STAT 153 Project

Group: Haoyuan Li, Ziyuan Tang
Blogpost(click here)

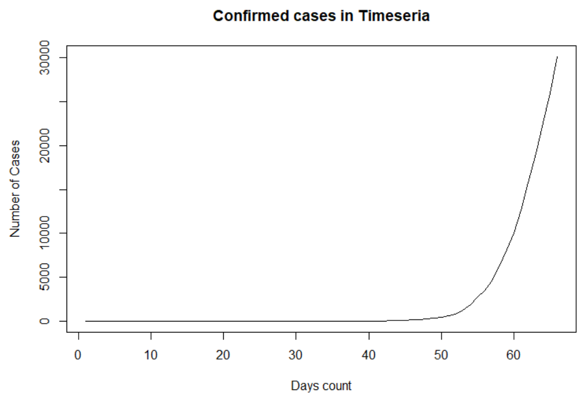
May 3, 2020

Executive Summary:

Many countries are suffering from the novel coronavirus currently, Timeseria is one of them. Here we will use primarily parametric model and SARIMA model to analyze and predict the COVID-19 dataset. From the prediction, we can see how the situation is going to be worse if proper actions are not taken.

1 Exploratory Data Analysis

Here is the original plot of COVID-19 dataset.



Intuitively, the number of cases remains on the ground before it rises monotonously and drastically after day 50, with no seasonality detected.

2 Models Considered

First of all, the data pattern is weird because first half, or more of it, increased extremely slow, but the second half of it increase wildly. Therefore, there is no way to fit a linear model to the data without modifying it in the first place, even exponential factor cannot describe it properly.

By differencing the data, we can see that the the first 35 differenced data points are all 0 or 1, which can be interpreted as only 0 or 1 more people caught COVID-19 in the first 35 days. These data should be discarded because the actual patients in the first 35 days are way more than 1 or 0, but may not be tested due to lack of alertness.**Therefore, our analysis will only cover the latter 31 datapoints.**

Recall that

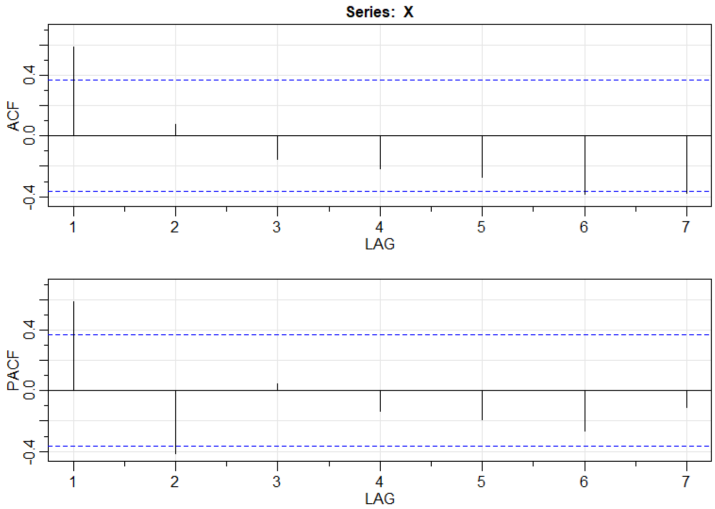
$$y_t = m_t + s_t + X_t$$

where m_t is the deterministic trend, s_t the seasonality, and X_t the white noise.

For the deterministic trend, the parametric model is suitable.All our individual models will use parametric model to fit the trend. As there is no apparent seasonality, we will neglect this part.(Hope the seasonality never occurs!) The major differences of our individual model are the tools we use to deal with the residuals. Haoyuan uses ARMA model and Ziyuan uses SARIMA model. Finally, differencing will be implemented as an alternative approach.

