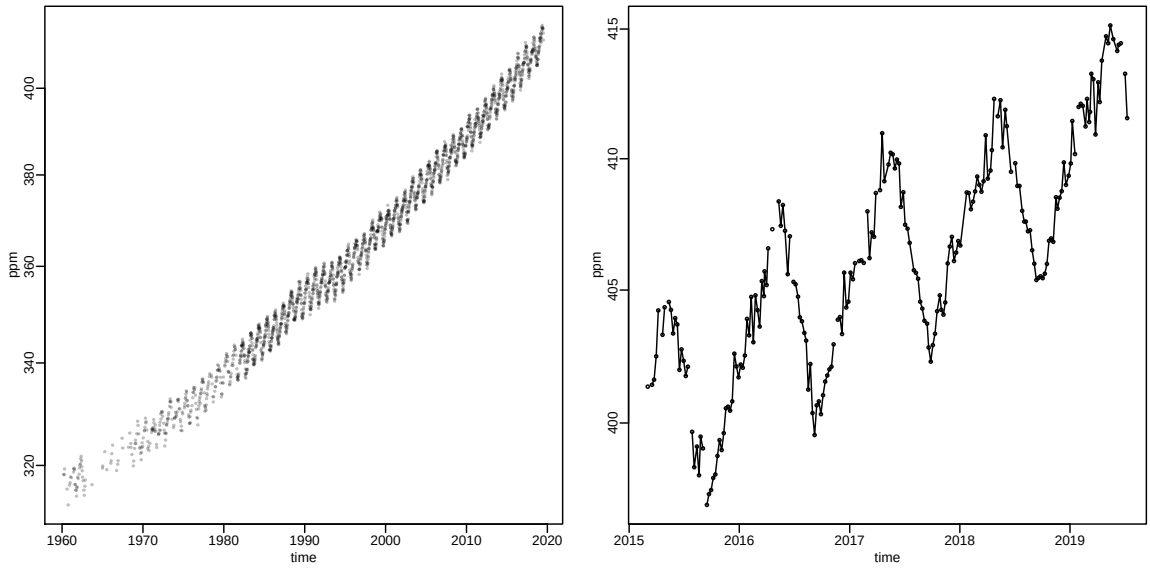
Homework 3, Smoothing

STA442 Methods of Applied Statistics

Due 11 November 2019

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 OPEC oil embargo which began in October 1973;
- the global economic recessions around 1980-1982;
- the fall of the Berlin wall almost exactly 30 years ago, preceding a dramatic fall in industrial production in the Soviet Union and Eastern Europe;
- China joining the WTO on 11 December 2001, which was followed by rapid growth in industrial production;
- the bankruptcy of Lehman Brothers on 15 September 2008, regarded as the symbolic start of the most recent global financial crisis; and
- the signing of the Paris Agreement on 12 December 2015, intended to limit CO2 emissions.

This last event is particularly important, as it suggests the growth rate of CO2 in the atmosphere should be lower now that it has been in the recent past.

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.
- you could consider estimating the derivative of the trend. The help files for `predict.gam` show how to do this, and some code doing it in `inla` are in the appendix.
- visual investigation is sufficient, you aren't expected for formally test for effects.

2 Heat

Figure 2 a shows daily maximum temperature data recorded on Sable Island, off the coast of Nova Scotia. Figure 2 b shows the period from 2016 to the present, with summer months (May to October inclusive) in black and winter in red. Figure 2 c shows an estimated time trend produced using code in the appendix, and Figure 2 d shows posterior samples of this trend. Notice that the winter temperatures are more variable than summer temperatures, and you can assume you have been advised by a reliable environmental scientist that it is advisable to consider only summer temperatures when modelling historical temperature time series (since winter temperatures are governed by a different and much more complex physical process).

The IPCC states

> Human activities are estimated to have caused approximately 1.0°C of global warming above pre-industrial levels, with a likely range of 0.8°C to 1.2°C. Global warming is likely to reach 1.5°C between 2030 and 2052 if it continues to increase at the current rate. (high confidence)

see www.ipcc.ch/sr15/resources/headline-statements

Your task is to prepare a short report (at most 2 pages of writing) discussing whether the data from Sable Island is broadly supportive of this statement from the IPCC. If you wish, you could write a report as a response to the following letter.

