STA 642 Homework3

Jae Hyun Lee, jl914

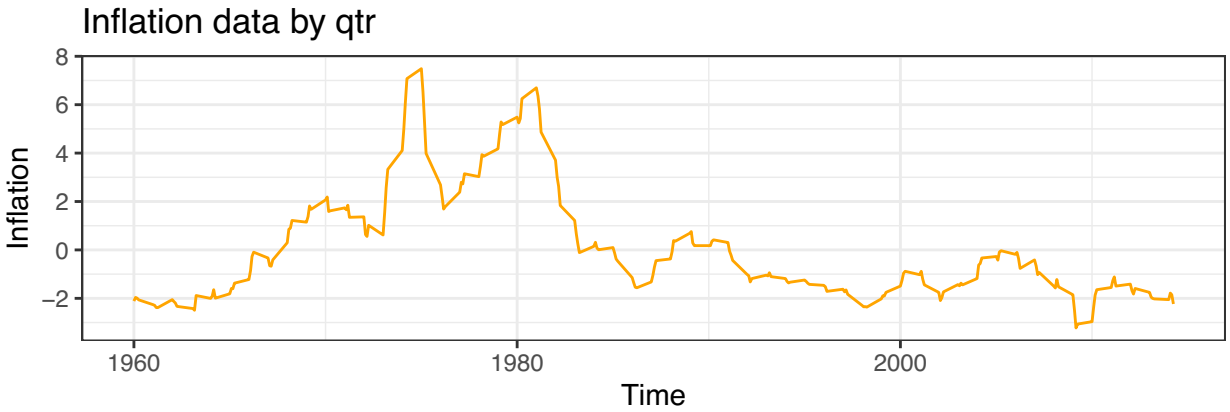
03 February, 2020

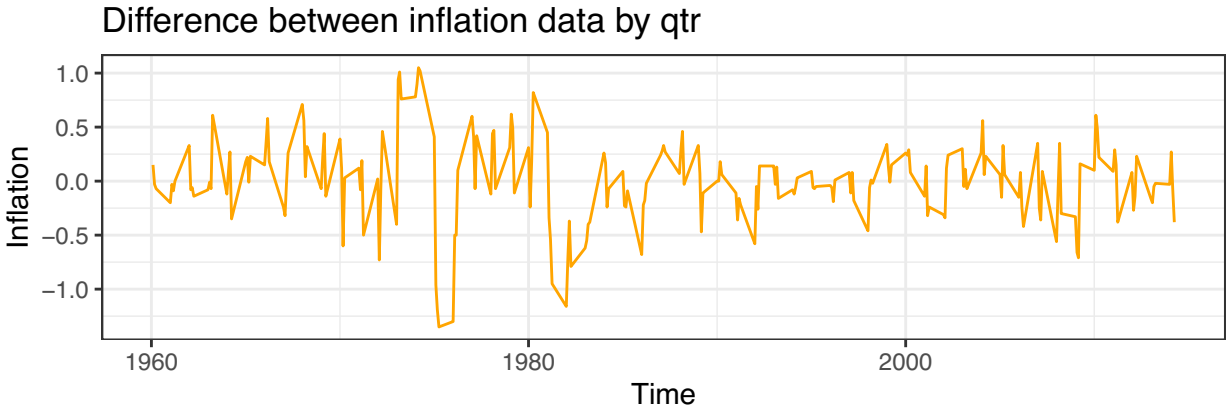
HW3 for STA-642

Exercise 3

Data EDA

```
#load data
data <- readxl::read_xlsx("USmacrodata1960-2014.xlsx")
#combine qtr and yr
time <- as.POSIXct(paste(data$yr,data$qtr,"01" ,sep = "-"))
#Y - data
inff <- data$Infln -mean(data$Infln)
inff_diff <- diff(inff)
#plot of original process
plot1 <- ggplot(mapping = aes(x = time, y = inff))+
  geom_line(col = "orange") +
  labs(title = "Inflation data by qtr", x = "Time", y = "Inflation") +
  theme_bw()

plot2 <- ggplot(mapping = aes(x = time[-1], y = inff_diff)) +
  geom_line(col = "orange") +
  labs(title = "Difference between inflation data by qtr", x = "Time", y = "Inflation") +
  theme_bw()
grid.arrange(plot1,plot2)
```





Function defining

```
AR_p <- function(data,p){
  #zero mean
  data <- data - mean(data)
  #y_n ~ y_
  y <- data[(p+1):length(data)]
  #Hankel matrix
  design = matrix(rep(NA,(length(data)-p)*p),ncol = p)
  for(i in 1:(length(data)-p)){
    design[i,] = rev(data[i:(i+p-1)])
  }
  #H'H
  B <- t(design) %*% design
  #B^{-1}H'y
  b <- solve(B) %*% t(design) %*% y
  #SSR
  Qb <- t(y-design %*% b) %*% (y - design %*% b)
  #residual
  res <- (y - design %*% b)
  return(list(B=B,b=b,Qb=Qb,design=design,res = res))
}
```

simulation over posterior distribution phi, v and future value

```
AR_p_sim <- function(data,p,nsim,npred){
  #y_t ~ y_(t-p+1)
  data <- data - mean(data)
  y = data[(p+1):length(data)]

  #statistics from design matrix
  B = AR_p(data,p)$B
  b = AR_p(data,p)$b
  Qb = AR_p(data,p)$Qb


  #simulation for posterior dist of v, phi, future y
  vsim <- rgamma(nsim,(length(data)-2*p)/2, Qb/2)^(-1)
  pisim <- matrix(data = rep(b,nsim), ncol = nsim) +
    t(chol(x = solve(B))) %*% t(MASS::mvrnorm(nsim, mu = rep(0,p), Sigma = diag(x=1, nrow = p))) *
    matrix(data = t(rep(sqrt(vsim),p)), nrow = p)

  #predict n+1 using n-p+1~n and replace n-p+1 with n+1
  y_pred <- t(MASS::mvrnorm(n = nsim, mu = rep(0,npred), Sigma = diag(x = 1, nrow = npred)))

  X <- matrix(data = rep(data[length(data):(length(data)-p+1)],nsim), ncol = nsim)
  for(t in 1:npred){
    y_pred[t,] <- colSums(X * pisim) + sqrt(vsim) * y_pred[t,]
    X <- rbind(y_pred[t,],X)
    X <- X[-nrow(X),]
  }
  return(list(phi = t(pisim), nu = vsim, pred = t(y_pred)))
}
```

extract simulated eigen values of evolution matrix

```
post_eigen <- function(phi){
  #phi is simulated phi
  #data allocation
  modulus <- matrix(rep(NA,dim(phi)[1]*dim(phi)[2]),ncol = dim(phi)[2])
  angle <- matrix(rep(NA,dim(phi)[1]*dim(phi)[2]),ncol = dim(phi)[2])
  period <- matrix(rep(NA,dim(phi)[1]*dim(phi)[2]),ncol = dim(phi)[2])
  #extract modulus and angle from complex eigen values
  for(i in 1:nrow(phi)){
    G <- rbind(matrix(phi[i,],nrow = 1),diag(x = length(phi[i,])))[1:length(phi[i,]),]
    lambda <- eigen(G)$values
    modulus[i,] <- Mod(lambda)
    angle[i,] <- Arg(lambda);
    angle[i,angle[i,] == 0] <- pi
    period[i,] <- 2*pi/angle[i,]
  }
  return(list(modulus = modulus, period = period, angle = angle))
}
```

