* R VISUALIZATION MODULE
* **Dplyr**
* s<- heights %>% filter(sex==”Male”) %>% summarize(average = mean(height), standard\_deviation = sd(height) )
* the resulting table stored in s is a dataframe : we can access the components of a dataframe with the $ sign.
* s$average
* s$standard\_deviation
* -> We can compute any summary that operates on vectors and returns a single value. Example :
* heights %>% filter(sex==”Male”) %>% summarize(median=median(height), minimum = min(height),
* maximum = max(height) )
* The . placeholder
* We are going to learn how to make **dplyr** functions return vectors as opposed to dataframes, using the US muders as an example.
* We already used **dplyr** to add a murder rate column :
* murders <- murders %>% mutate(murder\_rate = total / population \* 10^6)
* summarize(murders,average = mean(murder\_rate)
* In this computation, we are counting the small states the same way we are the large states. When we compute the average US murder rate we need to account for the large states more than the small states.
* US\_murder\_rate <- murders %>% summarize(rate = sum(total) / sum(population)\*10^6)
* Suppose we want to use a function that requires just a numeric value.
* We can’t us the US\_murder\_rate object because it is a dataframe. Most of **dplyr** functions return a dataframe.
* Useful trick to access value stored in data that is being piped with the %>% sign is to use the . character.
* US\_murder\_rate %>% .$rate
* Think of the . as a placeholder for the data that is being piped through the pipe.
* Because US\_murder\_rate is a dataframe, we can access its columns using a .$ sign and then the column name.
* For now, we will only use the . to produce numeric vectors from pipelines constructed with **dplyr**.
* GROUP BY
* Very common operation and data exploration is to first split data into groups and then compute the summaries for each group.
* Example : We want to compute the average and standard deviation for men and women heights separately.
* The **group by** function helps us do this :
* Heights %>% group\_by(sex) %>% summarize( average = mean(height),
* Standard\_deviation = sd(height) )
* The result will be a dataframe that has two rows now : one for men, one for women, with their respective average and standard deviation.
* Another example :
* murders %>% group\_by(region) %>% summarize(median\_rate = median(murder\_rate))
* Result is a dataframe with 4 rows, one for each region.
* SORTING DATA TABLES
* When examining a dataset, it is often convenient to sort the table by the different columns.
* The function **arrange** in **dplyr** very useful for this purpose.
* Example : We want to order the states by their population size.
* murders %>% arrange(population) %>% head()
* The default behavior of arrange is to sort the data in ascending order.
* The function **desc()** transforms a vector to be in descending order.
* Example : We want to get the states ordered by murder\_rate from the highest to the lowest
* murders %>% arrange(desc(murder\_rate)) %>% head()
* We can also do nested sorting to break the ties in the first columns by values of a second, third and/or fourth column.
* Example : We want to order the states by region and then within each region by murder\_rate
* murders %>% arrange(region, murder\_rate) %>% head()
* Another useful function is **top\_n()**.
* murders %>% top\_n(10, murder\_rate)
* Shows the 10 states with the highest murder\_rate not in order !
* To get ordered data :
* murders %>% arrange(desc(murder\_rate)) %>% top\_n(10)
* CASE STUDY : TRENDS IN WORLD HEALTH AND ECONOMICS
* The goal here is to demonstrate how simple **ggplot** and **dplyr** code can create insightful and aesthetically pleasing plots that helps us better understand trends in world health and economics.
* Data from gapminder : an organization dedicated to educating the public by using data to dispel common myths about the developing world.
* Two questions :
* - Is it fair to say the world is divided into rich and poor ?

- Has income inequality worsened during the last 40 years ?

* + Gapminder dataset

We can access the gapminder data using this code :

library(dslabs)

data(gapminder)

We start by testing our knowledge regarding differences in child mortality across different countries.

To answer this question :

gapminder %>% filter(year==2015 & country %in% c(“Sri Lanka”, “Turkey”)) %>% select(country, infant\_mortality)

* + Life expectancy and Fertility Rates
  + FACETING
* We can easily plot the 2012 data in the same way we did for 1962.
* But for comparison, side by side plots are preferable.
* In **ggplot**, this functionality is called **faceting** : we stratify the data by some variable and make the same plot for each strata.
* The function **facet\_grid()** ensures this functionality and can be added as a layer. It lets you facet up to 2 variables, using columns for the first one and rows for the second one.Rows and columns arguments are separated by a ~.
* filter(gapminder,year in c(1962,2012)) %>% ggplot(aes(fertility,life\_expectancy,col=continent))
* + geom\_point()
* + facet\_grid(continent~year)
* If we only want to facet by the year (1 variable):
* filter(gapminder,year in c(1962,2012)) %>% ggplot(aes(fertility,life\_expectancy,col=continent))
* + geom\_point()
* + facet\_grid(**.~year**)
* Analysis :
* The majority of the countries have moved from the developing world cluster to Western world one.
* They went from having short lifespans and large families to having longer lifespans and smaller families.
* Asia includes several countries that made great improvements inn the last 40 to 50 years.
* To explore how this transformation happened, we can make the plot for several years (1962,1972,1982,1992,2012)
* We might want the plots across different rows and columns : the function **facet\_wrap()** does this. It automatically wraps the series of plots so most displays has viewable dimensions.
* filter(gapminder,year in c(1962,1972,1982,1992,2012)) %>% ggplot(aes(fertility,life\_expectancy,col=continent))
* + geom\_point()
* + facet\_wrap(year)
* Analysis : this function clearly shows us that Asian countries have made great improvements throughout the years.
* When using facet\_wrap(), the range of the axes are determined by the data shown in all plots to keep the range consistent across them and facilitate the comparison.
  + TIME SERIES PLOTS
* The visualizations we have just seen clearly demonstrates that data no longer supports the Western vs developing world view.
* New questions emerge :
* - Which countries are improving more ? less ?
* - Was improvement constant during the last 50 years or was there an acceleration during a given period ?
* To answer these questions, we rely on **time series plots**.
* TSPs have time in x-axis and an outcome or measurement on the y-axis.
* Example : TSP for the US fertility rate (obtained with the geom\_point layer)
* The trend is not linear at all.
* When points are regularly spaced and densily packed (as in the example), we can create curves by joining points with lines.
* **Geom\_line()** function helps us do this.
* gapminder %>% filter(country==”United States”) %>% ggplot(aes(year,fertility))
* + geom\_line()
* It is particularly helpful when we look at two or more countries.

To let ggplot know that there are two curves that need to be made separately, we assign each point to a group, one for each country **through mapping**.

countries <- c(“South Korea”, “Germany”)

gapminder %>% filter(country %in% countries) %>% ggplot(aes(year,fertility,group=country))

+geom\_line()

However, we don’t know which line goes with which country. To see this, we can use color for example. The useful side effect of using color is that ggplot automatically groups the data by color value.

gapminder%>%filter(country%in%countries)%>%ggplot(aes(year,fertility,group=country,col=country))+geom\_line()

We get two lines, each with a color, and a legend has been added by default.

Analysis : This plot clearly shows how South Korea’s fertility rate dropped drastically during the 60s and 70s. And by the 1990s, it had a similar fertility rate as Germany.

For TSPs, we recommend labeling the curves rather than using legend as we did in the example. Labelling is usually preferred to legends. However, legends are easier to make and appear by default in many of ggplot’s functions.

Problem : How to add labels to a TSP ?

Solution : We demonstrate how to do this by using the life\_expectancy data.

We define a data table with the data locations:

labels <- data.frame(country=countries, x = c(1975, 1965), y=c(60, 72))

And then we use a second mapping just for the labels.

gapminder %>% filter(country %in% countries) %>% ggplot(aes(year, life\_expectancy, col = country) + geom\_line() + geom\_text(data=labels,aes(x,y,label=country),size = 5)

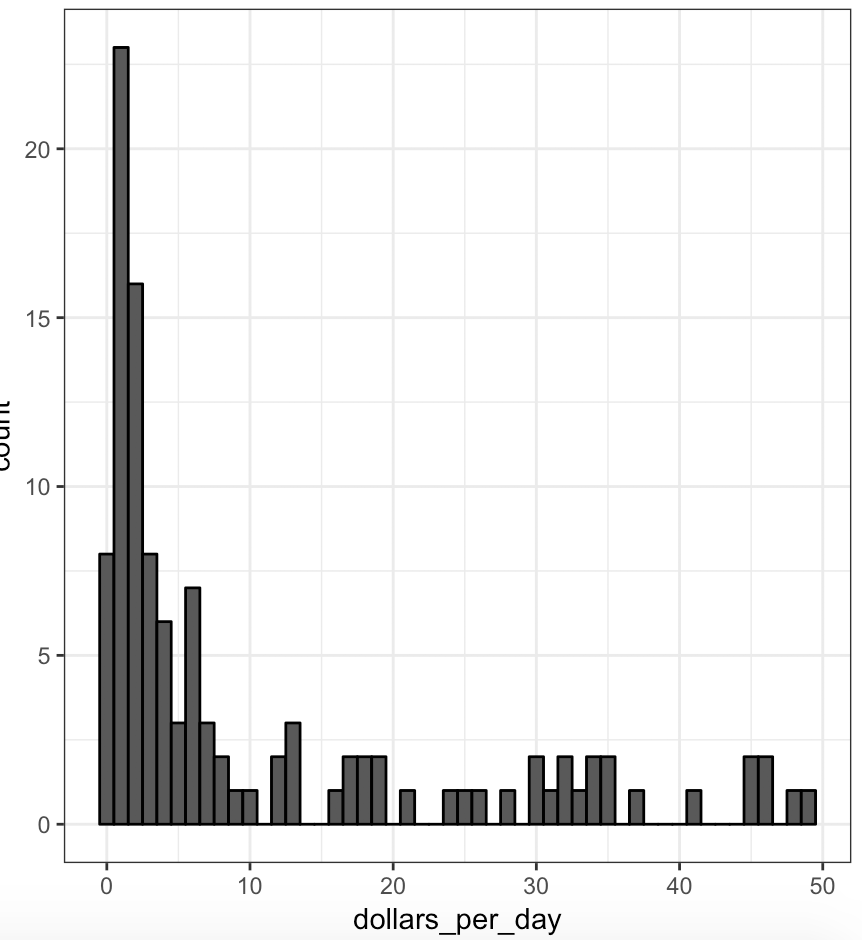
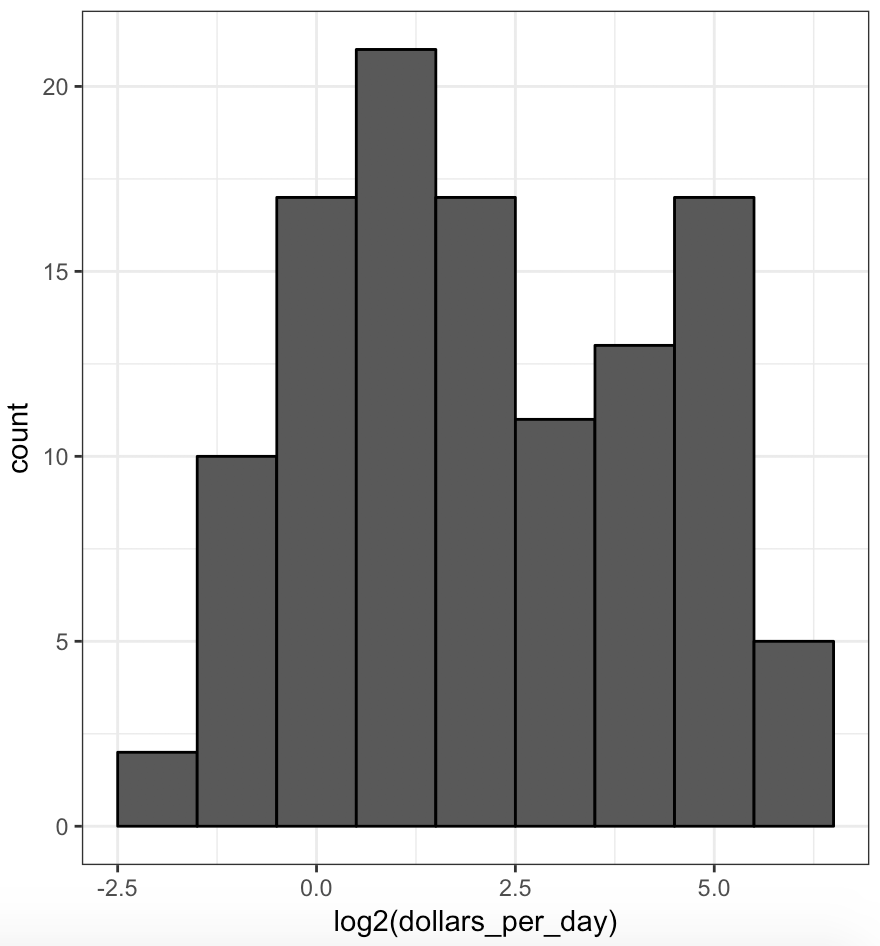
Analysis : We can see how the plot shows how an improvement in life\_expectancy followed the drops in fertility\_rates.

While in 1960, the Germans lived more than 15 years more on average than South Koreans by 2010 the gap is completely closed and the life expectancy in South Korea is slightly higher than in Germany.

Another commonly held notion is that wealth distribution across the world has worsened during the last decades.

