University of Waterloo
Faculty of Mathematics
Department of Statistics and Actuarial Science

ACTSC 445 PROJECT REPORT

ARIMA Process in Forecasting Financial Time Series Data

List of Tables

List of Figures

1. Introduction

Time-series data is one of the most important data types of the financial market data. People show great interest in time-series financial data because a good prediction is able to help to build an arbitrage portfolio and thus bring profit for certainty.

In recent time, various ways have been suggested and adopted to predict the financial time series data, including principal component analysis (PCA), artificial neural networks (ANN), long short-term memory LSTM [3]. For this project, we are focusing on discussion of ARIMA models, which are a general class of models which use differencing to eliminate non-stationary in the data and predict future values based on history [4].

Within the scope of ACTSC445, we have learned stylized facts about the financial time series data and the definition of an ARMA process with GARCH errors. In this project, we expect to make the people acquaint how ARIMA process could be used to predict a financial times series data successfully.

The project's synopsis could be separated into 4 parts which are: Definition, Process of ARIMA Forecasting, a case study about predicting the gold price and the conclusion.

---

2. Definition

In this section, this paper will introduce MA, AR and ARIMA process to whom are interested.

MA(q): Moving Average Suppose $Z_t \sim WN(0, \sigma^2)$

$$X_t = Z_t + \sigma_1 Z_{t-1} + ... + \sigma_q Z_{t-q}$$

From the definition, we can see that MA is a linear combination of the current and last q innovations Z_t.

AR(p): Auto Regression Suppose $Z_t \sim WN(0, \sigma^2)$

$$X_t = Z_t + \phi_1 X_{t-1} + ... + \phi_p X_{t-p}$$

AR uses a linear combination of past values of the variable as the prediction for current value, which is a regression of the variable against itself and thus named as auto-regression.

ARMA(p,q): Auto-regressive moving average process If X_t is stationary and statisfies

$$X_t - \phi_1 X_{t-1} - ... - \phi_p X_{t-p} = Z_t + \sigma_1 Z_{t-1} + ... + \sigma_q Z_{t-q}$$

then is called ARMA(p,q).

ARIMA: Auto-regressive Integrated Moving Average X_t is an ARIMA(p,d,q) process if d-times differencing is a causal ARIMA(p,d,q) process.

After have some primitive sense of what is ARIMA process, we can forward to discuss how to fit a ARIMA model by Box-Jenkins approach.

---

3. Process of ARIMA Forecasting

The approach we want to introduce for ARIMA (or SIRIMA) forecasting is Box-Jenkins approach, in which its process is summaried in the below figure [2].

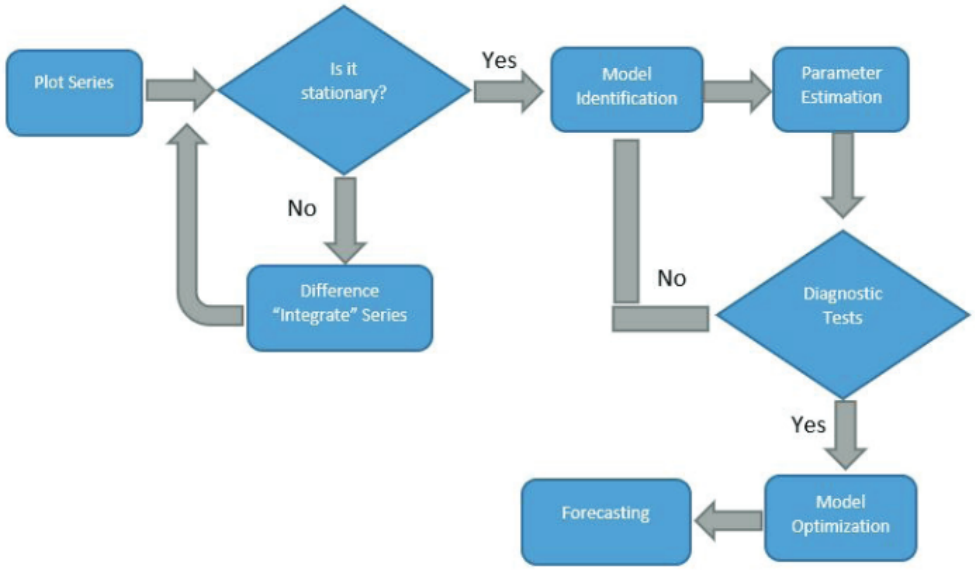

Figure 3.1: Steps of Box-Jenkins Approach

In this project, we want to add one more step to the original Box-Jenkins approach, that is power transform, which aims at removing non-constant variance in the data set and hence getting more randomized residuals after fitting the model [1].

Hence, for now, we have 5 steps in total for the ARIMA model forecasting time series data.

1. **Power transform**
   Suppose the original data is X_t and power is $\alpha \in \mathbb{R}$, power transformed data is

   $$Y_t = \begin{cases} log(X_t), \text{ if } \alpha = -1 \\ (X_T)^{\alpha}, \text{ otherwise} \end{cases}$$

   Our goal is to make sure Y_t has constant variance after applying power transform, which can be tested by Flinger-Kileen test (with null hypothesis of $\sigma_1^2 = ... = \sigma_t^2$). Also, there are many other methods available within the statistic scope. For instance, Box-Cox method mentioned in the literature [1].

2. **Training and Testing Set**
   The training set is usually 80%-90% of the whole data set while the testing set is about 10%-20%. The following transforms/fitting only performed on the training set.

3. **Differencing**
   Differencing is one way to make a non-stationary time series stationary by computing the differences between consecutive observations. It can help stabilise the mean of a time series by removing changes in the level of a time series, and therefore eliminating (or reducing) trend and seasonality.

   - Regular Differencing: used to eliminate trend

   $$\nabla X_t = X_t - X_{t-1}$$

- Seasonal Differencing: used to eliminate seasonality with period k

$$\nabla^k X_t = X_t - X_{t-k}$$

There is *diff()* function in R to help do the differencing.
ACF plot is important signal of using differencing. If the slow decay is observed in the ACF plot, regular differencing need to be applied to remove the trend; if observed seasonal pattern in the ACF plot, seasonal differencing need to be applied to remove the seasonality. For a stationary time series, the ACF will drop to zero relatively quickly. We only do differencing one at a time to avoid over-differencing, which would cause the variance increases exponentially.

