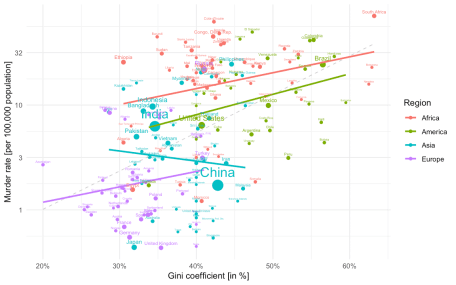
**Abstract:**

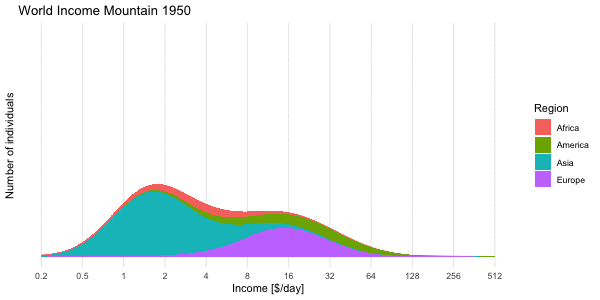
We follow up on last weeks post on using Gapminder data to study the world’s income distribution. In order to assess the inequality of the distribution we compute the Gini coefficient for the world’s income distribution by Monte Carlo approximation and visualize the result as a time series. Furthermore, we animate the association between Gini coefficient and homicide rate per country using the new version of gganimate.



**Introduction**

## **Abstract:**

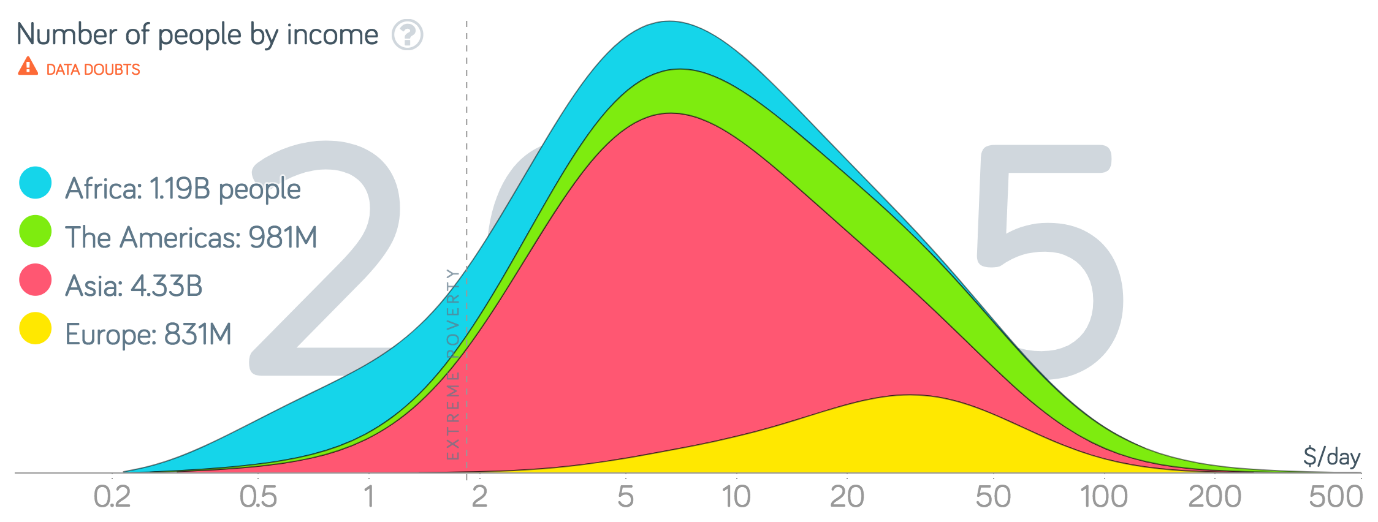
We work out the math behind the so called income mountain plots used in the book “Factfulness” by Hans Rosling and use these insight to generate such plots using tidyverse code. The trip includes a mixture of log-normals, the density transformation theorem, histogram vs. density and then skipping all those details again to make nice moving mountain plots.



**Introduction**

Reading the book Factfulness by Hans Rosling seemed like a good thing to do during the summer months. The ‘possibilistic’ writing style is contagious and his TedEx presentations and media interviews are legendary teaching material on how to support your arguments with data. What a shame he passed away in 2017.

What is really enjoyable about the book is that the Gapminder web page allows you to study many of the graphs from the book interactively and contains the data for download. Being a fan of **transparency** and **reproducibility**, I got interested in the so called **income mountain plots**, which show how incomes are distributed within individuals of a population:



Screenshot of the 2010 income mountain plot. Free material from www.gapminder.org.

