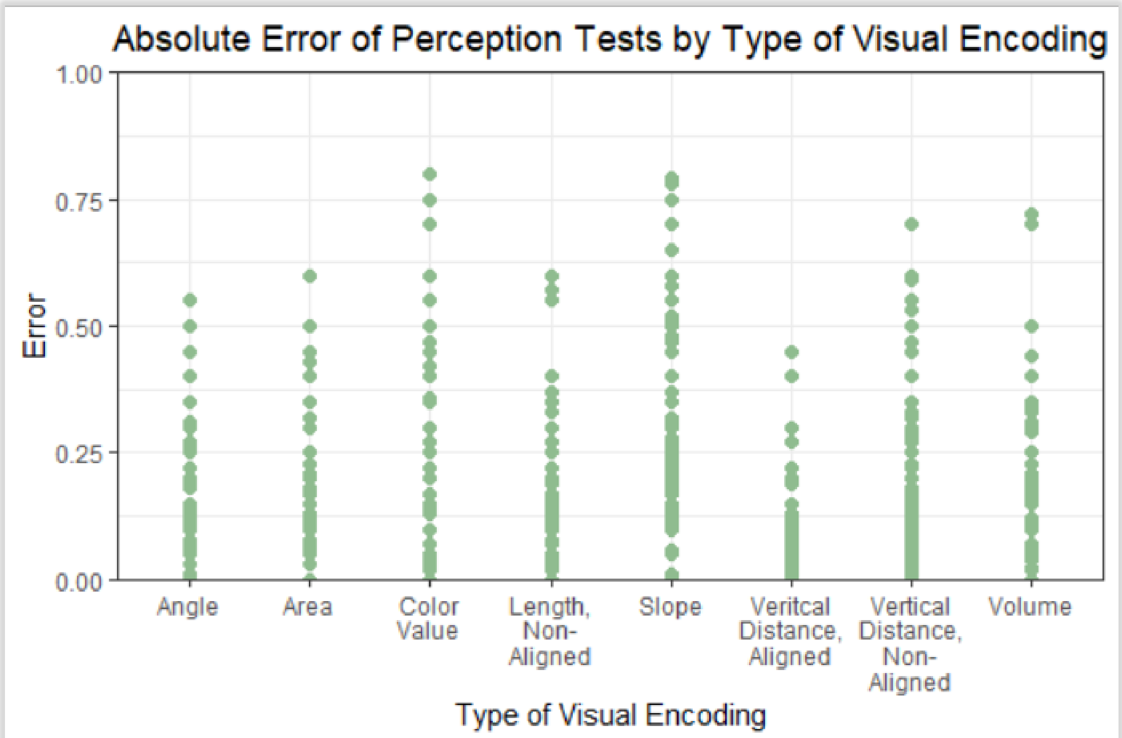
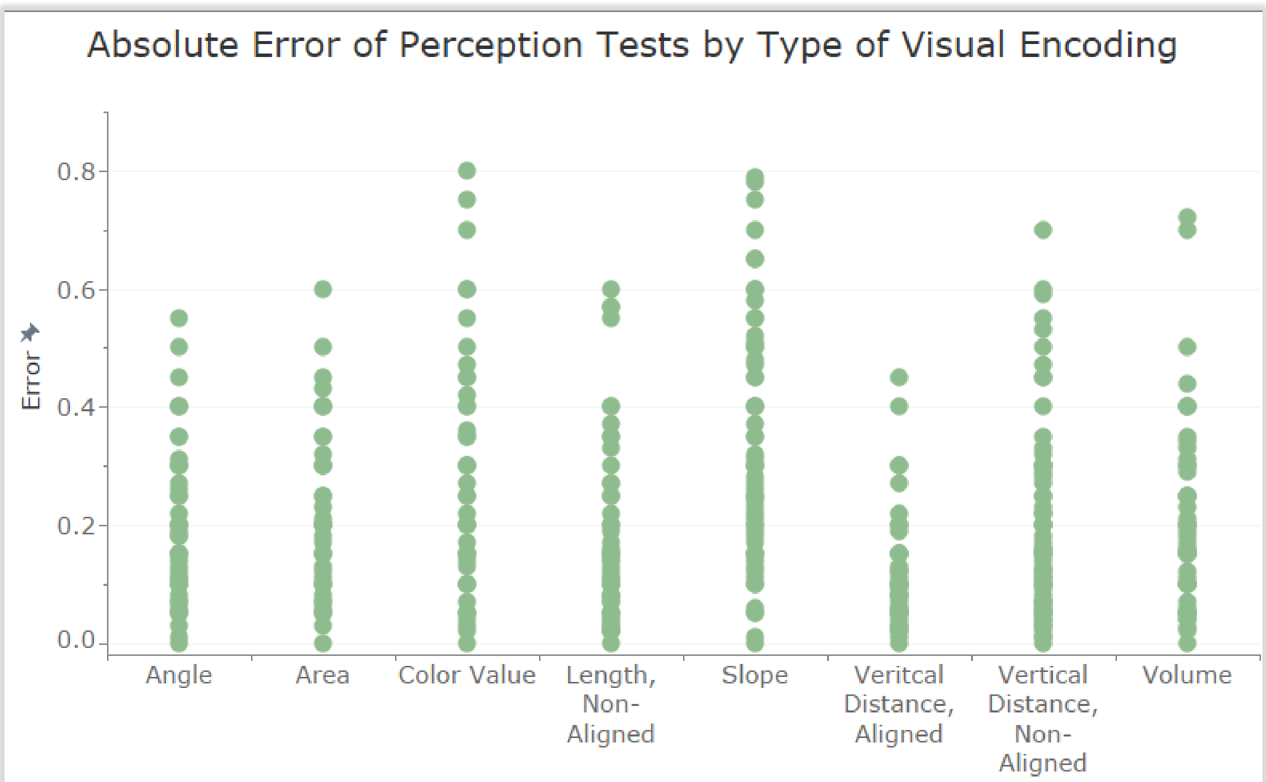
1a.

1a. R Studio This graph was made with test on the x-axis and absolute error on the y axis. By graphing the occurrences of each error, we can see the distribution of error, and determine where outliers might occur and sway the median graphs plotted in hw 1.



1a. Tableau This graph has Test in the columns, and ABS([Error]) in Rows. Finally implemented "Format Workbook" so now all graphs will be in verdana with these lovely axis ticks and rulers.



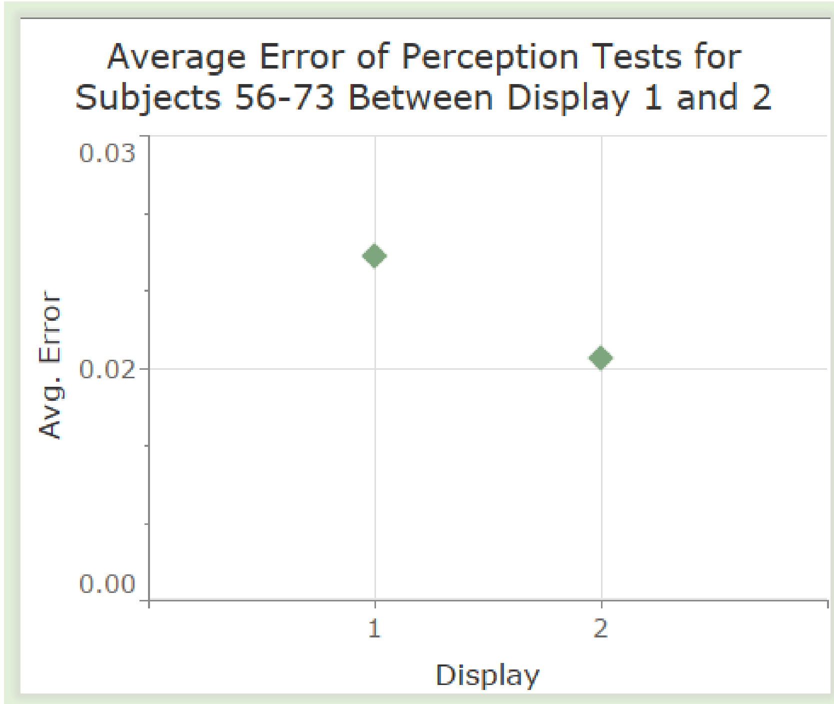
From these graphs it is clear that vertical distance aligned had the least error, with all occurrences of error hovering very close to 0. Color value has the most variation, with a few points far above the clustered errors, which could be due to forms of color blindness in participants.

1b.

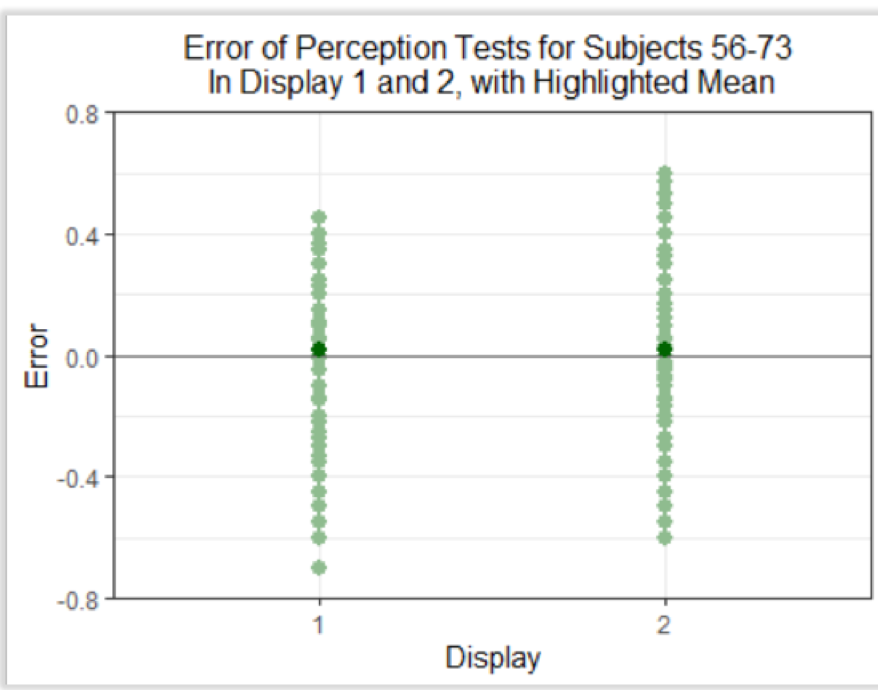
Graphing the same graphs as above, with error instead of absolute error, overlapped with a dot for the mean error per test, visually displays what neither of these graphs could show alone. This graph shows that while Slope has distribution from 0.7 to -0.7, the clustering around 0.2 – 0.3 is more intense than might have been perceived with only the scatterplot, meaning Slope was generally overestimated. Volume clearly has clustered around overestimation, and the median highlights this too.

1c.

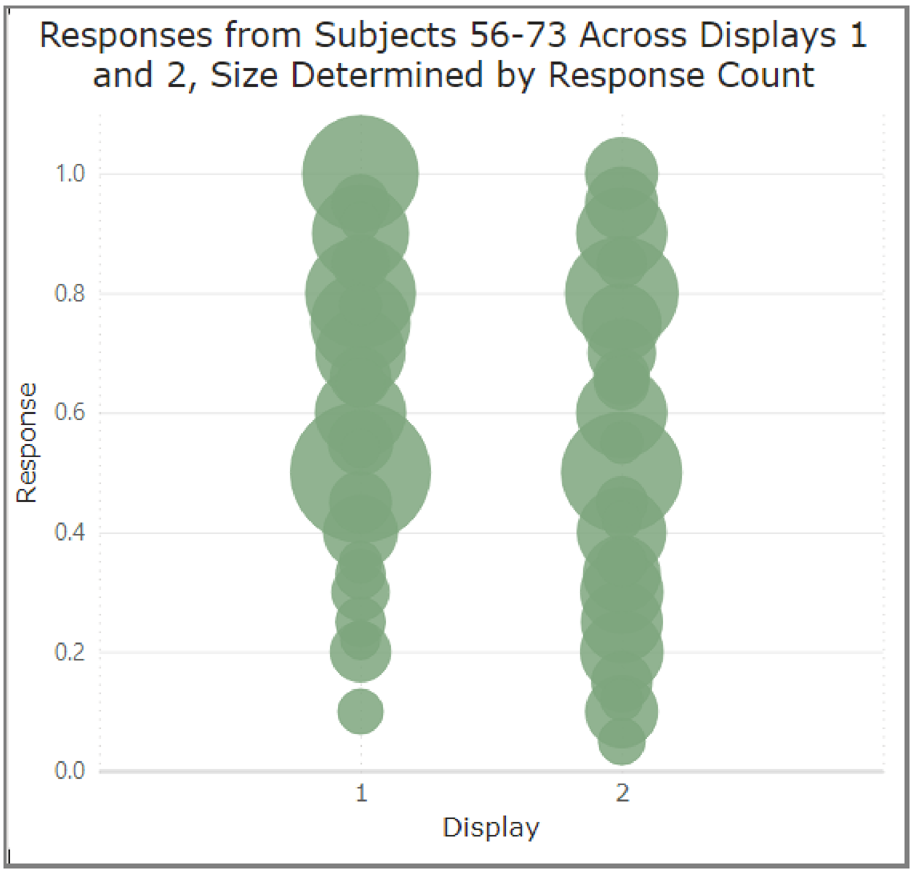
1c. Tableau This graph was created by placing Display in columns, and AVG(Error) in Rows, then Subject in filter, with the filter between 56-73. The most basic way to show a difference in response patterns.



1c. R Studio Filter with dplyr was the automatic step I took to create this graph, though time and memory wise, it is not the best technique. After creating the subset table with Subjects 56-73, ggplot was used to plot a similar scatterplot from 1a, but another geom_point for the mean of each test was added. However, the Tableau plot showed that means vary by about 0.005, so this difference in imperceivable. For this reason, and because the two have similar spreads to each other (though with Display 2 moving slightly towards overestimation), I would say there was no improvement in judging True Value between the first and second Display.

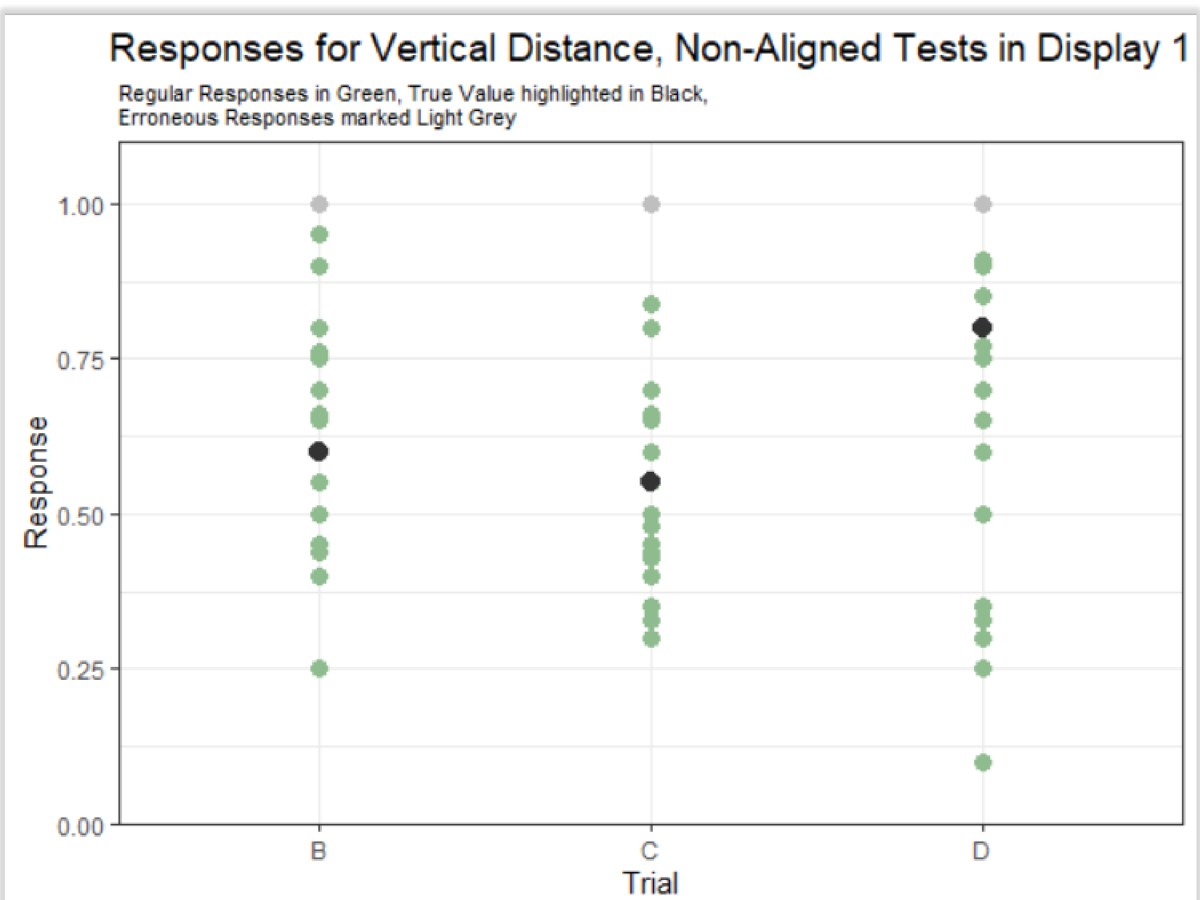


1c. Power BI This graph is a switch up from the previous two. In Power BI, I choose a scatterplot with Display in the x-axis and Response in y-axis. The filter was made for Subject greater than or equal to 56 and less than or equal to 73. The Size of the point was then determined by Count of Response. I chose this in order to highlight where the responses were clustered between both responses. Now the graph shows us what neither the above graphs shows us: differences in clusters between each group. In both displays, subjects many responses around 0.5, and again around 0.8, suggesting that those were where TrueValue locations tended to lie. There is a notable decrease in responses around 1.00 between the first and second display, showing subjects either largely changed their responses away from 1.00 between the two Displays...



