Homework 2

Erik Andersen

2024-04-23

Problem 1

a) The data for this problem set are from Wharton. I use the CRSP value weighted return with and without dividends.

b) In this part, I construct the dividend price ratio $\frac{D_t}{P_t}$ and the dividend growth ration $\frac{D_t}{D_{t-1}}$ using the returns. See the appendix for the code where this happens. Here I will derive how I construct these values using the given returns sequences.\ \ The formulas for the two return series I have access to are given below.

$$\text{returns with dividends} = \text{vwretd} = \frac{P_t + D_t}{P_{t-1}} - 1 \tag{1}$$

$$\text{returns no dividends} = \text{vwretx} = \frac{P_t}{P_{t-1}} - 1 \tag{2}$$

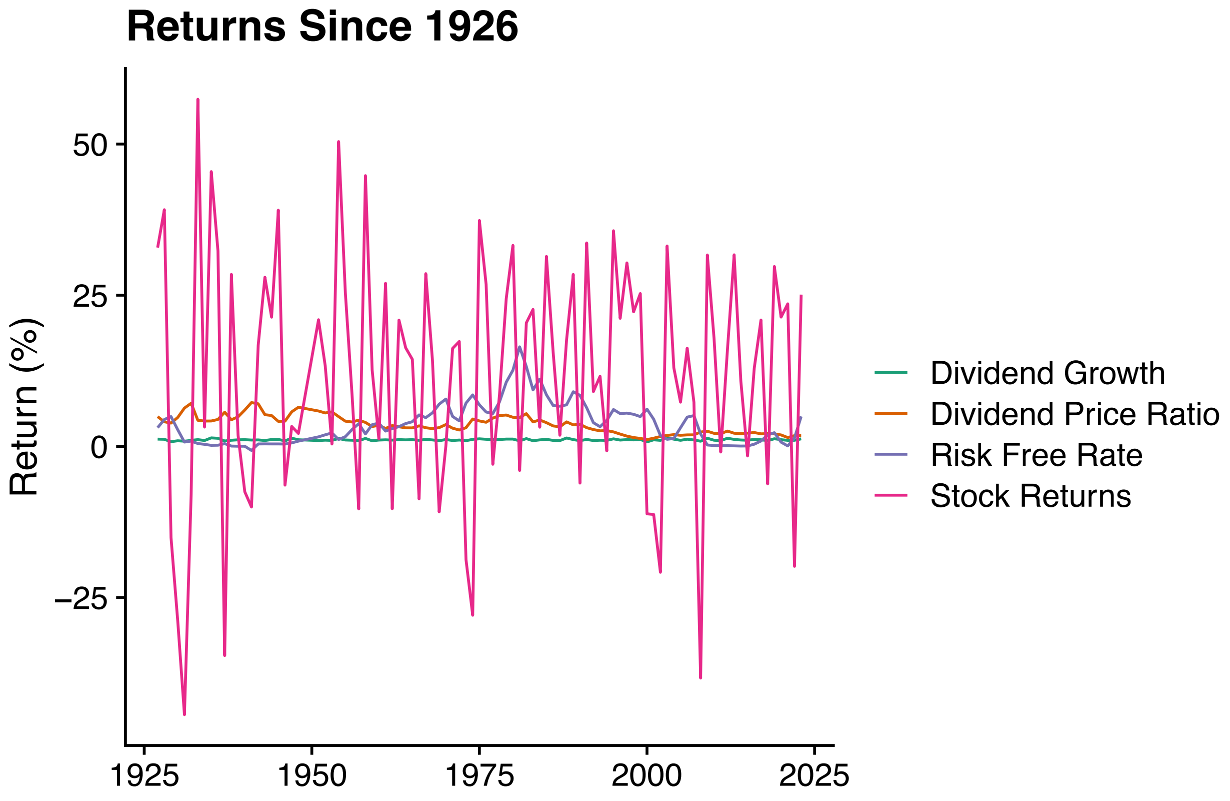
\ To construct the dividend price ratio, we will add one to (1) and (2) to remove the annoying constant, then divide (1) by (2). Finally, we subtract 1 from the ratio.

$$\begin{aligned}&\frac{\frac{P_t+D_t}{P_{t-1}}}{\frac{P_t}{P_{t-1}}} - 1\\ =&\frac{P_t + D_t}{P_t} - 1\\ =&\frac{D_t}{P_t}\end{aligned}$$