decompsing process by AR(p) model

```
decomp <- function(data, p){
  #zero mean
  data <- data - mean(data)

  #point estimate of phi
  b = AR_p(data,p)$b

  #evolution matrix and its eigenvalues
  G <- rbind(matrix(b,nrow = 1),diag(x = length(b)))[1:length(b),]
  lambda <- eigen(G)$values
  lambda_vec <- eigen(G)$vectors
  modulus <- Mod(lambda)
  angle <- Arg(lambda); angle[angle == 0] <- pi
  #E'FE^{-1}
  H <- diag(lambda_vec[1,]) %*% solve(lambda_vec)

  #sort by angle
  #G_sorted <- sort(angle)
  #G_sortidx <- match(G_sorted,angle)
  #modulus <- modulus[G_sortidx]
  #lambda_vec <- lambda_vec[G_sortidx,]
  #H <- H[G_sortidx,]

  n_complex <- sum(angle<0); i = angle>0
  n_total <- sum(i); modulus <- modulus[i]; angle <- angle[i]; H <- H[i,]
  H[1:n_complex, ] <- 2*Re(H[1:n_complex,])
  decomp <- H %*% cbind(data[(length(data) - p +1):length(data)],
                        t(AR_p(data,p)$design),
                        matrix(data = 0, nrow = p, ncol = p-1))
  #decomp <- decomp[, ncol(decomp):1]
  return(decomp)
}
```

plotting decomposed process

```
decomp_plot <- function(data, decomp, k){
  total <- length(data)
  nk <- min(k, nrow(decomp))

  mi <- min(data)
  ma <- max(data)
  d <- 1/(ma - mi)
  h <- 0.02 * total
  cen <- mean(d*(data-mi))

  plot_mat <- matrix(data = NA, nrow = total, ncol = nk+1)
  plot_mat[,1] <- d * (data - mi) - cen
  for(i in 1:nk){
    plot_mat[, i+1] = -cen +d *(Re(decomp[i,]) - mi)
  }
  plot_df <- as.data.frame(plot_mat)
  colnames(plot_df) <- c("Data", paste("Comp",1:nk, sep = ""))
```

```
  plot_list <- list()

  for(i in colnames(plot_df)){
    plot_list[[i]] <- ggplot(data = plot_df, mapping = aes_string(x = 1:length(data), y = i)) +
      geom_line()
  }
  grid.arrange(grobs = plot_list, nrow = ncol(plot_df))
}
```

(a)

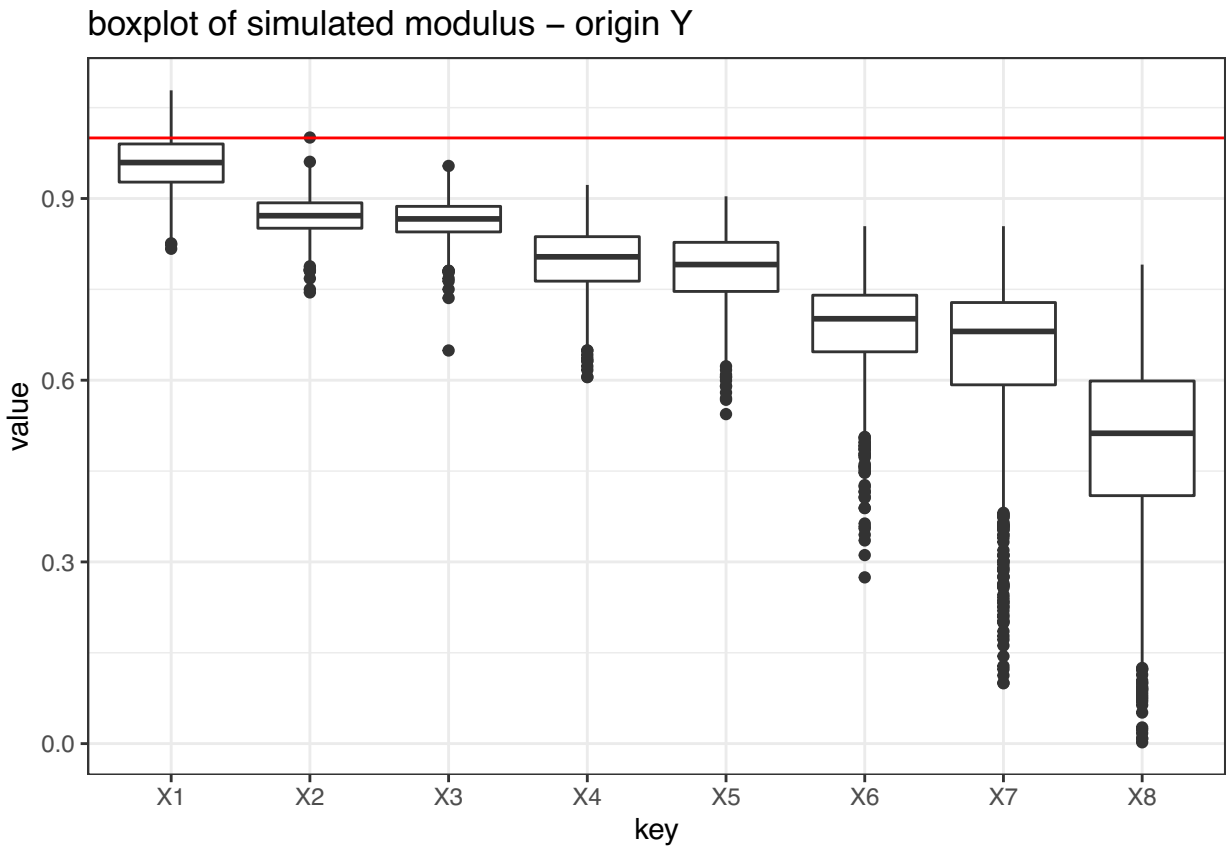
If the AR(8) is accepted as a good model for this data, do you think the data-model match supports stationarity? Give full numerical support for this based on the reference posterior.

```
p = 8
nsim = 1000
npred = 12
set.seed(1)
inff_result <- AR_p_sim(p = p, inff, nsim, npred)
diff_result <- AR_p_sim(p = p, inff_diff, nsim, npred)

inff_sim_eigen <- post_eigen(inff_result$phi)
diff_sim_eigen <- post_eigen(diff_result$phi)

data.frame(sum(inff_sim_eigen$modulus >1)/nsim,sum(diff_sim_eigen$modulus >1)/nsim) %>%
  `colnames<-`(c("origin_Y","diff_Y")) %>%
  kable(caption = "pr(Non-stationary | p = 8")
```

