Analysis on World_Bank_Pop

Aung Thura Htoo

2024-03-12

```
options(repos = c(CRAN = "https://cran.rstudio.com/"))
install.packages("tidyverse")
```

```
## Installing package into 'C:/Users/lenovo/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)
```

```
## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\lenovo\AppData\Local\Temp\Rtmpe2a8dG\downloaded_packages
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

options(repos = c(CRAN = "https://cran.rstudio.com/")) is a useful command when you prefer to use a specific CRAN mirror in RStudio CRAN mirror in package installions. This can be handy in knitting R Markdown to HTML.

Installing tidyverse can give me access to many datasets including world_bank_pop.

```
(.packages())
```

```
##  [1] "lubridate" "forcats"   "stringr"   "dplyr"     "purrr"     "readr"
##  [7] "tidyr"     "tibble"    "ggplot2"   "tidyverse" "stats"     "graphics"
## [13] "grDevices" "utils"     "datasets"  "methods"   "base"
```

This command (.packages()) is a base function in R that lists the names of the attached packages in your current R session. When you load a package in R using library() or require(), it becomes attached and ready to use.

```
my_data<- world_bank_pop
head(my_data,6)
```

```
## # A tibble: 6 x 20
##   country indicator       `2000`  `2001`  `2002`  `2003`  `2004`  `2005`  `2006`
##   <chr>   <chr>            <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 ABW     SP.URB.TOTL     4.16e4 4.20e+4 4.22e+4 4.23e+4 4.23e+4 4.24e+4 4.26e+4
## 2 ABW     SP.URB.GROW     1.66e0 9.56e-1 4.01e-1 1.97e-1 9.46e-2 1.94e-1 3.67e-1
## 3 ABW     SP.POP.TOTL     8.91e4 9.07e+4 9.18e+4 9.27e+4 9.35e+4 9.45e+4 9.56e+4
## 4 ABW     SP.POP.GROW     2.54e0 1.77e+0 1.19e+0 9.97e-1 9.01e-1 1.00e+0 1.18e+0
## 5 AFE     SP.URB.TOTL     1.16e8 1.20e+8 1.24e+8 1.29e+8 1.34e+8 1.39e+8 1.44e+8
## 6 AFE     SP.URB.GROW     3.60e0 3.66e+0 3.72e+0 3.71e+0 3.74e+0 3.81e+0 3.81e+0
## # i 11 more variables: `2007` <dbl>, `2008` <dbl>, `2009` <dbl>, `2010` <dbl>,
## #   `2011` <dbl>, `2012` <dbl>, `2013` <dbl>, `2014` <dbl>, `2015` <dbl>,
## #   `2016` <dbl>, `2017` <dbl>
```

