Computational Statistics Assessed Practical

P648

MSc. Statistical Science, TT 2021, Week 1

1 Exercise 1

This exercise includes questions related to estimation variance of GLM parameters using bootstrap methods and the "sandwich" estimator.

1.1 Model Specification

The Poisson GLM is fitted with the canonical link, using the **docvis** variable as response and 6 explanatory variables as independent main effects. The estimates of model parameters and standard errors can be easily obtained. In addition, estimates given by the "sandwich" estimators which are more robust to heteroscedasticity are also computed for the same model, with the estimation type set as "HC0", leading to using the White's estimator for the covariance. The table below summarises the model parameters and their corresponding standard errors by two estimators.

Term	intercept	age	hhninc	female	hhkids	educyrs	addins
$\hat{\beta}$	0.364	0.008	-0.064	0.338	-0.113	-0.023	0.406
standard se($\hat{\beta}$)	0.182	0.002	0.016	0.049	0.053	0.011	0.127
sandwich se($\hat{\beta}$)	0.254	0.003	0.021	0.068	0.074	0.014	0.195

Table 1: Model Estimation

The results show that the robust sandwich estimators give larger estimates of the parameter standard errors. This happens because the standard GLM variance estimator ignores the heteroscedasticity and underestimates the variances. In this case, the sandwich estimators would be more reliable, while using the standard GLM estimates would cause various problems, such as getting too narrow confidence intervals and too small p-values when conducting hypothesis tests.

1.2 Bootstrap Standard Errors

Then I use non-parametric paired bootstrap to estimate the standard errors of the coefficients. A brief outline of the method is given as below in Algorithm 1.

Algorithm 1: Non-parametric Paired Bootstrap

Data: $D = \{(X_1, y_1), ..., (X_n, y_n)\}$
Result: Compute bootstrap mean and covariance of model parameters
for $b = 1, ..., B$ **do**
  draw samples with replacement $\{(X_1^{*(b)}, y_1^{*(b)}), ..., (X_n^{*(b)}, y_n^{*(b)})\}$ from D;
  compute $\hat{\beta}^{*(b)}$ using poisson $glm(y^{*(b)} \sim X^{*(b)})$;
end
compute $\overline{\hat{\beta}^*} = \frac{1}{B}\sum_{b=1}^{B} \hat{\beta}^{*(b)}$;
compute $\hat{cov}(\hat{\beta}) = \frac{1}{B}\sum_{b=1}^{B}(\hat{\beta}^{*(b)} - \overline{\hat{\beta}^*})(\hat{\beta}^{*(b)} - \overline{\hat{\beta}^*})^T$;

Note that $\hat{cov}(\hat{\beta})$ is the covariance matrix of the parameters. The individual standard errors are the square roots of the corresponding diagonal entries as listed in the table below. The bootstrap standard errors are also larger than those obtained through standard GLM and approximately equal to the sandwich estimates for every term.

Term	intercept	age	hhninc	female	hhkids	educyrs	addins
bootstrap se(beta)	0.256	0.003	0.021	0.069	0.075	0.014	0.205

Table 2: Parameter standard errors estimated by paired bootstrap

1.3 Wald Tests

Based on the estimated coefficients and standard errors obtained by various methods, I then compute the Wald statistics and p-values to test the hypothesis $\hat{\beta}_{hhkids} = \hat{\beta}_{educyrs} = 0$. Note that the test result can be used as evidence for model selection between the full and the nested model with the above two terms left out.

Denote $\beta_t = (\beta_{hhkids}\ \beta_{educyrs})^T$. Under the null H0: $\beta_t = 0$, the Wald statistic is $\hat{\beta_t}^T(\hat{cov}(\hat{\beta}_t))^{-1}\hat{\beta}_t$ and has a χ_2^2 distribution asymptotically. The estimates $\hat{\beta}_t$ are fixed, and the standard, sandwich and bootstrap methods give different estimates for the covariance matrix. The resulting Wald test statistics for the three estimates are $8.71, 4.86$ and 4.79, and the p-values are thus $Pr(W > W_{obs})$ for $W \sim \chi_2^2$, and are $0.013, 0.088$ and 0.091 respectively.

The above results coincide with the conclusion drawn from the theory in the first

section. Ignoring the heteroscedasticity among the samples indeed cause the Wald statistic derived from the standard GLM covariance estimates being too significant and p-value being too small. Under the level of 5% significance, the null hypothesis might be "falsely" rejected using the standard GLM estimates, which could be avoided with the estimates derived from the bootstrap or sandwich estimator.

