Yugo Nagatake
22146468

【Summer Gasyuku Assignment for the 5th and 6th gens】
Assignment: From the chapter 1 to chapter 5 of "**Introductory Econometrics for Finance** (4th edition) by Chris Brooks (Cambridge University Press) ~ It is the review of your past two years in Math, Statistics, and Econometrics. So, I suppose you can make this easily, though Python might be new for some:)

1. Read and understand all contents including math for the preparation of **individual oral exam during Summer Gasyuku**
2. Try all Python code and summarize its result with your own interpretation (https://www.cambridge.org/highereducation/books/introductory-econometrics-for-finance[…]E5DC6565F57F04F/python-code/8BB3DBAE944CBED5FB8F1933281B2FDE) => Submit due of this assignment would be September 20th 2023

We will use the book for the coming semester, so please buy the book for this summer vacation and the next term starting from October 2023.

Code summary and interpretation

Chapter 2

Data_management.py: this code file contains basics of python including variable type and function.

fund_managers.py: this code file contained basic pandas and numpy data manipulation.

	Year	Risky Ricky	Safe Steve	Ricky Ranks	Steve Ranks
count	13.00000	13.000000	13.000000	13.00000	13.000000
mean	2011.00000	5.692308	4.692308	7.00000	6.846154
std	3.89444	20.360847	6.612924	3.89444	3.782551
min	2005.00000	-23.000000	-8.000000	1.00000	1.000000
25%	2008.00000	-7.000000	2.000000	4.00000	4.000000
50%	2011.00000	4.000000	4.000000	7.00000	7.000000
75%	2014.00000	14.000000	7.000000	10.00000	10.000000
max	2017.00000	56.000000	19.000000	13.00000	13.000000

describe method outputs basic information of the data frame including quantiles.

	Year	Risky Ricky	Safe Steve	Ricky Ranks	Steve Ranks
Year	1.000000	0.142928	0.385059	0.021978	-0.197996
Risky Ricky	0.142928	1.000000	0.870048	-0.934285	-0.766741
Safe Steve	0.385059	0.870048	1.000000	-0.796004	-0.944866
Ricky Ranks	0.021978	-0.934285	-0.796004	1.000000	0.763700
Steve Ranks	-0.197996	-0.766741	-0.944866	0.763700	1.000000

corr method outputs pearson correlation coefficient from the variables in the data frame. pearson correlation coefficient only captures linear relationship, but it can be used to find possible multicollinearity between two variables.

```
print(cashflow)

np.irr(cashflow)
```

```
0   -107
1      5
2      5
3      5
4      5
5    105
Name: Cashflow, dtype: int64


Out[8]: 0.034517484085994976
```

irr method in numpy library outputs IRR from input cashflow. it is the discount rate where NPV becomes 0. it is useful when the sign of the cashflow does not change more than once.

UKHP.py: this file contains basic matplotlib and pickle methods.

```
In [2]: print(data['Average House Price'].mean())
        print(data['Average House Price'].std())
        print(data['Average House Price'].skew())
        print(data['Average House Price'].kurtosis())

        data.describe()
```

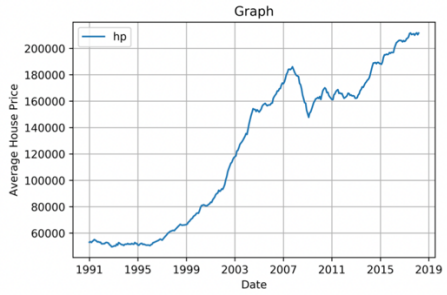
```
124660.48446512519
56387.16566469951
-0.11014547215
-1.59192678501
```

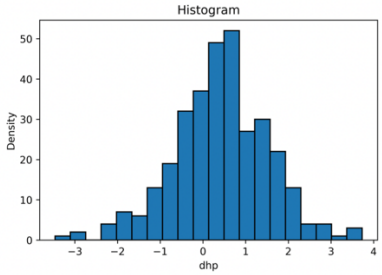
Basic information about the distribution of the data can be obtained using the pandas method. Not all information (such as number of sub-groups in the distribution) cannot be obtained from this. Just from this information, we can guess that the distribution is little bit skewed to right, and flatter than normal distribution.

```
def LogDiff(x):
    x_diff = 100*np.log(x/x.shift(1))
    return x_diff
```

```
                dhp
Month
1991-02-01  0.835451
1991-03-01 -1.135343
1991-04-01  1.472432
1991-05-01  1.310903
1991-06-01  1.318181
```

This function computes log difference of the data. log transformation is widely used especially when distribution is negatively skewed, although it is not always the case that log transformation is the most appropriate to account for heteroscedasticity.





Various graphs can be drawn using matplotlib. histogram can give visual information about the distribution that could not be seen from the statistical information computed before. Used data can be saved using pickle and to_excel method.

Chapter 3

Simple_linear_regression.py:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Spot   R-squared:                       1.000
Model:                            OLS   Adj. R-squared:                  1.000
Method:                 Least Squares   F-statistic:                 1.005e+06
Date:                Mon, 30 Jul 2018   Prob (F-statistic):               0.00
Time:                        14:53:33   Log-Likelihood:                -826.86
No. Observations:                 247   AIC:                             1658.
Df Residuals:                     245   BIC:                             1665.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -2.8378      1.489     -1.906      0.058      -5.771       0.095
Futures        1.0016      0.001   1002.331      0.000       1.000       1.004
==============================================================================
Omnibus:                      245.415   Durbin-Watson:                   1.326
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            10091.682
Skew:                          -3.814   Prob(JB):                         0.00
Kurtosis:                      33.371   Cond. No.                     5.05e+03
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 5.05e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               ret_spot   R-squared:                       0.989
Model:                            OLS   Adj. R-squared:                  0.989
Method:                 Least Squares   F-statistic:                 2.147e+04
Date:                Mon, 30 Jul 2018   Prob (F-statistic):          7.54e-240
Time:                        14:53:33   Log-Likelihood:                -157.16
No. Observations:                 246   AIC:                             318.3
Df Residuals:                     244   BIC:                             325.3
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0131      0.029      0.444      0.658      -0.045       0.071
ret_future     0.9751      0.007    146.543      0.000       0.962       0.988
==============================================================================
Omnibus:                       48.818   Durbin-Watson:                   2.969
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              671.062
Skew:                          -0.016   Prob(JB):                     1.91e-146
Kurtosis:                      11.091   Cond. No.                         4.45
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

First regression result illustrates linear relationship between return from S&P500 index and return from S&P500 futures. We can see that spot and future of S&P500 is closely related and the linear transformation of futures perfectly fits the spot price, explaining almost all the variances (rounded R^2 = 1). R^2 = 1 is usually odd, but in this case, it is understandable that these two variables are closely related although there may be other related variables.