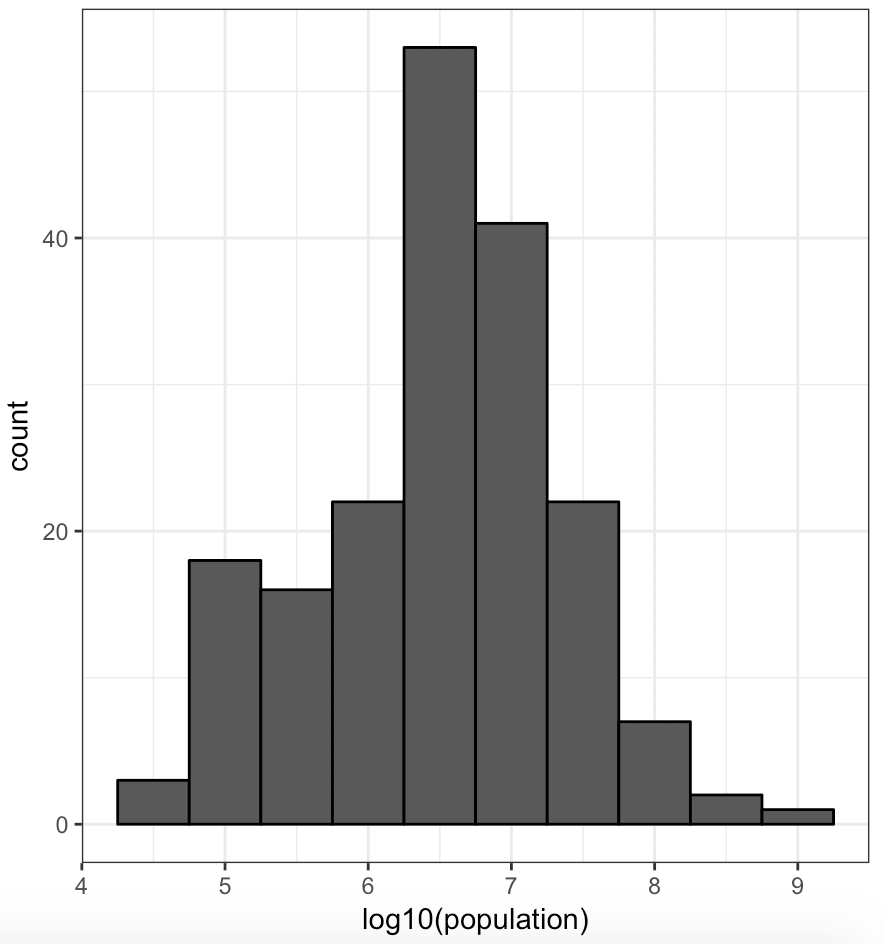
When general audiences are asked if poor countries have become poorer and rich countries have become richer, the majority answers yes.

By using histograms, density and box plots, we will be able to understand if this is in fact the case.

* + TRANSFORMATIONS
* Transformations can be very useful to better understand distributions.
* As an example, we look at income in the gapminder dataset. The gapminder dataset includes a column with the country’s Gross Domestic Product or GDP.
* GDP measures the market value of goods and services by a country in a given year.
* The GDP/person is often used as a rough summary of how rich a country is.
* Here we divide this quantity by 365 to obtain a more interpretable measure dollars per day.
* Using current US dollars as a unit, a person surviving on an income of less than 2$ per day, for example, is defined to be living in absolute poverty.
* gapminder %>% mutate(dollars\_per\_day = gdp/population/365)
* GDP values in our dataset are adjusted for inflation and represent current US dollars. These values are meant to be comparable across the years. Also note that these are country averages and that within each country there might be much variability.
* To draw the histogram of per day incomes from 1970 :
* gapminder %>% filter(year==1970 & !is.na(gdp)) %>% ggplot(aes(dollars\_per\_day)) + geom\_histogram(binwidth =1, color=”black”)
* ****
* Analysis : For the majority of countries, averages are below $10 a day. However, the majority of the x-axis is dedicated to 35 countries with averages above $10/day.
* Might be informative to be able to see how many countries make on average about :
* - $1/day – extremely poor
* - $2/day – very poor
* - $4/day – poor
* - $8/day – about the middle
* - $16/day – well-off
* - $32/day – rich
* - $64/day – very rich
* These changes are multiplicative. And here we introduce **the log transformations**.
* Log transformations change multiplicative changes into additive ones. Using the base 2 for example means that every time a value doubles, the log transformation increases by 1.
* So to get the distribution of the log2 transformed values, we simply transform the data and use the same code.
* gapminder %>% filter(year==1970 & !is.na(gdp)) %>% ggplot(aes(log2(dollars\_per\_day))) + geom\_histogram(binwidth =1, color=”black”)
* ****
* In this plot, we see something new : we see two clear bumps.
* Let’s introduce some commonly used statistical language.
* In statistics, these bumps are referred to as **modes**. The **mode of a distribution** is the value with the highest frequency.
* The mode of a normal distribution is the average.
* When a distribution is similar to the one observed in the plot, ie it doesn’t monotonically decrease from the mode, we call the location where it goes up and down **local modes** and we say that the distribution has multiple modes.
* The histogram here suggests that in 1970, country income distribution have two modes : one at about $2/day ( 1 in log2 scale) and another at about $32/day (5 in the log2 scale).
* The bimodality is consistent with a dichotomous view of the world made up of countries with average incomes less than $8/day (3 on the log2 scale) and countries above that.
* Let’s explain how we choose the base for the plot (histogram).
* In the histogram we just saw we choose log2**.** Other common choices are the natural log10. In general, **we do not recommend using the natural log for data exploration in visualization** because while we can quickly compute in our mind 2 to the 2, 2 to the 3, 2 to the 4 or 10 to the 1, 10 to the 2, 10 to the 3 etc…(log2 and log10 scales), it is not easy to compute E to the 2, E to the 3, etc…
* In the dollar per day example, we use base2 instead of base10 because the resulting range is easier to interpret. The range of the values being plotted started from about 0.3 and ended around 50.
* In base 10, this turns to a range that includes very few integers, just 0 and 1. In base 2, our range includes all integer between -2 and 5.
* Note that it is easier to compute 2 to the x and 10 to the x when x is an integer. So we prefer to have more integers in the transformed scale.

Another consequence of a limited range is that choosing the bin width is more challenging. With log base 2, we know that a bin width of 1 will translate to bins with range x to 2 to the x.

As an example in which base 10 makes more sense than base 2, consider population size. Using the base 10 makes more sense here since the range of the data varies from 45,000 to about 800 million.

****

Here is a histogram of the world population in 1970 if we transform the values with log base 10. Looking at the scale knowing that we’re in base 10, we can quickly determine that country population ranges from about 40,000 to about a billion.

How to use log transformations in plots?

There are two ways we can use log transformations : we can log the values before plotting them or we can use log scales in the axis.

Both approaches are useful and have different strengths.

If we log the data, we can more easily interpret intermediate values in the scale.

For example, if we use a scale that […]

However, the advantage of using log scales is that we see the original values on the axis. So this has an advantage because we see the original values displayed in the plot which makes it very easy to quickly see what numbers we’re actually dealing with.

For example, in the histogram, when we see $32/day, instead of 5log base $2/day.

Now let’s review how we make plots where the scales have been log transformed. We already learned this. We learned the **scale\_x\_continuous** function.

So we want to remake the histograms that we already made but now using scales that have been transformed. We simply add a layer using the scale\_x\_continuous function, and we no longer transform the data before plotting it.

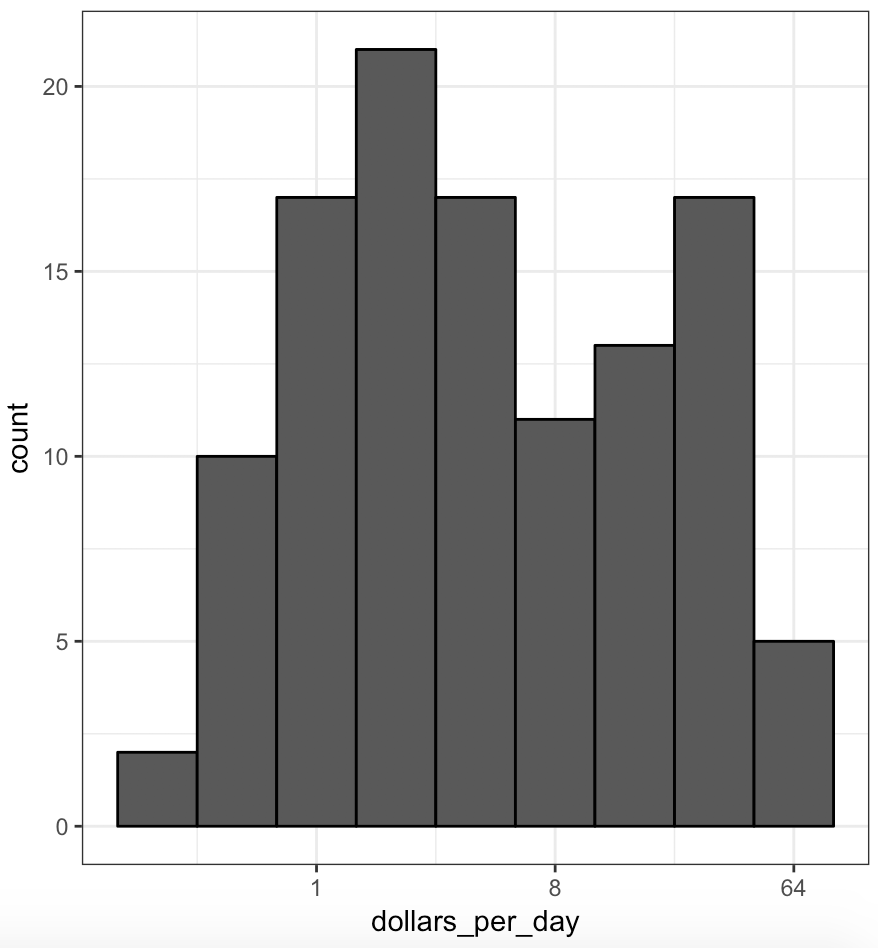
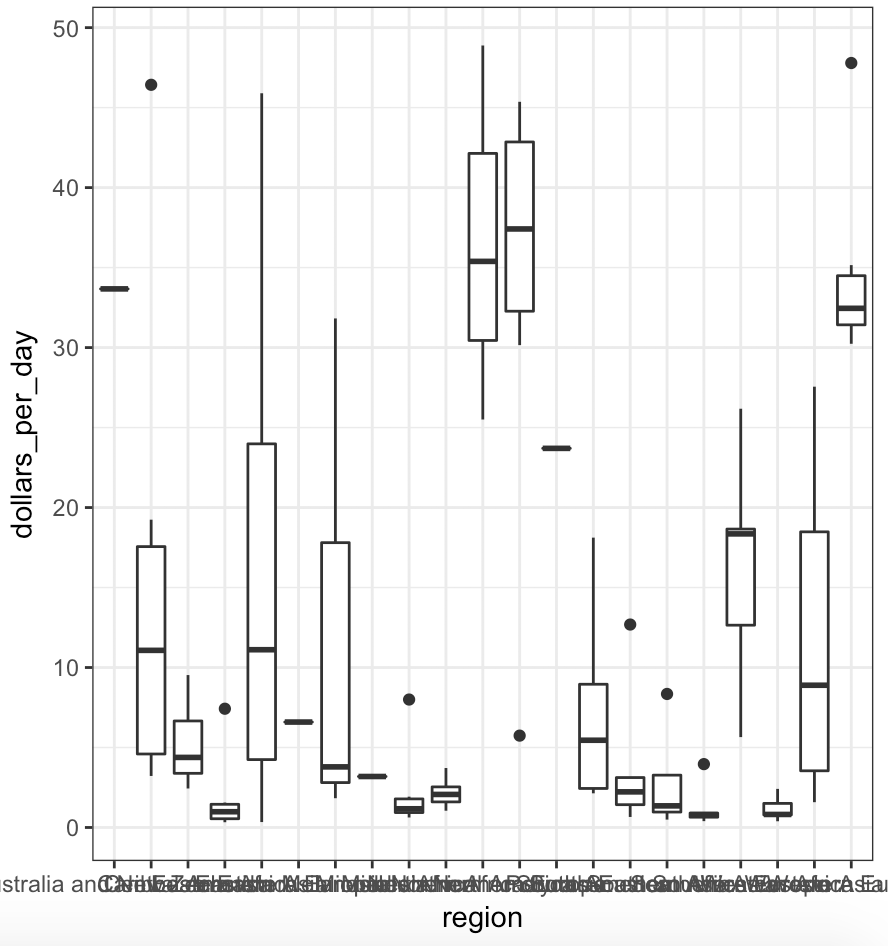
Code looks like this :

gapminder %>% filter(year == past\_year & !is.na(gdp))

%>% ggplot(aes(dollars\_per\_day)) +

geom\_histogram(binwidth = 1, color = “black”) +

scale\_x\_continuous(trans = “log2” )

* 
* Notice that the histogram looks exactly the same. The difference is that in the scales in the x-axis, instead of seeing the log-values, we see the original values in a log scale. So we can quickly interpret what that means in terms of dollars per day.
* STRATIFY AND BOXPLOT
* The histogram showed us that the income distribution values show a dichotomy.
* However, the histogram doesn’t show us if the two groups of countries are west versus the developing world.
* To see distributions by geographical regions, we first stratify the data into regions, and then examine the distribution for each.
* Now, the number of regions is large in this case (22) :
* length(levels(gapminder$region))
* Because of this, looking at histograms or smooth densities for each is not very useful in this case.
* instead, we can stack box plots next to each other. To do this, we simply write this code :
* p <- gapminder %>% filter(year==1970 & !is.na(gdp)) %>% ggplot(aes(region,dollars\_per\_day))
* p + geom\_boxplot()
* 

