Problem Set 2 Questions 2 and 4

Pedro Scatimburgo

03/06/2022

Preamble

This file contains questions 2 and 4 of Problem Set 2. Here we set the seed and load the main packages.

```
rm(list=ls())

set.seed(999)

library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(tidylog)
```

```
## Warning: package 'tidylog' was built under R version 4.0.5
```

```
##
## Attaching package: 'tidylog'
```

```
## The following objects are masked from 'package:dplyr':
##
##     add_count, add_tally, anti_join, count, distinct, distinct_all,
##     distinct_at, distinct_if, filter, filter_all, filter_at, filter_if,
##     full_join, group_by, group_by_all, group_by_at, group_by_if,
##     inner_join, left_join, mutate, mutate_all, mutate_at, mutate_if,
##     relocate, rename, rename_all, rename_at, rename_if, rename_with,
##     right_join, sample_frac, sample_n, select, select_all, select_at,
##     select_if, semi_join, slice, slice_head, slice_max, slice_min,
##     slice_sample, slice_tail, summarise, summarise_all, summarise_at,
##     summarise_if, summarize, summarize_all, summarize_at, summarize_if,
##     tally, top_frac, top_n, transmute, transmute_all, transmute_at,
##     transmute_if, ungroup
```

```
## The following objects are masked from 'package:tidyr':
##
##     drop_na, fill, gather, pivot_longer, pivot_wider, replace_na,
##     spread, uncount
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(urca)
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
path <- "C:\\Users\\psrov\\OneDrive - Fundacao Getulio Vargas - FGV\\Documentos\\EESP\\Disciplinas\\2 T
```

Question 2

We will run a Monte Carlo simulation for items 1. and 2. For the Monte Carlo simulation itself, I will use a standard `for` loop. Let's set the parameters. `capT` is the sample size T, `alpha` is the intercept α, `delta` is the slope δ and `M` is the number of Monte Carlo repetitions M.

```
capT = 10^4
alpha = 0
delta = 1
M = 10^4
```

Five degrees of freedom

Now we generate the $\{\epsilon_t\}$ process and run the simulations. `dgfr` stores the degrees of freedom and `siglevel` the level of significance. I use `siglevel` to calculate the `tscore` of a normal distribution. Then I create a '`results` dataframe that will store the calculated t-scores of the linear regression.

I loop in `i` through 1 to `M = 10^4`. I generate the $\{\epsilon_t\}$ process using the `rt` function and store the pseudo-random values in `epsilon`. Then I create the $\{Y_t\}$ process and store it in `Yt`. `x` is simply a sequence `1:10^4`. We will run a linear regression of $\{Y_t\}$ against a column of numbers from 1 to $10,000$. I store the linear regression in the `reg` object. Note that to change the null hypothesis, I use as formula the expression `Yt ~ x + offset(1.00*x)`. For more information about this, check: https://stats.stackexchange.com/questions/9825/changing-null-hypothesis-in-linear-regression

Finally, I store each t-score in the `results` dataframe. To calculate the rejection rate, I use `plyr::count` to check how many values attend the condition `abs(results) > abs(tscore)`.

```
dgfr = 5 # Degrees of freedom
siglevel = 0.1 # Significance level
tscore = qnorm(p = siglevel/2) # Tscore

results_5 <- data.frame(
  "tscore" = numeric(capT)
)

for(i in 1:M){

  epsilon = rt(n = capT, df = dgfr)

  Yt = alpha + delta*seq(1:capT) + epsilon
```

```
  x = seq(1:capT)

  reg <- lm(data = as.data.frame(Yt), formula = Yt ~ x + offset(1.00*x))

  results_5$tscore[i] <- summary(reg)[["coefficients"]][2,"t value"]

}

freq_5 <- plyr::count(abs(results_5) > abs(tscore))
freq_5$freq[2]/10000*100
```

```
## [1] 9.99
```

One degree of freedom

The rejection rate is 9.99, which is pretty close to the significante interval.

