---

Check Dataset for being Tidy

A dataset is tidy when each observation is in a row, and each column has a variable (not a value). For example, the following is NOT tidy, because the years 2005, 2006, 2007 which are columns are the value and not a variable.

---

#The first function we will use is ggplot(). This function allows us to define the data that we will be using to make the plot, as well as the aesthetic properties that will be mapped to the geometric objects. That is, we will tell ggplot which data (a data frame) we are interested in and how each of the variables in our dataset will be used (e.g. as an x or y coordinate, as a coloring variable or a size variable, etc).

#Below, we define our first ggplot object using the ggplot function, with the gapminder dataset and the x and y aesthetics defined by the gdpPercap and lifeExp variables, respectively.

The output of this function is a grid with gdpPercap as the x-axis and lifeExp as the y-axis. However, we have not yet told ggplot what type of geometric object the data will be mapped to, so no data has been displayed.

ggplot(gapminder, aes(x = gdpPercap, y = lifeExp))

Aesthetic mapping to layers

Next, we will add a “geom” layer to our ggplot object. For example, we could add a points layer which would automatically adopt the aesthetic mapping described in the previous line of code.

# describe the base ggplot object and tell it what data we are interested in along with the aesthetic mapping

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

# add a points layer on top

geom\_point()

What we have done is map each country (row) in the data to a point in the space defined by the GDP and life expectancy value. The end result is an ugly blob of points. Fortunately, there are many things that we can do to make this blob of points prettier.

For example, we can change the transparency of all points by setting the alpha argument to a low value, changing the color of the points to be blue instead of black, and making the points smaller.

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

geom\_point(alpha = 0.5, col = "cornflowerblue", size =

Note that the above argument changed the alpha value and color for all of the points at once.

One of the truly powerful features of ggplot2 is the ability to change these aesthetics based on the data itself. For example, perhaps we want to color each point by its continent. Instead of separating the data into five different subsets (based on the possible values of continent), and adding the different colored points separately, we can simply add all the points once and add an colour aesthetic map for continent.

Note that whenever we are using a variable from the data to describe an aesthetic property of a geom, this aesthetic property needs to be included in the aes() function.

unique(gapminder$continent)

We could also add aesthetic mappings for other features such as shape, size, transparency (alpha), and more! For example, changing the size based on population:

ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

geom\_point(alpha = 0.5)

So far, we have only seen scatterplots (point geoms), however, there are many other geoms we could add, including:

lines

histograms

boxplots and violin plots

barplots

smoothed curves

ggplot(gapminder, aes(x = year, y = lifeExp, group = country, color = continent)) +

geom\_line(alpha = 0.5)

---ggplot(gapminder, aes(x = continent, y = lifeExp, fill = continent)) +

geom\_boxplot()

--ggplot(gapminder, aes(x = lifeExp)) +

geom\_histogram(binwidth = 3)

ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, size = pop)) +

geom\_point(aes(color = continent), alpha = 0.5) +

geom\_smooth(se = FALSE, method = "loess", color = "grey30")

Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.

ℹ Please use `linewidth` instead.

`geom\_smooth()` using formula = 'y ~ x'

Warning: The following aesthetics were dropped during statistical transformation: size

ℹ This can happen when ggplot fails to infer the correct grouping structure in

the data.

ℹ Did you forget to specify a `group` aesthetic or to convert a numerical

variable into a factor?

We are going to return to our original scatterplot example to discuss scales, legend and positioning.

To remind you, this scatterplot showed GDP per capita against life expectancy for each country colored by continent and sized by population.

To keep things simple, let’s filter to a single year.

library(dplyr)

gapminder\_2007 <- gapminder %>% filter(year == 2007)

ggplot(gapminder\_2007, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

geom\_point(alpha = 0.5)

The scale of a plot describes the features of the space in which it is plotted. Arguably, it would be better to show gdpPercap on a logarithmic scale, rather than in its raw form. Fortunately, this is easy to do using a scale function, which can be considered another layer that transforms our plot.

ggplot(gapminder\_2007, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

geom\_point(alpha = 0.5) +

scale\_x\_log10()

The default x- (and y-) axes scales are scale\_x\_continuous and scale\_y\_continuous, but other options include scale\_x\_sqrt and scale\_x\_reverse.

