Tabulation and Visualization

1 Purpose

The purpose of this activity is to provide you with an understanding of basic quantitative data handling and both tabular and visual data summarization in the R statistical programming environment. The goal is that you develop hands-on experience quantitatively describing data. These skills will be invaluable to you in your future careers and will set a foundation for all of the further course content.

2 Overview

This lab can be completed during class time and at-home. You should allocate time to complete the relevant portions of the lab in line with the scheduled topics for each week.

3 Your Task

1. Open the R console or R studio. Install the "gapminder" package, which contains a simple dataset of country-level variables, using `install.packages("gapminder")`. Assuming you encountered no issues, the "gapminder" package is now installed on your machine but not loaded.

2. Load the gapminder package using `library("gapminder")` . The package is now loaded and attached, so that you can use the data therein. Type `? gapminder` to open the documentation of the dataset.

3. Using the documentation and the Rconsole, answer the following:

   - How many variables are contained in the dataset? Confirm this in R using `ncol(gapminder)` .
   - How many "observations" or cases are contained in the dataset? Confirm this in R using `nrow(gapminder)` .
   - What is the "unit of analysis" of these data? Is an observation a country, a year, a country-year, or something else?

4. Use some R functions to obtain glances at the data.

```
> head(gapminder) # first few rows of the data
      country continent year lifeExp      pop gdpPercap
1 Afghanistan      Asia 1952  28.801  8425333  779.4453
2 Afghanistan      Asia 1957  30.332  9240934  820.8530
3 Afghanistan      Asia 1962  31.997 10267083  853.1007
4 Afghanistan      Asia 1967  34.020 11537966  836.1971
5 Afghanistan      Asia 1972  36.088 13079460  739.9811
6 Afghanistan      Asia 1977  38.438 14880372  786.1134
```

```
> str(gapminder) # a detailed summary of the 'str'ucture of the data
Classes tbl_df, tbl and 'data.frame':       1704 obs. of  6 variables:
 $ country  : Factor w/ 142 levels "Afghanistan",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ year     : int  1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
 $ lifeExp  : num  28.8 30.3 32 34 36.1 ...
 $ pop      : int  8425333 9240934 10267083 11537966 13079460 14880372 12881816 138679
 $ gdpPercap: num  779 821 853 836 740 ...

> summary(gapminder) # a quantitative summary of the data
        country        continent        year         lifeExp           pop
 Afghanistan:  12   Africa  :624   Min.   :1952   Min.   :23.60   Min.   :6.001e+04
 Albania    :  12   Americas:300   1st Qu.:1966   1st Qu.:48.20   1st Qu.:2.794e+06
 Algeria    :  12   Asia    :396   Median :1980   Median :60.71   Median :7.024e+06
 Angola     :  12   Europe  :360   Mean   :1980   Mean   :59.47   Mean   :2.960e+07
 Argentina  :  12   Oceania : 24   3rd Qu.:1993   3rd Qu.:70.85   3rd Qu.:1.959e+07
 Australia  :  12                  Max.   :2007   Max.   :82.60   Max.   :1.319e+09
 (Other)    :1632
```

5. How many countries are represented in the dataset? There are many ways to answer this. Two possibilities are:

   - `length(table(gapminder[["country"]]))`
   - `length(unique(gapminder[["country"]]))`

   You should get 142.

6. In the above code, what is the `table()` function doing? Using the help documentation `? table` to find out.

7. In the above code, what is the `length()` function doing? Using the help documentation `? length` to find out.

8. How many *years* are represented in the dataset? Modify the above code examples to find the answer. The correct answer is 12.

9. What is the life expectancy at birth in years for Algeria in 1967? You can find this out visually by printing the entire dataset by just typing `gapminder` at the console, or by typing `fix(gapminder)` to view the data in a spreadsheet format.

