The following piece of code was written to replicate tables of Hirshleifer et al (2004) titled: Do Investors Overvalue Firms with Bloated Balance Sheets?

```
*************************************************************************

libname cleandat "C:\SASData\NewData";
libname result "C:\SAS Data\Output";

/* Clean out log */
dm "out;clear;log;clear;";
%let raw_1 = Compustat_hw4;
%let raw_2 = CRSP_hw4;
%let raw_3 = NYSEAMEX;
%let fstartyr = 1963;  /* define start year of compustat data */
%let fendyr = 2000;    /* define end year of compustat data */
%let startyr = 1964;   /* define start year of crsp data */
%let startm = 6;     /* define start month of crsp data */
%let endyr = 2002;         /* define end year of crsp data */

/* input raw data */
/* input compustat */
/* select NYSE/AMEX/NASDAQ stocks
11 - NYSE, 12 - AMEX, 14 - NASDAQ */
data compustat(drop = CONM SEQ IB REVT);
      set cleandat.&raw_1;
      if EXCHG= 11 or EXCHG= 12 or EXCHG= 14;
run;

/* input crsp */
/* select NYSE/AMEX/NASDAQ stocks: 1 - NYSE, 2 - AMEX, 3 - NASDAQ */
data crsp; set cleandat.&raw_2;
      if EXCHCD = 1 or EXCHCD = 2 or EXCHCD = 3;
      month = month(date);
      year = year(date);
run;

/* input NYSE/AMEX return */
data NYSEAMEX(keep = year month mktret);
      set cleandat.&raw_3;
      year = year(caldt);
      month = month(caldt);
      rename ewretd = mktret;
run;



/*********************************************/
/* Table 1, Part I NOA and other fundamentals */
/*********************************************/

/* Input Compustat */
/*
Old   Description                        New
1       Cash and short Term Investment    CHE
4       Current Asset                       ACT
5       Current Liabilities                LCT
```

```
6       Total Asset                          AT
9       Long Term Debt                    DLTT
14      Depreciation and Amortization        DP
25      Shares Outstanding                 CSHO
34      Debt included in current liabilities DLC
38      Minor Interests                   MIB
60      Book Value of Common Equity       CEQ
71      Income Tax Payable                  TXP
130     Preferred Stocks                     PSTK
178     Income From Continuing Operations OIADP
199     Fiscal Year End Closing Price        PRCC_F
*/

/* ********************************************************** */

/* clean compustat data */
data compustat (drop = gvkey_char);
      set compustat (rename = (gvkey = gvkey_char));
      cusip = substr(cusip,1,8);    /* reshape cusip to match crsp */
      gvkey = gvkey_char*1;   /* change gvkey to number */
      month = month(datadate);
      year = year(datadate);
      lagAT = lag(AT);  /* get total asset of last year */
    rename CEQ = BV;  /* book value of common equity */
run;

/* if some variables are missing, they can be reasonalby assumed
to be 0. see page 17 */
data compustat; set Compustat;
      if missing(DLC) then DLC = 0;
      if missing(TXP) then TXP = 0;
      if missing(DLTT) then DLTT = 0;
      if missing(MIB) then MIB = 0;
      if missing(PSTK) then PSTK = 0;
run;

/* if some variables are missing or negative, they must be dropped */
data compustat; set Compustat;
      if missing(CHE) = 0 and missing(ACT) = 0 and missing(LCT) = 0
         and missing(AT) = 0 and missing(DP) = 0 and missing(CSHO) = 0
         and missing(BV) = 0 and missing(OIADP) = 0 and missing(PRCC_F) = 0
         and missing(lagAT) = 0
         and AT > 0 and lagAT > 0 and CSHO > 0 and BV > 0 and PRCC_F > 0;
run;

/* ********************************************************** */

/* calculate fundamentals, meanings of variables see above chart and page 44
of the paper*/
data compustat; set compustat;
    OA = AT-CHE;
    OL = AT-DLC-DLTT-MIB-PSTK-BV;
      RawNOA = OA-OL;
    NOA = RawNOA/lagAT;
      Earnings = OIADP/lagAT;
      Rawaccruals = (ACT-lag(ACT))-(CHE-lag(CHE))-(LCT-lag(LCT))
                          +(DLC-lag(DLC))+(TXP-lag(TXP))-DP;
```

```
      /* be cautious to use lag function as the first line of each gvkey is
wrong
         will be adjusted later */
      Accruals = Rawaccruals/lagAT;
      Cashflows = Earnings-Accruals;
    MV = PRCC_F*CSHO;   /* market cap */
      Cash = CHE/lagAT;
      Equity = BV/lagAT;
      Debt = NOA-(Equity+Cash);
      BM = BV/MV; /* book to market */
run;

/* clean NOA and first line problem */
data compustat; set compustat; if missing(NOA) = 0; run;
/* delete the first row of each gvkey because its lag numbers are wrong */
proc sort data = compustat; by by gvkey year; run;
data compustat; set compustat; by gvkey year;
      if first.gvkey and first.year then delete;
run;

/* output necessary variables for Table 1 Part I calculation */
/* table_1 is for the calculation of Table 1 Part I */
data table_1 (keep = gvkey datadate fyear year month TIC cusip NOA