\ Using this we can now construct the dividend growth rate. To get that, we divide the dividend price ratio at time t+1 by the current period dividend price and multiply by returns with no dividends.

$$\begin{aligned}&\frac{\frac{D_{t_1}}{P_{t_1}}}{\frac{D_t}{P_t}} * \frac{P_{t_1}}{P_t}\\ =&\frac{D_{t_1}P_t}{D_tP_{t_1}}\frac{P_{t+1}}{P_t}\\ =& \frac{D_{t+1}}{D_t}\end{aligned}$$

c) Below is the graph of dividend growth, dividend price ratio, risk free return, and stock returns between 1926 and 2023. I only include the stock returns with dividends because including both is redundant. The noise of the stock returns drowns out everything else, but if you squint we see the characteristic shape of the dividend price ratio we saw in class.



Problem 2

In this problem, I will reproduce the following forecasting regressions and see how the standard errors differ under various different specifications.

$$
\begin{aligned}
R_{t,t+j} &= a + b\left(\frac{D}{P}\right)_t + \epsilon_{t+1} \\
R_{t,t+j} - R^f_{t,t+j} &= a + b\left(\frac{D}{P}\right)_t + \epsilon_{t+1} \\
\Delta D_{t,t+j} &= a + b\left(\frac{D}{P}\right)_t + \epsilon_{t+1}
\end{aligned}
$$

a) Below are the coefficients and R^2 values for the regressions with time horizons varying from 1 to 10 years. First I'll look at the first regression on returns. Ignore the far too many digits I don't know how to reduce them. Notice that both R^2 and β increase as the horizon increases. Both things indicate that over longer horizons the price dividend ratio explains more of the variation in returns, and has a larger impact on returns.

Table 1: Returns Regression

	Horizon Lengths									
	1	2	3	4	5	6	7	8	9	10
Beta	2.44366653695284	4.76917987127556	7.50747127768786	10.4028238913409	13.2904255210478	17.3727535070467	22.6101265956181	28.1463465465393	35.4053953663635	43.3012843295402
R-Squared	0.0153989141080492	0.0375215959476639	0.0588451697676149	0.0791352848813604	0.0981157192571636	0.124398473075757	0.148624756711374	0.169458090530031	0.194238057626269	0.200288660509278

Next I'll look at the values for the second regression which looks at excess returns above the risk free rate.

The pattern is the same here. In the short run, the dividend price ratio does a bad job predicting returns, but as the horizon gets longer it improves.

Table 2: Excess Returns Regression

	Horizon Lengths									
	1	2	3	4	5	6	7	8	9	10
Beta	1.6203247961923	2.01419659773439	4.42056447809811	3.28157901338726	5.03164925663726	4.31013813047258	8.78758422709141	5.20312486122837	17.2137754786496	9.69566606868518
R-Squared	0.00572323243466721	0.0128319459924064	0.0211054068562909	0.0160186108987976	0.0169581260894565	0.0146928811691185	0.0242628120414151	0.0109641116592573	0.0478334972438223	0.0191706921224648

Finally, we'll look at the final regression which looks at dividend growth. This time the pattern is very different. The R^2 decreases basically to 0 as the horizon increases. the coefficients remain around the same size and are very imprecisely estimated. We learn from this that at no horizon is the dividend price ratio predictive of dividend growth which contradicts the old style model of finance where all shifts in returns are driven by changes in dividends.

Table 3: Dividend Growth Regression

	Horizon Lengths									
	1	2	3	4	5	6	7	8	9	10
Beta	1.00785531020043	0.42194226644204	0.514971656622504	0.847141576007752	1.33228862052109	1.17966805352659	1.30496800379356	1.70805569366573	1.06698798046206	-0.141622233030912
R-Squared	0.00643352455524883	0.000823805122841135	0.000801401124377455	0.00166192562998034	0.0035663820065183	0.00239098921350862	0.0024358712957306	0.00361275082365127	0.00121014404203085	1.64022629212497e-05

b) The coefficients and standard errors are biased by the overlapping data. It is entirely reasonable to expect that there is strong autocorrelation in the time series. OLS is not robust to autocorrelated errors, so both estimates will be biased.

c) Now we will compare three different types of standard errors for the first regression. I will compare the standard OLS standard errors to Hansen-Hodrick errors, and OLS errors if we estimate the regression by omitting the overlapping data. The first table shows the standard errors over the various time horizons. Notice that the standard OLS errors are too small as the horizon gets large because of the autocorrelation. The Hansen-Hodrick errors correct for this and alre larger at every horizon and especially larger as the horizon gets large. The non-overlapping errors increase much faster than either of the other two. This is probably because they run out of data quickly. At the 10 year horizon for example, there are only 10 data points to use (1926-2016 by 10). This is probably not the best way to correct for the bias since it throws away so much data.

