Bar Plots to Illustrate Medians

James M. Murray, Ph.D.
University of Wisconsin - La Crosse

Updated: April 30, 2018

PDF file location: http://www.murraylax.org/rtutorials/medianplots.pdf

HTML file location: http://www.murraylax.org/rtutorials/medianplots.html

---

Note on required packages: The following code requires the packages `tidyverse`, `scales`, `stringr`, `Hmisc`, and `ggthemes`.

- The `tidyverse` package contains many packages that allow you to organize, summarize, and plot data.
- We use the `scales` library to customize the scales of our axes.
- The `stringr` package allows us to manipulate strings, which we use to manipulate string labels.
- The `Hmisc` provides mathematical and statistical functions to use with our plots.
- The `ggthemes` package provides multiple themes, which are combinations of parameters to change a plots look and feel
- The `psych` packages contains functions to compute medians and interpolated medians.

If you have not already done so, download, install, and load the libraries with the following code:

```
install.packages("tidyverse") # This only needs to be executed once for your machine
install.packages("scales") # This only needs to be executed once for your machine
install.packages("stringr") # This only needs to be executed once for your machine
install.packages("Hmisc") # This only needs to be executed once for your machine
install.packages("ggthemes") # This only needs to be executed once for your machine
install.packages("psych") # This only needs to be executed once for your machine
library("ggplot2") # This needs to be executed every time you load R
library("scales") # This needs to be executed every time you load R
library("stringr") # This needs to be executed every time you load R
library("Hmisc") # This needs to be executed every time you load R
library("ggthemes") # This needs to be executed every time you load R
library("psych") # This needs to be executed every time you load R
```

---

1 Introduction

In this tutorial, we show how to produce bar plots to illustrate interpolated medians of an ordinal variable for multiple categories of some categorical variable.

When illustrating means and their confidence intervals, we can call `stat_summary()` to take advantage of the functions `mean_sdl()` and `mean_cl_boot()`, which are both available in the `Hmisc` package. These functions

return the mean and confidence interval for the mean, respectively. Similar functions are not available for the interpolated median.

The following `source()` call downloads code I developed which replicate the behavior for `mean_sdl()` and `mean_cl_boot()` but for the interpolated median.

```
source("http://murraylax.org/code/R/imedian.R")
```

2 Downloading and Understanding the Data

This tutorial uses data from the National Financial Capability Study is performed by the FINRA (Financial Industry Regulatory Authority) Investor Education Foundation. The study includes a survey on financial behaviors and financial literacy. The dataset below includes observations from 10,000 individuals included in the 2015 survey.

The following command downloads the dataset and stores it in an R data frame called `df`.

```
load(url("http://murraylax.org/datasets/findata.RData"))
```

In this example, we compare the interpolated median income (`Incomecat`) for people with different levels of education (`Edu`). The variable `IncomeCat` is an ordered factor, or an ordinal variable. Rather than numeric values representing dollars, the possible responses are categories.

The code below verifies the scale of measurement with a call to `class()`, examines the category labels with `levels()`, and provides some counts for how many observations are in each category with `table()`:

```
class(df$Incomecat)
```

```
## [1] "ordered" "factor"
```

```
levels(df$Incomecat)
```

```
## [1] "Less than $15,000"
## [2] "At least $15,000 but less than $25,000"
## [3] "At least $25,000 but less than $35,000"
## [4] "At least $35,000 but less than $50,000"
## [5] "At least $50,000 but less than $75,000"
## [6] "At least $75,000 but less than $100,000"
## [7] "At least $100,000 but less than $150,000"
## [8] "$150,000 or more"
```

```
table(df$Incomecat)
```

```
##
##                         Less than $15,000
##                                      1278
##   At least $15,000 but less than $25,000
##                                      1151
##   At least $25,000 but less than $35,000
##                                      1104
##   At least $35,000 but less than $50,000
##                                      1497
##   At least $50,000 but less than $75,000
##                                      1992
##  At least $75,000 but less than $100,000
##                                      1282
## At least $100,000 but less than $150,000
##                                      1135
```

```
##                          $150,000 or more
##                                       561
```

3 Creating the Bar Plot

Because `Incomecat` is an ordinal variable, we cannot compute a mean. However, because ordinal variables can be ordered, we can compute medians and interpolated medians. Unfortunately, `R` returns an error if we try to compute either of these with a non-numeric variable. Instead, we must first convert the value to a numeric:

```
median(as.numeric(df$Incomecat))
```

```
## [1] 4
```

