Linear Modeling with More than Two Categorical Predictor Variables

ANTH 3720-001 Archaeological and Forensic Science Lab Methods: Data Analysis with R

Elic Weitzel

Jan. 29-31, 2021

If you would like the original R Markdown file, find it on my GitHub page at https://github.com/weitzele/Basic-R-Tutorial

1 Linear Modeling with More than Two Categorical Predictor Variables

Now that you have begun to understand linear modeling, we can start to add some complexity. We started off using the `lm()` function with just two categorical predictor variables, which is also known as a t-test. But often, you will find that you have values of more than just two groups that you want to compare. The `lm()` function can handle this quite easily.

1.1 Prepare the Data

As an example, let's return to our trusty `archdata` package.

```
library(archdata)
```

Since we're simply adding an extra categorical predictor variable to what we learned in the t-test tutorial, let's stick with the same dataset as well.

```
data(BarmoseI.pp)
```

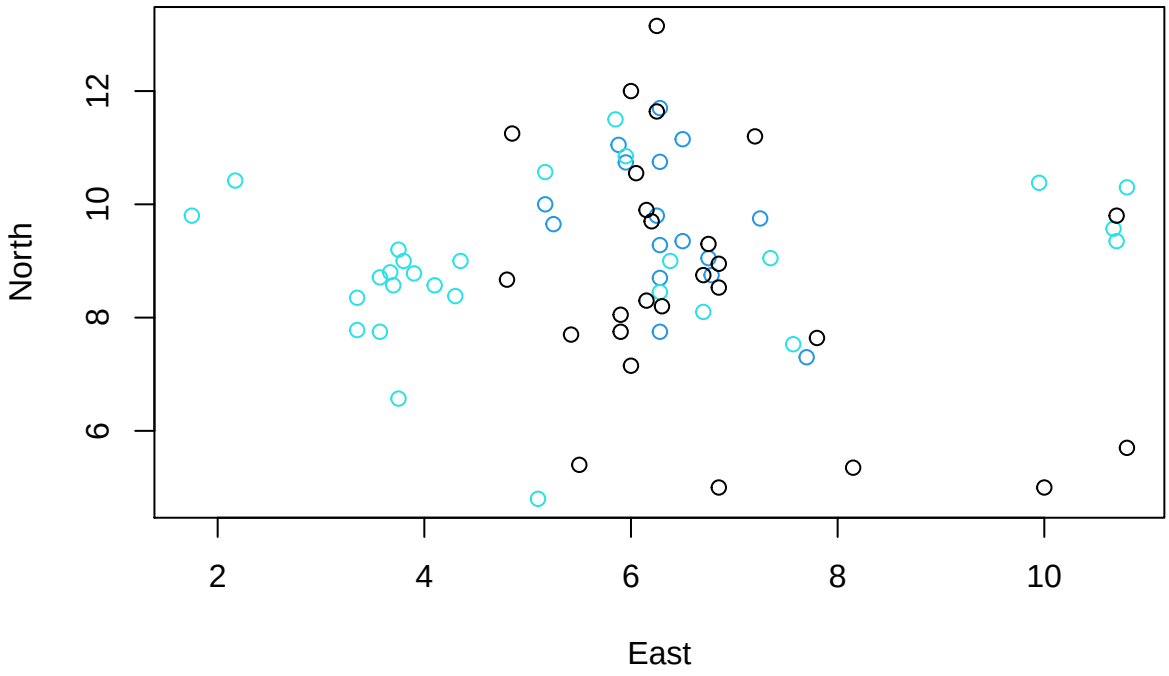
Now, let's pull out three of the artifact types from `BarmoseI.pp`. We'll do this using indexing (`data.frame[rows, columns]`) and logical operations (`column.name == "Artifact"`) as before, and we'll create a new object that contains only the columns for these three artifact types.

