ARMA-X Analysis Tutorial

Contents

Data

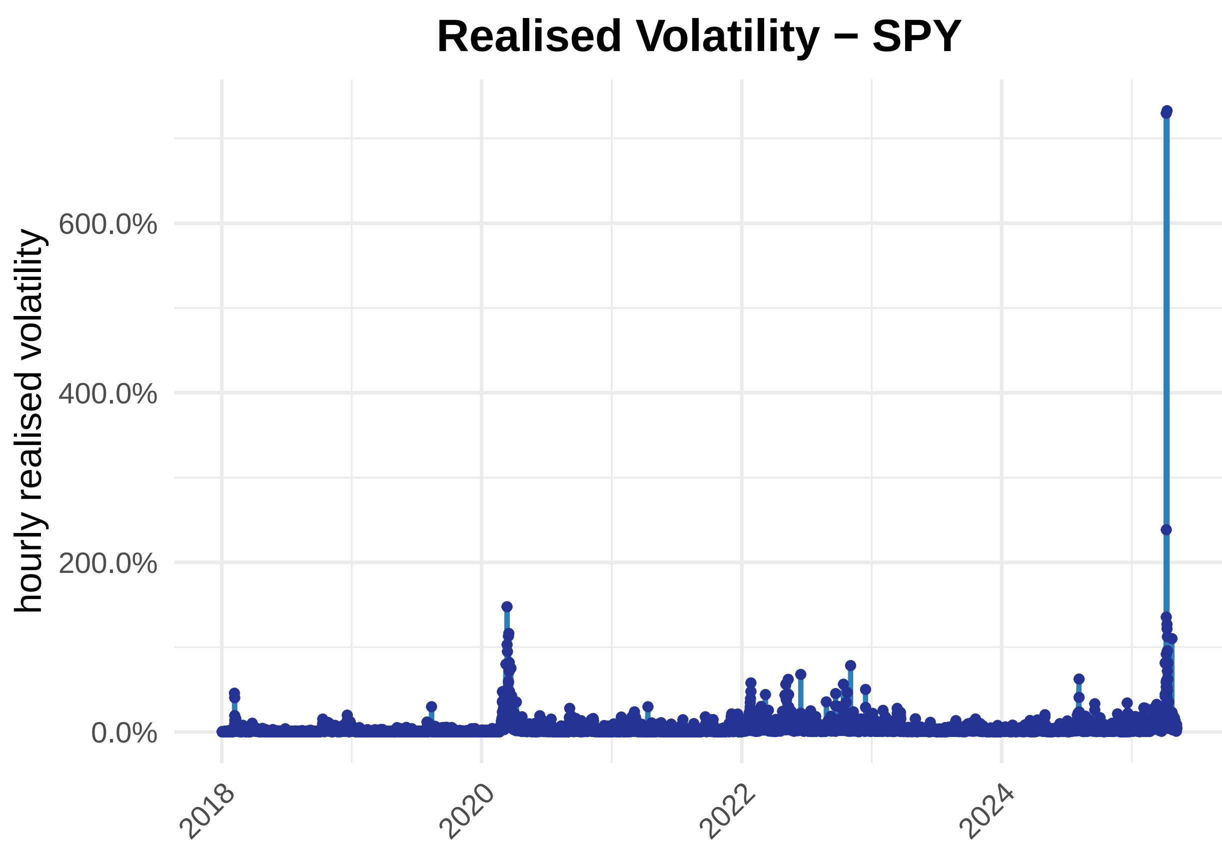
Load Base Data

```r
# 1. Load Political Social Media

#contains posts from Twitter & TruthSocial
social <- read.csv(here("data/mothership", "social.csv"))

social_hourly <- read.csv(here("data/mothership", "socialhourly.csv"))


# 2. Load Financial

#S&P500
SPY <- read.csv(here("data/mothership", "SPY.csv"))

#STOXX50
VGK <- read.csv(here("data/mothership", "VGK.csv"))

#CSI 300 (China)
ASHR <- read.csv(here("data/mothership", "ASHR.CSV"))
```

```r
#make posixct
SPY$timestamp = as.POSIXct(SPY$timestamp,format = "%Y-%m-%d %H:%M:%S")
VGK$timestamp = as.POSIXct(VGK$timestamp,format = "%Y-%m-%d %H:%M:%S")
ASHR$timestamp = as.POSIXct(ASHR$timestamp,format = "%Y-%m-%d %H:%M:%S")
social$timestamp = as.POSIXct(social$timestamp,format = "%Y-%m-%d %H:%M:%S")
social_hourly$timestamp = as.POSIXct(social_hourly$timestamp,format = "%Y-%m-%d %H:%M:%S")
social_hourly$adjusted_time = as.POSIXct(social_hourly$adjusted_time,format = "%Y-%m-%d %H:%M:%S")

#select timeframe
SPY = filter(SPY,between(timestamp, as.Date('2018-01-01'), as.Date('2025-05-07')))
VGK = filter(VGK,between(timestamp, as.Date('2018-01-01'), as.Date('2025-05-07')))
ASHR = filter(ASHR,between(timestamp, as.Date('2018-01-01'), as.Date('2025-05-07')))
social = filter(social,between(timestamp, as.Date('2018-01-01'), as.Date('2025-05-07')))
social_hourly = filter(social_hourly,between(timestamp, as.Date('2018-01-01'), as.Date('2025-05-07')))
```

Volatility

```r
#find hourly volatility
SPY_volatility = dplyr::select(SPY,timestamp,r_vol_h)

#aggregating per hour
SPY_volatility = SPY_volatility %>%
          mutate(timestamp = floor_date(timestamp, unit = "hour")) %>%
          distinct(timestamp, .keep_all = TRUE)

#plot
hvol_plotter(SPY_volatility,breaks="3 month",
             title="Realised Volatility - SPY")
```