To: You

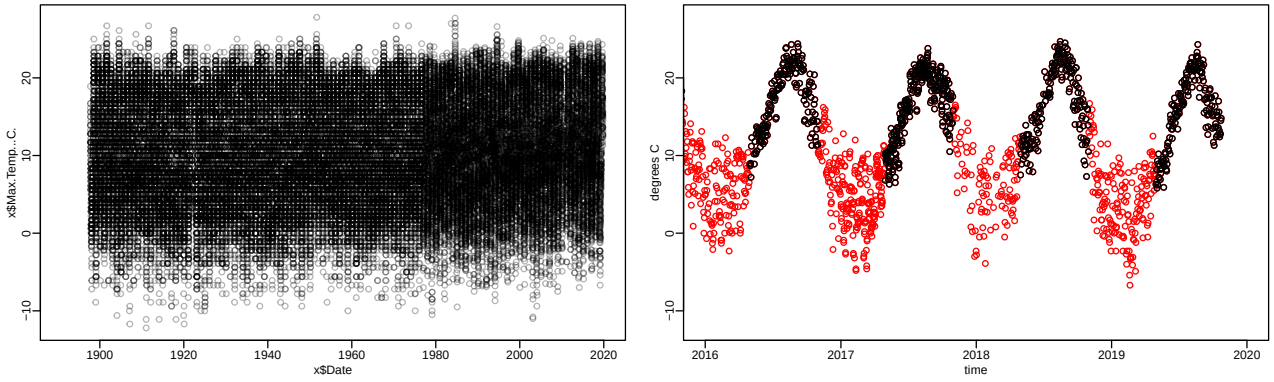
From: Maxim Burningier

Dear highly talented Statistician,

As you are no doubt aware, my political party believes that "Climate change alarmism is based on flawed models that have consistently failed at correctly predicting the future." and "CO2 is beneficial for agriculture and there has recently been a measurable "greening" of the world in part thanks to higher levels." (see www.peoplespartyofcanada.ca/platform). I have made a simple scatterplot of temperature measured at Sable Island over time and see no relationship whatsoever. I would like to employ you to write a two-page consulting report using the Sable Island data to refute the IPCC's irresponsible statements about global temperature rises. My enemies will no doubt give your report to deluded scientists who will comb over your methods and results looking for flaws, so you must clearly explain the model you are using and justify any model assumptions and prior distributions. Make your report self-contained so as to be understandable by someone who has not read the instructions I'm giving you. In return for this report I will compensate you with 100 barrels of unprocessed bitumen, a delightful substance which you may enjoy drinking with dinner or mixing into your bath water to assist with relaxation.

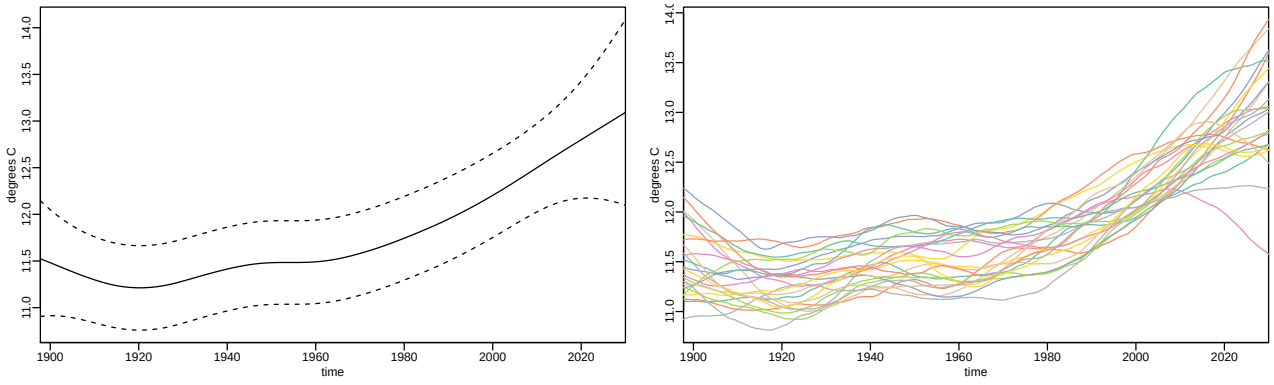
Many thanks in anticipation

Maxim-the-denier



(a) full

(b) zoom



(c) fit

(d) samples

Figure 2: Sable island temperature data

Appendix

CO2

```
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$time), format = "%Y-%m-%d %H:%M",
  tz = "UTC")
# remove low-quality measurements
co2s[co2s$quality >= 1, "co2"] = NA
```

```
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)
```