```
str(my_data)
```

```
## tibble [1,064 x 20] (S3: tbl_df/tbl/data.frame)
##  $ country  : chr [1:1064] "ABW" "ABW" "ABW" "ABW" ...
##  $ indicator: chr [1:1064] "SP.URB.TOTL" "SP.URB.GROW" "SP.POP.TOTL" "SP.POP.GROW" ...
##  $ 2000     : num [1:1064] 4.16e+04 1.66 8.91e+04 2.54 1.16e+08 ...
##  $ 2001     : num [1:1064] 4.20e+04 9.56e-01 9.07e+04 1.77 1.20e+08 ...
##  $ 2002     : num [1:1064] 4.22e+04 4.01e-01 9.18e+04 1.19 1.24e+08 ...
##  $ 2003     : num [1:1064] 4.23e+04 1.97e-01 9.27e+04 9.97e-01 1.29e+08 ...
##  $ 2004     : num [1:1064] 4.23e+04 9.46e-02 9.35e+04 9.01e-01 1.34e+08 ...
##  $ 2005     : num [1:1064] 4.24e+04 1.94e-01 9.45e+04 1.00 1.39e+08 ...
##  $ 2006     : num [1:1064] 4.26e+04 3.67e-01 9.56e+04 1.18 1.44e+08 ...
##  $ 2007     : num [1:1064] 4.27e+04 4.08e-01 9.68e+04 1.23 1.49e+08 ...
##  $ 2008     : num [1:1064] 4.29e+04 4.13e-01 9.80e+04 1.24 1.55e+08 ...
##  $ 2009     : num [1:1064] 4.31e+04 4.02e-01 9.92e+04 1.23 1.62e+08 ...
##  $ 2010     : num [1:1064] 4.32e+04 2.94e-01 1.00e+05 1.13 1.68e+08 ...
##  $ 2011     : num [1:1064] 4.35e+04 6.62e-01 1.01e+05 9.39e-01 1.75e+08 ...
##  $ 2012     : num [1:1064] 4.39e+04 8.49e-01 1.02e+05 8.10e-01 1.83e+08 ...