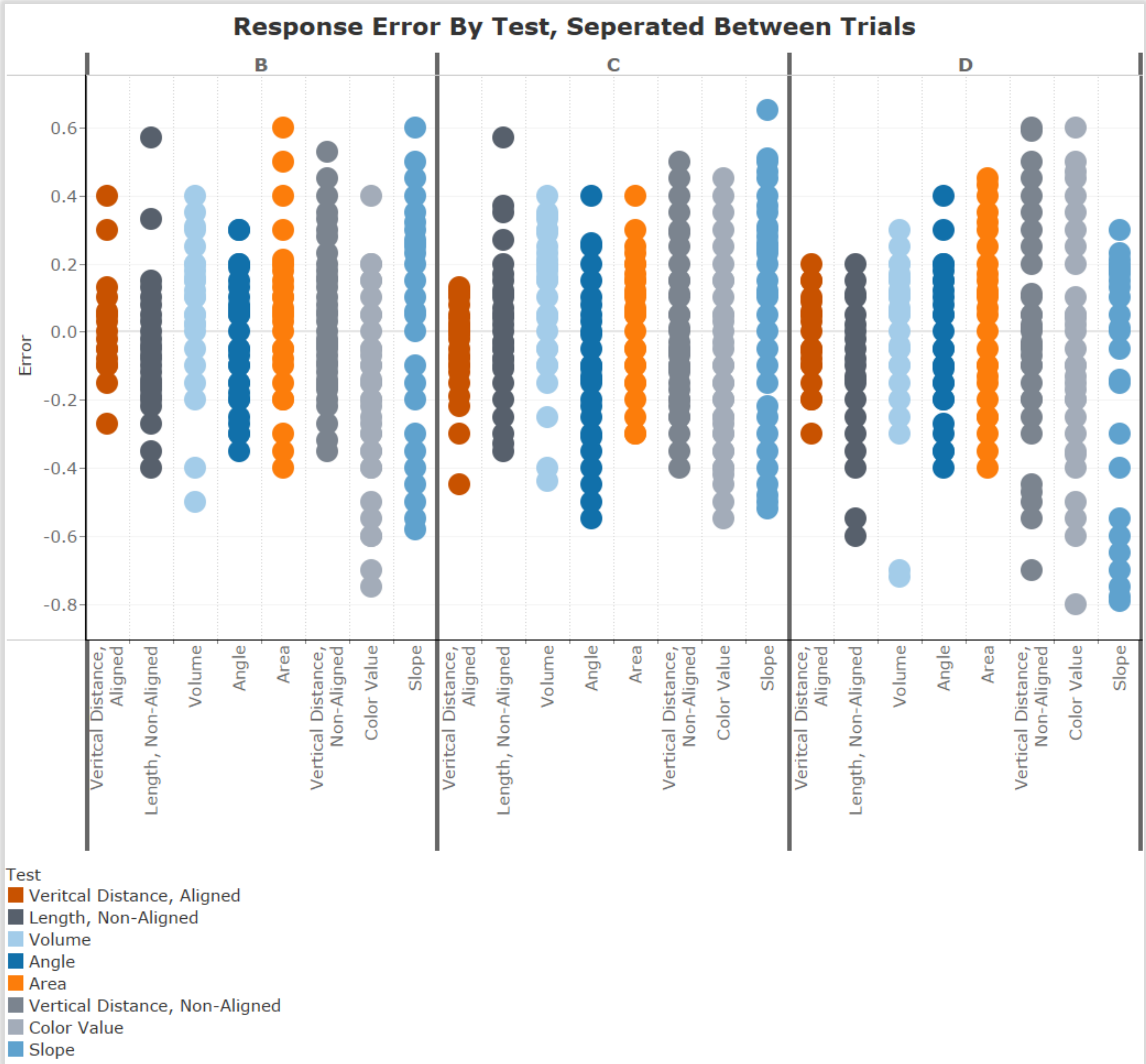
1d.

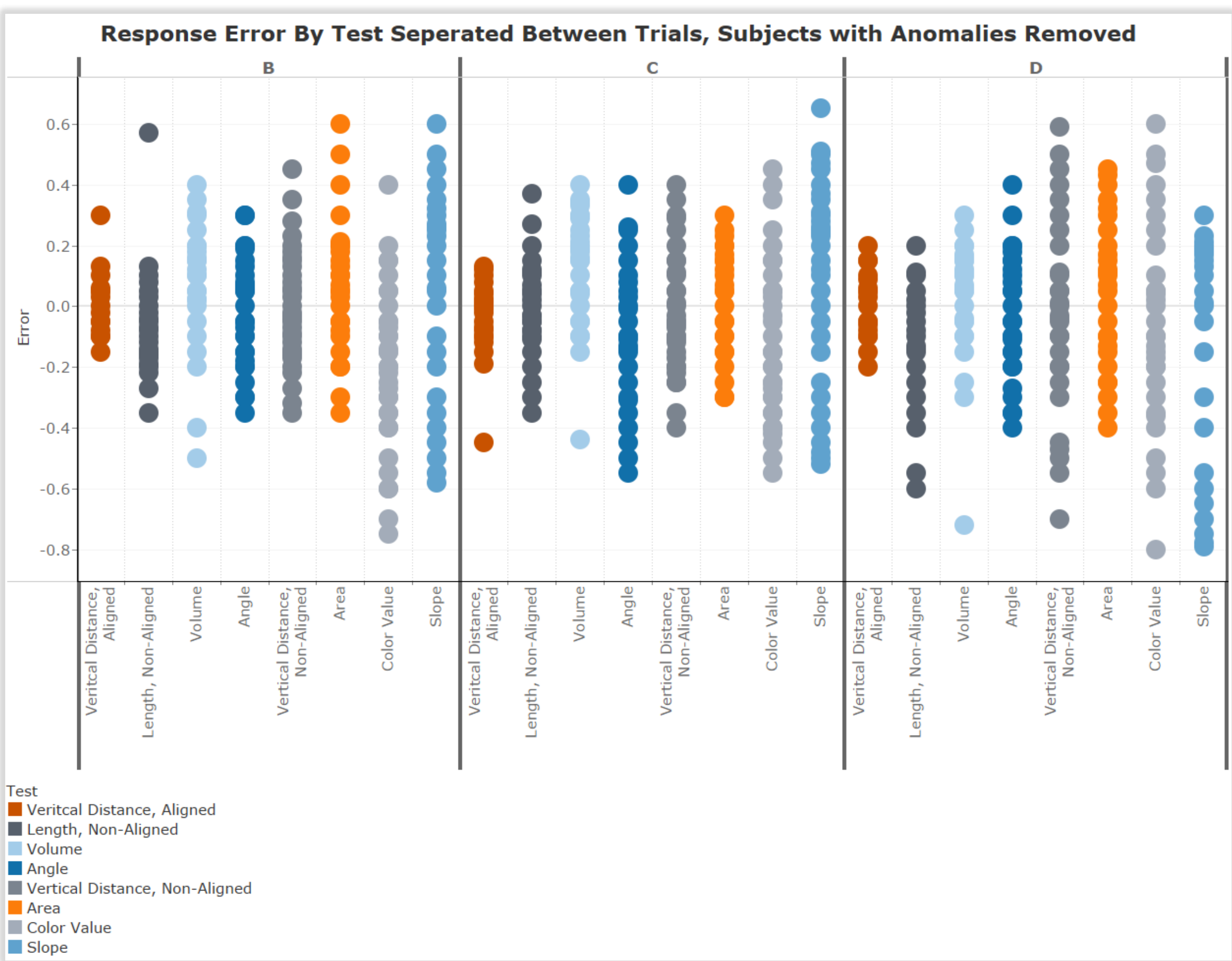
1d. R Studio To create the graph, I created a filter of responses for Vertical Distance, non-aligned and Display 1. The Trial v Response was point graphed similar to the previous graph. Then, I added a geom_point over this with the True Value highlighted, as referred in the 1c graph. Last, the responses at 1.00 were added in another geom_ppint overlay, this time in a light grey to illustrate that these points are erroneous. A subtitle was added explaining these points.



1e.

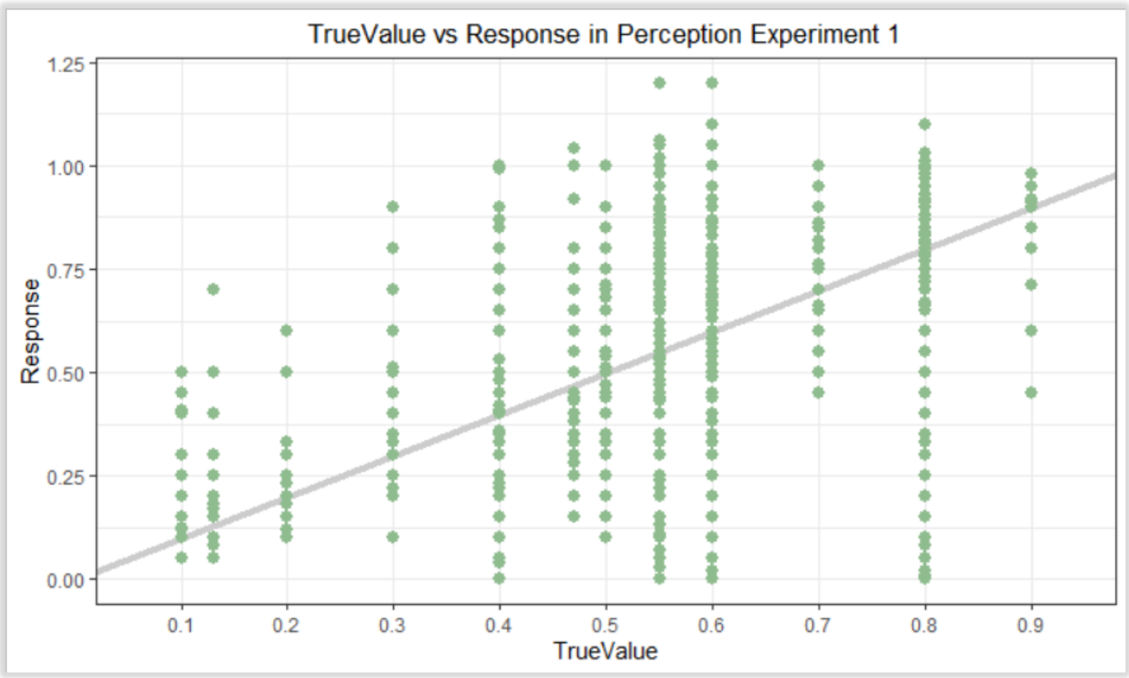
1e. Tableau One variable we didn't look at previously was Test. In this graph, created by Tableau, I placed Trial and Test in the Columns, and Error in Rows. For clarity, the colorblind color palette was applied to Test as well. Then I sorted the tests by variance, so that the tests with least variants are on the left and most variance on the right. This made it easier to view the graph than any other configuration, in my opinion. From here, it is immediately clear that Aligned Distance has the least error overall, with slope and color showing up all over the board. The variance in Vertical Distance, Non-Aligned is odd but this is caused by the error values. To fix this I created a set from subject that excluded the subjects that had the issue as determined by the earlier problem. However, that means that their responses are filtered from all test types, so this graph is arguably just as problematic as the first





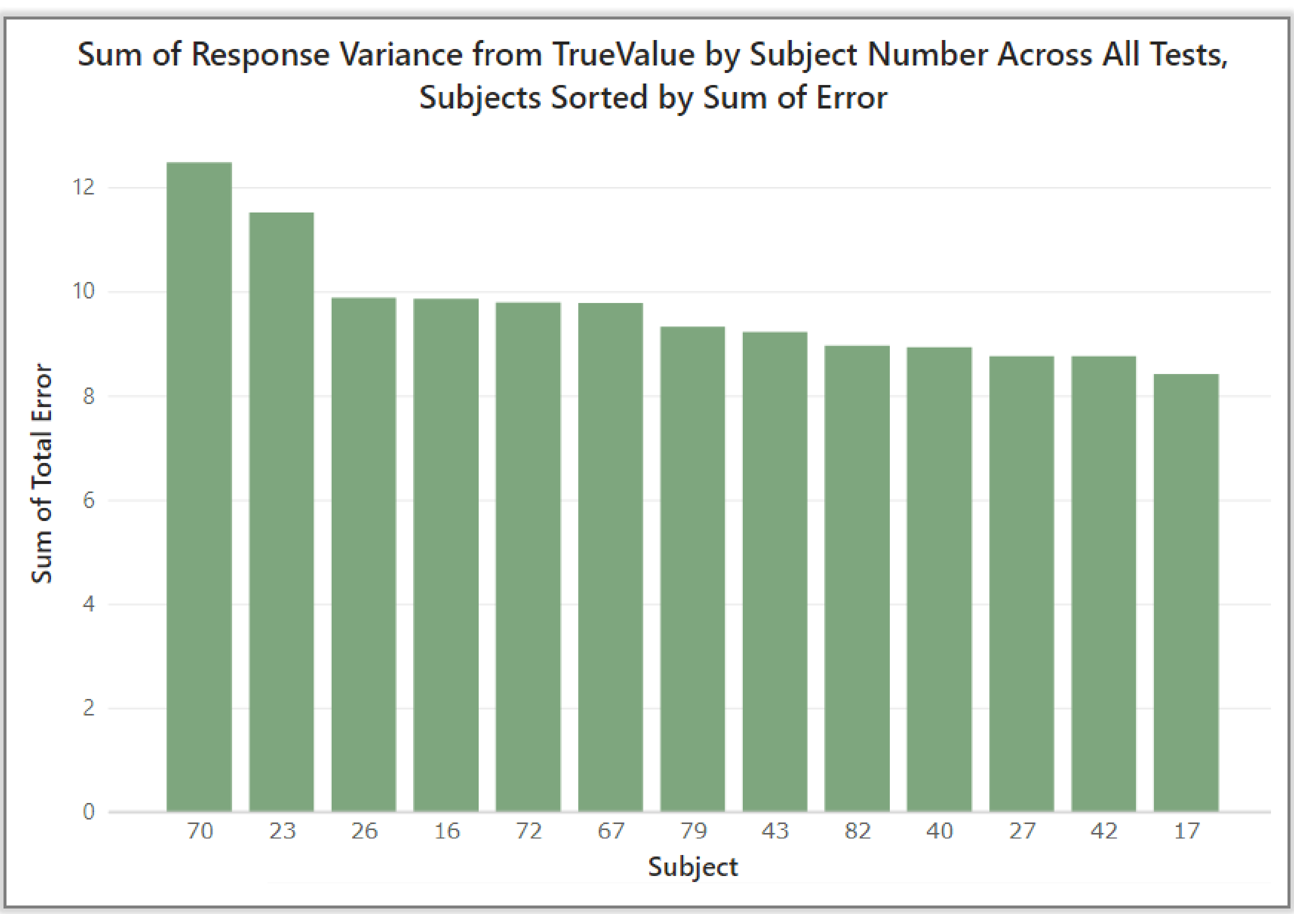
Regardless both graphs are interesting, as they both display a significant shift towards underestimation in Slope between Trial D and Trials B & C. This also highlights when the outliers occurred, such as with Length in Trial B, or Volume in all 3 trials. Where previous graphs would display these underestimates together, this visualization suggests that there may have just been 2 subjects who consistently underestimated Length in each Trial.

1e. R Studio For my R Studio testing, I utilized R's intuitive statistics abilities before deciding on a graph. Experimental tool Summary is always the best place to start, and right away I noticed variables that were not compared to before: TrueValue and Response. There was never a TrueValue of 0, but some subjects had chosen it. The response also spread farther in overestimation than underestimation which made me wonder if subjects were more likely to have responses near the TrueValue when the TV was closer to 0.



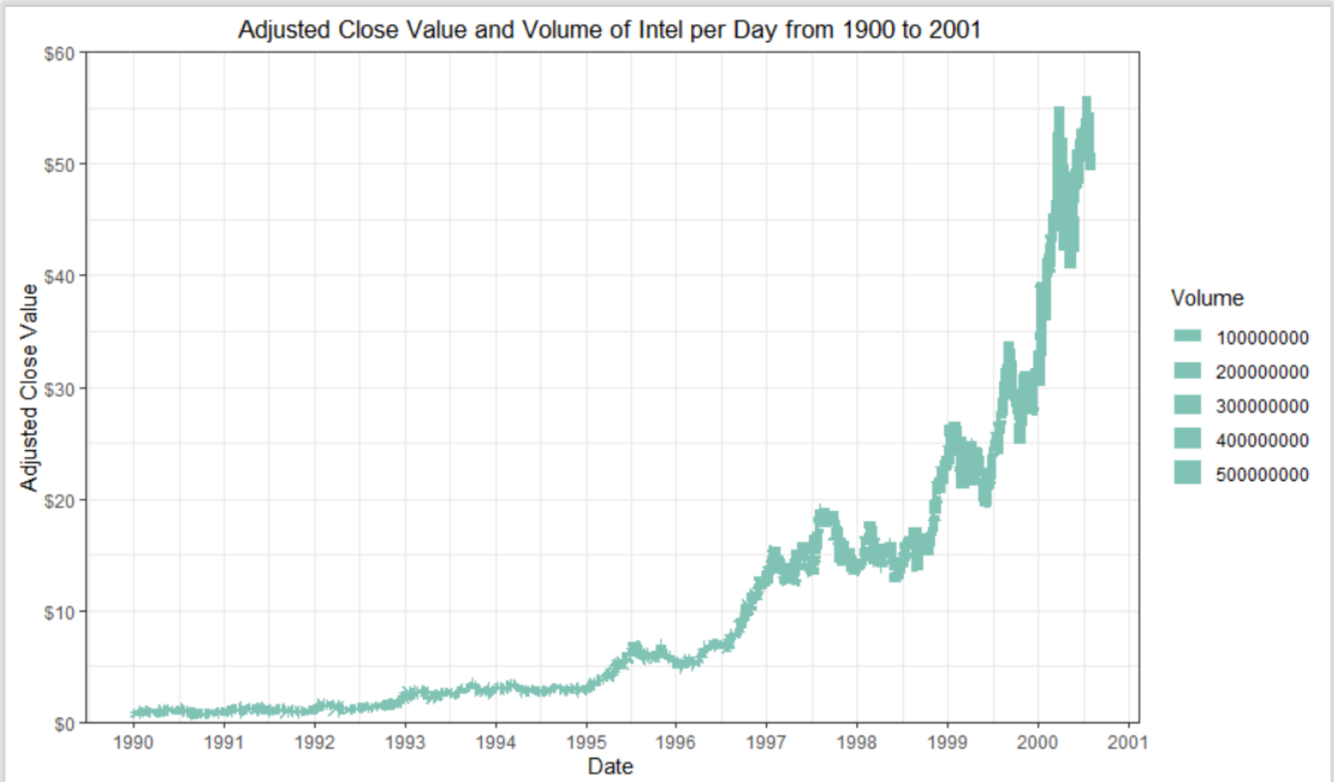
While this graph is interesting, it is a bit hard to understand…The line with slope 1 was added to clarify where the response data *should* cluster, but the unevenness of data for some TrueValues, versus the multitude of data for others makes it difficult to gauge how much of an impact the TrueValue had on Response, if at all.