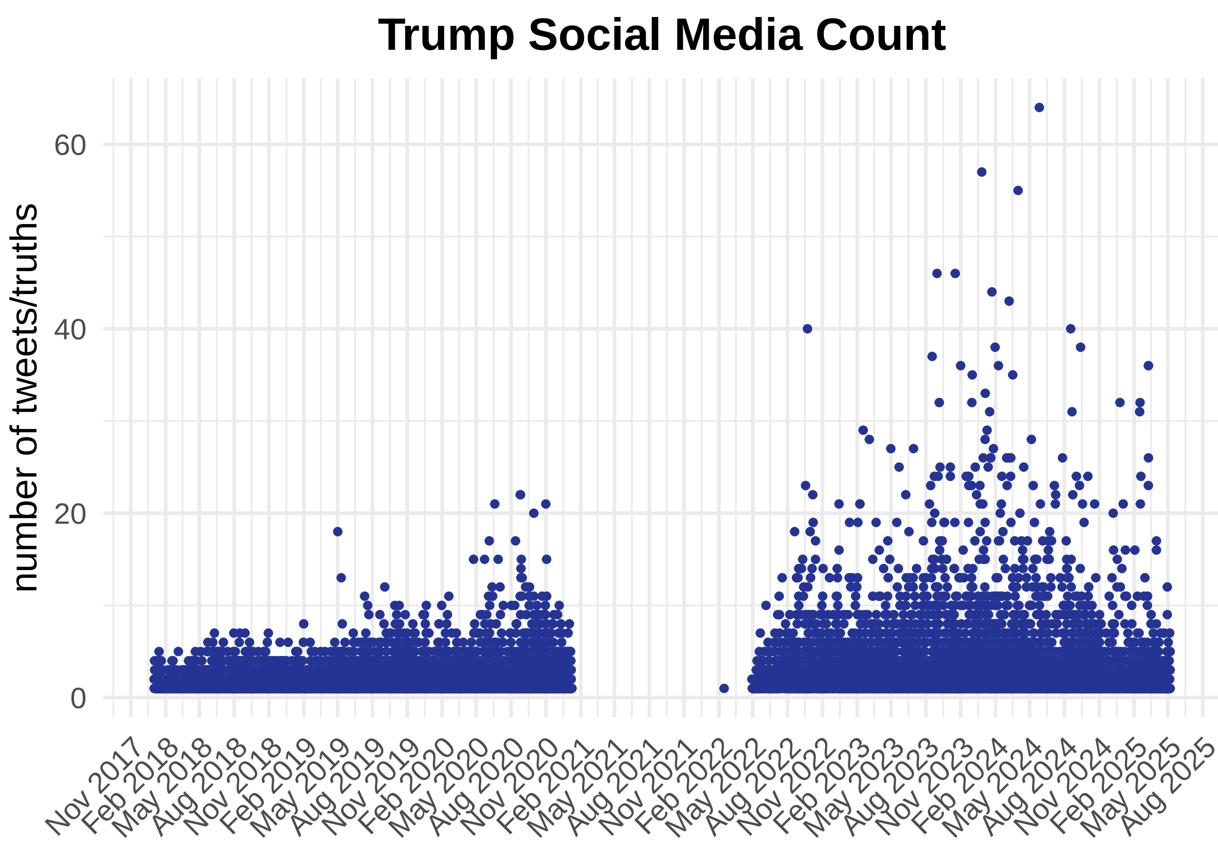

```r
#find hourly volatility
VGK_volatility = dplyr::select(VGK,timestamp,r_vol_h)

#aggregating per hour
VGK_volatility = VGK_volatility %>%
          mutate(timestamp = floor_date(timestamp, unit = "hour")) %>%
          distinct(timestamp, .keep_all = TRUE)
```

```r
#find hourly volatility
ASHR_volatility = dplyr::select(ASHR,timestamp,r_vol_h)

#aggregating per hour
ASHR_volatility = ASHR_volatility %>%
          mutate(timestamp = floor_date(timestamp, unit = "hour")) %>%
          distinct(timestamp, .keep_all = TRUE)
```

Number of Posts

```r
#find count
tweetcount = dplyr::select(social_hourly,timestamp,adjusted_time,N)
```

```r
#for taking count of closed market hours
tweetcount2 <- tweetcount %>%
  group_by(adjusted_time) %>%
  summarise(N = sum(N))

#plot
ggplot(tweetcount, aes(x = timestamp, y = N)) +
    geom_point(color = "#253494", size = 1) +
    scale_x_datetime(date_labels = "%b %Y", date_breaks = "3 month") +
    labs(title = "Trump Social Media Count",
         x = NULL,
         y = "number of tweets/truths") +
    theme_minimal(base_size = 14) +
    theme(axis.text.x = element_text(angle = 45, hjust = 1),
          plot.title = element_text(face = "bold", hjust = 0.5))
```



Dummy for Social Media Post

```r
#find dummy
tweetdummy = dplyr::select(social_hourly,timestamp,adjusted_time,dummy)

#for taking count of closed market hours
tweetdummy2 <- tweetdummy %>%
```

```r
  group_by(adjusted_time) %>%
  summarise(dummy = sum(dummy))
#peculiar interpretation for dummy: if dummy>1 it means that there were x
#out-hours which had tweets in them
```

Number of Tweets Mentioning Tariffs

```r
#find count
tariff = dplyr::select(social_hourly,timestamp,adjusted_time,total_tariff)

#for taking count of closed market hours
tariff2 <- tariff %>%
  group_by(adjusted_time) %>%
  summarise(total_tariff = sum(total_tariff))
```

Number of Tweets Mentioning Trade

```r
#find count
trade = dplyr::select(social_hourly,timestamp,adjusted_time,total_trade)

#for taking count of closed market hours
trade2 <- trade %>%
  group_by(adjusted_time) %>%
  summarise(total_trade = sum(total_trade))
```

Number of Tweets Mentioning China

```r
#find count
china = dplyr::select(social_hourly,timestamp,adjusted_time,total_china)

#for taking count of closed market hours
china2 <- china %>%
  group_by(adjusted_time) %>%
  summarise(total_china = sum(total_china))
```

Proportion of Positive

```r
#find count
positive = dplyr::select(social_hourly,timestamp,adjusted_time,prop_positive)

#how to count outside hours? since proportion?
```

Proportion of Negative

```r
#find count
negative = dplyr::select(social_hourly,timestamp,adjusted_time,prop_negative)
```

Merge

```r
#merge our dependant and independant vars

#case 1: ignore tweets outside trading hours
armax_data = left_join(SPY_volatility, VGK_volatility, by="timestamp")
armax_data = left_join(armax_data, ASHR_volatility, by="timestamp")
armax_data = left_join(armax_data, select(tweetdummy, -adjusted_time), by="timestamp")
armax_data = left_join(armax_data, select(tweetcount, -adjusted_time), by="timestamp")
armax_data = left_join(armax_data, select(tariff, -adjusted_time), by="timestamp")
armax_data = left_join(armax_data, select(trade, -adjusted_time), by="timestamp")
armax_data = left_join(armax_data, select(china, -adjusted_time), by="timestamp")
armax_data = left_join(armax_data, select(positive, -adjusted_time), by="timestamp")
armax_data = left_join(armax_data, select(negative, -adjusted_time), by="timestamp")

rm(armax_data)
#case 2: push tweets made outside market hours to the next open hour
armax_data = left_join(SPY_volatility, VGK_volatility, by="timestamp")
armax_data = left_join(armax_data, ASHR_volatility, by="timestamp")
armax_data <- armax_data %>%
  left_join(tweetdummy2, by = c("timestamp" = "adjusted_time"))
armax_data <- armax_data %>%
  left_join(tweetcount2, by = c("timestamp" = "adjusted_time"))
armax_data <- armax_data %>%
  left_join(tariff2, by = c("timestamp" = "adjusted_time"))
armax_data <- armax_data %>%
  left_join(trade2, by = c("timestamp" = "adjusted_time"))
armax_data <- armax_data %>%
  left_join(china2, by = c("timestamp" = "adjusted_time"))

#rename volatility columns
names(armax_data)[2] <- "SPY_vol"
names(armax_data)[3] <- "VGK_vol"
names(armax_data)[4] <- "ASHR_vol"

#convert NA to zeroes
armax_data$N[is.na(armax_data$N)] = 0
armax_data$dummy[is.na(armax_data$dummy)] = 0
armax_data$total_tariff[is.na(armax_data$total_tariff)] = 0
armax_data$total_trade[is.na(armax_data$total_trade)] = 0
armax_data$total_china[is.na(armax_data$total_china)] = 0
#armax_data$prop_positive[is.na(armax_data$prop_positive)] = 0
#armax_data$prop_negative[is.na(armax_data$prop_negative)] = 0
```

