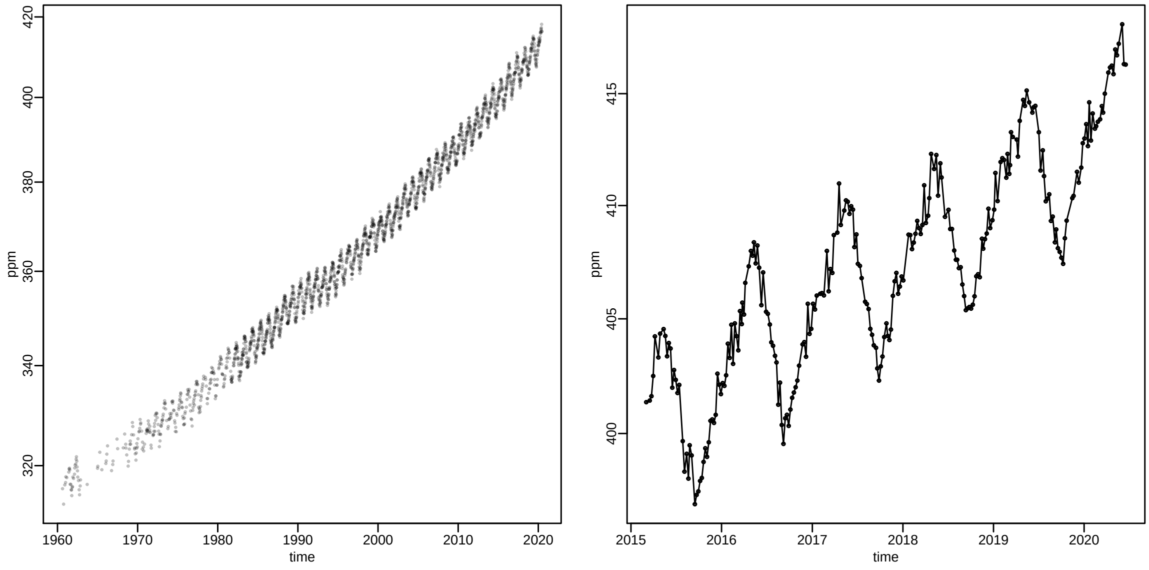
Homework 3, Smoothing

STA442 Methods of Applied Statistics

Due Friday 20 November 2020

1 CO2

Figure 1 shows atmoshperic Carbon Dioxide concentrations from an observatory in Haiwaii, made available by the Scripps CO_2 Program at scrippsco2.ucsd.edu. The figure was produced with code in the appendix.



(a) all

(b) recent

Figure 1: CO2 at Mauna Loa Observatory, Hawaii

Write a short consulting report (roughly a page of writing) discussing if the CO2 data appears to be impacted by the following events:

- the fall of the Berlin wall in November 1989 years ago, preceding a dramatic fall in industrial production in the Soviet Union and Eastern Europe;
- the global lockdown during the COVID-19 pandemic starting in February 2020, shutting down much of the global economy.

You should

- explain fully the model you are using and why you have chosen to use it
- make your graphs look nice
- at a minimum, plot the estimated smoothed trend of CO2 and discuss whether it appears shallower or steeper after the events listed above.
- visual investigation is sufficient, you aren't expected for formally test for effects.

2 Death

Daily mortality counts in Quebec is available from www.stat.gouv.qc.ca/statistiques/population-demographie/deces-mortalite/nombre-hebdomadaire-deces_an.html

An imaginary government official believes that the first wave of the COVID-19 epidemic, in March, April and May, primarily affected the elderly. The second wave, which began in September, is caused by irresponsible young people, primarily university undergraduates, acting irresponsibly. Evidence of this can be seen in the weekly mortality counts. Deaths amongst the elderly in the spring were well above the historical averages, whereas the under 50's had deaths in line with previous years. In the most recent death data, there is an increase in deaths in the under 50's whereas the over 70's have no more deaths than would be expected pre-covid.

Your task is to write a report explaining whether or not the mortality data support the above hypotheses. The code below calculates total excess mortality in the spring and the fall. You will probably find it useful to repeat the analysis twice, once for 70+ and once for <50.

Write a short report (2 pages of text, plus figures and tables).

