CUFE·CAFD

TITLE：Investment Midterm Assignment

GROUP NAME：GGM Never Die

MEMBERS：高雅萌、郭好格、孟舒晨

DATE：12/12/2019

Question 1

This table lists the equal-weighted average returns and risk characteristics of stocks sorted by realized betas. For each month, we calculate β, β-, β+, using the past 12 months returns, then, all stocks listed using daily continuously compounded returns over the next month（t） as our **raw return**, finally, we use market return factor to get the **factor-adjusted return** (as we don't have the risk free rate, we consider the risk free rate as zero, so that the market return is our market return factor), For each risk characteristic, we rank stocks into quintiles (1–5). The sample period is from January 2016 to June 2019, and observations are at a daily frequency.

There are two kinds of method we use to measure the risk characteristic. First, we form **equal-weighted average daily return** as the daily market return, and then we get the market-adjusted excess return for each stock, and finally we form **equal-weighted portfolios** at the beginning of each 12 months period. The column labelled "Return" reports the equal-weighted average return in excess of the portfolio sorted by β-. The row labelled "High-Low" reports the difference between the returns of portfolio 5 and portfolio 1. The labelled "t-stat" is the t-statistic computed to test whether the difference in return is different from zero. The columns labelled "β", "β-"and "β+" report the time-series and cross-sectional average of equal-weighted individual stock betas over the 12-month holding period.

Then, we form **value-weighted average daily return** as the daily market return, and then we get the market-adjusted excess return for each stock, and finally we form **value-weighted portfolios** at the beginning of each 12 months period. The column labelled "Return" reports the value-weighted average return in excess of the portfolio sorted by β-. The row labelled "High-Low" reports the difference between the returns of portfolio 5 and portfolio 1. The labelled "t-stat" is the t-statistic computed to test whether the difference in return is different from zero. The columns labelled "β", "β-"and "β+" report the time-series and cross-sectional average of equal-weighted individual stock betas over the 12-month holding period.

Table 1 Stocks sorted by β-

equal weighted portfolio and equal weight market ret					
rank	ret	adjret	beta	d_beta	u_beta
low β-	-0.07	0.02	-0.456	-0.647	-0.321
2	-0.045	0	-0.139	-0.221	-0.064
3	-0.133	-0.016	0.022	0.016	0.031
4	-0.375	-0.041	0.196	0.257	0.14
high β-	-0.784	-0.059	0.444	0.646	0.3
high-low	-0.714	-0.079	0.9	1.293	0.621
t_stats	[-0.95734]	[-7.47037]			

Table 2 Stocks sorted by β-

value weighted portfolio and vaule weight market ret					
rank	ret	adjret	beta	d_beta	u_beta
low β-	-0.127	-0.015	-0.352	-0.538	-0.217
2	-0.123	-0.024	0.034	0.017	0.055

3	-0.049	-0.039	0.22	0.304	0.157
4	-0.103	-0.058	0.389	0.593	0.231
high β-	-0.19	-0.066	0.656	1.12	0.396
high-low	-0.063	-0.051	1.008	1.658	0.613
t_stats	[-0.21249]	[-4.19908]			

Tables report the average realized excess return on month t in each equally-weighted quintile portfolio. The table also reports the average cross-sectional realized β, β-or β+ of each quintile portfolio. These betas are computed over the same 12-month period. This use of overlapping information is more efficient.

By construction, higher β- or higher β+ must also mean higher unconditional β, so high average returns are accompanied by high β-, β+ and regular β. Note that for these portfolios sorted by realized $^{-}$, the spread in realized β- and β+ is also similar to the spread in realized β.

Tables show that stocks with high contemporaneous β- do not have high returns in the next month. Stocks in the quintile with the lowest (highest) β- earn -0.127% (-0.19%) monthly return in excess of market return. The average difference between quintile portfolio 5 and 1 is not statistically significant. These results are not consistent with agents disliking downside risk and avoiding stocks that covary strongly when the market dips. In fact, stocks with high β- must carry a premium in order to entice agents to hold them. However, the data says "no".

The average β- spread between quintile portfolios 5 and 1 is very large (the difference is bigger than 1), but sorting on β- also produces variation in β and β+. However, the variation in β or β+ is not as disperse as the variation in β-.