Note that here, we're saying "extract all rows from BarmoseI.pp for which the value in the `Label` column is equal to"Microburins","Flake Axes", or"Denticulated/Notched Pieces". In R, and many other languages,"or" is denoted by a `|` symbol. And of course, as before, we're specifying all of these logical operations within the `[]` but before the comma, because these apply to the rows of the data frame. We want only the rows for which `Label` is equal to one of those three artifact types. And then we leave the indexing empty after the comma because we want all columns to be returned.

```
barm.data <- BarmoseI.pp[BarmoseI.pp$Label == "Microburins" |
                         BarmoseI.pp$Label == "Flake Axes" |
                         BarmoseI.pp$Label == "Denticulated/Notched Pieces", ]
```

Let's plot our artifact locations before we model anything. It's always important to visualize your data preliminarily to make sure that things look in order. If for example, northing and easting coordinates got flipped or some values had a large number added to them by mistake, we could spot such errors by visualization.

```
plot(North ~ East, data = barm.data, col = Label)
```

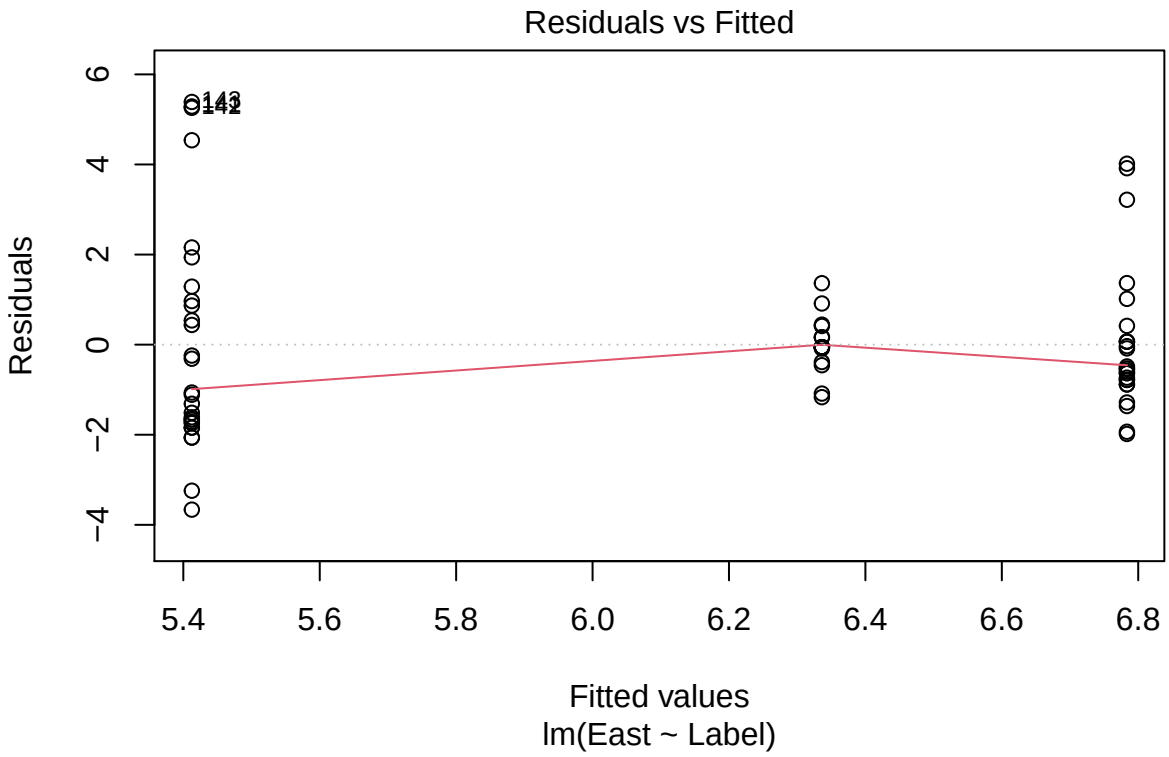


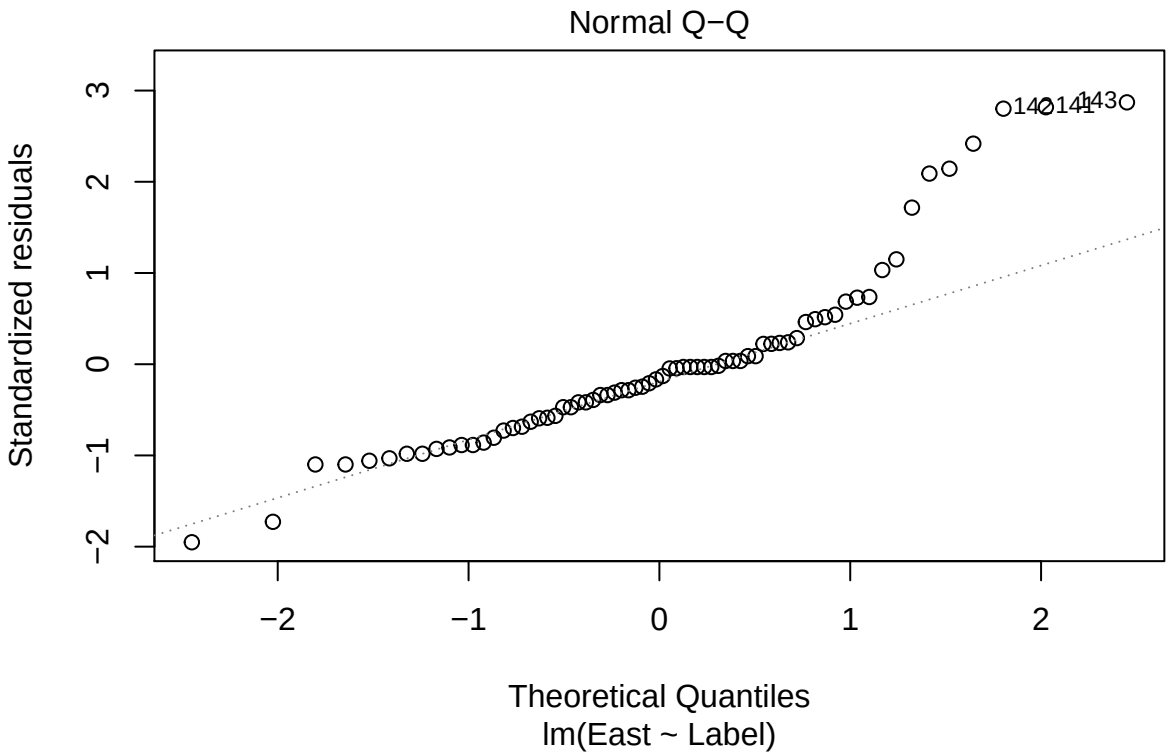
1.2 Build a Model

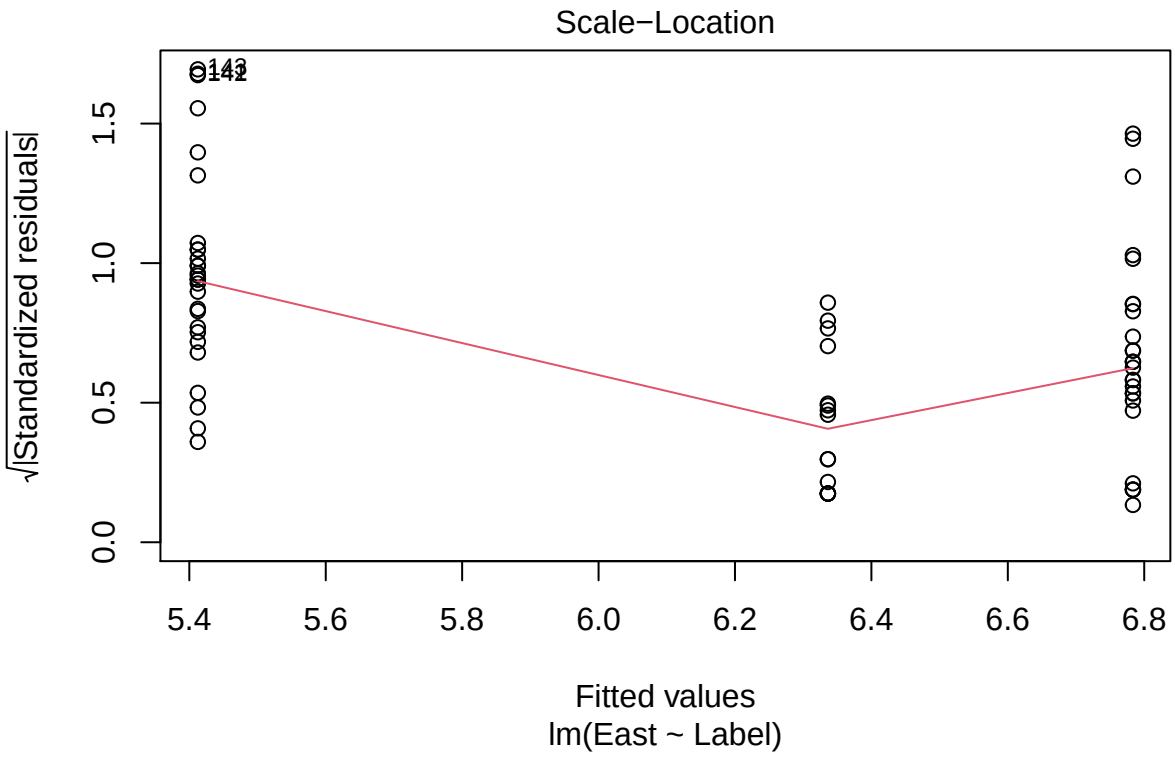
Now that we've plotted our artifact locations, let's make a model to determine whether the easting coordinates for all three types are the same. Once we make the model, let's also plot the model diagnostic plots.