1e. Power BI In honor of Power BI's general experience, I figured the best graph would be one that tallied error from subjects in order to expose which students racked up the biggest mistakes. This gave me opportunity to go deeper into format settings in Power BI, and eventually I settled on Subject on the x-axis, and SUM(ABS_Error) in the y-axis. The x-axis was changed to "categorical", and by increasing the "Minimum category width", I was able to show just the top dozen or so subjects with the highest total variance from TrueValue.
It should be noted that poor subject 70 did suffer from the error value in the Non-Aligned tests as determined earlier, which artificially increased their total error. But laughing at the expense of subject 23 is still valid.

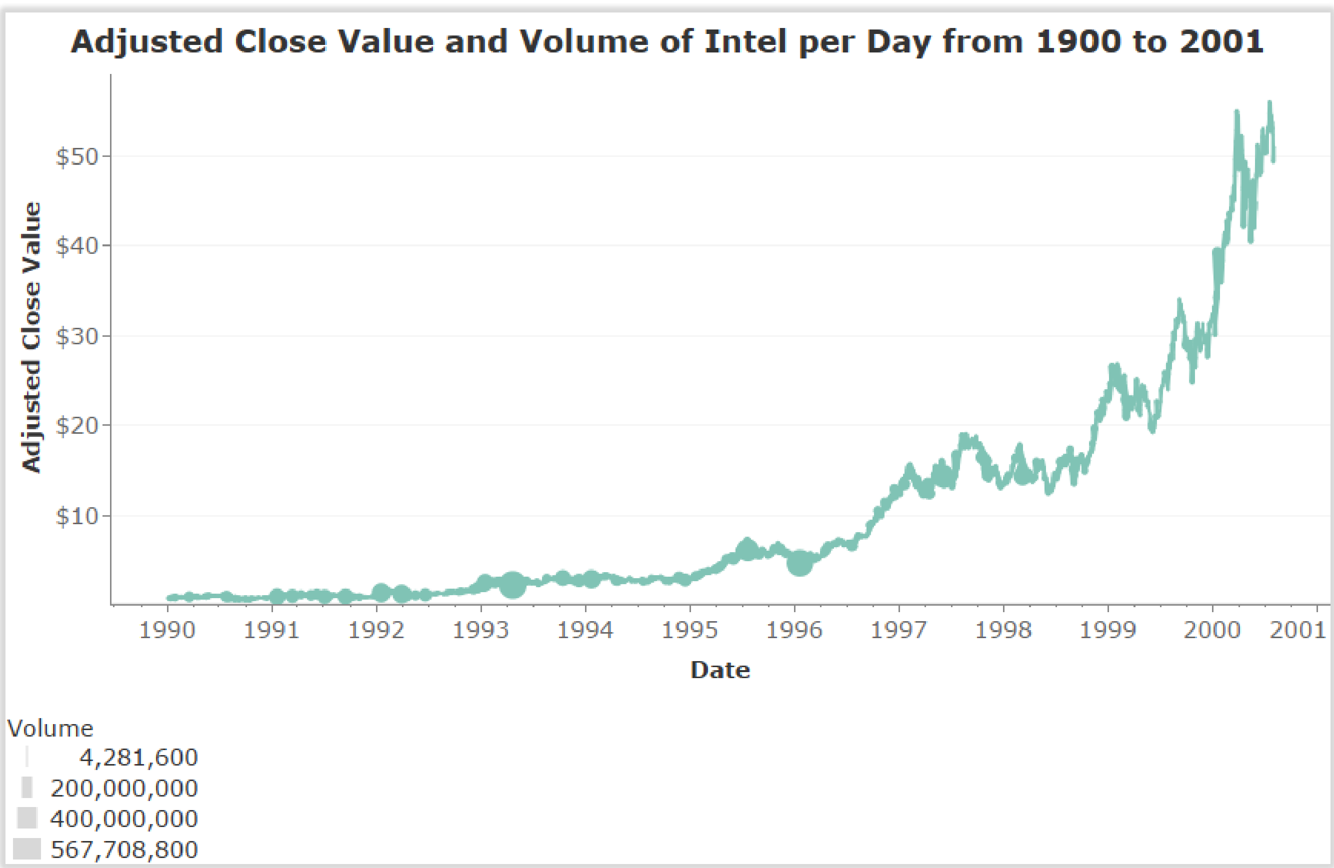


2a.

2a. R Studio This graph was created by first editing the Date using lubridate, then using ggplot.

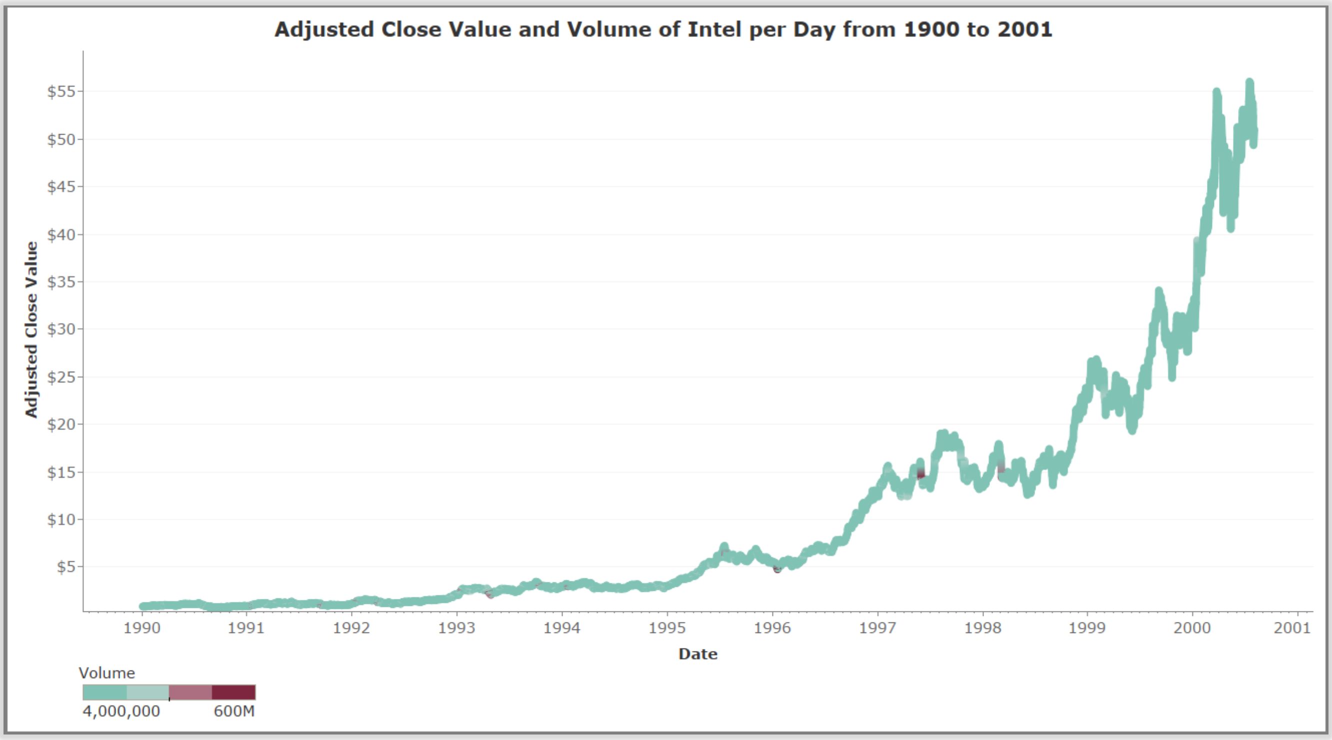


2a. Tableau For tableau, Date was placed in Columns, ADJ Close was placed in rows as a continuous dimension, and Volume was placed as a size mark as an attribute.

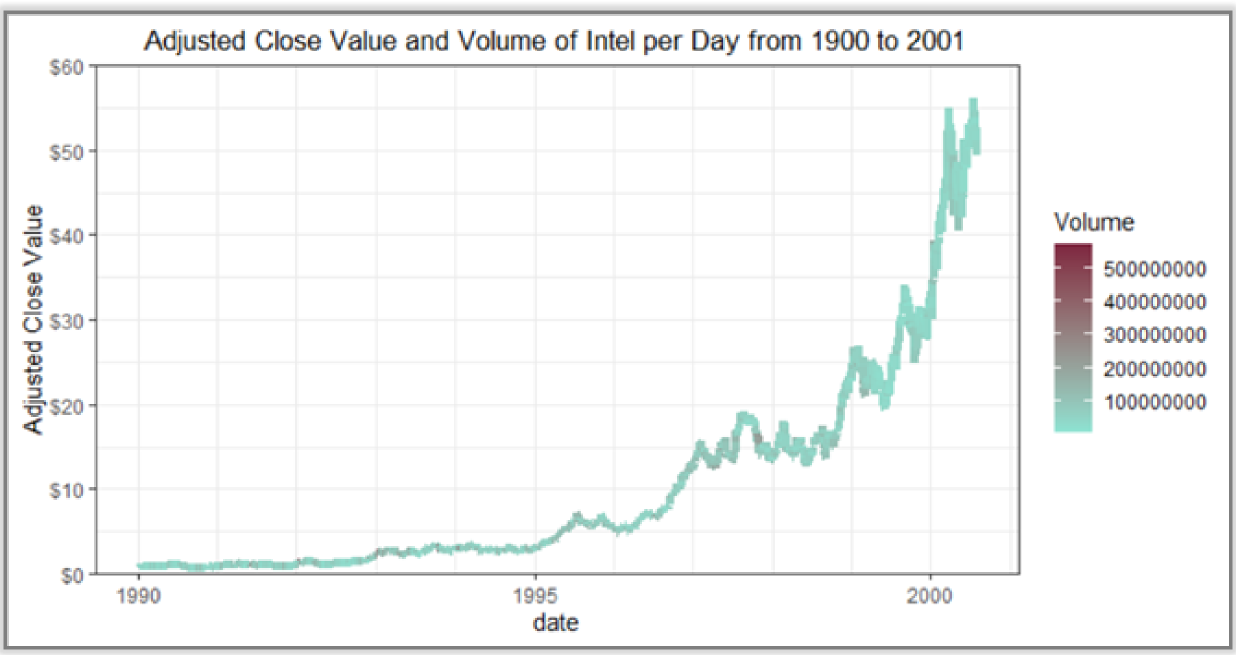


2b.

2b. Tableau The previous worksheet was duplicated, then ATTR(Volume) was adjusted to color. I chose a custom diverging color in an attempt to highlight the outliers with high volume by randomizing from Colorgorical, where Perpectual Distance with maximum score importance.



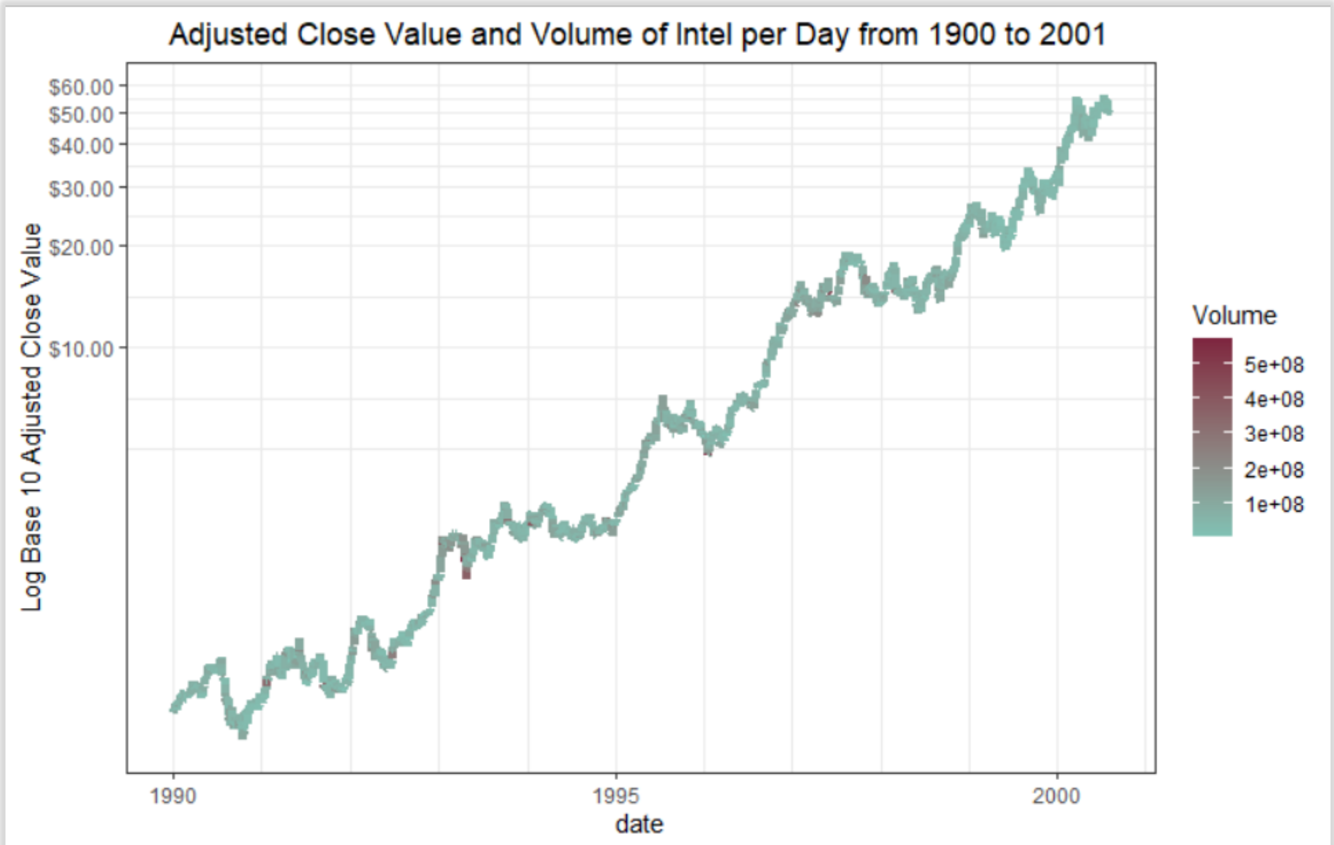
2b. RStudio this graph was made as a copy from the previous, with the same low and high color_gradient hex colors from the Tableau graph. An attempt to smooth the line using package bdscale was attempted, but it didn't help.



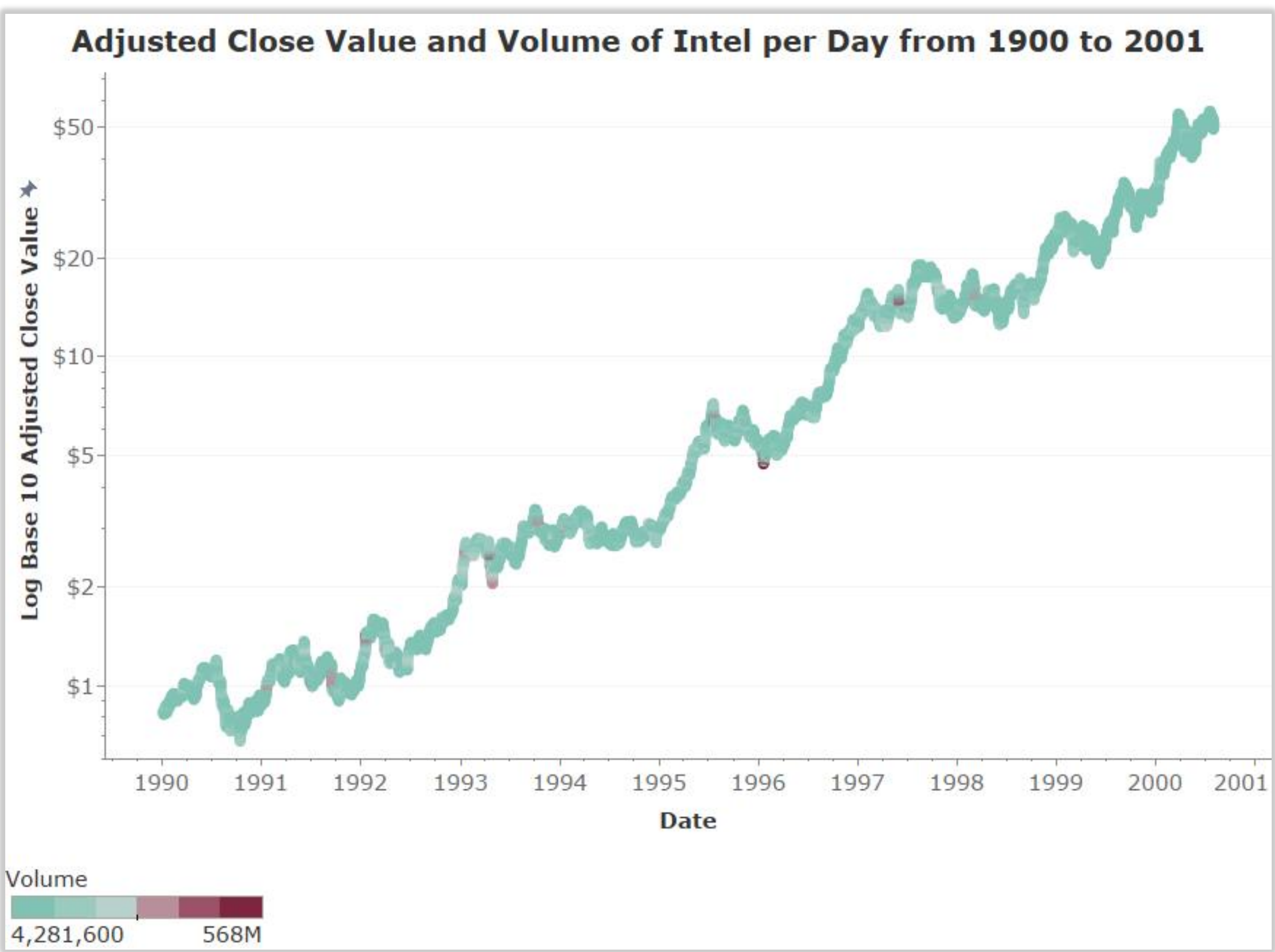
As far as the graphs from b versus c, I would say that the graph made by Tableau with size=Volume (a) does the best job, since it used a huge circle to demonstrate the large volume value. R Studio I just couldn't get the graph to smooth out, and as a result it was harder to discern where the Volume variations were. This problem was exacerbated by the slope of the graph, as values from 1990 through 1995 are so grouped together that it's hard to differentiate between them.