Each of these scale functions has many options including changing the limits, the breaks, etc. For example in the plot below, we manipulate the x-axis by providing arguments to our scale function of choice.

ggplot(gapminder\_2007, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

geom\_point(alpha = 0.5) +

# clean the x-axis breaks

scale\_x\_log10(breaks = c(1, 10, 100, 1000, 10000),

limits = c(1, 120000))

The scale of a plot describes the features of the space in which it is plotted. Arguably, it would be better to show gdpPercap on a logarithmic scale, rather than in its raw form. Fortunately, this is easy to do using a scale function, which can be considered another layer that transforms our plot.

ggplot(gapminder\_2007, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

geom\_point(alpha = 0.5) +

scale\_x\_log10()

The default x- (and y-) axes scales are scale\_x\_continuous and scale\_y\_continuous, but other options include scale\_x\_sqrt and scale\_x\_reverse.

Each of these scale functions has many options including changing the limits, the breaks, etc. For example in the plot below, we manipulate the x-axis by providing arguments to our scale function of choice.

ggplot(gapminder\_2007, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

geom\_point(alpha = 0.5) +

# clean the x-axis breaks

scale\_x\_log10(breaks = c(1, 10, 100, 1000, 10000),

limits = c(1, 120000))

Notice that we changed the name of the x-axis in the plot using the name argument. This could also be done using the labs function. As an example, below we add a title and change the name of the y-axis and legends using the labs function.

ggplot(gapminder\_2007, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

# add scatter points

geom\_point(alpha = 0.5) +

# log-scale the x-axis

scale\_x\_log10() +

# change labels

labs(title = "GDP versus life expectancy in 2007",

x = "GDP per capita (log scale)",

y = "Life expectancy",

size = "Popoulation",

color = "Continent")

We could also manipulate the scale of the size variable. Below, we expand the range of sizes and clean up the variable name. Since the variable we provided for size is a continuous variable (pop) we use the scale\_size\_continuous argument.

ggplot(gapminder\_2007, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

# add scatter points

geom\_point(alpha = 0.5) +

# log-scale the x-axis

scale\_x\_log10() +

# change labels

labs(title = "GDP versus life expectancy in 2007",

x = "GDP per capita (log scale)",

y = "Life expectancy",

size = "Popoulation (millions)",

color = "Continent") +

# change the size scale

scale\_size(range = c(0.1, 10),

breaks = 1000000 \* c(250, 500, 750, 1000, 1250),

labels = c("250", "500", "750", "1000", "12

###Faceting

Sometimes we want to be able to make multiple plots of the same thing across different categories. This can be achieved with minimal repetition using faceting.

In the example below, we will remake the plot above individually for each continent.

ggplot(gapminder\_2007, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

# add scatter points

geom\_point(alpha = 0.5) +

# log-scale the x-axis

scale\_x\_log10() +

# change labels

labs(title = "GDP versus life expectancy in 2007",

x = "GDP per capita (log scale)",

y = "Life expectancy",

size = "Popoulation (millions)",

color = "Continent") +

# change the size scale

scale\_size(range = c(0.1, 10),

breaks = 1000000 \* c(250, 500, 750, 1000, 1250),

labels = c("250", "500", "750", "1000", "1250")) +

# add faceting

facet\_wrap(~continent)

###Themes: making even more beautiful figures with ggplot2

One of the first things I usually do when I make a ggplot is edit the default theme. I actually really don’t like the grey background, nor do I like having a grid unless it really helps with the plot interpretation.

One of the simplest themes is theme\_classic, however there are several other themes to choose from. The ggthemes package offers many additional themes, but you could also make your own using the theme() function.

ggplot(gapminder\_2007, aes(x = gdpPercap, y = lifeExp, color = continent, size = pop)) +

# add scatter points

geom\_point(alpha = 0.5) +

# clean the axes names and breaks

scale\_x\_log10(breaks = c(1000, 10000),

limits = c(200, 120000)) +

# change labels

labs(title = "GDP versus life expectancy in 2007",

x = "GDP per capita (log scale)",

y = "Life expectancy",

size = "Popoulation (millions)",

color = "Continent") +

# change the size scale

scale\_size(range = c(0.1, 10),

breaks = 1000000 \* c(250, 500, 750, 1000, 1250),

labels = c("250", "500", "750", "1000", "1250")) +

# add a nicer theme

theme\_classic(base\_family = "Avenir")

