

ECOM90004 – Time Series Analysis and Forecasting – Assignment 2

Joshua Copeland (SID:1444772)

Tutorial: Wednesday 12pm

No other group members

Question 1

Given limited space for reporting results, answers to part (a) to (c) have been presented together.

Table 1: Charts required for Q1(a) to Q1(c).

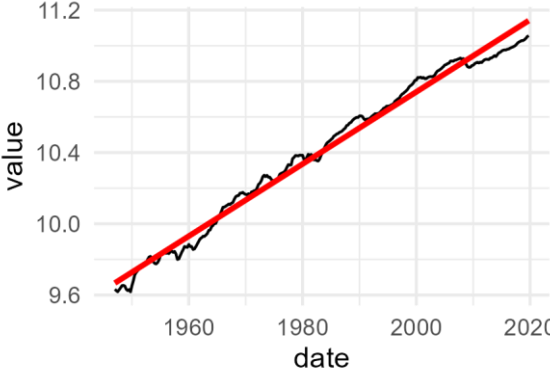
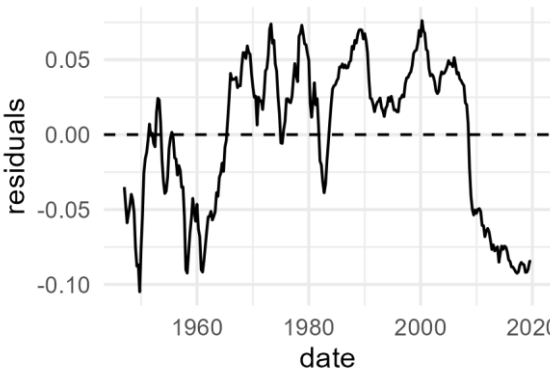
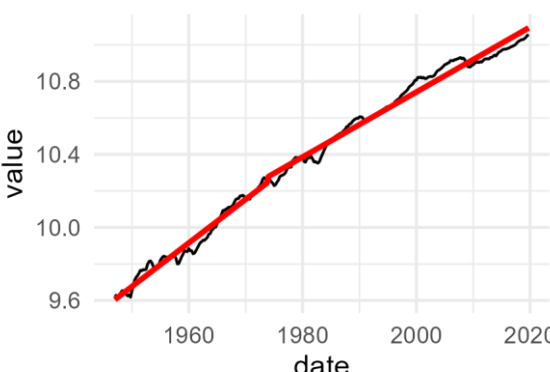
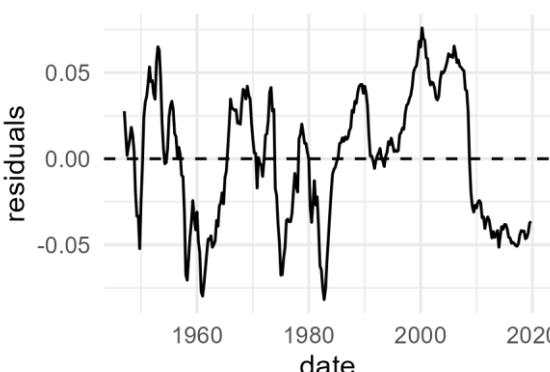
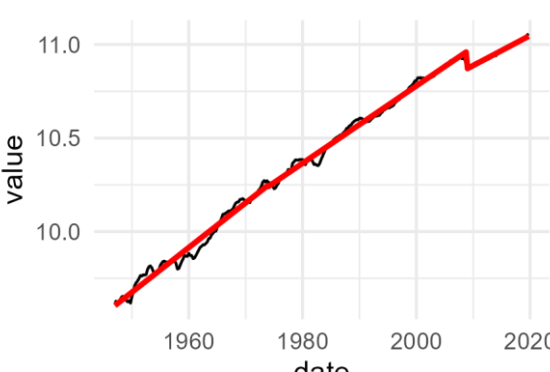
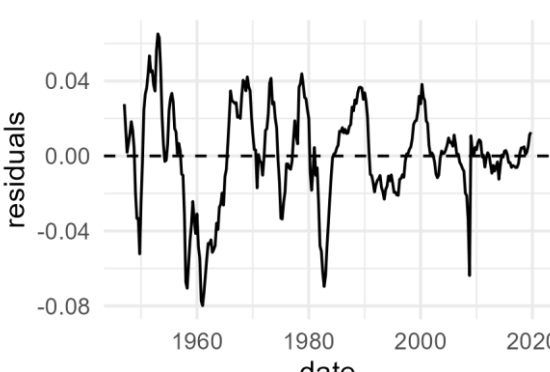
(a)	<p>Chart 1: Linear model fitted values</p> 	<p>Chart 2: Linear model residuals</p> 
(b)	<p>Chart 3: One-break model fitted values</p> 	<p>Chart 4: One-break model residuals</p> 
(c)	<p>Chart 5: Two-break model fitted values</p> 	<p>Chart 6: Two-break model residuals</p> 