Question 2

PANEL A: β^{-}SORTS CONTROLLING FOR SKEWNESS

SKEWNESS QUINTILES

PORTFOLIO	1	2	3	4	5	AVERAGE
1 LOWβ^{-}	-0.00696	-0.00679	-0.0068	-0.00685	-0.00666	-0.00681
2	-0.0073	-0.00834	-0.00825	-0.00845	-0.00846	-0.00816
3	-0.0079	-0.00847	-0.00904	-0.00924	-0.00928	-0.00879
4	-0.00843	-0.0091	-0.00949	-0.00984	-0.00962	-0.0093
5 HIGHβ^{-}	-0.01016	-0.01047	-0.01118	-0.0113	-0.01077	-0.01077
HML	-0.00319	-0.00367	-0.00438	-0.00446	-0.00411	**-0.00396**
T	- 7.03211	- 6.86681	- 6.68421	- 7.76738	- 8.50841	**-8.13262**

The table above measures the magnitude of the reward for exposure to downside beta, while explicitly controlling for the effect of skewness. We first form 5 portfolios sorted on skewness based on residual. Then, within each skewness portfolio, we sort stocks into five equally weighted portfolios based on downside beta. Both skewness and downside beta are computed over the same 12-month horizon before the month we want to test. After forming the 25 portfolios, we average the returns of downside beta portfolio over the five skewness portfolios. This characteristic control procedure creates a set of portfolios with near-identical levels of skewness risk, which is reported at the column labeled ''Average''. And the row labeled ''High-low'' reports the differences in

average returns between the first and fifth downside beta portfolios within each skewness rank. The average excess return of 0.396% in the bottom right cell is the difference in average returns between the fifth and firs portfolios that control for skewness risk. This difference has a robust t-statistic of -8.13. Hence, the downside beta is not subsumed to the idiosyncratic skewness measure.

Question 3

All the factors we need are as follows:

- *Beta*: use rolling past 12-month returns to estimate beta;

$$r_{it} = \alpha_{it} + \beta_{it} vwretd_{it} + \varepsilon_{it},$$

  where r_i is stock i's return in day t and vwretd is value weighted market return. β_{it} is our required estimated beta.

- *Momentum*: use rolling past 12-month to estimate the momentum

$$momentum = \sum_{i=1}^{n} \ln(1 + r_i),$$

  where n is the daily number of past 12 months.

- *Size*: log of market capitalization

$$Size = ln(Dsmvtll),$$

  where Dsmvtll denotes total market capitalization.

- $Residual\ Volatility = std(residual)$, where the residual uses past 12 months' regression residuals.
- *Book-to-Price*: we use PE ratio to standard for the book-to-price ratio.
- *Liquidity*: we use share turnover to capture the liquidity feature.
- $Non-linear\ size$:

$$size^3{}_{it} = \delta_{0it} + \delta_{1it} size_{it} + \varepsilon_{it}$$

  The ε_{it} is our required *non-linear size* which is orthogonalized to the *Size* factor.

1) correlation matrix of the downside beta and these Barra style factors:

Table 1: the correlation matrix between downside beta and other Barra style factors