   But if we want to find the answer *computationally*, we need to *subset* the data. We already did some subsetting above with `gapminder[["country"]]`, which uses `[[` bracket notation to extract one variable from the dataset. There are other ways to extract one variable from a dataset:

   (a) `gapminder[["country"]]`

   (b) `gapminder$country`

   (c) `gapminder[, "country"]`

All of these are (basically) equivalent. We can also subset so that we only see certain rows of a dataset. To do that, we'll use the third form of subsetting to find out the answer to our question about life expectancy in Algeria in 1967. We'll do this in several steps to see what is going on but only the last step is needed to find the answer:

(a) Extract only data for Algeria:
`gapminder[gapminder[["country"]] == "Algeria", ]`

(b) Extract only the `lifeExp` variable:
`gapminder[, "lifeExp"]`

(c) Extract all life expectancy data for Algeria:
`gapminder[gapminder[["country"]] == "Algeria", "lifeExp", drop = FALSE]` [1]

(d) Subset data to only Algeria for only 1967:
`gapminder[gapminder[["country"]] == "Algeria" & gapminder[["year"]] == 1967, ]`

(e) Extract life expectancy data for Algeria for only 1967:
`gapminder[gapminder[["country"]] == "Algeria" & gapminder[["year"]] == 1967, "lifeExp"]`

You should find that the answer is 51.407.

10. Now modify the above code to find the GDP per capita for Somalia in 1992. The correct answer should be 926.9603.

11. Now let's calculate some *statistics* about these data. For example, to find the mean (average) life expectancy across all countries in 1992, we can extract the vector of life expectancy data for that year:
`gapminder[gapminder[["year"]] == 1992, "lifeExp"]`

    Then we can use the `mean()` function to take the mean of this vector:
`mean(gapminder[gapminder[["year"]] == 1992, "lifeExp"])`

    The answer should be 64.16034.

12. R provides many other functions to calculate basic statistics. Try the following functions on the life expectancy variable and see what you find:

    - Median: `median()`
    - Variance: `variance()`
    - Standard deviation: `sd()`

13. Another useful way of understanding *categorical* or *ordinal* data is to tabulate it. This means to count how many observations in the data take a particular value of the categorical variable. For example, to see how many countries there are in the dataset that exist on each continent, we can use:

```
> table(gapminder[gapminder[["year"]] == 1992, "continent"])

  Africa Americas     Asia   Europe  Oceania
      52       25       33       30        2
```

[1] Note: the use of `drop = FALSE` means that our code returns a "data frame" rather than a "vector." If you set `drop = TRUE` (or just leave that part of the code out), you will get a vector of data instead.

Note how we are using the `table()` function on a subset of data. Test your knowledge: how many European countries are represented in the data in 1967?

14. The previous exercise demonstrated a "one-way" tabulation. But we may also want to produce "two-way" or "multi-way" tabulations, which we will typically call a *crosstabulation* or *crosstab* for short. To do that, we will use the `ftable()` function. For example, to see how many cases we have on each continent for each year, we can do:

    ```
    ftable(gapminder[["year"]], gapminder[["continent"]])
    ```

    Crosstabs are only useful for categorical data. Why is this? What happens when you try to tabulate a numerical/interval variable:
    `ftable(gapminder[["year"]], gapminder[["lifeExp"]])`

15. Tabulations only provide us with one statistic: a *count* (or "subgroup total"). If we want to describe the data in other ways — for example to know what the mean life expectancy was in each year — we need to transform our data through an *aggregation*. An aggregation is:

    > a new dataset created from the data where observations (rows) are combined by a function applied to subsets of the data defined by a "grouping factor" such that the new dataset has one observation per group.

    For example, a simple aggregation would create a new dataset by applying a count function (`length()` in R notation) where country is the grouping factor:
    `aggregate(.  ~ country, data = gapminder, length)`

    This code uses the `aggregate()` function to produce an aggregation. The `. ~country` portion of the code indicates that we want to aggregate every variable in our dataset (indicated by `.`) by `country` for the dataset `gapminder` using the aggregating function `length`. If we replace `.` with a variable name, we will only aggregate that variable as opposed to every variable in the dataset.

    Modifying this code, answer our original question: what is the mean life expectancy (across all countries) by year? You should find, among other things, that the mean life expectancy in 2002 was 65.69492.

16. We can generalize this code further by aggregating not just by one variable but by many. For example, if we wanted to know the mean life expectancy by year *by continent*, we could say:
    `aggregate(lifeExp ~ continent + year, data = gapminder, mean)`

    Using that information, can you confirm that average life expectancy in African countries in 2007 was 54.80604 years?

17. Now modify the above code to create an aggregated dataset that expresses the variability of life expectancy across countries by continent and year. Which continent had the most between-country variability in life expectancies in 2007? What about in 1952?

18. Now let's try graphing. R provides several different graphing libraries. Among the easiest to use is "ggplot2". You will need to install it using `install.packages("ggplot2")` and load it using `library("ggplot2")`. The article you read for this week by Hadley Wickham describes the theory or philosophy underlying the package and there is a useful cheatsheet available on Moodle. The package is very powerful, so we will only learn the basics at this point.