###As an example of further customization of the ggplot theme, below we do the following:

move the legend to the top (set legend.position = "top" in theme())

removing the population legend (set guide = "none" in scale\_size())

remove the axes lines (set axis.line = element\_blank() in theme())

add some text annotations (add geom\_text layer)

ggplot(gapminder\_2007) +

# add scatter points

geom\_point(aes(x = gdpPercap, y = lifeExp, color = continent, size = pop),

alpha = 0.5) +

# add some text annotations for the very large countries

geom\_text(aes(x = gdpPercap, y = lifeExp + 3, label = country),

color = "grey50",

data = filter(gapminder\_2007, pop > 1000000000 | country %in% c("Nigeria", "United States"))) +

# clean the axes names and breaks

scale\_x\_log10(limits = c(200, 60000)) +

# change labels

labs(title = "GDP versus life expectancy in 2007",

x = "GDP per capita (log scale)",

y = "Life expectancy",

size = "Popoulation",

color = "Continent") +

# change the size scale

scale\_size(range = c(0.1, 10),

# remove size legend

guide = "none") +

# add a nicer theme

theme\_classic() +

# place legend at top and grey axis lines

theme(legend.position = "top",

axis.line = element\_line(color = "grey85"),

axis.ticks = element\_line(color = "grey85"))

###Saving your plots

You can save your plots using the ggsave() function.

p <- ggplot(gapminder\_2007) +

# add scatter points

geom\_point(aes(x = gdpPercap, y = lifeExp, color = continent, size = pop),

alpha = 0.5) +

# add some text annotations for the very large countries

geom\_text(aes(x = gdpPercap, y = lifeExp + 3, label = country),

color = "grey50",

data = filter(gapminder\_2007, pop > 1000000000 | country %in% c("Nigeria", "United States"))) +

# clean the axes names and breaks

scale\_x\_log10(limits = c(200, 60000)) +

# change labels

labs(title = "GDP versus life expectancy in 2007",

x = "GDP per capita (log scale)",

y = "Life expectancy",

size = "Popoulation",

color = "Continent") +

# change the size scale

scale\_size(range = c(0.1, 10),

# remove size legend

guide = "none") +

# add a nicer theme

theme\_classic() +

# place legend at top and grey axis lines

theme(legend.position = "top",

axis.line = element\_line(color = "grey85"),

axis.ticks = element\_line(color = "grey85"))

ggsave("beautiful\_plot.png", p, dpi = 300, width = 6, height = 4)

---

List of number of trees in California by Location by Year

Location Plant Name Count\_2005 Count\_2006 Count\_2007

Danville Oak tree 100 120 222

San Ramon Sequoia 50 10 22

Pleasanton Oak tree 24 17 88

Alamo Oak tree NA 3 22

The above data is an example of WIDE data. However, the following is tidy or LONG:

Location Plant Name Year Count

Danville Oak tree 2005 100

Danville Oak tree 2006 120

San Ramon Sequoia 2005 50

Pleasanton Oak tree 2005 24

Alamo Oak tree 2005 NA

Here is how you can change the WIDE data to long assuming that the wide data set is called ‘wide\_data’:

long\_data <- pivot\_longer(

wide\_data,

cols = starts\_with("Count"),

names\_to = "Year",

names\_prefix = "Count\_",

values\_to = "Count"

)

And if you want to convert back to wide again

wide\_data\_again <- pivot\_wider(

long\_data,

names\_from = "Year",

names\_prefix = "Count\_",

values\_from = "Count"

)

How to add a row to a dataset

Use rbind. Example:

new\_df <- rbind(new\_bird <- c("female", 3.6, 3.9)

hbirds<- rbind(hbirds, new\_df)

Add a new column

Use cbind. Example:

depth\_ft <- c(4.15, 4.13, 4.12, 3.21, 3.23, 3.20, 5.67, 5.65, 5.66)

hsprings <- cbind(hsprings, depth\_ft)

If you want to choose a column:

Assume the dataset is called df:

select(df, contains("col"))

Here we are selecting all columns whose name has “col” in it

select\_if(df, is.numeric)

Here we are selecting all columns that are numeric

select\_if(df, ~!is.numeric(.))

Here we are selecting all columns that are NOT numeric. Note ! that means not but you have to add ~ as well

# Check Dataset for being Tidy

A dataset is tidy when each observation is in a row, and each column has a variable (not a value). For example, the following is NOT tidy, because the years 2005, 2006, 2007 which are columns are the value and not a variable.