1.4 Bootstrap p-value

I then move to implementing bootstrap to estimate the p-value for the Wald test statistic based on the sandwich variance estimator. I use parametric bootstrap to sample from Poisson distribution, with rate parameters being the fitted values obtained from the GLM. Note that under H0, the variables **hhkids** and **educyrs** are not included in the model (denote the data matrix by X_{-t}), so the GLM fitted here is different from the one used in previous sections.

Algorithm 2: Parametric Bootstrap

Data: $D = \{(X_1, y_1), ..., (X_n, y_n)\}$
Result: Compute p-value of sandwich estimator Wald statistic
fit $glm(y \sim X_{-t})$ with the two variables dropped;
compute the fitted values $\hat{\mu} = e^{\hat{\eta}} = e^{X_{-t}\hat{\beta_{-t}}}$;
for $b = 1, ..., B$ **do**
  draw samples $(y_1^{*(b)}, y_2^{*(b)}, ..., y_n^{*(b)})$ where $y_i^{*(b)} \sim Poi(\hat{\mu}_i)$;
  fit $glm(y^{*(b)} \sim X)$ to obtain $\hat{\beta}^{*(b)}$;
  use sandwich estimator to compute $\hat{cov}(\hat{\beta}^{*(b)})$;
end

Now a series of $\{\hat{\beta^{*(b)}}\}_{b=1}^{B}$ and their covariance matrices are obtained by bootstrap. To compute a p-value for the Wald test statistic based on the sandwich estimator, it suffices to compute the Wald statistics of the bootstrap samples $W^{*(b)} = \hat{\beta}^{*(b)T}(\hat{cov}(\hat{\beta}^{*(b)}))^{-1}\hat{\beta}^{*(b)}$, and an estimated p-value for $W = 4.86$ is

$$p = \frac{1}{B}\sum_{b=1}^{B} I(W^{*(b)} > 4.86) = 0.092$$

The result is very close to the theoretical p-value of the sandwich estimate (0.088) with a percentage difference of 4.68%, so the Wald statistic based on the sandwich estimator is approximately unchanged compared to that found in the last section.

1.5 Comments regarding model selection

The above results and comparisons among the three variance estimators reveal the problem of standard model selection procedure. The standard GLM covariance estimator ignores the heteroscedasticity in the samples, which is shown to cause the

estimates of variances and standard errors to be wrongly small. When testing nested models against full models, the Wald statistics would possibly be too large to retain H0 and wrongly favour the full model. The analysis above also justifies robustness of the sandwich variance estimator via comparison with bootstrap and the unchanged bootstrap p-value. Hence, when analysing the model such as doing model selection, if an estimate of the model parameter covariance is required, the sandwich variance estimator would be much more desirable than the standard GLM variance estimator.

2 Exercise 2

This chapter includes my analysis on the annual inflation rate for France from 1956 to 2020.

2.1 Exploratory Data Analysis

A natural graph to draw when analysing time series is the line chart. Fig.1 shows the inflation rate by year with 3- and 7-year moving averages and the fitted linear model equation. The inflation rate experienced large fluctuations prior to 1986. Due to the rapid changes taking place around 1960, 1975 and 1980, I do not expect the model to give precise predictions during those years. The rate exceeded 5% most of the time until the year 1986, after which the rate became more stable and never returned to the 5% level and the curves show some smaller ups and downs. The fitted line suggests that in general the inflation rate has a decreasing trend.

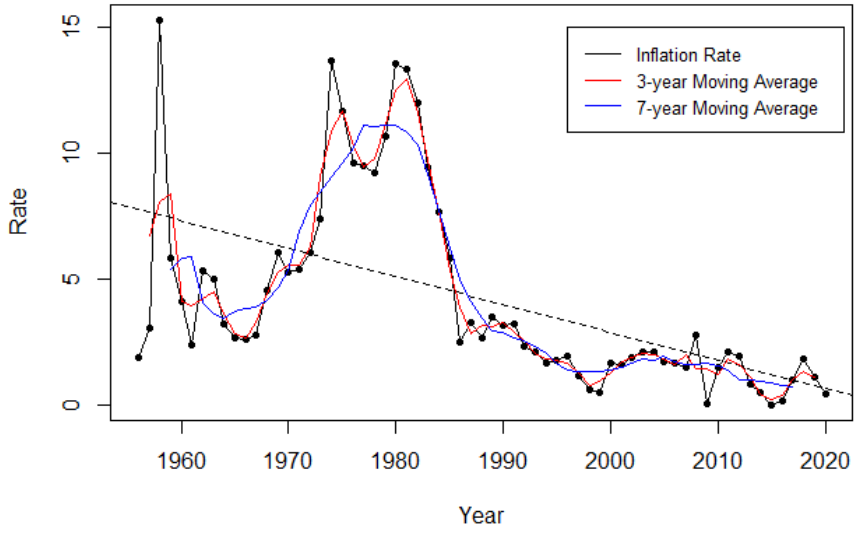

Fig. 1: Real Consumer Price Annual Inflation Rate

Properties such as order and number of lags of time series can be inferred from the auto- and partial auto-correlation sequences. The ACF plot is long tailed and the PACF has a cut off after the first lag, which means that the time series is approximately an AR(1) process.