Table 2: Reported coefficients for Q1(a) to Q1(c)

Coefficient	Linear model (a)	1 break model (b)	2 break model (c)
β_0	9.6621	9.5985	9.5985
β_1	0.0203	0.0239	0.0239
β_3	NA	0.1950	0.0798
β_4	NA	-0.0061	-0.0032
β_5	NA	NA	0.1942
β_6	NA	NA	-0.0046

d)

- i. In the two-break model, continuity means the fitted line passes smoothly through the breakpoints at 1973 and 2008, allowing slope changes but not any intercept jumps. The two break model is defined as:

$$Y_t = \beta_0 + \beta_1 Time_t + \beta_3 DU_{1,t} + \beta_4 DT_{1,t} + \beta_5 DU_{2,t} + \beta_6 DT_{2,t} + Z_t$$

Before 1973, all break terms are zero, so:

$$Y_t = \beta_0 + \beta_1 Time_t + Z_t$$

At break point $t = 1973$, continuity requires (noting we've removed Z_t because we're modelling fitted values):

$$\beta_0 + \beta_1(1973) = \beta_0 + \beta_3 + (\beta_1 + \beta_4)(1973)$$

Rearranging gives the restriction:

$$\beta_3 = -1973\beta_4$$

For the second breakpoint, the left hand side of the equality above is not the one-break fitted lines with 1973 terms, while the right-hand side includes the additional dummy for 2008. If we impose continuity at $t = 2008$ and solve the equality again we can rearrange for the following restrictions:

$$\beta_5 = -2008\beta_6$$

The definitions for β_3 and β_5 given above correspond with parameter restrictions required for continuity at 1973 and 2008 respectively.

- ii. Applying the continuity restriction means the fitted line can only change in slope, not in intercept. Therefore, we get a GFC style trend break by forcing the fitted line to pass smoothly through the break date. In the break models we use here, $DU_{x,t}$ impacts the level whereas $DT_{y,t}$ impacts the slope. In the example we used above, we are effectively neutralising the impact of $DU_{x,t}$ on the level in

either model, which generalises to the GFC trend break for retail sales modelled earlier in the semester:

$$\beta_3 DU_{1,t} + \beta_4 DT_{1,t} = -1973(\beta_4) DU_{1,t} + \beta_4 Time_t DU_{1,t} = \beta_4 (Time_t - 1973) DU_{1,t}$$

$$\beta_5 DU_{2,t} + \beta_6 DT_{2,t} = -2008(\beta_6) DU_{2,t} + \beta_6 Time_t DU_{2,t} = \beta_6 (Time_t - 2008) DU_{2,t}$$

iii. The estimates of the jumps (J_t) are:

$$J_{1973} = \beta_3 + \beta_4(1973) = 0.0798 + (-0.0032) \times 1973 = 0.0798 - 6.3136 = -6.2338$$

$$J_{2008} = \beta_5 + \beta_6(2008) = 0.1942 + (-0.0046) \times 2008 = 0.1942 - 9.2368 = -9.0426$$

We can test if their implementation is supported by using a Wald test to see if their values are statistically different from zero. The p-values of J_{1973} and J_{2008} are 0.0000 and 0.00036 respectively. Therefore the null hypothesis of no jump is rejected at any reasonable significance level: discontinuities are supported in 1973 and 2008.

Question 2

- a) Table 3 summarises the preferred AR lag orders $p^{(j)}$. Z_t is found to be an AR(4) process for the linear model, and an AR(3) process for the other two. Importantly, all models chosen have large Ljung-Box test p-values, which does not give us any evidence of autocorrelation in Z_t for any model.

Table 3: Summary of $p^{(j)}$ choice

Model	$p^{(j)}$ choice	AICc	Ljung-Box p-value
Linear	4	-1944.4000	0.3216
1 break	3	-1928.4130	0.3396
2 break	3	-1875.4320	0.6157

- b) Table 4 summarise the Augmented Dicky-Fuller (ADF) test statistics produced by each model. The residuals Z_t across all models (Chart 2, 4 and 6) show no clear trend and appear to oscillate around zero, although it is difficult to judge visually whether these fluctuations are perfectly balanced about the mean, so the drift specification is most appropriate for these tests. I have also set their maximum lags to be one greater than the AR order chosen in Q2a.