4. **Propose p,d,q**
After getting the stationary data by applying differencing, we can look at the ACF and PACF plot and propose model based on the following criteria table.Notice that, d is the order of differencing from the last step.

	ACF	PACF
MA(q)	cuts of after lag q	exponential decay and/or damped sinnnusoid
AR(p)	exponential decay and/or damped sinnnusoid	cuts of after lag p

Table 3.1: Table for Propose p & q

In the real data analysis, it is common that people find several pairs of (p, q) are convictive. Keep all reasonable model and carry it to the next step.

5. **Select best model**
When conducting fit ARIMA model in R, it will automatically generate model diagnostic plot like below plot. If all 4 plots implicates stationary, then we carry the model to the next step.

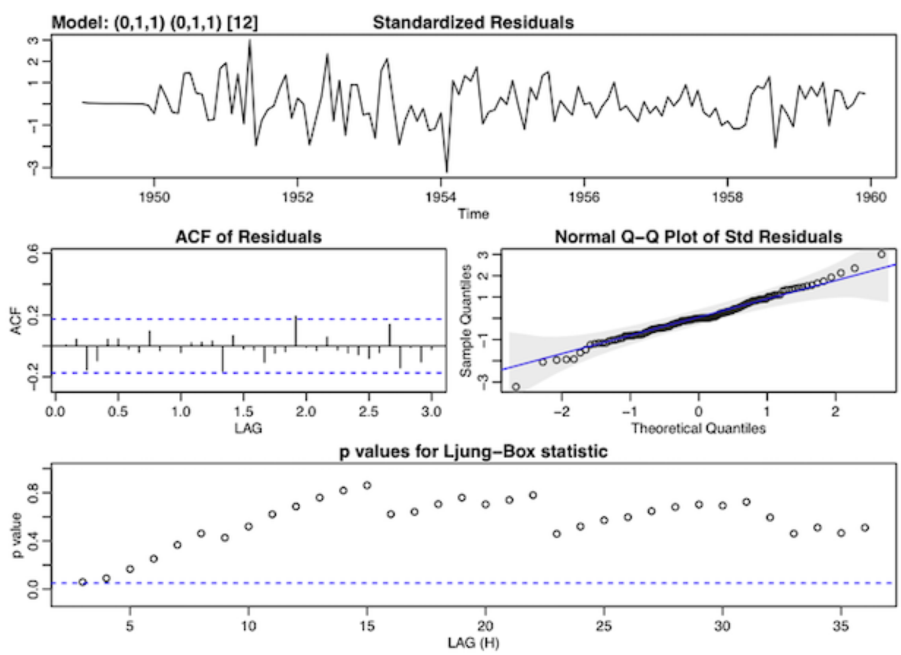

Figure 3.2: Model Diagnostic Plot

After filtering several models that fail the diagnose, a comparison of prediction mean square of error ($MSR_{prediction}$) will be conducted.

$$MSR_{prediction} = \frac{\sum_{y \in \text{Test Set}} (y - \hat{y})^2}{n}, \text{ where } \hat{y} \text{ is the prediction on the test set}$$

A model with minimum $MSR_{prediction}$ will be chosen.

6. **Forecasting**
After choosing the best model, we can applied the model to the future period to generate the forecasting data and its corresponding confidence interval.

---

4. Case Study: Gold Price

4.1. Data Set Description

We find the data from Kaggle, which contains the monthly gold price per gram from January 1979 to July 2021 from 18 different countries. We choose the Switzerland CHF currency as the data we want to analyze, because the price of the gold includes the fluctuations of the currency exchange rate. In order to make sure that the data only related to the nature volatility of gold price, we decided to choose the currency with more stable exchange rate, in which Switzerland is chosen, as that the country's zero-inflation policy, combined with its political independence, makes CHF an extremely powerful and stable currency [5].

Date	Switzerland.CHF.
31-01-1979	379.3
28-02-1979	413.6
30-03-1979	406.2
30-04-1979	420.0
31-05-1979	478.0
29-06-1979	457.7

Table 4.1: First 5 rows of the monthly gold price

Table 4.1 is the first 5 rows of our time-series data. Our goal is to fit a forecasting model on the data and predict the next 2 years' monthly gold price.

4.2. Modelling

4.2.1. Non-constant Variance

From figure 4.1, we can see that our data has trend but no seasonality, since the ACF plot has a slow decay but not signs of periodic patterns or slow decay on seasonal lag. We also used classic decomposition on the data and observe significant trend and insignificant seasonality (see figure A.1 and figure A.2 for details).