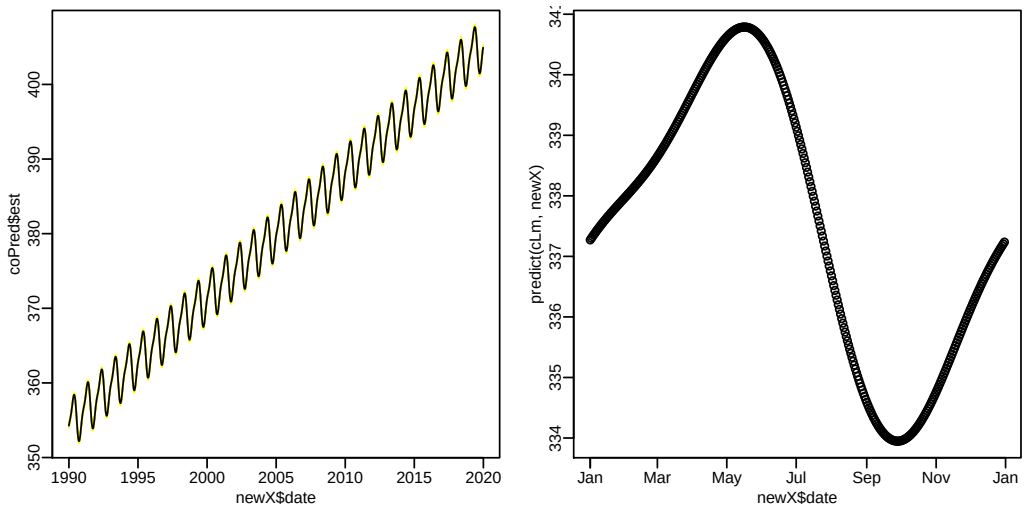
The code below might prove useful.

```
timeOrigin = ISOdate(1980, 1, 1, 0, 0, 0, tz = "UTC")
co2s$days = as.numeric(difftime(co2s$date, timeOrigin,
  units = "days"))
co2s$cos12 = cos(2 * pi * co2s$days/365.25)
co2s$sin12 = sin(2 * pi * co2s$days/365.25)
co2s$cos6 = cos(2 * 2 * pi * co2s$days/365.25)
co2s$sin6 = sin(2 * 2 * pi * co2s$days/365.25)
cLm = lm(co2 ~ days + cos12 + sin12 + cos6 + sin6,
  data = co2s)
summary(cLm)$coef[, 1:2]
```

```
                Estimate   Std. Error
(Intercept) 337.499286660 1.027025e-01
days          0.004651719 1.277835e-05
cos12        -0.898874589 9.274641e-02
sin12         2.884495702 9.153367e-02
cos6          0.657621761 9.245452e-02
sin6         -0.613041747 9.181487e-02
```

```
newX = data.frame(date = seq(ISOdate(1990, 1, 1, 0,
  0, 0, tz = "UTC"), by = "1 days", length.out = 365 *
  30))
newX$days = as.numeric(difftime(newX$date, timeOrigin,
  units = "days"))
newX$cos12 = cos(2 * pi * newX$days/365.25)
newX$sin12 = sin(2 * pi * newX$days/365.25)
newX$cos6 = cos(2 * 2 * pi * newX$days/365.25)
newX$sin6 = sin(2 * 2 * pi * newX$days/365.25)
coPred = predict(cLm, newX, se.fit = TRUE)
coPred = data.frame(est = coPred$fit, lower = coPred$fit -
  2 * coPred$se.fit, upper = coPred$fit + 2 * coPred$se.fit)
```

```
plot(newX$date, coPred$est, type = "l")
matlines(as.numeric(newX$date), coPred[, c("lower",
  "upper", "est")], lty = 1, col = c("yellow", "yellow",
  "black"))
newX = newX[1:365, ]
newX$days = 0
plot(newX$date, predict(cLm, newX))
```



