SPEC REU R Resources: Basic Data Visualization with ggplot2

Alix Ziff, Gaea Morales, Zacahary Johnson, Claudia Salas Gimenez, Ben Graham

Summer 2024

Welcome to the first part of the Data Visualization module. Visualizing data is key for analysis and communicating patterns effectively in research. By the end of this module, you'll be able to create, interpret, and customize basic univariate plots, and produce polished, publication-ready visuals that highlight key data insights.

In this walkthrough, we'll explore data visualization in R using the `ggplot2` package, focusing on univariate plots such as histograms, density plots, and bar charts. We'll work with data from the World Development Indicators to examine energy consumption over the past 25 years and create a bar plot using survey data from the SPEC Lab's LEWIS registry.

Initial Setup

Start by setting your working directory to the Training Data folder on your machine, loading the `tidyverse` and `ggplot2` packages, and loading the dataset `wdi_cleaned_part1.csv` ensuring that strings are read as character types.

Helpful Hint: Set `stringsAsFactors` to `FALSE` so that string variables (e.g. variables with letters in their values) are read in as character strings and not as factor variables.

```
# Set working directory
#setwd("YourFolderPath")

# Load required libraries
library(tidyverse)
library(dplyr)
library(ggplot2)

# Load the data
dat <- read.csv("wdi_cleaned_part1.csv", stringsAsFactors = FALSE)
```

Take a moment to reflect what do these variables represent. For more context, refer to the World Development Indicators data catalog for detailed descriptions of the indicators. Keep in mind that since these datasets have been modified for this walkthrough, the variable names may differ slightly from those in the original codebooks.

Let's Explore the Data

Before we begin visualizing the data, it's important to understand its structure and the types of variables it contains. To do this, we can use the `str()` function, which provides a snapshot of the dataset's structure, including the variable names, data types, the first few entries for each variable, and the overall dimensions. This step helps identify the kinds of visualizations and analyses we can apply to each variable.

```r
# Check the structure of the data
str(dat)
```

```
## 'data.frame':    5425 obs. of  5 variables:
##  $ country            : chr  "Afghanistan" "Albania" "Algeria" "American Samoa" ...
##  $ year               : int  1992 1992 1992 1992 1992 1992 1992 1992 1992 1992 ...
##  $ electricity_pop    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ energyuse_pop      : num  NA 418 884 NA NA ...
##  $ renewable_energyuse: num  62.24 46.03 0.29 NA NA ...
```

Introduction to ggplot2

The ggplot2 package, created by Hadley Wickham, employs Leland Wilkinson's "The Grammar of Graphics" principles to build graphs using a dataset, a set of geoms (visual marks that represent data points), and a coordinate system.

For most applications, the code to produce a graph in ggplot2 is roughly structured as follows:

```
ggplot(data = dat, aes(x = , y = , color = )) + geom_point() + labs(title = "Your Title Here")
```

- `ggplot()`: Function to initiate a graph in `ggplot2 package`.
- `data`: Dataset employed to produce the plot.
- `aes()`: Aesthetic mappings that describe the visual properties of the graph. The minimum values needed to be specified (for univariate data visualization) is the `x` and `y` parameter.
  - `x`: Maps a variable to the x-axis of the plot.
  - `y`: Maps a variable to the y-axis.
  - `color`: Maps a variable to color coding in the plot, differentiating data points by color.
  - `linetype`: Maps a variable to different line types in line graphs, distinguishing between groups by the style of the line.
- `geom()`: Specifies the type of plot to use.
  - `geom_point()`: Creates scatterplots, using points to represent data points.
  - `geom_line()`: Produces line graphs, connecting data points with lines.
  - `geom_boxplot()`: Generates boxplots, which summarize the distribution of a dataset.
  - `geom_bar()`: Makes bar charts for categorical data, showing counts or sums per category.
  - `geom_histogram()`: Constructs histograms, displaying the distribution of continuous data by binning it into intervals.

