```
*************************************************************************;
* This file computes variables required for the trading strategy tests  *;
* NOTE - this file includes size, BTM, and momentum as characteristics  *;
*                 Last updated: 31 August 2018                          *;
*************************************************************************;

libname home 'YOUR_DIR';
libname char "YOUR_DIR";
libname master "YOUR_DIR";


*************************************************************************;
** CAR: [y,xx] return minus size-btm-momentum matched portfolio return
** Form size breakpoints based on American exchanges, industry-
** adjusted BTM, and twelve-month momentum
*************************************************************************;

* Let's do quintiles for portfolio formation *;
%let size = 5;
%let btm_break = 5;
%let mom = 5;

* This file is divided into the following parts:
* I. The Size, BTM, and Momentum portfolios for a given date range.
* II. Determine the portfolio for each firm-quarter and find the long-term (i.e.,
monthly) returns for the portfolio
* III. The short-term (i.e., daily) returns for the portfolio
* IV. The short-run and long-run returns for the sample firms
* V. Calculate the BHAR over the specified ranges;

*****************************************************************************
*
I. Allocate all firms into Size/BTM/Momentum portfolios for a given date range
*****************************************************************************
*;

rsubmit;
proc sql;
      create table size as select
            date, permno, hexcd, shrout, prc
      from crsp.msf
      /* Specify exchanges */
      where hexcd in (1,2,3)
      /* Our sample begins in 2012, so pull in everything on or after 2010 to be
safe */
      and year(date) >= 2010
      order by permno, date;
quit;

* Ensure no duplicates *;
proc sort data=size out=size nodupkey; by permno date; run;

proc download data=size out=size; run;
endrsubmit;

* Calculate market value and create quintile breakpoint for each month *;
data size; set size;
      mkt_val = abs(prc)*shrout;
run;

proc sort data=size; by date mkt_val; quit;
```

```
62
63  * Create size quintiles *;
64  proc rank data=size out=size2 groups=&size.;
65        by date;
66        var mkt_val;
67        ranks size_quintile;
68  quit;
69
70  * Create size breaks which will be used later for missing values *;
71  proc means data=size2 noprint min nway missing;
72        class date size_quintile;
73        var mkt_val;
74        output out=size_breaks (drop=_type_ _freq_) min=min_value;
75  run;
76
77  data size_breaks; set size_breaks;
78        if size_quintile=. then delete;
79  run;
80
81  * Create a range for each quintile *;
82  proc sort data=size_breaks; by date size_quintile; quit;
83
84  proc expand data=size_breaks out=size_breaks;
85        by date;
86        id size_quintile;
87        convert min_value = max_value /transformout= (lead 1) method=none;
88  quit;
89
90  * Replace the "." for max_value with a large number to make matching easier *;
91  * Similarly, set min mkt_value for smallest quintile to be 0 *;
92  data size_breaks; set size_breaks;
93        if max_value=. then
94              max_value=999999999999;
95        if size_quintile=0 then
96              min_value =0;
97        /* Prevents firms at cutpoint from being double counted */
98        max_value = max_value-0.0001;
99  run;
100
101 **** Move onto BTM portfolio ****;
102 * Calculate BTM for all firms *;
103 * Download CRSP/Compustat data *;
104 rsubmit;
105 proc sql;
106       create      table compq as
107       select      a.gvkey, a.datadate, a.rdq, (a.fyearq * 10) + a.fqtr as fyearqtr,
108 a.atq, a.atq - a.ltq as book, a.ceqq, a.prccq
109       from  comp.fundq as a
110       where ((indfmt  = 'INDL') and (consol = 'C') and (popsrc = 'D') and (datafmt
111 = 'STD'))
112             /* This date range should work fine for BTM */
113             and '30jan2011'd le datadate le '31dec2015'd
114       order by gvkey, fyearqtr, atq descending;
115 quit;
116
117 * Eliminate duplicate quarters - very small number so just ignore them *;
118 proc sort data=compq nodupkey; by gvkey fyearqtr; run;
119
120 * Bring in permnos from CRSP *;
121 proc sql;
122       create table compq2 as select
```

```
            a.*, b.lpermno as permno
      from compq as a left join crsp.ccmxpf_linktable as b
      on a.gvkey eq b.gvkey
      and not missing(b.lpermno)
      and b.linktype in ('LU', 'LC', 'LD', 'LF', 'LN', 'LO', 'LS', 'LX')
      and b.linkprim in ('C', 'P')
      and (b.linkdt <= a.rdq or b.linkdt = .B)
      and (a.rdq <= b.linkenddt or b.linkenddt = .E);
quit;

* Remove duplicate entries *;
proc sort data=compq2 out=compq2 nodupkey; by gvkey fyearqtr; run;
* Kill obs with missing permnos *;
data compq2; set compq2; where not missing(permno); run;

proc download data=compq2 out=compq2; run;
endrsubmit;

* Save to characteristic-adjusted folder *;
data char.bm_data; set compq2; run;

* Take the most recent datadate for a given RDQ *;
proc sort data=char.bm_data out=btm; by permno rdq descending datadate; quit;
* Remove duplicates *;
proc sort data=btm out=btm nodupkey; by permno rdq; quit;
* Create RDQ 'windows' for which the quarterly BVE is valid *;
proc sort data=btm out=btm nodupkey; by permno datadate; quit;

data btm; set btm;
      /* Make sure we have valid RDQs */
      where not missing(rdq);
      /*  Reset RDQ to be 3 months after fiscal quarter-end if difference is
greater
            than 6 months. This occurs for a small number of observations (around
            95 out of near 100k) */