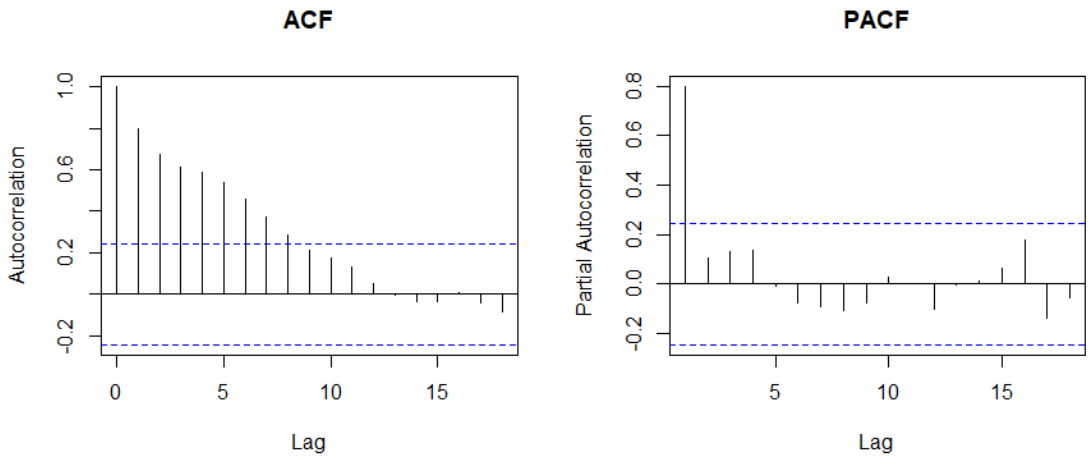

Fig. 2: Autocorrelation and Partial Autocorrelation Sequences

2.2 Fixed Coefficient AR(1) Model

I then fit a standard AR(1) model of the inflation rate data using the *arima* function in *R*. The model is specified as

$$Y_t = \alpha + Y_{t-1}\beta + W_t$$

where W is the random noise with unknown variance σ_W^2. The fitted model coefficients are $\hat{\alpha} = -0.067, \hat{\beta} = 0.801$, and $\hat{\sigma}_W^2 = 5.183$. Fig. 3 shows the model's fitted value, where the curve just corrects the inflation rate in the previous years moderately with same **scale** and **additive constant**. The residual against fitted value plot looks fine and shows no heteroscedatic behaviour, except for several samples corresponding to years when the inflation rate changed significantly. In fact, even though the random noise is estimated to have variance much larger than 1, most residuals lie between ± 2 and follow the standard normal distribution. The estimated variance is dominated by the samples with extreme values that the model cannot fit well and produce large residuals. Meanwhile, I choose not to treat them as outliers, as they are accurately recorded and can be found on the Internet.

However, this model is not desirable, since the fitted model cannot capture the sudden changes of the inflation rates, while accurate predictions only take place **by chance**. Hence, AR(1) model with fixed coefficients is not adequate to fit the data.

