```
libname mylib "c:\analysts";
libname inst "c:\s34";
* merge crsp and ibes;
%let ibes1= mylib.idsum;
%let crsp1= mylib.stocknames;
 /* link by cusip */
proc sort data=&ibes1 out=ibes1 (keep=ticker cusip cname sdates);
  where usfirm=1 and not(missing(cusip));
  by ticker cusip sdates;
run;
 /* create first and last 'start dates' for cusip link */
proc sql;
  create table ibes2
  as select *, min(sdates) as fdate, max(sdates) as ldate
  from ibes1
  group by ticker, cusip
  order by ticker, cusip, sdates;
quit;
 /* label date range variables and keep only most recent company name for cusip link */
data ibes2;
  set ibes2;
  by ticker cusip;
  if last.cusip;
  label fdate="first start date of cusip record";
  label ldate="last start date of cusip record";
  format fdate ldate date9.;
  drop sdates;
run;
 /* crsp: get all permno-ncusip combinations */
proc sort data=&crsp1 out=crsp1 (keep=permno ncusip comnam namedt nameenddt);
  where not missing(ncusip);
  by permno ncusip namedt;
run;
 /* arrange effective dates for cusip link */
proc sql;
  create table crsp2
  as select permno,ncusip,comnam,min(namedt)as namedt,max(nameenddt) as nameenddt
  from crsp1
  group by permno, ncusip
  order by permno, ncusip, namedt;
quit;
 /* label date range variables and keep only most recent company name */
data crsp2;
  set crsp2;
  by permno ncusip;
  if last.ncusip;
  label namedt="start date of cusip record";
  label nameenddt="end date of cusip record";
  format namedt nameenddt date9.;
run;
 /* create cusip link table */
proc sql;
  create table idlink1_1
  as select *
  from ibes2 as a, crsp2 as b
  where a.cusip = b.ncusip
```

```
  order by ticker, permno, ldate;
quit;
 * score links using cusip date range and company name spelling distance;
data idlink1_2;
  set idlink1_1;
  by ticker permno;
  if last.permno; * keep link with most recent company name;
  name_dist = min(spedis(cname,comnam),spedis(comnam,cname));
  if (not ((ldate<namedt) or="" (fdate="">nameenddt))) and name_dist < 30 then score = 0;
    else if (not ((ldate<namedt) or="" (fdate="">nameenddt))) then score = 1;
        else if name_dist < 30 then score = 2;
          else score = 3;
  keep ticker permno cname comnam score;
run;
 *identify remaining unmatched cases;
proc sql;
  create table nomatch1
  as select distinct a.*
  from ibes1 (keep=ticker) as a
  where a.ticker not in (select ticker from idlink1_2)
  order by a.ticker;
quit;
proc sql;
  create table nomatch2
  as select b.ticker, b.cname, b.oftic, b.sdates, b.cusip
  from nomatch1 as a, &ibes1 as b
  where a.ticker = b.ticker and not (missing(b.oftic))
  order by ticker, oftic, sdates;
quit;
*create first and last 'start dates' for exchange tickers;
proc sql;
  create table nomatch3
  as select *, min(sdates) as fdate, max(sdates) as ldate
  from nomatch2
  group by ticker, oftic
  order by ticker, oftic, sdates;
quit;
data nomatch3;
  set nomatch3;
  by ticker oftic;
  if last.oftic;
  label fdate="first start date of record";
  label ldate="last start date of record";
  format fdate ldate date9.;
  drop sdates;
run;
 * get entire list of crsp stocks with exchange ticker information;
proc sort data=&crsp1 out=crsp1 (keep=ticker comnam permno ncusip namedt nameenddt);
  where not missing(ticker);
  by permno ticker namedt;
run;
proc sql;
  create table crsp2
  as select permno,comnam,ticker as crsp_ticker,ncusip,
             min(namedt)as namedt,max(nameenddt) as nameenddt
  from crsp1
```

```
  group by permno, ticker
  order by permno, crsp_ticker, namedt;
quit;
 data crsp2;
  set crsp2;