List of number of trees in California by Location by Year

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Location | Plant Name | Count\_2005 | Count\_2006 | Count\_2007 |
| Danville | Oak tree | 100 | 120 | 222 |
| San Ramon | Sequoia | 50 | 10 | 22 |
| Pleasanton | Oak tree | 24 | 17 | 88 |
| Alamo | Oak tree | NA | 3 | 22 |

The above data is an example of WIDE data. However, the following is tidy or LONG:

|  |  |  |  |
| --- | --- | --- | --- |
| Location | Plant Name | Year | Count |
| Danville | Oak tree | 2005 | 100 |
| Danville | Oak tree | 2006 | 120 |
| San Ramon | Sequoia | 2005 | 50 |
| Pleasanton | Oak tree | 2005 | 24 |
| Alamo | Oak tree | 2005 | NA |

Here is how you can change the WIDE data to long assuming that the wide data set is called ‘wide\_data’:

long\_data <- pivot\_longer(

wide\_data,

cols = starts\_with("Count"),

names\_to = "Year",

names\_prefix = "Count\_",

values\_to = "Count"

)

And if you want to convert back to wide again

wide\_data\_again <- pivot\_wider(

long\_data,

names\_from = "Year",

names\_prefix = "Count\_",

values\_from = "Count"

)

## How to add a row to a dataset

Use rbind. Example:

new\_df <- rbind(new\_bird <- c("female", 3.6, 3.9)

hbirds<- rbind(hbirds, new\_df)

Add a new column

Use cbind. Example:

depth\_ft <- c(4.15, 4.13, 4.12, 3.21, 3.23, 3.20, 5.67, 5.65, 5.66)

hsprings <- cbind(hsprings, depth\_ft)

**If you want to choose a column:**

Assume the dataset is called df:

select(df, contains("col"))

Here we are selecting all columns whose name has “col” in it

select\_if(df, is.numeric)

Here we are selecting all columns that are numeric

select\_if(df, ~!is.numeric(.))

Here we are selecting all columns that are NOT numeric. Note ! that means not but you have to add ~ as well

# 1. Trend Analysis of Life Expectancy Over Time for Each Continent

This analysis will reveal how life expectancy has changed over the years across different continents.

**Approach:**

Group data by continent and year.

Calculate the average life expectancy for each continent per year.

Plot the trends over time.

library(ggplot2)

library(dplyr)

gapminder %>%

group\_by(continent, year) %>%

summarise(average\_lifeExp = mean(lifeExp, na.rm = TRUE)) %>%

ggplot(aes(x = year, y = average\_lifeExp, color = continent)) +

geom\_line() +

labs(title = "Life Expectancy Trends by Continent", x = "Year", y = "Average Life Expectancy") +

theme\_minimal()

# 2. Analysis of GDP per Capita and Life Expectancy

Investigate the relationship between GDP per capita and life expectancy, which can indicate how economic prosperity relates to health outcomes.

**Approach**:

Create a scatter plot with GDP per capita on the x-axis and life expectancy on the y-axis.

Color-code data points by continent to observe regional differences.

Optionally, use data from a specific year or range of years for clearer visualization.

gapminder %>%

filter(year == 2007) %>%

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

geom\_point(alpha = 0.5) +

scale\_x\_log10() + # GDP per capita values are skewed, so log scale is used for better visualization

labs(title = "Life Expectancy vs. GDP per Capita in 2007", x = "GDP per Capita (Log Scale)", y = "Life Expectancy") +

theme\_minimal()

# 3. Population Growth Analysis by Continent

Examine how the population in each continent has changed over time.

**Approach**:

Summarize total population by continent and year.

Plot population growth trends for each continent.

gapminder %>%

group\_by(continent, year) %>%

summarise(total\_population = sum(pop)) %>%

ggplot(aes(x = year, y = total\_population, color = continent)) +

geom\_line() +

labs(title = "Population Growth by Continent", x = "Year", y = "Total Population") +

theme\_minimal()

# 4. Distribution of Life Expectancy Across Continents for a Specific Year

This analysis will show the distribution and variation of life expectancy within continents for a given year.

**Approach**:

Filter data for a specific year.

