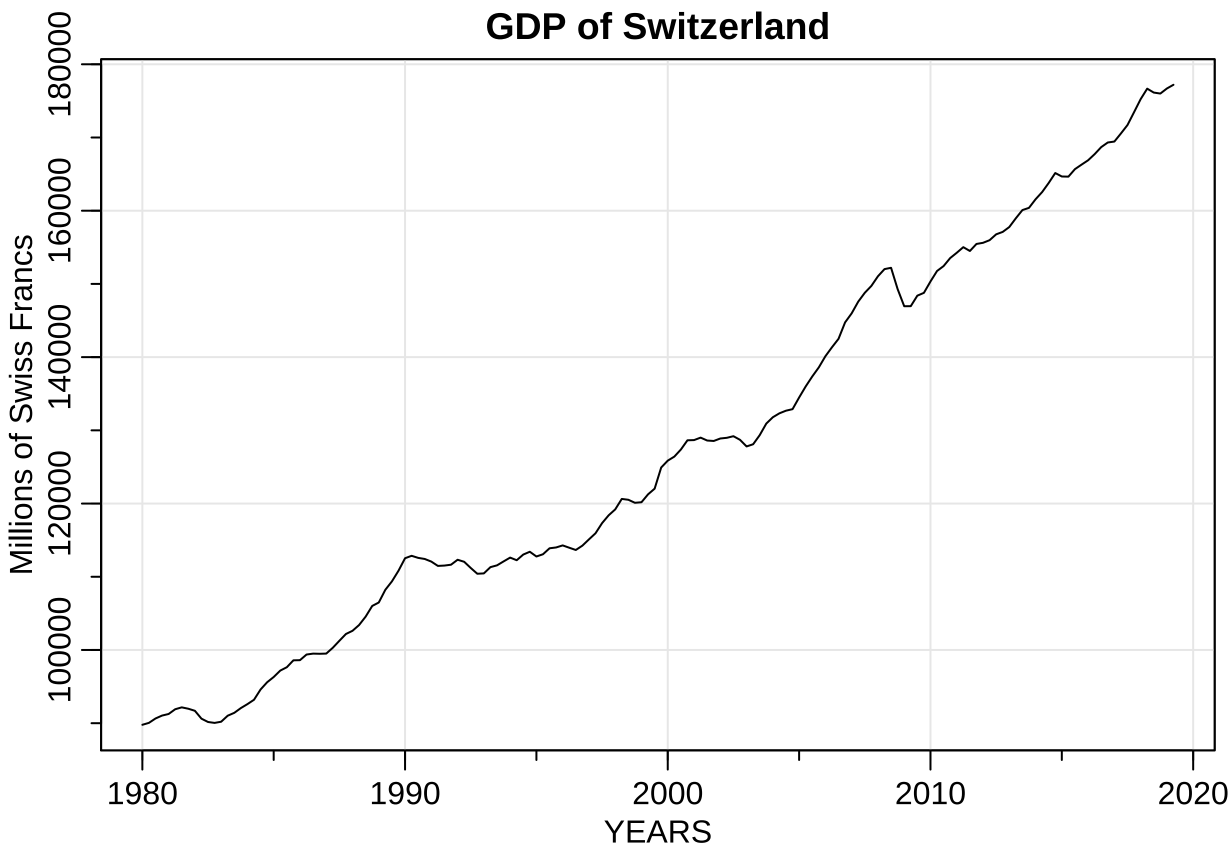
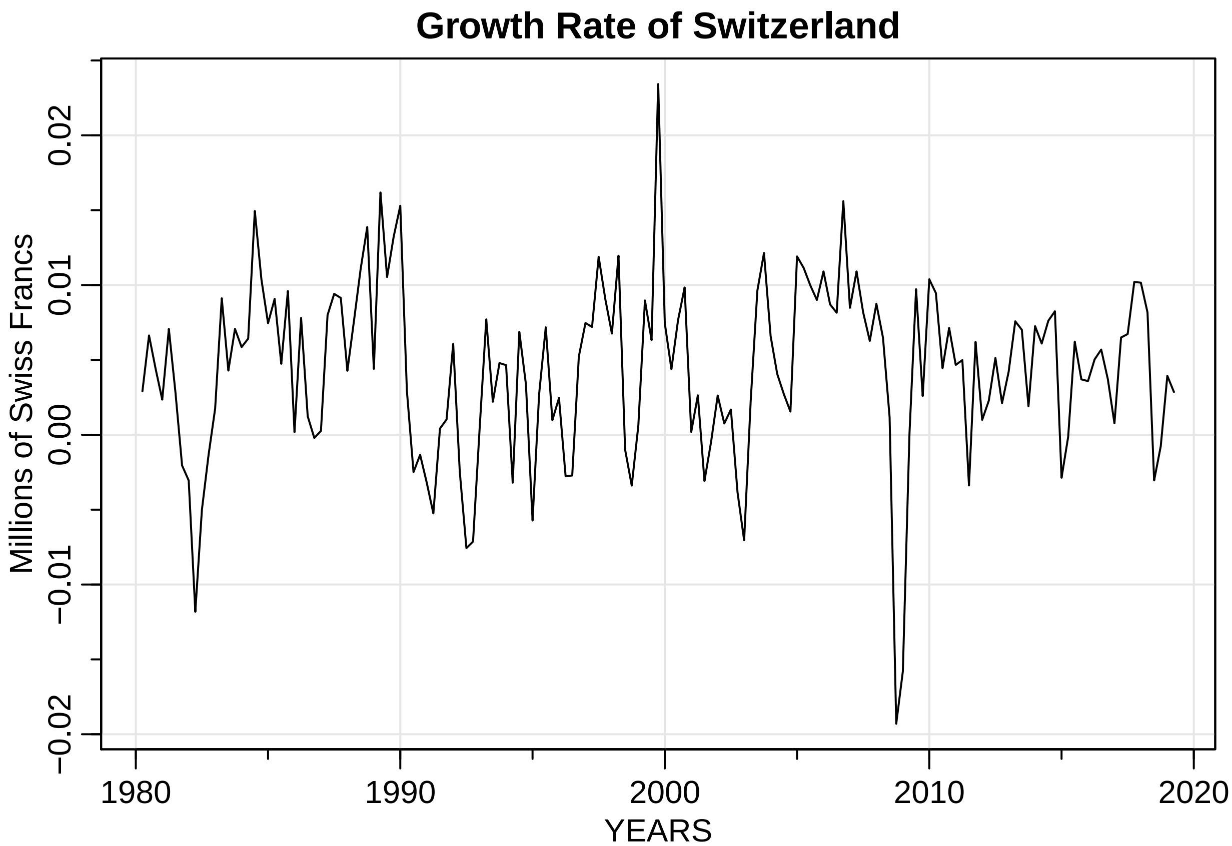
R Codes For Final Project