For an overview of the most important functions and geoms available in ggplot2, see ggplot2 cheatsheet.

Univariate visualizations (one variable visuals)

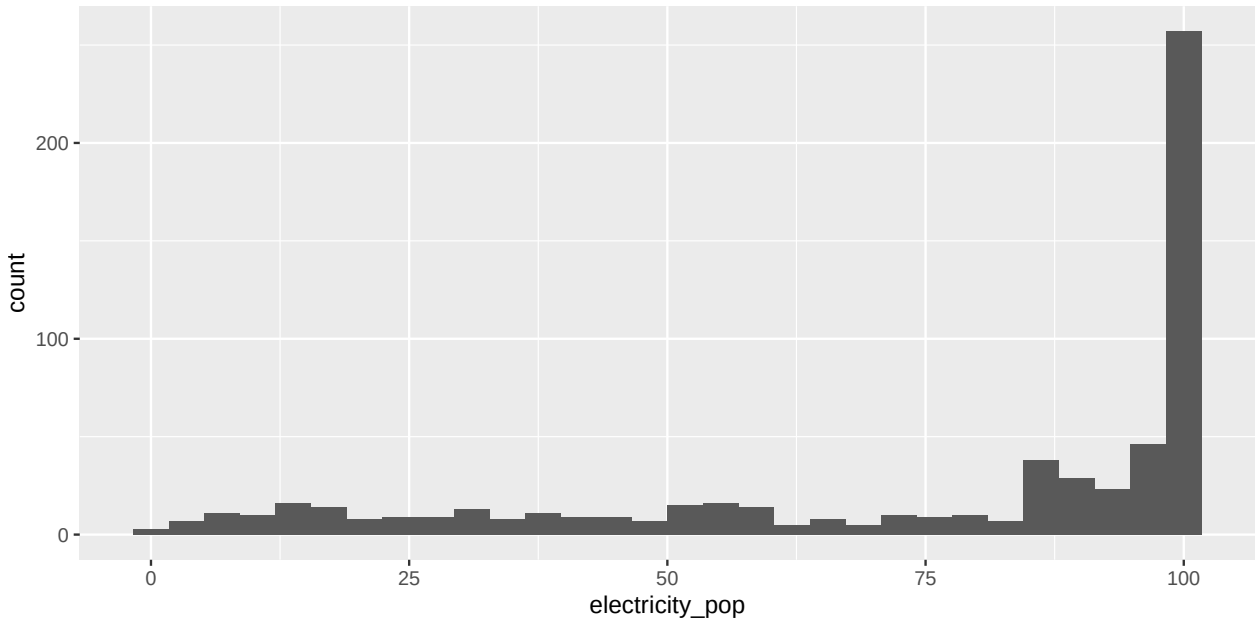
Histograms

Histograms are used to visualize the distribution of continuous variables by dividing them into bins. In this example, we'll graph the distribution of electricity access across countries. Note that since `electricity_pop` represents a percentage, its values are bounded between 0 and 100.

```r
# Summary statistics for 'electricity_pop'
summary(dat$electricity_pop)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    0.00   53.44   92.68   75.74  100.00  100.00    4789
```

```r
# Create a histogram to visualize the distribution of 'electricity_pop'
ggplot(dat, aes(electricity_pop)) +
  geom_histogram()
```