- Make your report self-contained, so it can be understood by someone who has not seen the assignment sheet.
- You are expected to use Bayesian inference and some sort of semi-parametric time trend, because that's what this assignment is evaluating you on. However, you're free to use different software or a different model if you'd like.
- clearly explain the statistical model you've used with equations, and state your prior distributions. Explain how your priors are reasonable.
- consider plotting quantiles in place of the posterior samples I've plotted
- make your figures and tables look nice and properly captioned.

Appendix

CO2

```r
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, ]
```

```r
plot(co2s$date, co2s$co2, log = "y", cex = 0.3, col = "#00000040",
  xlab = "time", ylab = "ppm")
plot(co2s[co2s$date > ISOdate(2015, 3, 1, tz = "UTC"),
  c("date", "co2")], log = "y", type = "o", xlab = "time",
  ylab = "ppm", cex = 0.5)
```

The code below might prove useful.

```r
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)))),
  # add this line if your computer has trouble
#  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]
```

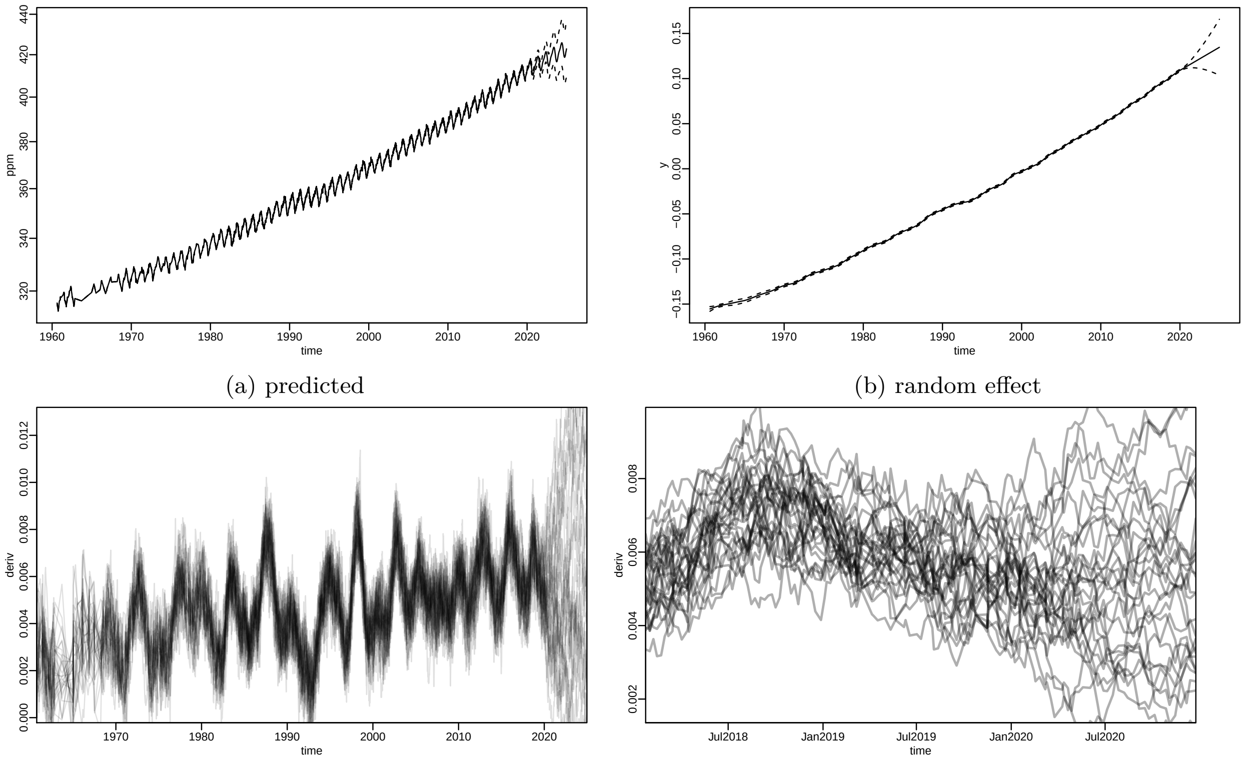
```
                   0.5quant   0.025quant   0.975quant
sd for gamma    2.89227e-06 2.831783e-06 2.941706e-06
sd for timeInla 2.67051e-03 2.546393e-03 2.749882e-03
```

```
# source('https://bioconductor.org/biocLite.R')
# biocLite('Biobase')
```

```
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$ID)
matplot(Stime, co2res$summary.random$timeInla[, qCols],
  type = "l", col = "black", lty = c(1, 2, 2), xlab = "time",
  ylab = "y")
matplot(Stime[-1], sampleDeriv, type = "l", lty = 1,
  xaxs = "i", col = "#00000020", xlab = "time", ylab = "deriv",
  ylim = quantile(sampleDeriv, c(0.01, 0.995)))
forX = as.Date(c("2018/1/1", "2021/1/1"))
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)))
axis(1, as.numeric(forX), format(forX, "%b%Y"))
```