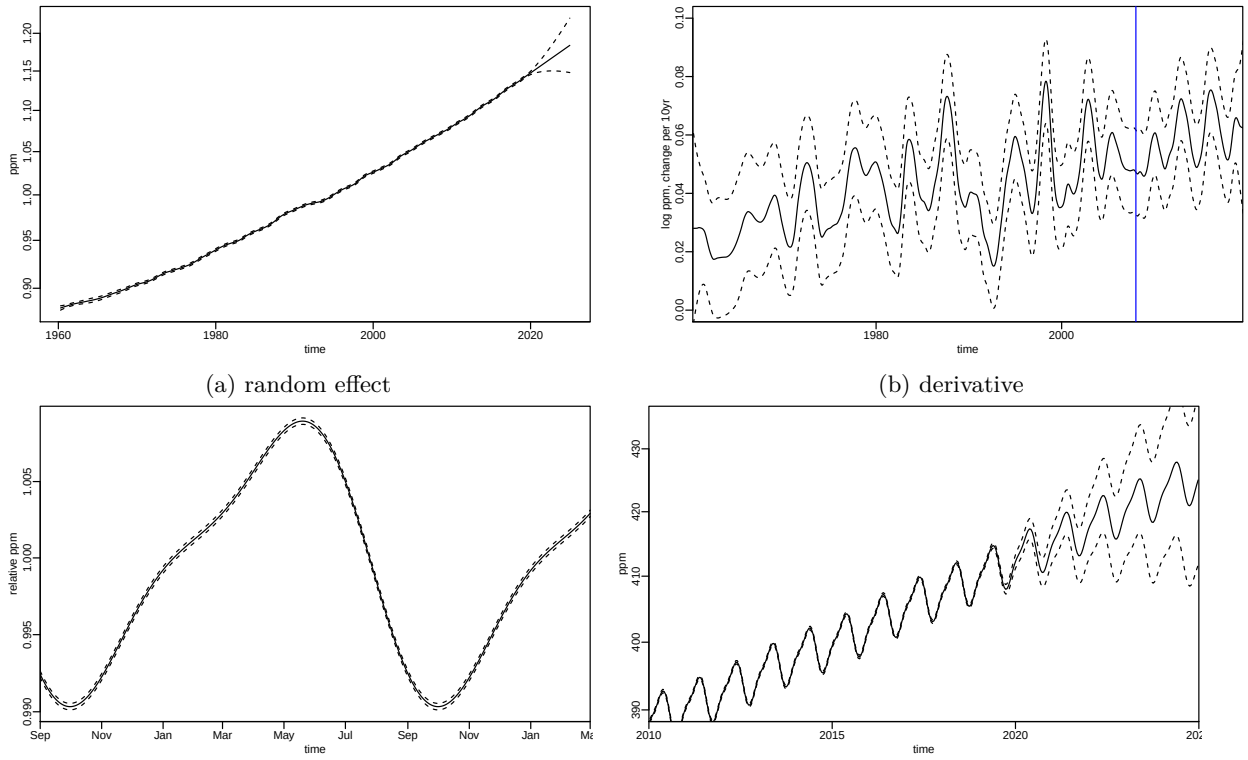
(a) forecast (b) cycle

Figure 3: Results

```
library("INLA")
# time random effect
timeBreaks = seq(min(co2s$date), ISOdate(2025, 1, 1,
  tz = "UTC"), by = "14 days")
timePoints = timeBreaks[-1]
co2s$timeRw2 = as.numeric(cut(co2s$date, timeBreaks))
# derivatives of time random effect
D = Diagonal(length(timePoints)) - bandSparse(length(timePoints),
  k = -1)
derivLincomb = inla.make.lincombs(timeRw2 = D[-1, ])
names(derivLincomb) = gsub("^lc", "time", names(derivLincomb))
# seasonal effect
StimeSeason = seq(ISOdate(2009, 9, 1, tz = "UTC"),
  ISOdate(2011, 3, 1, tz = "UTC"), len = 1001)
StimeYear = as.numeric(difftime(StimeSeason, timeOrigin,
  "days"))/365.35
seasonLincomb = inla.make.lincombs(sin12 = sin(2 *
  pi * StimeYear), cos12 = cos(2 * pi * StimeYear),
  sin6 = sin(2 * 2 * pi * StimeYear), cos6 = cos(2 *
    2 * pi * StimeYear))
names(seasonLincomb) = gsub("^lc", "season", names(seasonLincomb))
# predictions
StimePred = as.numeric(difftime(timePoints, timeOrigin,
  units = "days"))/365.35
predLincomb = inla.make.lincombs(timeRw2 = Diagonal(length(timePoints)),
  `(Intercept)` = rep(1, length(timePoints)), sin12 = sin(2 *
```

```
    pi * StimePred), cos12 = cos(2 * pi * StimePred),
  sin6 = sin(2 * 2 * pi * StimePred), cos6 = cos(2 *
    2 * pi * StimePred))
names(predLincomb) = gsub("^lc", "pred", names(predLincomb))

StimeIndex = seq(1, length(timePoints))
timeOriginIndex = which.min(abs(difftime(timePoints, timeOrigin)))

# disable some error checking in INLA
library("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(timeRw2, model = 'rw2',
    values = StimeIndex,
    prior='pc.prec', param = c(log(1.01)/26, 0.5)),
  data = co2s, family='gamma',  lincomb = c(derivLincomb, seasonLincomb, predLincomb),
  control.family = list(hyper=list(prec=list(prior='pc.prec', param=c(2, 0.5)))),
