Project 5

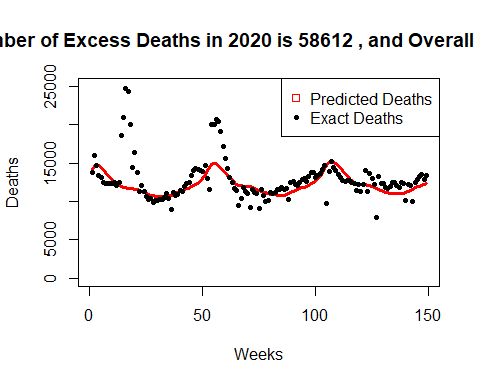
Robin

2022-12-02

## The author of this project is Sihong Lin, which can also be referred to as   
## Robin Lin.  
  
## The student number of the author is s2435943.  
  
## The author declares that everything in the project is his own work, and he  
## promises that he did not plagiarise from others.  
  
## The url of the GitHub repository is listed as follows.  
## https://github.com/RobinLam2435943/Statistical-Programming-Individual-Projects.git  
  
## -----------------------------------------------------------------------------  
  
## "Excess deaths" are the number of deaths over some period. The codes below   
## aims at predicting the expected number of deaths per week for England and Wales   
## from 2020. One way to do it is to compute the expected number of deaths in  
## each week, but the problem is that it will underestimate the number of deaths  
## if the population is ageing or growing. In view of this, there is an adjustment   
## to allow for seasonal variation in mortality rates. The excess deaths are to  
## be computed by comparing the predicted number of deaths with the actual number   
## of deaths, and they are to be modelled by using a Bayesian model in JAGS.

# pdf('excess.pdf', height = 10, width = 6)  
# par(mfrow = c(2, 1))

predict\_death <- function(Nf, Nm, mf, mm, d){  
   
 ## This function takes vectors of starting populations by one year age class(N  
 ## ), instantaneous per capita death rates per year(m), and number of deaths   
 ## in the j-th week(d) as inputs. Notice that the starting populations and the   
 ## death rates are separated into 2 groups, which are male and female, while   
 ## the number of deaths is not sex specific. It iterates the model forward for  
 ## length(d) weeks, and returns the predicted number of deaths each week as a   
 ## vector.   
   
 # Makes copies of the starting populations of the male and female groups.   
 Nf\_copy <- Nf  
 Nm\_copy <- Nm  
   
 # Expected proportions of age groups dying in a week.  
 qf <- 1 - exp(-mf / 52)  
 qm <- 1 - exp(-mm / 52)  
   
 # Initialises the vectors specifying deaths per week for male and female groups.  
 Death\_week\_f <- rep(0, length(d))  
 Death\_week\_m <- rep(0, length(d))  
   
 for(j in 1 : length(d)){# Loops through all of the elements of d.  
   
 # Obtains the predicted deaths of different age classes for male and female   
 # groups.  
 Df <- .9885 \* d[j] \* (qf \* Nf)  
 Dm <- .9885 \* d[j] \* (qm \* Nm)  
   
 # Obtains the total predicted deaths for male and female groups.  
 pred\_deaths\_f <- sum(Df)  
 pred\_deaths\_m <- sum(Dm)  
   
 # Stores the values above into the vectors that specify deaths per week.  
 Death\_week\_f[j] <- pred\_deaths\_f  
 Death\_week\_m[j] <- pred\_deaths\_m  
   
 # Removes deaths from the starting populations.  
 N\_star\_f <- Nf - Df  
 N\_star\_m <- Nm - Dm  
   
 # Obtains the previous N\_stars, where the first element is the starting   
 # population of age class of 1 year.  
 N\_star\_previous\_f <- c(Nf\_copy[1], N\_star\_f[1 : (length(Nf) - 1)])  
 N\_star\_previous\_m <- c(Nm\_copy[1], N\_star\_m[1 : (length(Nm) - 1)])  
   
 # Obtains the populations of the same age class at the start of the next week.  
 N\_plus\_f <- N\_star\_f \* 51 / 52 + N\_star\_previous\_f / 52  
 N\_plus\_m <- N\_star\_m \* 51 / 52 + N\_star\_previous\_m / 52  
   
 # Updates the starting populations.  
 Nf <- N\_plus\_f  
 Nm <- N\_plus\_m  
 }  
   
 # Returns the summation of the vectors specifying deaths per week of male and   
 # female groups.  
 return(Death\_week\_f + Death\_week\_m)  
}  
# Reads the two data-sets.  
it1720uk <- read.table('lt1720uk.dat', header = T)  
death1722uk <- read.table('death1722uk.dat', header = T)  
  
# Specifies the inputs of the deaths prediction function.   
fpop20 <- as.numeric(it1720uk$fpop20) # Starting population of the female group.  
mf <- as.numeric(it1720uk$mf) # Mortality rates of the female group.  
mpop20 <- as.numeric(it1720uk$mpop20) # Starting population of the male group.  
mm <- as.numeric(it1720uk$mm) # Mortality rates of the male group.  
  
