Building an Interactive Population Pyramid Explorer

Exploring demographic patterns across US counties through interactive visualization using {R-Shiny} and {mapgl}

Pukar Bhandari

pukar.bhandari@outlook.com

2025-06-25

Table of contents

1 Introduction

Population demographics are fundamental to transportation planning and urban analysis. Understanding age distributions helps inform infrastructure decisions, transit planning, and resource allocation. In this post, this blog will walk through building an interactive Shiny application that combines population pyramids with spatial visualization using {mapgl} package.

The app allows users to explore county-level demographic data across all US states, displaying both the geographic context through an interactive map and detailed age-sex distributions through population pyramids.

2 Application Overview

The Population Pyramid Explorer features:

- **Interactive state selection** with smooth map transitions
- **County-level demographic visualization** using Census tract data for context
- **Dynamic population pyramids** showing age-sex distributions
- **Responsive design** using MapLibre's story map format
- **Real-time data integration** from the US Census Bureau

3 Repository and Code Access

The complete source code for this application is available on GitHub: Population Pyramid Explorer Repository

The repository includes additional documentation, deployment instructions, and example usage scenarios.

4 Key Dependencies and Setup

The application leverages several powerful R packages for data processing, spatial analysis, and visualization:

```
# Custom function to load packages, installing them first if not already
installed
load_pkg <- function(pkg) {
  if (!require(pkg, character.only = TRUE)) {
    install.packages(pkg)
    library(pkg, character.only = TRUE)
  } else {
    library(pkg, character.only = TRUE)
  }
}

# Data manipulation and visualization
load_pkg("dplyr")
load_pkg("tidyr")
load_pkg("ggplot2")
```

```
load_pkg("scales")

# Spatial data and mapping
load_pkg("sf")
load_pkg("mapgl")
load_pkg("tigris")

# Census data integration
load_pkg("tidycensus")

# UI framework
load_pkg("shiny")
```

The `load_pkg()` function ensures packages are installed and loaded, making deployment more robust across different environments.

5 Application Architecture

User Interface Design

The UI implements {mapgl}'s story map concept, creating a scrollable narrative experience:

```
mapgl::story_maplibre(
  map_id = "map",
  font_family = "Poppins",
  sections = list(
    "intro" = mapgl::story_section(...),
    "state_detail" = mapgl::story_section(...)
  )
)
```

This approach provides intuitive navigation between the overview (state selection) and detailed analysis (county demographics).

Reactive Data Architecture

The server logic employs several reactive components that efficiently manage data flow:

State Geometry: Loads and transforms state boundaries using the Web Mercator projection (EPSG:3857) for optimal web mapping performance.

Census Tract Data: Dynamically fetches median age data at the tract level, providing geographic context for county selection. The data pipeline includes coordinate reference system transformation and popup content preparation.

Population Estimates: Retrieves detailed age-sex breakdowns from the Census Bureau's Population Estimates API, specifically formatted for pyramid visualization.

6 Data Processing Pipeline

Geographic Data Preparation

The application processes spatial data through several key transformations:

```
states_sf <- tigris::states(cb = TRUE) |>
  sf::st_transform(crs = 3857) |>
  sf::st_zm(drop = TRUE, what = "ZM")
```

Using cartographic boundary files (`cb = TRUE`) provides simplified geometries that render faster while maintaining visual accuracy at the national scale.

Census Data Integration

The demographic data pipeline combines two Census Bureau APIs:

1. **American Community Survey (ACS)** for tract-level median age visualization
2. **Population Estimates Program** for detailed county-level age-sex distributions

The tract-level data provides spatial context with median age choropleth mapping, while county estimates power the detailed population pyramids.

Data Transformation for Visualization

Population pyramid data requires specific transformations:

```
dplyr::mutate(
  value = ifelse(SEX == "Male", -value, value),
  age_min = ifelse(
    stringr::str_detect(AGEGROUP, "under"), 0,
    as.numeric(stringr::str_extract(AGEGROUP, "\\d+"))
  )
)
```

Male populations receive negative values for the traditional pyramid layout, and age groups are parsed to enable proper ordering.

7 Interactive Mapping Features

Map Initialization and Styling

The base map uses Carto's Voyager style for clean, readable cartography:

```
mapgl::maplibre(
  mapgl::carto_style("voyager"),
  center = c(-98.5, 39.5),
  zoom = 3,
  scrollZoom = FALSE
```

```
) |>
mapgl::set_projection(projection = "globe")
```

The globe projection adds visual appeal while the disabled scroll zoom prevents navigation conflicts with the story map interface.

Dynamic Layer Management

Map layers update responsively based on user selections:

- **State boundaries** provide national context with selective highlighting
- **Census tract fills** show median age patterns using a continuous color scale
- **Interactive popups** display detailed demographic information

The color scheme uses ColorBrewer's spectral palette, providing intuitive age visualization from young (red) to old (purple).

Smooth Transitions

Section-based navigation triggers smooth map animations:

```
mapgl::fly_to(
  center = c(-98.5, 39.5),
  zoom = 3,
  bearing = 0,
  pitch = 0,
  duration = 1500
)
```

These transitions maintain user orientation while providing engaging visual feedback.

8 Population Pyramid Visualization

Chart Design Philosophy

The population pyramid uses a horizontal bar chart approach with careful attention to readability:

- **Dual-color scheme**: Navy for males, dark red for females
- **Absolute value labeling**: Clear population counts despite negative male values
- **Minimal grid lines**: Focus attention on data patterns

Age Group Processing

Age categories from the Census API require standardization for consistent display:

```
scale_y_discrete(labels = ~ stringr::str_remove_all(.x, "Age\\s|\\syears"))
```

This removes redundant text while preserving essential age range information.

Dynamic Scaling

The x-axis automatically adjusts to population magnitudes using thousands formatting, ensuring readability across counties of varying sizes.

9 Technical Implementation Notes

Performance Optimization

Several design decisions optimize application performance:

- **{tigris} caching** reduces repeated API calls for geographic data
- **`selectize` server-side processing** handles large county lists efficiently
- **Reactive dependency management** prevents unnecessary data updates

Error Handling and Validation

The application includes robust input validation:

- `shiny::req()` ensures required inputs before processing
- {tidycensus} validation functions prevent invalid state/county combinations
- Graceful handling of missing or incomplete demographic data

10 Future Enhancements

Potential extensions to this application include:

- **Multi-county comparison** capabilities
- **Time series analysis** showing demographic changes over years
- **Additional demographic variables** (income, education, housing)
- **Export functionality** for charts and data
- **Integration with transportation metrics** for comprehensive planning analysis

11 Conclusion

This Population Pyramid Explorer demonstrates the power of combining modern web mapping with demographic visualization. By leveraging R's robust ecosystem for spatial data and statistical graphics, we can create compelling tools for understanding population patterns that inform transportation planning and policy decisions.

The modular architecture makes it straightforward to extend the application with additional demographic variables or geographic scales, while the story map format provides an intuitive user experience that encourages exploration and discovery.

Whether you're a transportation planner analyzing ridership demographics, an urban researcher studying neighborhood change, or a policy analyst examining service delivery patterns, this application framework provides a foundation for demographic-geographic analysis that can be adapted to various planning contexts.

Want to contribute or suggest improvements? Visit the project repository at: https://github.com/arpuuk/population-pyramid-explorer