  if  last.crsp_ticker;
  by permno crsp_ticker;
  label namedt="start date of exch. ticker record";
  label nameenddt="end date of exch. ticker record";
  format namedt nameenddt date9.;
run;
 proc sql;
  create table idlink2_1
  as select a.ticker,a.oftic, b.permno, a.cname, b.comnam, a.cusip, b.ncusip, a.ldate
  from nomatch3 as a, crsp2 as b
  where a.oftic = b.crsp_ticker and
         (ldate>=namedt) and (fdate<=nameenddt)
  order by ticker, oftic, ldate;
quit;
data idlink2_2;
  set idlink2_1;
  name_dist = min(spedis(cname,comnam),spedis(comnam,cname));
  if substr(cusip,1,6)=substr(ncusip,1,6) and name_dist < 30 then score=0;
  else if substr(cusip,1,6)=substr(ncusip,1,6) then score = 4;
  else if name_dist < 30 then score = 5;
      else score = 6;
run;
proc sort data=idlink2_2; by ticker score; run;
data idlink2_3;
  set idlink2_2;
  by ticker score;
  if first.ticker;
 * keep ticker permno cname comnam score;
run;
data mylib.idlink;
  set idlink1_2 idlink2_3;
run;
proc sort data=mylib.idlink; by ticker permno; run;
proc datasets lib=mylib nolist;
            modify idlink (label="ibes-crsp link table");
            label cname = "company name in ibes";
                        label comnam= "company name in crsp";
                        label score= "link score: 0(best) - 6";
            run;
quit;
* merge idlink it with stock level varset which includes permno, fsize, retvol,
yrqtr,# of analyst following and analystid;
proc sql;
  create table all1
  as select a.*, b.*
  from  mylib.varset a left join mylib.idlink b
  on a.permno=b.permno and b.score <= 0;
  order by permno yrqtr;
quit;
%let begdate = 01jan2005;
%let enddate = 31dec2009;
```

```
%let sfilter = (shrcd in (10,11));
%let msfvars = prc ret shrout cfacpr cfacshr;
%let msevars = ncusip exchcd shrcd ;
%INCLUDE ‖c:\stored_macros\crspm.sas=;
Options source2 /*nosource2*/ ORIENTATION=PORTRAIT;
%crspm(s=m,start=&begdate,end=&enddate,sfvars=&msfvars,sevars=&msevars,filters=&sfilter,
outset=mylib.crsp_m);
data crsp_m; format qdate date9.;
set mylib.crsp_m;
qdate = intnx('QTR',date,0,'E');
date = intnx('MONTH',date,0,'E');
p = abs(prc)/cfacpr;
tso=shrout*1000;
if tso<=0 then tso=.;
me = p*tso/1000000;
label p = "price at period end, adjusted";
label tso = "total shares outstanding, adjusted";
label me = "market capitalization, x$1m";
drop ncusip prc cfacpr shrout exchcd shrcd ret;
format ret percentn8.4 me p dollar12.3 tso comma12.;
run;
data crsp_m;
set crsp_m;
by permno qdate date;
if last.qdate;
drop date;
run;
 proc sql;
create table first_vint
as select distinct rdate, fdate, mgrno, mgrname
from inst.s34type1
group by mgrno, rdate
having fdate=min(fdate)
order by mgrno, rdate;
quit;
 data first_vint;
set first_vint;
by mgrno rdate;
length first_report 3;
first_report = (first.mgrno or intck('QTR',lag(rdate),rdate)>1);
run;
 proc sort data=first_vint nodupkey; by mgrno descending rdate; run;
data first_vint;
set first_vint;
by mgrno descending rdate;
length last_report 3;
last_report = (first.mgrno or intck('QTR',rdate,lag(rdate))>1);
if ("&begdate"d <= rdate <="&enddate"d);
run;
proc sql undo_policy=none;
create table first_vint
as select distinct *, count(mgrno) as numinst
from first_vint
group by rdate
order by fdate, mgrno;
quit;
```

```
 data holdings_v1 / view=holdings_v1;
merge first_vint(in=a drop=mgrname)
  inst.s34type3(in=b drop=type sole shared no);
by fdate mgrno;
if a and b and shares>0;
run;
/* Map 13F's Historical CUSIP to CRSP Unique Identifier PERMNO */
proc sql;
create view holdings_v2 as
select distinct a.rdate, a.fdate, a.mgrno, a.numinst,