William Kubin

```r
# Data import & manipulations
setwd("/Users/paa.willie/Fall2019/Time Series/Project")

mydata = read.csv("GDP_swiss.csv", header = TRUE)

library(astsa)

swiss.gdp = ts(mydata$GDP, start = c(1980, 1), end = c(2019, 2), frequency = 4)

tsplot(swiss.gdp, xlab="YEARS", ylab="Millions of Swiss Francs", main = "GDP of Switzerland")
```




```r
tsplot(diff(log(swiss.gdp)), xlab="YEARS", ylab="Millions of Swiss Francs", main = "Growth Rate of Swit
```



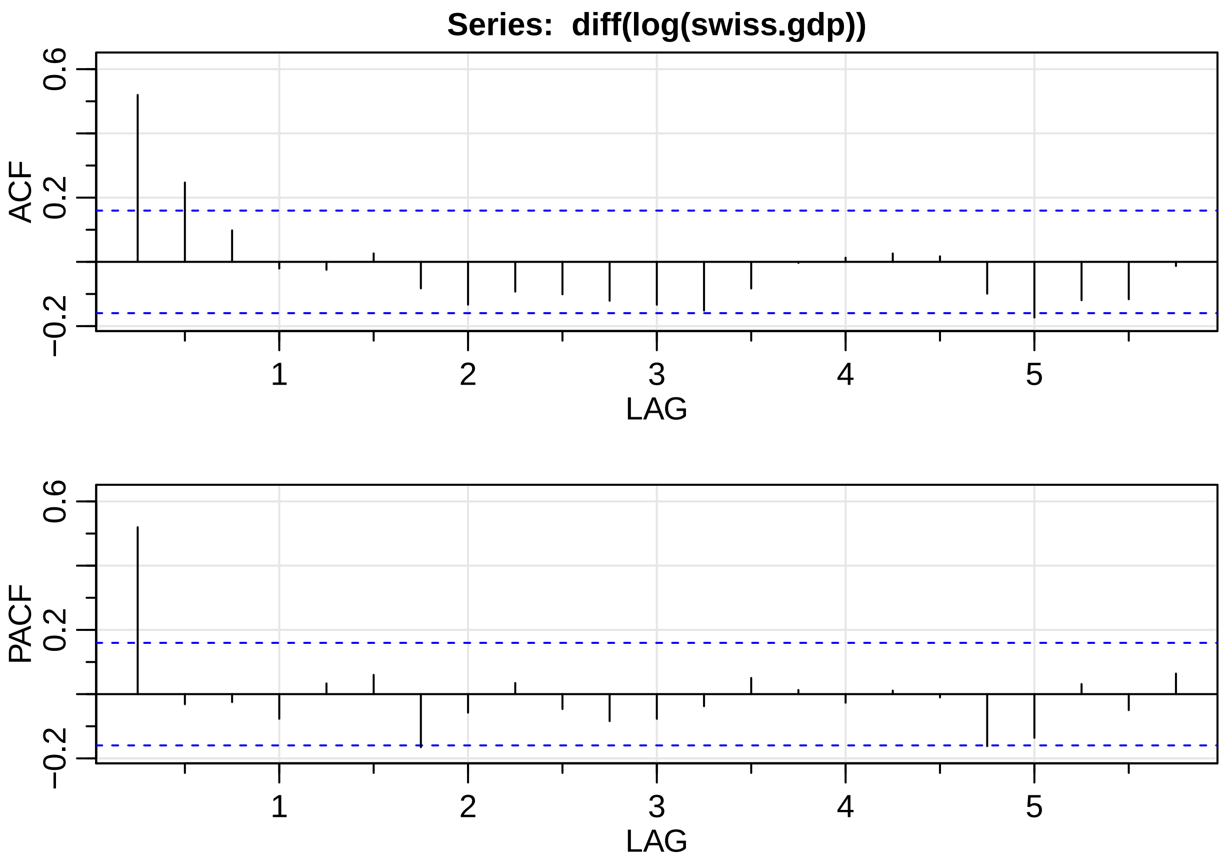

```r
acf2(diff(log(swiss.gdp)))
```

```
##         ACF  PACF
##  [1,]  0.52  0.52
##  [2,]  0.25 -0.03
##  [3,]  0.10 -0.02
##  [4,] -0.02 -0.08
##  [5,] -0.02  0.03
##  [6,]  0.03  0.06
##  [7,] -0.08 -0.17
##  [8,] -0.13 -0.06
##  [9,] -0.09  0.03
## [10,] -0.10 -0.05
## [11,] -0.12 -0.08
## [12,] -0.13 -0.08
## [13,] -0.15 -0.04
## [14,] -0.08  0.05
## [15,]  0.00  0.01
## [16,]  0.01 -0.03
## [17,]  0.03  0.01
## [18,]  0.02 -0.01
## [19,] -0.10 -0.16
## [20,] -0.17 -0.14
## [21,] -0.12  0.03
## [22,] -0.12 -0.05
## [23,] -0.01  0.06
```

```r
# Test of stationarity of growth rate of GDP
library("tseries")
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```