                              Earnings Accruals Cashflows BV MV BM);
      set compustat;
      if fyear >= &fstartyr and fyear <= &fendyr;
      /* use the same data range with the paper after we clean the data */
run;

/* ********************************************************** */

/* rank by NOA in each fiscal year */
proc sort data = table_1; by fyear; run;

proc rank data = table_1 out = table_1 group = 10;
      var NOA; by fyear;
      ranks NOA_rank;
run;

/* calculate mean and median stat of each NOA decile */
proc sort data = table_1; by NOA_rank; run;

proc means data = table_1 noprint;
      var NOA Earnings Accruals Cashflows BV MV BM;
      by NOA_rank;
      output out = meanstat mean = meanNOA meanEarning meanAccruals
meanCashflows meanBV meanMV meanBM;
run;

proc means data = table_1 noprint;
      var NOA Earnings Accruals Cashflows BV MV BM;
      by NOA_rank;
      output out = medianstat median = medianNOA medianEarning medianAccruals
medianCashflows medianBV medianMV medianBM;
run;

/* Output results */
```

```
proc export data = meanstat outfile = "C:\SAS Data\Output\HW4_meanstat.csv"
DBMS = csv replace; run;

proc export data = medianstat outfile = "C:\SAS
Data\Output\HW4_medianstat.csv" DBMS = csv replace; run;



/*****************************************/
/* Table 1, Part II Beta of each NOA decile */
/*****************************************/

/* merge compustat with crsp */
/* crspmerge is used to merge with compustat */
data crspmerge (keep = permno cusip date year month ret);
      set crsp;
      if missing(dlret) = 0 then ret = dlret;
      if missing(ret) or ret < -1 then delete;
run;

proc sort data = crspmerge; by permno year month; run;

/* prepare compustat for merge, compustatmerge is used to merge with
crspmerge */
/* move month of report date forward 4 month, the finnancial data will be
used
since this month and onward, as assumed in pate 23 */
data compustatmerge (keep = gvkey datadate fyear year fmonth cusip NOA MV
BM);
      set compustat;
      if month >= 1 and month <= 8 then month = month+4;
      else if month >= 9 and month <= 12 then do;
            month = month - 8;
            year = year+1;
      end;
      rename month = fmonth;  /* this is the month financial statement used
by investor */
run;

/* Add permnos to compustat by matching cusips */
data cusips (keep = permno cusip year); set crsp;
      if month = 1;
run;

proc sort data = cusips; by cusip year; run;
proc sort data = compustatmerge; by cusip year; run;

data compustatmerge;
      merge compustatmerge(in = k) cusips;
      by cusip year;
      if k;
run;

/* merge crspmerge and compustatmerge by permno
we think the reason not use cusip to merge is that cusip is not perfectly
number
fin_ret contains financial data and returns */
```

```
proc sort data = compustatmerge; by permno year; run;
data fin_ret;
      merge crspmerge compustatmerge(in = k);
      by permno year;
      if k;
run;

/* delete data that does no have valid permno (cannot be identified) */
data fin_ret; set fin_ret; if missing(permno) = 0; run;

/* adjust financial data. when merge, all physical year has same financial
data
however, investors only use the financial data after fmonth (when financial
data
of that year come out, before that month, invesor use last year financial
data */
proc sort data = fin_ret; by permno year month; run;
%macro fin_adj;
%do iter = 1 %to 11;
data fin_ret; set fin_ret;
      if fmonth = &iter+1 and month < fmonth then do;
      /* if month < fmonth, investor need to use last year data */
            NOA = lag&iter(NOA);
            MV = lag&iter(MV);      /* market cap in the end of fiscal year
*/
            BM = lag&iter(BM);      /* book to market in the end of fiscal
year */
      end;
%end;
%mend;
%fin_adj;

/* delete the data that is not correct */
data fin_ret; set fin_ret;
      by permno year month;
      if (first.permno or first.year) and month < fmonth then delete;
      /* during the first year and before the fmonth, investor has no
financial data */
      if missing(NOA) then delete;
      if missing(MV) or MV <= 0 then delete;
      if missing(BM) or BM <=0 then delete;
      /* BV (book value) and BM (book to market) are from compustat and are
in
      the end of fiscal year, as stated by the paper */
run;

/* after clean the data, use the data in the data range as stated in the
paper
from now fin_ret contains cleaned return and financial data with financial
data
at the right month that the investor should begin to use */
data fin_ret; set fin_ret; if year >= &startyr and year <= &endyr; run;
data fin_ret; set fin_ret; if year = &startyr and month < &startm then
delete; run;

/* ******************************************************** */
```

```
/* rank NOA for each month */