(a) predicted

(b) random effect

(c) derivative

(d) derivative, detail

Figure 2: INLA results

Covid

Download some data

```
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), ]
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 two different ways

```
plot(x[x$age == "Total", c("time", "dead")], type = "o",
  log = "y")
```

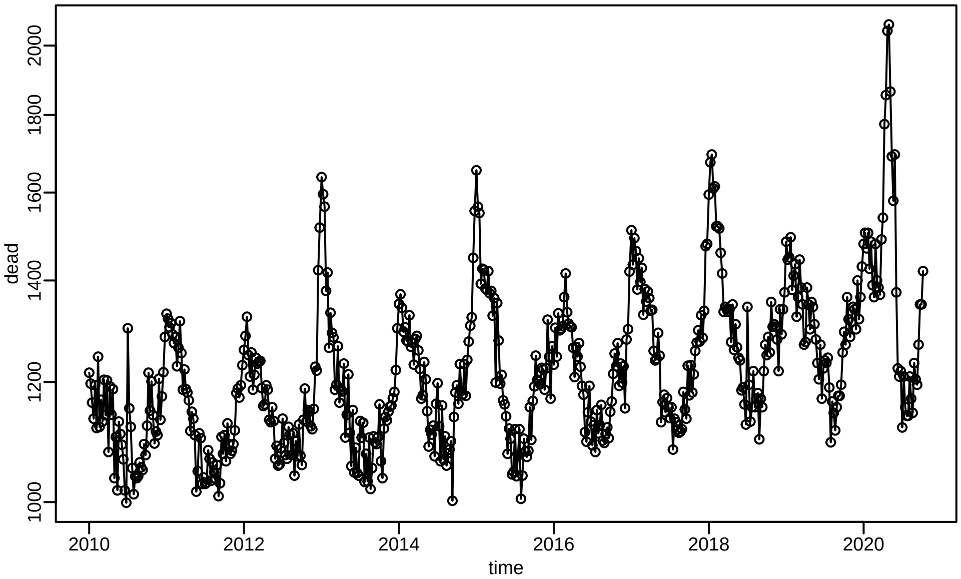


Figure 3

```
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)
```

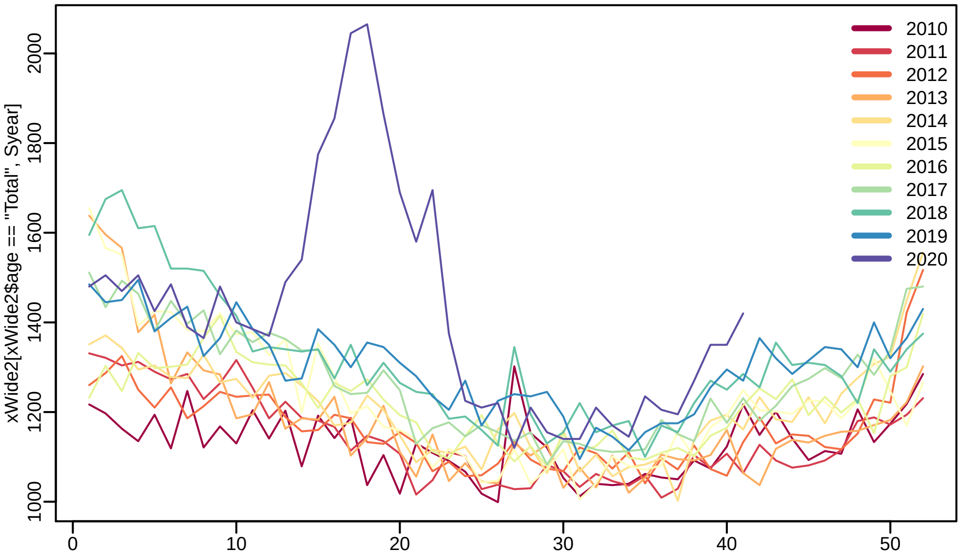


Figure 4

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),
  ]
```