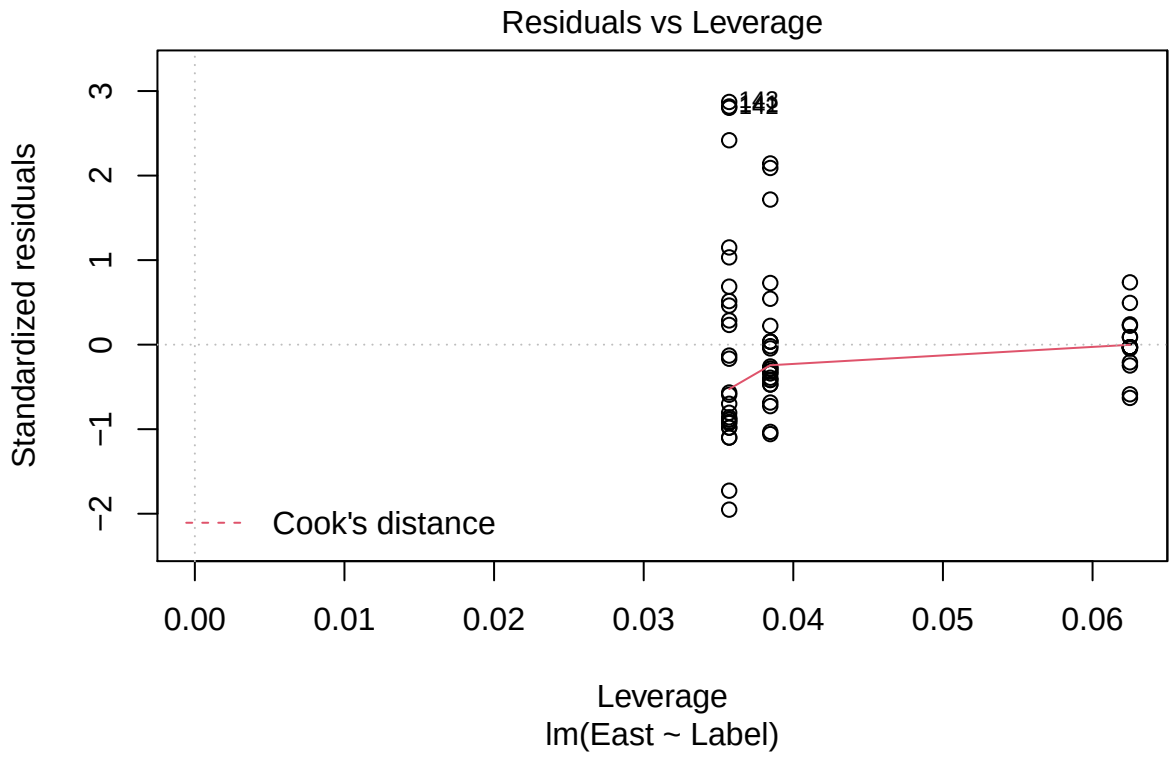
```
barm.mod <- lm(East ~ Label, data = barm.data)

plot(barm.mod)
```