      if intck('Month',datadate,rdq)>6 then rdq=intnx('Month',datadate,4)-1;
run;

proc expand data=btm method=none out=btm;
      by permno;
      id datadate;
      * Create lead RDQ variable *;
      convert rdq = rdq_lead /transformout= (lead 1) method=none;
quit;

data btm; set btm;
      rdq_lead = rdq_lead-1; /* So there arent two reports for a given date */
      if ceqq=. then ceqq=book;
      if ceqq=. then delete;
      if rdq_lead =. then rdq_lead = intnx('Month',rdq,3);
      if rdq_lead < rdq then delete;
run;

* Bring in calendar month-end dates from CRSP for merging *;
* Since are we forming portfolios as of each month-end *;
rsubmit;
proc sql;
      create table dates as select
            caldt
      from crsp.msp500 (where=(year(caldt)>=2010));
quit;
```

```
184
185  proc download data=dates out=dates;
186  quit;
187  endrsubmit;
188
189  * Join datasets *;
190  proc sql;
191        create table returns as select distinct
192              a.*, b.gvkey, b.permno, b.datadate,
193              b.rdq, b.rdq_lead, b.ceqq
194        from dates as a left join btm as b
195        on a.caldt between b.rdq_lead and b.rdq;
196  quit;
197
198  * Check to make sure there are no duplicates *;
199  proc sort data=returns nodupkey; by permno caldt; quit;
200
201  * Pull in monthly returns and price data as of calendar month-end *;
202  rsubmit;
203  proc upload data=returns out=returns; run;
204  proc sql;
205        create table returns as select
206              a.*, b.ret, abs(b.prc) as prc, b.shrout
207        from returns as a, crsp.msf as b
208        where a.permno = b.permno
209        and month(a.caldt) = month(b.date)
210        and year(a.caldt) = year(b.date);
211  quit;
212
213  * Also bring in SIC codes for FF48 industry classifications *;
214  proc sql;
215        create table returns as select distinct
216              a.*, b.sic as siccd
217        from returns as a left join comp.company as b
218        on a.gvkey = b.gvkey;
219  quit;
220
221  proc download data=returns out=returns; run;
222  endrsubmit;
223
224  * Calculate market value and BTM *;
225  data returns2; set returns;
226        mkt_val = prc*shrout;
227        label mkt_val = 'Market Value';
228        btm = ceqq*1000/(prc*shrout);
229        label btm = 'Book-to-market';
230  run;
231
232  * Bring in existing size quintiles *;
233  proc sql;
234        create table returns2 as select distinct
235              a.*, b.size_quintile, b.min_value, b.max_value
236        from returns2 as a left join size_breaks as b
237        on a.caldt = b.date
238        and a.mkt_val between b.min_value and b.max_value;
239  quit;
240
241  * Find the mean BTM average for each Fama French 48 industry for each month *;
242  * Fix a couple of SIC codes *;
243  data returns2; set returns2;
244        if SICCD in (3990,3999) then SICCD = 3991;
```

```
run;

* Bring in FF48 codes from macro file *;
%ff48(data=returns2, out=returns3, sic=SICCD, newvarname=FF_IND);

proc sort data=returns3; by FF_IND caldt; run;

* Deal with BTM extremes *;
%Winsorize_Truncate(dsetin=returns3, dsetout=returns3, vars=btm, type = W, pctl = 2
98);

* Determine BTM industry average for each calendar month-end *;
proc means data = returns3 noprint;
      by FF_IND caldt;
      var btm;
      output out = BTM_IND (drop=_Type_ _freq_)  mean=bmind;
run;

* Calculate long-term Industry BTM Average *;
data BTM_IND; set BTM_IND;
      by FF_IND caldt;
      retain avg n;
      if first.FF_IND then do;
            avg=bmind;
            n=1;
            bmavg=avg;
      end;
      else do;
      bmavg=((avg*n)+bmind)/(n+1);
      n+1;
      avg=bmavg;
      end;
      format bmavg comma8.2;
      drop avg n bmind;
run;

* Adjust firm-specific BTM for industry averages *;
proc sql;
      create table returns3 as select
            a.*, b.bmavg as BTM_AVG "LT Industry BTM Ratio" format comma8.2,
            (a.btm-b.bmavg) as BTM_ADJ "Adjusted BTM Ratio" format comma8.2
      from returns3 as a, BTM_IND as b
      where a.caldt=b.caldt and a.FF_IND=b.FF_IND;
quit;

* Determine BTM quintiles within each size quintile *;
proc sort data=returns3; by caldt size_quintile BTM_ADJ; quit;

* Create BTM quintiles *;
* Output to characteristic-adjusted folder *;
proc rank data=returns3 out=char.port_form1 groups=&btm_break.;
      by caldt size_quintile;
      var BTM_ADJ;
      ranks btm_quintile;
quit;

* Run this code to address observations that potentially fall off *;
* We create this now and pull it back in later *;
proc means data=char.port_form1 noprint min nway missing;
      class caldt size_quintile btm_quintile;
      var BTM_ADJ;
```

```
      output out=btm_breaks (drop=_type_ _freq_) min=min_value;
run;

data btm_breaks; set btm_breaks;
      if btm_quintile=. then delete;
run;

* Create a range for each quintile *;
proc sort data=btm_breaks; by caldt size_quintile btm_quintile; quit;

proc expand data=btm_breaks out=btm_breaks;
      by caldt size_quintile;
      id btm_quintile;
      convert min_value = max_value /transformout= (lead 1) method=none;
quit;

* Replace the "." for max_value with a large number to make matching easier *;
* Similarly, set min BTM_ADJ for smallest quintile to be -99999999999 *;
data btm_breaks; set btm_breaks;
      if max_value=. then
            max_value=999999999999;
      if btm_quintile=0 then
            min_value =-9999999999;
      /* Prevents firms at cutpoint from being double counted */
      max_value = max_value-0.0001;
run;

* Clean up variables, re-order, and only keep the observations which are non-
missing *;
* We do this now and then again after we do momentum *;