##  $ 2013     : num [1:1064] 4.42e+04 8.26e-01 1.03e+05 7.49e-01 1.90e+08 ...
##  $ 2014     : num [1:1064] 4.46e+04 8.11e-01 1.04e+05 6.92e-01 1.98e+08 ...
##  $ 2015     : num [1:1064] 4.49e+04 7.93e-01 1.04e+05 6.38e-01 2.07e+08 ...
##  $ 2016     : num [1:1064] 4.53e+04 7.85e-01 1.05e+05 5.90e-01 2.15e+08 ...
##  $ 2017     : num [1:1064] 4.56e+04 7.72e-01 1.05e+05 5.37e-01 2.24e+08 ...
```

head() and str() are useful commands in R to have a better understanding on your dataset. head() shows the first few rows of your dataframe while str() displays the structure of your dataframe or R object.

The data in world_bank_pop will be easier to analyze if we convert the columns from 2000 to 2017 to rows.

```
my_data2 <- my_data %>% pivot_longer(cols = -c(country, indicator), names_to = "Year", values_to = "Valu
head(my_data2)
```

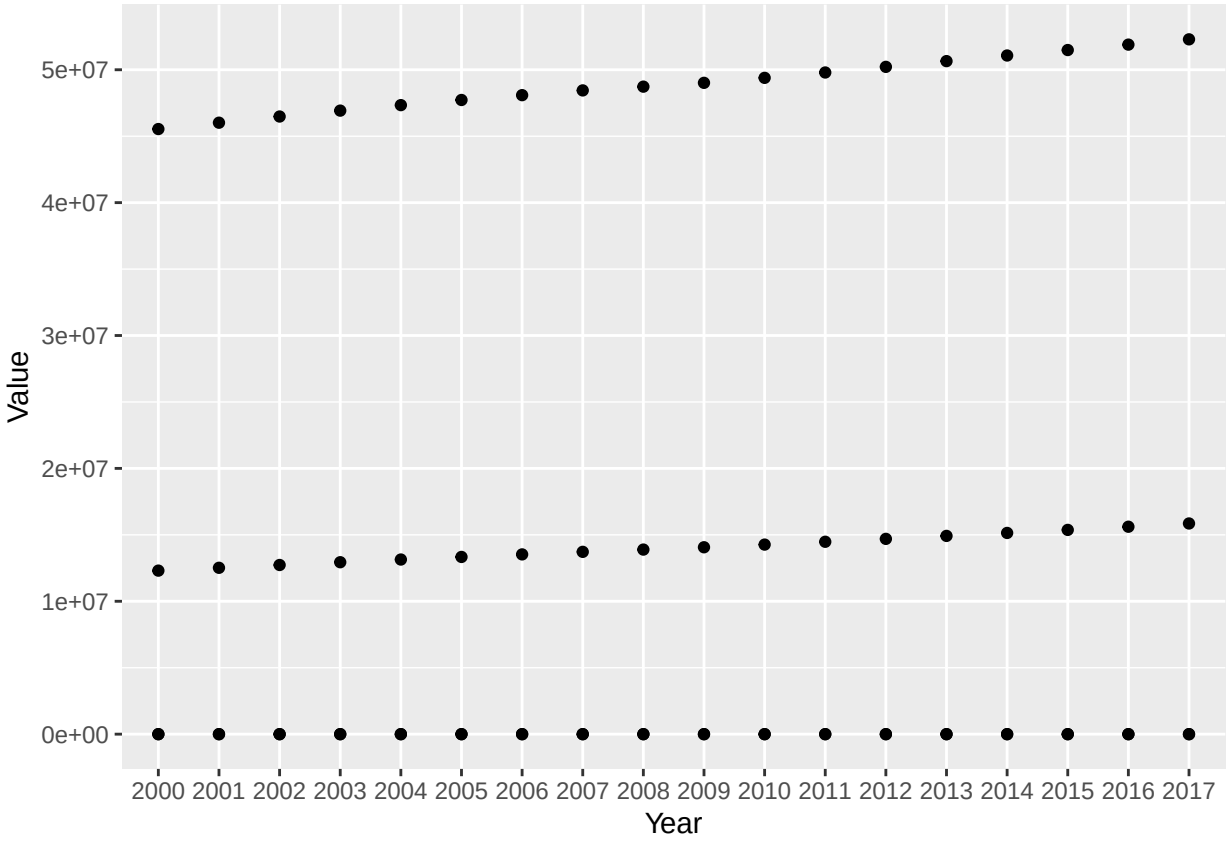
```
## # A tibble: 6 x 4
##   country indicator   Year  Value
##   <chr>   <chr>       <chr> <dbl>
## 1 ABW     SP.URB.TOTL 2000  41625
## 2 ABW     SP.URB.TOTL 2001  42025
## 3 ABW     SP.URB.TOTL 2002  42194
## 4 ABW     SP.URB.TOTL 2003  42277
## 5 ABW     SP.URB.TOTL 2004  42317
## 6 ABW     SP.URB.TOTL 2005  42399
```