Pearson 相关系数
Prob > |r| under H0: Rho=0
观测数

	downbeta	beta	momentum	size	resivol	pe	turnover	nlsize
downbeta	1.00000 52676	0.55855 <.0001 52676	0.01799 <.0001 52676	-0.12375 <.0001 52671	0.16184 <.0001 52676	0.04900 <.0001 50685	0.10857 <.0001 51212	-0.12631 <.0001 52671
beta	0.55855 <.0001 52676	1.00000 52693	0.07716 <.0001 52693	-0.10743 <.0001 52688	0.22059 <.0001 52693	0.04410 <.0001 50702	0.06163 <.0001 51229	-0.11233 <.0001 52688
momentum	0.01799 <.0001 52676	0.07716 <.0001 52693	1.00000 52693	0.36247 <.0001 52688	0.24087 <.0001 52693	0.04231 <.0001 50702	0.22502 <.0001 51229	0.34721 <.0001 52688
size	-0.12375 <.0001 52671	-0.10743 <.0001 52688	0.36247 <.0001 52688	1.00000 72025	-0.22533 <.0001 52688	-0.04134 <.0001 70039	-0.14692 <.0001 69605	0.99747 <.0001 72025
resivol 残差	0.16184 <.0001 52676	0.22059 <.0001 52693	0.24087 <.0001 52693	-0.22533 <.0001 52688	1.00000 52693	0.07212 <.0001 50702	0.55668 <.0001 51229	-0.22533 <.0001 52688
pe	0.04900 <.0001 50685	0.04410 <.0001 50702	0.04231 <.0001 50702	-0.04134 <.0001 70039	0.07212 <.0001 50702	1.00000 70039	0.05692 <.0001 67619	-0.04308 <.0001 70039
turnover	0.10857 <.0001 51212	0.06163 <.0001 51229	0.22502 <.0001 51229	-0.14692 <.0001 69605	0.55668 <.0001 51229	0.05692 <.0001 67619	1.00000 69605	-0.14831 <.0001 69605
nlsize	-0.12631 <.0001 52671	-0.11233 <.0001 52688	0.34721 <.0001 52688	0.99747 <.0001 72025	-0.22533 <.0001 52688	-0.04308 <.0001 70039	-0.14831 <.0001 69605	1.00000 72025

2) pick the one with the largest correlation with the downside beta and run a time-series regression to show me whether downside beta is still significant:

From the results of question 1), we find beta has the largest correlation with the downside beta, therefore we pick beta out and run a regression to see if the downside beta is still significant. The regression result is shown as follows:

Table 2: the regression result

Variable	_NAME_	COL1
Intercept	PARAM	0.003*
	T	[1.70]
	p	*
beta	PARAM	-0.009***
	T	[-5.90]
	p	***
downbeta	PARAM	0.002***
	T	[3.24]
	p	***

We can find from the regression result, the coefficient of downside beta is still significant, which means that beta' effect can not cover the downside beta's effect. Downside beta is still a risk factor after considering the beta's effect and it is evaluated through some excess returns.

3) what is the "style" of downside beta?

Downside beta captures the risk when market is down. According to previous literatures, higher downside beta should be compensated by higher average return. But in Chinese A-share market, this feature seems to be insignificant from our results. To be worse, we get the conclusion that,

in Chinese stock market, the higher the downside beta, the lower the average return. And this kind of downside beta's effect is robust after considering other factors, such as the idiosyncratically skewness, normal beta and so on.

Assignment code:

Question 1:

```
options mprint;
libname worklib '/folders/myfolders/sasuser.v94/midterm';
*select data we need;
proc sql;
create table all_data as
select
          fullsample_0620.stkcd,
          fullsample_0620.prc_change_pct as ret,
    fullsample_0620.trddt as date,
    fullsample_0620.year,
    fullsample_0620.month,
    fullsample_0620.dsmvtll as size
from
    worklib.fullsample_0620;
quit;
data all_data;
    set all_data;
    yearmonth=year(date)*100+month(date);
run;
proc sort data=all_data;
          by stkcd date;
run;
data all_data;
          set all_data;
          if stkcd=lag(stkcd) then
                    lsize=lag(size);
          else
                    lsize=".";
run;
*define excess return;
proc sort data=all_data;
          by date;
run;
*value weight to get daily market ret;
proc means data=all_data noprint;
          var ret;
          weight lsize;
          output out=d1(drop=_type_ _freq_)
```

```
        mean(ret)=m_ret;
        by date;
        where lsize^=.;
run;
data all_data;
        merge all_data d1;
        by date;
        ar=ret-m_ret;
run;
* Keep only securities with returns in all days;
proc sort data=all_data;
        by stkcd yearmonth;
run;
proc means data=all_data noprint;
        output out=d1(drop=_type_ _freq_)
        n(ret)=n_ret;
        by stkcd yearmonth;
run;
proc sort data=d1;
        by yearmonth;
run;
proc means data=d1 noprint;
        output out=d11(drop=_type_ _freq_)
        max(n_ret)=max_ret;
        by yearmonth;
run;
data d1;
    merge d1 d11;
    by yearmonth;
    if n_ret=max_ret;
run;
proc sort data=d1;
        by stkcd yearmonth;
run;
data all_data;
        merge all_data d1;
        by stkcd yearmonth;
        if n_ret^=.;
run;
*creat indicator to select months;

data indct;
        set all_data;
        keep yearmonth;
```

```
run;
proc sort data=indct nodupkey;
         by yearmonth;
run;
data indct;
         set indct;
         ind=_n_;
run;
proc sort data=all_data;
         by yearmonth;
run;
data all_data;
         merge all_data indct;
         by yearmonth;
run;

 ***************************************************
*macro;

%macro run_cycle(numcycle);
%do i=1 %to &numcycle;
        %let m=12+&i;
* data test;
data test;
        set all_data;
        where &m-12<=ind<=&m;
run;
*define downside beta;
data test;
        set test;
        if m_ret>0 then downside=0;
        else downside=1;
run;
*calculate beta for individual stocks;
data test1;
        set test;
        if &m-12<=ind<=&m-1;
run;

 proc sort data=test1;
        by stkcd;
run;
proc reg data=test1 outest =test_1(rename=(m_ret=beta)) noprint;
        model ar=m_ret;
```

```
        by stkcd;
run;

 *calculate downsidebeta for individual stocks(by regression);
data d_test;
        set test1;
        if downside=1;
run;
proc sort data=d_test;
        by stkcd;
run;
proc reg data=d_test outest =test_d(rename=(m_ret=d_beta)) noprint;
        model ar=m_ret;
        by stkcd;
run;
* sigel portfolio rank;
proc sort data=test_d;
        by d_beta;
run;

proc rank data=test_d out=d1 group=5;
   var d_beta;
   ranks rrank;
run;

 *calculate upsidebeta for individual stocks;
data u_test;
      set test1;
      if downside=0;
run;
proc sort data=u_test;
      by stkcd;
run;
proc reg data=u_test outest =test_u(rename=(m_ret=u_beta)) noprint;
      model ar=m_ret;
      by stkcd;
run;
*calculate portfolio returns and beta(value weighted);
