Practicum 2

Kieran Douglas

2. Examining Waugh's 1927 Asparagus Data

A.

We can see that the greatest difference between coeff estimates and those reported in the question are in the number of stalks and the variation in size (nostalks and disp). These end up being an absolute difference of 0.1767 and 0.0697 respecitvely.

```
# Set up environment and load data
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```
library(readr)
library(fixest)

waugh <- read_table("/Users/kieran/Documents/MASTERS/METRICS/code/metrics/practicum_2_files/w
```

```
-- Column specification --------------------------------------------------------
cols(
```

```
  GREEN = col_double(),
  NOSTALKS = col_double(),
  DISPERSE = col_double(),
  PRICE = col_double()
)

Warning: 200 parsing failures.
row col  expected    actual
  1  -- 4 columns 5 columns '/Users/kieran/Documents/MASTERS/METRICS/code/metrics/practicum_2
  2  -- 4 columns 5 columns '/Users/kieran/Documents/MASTERS/METRICS/code/metrics/practicum_2
  3  -- 4 columns 5 columns '/Users/kieran/Documents/MASTERS/METRICS/code/metrics/practicum_2
  4  -- 4 columns 5 columns '/Users/kieran/Documents/MASTERS/METRICS/code/metrics/practicum_2
  5  -- 4 columns 5 columns '/Users/kieran/Documents/MASTERS/METRICS/code/metrics/practicum_2
... ... ......... ......... ..............................................................
See problems(...) for more details.
```

```
waugh_clean <- waugh |>
  rename(
    green = "GREEN",
    nostalks = "NOSTALKS",
    disp = "DISPERSE",
    price = "PRICE"
  ) |>
   mutate(
      green = as.numeric(green),
      nostalks = as.numeric(nostalks),
      disp = as.numeric(disp),
      price = as.numeric(price)
    )

# Run a MLR
model1 = lm(price ~ green+nostalks+disp, data = waugh_clean)
summary(model1)
```

```
Call:
lm(formula = price ~ green + nostalks + disp, data = waugh_clean)

Residuals:
    Min      1Q  Median      3Q     Max
-56.004  -9.485  -0.122   9.422  49.097
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 40.761264   5.327837   7.651 8.82e-13 ***
green        0.137598   0.007099  19.382  < 2e-16 ***
nostalks    -1.357256   0.150822  -8.999  < 2e-16 ***
disp        -0.345283   0.129656  -2.663  0.00839 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 15.52 on 196 degrees of freedom
Multiple R-squared:  0.7268,    Adjusted R-squared:  0.7226
F-statistic: 173.8 on 3 and 196 DF,  p-value: < 2.2e-16
```

```r
# compare coefficient estimates
coeffs = coefficients(model1)
original_coeffs = c(green = 0.13826, nostalks = -1.53394, disp = -0.27554)
differences <- coeffs[names(original_coeffs)] - original_coeffs
print(differences)
```

```
        green      nostalks          disp
-0.0006617547  0.1766836397 -0.0697428425
```

```r
# We can see that the greatest difference between coeff estimates and those reported in the 
```

B.

It looks like the main issue here is that Waugh has transformed green color on the asparagus variable to inches rather than what I have in the raw data, being hundredths of inches. To fix this I need to rescale the variable green by dividing it by 100. After doing this the difference in means is no longer a problem.

```r
means <- c(price_avg = mean(waugh_clean$price), green_avg = mean(waugh_clean$green), nostalks
print(means)
```

```
   price_avg    green_avg nostalks_avg     disp_avg
      90.095      588.750       19.555       14.875
```

```r
means_reported <- c(price_avg = 90.095, green_avg = 5.8875, nostalks_avg = 19.555, disp_avg =
differences_avg <- means[names(means_reported)] - means_reported
print(differences_avg)
```

```
    price_avg     green_avg  nostalks_avg      disp_avg 
 0.000000e+00  5.828625e+02 -3.552714e-15  0.000000e+00 
```

```
# It looks like the main issue here is that Waugh has transformed green color on the asparagu
waugh_clean <- waugh_clean |>
  mutate(
    green = green/100
  )
# Recheck mean differences
means_recheck <- c(price_avg = mean(waugh_clean$price), green_avg = mean(waugh_clean$green),
print(means_recheck)
```

```
   price_avg    green_avg nostalks_avg     disp_avg 
     90.0950       5.8875      19.5550      14.8750 
```

C.

I notice relative variance in the relative size of the covariances reported. For example, the covariance between price and green according to Waugh is ~3430 while the one I found was 3448 (larger). On the other hand, Waugh found a covairance of ~-154 for green and disp while I found one of ~-180 (mine was smaller). In other cases though mine was larger like with the cov between green and price. I think that the pattern in differences may relate to the way my price variable is coded relative to Waugh's. This would explain the constant differences in price covariance versus the more consistent findings in ohter categories.

```
# I am now going to create a variance covariance matrix similar to Waugh's`
moments_subset <- waugh_clean[, c("price", "green", "nostalks", "disp")]
# to match the layout of the table provided I will modify green to be back to hundredths of a
moments_subset <- moments_subset |>
  mutate(
    green = green*100
  )
# Create the matric, format it correctly, and print
matrix <- cov(moments_subset)
print(matrix)
```

```
             price       green  nostalks       disp
price     868.73967  3448.18467 -93.38465  -87.43028
green    3448.18467 24439.38442 -17.09171 -180.05653
nostalks  -93.38465   -17.09171  60.73063   24.92399
disp      -87.43028  -180.05653  24.92399   83.48681
```

```r
matrix_fmt <- formatC(matrix, format="f", digits=1)
matrix_fmt[lower.tri(matrix_fmt)] <- ""
print(noquote(matrix_fmt))
```

```
         price green   nostalks disp  
price    868.7 3448.2  -93.4    -87.4 
green          24439.4 -17.1    -180.1
nostalks               60.7     24.9  
disp                            83.5  
```

D.