```
str(my_data2)
```

```
## tibble [19,152 x 4] (S3: tbl_df/tbl/data.frame)
##  $ country  : chr [1:19152] "ABW" "ABW" "ABW" "ABW" ...
##  $ indicator: chr [1:19152] "SP.URB.TOTL" "SP.URB.TOTL" "SP.URB.TOTL" "SP.URB.TOTL" ...
##  $ Year     : chr [1:19152] "2000" "2001" "2002" "2003" ...
##  $ Value    : num [1:19152] 41625 42025 42194 42277 42317 ...
```

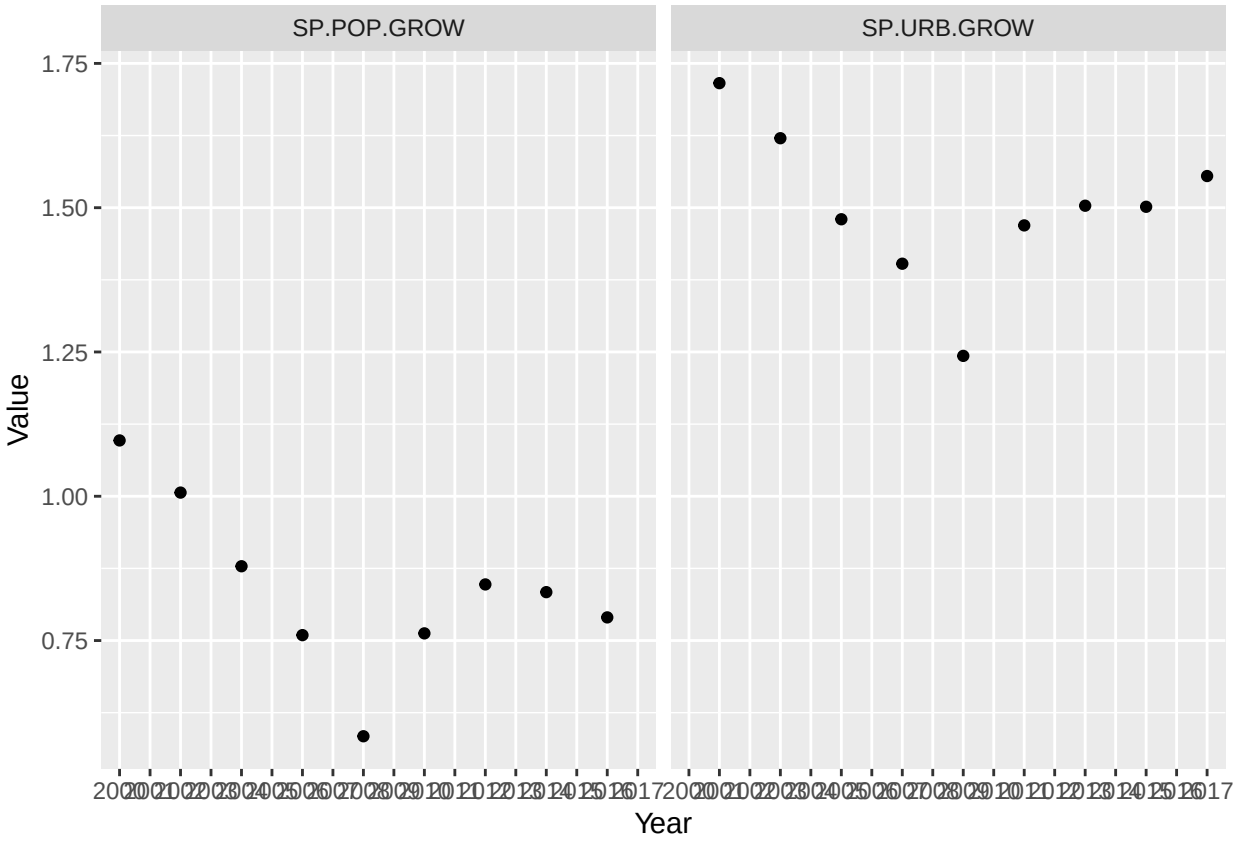
pivot_longer() is a command in tidyr, which reshapes the data from wide to long format. cols= -c(country,indicator) excludes the the two specific columns and all other columns are reshaped. names_to = "Year" specifies the name of the column that has the variable names, while values_to="Value" specifies the name of the column that has the values after pivoting.

