BDA - Project

Tomi Räsänen - 879626 & Erik Husgafvel - 528867

Contents

1. Introduction

One of the biggest challenges of humankind in the 2020s is figuring out ways to slow down the growth of greenhouse gas emissions and stop global warming (due to human activities) under 2 °C. The increasing trend of global temperature is easily seen in Figure 1 [cite NASA] in which the global surface temperature is illustrated relative to 1951-1980 average temperatures. Warming can also be seen with one's own eyes by observing the winters that are warming year by year, by noticing that the number of devastating hurricanes is increased, and by finding out the increased rate of ice melting in glaciers during summer.

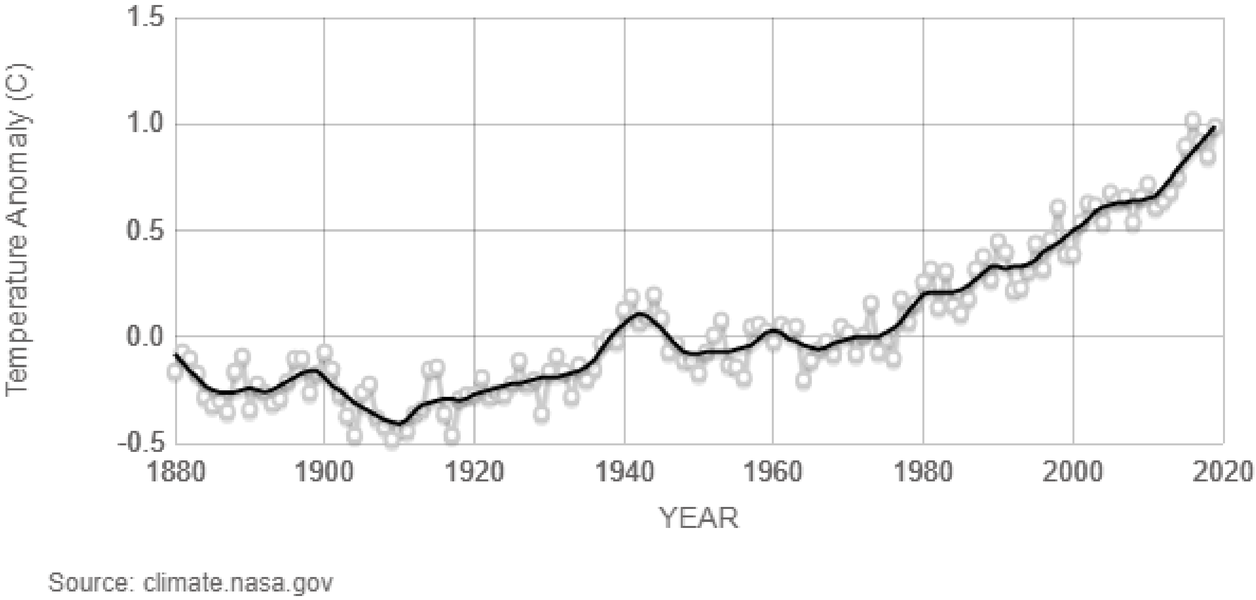


Figure 1: Global Land-Ocean Temperature Index

In response to that warming, many countries have declared a climate emergency to emphasize the criticality of the situation. In addition, young people have organized climate demonstrations around the world, politicians are talking more and more about climate change, and presidents and prime ministers are negotiating agreements and commitments to solve this, one of humanity's greatest, problem. But what if, despite attempts of negotiation, the necessary CO_2 reduction decisions are not achieved?

In this project, our goal is to model the historical emission trends of selected countries as well as attempts to model their future emissions. We are examining a scenario in which emissions continue to develop at a historical rate, and the necessary reductions are not achieved. In our modeling, the other parameters e.g. population growth and technical conditions, are similar to historical data in our modeling.