One notices that the “mountains” are plotted on a log-base-2 x-axis and without a y-axis annotation. Why? Furthermore, world income data usually involve mean income per country, so I got curious how/if these plots were made without access to finer granularity level data? Aim of this blog post is to answer these questions by using Gapminder data freely available from their webpage. The answer ended up as a nice tidyverse exercise and could serve as motivating application for basic probability course content.

## **Data Munging Gapminder**

Data on income, population and Gini coefficient were needed to analyse the above formulated questions. I have done this previously in order to visualize the Olympic Medal Table Gapminder Style. We start by downloading the GDP data, which is the annual gross domestic product per capita by Purchasing Power Parities (PPP) measured in international dollars, fixed 2011 prices. Hence, the inflation over the years and differences in the cost of living between countries is accounted for and can thus be compared - see the Gapminder documentation for further details. We download the data from Gapminder where they are available in wide format as Excel-file. For tidyverse handling we reshape them into long format.

##Download gdp data from gapminder - available under a CC BY-4 license.

if (!file.exists(file.path(fullFigPath, "gapminder-gdp.xlsx"))) {

download.file("https://github.com/Gapminder-Indicators/gdppc\_cppp/raw/master/gdppc\_cppp-by-gapminder.xlsx", destfile=file.path(fullFigPath,"gapminder-gdp.xlsx"))

}

gdp\_long <- readxl::read\_xlsx(file.path(fullFigPath, "gapminder-gdp.xlsx"), sheet=2) %>%

rename(country=`geo.name`) %>%

select(-geo,-indicator,-indicator.name) %>%

gather(key="year", value="gdp", -country,) %>%

filter(!is.na(gdp))

Furthermore, we rescale GDP per year to daily income, because this is the unit used in the book.

gdp\_long %<>% mutate(gdp = gdp / 365.25)

Similar code segments are written for (see the code on github for details)

* the gini (gini\_long) and population (pop\_long) data
* the regional group (=continent) each country belongs two (group)

The four data sources are then joined into one long tibble gm. For each year we also compute the fraction a country’s population makes up of the world population that year (column w) as well as the fraction within the year and region the population makes up (column w\_region) :

## # A tibble: 15,552 x 9

## country region code year gini gdp population w w\_region

## <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>

## 1 Afghanistan Asia AFG 1800 0.305 1.65 3280000 0.00347 0.00518

## 2 Albania Europe ALB 1800 0.389 1.83 410445 0.000434 0.00192

## 3 Algeria Africa DZA 1800 0.562 1.96 2503218 0.00264 0.0342

## 4 Andorra Europe AND 1800 0.4 3.28 2654 0.00000280 0.0000124

## 5 Angola Africa AGO 1800 0.477 1.69 1567028 0.00166 0.0214

## # ... with 1.555e+04 more rows

## **Income Mountain Plots**

The construction of the income mountain plots is thoroughly described on the Gapminder webpage, but without mathematical detail. With respect to the math it says: “Bas van Leeuwen shared his formulas with us and explained how to the math from ginis and mean income, to accumulated distribution shapes on a logarithmic scale.” Unfortunately, the formulas are not shared with the reader. It’s not black magic though: The income distribution of a country is assumed to be log-normal with a given mean μμ and standard deviation σσ on the log-scale, i.e. X∼LogN(μ,σ2)X∼LogN⁡(μ,σ2). From knowing the mean income ¯¯¯xx¯ of the distribution as well as the Gini index GG of the distribution, one can show that it’s possible to directly infer (μ,σ)(μ,σ) of the log-normal distribution.

Because the Gini index of the log-normal distribution is given byG=2Φ(σ√2)−1,G=2Φ(σ2)−1,where ΦΦ denotes the CDF of the standard normal distribution, and by knowing that the expectation of the log-normal is E(X)=exp(μ+12σ2)E(X)=exp⁡(μ+12σ2), it is possible to determine (μ,σ)(μ,σ) as:

σ=√2Φ−1(G+12)andμ=log(¯¯¯x)−12σ2.σ=2Φ−1(G+12)andμ=log⁡(x¯)−12σ2.

We can use this to determine the parameters of the log-normal for every country in each year.

### **Mixture distribution**

The income distribution of a **set of countries** is now given as a Mixture distribution of log-normals, i.e. one component for each of the countries in the set with a weight proportional to the population of the country. As an example, the world income distribution would be a mixture of the 192 countries in the Gapminder dataset, i.e.

fmix(x)=192∑i=1wi⋅fLogN(x;μi,σ2i),wherewi=populationi∑192j=1populationj,fmix(x)=∑i=1192wi⋅fLogN(x;μi,σi2),wherewi=populationi∑j=1192populationj,and fLogN(x;μi,σ2i)fLogN(x;μi,σi2) is the density of the log-normal distribution with country specific parameters. Note that we could have equally used the mixture approach to define the income of, e.g., a continent region. With the above definition we define standard R-functions for computing the PDF (dmix), CDF (pmix), quantile function (qmix) and a function for sampling from the distribution (rmix) - see the github code for details.

