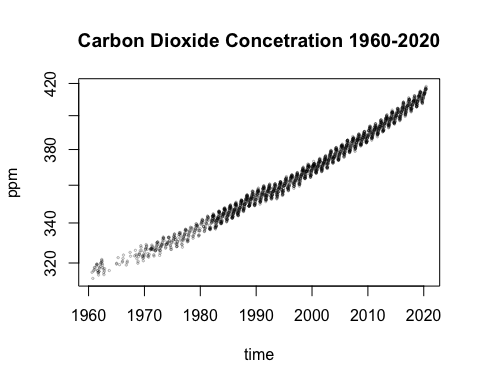
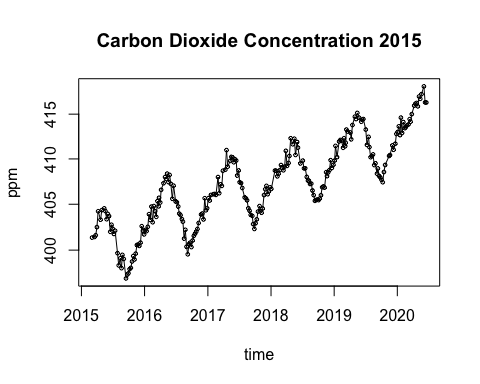
R Notebook

Form the plot we can see that the increasing trend of the carbon dioxide from 1960 to 2020.

cUrl = paste0("http://scrippsco2.ucsd.edu/assets/data/atmospheric/", "stations/flask\_co2/daily/daily\_flask\_co2\_mlo.csv")  
cFile = basename(cUrl)  
if (!file.exists(cFile)) download.file(cUrl, cFile)   
co2s = read.table(cFile, header = FALSE, sep = ",",  
skip = 69, stringsAsFactors = FALSE, col.names = c("day", "time", "junk1", "junk2", "Nflasks", "quality", "co2"))  
co2s$date = strptime(paste(co2s$day, co2s$time), format = "%Y-%m-%d %H:%M", tz = "UTC")  
# remove low-quality measurements  
co2s = co2s[co2s$quality == 0, ]  
plot(co2s$date, co2s$co2, log = "y", cex = 0.3, col = "#00000040", xlab = "time", ylab = "ppm", main = "Carbon Dioxide Concetration 1960-2020")



plot(co2s[co2s$date > ISOdate(2015, 3, 1, tz = "UTC"), c("date", "co2")], log = "y", type = "o", xlab = "time", ylab = "ppm", cex = 0.5, main = "Carbon Dioxide Concentration 2015")



co2s$day = as.Date(co2s$date)  
toAdd = data.frame(day = seq(max(co2s$day) + 3, as.Date("2025/1/1"),  
by = "10 days"), co2 = NA)  
co2ext = rbind(co2s[, colnames(toAdd)], toAdd)  
timeOrigin = as.Date("2000/1/1")  
co2ext$timeInla = round(as.numeric(co2ext$day - timeOrigin)/365.25,  
2)

co2ext$cos12 = cos(2 \* pi \* co2ext$timeInla)   
co2ext$sin12 = sin(2 \* pi \* co2ext$timeInla)   
co2ext$cos6 = cos(2 \* 2 \* pi \* co2ext$timeInla)  
co2ext$sin6 = sin(2 \* 2 \* pi \* co2ext$timeInla)  
library('INLA', verbose=FALSE)  
# disable some error checking in INLA  
mm = get("inla.models", INLA:::inla.get.inlaEnv())   
if(class(mm) == 'function') mm = mm()   
mm$latent$rw2$min.diff = NULL  
assign("inla.models", mm, INLA:::inla.get.inlaEnv())  
co2res = inla(co2 ~ sin12 + cos12 + sin6 + cos6 + f(timeInla, model = 'rw2',  
 prior='pc.prec', param = c(0.1, 0.5)), data = co2ext, family='gamma', control.family = list(hyper=list(prec=list(  
 prior='pc.prec', param=c(0.1, 0.5)))),  
 control.inla = list(strategy='gaussian'),  
 control.predictor = list(compute=TRUE, link=1), control.compute = list(config=TRUE), verbose=FALSE)  
qCols = c('0.5quant','0.025quant','0.975quant')   
Pmisc::priorPost(co2res)$summary[,qCols] # sd shows the change of slope so there is no change in slope(smooth)  
co2res$summary.fixed[,qCols] #sines and cosines account for the up and down of the graph which has 12 months cycle.