Table 4: Standard Errors

	Horizon Lengths									
	1	2	3	4	5	6	7	8	9	10
OLS	2.0047731907849	2.4913492942349	3.11334560068052	3.69973113055147	4.22400382385755	4.85839849176364	5.7361819822464	6.64248092980196	7.7311844435551	9.33017948260699
Hansen-Hodrick	2.41356771085058	3.13810526171058	4.18319430435705	5.41789880288018	7.8882055166958	9.94027831581315	11.4950646915258	14.559625687556	15.9622316429848	18.2297535552623
No Overlap	2.0047731907849	3.24752329859966	4.53744196937369	7.16944738781994	8.58044515783476	11.447695005446	18.1002420185932	16.858141745642	28.1117205574785	41.4091991274208

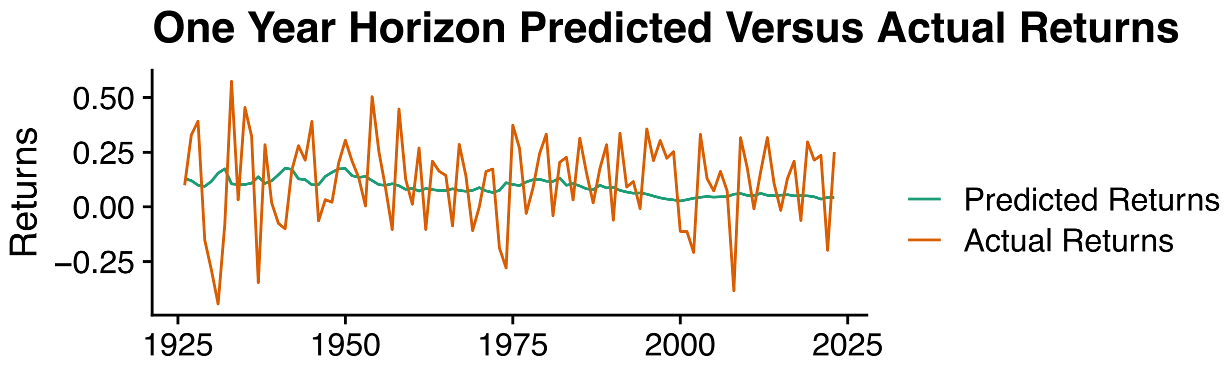
Next, we'll look a the T-statistics. What we can notice here is that using the standard OLS errors leads to us failing to reject the null of no predictive effect at almost every time horizon. Starting at the three year horizon, the t-stats are above 2, so all of them are significant at the 5% level. However, using the standard errors robust to the autocorrelation, we never reject the null except barly at the longest horizons with the Hansen-Hodrick errors.