We use the mixture approach to compute the density of the world income distribution obtained by “mixing” all 192 log-normal distributions. This is shown below for the World income distribution of the year 2015. Note the log2log2 x-axis. This presentation is Factfulness’ preferred way of illustrating the skew income distribution.

##Restrict to year 2015

gm\_recent <- gm %>% filter(year == 2015) %>% ungroup

##Make a data frame containing the densities of each region for

##the gm\_recent dataset

df\_pdf <- data.frame(log2x=seq(-2,9,by=0.05)) %>%

mutate(x=2^log2x)

pdf\_region <- gm\_recent %>% group\_by(region) %>% do({

pdf <- dmix(df\_pdf$x, meanlog=.$meanlog, sdlog=.$sdlog, w=.$w\_region)

data.frame(x=df\_pdf$x, pdf=pdf, w=sum(.$w), population=sum(.$population), w\_pdf = pdf\*sum(.$w))

})

## Total is the sum over all regions - note the summation is done on

## the original income scale and NOT the log\_2 scale. However, one can show that in the special case the result on the log-base-2-scale is the same as summing the individual log-base-2 transformed densities (see hidden CHECKMIXTUREPROPERTIES chunk).

pdf\_total <- pdf\_region %>% group\_by(x) %>%

summarise(region="Total",w=sum(w), pdf = sum(w\_pdf))

## Expectation of the distribution

mean\_mix <- gm\_recent %>%

summarise(mean=sum(w \* exp(meanlog + 1/2\*sdlog^2))) %$% mean

## Median of the distribution

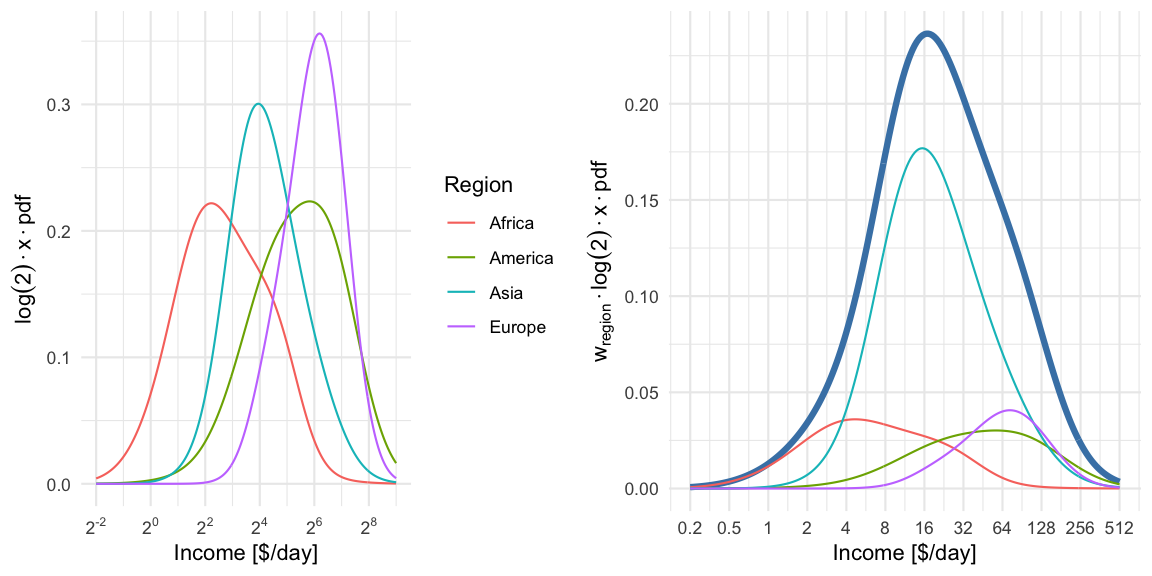
median\_mix <- qmix(0.5, gm\_recent$meanlog, gm\_recent$sdlog, gm\_recent$w)

## Mode of the distribution on the log2-scale (not transformation invariant!)

mode\_mix <- pdf\_total %>%

mutate(pdf\_log2x = log(2) \* x \* pdf) %>%

filter(pdf\_log2x == max(pdf\_log2x)) %$% x



For illustration we compute a mixture distribution for each region using all countries within region. This is shown in the left pane. Note: because a log-base-2-transformation is used for the x-axis, we need to perform a change of variables, i.e. we compute the density for Y=log2(X)=g(X)Y=log2⁡(X)=g(X) where X∼fmixX∼fmix, i.e.fY(y)=∣∣∣ddy(g−1(y))∣∣∣fX(g−1(y))=log(2)⋅2y⋅fmix(2y)=log(2)⋅x⋅fmix(x), where x=2y.fY(y)=|ddy(g−1(y))|fX(g−1(y))=log⁡(2)⋅2y⋅fmix(2y)=log⁡(2)⋅x⋅fmix(x), where x=2y.