proc sql;
      create table char.port_form2 as select distinct
            caldt, permno, size_quintile, btm_quintile, FF_IND, *
      from char.port_form1 (where=(not missing(size_quintile+btm_quintile)));
quit;

* Check for duplicates *;
%dupl(dset=char.port_form2,keys=permno caldt);

************************;
* Move to momentum now *;
************************;

* Calculate momentum based on the average monthly returns from the
trailing 12 months of return data. Require at least 6 months of data in order to
have a valid momentum as in Daniel et al. (1997) *;
proc sort data=char.port_form2; by permno caldt; quit;

* We already pulled in returns for each calendar month-end, so proc expand this *;
proc expand data=char.port_form2 (keep=permno caldt ret) out=momentum method=none;
      by permno;
      id caldt;
      convert ret = cret_12m / transformin=(+1) transformout=(MOVPROD 12 -1
trimleft 6);
quit;

data momentum2; set momentum;
      by permno caldt;
      * Rename momentum variable *;
      MOM = cret_12m;

```

```
        label MOM="12-Month Momentum Factor";
        label caldt="Formation Date";
        format MOM RET 11.5;
        drop cret_12m;
run;

* Bring in momentum to big dataset by calendar month-end *;
proc sql;
        create table returns4 as select distinct
                a.*, b.MOM
        from char.port_form2 as a left join momentum2 as b
        on a.permno = b.permno
        and a.caldt = b.caldt;
quit;

* Sort by calendar month-end and other quintiles *;
proc sort data=returns4; by caldt size_quintile btm_quintile; run;

* Create momentum quintile ranks *;
proc rank data=returns4 out=char.port_form3 groups=&mom.;
        by caldt size_quintile btm_quintile;
        var MOM;
        ranks mom_quintile;
quit;

* Create momentum breaks similar to other code *;
* Purpose of this is to address some observations that might fall through the
cracks *;
proc means data=char.port_form3 noprint min nway missing;
        class caldt size_quintile btm_quintile mom_quintile;
        var MOM;
        output out=mom_breaks (drop=_type_ _freq_) min=min_value;
run;

data mom_breaks;
        set mom_breaks;
        if mom_quintile=. then delete;
run;

* Create a range for each quintile *;
proc sort data=mom_breaks;
        by caldt size_quintile btm_quintile mom_quintile;
quit;

proc expand data=mom_breaks out=mom_breaks;
        by caldt size_quintile btm_quintile;
        id mom_quintile;
        convert min_value = max_value /transformout= (lead 1) method=none;
quit;

* Replace the "." for max_value with a large number to make matching easier *;
* Similarly, set min for smallest quintile to be -99999999999 *;
data mom_breaks; set mom_breaks;
        if max_value=. then
                max_value=999999999999;
        if mom_quintile=0 then
                min_value =-9999999999;
        * Prevents firms at cutpoint from being double counted *;
        max_value = max_value-0.00001;
run;
* 7200ish observations here *;
```

```
* 5x5x5x(4x12) should yield 6000 obs, but we have more since we are temporarily
expanding
  outside of 2012 and 2015 *;

* Duplicates code again *;
%dupl(dset=char.port_form3,keys=permno caldt);

* Clean up variables, re-order, and only keep the observations which are non-
missing *;
proc sql;
      create table char.port_form3 as select distinct
            caldt, permno, size_quintile, btm_quintile, mom_quintile, FF_IND, *
      from char.port_form3 (where=(not
missing(size_quintile+btm_quintile+mom_quintile)));
quit;
* Momentum is missing for a number of observations, which is likely attributable
to the 6-month requirement *;

proc sort data=char.port_form3 out=temp; by size_quintile btm_quintile
mom_quintile; run;


*************************************************************************
*;
*II. Find what size/BTM/momentum portfolio each sample firm falls into and
calculate
       benchmark returns for the portfolio;
*************************************************************************
*;

* Bring in the master dataset at this point *;
* This is our main in-sample dataset, but only bring in a few variables *;
data ret_data; set master.Step2_final(keep=gvkey fyearqtr rdq_to_use btm mkt_cap
permno); run;

* Bring in SIC codes again *;
rsubmit;
proc upload data=ret_data; run;
proc sql;
      create table ret_data2 as select
            a.*, b.sic as siccd
      from ret_data as a left join comp.company as b
      on a.gvkey = b.gvkey;
quit;

proc download data=ret_data2 out=ret_data2; run;
endrsubmit;

* Fix one block of SICs and permno *;
data ret_data2; set ret_data2;
      if SICCD in (3990,3999) then SICCD = 3991;
      * This is for GVKEY 151928 *;
      if permno = 89613 then permno = 16852;
run;

* Create industry codes again *;
%ff48(data=ret_data2, out=ret_data2, sic=SICCD, newvarname=FF_IND);

* Find portfolios that will be used as a benchmark based on date, size, BTM, and
momentum portfolio *;
proc sql;
```

```
      create table target_benchmark as select distinct
            a.gvkey, a.fyearqtr, a.rdq_to_use, a.btm as btm_current, a.mkt_cap*1000
as mkt_cap_current, a.permno, a.FF_IND,
            b.caldt, b.size_quintile, b.btm_quintile, b.mom_quintile
      from ret_data2 as a left join char.port_form3 as b
      on a.permno = b.permno
      /* Want to merge such that the portfolios are formed for the month preceding
a given EA */
      and intck('Month',a.rdq_to_use,b.caldt)=-1;
quit;

* Next blocks of code are to minimize loss of observations *;
* Merge in caldt *;
proc sql;
      create table target_benchmark as select distinct
            a.*, b.trading_date format YYMMDDN8.