```r
adf.test(diff(log(swiss.gdp)))          # ADF test
```

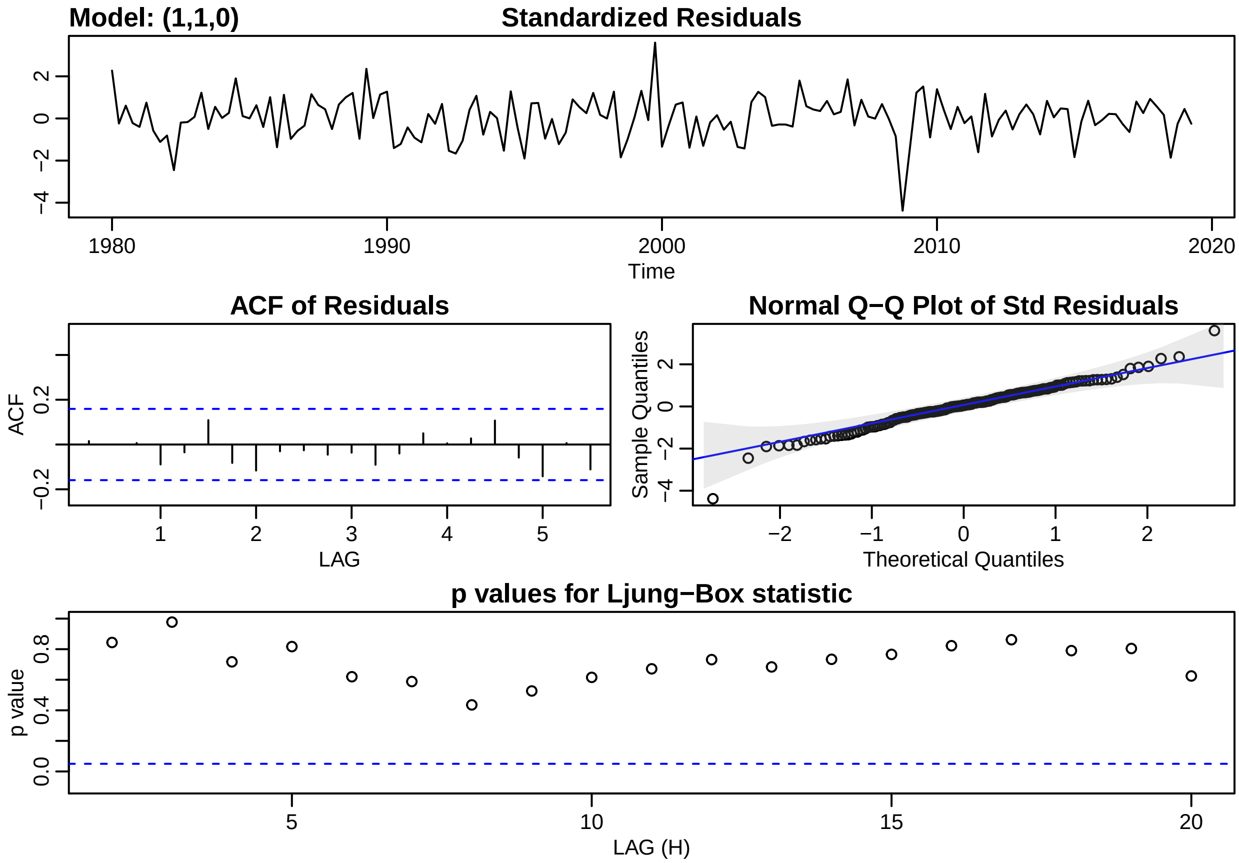
```
## Warning in adf.test(diff(log(swiss.gdp))): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(log(swiss.gdp))
## Dickey-Fuller = -4.1639, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```r
# ARIMA model fit
sarima(log(swiss.gdp), p = 1, d = 1, q = 0)     # ARMA(1,1,0) on growth rate
```

```
## initial  value -5.133969
## iter   2 value -5.291528
## iter   2 value -5.291528
## iter   2 value -5.291528
```