It is plausible that the principle findings concerning the effects of covariate variation could be affected in terms of the precision and statistical inference of coefficient estimates if there was underestimation or misreporting of variance and covariance among regressors. Depsite OLS estimates remaining unbiased under GM, Waugh's under or over-estiamted variance and coveriance could lead to underestiated standard errors and possibly inacurate significance. Similarly, cov among the regressors included may inflate the standard errors and lead to reductions in the validity/reliability of estimates. The difference in scaled coefficients between Waugh's and my own analysis do not differ too much, but there are some clear differences. For example, my estimate for the marginal effect of a unit increase in the number of stalks per bunch is about \$0.5 less than Waugh's. I estimate a slightly higher increase in price given a unit increase in green, and a roughly \$0.2 larger decrease in price given a unit increase in size variation. Overall, our estimates are very similar. I find the association between both green and nostalks with price to be highly statistically significant at the $p<0.001$ level and the association between disp and price to be statistically signifiacnt at $\mathbf{0.001<p<0.01}$ level. All of these observations are ceterus parabus.

::: {.cell}

```r
 # run model
coef(model1) * 2.782
```

::: {.cell-output .cell-output-stdout}

```
(Intercept)       green    nostalks        disp 
113.3978351   0.3827983  -3.7758872  -0.9605769 
```

:::

```r
waugh_scaled <- original_coeffs*2.782
print(waugh_scaled)
```

::: {.cell-output .cell-output-stdout}

```
     green   nostalks       disp 
 0.3846393 -4.2674211 -0.7665523 
```

:::

```r
# The difference in scaled coefficients between Waugh's and my own analysis do not differ too
```

:::

E.

To solve this I am going to experiment with a matrix.

```r
# my matrix solutions
# Extract covariance matrix of regressors (columns and rows 2 to 4)
xx <- matrix[2:4, 2:4]
# Extract covariance vector of PRICE with regressors (row 1, columns 2 to 4)
xy <- matrix[1, 2:4]
# Compute beta_hat = solve(Sigma_XX) %*% Sigma_XY
beta_hat <- solve(xx) %*% xy
print(beta_hat)
```

```
               [,1]
green     0.1375982
nostalks -1.3572564
disp     -0.3452828
```

```r
# Waugh's matrix solutions
vc_mat <- matrix(c(
  1063.64, 3430.89, -100.92, -82.35,
  3430.89, 24317.19, -17.01, -154.54,
  -100.92, -17.01, 61.33, 25.51,
  -82.35, -154.54, 25.51, 83.07
), nrow = 4, byrow = TRUE)
```

```r
# names for reference
row_col_names <- c("price","green","nostalks","disp")
dimnames(vc_mat) <- list(row_col_names, row_col_names)
# price-cov covariance vector
Sigma_yX <- vc_mat["price", c("green", "nostalks", "disp")]
# covariate-covariance matrix
Sigma_XX <- vc_mat[c("green", "nostalks", "disp"), c("green", "nostalks", "disp")]
# raw coefficients from matrix
beta_raw <- solve(Sigma_XX, Sigma_yX)
names(beta_raw) <- c("green", "nostalks", "disperse")
# scale
beta_final <- beta_raw * 2.782
beta_final
```

```
    green   nostalks   disperse
 0.3847304 -4.1520749 -0.7670883
```

```r
# Computing the least squares estimates from Waugh's VC matrix table, we can see that his es
```

3. Exploring Relationships among R^2, Coefficients of Determination, and Correlation Coefficients

A.

Based on the cor matrix, it is evident that the most highly correlated variables are price and green, price and nostalks, nostalks and disp, and price and disp (ordered from highest to lowest positive and negative correlation). The most orthogonal correlations are between green and nostalks and green and disp becasue each of their correlations is quite close to zero, thus implying low to no real linear relationship between the variables based on the data we have available.

```r
# create cor matrix
order <- c("price", "green", "nostalks", "disp")
cor_matrix <- cor(waugh_clean)
cor_matrix <- cor_matrix[order, order]

# round the matrix
cor_matrix_rounded <- round(cor_matrix, 5)
```

```
# lower triangle with empty strings
cor_matrix_char <- format(cor_matrix_rounded, nsmall = 5)
cor_matrix_char[lower.tri(cor_matrix_char)] <- ""

print(cor_matrix_char, quote = FALSE)
```

```
         price    green    nostalks disp    
price     1.00000  0.74834 -0.40656 -0.32464
green              1.00000 -0.01403 -0.12605
nostalks                    1.00000  0.35003
disp                                 1.00000
```

```
# based on the cor matrix, it is evident that the most highly correlated variables are price
```

B.