	Model 1
ar1	0.9812^{***}
	(0.0023)
ma1	-0.6787^{***}
	(0.0091)
ma2	-0.2105^{***}
	(0.0108)
ma3	-0.0106
	(0.0100)
ma4	0.0324^{***}
	(0.0088)
intercept	0.0325^{***}
	(0.0061)
dummy_lag_0	0.0013^{***}
	(0.0003)
dummy_lag_1	0.0008^{**}
	(0.0003)
dummy_lag_2	-0.0003
	(0.0003)
dummy_lag_3	-0.0010^{**}
	(0.0003)
dummy_lag_4	-0.0008^{*}
	(0.0003)
dummy_lag_5	-0.0008^{*}
	(0.0003)
dummy_lag_6	0.0000
	(0.0003)
dummy_lag_7	0.0009^{**}
	(0.0003)
AIC	-24011.5255
AICc	-24011.4883
BIC	-23899.5334
Log Likelihood	12020.7628
Num. obs.	12915

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 1: ARMAX Model Results

S&P500 Univariate ARMA-X Models

Tweet Dummy as Exogenous

```
#auto.armax selects the lowest AIC value given r (exogenous variable lags)
res1 = auto.armax(armax_data$SPY_vol,xreg=armax_data$dummy,nb.lags=7,
                latex=T,max.p = 7, max.q = 7, max.d=0)
```

```
#armax enables a custom armax specification with p,q,r
res2 = armax(armax_data$SPY_vol, xreg=armax_data$dummy, nb.lags=2,
                   p=5, q=0, d=0, latex=T)
```

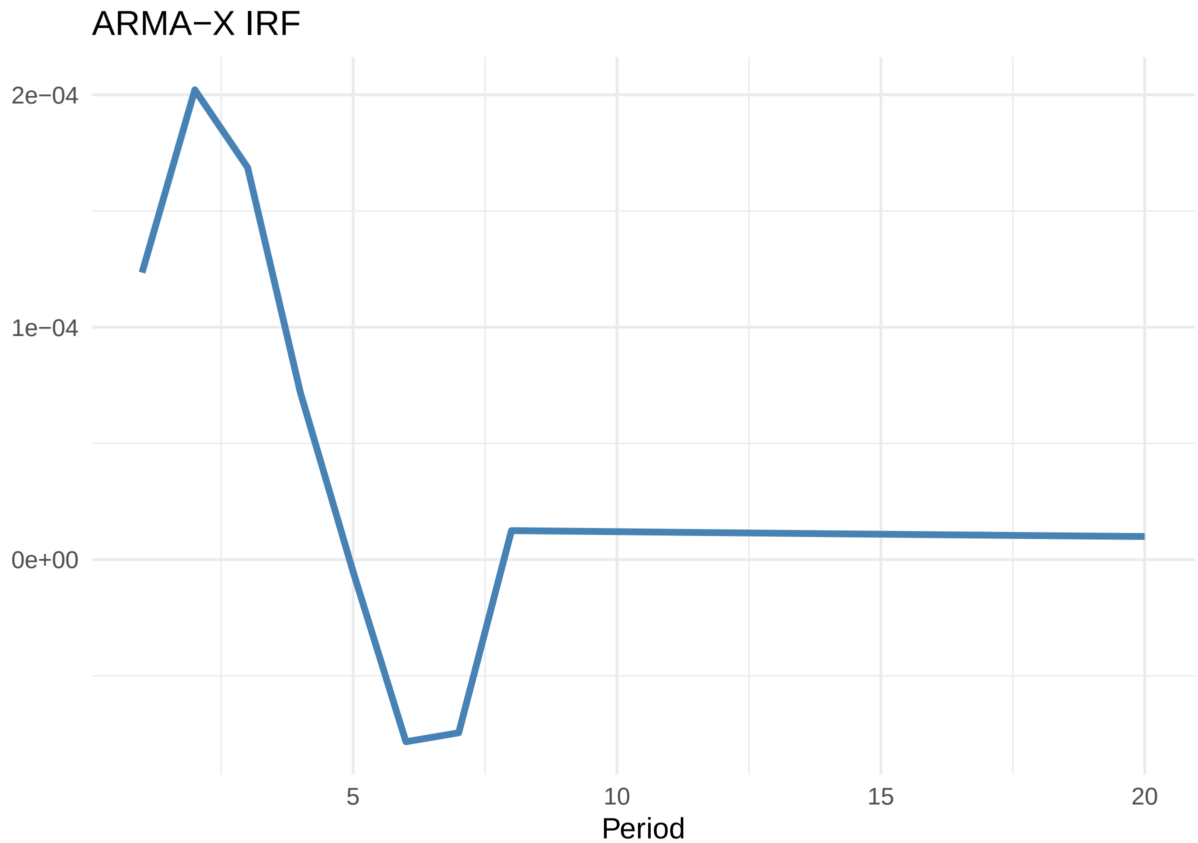
	Model 1
ar1	0.3535^{***}
	(0.0088)
ar2	0.0393^{***}
	(0.0093)
ar3	0.0970^{***}
	(0.0092)
ar4	0.1026^{***}
	(0.0093)
ar5	0.0779^{***}
	(0.0088)
intercept	0.0291^{***}
	(0.0027)
dummy_lag_0	0.0021^{***}
	(0.0003)
dummy_lag_1	0.0013^{***}
	(0.0003)
dummy_lag_2	0.0001
	(0.0003)
AIC	-23390.6502
AICc	-23390.6331
BIC	-23315.9848
Log Likelihood	11705.3251
Num. obs.	12920

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 2: ARMAX Model Results

```r
#auto.armax.r selects the lowest AIC checking all 3 p,q,r values
res3 = auto.armax.r(armax_data$SPY_vol, x=armax_data$dummy,
                    max_p = 7, max_q = 7, max_r = 3, criterion = "AIC", latex=T)
```

```r
#we want to plot the IRFs of these models
nb.periods = 20

irf.plot(res1,nb.periods)
```