2. Data description

Our CO_2 data was obtained from *Our World in Data* (OWID) web page [cite OWID_net] and the actual *CSV* file from OWID GitHub page [cite OWID_git]. As mentioned earlier, climate change is a hot topic in the daily news, and there is a lot of studies and research concerning how CO_2 emissions are influencing global warming. The data set was also used, for example, when researchers studied the climate impact of the different policy recommendation which targeted to reduce greenhouse gases from the atmosphere.

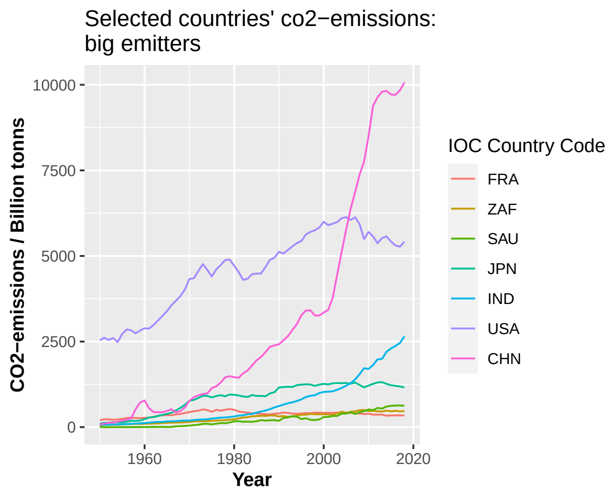
In our modeling, we are going to select 19 different countries from that OWID data set and examine CO_2 data between the years 1950-2018. The countries we have chosen cover the whole globe and are roughly evenly distributed across continents. There are both small and large polluters among the countries.

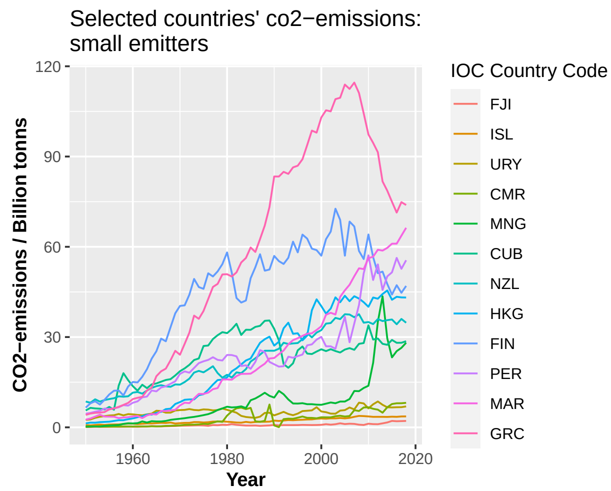
```r
library(rstan)
library(ggplot2)
library(reshape2)
library(gridExtra)
```