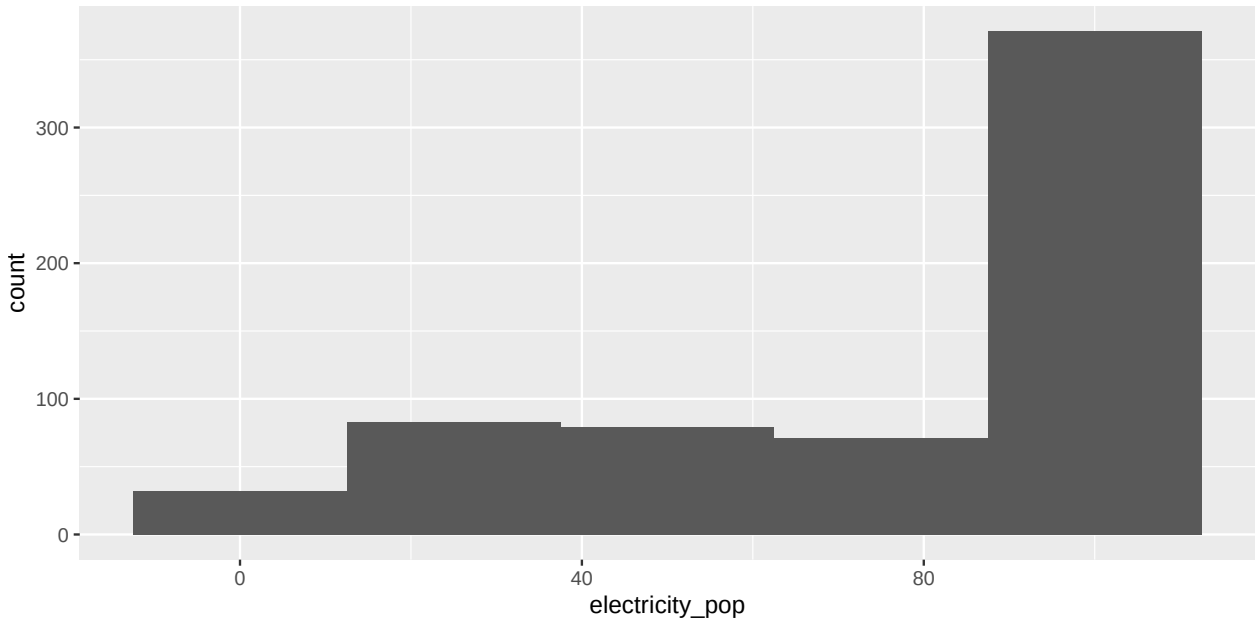


Question 1: What are the elements plotted on the x-axis and y-axis? What determines the width and height of each bar? (See the answer at the bottom of the walkthrough)

Question 2: What conclusions can you draw from the histogram above about the global distribution of electricity access? (See the answer at the bottom of the walkthrough)

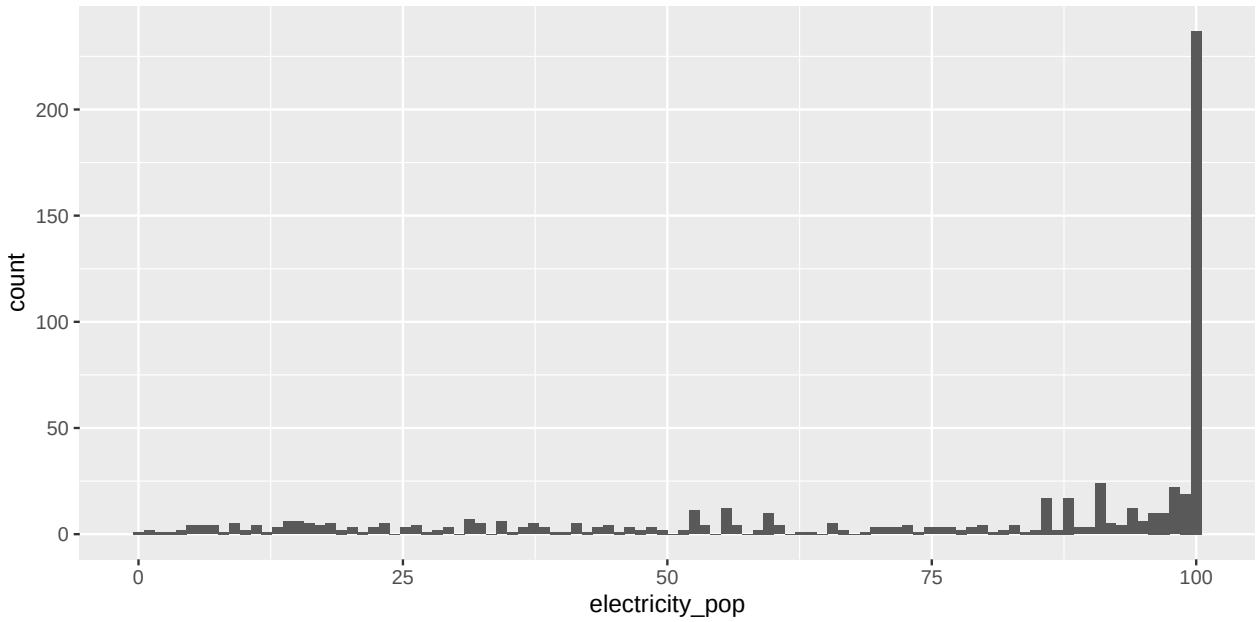
By default, the histogram uses 30 bins, meaning the entire range of the variable (0 to 100 here) is split into 30 equally spaced bins. To control the level of detail in the plot, you can adjust the number of bins. Fewer bins provide a more summarized view, while more bins reveal finer details. Let's set `bins = 5` to group the distribution into five bins (0-19, 20-39, etc.).

```r
# Adjust the number of bins to 5
ggplot(dat, aes(electricity_pop)) +
  geom_histogram(bins = 5)
```