In the right pane we then show the region specific densities each weighted by their population fraction. These are then summed up to yield the world income shown as a thick blue line. The median of the resulting world income distribution is at 20.0 $/day, whereas the mean of the mixture is at an income of 39.9$/day and the mode (on the log-base-2 scale) is 17.1$/day. Note that the later is not transformation invariant, i.e. the value is not the mode of the income distribution, but of log2(X)log2⁡(X).

To get the income mountain plots as shown in Factfulness, we additionally need to obtain number of people on the yy-axis and not density. We do this by partitioning the x-axis into non-overlapping intervals and then compute the number of individuals expected to fall into a given interval with limits [l,u][l,u]. Under our model this expectation is

n⋅(Fmix(u)−Fmix(l)),n⋅(Fmix(u)−Fmix(l)),

where FmixFmix is the CDF of the mixture distribution and nn is the total world population. The mountain plot below shows this for a given partition with n=7,305,116,647n=7,305,116,647. Note that 2.5⋅1082.5⋅108 corresponds to 250 mio people. Also note the log2log2 x-axis, and hence (on the linear scale) unequally wide intervals of the partitioning. Contrary to Factfulness’, I prefer to make this more explicit by indicating the intervals explicitly on the x-axis of the mountain plot, because it is about number of people in certain **income brackets**.

##Function to prepare the data.frame to be used in a mountain plot

make\_mountain\_df <- function(gm\_df, log2x=seq(-2,9,by=0.25)) {

##Make a data.frame containing the intervals with appropriate annotation

df <- data.frame(log2x=log2x) %>%

mutate(x=2^log2x) %>%

mutate(xm1 = lag(x), log2xm1=lag(log2x)) %>%

mutate(xm1=if\_else(is.na(xm1),0,xm1),

log2xm1=if\_else(is.na(log2xm1),0,log2xm1),

mid\_log2 = (log2x+log2xm1)/2,

width = (x-xm1),

width\_log2 = (log2x-log2xm1)) %>%

##Format the interval character representation

mutate(interval=if\_else(xm1<2, sprintf("[%6.1f-%6.1f]",xm1,x), sprintf("[%4.0f-%4.0f]",xm1,x)),

interval\_log2x=sprintf("[2^(%4.1f)-2^(%4.1f)]",log2xm1,log2x))

##Compute expected number of individuals in each bin.

people <- gm\_df %>% group\_by(region) %>% do({

countries <- .

temp <- df %>% slice(-1) %>% rowwise %>%

mutate(

prob\_mass = diff(pmix(c(xm1,x), meanlog=countries$meanlog, sdlog=countries$sdlog, w=countries$w\_region)),

people = prob\_mass \* sum(countries$population)

)

temp %>% mutate(year = max(gm\_df$year))

})

##Done

return(people)

}

##Create mountain plot data set for gm\_recent with default spacing.

(people <- make\_mountain\_df(gm\_recent))