Second regression is regressed after transforming data into log return (taking difference will give short run relationship). Estimating the coefficient on futures return on spot returns is identical to computing optimal hedging ratio from chapter 9 page 560~561 in the textbook. Formulas in the textbook is using change in spot price and futures price, but using log difference and scaling it will not affect the interpretation of the coefficient estimated in this regression. Therefore, the optimal hedging ratio based on the linear model is 0.9751 : 1. The intercept can be seen as cost of carry. By spot-future parity theorem, assuming continuous cost and return, it can be written that Future = Spot * e^(cost of carry) assuming efficient market because if else, there will be arbitrage opportunity. Log transforming this will give the same form of equation as the linear regression, meaning that the intercept can be seen as the cost of carry.

Chapter 4

hedging_revisited.py:

```
<F test: F=array([[ 2.58464834]]), p=0.10919227950361698, df_denom=245, df_num=1>
```

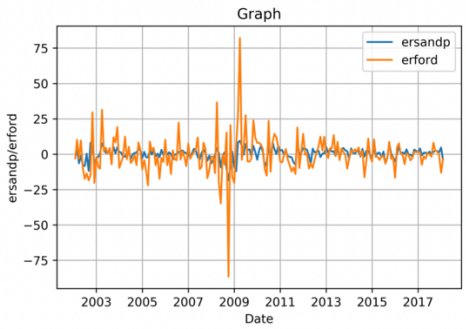
This is the result of the F-test (H0: beta = 1, H1:beta != 1) using the first regression (raw spot = beta*raw future + intercept). Idea behind this test is that the log run relationship between future and spot is 1:1. Looking at the result, we cannot reject the null hypothesis at 0.01 or 0.1 significance level, meaning that beta is likely to be 1. This is the expected result.

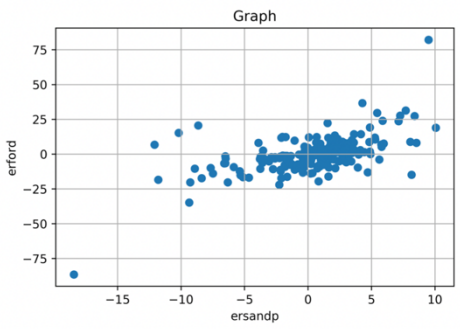
```
<F test: F=array([[ 14.02981315]]), p=0.00022456591761704512, df_denom=244, df_num=1>
```

This is the result of the F-test (H0: beta = 1, H1:beta != 1) using the second regression (transformed into log return). Idea behind this test is that the hedge ratio is usually not 1:1. the result suggests that we can reject H0 at 0.001 significance level, meaning that the optimal hedge ratio is not 1:1. This is the expected result.

Chapter 5

CAPM.py:





The graph on the left is the plot of two time series, log difference of S&P500 - US treasury bill rate (assumed to be risk free rate).It visually shows how two variables move together. the graph on the right the scatter plot that represents the relationship between two variables.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 erford   R-squared:                       0.337
Model:                            OLS   Adj. R-squared:                  0.334
Method:                 Least Squares   F-statistic:                     97.26
Date:                Mon, 30 Jul 2018   Prob (F-statistic):           8.36e-19
Time:                        15:11:43   Log-Likelihood:                -735.26
No. Observations:                 193   AIC:                             1475.
Df Residuals:                     191   BIC:                             1481.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -0.9560      0.793     -1.205      0.230      -2.520       0.608
ersandp        1.8898      0.192      9.862      0.000       1.512       2.268
==============================================================================
Omnibus:                       71.412   Durbin-Watson:                   2.518
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              677.387
Skew:                           1.079   Prob(JB):                    8.08e-148
Kurtosis:                      11.921   Cond. No.                         4.16
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

This is the result of linear regression where the dependent variable is return on FORD - risk free rate and the independent variable is market return (S&P500) - risk free rate, which represents CAPM formula. From these information, it is possible to test whether alpha is likely to be 0 or not and if beta is equal to some expected number.

```
<F test: F=array([[ 21.5604412]]), p=6.3653210360356986e-06, df_denom=191, df_num=1>
```

This is the result of the f test of H0: beta = 1 and H1: beta != 1. Null hypothesis can be rejected at 0.000001 significance level, meaning that beta is unlikely to be 1. This means that the movement of FORD and S&P500 is not equal.

Chapter 6

multi-hypothesis_CAPM.py: this file contains some explanation of pickle in addition to the F test conducted below

```
<F test: F=array([[ 12.9437402]]), p=5.3492561526069516e-06, df_denom=191, df_num=2>
```

This is the F test using the same model as code chapter 5, but now we conduct a joint test. We have the following hypotheses: H0: beta = 1 and alpha = 1, H1: one of the null hypotheses is violated. We can see that we reject the null hypothesis because the p-value of is very small. This is not surprising because we saw that beta = 1 was rejected in the previous section, and joint hypothesis will be rejected if at least one of them is rejected. Therefore, from this test, we cannot determine whether alpha is likely to be 1 or not.