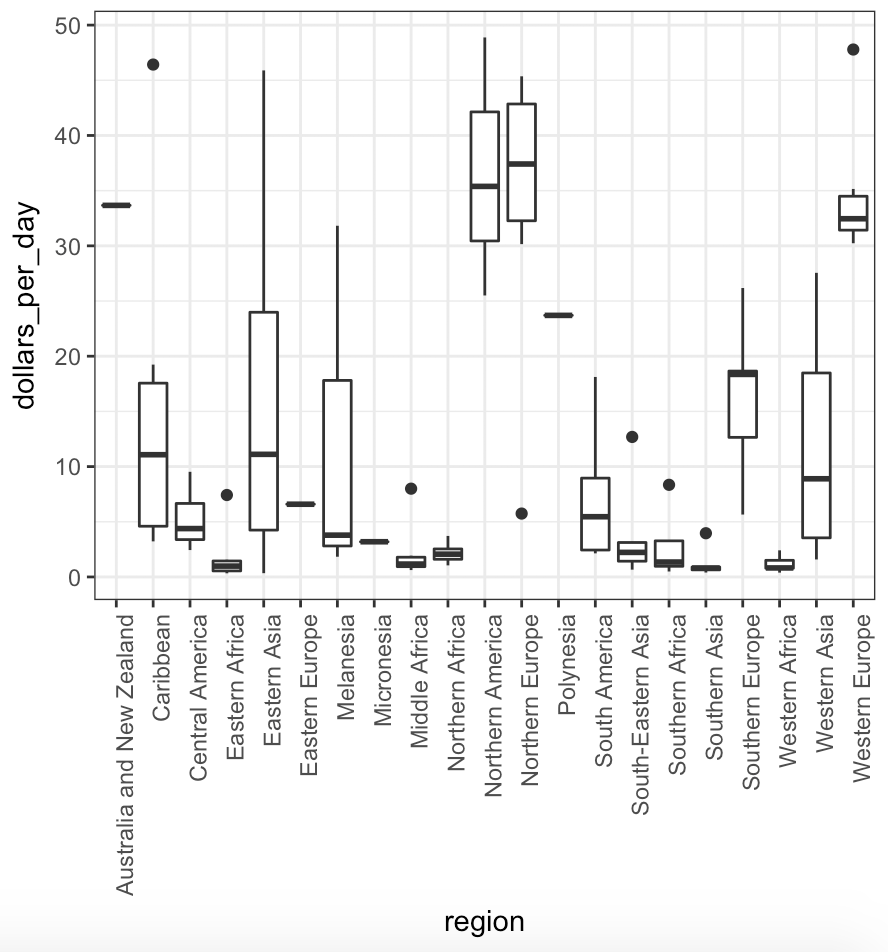
Now, note that we can’t read the region names because the default boxplot behavior is to write the labels horizontally and here we run out of room.

We can easily fix this by rotating the labels.

Consulting the documentation, we find that we can rotate the names by changing the theme through ‘*element\_text*’ . The ‘*hjust = 1*’ justifies this text so that it’s next to the axis.

p + geom\_boxplot()

+ theme(axis.text.x = element\_text(angle = 90, hjust = 1))



Now, we can read the names.

We can already see that there is indeed a west verses the rest dichotomy : if you look closely at the box plots that are high, we see that they’re North America, Northern Europe, Australia, New Zealand and Western Europe.

There are a few more adjustments we can make to this plot to help relay this message.

First, it helps to order the regions in some order that is not alphabetical : the function that is going to help us achieve this or the **reorder** function.

This function lets us change the order of the levels of a factor variable based on a summary computed on a numeric vector.

Let’s understand how the reorder function works using a simpler example.

fac <- factor(levels(“Asia”,”Asia”,”Asia”,”West”,”West”))

By turning this vector into a factor, the levels of this factor are ordered alphabetically. This is the default in R.

“Asia” is the first level and “West” is the second level.

Suppose that each of these elements of the original vector are associated with a value from a numerical vector.

value <- c(10,11,12,6,4)

Let’s suppose that we want to order the levels based on the mean of these numbers. In this case, “West” has a lower mean, it’s the mean of 12,6 and 4, compared to the mean of Asia, which is the mean of 10 and 11.

Now, if we use the function reorder using the function mean to summarize the values:

fac <- reorder(fac,value,FUN = mean)

We can see that the new factor has levels ordered differently. Now “West” is the first one because it has smaller mean value of the value vector.

The first thihng we are going to do to improve our plot is to simply reorder the regions by their median income level.

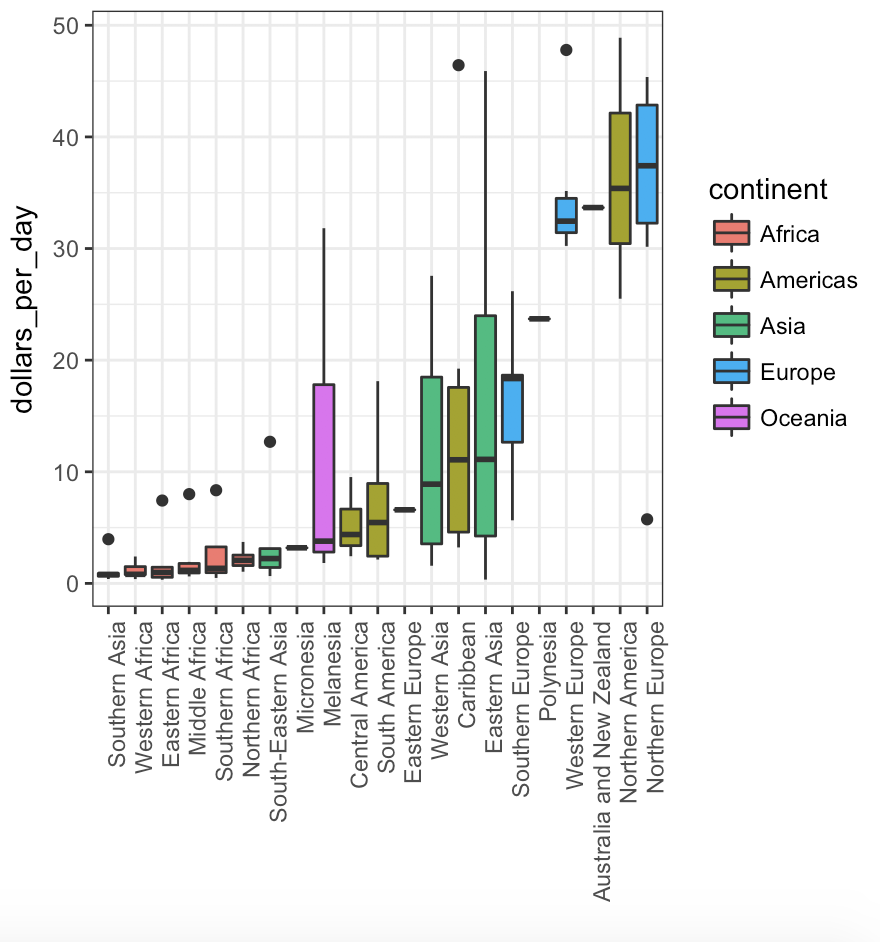
To achieve this, we write the same code as before but we add to mutate that changes region to a new factor where the levels are reordered.

* p <- gapminder %>% filter(year==1970 & !is.na(gdp)) %>%
* **mutate(region = reorder(region,dollars\_per\_day, FUN=median))** %>% ggplot(aes(region,dollars\_per\_day, fill = continent))

+ geom\_boxplot()

+ theme(axis.text.x = element\_text(angle = 90, hjust = 1))

+ xlab(“”)



Now we can see that the box plots are ordered by their medium value. And we quickly see there’s four box plots that stand out at the end, the four highest ones which are Western Europe, Australia and New Zealand, Northern America and Northern Europe.

Now there is another change we made to the plot to help convey this message, we used color to show the continents.

Remember that regions are parts of continents.

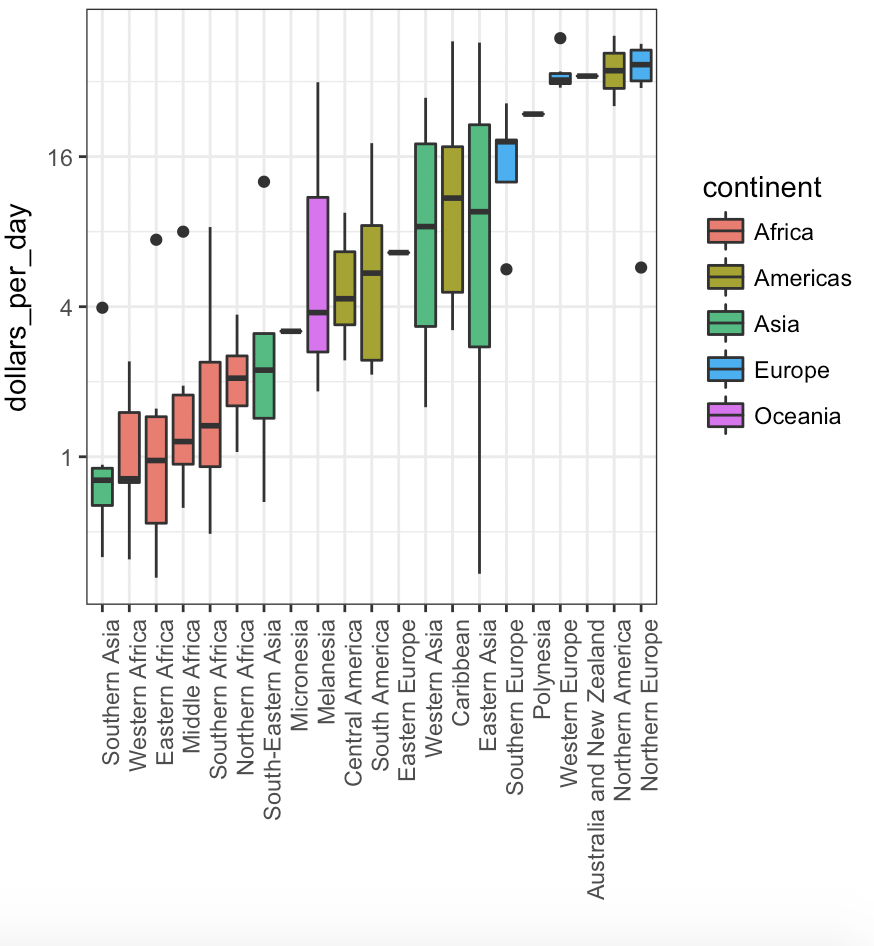
To add color to define the different continents, we use the **fill argument** in the aesthetic mappings of ggplot. And now each continent gets its color.

By adding the color to the continent, we observe that the blue box plots representing European countries are towards the right while the red countries representing African countries are to the left.

Another change we can make to this plot to help us see the data a little bit better, is to change the scale through the log scale so we add the scale\_y\_continuous layer and we use the log2 transformation.

* p <- gapminder %>% filter(year==1970 & !is.na(gdp)) %>%
* **mutate(region = reorder(region,dollars\_per\_day, FUN=median))** %>% ggplot(aes(region,dollars\_per\_day, fill = continent))
* + geom\_boxplot()
* + theme(axis.text.x = element\_text(angle = 90, hjust = 1))
* + xlab(“”)

+ scale\_y\_continuous(trans = “log2”)



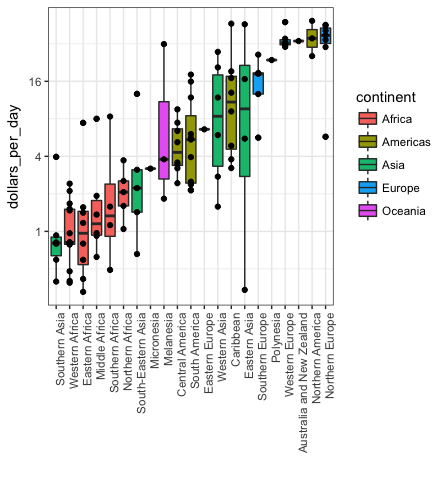
And now it helps us see the differences between the countries with the lower income. For example, we see a difference now between the African continent which is in red and Asia which is in green.

The last change we can make to this plot to make it tell the story a little bit better is to show the data.

In many cases we don’t show, the individual points, because it adds too much clutter to the plot and it obfuscates the message.

But in this particular example, we don’t have too many points. So we can add a layer of points by adding the geom\_point() layer.

p + geom\_point(show.legend = FALSE)



COMPARING DISTRIBUTIONS

The exploratory data analysis we have conducted has revealed two characteristics about average income distributions in 1970.

Using a histogram, we found a bimodal distribution with the most relating to poor and rich countries.

Then, by stratifying by region and examining box plots, we found that rich countries were mostly in Europe and Northern America along with Australia, New Zealand, and then the poor countries were mostly in the rest of the world.

So we are going to define a vector that defines the regions in the West.

west <- c("Western Europe","Northern Europe","Southern Europe","Northern America","Australia and New Zealand")

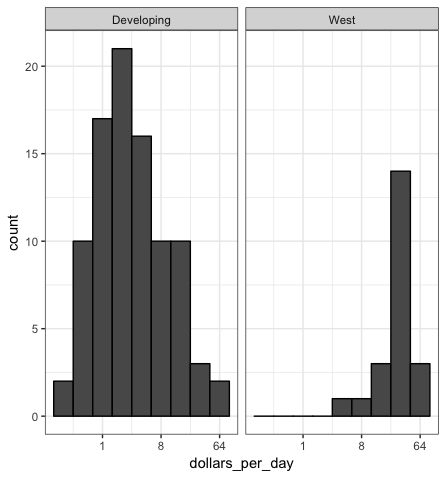
Now we want to focus on comparing the differences in distribution across time.