## # A tibble: 176 x 13

## # Groups: region [4]

## region log2x x xm1 log2xm1 mid\_log2 width width\_log2 interval interval\_log2x prob\_mass people year

## <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <chr> <dbl> <dbl> <chr>

## 1 Africa -1.75 0.297 0.25 -2 -1.88 0.0473 0.25 [ 0.2- 0.3] [2^(-2.0)-2^(-1.8)] 0.00134 1586808. 2015

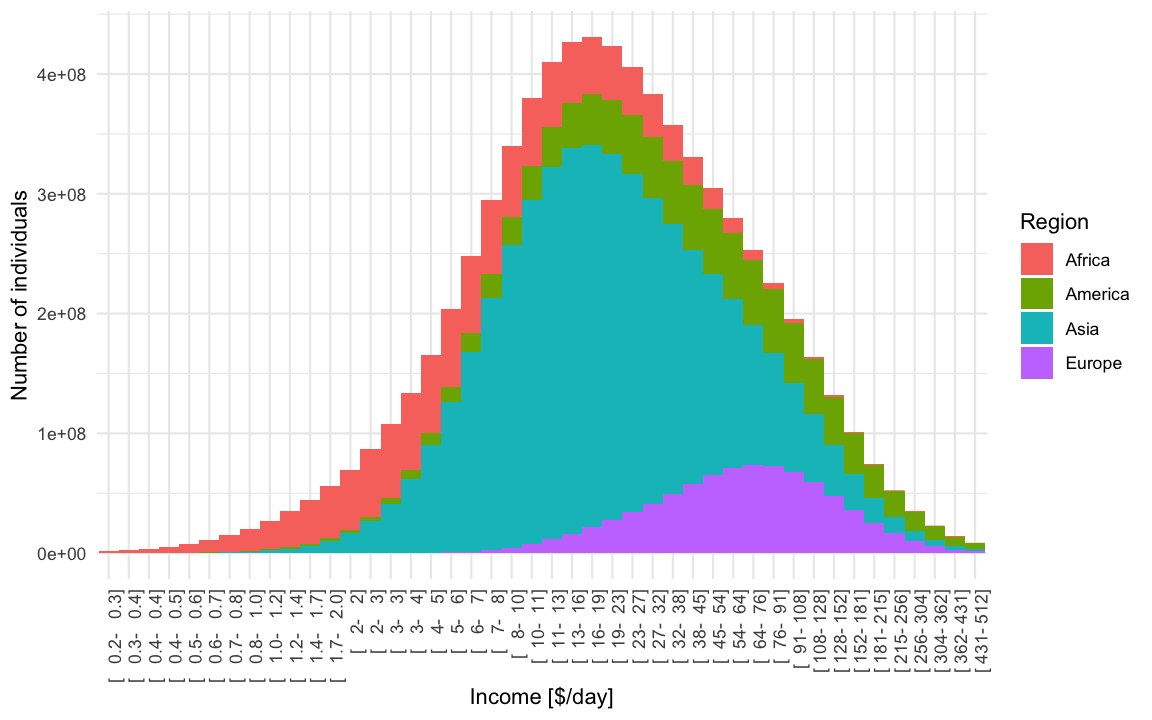
## 2 Africa -1.5 0.354 0.297 -1.75 -1.62 0.0563 0.25 [ 0.3- 0.4] [2^(-1.8)-2^(-1.5)] 0.00205 2432998. 2015

## 3 Africa -1.25 0.420 0.354 -1.5 -1.38 0.0669 0.25 [ 0.4- 0.4] [2^(-1.5)-2^(-1.2)] 0.00307 3639365. 2015

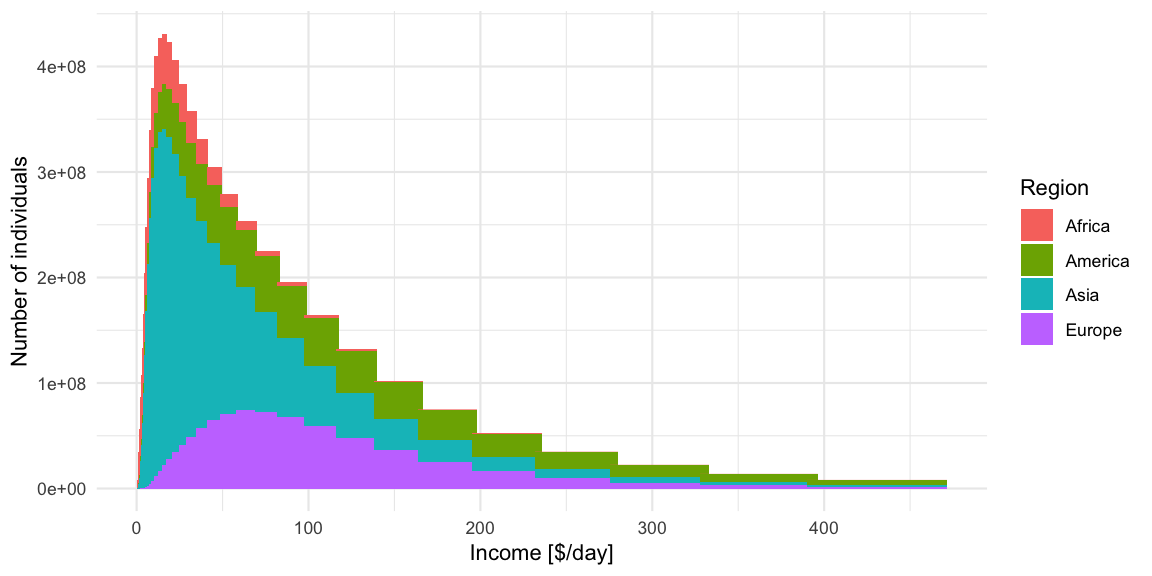
## 4 Africa -1 0.5 0.420 -1.25 -1.12 0.0796 0.25 [ 0.4- 0.5] [2^(-1.2)-2^(-1.0)] 0.00448 5305674. 2015

## 5 Africa -0.75 0.595 0.5 -1 -0.875 0.0946 0.25 [ 0.5- 0.6] [2^(-1.0)-2^(-0.8)] 0.00636 7537067. 2015

## # ... with 171 more rows

This can then be plotted with ggplot2:

In light of all the talk about gaps, it can also be healthy to plot the income distribution on the linear scale. From this it becomes obvious that linearly there indeed are larger absolute differences in income, but -as argued in the book- the exp-scale (base 2) incorporates peoples perception about the worth of additional income.

