Homework 7 - Data Manipulation

Peter Claussen

3/22/2018

Data

As with previous homework, all the numeric values you need, other than `0.05`, `0`, `1`, `2` and `3` are defined below:

```r
Year=c(1936, 1946, 1951, 1963, 1975, 1997, 2006)
CaloriesPerRecipeMean <- c(2123.8, 2122.3, 2089.9, 2250.0, 2234.2, 2249.6, 3051.9)
CaloriesPerRecipeSD <- c(1050.0, 1002.3, 1009.6, 1078.6, 1089.2, 1094.8, 1496.2)
CaloriesPerServingMean <- c(268.1, 271.1, 280.9, 294.7, 285.6, 288.6, 384.4)
CaloriesPerServingSD <- c(124.8, 124.2, 116.2, 117.7, 118.3, 122.0, 168.3)
ServingsPerRecipeMean <- c(12.9, 12.9, 13.0, 12.7, 12.4, 12.4, 12.7)
ServingsPerRecipeSD <- c(13.3, 13.3, 14.5, 14.6, 14.3, 14.3, 13.0)
sample.size <- 18
tenth.increment <- 0.10
idx.1936 <- 1
idx.2006 <- length(CaloriesPerRecipeMean)
idxs36_07 <- c(idx.1936,idx.2006)
alpha=0.05


CookingTooMuch.dat <- data.frame(
  Year=Year,
  CaloriesPerRecipeMean = CaloriesPerRecipeMean,
  CaloriesPerRecipeSD = CaloriesPerRecipeSD,
  CaloriesPerServingMean = CaloriesPerServingMean,
  CaloriesPerServingSD = CaloriesPerServingSD,
  ServingsPerRecipeMean = ServingsPerRecipeMean,
  ServingsPerRecipeSD = ServingsPerRecipeSD
)
```

Similar restrictions from Homework 4,5 and 6 apply. In this homework we will be manipulating data, so you should not need to call the `data.frame` function directly except in the initial steps for Exercise 1 using R; in SAS you can use DATA steps as necessary, although some exercises may be easier using PROC SQL.

There are 5 exercises in total, you are required to solve at least 4 and you must solve at least 1 exercise using R and at least 1 exercise using SAS. You can, as previously, solve only 4, using both R and SAS for one problem, or you can solve all 5, as long as you provide at least one R solution and at least one SAS solution.

To simplify grading, include exercise numbers in the uploaded file names.

Exercise 1.

Recreate the table from Exercise 1, Homework 5 but start with the table from Exercise 1, Homework 4. Specifically, you will start with a data table with 49 rows, but reduce it to 21 rows with a series of manipulations.

Part a.

Reproduce the code from Homework 4 here. This table should have 7×7 rows. Call this table 'CohenA' and print the table.

Part b.

Create a table `CohenB` from `CohenA` by selecting only those rows where `Year1` is not equal to `Year2`; alternatively, select only those rows when Cohen d is greater than 0.

Print this table.

Part c

This step is dependent on knowing that the original matrices from Homework 3 were symmetric and have unique values for each pair of treatment differences

Sort `CohenB` by d values. This will produce a table where there are pairs of rows, each representing the same mean comparison (that is, $|m_{1936} - m_{1946}|$ and $|m_{1946} - m_{1936}|$ will be consecutive rows). Remove one row from each pair by creating an index that will select every other row (i.e. all even number, all odd numbers, alternating true and false). If you do this in SAS, you can reference the automatic variable `_n_` in the DATA step - this variable counts the current row number.

Call this table `CohenC` and print this table.

Part d.

Sort `CohenC` by `Year1` and `Year2` to reproduce the order of rows in the table from Homework 6. Print this table.

Exercise 2

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 these files include comments for the data; you may need to specify comment character in import. (The SAS data was where I started).

To simplify this exercise, create a subset of the AppleData including only trees number 3, 7 and 10. (I was going to edit the files to only include these trees, but then I thought, "That's what computers are for!")

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`. If you use SAS, use `diam1-diam6` as column names.

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. If SAS, use PROC MEANS. Call with `var diam; by time;` for one table, and `var diam1-diam6;` for the other.