We start by confirming that the bimodality observed in 1970 is explained by a west versus developing world economy.

We do this by creating a histogram for the group previously defined.

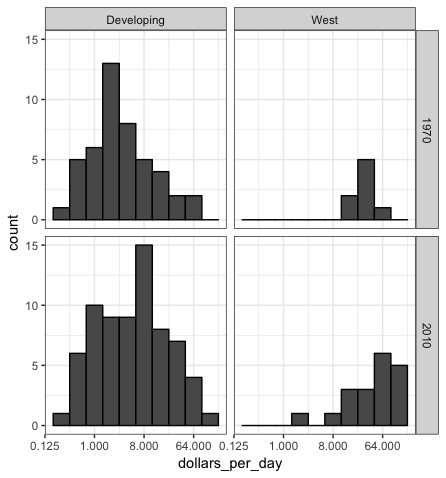
Note that we create two groups with and **ifelse** inside a mutate and then we use facet\_grid to make histograms for each group.

gapminder %>% filter(year==1970 & !is.na(gdp)) %>% mutate(group = ifelse(region%in%west,"West","Developing")) %>% ggplot(aes(dollars\_per\_day))+geom\_histogram(binwidth = 1, color = "black") + scale\_x\_continuous(trans = "log2")+facet\_grid(.~group)



We immediately see that the countries in the West have higher incomes. The histogram is shifted to the right. Countries in the developing world are shifted towards the left.

Now we’re ready to see if the separation is worse today than it was 40 years ago. We do that by faceting by both region and year.



Now we can see the histogram again for four different groups. When we look at this figure, we can see that the developing world has shifted to the right more than the West, meaning that the income distribution of the developing countries has gotten closer to those from the West.

Before we interpret the findings of this plot further, we note that there are more countries represented in the 2010 histograms than in the 1970s ones. The total account is larger. One reason for this is that several countries were founded after 1970.

For example, the Soviet Union turned into several countries, including Russia and Ukraine there in the 90s.

Another reason is that data is available for more countries in 2010 than in 1970.

So we are going to remake the plots, but only using countries which data is available for both years. We’re going to use this very simple code :

country\_list\_1 <- gapminder %>% filter(year ==1970 & !is.na(dollars\_per\_day)) %>% .$country

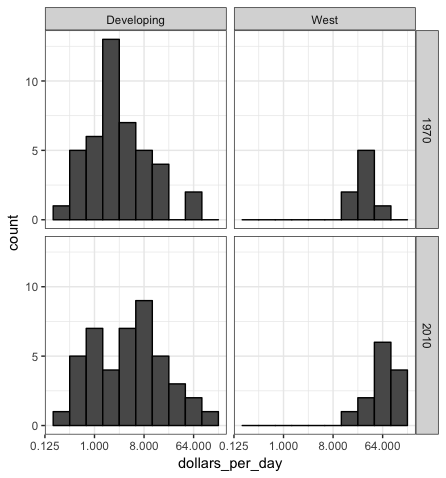
country\_list\_2 <- gapminder %>% filter(year ==2010 & !is.na(dollars\_per\_day)) %>% .$country