proc sort data = fin_ret; by year month; run;
proc rank data = fin_ret out = fin_ret group = 10;
      var NOA; by year month;
      ranks NOA_rank;
run;

proc sort data = NYSEAMEX; by year month; run;

/* betacal is for beta calculation */
data betacal(keep = permno year month NOA_rank ret); set fin_ret; run;

/* sort betacal in descending order for id identification (prior 60mon
return) */
proc sort data = betacal; by permno descending year month; run;

/* this macro is to calculate the beta of each NOA decile in each month. the
idea
is in each month, identify the stocks that belongs to on NOA decile and then
retrieve
the previous 60mon returns. use these returns to get equal weighted
historical return and
regress against NYSE/AMEX equal weighted index to get the beta of that NOA
decile in that month */
/* WARNING: This is a big loop and will take a lot of time.  Please reduce
the loop of
rank, year, and m to save time if you just want to test (e.g. set y = 1980 to
1981) */
%macro beta_cal;
%do rank = 0 %to 9;     /* loop of NOA decile */

data NOA_&rank._beta; run;
/* create empty dataset to store beta of each month for that NOA decile */

%do y = &startyr+5 %to &endyr;
/* %do y = &startyr+5 %to &endyr; */
      %do m = 1 %to 12; /* loop of year and month */

      data betacal; set betacal;
            by permno descending year month;
            retain id;
            /* use id to identify the returns that we want to use for
regression */
            if year = &y and month = &m and NOA_rank = &rank then id = 1;
            else if first.permno = 0 then id = id+1;
            else if first.permno then id = 61;
      run;

      /* drop id to prevent interfere to the next calculation */
      /* id < 60 are the recent 60 historical returns that we will use for
the regression
         to get the beta of this NOA decile in this month */
      data beta_sub(drop = id); set betacal; if missing(id) = 0 and id <= 60;
run;
      data betacal(drop = id); set betacal; run;

      proc sort data = beta_sub; by year month; run;
```

```
      /* calculate monthly equal weighted return of NOA portfolio */
      proc means data = beta_sub noprint;
            var ret; by year month;
            output out = return mean = NOAret;
      run;

      /* merge NOA portfolio return with NYSE/AMEX equal weighted return in
regression time period */
      data return (drop = _TYPE_ _FREQ_);
            merge return(in = k) NYSEAMEX;
            by year month;
            if k;
      run;

      /* delete data that is obviously wrong */
      data return; set return;
            if (&y-year)*12+(&m-month) < 0 or (&y-year)*12+(&m-month) >= 60
                  or missing(NOAret) or missing (mktret) then delete;
      run;

      /* regress NOA portfolio on NYSE/AMEX index, regression results are
stored in est dataset */
      proc reg data = return outest = est noprint;
            model NOAret = mktret;
      run;
      quit; /* stop the regression once it is done to speed up the
performance */

      /* store all beta in all month for this NOA decile in one dataset
(NOA_&rank.beta)
      the beta (coefficient of mktret) is store as mktret variable in est
dataset */
      data NOA_&rank._beta; set NOA_&rank._beta est(keep = mktret); run;
      %end;
%end;

/* rename the beta variable */
data NOA_&rank._beta; set NOA_&rank._beta;
      rename mktret = NOA_&rank;
run;
data NOA_&rank._beta; set NOA_&rank._beta;
      if missing(NOA_&rank) = 0;
      line = _n_;
run;
%end;
%mend;
%beta_cal;

/* ********************************************************** */

/* NOA_beta stores beta of each month for all NOA decile */
data NOA_beta; set NOA_0_beta; run;
%macro beta_merge;
%do iter = 1 %to 9;
data NOA_beta;
      merge NOA_beta NOA_&iter._beta;
```

```
      by line;
run;
%end;
%mend;
%beta_merge;

proc means data = NOA_beta noprint;
      var NOA_0 NOA_1 NOA_2 NOA_3 NOA_4 NOA_5 NOA_6 NOA_7 NOA_8 NOA_9;
      /* these are the betas for each NOA decile */
      output out = betamean mean = NOA_0 NOA_1 NOA_2 NOA_3 NOA_4 NOA_5 NOA_6
NOA_7 NOA_8 NOA_9;
run;

proc means data = NOA_beta noprint;
      var NOA_0 NOA_1 NOA_2 NOA_3 NOA_4 NOA_5 NOA_6 NOA_7 NOA_8 NOA_9;
      output out = betamedian median = NOA_0 NOA_1 NOA_2 NOA_3 NOA_4 NOA_5
NOA_6 NOA_7 NOA_8 NOA_9;
run;

/* Output results */
proc export data = betamean outfile = "C:\SAS Data\Output\HW4_betamean.csv"
DBMS = csv replace; run;
proc export data = betamedian outfile = "C:\SAS
Data\Output\HW4_betamedian.csv" DBMS = csv replace; run;



/*************************************************************/
/* Table 4, Part I. Abnormal returns of NOA decile portfolios */
/*************************************************************/

/* calculate next 1yr, 2yr, 3yr return for each NOA decile - leadret1
leadret2 leadret3 */