```
my_data2 %>%
  filter(country == "MMR") %>%
  ggplot(aes(Year, Value))+
  geom_point()
```



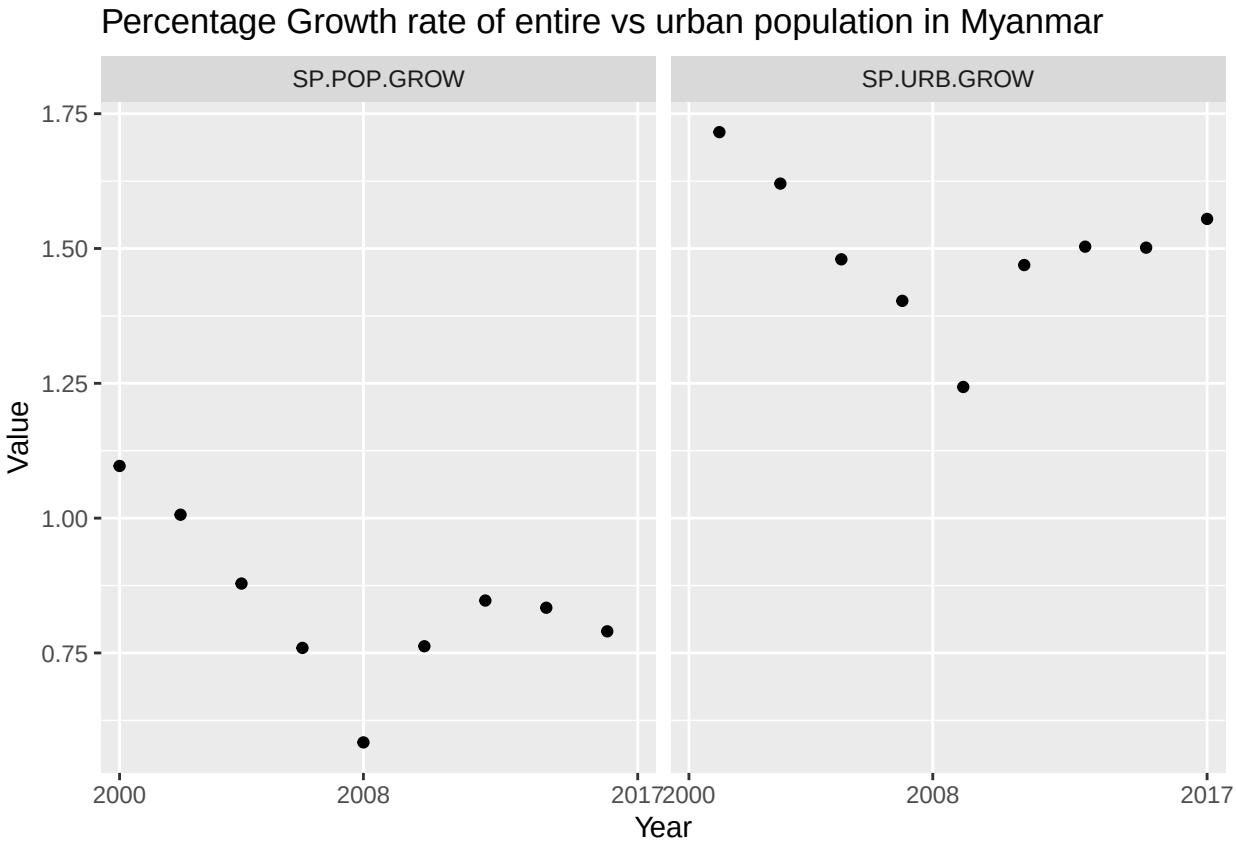
I have filtered the country "Myanmar" - code name "MMR", and plotted the year on the x-axis and values of population on y-axis. You can see three different dotted lines that represent different values by indicator. The thing that went wrong for two indicators namely SP.POP.GROW and SP.URB.GROW is that we plotted against very high numbers of SP.POP.TOTL and SP.URB.TOTL.