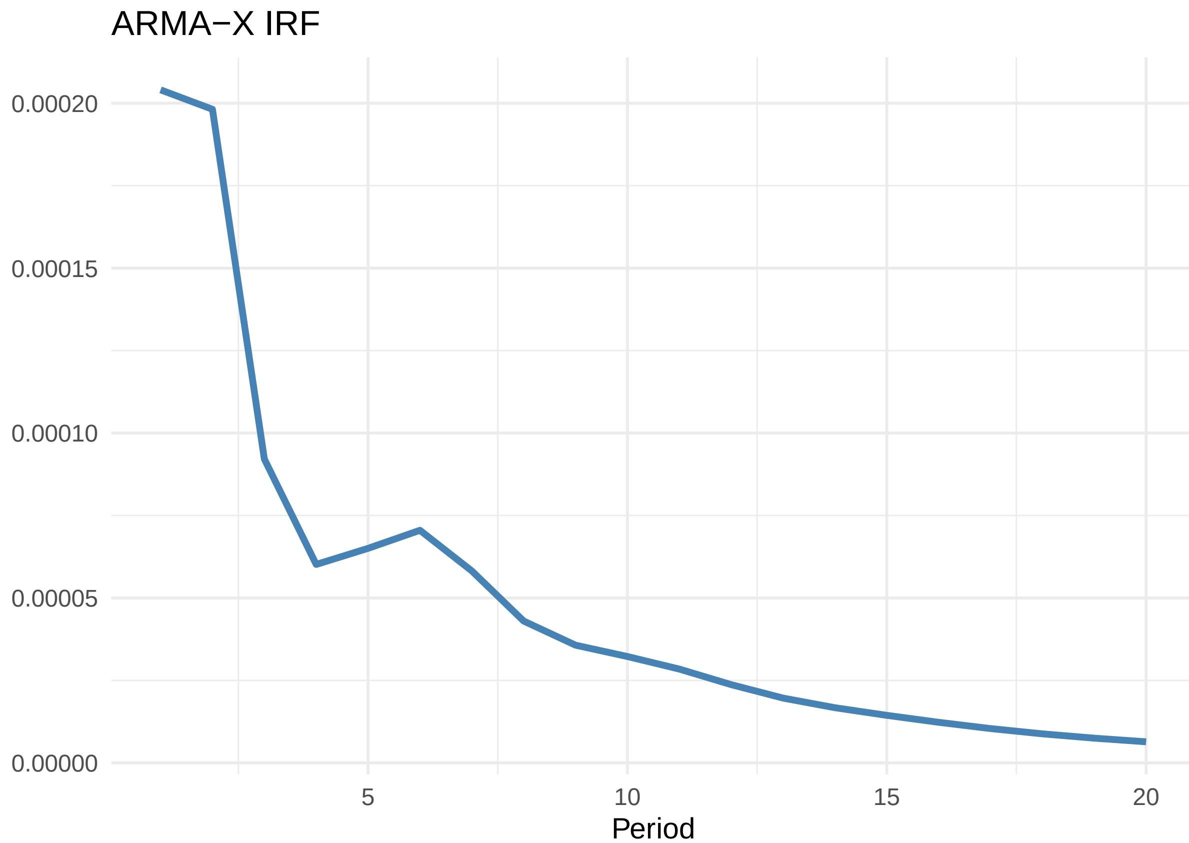



```r
irf.plot(res2,nb.periods)
```

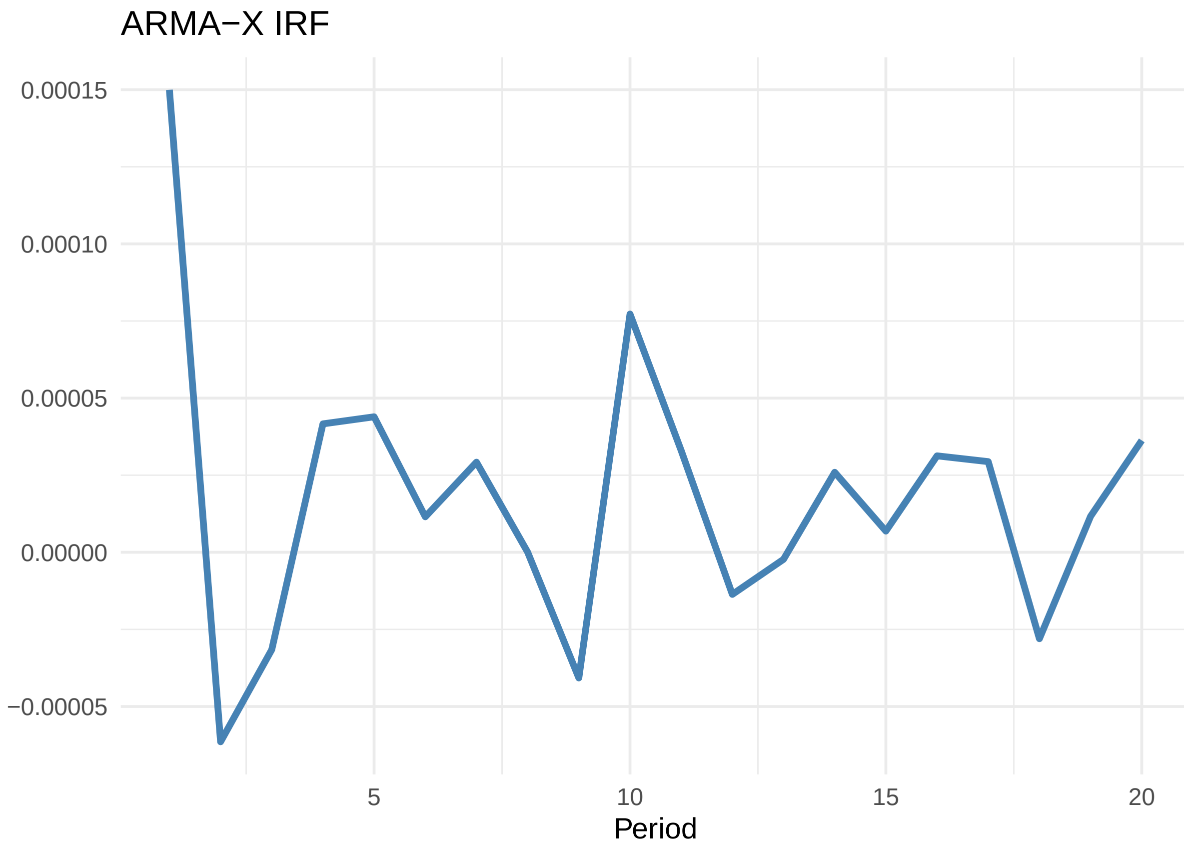
	Model 1
ar1	-0.8979^{***}
	(0.0157)
ar2	-0.5785^{***}
	(0.0172)
ar3	-0.1483^{***}
	(0.0156)
ar4	0.3598^{***}
	(0.0120)
ar5	0.6167^{***}
	(0.0153)
ar6	0.8030^{***}
	(0.0151)
ar7	0.6221^{***}
	(0.0125)
ma1	1.1955^{***}
	(0.0122)
ma2	0.9902^{***}
	(0.0178)
ma3	0.5642^{***}
	(0.0206)
ma4	-0.0238
	(0.0182)
ma5	-0.4950^{***}
	(0.0166)
ma6	-0.8420^{***}
	(0.0139)
ma7	-0.7527^{***}
	(0.0084)
intercept	0.0302^{***}
	(0.0060)
dummy_lag_0	0.0016^{***}
	(0.0003)
dummy_lag_1	0.0008^{**}
	(0.0003)
AIC	-24830.0899
AICc	-24830.0369
BIC	-24695.6910
Log Likelihood	12433.0450
Num. obs.	12921