```
## final  value -5.291528
## converged
## initial  value -5.293528
## iter   2 value -5.293535
## iter   3 value -5.293537
## iter   4 value -5.293537
## iter   5 value -5.293538
## iter   5 value -5.293538
## iter   5 value -5.293538
## final  value -5.293538
## converged
```



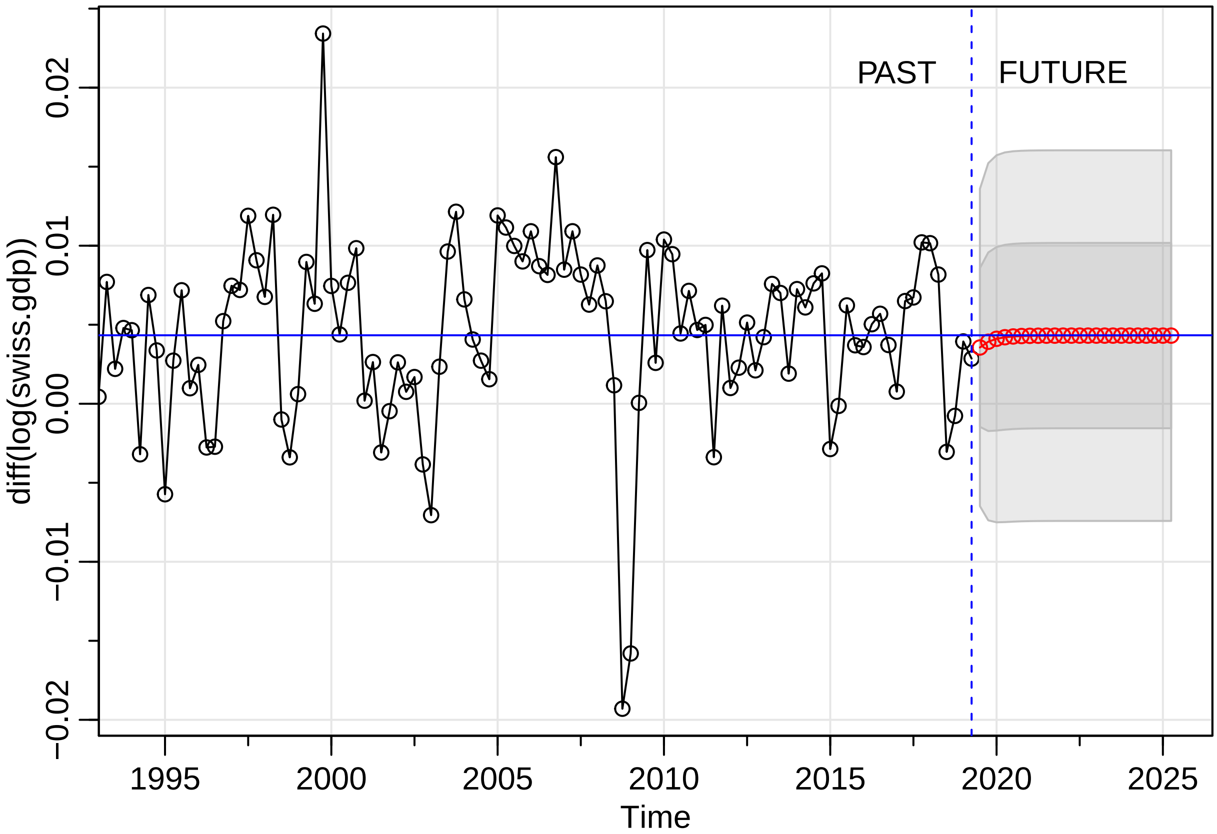

```
## $fit
##
## Call:
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,
##     Q), period = S), xreg = constant, transform.pars = trans, fixed = fixed,
##     optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##          ar1  constant
##       0.5168    0.0043
## s.e.  0.0678    0.0008
##
## sigma^2 estimated as 2.519e-05:  log likelihood = 608.31,  aic = -1210.62
##
## $degrees_of_freedom
```

```
## [1] 155
##
## $ttable
##          Estimate     SE t.value p.value
## ar1        0.5168 0.0678  7.6230       0
## constant   0.0043 0.0008  5.2109       0
##
## $AIC
## [1] -7.710982
##
## $AICc
## [1] -7.710486
##
## $BIC
## [1] -7.652582
```

```
mean.data = mean(diff(log(swiss.gdp)))

# Forecasting growth rate of GDP and Confidence Intervals
fCast = sarima.for(diff(log(swiss.gdp)), n.ahead = 24, p = 1, d = 0, q = 0)

fCast
```

```
## $pred
##             Qtr1        Qtr2        Qtr3        Qtr4
## 2019                         0.003559084 0.003922399
## 2020 0.004110166 0.004207207 0.004257359 0.004283278
## 2021 0.004296674 0.004303597 0.004307175 0.004309024
## 2022 0.004309980 0.004310474 0.004310729 0.004310861
## 2023 0.004310929 0.004310964 0.004310982 0.004310992
## 2024 0.004310997 0.004310999 0.004311001 0.004311001
## 2025 0.004311002 0.004311002
##
## $se
##             Qtr1        Qtr2        Qtr3        Qtr4
## 2019                         0.005018975 0.005649634
## 2020 0.005806502 0.005847689 0.005858641 0.005861563
## 2021 0.005862343 0.005862552 0.005862607 0.005862622
## 2022 0.005862626 0.005862627 0.005862628 0.005862628
## 2023 0.005862628 0.005862628 0.005862628 0.005862628
## 2024 0.005862628 0.005862628 0.005862628 0.005862628
## 2025 0.005862628 0.005862628
```