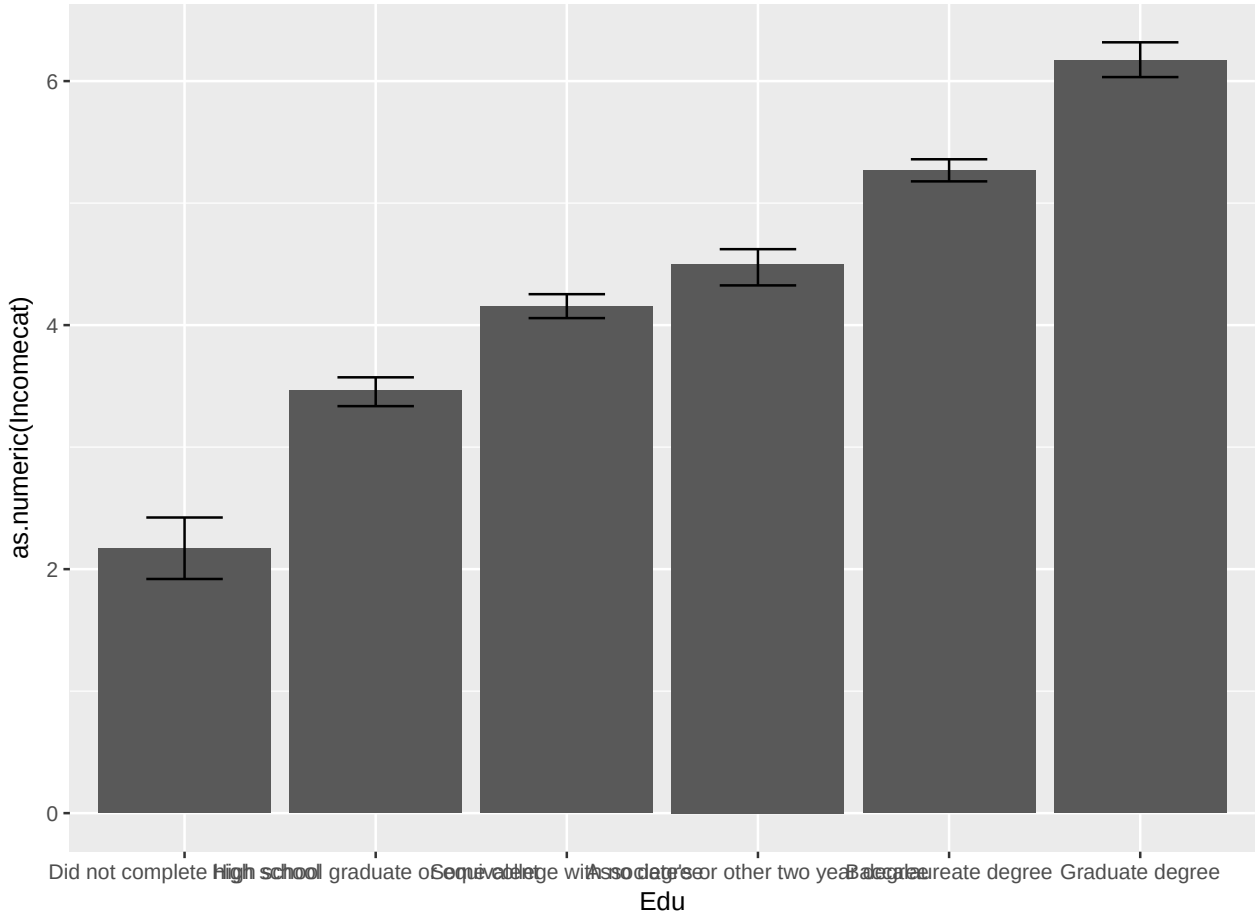
```
interp.median(as.numeric(df$Incomecat))
```

```
## [1] 4.47996
```

A median equal to 4 implies the median is the 4th level, or "At least $35,000 but less than $50,000"