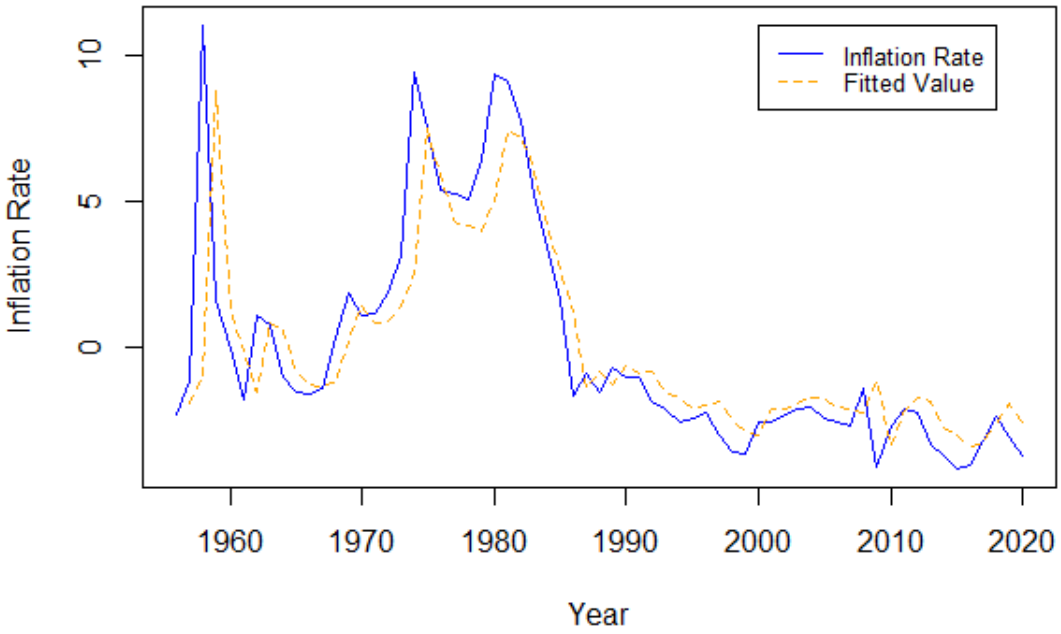

Fig. 3: Inflation rate and fitted values by AR(1)

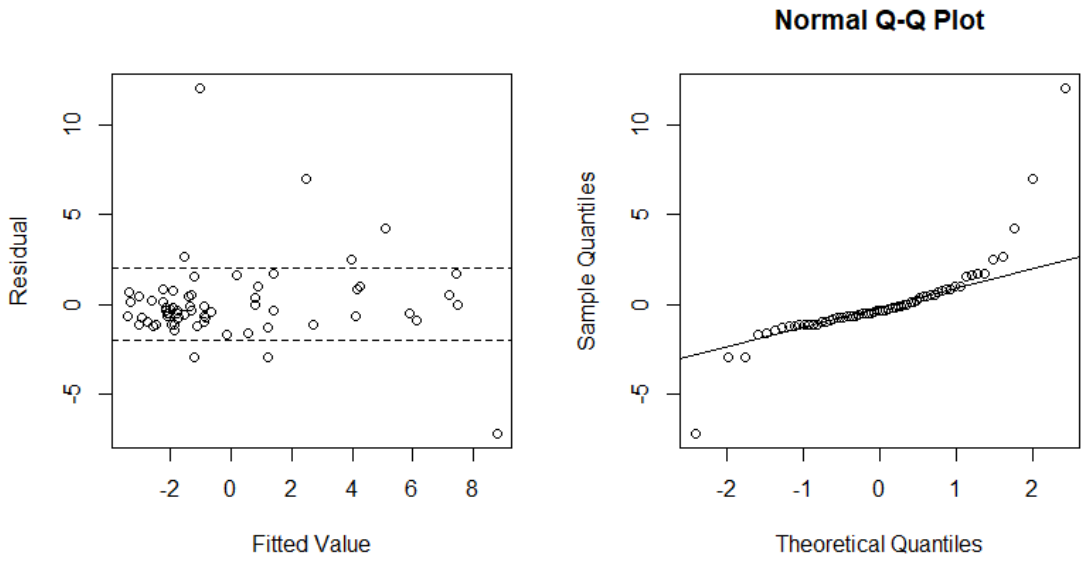

Fig. 4: Residual against Fitted Value and Q-Q Plot

2.3 AR(1) Model of Varying Coefficient

To better capture the features in the time series, I then fit AR(1) models with time varying coefficients as linear Gaussian state-space models.

The model is specified as

$$\beta_t = \beta_{t-1} + V_t \tag{1}$$
$$Y_t = Y_{t-1}\beta_t + W_t \tag{2}$$

for $t = 1, ..., T$=64. The initial state $\beta_0 \sim N(\mu_0, \sigma_0^2)$, $\{V_t\}_{t=1}^T$ and $\{W_t\}_{t=1}^T$ are state and observation noise, assumed to be independent $N(0, \sigma_v^2)$ and $N(0, \sigma_w^2)$ respectively. I will use Kalman filter to fit the data and estimate the hidden states $\{\beta_t\}_{t=0}^T$.

Variances of Noises Specified Firstly, assume that $\sigma_v^2 = 0.01$ and $\sigma_w^2 = 4$, I run the Kalman filter to compute the filtering mean and variance for the hidden state β_t given $y_0, ..., y_t$ for every t. Note that the observation matrix, denoted by H_t in the lecture, is given by Y_{t-1} from (2), namely the current observation Y_t depends on the hidden state β_t as a multiplicative factor on Y_{t-1}.