```
text(2017, 0.021, "PAST"); text(2022, 0.021, "FUTURE")

abline(h = mean.data, col = 4)                              # estimated mean

abline(v = 2019.25, lty = 2, col = 4)                 # vertical line showing second quarter of 2019
```



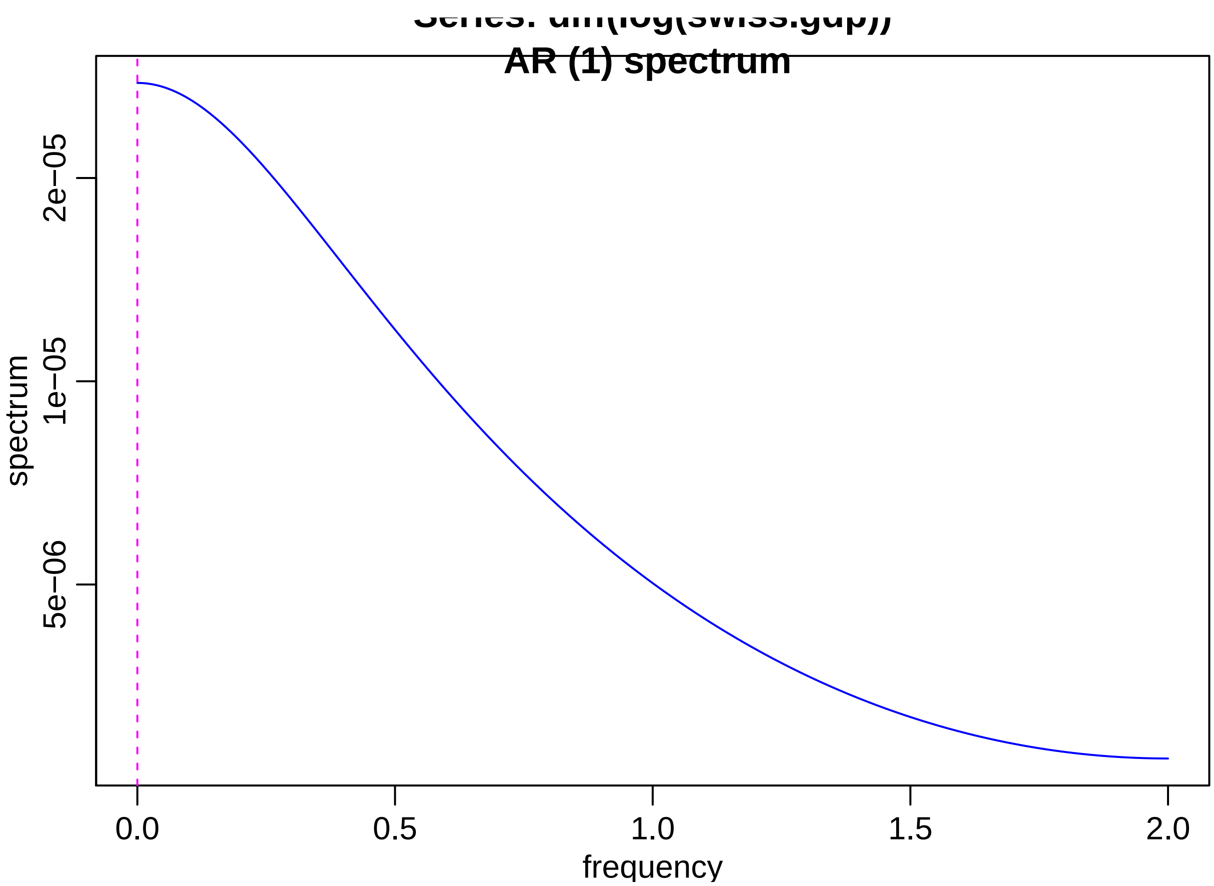

```r
predictions = fCast$pred
errors = fCast$se

lowerCI = c()

upperCI = c()

for (i in 1:24){

  lowerCI[i] = predictions[i] - 1.96*errors[i]

  upperCI[i] = predictions[i] + 1.96*errors[i]