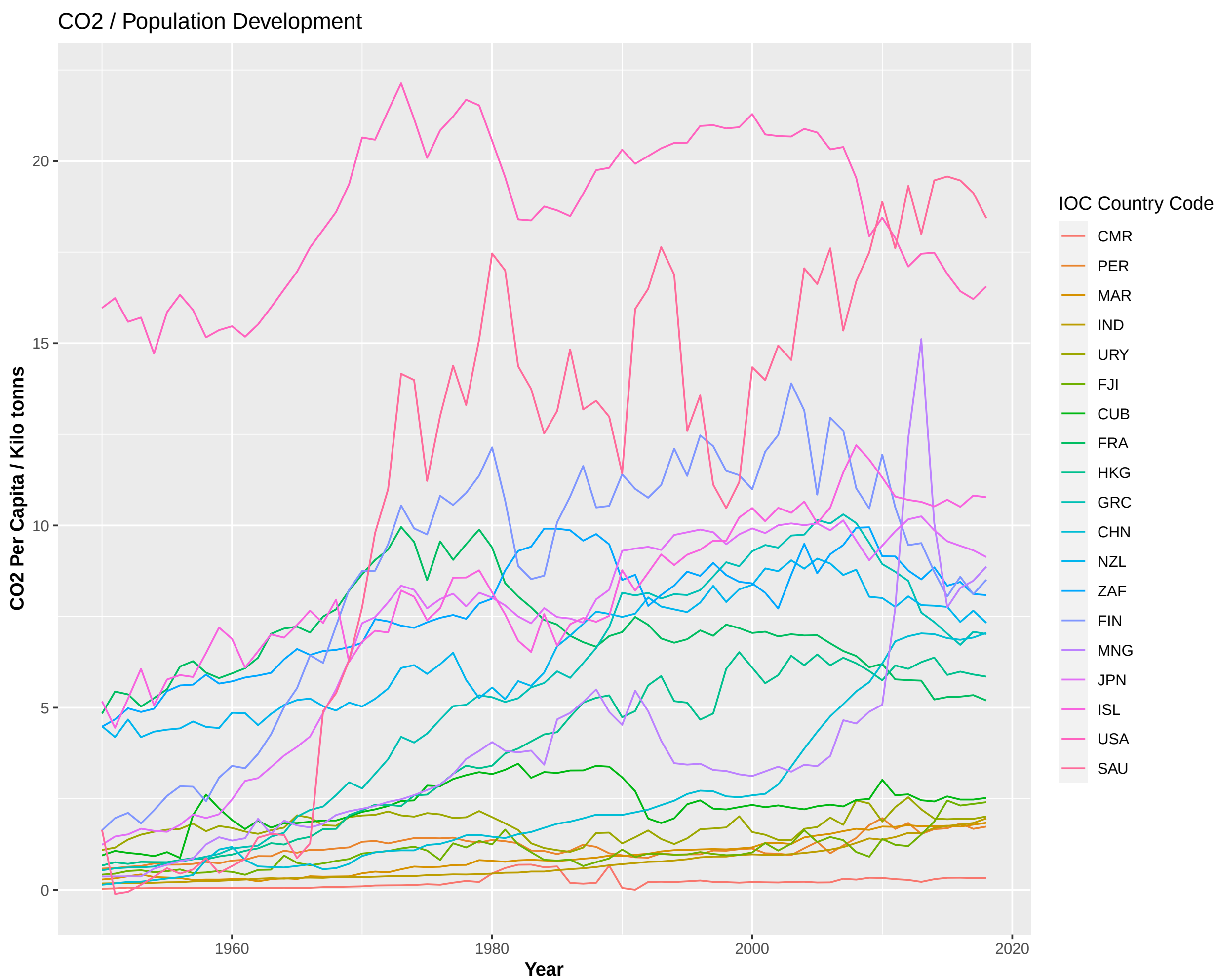
```r
# Read data to data frame
data_co2 <- read.csv("./data_co2.csv")
data_population <- head(read.csv("./data_population.csv"), -1)
data_co2_population = data_co2*10^6/data_population

# We discovered that the CO2-emissions difference between our selected countries is so vast
# that it's better to split the data into two different plots.

df_data1 <- data_co2[, (data_co2[dim(data_co2)[1], ]) >= 100]
df_data1 <- df_data1[,order(df_data1[69,])]
df_data1_2 <- data.frame(years=seq(1950,2018), df_data1)
df_plot1 <- melt(data = df_data1_2, id.vars = "years", variable.name = "Country")


df_data2 <- data_co2[, (data_co2[dim(data_co2)[1], ]) < 100]
df_data2 <- df_data2[,order(df_data2[69,])]
df_data2_2 <- data.frame(years=seq(1950,2018), df_data2)
df_plot2 <- melt(data = df_data2_2, id.vars = "years", variable.name = "Country")

# Population
# df_data3 <- data.frame(years=seq(1950,2018), data_population)
# df_plot3 <- melt(data = df_data3, id.vars = "years", variable.name = "Country")


df_data4 <- data.frame(years=seq(1950,2018), data_co2_population)
df_data4 <- df_data4[,order(df_data4[69,])]
df_plot4 <- melt(data = df_data4, id.vars = "years", variable.name = "Country")

# df_data5 <- data_c02_population[, (data_co2[dim(data_co2)[1], ]) < 100]
# df_data5 <- df_data5[,order(df_data5[69,])]
# df_data5_2 <- data.frame(years=seq(1950,2018), df_data5)
# df_plot5 <- melt(data = df_data5_2, id.vars = "years", variable.name = "Country")


plot1 <- ggplot(df_plot1, aes(x=years, y=value, colour=Country)) +
        geom_line() +
        ggtitle("Selected countries' co2-emissions: \nbig emitters") +
        xlab("Year") +
        ylab("CO2-emissions / Billion tonns") +
        labs(colour = "IOC Country Code") +
        theme(
          axis.title.x = element_text(face = "bold"),
          axis.title.y = element_text(face = "bold"))

plot2 <- ggplot(df_plot2, aes(x=years, y=value, colour=Country)) +
        geom_line() +
        ggtitle("Selected countries' co2-emissions: \nsmall emitters") +