proc sort data=d1;
      by stkcd;
run;
proc sort data=test;
      by stkcd;
run;
data test;
```

```
      merge test d1(keep=stkcd rrank);
      by stkcd;
      if rrank^=.;
run;
*get the stkcd monthly return on month t;
data test2;
      set test;
      where ind=&m;
      l_ar=log(1+ar/100);
run;
proc sort data=test2;
      by stkcd date;
run;
proc means data=test2 noprint;
      output out=test_m(drop=_type_ _freq_)
      sum(l_ar)=l_ar;
      by stkcd;
run;
data test_m;
      set test_m;
      ret=(exp(l_ar)-1)*100;
run;
* use the size before the first day of t month as the stkcd size;
proc sort data=test2 nodupkey;
      by stkcd;
run;
data test_m;
      merge test_m test2;
      by stkcd;
run;
*get the factor-adjusted return;
proc sort data=test;
      by rrank;
run;
proc reg data=test outest =test_a(rename=(intercept=adjret)) noprint;
      model ar=m_ret;
      by rrank;
run;
data all;
      merge d1(keep=stkcd d_beta rrank)
                  test_1(keep=stkcd beta)
                  test_u(keep=stkcd u_beta)
                  test_m(keep=stkcd ret lsize);
      by stkcd;
```

```
run;
proc sort data=all;
      by rrank;
run;
proc means data=all noprint;
      var ret beta d_beta u_beta;
      weight lsize;
      output out=result(drop=_type_ _freq_)
      mean(ret)=returns
      mean(beta)=beta
      mean(d_beta)=d_beta
      mean(u_beta)=u_beta;
      by rrank;
run;
data result;
      merge result test_a(keep=rrank adjret);
      by rrank;
run;
data result;
      set result;
      ind=&m;
      format returns adjret beta d_beta u_beta 5.3;
run;
*find the return differences;
data low high;
      set result;
      if rrank=0 then output low;
      if rrank=4 then output high;
run;
data low;
      set low;
      rename returns=r_low;
      rename adjret=adj_low;
      rename beta=b_low;
      rename d_beta=db_low;
      rename u_beta=ub_low;
run;
data high;
      set high;
      rename returns=r_high;
      rename adjret=adj_high;
      rename beta=b_high;
      rename d_beta=db_high;
      rename u_beta=ub_high;
```

```
run;
data diff;
      merge low high;
      ret_d=r_high-r_low;
      adjret_d=adj_high-adj_low;
      beta_d=b_high-b_low;
      d_beta_d=db_high-db_low;
      u_beta_d=ub_high-ub_low;
run;
data diff;
      set diff;
      keep ind ret_d adjret_d beta_d d_beta_d u_beta_d;
      format ret_d adjret_d beta_d d_beta_d u_beta_d 5.3;
run;
%if &i=1 %then
  %do;
                  data d_cycles;
                           set result;
                  run;

                  data f_diff;
                           set diff;
                  run;
  %end;
  %else
  %do;
    proc append base=d_cycles data=result;
    run;

    proc append base=f_diff data=diff;
    run;
  %end;
%end;
%mend run_cycle;
%run_cycle(30);
proc means data=f_diff noprint;
      output out=diff(drop=_type_ _freq_)
      mean(ret_d)=ret_d
      mean(adjret_d)=adjret_d
      mean(beta_d)=beta_d
      mean(d_beta_d)=d_beta_d
      mean(u_beta_d)=u_beta_d
      t(ret_d)=t_stats
      t(adjret_d)=t;
```

```
run;
proc sort data=d_cycles;
      by rrank;
run;
proc sort data=d_cycles;
      by rrank;
run;
proc means data=d_cycles noprint;
      output out=cycles(drop=_type_ _freq_)
      mean(returns)=ret
      mean(adjret)=adjret
      mean(beta)=beta
      mean(d_beta)=d_beta
      mean(u_beta)=u_beta;
      by rrank;
run;
```