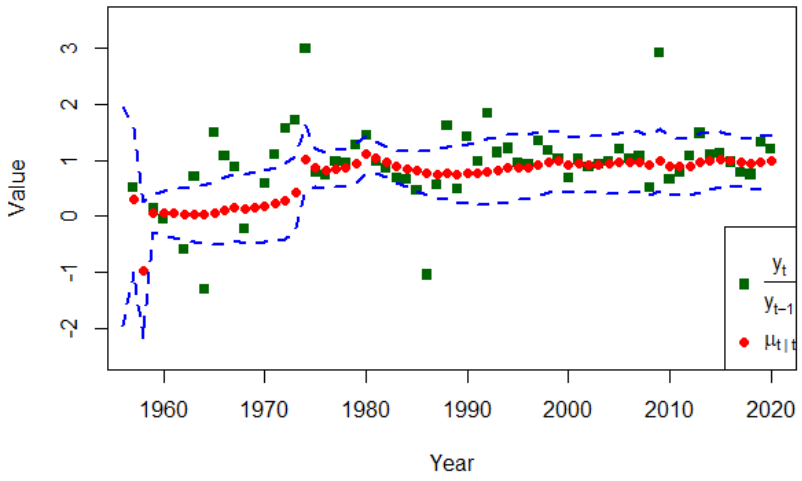

Fig. 5: Filtering mean and credible interval

I plot the fitted filtering mean $\mu_{t|t} = E(\beta_t|y_0, ..., y_t)$ and a 95% credible interval in Fig. 5. I include the ratios $\frac{y_t}{y_{t-1}}$ as a reference, based on the fact that $E(Y_t) = \beta_t E(Y_{t-1})$. The fitted mean is expected to be not far from the ratio which is satisfied as shown in the plot. One step predictions are given by $\hat{y}_t = y_{t-1}\mu_{t|t}$ and residuals are computed. The residual plots in Fig. 6 look better than those given by the fixed coefficient AR model, with no trends against fitted values and fewer samples having large residuals.

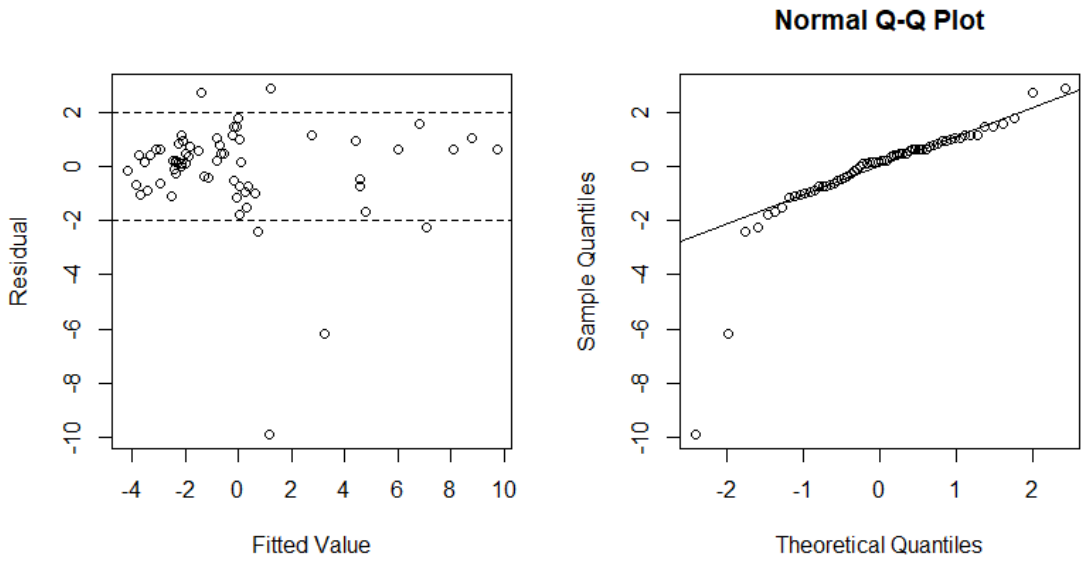

Fig. 6: Residual against Fitted Value and Q-Q Plot

Find Optimal Noise Variances Now I want to find values of σ_v^2 and σ_w^2 that maximise the log-likelihood $\log p(y_1, ..., y_T|y_0)$. It can be rewritten as $\log p(y_1|y_0) + \sum_{t=2}^{T} \log p(y_t|y_0, ..., y_{t-1})$, where each likelihood is Gaussian with mean $H_t\mu_{t|t-1}$ and variance $H_t^2\Sigma_{t|t-1} + \sigma_w^2$. Hence, by adapting the Kalman filter and adding the log-likelihood of y_t given $y_0, ..., y_{t-1}$ at each iteration, the log-likelihood can be computed conveniently. When setting $\sigma_v^2 = 0.01, \sigma_w^2 = 4$, the log-likelihood is -141.5208.