        a.first_report, a.last_report, b.permno, a.shares
from holdings_v1 as a,
   (select distinct ncusip, permno from crsp.msenames
    where not missing(ncusip)) as b
    where a.cusip=b.ncusip;
quit;
proc sql;
create table holdings as
select distinct a.rdate, a.mgrno, a.numinst, a.first_report, a.last_report,
      a.permno, a.shares as shares_adj label = "adjusted shares held"
from holdings_v2 as a, crsp_m as b
where a.permno=b.permno and a.fdate = b.qdate;
quit;
proc sort data=holdings nodupkey; by permno rdate mgrno; run;
proc sort data=crsp_m   nodupkey; by permno qdate;       run;
proc means data=holdings noprint;
where shares_adj>0;
by permno rdate;
var shares_adj first_report;
output out=io_metrics (drop=_freq_ _type_)
       n=numowners max(numinst)=numinst
       sum(shares_adj)=io_total ;
run;
data io;
merge io_metrics(in=a) crsp_m (in=b rename=(qdate=rdate));
by permno rdate;
if b and tso>0;
ior = io_total/tso;
if missing(ior) then ior=0;
format ior  percentn8.2;
year=year(rdate); qtr=qtr(rdate);
yrqtr=year || '.' || qtr;
if ior>1 then delete;
run;
%MACRO NWORDS (INVAR);
%local N W;
%let N = 0;
%let W = 1;
%do %while (%nrquote(%scan(&invar,&W,%str( ))) ^= %str());
  %let N = %eval(&N+1);
  %let W = %eval(&W+1);
%end;
&N
%MEND NWORDS;
%MACRO WIN (INSET=,OUTSET=,SORTVAR=,VARS=,PERC1=1,TRIM=0);
/* List of all variables */
```

```
%let vars = %sysfunc(compbl(&vars));
%let nvars = %nwords(&vars);
/* Display Output */
%put ### START.;
/* Trimming / Winsorization Options */
%if &trim=0 %then %put ### Winsorization; %else %put ### Trimming;
%put ### Number of Variables:  &nvars;
%put ### List   of Variables:  &vars;
options nonotes;
/* Ranking within &sortvar levels */
%put ### Sorting... ;
proc sort data=&inset; by &sortvar; run;
/* 2-tail winsorization/trimming */
%let perc2 = %eval(100-&perc1);
%let var2 = %sysfunc(tranwrd(&vars,%str( ),%str(__ )))__;
%let var_p1 = %sysfunc(tranwrd(&vars,%str( ),%str(__&perc1 )))__&perc1 ;
%let var_p2 = %sysfunc(tranwrd(&vars,%str( ),%str(__&perc2 )))__&perc2 ;
/* Calculate upper and lower percentiles */
proc univariate data=&inset noprint;
by &sortvar;
var &vars;
output out=_perc pctlpts=&perc1 &perc2 pctlpre=&var2;
run;
%if &trim=1 %then
%let condition = %str(if myvars(i)>=perct2(i) or myvars(i)<=perct1(i) then myvars(i)=. );
%else %let condition = %str(myvars(i)=min(perct2(i),max(perct1(i),myvars(i))) );
%if &trim=0 %then %put ### Winsorizing at &perc1.%... ;
%else %put ### Trimming at &perc1.%... ;
/* Save output with trimmed/winsorized variables */
data &outset;
merge &inset (in=a) _perc;
by &sortvar;
if a;
array myvars {&nvars} &vars;
array perct1 {&nvars} &var_p1;
array perct2 {&nvars} &var_p2;
do i = 1 to &nvars;
    if not missing(myvars(i)) then
    do;
    &condition;
    end;
end;
drop i &var_p1 &var_p2;
run;

/* House Cleaning */
proc sql; drop table _perc; quit;
options notes;
%put ### DONE . ; %put ;
%MEND WIN;
%win(inset=mylib.mycompq, outset=mycompqw, sortvar=yrqtr, vars=atq ltq ceqq, PERC1=1, trim=0 );
*also tried %win with (PERC1=0) and with sortvar=year in separately. results are robust;
%let start=2005;
%let end=2009;
    data mycompqw;
    set mycompqw;
```

```
    if &start<=fyear<=&end;
    *no double records;
    if indfmt='INDL' and datafmt='STD' and popsrc='D' and consol='C' ;
    run;
    /* match it with gvkey-permno identifier link*/
proc sql;
      create table all2 as
      select a.*, b.lpermno as permno
      from mycompqw a left join mylib.gvkeyperm b
        on a.gvkey=b.gvkey
        and b.lpermno ne .