  # add this line if your computer has trouble
#  control.inla = list(strategy='gaussian', int.strategy='eb'),
  verbose=TRUE)

matplot(timePoints, exp(co2res$summary.random$timeRw2[,
  c("0.5quant", "0.025quant", "0.975quant")]), type = "l",
  col = "black", lty = c(1, 2, 2), log = "y", xaxt = "n",
  xlab = "time", ylab = "ppm")
xax = pretty(timePoints)
axis(1, xax, format(xax, "%Y"))
derivPred = co2res$summary.lincomb.derived[grep("time",
  rownames(co2res$summary.lincomb.derived)), c("0.5quant",
  "0.025quant", "0.975quant")]
scaleTo10Years = (10 * 365.25/as.numeric(diff(timePoints,
  units = "days")))
matplot(timePoints[-1], scaleTo10Years * derivPred,
  type = "l", col = "black", lty = c(1, 2, 2), ylim = c(0,
    0.1), xlim = range(as.numeric(co2s$date)),
  xaxs = "i", xaxt = "n", xlab = "time", ylab = "log ppm, change per 10yr")
axis(1, xax, format(xax, "%Y"))
abline(v = ISOdate(2008, 1, 1, tz = "UTC"), col = "blue")
matplot(StimeSeason, exp(co2res$summary.lincomb.derived[grep("season",
  rownames(co2res$summary.lincomb.derived)), c("0.5quant",
  "0.025quant", "0.975quant")]), type = "l", col = "black",
  lty = c(1, 2, 2), log = "y", xaxs = "i", xaxt = "n",
  xlab = "time", ylab = "relative ppm")
xaxSeason = seq(ISOdate(2009, 9, 1, tz = "UTC"), by = "2 months",
  len = 20)
axis(1, xaxSeason, format(xaxSeason, "%b"))
timePred = co2res$summary.lincomb.derived[grep("pred",
  rownames(co2res$summary.lincomb.derived)), c("0.5quant",
  "0.025quant", "0.975quant")]
matplot(timePoints, exp(timePred), type = "l", col = "black",
  lty = c(1, 2, 2), log = "y", xlim = ISOdate(c(2010,
```

```
    2025), 1, 1, tz = "UTC"), ylim = c(390, 435),
  xaxs = "i", xaxt = "n", xlab = "time", ylab = "ppm")
xaxPred = seq(ISOdate(2010, 1, 1, tz = "UTC"), by = "5 years",
  len = 20)
axis(1, xaxPred, format(xaxPred, "%Y"))
```



