Assignment 2: Time Series Regression

Joshua Goldberg

April, 18 2019

Data Description

```
stock_data <- read_csv("Assign 2 TS regression.csv") %>%
  mutate(date = dmy(date)) %>%
  gather(key = stock, value = return, -1)

stock_data_ts <- stock_data %>%
  as_tsibble(key = id(stock), index = date)
```

All are daily stock exchange returns.

ISE: Istanbul stock exchange national 100 index

SP: Standard & Poor™s 500 return index

DAX: Stock market return index of Germany

FTSE: Stock market return index of UK

NIKKEI: Stock market return index of Japan

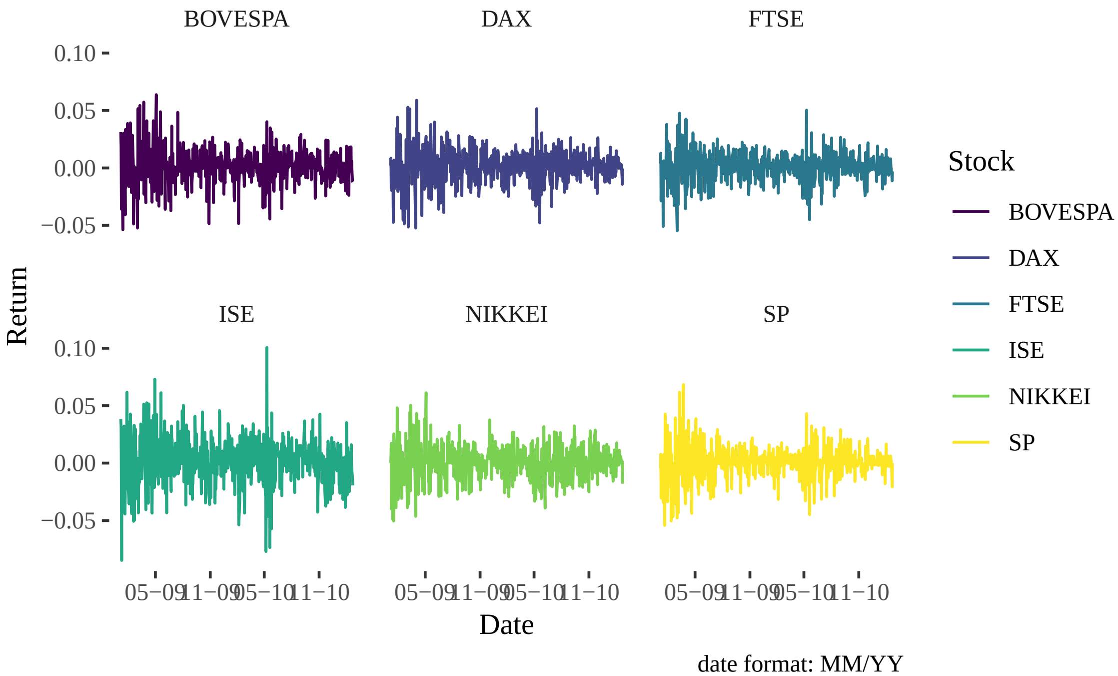
BOVESPA: Stock market return index of Brazil

Questions

Determine if all the TS are stationary:

1. qualitatively: the data for each stock all look stationary. μ and σ^2 remain constant overtime. Oscillations are offset by each other.

```
stock_data_ts %>%
  ggplot(aes(date, return, color = stock)) +
  geom_line() +
  scale_x_date(date_breaks = "6 month", date_minor_breaks = "3 month", date_labels = "%m-%y") +
  scale_color_viridis_d(name = "Stock") +
  facet_wrap( ~ stock) +
  labs(x = "Date",
       y = "Return",
       caption = "date format: MM/YY")
```

