formula_algorithms

Haolin Zhong

2022/4/27

The Posterior Distribution of the Parameters Θ

- Assumption:

$$Y_i(t+6) = \beta_{0,i} + \beta_{1,i}Y_i(t) + \beta_{2,i}\Delta_{i,1}(t) + \beta_{3,i}\Delta_{i,2}(t) + \beta_{4,i}\Delta_{i,3}(t) + \epsilon_i(t)$$

$$f(\boldsymbol{B} \mid \boldsymbol{\beta}, \boldsymbol{\Sigma}) = \prod_{i=1}^{n} \left\{ (2\pi)^{-5/2}|\Sigma|^{-1/2} \exp\{-\frac{1}{2}(\boldsymbol{\beta}_i - \boldsymbol{\beta})'\Sigma^{-1}(\boldsymbol{\beta}_i - \boldsymbol{\beta})\} \right\}$$

$$P\left(\sigma^2\right) \propto \frac{1}{\sigma^2}; \quad P(\boldsymbol{\beta}) \propto 1; \quad P\left(\Sigma^{-1}\right) \propto |\Sigma|^{-(d+1)} \exp\left(-\frac{1}{2}\Sigma^{-1}\right)$$

- Let's denote:

$$\eta_i(t) = \beta_{0,i} + \beta_{1,i}Y_i(t) + \beta_{2,i}\Delta_{i,1}(t) + \beta_{3,i}\Delta_{i,2}(t) + \beta_{4,i}\Delta_{i,3}(t)$$

- Distribution of $\epsilon_i(t)$:

$$\epsilon_i(t) \sim N(0, \sigma^2)$$

- Distribution of $Y_i(t+6)$

$$Y_i(t+6) \sim N(\eta_i(t), \sigma^2)$$

- pdf for $Y_i(t+6)$:

$$f(Y_i(t+6) \mid \boldsymbol{\beta}_i, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\Sigma}) = -\frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{1}{2\sigma^2}[Y_i(t+6) - \eta_i(t)]^2\}$$

- pdf for Y: ($t_{i(n)}$ is the time of the last record for hurricane i)

$$f(Y \mid \boldsymbol{B}, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\Sigma}) = \prod_{i=1}^{n} \prod_{t=0}^{t_{i(n)}-6} -\frac{1}{\sqrt{2\pi}\sigma} \exp\{-\frac{1}{2\sigma^2}[Y_i(t+6) - \eta_i(t)]^2\}$$

- posterior distribution of the parameters θ:

$$P(\Theta \mid Y) \propto f(Y|\Theta)P(\Theta) = f(Y \mid \boldsymbol{B}, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\Sigma}) f(\boldsymbol{B} \mid \boldsymbol{\beta}, \boldsymbol{\Sigma}) P(\boldsymbol{\beta}) P(\sigma^2) P(\boldsymbol{\Sigma}^{-1})$$

- for efficient computation, we take logarithm:

$$\begin{aligned}\log P(\Theta \mid Y) \propto & \sum_{i=1}^{n} \sum_{t=0}^{t_{i(n)}-6} \{\log(-\sqrt{2\pi}\sigma) - \frac{1}{2\sigma^2}[Y_i(t+6) - \eta_i(t)]^2\} \\ & + \sum_{i=1}^{n} \{-\frac{1}{2}(\boldsymbol{\beta}_i - \boldsymbol{\beta})' \Sigma^{-1} (\boldsymbol{\beta}_i - \boldsymbol{\beta})\} \\ & - \log(\sigma^2) - (d+1)\log(|\boldsymbol{\Sigma}|) - \frac{1}{2}\text{trace}(\boldsymbol{\Sigma}^{-1})\end{aligned}$$

a MCMC algorithm to generate the posterior distribution of Θ

Data Preparation

```
library(tidyverse)
library(lubridate)
library(extraDistr)

correct.year = function(date) {
  date$year = date$year - 100
  return(date)
}

raw = read_csv("data/hurrican703.csv") %>%
  janitor::clean_names() %>%
  select(-nature, -season, -month) %>%
  mutate(
    time = gsub("[()]", "", time),
    time = as.POSIXlt(parse_datetime(time, "%y-%m-%d %H:%M:%S")),
    time = as.POSIXct(correct.year(time))
  ) %>%
  filter(hour(time) %in% c(0, 6, 12, 18),
         minute(time) == 0,
         second(time) == 0)

placeholder = data.frame(id = "placeholder",
```

```
                        time = parse_datetime("00-01-01 00:00:00", "%y-%m-%d %H:%M:%S"),
                        latitude = 0,
                        longitude = 0,
                        wind_kt = 0)

dt = bind_cols(rbind(raw, placeholder),
          rbind(placeholder, raw),
          .name_repair = "unique")

dt = dt[2:(nrow(dt)-1),]

dt = dt %>%
  filter(id...1 == id...6,
         time...7 + 6*60*60 == time...2) %>%
  mutate(
    d_lat = latitude...3 - latitude...8,
    d_log = longitude...4 - longitude...9,
    d_wkt = wind_kt...5 - wind_kt...10
  ) %>%
  select(id = id...1, wkt_new = wind_kt...5, wkt_cur = wind_kt...10, d_lat, d_log, d_wkt)

hc = distinct(dt, id) %>% add_rownames("i")

dt = dt %>% left_join(hc)

head(dt)
```

```
## # A tibble: 6 x 7
##   id        wkt_new wkt_cur d_lat  d_log d_wkt i
##   <chr>       <dbl>   <dbl> <dbl>  <dbl> <dbl> <chr>
## 1 ABLE.1950      40      35 0.600 -0.800     5 1
## 2 ABLE.1950      45      40 0.5   -1.10      5 1
## 3 ABLE.1950      50      45 0.800 -1.20      5 1
## 4 ABLE.1950      50      50 1     -1.40      0 1
## 5 ABLE.1950      50      50 0.700 -1.10      0 1
## 6 ABLE.1950      55      50 0.600 -1.10      5 1
```

MCMC

```
#function calculating beta in inverse gamma distribution
beta_gamma <- function(dat, B) {
  res = NULL
  for (j in 1:700) {
    subdat = dat %>% filter(i == j)
    y = subdat[, 2]
    x = cbind(rep(1, nrow(subdat)), subdat[, 3:6]) %>% as.matrix()
    beta = 0.5*(sum((y - x %*% t(B[j, ]))^2))
    res = rbind(res, beta)
  }
  return(sum(res))
}
```

```r
#test function
B <- data.frame(matrix(1, nrow = 700, ncol = 5))
#beta_gamma <- beta_gamma(dt, B)


sigmasq <- function(dat, B) {
  alpha = nrow(dat)
  beta = beta_gamma(dat, B)
  sigmasq = rinvgamma(1, alpha = alpha, beta = beta)
  return(sigmasq)
}

set.seed(2022)
sigmasq(dt, B)
```

```
## [1] 2.914953
```