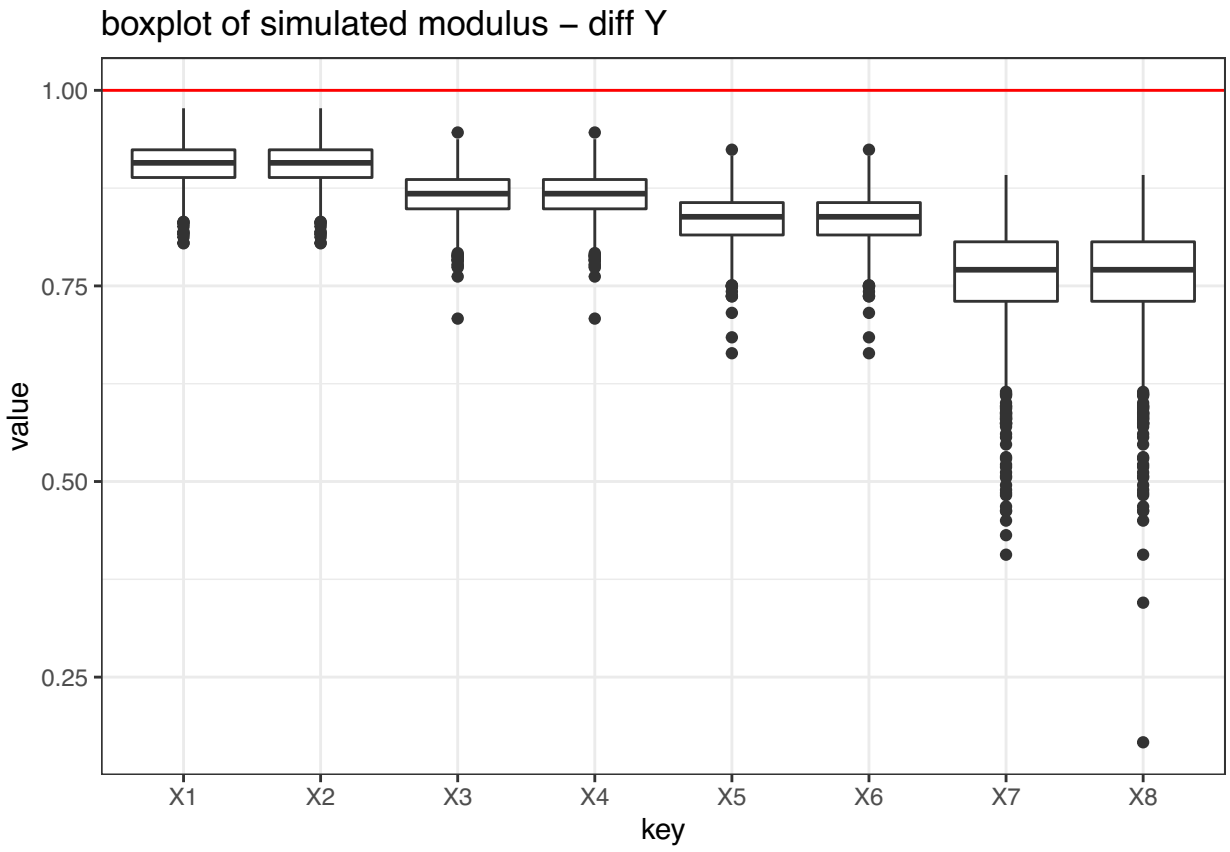
Table 1: pr(Non-stationary | p = 8

origin_Y	diff_Y
0.183	0

```
modulus_long <- gather(data.frame(inff_sim_eigen$modulus), key = "key", value = "value")
ggplot(data = modulus_long, mapping = aes(x = key, y = value)) +
  geom_boxplot() +
  geom_hline(yintercept = 1, col = "red") +
  theme_bw() +
  labs(title = "boxplot of simulated modulus - origin Y")
```




```
modulus_long <- gather(data.frame(diff_sim_eigen$modulus), key = "key", value = "value")
ggplot(data = modulus_long, mapping = aes(x = key, y = value)) +
  geom_boxplot() +
  geom_hline(yintercept = 1, col = "red") +
  theme_bw() +
  labs(title = "boxplot of simulated modulus - diff Y")
```



*** Original Inflation data

We can find that there are some simulated modulus of ϕ_1 that are larger than 1. The Monte-Carlo estimate of probability of non-stationarity in this process is 0.183. Thus we can conclude that original inflation rate is non-stationary process

*** Differenced Inflation data

We could not find any simulated modulus of ϕ that are larger than 1. Therefore differenced inflation rate can be considered stationary process.

(b)

Assuming that there is some indication of quasi-periodic behaviour under this posterior, summarise inferences on the maximum wavelength (a.k.a. period) of (quasi-)periodic components.

*** Original Inflation data

```
inff_period <- inff_sim_eigen$period
inff_modulus <- inff_sim_eigen$modulus
num_period <- apply(inff_period,1,function(x){sum(x!=2)})/2
kable(summary(as.factor(num_period)),caption = "The number of periodic components for 1000 simulations"
```

Table 2: The number of periodic components for 1000 simulations

	count
2	63
3	915
4	22

For original inflation data, we can find that the number of periodic components vary accroding to each simulation. Thus I decide to provide probability that each components are periodic.

```
proportion <- data.frame(matrix(apply(inff_period,2,
                                      function(x){sum(x == 2)/nrow(inff_period)}),ncol = 8)) %>%
  `colnames<-`(paste("comp",1:8))
kable(proportion, caption = "The probability that components are real")
```

Table 3: The probability that components are real

comp 1	comp 2	comp 3	comp 4	comp 5	comp 6	comp 7	comp 8
0.861	0.004	0.039	0.033	0.022	0.099	0.119	0.905

We can find that in most case, component1 and component8 has real eigenvalue. Let exclude components which are not periodic and make inference on cases which has 3 periodic components.

```
#extract periodic components
inff_period <- inff_period[num_period == 3,]
inff_period_only <- data.frame(t(apply(inff_period, 1, function(x){unique(abs(x[x!=2]))}))) %>%
  `colnames<-`(paste("comp",1:3))
inff_modulus <- inff_modulus[num_period == 3,]
inff_modulus_only <- data.frame(t(apply(inff_modulus, 1, function(x){x[duplicated(x)]}))) %>%
  `colnames<-`(paste("comp",1:3))

kable(summary(inff_period_only),caption = "summary of period")
```