Use a boxplot to display the distribution of life expectancy by continent.

gapminder %>%

filter(year == 2007) %>%

ggplot(aes(x = continent, y = lifeExp, fill = continent)) +

geom\_boxplot() +

labs(title = "Life Expectancy Distribution by Continent in 2007", x = "Continent", y = "Life Expectancy") +

theme\_minimal()

# 5. Economic Growth Over Time

Analyze how the GDP per capita has evolved over time for different countries or continents, highlighting economic growth patterns.

**Approach**:

Calculate the average GDP per capita for each year and continent.

Plot these averages over time to observe trends.

gapminder %>%

group\_by(year, continent) %>%

summarise(average\_gdpPercap = mean(gdpPercap, na.rm = TRUE)) %>%

ggplot(aes(x = year, y = average\_gdpPercap, color = continent)) +

geom\_line() +

labs(title = "Average GDP per Capita Over Time by Continent", x = "Year", y = "Average GDP per Capita") +

theme\_minimal()

# 6. Correlation between Population Growth and GDP per Capita

Investigate if there's a correlation between population growth rates and GDP per capita changes, which could suggest how economic conditions affect population dynamics.

**Approach**:

Calculate yearly growth rates for both population and GDP per capita.

Use scatter plots or correlation tests to explore relationships between these variables.

gapminder\_growth <- gapminder %>%

group\_by(country) %>%

arrange(year) %>%

mutate(pop\_growth\_rate = (lead(pop) - pop) / pop \* 100,

gdp\_growth\_rate = (lead(gdpPercap) - gdpPercap) / gdpPercap \* 100) %>%

filter(!is.na(pop\_growth\_rate) & !is.na(gdp\_growth\_rate))

# Scatter plot with correlation line

ggplot(gapminder\_growth, aes(x = pop\_growth\_rate, y = gdp\_growth\_rate)) +

geom\_point(alpha = 0.5) +

geom\_smooth(method = "lm", color = "blue") +

labs(title = "Correlation between Population Growth and GDP Growth",

x = "Population Growth Rate (%)",

y = "GDP Growth Rate (%)") +

theme\_minimal()

# 7. Impact of Health on Economic Development

Explore how health indicators, such as life expectancy, correlate with economic indicators like GDP per capita across countries.

**Approach**:

Perform a correlation analysis or linear regression between life expectancy and GDP per capita to assess the relationship strength and direction.

Visualize the relationship with scatter plots, possibly adding trend lines to highlight patterns.

library(ggplot2)

library(dplyr)

gapminder %>%

ggplot(aes(x = lifeExp, y = gdpPercap)) +

geom\_point(aes(color = continent)) +

geom\_smooth(method = "lm", se = FALSE) +

labs(title = "Life Expectancy vs GDP per Capita", x = "Life Expectancy", y = "GDP per Capita") +

theme\_minimal()

# 8. Life Expectancy and GDP per Capita Clusters

Use cluster analysis to identify groups of countries with similar life expectancy and GDP per capita characteristics, which can reveal patterns in global development levels.

**Approach**:

Utilize clustering techniques like K-means on life expectancy and GDP per capita.

Visualize the clusters using scatter plots or multidimensional scaling (MDS).

# Sample for K-means clustering (Pseudo-code, adjust as necessary)

library(cluster)

data\_filtered <- gapminder %>% filter(year == 2007) %>% select(lifeExp, gdpPercap) %>% na.omit()

clusters <- kmeans(data\_filtered, centers = 3)

ggplot(data\_filtered, aes(x = lifeExp, y = gdpPercap, color = as.factor(clusters$cluster))) + geom\_point()

# 9. Comparative Analysis of Continents

Perform a comparative analysis of continents across various indicators to understand differences in development, health outcomes, and economic conditions.

**Approach**:

Use data visualization techniques like facet plots to compare trends across continents.

Analyze disparities and similarities in development indicators.

# GDP per capita by continent over time

gapminder %>%

group\_by(year, continent) %>%

summarise(average\_gdpPercap = mean(gdpPercap, na.rm = TRUE)) %>%

ggplot(aes(x = year, y = average\_gdpPercap, color = continent)) +

geom\_line() +

labs(title = "Average GDP per Capita Over Time by Continent", x = "Year", y = "Average GDP per Capita") +

theme\_minimal()

# Life expectancy by continent over time

gapminder %>%

group\_by(year, continent) %>%

summarise(average\_lifeExp = mean(lifeExp, na.rm = TRUE)) %>%

ggplot(aes(x = year, y = average\_lifeExp, color = continent)) +

geom\_line() +

labs(title = "Average Life Expectancy Over Time by Continent", x = "Year", y = "Average Life Expectancy") +

theme\_minimal()