Comparing the square roots calculated of the R^2 I can see that they are equal except for sign. This is because the correlation coefficient can be negative (between -1 and 1) but the coiefficient of determination will always be positive (between 0 and 1) since its the squared correlation coefficient. If I had run the reverse regressios the R^2 measures would be the same as those from the correct regressions because when dealing with the same two variables, the ammount of variation explained in one variable by the other will be constant since again, it is the squares correlation coefficient. Reversing their order has no impact on the ammount of variation explained that our R^2 captures.

```
# price model
pricegreenmodel <- lm(data = waugh_clean, price ~ green)
pricenostalksmodel <- lm(data = waugh_clean, price ~ nostalks)
pricedispmodel <- lm(data = waugh_clean, price ~ disp)
#price on green
summary(pricegreenmodel)
```

```
Call:
lm(formula = price ~ green, data = waugh_clean)

Residuals:
   Min     1Q Median     3Q    Max 
-64.85 -11.52   1.29  12.55  47.43 
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   7.0275     5.4130   1.298    0.196
green        14.1091     0.8888  15.875   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 19.6 on 198 degrees of freedom
Multiple R-squared:   0.56, Adjusted R-squared:  0.5578
F-statistic:   252 on 1 and 198 DF,  p-value: < 2.2e-16
```

```r
sqrt(0.56)
```

```
[1] 0.7483315
```

```r
# price on nostalks
summary(pricenostalksmodel)
```

```
Call:
lm(formula = price ~ nostalks, data = waugh_clean)

Residuals:
    Min      1Q  Median      3Q     Max
-58.948 -17.317  -3.104   9.538  93.589

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 120.1645     5.1676  23.253  < 2e-16 ***
nostalks     -1.5377     0.2456  -6.262 2.32e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 27 on 198 degrees of freedom
Multiple R-squared:  0.1653,    Adjusted R-squared:  0.1611
F-statistic: 39.21 on 1 and 198 DF,  p-value: 2.32e-09
```

```r
sqrt(0.1653)
```

```
[1] 0.406571
```

```r
# price on disp
summary(pricedispmodel)
```

```

Call:
lm(formula = price ~ disp, data = waugh_clean)

Residuals:
    Min      1Q  Median      3Q     Max 
-57.153 -19.295  -3.114  15.451  86.272 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 105.6726     3.7826   27.94  < 2e-16 ***
disp         -1.0472     0.2168   -4.83 2.73e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 27.95 on 198 degrees of freedom
Multiple R-squared:  0.1054,    Adjusted R-squared:  0.1009 
F-statistic: 23.33 on 1 and 198 DF,  p-value: 2.731e-06
```

```r
sqrt(0.1054)
```

```
[1] 0.3246537
```

C.

I expect that if I add the regressor nostalks to the regression, the R^2 will increase. This is because we have seperately confirmed that variation in nostalks explains some of the variation in price, so if we add it to the regression, the models explanatory power should increase. Given the correlation between green and nostalks in the table I would expect the change in R^2 when nostalks is added to the regression to be large. My intuition is such that since the two variables are not super correlated BUT are indidivually correlated with price, they should explain a significant ammount of the variation in price when both included as regressors. Running the regression confirms my intuition, as the R^2 is increased by nearly 15 points, representing a significant jump. Further, comparing the R^2 between the model that regressed price on green to the model that regressed price on green AND disp, we can see another increase in the R^2 equating to about 5 points. This represents an interesting jump, despite it being smaller in magnitude to the previous change in R^2. I think this is consistent with the sample correlation between green and disp, which is bigger than that between green and nostalks but still not so big as to fail in adding additional explanation to the model. Finally, when comparing the price on nostalks model to the price on nostalks AND disp model, we also observe a jump in R^2 that is about 4 points. This is again smaller than previous jumps and can probably be explained by the higher correlation coefficient between nostalks and disp that comes out to 0.35003, an R that would suggest higher correlation between the two explanatory variables and thus a smaller increase to the R^2 when both included in a model explaining price. This is consistent with the previous findings regarding the effect of adding correlated explanatory variables to a model. I think this is why would prioritize using adjusted R^2 since that measure penalizes for adding less relavent variables.

```
green_nostalks_model <- lm(data = waugh_clean, price~green+nostalks)
summary(green_nostalks_model)
```

```
Call:
lm(formula = price ~ green + nostalks, data = waugh_clean)

Residuals:
    Min      1Q  Median      3Q     Max
-54.507  -9.997   0.746  10.008  50.997

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  36.9431     5.2100   7.091 2.33e-11 ***
green        14.0043     0.7148  19.593  < 2e-16 ***
nostalks     -1.4983     0.1434 -10.449  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 15.76 on 197 degrees of freedom
Multiple R-squared:  0.7169,    Adjusted R-squared:  0.714
F-statistic: 249.5 on 2 and 197 DF,  p-value: < 2.2e-16
```

```
summary(pricegreenmodel)
```

```
Call:
lm(formula = price ~ green, data = waugh_clean)

Residuals:
```

```
   Min     1Q Median     3Q    Max 
-64.85 -11.52   1.29  12.55  47.43 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   7.0275     5.4130   1.298    0.196    
green        14.1091     0.8888  15.875   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 19.6 on 198 degrees of freedom
Multiple R-squared:   0.56, Adjusted R-squared:  0.5578 
F-statistic:   252 on 1 and 198 DF,  p-value: < 2.2e-16
```

```r
green_disp_model <- lm(data = waugh_clean, price~green+disp)
summary(green_disp_model)
```

```
Call:
lm(formula = price ~ green + disp, data = waugh_clean)

Residuals:
    Min      1Q  Median      3Q     Max 
-64.610  -9.949   1.301  10.767  46.273 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  21.5318     5.7871   3.721 0.000259 ***
green        13.5529     0.8414  16.108  < 2e-16 ***
disp         -0.7549     0.1440  -5.244 4.03e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 18.41 on 197 degrees of freedom
Multiple R-squared:  0.6139,    Adjusted R-squared:   0.61 
F-statistic: 156.6 on 2 and 197 DF,  p-value: < 2.2e-16
```

```r
summary(pricegreenmodel)
```

```
Call:
```

```
lm(formula = price ~ green, data = waugh_clean)

Residuals:
   Min     1Q Median     3Q    Max 
-64.85 -11.52   1.29  12.55  47.43 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   7.0275     5.4130   1.298    0.196    
green        14.1091     0.8888  15.875   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 19.6 on 198 degrees of freedom
Multiple R-squared:   0.56, Adjusted R-squared:  0.5578 
F-statistic:   252 on 1 and 198 DF,  p-value: < 2.2e-16
```

```r
nostalks_disp_model <- lm(data = waugh_clean, price~nostalks+disp)
summary(nostalks_disp_model)
```

```
Call:
lm(formula = price ~ nostalks + disp, data = waugh_clean)

Residuals:
    Min      1Q  Median      3Q     Max 
-50.080 -16.153  -3.619  10.968  88.859 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 124.7557     5.2794  23.631  < 2e-16 ***
nostalks     -1.2626     0.2568  -4.917 1.85e-06 ***
disp         -0.6703     0.2190  -3.061  0.00252 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 26.44 on 197 degrees of freedom
Multiple R-squared:  0.2032,	Adjusted R-squared:  0.1951 
F-statistic: 25.12 on 2 and 197 DF,  p-value: 1.923e-10
```

```r
summary(pricenostalksmodel)
```

```
Call:
lm(formula = price ~ nostalks, data = waugh_clean)

Residuals:
    Min      1Q  Median      3Q     Max
-58.948 -17.317  -3.104   9.538  93.589

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 120.1645     5.1676  23.253  < 2e-16 ***
nostalks     -1.5377     0.2456  -6.262 2.32e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 27 on 198 degrees of freedom
Multiple R-squared:  0.1653,    Adjusted R-squared:  0.1611
F-statistic: 39.21 on 1 and 198 DF,  p-value: 2.32e-09
```

D.