Table 4: summary of period

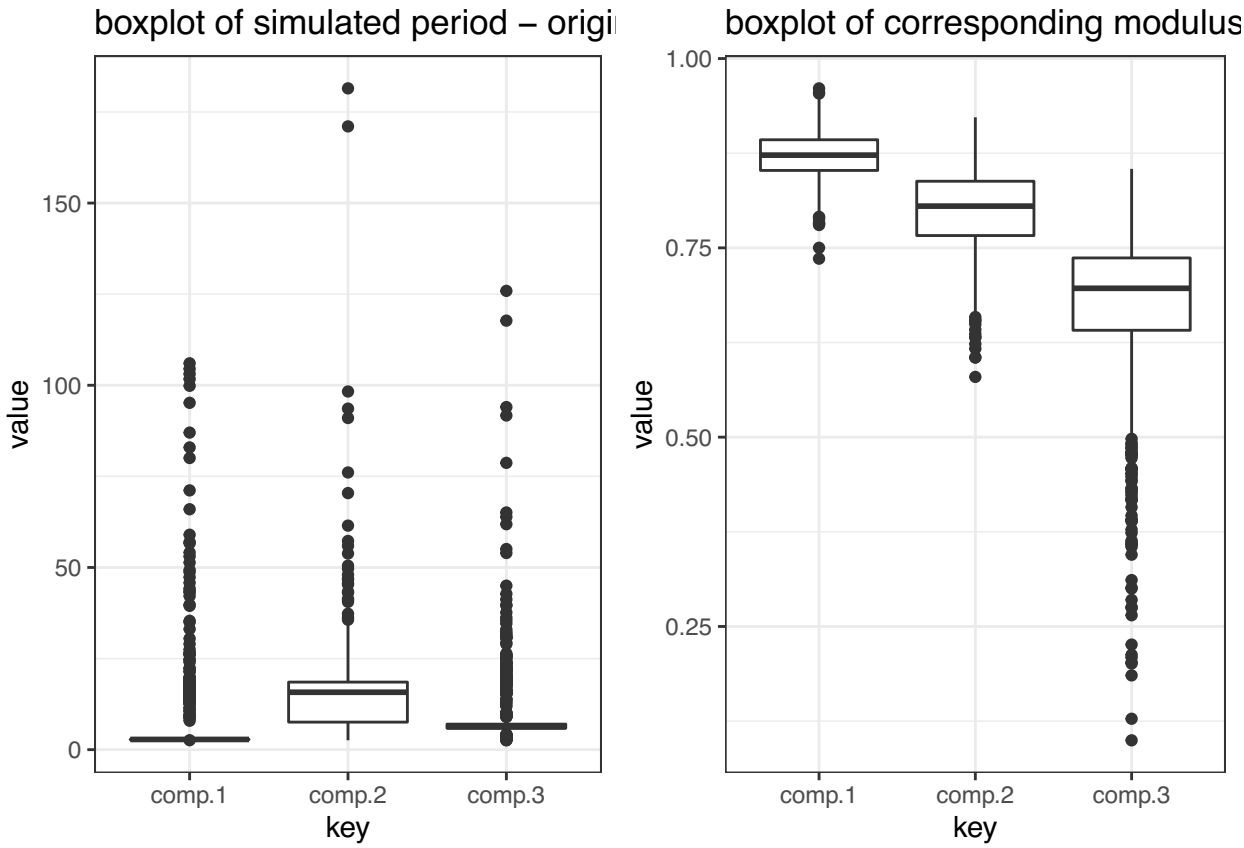
comp 1	comp 2	comp 3
Min. : 2.593	Min. : 2.587	Min. : 2.558
1st Qu.: 2.761	1st Qu.: 7.562	1st Qu.: 5.888
Median : 2.801	Median : 15.755	Median : 6.263
Mean : 6.418	Mean : 15.408	Mean : 9.046
3rd Qu.: 2.848	3rd Qu.: 18.541	3rd Qu.: 7.003
Max. :106.024	Max. :181.474	Max. :125.901

```
kable(summary(inff_modulus_only),caption = "summary of corresponding mod")
```

Table 5: summary of corresponding mod

comp 1	comp 2	comp 3
Min. :0.7357	Min. :0.5796	Min. :0.09961
1st Qu.:0.8522	1st Qu.:0.7663	1st Qu.:0.64115
Median :0.8725	Median :0.8051	Median :0.69659
Mean :0.8710	Mean :0.7986	Mean :0.67481
3rd Qu.:0.8927	3rd Qu.:0.8380	3rd Qu.:0.73669
Max. :0.9606	Max. :0.9224	Max. :0.85430

```
period_long <- gather(data.frame(inff_period_only), key = "key", value = "value")
plot1 <- ggplot(data = period_long, mapping = aes(x = key, y = value)) +
  geom_boxplot() +
  theme_bw() +
  labs(title = "boxplot of simulated period - origin Y")
modulus_long <- gather(data.frame(inff_modulus_only), key = "key", value = "value")
plot2 <- ggplot(data = modulus_long, mapping = aes(x = key, y = value)) +
  geom_boxplot() +
  theme_bw() +
  labs(title = "boxplot of corresponding modulus - origin Y")

grid.arrange(plot1,plot2,ncol = 2)
```



*** Period

We could find that first periodic components which has the largest modulus among candidiates has the shorted period on average. Over 75% simulated periods for first components were lower than 3. On the other hand, the second components has the longest period on average. Furthermore, variance in simulated period is also large. Lastly, period of third components is longer than first component and shorter than second components.

*** Decaying rate

We can check that the first periodic component damped by 0.87 at every 3 period on average. Decaying rate for the second component is faster than the first components. However, the third components has even faster decaying rate than the second component

*** Differenced Data

In contrast to original data, most of simulated processes for differenced data are completely consist of periodic components.

```
idx <- apply(diff_sim_eigen$period,1,function(x){sum(x==2)}) == 0
diff_period_only <- data.frame(t(apply(diff_sim_eigen$period,1,function(x){unique(abs(x))}))[idx,]) %>%
  `colnames<-`(paste("comp",1:4))
diff_modulus_only <- data.frame(t(apply(diff_sim_eigen$modulus[idx,],1,function(x){unique(abs(x))}))) %
  `colnames<-`(paste("comp",1:4))

kable(summary(diff_period_only),caption = "summary of period")
```

Table 6: summary of period

comp 1	comp 2	comp 3	comp 4
Min. : 2.478	Min. : 2.424	Min. : 2.379	Min. : 2.118
1st Qu.:16.640	1st Qu.: 3.029	1st Qu.: 2.952	1st Qu.: 2.424
Median :17.858	Median : 6.308	Median : 3.050	Median : 2.466
Mean :16.169	Mean : 6.741	Mean : 4.736	Mean : 2.728
3rd Qu.:18.899	3rd Qu.: 6.583	3rd Qu.: 6.412	3rd Qu.: 2.529
Max. :24.858	Max. :24.048	Max. :24.073	Max. :23.848