# 10. Life Expectancy and GDP per Capita over Time by Continent

gapminder %>%

ggplot(aes(x = year, y = lifeExp, color = continent)) +

geom\_line() + # Line plot for life expectancy trends

geom\_point(aes(size = gdpPercap), show.legend = FALSE) + # Point size represents GDP per capita

facet\_wrap(~continent, scales = "free\_y") + # Faceting by continent, allowing y-axis to vary

labs(title = "Life Expectancy and GDP per Capita over Time by Continent",

x = "Year",

y = "Life Expectancy",

size = "GDP per Capita") +

theme\_minimal() +

theme(axis.text.x = element\_text(angle = 45, hjust = 1))

# Comparison of GDP per Capita Growth Between Selected Countries

# Selected countries for comparison

selected\_countries <- c("United States", "China", "India", "Germany", "Brazil")

gapminder %>%

filter(country %in% selected\_countries) %>%

ggplot(aes(x = year, y = gdpPercap, color = country)) +

geom\_line() +

labs(title = "GDP per Capita Growth Comparison",

x = "Year",

y = "GDP per Capita (US$)") +

theme\_minimal() +

theme(legend.title = element\_blank())

# Continent-Wise Population Growth Trends

gapminder %>%

group\_by(year, continent) %>%

summarise(total\_population = sum(pop)) %>%

ggplot(aes(x = year, y = total\_population, color = continent)) +

geom\_line() +

labs(title = "Continent-Wise Population Growth Trends",

x = "Year",

y = "Total Population") +

theme\_minimal() +

theme(legend.title = element\_blank())

# Comparative Analysis of Life Expectancy vs. GDP per Capita by Continent

gapminder %>%

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

geom\_point(alpha = 0.5) +

scale\_x\_log10() + # Log scale for GDP per Capita to handle wide data range

facet\_wrap(~continent) +

labs(title = "Life Expectancy vs. GDP per Capita by Continent",

x = "GDP per Capita (Log Scale)",

y = "Life Expectancy") +

theme\_minimal() +

theme(legend.title = element\_blank())

# Yearly Comparison of Average Life Expectancy Between Two Continents

# Filtering for two continents for comparison

two\_continents <- gapminder %>%

filter(continent %in% c("Europe", "Asia"))

two\_continents %>%

group\_by(year, continent) %>%

summarise(average\_lifeExp = mean(lifeExp)) %>%

ggplot(aes(x = year, y = average\_lifeExp, color = continent)) +

geom\_line() +

labs(title = "Average Life Expectancy: Europe vs. Asia",

x = "Year",

y = "Average Life Expectancy") +

theme\_minimal() +

theme(legend.title = element\_blank())

# Comparative Analysis: Countries with the Highest GDP per Capita by Continent

This analysis aims to identify the country with the highest GDP per Capita within each continent for a specific year and then uses facets to compare these countries across different indicators such as life expectancy and population.

**Steps**:

1. Filter data for a specific year to simplify the analysis. Let's choose 2007, the last year in many datasets like Gapminder.
2. Identify the country with the highest GDP per Capita within each continent for the selected year.
3. Create a subset of the data that includes only these countries.
4. Use facets to compare these countries across life expectancy and population in the year 2007.

# Step 1: Filter data for the year 2007

data\_2007 <- gapminder %>%

filter(year == 2007)

# Step 2: Identify the country with the highest GDP per Capita within each continent

highest\_gdp\_country <- data\_2007 %>%

group\_by(continent) %>%

top\_n(1, gdpPercap) %>%

ungroup()

# Step 3: Comparative analysis using facets

# Compare life expectancy and population for the countries with the highest GDP per Capita

ggplot(highest\_gdp\_country, aes(x = country, y = lifeExp)) +

geom\_col() +

facet\_wrap(~continent, scales = "free\_x") +

labs(title = "Countries with the Highest GDP per Capita by Continent in 2007",

x = "Country",

y = "Life Expectancy") +

theme\_minimal() +

theme(axis.text.x = element\_text(angle = 45, hjust = 1))