Residuals vs Fitted
Residuals
6
4
2
0
−2
−4
5.4
5.6
5.8
6.0
6.2
6.4
6.6
6.8
Fitted values
lm(East ~ Label)


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(East ~ Label)
142
141
143
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
5.4
5.6
5.8
6.0
6.2
6.4
6.6
6.8
Fitted values
lm(East ~ Label)



The diagnostic plots look okay. We have some deviation from normality at high values, based on the Q-Q Plot, but this shouldn't be too much of a problem as the vast majority of points do fall along the dotted line. But this is the kind of thing you should be keeping an eye on. If you end up with results that seem a bit odd, you might want to rethink this assumption of normality.

Now that we know our model is looking okay, we can inspect the model summary.

```
summary(barm.mod)
```

```
##
## Call:
## lm(formula = East ~ Label, data = barm.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6625 -1.1528 -0.2775  0.4422  5.3875
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       6.3363     0.4778  13.262   <2e-16 ***
## LabelFlake Axes                  -0.9238     0.5989  -1.542    0.128
## LabelDenticulated/Notched Pieces  0.4472     0.6072   0.736    0.464
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.911 on 67 degrees of freedom
```

```
## Multiple R-squared:  0.0967, Adjusted R-squared:  0.06974
## F-statistic: 3.586 on 2 and 67 DF,  p-value: 0.03314
```

Because this summary output is still based on the `lm()` function, it can be interpreted in the same way as the previous summary outputs we've seen for t-tests.

The biggest difference is that there is now an extra row of coefficients. We see that there is a row for `(Intercept)`, a row for `LabelFlake Axes` and now a third row for `LabelDenticulated/Notched Pieces`. The `Estimate` is what we care most about here as it tells us the direction of the relationship. This can be interpreted like the coefficients for the t-tests previously. The estimated mean easting value for Flake Axes is 6.3363 - 0.9238 while the estimated mean for Denticulated/Notched Pieces is 6.3363 + 0.4472.

If you're still skeptical, we can compare these estimated means to the actual means.

```
6.3363 #the intercept (corresponding to the first artifact type we specified)
```

```
## [1] 6.3363
```

```
with(barm.data[barm.data$Label == "Microburins",],
     mean(East)) #calculate the mean East value with the Microburins from barm.data
```

```
## [1] 6.33625
```

```
6.3363 - 0.9238 #the intercept plus the coefficient for flake axes
```

```
## [1] 5.4125
```

```
with(BarmoseI.pp[BarmoseI.pp$Label == "Flake Axes",],
     mean(East)) #calculate the mean East value with the Flake Axes from barm.data
```

```
## [1] 5.4125
```

```
6.3363 + 0.4472 #the intercept plus the coefficient for D/N Pieces
```

```
## [1] 6.7835
```

```
with(BarmoseI.pp[BarmoseI.pp$Label == "Denticulated/Notched Pieces",],
     mean(East)) #calculate the mean East value with the D/N Pieces from barm.data
```

```
## [1] 6.783462
```

It is vitally important to remember that the p-values in the column labeled `Pr(>|t|)` are NOT the p-values for whether these artifact types have different easting values from each other!! The p-values here are simply expressions of how different each coefficient estimate is from 0. This is called a Wald Test. It is a common mistake to interpret these p-values as if they mean something for the overall ANOVA, but they don't! DO NOT look at these p-values to make any inferences about whether Flake Axes and Denticulated/Notched Pieces have different easting coordinates!

1.3 Analysis of Variance (ANOVA)

Again, as before, you may not have realized it but what you just ran was an ANOVA!

ANOVA stands for analysis of variance and, as with t-tests, an ANOVA is really just a specific type of linear model: one with a normally distributed response variable and categorical predictor variables.

Like t-tests, there are several different varieties of ANOVAs that we won't really get into. Once you understand the basics of linear models, you'll be able to read about them all yourself and it should make sense!