if (!requireNamespace("BiocManager", quietly = TRUE))  
 install.packages("BiocManager")  
  
BiocManager::install("Biobase")aas

sampleList = INLA::inla.posterior.sample(30, co2res, selection = list(timeInla = 0))   
sampleMean = do.call(cbind, Biobase::subListExtract(sampleList, "latent"))  
sampleDeriv = apply(sampleMean, 2, diff)/diff(co2res$summary.random$timeInla$ID)  
matplot(co2ext$day, co2res$summary.fitted.values[, qCols], type = "l", col = "black", lty = c(1, 2, 2), log = "y", xlab = "time", ylab = "ppm")   
Stime = timeOrigin + round(365.25 \* co2res$summary.random$timeInla$ID)   
matplot(Stime[-1], sampleDeriv, type = "l", lty=1, xaxs = "i", col = "#00000020", xlab = "time", ylab = "deriv",ylim = quantile(sampleDeriv, c(0.01, 0.995)), main= "FIrst derivative of CO2 concentration 1960-2020")  
  
forX = as.Date(c("2018/1/1", "2023/1/1"))  
forX = seq(forX[1], forX[2], by = "6 months")  
toPlot = which(Stime > min(forX) & Stime < max(forX))  
matplot(Stime[toPlot], sampleDeriv[toPlot, ], type = "l",  
lty = 1, lwd = 2, xaxs = "i", col = "#00000050",  
xlab = "time", ylab = "deriv", xaxt = "n", ylim = quantile(sampleDeriv[toPlot,  
], c(0.01, 0.995)),main = "FIrst derivaitve of CO2 concentration 2018-2020")  
axis(1, as.numeric(forX), format(forX, "%b%Y"))   
forY = as.Date(c("1985/1/1", "1996/1/1"))  
forY = seq(forY[1], forY[2], by = "6 months")  
toPlot1 = which(Stime > min(forY) & Stime < max(forY))  
matplot(Stime[toPlot1], sampleDeriv[toPlot1, ], type = "l",  
lty = 1, lwd = 1, xaxs = "i", col = "#00000040",  
xlab = "time", ylab = "deriv", xaxt = "n", ylim = quantile(sampleDeriv[toPlot,  
], c(0.01, 0.995)), main = "First derivative of CO2 1985-1995")  
axis(1, as.numeric(forY), format(forY, "%b%Y"))

1. Mortality on Quebec.

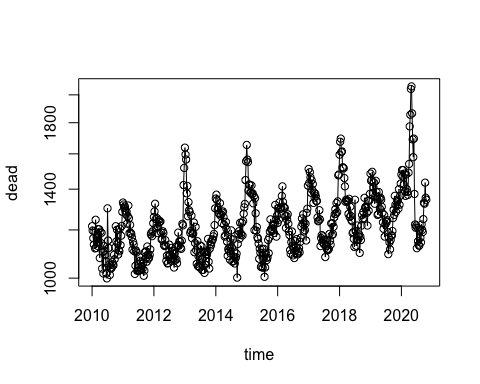
xWide = read.table(paste0("https://www.stat.gouv.qc.ca/statistiques/", "population-demographie/deces-mortalite/", "WeeklyDeaths\_QC\_2010-2020\_AgeGr.csv"), sep = ";", skip = 7, col.names = c("year", "junk",  
"age", paste0("w", 1:53)))  
xWide = xWide[grep("^[[:digit:]]+$", xWide$year), ]