/* fin_ret is created in Table 1, Part II, sort descending to calculate
future average return */
proc sort data = fin_ret; by permno descending year month; run;

data fin_ret; set fin_ret;
      by permno descending year month;
      leadret1 = 0; leadret2 = 0; leadret3 = 0;
      retain line;
      if first.permno then line = 1;
      else line = line+1;
      /* use line to control time step */
run;

/* This macro is used to calculate next 1yr, 2yr, 3yr return - leadret1
leadret2 leadret3 */
%macro lead_cal;
%do iter = 1 %to 36;
data fin_ret; set fin_ret;
      by permno descending year month;
      if &iter <= 12 then leadret1 = leadret1+lag&iter(ret);
      /* accumulative return from t+1 to t+12 month */
      if &iter <= 24 then leadret2 = leadret2+lag&iter(ret);
      /* accumulative return from t+1 to t+24 month */
      leadret3 = leadret3+lag&iter(ret);
```

```
      /* accumulative return from t+1 to t+36 month */
%end;
%mend;
%lead_cal;

/* the first 36 lines of each stock are wrong, as in these time period there
are no
sufficient data for t+3 return */
data fin_ret (drop = line); set fin_ret; if line <= 36 then delete; run;

/* perform similar calculation to calculate cumulative returns from t-2 to t-
12 month */
proc sort data = fin_ret; by permno year month; run;

data fin_ret; set fin_ret;
      by permno year month;
      lagret = 0;
      retain line;
      if first.permno then line = 1;
      else line = line+1;
      /* use line to control the calculation of PR1YR */
run;

/* This macro is used to calculate cumulative returns from t-2 to t-12 month
*/
%macro lag_cal;
%do iter = 2 %to 12;
data fin_ret; set fin_ret;
      by permno year month;
      lagret = (1+lagret)*(1+lag&iter(ret))-1;
      /* cumulative return from t-2 to t-12 month */
%end;
%mend;
%lag_cal;

/* the first 12 lines are calculated wrong */
data fin_ret (drop = line); set fin_ret; if line <= 12 then delete; run;

/* adjust returns to monthly average and delete data out of range */
data fin_ret;
      set fin_ret;
      leadret1 = leadret1/12; /* future average is athimetic average */
      leadret2 = leadret2/24;
      leadret3 = leadret3/36;
run;

/* ********************************************************** */

/* rank size, book/mkt, and PR1YR of each month, size(MV) and b/m(BM)
are value in each fiscal year end */
/* each month has 125 groups, this may lead to the problem that some
group only have one stock */
proc sort data = fin_ret; by year month; run;
proc rank data = fin_ret out = fin_ret group = 5;
      var MV; by year month;
      ranks size_rank;
run;
```

```
proc sort data = fin_ret; by year month size_rank; run;
proc rank data = fin_ret out = fin_ret group = 5;
      var BM; by year month size_rank;
      ranks BM_rank;
run;

proc sort data = fin_ret; by year month size_rank BM_rank; run;
proc rank data = fin_ret out = fin_ret group = 5;
      var lagret; by year month size_rank BM_rank;
      ranks ret_rank;
run;

/* calculate sum mktcap of 125 groups */
proc sort data = fin_ret; by year month size_rank BM_rank ret_rank; run;

proc means data = fin_ret noprint;
            var MV; by year month size_rank BM_rank ret_rank;
            output out = sumstat sum = sumcap;
run;

data fin_ret;
      merge fin_ret sumstat(drop = _TYPE_);
      by year month size_rank BM_rank ret_rank;
run;

/* ********************************************************** */

/* calculate S/BM/Mom benchmark return for t+1, t+2, t+3 */
/* This macro calculates the equal-weighted and value weighted benchmarks
S/BM/Mom */
%macro benchmark_ret(datain = );
/* use datain to control input data set */
%do code = 1 %to 2;
      data retcomp; set &datain; run;

      data retcomp; set retcomp;
            if &code = 1 then wt = MV/sumcap;
            else if &code = 2 then wt = 1/_FREQ_;
      run;

      /* Calculate benchmark weighted returns */
      proc sort data = retcomp; by year month size_rank BM_rank ret_rank;
run;

      proc means data = retcomp noprint;
            var leadret1 leadret2 leadret3;
            weight wt;
            by year month size_rank BM_rank ret_rank;
            output out = return mean = bmkret1_&code bmkret2_&code
bmkret3_&code;
      run;

      /* Output Returns */
      data return(drop = _TYPE_ _FREQ_); set return;
            if missing(bmkret1_&code) = 0 and missing(bmkret2_&code) = 0 and
               missing(bmkret3_&code) = 0 and missing(month) = 0;
```

```
      run;

      data &datain;
            merge &datain return;
            by year month size_rank BM_rank ret_rank;
      run;
%end;
%mend;
%benchmark_ret(datain = fin_ret);

/* calculate adjusted return against benchmark */
data fin_ret (drop = sumcap _FREQ_); set fin_ret;
      /* _1 is value/cap weighted */
      adjret1_1 = leadret1-bmkret1_1;
      adjret2_1 = leadret2-bmkret2_1;