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 3: ARMAX selected by AIC




```
irf.plot(res3$model,nb.periods)
```



Tweet Count as Exogenous

```
#auto.armax selects the lowest AIC value given r (exogenous variable lags)
res1 = auto.armax(armax_data$SPY_vol,xreg=armax_data$N,nb.lags=7,
                latex=T,max.p = 7, max.q = 7, max.d=0)
```

```
#armax enables a custom armax specification with p,q,r
res2 = armax(armax_data$SPY_vol, xreg=armax_data$N, nb.lags=2,
                   p=5, q=0, d=0, latex=T)
```

```
#auto.armax.r selects the lowest AIC checking all 3 p,q,r values
res3 = auto.armax.r(armax_data$SPY_vol, x=armax_data$N,
                max_p = 7, max_q = 7, max_r = 3, criterion = "AIC", latex=T)
```

```
#we want to plot the IRFs of these models
nb.periods = 20

irf.plot(res1,nb.periods)
```

	Model 1
ar1	0.9812^{***}
	(0.0023)
ma1	-0.6783^{***}
	(0.0091)
ma2	-0.2113^{***}
	(0.0108)
ma3	-0.0110
	(0.0100)
ma4	0.0330^{***}
	(0.0088)
intercept	0.0333^{***}
	(0.0060)
N_lag_0	0.0003^{**}
	(0.0001)
N_lag_1	0.0002
	(0.0001)
N_lag_2	-0.0001
	(0.0001)
N_lag_3	-0.0003^{**}
	(0.0001)
N_lag_4	-0.0003^{*}
	(0.0001)
N_lag_5	-0.0002^{*}
	(0.0001)
N_lag_6	-0.0000
	(0.0001)
N_lag_7	0.0003^{**}
	(0.0001)
AIC	-23991.7721
AICc	-23991.7349
BIC	-23879.7799
Log Likelihood	12010.8861
Num. obs.	12915

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 4: ARMAX Model Results

	Model 1
ar1	0.3547^{***}
	(0.0088)
ar2	0.0386^{***}
	(0.0093)
ar3	0.0968^{***}
	(0.0092)
ar4	0.1020^{***}
	(0.0093)
ar5	0.0778^{***}
	(0.0088)
intercept	0.0302^{***}
	(0.0027)
N_lag_0	0.0005^{***}
	(0.0001)
N_lag_1	0.0003^{***}
	(0.0001)
N_lag_2	0.0000
	(0.0001)
AIC	-23367.7281
AICc	-23367.7111
BIC	-23293.0628
Log Likelihood	11693.8641
Num. obs.	12920

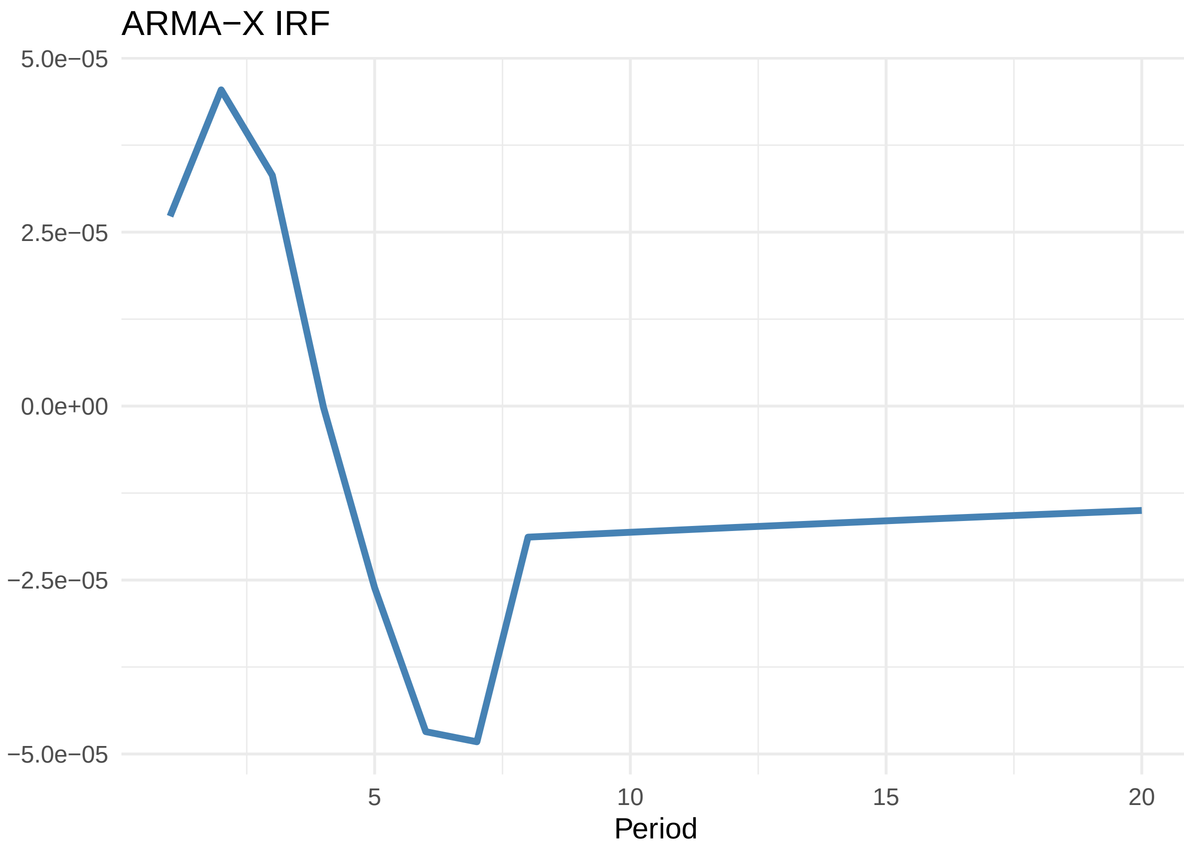
$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 5: ARMAX Model Results