Chapter 7

APT.py:

OLS Regression Results

Dep. Variable:	ermsoft	R-squared:	0.345
Model:	OLS	Adj. R-squared:	0.333
Method:	Least Squares	F-statistic:	28.24
Date:	Thu, 09 Aug 2018	Prob (F-statistic):	3.52e-31
Time:	11:22:44	Log-Likelihood:	-1328.3
No. Observations:	383	AIC:	2673.
Df Residuals:	375	BIC:	2704.
Df Model:	7		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.3260	0.475	2.789	0.006	0.391	2.261
ersandp	1.2808	0.094	13.574	0.000	1.095	1.466
dprod	-0.3030	0.737	-0.411	0.681	-1.752	1.146
dcredit	-0.0254	0.027	-0.934	0.351	-0.079	0.028
dinflation	2.1947	1.264	1.736	0.083	-0.291	4.681
dmoney	-0.0069	0.016	-0.441	0.659	-0.037	0.024

dspread	2.2601	4.140	0.546	0.585	-5.881	10.401
rterm	4.7331	1.716	2.758	0.006	1.359	8.107

Omnibus:	21.147	Durbin-Watson:	2.097
Prob(Omnibus):	0.000	Jarque-Bera (JB):	63.505
Skew:	-0.006	Prob(JB):	1.62e-14
Kurtosis:	4.995	Cond. No.	293.

This is the result of the multiple linear model expressing the APT model. Significant variables are ersandp and rterm at 0.01 significance level. ersandp is log difference of S&P500 and US treasury bill (risk free rate), rterm is the change in the difference between long dated bond and short dated bond. It is intuitive that the log return of the MICROSOFT is positively related to the market log return (S&P500). The coefficient of 1.2808 does not seem problematic because beta greater than 1 indicate greater systematic risk compared to market portfolio, and MICROSOFT stock alone is likely to be riskier than market portfolio (S&P500). rterm has the coefficient of 4.7331, which means the unit change in the difference between long dated and short dated bond price will change the long return of MICROSOFT by 4.73313%. Long dated bond is usually riskier, and it is intuitive that MICROSOFT (which is likely to have beta > 1, indicating higher systematic risk than market portfolio) is positively related to relatively risker asset. there may be many other interpretation about this.

```
<F test: F=array([[ 0.41387856]]), p=0.7986453783444192, df_denom=375, df_num=4>
```

This is the result of the F test where the hypotheses are H0: dpod = dcredit = dmoney = dspread = 0 and H1: at least one of the tested coefficient is non-zero. This is jointly testing all the insignificant variable (based on the individual t statistic). The result of F test and t test should be consistent (otherwise there may be multicollinearity), and we can see that F test outputs same results as t test at 0.01 significance level as expected.

step-wise_reg.py:

OLS Regression Results

Dep. Variable:	ermsoft	R-squared:	0.339
Model:	OLS	Adj. R-squared:	0.334
Method:	Least Squares	F-statistic:	64.58

Date:	Tue, 31 Jul 2018	Prob (F-statistic):	1.02e-33
Time:	11:34:44	Log-Likelihood:	-1323.8
No. Observations:	381	AIC:	2656.
Df Residuals:	377	BIC:	2671.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.0225	0.404	2.533	0.012	0.229	1.816
ersandp	1.2595	0.092	13.642	0.000	1.078	1.441
rterm	4.8402	1.721	2.812	0.005	1.456	8.224
dinflation	2.2094	1.217	1.816	0.070	-0.183	4.602

Omnibus:	21.219	Durbin-Watson:	2.075
Prob(Omnibus):	0.000	Jarque-Bera (JB):	64.060
Skew:	0.009	Prob(JB):	1.23e-14
Kurtosis:	5.009	Cond. No.	18.7

This is the result of the forward stepwise regression. Forward stepwise regression starts with 0 variables and iteratively adds the variables and only the significant variables remain. As criticized in chapter 5 of the textbook, starting with 0 variable is problematic than starting with overspecified model. This is due to the effect of underspecification on the statistical inference. However, the stepwise model has reached the similar conclusion with overspecified model shown in the previous section, so it may not be as problematic in this case. ersandp and rterm remained significant, and dinflation remained weakly significant (0.1 significance level). the signs of the coefficients are identical, and the scale of the coefficients are very similar. All the insignificant variable in the previous regression are removed. Note that the adjusted R^2 is a little bit higher than the overspecified model, although maximization of the R^2 is not the objective of the estimation.

Chapter 8

CAPM_quantreg.py:

0.1 quantile

QuantReg Regression Results

Dep. Variable:	erford	Pseudo R-squared:	0.2036
Model:	QuantReg	Bandwidth:	5.609
Method:	Least Squares	Sparsity:	47.54
Date:	Thu, 23 Aug 2018	No. Observations:	193
Time:	20:55:13	Df Residuals:	191
		Df Model:	1

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-11.9291	1.045	-11.420	0.000	-13.989	-9.869
ersandp	2.3404	0.348	6.720	0.000	1.653	3.027

0.2 quantile

QuantReg Regression Results

Dep. Variable:	erford	Pseudo R-squared:	0.2005
Model:	QuantReg	Bandwidth:	4.710
Method:	Least Squares	Sparsity:	25.26
Date:	Thu, 23 Aug 2018	No. Observations:	193
Time:	20:55:13	Df Residuals:	191
		Df Model:	1

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-7.3496	0.733	-10.021	0.000	-8.796	-5.903
ersandp	1.8044	0.186	9.694	0.000	1.437	2.172

0.3 quantile

QuantReg Regression Results

Dep. Variable:	erford	Pseudo R-squared:	0.1923
Model:	QuantReg	Bandwidth:	5.074
Method:	Least Squares	Sparsity:	19.85
Date:	Thu, 23 Aug 2018	No. Observations:	193
Time:	20:55:13	Df Residuals:	191
		Df Model:	1

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-4.8783	0.661	-7.381	0.000	-6.182	-3.575
ersandp	1.6596	0.163	10.187	0.000	1.338	1.981