Table 4: Summary of asymptotic Augmented Dicky-Fuller (ADF) tests

<i>Model</i>	<i>Max lag</i>	<i>t-statistic</i>	<i>p-value</i>
Linear	5	-2.0432	0.2683
1 break	4	-3.5591	0.0066
2 break	4	-4.5163	0.0002

- c) Table 5 summarises the ADF critical values produced by the simulation process for each model, all using 1000 repetitions. Max lag specification is consistent with earlier answers.

Table 5: Simulated critical values for ADF statistics

<i>Model</i>	<i>Max lag</i>	<i>5% critical value</i>
Linear	5	-2.90456
1 break	4	-2.95266
2 break	4	-2.87343

- d) Table X summarises the p-values implied by the ADF simulation process.

Table 6: Simulated p-values for ADF statistics

<i>Model</i>	<i>Simulated ADF test statistic</i>	<i>Simulated p-value</i>
Linear	-2.04324	0.274
1 break	-3.55911	0.008
2 break	-4.51627	0.001

- e) The ADF test results for the linear model cannot reject the unit root null hypothesis given its large p-value (0.27). Conversely, the structural break models strongly reject it given their small p-values (0.007 and <0.001 respectively). The simulated critical values and p-values are very close to their asymptotic results, reassuring us that having a finite-sample distribution does not materially alter conclusions. Therefore, these results only support stationarity in Z_t where a structural break is applied.
- f) There are two key implications of these results. Firstly, the close match between asymptotic and simulated critical values indicates that finite-sample effects are negligible here, so the asymptotic distribution provides a reliable basis for inference. Secondly, the results highlight the importance of accounting for structural breaks in this series. Without doing this, we cannot specify a model that meets the basic requirements of time series analysis: zero conditional mean errors.

R code appendix

```
library(tidyverse)
```

```
library(lubridate)
```

```
library(forecast)
```

```
library(urca)
```

```
library(car)
```

Question 1

(a) – (c) – given together for efficiency

```
# Data import and cleaning
```

```
data <- read_csv("USRealGDPPerCapita.csv") %>%
```

```
  select(date = observation_date, value = RealGDPPerCap) %>%
```

```
  mutate(date = dmy(date)) %>%
```

```
  mutate(value = log(value)) %>%
```

```
  na.omit() %>%
```

```
# Variables for models
```

```
mutate(time = row_number()*0.25) %>%
```

```
mutate(du_1 = if_else(lubridate::year(date) > 1973, 1, 0), dt_1 = if_else(year(date) > 1973, time, 0)) %>%
```

```
mutate(du_2 = if_else(lubridate::year(date) > 2008, 1, 0), dt_2 = if_else(year(date) > 2008, time, 0))
```

```
# Creating models
```

```
linear_model <- lm(value ~ time, data = data)
```

```
one_break_model <- lm(value ~ time + du_1 + dt_1, data = data)
```

```
two_break_model <- lm(value ~ time + du_1 + dt_1 + du_2 + dt_2, data = data)
```

```
# Creating table of coefficients
```

```
tab <- cbind(
```

```
  "Linear model (a)" = coef(linear_model)[c("(Intercept)","time","du_1","dt_1","du_2","dt_2")],
```

```
  "1 break model (b)" = coef(one_break_model)[c("(Intercept)","time","du_1","dt_1","du_2","dt_2")],
```

```
  "2 break model (c)" = coef(two_break_model)[c("(Intercept)","time","du_1","dt_1","du_2","dt_2")]
```

```
)
```

```
rownames(tab) <- c("beta_0","beta_1", "beta_3","beta_4","beta_5","beta_6")
```

```
round(tab, 4)
```

```

# Creating charts of fitted values and residuals

models <- list(linear_model, one_break_model, two_break_model)

labs <- c("Linear model", "One-break model", "Two-break model")

invisible(lapply(seq_along(models), function(i){

  m <- models[[i]]; lab <- labs[i]

  aug <- data.frame(date = data$date, value = data$value, fitted = fitted(m), residuals = resid(m))

  p1 <- ggplot(aug, aes(date, value)) + geom_line() + geom_line(aes(y = fitted), color = "red",
    linewidth = 1) +

    labs(title = paste0("Chart ", (i-1)*2+1, ": ", lab, " \n fitted values"), x = "date", y = "value")

  p2 <- ggplot(aug, aes(date, residuals)) + geom_line() + geom_hline(yintercept = 0, linetype =
    "dashed") +

    labs(title = paste0("Chart ", (i-1)*2+2, ": ", lab, " \n residuals"), x = "date", y = "residuals")

  ggsave(paste0("chart", (i-1)*2+1, ".png"), p1, width = 3, height = 2.5, units = "in", dpi = 300)
  ggsave(paste0("chart", (i-1)*2+2, ".png"), p2, width = 3, height = 2.5, units = "in", dpi = 300)

}))