Create some time variables, including sines and cosines. Time in years and centred is numerically stable in INLA.

```
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)
```

fit a model for total deaths in INLA

```
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_20.08.11-1 built 2020-08-11 09:32:13 UTC.
 - See www.r-inla.org/contact-us for how to get help.
 - To enable PARDISO sparse library; see inla.pardiso()
 - Save 425.1Mb of storage running 'inla.prune()'
```

```
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')
```

parameters

```
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)        7.10132633 7.095135708 7.10730664
sin12              0.05140121 0.044891417 0.05796350
```

```
sin6               0.01127193 0.005632171 0.01690051
cos12              0.09736361 0.090828734 0.10395743
cos6               0.01302345 0.007378958 0.01865081
SD for timeIid     0.03564929 0.032055718 0.03955956
SD for timeForInla 0.13827167 0.089468447 0.21211178
```

Plot predicted intensity and random effect

```
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")
```

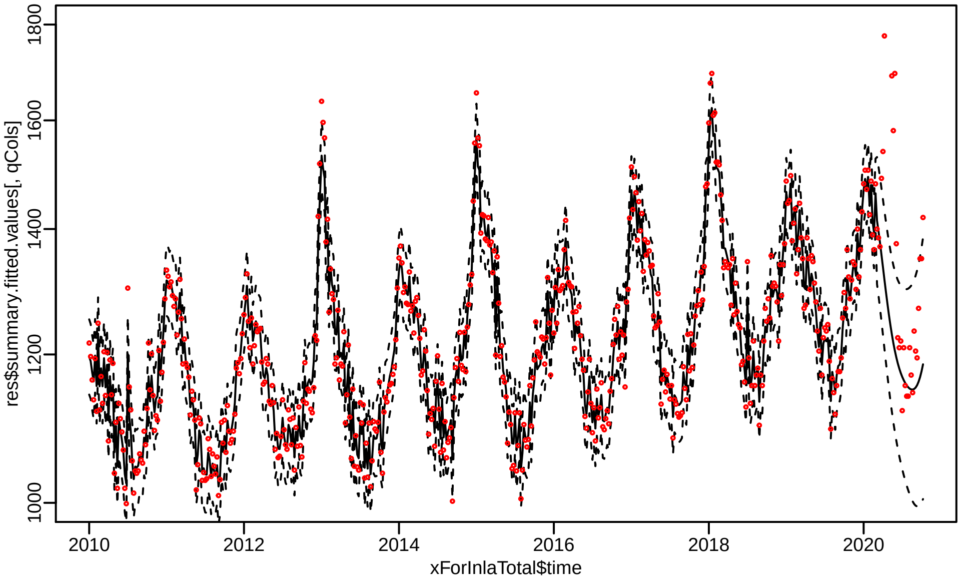


Figure 5

```
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)
```

Take posterior samples of the intensity

```
sampleList = INLA::inla.posterior.sample(30, res, selection = list(Predictor = 0))
sampleIntensity = exp(do.call(cbind, Biobase::subListExtract(sampleList,
  "latent")))
sampleDeaths = matrix(rpois(length(sampleIntensity),
  sampleIntensity), nrow(sampleIntensity), ncol(sampleIntensity))
```

plot samples and real data

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

```
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)
```