Here I notice the same trend, where the R^2 for the full regression is 0.7268 while the summed R^2 from the single regressions is 0.8307. I think this is because in this case, the model accounts for overlaps in explanatory power between the regressors that are not even a question in the single models. I think this is becasue the multiple regression avoids double counting, providing the joint effect on price.

```r
fullmodel <- lm(data = waugh_clean, price~nostalks+green+disp)
summary(fullmodel)
```

```
Call:
lm(formula = price ~ nostalks + green + disp, data = waugh_clean)

Residuals:
    Min      1Q  Median      3Q     Max
-56.004  -9.485  -0.122   9.422  49.097
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  40.7613     5.3278   7.651 8.82e-13 ***
nostalks     -1.3573     0.1508  -8.999  < 2e-16 ***
green        13.7598     0.7099  19.382  < 2e-16 ***
disp         -0.3453     0.1297  -2.663  0.00839 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 15.52 on 196 degrees of freedom
Multiple R-squared:  0.7268,    Adjusted R-squared:  0.7226
F-statistic: 173.8 on 3 and 196 DF,  p-value: < 2.2e-16
```

```r
# R^2 of 0.7268
0.56+0.1653+0.1054
```

```
[1] 0.8307
```

```r
# sum of individual R^2 = 0.8307
```

E.

Waugh's interpretation of the coefficient of determination is incorrect. His method of adding up individual coeficients of determination is not the right way to do it, as we know the explanatory variables he is including are not orthogonal. Waugh should have first correctly calculated and second explained that the coefficient of determination is the portion of variance in price that is jointly explained by variance in the explanatory variables, and in the case of this model, it is equal to about 0.72 or 72%.

```r
# nostalks = 0.14554
-1.53394 * (-100.92 / 1063.64)
```

```
[1] 0.1455429
```

```r
# green = 0.44597
0.13826 * (3430.89 / 1063.64)
```

```
[1] 0.4459731
```

```r
# disp = 0.02133
-0.27554 * (-82.35 / 1063.64)
```

```
[1] 0.02133308
```

F.

Right off the bat I notice that the R^2 calculated are the same! I think this happened because by regressing price on a vector of fitted values that were fitted on a regression on price, I am effectively decomposing price to simply its predicted and residual parts. As for the intercept coefficient of 0 and the slope coefficient of 1, I think this is due to the fact that the vector of fitted values is a linear transformation of the original regressors, meaning that when I go onto regress price on that vector, the best linear fit for it would be a zero intercept and a 1 slope. This is because the fitted values generated are the closest possible linear predictors of the observed data.

```r
fullmodel <- lm(data = waugh_clean, price~nostalks+green+disp)
summary(fullmodel)
```

```
Call:
lm(formula = price ~ nostalks + green + disp, data = waugh_clean)

Residuals:
    Min      1Q  Median      3Q     Max 
-56.004  -9.485  -0.122   9.422  49.097 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  40.7613     5.3278   7.651 8.82e-13 ***
nostalks     -1.3573     0.1508  -8.999  < 2e-16 ***
green        13.7598     0.7099  19.382  < 2e-16 ***
disp         -0.3453     0.1297  -2.663  0.00839 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 15.52 on 196 degrees of freedom
Multiple R-squared:  0.7268,	Adjusted R-squared:  0.7226 
F-statistic: 173.8 on 3 and 196 DF,  p-value: < 2.2e-16
```

```r
fitted <- fitted(fullmodel)
fitmodel <- lm(waugh_clean$price~fitted)
summary(fitmodel)
```

```
Call:
lm(formula = waugh_clean$price ~ fitted)

Residuals:
    Min      1Q  Median      3Q     Max 
-56.004  -9.485  -0.122   9.422  49.097 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 1.061e-13  4.075e+00    0.00        1    
fitted      1.000e+00  4.357e-02   22.95   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 15.44 on 198 degrees of freedom
Multiple R-squared:  0.7268,    Adjusted R-squared:  0.7254 
F-statistic: 526.7 on 1 and 198 DF,  p-value: < 2.2e-16
```

4. Assessing the Stability of the Hedonic Price Equation for the First and Second-Generation Computers

A.

B.

Chow's model: LN_RENT= 0 + 1LN_MEM + 2LN_MULT + 3LN_ACCESS + u My first f-test yielded the following statistics: Chow F-statistic: 0.5574, df = (20, 58), p-value = 0.9256 My second f-test yielded the following statitstics: Chow F-statistic: 0.6243, df = (20, 31), p-value = 0.863723