Part d.

I choose this example for a test case because it shows a case where the best repeated measures model is an autoregressive 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.

```r
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)))
```

If you use SAS, the equivalent code is part of the PROC GLM block, look for the table title **Partial Correlation Coefficients from the Error SSCP Matrix / Prob > |r|**.

Exercise 3

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

The data are provided:

```r
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 the column `Difference` by subtracting `Y1` from `Y2`. For further analysis, exclude any rows where the difference is 0.

Next add the column `Rank` as the rank of the absolute value of `Difference`.

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`).

Part d.

Compute the sum of the positive ranks, and the absolute value of the sum of the negative ranks.

Let W be the minimum of these two sums. Print W.

The expected mean of W is calculated by $\mu_W = N_r * (N_r + 1)/4$ with a standard deviation of

$$\sigma_W = \sqrt{\frac{N_r(N_r+1)(2N_r+1)}{24}}$$

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

$$z = (\mu_W - W)/\sigma_W$$

and the probability of greater z. Print both z and p.

The NIST page gives a p-value, 0.5677, based on the continuity correction. We are not computing this correction. You can compare the p of your z in R by

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

```
## Warning in wilcox.test.default(NATR332.DAT$Y1, NATR332.DAT$Y2, paired =
## TRUE, : cannot compute exact p-value with ties

## Warning in wilcox.test.default(NATR332.DAT$Y1, NATR332.DAT$Y2, paired =
## TRUE, : cannot compute exact p-value with zeroes

##
##  Wilcoxon signed rank test
##
## data:  NATR332.DAT$Y1 and NATR332.DAT$Y2
## V = 13.5, p-value = 0.534
## alternative hypothesis: true location shift is greater than 0
```

Exercise 4

In the 'fastest.csv' data set from Homework 5, Exercise 5, there are several motorcycles that were the fastest in production in their time, but were not the fastest in history, up to that point. That is, some motorcycles were fastest because a faster motorcycle had gone out of production.

Part a

Programmatically determine these motorcycles. If you can show that the gaps between the *fastest ever* and *currently fastest* is not greater than one model, you can simply compare $MPH_i < MPH_{i-1}$; otherwise you will need to iterate over each row and deterimine the maximimum MPH for all preceding rows.

Create and print a vector of the model names for these motorcycles.

Part b.

Create a subset of the original `fastest.csv` that contains those motorcycles not found in the motorcycles identified in part a. Print this table.

Part c.

Compare these two data sets by creating a staircase or step plot including both sets, plotting MPH as the dependent variable and Year as the independent variable. Use different colors for the lines for each set.

Add a vertical line at 1949, marking the year that the Vincent Black Lightning was introduced.

Exercise 5

Part a

Go to http://www.itl.nist.gov/div898/strd/anova/AtmWtAg.html and download the file listed under Data File in Table Format (ttps://www.itl.nist.gov/div898/strd/anova/AtmWtAgt.dat). You may have done this in the previous homework. This file is in the wide format; you will be expected to reshape or transpose this to the long format.

Do not read the file listed under Data File in Two-Column Format. This data file is in the long format. You can use it to check your work, but for this exercise you might write code to obtain data in the long form.

Part b

Reshape or tranpose this table from the wide format to the long format. Make sure the resulting table has two columns - AgWt and Instrument. Name this table AtmWtAg.long and print this table. If you choose SAS, you may need to a row number as ID - you might use the automatic variable _n_ in the DATA step; use the table name AtmWtAgLong.

Part c

To confirm that the table was reshaped correctly, use aggregate or tapply to calculate means from the long table and use apply or colMeans to calculate means from the wide table. Print and compare the results. If you choose SAS, see the instructions for Exercise 2, Part c.

Part d.

Convert the Instrument column in the long data set to a factor, then set eval=TRUE in the code chunk below. Compare the intercept to the mean of instrument 1, and the slope to the different between the two means. Use the PROC GLM block for SAS.

```
AtmWtAg.long$Instrument <- as.factor(AtmWtAg.long$Instrument)
summary(lm(AgWt ~ Instrument,data=AtmWtAg.long))
```