## Warning in grep("^[[:digit:]]+$", xWide$year): input string 47 is invalid in  
## this locale

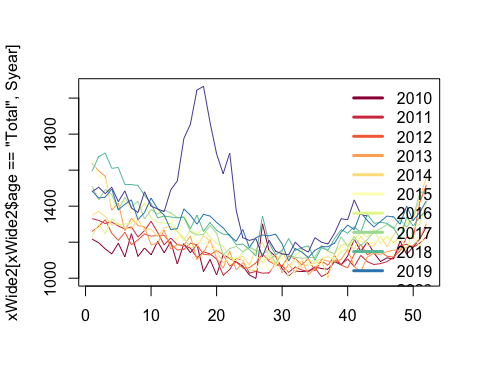
## Warning in grep("^[[:digit:]]+$", xWide$year): input string 50 is invalid in  
## this locale

## Warning in grep("^[[:digit:]]+$", xWide$year): input string 59 is invalid in  
## this locale

x = reshape2::melt(xWide, id.vars = c("year", "age"),  
measure.vars = grep("^w[[:digit:]]+$", colnames(xWide)))   
x$dead = as.numeric(gsub("[[:space:]]", "", x$value))   
x$week = as.numeric(gsub("w", "", x$variable))  
x$year = as.numeric(x$year)  
x = x[order(x$year, x$week, x$age), ] #convert the ‘week’ variable to time  
newYearsDay = as.Date(ISOdate(x$year, 1, 1))   
x$time = newYearsDay + 7 \* (x$week - 1)  
x = x[!is.na(x$dead), ]  
x = x[x$week < 53, ]  
  
plot(x[x$age == "Total", c("time", "dead")], type = "o", log = "y") # we get bigger spikes in the past with the flue vaccine. So these are often elderly people who have all sorts of other health problems.



xWide2 = reshape2::dcast(x, week + age ~ year, value.var = "dead")  
Syear = grep("[[:digit:]]", colnames(xWide2), value = TRUE)  
Scol = RColorBrewer::brewer.pal(length(Syear), "Spectral")   
matplot(xWide2[xWide2$age == "Total", Syear], type = "l",  
lty = 1, col = Scol)  
legend("topright", col = Scol, legend = Syear, bty = "n",  
lty = 1, lwd = 3)



head(x)

## year age variable value dead week time  
## 34 2010 0-49 years old w1 79 79 1 2010-01-01  
## 23 2010 50-69 years old w1 277 277 1 2010-01-01  
## 12 2010 70 years old and over w1 861 861 1 2010-01-01  
## 1 2010 Total w1 1 217 1217 1 2010-01-01  
## 78 2010 0-49 years old w2 82 82 2 2010-01-08  
## 67 2010 50-69 years old w2 286 286 2 2010-01-08

# to see excess deaths which makes this problem as GAMM problem. To fit again counterfactually if what happended in the pat world it continued what would be expeceted to see this year so the air accessing both mean amplitutde well, the mean there certainly seasonality in here.  
  
# When the covid death data was first coming, the first questions are there are some deaths taht we are not seeing..   
# In our data set there are total deaths by ages.   
# once for the 70 and once for less than 50

Divide the data into pre and post covid, add extra dates to data so that INLA will create forecasts.

dateCutoff = as.Date("2020/3/1")  
xPreCovid = x[x$time < dateCutoff, ]  
xPostCovid = x[x$time >= dateCutoff, ]  
toForecast = expand.grid(age = unique(x$age), time = unique(xPostCovid$time),  
dead = NA)  
xForInla = rbind(xPreCovid[, colnames(toForecast)],toForecast)  
xForInla = xForInla[order(xForInla$time, xForInla$age),]

dim(xForInla)

## [1] 2252 3

head(xForInla)

## age time dead  
## 34 0-49 years old 2010-01-01 79  
## 23 50-69 years old 2010-01-01 277  
## 12 70 years old and over 2010-01-01 861  
## 1 Total 2010-01-01 1217  
## 78 0-49 years old 2010-01-08 82  
## 67 50-69 years old 2010-01-08 286

xForInla[2235:2244,] # add the new data that I don't have dead.

