Lab 5: Analysis of variance (ANOVA)

In this lab, you'll analyze bee SD interbeat interval data from the second paper of the course, using one-way ANOVA followed by Tukey's HSD post-hoc tests. Then, you'll analyze a phenotypic assay of LPS and BaP treatment on mice from the third paper of the course. You'll build hypotheses, manage data, generate figures, run tests, and interpret outputs. In addition, you'll learn a basic way to write model formulas for statistical tests in R, and use the following new functions:

```
pivot_longer()
ordered()
colMeans()
apply()
geom_errorbar()
geom_jitter()
aov()
summary()
TukeyHSD()
hist()
```

There are 13 question checkpoints in **bold**. Answer them directly in your R Markdown file, outside of the chunks.

Section 1: One-way ANOVA visualization

1. Open a new R Markdown file. Then, load in the packages we'll be using today from the code below. Remember that you can use `install.packages()` if either package is not already installed on your computer.

```
library("ggplot2")
library("tidyr")
```

2. Download and import `bee_interbeat.csv` from the course website. Assign it to the variable `IBI.wide`. These data are approximated from figure 3c.

3. We're going to use a one-way ANOVA to analyze this dataset. Due to the experimental design, we can assume that the data points are independent. As for normality and homogeneity of variance, the function we use to conduct our ANOVA will give us a warning if there's cause for alarm that these assumptions may be violated.

    - The ANOVA will test to see whether there is any difference in SD interbeat interval (IBI) between bees from the four differently-polluted sites.
    - **Q1. Why is an ANOVA more appropriate than a t-test here?**
    - **Q2. What are the complementary alternative and null hypotheses for this one-way ANOVA? Please label which hypothesis is which.**

4. If you take a look at the dataset, you'll notice that it's in a wide format. However, a long format will be useful in data visualization and building the ANOVA. To get the long format, use a function called `pivot_longer()`, which is part of the `tidyr` package and a counterpart to the `pivot_wider()` function you learned in an earlier lab. Below is the code you'll need.

    - `IBI.wide` is the main data frame input that you want to take from a wide format to a long format.
    - `1:4` are the columns from `IBI.wide` to be included; in this case, it's simply all the columns.
    - `names_to="site"` and `values_to="IBI"` specify the names of the new name and value columns respectively.

```
IBI.long <- pivot_longer(IBI.wide, 1:4, names_to="site", values_to="IBI")
```

5. Before running the test, visualize the data using a bar plot with error bars. For this, you'll need three columns of a dataframe: the x-axis values (sites), the y-axis values (means), and standard deviations from which to build error bars. Starting with the x-axis, create a vector that simply lists each site. However, ggplot likes to reorder categories in alphabetical order and it would be better to keep the sites in ascending order of pollution. Therefore, you'll need to add in a function called `ordered()` to the vector, which will tell ggplot what order to display. See below for how to use this function. The first `c()` vector is the actual vector being created, while the second `c()` vector specifies the order that ggplot will use.

```
site <- ordered(c("rural", "low", "moderate", "high"),
                c("rural", "low", "moderate", "high"))
```

6. Next you'll need the y-axis values, which will be the means for each site. While you could simply use the `mean()` function several times for each column, there's an easier way to achieve this. Use the `colMeans()` function with the wide-format `IBI.wide` as the input to get a vector consisting of each column's mean. Assign this to the variable `IBI.mean`

7. Finally, get the standard deviations. Although there's no function to specifically get standard deviations of each mean, there's still a function you can use for this called `apply()`. This function will take a given function and repeat it for the input array across each column or row, depending on which you specify.

    - Look up the documentation for `apply()` in the help tab and find the usage. You should see the arguments `X`, `MARGIN`, and `FUN`. `simplify` is also included, but you can leave that as the default.
    - `X` is the input, which should be wide-format `IBI.wide`
    - `MARGIN` specifies whether you want to repeat the function over rows or columns. You'll see from the documentation that "1 indicates rows, 2 indicates columns, c(1, 2) indicates rows and columns". Set `MARGIN=2` to repeat across columns.
    - `FUN` is the function that will be applied to each column in this case. Remember that the function for standard deviation is `sd()`. However, in `apply()`, you want to make sure that you don't include the parentheses in the argument, so simply set `FUN=sd`.
    - Assign the resulting vector to the variable name `IBI.sd`

8. Combine the x-axis (site), y-axis (mean), and standard deviation vectors into one data frame and rename the columns if you'd like. This way, you have a data frame to input into ggplot.