}

lowerCI
```

```
##  [1] -0.006278108 -0.007150883 -0.007270578 -0.007254264 -0.007225578
##  [6] -0.007205385 -0.007193519 -0.007187004 -0.007183535 -0.007181715
## [11] -0.007180768 -0.007180276 -0.007180021 -0.007179889 -0.007179821
## [16] -0.007179786 -0.007179768 -0.007179758 -0.007179753 -0.007179751
## [21] -0.007179750 -0.007179749 -0.007179749 -0.007179748
```

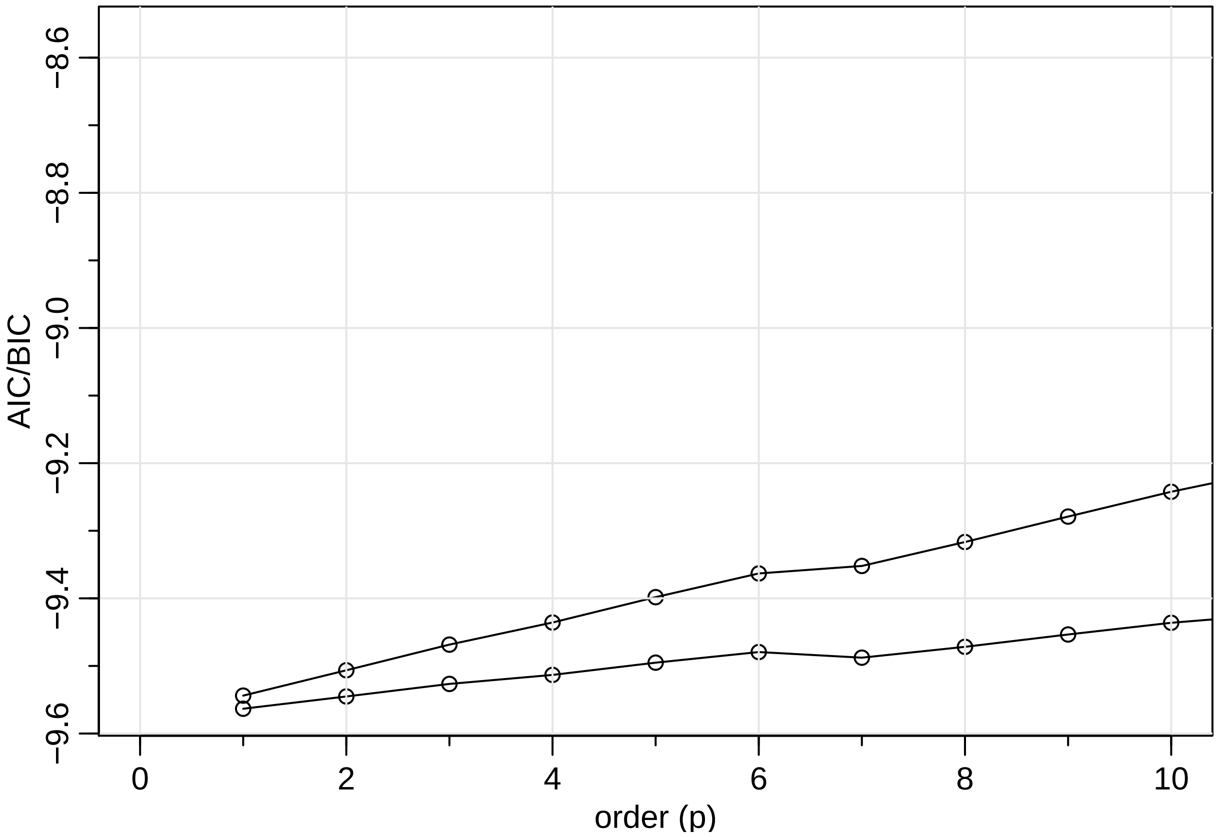
```r
upperCI
```

```
##  [1] 0.01339628 0.01499568 0.01549091 0.01566868 0.01574030 0.01577194
##  [7] 0.01578687 0.01579420 0.01579789 0.01579976 0.01580073 0.01580122
## [13] 0.01580148 0.01580161 0.01580168 0.01580171 0.01580173 0.01580174
## [19] 0.01580175 0.01580175 0.01580175 0.01580175 0.01580175 0.01580175
```

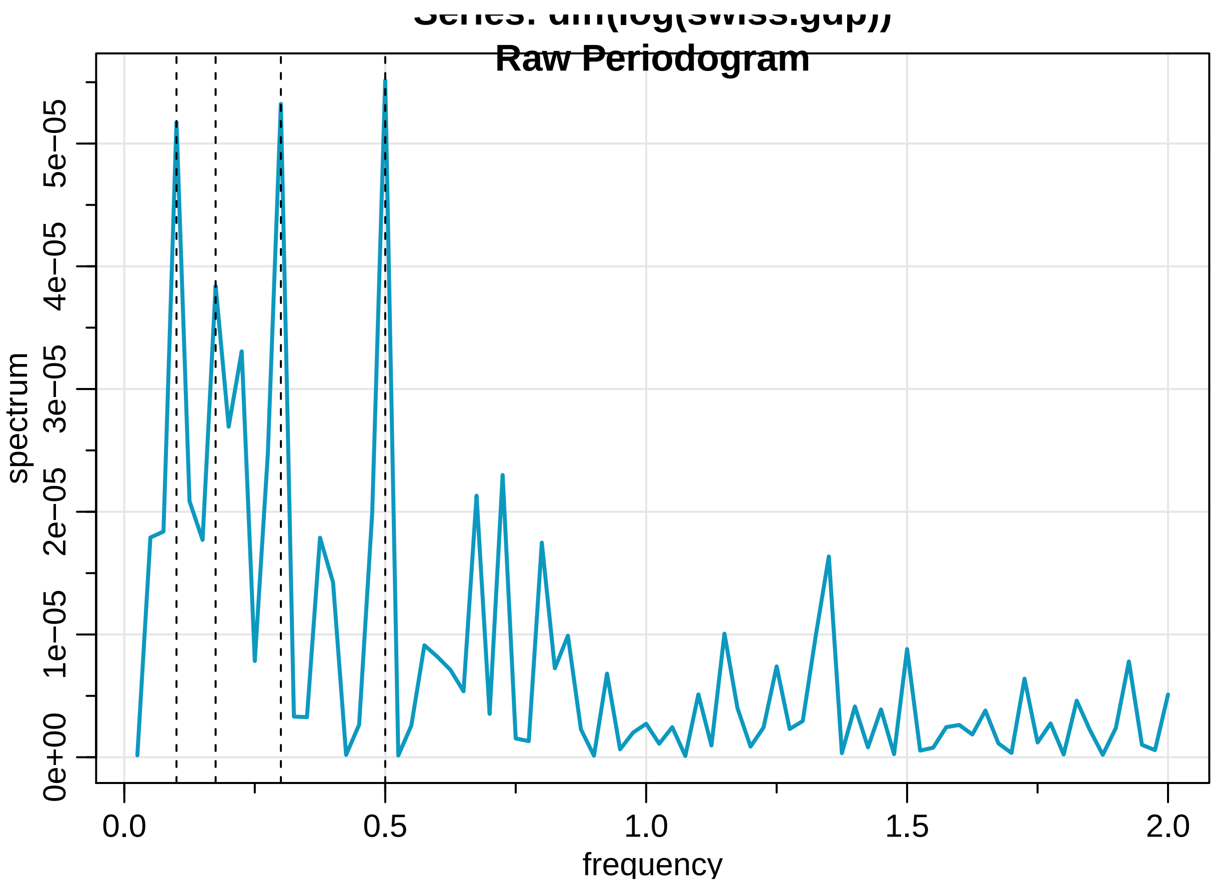
```r
# Spectral Estimation
# parametric spectral estimation - AR estimation
AR.spec = spec.ar(diff(log(swiss.gdp)), col = 4)