```
my_data2 %>%
  filter(country == "MMR", indicator == c("SP.POP.GROW", "SP.URB.GROW")) %>%
  ggplot(aes(Year, Value))+
  geom_point()+
  facet_wrap(~indicator)
```



By applying filter on two proportion value and facet_wrap() function, we can clearly see the scatter plot for the population in each year by each indicator type. But the labels on x-axis are overlapped.

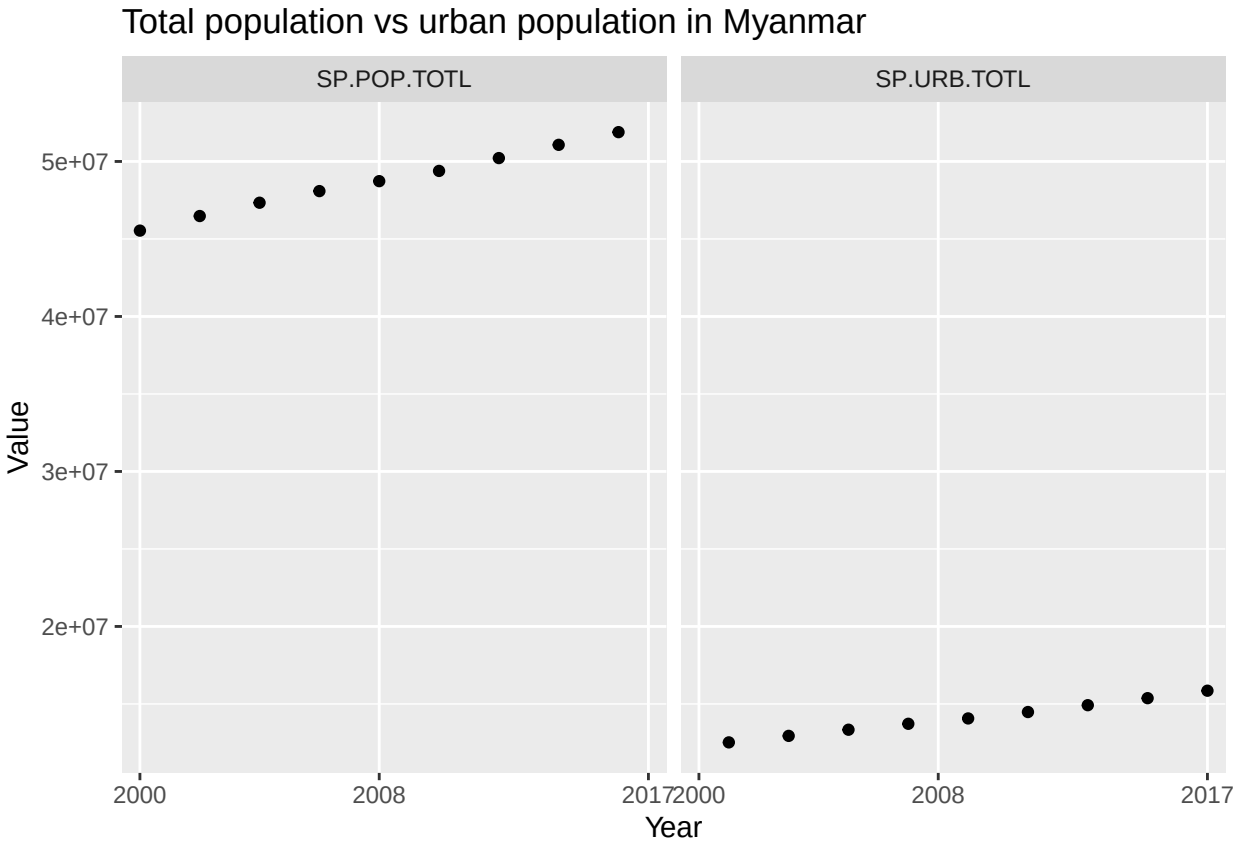
```
my_data2 %>%
  filter(country == "MMR", indicator == c("SP.POP.GROW", "SP.URB.GROW")) %>%
  ggplot(aes(Year, Value))+
  geom_point()+
  facet_wrap(~indicator)+
  scale_x_discrete(breaks = function(x) c(first(x), last(x), x[length(x) %/% 2]))+
  labs(title = "Percentage Growth rate of entire vs urban population in Myanmar")
```