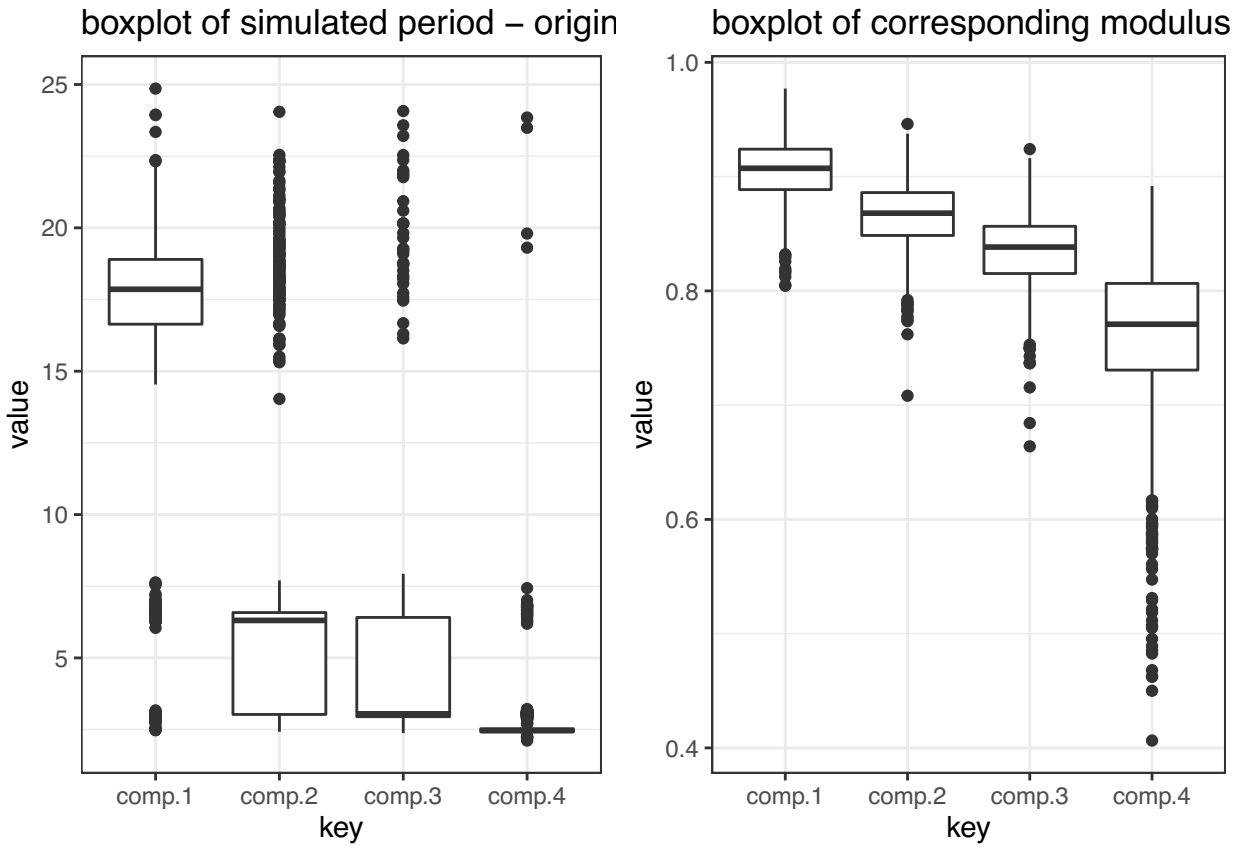
```
kable(summary(diff_modulus_only),caption = "summary of corresponding mod")
```

Table 7: summary of corresponding mod

comp 1	comp 2	comp 3	comp 4
Min. :0.8044	Min. :0.7082	Min. :0.6640	Min. :0.4065
1st Qu.:0.8887	1st Qu.:0.8485	1st Qu.:0.8152	1st Qu.:0.7307
Median :0.9073	Median :0.8681	Median :0.8384	Median :0.7708
Mean :0.9057	Mean :0.8660	Mean :0.8350	Mean :0.7593
3rd Qu.:0.9240	3rd Qu.:0.8860	3rd Qu.:0.8566	3rd Qu.:0.8065
Max. :0.9771	Max. :0.9461	Max. :0.9241	Max. :0.8918

```
period_long <- gather(data.frame(diff_period_only), key = "key", value = "value")
plot1 <- ggplot(data = period_long, mapping = aes(x = key, y = value)) +
  geom_boxplot() +
  theme_bw() +
```

```
  labs(title = "boxplot of simulated period - origin Y")
modulus_long <- gather(data.frame(diff_modulus_only), key = "key", value = "value")
plot2 <- ggplot(data = modulus_long, mapping = aes(x = key, y = value)) +
  geom_boxplot() +
  theme_bw() +
  labs(title = "boxplot of corresponding modulus - origin Y")

grid.arrange(plot1,plot2,ncol = 2)
```



*** Period

We could find that from first component to fourth component, period of each components has been decreased. First component has the longest period which is 16 on average and it has median value that is 18. On contrary, fourth component has the shortest period which is 2.7 on average.

*** Decaying rate

We could find that modulus for each components has same order as their period. First component has the largest modulus which indicates slow decaying. At every period, first component damped by 0.9 on average. On contrary, fourth component has the fastest decaying rate which damped by 0.76 on average at every period.

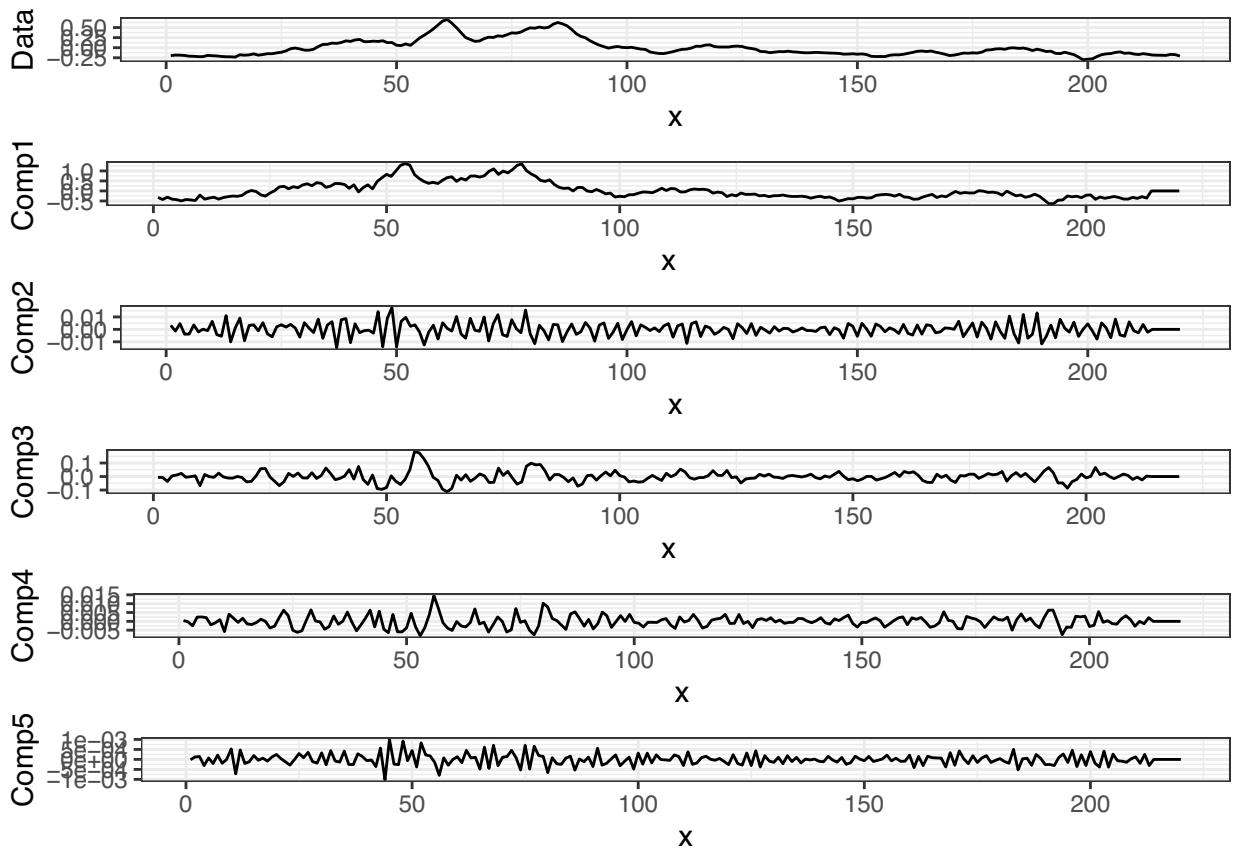
(c)