abline(v = 0, col = 6, lty = "dashed")         # ?? year cycle
```




```r
N = length(swiss.gdp)

c() -> AIC -> BIC

for (k in 1:30){

  sigma2 = ar(diff(log(swiss.gdp)), order = k, aic = FALSE)$var.pred

  BIC[k] = log(sigma2) + k*log(N)/N

  AIC[k] = log(sigma2) + (N + 2*k)/N

}

IC = cbind(AIC, BIC + 1)

ts.plot(IC, type = "o", xlab = "order (p)", ylab = "AIC/BIC", xlim = c(0, 10))

Grid()
```




```
# non-parametric spectral estimation - Periodogram
# raw periodogram
per = mvspec(diff(log(swiss.gdp)), col = rgb(.05, .6, .75), lwd = 2)

abline(v = 1/10, lty = "dashed")    # 10 year cycle

abline(v = 0.175, lty = "dashed")   # close to the El Nino Effect

abline(v = 3/10, lty = "dashed")

abline(v = 5/10, lty = "dashed")
```

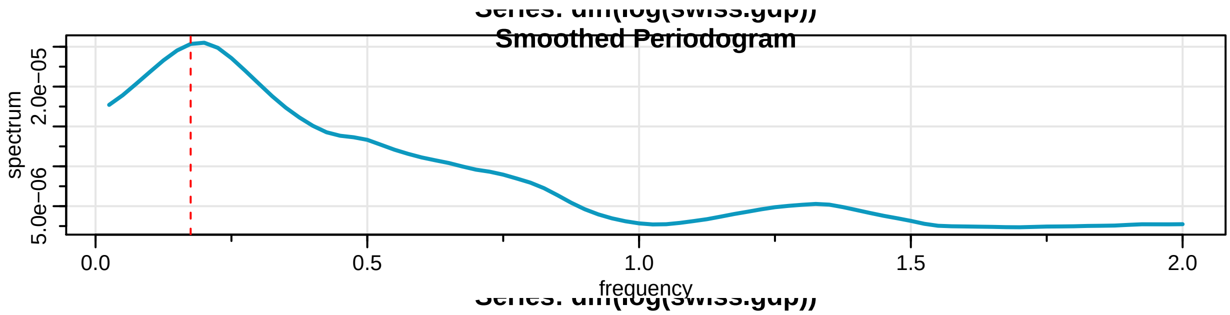


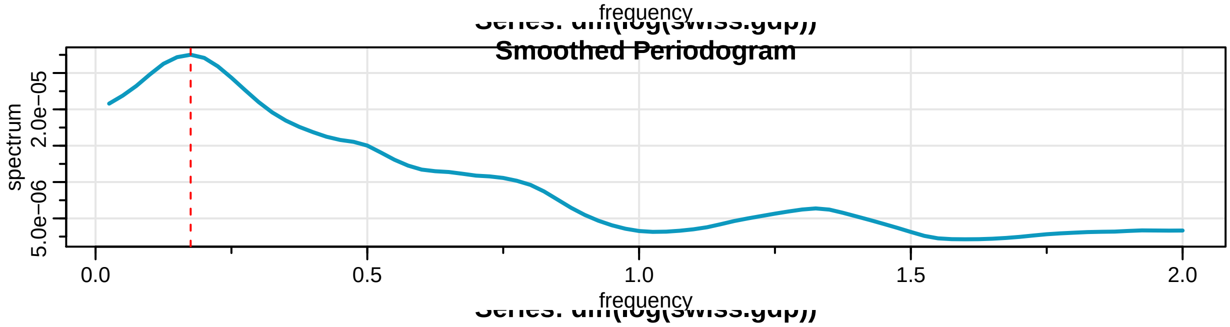

