PSTAT 122

Lab 5

Due on Tuesday 11/5 at midnight

Please upload your Rmd file AND your pdf output file to Canvas.

Part 1: Simulation of non-normal data

As mentioned previously, simulation is a very useful technique in probability and statistics when we wish to investigate certain statistical properties.

Here, we will simulate non-normal data according to a Gamma distribution. Recall from e.g. PSTAT 120A and/or PSTAT 120B that if $X \sim Gamma(\alpha, \beta)$, then:

$$E(X) = \alpha\beta$$
$$Var(X) = \alpha\beta^2$$

where α and β are the "shape" and "scale" parameters in the probability density function of the Gamma distribution, and are also identified as such in the `rgamma` function that you can use to simulate data coming from a Gamma distribution.

Simulate data according to the following specifications and store it in a dataframe:

- Three groups, each with $n_i = 20$ and with each observation coming from a Gamma distribution
- Group 1 should have $\alpha = 0.5, \beta = 1$
- Group 2 should have $\alpha = 1, \beta = 2$
- Group 3 should have $\alpha = 2, \beta = 3$

Create a graphical representation of your simulated data, and also make a table where each row is labeled by the shape/scale combination chosen, and then columns for the actual sample means and sample variances from your simulated data (note that due to random chance they may or may NOT be close to the theoretical values and that is ok!)

We will then use this dataframe in the next section.

Part 2: ANOVA Permutation Test

(a)

Write a function to perform the ANOVA version of a permutation test. The inputs to this function should be a dataframe and the desired number of repetitions. Your function can expect that the dataframe it receives will have two columns: a quantitative outcome variable in the first column, and a categorical treatment variable in the second column.

(b)

Test your function on the dataframe created in the previous part. Use `reps=10000`. Compare your result to that obtained with the `aov` function and comment briefly on what you observe, including a statement about Type I or Type II Error if relevant, and why you might have observed what you did.

Part 3: Simulation Study

Now, we will perform a small simulation study in which we simulate data according to the specifications in Part 1, lots of times. Because a real simulation study would take a long time to run, we will bump things down for this assignment. However, what you do here forms the foundation of what you would do if you choose a simulation option for the project; in that case, you would indeed want to be prepared for your simulations to take a very long time, and to investigate many different scenarios.

Here, do the following:

- Use `reps=1000` (this will be less accurate, but again we need to do so for the sake of how long it will take to run).
- Using the setup that you created in Part 1, write a `for` loop to randomly create a new dataframe under that exact same setup (that is, you can literally copy the code you wrote for that part and place it into a `for` loop). Have your `for` loop run 100 times. Each time, run `aov` and your permutation test, and store the p-value.
- At the end, calculate the proportion of the time that `aov` returns a p-value < 0.05 and the proportion of the time that your permutation test returns a p-value < 0.05. Comment on your result, referring to either Type I Error rate, Type II Error rate, and/or Power as appropriate, based on what we know to be true about the simulated data.

You can expect that this simulation study will take about 1-3 minutes for 100 overall repetitions and 1000 reps for the permutation test (a timecheck on my machine gave me 67 seconds). While you are still testing it out and making sure that will actually run and look reasonable, I recommend that you bump both of these numbers down even more (e.g. perhaps 10 overall reps and 10 permutation test reps) so that you don't have to wait very long to see anything.

Furthermore, so that you don't have to wait that long every time you knit your Markdown file after you've successfully run it, there are two options:

1. You can set the option `cache=TRUE` in the header of your code chunk; that is,

````
```{r, cache=TRUE}
````

This will make it so that code chunk will ONLY be re-run if something in it has been changed since the last time you knitted; otherwise, it will just use the result from the last time you knitted.

2. Alternatively, you can code your entire simulation in an R script file, have the script file save your output, and then simply load the output into your R Markdown file. That is, in your script file, end it with:

```
save.image("name.RData")
```

and then in your Rmd file, include in a code chunk:
```

```
load("name.RData")
```

which should work just like this as long as this file is in the same directory as your .Rmd file.

Part 4: Other Exercises

Input data into R as needed for the following exercises.

Exercise 3.23. Start by making an appropriate graphical representation of the data. Then do parts (a) and (b). Use standard ANOVA for part (a). Be careful to make sure that R is treating the circuit type values as categorical (as opposed to numeric).

Exercise 4.8. Include a graphical representation of the data. Also be careful to make sure that R is treating the solution values and day values as categorical (as opposed to numeric).