Tidy Data and Analysis using R Rstudio

Bernardo Vimpi

10/04/2020

In this project, we use another important data science/analysis technique and convert our nontidy dataset to tidy.

Packages

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(readr)
library(tidyr)
library(nycflights13)
library(fivethirtyeight)
library(tinytex)
```

(LC4.1) What are common characteristics of "tidy" data frames?

Answer: The common characteristics of "tidy" data frames are: the rows, tables and columns are matched up with observations, variables and types. Also, each variable forms a column, each observation forms a row and each type of observational unit forms a table. Also data frames that are "tidy" have a particular format of "long/narrow" while data frames that are "not tidy" are are "wide".

(LC4.2) What makes "tidy" data frames useful for organizing data?

Answer Having "tidy" data frames makes it useful for organizing data because it enables us to map a dataset to its structure and it makes it much easier for us to visualize the data frame, especially when using packages such as *ggplot2* and *dplyr*. By having a "tidy" data frame, we are also able to plot the data and display any relationship among or between the variables. Also, each variable will have its own corresponding

column which is needed for further analysis of the data. Also, observations that correspond to the same observational units should be saved in the same table or data frame.

(LC4.3) Take a look the *airline_safety* data frame included in the *fivethirtyeight* data package. Run the following:

```
airline_safety
```

```
## # A tibble: 56 x 9
##    airline incl_reg_subsid~ avail_seat_km_p~ incidents_85_99 fatal_accidents~
##    <chr>   <lgl>                       <dbl>           <int>            <int>
##  1 Aer Li~ FALSE                   320906734               2                0
##  2 Aerofl~ TRUE                   1197672318              76               14
##  3 Aeroli~ FALSE                   385803648               6                0
##  4 Aerome~ TRUE                    596871813               3                1
##  5 Air Ca~ FALSE                  1865253802               2                0
##  6 Air Fr~ FALSE                  3004002661              14                4
##  7 Air In~ TRUE                    869253552               2                1
##  8 Air Ne~ TRUE                    710174817               3                0
##  9 Alaska~ TRUE                    965346773               5                0
## 10 Alital~ FALSE                   698012498               7                2
## # ... with 46 more rows, and 4 more variables: fatalities_85_99 <int>,
## #   incidents_00_14 <int>, fatal_accidents_00_14 <int>, fatalities_00_14 <int>
```

After reading the help file by running ?airline_safety, we see that airline_safety is a data frame containing information on different airlines companies' safety records. This data was originally reported on the data journalism website FiveThirtyEight.com in Nate Silver's article "Should Travelers Avoid Flying Airlines That Have Had Crashes in the Past?". Let's ignore the incl_reg_subsidiaries and avail_seat_km_per_week variables for simplicity:

```
airline_safety_smaller <- airline_safety%>%
  select(-c(incl_reg_subsidiaries, avail_seat_km_per_week))
airline_safety_smaller
```

```
## # A tibble: 56 x 7
##    airline incidents_85_99 fatal_accidents~ fatalities_85_99 incidents_00_14
##    <chr>             <int>            <int>            <int>           <int>
##  1 Aer Li~               2                0                0               0
##  2 Aerofl~              76               14              128               6
##  3 Aeroli~               6                0                0               1
##  4 Aerome~               3                1               64               5
##  5 Air Ca~               2                0                0               2
##  6 Air Fr~              14                4               79               6
##  7 Air In~               2                1              329               4
##  8 Air Ne~               3                0                0               5
##  9 Alaska~               5                0                0               5
## 10 Alital~               7                2               50               4
## # ... with 46 more rows, and 2 more variables: fatal_accidents_00_14 <int>,
## #   fatalities_00_14 <int>
```

This data frame is not in "tidy" format. How would you convert this data frame to be in "tidy" format, in particular so that it has a variable incident_type_years indicating the incident type/year and a variable count of the counts?

Answer: To convert this data frame from not "tidy" to "tidy" we use the *gather()* function as follow:

```
airline_safety_smaller_tidy <-airline_safety_smaller%>%
  gather( key= incident_type_years, value = count, -airline)
airline_safety_smaller_tidy
```

```
## # A tibble: 336 x 3
##    airline               incident_type_years count
##    <chr>                 <chr>               <int>
##  1 Aer Lingus            incidents_85_99         2
##  2 Aeroflot              incidents_85_99        76
##  3 Aerolineas Argentinas incidents_85_99         6
##  4 Aeromexico            incidents_85_99         3
##  5 Air Canada            incidents_85_99         2
##  6 Air France            incidents_85_99        14
##  7 Air India             incidents_85_99         2
##  8 Air New Zealand       incidents_85_99         3
##  9 Alaska Airlines       incidents_85_99         5
## 10 Alitalia              incidents_85_99         7
## # ... with 326 more rows
```

(LC4.4) Convert the *dem_score* data frame into a tidy data frame and assign the name of *democracy_tidy* to the resulting long-formatted data frame.

Answer Lets do this in steps:

Step 1: Let's import the Democracy Score dataset and save it in *demo_score* data frame.

```
dem_score <- read_csv("https://moderndive.com/data/dem_score.csv")
```

```
## Parsed with column specification:
## cols(
##   country = col_character(),
##   `1952` = col_double(),
##   `1957` = col_double(),
##   `1962` = col_double(),
##   `1967` = col_double(),
##   `1972` = col_double(),
##   `1977` = col_double(),
##   `1982` = col_double(),
##   `1987` = col_double(),
##   `1992` = col_double()
## )
```

```
dem_score
```

```
## # A tibble: 96 x 10
##    country    `1952` `1957` `1962` `1967` `1972` `1977` `1982` `1987` `1992`
##    <chr>       <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
##  1 Albania        -9     -9     -9     -9     -9     -9     -9     -9      5
##  2 Argentina      -9     -1     -1     -9     -9     -9     -8      8      7
##  3 Armenia        -9     -7     -7     -7     -7     -7     -7     -7      7
##  4 Australia      10     10     10     10     10     10     10     10     10
##  5 Austria        10     10     10     10     10     10     10     10     10
```

```
##  6 Azerbaijan     -9     -7     -7     -7     -7     -7     -7     -7      1
##  7 Belarus        -9     -7     -7     -7     -7     -7     -7     -7      7
##  8 Belgium        10     10     10     10     10     10     10     10     10
##  9 Bhutan        -10    -10    -10    -10    -10    -10    -10    -10    -10
## 10 Bolivia        -4     -3     -3     -4     -7     -7      8      9      9
## # ... with 86 more rows
```

Step 2: Since this data is not "tidy" we'll create a new "tidy" data frame *dem_score_tidy* with "year" as the key and "democracy_score" as the value. We'll not "tidy the"country" variable.

```
dem_score_tidy <- dem_score%>%
  gather(key= year, value = democracy_score, -country)
dem_score_tidy
```

```
## # A tibble: 864 x 3
##    country    year  democracy_score
##    <chr>      <chr>           <dbl>
##  1 Albania    1952               -9
##  2 Argentina  1952               -9
##  3 Armenia    1952               -9
##  4 Australia  1952               10
##  5 Austria    1952               10
##  6 Azerbaijan 1952               -9
##  7 Belarus    1952               -9
##  8 Belgium    1952               10
##  9 Bhutan     1952              -10
## 10 Bolivia    1952               -4
## # ... with 854 more rows
```

Step 3: Since the "year" variable has a column type of "character". For plotting and further analysis, we need to use the *mutate()* function to convert "year" to numeric as follows:

```
dem_score_tidy <- dem_score_tidy%>%
  mutate( year = as.numeric(year))
dem_score_tidy
```

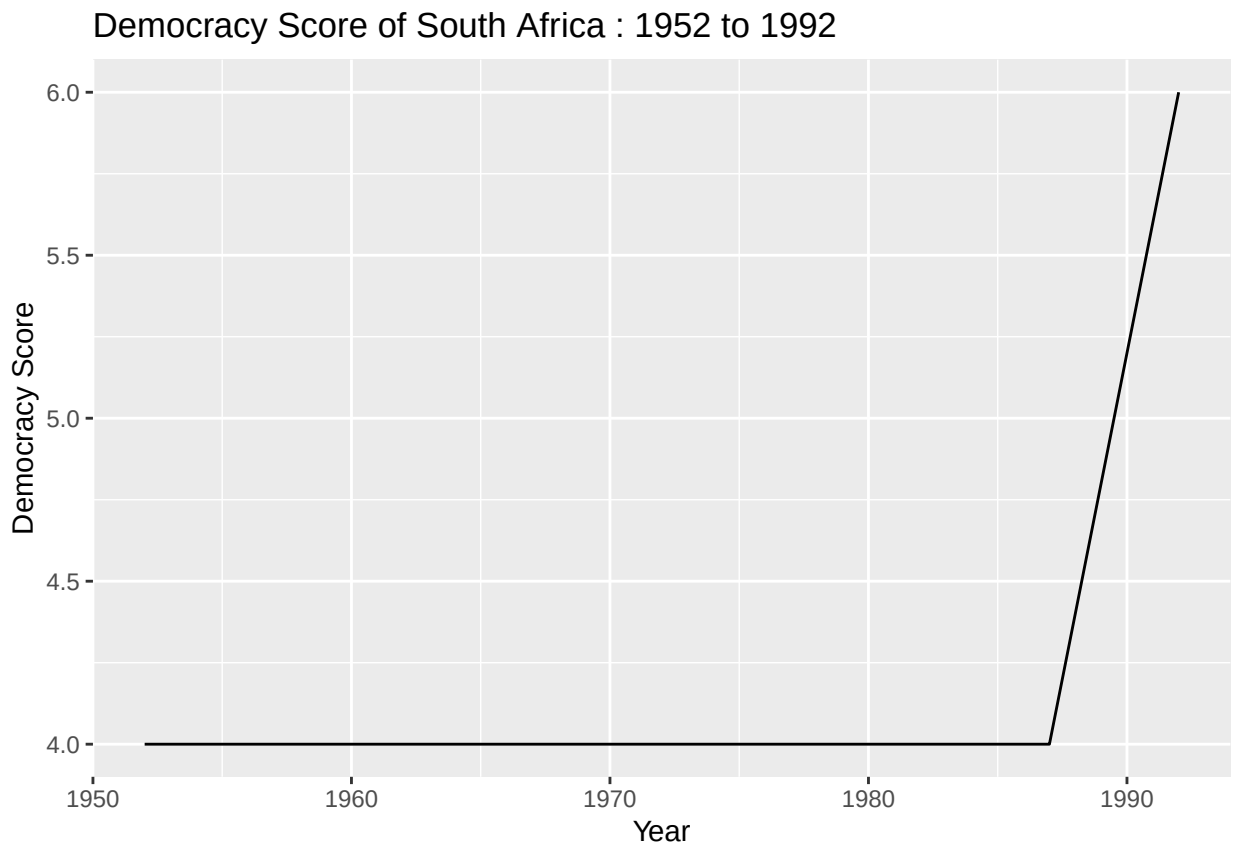
```
## # A tibble: 864 x 3
##    country     year democracy_score
##    <chr>      <dbl>           <dbl>
##  1 Albania     1952              -9
##  2 Argentina   1952              -9
##  3 Armenia     1952              -9
##  4 Australia   1952              10
##  5 Austria     1952              10
##  6 Azerbaijan  1952              -9
##  7 Belarus     1952              -9
##  8 Belgium     1952              10
##  9 Bhutan      1952             -10
## 10 Bolivia     1952              -4
## # ... with 854 more rows
```

***Step 4**: Now that the "year" is of a type numeric, let's filter one country and visualize the data via time series. Let's plot a time series of democracy score for South Africa.