To numerically search for the maximum likelihood estimator of σ_v^2 and σ_w^2, I grid search over two lists of candidate values for the two variances. The MLE is $(\hat{\sigma}_v^2, \hat{\sigma}_w^2) = (0.01, 4.02)$ which give the log-likelihood -141.5203.

Refit Under MLE(?) Despite the difference between the default setting of the variances and the MLE, refitting the model under the MLE does not change the fitted filtering mean and credible interval too much. The plot looks exactly the same as Fig. 5 so I omit it here. This happens since the variance and likelihood are approximately equal to those of the original model.

Interpretation Here I want to summarise some observations from Fig. 5.

- From 1960 to 1975, the filtering mean remains at approximately zero, while the ratio $\frac{y_t}{y_{t-1}}$ shows significant variation. (In fact, some extremely large values of the ratio were not shown on the plot). Since $\frac{Y_t}{Y_{t-1}} = \beta_t + \frac{W_t}{Y_{t-1}}$, during this period, the fluctuations are mostly treated as the effects by the observational noise W_t.

- After 1975, the filtering mean starts to stabilise around 1, indicating that the inflation rate Y_t becomes stable, which can be verified from the curves in Fig. 1. The 95% CI for the filtering mean contains the ratio $\frac{Y_t}{Y_{t-1}}$ most of the time.
- From 1990, the CI starts to have constant width as well. Hence the Gaussian hidden state β_t becomes approximately identically distributed since the year 1990, as the filtering mean and variance of β_t both become constant.
- Throughout the whole period, the filtering mean is stable except for the sudden changes at 1958 and 1975 when the inflation rate changed drastically. This coincide with the deterministic equation (1) for the hidden state, where the change in β_t has variance $\sigma_v^2 = 0.01$.

2.4 Conclusion & Discussion

In this Chapter, I explored the Inflation Rate data set, conducted some exploratory analysis in Section 2.1 and favoured AR(1) models based on the auto-correlation plots. After discussing the incompatibility of the fixed coefficient model with the data, I moved to fitting linear Gaussian state-space models using Kalman filter, with model components specified in equations (1) and (2) in 2.3. I firstly specified variances of the hidden states σ_v^2 and observations σ_w^2 to be 0.01 and 4 respectively and adjusted the Kalman filter function to calculate the log-likelihood of the data under the model. The optimal values for the variances which gave the maximum likelihood were very close to the initial settings. Therefore, I opted the original model and interpreted my findings of the filtering mean and credible interval from Fig. 5.

There are several points I want to make regarding the strengths and weaknesses of the statistical methods I used. The fitted AR(1) model is capable of modelling the data after 1975, while prior to that when the inflation rate experienced large fluctuations, the model cannot capture the serial correlation in the data. The Kalman filter used to fit the model can make predictions based on data until the current year. If the goal is instead to **fit the data**, **Kalman smoother** which takes **all observations** into account might give better performance. Furthermore, the Gaussian state-space model makes the problem easily solvable, as the filtering mean and variance can be analytical computed. However, normality might not be satisfied in real world problems, and the variances of observations and hidden states might also be time varying, which are naively assumed to be constant throughout.

I have some final remarks that might be considered for future work. On one hand, the scale of inflation rate changed by time, so we might consider models whose variance of noise is also time varying. On the other hand, we can explore other models, such as multivariate time series models by collecting other features as explanatory variables, or non-Gaussian models such as GLMs, especially when normality of the model components is not satisfied. However, based on the available data, the AR(1) model fitted by Kalman filter is favoured by my analysis.