      adjret3_1 = leadret3-bmkret3_1;
      /* _2 is equal weighted */
      adjret1_2 = leadret1-bmkret1_2;
      adjret2_2 = leadret2-bmkret2_2;
      adjret3_2 = leadret3-bmkret3_2;
      if _FREQ_ <= 1 then delete;
      /* some groups only have one stocks, it is excluded */
run;

/* ********************************************************** */

/* calculate raw return and adjust return of each NOA decile */
/* monthly NOA rank are set in Table 1 Part II */

/* this macro is listed first and will be used in the next macro */
/* calculate the equal weighted and value weighted average future return of
the NOA decile */
/* there are several weighted return macros in this file, we don't have time
to
consolidate it in to one as we did last time */
%macro NOA_ret(datain = , rank = );
/* use datain to control input data set */
%do code = 1 %to 2;
      data retcomp; set &datain; run;

      /* select weighting method */
      data retcomp; set retcomp;
            if &code = 1 then wt = MV/sumcap;
            else if &code = 2 then wt = 1/_FREQ_;
      run;

      proc sort data = retcomp; by year month; run;

      /* calculate monthly weighted returns */
      proc means data = retcomp noprint;
            var leadret1 adjret1_&code adjret2_&code adjret3_&code;
            weight wt; by year month;
            output out = return mean = leadret1 adjret1 adjret2 adjret3;
      run;

      /* output returns */
      data ret_&datain._&code(drop = _TYPE_ _FREQ_);
```

```
            set return; NOA_rank = &rank;
            if missing(leadret1) = 0 and missing(adjret1) = 0 and
               missing(adjret2) = 0 and missing(adjret3) = 0 and
               missing(month) = 0;
      run;
%end;
%mend;

/* seperate each NOA decile in to each sub database and calculate return */
%macro NOA_decile;
%do iter = 0 %to 9;
      data NOA_&iter; set fin_ret;
            if NOA_rank = &iter;
      run;

      proc sort data = NOA_&iter; by year month; run;

      proc means data = NOA_&iter noprint;
            var MV; by year month;
            output out = sumstat sum = sumcap;
      run;

      data NOA_&iter;
            merge NOA_&iter sumstat(drop = _TYPE_);
            by year month;
      run;

      %NOA_ret(datain = NOA_&iter, rank = &iter);
%end;
%mend;
%NOA_decile;

/* ********************************************************** */

/* vertical merge return data */
data NOA_cap; set ret_NOA_0_1; run; /* cap use all _1 */
data NOA_eq; set ret_NOA_0_2; run;  /* eq use all _2 */

%macro NOA_merge;
%do iter = 1 %to 9;
      data NOA_cap; set NOA_cap ret_NOA_&iter._1; run;
      data NOA_eq; set NOA_eq ret_NOA_&iter._2; run;
%end;
%mend;
%NOA_merge;

/* calculate L-H for eq and cap, assign it to NOA_rank = 10 */
/* cap weighted */
data ret_NOA_9_1; set ret_NOA_9_1;
      rename leadret1 = leadret1_9;
      rename adjret1 = adjret1_9;
      rename adjret2 = adjret2_9;
      rename adjret3 = adjret3_9;
run;

data ret_NOA_10_1;
      merge ret_NOA_0_1 (drop = NOA_rank) ret_NOA_9_1 (drop = NOA_rank);
```

```
      by year month;
run;

data ret_NOA_10_1 (drop = leadret1_9 adjret1_9 adjret2_9 adjret3_9);
      set ret_NOA_10_1;
      leadret1 = leadret1-leadret1_9;
      adjret1 = adjret1-adjret1_9;
      adjret2 = adjret2-adjret2_9;
      adjret3 = adjret3-adjret3_9;
      NOA_rank = 10;
run;
/* vertical merge NOA_10_1 to NOA_cap */
data NOA_cap; set NOA_cap ret_NOA_10_1; run;

/* equal weighted */
data ret_NOA_9_2;
      set ret_NOA_9_2;
      rename leadret1 = leadret1_9;
      rename adjret1 = adjret1_9;
      rename adjret2 = adjret2_9;
      rename adjret3 = adjret3_9;
run;

data ret_NOA_10_2;
      merge ret_NOA_0_2 (drop = NOA_rank) ret_NOA_9_2 (drop = NOA_rank);
      by year month;
run;

data ret_NOA_10_2 (drop = leadret1_9 adjret1_9 adjret2_9 adjret3_9);
      set ret_NOA_10_2;
      leadret1 = leadret1-leadret1_9;
      adjret1 = adjret1-adjret1_9;
      adjret2 = adjret2-adjret2_9;
      adjret3 = adjret3-adjret3_9;
      NOA_rank = 10;
run;
/* vertical merge NOA_10_2 to NOA_eq */
data NOA_eq; set NOA_eq ret_NOA_10_2; run;

/* ********************************************************** */

/* calculate the average weighted returns of each NOA decile */
proc sort data = NOA_cap; by NOA_rank year month; run;
proc means data = NOA_cap noprint;
      var leadret1 adjret1 adjret2 adjret3;
      by NOA_rank;
      output out = table4_cap mean = leadret1 adjret1 adjret2 adjret3;
run;
proc export data = table4_cap outfile =
            "C:\SAS Data\Output\HW4_table4_cap.csv" DBMS = csv replace; run;

proc sort data = NOA_eq; by NOA_rank year month; run;
proc means data = NOA_eq noprint;
      var leadret1 adjret1 adjret2 adjret3;
      by NOA_rank;
      output out = table4_eq mean = leadret1 adjret1 adjret2 adjret3;
run;
```

```