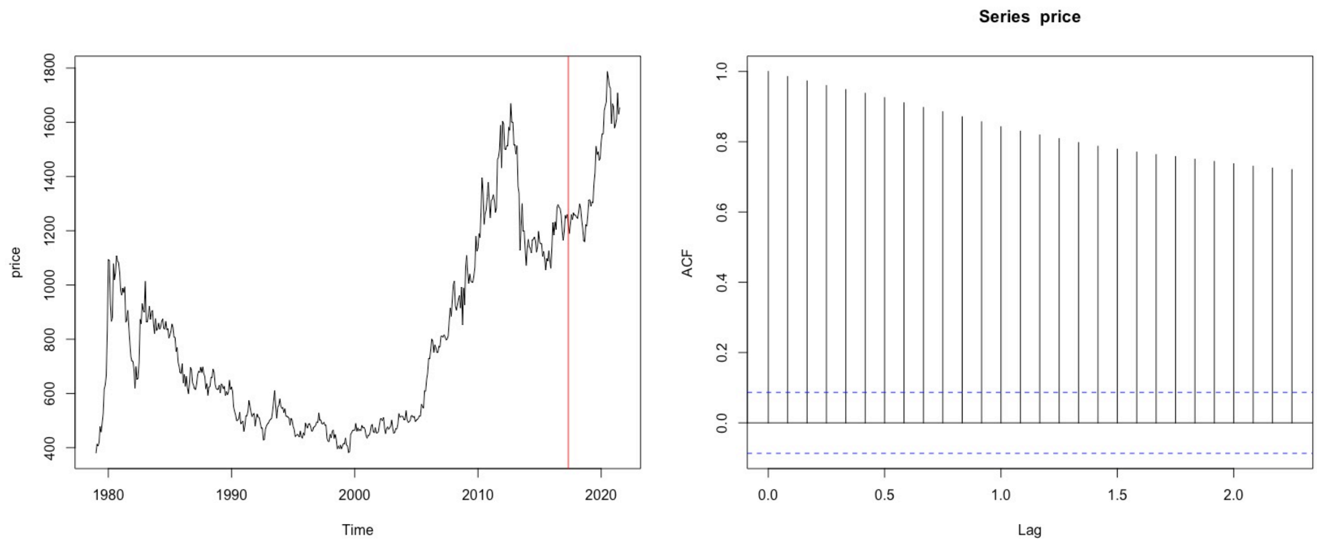

Figure 4.1: Time Series plot and ACF plot

Additionally, we didn't observe any non-constant variance in the plot, but to double check, we performed the Fligner-Killeen test on the dataset, and the results has very small p-value. Hence, we have non-constant variance in our data-set. However, when we try to use power transformation and boxcox method to remove the non-constant variance, neither method is helpful (see Appendix 4.2.1 for the result of F-test). Hence, we decide to move forward to try to fit models, but note that the results might not be valid since the data is non-stationary.

We got the following when performing the Fligner test on the entire data:

```
Fligner-Killeen test of homogeneity of variances
data:  price and seg
Fligner-Killeen:med chi-squared = 197.01, df = 4, p-value < 2.2e-16
```

We want to choose α from $\{-2, -1.5, -1, -0.5, 0, 0.5, 1.5, 2\}$. When we use the Boxcox method, we get the following graph:

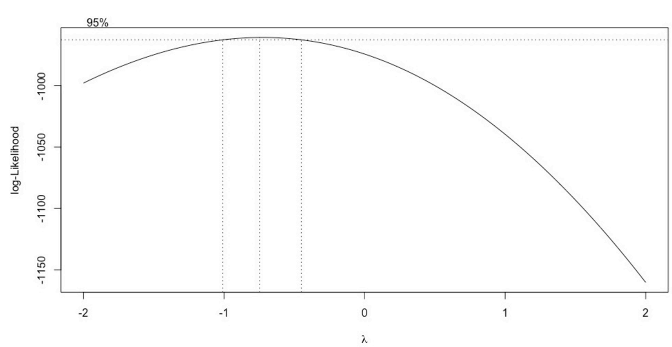

Figure 4.2: Boxcox result

When we again test on the power transformed data with $\alpha = -0.7474747$, we get

```
Fligner-Killeen test of homogeneity of variances
data:  price^bx$x[which.max(bx$y)] and seg
Fligner-Killeen:med chi-squared = 107.15, df = 4, p-value < 2.2e-16
```

The extremely small p-value doesn't improve our results above.

We also compare the fligner test result on each α in the above list to find the optimal α. The p-values from each test is as follows:

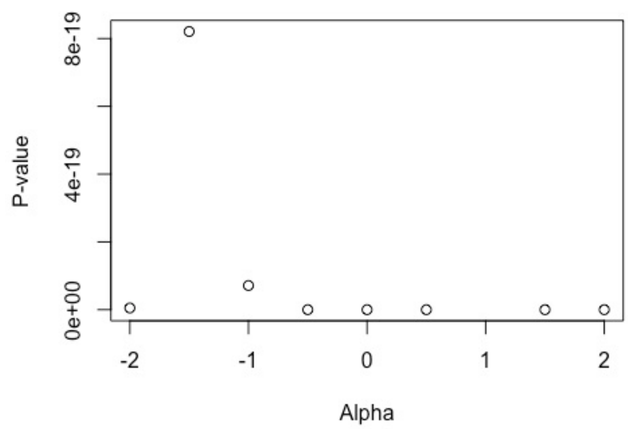

Figure 4.3: Fligner p-values

All the p values are extremely small, which means that any of α doesn't remove non-constant variance in our data. Hence, no more power transform in the data set.

4.2.2. Training and Testing Set

We split the data-set into a training set and a test set, and use the test set MSE to select the best model within each category. The 90% training set is from 1979-01 to 2017-03, and the 10% test set is from 2017-04 to 2021-12 (see the red line in figure 4.1). Our goal is to predict the future 30 months of the gold price (up until 2023-12).

4.2.3. ARIMA Process

From figure 4.1, we see that the data contains significant trend and no seasonality. Hence, we will perform regular differencing to first remove non-stationarity from the data.

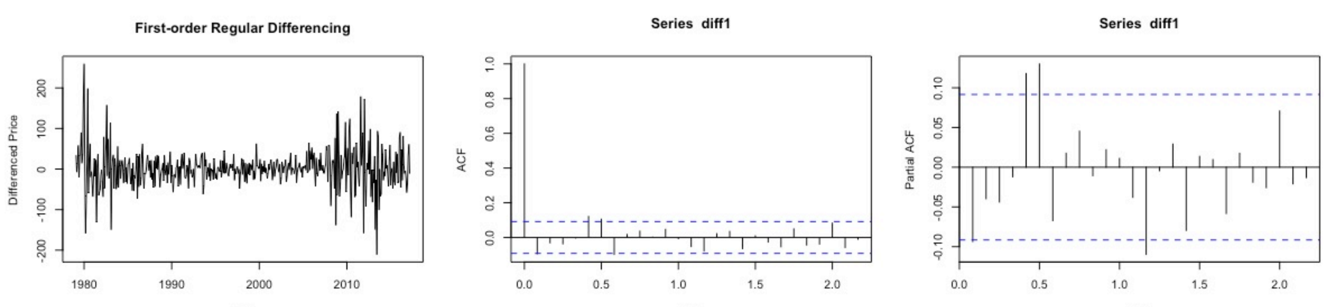

Figure 4.4: First order Regular Differencing

After performing one regular differencing on the data, the trend seems to be removed from the data since there is no linear decay in the ACF plot. To be sure that no more differencing is needed, we performed another differencing, and figure B.1 confirms this conclusion. To avoid over-differencing, we stop here and move on to propose models.

Based on the ACF and PACF plot (4.4), we proposed three ARIMA model with no seasonality:

ARIMA(0,1,1) We observe that the ACF plot cuts off after lag 1, and the PACF plot has a damped sine wave. The spikes in the ACF plot after lag 1 is due to the 95% confidence interval.

ARIMA(1,1,0) We observe that the ACF plot has a damped sine wave, and the PACF plot cuts off after lag 1 (we can see a little spike on lag 1). The spikes in the PACF plot after lag 1 is due to the 95% confidence interval.

