  
if(positive.sum<negative.sum){  
 w = positive.sum  
 print(w)  
}else{  
 w = negative.sum  
 print(w)  
}

## [1] 13.5

The expected mean of is calculated by

with a standard deviation of

where is the number of ranked values (excluding differences of 0). Calculate a score by

Print both and .

n.r = length(NATR332.DAT2$SignedRank)  
  
mu.w = n.r \* (n.r + 1) / 4  
  
si.w.nu = n.r \* (n.r + 1) \* (2 \* n.r + 1)  
#hard code 6 in the denominator..  
si.w = sqrt(si.w.nu/6)  
z = (mu.w - w)/si.w  
print(mu.w)

## [1] 14

print(z)

## [1] 0.04225771

The NIST page gives a p-values based on the continuity correction. We are not computing this correction. You can compute the of your (using the normal distribution) and compare it to

p = pnorm(z,lower.tail = TRUE)  
print(p)  
  
wilcox.test(NATR332.DAT$Y1, NATR332.DAT$Y2, paired = TRUE, correct = FALSE, alternative = "less")  
  
#pnorm shows a p-value that is higher than the wilcox.test

while the corrected p-values are given by

wilcox.test(NATR332.DAT$Y1, NATR332.DAT$Y2, paired = TRUE, correct = TRUE, alternative = "less")  
wilcox.test(NATR332.DAT$Y1, NATR332.DAT$Y2, paired = TRUE, correct = TRUE, alternative = "greater")