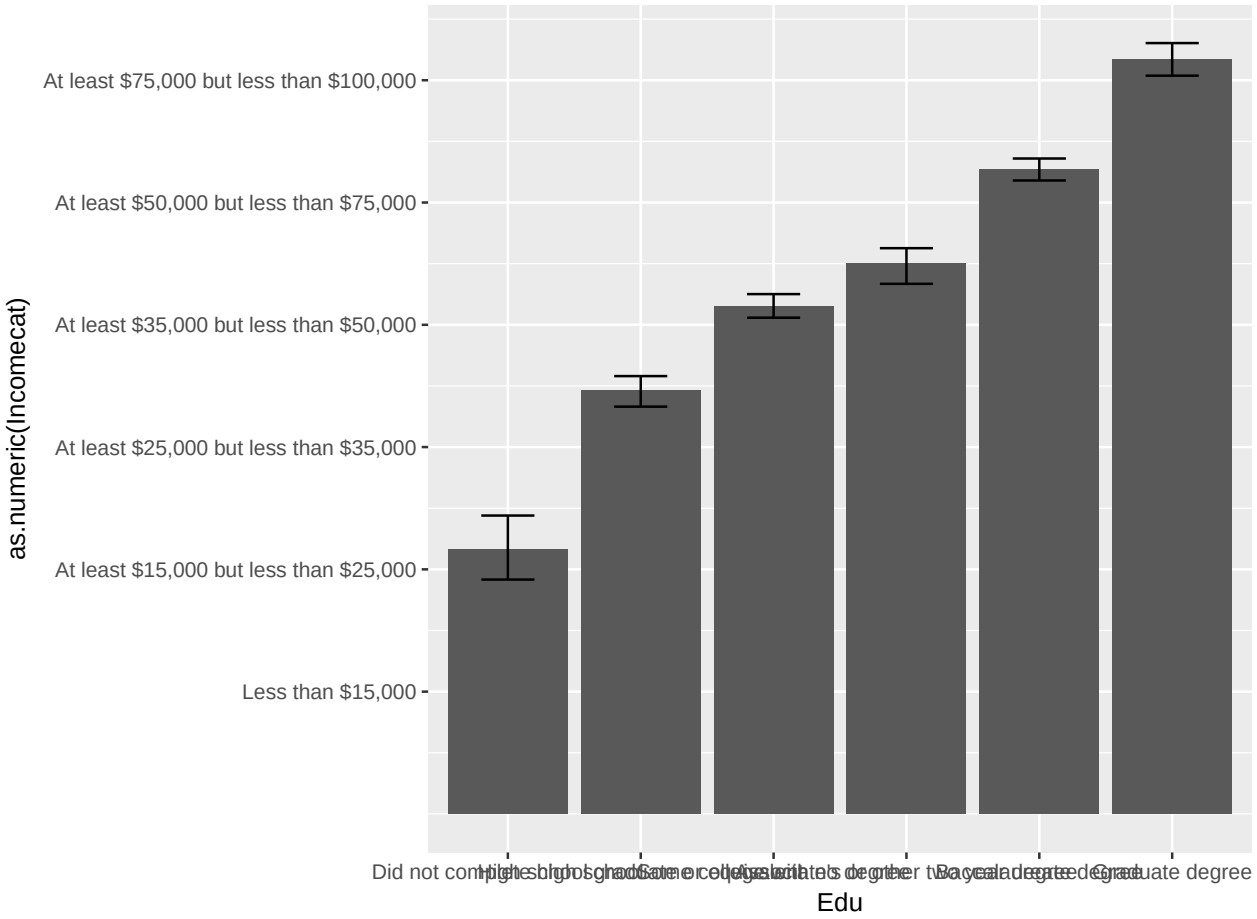
We can use the functions `imedian_sdl()` and `imedian_cl_boot()` from the `imedian.R` file we included above to compute the interpolated median and bootstrapped confidence interval for the interpolated and include these in a plot. Again, we have to coerce `Incomecat` to numeric for the procedure to work.

```
ggplot(df, aes(x=Edu, y=as.numeric(Incomecat))) +
  stat_summary(fun.data=imedian_sdl, geom="bar") +
  stat_summary(fun.data=imedian_cl_boot, geom="errorbar", width=0.4)
```



The vertical scale currently shows values 0-6 which is not too informative. The scale is the levels for income category, for which there are 7, and which have the descriptive labels above that give income ranges. The code below alters the vertical scale to include the appropriate descriptions for the vertical scale. It uses the `scale_y_orderedfactor()` function, which is also included in `imedian.R`. We pass to this function the levels we wish to use for the labels.

```
ggplot(df, aes(x=Edu, y=as.numeric(Incomecat))) +
  stat_summary(fun.data=imedian_sdl, geom="bar") +
  stat_summary(fun.data=imedian_cl_boot, geom="errorbar", width=0.4) +
  scale_y_orderedfactor( levels(df$Incomecat) )
```



All that remains is to pretty it up. Below I break apart the labels so they wrap to new lines (two lines that call `str_wrap()`), add color to the bars (`fill="darkgreen"`), give the bars some transparency (`alpha=0.7`), give it a title and clean up the x- and y-axis labels (call to `labs()`), give it a white background (`theme_bw()`), remove the vertical lines in the grid which are not necessary for bar charts (associated with the calls to `element_blank()`), and thicken the horizontal lines up a bit (associated with the call to `element_line()`).

```
levels(df$Edu) <- str_wrap(levels(df$Edu), 12)
levels(df$Incomecat) <- str_wrap(levels(df$Incomecat), 12)

ggplot(df, aes(x=Edu, y=as.numeric(Incomecat))) +
  stat_summary(fun.data=imedian_sdl, geom="bar", fill="darkgreen", alpha=0.7) +
  stat_summary(fun.data=imedian_cl_boot, geom="errorbar", width=0.4) +
  scale_y_orderedfactor(df$Incomecat) +
  labs(title="Income by Education Level", x="", y="") +
  theme_bw() +
```

```
  theme(panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
        panel.grid.major.y = element_line(size=0.75))
```