I also find associated p-values of 0.9256 and 0.8637 respectively. Based on these findings I fail to reject the null hypothesis, finding lack of evidence that the slopes differ. This is similar to Chow's findings.

```
library("readxl")
chow <- read_excel('/Users/kieran/Documents/MASTERS/METRICS/code/metrics/practicum_2_files/P

# data cleaning and constructions
chow_clean <- chow |>
  rename(
    obs = "Obs",
    volume = "VOLUME",
    rent = "RENT",
    binary = "BINARY",
    digits = "DIGITS",
    words = "WORDS",
    add = "ADD",
    mult = "MULT",
    access = "ACCESS",
    year = "YEAR",
    order = "ORDER",
    ibmdum = "IBMDUM"
  ) |>
    mutate(
      ln_rent = log(rent),
      ln_mult = log(mult),
      ln_access = log(access),
      ln_add = log(add),
      mem = words*binary*digits,
      ln_mem = log(mem)
    )

#constrained data and model
constr_chow <- chow_clean |>
  filter(year %in% c(60, 61, 62, 63, 64, 65))

# constructing clope coefficients like in 3.a.
# create cor matrix
order_chow <- c("ln_rent", "ln_mem", "ln_mult", "ln_access")
cor_matrix_chow <- cor(chow_clean)
# cor_matrix_chow <- cor_matrix[order_chow, order_chow]

# round the matrix
cor_matrix_rounded_chow <- round(cor_matrix_chow, 5)

# lower triangle with empty strings
```

```r
cor_matrix_char_chow <- format(cor_matrix_rounded_chow, nsmall = 5)
cor_matrix_char_chow[lower.tri(cor_matrix_char_chow)] <- ""

print(cor_matrix_char_chow, quote = FALSE)
```

```
          obs     volume   rent     binary   digits   words    add     
obs        1.00000  0.21965  0.03746 -0.30001  0.01232  0.29663 -0.29272
volume              1.00000 -0.13971  0.09402 -0.25418 -0.00153 -0.02353
rent                         1.00000 -0.12260  0.37107  0.51318 -0.17710
binary                                1.00000 -0.64290 -0.13430  0.18879
digits                                         1.00000 -0.02091 -0.05012
words                                                   1.00000 -0.12115
add                                                              1.00000
mult                                                                    
access                                                                  
year                                                                    
order                                                                   
ibmdum                                                                  
ln_rent                                                                 
ln_mult                                                                 
ln_access                                                               
ln_add                                                                  
mem                                                                     
ln_mem                                                                  
          mult     access   year     order    ibmdum   ln_rent  ln_mult 
obs       -0.32512 -0.53380  0.98968  1.00000 -0.01696  0.04020 -0.67477
volume    -0.02577 -0.13205  0.21132  0.21965  0.18270 -0.16712  0.04120
rent      -0.18803 -0.30584  0.04531  0.03746  0.35135  0.78977 -0.48891
binary     0.18506  0.05802 -0.30849 -0.30001  0.22482 -0.02370  0.40788
digits    -0.04427  0.06432  0.02960  0.01232 -0.20199  0.16764 -0.20710
words     -0.12831 -0.21607  0.27597  0.29663  0.29011  0.45336 -0.37587
add        0.99334  0.49600 -0.30844 -0.29272 -0.13647 -0.33007  0.51617
mult       1.00000  0.52413 -0.34138 -0.32512 -0.14532 -0.34507  0.53977
access              1.00000 -0.54836 -0.53380 -0.26138 -0.46524  0.72318
year                         1.00000  0.98968 -0.02983  0.04176 -0.68294
order                                 1.00000 -0.01696  0.04020 -0.67477
ibmdum                                         1.00000  0.33318 -0.12620
ln_rent                                                 1.00000 -0.58474
ln_mult                                                          1.00000
ln_access                                                               
ln_add                                                                  
mem                                                                     
```

```
ln_mem
           ln_access ln_add   mem      ln_mem
obs        -0.70016  -0.62843  0.22658  0.34817
volume     -0.12977  -0.00407 -0.07879 -0.12772
rent       -0.40202  -0.46438  0.80777  0.65657
binary      0.15272   0.39932 -0.19094 -0.20511
digits     -0.00773  -0.17664  0.36405  0.21697
words      -0.33875  -0.35789  0.62968  0.52719
add         0.42748   0.57882 -0.11318 -0.44234
mult        0.45680   0.59353 -0.12088 -0.46409
access      0.86526   0.74038 -0.18763 -0.43530
year       -0.71031  -0.64090  0.22517  0.34420
order      -0.70016  -0.62843  0.22658  0.34817
ibmdum     -0.21527  -0.13584  0.14798  0.17801
ln_rent    -0.55393  -0.59442  0.51569  0.84689
ln_mult     0.87557   0.96936 -0.41591 -0.65735
ln_access   1.00000   0.88256 -0.30229 -0.57526