proc export data = table4_eq outfile =
            "C:\SAS Data\Output\HW4_table4_eq.csv" DBMS = csv replace; run;

/* ******************************************************** */

/* calculate the t-stat */
proc ttest data = NOA_cap;
      var leadret1 adjret1 adjret2 adjret3;
      by NOA_rank;
      ods output Ttests = NOA_cap_ttest;
run;

proc export data = NOA_cap_ttest outfile =
            "C:\SAS Data\Output\HW4_NOA_cap_ttest.csv" DBMS = csv replace;
run;

proc ttest data = NOA_eq;
      var leadret1 adjret1 adjret2 adjret3;
      by NOA_rank;
      ods output Ttests = NOA_eq_ttest;
run;

proc export data = NOA_eq_ttest outfile =
            "C:\SAS Data\Output\HW4_NOA_eq_ttest.csv" DBMS = csv replace;
run;



/**************************/
/* Table 4, Part II Alphas */
/**************************/

/* construct market/FAMA-French/four factor portforlio */
/* FAMA-French/four factor portforlio are constructed using code of last hw
*/
data compustat(drop = datadate gvkey_char);
      set cleandat.&raw_1(rename = (gvkey = gvkey_char));
      rename ib = income;
      rename seq = book;
      rename fyear = year;
      rename tic = comp_ticker;
      cusip = substr(cusip,1,8);
      gvkey = gvkey_char*1;
      if fyear >= &startyr-2 and fyear <= &endyr-2;
run;

/* Input CRSP */
data crsp(drop = date dlret prc shrout); set cleandat.&raw_2;
      month = month(date);
      year = year(date);
      if missing(dlret) = 0 then ret = dlret;
      if ret < -1 then delete;
      mktcap = abs(prc)*abs(shrout);
      if year >= &startyr-2 and year <= &endyr;
run;

/* Add permnos to compustat by matching cusips */
```

```
data cusips(keep = permno cusip ticker year); set crsp;
      if month = 1;
run;

proc sort data = cusips; by cusip year; run;
proc sort data = compustat; by cusip year; run;

data compustat;
      merge compustat(in = k) cusips;
      by cusip year;
      if k;
      year = year+2;
run;

/* Add Market cap (Dec of y-1) data to financial data */
data mktcap_dec(keep = year permno mktcap_dec); set crsp;
      if month = 12;
      year = year+1;
      rename mktcap = mktcap_dec;
run;

proc sort data = mktcap_dec; by year permno; run;
proc sort data = compustat; by year permno; run;

data universe;
      merge compustat(in = k) mktcap_dec;
      by year permno;
      if k;
run;

proc means data = universe;
      var mktcap_dec book income;
      by year;
      output out = sumstat sum = sumcap sumbook sumincome;
run;

data universe;
      merge universe sumstat(drop = _TYPE_);
      by year;
run;

/* Calculate book/cap value */
data universe; set universe; BKMK = book/mktcap_dec; run;

/* Rank market cap and book/cap */
proc rank data = universe out = universe group = 10;
      var mktcap_dec; by year;
      ranks cap_rank;
run;

proc rank data = universe out = universe group = 10;
      var BKMK; by year;
      ranks BKMK_rank;
run;

/* Merge universe with return data from CRSP */
proc sort data = crsp(keep = permno year month ret retx) out = crsp_subset;
```

```
      by permno year;
run;
proc sort data = universe; by permno year; run;

data universe;
      merge crsp_subset universe(in = k);
      by permno year;
      if k;
run;

/* ********************************************************** */

/* this macro calculates weighted average return */
%macro ret_cal(datain = , start_code = , end_code = , rebal = , rebal_mon =
);
/* use datain to control input data set */
/* use start_code and end_code to control weighting method */
/* code 1-cap weighted; 2-equal weighted; 3-earning weighted; 4-book weighted
*/
/* rebal control rebalance frequency. 1-annually; 2-monthly */
/* rebal_mon control on which month to perform annually rebalance */

%do code = &start_code %to &end_code;
      data retcomp; set &datain; run;

      data retcomp; set retcomp;
            if &code = 1 then weight = mktcap_dec/sumcap;
            else if &code = 2 then weight = 1/_FREQ_;
            else if &code = 3 then weight = income/sumincome;
            else if &code = 4 then weight = book/sumbook;
      run;

      /* Calculate dynamic weights */
      proc sort data = retcomp; by permno year month; run;

      data retcomp; set retcomp;
            by permno year month;
            lagretx = lag(retx);
            if first.permno then lagretx = 0;
      run;

      /* Calculate rebalance */
      %if &rebal = 1 %then %do;
      /* &rebal = 1,Annual rebalance */
            data retcomp; set retcomp;
                  by permno year month;
                  retain dyn_wt;
                  if first.permno or month = &rebal_mon then dyn_wt = weight;
                  /* rebalance in the rebal_mon month */
                  else dyn_wt = dyn_wt*(1+lagretx);   /* Rebalance of the
portfolio */
            run;
      %end;
      %else %if &rebal = 2 %then %do;