      from target_benchmark as a left join char.trading_dates_final as b
      on a.rdq_to_use = b.orig_date;
quit;

proc sql;
      create table target_benchmark as select distinct
            a.*, b.caldt as caldt2
      from target_benchmark as a left join dates as b
      on intck('month', b.caldt, a.trading_date) = 1;
quit;

* Merge in mkt_cap at caldt2 and BTM from most recent rdq before caldt2 *;
rsubmit;
proc upload data=target_benchmark; quit;
proc sql;
      create table target_benchmark2 as select
            a.*, abs(b.prc)*shrout as mkt_cap
      from target_benchmark as a, crsp.msf as b
      where a.permno = b.permno
      and month(a.caldt2) = month(b.date)
      and year(a.caldt2) = year(b.date);
quit;

proc sql;
      create      table compq as
      select      a.gvkey, a.datadate, a.rdq, (a.fyearq * 10) + a.fqtr as fyearqtr,
a.atq, a.atq - a.ltq as book, a.ceqq, a.prccq*a.cshoq*1000 as mkt_cap2
      from  comp.fundq as a
      where ((indfmt  = 'INDL') and (consol = 'C') and (popsrc = 'D') and (datafmt
= 'STD'))
            and '30jan2011'd le datadate le '31dec2015'd
      order by gvkey, fyearqtr, atq descending;
quit;

* Eliminate duplicate quarters - very small number so just ignore them *;
proc sort data=compq nodupkey; by gvkey fyearqtr; run;

proc sql;
      create table target_benchmark2 as select
            a.*, b.ceqq, b.book, b.mkt_cap2, b.rdq as rdq_prior
      from target_benchmark2 as a left join compq as b
      on a.gvkey = b.gvkey
      /* Six-month look back period for BTM */
      and intck('month',a.caldt2,b.rdq) between -6 and 0
      order by gvkey, fyearqtr, rdq descending;
```

```
quit;

proc sort data=target_benchmark2 nodupkey; by gvkey fyearqtr; run;

* Bring back down locally *;
proc download data=target_benchmark2; quit;
endrsubmit;

* Re-create BTM variable for missing observations *;
data target_benchmark3 (drop=btm_current ceqq book mkt_cap2 mkt_cap_current
rdq_prior);
      set target_benchmark2;
      if mkt_cap=. then mkt_cap=mkt_cap2;
      if mkt_cap=. then mkt_cap=mkt_cap_current;
      if ceqq=. then ceqq=book;
      btm=ceqq/mkt_cap*1000;
      if btm=. then btm=btm_current;
run;

* Merge in industry average BTM *;
proc sql;
      create table target_benchmark3 as select distinct
            a.*, b.bmavg as BTM_AVG "LT Industry BTM Ratio",
            a.btm-b.bmavg as BTM_ADJ "Adjusted BTM Ratio" format comma8.2
      from target_benchmark3 as a left join BTM_IND as b
      on a.caldt2 = b.caldt
      and a.FF_IND = b.FF_IND;
quit;

* Now let's also bring in momentum again to account for these issues *;
rsubmit;
proc upload data=target_benchmark3; quit;
proc sql;
      create table target_benchmark4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as MOM, count(b.ret) as n
      from target_benchmark3 as a left join crsp.msf as b
      on a.permno eq b.permno

      and a.caldt2 ge b.date
      and (intnx('day',a.caldt2,0) - intnx('day',b.date,0)) <= 340
      group by a.permno, a.fyearqtr;
quit;

proc download data=target_benchmark4; quit;
endrsubmit;

* Set MOM to missing if missing more than 6 months of data *;
data target_benchmark5; set target_benchmark4;
      if n < 7 then MOM = .;
run;

proc freq data=target_benchmark5; tables n; run;

data target_benchmark5; set target_benchmark3; run;

* Now merge in breakpoints *;
* First do size_quintile *;
proc sql;
      create table target_benchmark6 as select
            a.*, b.size_quintile as size_quintile2, b.min_value, b.max_value
      from target_benchmark5 as a left join size_breaks as b
```

```
611         on a.caldt2 = b.date
612         and a.mkt_cap between b.min_value and b.max_value;
613   quit;
614
615   * Merge in btm_quintile based on caldt and size_quintile *;
616   proc sql;
617         create table target_benchmark6 as select
618               a.*, b.btm_quintile as btm_quintile2, b.min_value as min_value_btm,
619   b.max_value as max_value_btm
620         from target_benchmark6 as a left join btm_breaks as b
621         on a.caldt2 = b.caldt
622         and a.size_quintile2 = b.size_quintile
623         and a.BTM_ADJ between b.min_value and b.max_value;
624   quit;
625
626   * Finally, merge in mom_quintile based on caldt, size_quintile, and btm_quintile *;
627   proc sql;
628         create table target_benchmark6 as select
629               a.*, b.mom_quintile as mom_quintile2, b.min_value as min_value_mom,
630   b.max_value as max_value_mom
631         from target_benchmark6 as a left join mom_breaks as b
632         on a.caldt2 = b.caldt
633         and a.size_quintile2 = b.size_quintile
634         and a.btm_quintile2 = b.btm_quintile
635         and a.MOM between b.min_value and b.max_value;
636   quit;
637
638   * Keep relevant variables and assign quintiles in the event they were originally
639   missing *;
640   data target_benchmark6 (keep=rdq_to_use permno trading_date caldt size_quintile
641   btm_quintile mom_quintile);
642         set target_benchmark6;
643         caldt=caldt2;
644         if size_quintile=. then size_quintile=size_quintile2;
645         if btm_quintile=. then btm_quintile=btm_quintile2;
646         if mom_quintile=. then mom_quintile=mom_quintile2;
647   run;
648
649   * Save down target_benchmark6 *;
650   data char.target_benchmark1; set target_benchmark6; run;
651
652   * Only keep portfolio data that will be used as a target benchmark *;