Explore and discuss aspects of inference on the implied decomposition of the series into underlying components implied by the eigenstructure of the AR model.

I have decomposed each process by implied components.

*** Original Inflation data

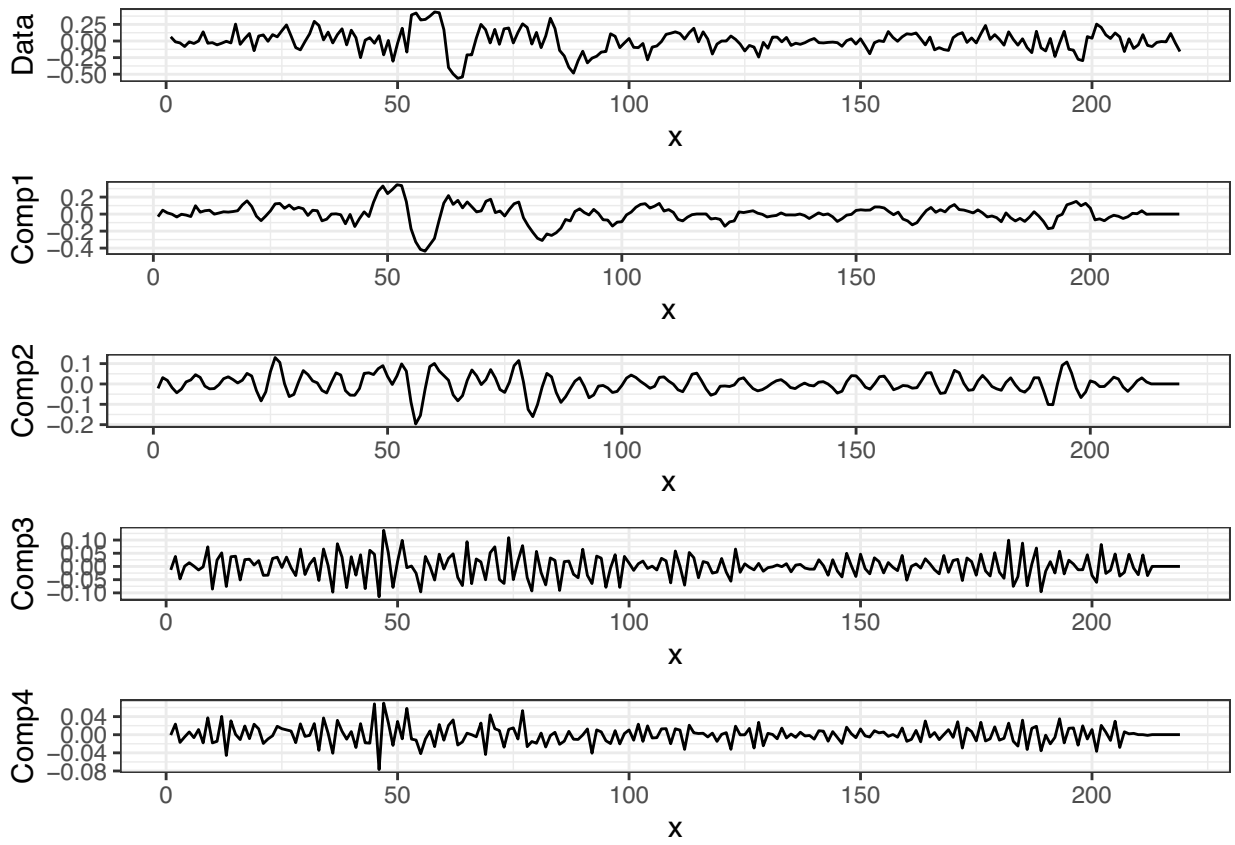
```
inff_decomp <- decomp(data = inff, p = 8)
decomp_plot(data = inff, decomp = inff_decomp, k = 8)
```



We can find that Original data is consist of distinct 5 components. When we investigate plot of first component, we can see that it is very similar with actual data. As we saw at previous question, first component usually has real value with high probability. After first components, we can also find that implied three periodic components. The order of plot is determiend by their modulus which indicate their contribution for actual process. Moreover, as we could see at question2, the period of first component among periodic components has the shortest period whereas the second component which is denoted as comp3 has the longest period among periodic components.

*** Differenced data

```
diff_decomp <- decomp(data = inff_diff, p = 8)
decomp_plot(data = inff_diff, decomp = diff_decomp, k = 8)
```