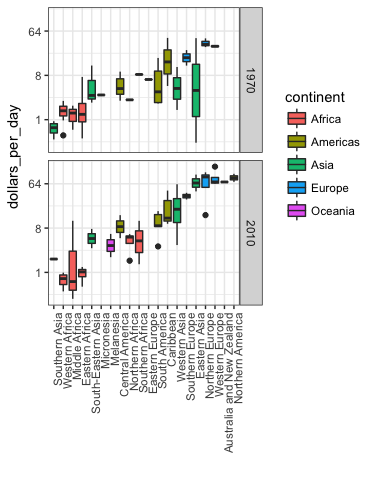
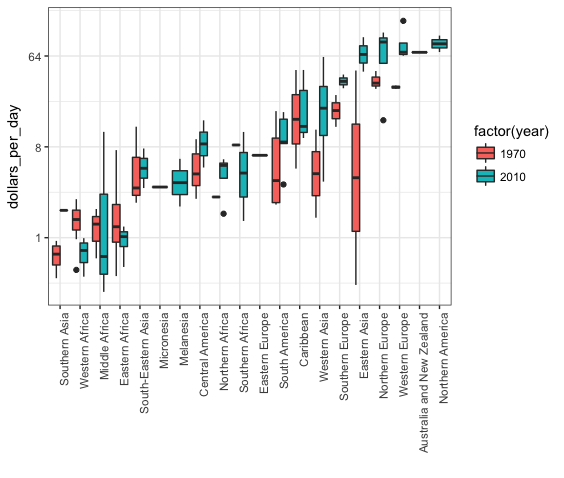
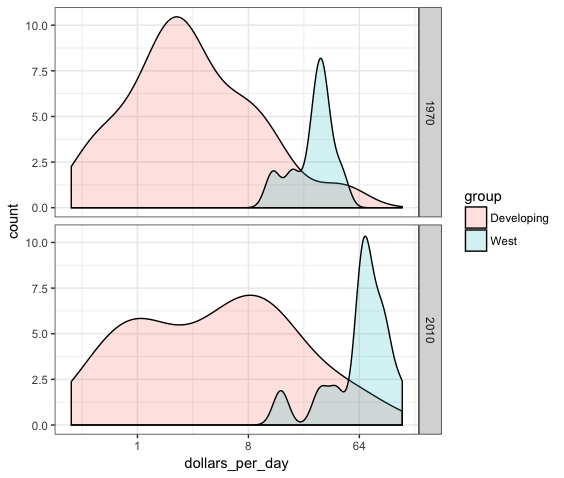
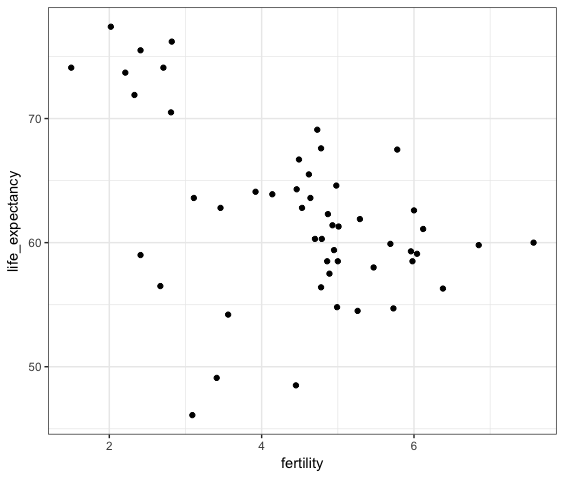
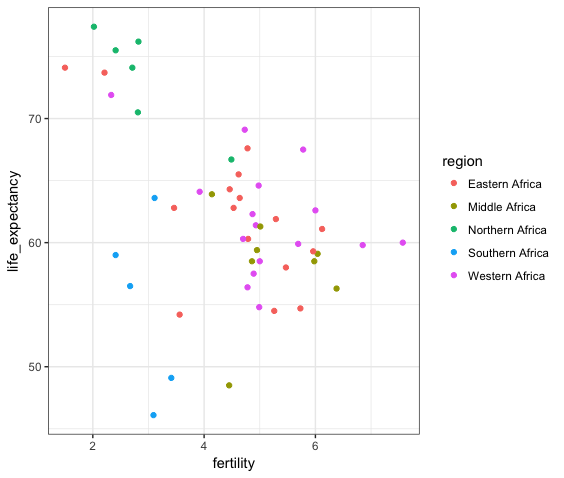
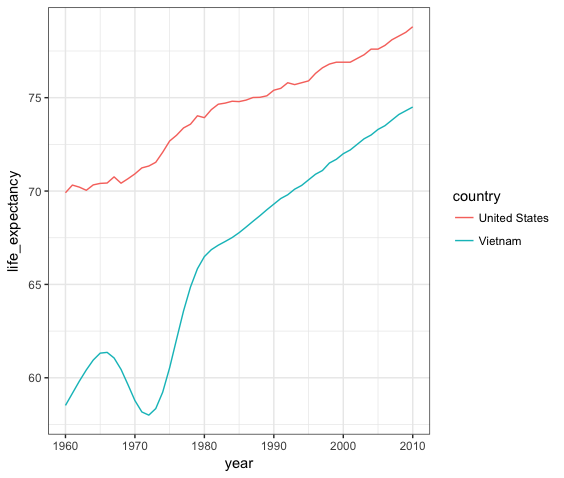
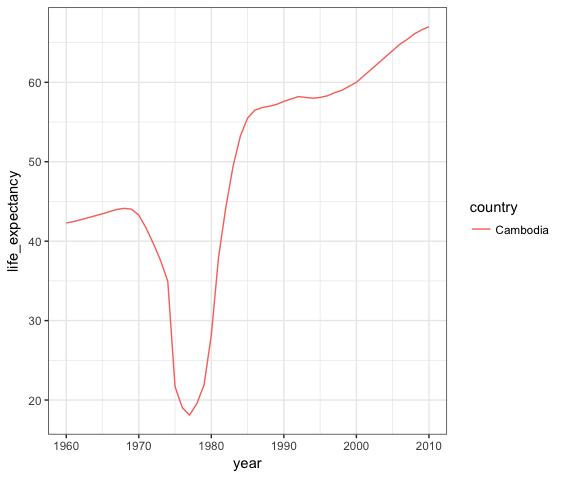
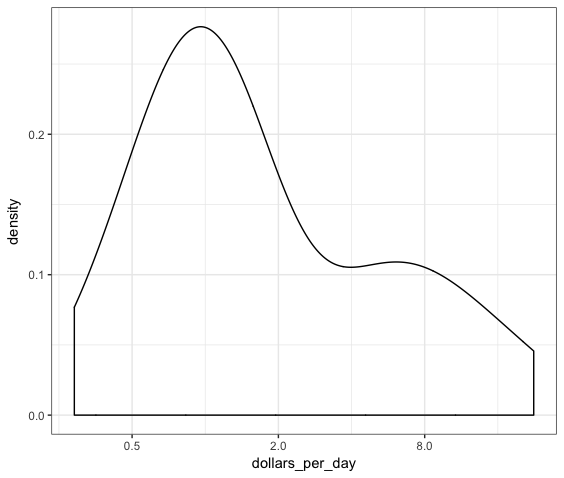
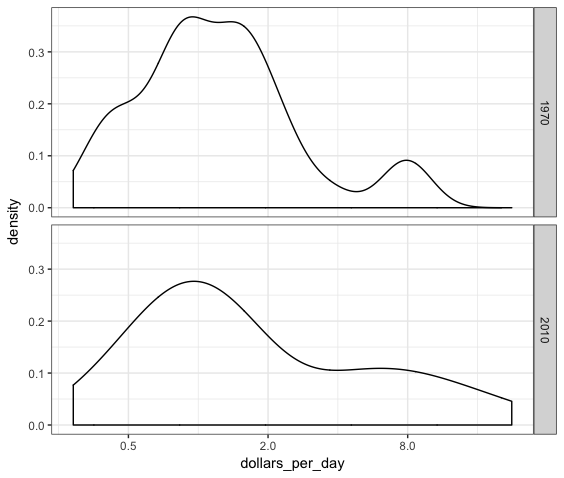
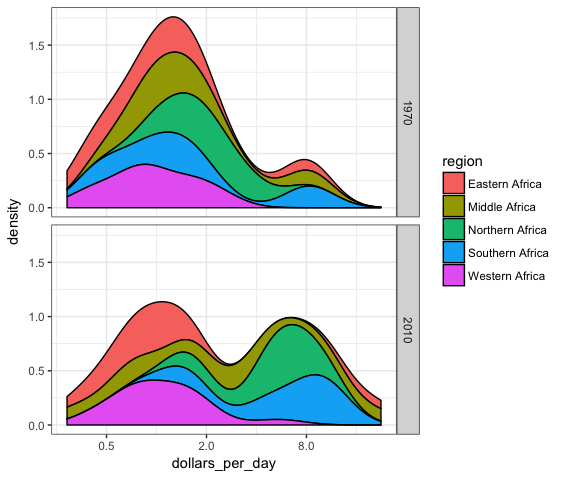
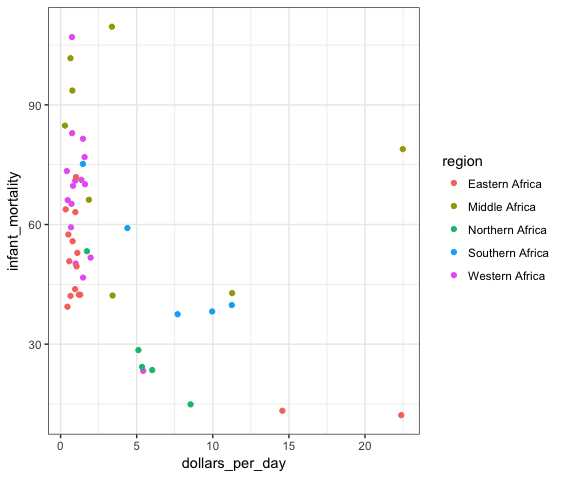