Now we perform the same procedure, but with `dgfr = 1`:

```
dgfr = 1 # Degrees of freedom
siglevel = 0.1 # Significance level
tscore = qnorm(p = siglevel/2) # Tscore

results_1 <- data.frame(
  "tscore" = numeric(capT)
)

for(i in 1:M){

  epsilon = rt(n = capT, df = dgfr)

  Yt = alpha + delta*seq(1:capT) + epsilon

  x = seq(1:capT)

  reg <- lm(data = as.data.frame(Yt), formula = Yt ~ x + offset(1.00*x))

  results_1$tscore[i] <- summary(reg)[["coefficients"]][2,"t value"]

}

freq_1 <- plyr::count(abs(results_1) > abs(tscore))
freq_1$freq[2]/10000*100
```

```
## [1] 8.77
```

Now, the rejection rate is 8.77, which is lower than the significance level.

The rejection rates in items 1 and 2, 9.99 and 8.77, respectively, are very different from each other, considering that we are running ten thousand simulations and using a sample size of also ten thousand. This difference is explained by the fact that a t-distribution with a lower degree of freedom has higher variance. This reduces the t-statistic and, therefore, causes underrejection.

Question 4

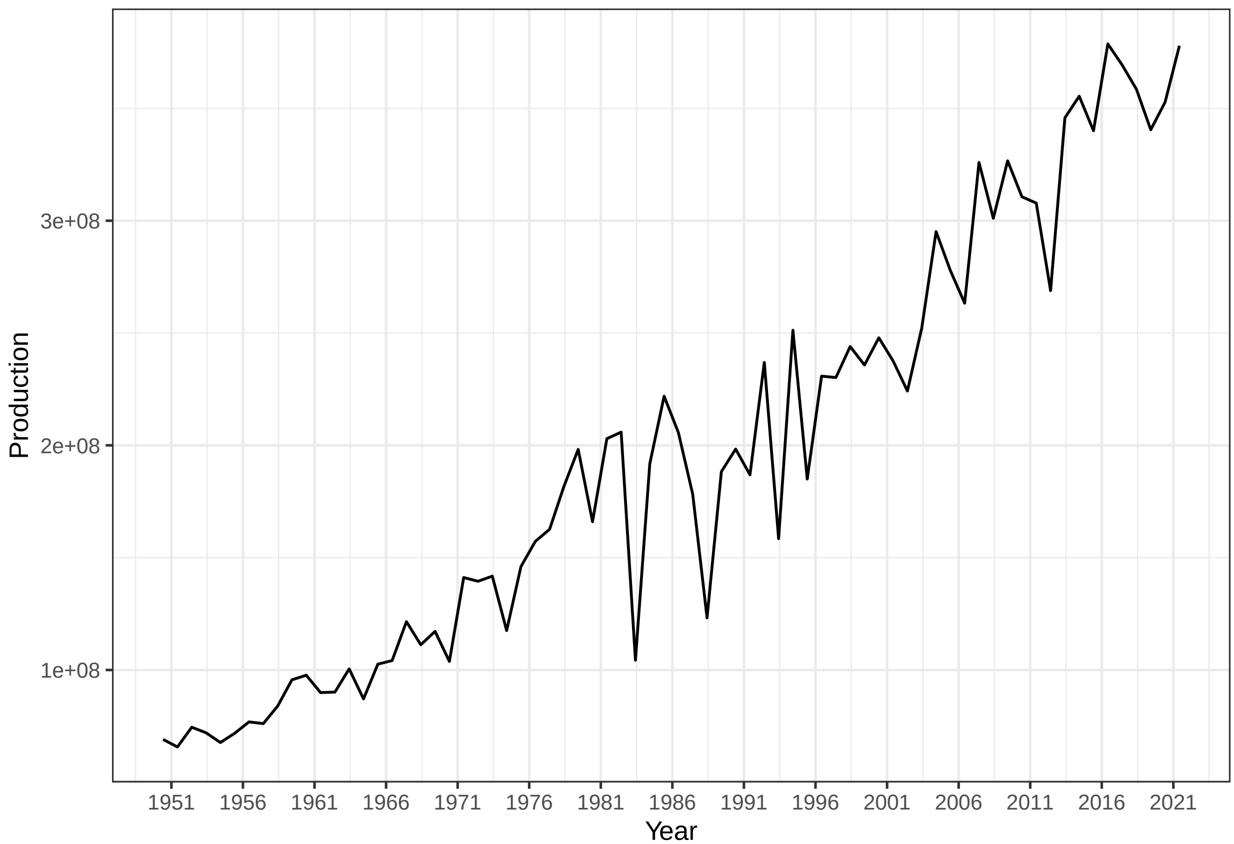
Let's load and prepare the data:

```r
corn_production <- readr::read_csv(paste0(path,"corn-production-land-us.csv")) %>%
  filter(Year >= 1950) %>%
  mutate(
    year = as.Date(as.character(Year), "%Y"),
    production = `Corn production (tonnes)`
  ) %>%
  select(year,production)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Entity = col_character(),
##   Code = col_character(),
##   Year = col_double(),
##   `Corn, area harvested (hectares)` = col_double(),
##   `Corn production (tonnes)` = col_double()
## )

## filter: removed 84 rows (54%), 72 rows remaining

## mutate: new variable 'year' (Date) with 72 unique values and 0% NA

##         new variable 'production' (double) with 72 unique values and 0% NA

## select: dropped 5 variables (Entity, Code, Year, Corn, area harvested (hectares), Corn production (to
```