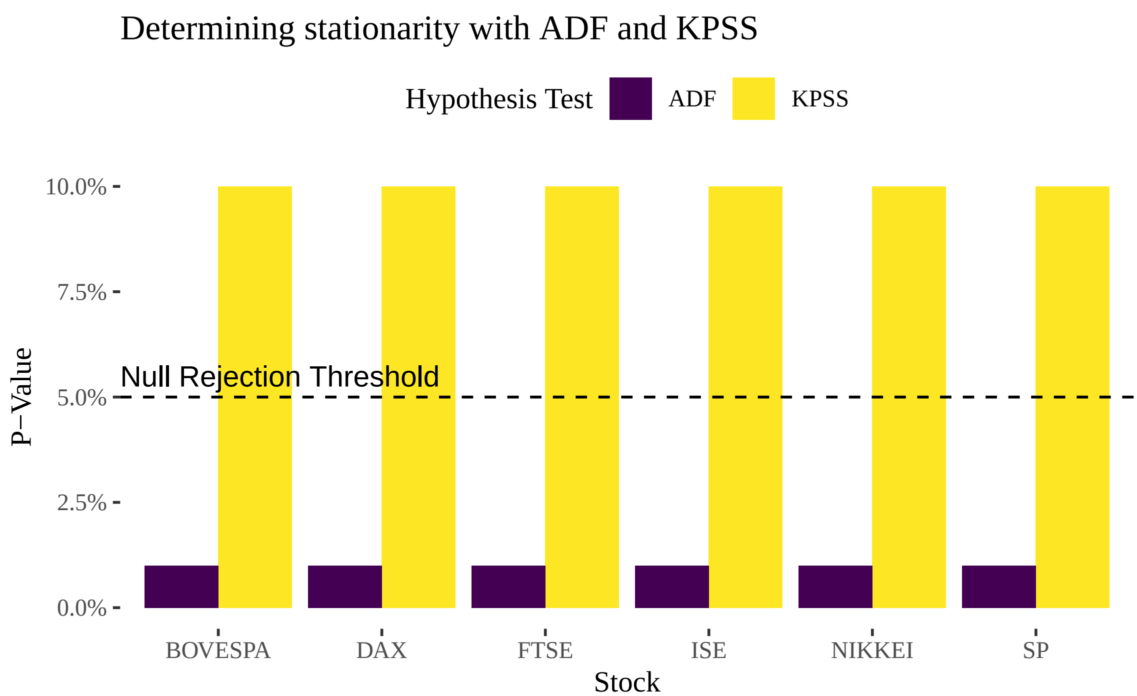


2. quantitatively: use **ADF** and **KPSS** from package tseries.

```
(stationary_tests <- stock_data_ts %>%
  nest(-stock) %>%
  mutate(adf_test = map(data, ~ suppressWarnings(adf.test(.x$return))),
         kpss_test = map(data, ~ suppressWarnings(kpss.test(.x$return))),
         adf_p_value = map_df(adf_test, ~ glance(.x)) %>% pull(p.value),
         kpss_p_value = map_df(kpss_test, ~ glance(.x)) %>% pull(p.value)))
```

```
## # A tibble: 6 x 6
##   stock   data               adf_test    kpss_test  adf_p_value kpss_p_value
##   <chr>   <list>             <list>      <list>           <dbl>        <dbl>
## 1 BOVESPA <tsibble [536 x ~ <S3: htest> <S3: htes~        0.01          0.1
## 2 DAX     <tsibble [536 x ~ <S3: htest> <S3: htes~        0.01          0.1
## 3 FTSE    <tsibble [536 x ~ <S3: htest> <S3: htes~        0.01          0.1
## 4 ISE     <tsibble [536 x ~ <S3: htest> <S3: htes~        0.01          0.1
## 5 NIKKEI  <tsibble [536 x ~ <S3: htest> <S3: htes~        0.01          0.1
## 6 SP      <tsibble [536 x ~ <S3: htest> <S3: htes~        0.01          0.1
```

```
stationary_tests %>%
  gather(key = key, value = value, -c(1:4)) %>%
  ggplot(aes(stock, value, fill = key)) +
  geom_col(position = "dodge") +
  geom_hline(yintercept = .05, linetype = 2) +
  annotate("text", -Inf, .0575, label = "Null Rejection Threshold", hjust = 0, vjust = 1) +
  scale_y_continuous(labels = scales::percent) +
  scale_fill_viridis_d(name = "Hypothesis Test", labels = c("ADF", "KPSS")) +
  labs(title = "Determining stationarity with ADF and KPSS",
       x = "Stock",
       y = "P-Value") +
  theme(legend.position = "top")
```