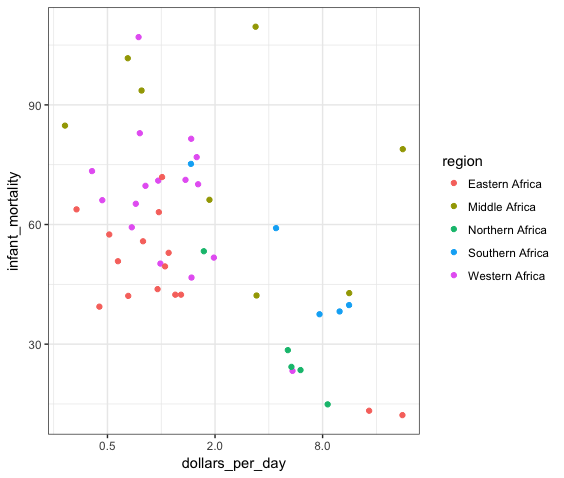
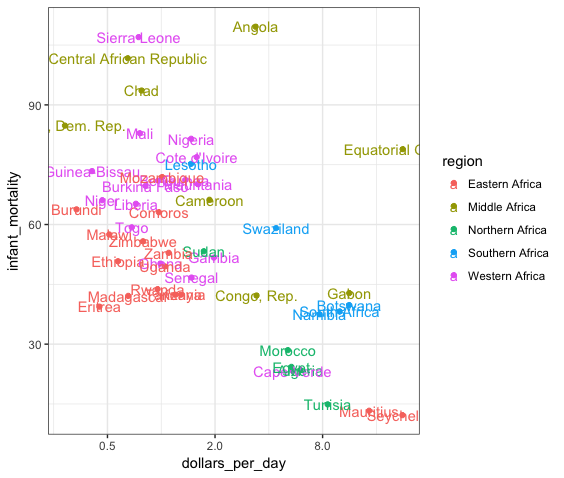
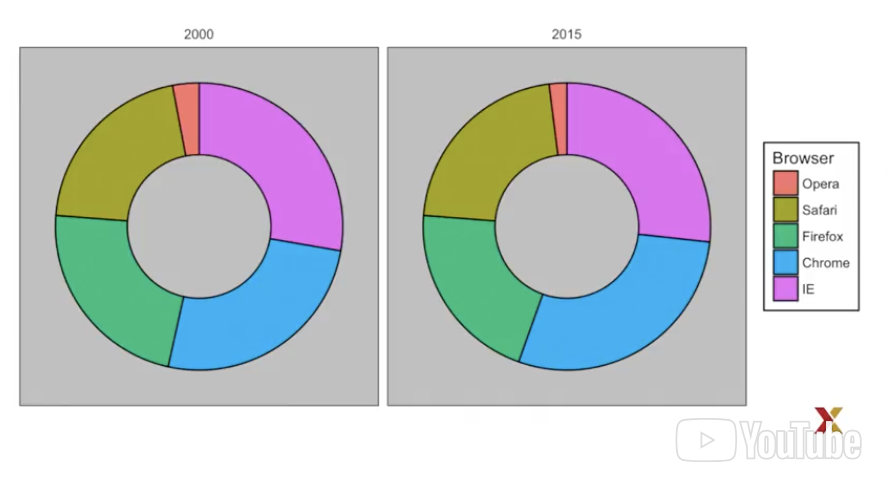
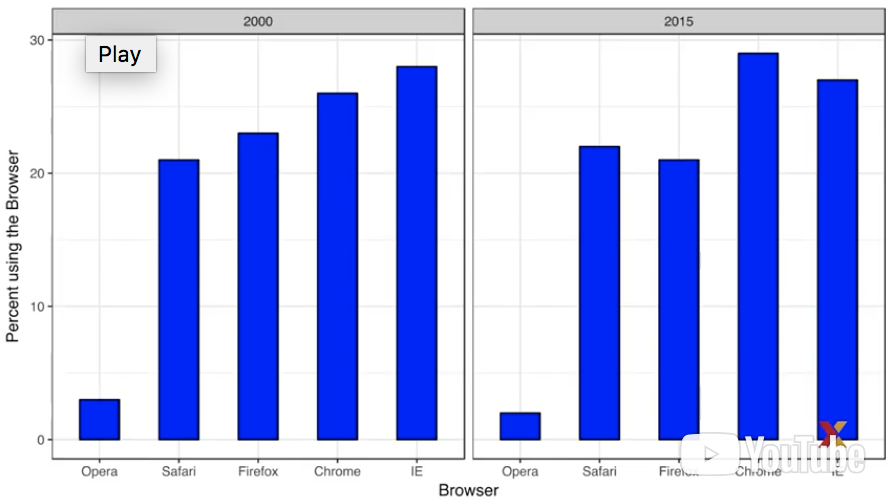
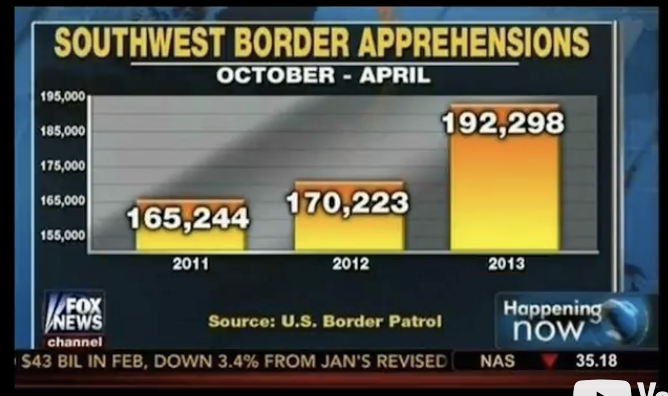
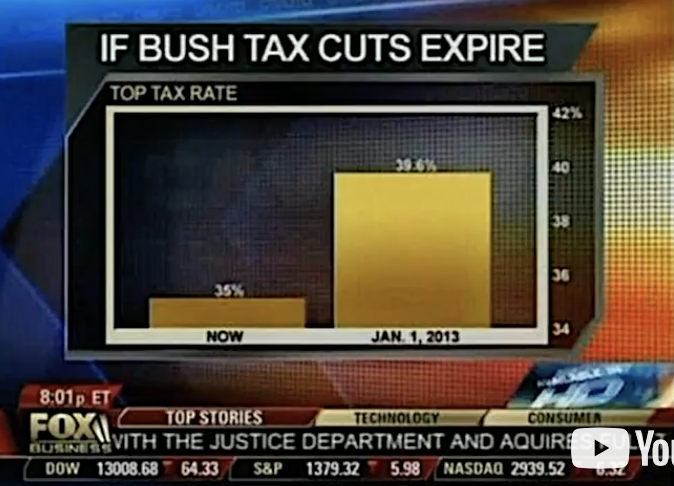
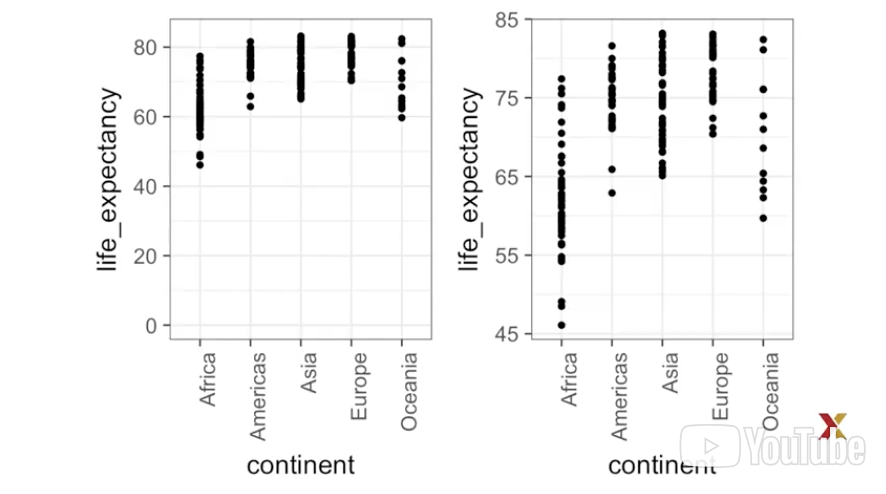
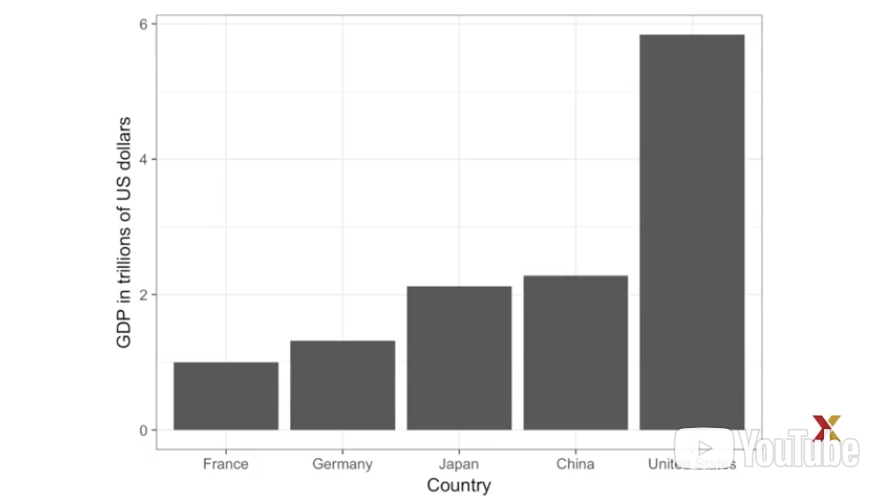
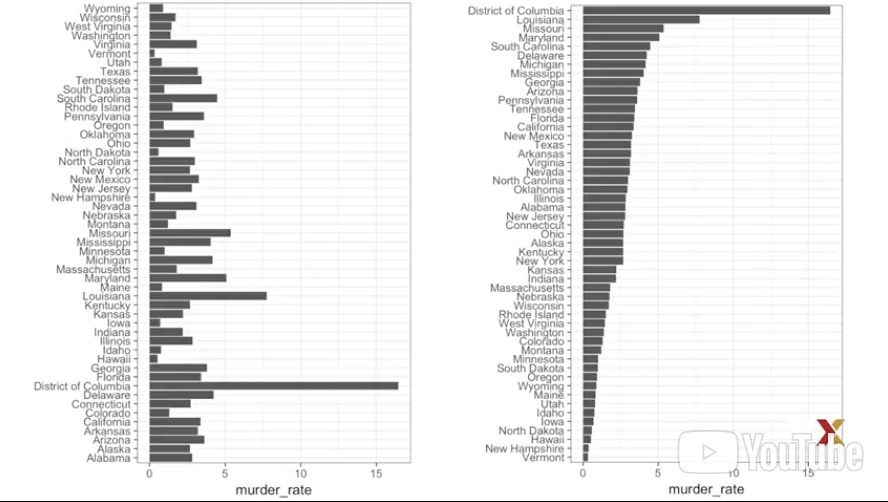
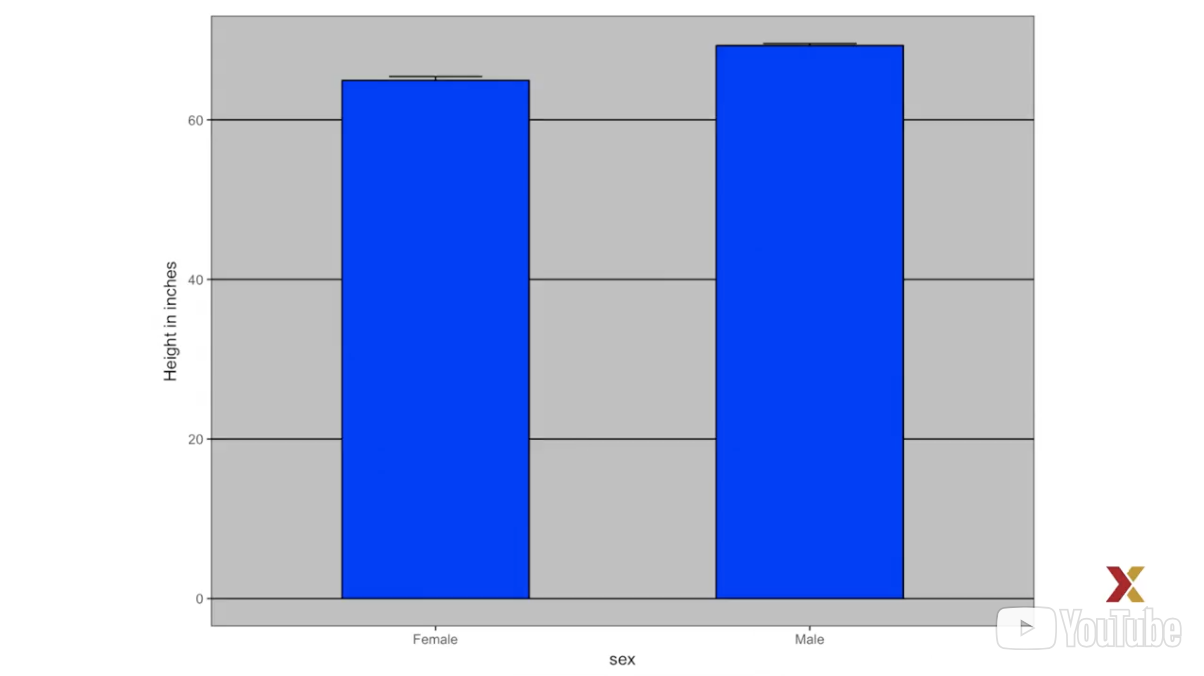
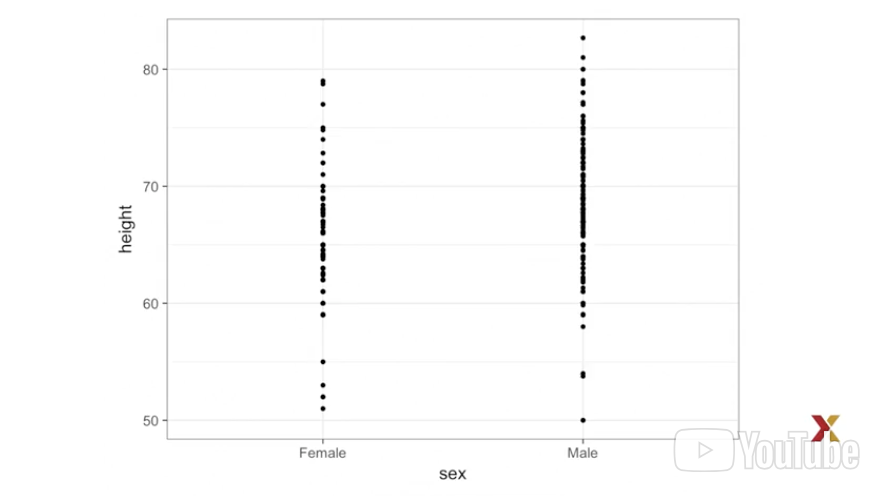