0.4 quantile

QuantReg Regression Results

Dep. Variable:	erford	Pseudo R-squared:	0.1753
Model:	QuantReg	Bandwidth:	5.192
Method:	Least Squares	Sparsity:	18.09
Date:	Thu, 23 Aug 2018	No. Observations:	193
Time:	20:55:13	Df Residuals:	191
		Df Model:	1

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-3.3479	0.641	-5.223	0.000	-4.612	-2.084
ersandp	1.5005	0.155	9.669	0.000	1.194	1.807

0.5 quantile

QuantReg Regression Results

Dep. Variable:	erford	Pseudo R-squared:	0.1724
Model:	QuantReg	Bandwidth:	5.340
Method:	Least Squares	Sparsity:	16.78
Date:	Thu, 23 Aug 2018	No. Observations:	193
Time:	20:55:13	Df Residuals:	191
		Df Model:	1

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-1.4896	0.606	-2.457	0.015	-2.685	-0.294
ersandp	1.4384	0.146	9.822	0.000	1.150	1.727

0.6 quantile

QuantReg Regression Results

Dep. Variable:	erford	Pseudo R-squared:	0.1679
Model:	QuantReg	Bandwidth:	5.155
Method:	Least Squares	Sparsity:	17.05
Date:	Thu, 23 Aug 2018	No. Observations:	193
Time:	20:55:13	Df Residuals:	191
		Df Model:	1

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-0.0188	0.605	-0.031	0.975	-1.212	1.174
ersandp	1.2967	0.150	8.658	0.000	1.001	1.592

0.7 quantile

QuantReg Regression Results

Dep. Variable:	erford	Pseudo R-squared:	0.1611
Model:	QuantReg	Bandwidth:	4.881
Method:	Least Squares	Sparsity:	20.29
Date:	Thu, 23 Aug 2018	No. Observations:	193
Time:	20:55:13	Df Residuals:	191
		Df Model:	1

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.7568	0.676	2.597	0.010	0.423	3.091
ersandp	1.5283	0.182	8.396	0.000	1.169	1.887

0.8 quantile

QuantReg Regression Results

Dep. Variable:	erford	Pseudo R-squared:	0.1357
Model:	QuantReg	Bandwidth:	4.653
Method:	Least Squares	Sparsity:	29.42
Date:	Thu, 23 Aug 2018	No. Observations:	193
Time:	20:55:13	Df Residuals:	191
		Df Model:	1

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	4.0013	0.856	4.674	0.000	2.313	5.690
ersandp	1.6199	0.251	6.454	0.000	1.125	2.115

0.9 quantile

QuantReg Regression Results

Dep. Variable:	erford	Pseudo R-squared:	0.1063
Model:	QuantReg	Bandwidth:	4.810
Method:	Least Squares	Sparsity:	88.43
Date:	Thu, 23 Aug 2018	No. Observations:	193
Time:	20:55:13	Df Residuals:	191
		Df Model:	1

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	10.4563	1.951	5.359	0.000	6.608	14.305
ersandp	1.8473	0.699	2.644	0.009	0.469	3.225

Above are the results of the quantile linear regression where the selected quantiles are 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9. below is the summarized version of the quantile linear

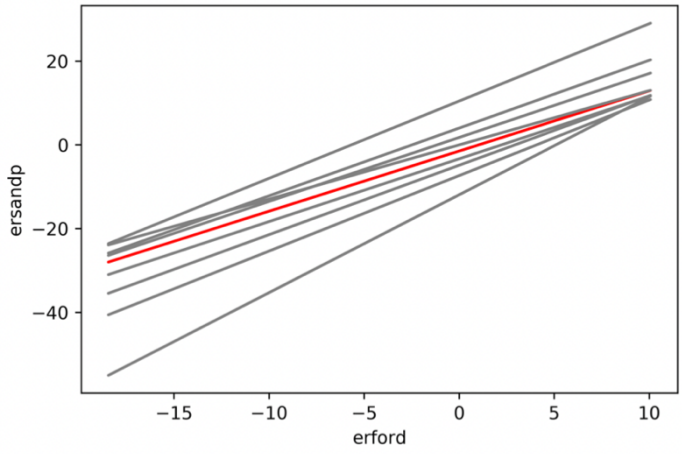
regression results.

	q	alpha	beta	lb	ub
0	0.1	-11.929063	2.340422	1.653477	3.027366
1	0.2	-7.349600	1.804414	1.437253	2.171575
2	0.3	-4.878332	1.659638	1.338293	1.980982
3	0.4	-3.347896	1.500533	1.194442	1.806624
4	0.5	-1.489629	1.438449	1.149566	1.727332
5	0.6	-0.018844	1.296706	1.001296	1.592115
6	0.7	1.756783	1.528341	1.169294	1.887389
7	0.8	4.001336	1.619863	1.124772	2.114954
8	0.9	10.456266	1.847333	0.469290	3.225376

In each regression, the data points are weighted differently in the loss function according to the selected quantile so that the estimated line captures the tendency at selected quantile level. The dependent variable is log return of FORD and independent variable is log return of S&P500 in all regression. Alpha is the estimated intercept, beta is the coefficient or the slope, and 1b ub is the coefficient at [0.025 ~ 0.975] confidence interval. We observe that each model has different alpha and beta, capturing g different tendency at different quantile level.

Date	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
2002-02-01	-17.183667	-11.400785	-8.604472	-6.716822	-4.719166	-2.930147	-1.674577	0.364496	6.308721
2002-03-01	-3.841712	-1.114428	0.856563	1.837213	3.480946	4.461936	7.037983	9.598789	16.839746
2002-04-01	-27.105056	-19.049959	-15.639919	-13.077801	-10.816960	-8.427072	-8.153439	-6.502338	-1.522394
2002-05-01	-14.407484	-9.260408	-6.635828	-4.936906	-3.012894	-1.392009	0.138325	2.285960	8.500008
2002-06-01	-29.869803	-21.181518	-17.600453	-14.850385	-12.516204	-9.958874	-9.958874	-8.415887	-3.704654

This is the first five rows of the estimated values of dependent variable at different quantile level, ad below is the graph representing the estimated lines.