      /* &rebal = 2, Monthly rebalance */
            data retcomp; set retcomp;
                  dyn_wt = weight;
```

```
            run;
        %end;

        /* Calculate Returns */
        proc sort data = retcomp; by year month; run;

        proc means data = retcomp noprint;
            var dyn_wt; by year month;
            output out = sumwt sum = sumwt;
        run;

        data retcomp(drop = _TYPE_ _FREQ_ sumwt);
            merge sumwt retcomp; by year month;
            dyn_wt = dyn_wt/sumwt;
        run;

        proc means data = retcomp noprint;
            var ret retx;
            weight dyn_wt;
            by year month;
            output out = return mean = TR_&datain PR_&datain;
        run;

        /* Output Returns */
        data ret_&datain._&code._&rebal(drop = _TYPE_ _FREQ_);
            set return;
            if missing(TR_&datain) = 0 and missing(month) = 0;
        run;
    %end;
    %mend;

    /* Market/CRSP annual rebalance */
    %ret_cal(datain = universe, start_code = 1, end_code = 1, rebal = 1,
    rebal_mon = 1);
    /* Get market return */
    proc export data = ret_universe_1_1 outfile = "C:\SAS
    Data\Output\mkt_ret.csv" DBMS = csv replace; run;

    /* ******************************************************** */

    /* Fama-French */
    /* FF portfolio, drop data that do not have marketcap or book data in T, T-1,
    and T-2 or do not have active market return */
    data ff; set universe;
        if missing(mktcap_dec) = 0 and missing(book) = 0 and
           missing(lag1(mktcap_dec)) = 0 and missing(lag1(book)) = 0 and
           missing(lag2(mktcap_dec)) = 0 and missing(lag2(book)) = 0 and
           missing(ret) = 0;
    run;

    /* Create big-growth, big-neutral, big-value, small-growth, small-neutral,
    small-value portfolio */
    data ff_bg; set ff; if cap_rank >= 8 and BKMK_rank >= 0 and BKMK_rank <= 2;
    run;
    data ff_bn; set ff; if cap_rank >= 8 and BKMK_rank >= 3 and BKMK_rank <= 6;
    run;
```

```
data ff_bv; set ff; if cap_rank >= 8 and BKMK_rank >= 7 and BKMK_rank <= 9;
run;
data ff_sg; set ff; if cap_rank >= 0 and cap_rank <= 7 and BKMK_rank >= 0 and
BKMK_rank <= 2; run;
data ff_sn; set ff; if cap_rank >= 0 and cap_rank <= 7 and BKMK_rank >= 3 and
BKMK_rank <= 6; run;
data ff_sv; set ff; if cap_rank >= 0 and cap_rank <= 7 and BKMK_rank >= 7 and
BKMK_rank <= 9; run;

/* Size and Value portfolios are reconstituted annually at the end of June,
returns are cap weighted */
%ret_cal(datain = ff_bg, start_code = 1, end_code = 1, rebal = 1, rebal_mon =
6);
%ret_cal(datain = ff_bn, start_code = 1, end_code = 1, rebal = 1, rebal_mon =
6);
%ret_cal(datain = ff_bv, start_code = 1, end_code = 1, rebal = 1, rebal_mon =
6);
%ret_cal(datain = ff_sg, start_code = 1, end_code = 1, rebal = 1, rebal_mon =
6);
%ret_cal(datain = ff_sn, start_code = 1, end_code = 1, rebal = 1, rebal_mon =
6);
%ret_cal(datain = ff_sv, start_code = 1, end_code = 1, rebal = 1, rebal_mon =
6);

/* Momentum */
/* Momentum portfolio, drop data that does not have mktcap and return value
*/
/* first 12 month are dropped later */
data mom; set universe;
      if missing(mktcap_dec) = 0 and missing(ret) = 0;
run;

proc sort data = mom; by permno year month; run;

data mom; set mom;
      by permno year month;
      cum_ret = 0;
      retain line;
      if first.permno then line = 1;
      else line = line+1;
      /* use line to control the calculation of PR1YR */
run;

/* This macro is used to calculate cumulative returns from t-2 to t-12 */
%macro cum_cal;
%do iter = 2 %to 12;
data mom; set mom;
      by permno year month;
      cum_ret = (1+cum_ret)*(1+lag&iter(ret))-1;
      /* accumulative return from t-2 to t-12 */
      %end;
%mend cum_cal;
%cum_cal;

data mom; set mom; if line >= 13; run;
/* drop the first 12 month */
```

```
proc sort data = mom; by year month; run;
proc rank data = mom out = mom group = 10;
      var cum_ret; by year month;
      ranks ret_rank;
run;

/* Create big-up, big-down, small-up, small-down portfolio */
data mom_bd; set mom; if cap_rank >= 8 and ret_rank >= 0 and ret_rank <= 2;
run;
data mom_bu; set mom; if cap_rank >= 8 and ret_rank >= 7 and ret_rank <= 9;
run;
data mom_sd; set mom; if cap_rank >= 0 and cap_rank <= 7 and ret_rank >= 0
and ret_rank <= 2; run;
data mom_su; set mom; if cap_rank >= 0 and cap_rank <= 7 and ret_rank >= 7
and ret_rank <= 9; run;

/* After getting the portfolio, redo the monthly rebalanced, cap weighting
return */
%ret_cal(datain = mom_bd, start_code = 1, end_code = 1, rebal = 2, rebal_mon
= 1);
%ret_cal(datain = mom_bu, start_code = 1, end_code = 1, rebal = 2, rebal_mon
= 1);
%ret_cal(datain = mom_sd, start_code = 1, end_code = 1, rebal = 2, rebal_mon
= 1);
%ret_cal(datain = mom_su, start_code = 1, end_code = 1, rebal = 2, rebal_mon
= 1);