By adding the scale_x_discrete(), we cleaned up the labels on x axis, by showing only the first, middle, and last labels. Note: since the labels are discrete, we use scale_x_discrete() instead of scale_x_continuous().

Let's go back to the population value SP.POP.TOTL and SP.URB.TOTL.

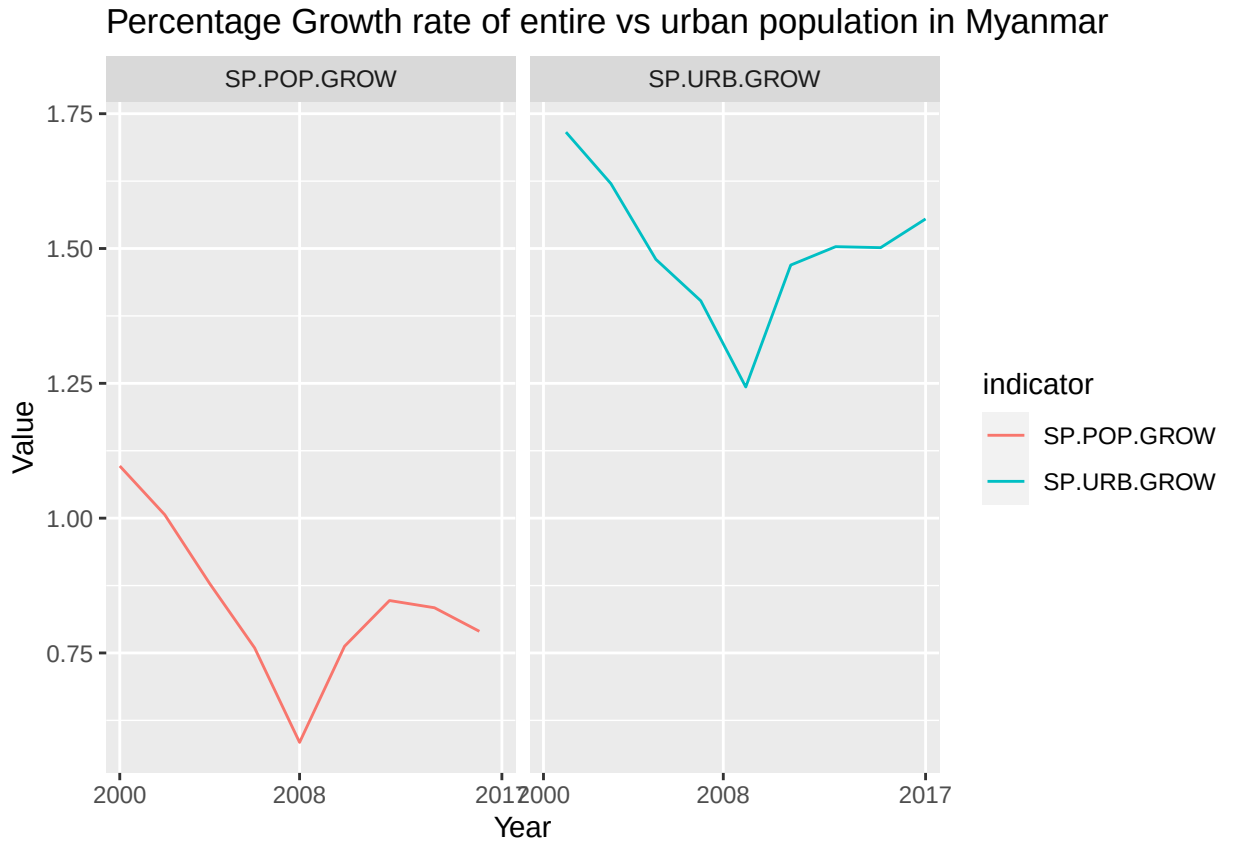
```
my_data2 %>%
  filter(country == "MMR", indicator == c("SP.POP.TOTL", "SP.URB.TOTL")) %>%
  ggplot(aes(Year, Value))+
  geom_point()+
  facet_wrap(~indicator)+
  scale_x_discrete(breaks = function(x) c(first(x), last(x), x[length(x) %/% 2]))+
  labs(title = "Total population vs urban population in Myanmar")
```



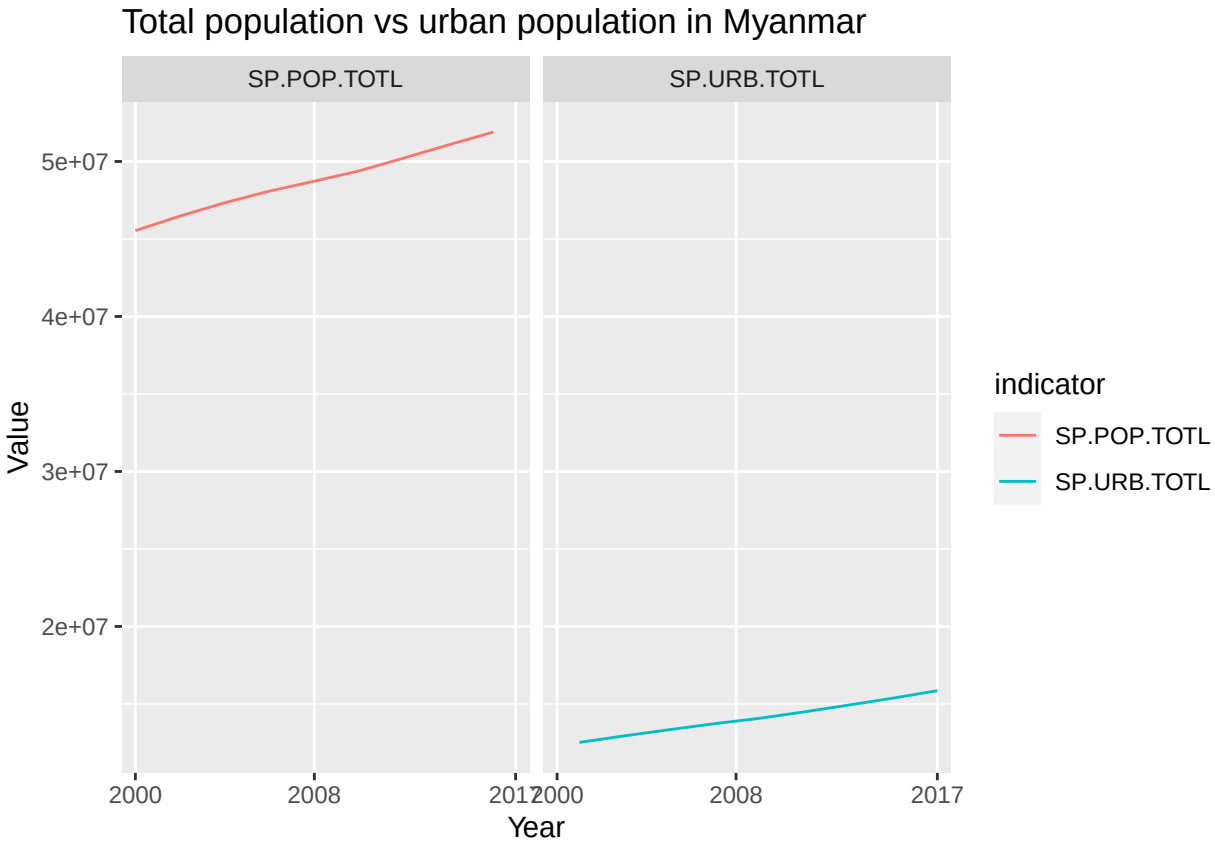
These graphs look great, let's try with line graphs.

By adding the group and color aesthetic in ggplot()function, different colors are denoted to values in each indicator graph.

```
my_data2 %>%
  filter(country == "MMR", indicator == c("SP.POP.GROW", "SP.URB.GROW")) %>%
  ggplot(aes(Year, Value, group = indicator, color = indicator)) +
  geom_line() +
  facet_wrap(~indicator) +
  scale_x_discrete(breaks = function(x) c(first(x), last(x), x[length(x) %/% 2]))+
  labs(title = "Percentage Growth rate of entire vs urban population in Myanmar")
```




```
my_data2 %>%
  filter(country == "MMR", indicator == c("SP.POP.TOTL", "SP.URB.TOTL")) %>%
  ggplot(aes(Year, Value, group = indicator, color = indicator)) +
  geom_line() +
  facet_wrap(~indicator) +
  scale_x_discrete(breaks = function(x) c(first(x), last(x), x[length(x) %/% 2]))+
  labs(title = "Total population vs urban population in Myanmar")
```


Total population vs urban population in Myanmar
SP.POP.TOTL
SP.URB.TOTL
5e+07
4e+07
3e+07
2e+07
Value
2000
2008
2017
2000
2008
2017
Year
indicator
SP.POP.TOTL
SP.URB.TOTL