1.4 Interpreting the Model

To find the real p-value that we care about, we need to run the `anova()` function on our linear model.

```
anova(barm.mod)
```

```
## Analysis of Variance Table
##
## Response: East
##           Df  Sum Sq Mean Sq F value  Pr(>F)
## Label      2  26.197 13.0985  3.5863 0.03314 *
## Residuals 67 244.706  3.6523
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here, we see that our p-value is 0.03314, which is significant at the 0.05 level. To report this result, we would say something like:

An analysis of variance determined that there is a significant difference in easting values between these three artifact types ($F = 3.586$; $df = 2, 67$; $p < 0.05$). Alternatively, you can report the F statistic and degrees of freedom together as *$F(2, 67) = 3.586$.*

Just as you should report the t statistic and degrees of freedom for a t-test, you should report the F statistic and degrees of freedom for an ANOVA. For ANOVAs, there are actually two different degrees of freedom, and you should report both. The p-value is based on these numbers, and though we won't get into what this really means here, a p-value of 0.3314 is what you get from an F distribution at 2 and 67 degrees of freedom.

And actually, this p-value was reported in the original summary output for the linear model: look at the very bottom right value.

```
summary(barm.mod)
```

```
##
## Call:
## lm(formula = East ~ Label, data = barm.data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6625 -1.1528 -0.2775  0.4422  5.3875
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     6.3363     0.4778  13.262   <2e-16 ***
```

```
## LabelFlake Axes                    -0.9238     0.5989  -1.542    0.128
## LabelDenticulated/Notched Pieces    0.4472     0.6072   0.736    0.464
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.911 on 67 degrees of freedom
## Multiple R-squared:  0.0967, Adjusted R-squared:  0.06974
## F-statistic: 3.586 on 2 and 67 DF,  p-value: 0.03314
```

So now we know how to calculate a p-value for an ANOVA, and how to report this test result!

In R, there is also a built-in ANOVA function that is not actually the same as the `anova()` function. This can get a bit confusing... The `anova()` function requires its input to be an `lm()` object, and will return the results of an ANOVA. But there is also a built-in ANOVA function in R for which you can simply specify your data as you would in a linear model, and the result is an ANOVA. This function is `aov()`, and returns the same results as `anova(lm)`.

```
aov.obj <- aov(East ~ Label, data = barm.data)
summary(aov.obj)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## Label        2   26.2  13.098   3.586 0.0331 *
## Residuals   67  244.7   3.652
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2 Post Hoc Tests and the Multiple Comparisons Problem

Now, after running this ANOVA, we know that our three artifact types do not have the same easting coordinates. But this ANOVA doesn't actually tell us which one, or ones, are driving this result. Perhaps all three are very different from each other, or perhaps two are similar but one is different. Both of these scenarios would lead to a significant ANOVA result, but an ANOVA alone doesn't tell us which case we're dealing with.

We need to perform another statistical test.

If you get a significant result from an ANOVA, you can then run what is called a **post hoc test**.

There are many different varieties of post-hoc tests that work well with ANOVAs. Here, I'll just introduce you to one common variety, the **pairwise t-test**. But first, let's discuss why we need to do all of this.

2.1 Multiple Comparisons

A common beginner's mistake when running ANOVAs is to simply run a series of t-tests on all possible combinations of your data. In the example we're using here, we have three artifact types and we're interesting in comparing their easting values. Folks often assume that you could simply run a t-test comparing Microburins and Flake Axes, a second t-test on Microburins and D/N Pieces, and a third t-test on Flake Axes and D/N Pieces. In this case, you would get three p-values telling you whether Microburins and Flake Axes have statistically different mean easting values, and the same for Microburins and D/N Pieces and Flake Axes and D/N Pieces.

There is a problem here, however, that makes such an analysis incorrect. It's called the **multiple comparisons problem**. In essence, whenever you run more than one statistical test on the same set of data, you're increasing the likelihood of encountering a false positive and your p-value is no longer accurate.

To make sense of this, think back on the definition of a p-value. A p-value expresses the probability that you would obtain your result, or a more extreme result, if the null hypotheses of no difference were true (and if the sampling were replicated an infinite number of times).