```

(d)(iii)

H0: no jump at 1973

```
linearHypothesis(two_break_model, "du_1 + 1973*dt_1 = 0")
```

H0: no jump at 2008

```
linearHypothesis(two_break_model, "du_2 + 2008*dt_2 = 0")
```

#Question 2

(a)

function: fit AR(p) to residuals (Z), compute AICc & Ljung-Box, mark min AICc

```

resid_ar_table <- function(model, label, pmax = 9) {

  Z <- resid(model); AICc <- LBp <- numeric(pmax + 1)    # storage

  for (p in 0:pmax) {                                # loop over AR orders

    eq <- Arima(Z, order = c(p, 0, 0), include.mean = FALSE, method = "ML")

    AICc[p + 1] <- eq$aicc                             # record AICc

    LBp[p + 1] <- Box.test(residuals(eq), lag = p + 4,
      type = "Ljung-Box", fitdf = p)$p.value

  }

  # results table for display

  Stats <- cbind(sprintf("%.4f", LBp),

```

```

        sprintf("%.2f", AICc),
        replace(rep("", pmax + 1), which.min(AICc), "<="))
rownames(Stats) <- paste0("p=", 0:pmax); colnames(Stats) <- c("LBp", "AICc", "minAICc")
cat("\n", label, "\n", sep = ""); print(Stats, quote = FALSE)
# return summary row for this model
data.frame(model = label, p = which.min(AICc) - 1,
           AICc = min(AICc), LB_p = LBp[which.min(AICc)])
}
# Run for all models
models <- list(linear_model, one_break_model, two_break_model) # model objects
names(models) <- c("linear_model", "one_break_model", "two_break_model") # add labels
# combine results into final dataframe
final <- do.call(rbind, Map(function(m, lab) resid_ar_table(m, lab, pmax = 9),
                           models, names(models)))
print(final)

```

(b)

```

# models, labels, and max lags (linear=4, others=5)
models <- list(linear_model, one_break_model, two_break_model)
labels <- c("linear_model", "one_break_model", "two_break_model")
lags <- c(5, 4, 4)
# compact ADF on residuals -> selected lag, t-stat, p-value (via punitroot)
adf_rows <- t(mapply(function(m, L) {
  adf <- ur.df(resid(m), type="drift", selectlags="AIC", lags=L)
  tval <- as.numeric(adf@teststat[1]) # tau2 for 'drift'
  pval <- urca::punitroot(tval, trend="c", statistic="t")
  c(lag = adf@lags, t_stat = tval, p_value = pval)
}, models, lags, SIMPLIFY = TRUE))
adf_results <- data.frame(model = labels, adf_rows, row.names = NULL)
print(adf_results)

```

(c)

```

# Sample sizes of residuals
n_vec <- sapply(models, function(m) length(resid(m)))

```



```

# Simulate null distributions and store 5% critical values
reps <- 1000
tstats_list <- vector("list", length(models))
cv5 <- numeric(length(models))
for(i in seq_along(models)){
  tstats <- replicate(reps, {
    y <- cumsum(rnorm(n_vec[i]))          # random walk under H0
    as.numeric(ur.df(y, type="drift", selectlags="AIC", lags=lags[i])@teststat[1]) # tau2
  })
  tstats_list[[i]] <- tstats
  cv5[i] <- unname(quantile(tstats, 0.05, na.rm = TRUE)) # 5% critical value
}
cv_table <- data.frame(model = labels, Lmax = lags, cv_5pct = cv5)
print(cv_table)

```

(d)

```

# Observed ADF t statistics on residuals (drift, AIC, same lag caps)
obs_t <- numeric(length(models))
for(i in seq_along(models)){
  adf <- ur.df(resid(models[[i]]), type="drift", selectlags="AIC", lags=lags[i])
  obs_t[i] <- as.numeric(adf@teststat[1])
}

# Simulated p-values using the saved null draws in tstats_list
pvals <- numeric(length(models))
for(i in seq_along(models)){
  tsim <- tstats_list[[i]]
  pvals[i] <- mean(tsim <= obs_t[i]) # left-tail probability under H0
}

# Table of simulated p-values
p_table <- data.frame(model = labels, t_stat = obs_t, p_sim = pvals)
print(p_table)

```