        and b.linktype in ("LC" "LN" "LU" "LX" "LD" "LS")
        and b.linkprim in ("C", "P")
        and ((a.datadate >= b.linkdt) or b.linkdt eq .B) and
           ((a.datadate <= b.linkenddt) or b.linkenddt eq .E)   ;
quit;
    * merge it with fsize, volatility,
# of analyst following, and analystid;
proc sql;
        create table all3 as
        select a.*, b.*,(atq-ltq)/ceqq as bm
        from all2 as a, all1 as b
        where a.permno = b.permno and a.yrqtr=b.yrqtr and 2005.1<=yrqtr<=2009.4;
quit;
proc sql;
  create table all4
  as select a.*, b.ior
  from all3 a, io b
  where a.permno=b.permno and a.yrqtr=b.yrqtr  ;
quit;
* merge crsp/comp/13f final data with the set which
includes analyst characteristics,
chpi and housing controls for each analyst;
proc sql;
  create table all5
  as select a.*, b.chpi, b.errors, b.bias, b.nrevs, b.horizon, b.star, b.portsize, b.ivy, b.tenure,
b.exp, b.cohort, b.firstemp, b.analystzip, b.empid, b.zipyrqtr, b.empidyrqtr, b.empcbsa,
b.firmyrqtr, b.problayoff, b.owner, b.recourse, b.cltv90, b.equity, b.foreclosure
from all4 a, mylib.analysts b
  where a.analystid=b.analystid and a.yrqtr=b.yrqtr  ;
quit;
option nocenter nodate nonumber ps=max ls=max;
*macro for regressions with fixed effects and double clustered standard errors;
%macro clstr(data=,dependent=,var=,fe=,class=, format=,
cluster=,endogenous=,instruments=,weight=,multi=,threshold=0.000000001,maxiter=100, n
estedness=yes,deldata=yes);


    %if &fe=%str() and &instruments=%str() %then %do;
        %clust3(data=&data,dependent=&dependent, var=&var, class=&class,format=&format,
                        cluster=&cluster,weight=&weight,
                        multi=&multi, output=&data._est_cl);
        %goto myend;
    %end;

    %if &class ne %str() or &format ne %str() %then %do;
        %put %str(class variables (and their formats) not yet
```

```
available with fixed effects or instrumental variables ! create dummies !);
        %goto myend;
    %end;

    %let methodnes1=%str();
    %let methodnes2=%str();

    %if &fe ne %str() and &cluster ne %str() %then %do;
        %let methodnes1=correction for nestedness of the 3 first fixed effects in
the 3 first clusters;
        %let methodnes2=no correction for nestedness;
    %end;

    %let within=%str();
    %if &fe ne %str() %then %do;
        %let demean=_demean;
        %let within=within;
    %end;
    %else %let demean=_clean;

    %let fe1=%scan(&fe,1);
    %let fe2=%scan(&fe,2);
    %let fe3=%scan(&fe,3);
    %let fe4=%scan(&fe,4);

    %let fe_other=%str();
    %if &fe4 ne %str() %then %let fe_other=%substr(&fe,%index(&fe,&fe4),
%length(&fe-%index(&fe,&fe4)));

    %let fe1_det=%str();
    %let fe2_det=%str();
    %let fe3_det=%str();

    %let cluster1=%scan(&cluster,1);
    %let cluster2=%scan(&cluster,2);
    %let cluster3=%scan(&cluster,3);

    %let cluster1_det=%str();
    %let cluster2_det=%str();
    %let cluster3_det=%str();

    %let n_fe1=0;
    %let n_fe2=0;
    %let n_fe3=0;
    %let df_fe_all=0;

    %let n_fe1_cl1=0;