2.1 Haoyuan's Model

First let's look at the ACF and PACF of the residual:

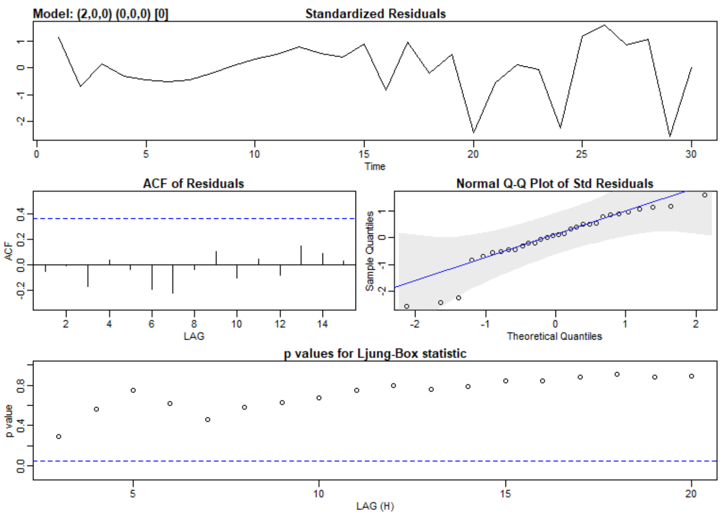


The deterministic function used in my model is:

$$f(t) = a + bt^2 + ct^3 + dt^4$$

Comment on trend model: This model fits the modified data quite well in an acceptable complexity.

Although there is a cutoff in the ACF at lag 1, the spikes at lag 6 and lag 7 indicate that it is not appropriate to use MA model here. Meanwhile,there is a cutoff in PACF at lag 2 with no significant spike occurring afterwards,thus AR(2) may be suitable.(It also performs best in trial and errors.) Here's the sarima() output:



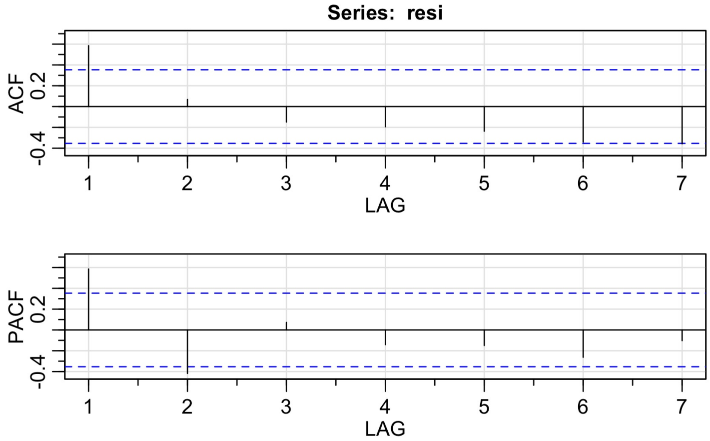
There is no spike in ACF which means that the new residual is likely to be white noise. In the meantime, the p values are reasonably high. Therefore I would say this a good fit of the data, given that AR(2) is a model with relatively lower complexity. AIC will be discussed in Section 3.

2.2 Ziyuan's Model

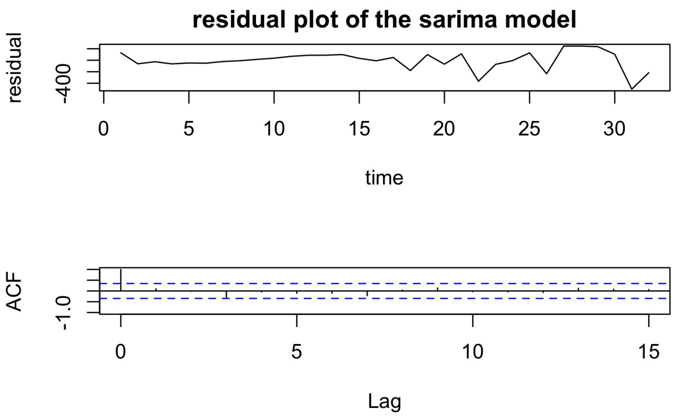
I fit the latter 31 data points with model:

$$f(t) = a + bt + ct^2 + dt^3 + et^4$$

Here's the ACF and PACF plot of the residual:

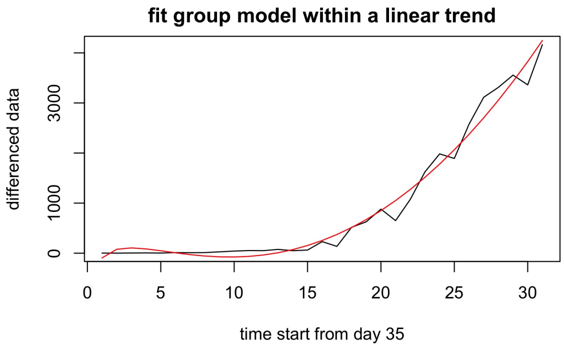


There is one spike at the residual's ACF plot and 2 spikes at the residual's PACF plot. Intuitively, I consider a ARMA(2,1) model to fit the residual. Also, the residual has an obvious seasonality. However, possibly due to the reason that I have only 31 datapoints, the residual can be fitted well using ARMA with a lot different choices of p and q. Therefore, I choose to use cross validation here to select the best model. My trial are based on ARMA(1,1) ,ARMA(1,0), ARMA(2,0) and ARMA(2,1), given a seasonality of 4, 8 ,16. Finally, I choose SARIMA(p = 1, d = 0, q = 1, S = 8, P = 1, Q = 1, D = 0) as my model.

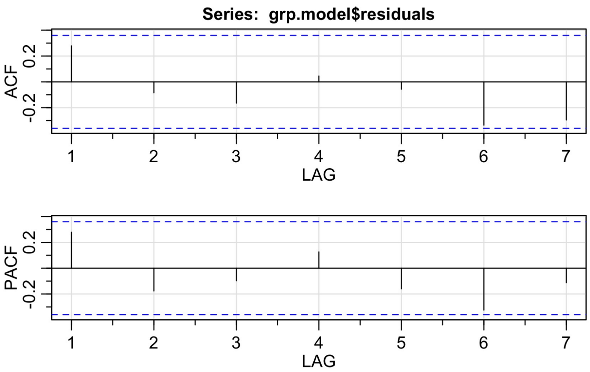


2.3 The Group Model

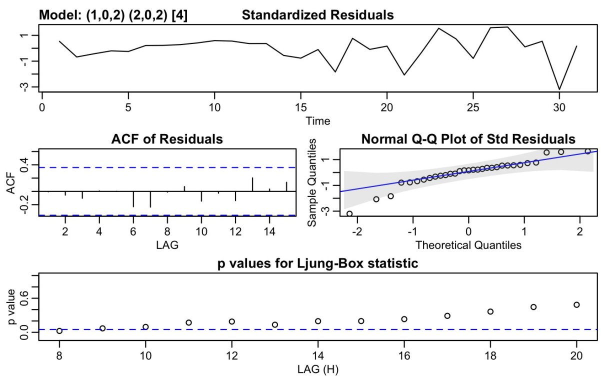
It is obvious that the predictions of the data are strongly dominated by the linear model we use. In other words, the ARMA model we try to fit the residual is not indispensable. In fact, we can prove it by simply obsevering the prediction data coming from the linear model and the ARMA model. The linear model has its flaw, as it fit the data well only when the data has a well deterministic trend. But in this dataset, the trend we are predicting is of rapid growth. The linear trend may be misleading, even it fit the current data well, if the trend itself has even a minor error. We can hedge this risk by soaring the dominance of the residual model. Therefore, we consider to difference the data. We difference the data points from day 35.



The differenced data does not show a smoothed trend as before, which means after fitting the linear model on it, the model use to fit the residual data can also be significant. By then, the ACF and PACF of the residual is pretty nice already.