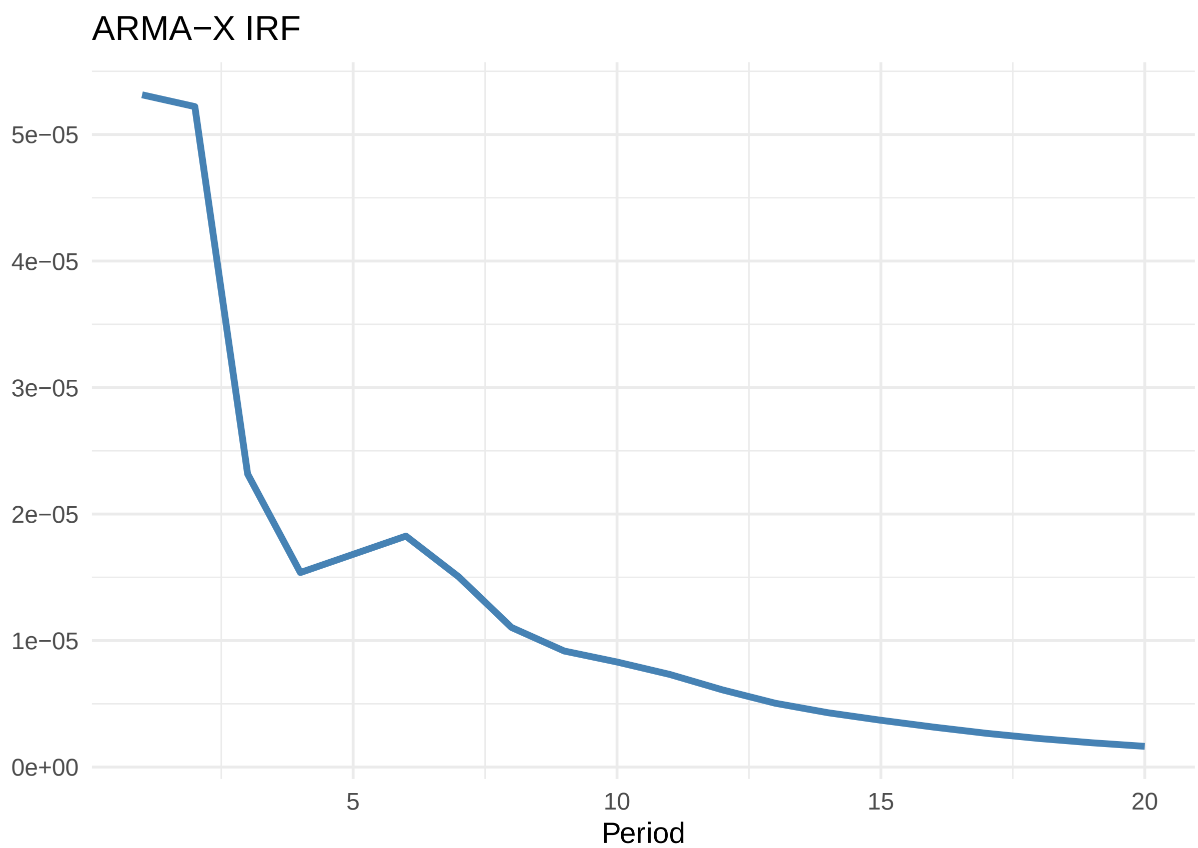
	Model 1
ar1	-0.8773^{***}
	(0.0164)
ar2	-0.5729^{***}
	(0.0187)
ar3	-0.1390^{***}
	(0.0170)
ar4	0.3523^{***}
	(0.0128)
ar5	0.6138^{***}
	(0.0157)
ar6	0.8078^{***}
	(0.0154)
ar7	0.6232^{***}
	(0.0129)
ma1	1.1788^{***}
	(0.0130)
ma2	0.9765^{***}
	(0.0190)
ma3	0.5452^{***}
	(0.0222)
ma4	-0.0321
	(0.0191)
ma5	-0.5012^{***}
	(0.0171)
ma6	-0.8469^{***}
	(0.0141)
ma7	-0.7537^{***}
	(0.0086)
intercept	0.0310^{***}
	(0.0066)
N_lag_0	0.0004^{***}
	(0.0001)
N_lag_1	0.0002^{*}
	(0.0001)
AIC	-24812.3182
AICc	-24812.2652
BIC	-24677.9192
Log Likelihood	12424.1591
Num. obs.	12921

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

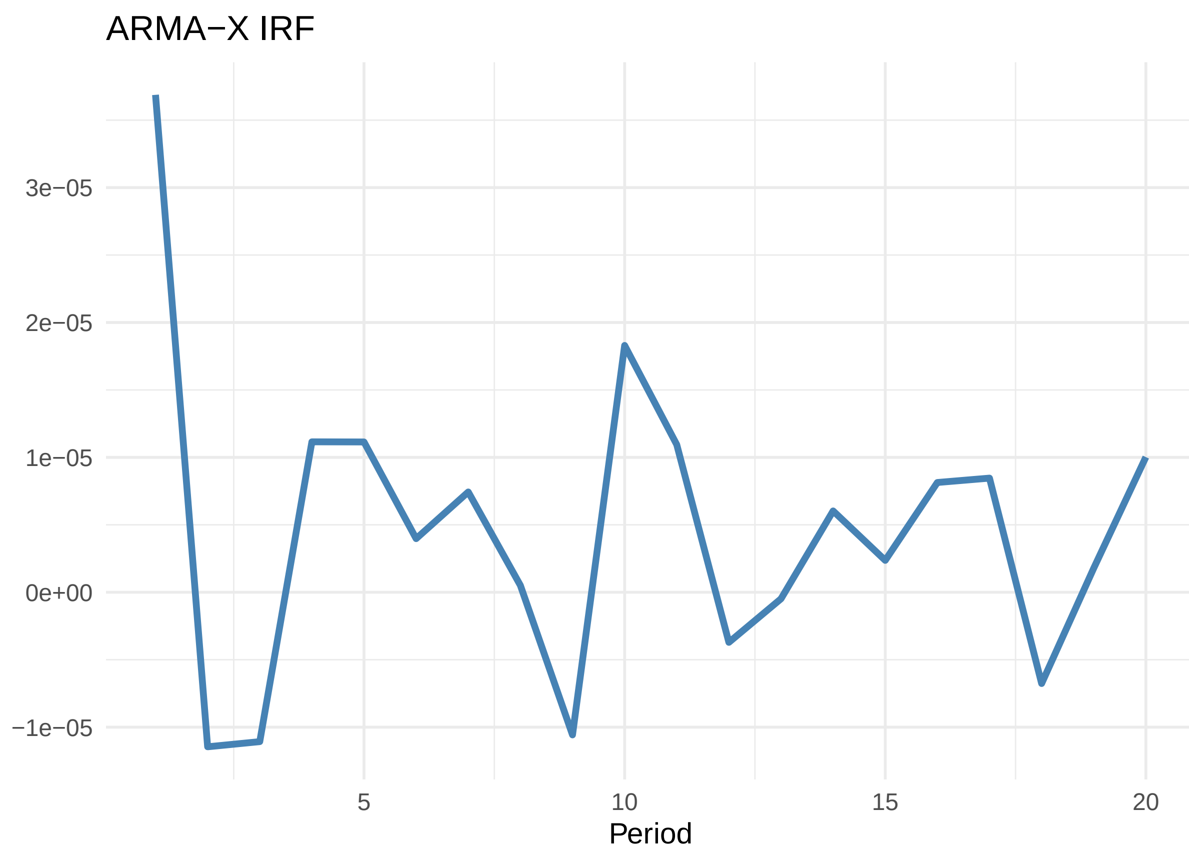
Table 6: ARMAX selected by AIC




```
irf.plot(res2,nb.periods)
```




```r
irf.plot(res3$model,nb.periods)
```