So what we're calculating in a p-value is the probablity of getting your result if there's actually no real difference. If we're comparing the easting values of Microburins and Flake Axes, a p-value of 0.05 means there's only a 5% chance you'd get the easting values in our dataset if these two types of artifacts actually have the same easting distribution. This means that it's pretty unlikely such a thing would happen, so we could reasonably conclude that these two artifacts are not distributed the same.

For this reason, you can often (simplistically) think of a p-value as a false positive rate if the null is true. Only 5% of the time would you get a false positive (a significant result) if there isn't actually a difference.

But if we then compare the easting coordinates of Microburins to D/N Pieces and calculate a p-value, we hit a snag. We've now run another test on the same data (Microburins), and now our chance of getting a false positive is no longer the same as it was.

There's an xkcd comic about this phenomenon that might help to explain the issue: https://xkcd.com/882/

In this comic strip, scientists run twenty different tests to explore the relationship between consuming different jelly bean colors and developing acne. Nineteen of these tests have non-significant p-values, but one of them - for green jelly beans - has a p-value of less than 0.05. The newspapers run with this, saying that green jelly beans are linked to acne. But are they really?

Reflecting on our understanding of p-values as a sort of false positive rate, our cutoff point of 0.05 means that 5% of the time, we'll find a significant result when there actually isn't one. This means we'll make a mistake 5% of the time if we keep replicating our analysis on the same data!

So in this comic, that's what is happening. They keep replicating their analysis on the same response variable (acne) with twenty different predictor variables (jelly bean colors). One of these twenty predictor variables has a p-value of less than 0.05. But 1 out of 20 is 5%... This is likely simply a false positive!

This is exactly the mistake we'd be making if we ran a bunch of t-tests comparing all the different possible combinations of our data. We'd be operating under the assumption that our significance cutoff is still 0.05, but in actuality, the more tests we run, the more that cutoff changes. A simple method for recalculating this significance threshold is called the Bonferroni correction. Don't worry about the math behind it, but it basically recalculates what a significant p-value would have to be based on how many tests you've run. In the jelly bean and acne example, with twenty tests and a Bonferroni-corrected significance cutoff, our actual alpha value for significance would be 0.00256! That's much smaller than 0.05! This means that if we get a p-value of 0.01 for the relationship between green jelly bean consumption and acne, that's not actually a significant p-value anymore! It would have to be less than 0.00256 for it to be significant now.

This is why we have to run an ANOVA when comparing more than two groups, and why we then need a particular type of post hoc test if we get a significant ANOVA result. Otherwise, our interpretations of our p-values become meaningless and our conclusions could end up being very wrong...

2.2 The Pairwise T-Test

So, if you get a significant p-value from an ANOVA, you can then do a post hoc test. Don't do a post hoc test if your ANOVA is not significant.

As stated above, there are many different types of post hoc tests, but a common one is simply a pairwise t-test with a p-value correction. This is basically doing what I just warned you not to do, but with a special type of function that internally recalibrates your p-values.

This can be done in R using the `pairwise.t.test()` function.

```r
pairwise.t.test(barm.data$East, barm.data$Label, p.adjust.method = "bonf")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  barm.data$East and barm.data$Label
##
##                             Microburins Flake Axes
## Flake Axes                  0.383       -
## Denticulated/Notched Pieces 1.000       0.031
##
## P value adjustment method: bonferroni
```

You can inspect the help page for this function with `?pairwise.t.test`. In this function, we specify the easting coordinates as the vector of data which will be analyzed. Then, we specify that the groups we're interested in are based on the Label column. Finally, we specify that we want to adjust our p-values using a Bonferroni correction, which is "bonf" here.

This results in an output of three different p-values presented in a table. We can now see that the easting values for our three artifact types are not all different from each other: only one pair is driving our significant ANOVA result.

The Bonferroni-adjusted p-value for a t-test comparing Microburins to Flake Axes is 0.383: not significant. The same is true for the p-value of Microburins and D/N Pieces. However, the adjusted p-value for Flake Axes and D/N Pieces is 0.031: a significant result! Now we know that while these three artifact types do not all share the same distribution of easting values (based on our ANOVA), it's really flake axes and D/N pieces that have different easting coordinates. You can then report these p-values after your ANOVA results, but remember that you can't run a post hoc test on a non-significant ANOVA, and you must always run an ANOVA first.