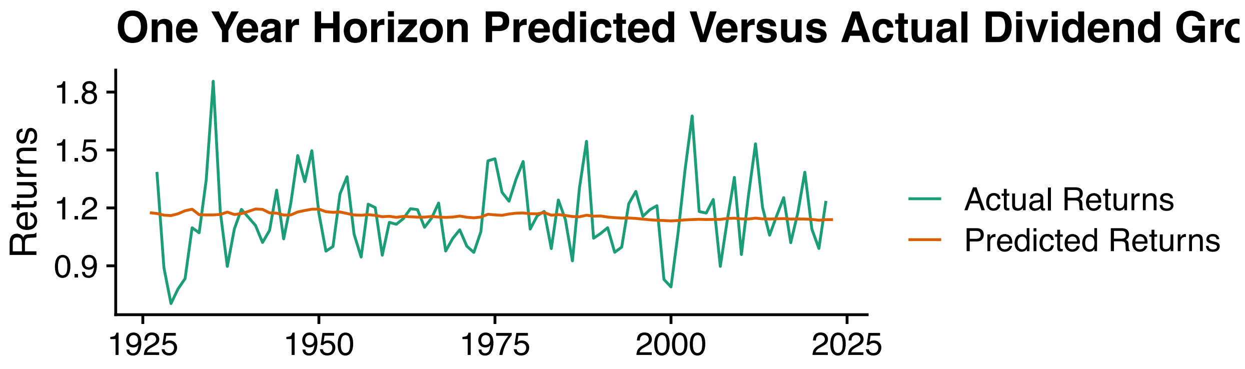
Table 5: T Statistics

	Horizon Lengths									
	1	2	3	4	5	6	7	8	9	10
OLS	1.21892418962172	1.91429595292465	2.4113838425284	2.81177834936084	3.14640470872264	3.57581897337126	3.94166828486211	4.23732440393749	4.57955642176902	4.64099157044739
Hansen-Hodrick	1.01247067814462	1.51976414859834	1.79467429229103	1.92008456965101	1.68484777595081	1.74771298701062	1.96694209231257	1.93317789554134	2.21807302125726	2.37530826723879
No Overlap	1.21892418962172	1.4685590934273	1.65456028492726	1.45099382541172	1.5489202805419	1.51757655133038	1.24916156217094	1.66959958999131	1.25945316274655	1.045692388213

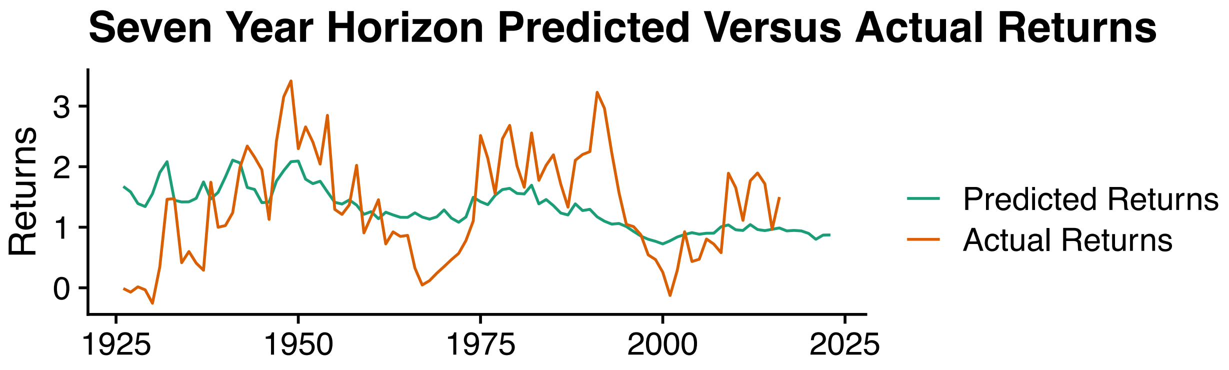
Problem 3

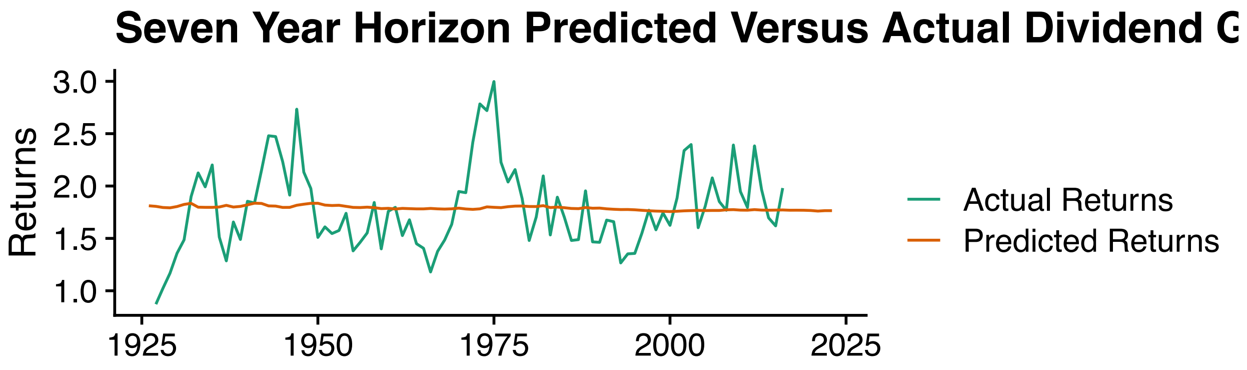
Now that we've run the regressions to see how well the dividend price ratio can predict things and done some statistics, we can visually inspect how well our predictions do by graphing them. First we'll look at our predictions for returns and dividend growth using the one year time horizon. The graphs are below. We see that the dividend price ratio looks like it can do something to predict returns, but nothing at all to predict dividend growth. The line is basically flat.