R Code

```
#########################################################
#################### Exercise 1 ########################
#########################################################

library(readr)
dvis <- read_csv("D:/MSc/Practicals/CompStats-Week1-TT21/dvis.csv")

glm <- glm(docvis~1+age+hhninc+female+hhkids+educyrs+addins,
           data=dvis, family=poisson())
beta.hat <- glm$coefficients
summary(glm) # Extract s.e's

#################################################################
# (Intercept)     age      hhninc     female      hhkids      #
# 0.182105    0.002273     0.015997   0.048738    0.053270    #
# educyrs      addins                                         #
# 0.010740     0.126796                                       #
#################################################################

library(sandwich)
rcov1 <- vcovHC(glm,type='HC0') # sandwich estimators of covariance
se1 <- sqrt(diag(rcov1)) # s.e = square root of diagonal entry
se1

#################################################################
# (Intercept)     age      hhninc      female      hhkids     #
# 0.254294467 0.003312846 0.020792797 0.068652805 0.074212943 #
# educyrs      addins                                         #
# 0.014424773 0.194750943                                     #
#################################################################

B <- 1e5
n <- nrow(dvis)
beta.mat <- matrix(data=NA,nrow=B,ncol=7)
for (i in 1:B){
  rows <- sample(1:n, replace=T) # Indices of subsampled data
  glm.boot <- glm(docvis~1+age+hhninc+female+hhkids+educyrs+addins,
              data=dvis[rows,], family=poisson()) # Fit bootstrapped data
  beta.boot <- glm.boot$coefficients # Extract estimates
  beta.mat[i,] <- beta.boot # Record
}

se.boot <- sqrt(diag(var(beta.mat))) # Compute s.e's

#################################################################
# (Intercept)         age      hhninc      female      hhkids #
# 0.255553659 0.003329749 0.020978023 0.068824701 0.074631902 #
# educyrs      addins                                         #
# 0.014497522 0.205093932                                     #
#################################################################

betas <- beta.hat[c('hhkids','educyrs')] # Components of Interest
# Calculate Wald statistics based on different variance estimates
W.glm <- t(betas)%*%solve(vcov(glm)[5:6,5:6])%*%betas #8.71
```

```
p.glm <- pchisq(8.71,2,lower=FALSE) # 0.01284244
W.sandwich <-  t(betas)%*%solve(rcov1[5:6,5:6])%*%betas #4.86
p.sandwich <- pchisq(4.86,2,lower=FALSE) # 0.08803683
W.boot <- t(betas)%*%solve(var(beta.mat)[5:6,5:6])%*%betas #4.79
p.boot <- pchisq(4.79,2,lower=FALSE) # 0.09117268


################################################
######### Parametric Bootstrap under H0 #########
################################################
W.vec.para <- vector(length=B)
glm0 <- glm(docvis~1+age+hhninc+female+addins,data=dvis, family=poisson())
mu.hat0 <- glm0$fitted.values
for (i in 1:B){
  y.boot <- rpois(1204, mu.hat0) # Sample from H0
  glm.boot.para <- glm(y.boot~1+age+hhninc+female+hhkids+educyrs+addins,
                 data=dvis, family=poisson()) # Fit full model
  beta.boot.para <- glm.boot.para$coefficients
  beta.s <- beta.boot.para[c('hhkids','educyrs')]
  rcov.boot <- vcovHC(glm.boot.para,type='HC0')
  W.vec.para[i] = t(beta.s)%*%solve(rcov.boot[5:6,5:6])%*%beta.s # Wald statistic
}

p.value.para <- sum(W.vec.para>4.86)/B #0.09216
(p.value.para-0.088)/0.088 = 0.04683457 # 4.68% difference from theoretical value

#########################################################
#################### Exercise 2 ########################
#########################################################

dat.infl <- read_csv("D:/MSc/Practicals/CompStats-Week1-TT21/infl.csv")

inflFR<-dat.infl$inflFR
year<-dat.infl$year

#################################
############# EDA ###############
#################################

par(mfrow=c(1,1))

plot(inflFR~year, type='l',
     xlab='Year', ylab='Rate')
points(year,inflFR, pch=16,cex=0.7)
library(zoo)
ma3 = rollmean(inflFR,3,fill = list(NA,NULL,NA))