        xlab("Year") +
        ylab("CO2-emissions / Billion tonns") +
```

```r
        labs(colour = "IOC Country Code") +
        theme(
          axis.title.x = element_text(face = "bold"),
          axis.title.y = element_text(face = "bold"))

# plot3 <- ggplot(df_plot3, aes(x=years, y=value, colour=Country)) +
#          geom_line() +
#          ggtitle("Selected countries' population development") +
#          xlab("Year") +
#          ylab("Population")

plot4 <- ggplot(df_plot4, aes(x=years, y=value, colour=Country)) +
        geom_line() +
        ggtitle("CO2 / Population Development") +
        xlab("Year") +
        ylab("CO2 Per Capita / Kilo tonns") +
        labs(colour = "IOC Country Code") +
        theme(
          axis.title.x = element_text(face = "bold"),
          axis.title.y = element_text(face = "bold"))

# plot5 <- ggplot(df_plot5, aes(x=years, y=value, colour=Country)) +
#          geom_line() +
#          ggtitle("co2 / Population development small") +
#          xlab("Year") +
#          ylab("co2 per capita")
```

```r
grid.arrange(
  grobs = list(plot1, plot2, plot4),
  layout_matrix = rbind(c(1, 1, 2, 2),
                        c(3, 3, 3, 3),
                        c(3, 3, 3, 3))
)
```








```
#plot5
```

```
data_co2_population = data_co2*10^6/data_population

df_data4 <- data.frame(years=seq(1950,2018), data_co2_population)
df_data4 <- df_data4[,order(df_data4[69,])]
```

```r
df_plot4 <- melt(data = df_data4, id.vars = "years", variable.name = "country")

#df_data5 <- data_c02_population[, (data_co2[dim(data_co2)[1], ]) < 100]
#df_data5 <- df_data5[,order(df_data5[69,])]
#df_data5_2 <- data.frame(years=seq(1950,2018), df_data5)
#df_plot5 <- melt(data = df_data5_2, id.vars = "years", variable.name = "country")

plot4 <- ggplot(df_plot4, aes(x=years, y=value, colour=country)) +
        geom_line() +
        ggtitle("CO2 / Population Development") +
        xlab("Year") +
        ylab("CO2 Per Capita / Kilo tonns")
```