ARIMA(0,1,5) We observe that the ACF plot cuts off after lag 5, and the PACF plot has a damped sine wave. The spikes in the ACF plot after lag 5 is due to the 95% confidence interval.

ARIMA(0,1,6) We observe that the ACF plot cuts off after lag 6, and the PACF plot has a damped sine wave. The spikes in the ACF plot after lag 6 is due to the 95% confidence interval.

ARIMA(6,1,0) We observe that the ACF plot has a damped sine wave, and the PACF plot cuts off after lag 6. The spike in the PACF plot after lag 6 is due to the 95% confidence interval.

ARIMA(1,1,1) As above, we observe damped sine wave in both ACF and PACF plots.

ARIMA(,1,) We can also try other ARMA methods with larger p and q values.

4.2.4. Model Selection

Model	MSE_{pred}	AIC	AICc	BIC
ARIMA(0,1,1)	43795.86	10.48568	10.48573	**10.51271**
ARIMA(1,1,0)	**43692.13**	10.48642	10.48648	10.51345
ARIMA(0,1,5)	46688.48	10.47991	10.48032	10.54299
ARIMA(0,1,6)	49255.66	10.47709	10.47764	10.54918
ARIMA(6,1,0)	47231.58	**10.47324**	**10.47378**	10.54532
ARIMA(1,1,1)	44388.76	10.48873	10.48884	10.52477
ARIMA(1,1,2)	44479.14	10.49295	10.49314	10.53800
ARIMA(2,1,1)	44517.89	10.49288	10.49308	10.53794
ARIMA(2,1,2)	44396.06	10.49741	10.49770	10.55148

Table 4.2: Comparison Between Proposed ARIMA Models

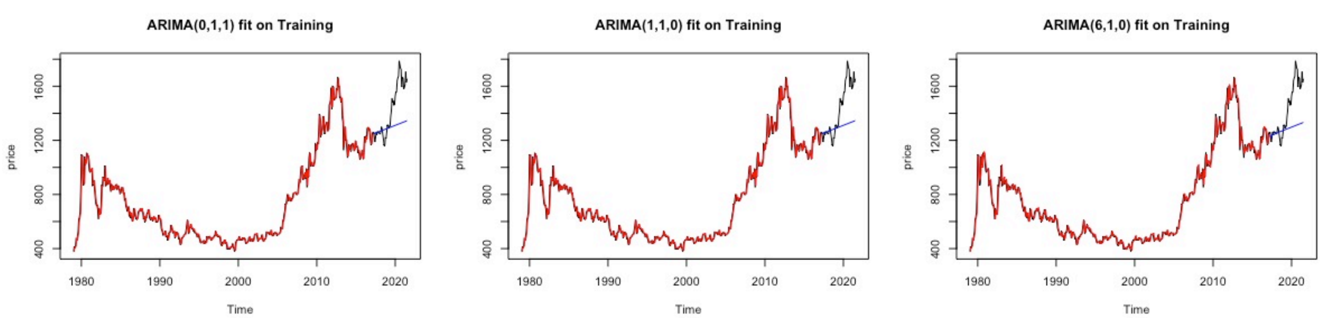

Figure 4.5: ARIMA models fit on Training set

After calculating the MSE, AIC, AICc and BIC, which are shown in table 4.2, we find that ARIMA(6,1,0), ARIMA(0,1,1), and ARIMA(1,1,0) all have better prediction power than the rest proposed models. We include the later two here because ARIMA(6,1,0) have 6 parameters comparing to the other two, and we try to avoid choosing a model with too many parameters. Looking at figure 4.5, we see that all three models have similar fit and prediction, so we will perform residual analysis on all three models to find the best model.

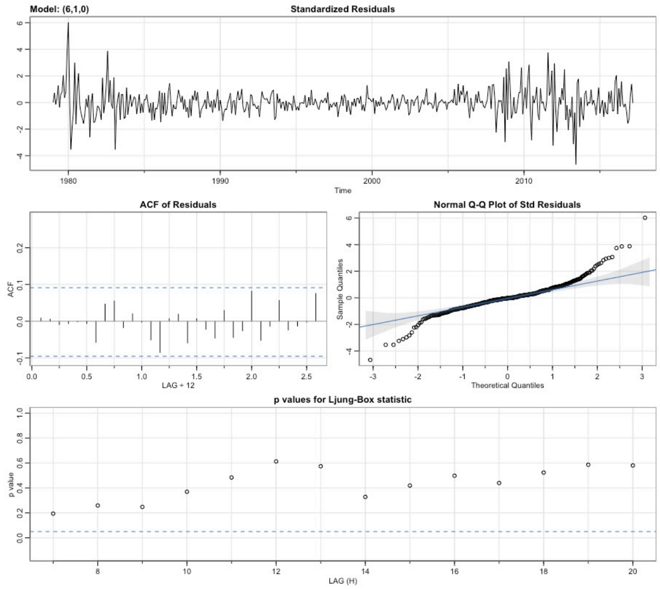

Figure 4.6: ARIMA(6,1,0) on Training set

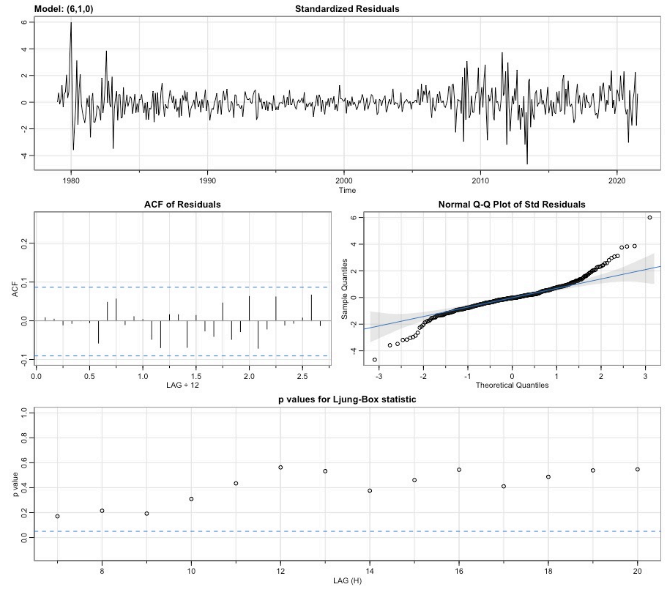

Figure 4.7: Final ARIMA Model Residual Analysis

In Figure 4.6, we can see that the residuals have constant mean, no seasonality, not normal (might due to the variance issue 4.2.1), and the p values look great. Other than non-constant variance, the residuals are pretty much stationary. We also include the residual analysis for ARIMA(1,1,0) (figure B.3) and ARIMA(0,1,1) (figure B.2) in Appendix, and they don't look as good as this model.