Question 2:

```
options ls=80;
libname worklib"D:\investment\data\midterm";
**********以dailyret计算beta**********;
data ret;
set worklib.Fullsample_0620;
run;/*导入数据到work*/
proc sort data=ret;
by stkcd trddt;
run;
proc expand data=ret out=ret method=none;
by stkcd;convert clsprc = lag_clsprc / transformout=(lag 1);
run;
data ret;
set ret;
ret=dsmvtll;
run;/*以价格的变化率作为收益率*/
proc sort data=ret;
by stkcd year month;
run;
data ret;
set ret;
yearmonth=year*100+month;
run;
**********以size为权重，计算每天的市场收益率**********;
data indct;
set ret;
```

```
keep year month;
run;
data indct;
set indct;
yearmonth=year*100+month;
run;
proc sort data=indct nodupkey;
by yearmonth;
run;
data indct;
set indct;
sequence = _n_;
run;/*对月排序标数*/
proc sql;
create table ret as
select
ret.*,
indct.sequence
from
ret,indct
where
ret.yearmonth=indct.yearmonth;
quit;/*将sequence加入总表ret*/
proc sort data=ret;
by stkcd yearmonth;
run;
data ret;
set ret;
size=dsmvtll;
run;/*对每个股票每天计算市值size*/
proc sort data=ret;
by trddt stkcd;
run;
proc means data=ret noprint;
weight size;
output out=mkt
mean(ret)=Dmkt;
by trddt;
run;/*对每天每个股票日收益率以市值size为权重算股票收益率，每天对应Dret，缺失值为每个股票的201601第一天lsize*/
proc sql;
create table ret as
select
ret.*,
```

```
mkt.dmkt
from
ret,mkt
where
ret.trddt=mkt.trddt;
quit;/*将市场日收益率加入总表*/
data downret;
set ret;
if Dmkt>0 then delete;
run;/*为计算downsidebeta只保留市场收益率为负数的天数*/
/************************对每个股票每个月算
rollingbeta***********************/
options symbolgen;
%macro rolling(startyearmonth,num);
%do i=1 %to &num;
/*****************************startyearmonth为循环初始月，num为循环次数，
yearmonth_cycle求每次循环所对应的初始月份
*******************************88********/
    %let m=mod(&startyearmonth,100);
    %let y=int(&startyearmonth/100);
    %let numcycle=12*(&y-2016)+&m+&i-1;
    data numcycle;
    numcycle=&numcycle;
    run;
/*********************************对每个numcycle求
beta*************************/

proc sql;
create table numcycle as
select
numcycle.*,
indct.yearmonth
from
numcycle,indct
where
indct.sequence=numcycle.numcycle;
quit;/*yearmonth加入numcycle*/
data test;
set downret;
where &numcycle-12<=sequence<=&numcycle-1;
run;
proc sort data=test;
by stkcd trddt;
run;
```

```
proc reg data=test noprint;
    model ret=Dmkt;
    output out=regresult residual=r;
by stkcd;
quit;/*得到r*/

proc reg data=test outest=regresultbeta(rename=(Dmkt=downsidebeta))
noprint;
     model ret=Dmkt;
by stkcd;
quit;
proc sort data=regresult;
by stkcd;
run;
proc sort data=regresultbeta;
by stkcd;
run;
proc sql;
create table regresult as
select
regresult.*,
regresultbeta.downsidebeta
from
regresult,regresultbeta
where
regresult.stkcd=regresultbeta.stkcd;
quit;/*将beta加入regresult*/
proc sort data=regresult;
by stkcd;
run;
Proc univariate data=regresult noprint;
var r;
Output out=ske
Skewness=ske;
By stkcd;
Run;/*求残差的偏度*/
proc sql;
create table regresult as
select
regresult.*,
ske.ske
from
regresult,ske
where
```

```
regresult.stkcd=ske.stkcd;
quit;/*将ske加入regresult*/
proc rank data=regresult out=skerank group=5;
var ske;
ranks skerank;
run;/**根据所求月ske大小将公司股票分为五组*/

proc sort data=skerank nodupkey;by stkcd;run;
proc sort data=skerank;by skerank;run;
proc rank data=skerank out=skebetarank group=5;
var downsidebeta;
ranks betarank;
by skerank;
run;/*doublerank*/
proc sort data=skebetarank;by skerank betarank;run;
data skebetarank1;
set skebetarank;
if skerank=. then delete;
run;
data skebetarank2;
set skebetarank1;
skebetarank=skerank*10+betarank;
run;
proc sql;
create table startret as
select
downret.*
from
downret
where