2. Split the data into train and test, keeping only the last 10 rows for test (from date 9-Feb-11). Remember to use only train dataset.

```
model_data <- stock_data_ts %>%
  spread(stock, return)

train <- model_data %>%
  filter(date < "2011-02-09")

test <- model_data %>%
  anti_join(train, "date")
```

3. Linearly regress ISE against the remaining 5 stock index returns. Determine which coefficients are equal or better than 0.02 (*) level of significance.

```
lm_model <- lm(ISE ~ BOVESPA + DAX + FTSE + NIKKEI + SP, data = train)
summary(lm_model)
```

```
##
## Call:
## lm(formula = ISE ~ BOVESPA + DAX + FTSE + NIKKEI + SP, data = train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.071180 -0.009248  0.000083  0.009304  0.051863
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.0008833  0.0006640   1.330 0.183979
## BOVESPA     0.1117630  0.0626647   1.784 0.075087 .
## DAX         0.3417440  0.0961243   3.555 0.000412 ***
## FTSE        0.6033493  0.1077621   5.599 3.50e-08 ***
## NIKKEI      0.3266529  0.0462163   7.068 5.09e-12 ***
```

```
## SP          -0.0607521  0.0770823  -0.788 0.430970
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0152 on 520 degrees of freedom
## Multiple R-squared:  0.493,  Adjusted R-squared:  0.4881
## F-statistic: 101.1 on 5 and 520 DF,  p-value: < 2.2e-16
```

```r
signif_vars <- function(model) {
  model %>%
    tidy() %>%
    slice(-1) %>%
    filter(p.value < .02) %>% pull(term)
}

signif_vars(lm_model)
```

```
## [1] "DAX"    "FTSE"   "NIKKEI"
```

Significant variables: DAX, FTSE, NIKKEI.

4. For the non-significant coefficients, continue to lag by 1 day until all coefficients are significant at 0.01 (*). Use `slide()` function from package **DataCombine**. Remember you will need to lag, so you slideBy = -1 each step. How many lags are needed for each independent variable?

```r
# Define shift function to take a dataframe, variable, and shift direction and return a respective data
shift_var <- function(.data, .var, .shift_by) {
  .var <- enquo(.var)
  shift_direction <- ifelse(.shift_by > 0, "lead", "lag")
  column_name <- sym(paste0(quo_name(.var), "_", shift_direction, abs(.shift_by)))

  .data %>%
    mutate(!! column_name := DataCombine::shift(!! .var, shiftBy = .shift_by, reminder = FALSE))
}

lagged_train <- train %>%
  shift_var(BOVESPA, .shift_by = -1) %>%
  shift_var(SP, .shift_by = -2)

lagged_train %>%
  select(date, contains("lag")) %>%
  head()
```

```
## # A tsibble: 6 x 3 [1D]
##   date       BOVESPA_lag1   SP_lag2
##   <date>            <dbl>     <dbl>
## 1 2009-01-05     NA        NA
## 2 2009-01-06      0.0312   NA
## 3 2009-01-07      0.0189   -0.00468
## 4 2009-01-08     -0.0359    0.00779
## 5 2009-01-09      0.0283   -0.0305
## 6 2009-01-12     -0.00976   0.00339
```