(a) random effect

(b) derivative

(c) seasonal

(d) predicted

Figure 4: INLA results

Heat

```
heatUrl = "http://pbrown.ca/teaching/appliedstats/data/sableIsland.rds"
heatFile = tempfile(basename(heatUrl))
download.file(heatUrl, heatFile)
x = readRDS(heatFile)
```

```
x$month = as.numeric(format(x$Date, "%m"))
xSub = x[x$month %in% 5:10 & !is.na(x$Max.Temp...C.),
  ]
weekValues = seq(min(xSub$Date), ISOdate(2030, 1, 1,
  0, 0, 0, tz = "UTC"), by = "7 days")
xSub$week = cut(xSub$Date, weekValues)
xSub$weekIid = xSub$week
xSub$day = as.numeric(difftime(xSub$Date, min(weekValues),
  units = "days"))
xSub$cos12 = cos(xSub$day * 2 * pi/365.25)
xSub$sin12 = sin(xSub$day * 2 * pi/365.25)
xSub$cos6 = cos(xSub$day * 2 * 2 * pi/365.25)
```

```
xSub$sin6 = sin(xSub$day * 2 * 2 * pi/365.25)
xSub$yearFac = factor(format(xSub$Date, "%Y"))

lmStart = lm(Max.Temp...C. ~ sin12 + cos12 + sin6 +
  cos6, data = xSub)
startingValues = c(lmStart$fitted.values, rep(lmStart$coef[1],
  nlevels(xSub$week)), rep(0, nlevels(xSub$weekIid) +
  nlevels(xSub$yearFac)), lmStart$coef[-1])

INLA::inla.doc('^t$')

library("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())

sableRes = INLA::inla(
  Max.Temp...C. ~ 0 + sin12 + cos12 + sin6 + cos6 +
  f(week, model='rw2',
    constr=FALSE,
    prior='pc.prec',
    param = c(0.1/(52*100), 0.05)) +
  f(weekIid, model='iid',
    prior='pc.prec',
    param = c(1, 0.5)) +
  f(yearFac, model='iid', prior='pc.prec',
    param = c(1, 0.5)),
  family='T',
  control.family = list(
    hyper = list(
      prec = list(prior='pc.prec', param=c(1, 0.5)),
      dof = list(prior='pc.dof', param=c(10, 0.5)))),
  control.mode = list(theta = c(-1,2,20,0,1),
             x = startingValues, restart=TRUE),
  control.compute=list(config = TRUE),
# control.inla = list(strategy='gaussian', int.strategy='eb'),
  data = xSub, verbose=TRUE)

sableRes$summary.hyper[, c(4, 3, 5)]

                                          0.5quant   0.025quant
precision for the student-t observations 3.211698e-01 3.141445e-01
degrees of freedom for student-t         1.379119e+01 1.142332e+01
Precision for week                       3.439837e+09 2.608100e+09
Precision for weekIid                    8.335287e-01 7.665303e-01
Precision for yearFac                    2.038099e+00 1.348922e+00
                                           0.975quant
precision for the student-t observations 3.312381e-01
degrees of freedom for student-t         1.673788e+01
Precision for week                       4.543424e+09
Precision for weekIid                    8.875598e-01
Precision for yearFac                    2.753691e+00

sableRes$summary.fixed[, c(4, 3, 5)]

        0.5quant 0.025quant  0.975quant
```

```
sin12 -4.6739279  -5.189943 -4.15843811
cos12  4.7607402   4.470423  5.05090575
sin6  -2.0499222  -2.273244 -1.82670780
cos6  -0.1529845  -0.324455  0.01837625
```

```
Pmisc::priorPost(sableRes)$summary[, c(1, 3, 5)]
```

```
                        mean   0.025quant   0.975quant
sd for t       1.763458e+00 1.737520e+00 1.784166e+00
sd for week    1.709116e-05 1.483571e-05 1.958113e-05
sd for weekIid 1.097193e+00 1.061454e+00 1.142182e+00
sd for yearFac 7.091899e-01 6.026184e-01 8.610067e-01
```

```
mySample = inla.posterior.sample(n = 24, result = sableRes,
  num.threads = 8, selection = list(week = seq(1,
    nrow(sableRes$summary.random$week))))
length(mySample)
names(mySample[[1]])
weekSample = do.call(cbind, lapply(mySample, function(xx) xx$latent))
dim(weekSample)
head(weekSample)
```

```
plot(x$Date, x$Max.Temp...C., col = mapmisc::col2html("black",
  0.3))
forAxis = ISOdate(2016:2020, 1, 1, tz = "UTC")
plot(x$Date, x$Max.Temp...C., xlim = range(forAxis),
  xlab = "time", ylab = "degrees C", col = "red",
  xaxt = "n")
points(xSub$Date, xSub$Max.Temp...C.)
axis(1, forAxis, format(forAxis, "%Y"))
matplot(weekValues[-1], sableRes$summary.random$week[,
  paste0(c(0.5, 0.025, 0.975), "quant")], type = "l",
  lty = c(1, 2, 2), xlab = "time", ylab = "degrees C",
  xaxt = "n", col = "black", xaxs = "i")
forXaxis2 = ISOdate(seq(1880, 2040, by = 20), 1, 1,
  tz = "UTC")
axis(1, forXaxis2, format(forXaxis2, "%Y"))
myCol = mapmisc::colourScale(NA, breaks = 1:8, style = "unique",
  col = "Set2", opacity = 0.3)$col
matplot(weekValues[-1], weekSample, type = "l", lty = 1,
  col = myCol, xlab = "time", ylab = "degrees C",
  xaxt = "n", xaxs = "i")
axis(1, forXaxis2, format(forXaxis2, "%Y"))
```