653   proc sql;
654         create table benchmark_portfolio as select distinct
655               a.*,
656               case
657                     when b.caldt=.
658                     then 0
659               else 1
660               end
661         as valid_benchmark
662         from char.port_form3 as a left join char.target_benchmark1 as b
663         on (intck('month',a.caldt,b.caldt) between 0 and 2)
664         and a.size_quintile = b.size_quintile
665         and a.btm_quintile = b.btm_quintile
666         and a.mom_quintile = b.mom_quintile;
667   quit;
668
669   data benchmark_portfolio; set benchmark_portfolio;
670         if valid_benchmark;
671   run;
```

```
672
673
674   ********************************************************************************
675   *;
676   *III. Calculate benchmark daily returns for the portfolio;
677   ********************************************************************************
678   *;
679
680   proc sort data=benchmark_portfolio nodupkey; by caldt permno; run;
681
682   * Keep only necessary variables *;
683   data char.benchmark_portfolio;
684         set benchmark_portfolio(drop=RET prc SHROUT ceqq min_value max_value);
685   run;
686   data benchmark_portfolio_temp; set char.benchmark_portfolio; run;
687
688   * Need to do all of these next steps in WRDS because datasets get huge *;
689   rsubmit;
690   proc upload data=benchmark_portfolio_temp out=benchmark_portfolio_temp; quit;
691   proc upload data=char.trading_dates_final out=trading_dates_final; quit;
692   * Pull in simple base returns first *;
693   proc sql;
694         create table benchmark_portfolio_temp2 as select
695               a.*, b.date, b.ret
696         from benchmark_portfolio_temp as a left join crsp.dsf
697   (where=(year(date)>=2010)) as b
698         on a.permno=b.permno
699         and intck('month',a.caldt,b.date) between 1 and 5;
700   quit;
701
702   * Short-term Delisting Returns *;
703   proc sql;
704         create table benchmark_portfolio_temp2 as select distinct
705               a.*, b.dlret
706         from benchmark_portfolio_temp2 a left join crsp.DSEDELIST (where=(dlret^=.
707   and DLSTDT^=.)) b
708         on a.permno = b.permno
709         and intck('month', a.caldt, b.DLSTDT) between 1 and 5;
710   quit;
711
712   * Reset returns for delisting *;
713   data benchmark_portfolio_temp2;
714         set benchmark_portfolio_temp2;
715         ret2 = ret;
716         if ret2=. then
717               ret2=dlret;
718         drop dlret ret;
719         rename ret2 = ret;
720   run;
721
722   proc sort data=benchmark_portfolio_temp2 nodupkey; by permno date descending caldt;
723   run;
724   proc sort data=benchmark_portfolio_temp2 nodupkey; by permno date; run;
725
726   * Trading date ranges *;
727   proc sql;
728         create table benchmark_portfolio_temp2 as select distinct
729               a.*, b.trading_date_plus1, b.trading_date_plus2, b.trading_date_plus3,
730   b.trading_date_plus4,
731               b.trading_date_plus5, b.trading_date_plus10, b.trading_date_plus20,
732   b.trading_date_plus40,
```

```
733             b.trading_date_plus60
734       from benchmark_portfolio_temp2 as a left join trading_dates_final as b
735       on a.date = b.orig_date;
736 quit;
737
738 * Need to partition this dataset for computational purposes *;
739 * Before calculating buy-and-hold returns *;
740 data temp1; set benchmark_portfolio_temp2;
741       where permno <= 32299;
742 run;
743 data temp2; set benchmark_portfolio_temp2;
744       where (permno>32299 and permno<=81138);
745 run;
746 data temp3; set benchmark_portfolio_temp2;
747       where (permno>81138 and permno<=89500);
748 run;
749 data temp4; set benchmark_portfolio_temp2;
750       where permno>89500;
751 run;
752
753 proc download data=temp1; run;
754 proc download data=temp2; run;
755 proc download data=temp3; run;
756 proc download data=temp4; run;
757 endrsubmit;
758
759 * Need to get mean return per portfolio-date *;
760 * Do the first sample partition *;
761 rsubmit;
762 proc upload data=temp1; run;
763 * For now, start with [2,60] window *;
764 proc sql;
765       create table temp1 as select distinct
766             a.*, exp(sum(log(1+b.ret)))-1 as ret260
767       from temp1 as a left join temp1 as b
768       on a.permno = b.permno
769       and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
770       group by a.permno, a.date;
771 quit;
772 * Also do [2,40] *;
773 proc sql;
774       create table temp1 as select distinct
775             a.*, exp(sum(log(1+b.ret)))-1 as ret240
776       from temp1 as a left join temp1 as b
777       on a.permno = b.permno
778       and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
779       group by a.permno, a.date;
780 quit;
781 * And also [2,20] *;
782 proc sql;
783       create table temp1 as select distinct
784             a.*, exp(sum(log(1+b.ret)))-1 as ret220
785       from temp1 as a left join temp1 as b
786       on a.permno = b.permno
787       and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
788       group by a.permno, a.date;
789 quit;
790 * Also do [2,10] *;
791 proc sql;
792       create table temp1 as select distinct
793             a.*, exp(sum(log(1+b.ret)))-1 as ret210
```

```
794         from temp1 as a left join temp1 as b
795         on a.permno = b.permno