```
dem_score_tidy_SA <-dem_score_tidy%>%
  filter(country == "South Africa")
dem_score_tidy_SA
```

```
## # A tibble: 9 x 3
##   country       year democracy_score
##   <chr>        <dbl>           <dbl>
## 1 South Africa  1952               4
## 2 South Africa  1957               4
## 3 South Africa  1962               4
## 4 South Africa  1967               4
## 5 South Africa  1972               4
## 6 South Africa  1977               4
## 7 South Africa  1982               4
## 8 South Africa  1987               4
## 9 South Africa  1992               6
```

```
ggplot(dem_score_tidy_SA, aes( x= year, y= democracy_score)) + geom_line() +
  labs( x= "Year", y= "Democracy Score") +
  ggtitle("Democracy Score of South Africa : 1952 to 1992")
```



Analysis of our results:

Since the democracy score ranges from -10 to 10 with -10 corresponding to authoritarian or autocratic governments and 10 to democratic government, based on the existing data we see that South Africa had a

score of 4 from around 1952 to 1991. This is also during the time of Apartheid regime in South Africa. This lasted for about 50 years. However in 1989 President Klerk was elected as the new South African President. In 1990 President Klerk released Nelson Mandala from prison(after 27 years of incarceration). In 1991 President Klerk started making political/legislative reforms to repeal apartheid. This resulted in the boost of the South African democracy score from 4 to 6 starting in 1992, as we also notice in the time series graph. Two years later, on May 10th 1994, Nelson Mandela became the first black democratically elected President of South Africa.

(LC4.5) Read in the life expectancy data stored at https://moderndive.com/data/le_mess.csv and convert it to a tidy data frame.

Step 1: Let's import the csv from the URL using the _read_csv()___ function.

```
life_exp_untidy <- read_csv("https://moderndive.com/data/le_mess.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   country = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
life_exp_untidy
```

```
## # A tibble: 202 x 67
##    country `1951` `1952` `1953` `1954` `1955` `1956` `1957` `1958` `1959` `1960`
##    <chr>    <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
##  1 Afghan~   27.1   27.7   28.2   28.7   29.3   29.8   30.3   30.9   31.4   31.9
##  2 Albania   54.7   55.2   55.8   56.6   57.4   58.4   59.5   60.6   61.8   62.9
##  3 Algeria   43.0   43.5   44.0   44.4   44.9   45.4   45.9   46.4   47.0   47.5
##  4 Angola    31.0   31.6   32.1   32.7   33.2   33.8   34.3   34.9   35.4   36.0
##  5 Antigu~   58.3   58.8   59.3   59.9   60.4   60.9   61.4   62.0   62.5   63.0
##  6 Argent~   61.9   62.5   63.1   63.6   64.0   64.4   64.7   65     65.2   65.4
##  7 Armenia   62.7   63.1   63.6   64.1   64.5   65     65.4   65.9   66.4   66.9
##  8 Aruba     59.0   60.0   61.0   61.9   62.7   63.4   64.1   64.7   65.2   65.7
##  9 Austra~   68.7   69.1   69.7   69.8   70.2   70.0   70.3   70.9   70.4   70.9
## 10 Austria   65.2   66.8   67.3   67.3   67.6   67.7   67.5   68.5   68.4   68.8
## # ... with 192 more rows, and 56 more variables: `1961` <dbl>, `1962` <dbl>,
## #   `1963` <dbl>, `1964` <dbl>, `1965` <dbl>, `1966` <dbl>, `1967` <dbl>,
## #   `1968` <dbl>, `1969` <dbl>, `1970` <dbl>, `1971` <dbl>, `1972` <dbl>,
## #   `1973` <dbl>, `1974` <dbl>, `1975` <dbl>, `1976` <dbl>, `1977` <dbl>,
## #   `1978` <dbl>, `1979` <dbl>, `1980` <dbl>, `1981` <dbl>, `1982` <dbl>,
## #   `1983` <dbl>, `1984` <dbl>, `1985` <dbl>, `1986` <dbl>, `1987` <dbl>,
## #   `1988` <dbl>, `1989` <dbl>, `1990` <dbl>, `1991` <dbl>, `1992` <dbl>,
## #   `1993` <dbl>, `1994` <dbl>, `1995` <dbl>, `1996` <dbl>, `1997` <dbl>,
## #   `1998` <dbl>, `1999` <dbl>, `2000` <dbl>, `2001` <dbl>, `2002` <dbl>,
## #   `2003` <dbl>, `2004` <dbl>, `2005` <dbl>, `2006` <dbl>, `2007` <dbl>,
## #   `2008` <dbl>, `2009` <dbl>, `2010` <dbl>, `2011` <dbl>, `2012` <dbl>,
## #   `2013` <dbl>, `2014` <dbl>, `2015` <dbl>, `2016` <dbl>
```

Step 2: Let's convert the "untidy" data frame to a "tidy" data frame and store it in "life_exp_tidy" using "year" as the key and "life_expectancy" as value while the "country" column will remain untouched.In the same query we'll convert the year variable from character to numeric as well.