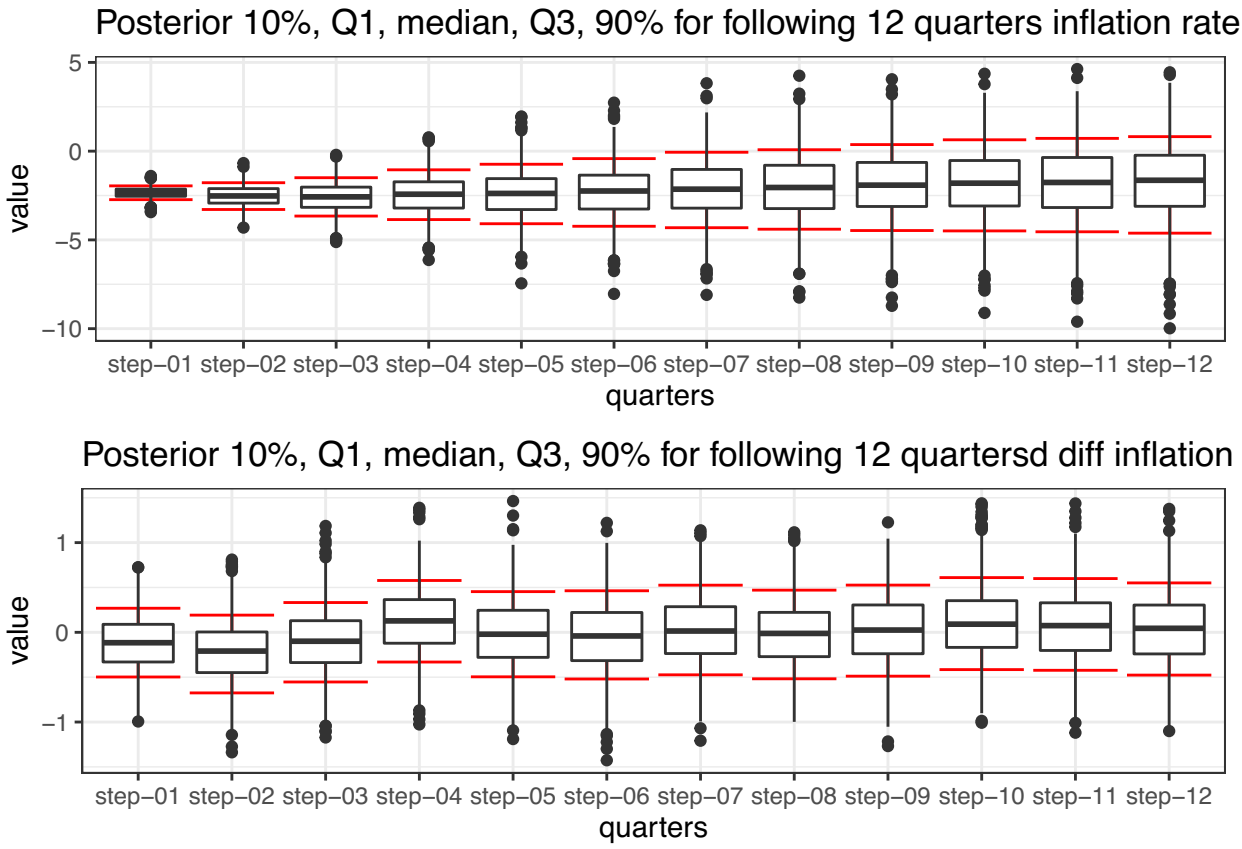
We can find that differenced data is consist of four implied periodic components. As original data's plots, these plots are ordered by their contribution. The period of each components are corresponding with result from question2. First components has the largest contribution to process and the longest period. We can see that process of component1 is very simliar with original data. Moreover, we can find that component4 has the shortest period.

(d)

Produce and display graphical summaries– in terms of (Monte Carlo based) posterior medians, upper and lower quartiles, and 10% and 90% points of the predictive distributions of actual inflation over the 12 quarters following the end of the data series.

```
inff_pred <- data.frame(inff_result$pred) %>%
  `colnames<-`(paste("step",formatC(1:12,flag = 0,digits = 1),sep = "-"))
diff_pred <- data.frame(diff_result$pred) %>%
  `colnames<-`(paste("step",formatC(1:12,flag = 0,digits = 1),sep = "-"))

inff_pred_long <- gather(inff_pred, key = "key", value = "value")
diff_pred_long <- gather(diff_pred, key = "key", value = "value")

inff_pred_intv <- apply(inff_pred, 2, function(x){quantile(x,c(0.1,0.9))}) %>%
  t() %>%
  as.data.frame()
diff_pred_intv <- apply(diff_pred, 2, function(x){quantile(x,c(0.1,0.9))}) %>%
  t() %>%
```

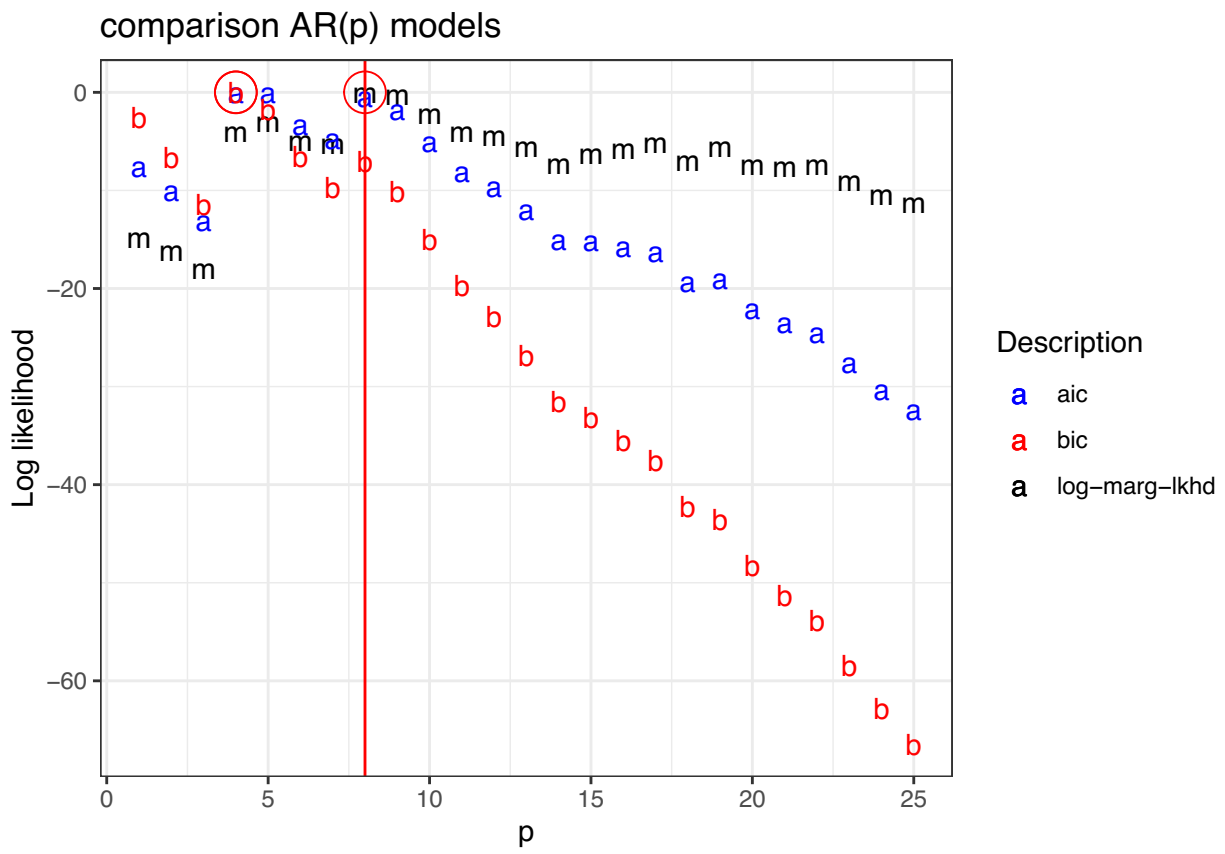
```
  as.data.frame()


inff_intv_plot <- ggplot() +
  geom_errorbar(data = inff_pred_intv,
                mapping = aes(x = paste("step",formatC(1:12,flag = 0,digits = 1),sep = "-"), ymin = `10%
                col = "red") +
  geom_boxplot(data = inff_pred_long, mapping = aes(x = key, y = value)) +
  labs(title = "Posterior 10%, Q1, median, Q3, 90% for following 12 quarters inflation rate", x = "quar

diff_intv_plot <- ggplot() +
  geom_errorbar(data = diff_pred_intv,
                mapping = aes(x = paste("step",formatC(1:12,flag = 0,digits = 1),sep = "-"), ymin = `10%
                col = "red") +
  geom_boxplot(data = diff_pred_long, mapping = aes(x = key, y = value)) +
  labs(title = "Posterior 10%, Q1, median, Q3, 90% for following 12 quartersd diff inflation rate", x =

grid.arrange(inff_intv_plot,diff_intv_plot, nrow = 2)
```