But we still want to fit an ARMA model by using cross validation. The best model from around 20 trials is SARIMA(p = 1, d = 0, q = 2, S = 4, P = 2, Q = 2, D = 0).



3 Model Comparison and Selection

The table below illustrates some comparison between models. Since AIC, BIC and AICc give exactly the same result, we'll use AIC and RMSE as major criteria to evaluate them.

Model Name	Description	AIC	RMSE
Haoyuan	Linear trend and AR(2)	13.58	946
Ziyuan	Linear trend and SARIMA(1,0,1,1,0,1,8)	13.84	730
Group	Differencing and SARIMA(1,0,2,2,0,2,4)	13.53	487

Table 1: The RMSE comes from the cross-validation.

From the table, we found that the group model, the Differencing with SARIMA model, is dominant under both AIC and RMSE criteria. Therefore, we choose our group model as the best model.

4 Results

This section will present results including parameter estimations and predictions.

First, for SARIMA(p=1,d=0,q=2,P=2,D=0,Q=2,S=4), whose format is

$$\Phi(B^4)\phi(B)X_t = \Theta(B^4)\theta(B)W_t$$

4.1 Estimation of model parameters

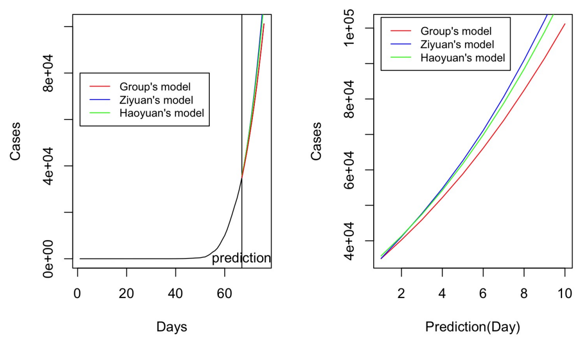
From sarima() function, R gives the estimates:

Parameter	Estimate	(s.e)
ϕ_1	-0.4046	(0.7491)
θ_1	0.8088	(0.7210)
θ_2	0.3287	(0.3199)
Φ_1	-1.0005	(0.4211)
Φ_2	-0.3940	(0.3122)
Θ_1	1.3381	(0.4092)
Θ_2	0.9999	(0.4382)
μ	-0.0012	(50.5291)
σ^2	18594	—

Table 2: These are our parameter estimates and corresponding standard errors for the SARIMA model, where μ represents the mean value of X_t and σ the variance of W_t

4.2 Prediction

Now for the prediction! These predictions are under the assumption that no extra government restriction is implemented(lockdown, social-distancing,etc).Visualization is the best way to remind everyone how disastrous it will be if precautionary actions are out of place.



5 Appendix

5.1 Haoyuan's model-R code

```
library(astsa)
covid = read.csv('C:/Users/80562/Desktop/Time Series Project/covid.csv')
plot.ts(covid$Count)
time = covid$Day
d = diff(covid$Count)
plot(d,type = 'l')
plot(covid$Day,covid$Count,type='l',xlab = 'Days count',ylab = 'Number of Cases',main = 'Confirmed cases in Timeseria')
d_eff = time[35:66]
count_eff = covid$Count[35:66]
plot(covid$Day[35:66],covid$Count[35:66],type='l',xlab = 'Effective days count',ylab = 'Number of Cases',main = 'Data starts from day 35')
model = lm(count_eff~ I(d_eff^2)+I(d_eff^3)+I(d_eff^4))
lines(model$fitted.values, col='red')

X = model$residuals
plot(X,type='l')
acf2(X)

model1 = sarima(X,p=0,d=0,q=1,P=0,D=0,Q=0,S=0)
model2 = sarima(X,p=1,d=0,q=0,P=0,D=0,Q=0,S=0)
model3 = sarima(X,p=1,d=0,q=1,P=0,D=0,Q=0,S=0)
model4 = sarima(X,p=2,d=0,q=0,P=0,D=0,Q=0,S=0)
model5 = sarima(X,p=1,d=0,q=2,P=0,D=0,Q=0,S=0)
model6 = sarima(X,p=2,d=0,q=1,P=0,D=0,Q=0,S=0)


X_forcast = sarima.for(X,n.ahead = 10,p=2,d=0,q=1,P=0,D=0,Q=0,S=0)

newcount = rep(0,10)

intcep = model$coefficients[1]
c_1 = model$coefficients[2]
c_2 = model$coefficients[3]
c_3 = model$coefficients[4]
for(i in 1:10){
a[i] = intcep + c_1*((66+i)^2)+c_2*((66+i)^3)+c_3*((66+i)^4)
newcount[i] = a[i]
}

prediction = newcount + X_forcast$se

plot.ts(c(count_eff,newcount))
model7 = ar(X)
```