Because the intervals are not equally wide, only the height of the bars should be interpreted in this plot. However, the eye perceives area, which in this case is misguiding. Showing histograms with unequal bin widths is a constant dilemma between area, height, density and perception. The recommendation would be that if one wants to use the linear-scale, then one should use equal width linear intervals or directly plot the density. As a consequence, plots like the above are not recommended, but they make obvious the tail behaviour of the income distribution - a feature which is somewhat hidden by the log-base-2-scale plots.

Of course none of the above plots looks as nice as the Gapminder plots, but they have proper x and y-axes annotation and, IMHO, are clearer to interpret, because they do not mix the concept of density with the concept of individuals falling into income bins. As the bin-width converges to zero, one gets the density multiplied by nn, but this complication of infinitesimal width bins is impossible to communicate. In the end this was the talent of Hans Rosling and Gapminder - to make the complicated easy and intuitive! We honor this by skipping the math1 and celebrate the result as the **art** it is!

##Make mountain plot with smaller intervals than in previous plot.

ggplot\_oneyear\_mountain <- function(people, ymax=NA) {

##Make the ggplot

p <- ggplot(people %>% rename(Region=region), aes(x=mid\_log2,y=people, fill=Region)) +

geom\_col(width=min(people$width\_log2)) +

ylab("Number of individuals") +

xlab("Income [$/day]") +

scale\_x\_continuous(minor\_breaks = NULL, trans="identity",

breaks = trans\_breaks("identity", function(x) x,n=11),

labels = trans\_format(trans="identity", format=function(x) ifelse(x<0, sprintf("%.1f",2^x), sprintf("%.0f",2^x)))) +

theme(axis.text.y=element\_blank(), axis.ticks.y=element\_blank()) +

scale\_y\_continuous(minor\_breaks = NULL, breaks = NULL, limits=c(0,ymax)) +

ggtitle(paste0("World Income Mountain ",max(people$year))) +

NULL

#Show it and return it.

print(p)

invisible(p)

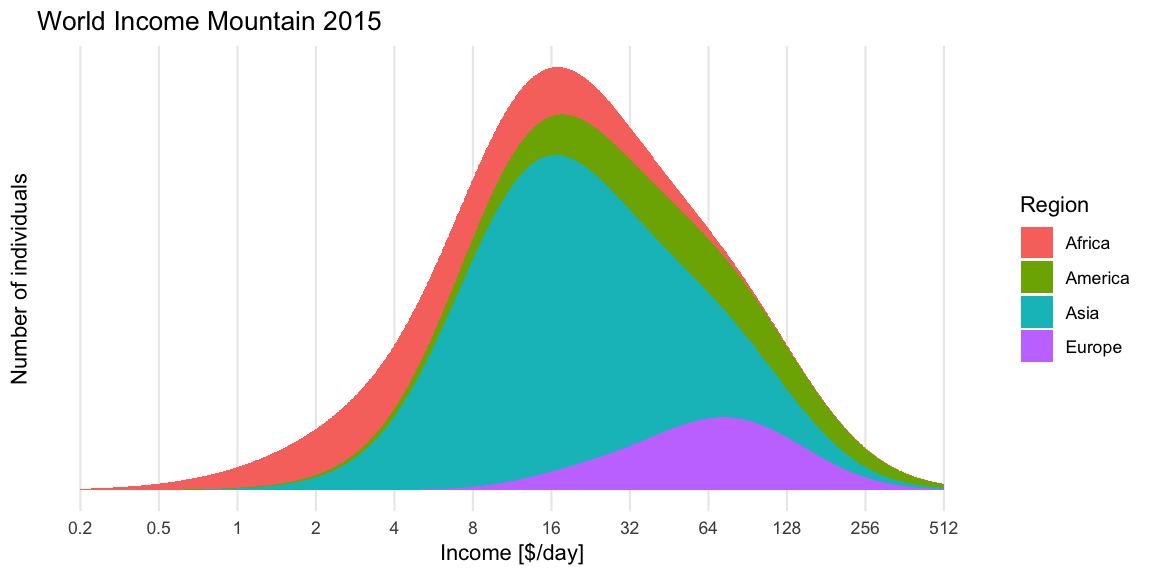
}

##Create the mountain plot for 2015

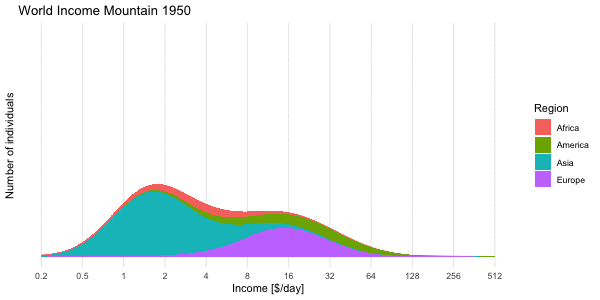
gm\_recent %>%

make\_mountain\_df(log2x=seq(-2,9,by=0.01)) %>%

ggplot\_oneyear\_mountain()



We end the post by animating the dynamics of the income mountains since 1950 using gganimate. To put it in possibilistic terms: Let the world move forward! It is not as bad as it seems. Facts matter.