2c.

2c. R Studio I found the auto graph created using the scale_y_log10 to be unclear in the log scale, and recycled code utilizing the :scales: package I used in previous project. After setting the scale to transform to "log", the graph has evenly spaced ticks with disastrous numbers. Adding extended breaks cleans this up and separates the ticks to make it clear the graph is increasing exponentially



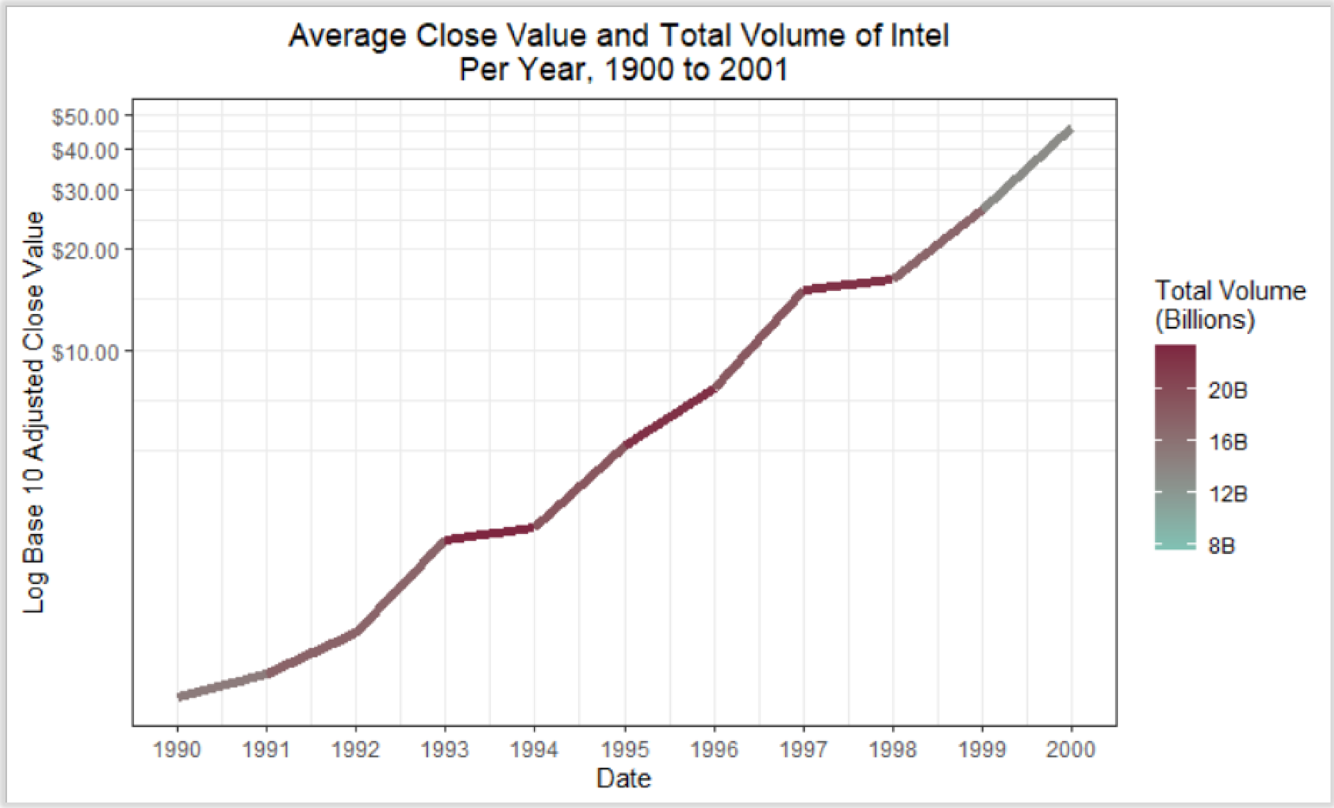
2c. Tableau The graph from 2b was duplicated, then log was applied. Tableau doesn't allow for as much customization as R, so all I changed was the scale to log, and the y-axis title.



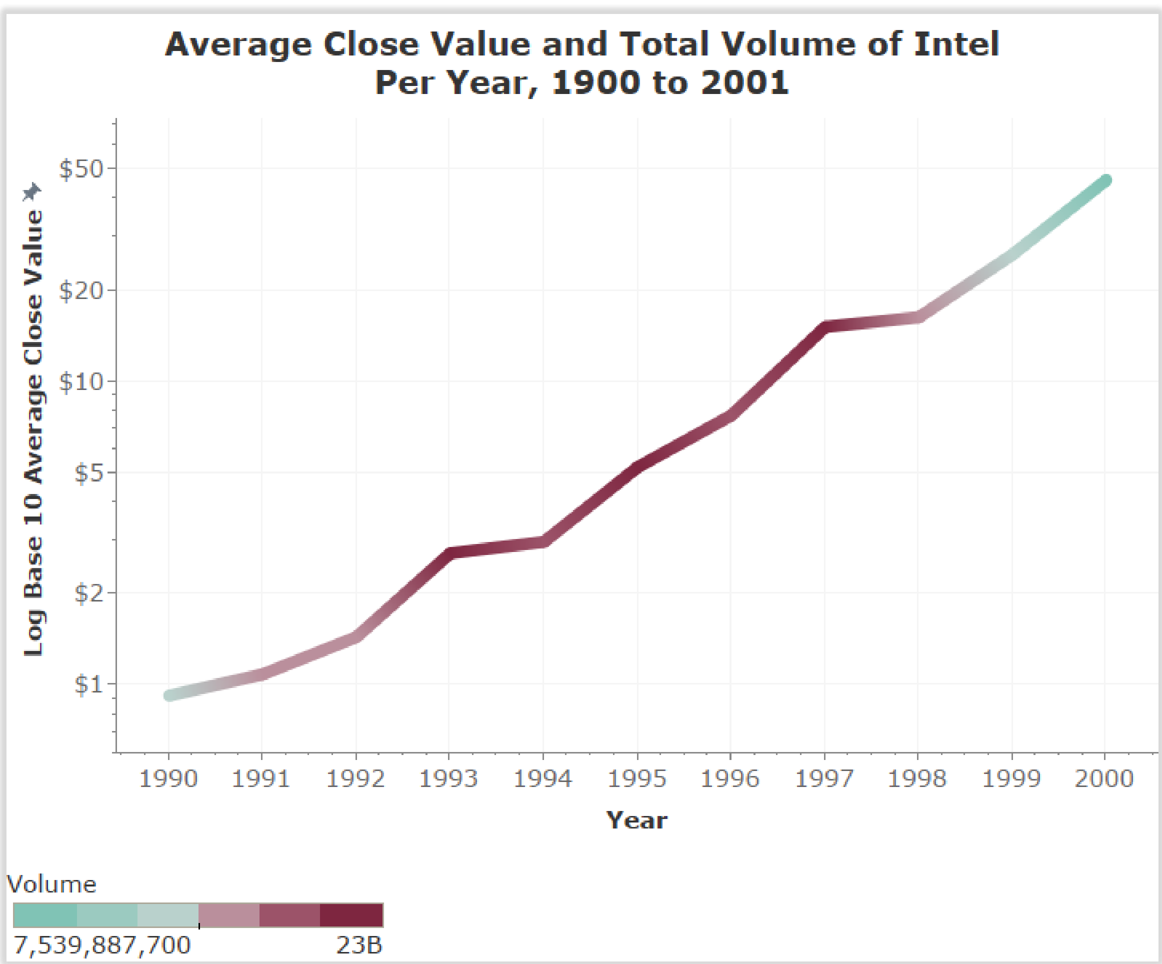
This changes the graph by making a positive linear average slope. This means that the price increased exponentially in the graphed time span.

2d.

2d. R Studio Finally fixed the legend! In order to make this graph work, I first created a column that had the floor_date from lubraidate, which just rounds to the first day of the year. Then I grouped by this new column, and since all data in the same year had the same date, this grouped easily and was then summarized by MEAN for Adj Close and SUM for Volume. From my r code it is clear I couldn't figure out the correct popping and instead made a new dataframe to hold this info.

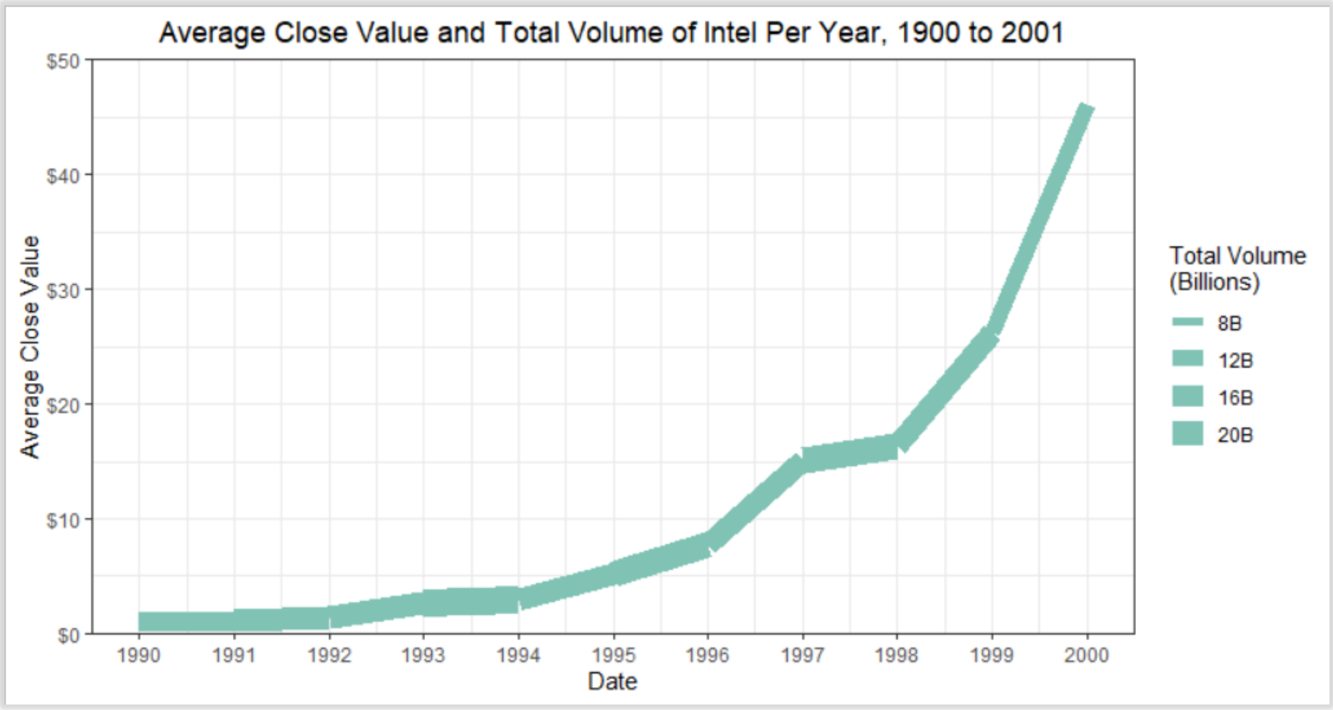


2d. Tableau Volume was changed to measure (SUM) and Adj Close to (AVG), and the Date to Year. Besides ratio changes, no other changes were made.

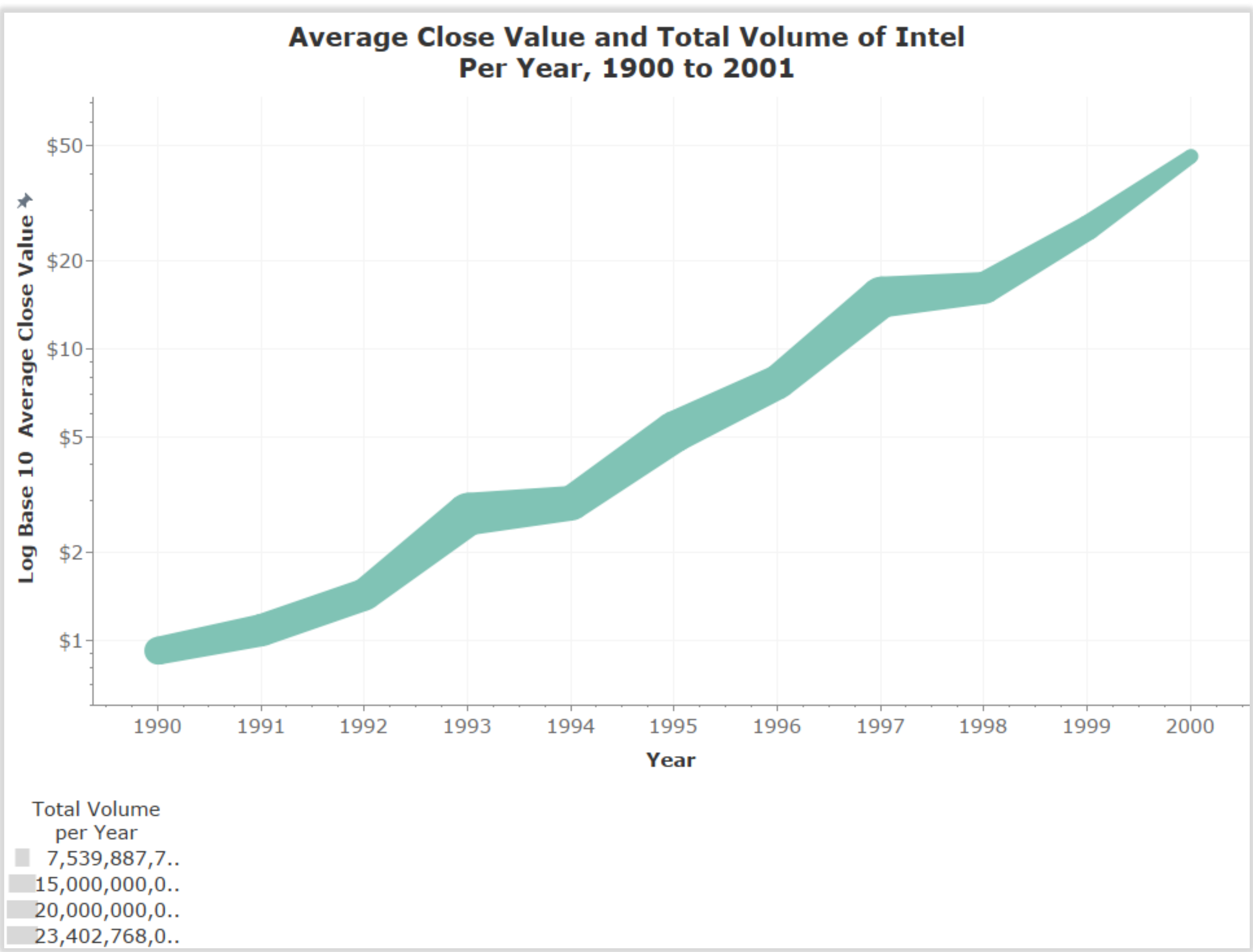


1992, 1994, and 1998-2000 have appear to have the biggest slope, and the greatest percentage increase from the previous years.

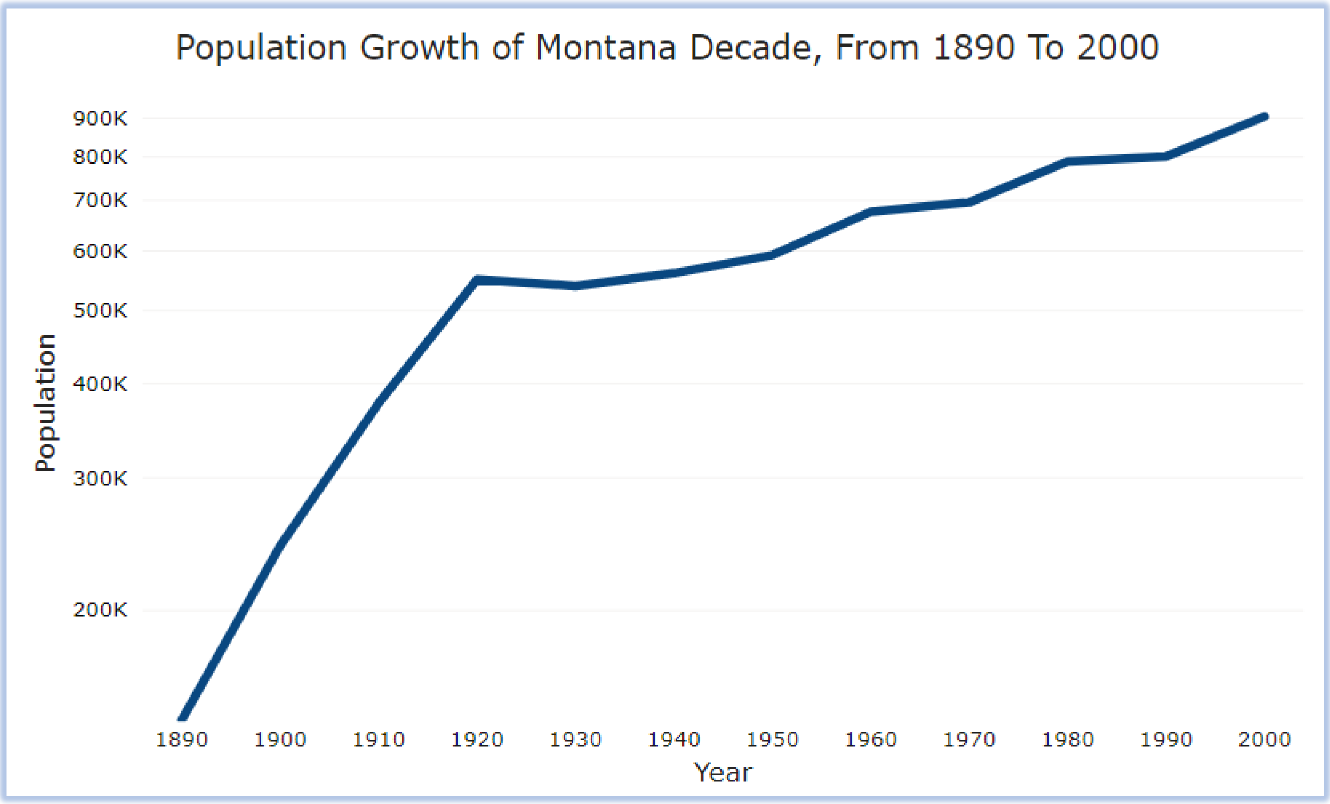
2e. R Studio It isn't clear whether or not to stay in Log10 base and arguably the graph looked better without it. Here in R, it is very clear that there is less volume per year in 1990 and 1999 than 93 and 95. However, knowing what I know about outliers, this graph may have distortion.