Tariff as Exogenous

```
#auto.armax selects the lowest AIC value given r (exogenous variable lags)
res1 = auto.armax(armax_data$SPY_vol,xreg=armax_data$total_tariff,nb.lags=2,
                latex=T,max.p = 6, max.q = 6, max.d=0)
```

```
#armax enables a custom armax specification with p,q,r
res2 = armax(armax_data$SPY_vol, xreg=armax_data$total_tariff, nb.lags=2,
                   p=5, q=0, d=0, latex=T)
```

```
#auto.armax.r selects the lowest AIC checking all 3 p,q,r values
res3 = auto.armax.r(armax_data$SPY_vol, x=armax_data$total_tariff,
                max_p = 6, max_q = 6, max_r = 6, criterion = "AIC", latex=T)
```

```
#we want to plot the IRFs of these models
nb.periods = 20

irf.plot(res1,nb.periods)
```

	Model 1
ar1	1.6392^{***}
	(0.1237)
ar2	-0.8082^{***}
	(0.1443)
ar3	0.1591^{***}
	(0.0238)
ma1	-1.3393^{***}
	(0.1247)
ma2	0.4068^{***}
	(0.1121)
intercept	0.0314^{***}
	(0.0057)
total_tariff_lag_0	0.0044^{*}
	(0.0018)
total_tariff_lag_1	0.0206^{***}
	(0.0019)
total_tariff_lag_2	0.0113^{***}
	(0.0018)
AIC	-24097.7721
AICc	-24097.7551
BIC	-24023.1068
Log Likelihood	12058.8861
Num. obs.	12920

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 7: ARMAX Model Results

	Model 1
ar1	0.3538***
	(0.0088)
ar2	0.0402***
	(0.0093)
ar3	0.0877***
	(0.0093)
ar4	0.0955***
	(0.0093)
ar5	0.0825***
	(0.0088)
intercept	0.0313***
	(0.0025)
total_tariff_lag_0	0.0047**
	(0.0018)
total_tariff_lag_1	0.0203***
	(0.0019)
total_tariff_lag_2	0.0110***
	(0.0018)
AIC	−23455.1331
AICc	−23455.1161
BIC	−23380.4678
Log Likelihood	11737.5666
Num. obs.	12920

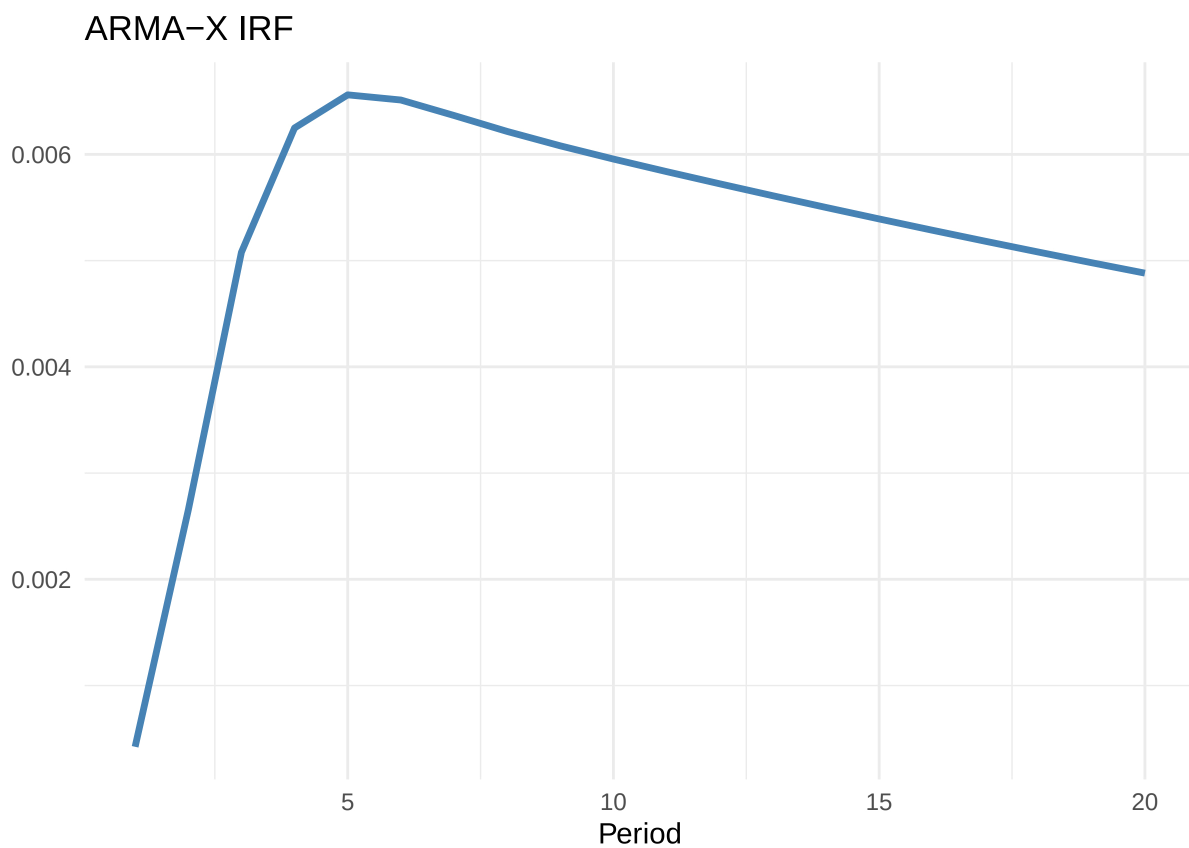
$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 8: ARMAX Model Results