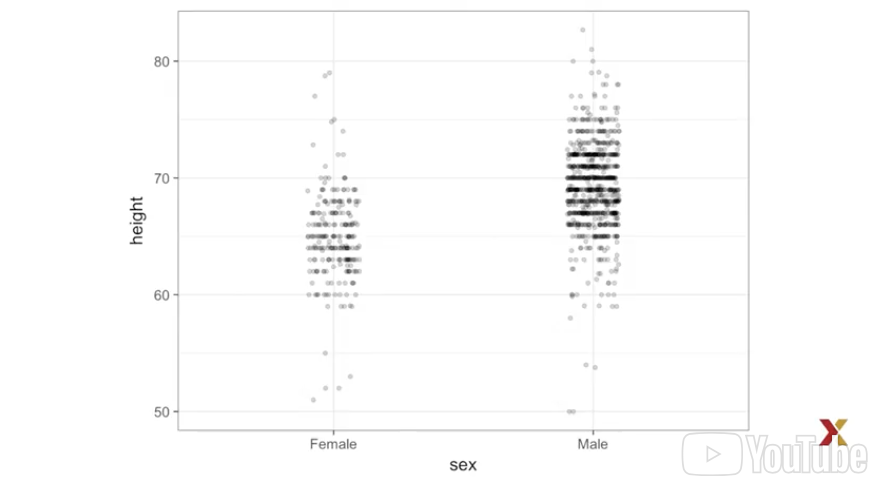
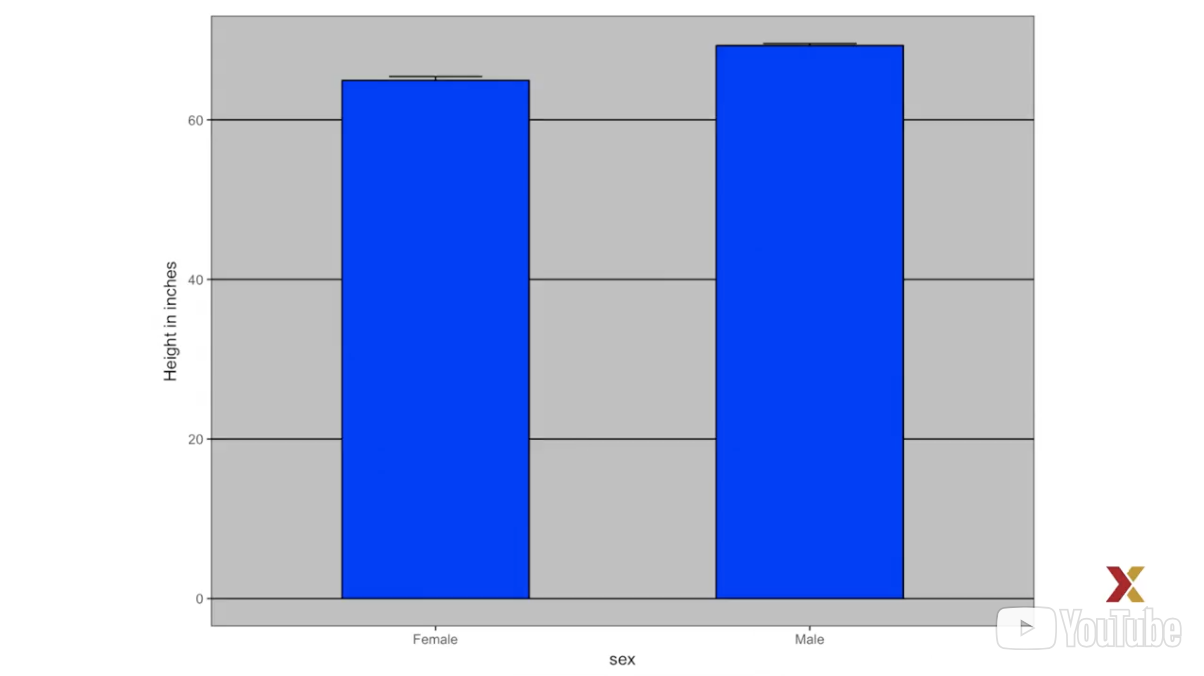
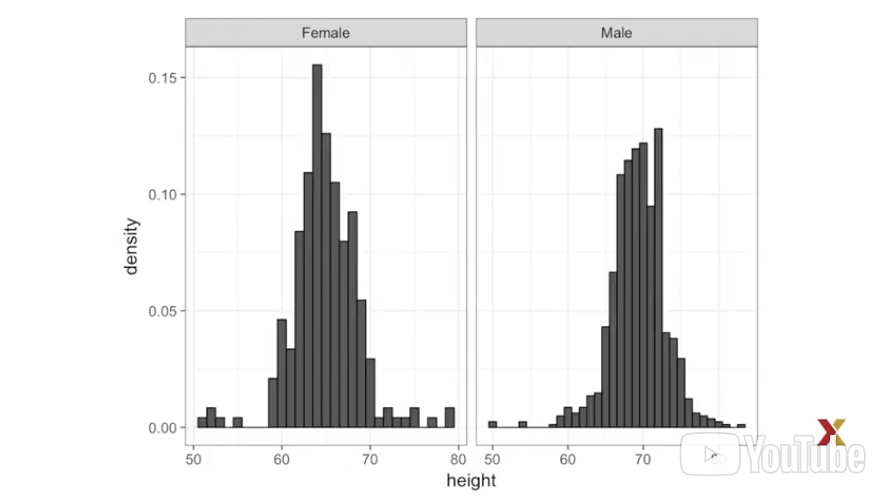
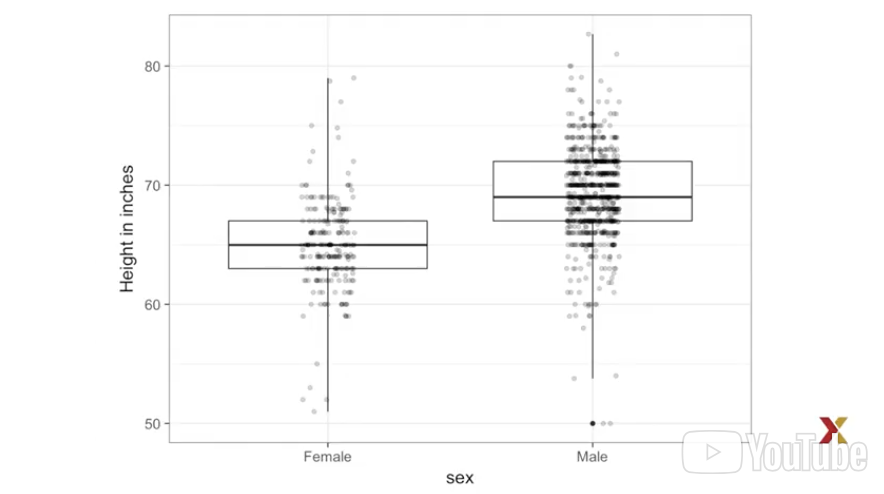
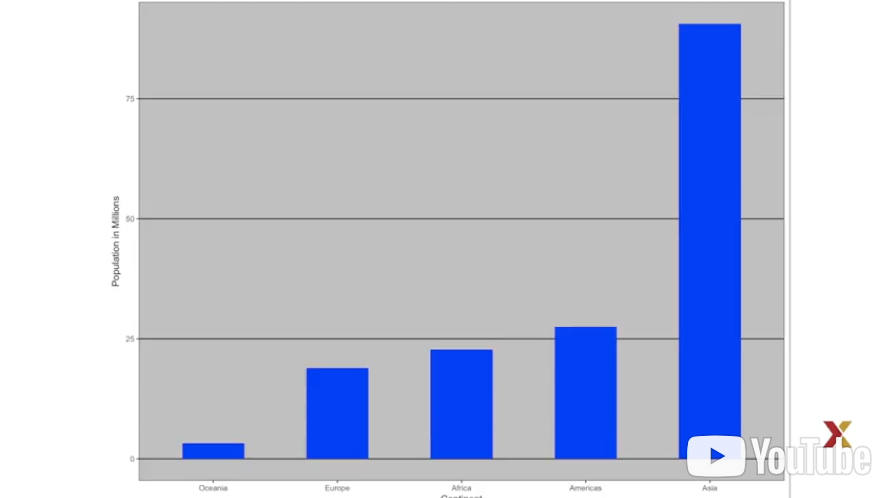
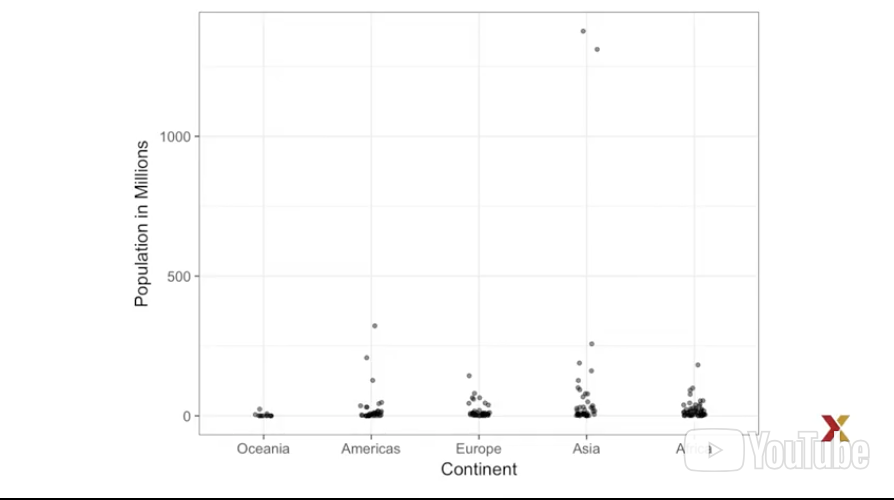
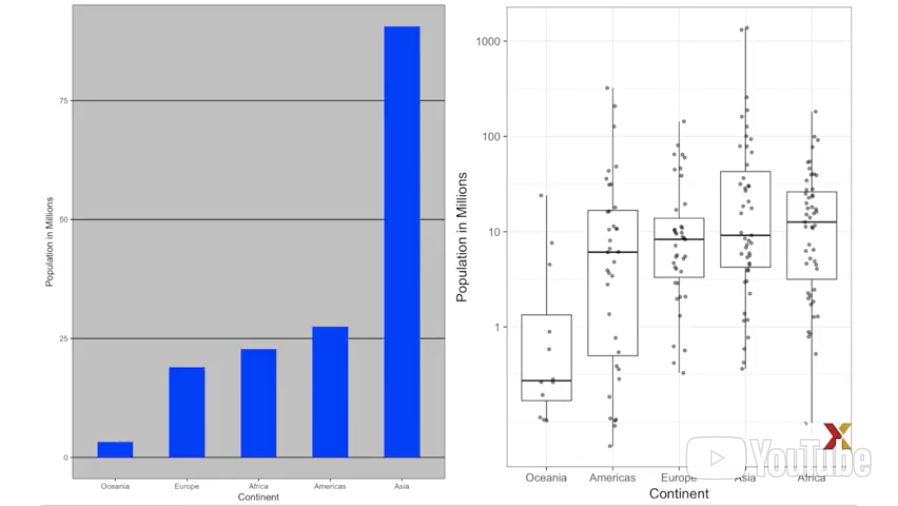
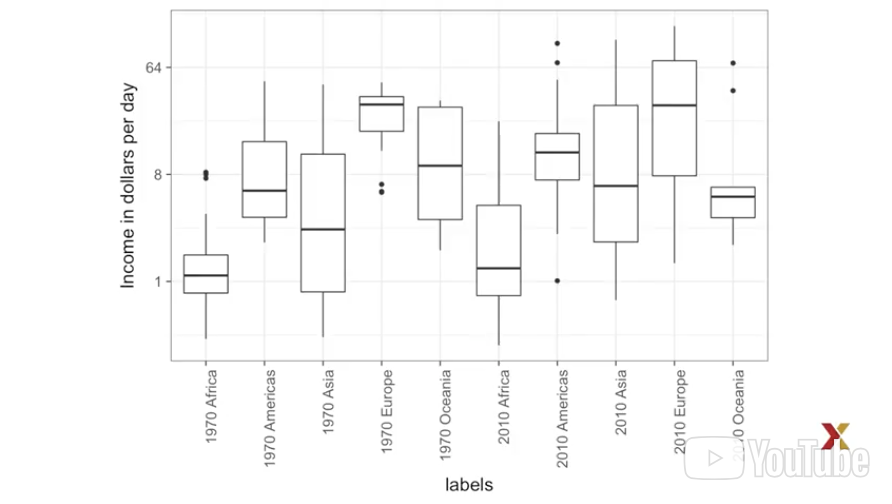
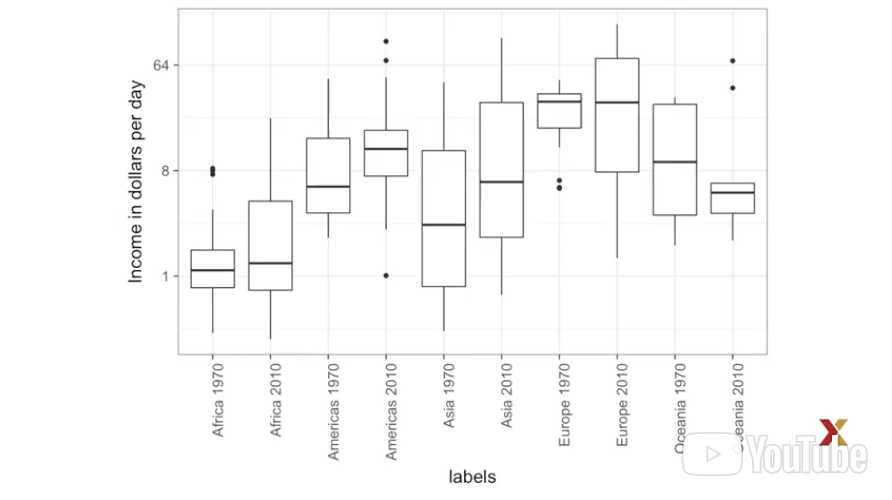
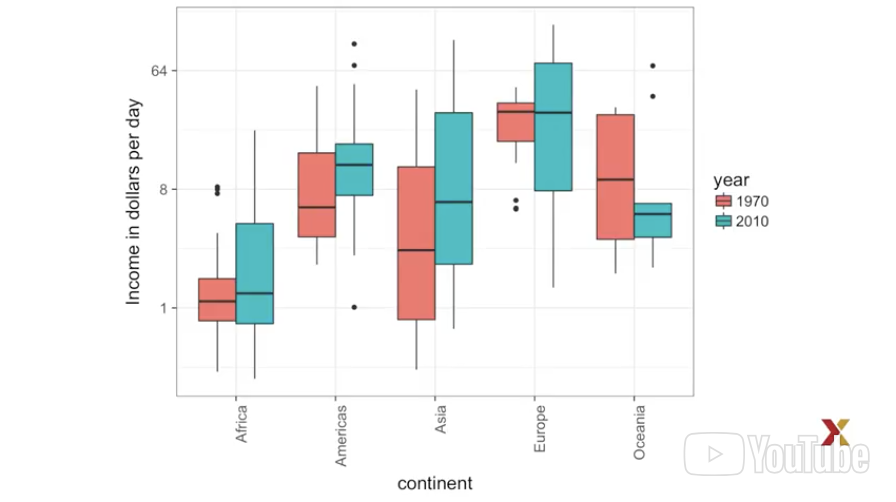
country\_list <- intersect(country\_list\_1,country\_list\_2)