(e)

Assuming an AR(p) model is agreeable, do you think p = 8 makes sense for the differenced inflation series? Consider, for example, features of the fitted residuals from the model. In addition to exploratory and graphical analysis– and all you know about applied evaluation of linear regression model "fits"– the arpcompare.m function (noted and advised to review in Homework #2) may be of interest.

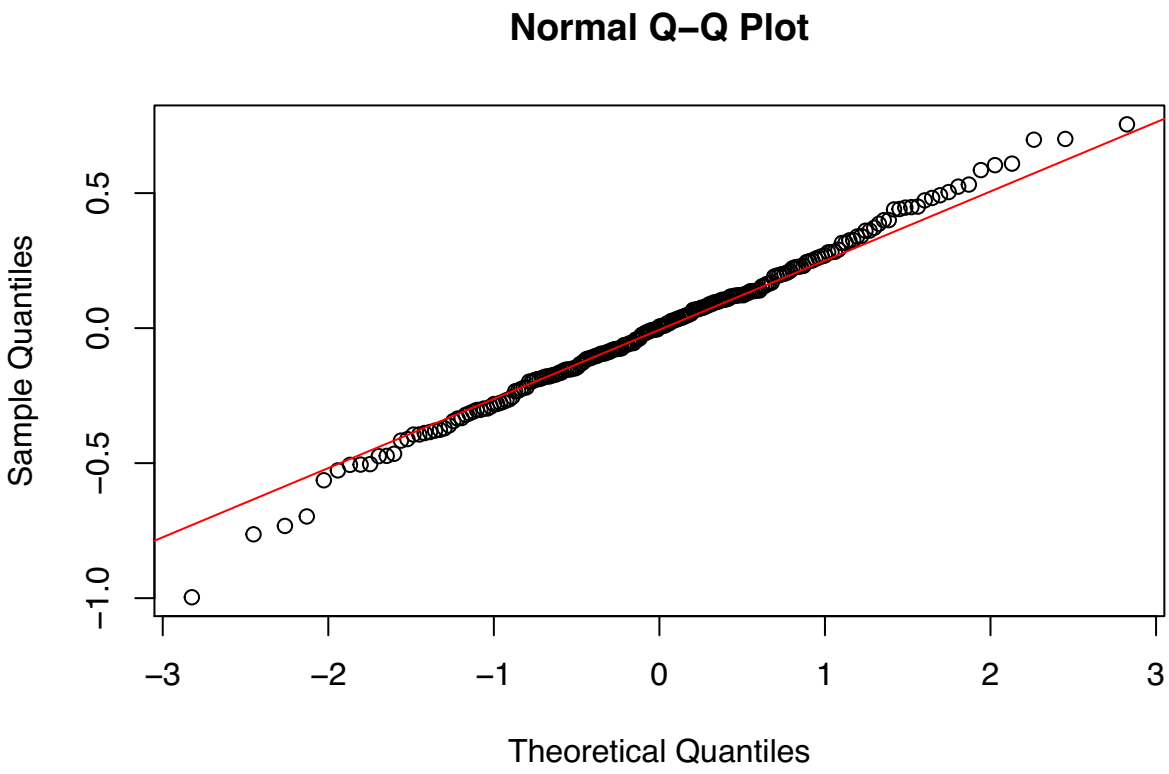
```r
AR_compare <- function(data, maxp){
  data <- data - mean(data)
  loglkhd <- aic <- bic <- rep(NA,maxp)
  #y_n ~ y_
  for(p in 1:maxp){
    y <- data[(p+1):length(data)]
    #Hankel matrix
    design = matrix(rep(NA,(length(data)-p)*p),ncol = p)
    for(i in 1:(length(data)-p)){
      design[i,] = rev(data[i:(i+p-1)])
    }
    #H'H
    B <- t(design) %*% design
    #B^{-1}H'y
    b <- solve(B) %*% t(design) %*% y
    #SSR
    Qb <- t(y-design %*% b) %*% (y - design %*% b)
    #loglkhd
    n <- length(y)
    nu <- n-p

    loglkhd[p] <- (-nu*log(pi) - log(det(B)) - nu*log(Qb))/2 + log(gamma(nu/2))
    ic <- n*log(Qb/nu)
    aic[p] <- -(2*p+ic)/2
    bic[p] <- -(log(n)*p+ic)/2
  }
  return(list(loglkhd = loglkhd, aic = aic, bic = bic))
}
maxp <- 25
compare <- AR_compare(inff_diff, maxp)
scaled_AR <- map_df(compare,function(x){x - max(x)})
maxpoint <- map_df(scaled_AR, function(x){c(max(x), which.max(x))}) %>% t() %>% as.data.frame()

ggplot(data = scaled_AR) +
  geom_text(mapping = aes(x = 1:maxp, y = loglkhd, label = "m", col = "log-marg-lkhd")) +
  geom_text(mapping = aes(x = 1:maxp, y = aic, label = "a", col = "aic")) +
  geom_text(mapping = aes(x = 1:maxp, y = bic, label = "b", col = "bic")) +
  geom_point(data = maxpoint, mapping = aes(x = V2, y = V1),pch = 1, fill=NA, size = 7,col = "red") +
  geom_vline(xintercept = 8, col = "red") +
  labs(title = "comparison AR(p) models", x = "p", y = "Log likelihood") +
  scale_color_manual("Description", values = c("blue","red","black"))
```



*** Comparison among candidate models

I think AR(8) model is the best model in terms of model's log marginal likelihood. As we can see at above graph, at p = 8, the model has the largest log marginal likelihood. However, if we use BIC as criteria, AR(8) is not the best model. We can find that BIC indicate that p = 4 is the best model. We can find that AIC shows simliar result at p = 4 and p = 8. These difference might be caused from penalty AIC and BIC give on the number of parameters and BIC often gives more strict penalty on the number of parameter. Therefore, if we do not consider penalty of the number of parameters and focus on perdiction power of model, p = 8 can be considered as appropriate model for differenced inflation data.

```
res <- AR_p(inff_diff,8)$res
qqnorm(res)
qqline(res, col = "red")
```



*** Assumption of normality

When we examine normality of residual from AR(8) model, we can find hat normality of residual seems to be satisfied.