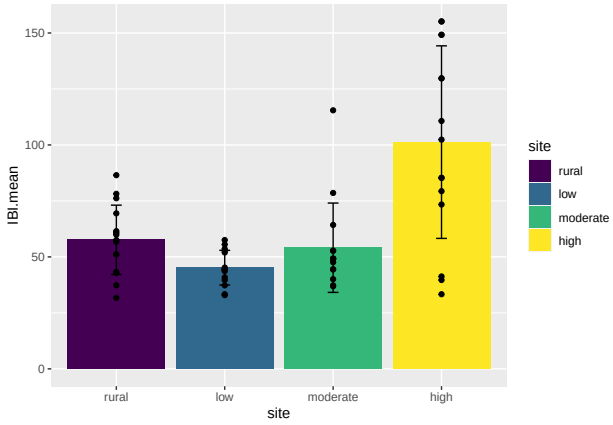
9. Now it's time to make the plot. Create a new ggplot with the data frame from the last step as the `data` argument, the site column as the x, and the mean column as the y. Put these in the `ggplot()` line so that the other layers will inherit it. Add a `geom_col()` layer below that, and optionally add `aes(fill=site)` to make it a little prettier.

10. This looks good, but it doesn't tell us anything about the spread of the data, so it would be good to add some error bars using `geom_errorbar()`. Below is the base code for an error bar layer, with some blanks that you need to fill in yourself. Delete the underscores and replace them with the appropriate inputs, detailed below.

    - `ymin` is the lower range for the error bars. This should be the mean minus the standard deviation. Luckily, you have columns for both of these values. Since they match up in the data frame, you can simply specify that you want the mean column - the standard deviation column and ggplot will know to match the position of each to the correct position on the x-axis.
    - `ymax` is the upper range. Do the same as you did for `ymin`, but this time you need to add the mean and standard deviation together.

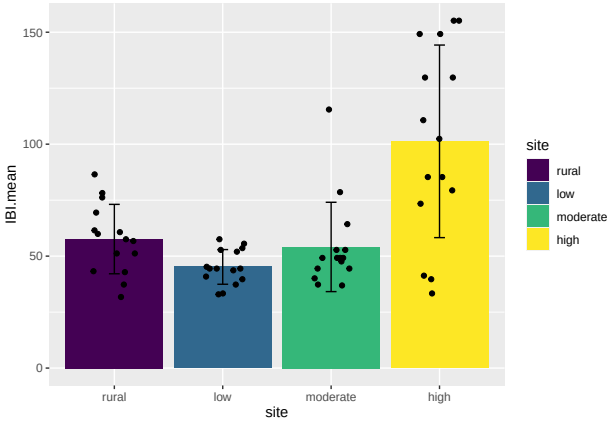
```
geom_errorbar( aes(ymin = _______,
                   ymax = _______,
                 width=0.1 )
```

11. Time to add one final layer! Don't forget the `+` symbols before each preceding layer each time you add a new one. For an even better look at the data, add a scatterplot on top. This will need a separate data frame input since you need to input each individual data point instead of only the means. You created `IBI.long` earlier with this information, so use that as the `data` argument in the `geom_point()` scatterplot layer. This will overwrite the `data` argument that was inherited from the `ggplot()` layer, but for only the scatterplot layer. Within `aes()`, the `x` should be `site` and the `y` should be `IBI`.



12. If you run the current plot, a lot of the points will be overlapping making it difficult to see. To fix this, simply change your scatterplot layer to `geom_jitter()` with the same exact inputs. This will add a jitter effect to the points so they don't overlap as much.

    - Now, in the geom_jitter() inputs, add `width=0.2, height=0` so that the jitter effect isn't too strong.
    - **Q3. Your plot is now complete! Using only visual inspection without running any tests, which pairs of sites do you think have a significant difference?**



Section 2: One-way ANOVA

13. It's finally time to run an ANOVA. The function here is `aov()`, which stands for Analysis Of Variance (remember that ANOVA is also ANalysis Of VAriance). If you look at the documentation, you'll see that it's asking for a `formula` as a requirement, and you'll also want to specify the `data` argument so that it knows where to get data from.

    - In R, formulas that specify a model use the tilde, `~`, to designate the dependent and independent variables. Dependent variables go to the left of the tilde, while independent variables go to the right of the tilde. Since we're interested in the variation in IBI (dependent variable) compared to the different sites (independent variable), our formula will look like this: `IBI ~ site`.
    - The `data` argument is `IBI.long`, since the long formate contains a column for each of the two variables.
    - Assign the results of `aov()` to the variable name `IBI.anova` for later use. Because you're assigning to a variable, R shouldn't output anything yet when you run this.