796         and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
797         group by a.permno, a.date;
798   quit;
799   * Also do [2,5] *;
800   proc sql;
801         create table temp1 as select distinct
802               a.*, exp(sum(log(1+b.ret)))-1 as ret205
803         from temp1 as a left join temp1 as b
804         on a.permno = b.permno
805         and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
806         group by a.permno, a.date;
807   quit;
808   * Also do [2,4] *;
809   proc sql;
810         create table temp1 as select distinct
811               a.*, exp(sum(log(1+b.ret)))-1 as ret204
812         from temp1 as a left join temp1 as b
813         on a.permno = b.permno
814         and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
815         group by a.permno, a.date;
816   quit;
817   * Also do [2,3] *;
818   proc sql;
819         create table temp1 as select distinct
820               a.*, exp(sum(log(1+b.ret)))-1 as ret203
821         from temp1 as a left join temp1 as b
822         on a.permno = b.permno
823         and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
824         group by a.permno, a.date;
825   quit;
826
827   proc download data=temp1 out=temp1; run;
828   endrsubmit;
829
830   * Save down first group *;
831   data char.port_ret_group1; set temp1; run;
832
833   * Do this for the other temporary groups *;
834   rsubmit;
835   proc upload data=temp2; run;
836   proc sql;
837         create table temp2 as select distinct
838               a.*, exp(sum(log(1+b.ret)))-1 as ret260
839         from temp2 as a left join temp2 as b
840         on a.permno = b.permno
841         and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
842         group by a.permno, a.date;
843   quit;
844   proc sql;
845         create table temp2 as select distinct
846               a.*, exp(sum(log(1+b.ret)))-1 as ret240
847         from temp2 as a left join temp2 as b
848         on a.permno = b.permno
849         and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
850         group by a.permno, a.date;
851   quit;
852   proc sql;
853         create table temp2 as select distinct
854               a.*, exp(sum(log(1+b.ret)))-1 as ret220
```

```
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
      group by a.permno, a.date;
quit;
proc sql;
      create table temp2 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret210
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
      group by a.permno, a.date;
quit;
proc sql;
      create table temp2 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret205
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
      group by a.permno, a.date;
quit;
proc sql;
      create table temp2 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret204
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
      group by a.permno, a.date;
quit;
proc sql;
      create table temp2 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret203
      from temp2 as a left join temp2 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
      group by a.permno, a.date;
quit;

proc download data=temp2 out=temp2; run;
endrsubmit;

* Save down second partition *;
data char.port_ret_group2; set temp2; run;

* Third group *;
rsubmit;
proc upload data=temp3; run;
proc sql;
      create table temp3 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret260
      from temp3 as a left join temp3 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
      group by a.permno, a.date;
quit;
proc sql;
      create table temp3 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret240
      from temp3 as a left join temp3 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
```

```
916         group by a.permno, a.date;
917   quit;
918   proc sql;
919         create table temp3 as select distinct
920               a.*, exp(sum(log(1+b.ret)))-1 as ret220
921         from temp3 as a left join temp3 as b
922         on a.permno = b.permno
923         and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
924         group by a.permno, a.date;
925   quit;
926   proc sql;
927         create table temp3 as select distinct
928               a.*, exp(sum(log(1+b.ret)))-1 as ret210
929         from temp3 as a left join temp3 as b
930         on a.permno = b.permno
931         and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
932         group by a.permno, a.date;
933   quit;
934   proc sql;
935         create table temp3 as select distinct
936               a.*, exp(sum(log(1+b.ret)))-1 as ret205
937         from temp3 as a left join temp3 as b
938         on a.permno = b.permno
939         and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
940         group by a.permno, a.date;
941   quit;
942   proc sql;
943         create table temp3 as select distinct
944               a.*, exp(sum(log(1+b.ret)))-1 as ret204
945         from temp3 as a left join temp3 as b
946         on a.permno = b.permno
947         and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
948         group by a.permno, a.date;
949   quit;
950   proc sql;
951         create table temp3 as select distinct
952               a.*, exp(sum(log(1+b.ret)))-1 as ret203
953         from temp3 as a left join temp3 as b
954         on a.permno = b.permno
955         and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
956         group by a.permno, a.date;
957   quit;
958