5.2 Ziyuan's model-R code

```
#load
library(astsa)
covid = read.csv("covid.csv")
plot.ts(covid$Count)

#by differencing, data before day 35 is less informative
d0 = covid$Count
d1 = c(0, diff(covid$Count))
d1
d3 = d1[35:66]

{r}
#using d3 as valid dataset, fit the linear model
time = 1:length(d3)
covid2 = data.frame(time, "Count" = d0[35:66])
lg.covid = log(covid2$Count)
linearM = lm(covid2$Count ~ time + I(time^2) + I(time^3) + I(time
    ^4))
linearM
#linear model plot with fitted curve
plot(time, covid2$Count, type = "l")
lines(linearM$fitted.values,col = "red")
#linear model's residual
plot(linearM$residuals, type = "l")
{r}
#residual analysis
resi = linearM$residuals
plot(resi, type = "l")
acf2(resi)

{r}

#Cross Validation
resi = linearM$residuals
sse = matrix(NA, nrow=3,ncol=10)
for(i in 1:3){

# Split train/test
train = window(resi,start=1,end=24+i)
test = window(resi, start=24+i, end=24 + i + 4)

# Fit
model1 = arima(train,order = c(1,0,1), seasonal = list(order = c
    (1,0,1), period = 4), method = "ML")
model2 = arima(train,order = c(1,0,1), seasonal = list(order = c
    (1,0,1), period = 8), method = "ML")
model3 = arima(train,order = c(1,0,1), seasonal = list(order = c
    (1,0,1), period = 12), method = "ML")
model4 = arima(train,order = c(1,0,1), seasonal = list(order = c
    (1,1,1), period = 8), method = "ML")
model5 = arima(train,order = c(1,1,0), seasonal = list(order = c
    (0,1,1), period = 8), method = "ML")
model6 = arima(train,order = c(1,1,1), seasonal = list(order = c
    (1,1,0), period = 8), method = "ML")
model7 = arima(train,order = c(1,0,1), seasonal = list(order = c
    (1,0,1), period = 0), method = "ML")
```

```
model8 = arima(train,order = c(1,1,0), seasonal = list(order = c(0,0,1), period = 0), method = "ML")
model9 = arima(train,order = c(1,1,1), seasonal = list(order = c(1,0,0), period = 0), method = "ML")
model10 = arima(train,order = c(1,0,1), seasonal = list(order = c(1,1,0), period = 0), method = "ML")



#sarima(resi, p = 1, d = 0, q = 0, P=0,D=0,Q=0,S=0)

#predic model
preM1 = predict(model1, n.ahead = 5)
preM2 = predict(model2, n.ahead = 5)
preM3 = predict(model3, n.ahead = 5)
preM4 = predict(model4, n.ahead = 5)
preM5 = predict(model5, n.ahead = 5)
preM6 = predict(model6, n.ahead = 5)
preM7 = predict(model7, n.ahead = 5)
preM8 = predict(model8, n.ahead = 5)