Question 3: How does the graph change if we specify `bins = 100`? (See answer at the bottom of the walkthrough)

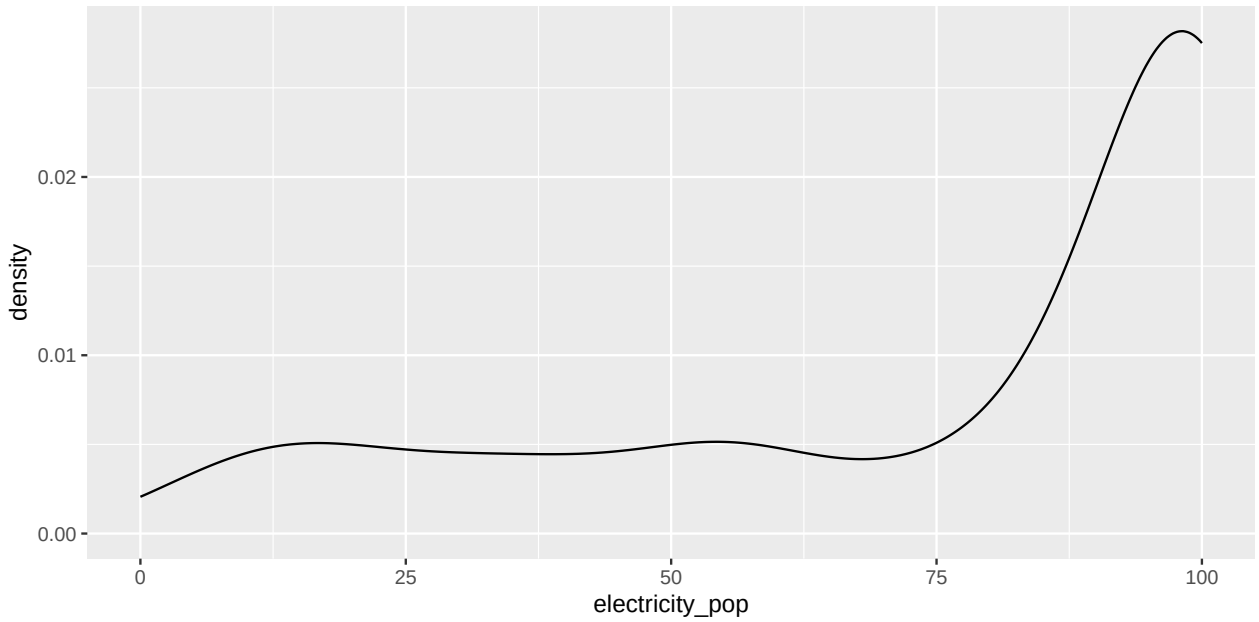
```
# Adjust the number of bins to 100
ggplot(dat, aes(electricity_pop)) +
  geom_histogram(bins = 100)
```