## age time dead  
## 1199 70 years old and over 2020-09-23 NA  
## 12010 Total 2020-09-23 NA  
## 12110 0-49 years old 2020-09-30 NA  
## 1221 50-69 years old 2020-09-30 NA  
## 1232 70 years old and over 2020-09-30 NA  
## 1243 Total 2020-09-30 NA  
## 1254 0-49 years old 2020-10-07 NA  
## 1265 50-69 years old 2020-10-07 NA  
## 1276 70 years old and over 2020-10-07 NA  
## 1287 Total 2020-10-07 NA

xForInla$timeNumeric = as.numeric(xForInla$time)  
xForInla$timeForInla = (xForInla$timeNumeric - as.numeric(as.Date("2015/1/1")))/365.25   
xForInla$timeIid = xForInla$timeNumeric  
xForInla$sin12 = sin(2 \* pi \* xForInla$timeNumeric/365.25)  
xForInla$sin6 = sin(2 \* pi \* xForInla$timeNumeric \*  
2/365.25)  
xForInla$cos12 = cos(2 \* pi \* xForInla$timeNumeric/365.25)   
xForInla$cos6 = cos(2 \* pi \* xForInla$timeNumeric \*2/365.25)  
xForInlaTotal= xForInla[xForInla$age == 'Total', ]   
library(INLA, verbose=FALSE)

## Loading required package: Matrix

## Loading required package: sp

## Loading required package: parallel

## Loading required package: foreach

## This is INLA\_20.03.17 built 2020-10-27 02:19:26 UTC.  
## See www.r-inla.org/contact-us for how to get help.  
## To enable PARDISO sparse library; see inla.pardiso()

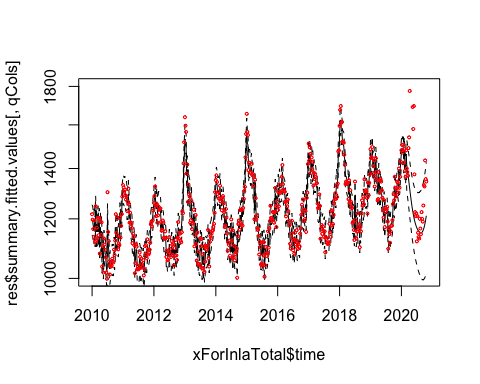
res = inla(dead ~ sin12 + sin6 + cos12 + cos6 +  
f(timeIid, prior='pc.prec', param= c(log(1.2), 0.5)) + f(timeForInla, model = 'rw2', prior='pc.prec', param= c(0.01, 0.5)),  
data=xForInlaTotal,  
control.predictor = list(compute=TRUE, link=1), control.compute = list(config=TRUE),  
# control.inla = list(fast=FALSE, strategy='laplace'),  
family='poisson')  
  
qCols = paste0(c(0.5, 0.025, 0.975), "quant")   
rbind(res$summary.fixed[, qCols], Pmisc::priorPostSd(res)$summary[,qCols])

## 0.5quant 0.025quant 0.975quant  
## (Intercept) 7.10144269 7.094837392 7.10779179  
## sin12 0.05140123 0.044891528 0.05796335  
## sin6 0.01127193 0.005632170 0.01690054  
## cos12 0.09736359 0.090828803 0.10395724  
## cos6 0.01302343 0.007378945 0.01865080  
## SD for timeIid 0.03564886 0.032055158 0.03955971  
## SD for timeForInla 0.13827806 0.089458363 0.21216269