959   proc download data=temp3 out=temp3; run;
960   endrsubmit;
961
962   * Save down third partition *;
963   data char.port_ret_group3; set temp3; run;
964
965   * Final group *;
966   rsubmit;
967   proc upload data=temp4; run;
968   proc sql;
969         create table temp4 as select distinct
970               a.*, exp(sum(log(1+b.ret)))-1 as ret260
971         from temp4 as a left join temp4 as b
972         on a.permno = b.permno
973         and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
974         group by a.permno, a.date;
975   quit;
976   proc sql;
```

```
      create table temp4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret240
      from temp4 as a left join temp4 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
      group by a.permno, a.date;
quit;
proc sql;
      create table temp4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret220
      from temp4 as a left join temp4 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
      group by a.permno, a.date;
quit;
proc sql;
      create table temp4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret210
      from temp4 as a left join temp4 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
      group by a.permno, a.date;
quit;
proc sql;
      create table temp4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret205
      from temp4 as a left join temp4 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
      group by a.permno, a.date;
quit;
proc sql;
      create table temp4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret204
      from temp4 as a left join temp4 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
      group by a.permno, a.date;
quit;
proc sql;
      create table temp4 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret203
      from temp4 as a left join temp4 as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
      group by a.permno, a.date;
quit;

proc download data=temp4 out=temp4; run;
endrsubmit;

* Save down fourth and final partition *;
data char.port_ret_group4; set temp4; run;

data temp1; set char.port_ret_group1; run;
data temp2; set char.port_ret_group2; run;
data temp3; set char.port_ret_group3; run;
data temp4; set char.port_ret_group4; run;

* Combine the partitioned datasets *;
data benchmark_portfolio_temp3;
```

```
      set temp1 temp2 temp3 temp4;
run;

data benchmark_portfolio_temp3;
      set benchmark_portfolio_temp3;
      if caldt=. or size_quintile=. or btm_quintile=. or mom_quintile=. or date=.
then delete;
run;

* Calculate equal-weighted returns by portfolio *;
proc means data=benchmark_portfolio_temp3 mean noprint;
      class caldt date size_quintile btm_quintile mom_quintile;
      var ret ret260 ret240 ret220 ret210 ret205 ret204 ret203;
      output out=portfolio_returns (drop=_type_ _freq_) mean=ew_ret ew_ret_260
ew_ret_240 ew_ret_220 ew_ret_210 ew_ret_205

      ew_ret_204 ew_ret_203;
run;

data portfolio_returns; set portfolio_returns;
      if caldt=. or size_quintile=. or btm_quintile=. or mom_quintile =. or date=.
then delete;
run;

* Save down *;
data char.port_ret_combined; set portfolio_returns; run;
data char.benchmark_portfolio; set benchmark_portfolio_temp3; run;

*************************************************************************************
*;
* IV. Calculate firm returns
*************************************************************************************
*;

data benchmark_portfolio_temp3; set char.benchmark_portfolio; run;
data target_benchmark6; set char.target_benchmark1; run;

* Let's get the firm raw returns *;
proc sort data=target_benchmark6; by permno rdq_to_use; run;
proc sql;
      create table target_benchmark7 as select
            a.*, b.ret260, b.ret240, b.ret220, b.ret210, b.ret205, b.ret204,
b.ret203,
            b.trading_date_plus1, b.trading_date_plus2, b.trading_date_plus3,
b.trading_date_plus4,
            b.trading_date_plus5, b.trading_date_plus10, b.trading_date_plus20,
b.trading_date_plus40,
            b.trading_date_plus60
      from target_benchmark6 as a left join benchmark_portfolio_temp3 as b
      on a.permno = b.permno
      and a.trading_date = b.date;
quit;

proc sql;
      create table target_benchmark7 as select
            a.*, b.trading_date_plus1, b.trading_date_plus2, b.trading_date_plus3,
b.trading_date_plus4,
            b.trading_date_plus5, b.trading_date_plus10, b.trading_date_plus20,
b.trading_date_plus40,
            b.trading_date_plus60
      from target_benchmark7 as a left join char.trading_dates_final as b
```

```
      on a.rdq_to_use = b.orig_date;
quit;

* Daily returns are missing sometimes, so double check the DSF and pull the returns
in as applicable *;
rsubmit;
proc upload data=target_benchmark7; run;
proc sql;
      create table target_benchmark7 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret260_alt
      from target_benchmark7 as a left join crsp.dsf as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus60
      group by a.permno, a.trading_date;
quit;
proc sql;
      create table target_benchmark7 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret240_alt
      from target_benchmark7 as a left join crsp.dsf as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus40
      group by a.permno, a.trading_date;
quit;
proc sql;
      create table target_benchmark7 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret220_alt