    %let n_fe1_cl2=0;
    %let n_fe1_cl3=0;

    %let n_fe2_cl1=0;
    %let n_fe2_cl2=0;
    %let n_fe2_cl3=0;

    %let n_fe3_cl1=0;
    %let n_fe3_cl2=0;
```

```
    %let n_fe3_cl3=0;

    %let r2=%str();
    %let warning=%str();
    %let print=yes;

    %if &instruments=%str() %then %let model=ols;
    %else %let model=iv;

    %if &fe=%str() %then %let model_fe=%str();
    %else %let model_fe=fixed effects;

    %if &cluster=%str() %then %let model_cl=%str();
    %else %let model_cl=cluster;


*subprogram for using list of variables type v1-v10 and aa--zz;
    %let hypen1=%str( -);
    %do %while (%index(&instruments,&hypen1)>0);
        %let length=%index(&instruments,&hypen1);
        %let instruments=%qsubstr(&instruments,1,%eval(&length-1))%str(-)
%qsubstr(&instruments,%eval(&length+2));
    %end;
    %do %while (%index(&var,&hypen1)>0);
        %let length=%index(&var,&hypen1);
        %let var=%qsubstr(&var,1,%eval(&length-1))%str(-)
%qsubstr(&var,%eval(&length+2));
    %end;


    %let hypen2=%str(- );
    %do %while (%index(&instruments,&hypen2)>0);
        %let length=%index(&instruments,&hypen2);
        %let instruments=%qsubstr(&instruments,1,%eval(&length-1))
%str(-)%qsubstr(&instruments,&length+2);
    %end;
    %do %while (%index(&var,&hypen2)>0);
        %let length=%index(&var,&hypen2);
        %let var=%qsubstr(&var,1,%eval(&length-1))%str(-)%qsubstr(&var,&length+2);
    %end;

    %if %index(&var,--)>0 or %index(&instruments,--)>0  %then %do;
        proc contents data=&data noprint; run;
    %end;
    %do %while (%index(&instruments,--)>0);
        %let i=1;
        %do %while(%index(%scan(&instruments,&i,%str( )),--)=0);
            %let i=%eval(&i+1);
        %end;
        %let length2=%length(%scan(&instruments,&i,%str( )));
        %let vd=%scan(&instruments,&i);
        %let length=%index(&instruments,&vd);
        %let vf=%scan(&instruments,&i+1);
        %local z; %local zz;
        %let z=%sysfunc(open(&data));
            %let posd=%sysfunc(varnum(&z,&vd));
```

```
            %let posf=%sysfunc(varnum(&z,&vf));
            %let j=%eval(&posd+1);
            %let white=%str( );
            %let allvar=&vd&white;
            %do %while (&j<&posf);
                %let vj=%sysfunc(varname(&z,&j));
                %let allvar=&allvar&vj&white;
                %let j=%eval(&j+1);
            %end;
            %let allvar=&allvar&vf;
        %let zz=%sysfunc(close(&z));
        %let instruments=%qsubstr(&instruments,1,%eval(&length-1))
%str(&allvar)%qsubstr(&instruments,%eval(&length+&length2));
    %end;

    %do %while (%index(&instruments,-)>0);
        %let i=1;
        %do %while(%index(%scan(&instruments,&i,%str( )),-)=0);
            %let i=%eval(&i+1);
        %end;
        %let length2=%length(%scan(&instruments,&i,%str( )));
        %let vd=%scan(&instruments,&i,%str( -));
        %let length=%index(&instruments,&vd);
        %let vf=%scan(&instruments,&i+1,%str( -));
        %let k=%length(&vd);
        %do %while(%verify(%substr(&vd,&k),0123456789)=0);