14. If you type `IBI.anova` you'll get an output that's not super helpful, but it does contain one line: "Estimated effects may be unbalanced". This means that you need to investigate the assumptions of normality and variance homogeneity more closely. If this had said that they were balances, that lets you know that you don't have to worry about violation of assumptions of normality and homogeneity of variances.

    - Test for normality using a histogram of the residuals of the ANOVA, like so: `hist(IBI.anova$residuals)`. This histogram isn't as pretty as the ggplot ones, but it's good enough for a quick inspection of the distribution.
    - The easiest way to bypass the homogeneity of variances assumption is to have equal sample sizes for each group, which can be achieved through the experimental design.
    - **Q4. How many observations are there in each group for the bee IBI data?**

15. Although looking at the variable itself didn't help too much, the actual best way to look at the results of `aov()` is the `summary()` function. Copy and paste in the below code, then run it to see the outputs.

```
summary( IBI.anova )
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## site         3  28209    9403   14.76 3.37e-07 ***
## Residuals   56  35665     637
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

16. Time to interpret this. Df is the degrees of freedom, Sum Sq is the sum of squares (each value subtracted by the mean, squared, and then all summed up), Mean Sq is the mean of squares (Sum Sq divided by Df), F value is the F value, and Pr(>F) is the P-value. There's a row for `site`, which is the sites you're comparing (the independent variable), and `Residuals`, which is like the error of the model (how far your values are from the mean).

    - **Q5. How would you calculate the degrees of freedom for the independent variable, `site`? (This comes out to 3)**
    - **Q6. How would you calculate the degrees of freedom for the residuals? (This comes out to 56)**
    - **Q7. Based on the p-values and an alpha of 0.05, would you reject the null hypothesis? In your own words, what does this mean?**

17. Notice that the ANOVA gave us insight into whether there was significant variance between the sites, but it didn't tell us which sites significantly varied with which. We'll have to run a test on each possible pair of comparisons to figure that out, but remember that running tests so many times is bound to give you a significant P-value due to random chance. So, we're going to use Tukey's method to correct for that. The function is `TukeyHSD()`, and the input in the parentheses should be the results from the ANOVA, `IBI.anova`.

    - Note: The HSD in Tukey HSD stands for Honest Significant Differences.

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = IBI ~ site, data = IBI.long)
##
## $site
##                       diff       lwr       upr     p adj
## low-high       -56.084656 -80.48483 -31.68448 0.0000007
## moderate-high  -47.169312 -71.56949 -22.76914 0.0000228
## rural-high     -43.650794 -68.05097 -19.25062 0.0000881
## moderate-low     8.915344 -15.48483  33.31552 0.7683603
## rural-low       12.433862 -11.96631  36.83404 0.5360140
## rural-moderate   3.518519 -20.88166  27.91869 0.9808653
```

18. The `$site` column lists what each comparison is, so `low-high` is a comparison between the low-pollution site and the high-pollution site. `diff` is the difference in means between the two groups, while `lwr` and `upr` give you the range for the confidence intervals (currently set to 95% confidence by default, which you can see in the second line of output). `p adj` lists the p-values for each comparison, after adjustment due to multiple comparisons.

    - **Q8. In a paragraph or two, write up a final report for the results of your ANOVA and Tukey's HSD tests. Use this example from the last slide of the lecture as a model, and make sure you include: description of the variables, type of tests run, test statistics, p-values, whether there is significance, degrees of freedom, and sample size. (Not necessarily in that order). Normally you need to include the specific results too, such as the mean differences, but that can be seen in the plot you made earlier.**

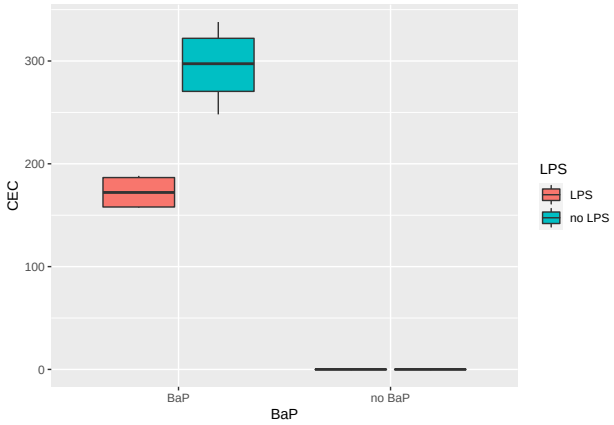
Section 3: Two-way ANOVA

19. For the next dataset, we'll be looking at a phenotypic assay from the third paper of the course. This assay measures the activity of 3-cyano-7-ethoxycoumarin (CEC) under four different conditions: a control, lipopolysaccharide exposure (LPS, to induce inflammation), benzo[a]pyrene exposure (BaP, a