ln_add                1.00000 -0.37522 -0.66034
mem                            1.00000  0.60344
ln_mem                                  1.00000
```

```r
# now its time to run the constrained and unconstrained models
# Common slopes, different intercepts by year
pooled <- lm(data = constr_chow, ln_rent~factor(year)+ln_mem+ln_mult+ln_access)
rss_pooled <- sum(resid(pooled)^2)

# allowed to differ
library(broom)
by_year <- constr_chow %>%
  group_by(year) %>%
  do(tidy(lm(ln_rent ~ ln_mem + ln_mult + ln_access, data = .)))
by_year
```

```
# A tibble: 24 x 6
# Groups:   year [6]
    year term        estimate std.error statistic  p.value
   <dbl> <chr>          <dbl>     <dbl>     <dbl>    <dbl>
 1    60 (Intercept)  1.20       1.54      0.785   0.462
 2    60 ln_mem       0.423      0.180     2.36    0.0566
 3    60 ln_mult     -0.152      0.101    -1.51    0.182
 4    60 ln_access   -0.121      0.0783   -1.54    0.174
 5    61 (Intercept)  0.00484    0.934     0.00519 0.996
```

```
 6    61 ln_mem       0.551      0.108    5.11    0.000920
 7    61 ln_mult     -0.0615     0.0729  -0.843   0.424
 8    61 ln_access   -0.175      0.0519  -3.38    0.00962
 9    62 (Intercept) -2.40       1.20    -2.01    0.0844
10    62 ln_mem       0.826      0.152    5.42    0.000987
# i 14 more rows
```

```r
# cell-means parameterization: per-year intercepts and per-year slopes
m_interacted_cm <- lm(ln_rent ~ factor(year) + factor(year) + factor(year), data = chow_clea
summary(m_interacted_cm)
```

```
Call:
lm(formula = ln_rent ~ factor(year) + factor(year) + factor(year),
    data = chow_clean)

Residuals:
    Min      1Q  Median      3Q     Max
-3.6240 -0.7671 -0.1120  0.9548  2.8111

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     1.92967    0.46991   4.106 7.21e-05 ***
factor(year)55 -0.43900    0.62655  -0.701    0.485
factor(year)56 -0.24626    0.60111  -0.410    0.683
factor(year)57  0.34655    0.62655   0.553    0.581
factor(year)58  0.20080    0.62655   0.320    0.749
factor(year)59  0.49038    0.61269   0.800    0.425
factor(year)60 -0.17661    0.61269  -0.288    0.774
factor(year)61  0.33439    0.59129   0.566    0.573
factor(year)62  0.09921    0.60111   0.165    0.869
factor(year)63  0.16512    0.56909   0.290    0.772
factor(year)64  0.37267    0.55379   0.673    0.502
factor(year)65 -0.31812    0.56340  -0.565    0.573
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.243 on 125 degrees of freedom
Multiple R-squared:  0.05719,	Adjusted R-squared:  -0.02578
F-statistic: 0.6893 on 11 and 125 DF,  p-value: 0.7467
```

```
# fit pooled model and get its rss
pooled <- lm(ln_rent ~ factor(year) + ln_mem + ln_mult + ln_access, data = constr_chow)
RSS_pooled <- sum(resid(pooled)^2)

# run separate regressions by year and sum RSSs
years <- unique(constr_chow$year)
RSS_years <- 0
nyears <- length(years)
k <- 4
N_total <- 0
for (yy in years) {
  sub <- subset(constr_chow, year == yy)
  fit <- lm(ln_rent ~ ln_mem + ln_mult + ln_access, data = sub)
  RSS_years <- RSS_years + sum(resid(fit)^2)
  N_total <- N_total + nrow(sub)
}

# calculate degrees of freedom
numerator_df <- k * (nyears - 1)
denominator_df <- N_total - k * nyears

# calculate Chow F-statistic
F_stat <- ((rss_pooled - RSS_years) / numerator_df) / (RSS_years / denominator_df)

# find p-value
p_value <- pf(F_stat, numerator_df, denominator_df, lower.tail = FALSE)
cat(sprintf("Chow F-statistic: %.4f, df = (%d, %d), p-value = %.4g\n", F_stat, numerator_df,
```

```
Chow F-statistic: 0.5574, df = (20, 58), p-value = 0.9256
```

```
# now to do the same but for the years 50-59
# filter data for years 54-59
constr_chow_old <- chow_clean %>%
  filter(year %in% c(54, 55, 56, 57, 58, 59))

# fit pooled model for those years
pooled_old <- lm(ln_rent ~ factor(year) + ln_mem + ln_mult + ln_access, data = constr_chow_ol
RSS_pooled_old <- sum(resid(pooled_old)^2)

# run separate regressions and sum RSSs for each year (from constr_chow_old)
years_old <- unique(constr_chow_old$year)
```

```r
RSS_years_old <- 0
nyears_old <- length(years_old)
k_old <- 4
N_total_old <- 0
for (yy in years_old) {
  sub_old <- subset(constr_chow_old, year == yy)
  fit_old <- lm(ln_rent ~ ln_mem + ln_mult + ln_access, data = sub_old)
  RSS_years_old <- RSS_years_old + sum(resid(fit_old)^2)
  N_total_old <- N_total_old + nrow(sub_old)
}

# calculate degrees of freedom
numerator_df_old <- k_old * (nyears_old - 1)
denominator_df_old <- N_total_old - k_old * nyears_old

# calculate Chow F-statistic
F_stat_old <- ((RSS_pooled_old - RSS_years_old) / numerator_df_old) / (RSS_years_old / denomi
p_value_old <- pf(F_stat_old, numerator_df_old, denominator_df_old, lower.tail = FALSE)

cat(sprintf(
  "Chow F-statistic: %.4f, df = (%d, %d), p-value = %.4g\n",
  F_stat_old, numerator_df_old, denominator_df_old, p_value_old))
```

```
Chow F-statistic: 0.6243, df = (20, 31), p-value = 0.8637
```