After checking the assumptions, we fit the model on the entire data-set and estimate the parameters.

Model	MSE	AIC	AICc	BIC
ARIMA(6,1,0)	2012.242	10.47854	10.47897	10.54496

Table 4.3: Final ARIMA Fit and Prediction Power

We can see that the residuals (figure 4.7) match the model assumptions, except non-constant variance and deviance in normality.

5. Conclusions

5.1. Statistical Conclusions

Now, we will use ARIMA(6,1,0) to predict the next 30 months' gold price.

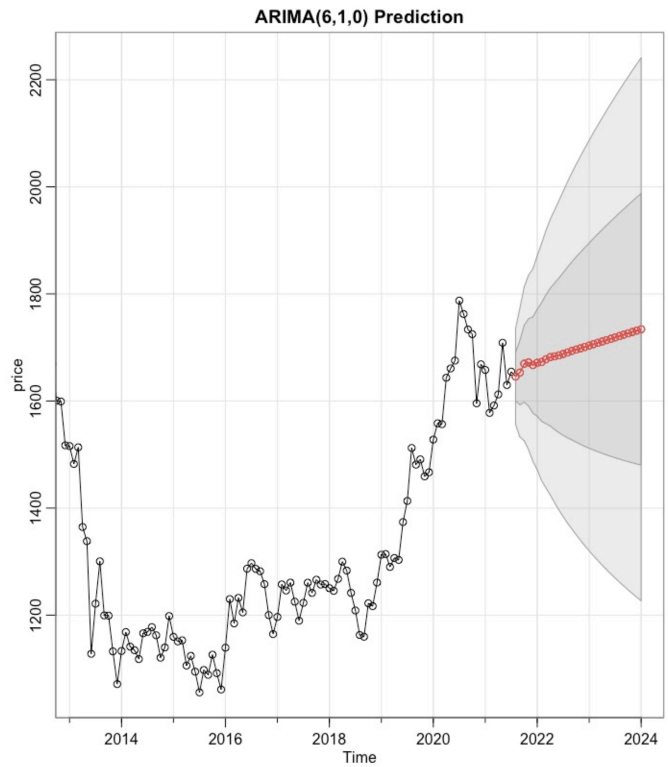

Figure 5.1: ARIMA(6,1,0) Prediction

year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2021								1645.730	1653.005	1669.634	1672.018	1667.086
2022	1671.276	1672.802	1677.870	1681.929	1683.312	1685.172	1687.807	1690.540	1693.578	1696.108	1698.402	1700.877
2023	1703.453	1706.077	1708.667	1711.164	1713.662	1716.197	1718.749	1721.302	1723.837	1726.362	1728.893	1731.431

Table 5.1: ARIMA(6,1,0) Prediction values

However, due to the non-constant variance we have within the residuals, the validity of the prediction interval remain questionable.

5.2. Connect to the Context

In this simulation study, we find that the gold price is very hard to predict using a time-series model due to the complexity and variation existing within this topic. The price is not only tied to time but also many other factors.

As a result, there are several change points in the data-set, e.g. around the 1980s, 2010s, and 2020s. We can guess that the first is due to the stock market recession in 1980-1981, the second is due to the financial crisis in 2007-1008, and the last is due to the COVID-19 pandemic. All these global incidents cause the high variation in the data-set, which is another reason why it's hard to remove the non-constant variance (4.2.1).

Our predictions above are the monthly gold prices from 2021 August to 2023 December. The monthly gold price will have a steady and slowly increasing trend, but the variation in our results exists and is pretty huge.

5.3. Implication related to QRM

The initial goal of the project is to have some naive practice of forecasting a financial time series data by using the ARIMA process, and thus give some implications when calculating risk measurements (e.g. VaR and ES).

The advantages of this model could be seen as:

- Only requires the prior data of a time series to generalize the forecast.
- Performs well on short term forecasts.
- Models non-stationary time series.

However, from the simulation study we can see that for the real data, ARIMA model is not that powerful as we thought of. The possible reasons we propose are as follows:

- Due to the stylized facts of financial times seires data from ACTSC 445, the extreme events comes in cluster. However, the ARIMA model doesn't pick up that.
- The variance of different time intervals fluctuates a lot. Hence, the data is not stationary even after diffencing, which doesn't satisfy the assumption of ARIMA model.

Hence, through this simulation study, we can see a reason that why quantitative risk management introduces Block Maxima Model and Peaks Over Threshold two approximating return distribution methods. The precise distribution, to some extent, is hard to determine and also does not guarantee the prediction power.