We observe that the slope of the line is increasing at the upper quantile and lower quantile. this can be interpreted that systematic risk of FORD tend to be higher in both upper and lower extreme situation.

Chapter 9

PCA.py:

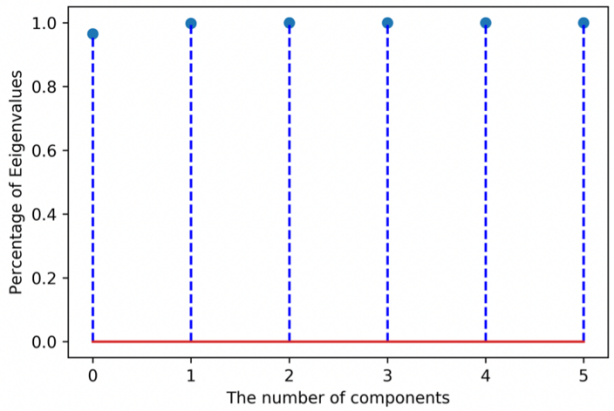
Principle components are computed after normalizing the US treasury bill series. PCA is computed using the existing library in numpy.

	GS3M	GS6M	GS1	GS3	GS5	GS10
0	0.407769	0.409146	0.411719	0.414379	0.409883	0.396356
1	0.417793	0.391519	0.293202	-0.091793	-0.361692	-0.668541
2	-0.468801	-0.152028	0.231295	0.588798	0.293663	-0.520283
3	-0.537538	0.192442	0.624653	-0.154963	-0.414705	0.296368
4	-0.307086	0.506547	-0.082492	-0.524346	0.580709	-0.173614
5	0.236960	-0.602142	0.542243	-0.417407	0.325168	-0.085348

Above is the estimated principal component (eigenvectors of the covariance matrix) of the US treasury bills.

```
0    0.965461
1    0.997919
2    0.999615
3    0.999924
4    0.999976
5    1.000000
Name: Eigenvalue, dtype: float64
```

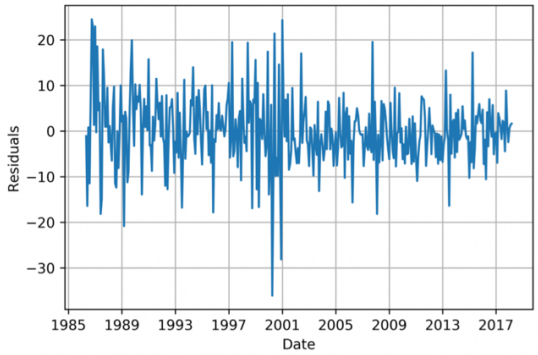
This is the ordered cumulative eigenvalues, or the variance explained by the variance-optimized linear combination of the variables. We can see that the first principal component explains most (96.5%) of the variance in the variable space and other principal components explain less than 4% of the total variance. This means that the dimension of the variables used in this analysis can be effectively reduced to one (or maybe two) variables, possibly reducing the redundancy. This might also mean that all variables share similar underlying pattern or factor, which is intuitive because they are all US treasury bills.



Above is the graphical representation of the ordered cumulative variance explained by each principal components, and we can see that most of the variance is explained by the first component.

Chapter 10

heteroscedasticity_testing.py:



The graph above is the time series plot of the residuals from the APT model estimated in code chapter 7.

```
[('Lagrange multiplier statistic', 3.1606601504201448),
 ('p-value', 0.86975267437865456),
 ('f-value', 0.4457702552712059),
 ('f p-value', 0.87290034041883569)]
```

Above is the result of the breusch-pagan heteroskedasticity test of the residuals of the estimated APT model. Heteroskedasticity is one of the essential assumptions of CLRM, which allows the estimated coefficients to be BLUE. The coefficients will still be unbiased and consistent under other CLRM assumptions, but they won't be BLUE anymore because standard error is higher than possible BLUE estimator. This will affect the accuracy of the estimator and reliability of the statistical inference that uses standard error in the process. Therefore heteroskedasticity should be diagnosed, and detecting it is one way to start. Breusch-pagan test was not mentioned in the textbook, but it is identical to Lagrange multiplier test mentioned in the textbook. Breusch-pagan test is based on the Lagrange multiplier test principle which utilizes lagrange multipler method to find the maximum of the likelihood function, and it tests for heteroskedasticity by utilizing the derived test statistics that follows chi-square distribution. The code estimated the F statistic from the auxiliary regression, which is same as the White's test explained in the textbook. Looking at the result, both White's test and breusch-pagan test output high p-value, not rejecting the joint null hypothesis that all the coefficients of the auxiliary regression model are 0 (independent

variables in the auxiliary model are independent variables in the original regression and multiple of those variables). This means that the residual is not a function of independent variables in the original model, supporting the homoskedasticity assumption (i.e. residual variance is constant = unlikely to depend on other variables).

white_modifed_standard_error.py:

OLS Regression Results

Dep. Variable:	ermsoft	R-squared:	0.345
Model:	OLS	Adj. R-squared:	0.333
Method:	Least Squares	F-statistic:	29.89
Date:	Thu, 09 Aug 2018	Prob (F-statistic):	9.27e-33
Time:	11:27:38	Log-Likelihood:	-1328.3
No. Observations:	383	AIC:	2673.
Df Residuals:	375	BIC:	2704.
Df Model:	7		
Covariance Type:	HC1		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	1.3260	0.459	2.888	0.004	0.426	2.226
ersandp	1.2808	0.093	13.778	0.000	1.099	1.463
dprod	-0.3030	0.635	-0.478	0.633	-1.547	0.941
dcredit	-0.0254	0.021	-1.219	0.223	-0.066	0.015
dinflation	2.1947	1.307	1.679	0.093	-0.367	4.756
dmoney	-0.0069	0.011	-0.630	0.528	-0.028	0.014
dspread	2.2601	3.428	0.659	0.510	-4.458	8.978
rterm	4.7331	1.727	2.741	0.006	1.349	8.117

Omnibus:	21.147	Durbin-Watson:	2.097
Prob(Omnibus):	0.000	Jarque-Bera (JB):	63.505
Skew:	-0.006	Prob(JB):	1.62e-14
Kurtosis:	4.995	Cond. No.	293.