Now we'll look at the same thing but over a 7 year horizon. Now the returns look pretty well predicted by the dividend price ratio. It doesn't capture all of the noise, but it tracks the broad movements. Dividend growth still is not at all predictable and the line is still basically flat.





Problem 4

In this question, I will estimate the variance decomposition of the price dividend ratio. See the code appendix for the calculation. For the dividend part of the variation, I get a value of -16.4686 which matches fairly closely the value from table 20.3. It is slightly lower but that could be due to the changing macro environment since the table was created. I get a value of 143.0285 for the portion of the variation due to returns which matches the table almost exactly. Matching Cochrane's analysis, I find that variation in returns explains basically all the variation in the price dividend ratio.

Code Appendix

Erik Andersen

2024-04-18

```r
here::i_am("HW2/code/HW2.Rmd")
```

```
## here() starts at /Users/erikandersen/Documents/Classes/FIN-592
```

```r
# Load packages
pacman::p_load(tidyverse, magrittr, sandwich, kableExtra)
```

```r
# Load data
equity_df = read_csv(here::here("HW2", "data", "equity_returns.csv"))
```

```
## Rows: 99 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (2): vwretd, vwretx
## date (1): caldt
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
tbill_df = read_csv(here::here("HW2", "data", "tbill_returns.csv"))
```

```
## Rows: 99 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (1): t90ret
## date (1): caldt
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Merge into one data set
returns_df = left_join(equity_df, tbill_df) |>
  mutate(date = year(caldt)) |> # Extract the year
  select(date, everything(), -caldt) |>
  filter(!is.na(vwretd)) # Remove 1925 because it has no data
```

```
## Joining with `by = join_by(caldt)`
```

```r

# Construct dividend price ratio from equity returns data
# To get there, we will do the following. First, subtracting returns without dividends from returns wit

# Dividend growth we can construct by multiplying current dividend price by returns without dividends d

returns_df =
  returns_df |>
    mutate(div_price = (vwretd + 1) /(vwretx + 1) - 1,
           div_growth = div_price / lag(div_price) + vwretx)
```

b)

```
# Plot dividend price, dividend returns, stock returns, and risk free returns on the same graph

returns_plot = returns_df |>
  drop_na() |>
  mutate(across(-c(date, div_growth), function(x) x*100)) |> # Convert to %s
  select(-c(vwretx)) |> # Its redundant to plot returns with and without dividends
  pivot_longer(cols = -date) |>  # Rearrange the data so it makes a nice legend
  ggplot(aes(x = date, y = value, color = name)) +
  geom_line() +
  cowplot::theme_cowplot() +
  scale_color_brewer(name = "", palette = "Dark2", labels = c("Dividend Growth", "Dividend Price Ratio"
  labs(y = "Return (%)", x = "", title = "Returns Since 1926" )

ggsave(here::here("HW2", "plots", "returns.pdf"))
```

