OLS coefficients by hand in R and Python

1. Deriving the OLS coefficients

The independent and depenendent variables in a multivariate regression can be represented in matrix notation as

$$y = X\beta + u,$$

where

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1k} \\ x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ x_{T1} & x_{T2} & \cdots & x_{Tk} \end{pmatrix}, \quad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_T \end{pmatrix}, \quad u = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_T \end{pmatrix}.$$

In matrix notation, the criterion function to be minimized is

$$SSE(\beta) = (y - X\beta)'(y - X\beta),$$

and the first-order conditions are

$$\frac{\partial SSE(\beta)}{\partial \beta} = -2X'(y - X\hat{\beta}) = 0,$$

which yields the normal equations,

$$(X'X)\hat{\beta} = X'y.$$

As long as $(X'X)$ is of full rank, then

$$\hat{\beta} = (X'X)^{-1}X'y.$$

It can be shown via the Gauss-Markov theorem that under the classical assumptions, the OLS estimator has the least variance in the class of all linear unbiased estimators of β. However, the point of this document is to show how to calculate the OLS coefficients by hand using the computer programs R and Python. Let's start with R.

2. Calculating $\hat{\beta}$ by hand in R

```r
# Number of observations
N <- 500

# Generate data for the independent variables
set.seed(4)
x0 <- runif(N, min = 1, max = 1)
x1 <- rnorm(N, 0, sd = 1)
x2 <- rnorm(N, 2, sd = 4)
x3 <- rnorm(N, -1, sd = 0.5)

# Create independent variable and define the betas
y = 2 + 5*x1 -2*x2 + 1.5*x3 + rnorm(N, 0, sd = 1)

# Convert to a data frame
df <- data.frame(x0, x1, x2, x3, y)
head(df)
```

```
##   x0         x1         x2         x3         y
## 1  1  0.2167549 -4.5787213 -0.9116930 10.132975
## 2  1 -0.5424926 -1.2799037 -0.1554771  1.427177
## 3  1  0.8911446 -4.7129586 -1.6736710 13.236882
## 4  1  0.5959806  3.6966599 -0.4621888 -4.048841
## 5  1  1.6356180  0.6048537 -1.2281045  6.776754
## 6  1  0.6892754  1.7906728 -1.3407223  1.410808
```

```r
# Convert data to matrix form
Y <- as.matrix(df[, "y"])
X <- as.matrix(df[, c("x0","x1","x2","x3")])

# Manually calculate OLS coefficients
beta_hat <- solve(t(X) %*% X) %*% t(X) %*% Y
beta_hat
```

```
##         [,1]
## x0  2.066855
## x1  4.972577
## x2 -1.995047
## x3  1.516274
```

```r
# Run OLS regression and compare results
model <- lm(y ~ x1 + x2 + x3)
coef(model)
```

```
## (Intercept)          x1          x2          x3
##    2.066855    4.972577   -1.995047    1.516274
```

```r
library(equatiomatic)
extract_eq(model, use_coefs = TRUE)
```

$$\widehat{\mathrm{y}} = 2.07 + 4.97(\mathrm{x1}) - 2(\mathrm{x2}) + 1.52(\mathrm{x3}) \tag{1}$$

3. Calculating $\hat{\beta}$ by hand in Python

```python
import pandas as pd
import numpy as np
import statsmodels.api as st

# Number of observations

N = 500

# Generate data for the independent variables
np.random.seed(4)
x0 = np.ones(500)
x1 = np.random.normal(0, 1, 500)
x2 = np.random.normal(2, 4, 500)
x3 = np.random.normal(-1, 0.5, 500)

# Create independent variable and define the betas
y = 2 + 5*x1 -2*x2 + 1.5*x3 + np.random.normal(0, 1, 500)

# Create a matrix of independent variables
```

```
x = np.column_stack((x0, x1, x2, x3))

# Convert to a data frame
df = pd.DataFrame(data = x)

# Examine first 5 rows of data frame
df[:5]

# Manually calculate OLS coefficients
```

```
##      0         1         2         3
## 0  1.0  0.050562 -1.137449 -1.065155
## 1  1.0  0.499951  3.426072 -1.944436
## 2  1.0 -0.995909  9.263411 -1.947148
## 3  1.0  0.693599  3.353012 -1.382067
## 4  1.0 -0.418302  1.802297 -0.587590
```

```
beta_hat = np.dot(np.linalg.inv(np.dot(x.transpose(),x)), np.dot(x.transpose(),y))
print(beta_hat)

# Run OLS regression and compare results
```

```
## [ 2.0170067   4.99170179 -1.99570408  1.52646976]
```

```
model = st.OLS(y, x)
results = model.fit(cov_type = 'HC1')
print(results.summary())
```

```
##                             OLS Regression Results
## ==============================================================================
## Dep. Variable:                      y   R-squared:                       0.989
## Model:                            OLS   Adj. R-squared:                  0.989
## Method:                 Least Squares   F-statistic:                 1.438e+04
## Date:                Thu, 27 Jan 2022   Prob (F-statistic):               0.00
## Time:                        15:59:57   Log-Likelihood:                -691.20
## No. Observations:                 500   AIC:                             1390.
## Df Residuals:                     496   BIC:                             1407.
## Df Model:                           3
## Covariance Type:                  HC1
## ==============================================================================
##                  coef    std err          z      P>|z|      [0.025      0.975]
## ------------------------------------------------------------------------------
## const          2.0170      0.095     21.167      0.000       1.830       2.204
## x1             4.9917      0.044    113.271      0.000       4.905       5.078
## x2            -1.9957      0.011   -177.707      0.000      -2.018      -1.974
## x3             1.5265      0.087     17.639      0.000       1.357       1.696
## ==============================================================================
## Omnibus:                        3.809   Durbin-Watson:                   1.939
## Prob(Omnibus):                  0.149   Jarque-Bera (JB):                3.581
## Skew:                           0.188   Prob(JB):                        0.167
## Kurtosis:                       3.177   Cond. No.                         12.7
## ==============================================================================
##
## Notes:
## [1] Standard Errors are heteroscedasticity robust (HC1)
```