Racial Wealth Gap Visualization Design Document

Ryan McNeil

The Graduate Center, CUNY

Department of Computer Science

CSC83060

Completed animation: <https://youtu.be/2-B77r16KO0>

Special thank you to Lisa Ciaccio, software engineer.

I explicitly aim to illustrate the trend of the white-black wealth gap in American over time from Emancipation through the current day. I do so using data from the landmark recent study, Derenoncourt et. al. (2022) (<https://www.nber.org/papers/w30101>), which constructed a dataset tracking the black-white racial wealth gap in the United States from 1860 to 2019, assembled from numerous (often fragmented) sources and calculations. To date, this is the only study which successfully does so for such a large time range. Prior datasets on the matter (including the dataset I would have otherwise used, from the World Bank website) only begin in the mid-20th century, and are typically discontinuous.

I choose to produce an *animated* graph for this dataset because I want to allow users to inspect the racial wealth gap from the start of the dataset *relative to any chosen year*, which an animation/video with controls makes possible. I believe that a low-fps continuous framerate animation more effectively communicates the scale of the time period being represented. This is intended to mitigate the problems of logarithmic perception and the limits of human perception of magnitude in static visualizations. This technique (similar in effect and use to panning in a 3-dimensional space) is especially useful in domains where extreme scale can lead to loss of perspective (i.e. astronomy). In other words, static graphs can fail to communicate the true scale of this 160-year window, and the slow rate of reduction in the racial wealth gap over this period. Additionally, as the time axis compresses, each additional year (yframe+1) takes up less and less visual space. Thus, the animation accentuates that *despite* the long period over which the racial wealth gap fell from, say, 56:1 to 10:1, much of the 20th century has seen progress slow to a crawl; the wealth gap ratio has taken the better part of a century to fall from 10:1 to 5:1, and then mostly stagnated.

* I tried to follow a Tufte-ian approach to my design. My intuition being that an animated graph would be particularly susceptible to appearing “too busy,” so I tried to limit the amount of information on-screen at any one time.
* Fonts, background, and styling:
  + I tried to emulate 4 different styles, allowing the graph to physically tell the story of changing eras through not just its informational content, but through its appearance.
    - The greatest drawback in producing this visualization was the many challenges and limitations that animation posed to visual design. If I had been able, I would have preferred to control more elements and provide more detailing (especially for older styles).
  + 1860-1899: Styled after Minard & Nightingale
    - Hand-written fonts, primarily in delicate script.
    - Reduced alpha to emulate faded ink
    - Background to mimic weathered canvas.
  + 1900-1939: Styled after WEB Du Bois.
    - All fonts & styles mimic those used in the famous W.E.B. Du Bois World’s Fair visualizations.
    - Reduced alpha and size to mimic era typography.
    - Background to emulate weathered paper.
  + 1940-1979: Styled after early IBM spreadsheets & early computer displays
    - Data visualization “dark age” in the early 20th century, as some sources put it offered additional challenge to find a appealing style.
    - 2 variants of console/typewriter script.
    - Background to evoke cheap paper or a dim CRT display.
    - On its own, I would have inverted the colors with bright white graph on black background (to mimic a vector display), but this creates too jarring of a transition.
  + 1980-2019: Plain modern style, like you might see in a newspaper or clean output.
    - Times New Roman title, to mimic newspaper. Plain sans-serif plot font (recommended in Data Vis textbook for readability).
    - Brighter background invoking modern LCD display.
    - If it were possible, I would have wanted to incorporate Matplotlib’s new Viridis color scheme (for both aesthetic and accessibility), but the elements I have access to manipulate mid-animation were limited.
* X-axis ticks: I chose to change the interval size as the animation progresses in order to prevent too many ticks from being rendered on screen at once, avoiding visual clutter.
* Y-axis ticks: I chose a maximum of 60 & interval of 10. Any higher interval becomes hard to compare to the trendline, any lower leads to clutter.
  + Although the starting graph value is 56, 60 offers a “nice” number allowing neat 10-long tick intervals.
* Secondary X-axis: Following the rule of 3s laid out in *Design is Storytelling*, I didn’t want more than 3 items of information to strike viewers at first glance. Here I want them to only start by seeing 1) Year, 2) Events (to better contextualize Year), and 3) Wealth gap ratio.
* Gridlines: I chose not to include typical gridlines at year intervals, as the way they animated made the graph busy and difficult to read.
  + I draw gridlines at events to allow easier comparison to ratio.
  + I do this instead of drawing gridlines at years because I believe this would both be too crowded, and was too jarring when year tick intervals change.
* Labels on plot:
  + I report the wealth gap during years of significant events by showing a small bit of text at the intersection of the trendline and each gridline. I keep this text small so that it is unobtrusive to the main 3 pieces of information I want to strike viewers first (aforementioned). These are intended for a viewer who may want to pause and inspect the graph more closely.
* Trendline:
  + The trendline I do not iterate *at all*. The alpha, width, and color all remain constant. *Any* changes in the trendline end up very disorienting during transitions.
* Points at intersections
  + Meant to better highlight intersections as significant joints in the trend line.
  + Static black color. Line or point color changed during transitions ends up being very disorienting.