ma7 = rollmean(inflFR,7,fill = list(NA,NA,NA,NULL,NA,NA,NA))
abline(lm(inflFR~year),lty=2)
lines(year,ma3, col='red')
lines(year,ma7, col='blue')
legend(1996,15,legend=c("Inflation Rate", "3-year Moving Average",
                        "7-year Moving Average"),
       col=c("black","red", "blue"),lty=1,cex=0.8)
title('Real Consumer Price Annual Inflation Rate')

par(mfrow=c(1,2))
```

```
acf(inflFR, ylab='Autocorrelation',xlab='Lag',main="ACF")
pacf(inflFR,ylab='Partial Autocorrelation',xlab='Lag',main="PACF")

inflFR <- inflFR - mean(inflFR) # Centering

####################################
############## AR(1) ###############
####################################

AR <- arima(inflFR, order = c(1,0,0))
print(AR)
# intercept is the value of \mu.
# c would be \mu*(1-\phi_1)
residuals <- residuals(AR)
AR.fitted <- inflFR - residuals

plot(dat.infl$year,inflFR, type='l',col='blue',xlab='Year',ylab='Inflation Rate')
points(year[2:65],AR.fitted[2:65], type = "l", col = 'orange', lty = 2)
legend(2000,11,legend=c("Inflation Rate", "Fitted Value"),
       col=c("blue","orange"),lty=c(1,2),cex=0.8)

# Residual Plots
par(mfrow=c(1,2))
plot(array(AR.fitted),array(residuals),type='p',
     xlab='Fitted Value',ylab='Residual')
abline(a=2,b=0,lty=2)
abline(a=-2,b=0,lty=2)

qqnorm(residuals)
qqline(residuals)

##################################################################
###################### Kalman Filter #############################
##################################################################

kalman = function(y, F, G, H, Q, R, mu0, Sigma0){

  T = ncol(y)
  ## INITIALIZATION ##

  mu.p =  array(0, T)
  Sigma.p = array(0, T)
  mu.f =  array(0, T)
  Sigma.f = array(0, T)

  ## FORWARD RECURSION
  ## Time 1
  mu.p[1] = F * mu0
  Sigma.p[1] = F**2*Sigma0 + G**2 * Q
  nu1 = y[1] - H[1]*mu.p[1]
  S1 = H[1]**2 * Sigma.p[1] + R
  K1 = Sigma.p[1] * H[1] / S1
  mu.f[1] = mu.p[1] + K1  * nu1
  Sigma.f[1] = (1 - K1*H[1]) * Sigma.p[1]
  log.lik = dnorm(y[1],H[1]*mu.p[1],sqrt(S1),log=TRUE) # First term in log.lik
  # Time 2:T
```

```
  for (t in (2:T)) {
    # Prediction
    mu.p[t] = F * mu.f[t-1]
    Sigma.p[t] = F**2 * Sigma.f[t-1]+ G**2*Q

    # Update
    nut = y[t] - H[t] * mu.p[t]
    St = H[t]**2 * Sigma.p[t]+ R
    Kt = Sigma.p[t]*H[t]/St
    mu.f[t] = mu.p[t] + Kt * nut
    Sigma.f[t] = (1- Kt*H[t])*Sigma.p[t]
    # Iteratively add on log.lik
    log.lik = log.lik + dnorm(y[t],H[t]*mu.p[t],sqrt(St),log=TRUE)
  }

  return(list(mu.f = mu.f, Sigma.f = Sigma.f,
              mu.p = mu.p, Sigma.p = Sigma.p,
              log.lik = log.lik))

}

# Model components
F = 1
G = 1
Qa = 0.01
Ra = 4
Sigma0 = 1
mu0 = 0
T0 = length(inflFR)
y = t(inflFR[2:T0])
H = t(inflFR[1:(T0-1)]) # Transition matrix is y_(t-1)

results.KF = kalman(y, F, G, H, Qa, Ra, mu0, Sigma0)

mu.f = results.KF$mu.f
Sigma.f = results.KF$Sigma.f
se.f = sqrt(Sigma.f)
mu.f0 = c(mu0,mu.f)
se.f0 = c(sqrt(Sigma0),se.f)
alpha = 0.05
cv95 = qnorm(1-alpha/2)
CIupper = mu.f0+cv95*se.f0
CIlower = mu.f0-cv95*se.f0

plot(year[-1],y/H, col='darkgreen',pch=15,
     xlab='Year',ylab='Value',ylim=c(-2.5,3.5)) # Reference
points(year[-1],mu.f,col='red',pch=16) # Fitted filtering mean
points(dat.infl$year,CIupper,col='blue',type="l",lty=2,lwd=2) # 95% CI
points(dat.infl$year,CIlower,col='blue',type="l",lty=2,lwd=2)
legend('bottomright',pch=c(15,16),col=c('darkgreen','red'),
       legend=c(expression(frac('y'[t],'y'[t-1]) ),expression(mu[t~"|"~t])) )

# Residual Plots
residual.KF <- y.p - array(y)
par(mfrow=c(1,2))
plot(y.p,residual.KF,type='p',
```

```
     xlab='Fitted Value',ylab='Residual')
abline(a=2,b=0,lty=2)
abline(a=-2,b=0,lty=2)

qqnorm(residual.KF)
qqline(residual.KF)

# Grid Search
Q.range <- c(1:100)/1000
R.range <- c(380:420)/100
lis <- matrix(nrow=4100,ncol=3)
i=1
for (q in Q.range){
  for (r in R.range){
    results.KF.qr <- kalman(y, F, G, H, q, r, mu0, Sigma0)
    log.lik.qr <- results.KF.qr$log.lik
    lis[i,] = c(q,r,log.lik.qr)
    i = i+1
    }
}
n = which.max(lis[,3]) # Which is the MLE
lis[n,] # MLE and max.log.lik # 0.01 4.02 -141.5203
```