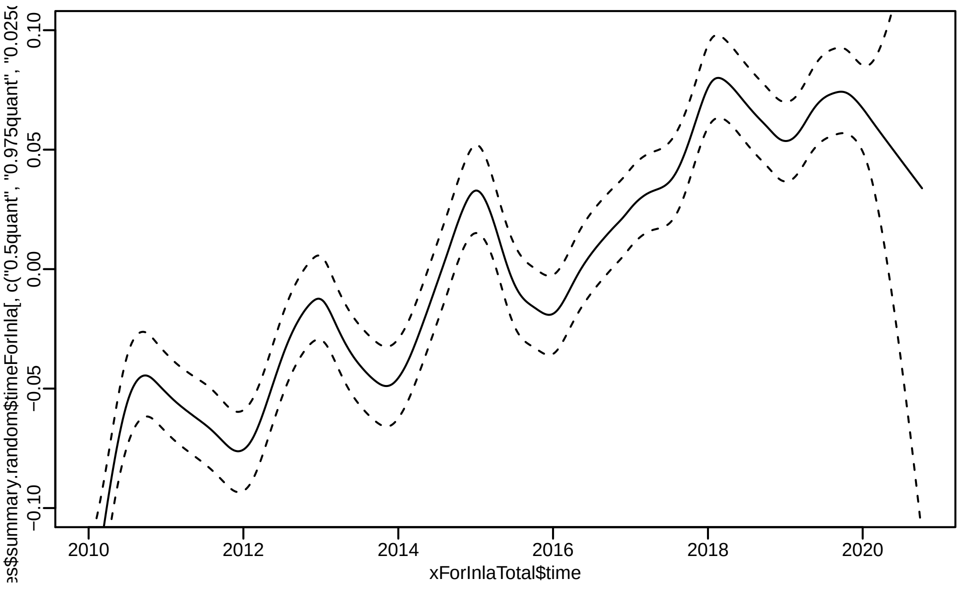


Figure 6

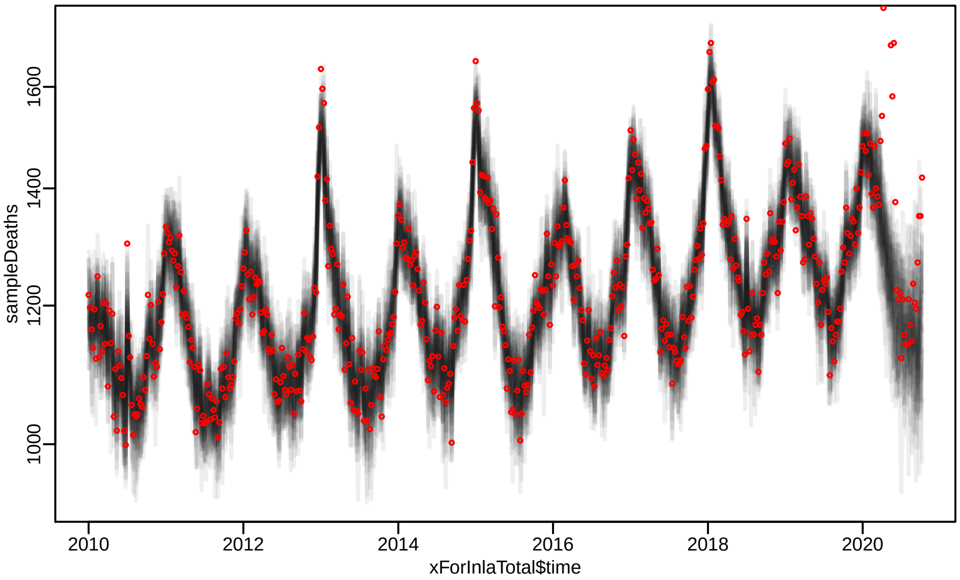


Figure 7

```
points(x[x$age == "Total", c("time", "dead")], col = "red",
  cex = 0.5)
```

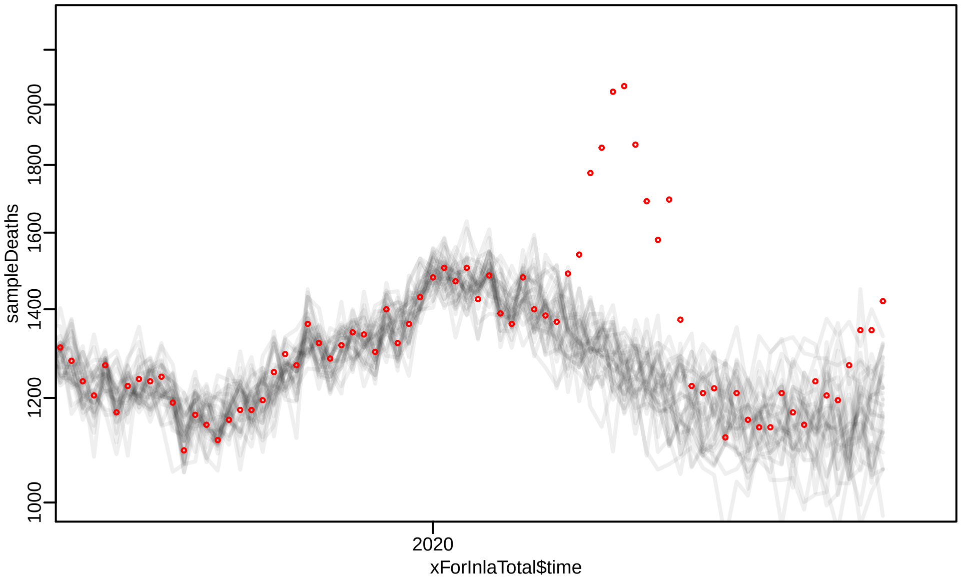


Figure 8

calculate excess deaths

```
xPostCovidTotal = xPostCovid[xPostCovid$age == "Total",
  ]
xPostCovidForecast = sampleDeaths[match(xPostCovidTotal$time,
  xForInlaTotal$time), ]
excessDeaths = xPostCovidTotal$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")
```

```
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"), ]
excessDeathsInPeriod = apply(excessDeathsSub, 2, sum)
round(quantile(excessDeathsInPeriod))
```

```
  0%  25%  50%  75% 100%
4098 4568 4727 5071 5859
```

