```
* Output of the program:

Program 1: Read the data, and prepares c and X.
  c and X successfully generated!

Program 2: Computes the average SSR for theta = [0; 0; 0].
  The average SSR for theta=[0, 0, 0] is: 147.273

Program 3: Optimizes SSR function using fminsearch.
  The Theta that optimizes SSR is: [2.149233, 0.297950, 1.001049]
  The Optimal SSR is: 9.525324e-05

Program 4: Restrict rho and optimize with fminsearch.
  The Theta that optimizes SSR is: [0, 0, 1]
  The Optimal SSR is: 2.015572e-04

Program 5: Own alternatives for fminsearch.
  Alternative 1: A steepest decent algorithm.
  The Theta that optimizes SSR is: [2.07766, 0.292619, 1.00104]
  The Optimal SSR is: 9.528267e-05

  Alternative 2: Newton-Raphson.
  The Theta that optimizes SSR is: [5.48767, 0.537869, 1]
  The Optimal SSR is: 1.240667e-04

  Own algorithms are not capable of finding an optimum.
```

* Conclusion:

1. Creating functions and scripts for different purposes would help us a lot and make the program clear and tidy.

2. It is obvious both of our own alternatives do not reach satisfactory results. Comparing the results from Program 3 and Program 5, we see that the optimal value that we found by using the function 'fminsearch' is different from those we found by own alternatives. That means a steepest decent algorithm and Newton-Raphson are not good and complete enough to find the minimum. There are some problems in these two methods which would make the value converge to a stationary point that is not necessarily the optima.

* What we did:

The whole program consists of 5 sub-programs (functions), which can be found under the path '\program'.

```
Program 1: LoadData.m
Program 2: AverageSSR.m
Program 3: OptSSR.m
Program 4: OptWithRes.m
Program 5: Alternative1.m       and     Alternative2.m
```

Each program might consists of one or more functions, which can be found under the path '\include'.

1. The first program:
	1) Read the data.
		Using 'importdata' command, and give the value into mConsumption.
	2) Prepare c and X for the correct time period.
		The data file contains: rank, year, rank, income, expenditure. Transform columes of income and expenditure seperately into different vectors, namely vY and vC. Transform them into logarithm form. Select the correct tiem period to form the matrix and vectors we want.
	3) FYI, averages and standard deviations in Table. We can check whether we are right.

2. The second program:
	Make two functions that convert theta into beta and convert beta into theta.
	Calculate the SSR with the beta corresponding to theta=0.

3. The third program
	find the initial point by OLS.
	Claculate by OLS the initial point of beta, and convert it into starting value of theta.
	Create a seperate function that the SSR is the function of Theta. This function is created by combining SSRfuncBeta and ThetaToBeta. Thus, we can use fminsearch to find the minimum.

4. Minimize with restriction
	Rho is defined within interval (-1, 1).
	First function, transform theta to new parameter thetastar.
		transform parameter. rho1=(rho+1)/2, within (0, 1).
		transform parameter. rho2=log(rho1/(1-rho1)), within (-infinite, +infinite).
	Second function, back-transform the thetastar to theta.
		Back-transform parameter. rho1=exp(rho2)/(1+exp(rho2)).
		Back-transform parameter. rho=2*rho1-1
	Note: there is probability that initial rho calculated by OLS is not within interval (-1,1). Transform it into new parameter might cause error. Thus we should discuss the case by assuming rho is within the interval or not. If yes, then we can transform it into new parameter, if not, we can abtrarilly assign the original theta to be [0, 0, 0].
	Then, we create a function that the SSR is the function of ThetaStar. And thus we can optimize SSR with function fminsearch. After getting the optimizer ThetaStar, we can transfer it back to Theta.

5. Own alternatives
	First, a steepest decent algorithm.
		Find the initial theta.
		Calculate the gradients.
		Move a short distance towards the opposite direction of gradients, which means that multiplying the gradients by a small negative number

```
alpha.
		Add this distance into initial theta.
		Repeatedly doing this until the gradients is very close to zero.
Let us abitrarily set it to be 0.0001.
	Second, Newton-Raphson.
		Find the initial theta.
		Calculate the gradient and hessian on this point.
		Calculate the h with Newton-Raphson。
		Plus initial theta with h to get the new point.
		Repeatedly doing this procedure, until |h| is small enough. Let
us abitrarily set it to be 0.0001.


Sun, Junze
Zhu, Yuhao
3 Nov. 2013
```