# The overall mortality rate modifiers.  
death\_num\_overall <- as.numeric(death1722uk$d)[157 : length(death1722uk$d)]  
  
# The overall predicted deaths numbers.  
predicted\_deaths\_overall <- predict\_death(fpop20, mpop20, mf, mm, death\_num\_overall)  
  
# The overall real deaths numbers.  
real\_deaths\_overall <- death1722uk$deaths[157 : length(death1722uk$d)]  
  
# The overall excess deaths numbers.  
excess\_deaths\_overall <- real\_deaths\_overall - predicted\_deaths\_overall  
# The mortality rate modifiers in 2020.  
death\_num\_2020 <- as.numeric(death1722uk$d)[(157 : 208)]  
  
# The predicted deaths numbers in 2020.  
predicted\_deaths\_2020 <- predict\_death(fpop20, mpop20, mf, mm, death\_num\_2020)  
  
# The real deaths numbers in 2020.  
real\_deaths\_2020 <- death1722uk$deaths[(157 : 208)]  
  
# The excess deaths numbers in 2020.  
excess\_deaths\_2020 <- real\_deaths\_2020 - predicted\_deaths\_2020  
weeks <- 1 : length(excess\_deaths\_overall) # Specifies overall weeks.  
  
# Overall plot of predicted deaths.  
plot(predicted\_deaths\_overall ~ weeks, type = 'l', xlab = 'Weeks', ylab = 'Deaths', main = paste('Number of Excess Deaths in 2020 is', round(sum(excess\_deaths\_2020), 0), ', and Overall is', round(sum(excess\_deaths\_overall), 0)), col = 'red', lwd = 3, ylim = c(0, 2.5e4))   
  
# Overall plot of real deaths.  
points(real\_deaths\_overall ~ weeks, pch = 20)  
  
# Shows the legends.  
legend('topright', legend = c('Predicted Deaths', 'Exact Deaths'), col = c('red', 'black'), pch = c(0, 20))



# Overall cumulative excess deaths.  
cum\_excess\_deaths\_overall <- cumsum(excess\_deaths\_overall)  
  
# Plot of overall cumulative excess deaths.  
plot(cum\_excess\_deaths\_overall ~ weeks, type = 'p', xlab = 'Weeks', ylab = 'Cumulative Excess Deaths', main = 'Overall Cumulative Excess Deaths', col = 'black', lwd = 3)  
  
library(rjags) # Loads rjags.

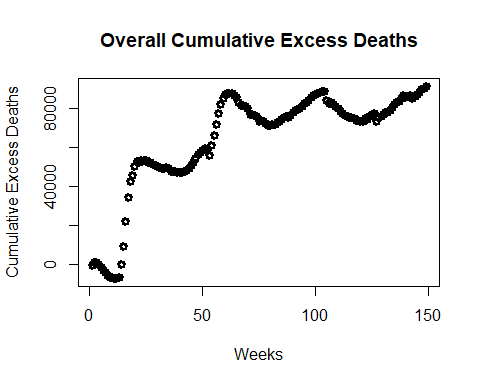
## Warning: package 'rjags' was built under R version 4.1.3

## Loading required package: coda

## Warning: package 'coda' was built under R version 4.1.3

## Linked to JAGS 4.3.0

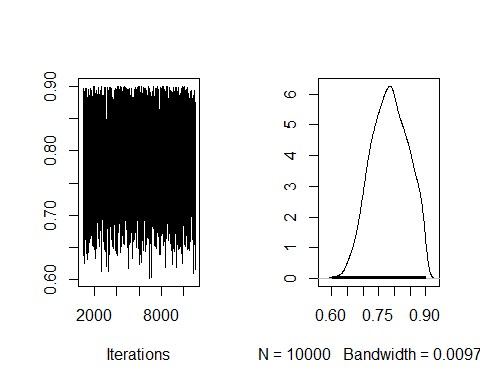
## Loaded modules: basemod,bugs



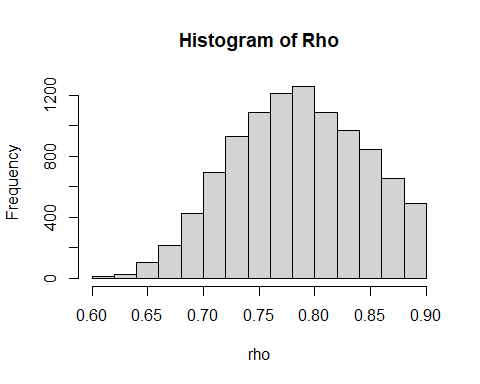
## Warning: package 'rjags' was built under R version 4.1.3  
## Loading required package: coda  
## Warning: package 'coda' was built under R version 4.1.3  
## Linked to JAGS 4.3.0  
## Loaded modules: basemod,bugs  
  
# In the weeks near Christmas and the New Year, the data have recording problems.  
# Hence, they should be set as NA.  
excess\_deaths\_overall\_copy <- excess\_deaths\_overall # Makes a copy.  
excess\_deaths\_overall\_copy[c(51, 52, 53, 105, 106)] <- NA # Sets values to be NA.  
  