/* Consolidate sub-FF and sub-Momentum portfolio */
/* Merge calculation results */
data ff_ret;
      merge ret_ff_bg_1_1 ret_ff_bn_1_1 ret_ff_bv_1_1
            ret_ff_sg_1_1 ret_ff_sn_1_1 ret_ff_sv_1_1;
            /* these datasets are generated by ret_cal macro */
      by year month;
run;

data mom_ret;
      merge ret_mom_bu_1_2 ret_mom_bd_1_2 ret_mom_su_1_2 ret_mom_sd_1_2;
      /* these datasets are generated by ret_cal macro */
      by year month;
run;

/* Calculate SMB, HML, MOM */
data ff_ret; set ff_ret;
      smb = 1/3*(TR_ff_sv+TR_ff_sn+TR_ff_sg)-
1/3*(TR_ff_bv+TR_ff_bn+TR_ff_bg);
      hml = 1/2*(TR_ff_bv+TR_ff_sv)-1/2*(TR_ff_bg+TR_ff_sg);
run;

data mom_ret; set mom_ret;
      mom = 1/2*(TR_mom_bu+TR_mom_su)-1/2*(TR_mom_bd+TR_mom_sd);
run;

data ffm (keep = year month smb hml mom);
      merge ff_ret mom_ret;
      by year month;
run;
```

```
proc export data = ffm outfile = "C:\SAS Data\Output\ffm.csv" DBMS = csv
replace; run;

/* ********************************************************** */

/* calculate alpha using hedge NOA portfolio in Table 2 Part I */
/* get rf from outside */
proc import datafile = "C:\SAS Data\Data\riskfree.csv"
      out = rf DBMS = csv replace;
run;

data rf; set rf;
      year = substr(date,5,6);
      month = substr(date,11,2);
run;

data rf(drop = date year_char month_char);
      set rf(rename = (year = year_char month = month_char));
      year = year_char*1;
      month = month_char*1;
      rf = rf/100;
      if year >= &startyr and year <= &endyr;
run;

/* This macro is to do the regression on CAPM, FF, and 4-factor. It returns
alpha and t-stat */
/* 1 - value weighted; 2 - equal weighted */
%macro factor_cal;
%do iter = 1 %to 2;

data factor_&iter; merge ret_NOA_10_&iter rf; by year month; run;
data factor_&iter; merge factor_&iter ret_universe_1_1; by year month; run;
data factor_&iter; merge factor_&iter ffm; by year month; run;

data factor_&iter; set factor_&iter;
      if missing(leadret1) or missing(adjret1) or missing(adjret2) or
missing(adjret3)
         or missing(rf) or missing(TR_universe) or missing(smb) or
missing(hml)
         or missing(mom) then delete;
run;

/* mktex is the market excessive return */
data factor_&iter; set factor_&iter; mktex = TR_universe-rf;

data alpha_&iter; run;

/* regress CAPM */
proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model leadret1 = mktex;
run; quit; /* stop the regression once it is done to speed up the performance
*/
data alpha_&iter; merge alpha_&iter est(keep = _TYPE_ intercept); rename
intercept = lead1_CAPM; run;

proc reg data = factor_&iter outest = est TABLEOUT noprint;
```

```
      model adjret1 = mktex;
run; quit;
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
adj1_CAPM; run;

proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model adjret2 = mktex;
run; quit;
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
adj2_CAPM; run;

proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model adjret3 = mktex;
run; quit;
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
adj3_CAPM; run;

/* regress FF */
proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model leadret1 = mktex smb hml;
run; quit; /* stop the regression once it is done to speed up the performance
*/
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
lead1_FF; run;

proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model adjret1 = mktex smb hml;
run; quit;
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
adj1_FF; run;

proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model adjret2 = mktex smb hml;
run; quit;
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
adj2_FF; run;

proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model adjret3 = mktex smb hml;
run; quit;
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
adj3_FF; run;

/* regress FF+Mom */
proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model leadret1 = mktex smb hml mom;
run; quit; /* stop the regression once it is done to speed up the performance
*/
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
lead1_FFM; run;

proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model adjret1 = mktex smb hml mom;
run; quit;
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
adj1_FFM; run;
```

```
proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model adjret2 = mktex smb hml mom;
run; quit;
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
adj2_FFM; run;

proc reg data = factor_&iter outest = est TABLEOUT noprint;
      model adjret3 = mktex smb hml mom;
run; quit;
data alpha_&iter; merge alpha_&iter est(keep = intercept); rename intercept =
adj3_FFM; run;

proc export data = alpha_&iter outfile = "C:\SAS
Data\Output\HW4_alpha_&iter..csv" DBMS = csv replace; run;
%end;
%mend;
%factor_cal;

/* Thank you for reading 1104 lines of code */
```