downret.sequence=&numcycle;
quit;/*保留所求月份的数据*/
proc sort data=skebetarank2;
by stkcd;
run;
proc sort data=startret;
by stkcd;
run;
proc sql;
create table startret as
select
startret.*,
skebetarank2.skerank,
skebetarank2.betarank,
```

```
skebetarank2.skebetarank
from
startret,skebetarank2
where
startret.stkcd=skebetarank2.stkcd;
quit;/*所求月各公司ret加上分组*/
proc sort data=startret;
by skebetarank;
run;
proc means data=startret noprint;
    var ret;
    output out=permonth25group
    mean(ret)=mean_ret
    n(ret)=n_ret;
    by skebetarank betarank;
run;/*求各组本月均值为（该组所有公司）*(所求月所有日期)的平均*/
data permonth25group;
set permonth25group;
portfolio=_n_;
run;/*为group排序*/
/***********计算high-low********8*/
/*挑选最高组和最低组*/
data permonthperskehigh;
set permonth25group;
if mod(portfolio,5)^=0 then delete;
run;
data permonthperskehigh;
set permonthperskehigh;
highret=mean_ret;
run;
data permonthperskelow;
set permonth25group;
if mod(portfolio+4,5)^=0 then delete;
run;
data permonthperskelow;
set permonthperskelow;
lowret=mean_ret;
run;
/**合并最高组和最低组并计算hml**/
proc sql;
create table permonthperskehml as
select
permonthperskehigh.*,
permonthperskelow.lowret
```

```
from
permonthperskehigh,permonthperskelow
where
permonthperskelow.portfolio+4=permonthperskehigh.portfolio;
quit;
data permonthperskehml;
set permonthperskehml;
hmlret=highret-lowret;
run;
/**对每个betagroup合并skegroup，相当于把portfolio分组**/
proc rank data=permonth25group out=prank group=5;
var betarank;
ranks prank;
run;/****prank=skewrank****/
/****求average****/
proc sort data=prank;
by betarank;
run;
proc means data=prank noprint;
    var mean_ret;
    output out=permonthperbetaAverageret
    mean(mean_ret)=average_ret
    n(mean_ret)=n_average_ret;
    by betarank;
run;
proc means data=permonthperskehml noprint;
    var hmlret;
    output out=averageret_hml
    mean(hmlret)=average_hmlret
    n(hmlret)=n_hmlret;
run;
/**每个月合并**/
  %if &i=1 %then
  %do;
    data group25;
        set Permonth25group;
    data perskehml;
        set Permonthperskehml;
    data perbetaAverageret;
        set permonthperbetaAverageret;
    data keyt;
        set Averageret_hml;
  %end;
  %else
```

```
  %do;
      proc append base=group25 data=permonth25group;
      proc append base=perskehml data=permonthperskehml;
      proc append base=perbetaAverageret
data=permonthperbetaAverageret;
      proc append base=keyt data=Averageret_hml;
run;

  %end;
  proc sort data=group25;
  by portfolio;
  run;
  proc means data=group25 noprint;
  var mean_ret;
  output out=group25Table
  mean(mean_ret)=ret;
  by portfolio;
  run;
  proc sort data=perskehml;
  by portfolio;
  run;
  proc means data=perskehml noprint;
  var hmlret;
  output out=perskehmlTable
  mean(hmlret)=ret
  t(hmlret)=t;
  by portfolio;
  run;
  proc sort data=perbetaAverageret;
  by betarank;
  run;
  proc means data=perbetaAverageret noprint;
  var average_ret;
  output out=averageretTable
  mean(average_ret)=ret
  t(average_ret)=t;
  by betarank;
  run;
  proc means data=keyt noprint;
  var average_hmlret;
  output out=keytTable
  mean(average_hmlret)=ret
  t(average_hmlret)=t;
  run;
```

```
  proc export data=group25table
outfile='C:\Users\m1371\Desktop\table' dbms=excel;
  proc export data=averagerettable
outfile='C:\Users\m1371\Desktop\table' dbms=excel;
  proc export data=perskehmlTable
outfile='C:\Users\m1371\Desktop\table' dbms=excel;
  proc export data=keyttable outfile='C:\Users\m1371\Desktop\table'
dbms=excel;run;
%end;
%mend rolling;
%rolling(201701,30);
```