To determine what kind of test we should do, let's plot the graph and do some visual analysis:

```r
production_plot <- ggplot(data = corn_production) +
  geom_line(aes(x = year, y = production)) +
  scale_x_date(name = "Year",breaks = "5 years",date_labels = "%Y") +
  scale_y_continuous(name = "Production",breaks = waiver()) +
  theme_bw(base_size = 10)

print(production_plot)
```



It looks like there is a trend, so we will use the `ur.df` function with parameter `type = "trend"`. But firstly, consider the model:

$$\Delta Y_t = \rho Y_{t-1} + \delta t + \alpha \sum_{i=1}^{p} \beta_i \Delta Y_t t - i + 1 + \epsilon_t \tag{1}$$

We have the following convention:

$(\phi 2)$ $H_0 : \rho = 1$ and $\delta = 0$ and $\alpha = 0$

$(\phi 3)$ $H_0 : \rho = 1$ and $\delta = 0$

$(\tau 3)$ $H_0 : \rho = 1$

Let's test it:

```
production_urtest <- ur.df(y = corn_production$production, type = "trend", selectlags = "BIC")

summary(production_urtest)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
```

```
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -88189259  -6955484   2877956  14633714  43297503
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.684e+07  8.353e+06   3.213  0.00203 **
## z.lag.1     -6.705e-01  1.543e-01  -4.347 4.88e-05 ***
## tt           2.982e+06  6.839e+05   4.360 4.67e-05 ***
## z.diff.lag  -1.808e-01  1.207e-01  -1.498  0.13899
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25880000 on 66 degrees of freedom
## Multiple R-squared:  0.4271, Adjusted R-squared:  0.4011
## F-statistic:  16.4 on 3 and 66 DF,  p-value: 4.537e-08
##
##
## Value of test-statistic is: -4.3468 7.8636 9.5984
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -4.04 -3.45 -3.15
## phi2  6.50  4.88  4.16
## phi3  8.73  6.49  5.47
```

Since $-4.3468 < -4.04$, $7.8636 > 6.50$ and $9.5984 > 8.73$, all of the null hypothesis defined above are reject at the 1% level. Therefore, there is no unit root under the null, but we do have a time trend and a drift.