19. Every graph made with ggplot2 starts with the `ggplot()` function. This function expresses *what* data we want to visualize and then we add graphing elements to describe *how* we want to visualize those data.

    To begin, you can specify that we want to visualize the life expectancy variable:
    `ggplot(gapminder, aes(x = lifeExp))`

    You will get a plot but it will be blank because we haven't specified how we want those data visualized. For example, if we just want to see the general distribution of the data, we request a histogram:

    ```
    ggplot(gapminder, aes(x = lifeExp)) + geom_histogram()
    ```

    What is the most prevalent life expectancy? What is the general shape of the distribution? Now try drawing a histogram for the `gdpPercap` variable. How does this distribution compare to the last one?

20. ggplot2 is most useful for visualizing multivariate relationships. For example, if we want to visualize the relationship between GDP per capita and life expectancy for all country–years, we can do:

    ```
    ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) + geom_point()
    ```

    What is the general shape of the relationship? Are the two variables closely related (correlated)? Calculate a Pearson's correlation coefficient for the relationship using:
    `cor(gapminder[["gdpPercap"]], gapminder[["lifeExp"]])`

    Recalling that the correlation coefficient ranges from -1 (perfectly negatively correlated) to 0 (linearly unrelated) to +1 (perfectly positively correlated), how strong is the relationship?

21. A common procedure in describing data is to transform it. Data that are highly *skewed*, as is GDP per capita, are often "log transformed" or "logged" by applying the logarithm function to the data in order to make them more easily interpretable. Try logging the GDP per capita variable and seeing how this changes the relationship between it and life expectancy:

    ```
    ggplot(gapminder, aes(x = log(gdpPercap))) + geom_histogram()
    ggplot(gapminder, aes(x = log(gdpPercap), y = lifeExp)) + geom_point()
    cor(log(gapminder[["gdpPercap"]]), gapminder[["lifeExp"]])
    ```

    How does your understanding of the relationship between the variables change?

22. Now let's try to look at even more complex relationships. Recall above that calculated mean life expectancy by country. Let's try to plot those results so that they are more easily understandable. Draw a scatterplot showing the relationship between year and life expectancy:

    ```
    ggplot(gapminder, aes(x = year, y = lifeExp)) + geom_point()
    ```

    How does the relationship here compare to the results we saw above from aggregating the data? One thing that can help clarify this relationship is to use a different "geom" to visualize the data. Try:

    ```
    ggplot(gapminder, aes(x = year, y = lifeExp)) + geom_smooth()
    ```

    What's the trend in life expectancy over time? Now practice a little more sophisticated ggplot2 skill by combining the scatterplot and smoothed trend line:

    ```
    ggplot(gapminder, aes(x = year, y = lifeExp)) +
      geom_point() + geom_smooth()}
    ```

23. This visualization is helpful for describing a time trend, but we haven't yet expressed the between-continent variation that was perhaps the most interesting part of work so far. To do that, we have to instruct the `ggplot()` function to account for the continent variable. To do that, let's add a grouping factor:

```
ggplot(gapminder, aes(x = year, y = lifeExp, color = continent)) +
  geom_point() + geom_smooth()
```

What's going on here? How would you describe or characterize the continent-specific time trends? How much has life expectancy changed over the period of our data? Is the trend the same in all continents?

24. If you look closely at this last visualization, you should see that it is displaying an aggregation. The `geom_smooth()` component of our visualization code is aggregating our data by continent and year and then visualizing that pattern. To check that this is in fact what is happening, create an aggregation of this relationship and then visualize it.

In case you're not sure how to do that, try the following:

```
newdat <- aggregate(lifeExp ~ continent + year, data = gapminder, mean)
ggplot(newdat, aes(x = year, y = lifeExp)) + geom_line(aes(color = continent))
```

How do the two graphs compare?[2]

25. One of the interesting things about the data so far has been that the variance of life expectancy across countries within a continent seemed to change over time. To get a visual sense of the distribution of the data, let's use a boxplot (or "box and whiskers") visualization:

```
ggplot(gapminder, aes(x = factor(year), y = lifeExp)) + geom_boxplot()
```

This graph shows the "five number summary" (minimum, first quartile, median, third quartile, and maximum) of the data. Compare the visualization against a tabular representation of the same:

```
aggregate(lifeExp ~ year, data = gapminder, fivenum)
```

Do they match?