```
life_exp_tidy <- life_exp_untidy%>%
  gather(key = year, value = life_expectancy, -country)%>%
  mutate(year = as.numeric(year))
life_exp_tidy
```

```
## # A tibble: 13,332 x 3
##    country              year life_expectancy
##    <chr>               <dbl>           <dbl>
##  1 Afghanistan          1951            27.1
##  2 Albania              1951            54.7
##  3 Algeria              1951            43.0
##  4 Angola               1951            31.0
##  5 Antigua and Barbuda  1951            58.3
##  6 Argentina            1951            61.9
##  7 Armenia              1951            62.7
##  8 Aruba                1951            59.0
##  9 Australia            1951            68.7
## 10 Austria              1951            65.2
## # ... with 13,322 more rows
```

Step 3: Let's now filter and graph the life expectancy for Angola using time series plot.

filter: Filter the data frame for Angola and store in *life_exp_Angola*.

```
life_exp_Angola <- life_exp_tidy%>%
  filter(country == "Angola")
life_exp_Angola
```

```
## # A tibble: 66 x 3
##    country  year life_expectancy
##    <chr>   <dbl>           <dbl>
##  1 Angola   1951            31.0
##  2 Angola   1952            31.6
##  3 Angola   1953            32.1
##  4 Angola   1954            32.7
##  5 Angola   1955            33.2
##  6 Angola   1956            33.8
##  7 Angola   1957            34.3
##  8 Angola   1958            34.9
##  9 Angola   1959            35.4
## 10 Angola   1960            36.0
## # ... with 56 more rows
```

use tail():Let's display the last 10 rows of the _life_exp_Angola.

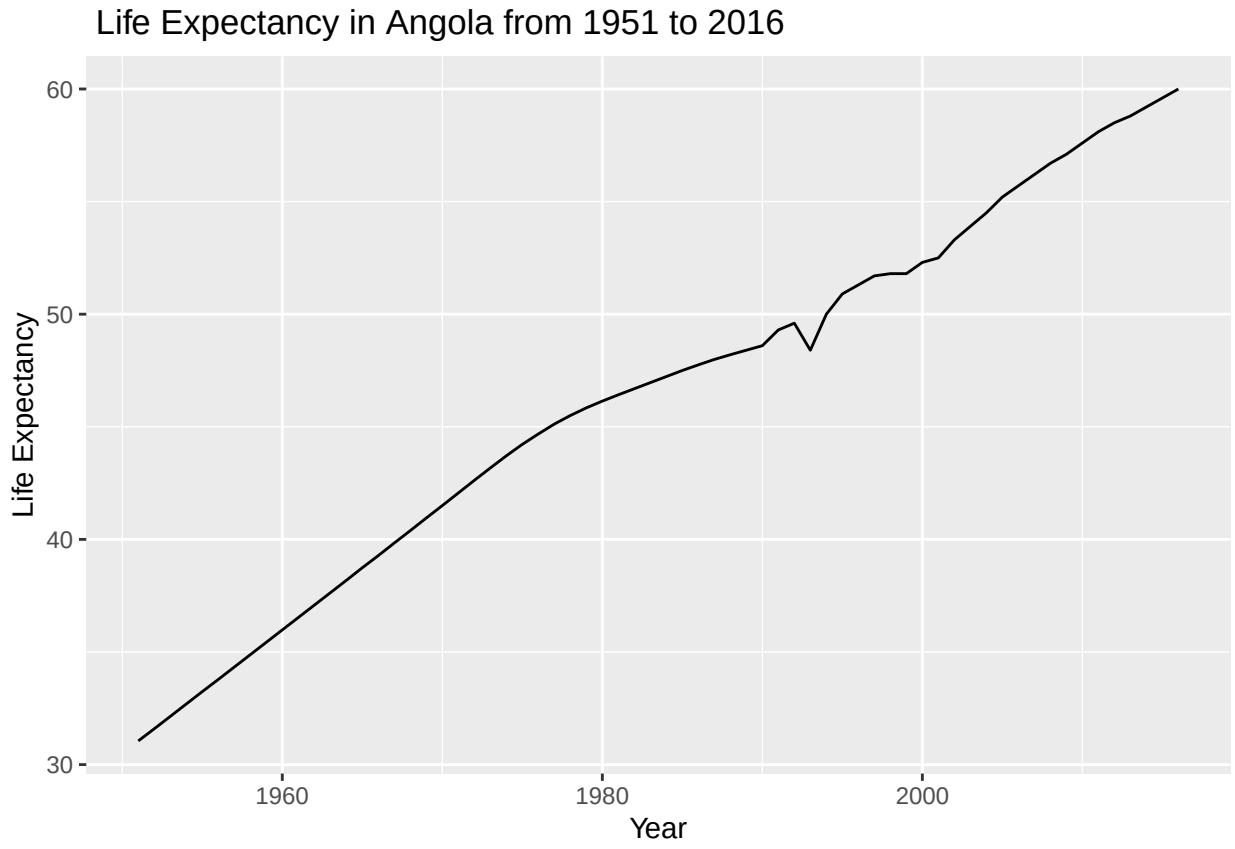
```
tail(life_exp_Angola, 10)
```

```
## # A tibble: 10 x 3
##    country  year life_expectancy
##    <chr>   <dbl>           <dbl>
##  1 Angola   2007            56.2
##  2 Angola   2008            56.7
```

```
##  3 Angola   2009            57.1
##  4 Angola   2010            57.6
##  5 Angola   2011            58.1
##  6 Angola   2012            58.5
##  7 Angola   2013            58.8
##  8 Angola   2014            59.2
##  9 Angola   2015            59.6
## 10 Angola   2016            60
```

Step 3: Let's use time series plot to visualize the life expectancy for Angola across years.