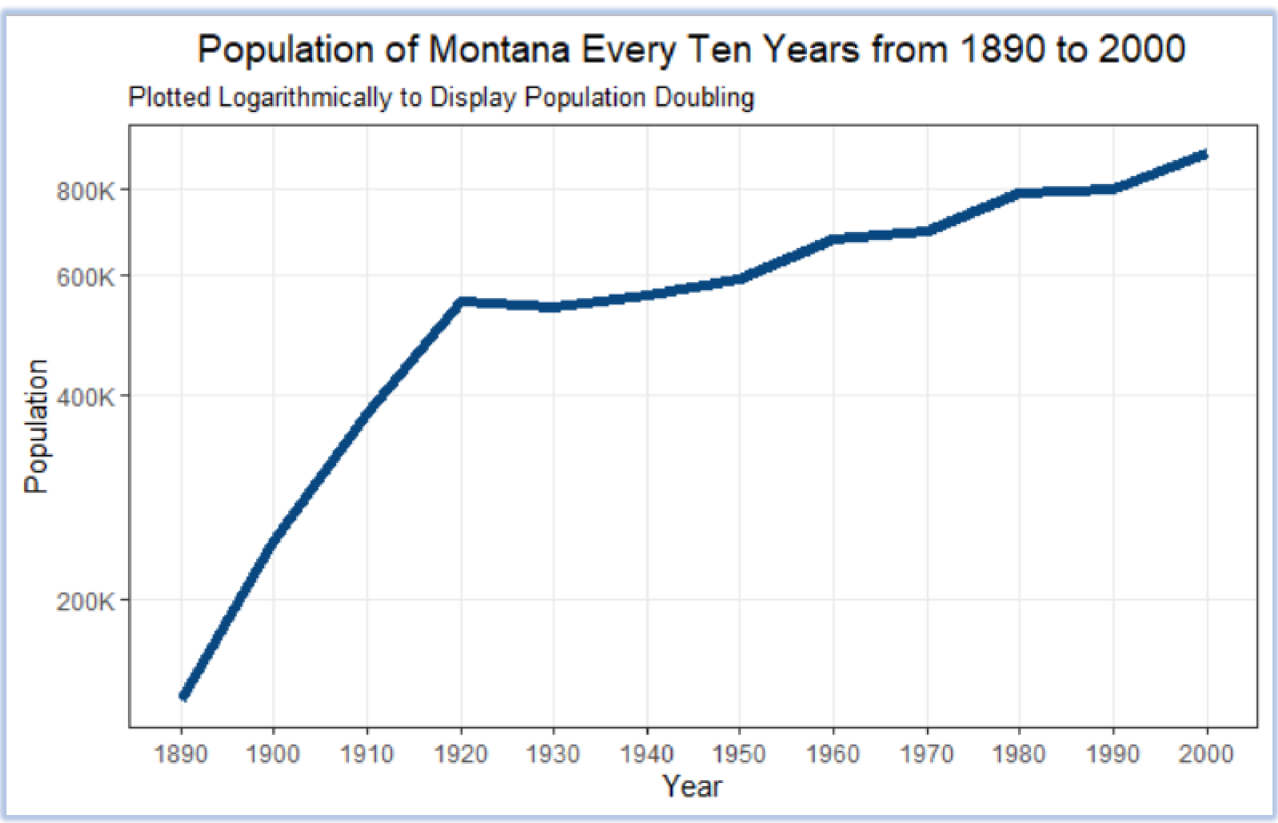
2e. Tableau For Tableau I decided to keep the log base10 scale just for the diversity in visualizations. In this viz, it is much easier to see the difference in Volume when there are just 11 data points.



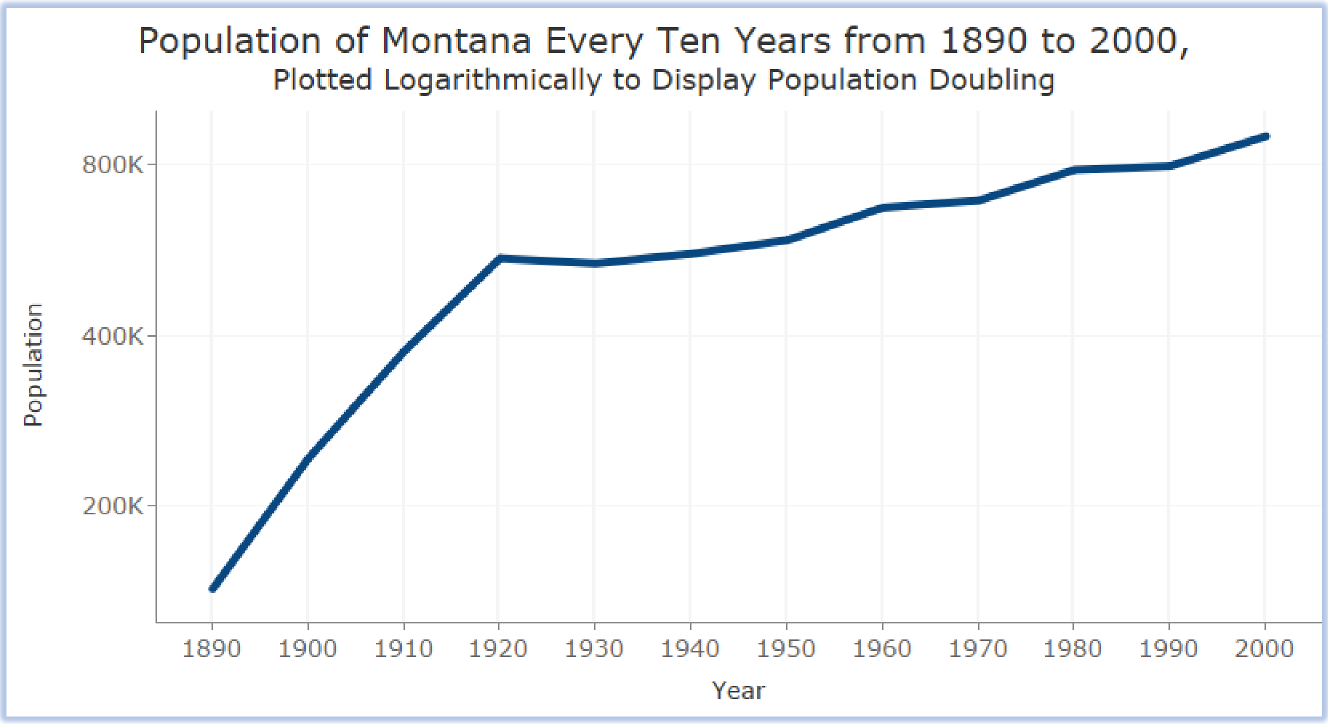
3a. Power BI This problem looked like a great opportunity to bring back my old…friend, Power BI. Power BI supports Logarithmic scales sometimes, and worked with this dataset. Since the audience is technical, I left the visualization with the stepped y-axis. Ultimately Power BI's graph looked the best in my opinion.



3a. R Studio extended_breaks didn't really work with this graph, even though it was great in the previous ones, but I discovered pretty_breaks, which automatically made this perfectly spaced axis. To clarify the dates, the axis was set to place a gridline and year for each data point.



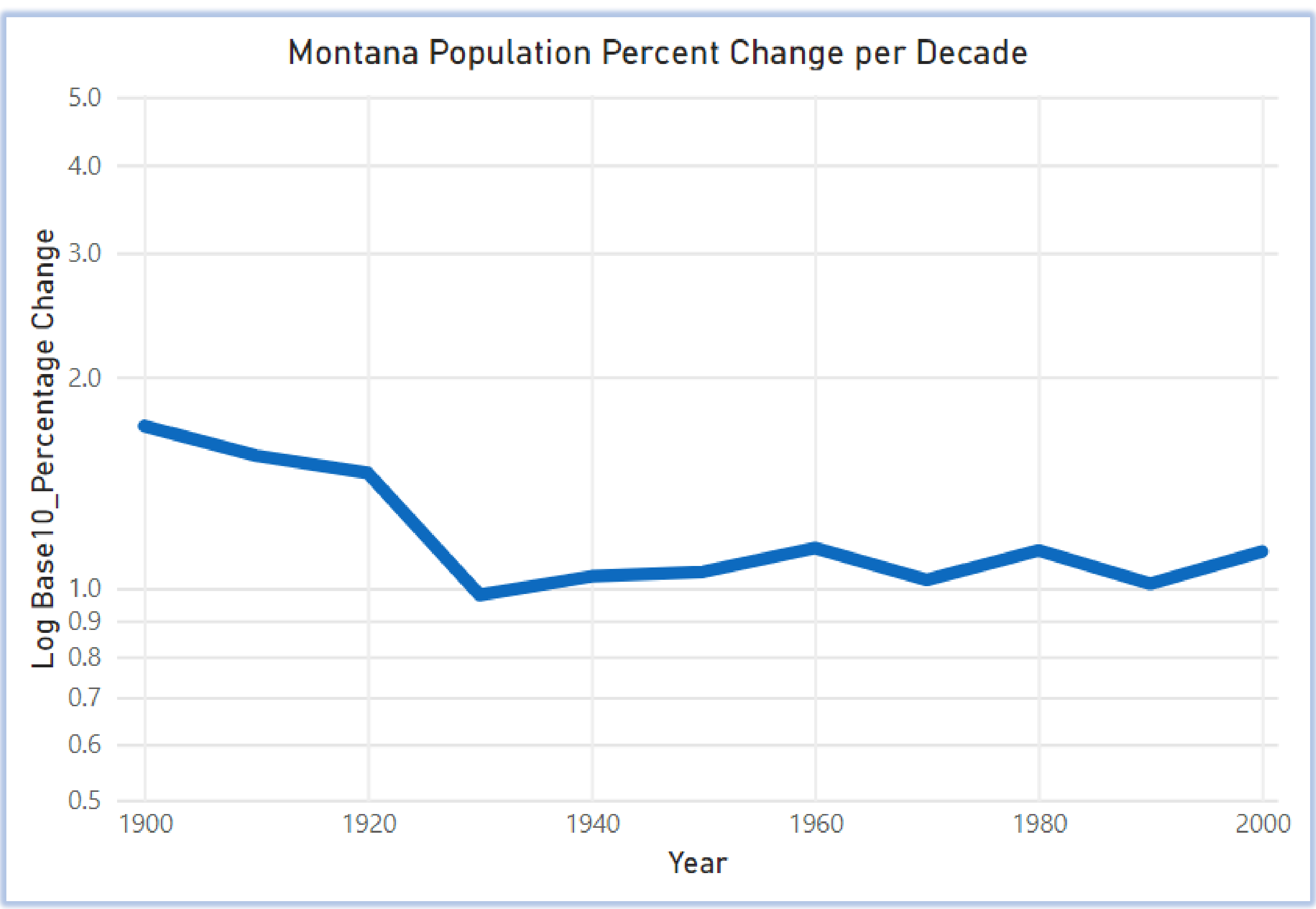
3a. Tableau Tableau's automatic settings for log based scales clashed with this dataset. The powers, tick marks, and range had to be adjusted.



The population has doubled 3 times and in 2000 has a size about 8 times what it was in 1890.

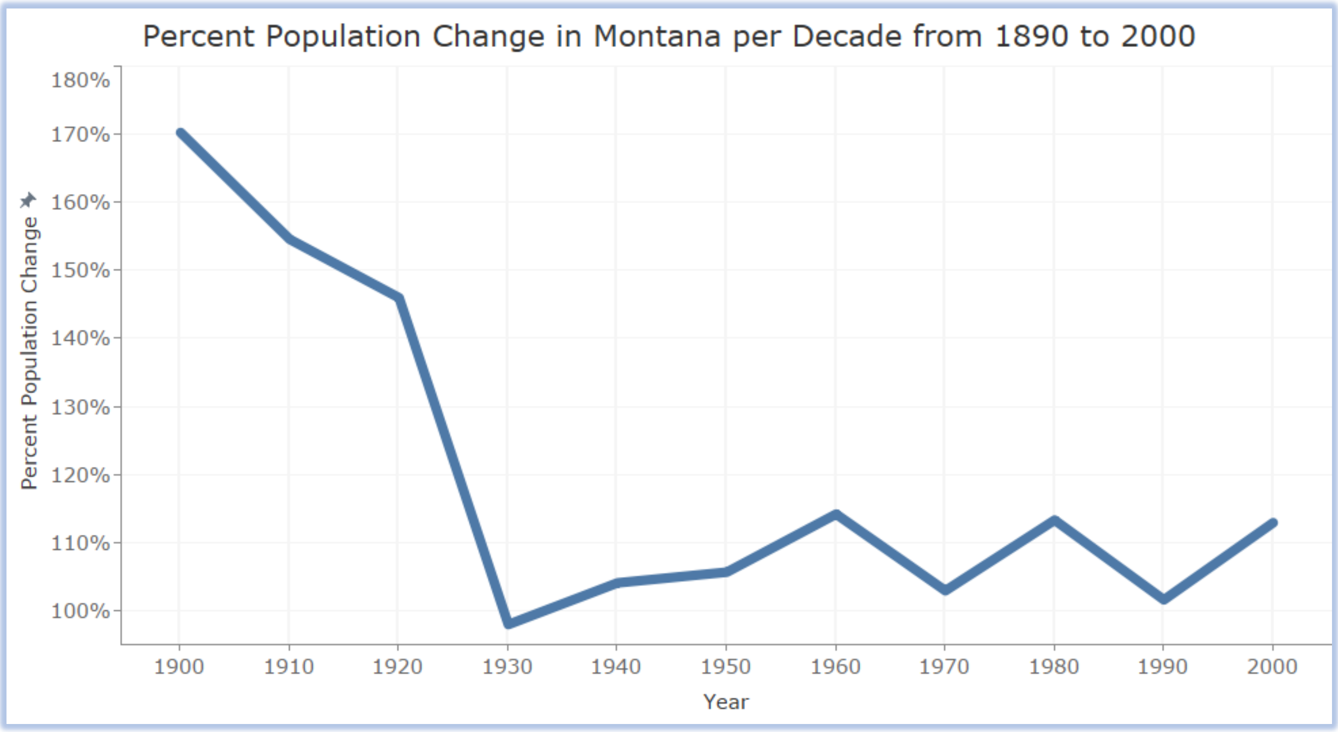
3b.

3b. Power BI We talked in class about Power BI but wow, this was really confusing to make work. I created an Index Column, then another column that input the population in the row above (using an if/else to put 0 in the first row), then Another column that was [Population]-[PrevPopulation]/[PrevPopulation] for change. I wasn't sure how to deal with infinity, because I couldn't figure out how to refer to infinity in power query, so I then made ANOTHER column, that conditionally said to put null if the value was over 99999, and column value otherwise.
Anyway, remember when I said that power BI doesn't often allow log scale? This was one of those instances. From my understanding, it's because the percentage goes below 0. Okay fine, I created yet another column called "PercentChange" that added 1 to the data

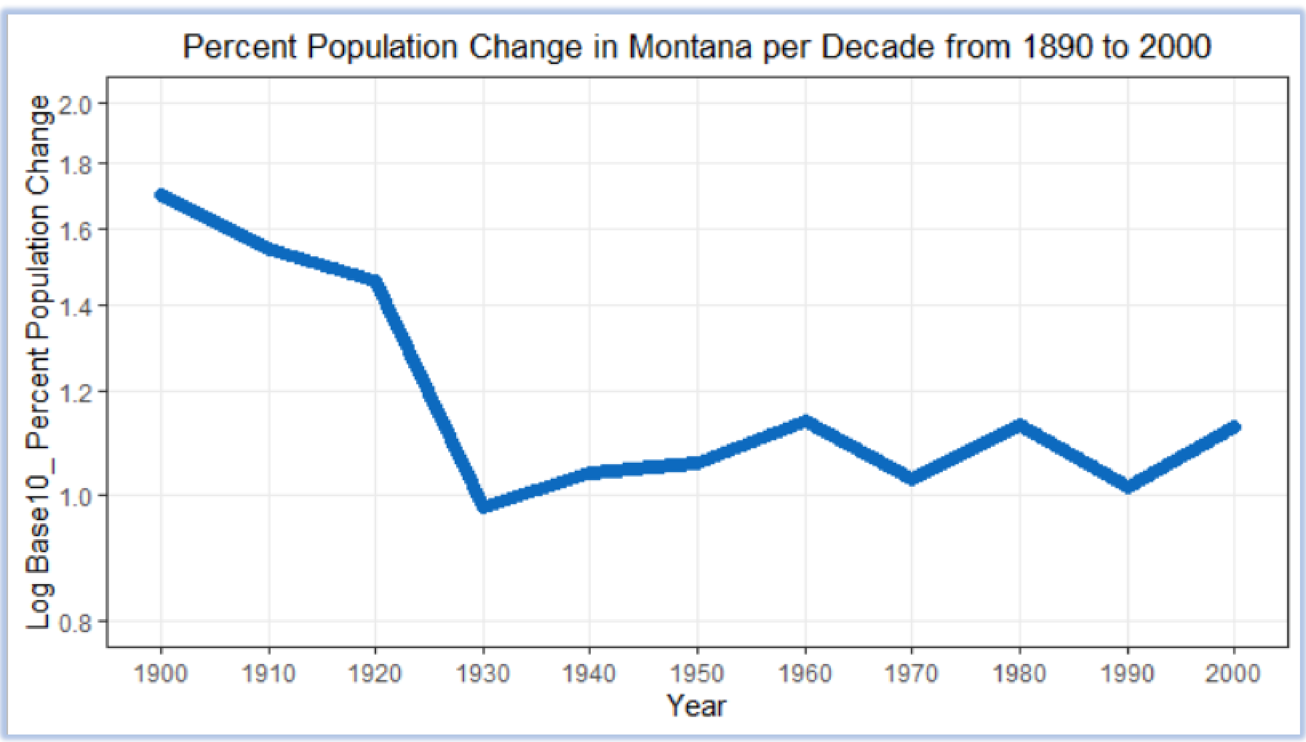


3b. Tableau I chose "Table Calculation" then "Percent Difference From" "Year of Year" Relative to "Previous" and Tableau automatically made the exact same graph as above just like that. In 2 clicks. To mix things up, I chose "percent from" for the visualization below to show the percent of population relative to the previous decade. Same graph, different axis numbers. There was a null value, from 1890, which was filtered.