c)

```
## Saving 6.5 x 4.5 in image
```

Question 2

```
# Calculate cumulative returns for every horizon from 1 to 10 years.
# I couldn't come up with a good way to do this so I'm doing it manually
horizons_return_df = returns_df |>
  mutate(demean = vwretd + 1,
         r1 = demean * lead(demean)-1,
         r2 = (r1 + 1)*lead(demean,2)-1,
         r3 = (r2 + 1)*lead(demean,3)-1,
         r4 = (r3 + 1)*lead(demean,4)-1,
         r5 = (r4 + 1)*lead(demean,5)-1,
         r6 = (r5 + 1)*lead(demean,6)-1,
         r7 = (r6 + 1)*lead(demean,7)-1,
         r8 = (r7 + 1)*lead(demean,8)-1,
         r9 = (r8 + 1)*lead(demean,9)-1,
         r10= (r9 + 1)*lead(demean,10)-1)

# Calculate the regressions for various return horizons on dividend price ratio
return_horizons = sapply(paste0("r", 1:10), function(x){
  temp = horizons_return_df %>%
    lm(paste(x, "~", "div_price"),.) |> summary()

  # Extract relevant stats
  ret = tibble(intercept = coef(temp)[1,1], beta = coef(temp)[2,1], se = coef(temp)[2,2], t = coef(temp

  return(ret)
})
colnames(return_horizons) = 1:10
```

```r
rownames(return_horizons) = c("Intercept", "Beta", "Standard Error", "T-Stat", "R-Squared")

# Add excess returns
horizons_return_df %<>%
  mutate(rf = t90ret + 1,
         rf1 = r1 - (rf * lead(rf)-1),
         rf2 = r2 - (rf1 + 1) * lead(rf,2)-1,
         rf3 = r3 - (rf2 + 1) * lead(rf,3)-1,
         rf4 = r4 - (rf3 + 1) * lead(rf,4)-1,
         rf5 = r5 - (rf4 + 1) * lead(rf,5)-1,
         rf6 = r6 - (rf5 + 1) * lead(rf,6)-1,
         rf7 = r7 - (rf6 + 1) * lead(rf,7)-1,
         rf8 = r8 - (rf7 + 1) * lead(rf,8)-1,
         rf9 = r9 - (rf8 + 1) * lead(rf,9)-1,
         rf10=r10 - (rf9 + 1) * lead(rf,10)-1)

# Calculate regressions
risk_free_horizons = sapply(paste0("rf", 1:10), function(x){
  temp = horizons_return_df %>%
    lm(paste(x, "~", "div_price"),.) |> summary()

  # Extract relevant stats
  ret = tibble(intercept = coef(temp)[1,1], beta = coef(temp)[2,1], se = coef(temp)[2,2], t = coef(temp

  return(ret)
})
colnames(risk_free_horizons) = 1:10
rownames(risk_free_horizons) = c("Intercept", "Beta", "Standard Error", "T-Stat", "R-Squared")

# Add Dividend growth
horizons_return_df %<>%
  mutate(d1 = div_growth * lead(div_growth),
         d2 = d1 * lead(div_growth,2),
         d3 = d2 * lead(div_growth,3),
         d4 = d3 * lead(div_growth,4),
         d5 = d4 * lead(div_growth,5),
         d6 = d5 * lead(div_growth,6),
         d7 = d6 * lead(div_growth,7),
         d8 = d7 * lead(div_growth,8),
         d9 = d8 * lead(div_growth,9),
         d10= d9 * lead(div_growth,10))

# Calculate regrssions
div_growth_horizons = sapply(paste0("d", 1:10), function(x){
  temp = horizons_return_df %>%
    lm(paste(x, "~", "div_price"),.) |> summary()

  # Extract relevant stats
  ret = tibble(intercept = coef(temp)[1,1], beta = coef(temp)[2,1], se = coef(temp)[2,2], t = coef(temp

  return(ret)
})
colnames(div_growth_horizons) = 1:10
```

```r
rownames(div_growth_horizons) = c("Intercept", "Beta", "Standard Error", "T-Stat", "R-Squared")
```

```r
# Calculate the Hansen Hoddrick standard errors
hansen_hodrick = sapply(paste0("r", 1:10), function(x){
  temp = horizons_return_df %>%
    lm(paste(x, "~", "div_price"),.) |>
    kernHAC(kernel = 'Truncated', prewhite = F, adjust = T, sandwich = T)

  se = tibble(se = sqrt(temp[2,2]))

  return(se)
}) |> unlist()

# Calcualte t-stats
hh_t = as.numeric(return_horizons[2,])/hansen_hodrick
```

```r
# Calculate non-overlapping standard errors
# What we're doing in this loop is subsetting the data into chunks that are the length of the return we
no_overlap = sapply(1:10, function(i){
  temp = horizons_return_df %>%
    # This next bit is complicated and dense because it uses non dplyr fuctions so they don;t work in a
    # Starting from the inside, first we find the row number in the dataset. Why its only the date colu
    # Take the modulo of that for each legth of time. Then select only those rows for which that equals
    slice(which((row_number(horizons_return_df$date) -1) %% i == 0)) %>%
    lm(paste0("r", i, "~", "div_price"),.) |>
    summary() |>
    coef()

  se = temp[2,2]

  return(tibble(se))

}) |> unlist()

# Calculate t-stat
no_overlap_t = as.numeric(return_horizons[2,])/no_overlap
```

c)