preM9 = predict(model9, n.ahead = 5)
preM10 = predict(model10, n.ahead = 5)
# Test
sse[i,1] = sum((test - preM1$pred)^2)
sse[i,2] = sum((test - preM2$pred)^2)
sse[i,3] = sum((test - preM3$pred)^2)
sse[i,4] = sum((test - preM4$pred)^2)
sse[i,5] = sum((test - preM5$pred)^2)
sse[i,6] = sum((test - preM6$pred)^2)
sse[i,7] = sum((test - preM7$pred)^2)
sse[i,8] = sum((test - preM8$pred)^2)
sse[i,9] = sum((test - preM9$pred)^2)
sse[i,10] = sum((test - preM10$pred)^2)

}

apply(sse,2,mean)

{r}
#Analysis the selected model
a = arima(resi,order = c(1,0,1), seasonal = list(order = c(1,0,1), period = 8), method = "ML")
par(mfrow = c(2, 1))
plot(a$residuals, main = "residual plot of the sarima model", ylab = "residual", xlab = "time")
acf(a$residuals, ylim =c(-1,1), main = "")
#pacf(a$residuals, ylim =c(-1,1), main = "")
{r}
#fcast
par(mfrow = c(1,1))
model2 = arima(linearM$residuals,order = c(1,0,1), seasonal = list(order = c(1,0,1), period = 8), method = "ML")
resiFcast = predict(model2, n.ahead = 10)
{r}
#Calculated the forecast result
intcep = linearM$coefficients[1]
co1 = linearM$coefficients[2]
co2 = linearM$coefficients[3]
```

```r
co3 = linearM$coefficients[4]
co4 = linearM$coefficients[5]
#summary(linearM)
newpt = rep(0, 10)
for(i in 1:10) {
a = intcep + (32 + i)*co1 + ((32 + i)^2)*co2 + ((32 + i)^3)*co3 +
    ((32 + i)^4)*co4
newpt[i] = a
}

resiFcast$pred
{r}
#plot the prediction with the 95% CI
newpt = newpt + resiFcast$pred

confi = confint(linearM)

upperError =c(0,10)
for(i in 1:10) {
a = confi[1,2] + (32 + i)*confi[2,2] + ((32 + i)^2)*confi[3,2] +
    ((32 + i)^3)*confi[4,2] + ((32 + i)^4)*confi[5,2]
upperError[i] = a
}
upperError = upperError + 2*resiFcast$se

lowerError =c(0,10)
for(i in 1:10) {
a = confi[1,1] + (32 + i)*confi[2,1] + ((32 + i)^2)*confi[3,1] +
    ((32 + i)^3)*confi[4,1] + ((32 + i)^4)*confi[5,1]
lowerError[i] = a
}
lowerError = lowerError - 2 * resiFcast$se
time2 = 1:(length(covid2$time) + 10)
{r}
plot(time2, c(covid2$Count, newpt), type = "l", ylim = c(-200000,
    200000))
points(33:42, newpt, col = "red", type = "l")
lines(33:42, upperError, col = "blue")
lines(33:42, lowerError, col = "blue")

plot(time2, c(covid2$Count, newpt), type = "l", ylim = c(-200000,
    200000))
lines(33:42, newpt, col = "red")
lines(33:42,newpt - 2*resiFcast$se, col = "blue")
lines(33:42, newpt + 2*resiFcast$se, col = "blue")
```