Gini coefficients to investigate the association with the number of homicides in the country.

**Gini coefficient**

There are different ways to measure income inequality, both in terms of which response you consider and which statistical summary you compute for it. Not going into the details of these discussion we use the GDP/capita in [Purchasing Power Parities (PPP)](https://en.wikipedia.org/wiki/Purchasing_power_parity) measured in so called international dollars (fixed prices 2011). In other words, comparison between years and countries are possible, because the response is adjusted for inflation and differences in price of living.

The [**Gini coefficient**](https://en.wikipedia.org/wiki/Gini_coefficient) is a statistical measure to quantify inequality. In what follows we shall focus on computing the Gini coefficient for a continuous probability distribution with a known probability density function. Let the probability density function of the non-negative continuous income distribution be defined by \(f\), then the Gini coefficient is given as **half the relative mean difference**:

\[  
G  
= \frac{1}{2\mu}\int\_0^\infty \int\_0^\infty |x-y| \> f(x) \> f(y) \>  
dx\> dy, \quad\text{where}\quad \mu = \int\_{0}^\infty x\cdot f(x) dx.  
\]

Depending on \(f\) it might be possible to [solve these integrals analytically](https://en.wikipedia.org/wiki/Gini_coefficient#Continuous_probability_distribution), however, a straightforward computational approach is to use Monte Carlo sampling – as we shall see shortly. Personally, I find the above relative mean difference presentation of the Gini index much more intuitive than the area argument using the Lorenz curve. From the eqution it also becomes clear that the Gini coefficient is invariant to multiplicative changes in the income: if everybody increases their income by factor \(k>0\) then the Gini coefficient remains the same, because \(|k x – k y| = k | x – y|\) and \(E(k \cdot X) = k \mu\).

The above formula indirectly also states how to compute the Gini coefficient for a discrete sample of size \(n\) and with incomes \(x\_1,\ldots, x\_n\): \[  
G = \frac{\sum\_{i=1}^n \sum\_{j=1}^n |x\_i –  
x\_j| \frac{1}{n} \frac{1}{n}}{2 \sum\_{i=1}^n x\_i \frac{1}{n}} =  
\frac{\sum\_{i=1}^n \sum\_{j=1}^n |x\_i – x\_j|}{2 n \sum\_{j=1}^n x\_j}.  
\]

**Approximating the integral by Monte Carlo**

If one is able to easily sample from \(f\) then can instead of solving the integral analytically use \(k\) pairs \((x,y)\) both drawn at random from \(f\) to approximate the double integral:

\[  
G \approx \frac{1}{2\mu K} \sum\_{k=1}^K |x\_k – y\_k|, \quad\text{where}\quad  
x\_k \stackrel{\text{iid}}{\sim} f \text{ and } y\_k \stackrel{\text{iid}}{\sim} f,  
\] where for our mixture model \[  
\mu = \sum\_{i=1}^{192} w\_i \> E(X\_i) = \sum\_{i=1}^{192} w\_i \exp\left(\mu\_i + \frac{1}{2}\sigma\_i^2\right).  
\] This allows us to compute \(G\) even in case of a complex \(f\) such as the log-normal mixture distribution. As always, the larger \(K\) is, the better the Monte Carlo approximation is.

##Precision of Monte Carlo approx is controlled by the number of samples

n <- 1e6

##Compute Gini index of world income per year

gini\_year <- gm %>% group\_by(year) %>% do({

x <- rmix(n, meanlog=.$meanlog, sdlog= .$sdlog, w=.$w)

y <- rmix(n, meanlog=.$meanlog, sdlog= .$sdlog, w=.$w)

int <- mean( abs(x-y) )

mu <- sum(exp( .$meanlog + 1/2 \* .$sdlog^2) \* .$w)

data.frame(gini\_all\_mc=1/(2\*mu)\*int,

country\_gini=gini(.$gdp\*.$population))

}) %>%

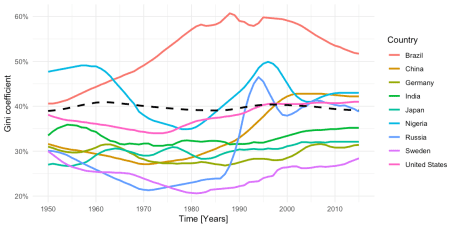
rename(`World Population GDP/capita`=gini\_all\_mc, `Country GDP/capita`=country\_gini)

