Amrith Srinivas
Lillianne Davis
Jake Samuel
Nitin Chandra

In this project, we used AI to help us write a lot of the boilerplate Javascript D3 code that composed our visualization. We also used it (to varying degrees of success) to generate some of the more complex components of our code that created the radial design of our visualization. We used ChatGPT 5.1 and Gemini 3, and both had very similar strengths and limitations. The visualization required us to build a number of different parts, and then move them together. After writing another script to manually average the original dataset, we used this averaged data as our baseline across the remaining visualization.

Our AI usage and debugging largely focused around 5 different components of the visualization, which were Arc generation (the radial), Radial axes for the metric names, Label rotation, Interactive bubbles, and the tooltips and hover effects that would show upon interacting with those bubbles. We were largely happy with the output of the AI models throughout the project, except for some understandable minor debugging.

The earliest challenge we had was with configuring the Arc within the SVG canvas. The AI model was able to properly debug the error, which was a simple parameter type error. This showed us that the models had intimate knowledge of the documentation, and were able to help us debug far faster (especially for such a small issue).

There were also issues with the Axis labeling, since the visualization example we were given showed complex spokes representative of the metrics, which were hard to implement without a sequence of iterative changes we had to make to replicate the feature. The largest issues we found were placement and orientation (since the spokes are rendered dynamically for each metric), while also keeping it aesthetic and visible like in our example image.

There were also issues implementing the bubble placement and the corresponding tooltips, which were probably the hardest task for the AI to solve. The regional bubbles laid on the axis were hard to implement since they were being dynamically rendered again (and on a scale) over the already dynamically rendered axes. It took a lot of tweaking and refining prompts to get the feature functional. The tooltips were less hard, since the feature is something we've implemented in previous labs, but the larger issue was the generation of code that was "correct", but not aesthetically pleasing or visually clear.

Overall, I think our group had a pleasant experience using AI to implement this project, since it largely cut down the time required to generate what is a 680 line Javascript file, with minimal mistakes. I feel that the limiting factor of the AI is its inability to see and compare the changes it made, as well as to objectively consider the aesthetics and appeal of the generated visualization while designing it (which is somewhat fair, considering it does not have access to the output visual at each stage). The other issue is that, for this large of a file, context became an issue across ChatGPT 5.1, with Gemini being able to handle the longer conversations better.

Complete List of Prompts for This Project

1. Initial Chart & Bug Fixing

Prompts

- *“I get the following error from the devtool:*
  d3.v7.min.js:2 Error: <path> attribute d: Expected arc flag…”
- *“This is the code: [your* `drawband()` *function]”*
- *“ok. That seemed to fix it.”*

---

2. Radial Layout & Axes

Prompts

- *“Now, I want to do the next part which is ‘Radial Layout & Axes’. Implement the core D3 radial layout. Define the angular and radial scales. Render the metric axes (at least 10) radiating from the center.”*
- *“The entire code is as follows: [full script.js pasted]”*

---

3. Axis Label Orientation

Prompts

- *“I want the labels to be rotated within the actual lines.”*

- *“Still not correct. The text label should be slanted and on each of the lines inside the circle.”*
- *“No. I need the text to be rotated within the axis.”*
- *“Still didn't do this. I need the labels to be on each of the lines inside the circle. The label is not always horizontal text. It will change to sit on the lines.”*

---

4. Region Bubbles & Distribution Lines

Prompts

- *“I want to implement the drawlinesandBubbles method. Implement the logic to place the six region dots along each metric axis based on the pre-calculated average values, including the non-overlapping placement logic.”*
- *“The bubbles should not have any fill, just the line colors should change.”*
- *“The original, first circle should not be filled either.”*
- *“The circles should be closer to the axis lines.”*
- *“The purple bubbles should have z-index highest.”*
- *“When I first load the page, the circles are filled with purple. I want to add an opacity for the purple circles.”*
- *“I want to make the top level circle wider.”*
- *“How do I make sure that the small circles do not touch the large circle. That is, I want to add a padding to the main large circle.”*

---

5. Hover Interactions

Prompts

- *“Implement the hover effect for the metrics/regions — change the background color to gray upon hover.”*
- *“The hover only works if I'm on the actual label, but I want the hover to work when I'm in the same wedge.”*

---

6. Tooltip System

Prompts

- *“I want to implement circle hover tooltip. Implement the tooltip (or text overlay) that appears when hovering over a region's circle, displaying the Metric Name, Region Name, and Metric Value.”*
- *“I get the following error: tooltip.style is not a function.”*
- *“I want to make sure that everything is responsive.”*
- *“I want the tooltip to show the percentage.”*

---

7. Meta

Prompt

- *“Now can you give me all the prompts I asked you about this project because I need to write a report about it.”*
- *“I like this can you do it again but also add a short summary of what went wrong and what went well.”*

Here is a concise summary of what went **wrong**, what went **well**, and what you ultimately **implemented successfully**.

✔️ What Went Wrong

1. SVG Arc Generation Error

- The initial `arc()` path string caused SVG to throw:
  "Expected arc flag…"
- Reason: D3 transition interpolated flags (`0`/`1`) into invalid decimals.
- Fix: Use `attrTween` to animate the arc instead of interpolating the raw `"d"` string.

2. Labels Would Not Rotate Correctly

- Labels initially stayed horizontal or misaligned.
- Problem: misunderstanding of whether labels should be radial, tangential, or aligned with axis lines.
- Fix: Rotate around `(0,0)` then translate outward along axis or tangent.

3. Circles Overlapping & Wrong Position

- Bubbles could overlap each other or the outer circle.
- Fixes included:
  - Adding perpendicular tangent offsets
  - Increasing offset spacing
  - Adding padding from outer radius

4. Purple Circles Appearing Filled

- First load showed filled purple bubbles despite `fill="none"`.
- Reason: old DOM state or CSS fill default.

- Fix: explicitly enforce `fill="none"` and switch to `stroke-opacity`.

5. Tooltip Crashed

- Error: `tooltip.style is not a function`.
- Reason: `tooltip` was not a D3 selection.
- Fix: Create tooltip with `d3.select("body").append("div")`.

6. Hover Highlight Only Worked on Labels

- Wedges were not receiving pointer events.
- Fix: Add invisible wedge paths with `pointer-events: all`.

✔️ What Went Well

1. The Radial Architecture Was Solid

- Your base structure (scales, ordering of metrics, region averaging) was well-organized.
- Allowed clean integration of axes, arcs, bubbles, and interactions.

2. Angle & Radius Math Was Correct

- Your radial coordinate system (angles in radians, top at -90°) was consistent.
- This made positioning, labels, arcs, and bubbles predictable.

3. The Bubble Logic Extended Cleanly

- Because each bubble datum contained `{metric, region, value, normalized}`, adding:

    - color rules
    - z-order rules
    - tooltip logic
    - hover logic
      was smooth and straightforward.

4. The Visualization Became Fully Interactive

You successfully implemented:

- Metric/region hover highlighting
- Responsive resizing through `viewBox`
- Tooltips showing raw values and percentage
- Z-index ordering for selected region
- Axis labels positioned dynamically along lines
- Non-overlapping region bubbles
- Custom arc transitions without SVG errors

5. Your Iteration Process Was Excellent

You refined:

- label orientation
- bubble styling
- hover mechanics
- tooltip content
- padding & spacing

- circle layering

step-by-step, producing a polished radial dashboard.