C.

I find a f statistic of 3.6926, with a corresponding p value of 0.01389. Since the p-value shows that the results are significant at a p<0.05 level, I reject the null hypothesis. I determine that there is statistically significant evidence that the relationship between ln(rent) and one of more of the explanatory variables used in the model changed between the generations of computers. This does not really surprise me, since I would expect technological value relationships to change over time as technology becomes more capable. I think this demonstrates that a single hedonic pricing framework fails to fit both generations correctly due to external dynamics shifting between generations. Next I relax the assumption of slope parameter equality within each generation and test the null hypothesis that slope parameters are equal over the entire 1954-1965 time span against the alternative hypothesis that these slope coefficients varied from year to year. I find an f-statistic of 0.7328 and a corresponding p-value of 0.8721. This means that there is no significant evidence that the relationship between features and rent changes from year to year over the entire period, leading me to fail in rejecting the null hypothesis. These findings are consistent with those found earlier since it is plausible that the relationship could only change at the generational boundary while remaining constant withinm each generation. These results point towards a large shift but not indefinite instability between sub-periods.

```r
# full regression not filtering for year; restricted model
full_model <- lm(ln_rent ~ factor(year) + ln_mem + ln_mult + ln_access, data = chow_clean)
summary(full_model)
```

```

Call:
lm(formula = ln_rent ~ factor(year) + ln_mem + ln_mult + ln_access,
    data = chow_clean)

Residuals:
     Min       1Q   Median       3Q      Max
-0.92776 -0.23149  0.02861  0.22199  0.89756

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     1.39309    0.28848   4.829 4.02e-06 ***
factor(year)55 -0.05461    0.18593  -0.294 0.769482
factor(year)56 -0.21229    0.17899  -1.186 0.237902
factor(year)57 -0.28449    0.18668  -1.524 0.130103
factor(year)58 -0.47597    0.18757  -2.538 0.012421 *
factor(year)59 -0.69406    0.18494  -3.753 0.000269 ***
factor(year)60 -1.13892    0.18626  -6.115 1.20e-08 ***
factor(year)61 -1.23887    0.18218  -6.800 4.12e-10 ***
factor(year)62 -1.62179    0.19082  -8.499 5.59e-14 ***
factor(year)63 -1.73257    0.18521  -9.355 5.21e-16 ***
factor(year)64 -2.02818    0.18414 -11.014  < 2e-16 ***
factor(year)65 -2.30576    0.18544 -12.434  < 2e-16 ***
ln_mem          0.51912    0.02751  18.873  < 2e-16 ***
ln_mult        -0.06351    0.02331  -2.725 0.007381 **
ln_access      -0.16059    0.01985  -8.091 5.00e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3674 on 122 degrees of freedom
Multiple R-squared:  0.9197,    Adjusted R-squared:  0.9104
F-statistic: 99.74 on 14 and 122 DF,  p-value: < 2.2e-16
```

```r
# unrestricted
chow_clean <- chow_clean %>%
  mutate(gen = ifelse(year <= 59, "first", "second"))

model_by_gen <- lm(ln_rent ~ factor(year) + ln_mem * gen + ln_mult * gen + ln_access * gen, d
summary(model_by_gen)
```

```
Call:
lm(formula = ln_rent ~ factor(year) + ln_mem * gen + ln_mult *
    gen + ln_access * gen, data = chow_clean)

Residuals:
     Min       1Q   Median       3Q      Max
-0.95543 -0.21900  0.01621  0.19941  0.83803

Coefficients: (1 not defined because of singularities)
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)          2.123953   0.490932   4.326 3.17e-05 ***
factor(year)55      -0.035690   0.180811  -0.197 0.843862
factor(year)56      -0.139654   0.175530  -0.796 0.427839
factor(year)57      -0.209955   0.183478  -1.144 0.254795
factor(year)58      -0.500682   0.183374  -2.730 0.007288 **
factor(year)59      -0.670951   0.182039  -3.686 0.000345 ***
factor(year)60      -2.228408   0.570994  -3.903 0.000158 ***
factor(year)61      -2.368206   0.574847  -4.120 7.04e-05 ***
factor(year)62      -2.717515   0.560047  -4.852 3.74e-06 ***
factor(year)63      -2.822257   0.556268  -5.074 1.45e-06 ***
factor(year)64      -3.153227   0.558967  -5.641 1.16e-07 ***
factor(year)65      -3.391577   0.555695  -6.103 1.34e-08 ***
ln_mem               0.410785   0.047161   8.710 2.08e-14 ***
gensecond                  NA         NA      NA       NA
ln_mult             -0.067905   0.045973  -1.477 0.142303
ln_access           -0.191971   0.029998  -6.399 3.21e-09 ***
ln_mem:gensecond     0.168545   0.057426   2.935 0.004004 **
gensecond:ln_mult    0.002539   0.052966   0.048 0.961851
gensecond:ln_access  0.051367   0.040461   1.270 0.206723
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3557 on 119 degrees of freedom
Multiple R-squared:  0.9265,    Adjusted R-squared:  0.9161
```

```
F-statistic:  88.3 on 17 and 119 DF,  p-value: < 2.2e-16
```

```
# test! set up equation
rss_restricted <- sum(resid(full_model)^2)
rss_unrestricted <- sum(resid(model_by_gen)^2)

numerator_df <- 3
k_unrestricted <- length(coef(model_by_gen))

denominator_df <- nrow(chow_clean) - k_unrestricted
F_stat <- ((rss_restricted - rss_unrestricted) / numerator_df) / (rss_unrestricted / denomina
p_value <- pf(F_stat, numerator_df, denominator_df, lower.tail = FALSE)
cat(sprintf("Chow F-statistic: %.4f, df = (%d, %d), p-value = %.4g\n", F_stat, numerator_df,
```

```
Chow F-statistic: 3.6926, df = (3, 118), p-value = 0.01389
```

```
#Now I relax the assumption of slope parameter equality within each generation and test the r
# run the model
pooled <- lm(ln_rent ~ factor(year) + ln_mem + ln_mult + ln_access, data = chow_clean)
RSS_pooled <- sum(resid(pooled)^2)
# set up the test
years <- unique(chow_clean$year)
RSS_years <- 0
nyears <- length(years)
k <- 4
N_total <- 0
for (yy in years) {
  sub <- subset(chow_clean, year == yy)
  fit <- lm(ln_rent ~ ln_mem + ln_mult + ln_access, data = sub)
  RSS_years <- RSS_years + sum(resid(fit)^2)
  N_total <- N_total + nrow(sub)
}
# calculate p value and interpret
numerator_df <- k * (nyears - 1)
denominator_df <- N_total - k * nyears
F_stat <- ((RSS_pooled - RSS_years) / numerator_df) / (RSS_years / denominator_df)
p_value <- pf(F_stat, numerator_df, denominator_df, lower.tail = FALSE)
cat(sprintf("Chow F-statistic: %.4f, df = (%d, %d), p-value = %.4g\n", F_stat, numerator_df,
```

```
Chow F-statistic: 0.7328, df = (44, 89), p-value = 0.8721
```

5. Using Time-Varying Hedonic Price Equations to Construct Chained Price Indexes for Computers

A.