##Convert to long format

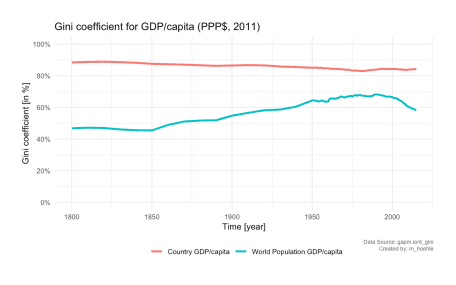
gini\_ts <- gini\_year %>% gather(key="type", value="gini", -year)

**Results**

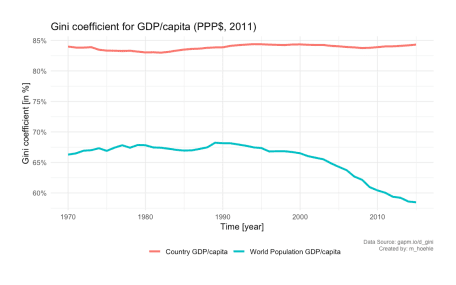
We start by showing the country specific Gini coefficient per year since 1950 for a somewhat arbitrary selection of countries. The dashed black line shows the mean Gini coefficient each year over all 192 countries in the dataset.



In addition, we now show the Monte Carlo computed Gini coefficient for the world’s income distribution given as a log-normal mixture with 192 components.

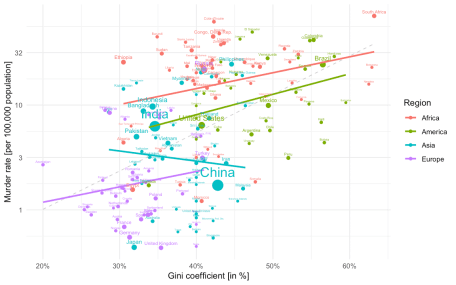


We notice that the Gini coefficient taken over the 192 countries’ GDP/capita remains very stable over time. This, however, does not take the large differences in populations between countries into account. A fairer measure is thus the Gini coefficient for the world’s income distribution. We see that this Gini coefficient increased over time until peaking around 1990. From then on it has declined. However, the pre-1950 Gini coefficients are rather guesstimates as stated by Gapminder, hence, we zoom in on the period from 1970, because data are more reliable from this point on.



**Gini coefficient and Homicide Rate**

Finally, we end the post by illustrating the association between the Gini coefficient and the homicide rate per country using a 2D scatterplot over the years. The Gapminder data download page also contains [data](https://docs.google.com/spreadsheet/pub?key=tZgPgT_sx3VdAuyDxEzenYA&output=xlsx) for this for the years 1950- 2005. Unfortunately, no data for more recent years are available from the Gapminder homepage, but the plot shown below is the situation in r show\_year with a log-base-10 y-axis for the homicide rates. For each of the four Gapminder regional groups we also fit a simple linear regression line to the points of all countries within the region. Findings such as Fajnzylber, Lederman, and Loayza (2002) suggest that there is a strong positive correlation between Gini coefficient and homicide rate, but we see from the plot that there are regional differences and of course correlation is not causality



We extend the plots to all years 1950-2005. Unfortunately, not all countries are available every year – so we only plot the available countries each year. This means that many African countries are missing from the animation. An improvement would be to try some form of linear interpolation. Furthermore, for the sake of simplicity of illustration, we fix countries with a reported murder rate of zero in a given year (happens for example for Cyprus, Iceland, Fiji in some years) to 0.01 per 100,000 population. This can be nicely animated using the new version of the gganimate pkg.

## New version of gganimate. Not on CRAN yet.

## devtools::install\_github('thomasp85/gganimate')

require(gganimate)

p <- ggplot(gm2\_nozero, aes(x=gini, y=murder\_rate,size=population, color=Region)) +

geom\_point() +

scale\_x\_continuous(labels=scales::percent) +

scale\_y\_continuous(trans="log10",

breaks = trans\_breaks("log10", function(x) 10^x,n=5),

labels = trans\_format("log10", function(x) ifelse(x<0, sprintf(paste0("%.",ifelse(is.na(x),"0",round(abs(x))),"f"),10^x), sprintf("%.0f",10^x)))) +

geom\_smooth(se=FALSE, method="lm", formula=y~x) +

geom\_text(data=gm2, aes(x=gini,y=murder\_rate, label=country), vjust=-0.9, show.legend=FALSE) +

ylab("Murder rate (per 100,000 population)") +

xlab("Gini coefficient (in %)") +

guides(size=FALSE) +

labs(title = 'Year: {frame\_time}') +

transition\_time(year) +

ease\_aes('linear')

animate(p, nframes=length(unique(gm2$year)), fps=4, width=800, height=400, res=100)