A. Additional Figures

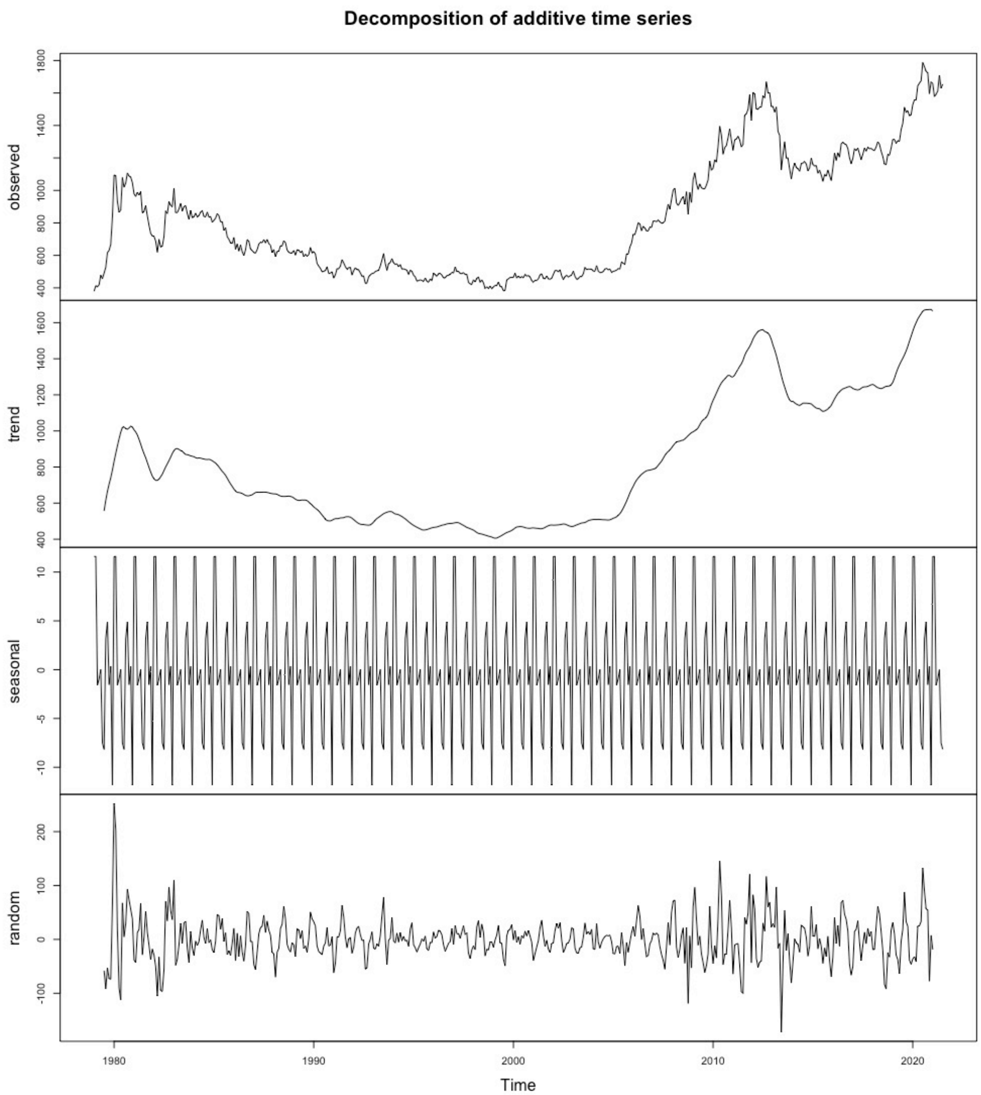

Figure A.1: Additive decomposition plot

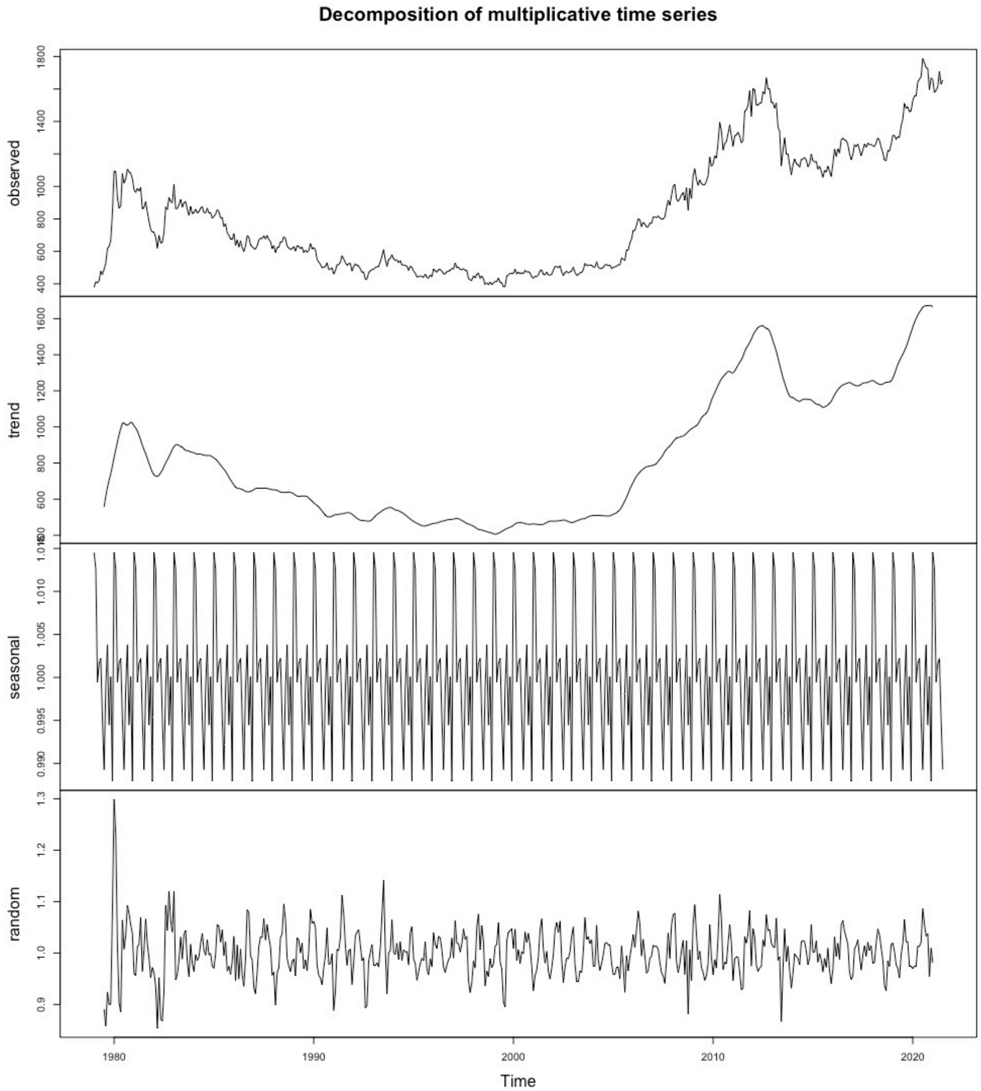

Figure A.2: Multiplicative decomposition plot

B. Supplementary Material

Second-order regular differencing:

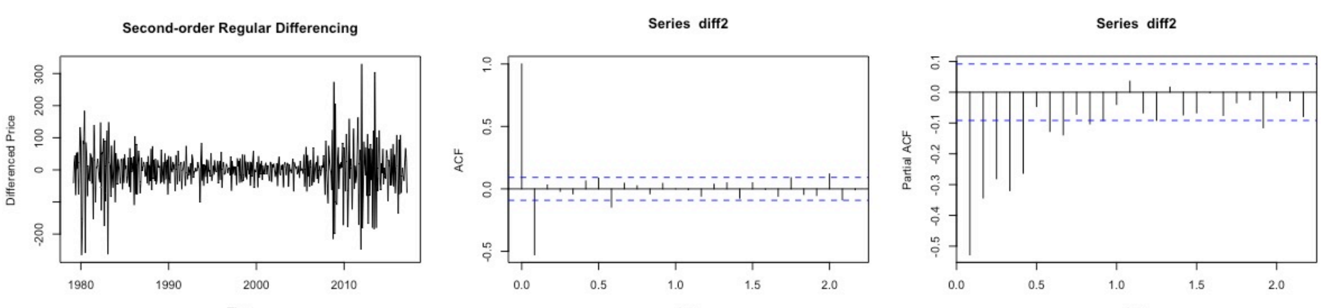

Figure B.1: Second Regular Differencing

Analysis for ARIMA(1,1,0) and ARIMA(0,1,1)

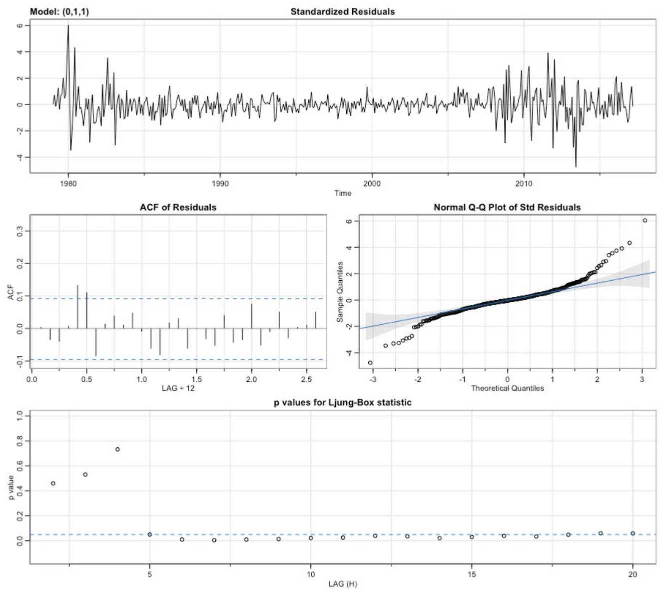

Figure B.2: ARIMA(0,1,1) on Training set

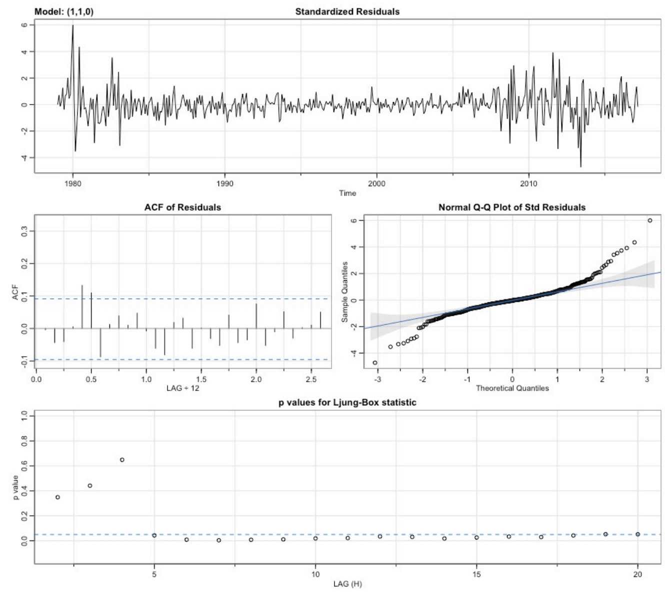

Figure B.3: ARIMA(1,1,0) on Training set

C. R Code

```r
# import data
data <- read.csv("Data_Group2.csv", header=T)[c("Date", "Switzerland.CHF.")]
head(data)
price <- ts(data$Switzerland.CHF., start=c(1979, 1), end=c(2021, 7), frequency=12)
time <- time(price)
boxplot(price)
par(mfrow=c(1,2))
ts.plot(price)
abline(v=2017+1/3, col="red")
acf(price)

decompose.add <- decompose(price, type="additive")
decompose.mult <- decompose(price, type="multiplicative")
plot(decompose.add)
plot(decompose.mult)


# 90% training 10% test set
training.price = window(price, start=c(1979,1), end=c(2017,3), frequency=12)
training.time = window(time, start=c(1979,1), end=c(2017,3), frequency=12)
test.price = window(price, start=c(2017, 4), frequency=12)
test.time = window(time, start=c(2017, 4), frequency=12)
test = data.frame(time=test.time)
training <- data.frame(price=training.price, time=training.time)
df <- data.frame(price=price, time=time)
future.time = time(ts(start=c(2021,7), end=c(2023,12), frequency=12))
future = data.frame(time=future.time)

library(MASS)

## non-constant variance
seg = factor(c(rep(1:4, each=102), rep(5, 103)))
fligner.test(price, seg)