# For population comparison, simply change the y aesthetic to 'pop'

ggplot(highest\_gdp\_country, aes(x = country, y = pop)) +

geom\_col() +

facet\_wrap(~continent, scales = "free\_x") +

labs(title = "Population of Countries with the Highest GDP per Capita by Continent in 2007",

x = "Country",

y = "Population") +

theme\_minimal() +

theme(axis.text.x = element\_text(angle = 45, hjust = 1))

# 11. Find the top 3 countries by continent with the most population change between 1952 and 2007

# Calculate absolute population change and select top 3 countries per continent

population\_change <- gapminder %>%

filter(year %in% c(1952, 2007)) %>%

group\_by(country, continent) %>%

summarize(start\_pop = min(pop[year == 1952]), end\_pop = max(pop[year == 2007]), .groups = 'drop') %>%

mutate(abs\_pop\_change = abs(end\_pop - start\_pop)) %>%

arrange(desc(abs\_pop\_change))

# Identify top 3 countries in each continent with the highest absolute population changes

top\_changes\_per\_continent <- population\_change %>%

group\_by(continent) %>%

top\_n(3, abs\_pop\_change) %>%

ungroup() %>%

arrange(continent, desc(abs\_pop\_change))

# For visualization, we can plot the population change of these countries

ggplot(top\_changes\_per\_continent, aes(x = reorder(country, abs\_pop\_change), y = abs\_pop\_change, fill = continent)) +

geom\_col() +

coord\_flip() +

labs(title = "Top 3 Countries with Highest Absolute Population Change by Continent (1952-2007)",

x = "Country",

y = "Absolute Population Change") +

facet\_wrap(~continent) +

theme\_light() +

theme(axis.text.x = element\_text(angle = 60, hjust = 1)) # Adjusting the x-axis labels to be at a 60-degree angle

# To graph max, min, and mean Life expectancy for all the years by continent

# Assuming gapminder is your dataset

lifeExp\_summary <- gapminder %>%

group\_by(continent) %>%

summarise(

Min\_LifeExp = min(lifeExp, na.rm = TRUE),

Mean\_LifeExp = mean(lifeExp, na.rm = TRUE),

Max\_LifeExp = max(lifeExp, na.rm = TRUE)

) %>%

pivot\_longer(cols = c(Min\_LifeExp, Mean\_LifeExp, Max\_LifeExp), names\_to = "Statistic", values\_to = "LifeExp")

# Plotting

ggplot(lifeExp\_summary, aes(x = continent, y = LifeExp, fill = Statistic)) +

geom\_bar(stat = "identity", position = "dodge") +

theme\_minimal() +

labs(title = "Life Expectancy Statistics by Continent",

x = "Continent", y = "Life Expectancy") +

scale\_fill\_brewer(palette = "Pastel1") +

theme(axis.text.x = element\_text(angle = 45, hjust = 1))

# Find top 5 countries by population in each continent for a particular year

gapminder %>%

filter(year == 2007) %>%

arrange(desc(pop)) %>%

group\_by(continent) %>%

mutate(rank = row\_number()) %>%

filter(rank <= 5)

link to resource: [[Rebecca Barter's](http://www.rebeccabarter.com/blog/2017-11-17-ggplot2_tutorial)](<http://www.rebeccabarter.com/blog/2017-11-17-ggplot2_tutorial/>)

# Use of Pivot\_wider to make the table wide

gapminder %>%

filter(year == 1952 | year == 2007) %>%

pivot\_wider(names\_from = year, values\_from = pop, names\_prefix = "pop") %>%

group\_by(country) %>%

summarize(across(starts\_with("pop"), sum, na.rm = TRUE), .groups = "drop")%>%

mutate(popgrowth=pop2007 -pop1952) %>%

arrange(desc(popgrowth))

The above shows countries with the largest population growth since 1952

If you want countries with largest GDP growth since 1952, you can use the following

gapminder %>%

filter(year == 1952 | year == 2007) %>%

pivot\_wider(names\_from = year, values\_from = gdpPercap, names\_prefix = "gdp") %>%

group\_by(country) %>%

summarize(across(starts\_with("gdp"), sum, na.rm = TRUE), .groups = "drop")%>%

mutate(gdpgrowth=gdp2007 -gdp1952) %>%

arrange(desc(gdpgrowth) %>%

top\_n(5) %>%

ggplot(aes(x=country, y=gdpgrowth, fill=country)) +

geom\_col()

If you want to filter a set of values use %in% like the following

filter(country %in% c("Brazil", "China", "India", "Indonesia", "United States")