FAQ for Python Stream Assignment

This memorandum assembles frequently asked questions during the office hours. I hope that you will find it helpful when finishing the assignment. Yet, as there can be multiple approaches to solve a coding problem, the hints/functions/commands provided here are for reference only.

Some notations to keep in mind:

varlist: a list of variables

pd: Pandas module

df: a Pandas-dataframe

Part I (a)

1. What file on TRACE do I have to use?

If you are writing SQL scripts, then you can select data from trace; if you are using WRDS web to download the data, you should use BTDS file on TRACE.

2. What variables do I have to use?

cusip_id, yld_pt, trd_exctn_dt, trd_exctn_tm

yld_pt: bond yields in percentage points;

trd_exctn_dt: trade execution date;

trd_exctn_tm: trade execution time;

3. How do I keep one observation and drop all other observations for each group?

First, sort your dataframe such that the desired observation is the first one (or last one) within each group. Then drop other observations within each group by invoking drop_duplicates( ) function.

Useful commands:

1. df. sort_values(by= varlist) (For more details, see here)

2. df.drop_duplicates() (For more details, see here)

4. *How do I aggregate daily observations to monthly observations? In other words, how do I find out the median among daily observations?*

There are several ways to solve the problem. For example, once you obtain daily yields, you can sort and group the data, such that each group contains sorted yld_pt values for a bond within a month. Then pick the observation in the middle within each group.

Useful commands:

1. df. sort_values(by= varlist) (For more details, see here)

2. df.groupby([varlist])[X].rank( ) (For more details, see here)

This function ranks the variable X within each group generated by the varlist

3. df.groupby([varlist]).size() **(For more details, see here)**

This function returns the number of observations within each group generated by the varlist;

5. What is the expected result of part I (a)?

	cusip_id	yld_pt	trd_exctn_dt
33	00077TAB0	6.842175	2015-01-06
34	00077TAB0	7.002821	2015-02-17
35	00077TAB0	6.816467	2015-03-31
36	00077TAB0	6.029157	2015-04-24
38	00077TAB0	6.484927	2015-06-10

This table shows part of the expected result for the bond with cusip_id 00077TAB0. Each month from January 2015 to June 2015 has one observation, except May in which the bond was not traded.

Part I (b)

1. What file on CRSP do I have to use?

The dsf file.

2. What variables to I have to use?

permno, permco, date, ret

permno: stock identifier

permco: stock issuer identifier

ret: stock return

3. What does 180-day volatility mean?

It means to calculate the volatility using previous 180-day data. For example, if you try to calculate the 180-day volatility on Jan 6th, 2015, you should use 180-trading-day stock return data on and before Jan 6th, 2015.

4. How do I calculate the 180-day volatility?

Useful command:

df.groupby('permno')['ret'].rolling(180).std() (For more details, see here)

Part I (c):

1.What is the identifier in Compustat?

It is gvkey. Once you finish part I (b), you can find a linking table in WRDS linking permno/permco to gvkey.

2. What file on Compustat do I have to use? What variables do I have to use?

Data: fundq;

Variables: gvkey, PRCCQ, CSHOQ, DLTTQ, LCTQ, ATQ, datadate

PRCCQ: stock closing price at the end of each quarter

CSHOQ: the number of common shares outstanding

DLTTQ: long-term debt

LCTQ: current total liabilities

ATQ: total assets

Part I (d):

1. What are the useful commands?

pd.merge_asof(df1 , df2, left_on=, right_on=, by='issue_id', direction='backward')

where df1 and df2 are two dataframes. (For more details, see here)

Part I (e):

1. How to calculate the time to maturity?

Make sure you have merged the bond transaction data you obtain from Part I (a) with the data in "acf351b_python_data.csv", such that your data contains both maturity and trd_exctn_dt (trade execution date).

Before calculating the difference, make sure that you have transformed the date variables of interest from text format to date format by invoking pd.to_datetime() function.

Useful commands:

1. pd.to_datetime() (For more details, see here)

2. Where can I download zero coupon yields?

http://www.federalreserve.gov/econresdata/researchdata/feds200628.xls

This excel spreadsheet contains daily zero coupon yield information. SVENYXX indicate zero coupon yields, where XX represents the maturity. For example, SVENY01 means the 1-year Treasury zero coupon yield; SVENY30 means the 30-year Treasury zero coupon yield.