Though Tableau lets me use log for this scale, it doesn't really help the visualization, and if anything would be more confusing, even for a technical audience, as it made the percent change appear more significant but did not allow for appropriate gridline adjustments.



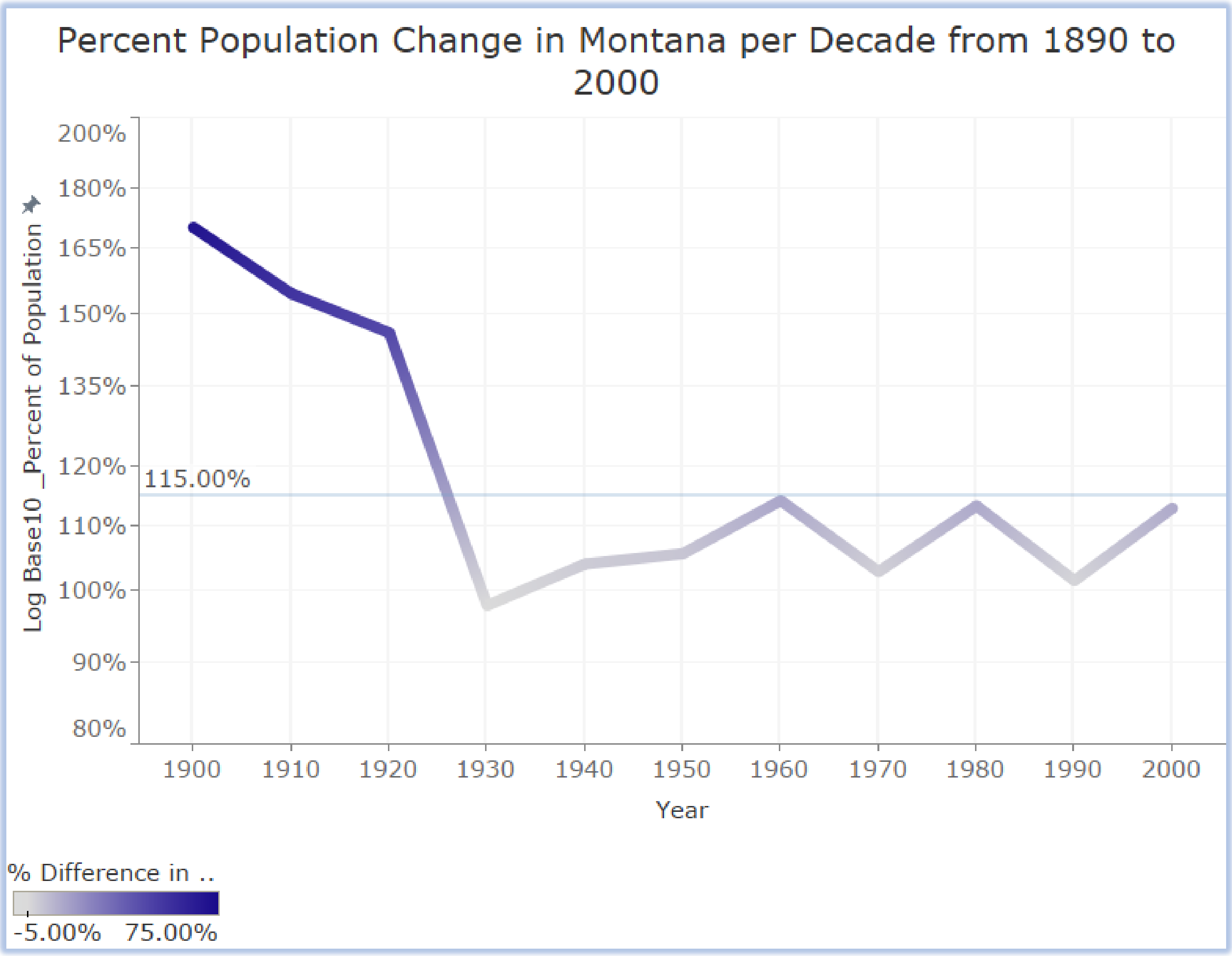
3b. R Studio This calculation was created simply by using the lag() function in R. Basically, lag(MontanaPopulation$Population, 1) was used as V1 in a (V2-V1)/ V1 calculation. +1 was added like in Power BI so log ran correctly. Then just chart tweaks.



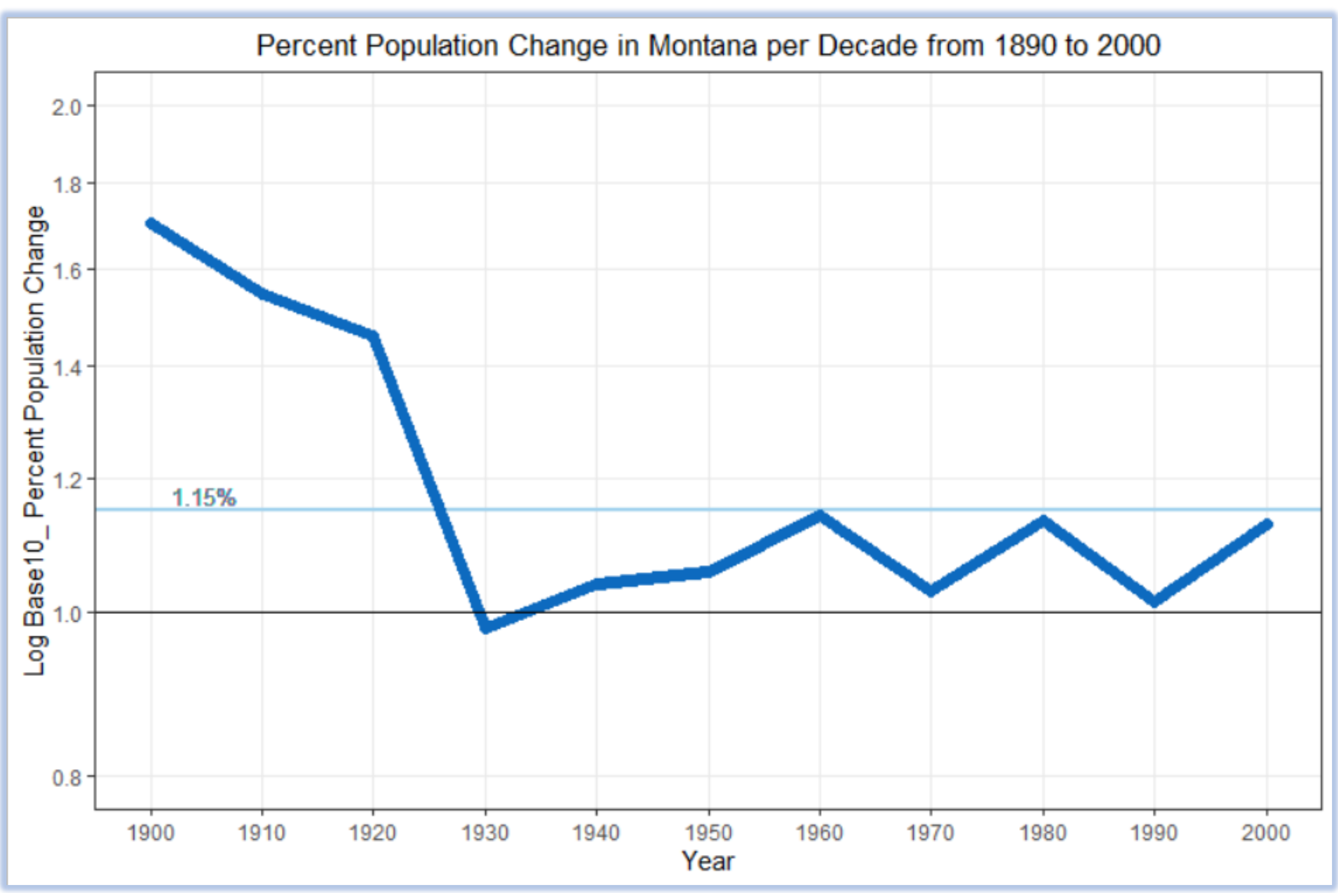
The percentage rate of change had a high positive value, then dropped in 1930. Now there is a consistent rate of change around the 1.1 mark. The greatest population changes occurred from 1890-1920.

3c.

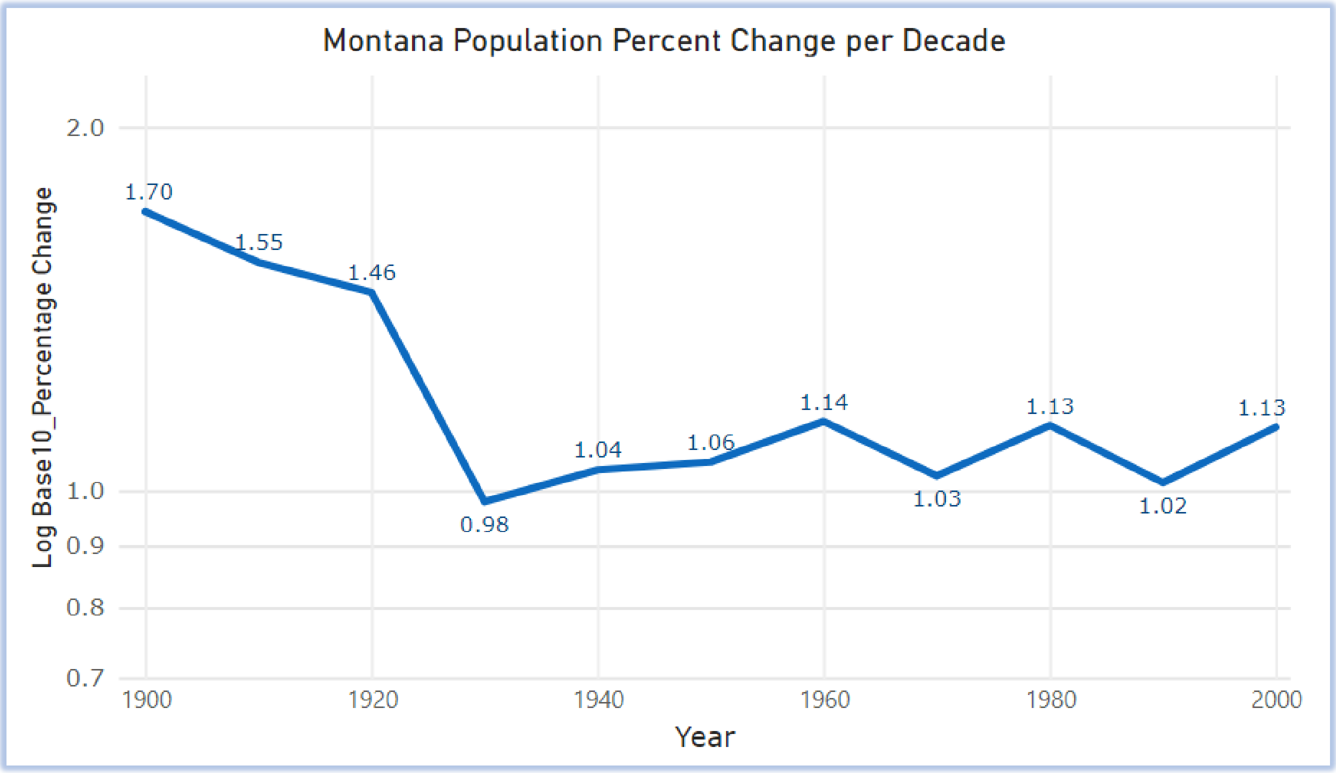
3c. Tableau This is similar to the previous graph, a reference line at 15% increase was added, and color was applied to the percent change



3c. RStudio geom_hline was created to make the y-intercept line at 1.15, then geom_text was used to add the label.

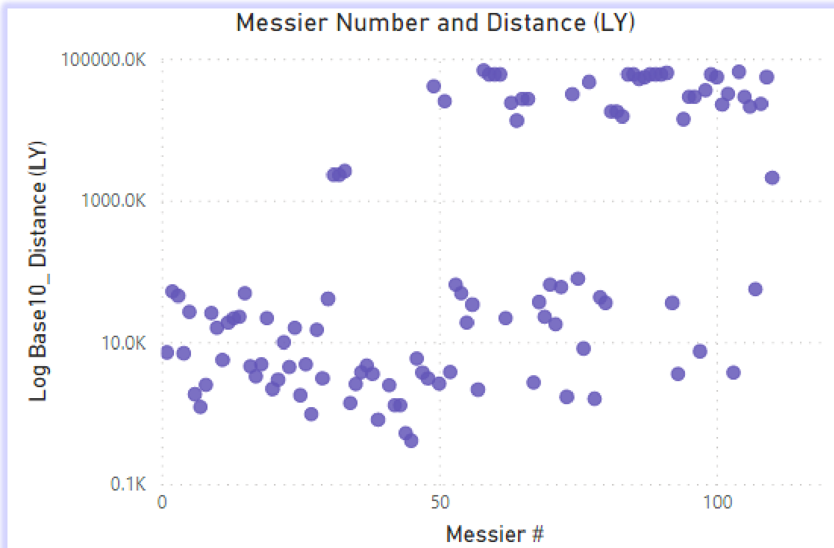


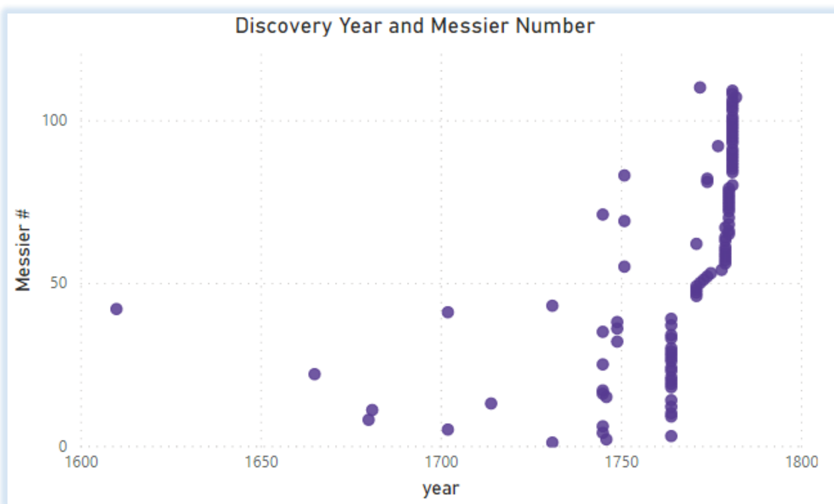
3c. Power BI Decided to switch it up with Power BI by adding the actual value labels to the graph, since Power BI actually places them pretty nicely through the graph.

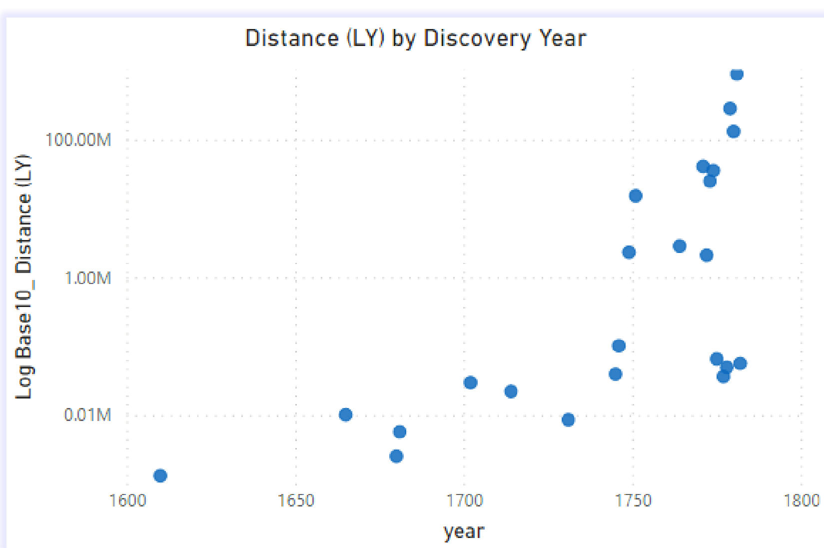


The years with a population increase greater than 15% were 1900, 1910, and 1920.

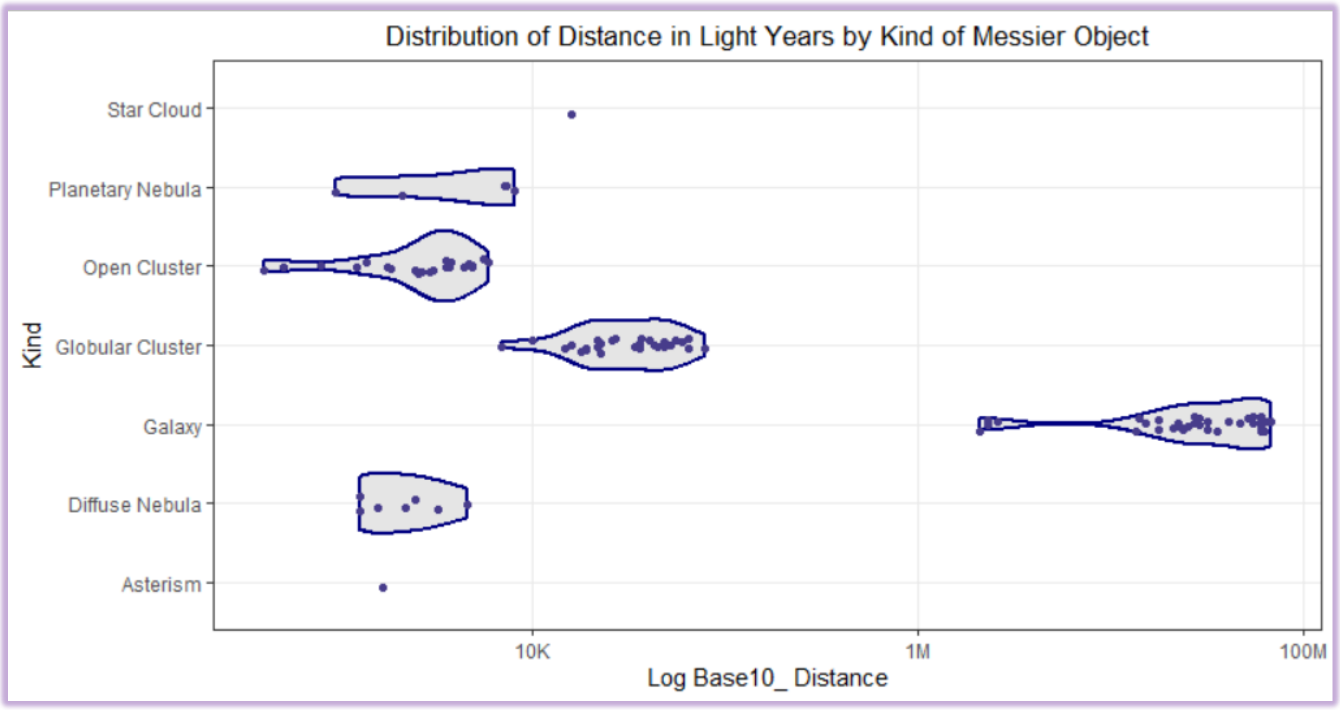
4a. Power BI When graphing Messier# vs Distance, the Distance values increase as the messier numbers do, with the furthest distances occurring in Messier values over 50. Another graph of Year vs Messier number illustrates that though there is a significant number of discoveries around the 1760s-1780s, that all Messier Numbers above 50 occurred after 1745. A third graph of Year vs Distance in Base10 shows that discoveries over 1M lightyears away started at this same time, and was followed by an uptick of objects further and further away. This suggests that Messier's numbers may have been ordered as he found out about them, meaning the most recently found objects have higher numbers, and that the ability/interest in finding these objects increased over time, which made it possible for further objects to be found as time went by.



