MCMC

Data preprocessing (same as EDA)

```r
origin_df <- read.csv("hurrican703.csv")

hurricane_df <- origin_df %>%
  mutate(
    Month = factor(Month, levels = month.name[-c(2:3)]), # April-January (January ref, may choose anoth
    Nature = as.factor(Nature), # TS,ET,DS,SS,NR (DS ref, may choose another)
    # note: one hurricane can have multiple natures throughout its life
    time = gsub("[()]", "", time),
    time = paste0(ifelse(substr(time, 1, 2) > 23, "19", "20"), time),
    time = as.POSIXct(time, format = "%Y-%m-%d %H:%M:%S"),
    hour = substr(time, 12, 19)
  ) %>%
  # remove data not at six-hour time intervals. (613 observations)
  filter(hour %in% c("00:00:00", "06:00:00", "12:00:00", "18:00:00")) %>%
  dplyr::select(-hour)

# remove hurricanes that has only 2 (<3) observations (change the threshold if you wish)
few_id <- hurricane_df %>%
  group_by(ID) %>%
  summarize(obs = n()) %>%
  filter(obs < 3) %>%
  .$ID
hurricane_df <- hurricane_df %>% filter(!(ID %in% few_id)) # remove 2 hurricanes

# manually correct those data
hurricane_df <-
  hurricane_df %>%
  mutate(
    # 2 hurricanes with the name ALICE.1954
    ID = ifelse(ID == "ALICE.1954" & Month == "June", "ALICE.1954(1)", ID),
    ID = ifelse(ID == "ALICE.1954", "ALICE.1954(2)", ID),
    # 4 hurricanes with the name SUBTROP:UNNAMED.1974
    ID = ifelse(ID == "SUBTROP:UNNAMED.1974" & Month == "June", "SUBTROP:UNNAMED.1974(1)", ID),
    ID = ifelse(ID == "SUBTROP:UNNAMED.1974" & Month == "July", "SUBTROP:UNNAMED.1974(2)", ID),
    ID = ifelse(ID == "SUBTROP:UNNAMED.1974" & Month == "August", "SUBTROP:UNNAMED.1974(3)", ID),
    ID = ifelse(ID == "SUBTROP:UNNAMED.1974", "SUBTROP:UNNAMED.1974(4)", ID),
    # 2 hurricanes with the name SUBTROP:UNNAMED.1976
    ID = ifelse(ID == "SUBTROP:UNNAMED.1976" & Month == "May", "SUBTROP:UNNAMED.1976(1)", ID),
    ID = ifelse(ID == "SUBTROP:UNNAMED.1976", "SUBTROP:UNNAMED.1976(2)", ID)
  )

diff_df <-
  hurricane_df %>%
  group_by(ID) %>%
  mutate(
    lat_diff = lead(Latitude) - Latitude,
    long_diff = lead(Longitude) - Longitude,
    wind_diff = lead(Wind.kt) - Wind.kt,
    time_j = round(difftime(time, min(time), units = "hours")/6) %>% as.integer
  ) %>%
  dplyr::select(ID, lat_diff, long_diff, wind_diff, time_j)
```

Data partition (may be useless.)

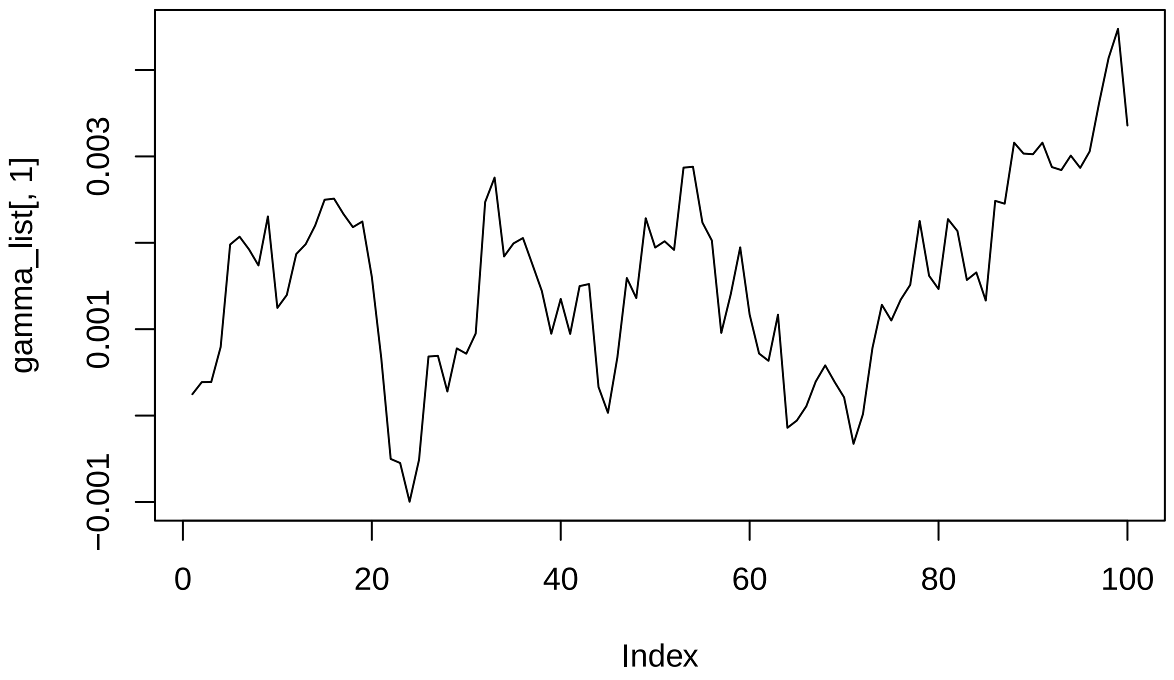
```r
# data partition
id <- hurricane_df %>% dplyr::select(ID) %>% unique %>% as.vector %>% unlist
set.seed(1)
index <- sample(1:length(id), size = 0.8*length(id))
training_id <- id[index]
test_id <- id[-index]
Training <- hurricane_df %>% filter(ID %in% training_id)
Test <- hurricane_df %>% filter(ID %in% test_id)
```