26. This interesting part of that relationship, though, was between-continent differences. To show them, we can try a couple of different things. one is to group the results by continent using colour:

```
ggplot(gapminder, aes(x = factor(year), y = lifeExp)) +
    geom_boxplot(aes(color = continent))
```

That's kind of ugly and unreadable though. Colour here is just like an axis, we're adding a dimension to the visualization captured by colour. Rather than using colour, we could show that dimension in another way. For example, we could swap the use of year and continent:

[2]Note: If you want to compare graphs, it can be useful to save them to your computer's hard drive. To do this, plot the graph, then use the `ggsave()` function and specify a filename, like `ggsave("graph1.png")`. Note that graphs will be saved, by default, in your working directory, which you can identify by typing `getwd()` at the console

```
ggplot(gapminder, aes(x = continent, y = lifeExp)) +
    geom_boxplot(aes(color = factor(year)))
```

In that visualization, we see the separate trends for each continent, with year treated as a colour. This is better because it shows the continent-specific trends more clearly and it also shows how much between-country variation there is in Africa and Asia and how little there is in Asia.

What's going on there? Why is the variation in Oceania so small? Create an subset of just the data by extracting just the observations for countries in Oceania. How many are there and which countries are they?

27. The last visualization used colour as a grouping factor. Another approach is to do "faceting" where several smaller graphs are made, and each group represents a subset of the data. To do that, we will add a `facet_wrap()` feature to the visualization:

```
ggplot(gapminder, aes(x = factor(year), y = lifeExp)) +
    geom_boxplot() + facet_wrap(~continent)
```

If you don't like the look, you can customize it, for example, the following might be better:

```
ggplot(gapminder, aes(x = factor(year), y = lifeExp)) +
    geom_boxplot() + facet_wrap(~continent, nrow = 1)
```

Substantively, this is the same data as shown above but instead of using colours to represent year or continent, we are using facets. In this way, it should be clear that an axis, a colour, and a facet are at the most basic level just an axis — that is, a way to add dimensionality to a visualization.

28. We can add even more dimensions when we draw scatterplots. Let's go back to the relationship between GDP per capita and life expectancy. We can colour the graph to show continent-specific data:

```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, color = continent)) +
  geom_point()
```

But we could also show that dimensionality using facets:

```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
  geom_point() + facet_wrap(~continent)
```

Or we could convey years as facets:

```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
  geom_point() + facet_wrap(~year)
```

Or see all of our data by facetting by year and continent:

```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
  geom_point() + facet_wrap(~year+continent)
```

That basically becomes unreadable, though. So we might be better of displaying one of those dimensions through colour and maybe log transform for clarity:

```
ggplot(gapminder, aes(x = log(gdpPercap), y = lifeExp)) +
  geom_point(aes(colour = continent)) + facet_wrap(~year)
```

We can add further dimensions, such as population size, as well. Now not only axes, colour, and facet are used as dimensions, but so is the size of the points representing each country:

```
ggplot(gapminder, aes(x = log(gdpPercap), y = lifeExp)) +
  geom_point(aes(colour = continent, size = pop)) +
  facet_wrap(~year)
```

What are the large outlier countries in Asia?

29. Finally, it is worth noting that ggplot2 visualizations are highly customizable. One of the simplest ways to customize them is to use "themes". These change the colours and other features of graphs in ways that may be attractive. To show how they work, we will save our graph as an object called `g` and then change the theme associated with it to see various options:

```
g <- ggplot(gapminder, aes(x = log(gdpPercap), y = lifeExp)) +
  geom_point(aes(colour = continent, size = pop)) +
  facet_wrap(~year)

g + theme_bw()
g + theme_gray()
g + theme_minimal()
```

See `? theme_bw` for options. There is an another package called "ggthemes" that adds additional options. Install and load it to use these themes, such as:

```
library("ggthemes")
g + theme_economist()
g + theme_solarized(light = FALSE)
```

It is also possible to modify other features such as:

- Title, using `ggtitle()`
- x-axis and y-axis labels, using `xlab()` and `ylab()`
- x-axis and y-axis labels and dimensions, using `scale_x_continuous()`, `scale_x_continuous()`, etc.

See the documentation and the ggplot2 cheatsheet for guidance on these modifications.

4 Feedback and Assistance

There is no explicit feedback on this activity, but please attend office hours if you have questions about using R or about any of the substantive material in this lab activity.