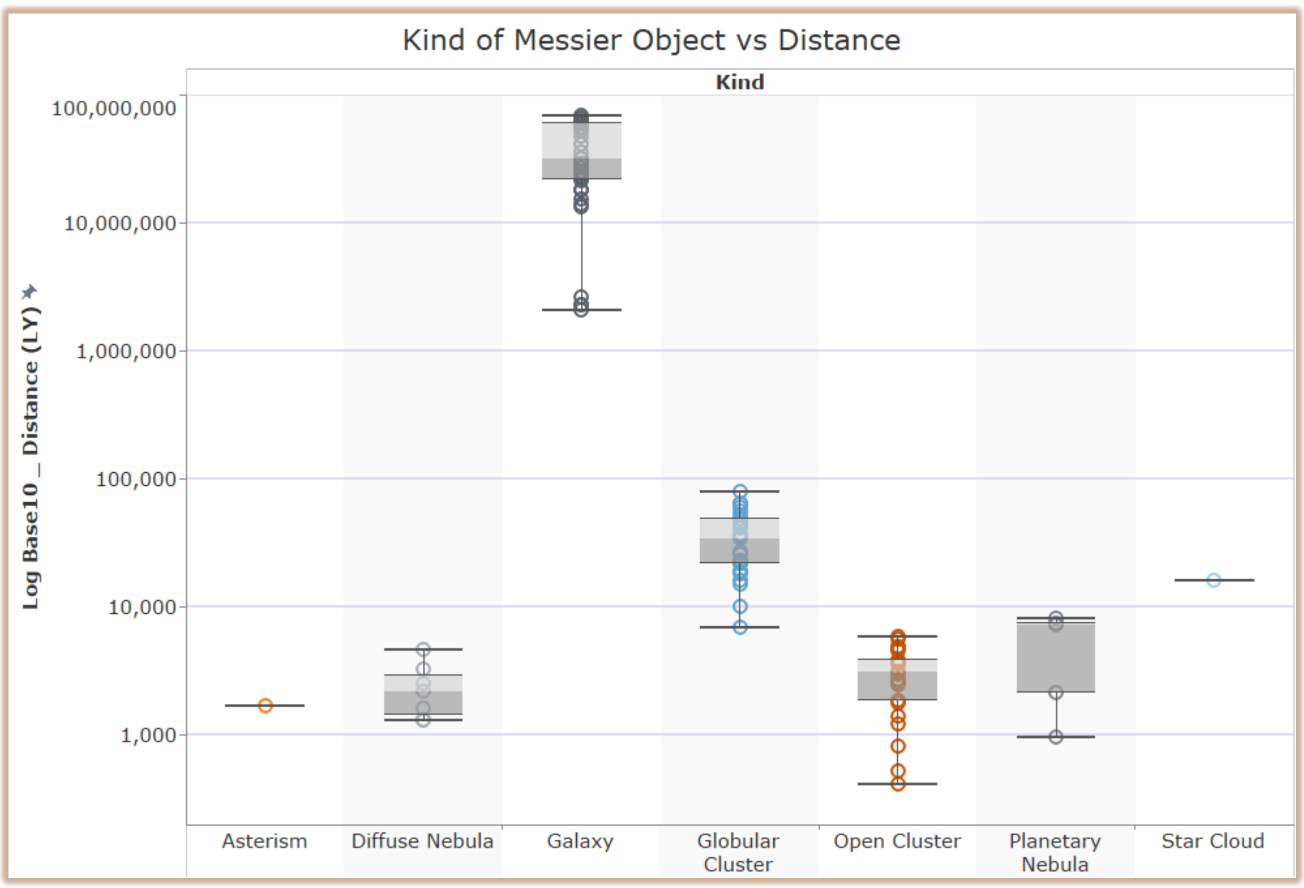




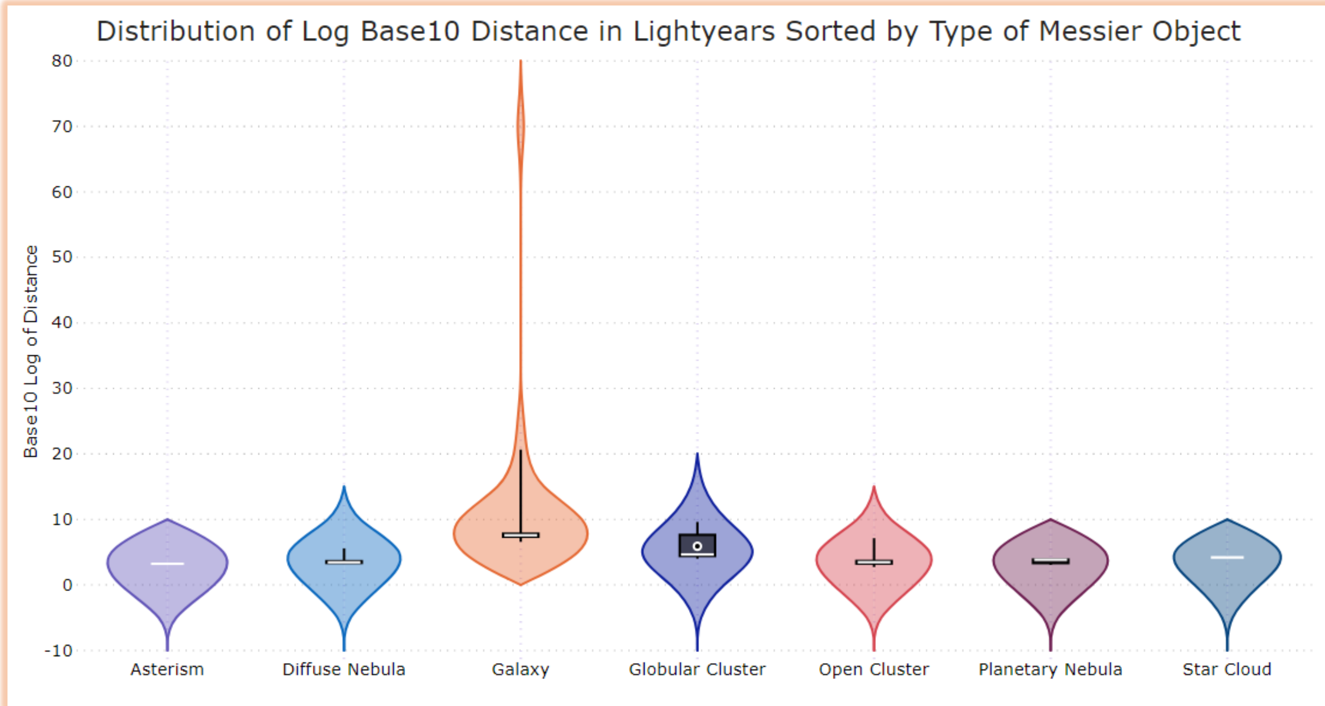
4b. R Studio Messier #40 doesn't have distance or kind entered. While R plots nothing for the NA in distance, it still listed the Kind, which messed up the graph. So, I threw a WHICH clause in the ggplot to remove data that was "". After running the geom_violin, I added geom_jitter on top, then set the log scale on the x axis



4b. Tableau Created with "Kind" in Columns and Distance in Rows. The y-axis was set to a logarithmetic scale, then fixed axis to begin at 200. Added different colors per Kind for fun.

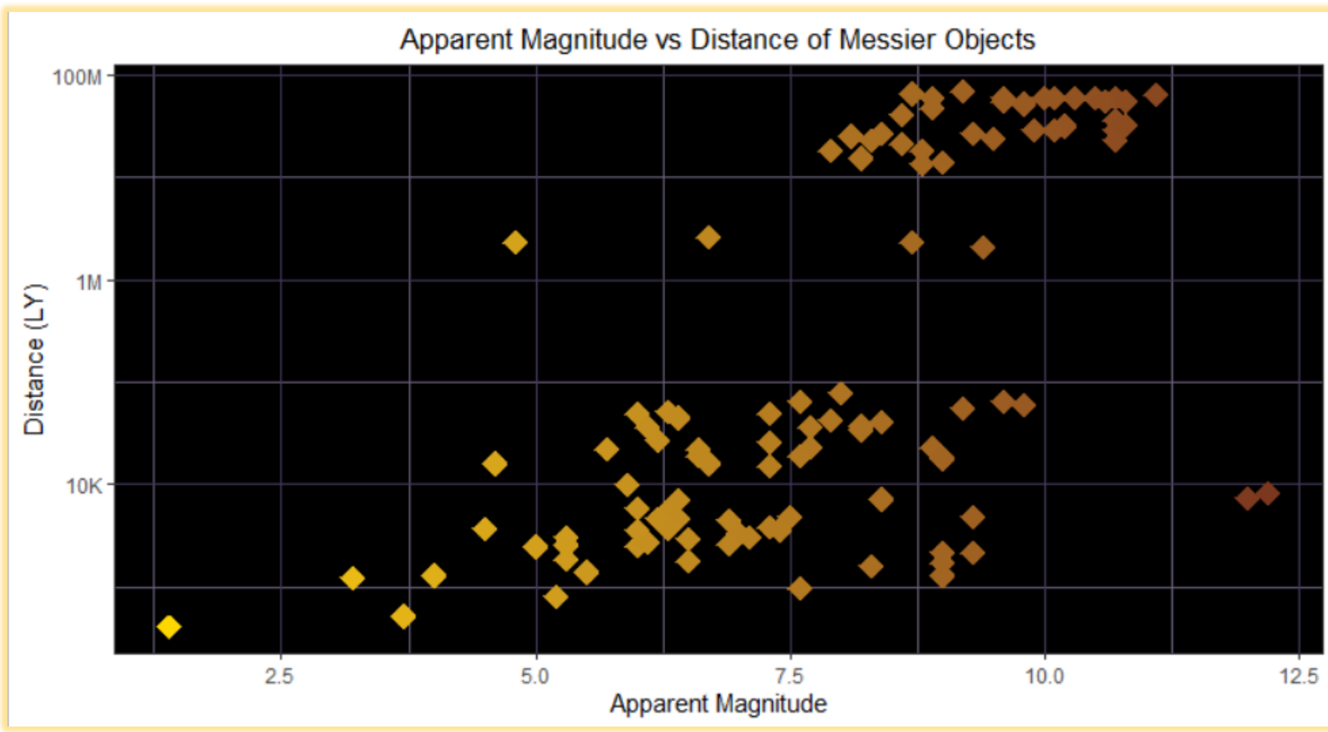


4b. Power BI I really wanted to try the violin plot for this one by there wasn't a log scale allowed in axis so a new column with mathematically determined log was created. Added colors again.

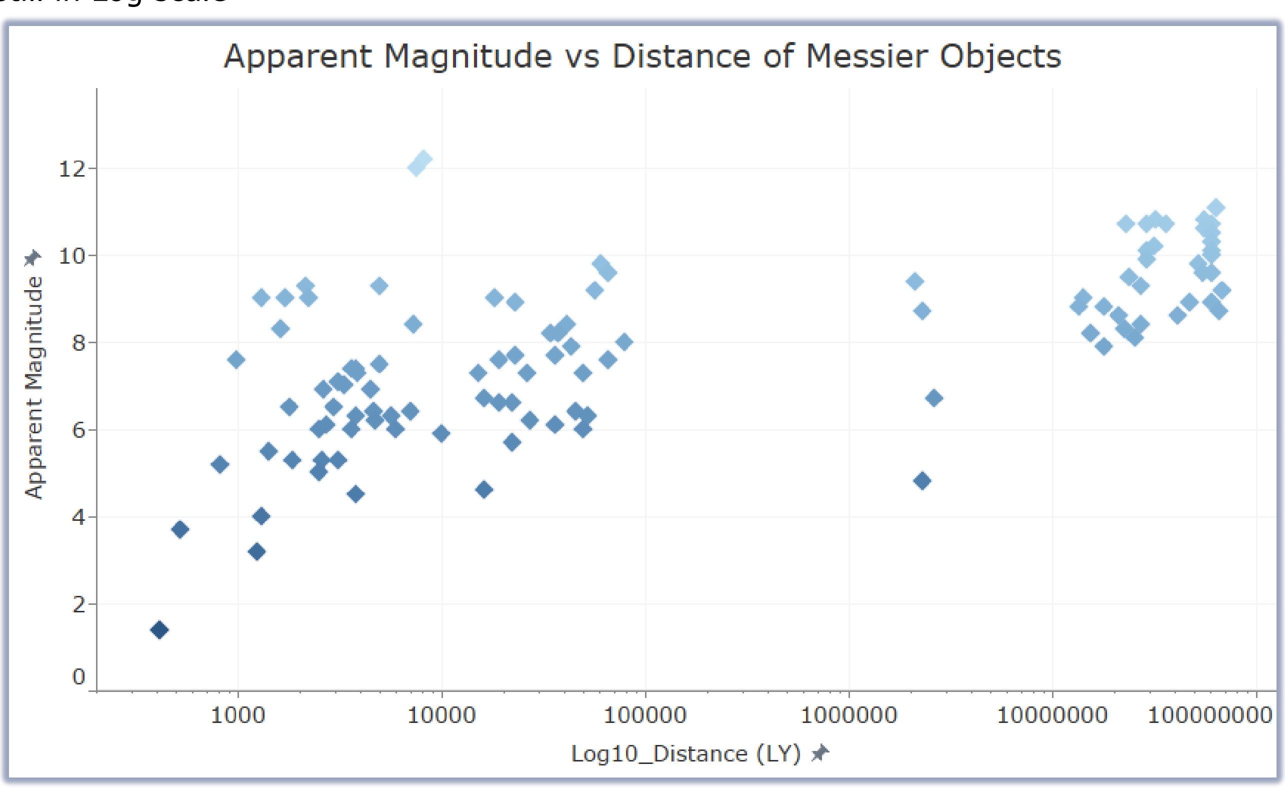


4c.

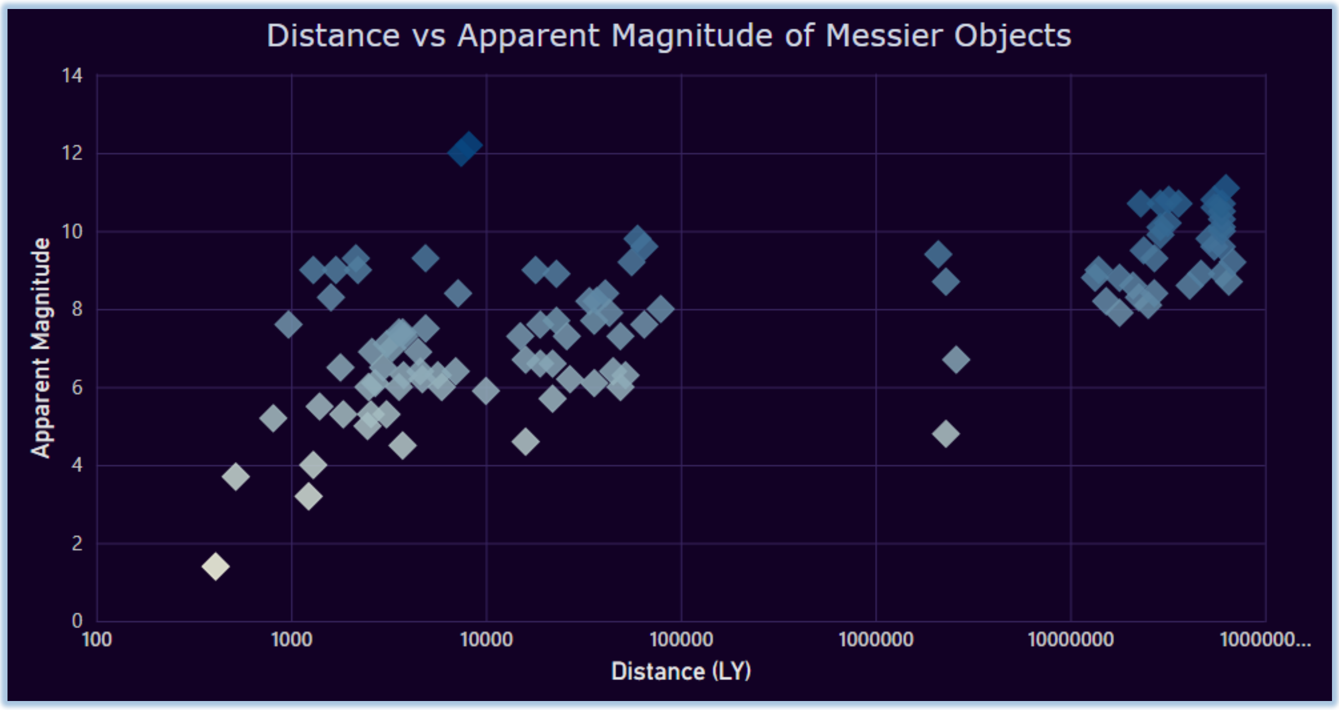
4c. R Studio Was excited about this because upon reading the prompt I knew the graph would look like stars... Changed theme elements and set continuous color scale.