Above is the result of the APT regression estimated in code chapter 7, but after correcting the standard error by utilizing White's estimator. if residuals are heteroskedastic, then the standard error of the estimators are inflated. This will affect the statistical inference made after estimation, so there is a need for diagnosis. Although it does not get rid of heteroskedasticity from the data, White's estimator allows us to estimate more realistic standard error assuming heteroskedasticity. It is based on White's proof that non-constant covariance matrix (covariance matrix that does not assume homoskedasticity) of error term can also be estimated from the residuals, and it should be utilized when heteroskedasticity is likely to affect the analysis and when it is difficult to get rid of it. We observe from the regression result that the absolute value of the t statistic became larger. This is because the standard error is corrected and became smaller than the original regression model in code chapter 7. This will make it easier for each variables to be significant in stricter significance

level.

newey-west_standard_error.py:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                ermsoft   R-squared:                       0.345
Model:                            OLS   Adj. R-squared:                  0.333
Method:                 Least Squares   F-statistic:                     24.93
Date:                Thu, 09 Aug 2018   Prob (F-statistic):           6.73e-28
Time:                        11:28:44   Log-Likelihood:                -1328.3
No. Observations:                 383   AIC:                             2673.
Df Residuals:                     375   BIC:                             2704.
Df Model:                           7
Covariance Type:                  HAC
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      1.3260      0.503      2.638      0.008       0.341       2.311
ersandp        1.2808      0.100     12.822      0.000       1.085       1.477
dprod         -0.3030      0.522     -0.581      0.561      -1.325       0.719
dcredit       -0.0254      0.022     -1.135      0.256      -0.069       0.018
dinflation     2.1947      1.314      1.671      0.095      -0.380       4.769
dmoney        -0.0069      0.011     -0.628      0.530      -0.028       0.015
dspread        2.2601      2.842      0.795      0.426      -3.310       7.830
rterm          4.7331      1.759      2.692      0.007       1.286       8.180
==============================================================================
Omnibus:                       21.147   Durbin-Watson:                   2.097
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               63.505
Skew:                          -0.006   Prob(JB):                     1.62e-14
Kurtosis:                       4.995   Cond. No.                         293.
==============================================================================
```

Above is the regression result of the APT model estimated in chapter 7, but after correcting the standard error estimate by Newey-West procedure. Different from the White's estimator, this correction method also considers the presence of the autocorrelation that will also have negative effect on BLUE. Although the possible number of lags are determined arbitrary (there may be formal way to choose), it is a useful method to account for two essential assumption of CLRM. From the given results, we observe that the absolute value of the test statistic for some variables have increased, while it didn't increase for the variables that were significant in the original regression. The difference with the White's method of correction comes from consideration of possible autocorrelation, but it made it easier for some variables to be significant at lower significant level similar to the White's method.

autocorrelation.py:

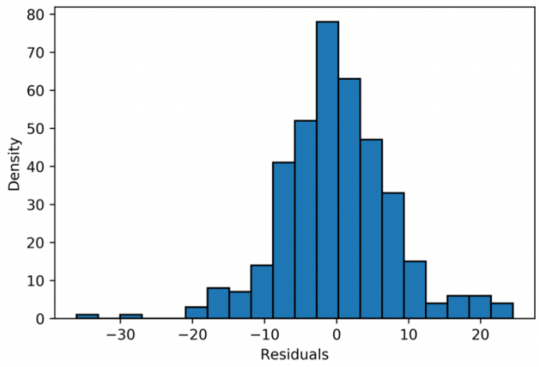
```
2.0973940504299913
```

Above is the Durbin Watson statistic derived from the APT model estimated in code chapter 7. Durbin Watson statistic of value close to 2 means there is no autocorrelation (assuming lag = 1). Note that the DW test imposes some assumptions on the original model. The result shows that there is no first order lag autocorrelation in the residuals. To test the presence of higher order lag autocorrelation, Breusch-Godfrey test is commonly used.

```
[('Lagrange multiplier statistic', 4.766591436782349),
 ('p-value', 0.9062145800242255),
 ('f-value', 0.45998207324796836),
 ('f p-value', 0.91501799815728901)]
```

Above is the result of Breusch-Godfrey test. the procedure and underlying principle is similar to the breusch-pagan test of heteroskedasticity. Different from the DW test, the estimated regression model contains higher order lagged residuals and tests whether the residuals can be expressed as the function of previous residuals. Both lagrange multiplier test and f test fails to reject the joint null hypothesis that all the coefficients of the lagged residuals are 0. This means that it is unlikely that there is autocorrelation even in the higher order lags, supporting the CLRM assumption.

non-normality.py:



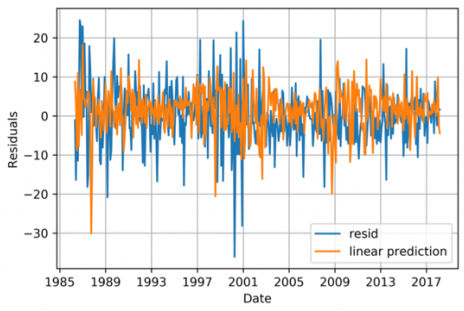
Above is the histogram representing the distribution of the APT model estimated in chapter 7. Visual representation gives us informal information about the distribution, but more formal information is desired to check the normality assumption of the residuals. this assumption is essential for the statistical inference after estimation.