```

```
power.alpha <- c(-2,-1.5,-1,-0.5,0,0.5,1.5,2)
bx = boxcox(lm(training.price~1), lambda=power.alpha)
fligner.test(price^bx$x[which.max(bx$y)], seg)
f.p <- c()
for (a in power.alpha) {
  if (a == 0) {
    y.star = log(price)
  } else {
    y.star = price^a
  }
  f.test = fligner.test(y.star, seg)
  f.p = c(f.p, f.test$p.value)
}
f.p
plot(power.alpha, f.p, xlab="Alpha", ylab="P-value")
abline(h=0.1, col="red")
print(power.alpha[which(f.p>0.1)])

power.alpha.opt = bx$x[which.max(bx$y)]
training.price.star = training.price^power.alpha.opt
test.price.star = test.price^power.alpha.opt
price.star = price^power.alpha.opt
fligner.test(training.price.star, seg)
par(mfrow=c(2,1))
ts.plot(training.price.star)
acf(training.price.star)

## ARIMA

# one regular differencing
diff1 = diff(training.price)
par(mfrow=c(1,3))
plot(diff1, ylab="Differenced Price", main="First-order Regular Differencing")
acf(diff1)
pacf(diff1)
# second regular differencing...?
diff2 = diff(diff1)
par(mfrow=c(1,3))
plot(diff2, ylab="Differenced Price", main="Second-order Regular Differencing")
acf(diff2)
pacf(diff2)

# fit models
library(astsa)
model1 = sarima(training.price, 0,1,1)
model2 = sarima(training.price, 1,1,0)
model3 = sarima(training.price, 0,1,5)
model4 = sarima(training.price, 0,1,6)
model5 = sarima(training.price, 6,1,0)
model6 = sarima(training.price, 1,1,1)
model7 = sarima(training.price, 1,1,2)
model8 = sarima(training.price, 2,1,1)
model9 = sarima(training.price, 2,1,2)

pred_model1 = sarima.for(training.price, p=0, d=1, q=1, n.ahead = 52)
pred_model2 = sarima.for(training.price, p=1, d=1, q=0, n.ahead = 52)
pred_model3 = sarima.for(training.price, 0,1,5, n.ahead = 52)
pred_model4 = sarima.for(training.price, 0,1,6, n.ahead = 52)
pred_model5 = sarima.for(training.price, 6,1,0, n.ahead = 52)
pred_model6 = sarima.for(training.price, p=1, d=1, q=1, n.ahead = 52)
pred_model7 = sarima.for(training.price, 1,1,2, n.ahead = 52)
pred_model8 = sarima.for(training.price, 2,1,1, n.ahead = 52)
```

```
pred_model9 = sarima.for(training.price, 2,1,2, n.ahead = 52)
ds1 = data.frame(
  Model = c("ARIMA(0,1,1)", "ARIMA(1,1,0)", "ARIMA(0,1,5)",
            "ARIMA(0,1,6)", "ARIMA(6,1,0)", "ARIMA(1,1,1)",
            "ARIMA(1,1,2)", "ARIMA(2,1,1)", "ARIMA(2,1,2)"),
  MSE = c(MSE(test.price, pred_model1$pred),MSE(test.price, pred_model2$pred),MSE(test.price, pred_model3$pred),
          MSE(test.price, pred_model4$pred),MSE(test.price, pred_model5$pred),MSE(test.price, pred_model6$pred),
          MSE(test.price, pred_model7$pred),MSE(test.price, pred_model8$pred),MSE(test.price, pred_model9$pred)),
  AIC = c(model1$AIC, model2$AIC, model3$AIC,
          model4$AIC, model5$AIC, model6$AIC,
          model7$AIC, model8$AIC, model9$AIC),
  AICc = c(model1$AICc, model2$AICc, model3$AICc,
           model4$AICc, model5$AICc, model6$AICc,
           model7$AICc, model8$AICc, model9$AICc),
  BIC = c(model1$BIC,model2$BIC,model3$BIC,
          model4$BIC,model5$BIC,model6$BIC,
          model7$BIC,model8$BIC,model9$BIC)
)
ds1

# residual diagnostic of ARIMA(6,1,0)
fit_train5 = training.price - model5$fit$residuals
fit_train1 = training.price - model1$fit$residuals
fit_train2 = training.price - model2$fit$residuals
par(mfrow=c(1,3))
ts.plot(price, main="ARIMA(0,1,1) fit on Training")
points(training.time, fit_train1, type="l", col="red")
points(test.time, pred_model1$pred, type="l", col="blue")
ts.plot(price, main="ARIMA(1,1,0) fit on Training")
points(training.time, fit_train2, type="l", col="red")
points(test.time, pred_model2$pred, type="l", col="blue")
ts.plot(price, main="ARIMA(6,1,0) fit on Training")
points(training.time, fit_train5, type="l", col="red")
points(test.time, pred_model5$pred, type="l", col="blue")

par(mfrow=c(2,2))
plot(fit_train, model5$fit$residuals, xlab="Fitted values", ylab="Residuals")
abline(h=0, col="red")
plot(model5$fit$residuals, ylab="Residuals")
abline(h=0, col="red")
car::qqPlot(model5$fit$residuals, pch=16, col=adjustcolor("black", 0.7),
            xlab = "Theoretical Quantiles (Normal)",
            ylab = "Sample Quantiles (r.hat)",
            main = "Normal Q-Q Plot")
acf(model6$fit$residuals, main="Sample ACF of Residuals")

#assumption test
shapiro.test(model5$fit$residuals)
randtests::difference.sign.test(model5$fit$residuals)

# fit to all data
final_arima = sarima(price, 6,1,0)
fit_val = price - final_arima$fit$residuals

# MSE, AIC, BIC
ds2 = data.frame(
  Model = c("ARIMA(6,1,0)"),
  MSE = c(MSE(price, fit_val)),
  AIC = c(final_arima$AIC),
```

```
  AICc = c(final_arima$AICc),
  BIC = c(final_arima$BIC)
)
ds2

pred = sarima.for(price,6,1,0, n.ahead = 29, main="ARIMA(6,1,0) Prediction")
lower <- pred$pred-1.96*pred$se
upper <- pred$pred+1.96*pred$se
x = c(time(upper) , rev(time(upper)))
y = c(upper , rev(lower))
plot(price, xlim = c(1979,2024), ylim = c(370, 2100))
polygon(x, y, col = "grey" , border =NA)
lines(pred$pred, col = "red")
```

References

[1] Simon Gluzman. "Critical indices and self-similar power transform". In: (June 2021). DOI: 10.13140/RG.2.2.21733.93927.

[2] Tamerlan Mashadihasanli. "Stock Market Price Forecasting Using the Arima Model: an Application to Istanbul, Turkiye". In: *Journal of Economic Policy Researches / İktisat Politikası Araştırmaları Dergisi* 9 (July 2022), pp. 439–454. DOI: 10.26650/JEPR1056771.

[3] Jingyi Shen and M. Shafiq. "Short-term stock market price trend prediction using a comprehensive deep learning system". In: *Journal of Big Data* 7 (Aug. 2020). DOI: 10.1186/s40537-020-00333-6.

[4] Capital One Team. *Understanding ARIMA Models for Machine Learning*. URL: https://www.capitalone.com/tech/machine-learning/understanding-arima-models/.

[5] CFI Team. *Swiss Franc (CHF)*. URL: https://corporatefinanceinstitute.com/resources/foreign-exchange/swiss-franc-chf/.