```
ggplot(life_exp_Angola, aes(x= year, y=life_expectancy)) +
  geom_line() + labs(x= "Year", y= "Life Expectancy") +
  ggtitle(" Life Expectancy in Angola from 1951 to 2016 ")
```



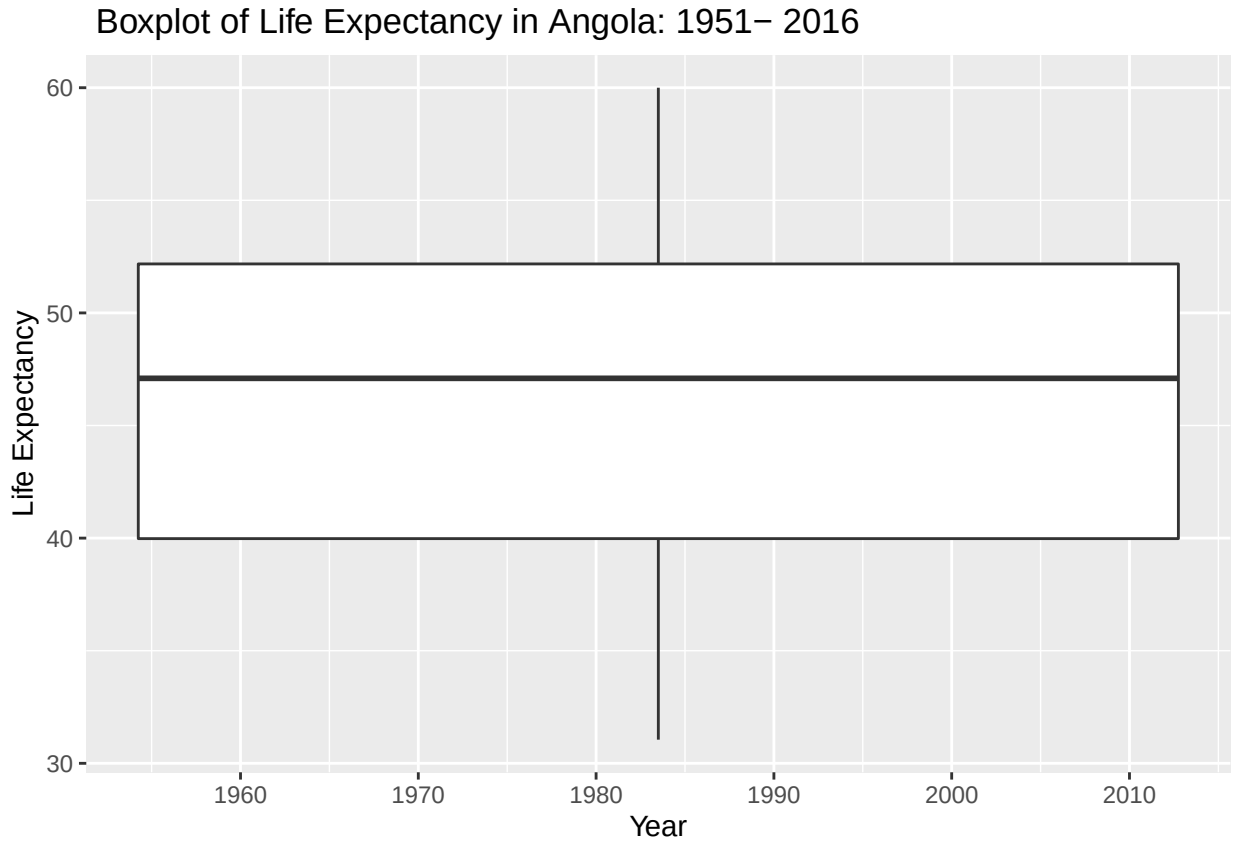
Let's display the summary the statistics for this data frame

```
summary(life_exp_Angola)
```

```
##    country                year      life_expectancy
##  Length:66          Min.   :1951   Min.   :31.05
##  Class :character   1st Qu.:1967   1st Qu.:39.98
##  Mode  :character   Median :1984   Median :47.09
##                     Mean   :1984   Mean   :46.33
##                     3rd Qu.:2000   3rd Qu.:52.17
##                     Max.   :2016   Max.   :60.00
```

```
ggplot(life_exp_Angola, aes( x=year, y= life_expectancy)) + geom_boxplot() + xlab("Year") + ylab( "Life
```

```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```



Analysis of our results:

From the time series we notice that Angola has experienced an exponential/positive increase in the life expectancy over the years.Back in the 1950s the life expectancy was very low (around 31 years of age). Angola was not a country then. It was one of Portugal's colonies in Africa. In 1975 Angola became independent from Portugal and in 1975 due to power struggle a civil war broke. One of the years in which the civil war caused the largest number of death, especially among Angolan men, was in 1992. Therefore, we notice on the time series plot that Angola experiences a decline in life expectancy during the same period. As we move further in the 1990s the life expectancy starts to go up again and by early 2000s life expectancy continues to rise as the civil war ended in 2002. From the summary statistics results we note that life expectancy has almost double between 1951 and 2016, from 31.05 to 60 years.A booming economy and increased in urban life style, jobs, access to healthcare, education and lower infant mortality are factors that might have contributed to the increase in life expectancy. These factors can be analyzed via data for another individual project.

Thank You

Bernardo Vimpi