Question 3

```r
# Predict returns for one year horizon using the returns_horizon object and plot
one_year_returns = horizons_return_df |>
  mutate(predicted_returns = as.numeric(return_horizons[2,1]) * div_price) |>
  select(date, vwretd, predicted_returns) |>
  pivot_longer(cols = -date) |>
  ggplot(aes(x = date, y = value, color = name)) +
  geom_line() +
  scale_color_brewer(palette = "Dark2", labels = c("Predicted Returns", "Actual Returns")) +
  labs(y = "Returns", x = "", color = "") +
  ggtitle("One Year Horizon Predicted Versus Actual Returns") +
  cowplot::theme_cowplot()
```

```r
ggsave(here::here("HW2", "plots", "one_year_returns.png"))
```

```
## Saving 6.5 x 4.5 in image
```

```r
# Same thing but for dividend growth
# Predict returns for one year horizon using the div_growth_horizons object and plot
one_year_div = horizons_return_df |>
  mutate(predicted_div_growth = as.numeric(div_growth_horizons[1,1]) +  as.numeric(div_growth_horizons[
  select(date, d1, predicted_div_growth) |>
  pivot_longer(cols = -date) |>
  ggplot(aes(x = date, y = value, color = name)) +
  geom_line() +
  scale_color_brewer(palette = "Dark2", labels = c("Actual Returns", "Predicted Returns")) +
  labs(y = "Returns", x = "", color = "") +
  ggtitle("One Year Horizon Predicted Versus Actual Dividend Growth") +
  cowplot::theme_cowplot()

ggsave(here::here("HW2", "plots", "one_year_div.png"))
```

```
## Saving 6.5 x 4.5 in image

## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```r
# Now graph the same things but over 7 year horizon
seven_year_returns = horizons_return_df |>
  mutate(predicted_returns = as.numeric(return_horizons[1, 7]) + as.numeric(return_horizons[2, 7]) * di
  select(date, r7, predicted_returns) |>
  pivot_longer(cols = -date) |>
  ggplot(aes(date, value, color = name)) +
  geom_line() +
  scale_color_brewer(palette = "Dark2", labels = c("Predicted Returns", "Actual Returns")) +
  labs(y = "Returns", x = "", color = "") +
  ggtitle("Seven Year Horizon Predicted Versus Actual Returns") +
  cowplot::theme_cowplot()

ggsave(here::here("HW2", "plots", "seven_year_returns.png"))
```

```
## Saving 6.5 x 4.5 in image

## Warning: Removed 7 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```r
# Finally seven year horizon for dividend growth
seven_year_div = horizons_return_df |>
  mutate(predicted_div_growth = as.numeric(div_growth_horizons[1,7]) +  as.numeric(div_growth_horizons[
  select(date, d7, predicted_div_growth) |>
  pivot_longer(cols = -date) |>
  ggplot(aes(x = date, y = value, color = name)) +
  geom_line() +
  scale_color_brewer(palette = "Dark2", labels = c("Actual Returns", "Predicted Returns")) +
  labs(y = "Returns", x = "", color = "") +
  ggtitle("Seven Year Horizon Predicted Versus Actual Dividend Growth") +
  cowplot::theme_cowplot()

ggsave(here::here("HW2", "plots", "sevel_year_div.png"))
```

```
## Saving 6.5 x 4.5 in image
```

```
## Warning: Removed 8 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

Question 4

```r
# Construct variance decomposition for the price dividend ratio. First we'll do the part for dividends
#
var_dividends = sapply(1:nrow(horizons_return_df), function(i){ # Loop over each row
  # Select each row because we want to calculate the discounted sum of dividend growth for each time pe
  temp = horizons_return_df[i,]
  sapply(1:10, function(j){ # Question asks for 10 year horizon
    # We already calculated the dividend growth up to 10 years indexed by d1:d10, so we just discount t
    as.numeric(0.96^(j-1) * temp[paste0("d", j)])
  }) |> sum()
})

# Calculate the relevant statistic
# We have to take the negative log of dividend price because that's that the table uses
div_decomp = 100 *cov(var_dividends,
                 -log(horizons_return_df$div_price),
                 # Complete.obs gets rid of all the NAs
                 "complete.obs") / var(-log(horizons_return_df$div_price), na.rm = T)
```

```r
# Do the same thing for returns
var_returns = sapply(1:nrow(horizons_return_df), function(i){
  temp = horizons_return_df[i,]
  sapply(1:10, function(j){
    # Only difference is we iterate over returns indexed by r1:r10
    as.numeric(0.96^(j-1) * temp[paste0("r", j)])
  }) |> sum()
})

# Calculate stat
return_decomp = cov(var_returns,
                    horizons_return_df$div_price,
                    "complete.obs") / var(horizons_return_df$div_price, na.rm = T)
```