5.3 Group model-R code(including cross-validation test)

```
#start group model
time = 1:(length(d3) - 1)
diff.data = diff(covid2$Count)
grp.model = lm(diff.data~ time + I(time^2) + I(time^3) + log(time))
summary(grp.model)
plot(diff.data, type = "l", main = "fit group model within a linear trend", xlab = "time start from day 35", ylab = "differenced data")
lines(grp.model$fitted.values, col = "red")
plot(grp.model$residuals, type = "l")#, ylim = c(-1,1))
acf2(grp.model$residuals)
{r}
#group model forecast
intcep = grp.model$coefficients[1]
co1 = grp.model$coefficients[2]
co2 = grp.model$coefficients[3]
co3 = grp.model$coefficients[4]
co4 = grp.model$coefficients[5]
#co5 = grp.model$coefficients[6]
gp.newpt = rep(0, 10)
for(i in 1:10) {
a = intcep + (31 + i)*co1 + ((31 + i)^2)*co2 + ((31 + i)^3)*co3 + (log(31 + i))*co4 #+ co5 * exp(32+i)
gp.newpt[i] = a
}

plot(grp.model$fitted.values, type = "l", xlim = c(0, 42), ylim = c(0, 10000))
lines(32:41, gp.newpt, col = "red")



{r}

#Cross Validation for group model
resi = grp.model$residuals
sse2 = matrix(NA, nrow=3,ncol=10)
for(i in 1:3){

# Split train/test
train = window(resi,start=1,end=23+i)
test = window(resi, start=23+i, end = 23 + i + 4)

# Fit
model1 = arima(train,order = c(1,0,1), seasonal = list(order = c(1,0,1), period = 4), method = "ML")
model2 = arima(train,order = c(1,1,1), seasonal = list(order = c(0,0,1), period = 8), method = "ML")
model3 = arima(train,order = c(0,0,1), seasonal = list(order = c(1,0,1), period = 4), method = "ML")
model4 = arima(train,order = c(1,0,0), seasonal = list(order = c(0,1,1), period = 8), method = "ML")
model5 = arima(train,order = c(1,1,0), seasonal = list(order = c(1,1,0), period = 8), method = "ML")
```

```
model6 = arima(train,order = c(1,1,1), seasonal = list(order = c
    (1,1,0), period = 8), method = "ML")
model7 = arima(train,order = c(1,0,1), seasonal = list(order = c
    (1,0,1), period = 0), method = "ML")
model8 = arima(train,order = c(1,1,0), seasonal = list(order = c
    (0,0,1), period = 0), method = "ML")
model9 = arima(train,order = c(1,0,1), seasonal = list(order = c
    (1,0,0), period = 0), method = "ML")
model10 = arima(train,order = c(1,0,2), seasonal = list(order = c
    (2,0,2), period = 4), method = "ML")


#sarima(resi, p = 1, d = 1, q = 0, P=0,D=1,Q=0,S=4)
#sarima(resi, p = 1, d = 0, q = 0, P=0,D=0,Q=0,S=0)

#predic model
preM1 = predict(model1, n.ahead = 5)
preM2 = predict(model2, n.ahead = 5)
preM3 = predict(model3, n.ahead = 5)
preM4 = predict(model4, n.ahead = 5)
preM5 = predict(model5, n.ahead = 5)
preM6 = predict(model6, n.ahead = 5)
preM7 = predict(model7, n.ahead = 5)
preM8 = predict(model8, n.ahead = 5)
preM9 = predict(model9, n.ahead = 5)
preM10 = predict(model10, n.ahead = 5)
# Test
sse2[i,1] = sum((test - preM1$pred)^2)
sse2[i,2] = sum((test - preM2$pred)^2)
sse2[i,3] = sum((test - preM3$pred)^2)
sse2[i,4] = sum((test - preM4$pred)^2)
sse2[i,5] = sum((test - preM5$pred)^2)
sse2[i,6] = sum((test - preM6$pred)^2)
sse2[i,7] = sum((test - preM7$pred)^2)
sse2[i,8] = sum((test - preM8$pred)^2)
sse2[i,9] = sum((test - preM9$pred)^2)