Two and one lag(s) were needed for `SP` and `BOVESPA`, respectively.

```r
lm_model_lag <- lm(ISE ~ BOVESPA_lag1 + DAX + FTSE + NIKKEI + SP_lag2, data = lagged_train)
summary(lm_model_lag)
```

```
##
## Call:
## lm(formula = ISE ~ BOVESPA_lag1 + DAX + FTSE + NIKKEI + SP_lag2,
##     data = lagged_train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.063412 -0.009491  0.000468  0.008739  0.050599
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.0007513  0.0006491   1.157 0.247635
## BOVESPA_lag1  0.2057244  0.0452856   4.543 6.91e-06 ***
## DAX           0.3355329  0.0890788   3.767 0.000184 ***
## FTSE          0.6368064  0.1024472   6.216 1.05e-09 ***
## NIKKEI        0.2395311  0.0489366   4.895 1.32e-06 ***
## SP_lag2      -0.1082670  0.0455658  -2.376 0.017861 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01481 on 518 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.516,  Adjusted R-squared:  0.5113
## F-statistic: 110.4 on 5 and 518 DF,  p-value: < 2.2e-16
```

5. Find correlations between ISE and each independent variable. Sum the square of the correlations. How does it compare to R-squared from #4?

```r
cor.test(lagged_train$ISE, lagged_train$BOVESPA)
```

```
##
##  Pearson's product-moment correlation
##
## data:  lagged_train$ISE and lagged_train$BOVESPA
## t = 11.449, df = 524, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.3762042 0.5131800
## sample estimates:
##       cor
## 0.4473112
```

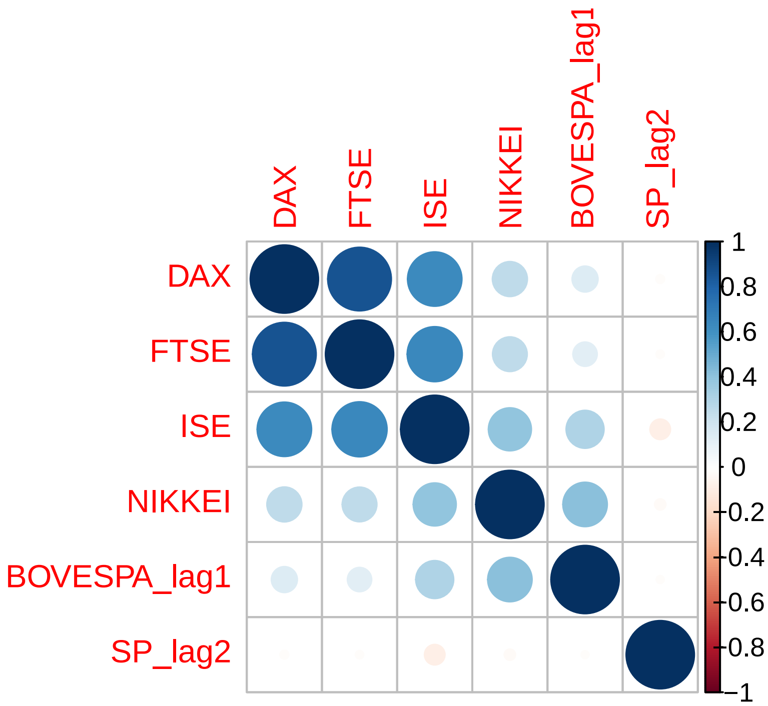
```r
vars <- c("BOVESPA_lag1", "DAX", "NIKKEI", "FTSE", "SP_lag2")

cors <- map(vars, ~ cor.test(lagged_train$ISE, lagged_train[, .x][[1]])) %>%
  map_dbl("estimate")

sum(cors^2)
```

```
## [1] 1.075558
```

Sum the square of the correlations is 1.0755584.