# The JAGS model.  
mod <- jags.model('model.jags', data = list(x = excess\_deaths\_overall\_copy, N = length(excess\_deaths\_overall)))

## Compiling model graph  
## Resolving undeclared variables  
## Allocating nodes  
## Graph information:  
## Observed stochastic nodes: 144  
## Unobserved stochastic nodes: 9  
## Total graph size: 607  
##   
## Initializing model

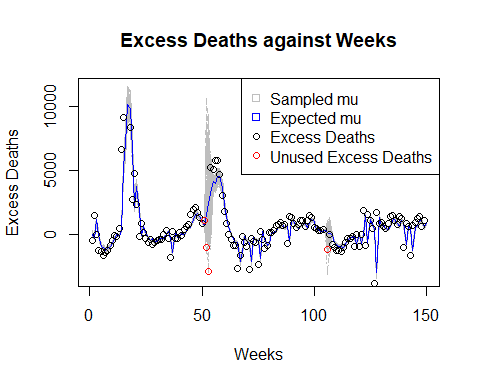
## Compiling model graph  
## Resolving undeclared variables  
## Allocating nodes  
## Graph information:  
## Observed stochastic nodes: 144  
## Unobserved stochastic nodes: 9  
## Total graph size: 607  
##   
## Initializing model  
  
# Draws 10000 samples from the posterior densities of mu, rho, and k.  
samples <- coda.samples(mod, c('mu', 'rho', 'k'), n.iter = 1e4)  
  
# MCMC of the posterior of rho.  
rho <- samples[, (3 + length(excess\_deaths\_overall))]  
  
# Trace plot and histogram of rho.  
plot(rho)



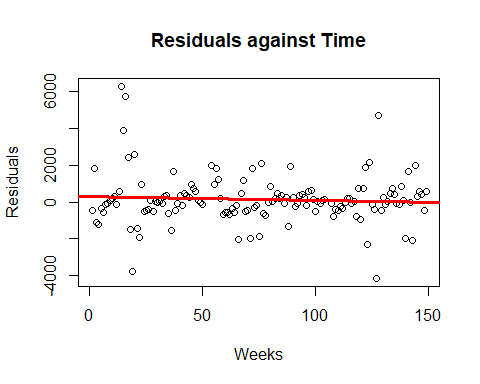
hist(as.numeric(rho[[1]]), xlab = 'rho', main = 'Histogram of Rho')



# Posterior expected value vector for mu.  
sample\_matrix <- as.matrix(samples[[1]], nrow = 1e4) # Matrix of all parameter   
# values.  
mu\_mean <- apply(sample\_matrix[, (2 : (2 + length(excess\_deaths\_overall)))], 2, mean)  
# Vector of the mean of mu.  
  
# Extracts every 50th sampled mu vector.  
mu\_50 <- sample\_matrix[seq(50, 1e4, by = 50), (2 : (2 + length(excess\_deaths\_overall)))]  
  
# Plots the excess deaths against weeks.  
matplot((1 : dim(mu\_50)[2]), t(mu\_50), type = "l", col = 'grey', xlab = 'Weeks', ylab = 'Excess Deaths', main = 'Excess Deaths against Weeks')  
# Plots every 50th sampled mu vector.  
matlines((1 : dim(mu\_50)[2]), mu\_mean, type = "l", col = 'blue')  
# Plots the overall excess deaths.  
matpoints((1 : length(excess\_deaths\_overall)), excess\_deaths\_overall, pch = 1, col = 'black')  
# Plots the unused excess deaths.  
matpoints(c(51, 52, 53, 105, 106), excess\_deaths\_overall[c(51, 52, 53, 105, 106)], pch = 1, col = 'red')  
# Shows the legends.  
legend('topright', legend = c('Sampled mu', 'Expected mu', 'Excess Deaths', 'Unused Excess Deaths'), col = c('grey', 'blue', 'black', 'red'), pch = c(0, 0, 1, 1))



# Residuals against time plot.  
residuals <- excess\_deaths\_overall\_copy - mu\_mean[1 : (length(mu\_mean) - 1)]  
plot(residuals ~ weeks, xlab = 'Weeks', ylab = 'Residuals', main = 'Residuals against Time')  
abline(lm(residuals ~ weeks), col = 'red', lwd = 3) # Adds a trend line.



# dev.off()