```
[('Jarque-Bera', 63.50547267158968),
 ('Chi^2 two-tail prob.', 1.6216675409916346e-14),
 ('Skew', -0.00561267747998391),
 ('Kurtosis', 4.99482560590665)]
```

Jarque Bera test is a popular method for testing normality, and it is based on the lagrange multiplier test principle. it is proven that the test statistic derived by lagrange multiplier

follows chi-square distribution. From the shown result, we can reject the null hypothesis that the distribution of the residuals is normal at strict significance level.

dummy_variable.py:




```
Date
2000-04-01    -36.075347
2000-12-01    -28.143156
dtype: float64
```

The above graph on the left is the time series plot of the predicted value of the dependent variable and the residuals of the regression. We want to reduce the effect of large outliers on the regression result, so we detected two large negative residuals shown in the right figure above.

OLS Regression Results

Dep. Variable:	ermsoft	R-squared:	0.406
Model:	OLS	Adj. R-squared:	0.392
Method:	Least Squares	F-statistic:	28.33
Date:	Thu, 09 Aug 2018	Prob (F-statistic):	2.22e-37
Time:	11:31:57	Log-Likelihood:	-1309.7
No. Observations:	383	AIC:	2639.
Df Residuals:	373	BIC:	2679.
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.4198	0.454	3.125	0.002	0.526	2.313
ersandp	1.2539	0.090	13.897	0.000	1.076	1.431
dprod	-0.3211	0.705	-0.456	0.649	-1.707	1.065
dcredit	-0.0157	0.026	-0.603	0.547	-0.067	0.035
dinflation	1.4421	1.215	1.187	0.236	-0.946	3.830
dmoney	-0.0057	0.015	-0.383	0.702	-0.035	0.024
dspread	1.8693	3.955	0.473	0.637	-5.908	9.647
rterm	4.2642	1.641	2.599	0.010	1.038	7.490
APR0DUM	-37.0288	7.576	-4.888	0.000	-51.926	-22.132
DEC0DUM	-28.7300	7.546	-3.807	0.000	-43.569	-13.891

Omnibus:	20.084	Durbin-Watson:	2.088
Prob(Omnibus):	0.000	Jarque-Bera (JB):	28.877
Skew:	0.404	Prob(JB):	5.36e-07
Kurtosis:	4.076	Cond. No.	560.

Above is the regression result of the APT model estimated in code chapter 7, but in this regression it includes two dummy variables: two outliers detected previously. This is identical to eliminating the effect of those two outliers on the coefficient of other variables. We observe that the test statistics increased from the original model in chapter 7, and adjusted R^2 increased as expected. The model now better captures the (linear) general tendency of the

data.

multicollinearity.py:

	dprod	dcredit	dinflation	dmoney	dspread	rterm
dprod	1.000000	0.094273	-0.143551	-0.052514	-0.052756	-0.043751
dcredit	0.094273	1.000000	-0.024604	0.150165	0.062818	-0.004029
dinflation	-0.143551	-0.024604	1.000000	-0.093571	-0.227100	0.041606
dmoney	-0.052514	0.150165	-0.093571	1.000000	0.170699	0.003801
dspread	-0.052756	0.062818	-0.227100	0.170699	1.000000	-0.017622
rterm	-0.043751	-0.004029	0.041606	0.003801	-0.017622	1.000000

Above is the correlation matrix of the independent variables used in the APT model regression in code chapter 7. Although informally, we observe that each pearson correlation coefficient is relatively low, indicating less likelihood of multicollinearity. However, more formal approach should be taken to examine multicollinearity such as VIF, comparison of t and F test, and testing the sensitiveness to the change in the regression settings.

reset_ramsey.py:

```
<class 'Statsmodels.stats.contrast.ContrastResults'>
<F test: F=array([[ 0.9937673]]), p=0.39574412394082514, df_denom=372, df_num=3>
```

Above is the result of the Ramsey RESET test for the APT model estimated in code chapter 7. RESET test aims to check whether the linear model is the appropriate specification for the data. It regresses the residuals from the original model on the powers of the estimated dependent variable and performs F test, which is identical to testing if the omitted non-linear variables are significant or not. We observe that we cannot reject the null hypothesis that all coefficients are 0, meaning that we cannot reject the statement that linear model is the appropriate model.

chow_test.py:

```
1.9894610789615745
```

Above is the test statistics of the chow test for the APT model estimated in chapter 7. This test assumes that the structural break point is known or obvious and tests if the estimated parameters are consistent throughout the considered time period. The p-value for this test statistic is approximately 0.1151, which means we cannot reject that the parameters are stable (i.e. at chosen breaking point, parameters are likely to be stable).