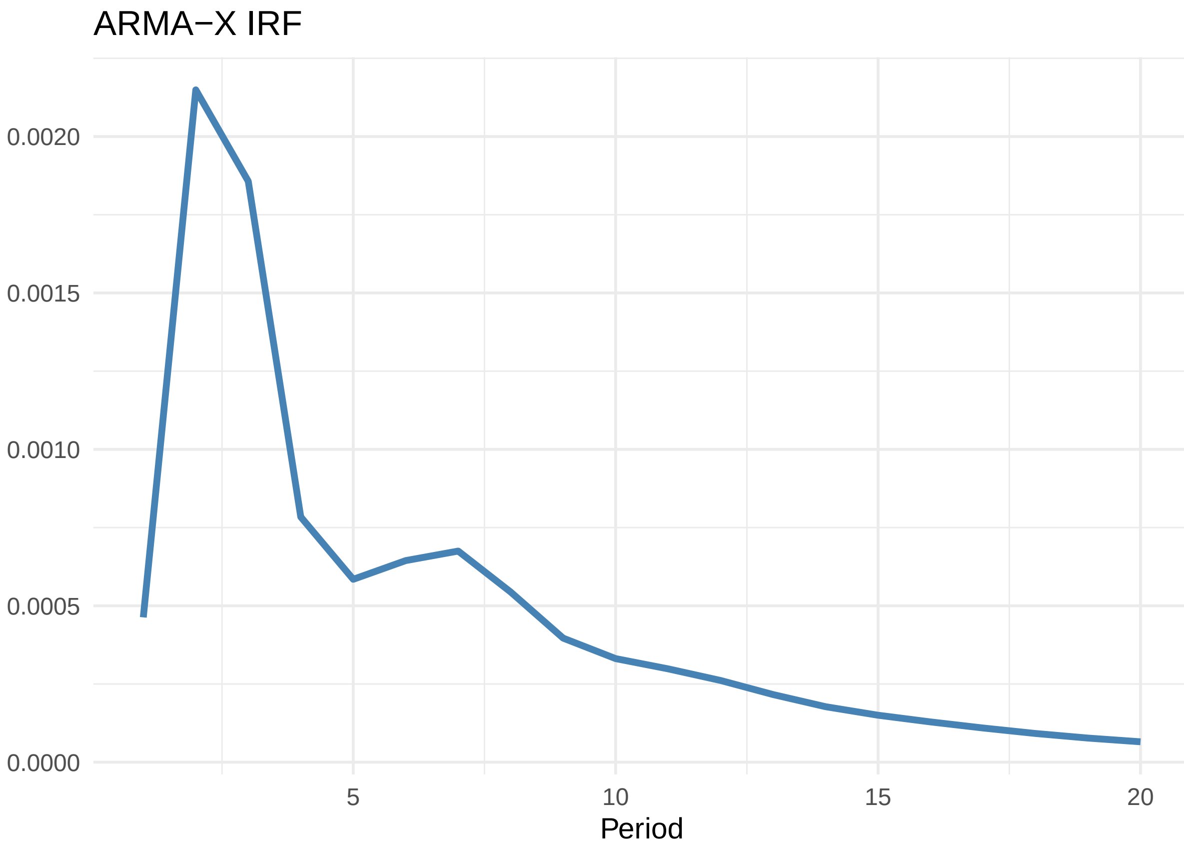
	Model 1
ar1	-0.6539^{***}
	(0.0121)
ar2	0.0271
	(0.0156)
ar3	0.0145
	(0.0109)
ar4	0.1316^{***}
	(0.0139)
ar5	0.6410^{***}
	(0.0134)
ar6	0.6900^{***}
	(0.0094)
ma1	0.9532^{***}
	(0.0092)
ma2	0.2810^{***}
	(0.0160)
ma3	0.1801^{***}
	(0.0147)
ma4	0.0658^{***}
	(0.0124)
ma5	-0.6168^{***}
	(0.0129)
ma6	-0.8019^{***}
	(0.0070)
intercept	0.0315^{***}
	(0.0057)
total_tariff_lag_0	0.0070^{***}
	(0.0016)
total_tariff_lag_1	0.0159^{***}
	(0.0017)
total_tariff_lag_2	0.0083^{***}
	(0.0016)
AIC	-24961.8383
AICc	-24961.7908
BIC	-24834.9072
Log Likelihood	12497.9191
Num. obs.	12920

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

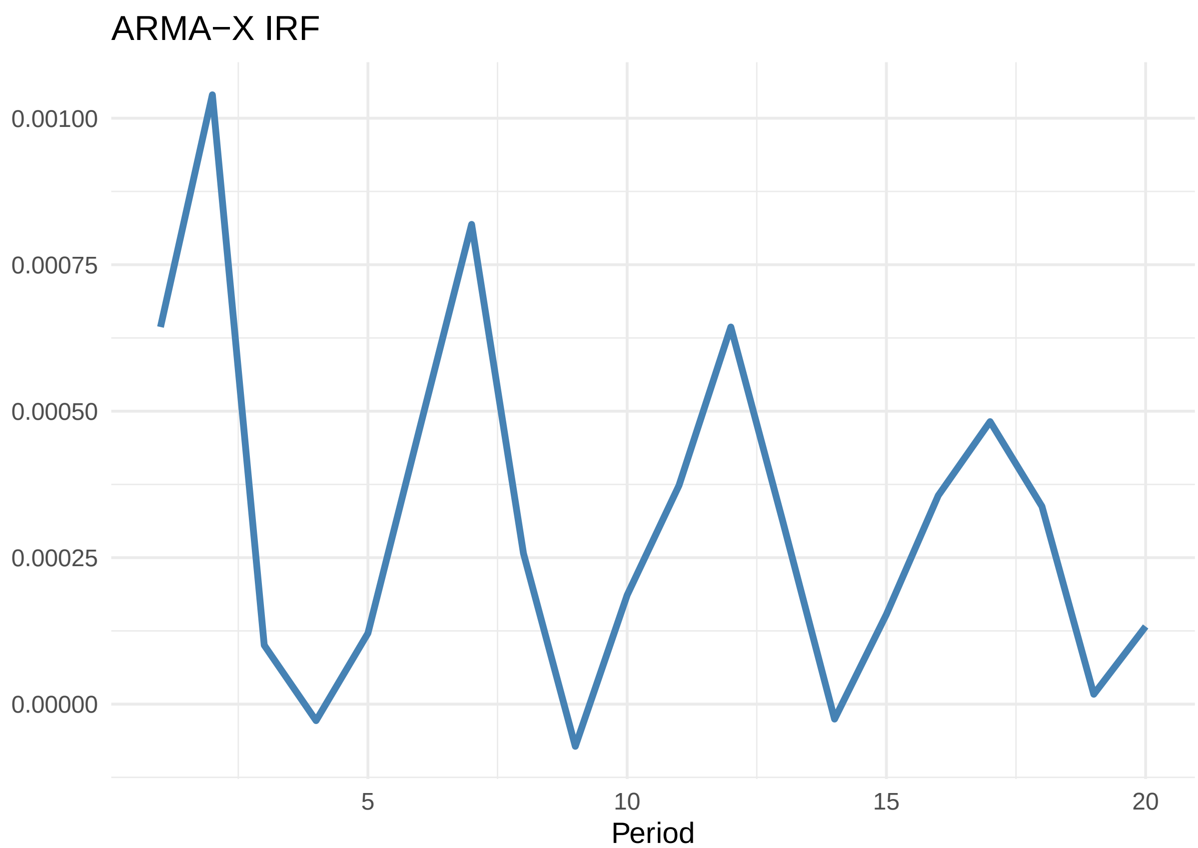
Table 9: ARMAX selected by AIC




```r
irf.plot(res2,nb.periods)
```




```r
irf.plot(res3$model,nb.periods)
```



Interaction Terms

Dummy * Tariff