Create X, Y, Z and $\boldsymbol{m} = (m_i)$ in R

```r
n <- length(unique(Training$ID))

Y <- split(Training$Wind.kt, Training$ID) %>%
  lapply(function(x) x[-c(1:2)])

X <- Training %>%
  group_by(ID) %>%
  slice(1) %>%
  dplyr::select(ID, Season, Month, Nature) %>%
  ungroup(ID)
X <- model.matrix(~., X[-1])[,-1]

Z <- Training %>%
  group_by(ID) %>%
  mutate(
    intercept = 1,
    wind_pre = lag(Wind.kt),
    lat_diff = lag(Latitude) - lag(Latitude, 2),
    long_diff = lag(Longitude) - lag(Longitude, 2),
    wind_diff = lag(Wind.kt) - lag(Wind.kt, 2),
  ) %>%
  drop_na %>%
  dplyr::select(ID, intercept, wind_pre, lat_diff, long_diff, wind_diff)
Z <- split(Z[, names(Z)[-1]], Z$ID) %>%
  lapply(as.matrix)

m <- Training %>%
  group_by(ID) %>%
  summarize(obs = n() - 2) %>%
  .$obs
```

MCMC

```r
# constants
nu <- 5 # suggested
V <- diag(5) # try other
S <- diag(5) # try other

# initial values of variables (please try other)
B0 <- matrix(rep(0, 5*n), ncol = n, nrow = 5) # each column is beta_i
mu0 <- rep(0, 5)
Sigma0 <- diag(5)
gamma0 <- rep(0, 14)
sigma0 <- 10



B_sample <- function(mu, Sigma, gamma, sigma) {
  Sigma.inv <- solve(Sigma)
  B_mean_cov <- function(i) {
    cov <- solve(Sigma.inv + 1/sigma^2 * t(Z[[i]]) %*% Z[[i]])
    mean <- cov %*% (Sigma.inv %*% mu + 1/sigma^2 * colSums((Y[[i]] - (X[i,] %*% gamma)[,]) * Z[[i]]))
    list(mean = mean, cov = cov)
  }
  mean_cov_list <- lapply(1:n, B_mean_cov)
  B <- sapply(mean_cov_list, function(x) {mvrnorm(mu = x$mean, Sigma = x$cov)})
  return(B)
}
# helper function for sampling B


mu_sample <- function(B, Sigma) {
  cov <- V %*% solve(n*V + Sigma)
  mean <- cov %*% rowSums(B)
  mu <- mvrnorm(mu = mean, Sigma = cov)
  return(mu)
}

Sigma_sample <- function(B, mu) {
  Sigma.inv = rWishart(n = 1, Sigma = S + (B - mu) %*% t(B - mu), df = n + nu)[,,]
  Sigma = solve(Sigma.inv)
  return(Sigma)
}

gamma_sample <- function(B, sigma){
  X_trans <- sqrt(m) * X
  cov <- solve(400*diag(14) + 1/sigma^2 * t(X_trans) %*% X_trans)
  total <- rowSums(sapply(1:n, function(i) sum(Z[[i]] %*% B[,i] - Y[[i]]) * X[i,]))
  mean <- cov %*% (-1/sigma^2 * total)
  gamma <- mvrnorm(mu = mean, Sigma = cov)
  return(gamma)
}

# MH algorithm, not finished
sigma_sample <- function(B, gamma) {
  ####
  return(sigma)
}

MCMC <- function(B0, mu0, Sigma0, gamma0, sigma0, iter = 100) {
  B <- B0
  mu <- mu0
  Sigma <- Sigma0
  gamma <- gamma0
  sigma <- sigma0
  res <- vector("list", iter)
  for (i in 1:iter) {
    B <- B_sample(mu, Sigma, gamma, sigma)
    mu <- mu_sample(B, Sigma)
    Sigma <- Sigma_sample(B, mu)
    gamma <- gamma_sample(B, sigma)
    # sigma <- sigma_sample(B, gamma)
    res[[i]] <- list(B = B, mu = mu, Sigma = Sigma, gamma = gamma, sigma = sigma)
  }
  return(res)
}

set.seed(1)
res <- MCMC(B0, mu0, Sigma0, gamma0, sigma0, iter = 100) # try 100 iterations
B_list <- t(mapply(function(x) x$B, res))
mu_list <- t(mapply(function(x) x$mu, res))
colnames(mu_list) <- colnames(Z[[1]])
Sigma_list <- t(mapply(function(x) x$Sigma, res))
gamma_list <- t(mapply(function(x) x$gamma, res))
sigma_list <- t(mapply(function(x) x$sigma, res))
plot(gamma_list[,1], type = "l") # try plotting gamma_year
```


gamma_list[, 1]
−0.001
0.001
0.003
0
20
40
60
80
100
Index