PAH), and both LPS + BaP exposure. CEC is the substrate of CYP1A enzymes, making it possible to estimate phenotypic variance of the different treatments from the lens of *Cyp1a* gene expression. Download and import `Shi_CEC.csv` from the course website, then assign it to a variable. These data are simulated based on the means and standard deviations in the paper, with 4 mice in each group.

- Inspect the data frame to see a column for the treatment, a column for CEC activity (in RFU/min, where RFU stands for Relative Fluorescence Unit), a column indicating whether BaP is present, and a column indicating whether LPS is present.

20. Before trying ANOVA, visualize the data using a box plot. Although you could test the data using only the `treatment` column, testing `BaP` and `LPS` separately will allow you to see how and whether they interact. Since there are 2 independent variables this time (BaP treatment and LPS treatment), the boxes of the plot need to be clustered accordingly. Using the CEC assay data frame, make a box plot with `BaP` on the x-axis, `CEC` on the y-axis, and `LPS` as the "fill" argument, all within the `aes()` function. For box plots, using `fill` will automatically cluster the groups.



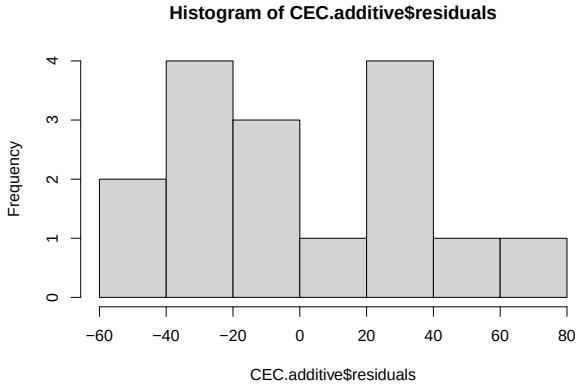
21. Assume an alpha of 0.05, and run a two-way ANOVA. This is done similarly to a one-way ANOVA, but the formula is a little different. In a one way ANOVA the formula was `dependent ~ independent`. For a two-way, we have two independent variables, so we need to combine them using either `+` or `*`. Using the `+` will test every combination, but it won't test for the interaction between independent variables. On the other hand, `*` does test for that interaction. Try a version of each, so two two-way ANOVAs total.

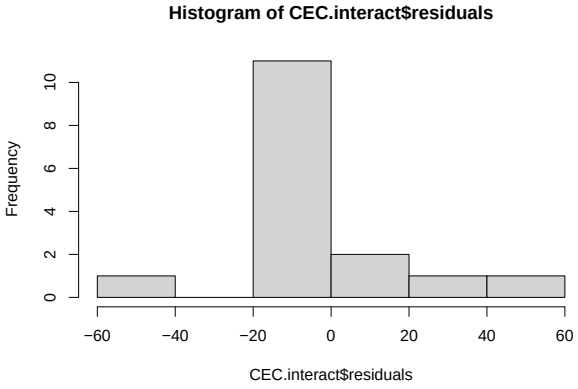
- Remember that the function is `aov()`, and that the inputs should be a model formula and a data frame, so the format should be something like 'aov(MODEL.FORMULA.HERE, data = DATA.FRAME.HERE)
- Again, use `dependent ~ independent1 + independent2` for the model formula without interaction, and `dependent ~ independent1 * independent2` to include interaction. Then, use the CEC data frame for the data argument of both.
- Assign the results of the ANOVA to variables called `CEC.additive` and `CEC.interact` respectively so we can use them later.
- **Q9. What are the three pairs of complementary alternative and null hypotheses for the ANOVA with interaction (no blocking) that compares CEC activity as the dependent variable, and BaP and LPS as the independent variables? Please specify which variables are being compared in your hypotheses, label each hypotheses by alternative/null and pair.**
- **Q10. What are the two pairs of complementary alternative and null hypotheses for the additive ANOVA that compares CEC activity as the dependent variable, and BaP and LPS as the independent variables? Please specify which variables are being compared in your hypotheses.**