Now, there is 108 countries in this list which accounts for 86% of the total population. So this subset should be representative of the entire world.

Let’s make this plot again, but this time using only the subset of countries for which data is present for 1970 and 2010.

* gapminder
* %>% filter(year==c(1970,2010) & **country %in% country\_list**)
* %>% mutate(group = ifelse(region%in%west,"West","Developing"))
* %>% ggplot(aes(dollars\_per\_day))
* +geom\_histogram(binwidth = 1, color = "black")
* + scale\_x\_continuous(trans = "log2")
* +facet\_grid(year~group)



* We now see that while the rich countries have become a bit richer percentage wise, the poorer countries have improved more.
* The histogram has shifted more to the right than for the rich countries. In particular, we see that the proportion of developing countries earning more than 16$/day increased substantially.
* To see which specific regions improved the most, we can remake the box plots but now adding 2010. We use the same code but adding the layer facet\_grid to divide into 1970 and 2010.
* gapminder %>% filter(year==c(1970,2010) & country%in%country\_list) %>% mutate(region = reorder(region,dollars\_per\_day, FUN=median)) %>%ggplot(aes(region,dollars\_per\_day,fill=continent))
* +geom\_boxplot()
* +theme(axis.text.x = element\_text(angle = 90,hjust =1))
* +xlab("")+scale\_y\_continuous(trans="log2")
* +facet\_grid(year~.)
* 
* Now these box plots are a little bit hard to compare because we are trying to compare box plots that are on top of each other. It’s helpful to see them next to each other.
* So we are going to learn **to ease the comparisons**.
* Let’s pause to introduce another powerful ggplot feature.
* Because we want to compare each region before and after, it would be convenient to have the 1970 box plot next to the 2010 box plot.
* In general, comparisons are easier when data are plotted next to each other.
* So instead of faceting, we keep the data from each year together, but ask ggplot to color the box block depending on the year. Ggplot automatically separates them and puts the two plots next to each other.
* Because year is a number, we turn it into a factor so that each is a category.
* This is because ggplot automatically assigns a color to each level of a factor if we assign that factor to the color argument.
* gapminder %>% filter(year==c(1970,2010) & country%in%country\_list) %>% mutate(region = reorder(region,dollars\_per\_day, FUN=median)) %>%ggplot(aes(region,dollars\_per\_day,fill=factor(year)))
* +geom\_boxplot()
* +theme(axis.text.x = element\_text(angle = 90,hjust = 1))
* +xlab("")
* +scale\_y\_continuous(trans="log2")
* 
* Look at Eastern Asia for example, how it went from way down around $8/day all the way up to almost $64.
* And finally we point out that if what we most interested in is in comparing before and after values, it might make more sense to plot the ratios, or differences in the log scale.
* We are still not ready to learn this code yet.
* DENSITY PLOTS
* We have used data exploration to discover that income gap between rich and poor countries has closed considerably during the last forty years.
* We used a series of histograms and box plots to see this.
* Here, we suggest a succinct way to convey this message with just one plot.
* We will use **smooth density plots**.
* Let’s start by noting that the density plot for income distribution in 1970 and 2010 deliver the message that the gap is closing.
* In the 1970 plot, we see two clear mode, poor and rich.
* In the 2010 plot, it appears that some of the poorer countries have shifted towards the right, closing the gap.
* The next message we need to convey is that the reason for this change in distribution is that poor countries became richer rather than some rich countries becoming poorer.
* To do this, all we need to do is assign a color to the groups we identified during the data exploration.
* However, before we can do this, we need to learn how to make the smooth density plots in a way that preserve the information of how many countries are in each group.
* To understand why we need to do this, note the discrepancy in the size of each group.
* If we divide the world into developing countries and the West, we have 87 developing countries and 21 Western countries.
* gapminder %>% filter(year==1970 & country%in%country\_list)
* %>% mutate(group = ifelse(region %in%west,"West","Developing"))
* %>% group\_by(group)
* %>% summarize(n=n())
* %>% knitr::kable()
* If we overlay the two densities, the default is to have the area represented by each distribution add up to 1 regardless of the size of each group.
* This makes it seem like there is the same number of countries in each group, which is incorrect.
* To change this, we will need to learn to access computed variables with the geom\_density function.
* To have the areas of the densities be proportional to the size of the groups, we can simply multiply the y-axis values by the size of the group.
* From the geom\_density help file, we see that the function computes a variable called count that does exactly this.
* We want this variable to be on the y-axis rather than the density value.
* In ggplot, we can access these variables by surrounding their names with ..
* So we will use the following mapping :
* gapminder %>% filter(year==c(1970,2010) & country%in%country\_list) %>% mutate(group = ifelse(region %in%west,"West","Developing"))
* %>%ggplot(**aes(dollars\_per\_day,y=..count..** ,fill=group))
* +geom\_density(alpha = 0.2)
* +scale\_x\_continuous(trans="log2")
* +facet\_grid(year~.)
* 
* Notice that now we can clearly see that the developing world has more countries.
* If you want the densities to be smoother, because we can see in the Western countries there was a lot of unsmoothness, we can change the bw argument.
* 
* The plot now shows us what is happening very clearly. The developing world is changing. A third mode appears consisting of the countries that most closed the gap.
* We can actually make this figure somewhat more informative.
* From the exploratory data analysis, we noticed that many of the countries that most improved were from Asia.
* We can easily alter the plot to show key regions separately.
* To do this, we introduce a new function called **case\_when**. Very useful for defining groups.
* gapminder %>% mutate(group=case\_when(
* .$region %in% west ~ "West",
* .$region %in% c("Eastern Asia","South-Eastern Asia") ~ "East Asia",
* .$region %in% c("Caribbean","Central America","South America") ~ "Latin America",
* .$continent=="Africa" & .$region!="Northern Africa" ~ "Sub-Saharan Africa",
* TRUE ~ "Others"))
* We’re assigning groups depending on the region.
* Now we turn this group variable into a factor to control the order of the levels.
* We do it like this :
* ECOLOGICAL FALLACY
* ASSESSMENT
* *1. Using ggplot and the points layer, create a scatter plot of life expectancy versus fertility for the African continent in 2012.*
* gapminder %>% filter(year==2012 & continent=="Africa") %>%
* ggplot(aes(fertility ,life\_expectancy )) +
* geom\_point()
* 
* *2. Remake the plot from the previous exercises but this time use color to dinstinguish the different regions of Africa to see if this explains the clusters.*
* gapminder %>% filter(year==2012 & continent=="Africa")
* %>% ggplot(aes(fertility ,life\_expectancy,color=region ))
* + geom\_point()
* 
* *3. Create a table showing the country and region for the African countries (use select) that in 2012 had fertility rates of 3 or less and life expectancies of at least 70.*
* df<-gapminder
* %>%filter(year==2012&continent=="Africa"&life\_expectancy>=70&fertility<=3)
* %>%select(country,region)
* *4. Use filter to create a table with data for the years from 1960 to 2010 in Vietnam and the United States.*
* tab<-gapminder%>%filter(country%in%c("Vietnam","United States") & year>=1960 & year<= 2010)
* *5. Use geom\_line to plot life expectancy vs year for Vietnam and the United States. The data table is stored in tab.*
* tab %>% ggplot(aes(year,life\_expectancy,color=country))+geom\_line()
* 
* *6. Use a single line of code to create a time series plot from 1960 to 2010 of life expectancy vs year for Cambodia.*
* gapminder%>%filter(country=="Cambodia" & year>=1960 & year<= 2010)%>%ggplot(aes(year,life\_expectancy,color=country))+geom\_line()
* 
* *7. Use mutate to create a dollars\_per\_day variable, which is defined as gdp/population/365. Create the dollars\_per\_day variable for African countries for the year 2010.*
* daydollars <-gapminder%>%mutate(dollars\_per\_day=gdp/population/365)%>%filter(continent=="Africa"& year==2010 & !is.na(gdp) )
* *8. The dataset including the dollars\_per\_day variable is preloaded as daydollars. Create a smooth density plot of dollars per day from daydollars. Use a log (base 2) scale for the x axis.*
* daydollars%>%ggplot(aes(dollars\_per\_day))+geom\_density()+scale\_x\_continuous(trans="log2")
* 
* *9. Create the dollars\_per\_day variable as in Exercise 7, but for African countries in the years 1970 and 2010 this time. Make sure you remove any NA values. Create a smooth density plot of dollars per day for 1970 and 2010 using a log (base 2) scale for the x axis. Use facet\_grid to show a different density plot for 1970 and 2010.*
* daydollars <- gapminder
* %>% mutate(dollars\_per\_day=gdp/population/365)
* %>% filter(continent=="Africa"& year%in%c(1970,2010) & !is.na(gdp) )
* daydollars %>% ggplot(aes(dollars\_per\_day))
* + geom\_density()
* + scale\_x\_continuous(trans="log2")
* + facet\_grid(year~.)
* 
* *10. Create the dollars\_per\_day variable as in Exercise 7, but for African countries in the years 1970 and 2010 this time. Make sure you remove any NA values. Create a smooth density plot of dollars per day for 1970 and 2010 using a log (base 2) scale for the x axis. Use facet\_grid to show a different density plot for 1970 and 2010. Make sure the densities are smooth by using bw = 0.5. Use the fill and position arguments where appropriate to create the stacked histograms of each region.*
* daydollars <- gapminder
* %>% mutate(dollars\_per\_day=gdp/population/365)
* %>% filter(continent=="Africa"& year%in%c(1970,2010) & !is.na(gdp) )
* daydollars%>% ggplot(aes(dollars\_per\_day,fill=region))
* + geom\_density(bw=0.5,position="stack")
* + scale\_x\_continuous(trans="log2")
* + facet\_grid(year~.)
* 
* *11. Generate dollars\_per\_day using mutate and filter for the year 2010 for African countries. Remember to remove NA values. Store the mutated dataset in gapminder\_Africa\_2010. Make a scatter plot of infant\_mortaility versus dollars\_per\_day for countries in the African continent. Use color to denote the different regions of Africa.*
* gapminder\_Africa\_2010 <- gapminder
* %>% mutate(dollars\_per\_day=gdp/population/365)
* %>% filter(continent=="Africa"& year==2010 & !is.na(gdp) )
* gapminder\_Africa\_2010 %>% ggplot(aes(dollars\_per\_day,infant\_mortality,color=region))
* +geom\_point()
* 
* *12. Transform the x axis to be in the log (base 2) scale.*
* gapminder\_Africa\_2010 %>% ggplot(aes(dollars\_per\_day,infant\_mortality,color=region))
* +geom\_point()
* +scale\_x\_continuous(trans="log2")
* 
* *13. Add a layer to display country names instead of points.*
* gapminder\_Africa\_2010 %>% ggplot(aes(dollars\_per\_day,infant\_mortality,label=country,color=region))
* + geom\_point()
* + scale\_x\_continuous(trans="log2")
* +geom\_text()
* 
* *14. Use facet\_grid to show different plots for 1970 and 2010.*
* gapminder%>% mutate(dollars\_per\_day=gdp/population/365)
* %>% filter(continent=="Africa"& year%in%c(1970,2010)& !is.na(dollars\_per\_day) & !is.na(infant\_mortality))
* %>% ggplot(aes(dollars\_per\_day,infant\_mortality,
* label=country,color=region))
* + geom\_point()
* + geom\_text()
* + geom\_label()
* + facet\_grid(year~.)
* + scale\_x\_continuous(trans="log2")
* INTRODUCTION TO DATA VISUALIZATION PRINCIPLES
* Here we aim to provide some general principles we can use as guidelines for effective data visualization.
* Much of this part of the course is based on a talk by Karl Broman entitled “ Creating Effective Figures and Tables ” and from class notes from Peter Aldhous titled “ Introduction to Data Visualization ” .
* In many of our examples we follow Karl’s approach.
* We show some examples of plot styles we should avoid, explain how to improve them, and then use these as motivation for a list of principles.
* We compare and contrast plots that follow these principles to those that don’t.
* The principles are mostly based on research related to how humans detect patterns and make visual comparisons.
* The preferred approaches are those that best fit the way our brain processes visual information.
* It is also important to keep our goal in mind. We may be comparing a viewable number of quantities, describing distributions for categories or numeric values, comparing the data from two groups, or describing the relationship between two variables.
* As a final note, we also know that for a data scientist it is important to adapt and optimize graphs to the audience.
* For example, an exploratory plot made for ourselves will be different than a chat intended to communicate a finding to a general audience.
* ENCODING DATA USING VISUAL CUES
* We start by describing some principles for encoding data.
* There are several approaches to our disposal, including position, aligned lengths, angles, area, brightness and color hue.
* In our first example, to illustrate how some of these strategies compare, let’s suppose we want to report results from two hypothetical polls, asking what is your browser preference and the polls were taken in 2000 and 2015.
* Here, for each year, we are simply comparing four quantities, four percentages.
* A widely used graphical representation of percentages, popularized by Microsoft Excel, is the **pie chart**. There are two pie charts, one for 2000, one for 2015.
* 
* Here, we are representing quantities with both areas and angles, since both the angle and area of each pie slice is proportional to the quantity it represents.
* This turns out to be a suboptimal choice, as demonstrated by perception studies, humans are not good at precisely quantifying angles, and are even worse when only area is available.
* This makes the donut chart, which only uses area, even worse than the pie chart.
* 
* To see how hard it is to quantify angles and area, note that the rankings in the plots we just saw changed from 2000 to 2015.
* Can you determined the actual percentages and rank the browser’s popularity ? Can you see how percentages changed from 2000 to 2015 ?
* it is not easy to tell from the plot. In this case, simply showing the numbers is not only clear, but it would save us our print costs, if making a paper version of our results.
* If we write out the percentages, we quickly see which browser is more popular, and how they changed from 2000 to 2015. If we insist on a plot, the preferred way to plot these quantities is to use length and positions, since humans are much better at judging linear measures.
* The bar plot uses this approach by using bars of length proportional to the quantity of interest. By adding a horizontal line at strategically chosen values, in this case every multiple of 10, we ease the quantifying through the position of the top of the bars.
* 
* Compare these two plots. Notice how easier it is to see the differences in the bar plot. In fact, we can now determine the actual percentages by following a horizontal line to the y-axis.
* If for some reason, you need to make a pie chart, do include percentages as numbers to avoid having to infer them from the angles or area.
* **In summary, position and length are the preferred way to display quantities over angles, which are preferred over area**.
* Brightness and color are even harder to quantify than angles and area. But as we will see later, there are sometimes useful when more than two dimensions are being displayed.
* KNOW WHEN TO INCLUDE 0
* **When using bar plots, it is dishonest not to start the bars at 0**. This is because by using a bar plot, we are implying the length is proportional to the quantities being displayed. By avoiding 0, relatively small differences can be made to look much bigger than they actually are.
* This approach is often used by politicians or media organizations trying to exaggerate the difference.
* Here is an illustrative example :
* 
* This is a bar plot made by Fox News showing southwest border apprehensions in 2011, 2012, and 2013. Look how much bigger the 2013 bar looks compared to the 2011. From this plot, it appears that apprehensions have almost triplet, when iin fact, if you look at the numbers, they have only increased by 16%.
* Starting the graph at 0 illustrates this clearly. This is what it looks like if the plot includes 0.
* 
* Here is another example, again from Fox News, that is showing us what would happen if Bush tax cuts expires for the top tax rate. It’s comparing January 1,2013 to the time in which this broadcast. When we look at the bar plots, it looks like January 1,2013 is about 5 times bigger than the now bar plot.
* 
* Here is what it looks like with the appropriate plot, a much different story.
* When using position rather than length, then it’s not necessary to include 0.
* This is particularly the case when we want to compare differences between groups relative to the variability seen within the groups.
* 
* Here is an illustrative example showing country average life expectancies, stratified into continents, in 2012.
* In the plot on the left, which includes 0, the space between 0 and 43 adds no information, and makes it harder to appreciate the between and within variability. For this reason, on the plot on the right, we restrict the range to include the points.
* DO NOT DISTORT QUANTITIES
* Our next principle is **do not distort quantities**.
* 
* Here’s an example : during President Barack Obama’s 2011 State of the Union address, the following chart was used to compare the US GDP to the GDP of four competing nations. Note that judging by the area of the circles, the US appears to have an economy over 5 times larger than China, and over 30 times larger than France. However, when looking at the actual numbers, one sees that this is not the case.
* The actual ratios are 2.6, and 5.8 times bigger than China and France respectively.
* The reason for this distortion is that the radius, rather than the area, was made to be proportional to the quantity, which implies that the proportions between the areas is squared.
* So 2.6 turns into 6.5 and 5.8 turns into 34.1.
* 
* Here is a comparison of the circles we get if we make the values proportional to the radius, that’s on the left, and so the area, that’s on the right.
* Not surprisingly, ggplot defaults to using area rather than the radius.
* Of course, in this case, we really should not be using area at all, since we can use position and length. Here’s the bar plot comparing the GDPs.
* 
* ORDER BY A MEANINGFUL VALUE
* Our next principle is **order by a meaningful value.**
* When one of the axes is used to show categories, as done bar plots, the default ggplot behavior is to order the categories alphabetically when they are defined by character strings. If they are defined by factors, they are ordered by the factor levels.
* But remember, factor levels default to ordering by alphabetical order. We rarely want to use alphabetical order since it’s arbitrary.
* Instead, we should order by a meaningful quantity.
* In all the cases discussed, the bar plots were ordered by the values being displayed.
* The exception was the graph showing bar plots comparing browsers. In this case, we kept the order the same across the bar plots to ease the comparison.
* Instead, we ordered by the average value of 2000 and 2015.
* We previously learned how to use the reorder function, which helps achieve this goal.
* To appreciate how the right order helps convey a message, suppose we want to create a plot to compare the murder rates across states. We are particularly interested in the most dangerous and the safest states.
* 
* Note the difference when we order alphabetically, the default behavior, versus when we order by the actual rate.
* The information we want is much easier to extract from the plot on the right.
* Note that the reorder function lets us reorder groups as well.
* In an earlier class, we saw an example related to income distribution across regions. Here are these two plots again :
* In the first one, we simply order alphabetically. In the second one, we order by the median value of each group.
* So we have seen a few examples of how ordering by meaningful values make much better graphs.
* SHOW THE DATA
* In this class, we describe yet another principle, **show the data**.
* We have focused on displaying single quantities across categories. We now shift our attention to this plane data **with a focus on comparing groups**.
* To motivate this principle, we go back to our official example describing heights to an extraterrestrial, ET.
* This time, let’s assume ET is interested in the difference in heights between male and female. A commonly seen plot used for comparison between groups, popularized by software such as Microsoft Excel, shows the average and the standard error. Now keep in mind, standard errors, which we define later, are not the same as standard deviation.
* Here is what the plot looks like :
* 
* The average of each group is represented by the top of each bar and the antenna that we see that expands out is the average plus two standard errors.
* If all ET receives is this plot, he will have little information on what to expect if he meets a group of humans, males and females.
* Note that the bars go to 0.
* Does this mean there are tiny humans measuring less than one foot?
* Are all males taller than the tallest female?
* Is there a range of heights?
* ET can’t answer these questions since we have provided almost no information on the height distribution.
* This brings us back to our principle, show the data.
* The following simple ggplot code already generate a more informative plot than the bar plot by simply showing all the points.
* heights %>% ggplot(aes(sex,height)) + geom\_point()
* 
* Just this little line of code shows you the points, the heights for females and the heights for males.
* However, this plot has limitations as well since we can’t really see all the 216 and 708 points plotted for females and males, respectively. And many points are plotted above each other so we don’t know how many there are.
* As we have described, visualizing the distribution is much more informative. But before doing this, we point out two ways we can improve a plot showing all the points.
* The first is to add **jitter**. Jitter is adding a small random shift to each point. in this case, **adding horizontal jitter does not alter the interpretation** since the height of the points doesn’t change. But we minimize the number of points that fall on top of each other and, therefore, get a better sense of how many points there are and how the data is distributed.
* A second improvement comes from **alpha blending**, making the point somewhat transparent. Without alpha blending, the more points fall on top of each other, the darker the plot gets in that region, which also helps us get a sense of how the points are distributed.
* Here is the same plot with jitter and alpha blending :
* 
* It just requires us to change a couple of arguments, and immediately we get a much better sense of what the distribution of the data is.
* Now since there are so many points, it is more effective to show distribution rather than show the individual points.
* in the next class, we’ll show distributions, and we’ll learn some principles of how to best compare these distributions.
* EASE COMPARISONS : USE COMMON AXES
* Earlier, we saw this plot used to compare male and females heights.
* 
* We criticized it for not showing all the data.
* Now in this case, showing all the data is not as effective as showing distributions. So let’s start by creating histograms for each group. They look like this:
* 
* However, from this plot it’s not immediately obvious that males are, on average, taller than females. We have to look carefully to notice that the x-axis has a higher range of values in the male histogram.
* This brings us to another important principle, and it’s to **keep the axes the same when comparing data across plots**.
* Know how the comparison becomes easier when we keep the axes the same.
* Now we do see that the male histogram is shifted to the right, compared to the female histogram.
* Now, there’s another principle that we need to follow here to make it even easier to compare, and it’s **to align plots vertical to see horizontal changes, and horizontally to see vertical changes**.
* In these histograms, the visual cue related to decrease or increase height, are shifts to the left, or right, respectively.
* Horizontal changes, aligning in the plots vertically, helps us see the change when the axes are fixed. Look at how much easier it is to see it now :
* 
* This one clearly shows that the male heights are shifted to the right.
* If instead of histograms, we want the more compact summary provided by box plots, then we align horizontally, since by default box plots move up and down with changes in height, vertically.
* Here are the two box blocks plotted next to each other, horizontally next to each other. We can appreciate the vertical changes.
* 
* In this case, we also add all the data, all the points, using ***jitter and alpha blending***.
* Now contrast and compare these three plots, based on exactly the same data.
* Note how much more we learn from the two plots on the right. Bar plots are useful for showing one number, but not very useful when wanting to describe distributions.
* CONSIDER TRANSFORMATIONS
* Another important principle is **to consider transformations**.
* We have motivated the use of **the log transformation in cases where the changes are multiplicative**. Population size was an example in which we found a log transformation to yield a more informative plot. The combination of incorrectly using bar plots, when a log transformation is merited, can be particularly distorting.
* As an example, consider this bar plot showing the average population sizes for each continent in 2015.
* 
* From this plot, one would conclude that countries in Asia are much more populous than other continents. Following the show-the-data principle, we quickly notice that this is due to two very large countries, which we assume are India and China.
* 
* You can see those two points way up there.
* Here, using a log transformation provides a much more informative plot.
* We compare the original bar plot to a box plot using the log-scale transformation for the y-axis.
* 
* Note how much more informative that box plot is over the bar plot. In fact, we see that Africa has a higher median population size than Asia.
* Other transformation you should consider are :
* - the logistic transformation – useful to better see full changes in odds –
* - the square root transformation, useful for count data.
* EASE COMPARISONS: COMPARED VISUAL CUES SHOULD BE ADJACENT
* In this class, we introduce yet another principle. It helps us ease comparison. And the principle is **that visual cues to be compared should be adjacent.**
* When comparing income data between 1970 and 2010, across regions, we made a figure similar to this one :
* 
* A difference is that here, we look at continents instead of regions. But this is not relevant to the point we are trying to make.
* Note that for each continent, we want to compare the distribution from 1970 to 2010. The default in ggplot is to order alphabetically. So the labels with 1970 come before the labels with 2010, making that comparison challenging.
* Know how much easier it is to make the comparison when the box blocks that we want to compare are next to each other.
* 
* Here, we have reordered them to do just that. The comparison becomes even easier if we use color to distinguish 1970 to 2010.
* 
* **Using colors is another way to ease comparisons**. When picking colors keep in mind that about 10% of the population is color blind.
* Unfortunately, the default colors used in ggplot are not optimal for this group.
* However, ggplot does it make it easy to change the color palette used in the plots. Here’s an example of how we can use a color blind friendly palette using ggplot.
* You can see that by adding some layers, and picking the right colors, we can in fact, make the use of color blind friendly colors.