In comparing the year-to-year changes in the estimated coefficients of the 11 dummy variables with the levels of the t estimates, I can see that the estimates are relatively close to each other between methods, with directionality consistently matching. I do see the potential for higher order interactions to exist that arent captured in the smaller adjacent year regressions. This could be due to slope inconsistency or potentially omitted interactions. I think it is appropriate to compare year-to-year changes in the estimated dummy variable coefficients with levels of the estimated t because both methods provide estimates of the annual adjusted price change. THey are both expected to be similar if the model is well specified and if the sample sizes are of reasonable size. I notice some more substantial differences between the years 56, 57, and 62, which could be the result of the adjacent year models' use of only two years of data per estimate. It could similarly have to do with the pooled version using data from all of the years which could effectively smooth away more volatility per year. I think the differences could be the result of either actual heterogeneity being picked up by the models, or some sort of misspecification due to the nature of the approaches taken. ### B. I think the logic behind referring to this as a "chained" price index rests on the fact that each coefficient is effectively linked to the others forming a chain of sorts. With each year price change being calculated relative to the preceding year and the changes then being compounded over time, each period is clearly building on the last thus developing a chain of prices. For the final part of this question, I compare the hedonic to the chained price indeces. To start, I notice that the chained index is greater than 1 for several years while the hedonic index is not. The chained index also has a pretty dramatic drop off and then becomes increasingly close to 0 while the hedonic index decreases over time much more gradually. I think that since the chained index compounds on previous years, it can reflect volatility but also may be more prone to noise. The hedonic index on the other hand smoothes out extreme volatility over time showing a cleaner long run trend. For this reason I generally would prefer the the hedonic index. It generally feels more realistic, more interpretable, and is less volatile. I also think that in this case, the chained index may be heavily influenced by some of the beta values due to its extreme jumps and high values. This could imply that for these data, it is not the best way to understand time varying dynamics.

```r
# create lists to store results
adjacent_betas <- numeric()
years <- 54:64 # last pair is 64-65

for (yy in years) {
  # filter data for the adjacent years
  dat_pair <- chow_clean %>%
```

```r
    filter(year %in% c(yy, yy + 1)) %>%
    mutate(dummy = ifelse(year == (yy + 1), 1, 0))

  # run regression
  fit <- lm(ln_rent ~ dummy + ln_mem + ln_mult + ln_access, data = dat_pair)
  # extract beta for the adjacent year dummy (beta t)
  beta_t <- coef(fit)["dummy"] # get coefficient by name
  adjacent_betas <- c(adjacent_betas, beta_t)
}

# year names for each beta coefficient
names(adjacent_betas) <- paste0('beta_', as.character(55:65))

# print out the adjacent year beta estimates
print(adjacent_betas)
```

```
    beta_55     beta_56     beta_57     beta_58     beta_59     beta_60
-0.06745630 -0.13118744 -0.12639924 -0.25751042 -0.20202314 -0.50355965
    beta_61     beta_62     beta_63     beta_64     beta_65
-0.08546431 -0.28575199 -0.12956105 -0.31576839 -0.21823508
```

```r
# now for the traditional hedonic approach on the 11 dummies and other variables
hedonic_time_dummy <- lm(ln_rent ~ factor(year) + ln_mem + ln_mult + ln_access, data = chow_c
summary(hedonic_time_dummy)
```

```

Call:
lm(formula = ln_rent ~ factor(year) + ln_mem + ln_mult + ln_access,
    data = chow_clean)

Residuals:
     Min       1Q   Median       3Q      Max
-0.92776 -0.23149  0.02861  0.22199  0.89756

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     1.39309    0.28848   4.829 4.02e-06 ***
factor(year)55 -0.05461    0.18593  -0.294 0.769482
factor(year)56 -0.21229    0.17899  -1.186 0.237902
factor(year)57 -0.28449    0.18668  -1.524 0.130103
factor(year)58 -0.47597    0.18757  -2.538 0.012421 *
```

```
factor(year)59 -0.69406    0.18494  -3.753 0.000269 ***
factor(year)60 -1.13892    0.18626  -6.115 1.20e-08 ***
factor(year)61 -1.23887    0.18218  -6.800 4.12e-10 ***
factor(year)62 -1.62179    0.19082  -8.499 5.59e-14 ***
factor(year)63 -1.73257    0.18521  -9.355 5.21e-16 ***
factor(year)64 -2.02818    0.18414 -11.014  < 2e-16 ***
factor(year)65 -2.30576    0.18544 -12.434  < 2e-16 ***
ln_mem          0.51912    0.02751  18.873  < 2e-16 ***
ln_mult        -0.06351    0.02331  -2.725 0.007381 **
ln_access      -0.16059    0.01985  -8.091 5.00e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3674 on 122 degrees of freedom
Multiple R-squared:  0.9197,    Adjusted R-squared:  0.9104
F-statistic: 99.74 on 14 and 122 DF,  p-value: < 2.2e-16
```

```r
# Now for part b I will calculate a hedonic price index
hedonic_coeffs <- coef(hedonic_time_dummy)

dummy_names <- paste0('factor(year)', 55:65)
hedonic_index <- exp(c(0, hedonic_coeffs[dummy_names])) # prepend 0 for base year
names(hedonic_index) <- 54:65
print(hedonic_index)
```

```
        54         55         56         57         58         59         60
1.00000000 0.94685568 0.80873141 0.75239464 0.62127942 0.49954370 0.32016382
        61         62         63         64         65
0.28971066 0.19754519 0.17683000 0.13157444 0.09968332
```

```r
# Now I will construct a chained price index as the second half of the question
years <- 54:65
chained_log_sum <- c(0, cumsum(hedonic_coeffs))
chained_index <- exp(chained_log_sum)
names(chained_index) <- years
print(chained_index)
```

```
          54           55           56           57           58           59
1.000000e+00 4.027282e+00 3.813255e+00 3.083899e+00 2.320309e+00 1.441560e+00
          60           61           62           63           64           65
7.201224e-01 2.305571e-01 6.679486e-02 1.319500e-02 2.333272e-03 3.069990e-04
```

```
        <NA>         <NA>         <NA>         <NA>
3.060268e-05 5.142941e-05 4.826488e-05 4.110421e-05
```