6. Concept question 1: why do you think the R-squared in #4 is so much less than the sum of square of the correlations? The much higher result compared to R^2 is due to collinearity between the independent variables:

```r
cor(lagged_train %>%
      .[complete.cases(.), ] %>%
      as_tibble() %>%
      select(-date, -BOVESPA, -SP) %>%
      as.matrix()) %>%
  corrplot::corrplot()
```

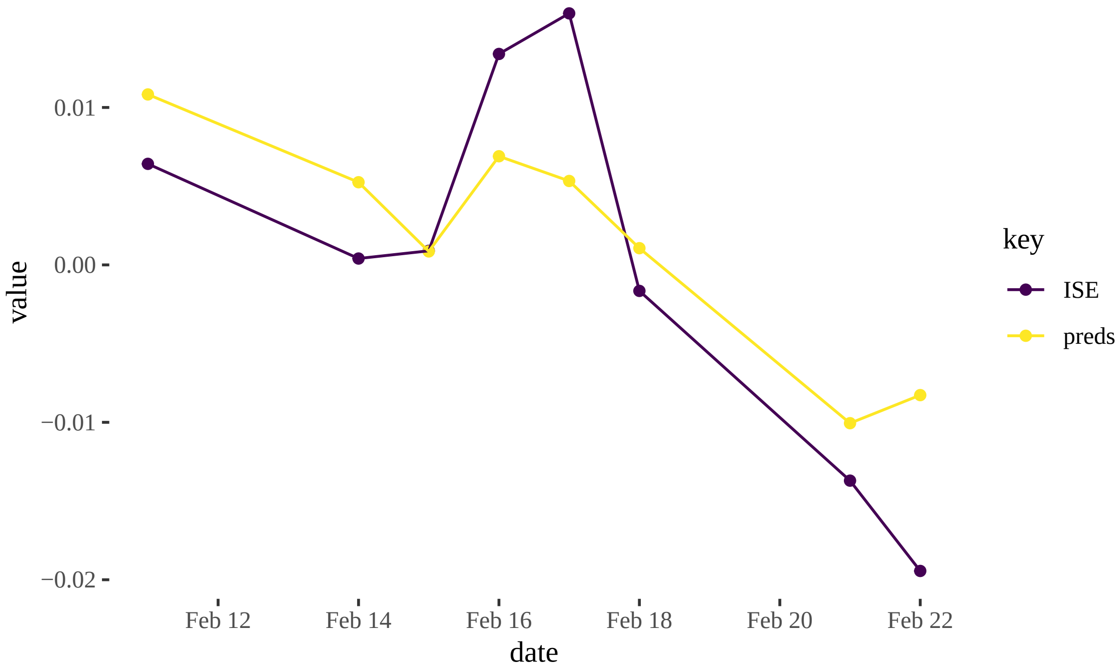


7. Take the test dataset and perform the same lags from #4 and call `predict()` function using the lm regression object from #4. Why do you need to use the lm function object from #4? Because this is the model we used for the training data with the lagged sequences, which has statistically significant variables.

```r
lagged_test <- test %>%
  shift_var(BOVESPA, .shift_by = -1) %>%
  shift_var(SP, .shift_by = -2)

test_predictions <- lagged_test[complete.cases(lagged_test), ] %>%
  mutate(preds = lm_model_lag %>% predict(newdata = lagged_test[complete.cases(lagged_test), ]),
         squared_errors = (preds - ISE)^2,
         rmse = mean(sqrt(squared_errors))) %>%
  select(date, ISE, preds, squared_errors, rmse)
```

We see that the predictions roughlt follow the trend, but not perfectly. So we have some bias in the model.

```r
test_predictions %>%
  select(-rmse, -squared_errors) %>%
  gather(key = key, value = value, -date) %>%
  ggplot(aes(date, value, color = key)) +
  geom_point() +
  geom_line() +
  scale_color_viridis_d()
```



Concept question 2: what do you find in #1 and why?

We find that both qualitatively and quantitatively that the time series are stationary. Since both of these methods agree, the conclusion is likely sound.