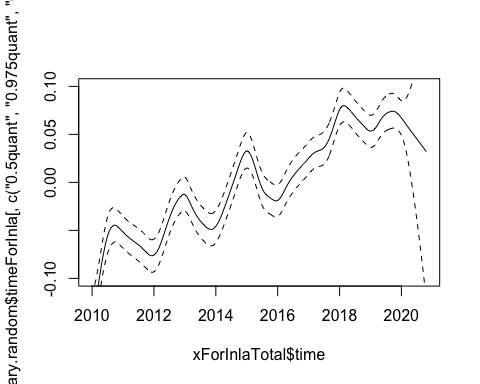
$$ Y\_{i} ~ poisson(O\_{i}\_{i})

$$

matplot(xForInlaTotal$time, res$summary.fitted.values[,  
qCols], type= "l", ylim = c(1000, 1800), lty = c(1,2, 2),col ="black", log = "y")  
points(x[x$age== "Total", c("time", "dead")], cex = 0.4,col = "red")

 Th

matplot(xForInlaTotal$time, res$summary.random$timeForInla[, c("0.5quant", "0.975quant", "0.025quant")], type = "l", lty = c(1, 2, 2), col = "black", ylim = c(-1, 1) \*  
0.1) # this is after taking out seasonality

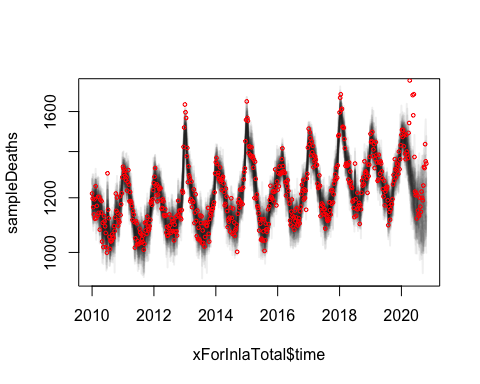


Take posterior samples of the intensity

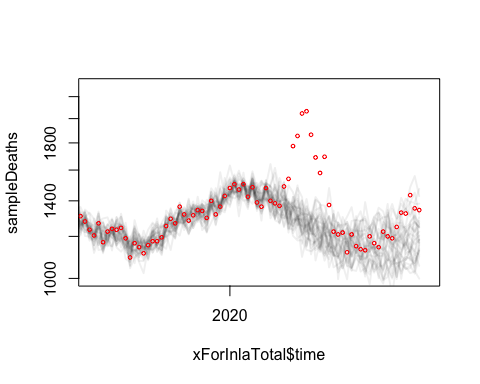
# get posterior samples  
sampleList = INLA::inla.posterior.sample(30, res, selection = list(Predictor = 0)) # prdictors = 0 is samples of log(\lmabda(t)) turning that into a matrix   
sampleIntensity = exp(do.call(cbind, Biobase::subListExtract(sampleList,"latent"))) # and exponentiate it them to put them into natural sclae  
sampleDeaths = matrix(rpois(length(sampleIntensity), #generating deaths from poisson distribution  
sampleIntensity), nrow(sampleIntensity), ncol(sampleIntensity))

plot samples and real data

matplot(xForInlaTotal$time, sampleDeaths, col = "#00000010", lwd = 2, lty = 1, type = "l", log = "y") # graph of posterior samples  
points(x[x$age == "Total", c("time", "dead")], col = "red", cex = 0.5) # actual datas



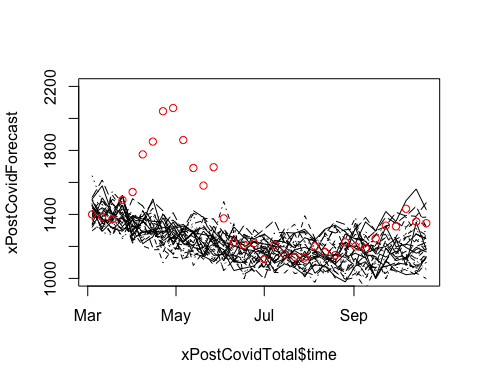
matplot(xForInlaTotal$time, sampleDeaths, col = "#00000010",  
lwd = 2, lty = 1, type = "l", log = "y", xlim = as.Date(c("2019/6/1",  
"2020/11/1")), ylim = c(1, 2.3) \* 1000)  
points(x[x$age == "Total", c("time", "dead")], col = "red", cex = 0.5) # these red dots are real deaths after the covid.