22. Before viewing the results, check the assumptions for an ANOVA:

    - Independence: this holds true thanks to proper experimental design.
    - Normality: use a quick histogram of the residuals (the error, or distance from the mean for each observation value) to check for this, by copying the code below. They're not perfectly normal, but they should be close enough.
    - Homogeneity: there is an equal number of observations in each group, so this is fine.

```
hist(CEC.additive$residuals)
hist(CEC.interact$residuals)
```





23. Use `summary()` to take a look at the results, just as you did for the one-way ANOVA. There should be two lines here: one for the additive ANOVA, and one for the ANOVA with interaction.

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## BaP          1 218704  218704 137.145 2.8e-08 ***
## LPS          1  15055   15055   9.441  0.0089 **
## Residuals   13  20731    1595
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## BaP          1 218704  218704  462.38 5.98e-11 ***
## LPS          1  15055   15055   31.83 0.000109 ***
## BaP:LPS      1  15055   15055   31.83 0.000109 ***
## Residuals   12   5676     473
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

24. The first column of the output lists either an independent variable, an interaction between independent variables (denoted by a colon, `:`), or `Residuals`. The rest of the output is the same as the one-way ANOVA.

    - **Q11. Interpret the output for the interaction ANOVA specifically in one short paragraph. Please mention BOTH the P-values and F-values in your interpretation.**

25. Finally, use Tukey's HSD as a post-hoc test on each ANOVA. Remember that the function is `TukeyHSD()` and the input should be the results from your ANOVAs, which you should have saved as `CEC.additive` and `CEC.interact`. These should be two separate lines of code.

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = CEC ~ BaP + LPS, data = CEC.df)
##
## $BaP
##                  diff       lwr       upr p adj
## no BaP-BaP -233.8291 -276.9647 -190.6935     0
##
## $LPS
##                diff      lwr      upr     p adj
## no LPS-LPS 61.34955 18.21394 104.4852 0.0089045
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = CEC ~ BaP * LPS, data = CEC.df)
##
## $BaP
##                  diff       lwr       upr p adj
## no BaP-BaP -233.8291 -257.5219 -210.1363     0
##
## $LPS
##                diff      lwr      upr     p adj
## no LPS-LPS 61.34955 37.65674 85.04235 0.0001087
##
## $`BaP:LPS`
##                                     diff        lwr        upr p adj
## no BaP:LPS-BaP:LPS       -1.724796e+02 -218.13658 -126.82260 5e-07
## BaP:no LPS-BaP:LPS        1.226991e+02   77.04210  168.35608 2e-05
## no BaP:no LPS-BaP:LPS    -1.724796e+02 -218.13658 -126.82260 5e-07
## BaP:no LPS-no BaP:LPS     2.951787e+02  249.52169  340.83567 0e+00
## no BaP:no LPS-no BaP:LPS  1.332268e-14  -45.65699   45.65699 1e+00
## no BaP:no LPS-BaP:no LPS -2.951787e+02 -340.83567 -249.52169 0e+00
```

26. This time, you'll notice that the output for ANOVA with interaction is much longer than it was for the one-way ANOVA. This is because it's also testing the interactions between independent variables.
    - **Q12. What differences do you notice between the additive ANOVA and the ANOVA with interaction? Look at both the ANOVAs and post-hoc tests to contrast the two ANOVA types.**
    - **Q13. What conclusions could be drawn from specifically the test with interaction? Use specific values to back up your claims, from both the ANOVA results and the post-hoc test results.**
27. To wrap up, knit your file to PDF or HTML when you're done and make sure you've answered Q1-Q13 in the R Markdown.

Reference

Thimmegowda, G.G., Mullen, S., Sottilare, K. et. al. **A field-based quantitative analysis of sublethal effects of air pollution on pollinators.** Proceedings of the National Academy of Sciences, Aug 2020, 117 (34) 20653-20661; DOI: 10.1073/pnas.2009074117

Shi, Q., Fijten, R.R., Spina, D. et. al. **Altered gene expression profiles in the lungs of benzo[a]pyrene-exposed mice in the presence of lipopolysaccharide-induced pulmonary inflammation.** Toxicol Appl Pharmacol, Dec 2017, 336:8-19. doi: 10.1016/j.taap.2017.09.023. PMID: 28987381; PMCID: PMC5703654.