Example code for pooled model from assignment

data { int <lower=0> N; // number of observations vector[N] y; // observations }

parameters { real mu; real<lower=0> sigma; }

model { mu ~ lognormal(0, 10); // priors from last week sigma ~ inv_chi_square(1); // priors from last week

// pooled model likelihood, common mu and sigma for all observations y ~ normal(mu, sigma); }

generated quantities { real ypred; //vector[N] log_lik;

//predictive distribution for any machine ypred = normal_rng(mu, sigma);

//for (i in 1:N){ // log_lik[i] = normal_lpdf(y[i] | mu, sigma); //} }

```r
# Setting seed to get same "random" results
SEED <- 12345

vectored_data <- data.frame(df_plot2[,'value'])
N <- nrow(vectored_data)

# # Printing out our hierarchical model
# writeLines(readLines("assignment9_hierarchical_model.stan"))


num_of_chains = 4
pool_data <- list(N = N,
                  y = vectored_data[,1])

pool_model <- rstan::stan_model(file = "pooled_model_stan.stan")
```

```
## Running /usr/lib/R/bin/R CMD SHLIB foo.c
## clang -flto=thin -std=gnu99 -I"/usr/share/R/include" -DNDEBUG   -I"/usr/local/lib/R/site-library/Rcpp
## In file included from <built-in>:1:
## In file included from /usr/local/lib/R/site-library/StanHeaders/include/stan/math/prim/mat/fun/Eigen
## In file included from /usr/local/lib/R/site-library/RcppEigen/include/Eigen/Dense:1:
## In file included from /usr/local/lib/R/site-library/RcppEigen/include/Eigen/Core:88:
## /usr/local/lib/R/site-library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:1: error: unknown ty
## namespace Eigen {
## ^
## /usr/local/lib/R/site-library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:16: error: expected
## namespace Eigen {
##                ^
##                ;
```

```
## In file included from <built-in>:1:
## In file included from /usr/local/lib/R/site-library/StanHeaders/include/stan/math/prim/mat/fun/Eigen
## In file included from /usr/local/lib/R/site-library/RcppEigen/include/Eigen/Dense:1:
## /usr/local/lib/R/site-library/RcppEigen/include/Eigen/Core:96:10: fatal error: 'complex' file not fou
## #include <complex>
##          ^~~~~~~~~
## 3 errors generated.
## make: *** [/usr/lib/R/etc/Makeconf:168: foo.o] Error 1
```

```r
pool_fit <- rstan::sampling(object = pool_model,
                            data = pool_data,
                            iter = 2000,
                            warmup = 1000,
                            seed = SEED)
```

```
##
## SAMPLING FOR MODEL 'pooled_model_stan' NOW (CHAIN 1).
## Chain 1: Rejecting initial value:
## Chain 1:   Error evaluating the log probability at the initial value.
## Chain 1: Exception: lognormal_lpdf: Random variable is -1.91091, but must be >= 0!  (in 'model75b5a03
##
## Chain 1: Rejecting initial value:
## Chain 1:   Error evaluating the log probability at the initial value.
## Chain 1: Exception: lognormal_lpdf: Random variable is -1.87978, but must be >= 0!  (in 'model75b5a03
##
## Chain 1:
## Chain 1: Gradient evaluation took 2.1e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.055376 seconds (Warm-up)
## Chain 1:                0.043356 seconds (Sampling)
## Chain 1:                0.098732 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'pooled_model_stan' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.1e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.11 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
```

```
## Chain 2: 
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2: 
## Chain 2:  Elapsed Time: 0.050893 seconds (Warm-up)
## Chain 2:                0.045864 seconds (Sampling)
## Chain 2:                0.096757 seconds (Total)
## Chain 2: 
## 
## SAMPLING FOR MODEL 'pooled_model_stan' NOW (CHAIN 3).
## Chain 3: 
## Chain 3: Gradient evaluation took 1e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.1 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3: 
## Chain 3: 
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3: 
## Chain 3:  Elapsed Time: 0.058536 seconds (Warm-up)
## Chain 3:                0.047852 seconds (Sampling)
## Chain 3:                0.106388 seconds (Total)
## Chain 3: 
## 
## SAMPLING FOR MODEL 'pooled_model_stan' NOW (CHAIN 4).
## Chain 4: 
## Chain 4: Gradient evaluation took 1.2e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4: 
## Chain 4: 
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
```

```
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.051911 seconds (Warm-up)
## Chain 4:                0.039784 seconds (Sampling)
## Chain 4:                0.091695 seconds (Total)
## Chain 4:
```

References