```
[('test statistic', 1.5352572458140492),
 ('pval', 0.017937116553892265),
 ('crit', [(1, 1.63), (5, 1.36), (10, 1.22)])]
```

Above is the result of the CUSUM test for APT model estimated in code chapter 7. CUSUM test is one of the recursive least square method that iteratively add the data points and see if the residuals (which are high at the start because few data points are used) become smaller. If residuals does not become smaller, that may indicate that the added data has different underlying structure. CUSUM test uses sum of normalized residuals to give a formal conclusion about the difference between residuals from small sample and large sample (i.e. the stability or the constancy of predictive power of the parameters). Different from the chow test, this test does not assume known structural breaking point because we are checking almost all the possible breaking points during the iterative process. From the test result, we observe that we can reject the null hypothesis that the parameters are stable, meaning that there is a significant structural breaking point in the data.

Notes

Following are my notes and thoughts on the subjects mentioned in the textbook that I felt like researching a little bit more on.

Notes on eigenvalue & eigenvector:

$$A\boldsymbol{x} = \lambda\boldsymbol{x}$$

The computation process of eigenvalue decomposes any linear transformation (in the form of matrix) into direction and size of transformation. It can also be seen as creating a pseudo scaler to express a matrix with respect to certain vector.

linear transformation can actually express many kinds of geometric transformation

Eigenvalues of geometric transformations

	Scaling	Unequal scaling	Rotation	Horizontal shear	Hyperbolic rotation
Illustration					
Matrix	$\begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix}$	$\begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}$	$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$	$\begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} \cosh\varphi & \sinh\varphi \\ \sinh\varphi & \cosh\varphi \end{bmatrix}$
Characteristic polynomial	$(\lambda - k)^2$	$(\lambda - k_1)(\lambda - k_2)$	$\lambda^2 - 2\cos(\theta)\lambda + 1$	$(\lambda - 1)^2$	$\lambda^2 - 2\cosh(\varphi)\lambda + 1$
Eigenvalues, λ_i	$\lambda_1 = \lambda_2 = k$	$\lambda_1 = k_1$ $\lambda_2 = k_2$	$\lambda_1 = e^{i\theta} = \cos\theta + i\sin\theta$ $\lambda_2 = e^{-i\theta} = \cos\theta - i\sin\theta$	$\lambda_1 = \lambda_2 = 1$	$\lambda_1 = e^{\varphi} = \cosh\varphi + \sinh\varphi$ $\lambda_2 = e^{-\varphi} = \cosh\varphi - \sinh\varphi$
Algebraic mult., $\mu_i = \mu(\lambda_i)$	$\mu_1 = 2$	$\mu_1 = 1$ $\mu_2 = 1$	$\mu_1 = 1$ $\mu_2 = 1$	$\mu_1 = 2$	$\mu_1 = 1$ $\mu_2 = 1$
Geometric mult., $\gamma_i = \gamma(\lambda_i)$	$\gamma_1 = 2$	$\gamma_1 = 1$ $\gamma_2 = 1$	$\gamma_1 = 1$ $\gamma_2 = 1$	$\gamma_1 = 1$	$\gamma_1 = 1$ $\gamma_2 = 1$
Eigenvectors	All nonzero vectors	$\mathbf{u}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ $\mathbf{u}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$	$\mathbf{u}_1 = \begin{bmatrix} 1 \\ -i \end{bmatrix}$ $\mathbf{u}_2 = \begin{bmatrix} 1 \\ +i \end{bmatrix}$	$\mathbf{u}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$	$\mathbf{u}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ $\mathbf{u}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$

It can simplify/summarize complex linear transformation into two kinds of information (direction and size) which is more intuitive in many cases.

Because of its wide application, it is difficult to understand what makes eigenvalues so useful. It arises in optimization problem or shows up as a computational process to simplify the problem. It usually illustrates key characteristics to infer something from the given information. For example it arises in the PCA algorithm when finding a linear combination of given data that maximizes the explained variance. Lagrange multiplier method is used during the computation process of PCA, and it ends up as a form of eigenvalue problem. It's interesting that it can be shown that maximum explained variance is the eigenvalue of a covariance matrix. Eigenvalue also arises in other areas such as vibration problems and Schrödinger equation in physics, image processing (known as eigenfaces), and basic reproduction number in the field of epidemiology.

Maybe it is better to say that it "arises" rather than it is "used". Because linear transformation can come in various forms, and many things can be expressed as linear transformation (sometimes unexpectedly) by applying matrix/vector representation, the equation form of eigenvalue computation arises in many places maybe due to its simplifying characteristics and special case properties (such as orthogonality).

https://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors

Notes on copula:

$$f(x_1, \ldots, x_n) = c(F_1(x_1), \ldots, F_n(x_n)) \prod_{i=1}^{n} f_i(x_i)$$

from this equation, copula looks like a function that represents a ratio of joint distribution

density function when variables are independent and JDDF when dependent. this is kind of intuitive as copula being a representation of dependency expressed as a function.

$$\Pr(X_1 \leq x_1, \ldots, X_n \leq x_n) = F(x_1, \ldots, x_n) = C(F_1(x_1), \ldots, F_n(x_n))$$

but only when it is cumulative function.

In probabilistic terms, $C : [0,1]^d \to [0,1]$ is a *d*-dimensional **copula** if *C* is a joint cumulative distribution function of a *d*-dimensional random vector on the unit cube $[0,1]^d$ with uniform marginals.[4]

you make it a cumulative function and perform a change in variable (変数変換…確率積分変換) to make the distribution fit [0,1] probably because you want to satisfy the conditions for the existence of copula. also, it is known that if variable between [0,1] and probability density function is monotonic, function resulting from change in variable will be a uniform distribution. this is why cumulative distribution of x which is packed between [0,1] from the perspective of y will be converted to uniform distribution marginal by performing change in variable. interestingly, change in variable is same as getting an inverse function in this case, and distribution of y becomes uniform distribution for any distribution of x. this is because taking distribution is just counting the number of points without measuring the distance. just counting the frequency from y's perspective in cumulative distribution will obviously give uniform distribution because x and y has one-to-one relationship. changing the variable of PDF to the y (to pack the distribution between [0,1]) means taking marginal distribution of uniform distribution (=counting frequency from y's perspective). I wonder if we can pack the distribution between [0,1] without making it a uniform distribution, and I also wonder how it will change the application and interpretation.

horizontally smoosh the time series data => find a proxy PDF for the distribution => make it a cumulative DF => convert it to uniform margin by variable conversion => input fits the condition of the existence of copula … copula representing the same joint CDF as the real (although proxied) joint cumulative distribution without assuming independence, and PDF representing the dependence.

to conclude, I would say that copula is a method to model a joint distribution of multiple distribution that are not independent. I am still not clear about the interpretation of copula density function with uniform margin. my current interpretation is that it is controlling for the original distribution or frequency to see which part is more correlated or dependent with each other when each value pairs are equally probable, but I need to study more to get more clear understanding of the application and its interpretation.

https://fmurakami.securesite.jp/unifstudy1.html

https://multivariate-statistics.com/2021/12/30/statistics-probabliity-density-function/

https://nounai-librarian.com/2021-03-14-063625/
https://en.wikipedia.org/wiki/Copula_(probability_theory)
https://www.ism.ac.jp/editsec/toukei/pdf/68-1-087.pdf

Further notes and research summary on other concepts in the textbook is in this quizlet set below.
https://quizlet.com/jp/815926225/chapter-15-summer-assignment-flash-cards/?i=220rpq&x=1jqt