 calculate excess deaths

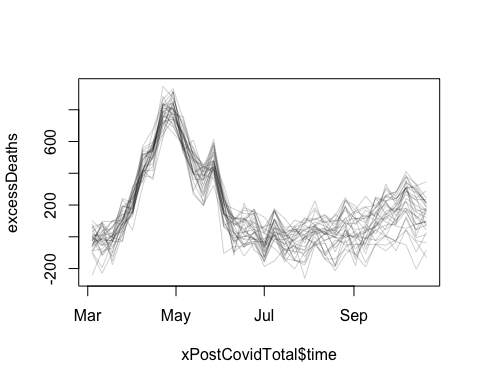
xPostCovidTotal = xPostCovid[xPostCovid$age == "Total", ] # this is number of deaths each week  
xPostCovidForecast = sampleDeaths[match(xPostCovidTotal$time, xForInlaTotal$time), ]#5 posterior samples of what we would have expected.   
excessDeaths = xPostCovidTotal$dead - xPostCovidForecast

plot samples of excess deaths

matplot(xPostCovidTotal$time, xPostCovidForecast, type = "l", ylim = c(1000, 2200), col = "black")  
points(xPostCovidTotal[, c("time", "dead")], col = "red") # so what is the difference between red and the black that is the next graph # the black line represents the covid free world and red is actual death



matplot(xPostCovidTotal$time, excessDeaths, type = "l",lty = 1, col = "#00000030")

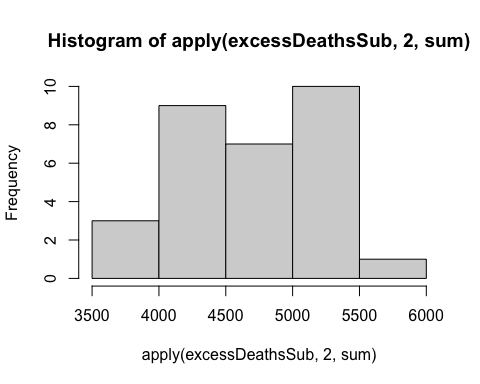


Total excess deaths march-may inclusive

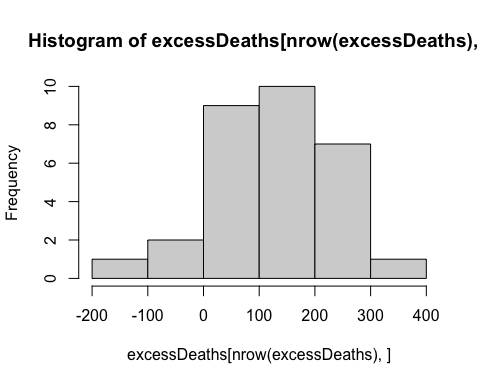
excessDeathsSub = excessDeaths[xPostCovidTotal$time > as.Date("2020/03/01") & xPostCovidTotal$time < as.Date("2020/06/01"), ] # sum up the excess deaths between March 1 and June 1  
excessDeathsInPeriod = apply(excessDeathsSub, 2, sum)   
round(quantile(excessDeathsInPeriod))

## 0% 25% 50% 75% 100%   
## 3561 4354 4738 5070 5963

hist(apply(excessDeathsSub,2,sum)) # there are between 4000~ 5000 excess deaths.



hist(excessDeaths[nrow(excessDeaths),])



Excess deaths in most recent week

round(quantile(excessDeaths[nrow(excessDeaths), ]))

## 0% 25% 50% 75% 100%   
## -129 78 162 202 346

matplot(excessDeathsSub)  
lines(excessDeathsSub[,1])  
lines(excessDeathsSub[,2], col = 'red')

