Website Link: <https://kyang04.github.io/DS4200-MBTA/>

HTML/CSS/JS/Python Files: <https://github.com/kyang04/DS4200-MBTA>

Visualization Designs

We begin our webpage with the first data visualization, a stacked bar chart made using Altair to demonstrate the number of active speed restrictions at the beginning and end of the month. We took data from the “MBTA Rapid Transit Speed Restrictions by Day” dataset. We aggregated the number of active speed restrictions at the beginning of the month, and the number of active speed restrictions at the end of the month. The stacked barplot was chosen to demonstrate how many speed restrictions are being cleared within a monthly time frame. The stacked bars are the marks, and the colors used for the stacked bars are the channels. Stacked bars make it easier for viewers to see how many restrictions were cleared or added within a monthly period, as they can compare the quantities of restrictions from the start and end of the month. The color helps to differentiate the start and end quantities of restrictions. Another feature of using a stacked bar chart is that users can see a clear trend with regards to the speed restrictions, if the height of the bars is increasing, then we know that there was an increasing number of speed restrictions during that monthly period, and if we see the height of the bars decreasing, it shows that the restrictions were being removed/cleared.

For our second visualization, we created a boxplot that synthesizes MBTA’s speed restriction data in mphs, with several key design liberties taken to shape the narrative. First, it assumes uniform formatting across CSV files in a folder, simplifying ingestion but ignoring potential inconsistencies. Speed restriction values are stripped down to their numeric components, omitting annotations or units for the sake of clarity. Monthly aggregation is used instead of daily or weekly data, offering a balance between granularity and readability, while averages are calculated using the mean, which may gloss over skewed distributions where medians could provide more insight. Color is manually assigned to four train lines, which may not reflect the full set of lines in the data but keeps the chart clean with our focus lines (orange, blue, green, and red). The boxplot sorts by descending median restriction speed, subtly steering viewer attention toward underperforming lines. In addition, there is tool-tip, so that the user can specifically see the IQR statistics of each box plot. Overall, the design frames the MBTA’s operational challenges in a visually digestible and comparative format, favoring high-level storytelling over exhaustive detail.

For our third visualization, we created a line plot in Altair showing ridership trends from late 2018 through 2024. The plot includes an interactive legend, allowing users to click on a label to highlight a specific line. This feature improves readability and enables users to focus on individual lines of interest. Although the dataset included all MBTA lines, we chose to highlight the major ones: the Blue, Orange, Green, and Red lines.

Our fourth visualization is a bar plot created in Plotly, displaying the total reported speed restrictions in 2025 for each MBTA line and branch. Each bar is labeled with the corresponding count of speed restrictions, and users can also hover over a bar to view the exact number and line name. A dropdown menu allows users to filter the data by line—for example, selecting the Green Line will display all its branches in the plot. This feature enables users to compare different lines and examine the variations across branches within a single line.

To visualize the evolution of MBTA subway average length of speed restrictions over time, in our fifth visualization, we created a responsive D3.js line chart that maps the number of speed restrictions per month from January 2022 through March 2025. The chart is built by first defining margins and dimensions to maintain spacing around the edges. I parsed the time-series data using d3.timeParse and established scales for both the x-axis (time) and y-axis (restriction count). Using d3.line(), we defined the line generator with a smoothing curve (d3.curveMonotoneX) to emphasize trend continuity. Axes were added with readable tick formats, and the main line was drawn in a vibrant MTA-style blue for clarity. We took a few design liberties: we applied subtle grid lines for better readability, smoothed the line for visual appeal even though actual data may fluctuate more sharply, and placed a multiline annotation inside the chart using tspan elements to narrate key events like the spike in early 2024. Colors were chosen to evoke clarity and professionalism, with a soft gray background and central alignment of the annotation to avoid clutter.