Density Plots

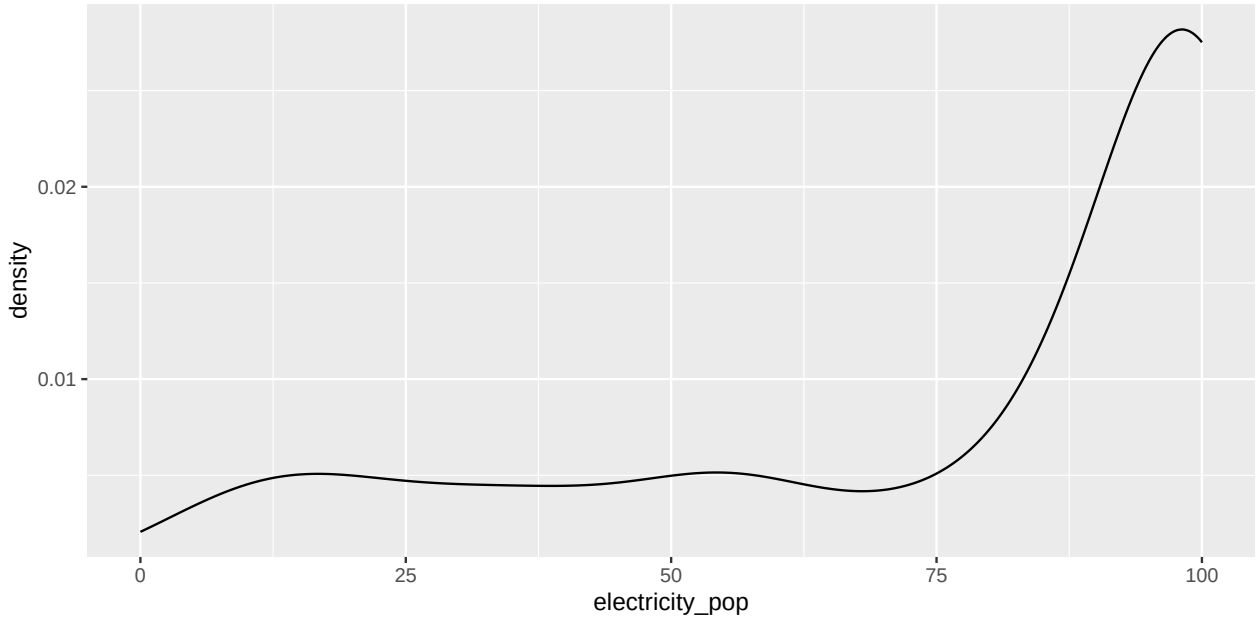
Density plots provide a smooth, bin-free, continuous alternative to histograms by estimating the probability distribution of a continuous variable. We use the `geom_density()` function to create a density plot.

```
# Create a density plot of 'electricity_pop' using geom_density()
ggplot(dat, aes(electricity_pop)) +
  geom_density()
```



If you prefer a line-based density plot without a filled polygon, you can use the `geom_line()` function with the `stat = "density"` parameter.

```
# Create a density plot of 'electricity_pop' using geom_line()
ggplot(dat, aes(x = electricity_pop)) +
  geom_line(stat = "density")
```