```
per$details[1:50,]
```

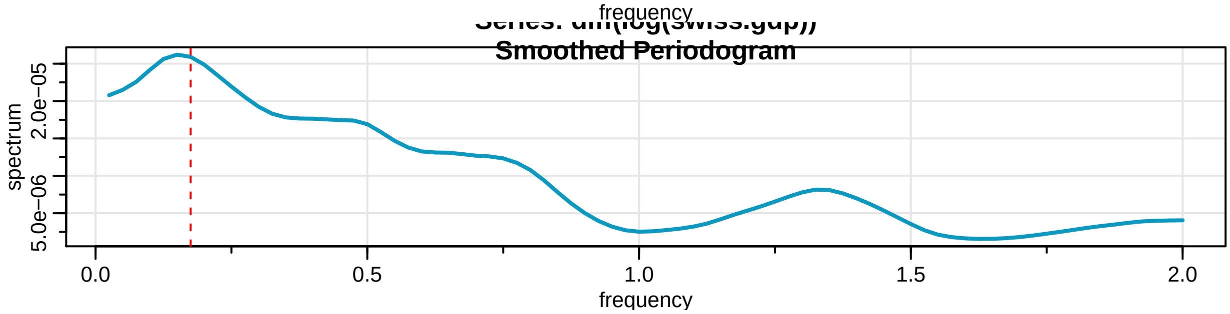
```
##       frequency  period spectrum
##  [1,]     0.025 40.0000    0e+00
##  [2,]     0.050 20.0000    0e+00
##  [3,]     0.075 13.3333    0e+00
##  [4,]     0.100 10.0000    1e-04
##  [5,]     0.125  8.0000    0e+00
##  [6,]     0.150  6.6667    0e+00
##  [7,]     0.175  5.7143    0e+00
##  [8,]     0.200  5.0000    0e+00
##  [9,]     0.225  4.4444    0e+00
## [10,]     0.250  4.0000    0e+00
## [11,]     0.275  3.6364    0e+00
## [12,]     0.300  3.3333    1e-04
## [13,]     0.325  3.0769    0e+00
## [14,]     0.350  2.8571    0e+00
## [15,]     0.375  2.6667    0e+00
## [16,]     0.400  2.5000    0e+00
## [17,]     0.425  2.3529    0e+00
## [18,]     0.450  2.2222    0e+00
## [19,]     0.475  2.1053    0e+00
## [20,]     0.500  2.0000    1e-04
## [21,]     0.525  1.9048    0e+00
## [22,]     0.550  1.8182    0e+00
## [23,]     0.575  1.7391    0e+00
```

```
## [24,]     0.600  1.6667    0e+00
## [25,]     0.625  1.6000    0e+00
## [26,]     0.650  1.5385    0e+00
## [27,]     0.675  1.4815    0e+00
## [28,]     0.700  1.4286    0e+00
## [29,]     0.725  1.3793    0e+00
## [30,]     0.750  1.3333    0e+00
## [31,]     0.775  1.2903    0e+00
## [32,]     0.800  1.2500    0e+00
## [33,]     0.825  1.2121    0e+00
## [34,]     0.850  1.1765    0e+00
## [35,]     0.875  1.1429    0e+00
## [36,]     0.900  1.1111    0e+00
## [37,]     0.925  1.0811    0e+00
## [38,]     0.950  1.0526    0e+00
## [39,]     0.975  1.0256    0e+00
## [40,]     1.000  1.0000    0e+00
## [41,]     1.025  0.9756    0e+00
## [42,]     1.050  0.9524    0e+00
## [43,]     1.075  0.9302    0e+00
## [44,]     1.100  0.9091    0e+00
## [45,]     1.125  0.8889    0e+00
## [46,]     1.150  0.8696    0e+00
## [47,]     1.175  0.8511    0e+00
## [48,]     1.200  0.8333    0e+00
## [49,]     1.225  0.8163    0e+00
## [50,]     1.250  0.8000    0e+00
```

```r
par(mfrow = c(3,1))

x.t0 = mvspec(diff(log(swiss.gdp)), spans = c(9,9), col = rgb(.05, .6, .75), lwd = 2)

abline(v = 0.175, lty = "dashed", col = 2)   # close to the El Nino Effect

# 20% tapering
x.t1 = mvspec(diff(log(swiss.gdp)), spans = c(9,9), taper = 0.2, col = rgb(.05, .6, .75), lwd = 2)

abline(v = 0.175, lty = "dashed", col = 2)

# 50% full tapering
x.t2 = mvspec(diff(log(swiss.gdp)), spans = c(9,9), taper = 0.5, col = rgb(.05, .6, .75), lwd = 2)

abline(v = 0.175, lty = "dashed", col = 2)
```


Smoothed Periodogram
spectrum
5.0e−06
2.0e−05
0.0
0.5
1.0
1.5
2.0
frequency


Smoothed Periodogram
spectrum
5.0e−06
2.0e−05
0.0
0.5
1.0
1.5
2.0
frequency


Smoothed Periodogram
spectrum
5.0e−06
2.0e−05
0.0
0.5
1.0
1.5
2.0
frequency