sse2[i,10] = sum((test - preM10$pred)^2)

}

apply(sse2,2,mean)

{r}
#group model analysis
sarima(grp.model$residuals, p = 1, d = 0, q = 2, P = 2, D = 0, Q =
    2, S = 4)
best.model = arima(train,order = c(1,0,2), seasonal = list(order = c
    (2,0,2), period = 4), method = "ML")
bmodel = sarima(resi, p=1,d=0,q=2,P=2, D = 0, Q = 2, S = 4)
grp.predict = predict(best.model, n.ahead = 10)
gp.pd = gp.newpt + grp.predict$pred

plot(c(diff.data, gp.pd), type = "l", xlim = c(0, 42), ylim = c(0,
    13000))
lines(32:41, gp.pd, col = "red")
{r}
#Summary of three model
final.data = cumsum(c(diff.data, gp.pd))
```

```
final.data
#group prediction data
plot(c(covid$Count[1:35],final.data), type = "l", col = "black", xlab = "Days", ylab = "Cases")
abline(v = 67)
#Jack's predition
lines(67:76,newpt, col = "blue")
text(67, 4, "prediction")
LiModel = c(35726.7, 41311.45, 47463.68, 54249.20, 61710.97, 69874.26, 78768.62, 88434.95, 98921.09, 110274.67)
lines(67:76, LiModel, col = "green")
lines(67:76, final.data[32:41], col = "red")
legend(1, 80000, legend=c("Group's model", "Ziyuan's model", "Haoyuan's model"),
col=c("red", "blue", "green"), lty=1:1, cex=0.8)


{r}
#cross validation for the final 3 model
jack.lm = linearM
jack.resi = jack.lm$residuals
jack.arma = arima(jack.resi,order = c(1,0,1), seasonal = list(order = c(1,0,1), period = 8), method = "ML")

time = 1:length(d3)
li.lm = lm(covid2$Count ~ I(time^2) + I(time^3) + I(time^4))
li.resi = li.lm$residuals
li.arma = arima(li.resi,order = c(2,0,0))

group.lm = grp.model
group.resi = group.lm$residuals
group.arma = best.model

AIC.compare = c(jack.arma$aic, li.arma$aic, group.arma$aic)
AIC.compare
sse3 = matrix(NA, nrow=3,ncol=3)
for(i in 1:3){

# Split train/test
jack.train = window(jack.resi,start=1,end=24+i)
li.train = window(li.resi,start=1,end=24+i)
group.train = window(group.resi,start=1,end=23+i)

jack.test = window(jack.resi, start=24+i, end = 24 + i + 4)
li.test = window(li.resi, start=24+i, end = 24 + i + 4)
group.test = window(group.resi, start=23+i, end = 23 + i + 4)

# Fit
model1 = arima(jack.train,order = c(1,0,1), seasonal = list(order = c(1,0,1), period = 8), method = "ML")
model2 = arima(li.train,order = c(2,0,0))
model3 = arima(group.train,order = c(1,0,2), seasonal = list(order = c(2,0,2), period = 4), method = "ML")


#sarima(resi, p = 1, d = 1, q = 0, P=0,D=1,Q=0,S=4)
#sarima(resi, p = 1, d = 0, q = 0, P=0,D=0,Q=0,S=0)

```

```
#predic model
preM1 = predict(model1, n.ahead = 5)
preM2 = predict(model2, n.ahead = 5)
preM3 = predict(model3, n.ahead = 5)


sse3[i,1] = sum((jack.test - preM1$pred)^2)
sse3[i,2] = sum((li.test - preM2$pred)^2)
sse3[i,3] = sum((group.test - preM3$pred)^2)

}

CrossValidation.result = apply(sse3,2,mean)

{r}
#plot the result
barplot.my <-barplot(CrossValidation.result,  xlab = "Average the sum of squares of errors",  col=c(rgb(0.3,0.1,0.4,0.6) , rgb(0.3,0.5,0.4,0.6) , rgb(0.3,0.9,0.4,0.6) ,  rgb(0.3,0.9,0.4,0.6)), names.arg = c(c("Jack","Li","Group" )), )
title(main = "Cross Validation result")

text(barplot.my,0, paste(round(CrossValidation.result), sep="") , cex=1, pos = 3)
lines(AIC.compare)

```