Question 3:

```
*calculate the beta;
proc sort data=ret;
by trddt;
proc sort data=mkret;
by trddt;
data all;
merge ret(keep=trddt yearmonth ret stkcd) mkret(keep= trddt vwretd);
by trddt;
run;
data all;
set all;
where vwretd^=.;
run;

proc sort data=all;
by yearmonth;
run;
proc sort data=d_months;
by yearmonth;
run;
data all;
merge all d_months;
by yearmonth;
rename i=num_month;
run;

%macro run_cycle(first_month,num_cycles);
%do i=1 %to &num_cycles;
%let month_cycle=%eval(&first_month+&i-1);
```

```
data all1;
set all;
where &month_cycle-12<=num_month<=&month_cycle-1;
run;
proc sort data=all1;
by stkcd;
run;

proc reg data=all1 outest=beta_all noprint;
model ret=vwretd;
output out=reg_all
residual=r;
by stkcd;
run;

data beta_all;
set beta_all;
rename vwretd=beta;
num_month=&month_cycle;
run;
   %if &i=1 %then
    %do;
     data all_testing;
        set beta_all;
     run;
     data all_r;
        set reg_all;
     run;
    %end;
    %else
    %do;
      proc append base=all_testing data=beta_all;
      run;
      proc append base=all_r data=reg_all;
      run;
    %end;
%end;
%mend run_cycle;
%run_cycle(13,30);
*calculate residual volatility;
%macro run_cycle(first_month,num_cycles);
%do i=1 %to &num_cycles;
%let month_cycle=%eval(&first_month+&i-1);
```

```
data all_r1;
set all_r;
where &month_cycle-12<=num_month<=&month_cycle-1;
run;
proc sort data=all_r1;
by stkcd;
run;

proc means data=all_r1 noprint;
var r;
output out=all_r_stats
std(r)=resivol;
by stkcd;
run;
data all_r_stats;;
set all_r_stats;
num_month=&month_cycle;
run;
 %if &i=1 %then
    %do;
     data all_rt;
        set all_r_stats;
     run;
     %end;
    %else
    %do;
      proc append base=all_rt data=all_r_stats;
      run;
    %end;
%end;
%mend run_cycle;
%run_cycle(13,30);
*calculate momentum;
proc sort data=ret;
by yearmonth;
proc sort data=d_months;
by yearmonth;
data ret;
merge ret d_months;
by yearmonth;
rename i=num_month;
run;
%macro run_cycle(first_month,num_cycles);
```

```
%do i=1 %to &num_cycles;
%let month_cycle=%eval(&first_month+&i-1);
*calculate residual volatility;
data ret1;
set ret;
where &month_cycle-12<=num_month<=&month_cycle-1;
run;
proc sort data=ret1;
by stkcd;
run;
proc means data=ret1 noprint;
var log_ret;
output out=mom_stats
sum(log_ret)=momentum;
by stkcd;
run;
data mom_stats;;
set mom_stats;
num_month=&month_cycle;
run;

   %if &i=1 %then
    %do;
     data all_mom;
       set mom_stats;
     run;
    %end;
    %else
    %do;
      proc append base=all_mom data=mom_stats;
      run;
    %end;
%end;
%mend run_cycle;
%run_cycle(13,30);
*calculate size, non-linear size,PE, liquidity;
data part;
set full(keep=stkcd trddt year month Dsmvtll pe turnover);
yearmonth=year*100+month;
size=log(Dsmvtll);
nlsize=(log(Dsmvtll))**3;
run;
proc sort data=part;
by yearmonth;
```

```
proc sort data=d_months;
by yearmonth;
data part;
merge part d_months;
by yearmonth;
rename i=num_month;
run;
/*******************/
*with daily data;
proc sort data=part;
by stkcd num_month;
proc sort data=all_rt;
by stkcd num_month;
proc sort data=all_testing;
by stkcd num_month;
proc sort data=d_testing;
by stkcd num_month;
proc sort data=all_mom;
by stkcd num_month;
data dc;
merge part
all_rt(keep=stkcd num_month resivol)
all_testing(keep=stkcd num_month beta)
d_testing(keep=stkcd num_month downbeta)
all_mom(keep=stkcd num_month momentum);
by stkcd num_month;
run;
proc corr data=dc;
var downbeta beta momentum size resivol pe turnover nlsize;
run;
proc sort data=d1;
by stkcd yearmonth;
proc sort data=dc;
by stkcd yearmonth;
data dc1;
merge dc(keep=stkcd yearmonth downbeta beta) d1;
by stkcd yearmonth;
run;
data dc1;
set dc1;
where downbeta^=. and beta^=.;
run;
proc reg data=dc1;
model ret=downbeta beta;
```

```
quit;
*with monthly data;
proc sort data=part;
by stkcd num_month;
proc means data=part noprint;
var pe size nlsize turnover;
output out=part1
mean(pe size nlsize turnover)=pe size nlsize turnover;
by stkcd num_month;
run;
proc sort data=part1;
by stkcd num_month;
proc sort data=all_rt;
by stkcd num_month;
proc sort data=all_testing;
by stkcd num_month;
proc sort data=d_testing;
by stkcd num_month;
proc sort data=all_mom;
by stkcd num_month;
data dc;
merge part1
all_rt(keep=stkcd num_month resivol)
all_testing(keep=stkcd num_month beta)
d_testing(keep=stkcd num_month downbeta)
all_mom(keep=stkcd num_month momentum);
by stkcd num_month;
run;
proc corr data=dc;
var downbeta beta momentum size resivol pe turnover nlsize;
run;
proc sort data=msf;
by yearmonth;
proc sort data=d_months;
by yearmonth;
data msf;
merge msf d_months;
by yearmonth;
run;
data msf;
set msf;
rename i=num_month;
run;
proc sort data=dc;
```

```
by stkcd num_month;
proc sort data=msf;
by stkcd num_month;
data dc1;
merge dc(keep=stkcd num_month downbeta beta) msf;
by stkcd num_month;
run;
data dc1;
set dc1;
where downbeta^=. and beta^=.;
run;
proc reg data=dc1;
model monthlyret=downbeta beta;
ods output ParameterEstimates=parms_out;
quit;
data parms;
 set parms_out;
 tvalue2=put(tvalue,7.2);
 if probt<0.1 then p='*  ';
 if probt<0.05 then p='** ';
 if probt<0.01 then p='***';
 T=compress('['||tvalue2||']');
 PARAM=compress(put(estimate,7.3)||p);
run;
proc sort data=parms;
by variable;
proc transpose data=parms out=parms1;
var param T p;
by variable;
run;
data parms1;
set parms1;
by variable;
if first.variable=0 then variable=.;
run;
```