4c. Tableau This one was made with white background, flipped palette. Distance in LY still in Log scale

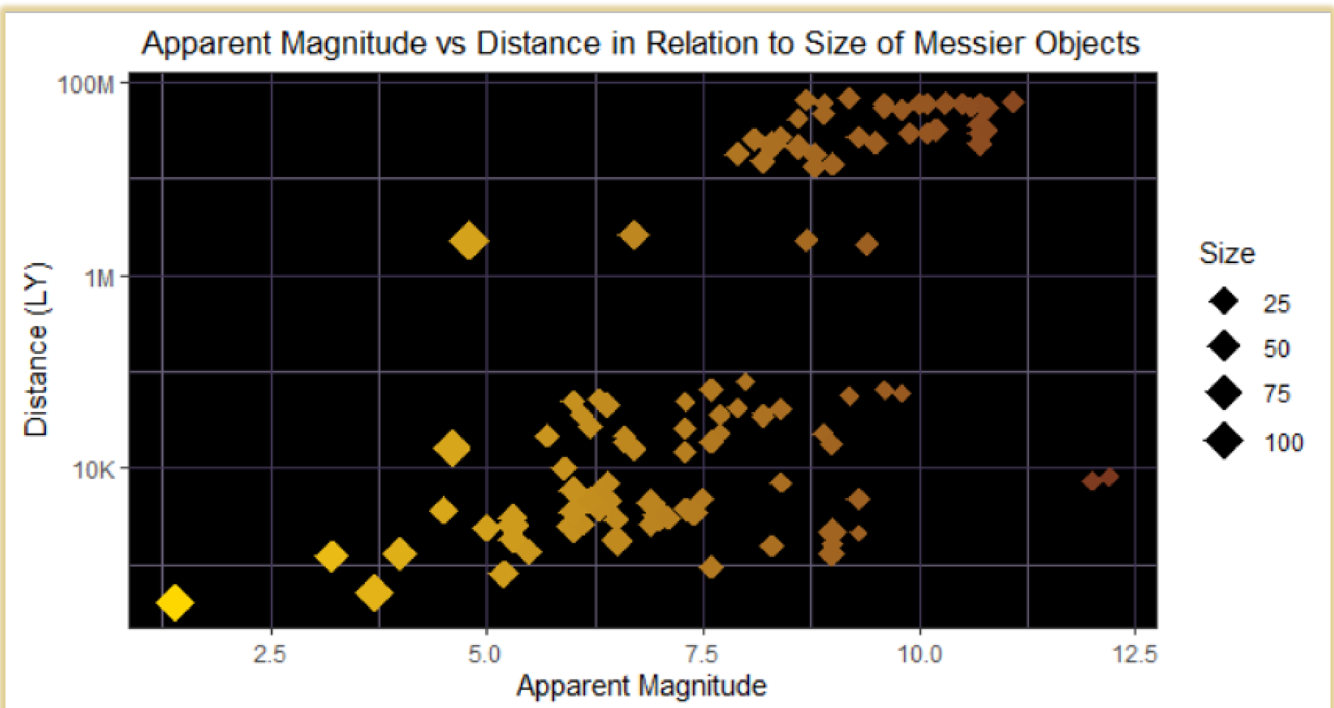


4c. Power BI Best graph award for this prompt is Power BI. The high and low colors were chosen by hand. I like the vibe, it makes me feel like we're in the planetarium

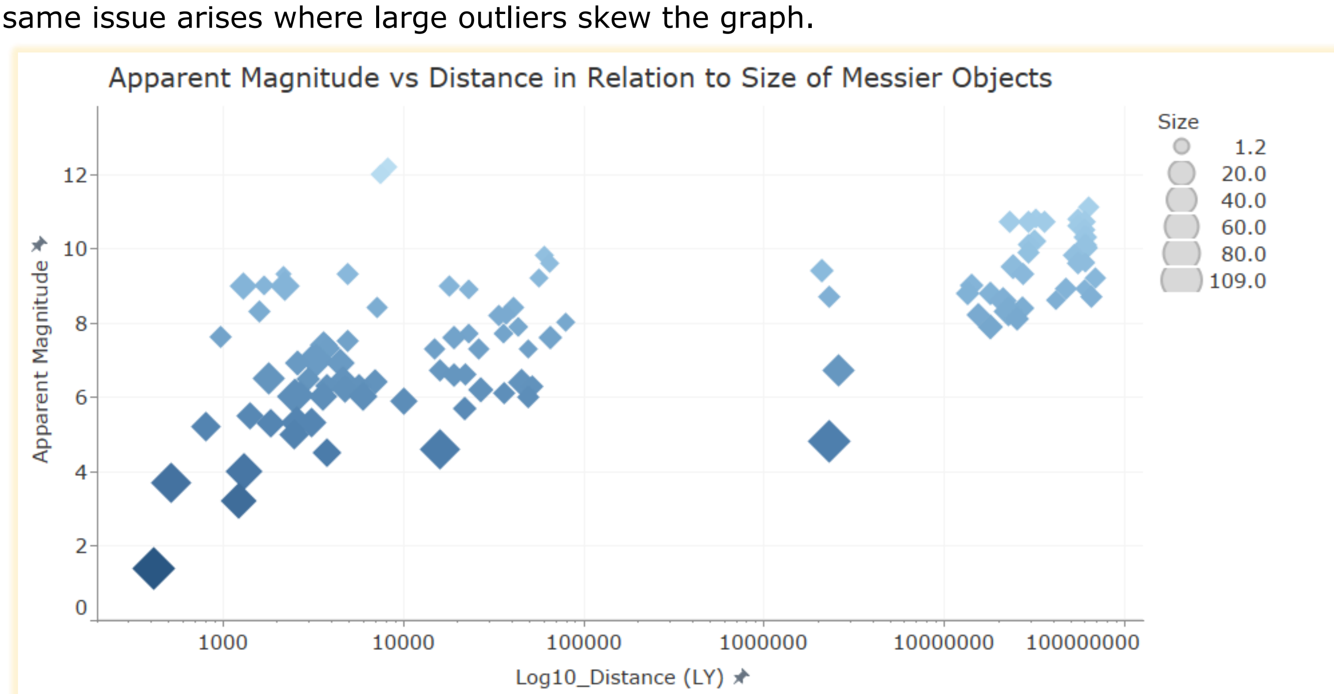


4d.

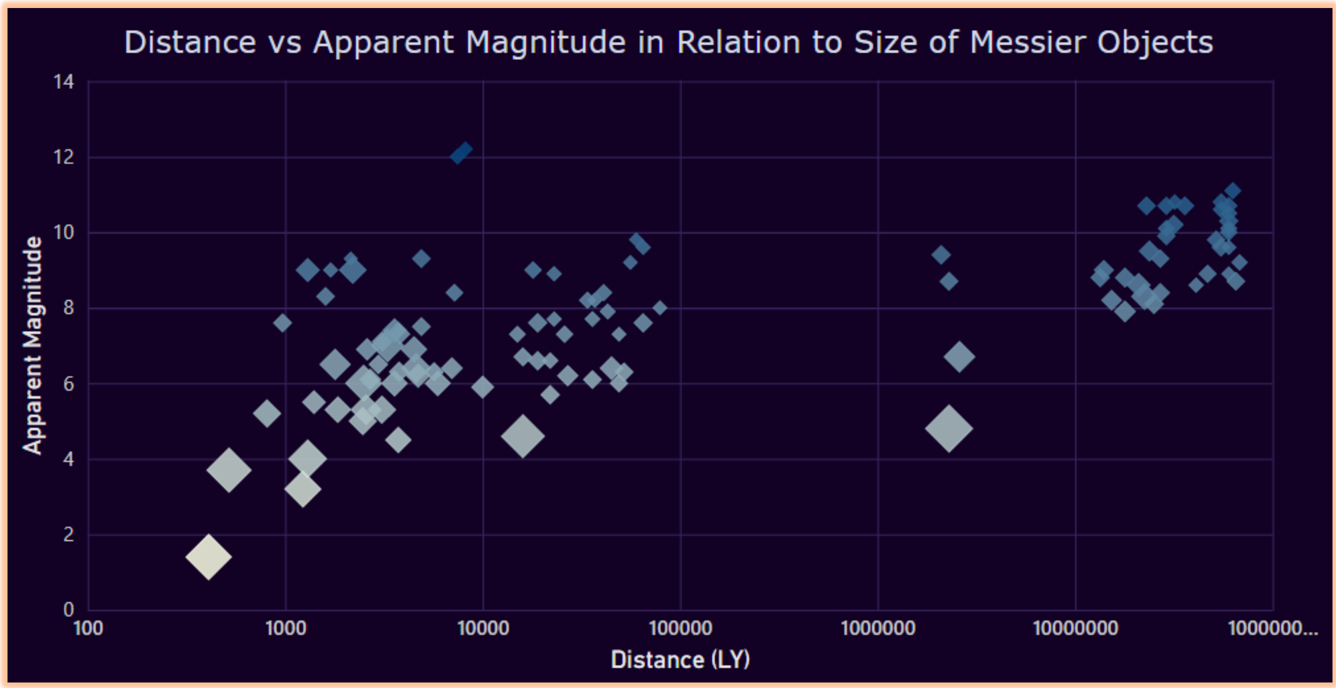
4d. R Studio This is the same graph as 4c but with size added. The issue with this is it is hiding a lot of size variation. There are objects small as 1.2 and large as 109



4d. Tableau Though Tableau has an easy way to adjust the size ranges of objects, the same issue arises where large outliers skew the graph.



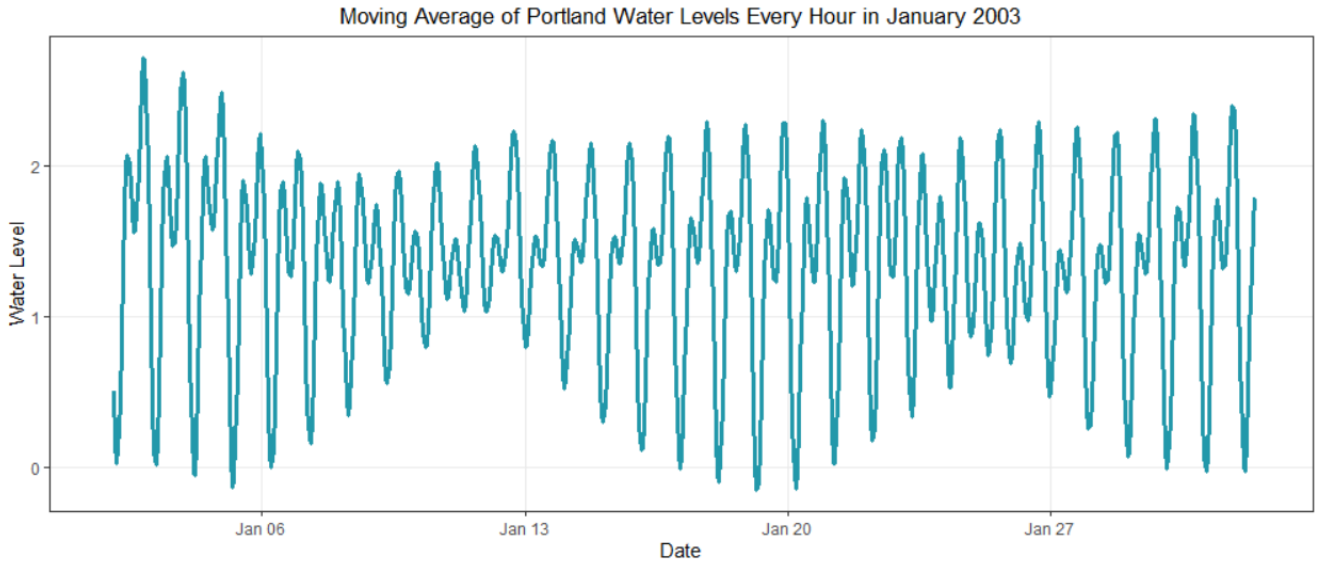
4d. Power BI This is a duplicate of the previous graph, with median of size added as a size component.



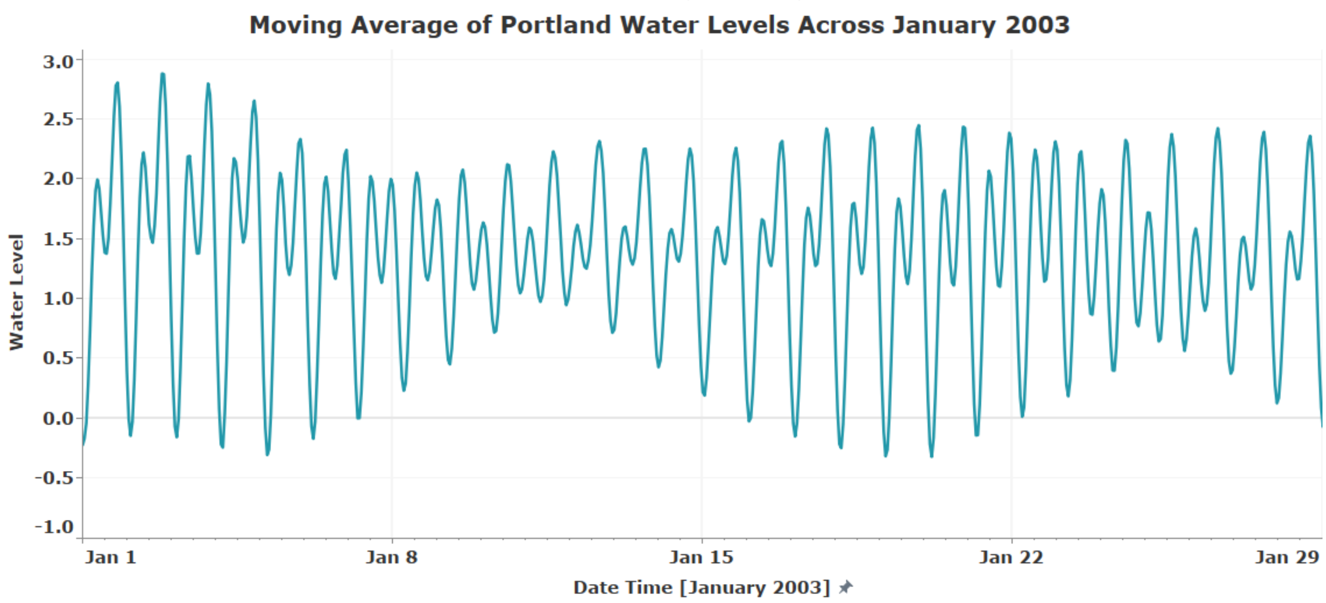
The issue with these graphs is they are all hiding a lot of size variation. There are objects small as 1.2 and large as 109, but most the object sizes are smaller than 20. Maybe graphing this without the larger sized outliers would help clarify the data

5a.

5a. R Studio First lubridate was used to create a separate DateTime column that combined the Time and Date columns into lubridate format. This column was used along the x-axis. Geom_moving average was used to make the lines.

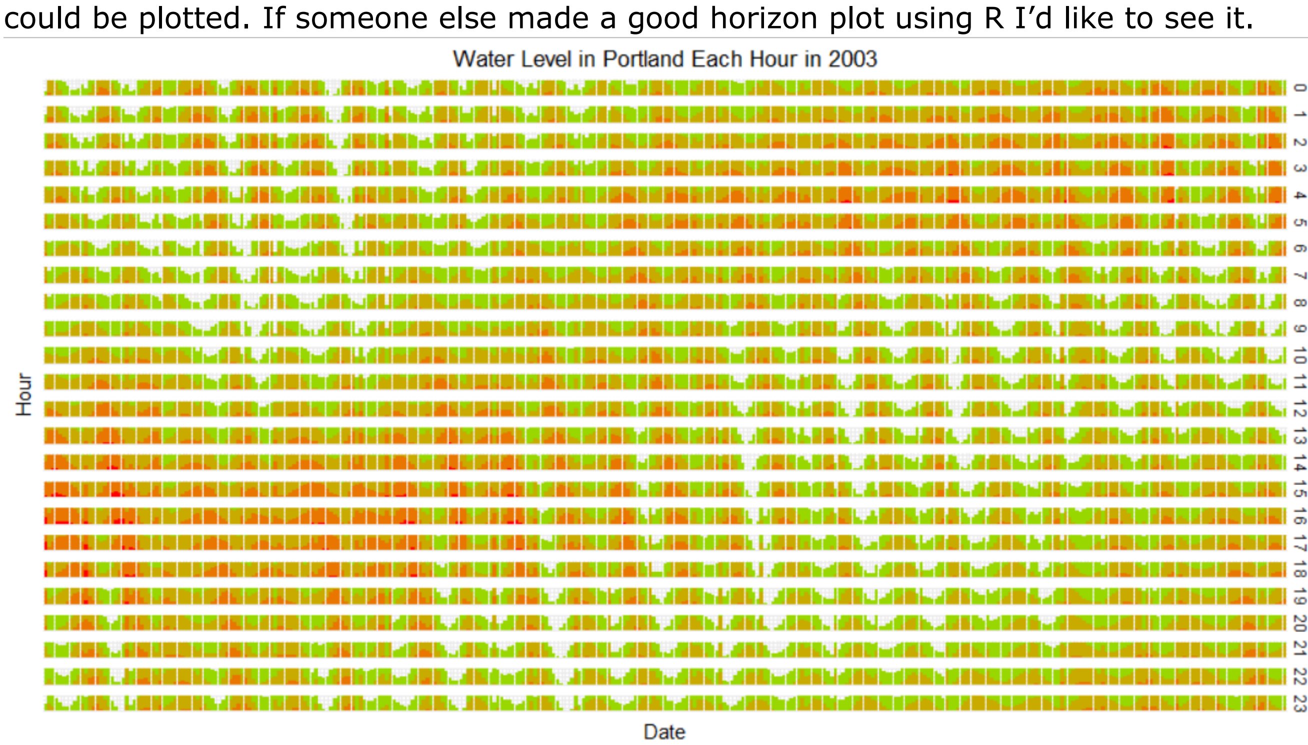


5a. Tableau For this I chose the range of dates from Jan 1 to Jan 29 since that's one full moon cycle, and Tableau shows the moving average quite clearly



5b.

5b. R Studio So… I discovered ggplot_horizon. The issue with it is the elements of ggplot don't work with it so there wasn't a way to format axis, or even the title of the legend, which is why there is no legend. This also meant that the text for the hour of day wasn't adjustable, which led me to remove the ":00" in time just so the hour number could be plotted. If someone else made a good horizon plot using R I'd like to see it.



5b. Tableau After being traumatized by R's horizon plot, Tableau was instead used to make this pretty graph that shows each water level at every time across the year. Moving average was calculated as shown in class. As usual I added sequential color pattern for flair.