      from target_benchmark7 as a left join crsp.dsf as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus20
      group by a.permno, a.trading_date;
quit;
proc sql;
      create table target_benchmark7 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret210_alt
      from target_benchmark7 as a left join crsp.dsf as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus10
      group by a.permno, a.trading_date;
quit;
proc sql;
      create table target_benchmark7 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret205_alt
      from target_benchmark7 as a left join crsp.dsf as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus5
      group by a.permno, a.trading_date;
quit;
proc sql;
      create table target_benchmark7 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret204_alt
      from target_benchmark7 as a left join crsp.dsf as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus4
      group by a.permno, a.trading_date;
quit;
proc sql;
      create table target_benchmark7 as select distinct
            a.*, exp(sum(log(1+b.ret)))-1 as ret203_alt
      from target_benchmark7 as a left join crsp.dsf as b
      on a.permno = b.permno
      and a.trading_date_plus2 <= b.date <= a.trading_date_plus3
```

```
      group by a.permno, a.trading_date;
quit;

* Check for de-listing returns *;
proc sql;
      create table target_benchmark7 as select
            a.*, b.dlret
      from target_benchmark7 as a left join crsp.DSEDELIST (where=(dlret^=. and
DLSTDT^=.)) as b
      on a.permno=b.permno
      and b.dlstdt between a.rdq_to_use and a.trading_date_plus2;
quit;

proc download data=target_benchmark7 out=target_benchmark7; run;
endrsubmit;

* If original returns were missing or delisted then update as applicable *;
data target_benchmark7 (drop=ret260_alt ret240_alt ret220_alt ret210_alt ret205_alt
ret204_alt
                                    ret203_alt dlret); set target_benchmark7;
      if ret260=. then ret260=ret260_alt;
      if ret260=. then ret260=dlret;
      if ret240=. then ret240=ret240_alt;
      if ret240=. then ret240=dlret;
      if ret220=. then ret220=ret220_alt;
      if ret220=. then ret220=dlret;
      if ret210=. then ret210=ret210_alt;
      if ret210=. then ret210=dlret;
      if ret205=. then ret205=ret205_alt;
      if ret205=. then ret205=dlret;
      if ret204=. then ret204=ret204_alt;
      if ret204=. then ret204=dlret;
      if ret203=. then ret203=ret203_alt;
      if ret203=. then ret203=dlret;
run;

* Merge in portfolio returns *;
proc sql;
      create table target_returns as select distinct
            a.*, b.date, b.ew_ret_260, b.ew_ret_240, b.ew_ret_220, b.ew_ret_210,
b.ew_ret_205, b.ew_ret_204, b.ew_ret_203
      from target_benchmark7 as a left join char.port_ret_combined as b
      on a.size_quintile = b.size_quintile
      and a.btm_quintile = b.btm_quintile
      and a.mom_quintile = b.mom_quintile
      and a.caldt = b.caldt
      and intck('day',a.trading_date,b.date) = 0;
quit;

** Save CAR and portfolio returns *;
data char.returns_main(keep=permno rdq_to_use trading_date ret260 ret240 ret220
ret210 ret205 ret204 ret203
                                    ew_ret_260 ew_ret_240 ew_ret_220 ew_ret_210
ew_ret_205 ew_ret_204 ew_ret_203);
      set target_returns;
run;


*****************************************************************;
** V. Calculate BHAR over specified range
*****************************************************************;
```

```
1221
1222  * Bring in main dataset again *;
1223  data main; set master.step2_final; run;
1224  * Fix dates for merging *;
1225  data main; set main;
1226        rdq2 = input(put(rdq,8.),yymmdd8.);
1227        * This is for GVKEY 151928 *;
1228        if permno = 89613 then permno = 16852;
1229        drop rdq;
1230  run;
1231  data main; set main;
1232        rdq = rdq2;
1233        drop rdq2;
1234        format rdq yymmddn8.;
1235  run;
1236
1237  * Merge firm and portfolio returns into main dataset *;
1238  proc sql;
1239        create table main2 as select
1240              a.*, b.ret260, b.ret240, b.ret220, b.ret210, b.ret205, b.ret204,
1241  b.ret203,
1242              b.ew_ret_260, b.ew_ret_240, b.ew_ret_220, b.ew_ret_210, b.ew_ret_205,
1243  b.ew_ret_204, b.ew_ret_203
1244        from main as a left join char.returns_main as b
1245        on a.permno = b.permno
1246        and a.rdq_to_use = b.rdq_to_use;
1247  quit;
1248
1249  * Calculate portfolio-adjusted returns *;
1250  * Multiply by 100 to express in terms of basis points *;
1251  data main2 (drop=ew_ret_260 ew_ret_240 ew_ret_220 ew_ret_210 ew_ret_205 ew_ret_204
1252  ew_ret_203); set main2;
1253        ar_260_sbm = 100*(ret260-ew_ret_260);
1254        ar_240_sbm = 100*(ret240-ew_ret_240);
1255        ar_220_sbm = 100*(ret220-ew_ret_220);
1256        ar_210_sbm = 100*(ret210-ew_ret_210);
1257        ar_205_sbm = 100*(ret205-ew_ret_205);
1258        ar_204_sbm = 100*(ret204-ew_ret_204);
1259        ar_203_sbm = 100*(ret203-ew_ret_203);
1260  run;
1261
1262  * Save down *;
1263  * Use this in Stata for the LT returns tests *;
1264  data master.sizebtmmom; set main2(keep=gvkey fyearqtr ar_260_sbm ar_240_sbm
1265  ar_220_sbm ar_210_sbm
1266                                                    ar_205_sbm ar_204_sbm ar_203_sbm);
1267  run;
1268
1269
1270  ******************;
1271  *** END OF FILE ***;
1272  ******************;
1273
```