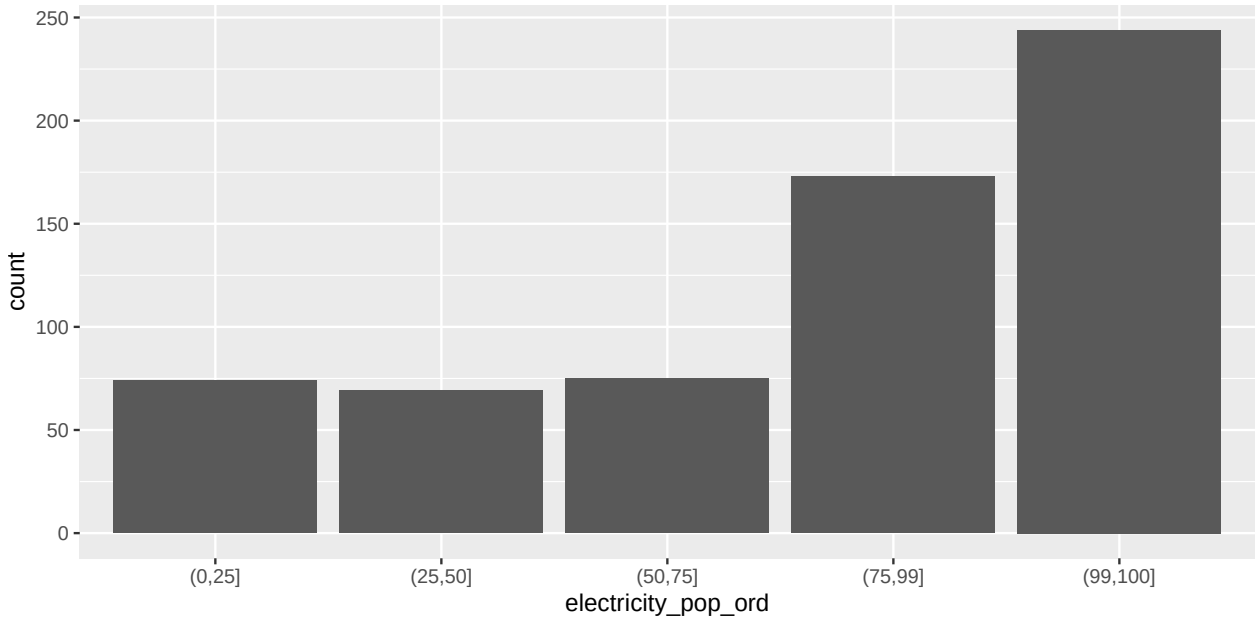
Bar Plots

For categorical variables, bar plots are the best way to visualize the frequency of each category. In this example, we will convert the continuous variable `electricity_pop` into specific categories (0, 25, 50, 75, 99, 100) and create a bar plot to show the frequency of those values in the data.

```
# Transform 'electricity_pop' to an ordinal variable with specific breaks (0, 25, 50, 75, 99, 100)
dat2 <- dat %>%
```

```r
  mutate(electricity_pop_ord = cut(electricity_pop, breaks = c(0,25, 50, 75, 99, 100))) %>%
  filter(electricity_pop_ord != "NA")
  # Remove any instances where 'electricity_pop_ord' is NA to clean up the data before plotting.

# Create a bar plot
ggplot(dat2, aes(x = electricity_pop_ord)) +
  geom_bar()
```



Answers

Question 1: Answer

A histogram displays the distribution of a variable. The x-axis represents the values of the variable, while the y-axis shows the number of observations for each value (or group of values). The width of each bar indicates which values are grouped into a single bin, and the height of the bar represents the number of observations within that bin.

Question 2: Answer

The distribution is not normal (i.e., not a bell curve). It is skewed to the left. There are far more observations at the upper end of the scale than at the lower end, meaning more country-years have high levels of electricity availability compared to low levels.

Question 3: Answer

Changing the number of bins affects the resolution of the histogram. More bins provide a more detailed view, while fewer bins give a broader summary of the data.

Conclusion

In this walkthrough, you were introduced to basic data visualizations in `ggplot2`, including histograms, density plots, and bar plots. These tools enable you to explore and effectively communicate the structure and distribution of both continuous and categorical data.

In Walkthrough 2, we will build on these concepts by exploring advanced styling options and interactive features in `ggplot2`. These skills will help you create publication-ready plots that can be shared and presented with a professional finish.