Excess deaths in most recent week

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

```
  0%  25%  50%  75% 100%
  84  158  230  289  443
```

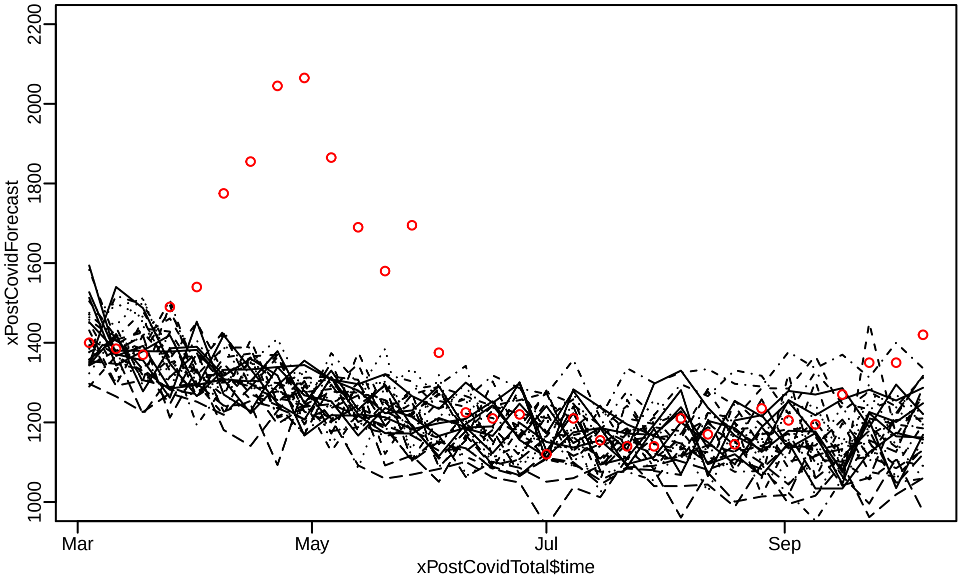


Figure 9

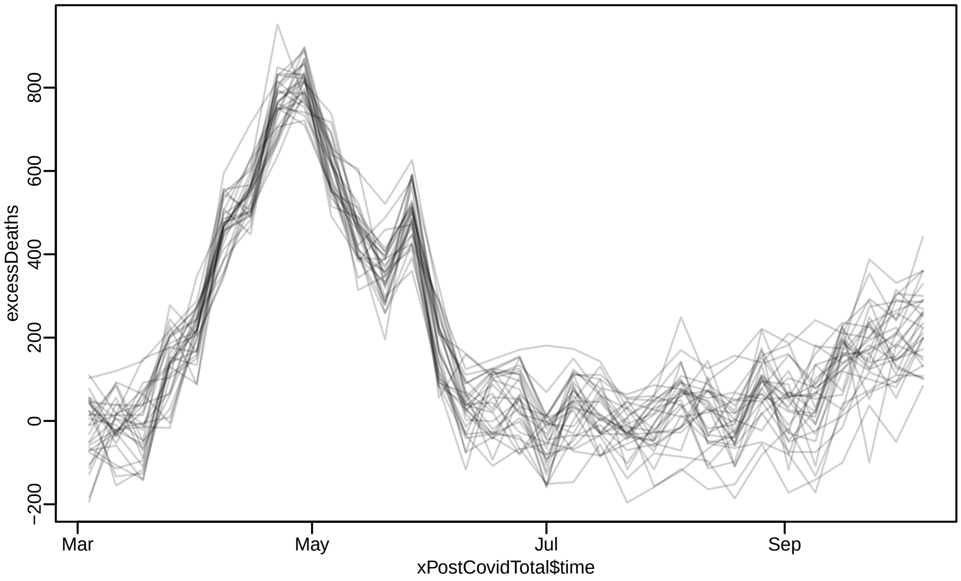


Figure 10