            %let posd=%substr(&vd,&k);
            %let k=%eval(&k-1);
        %end;
        %let k=%length(&vf);
        %do %while(%verify(%substr(&vf,&k),0123456789)=0);
            %let posf=%substr(&vf,&k);
            %let k=%eval(&k-1);
        %end;
            %let j=%eval(&posd+1);
            %let white=%str( );
            %let allvar=&vd&white;
            %do %while (&j<&posf);
                %let vj=%substr(&vd,1,&k);
                %let allvar=&allvar&vj&j&white;
                %let j=%eval(&j+1);
            %end;
            %let allvar=&allvar&vf;
        %let instruments=%qsubstr(&instruments,1,%eval(&length-1))%str(&allvar)
%qsubstr(&instruments,%eval(&length+&length2));
    %end;

    %do %while (%index(&var,--)>0);
        %let i=1;
        %do %while(%index(%scan(&var,&i,%str( )),--)=0);
            %let i=%eval(&i+1);
        %end;
        %let length2=%length(%scan(&var,&i,%str( )));
        %let vd=%scan(&var,&i);
        %let length=%index(&var,&vd);
        %let vf=%scan(&var,&i+1);
```

```
        %local z; %local zz;
        %let z=%sysfunc(open(&data));
            %let posd=%sysfunc(varnum(&z,&vd));
            %let posf=%sysfunc(varnum(&z,&vf));
            %let j=%eval(&posd+1);
            %let white=%str( );
            %let allvar=&vd&white;
            %do %while (&j<&posf);
                %let vj=%sysfunc(varname(&z,&j));
                %let allvar=&allvar&vj&white;
                %let j=%eval(&j+1);
            %end;
            %let allvar=&allvar&vf;
        %let zz=%sysfunc(close(&z));
        %let var=%qsubstr(&var,1,%eval(&length-1))%str(&allvar)
%qsubstr(&var,%eval(&length+&length2));
    %end;

    %do %while (%index(&var,-)>0);
        %let i=1;
        %do %while(%index(%scan(&var,&i,%str( )),-)=0);
            %let i=%eval(&i+1);
        %end;
        %let length2=%length(%scan(&var,&i,%str( )));
        %let vd=%scan(&var,&i,%str( -));
        %let length=%index(&var,&vd);
        %let vf=%scan(&var,&i+1,%str( -));
        %let k=%length(&vd);
        %do %while(%verify(%substr(&vd,&k),0123456789)=0);
            %let posd=%substr(&vd,&k);
            %let k=%eval(&k-1);
        %end;
        %let k=%length(&vf);
        %do %while(%verify(%substr(&vf,&k),0123456789)=0);
            %let posf=%substr(&vf,&k);
            %let k=%eval(&k-1);
        %end;
            %let j=%eval(&posd+1);
            %let white=%str( );
            %let allvar=&vd&white;
            %do %while (&j<&posf);
                %let vj=%substr(&vd,1,&k);
                %let allvar=&allvar&vj&j&white;
                %let j=%eval(&j+1);
            %end;
            %let allvar=&allvar&vf;
        %let var=%qsubstr(&var,1,%eval(&length-1))%str(&allvar)
%qsubstr(&var,%eval(&length+&length2));
    %end;
    *end of subprogram for using list of variables type v1-v10 and aa--zz;

    %let var_c = %sysfunc(tranwrd(&var,*,_x_));
    %let var_d = %sysfunc(tranwrd(&var,*,%str( )));

    %let instruments_c = %sysfunc(tranwrd(&instruments,*,_x_));
    %let instruments_d = %sysfunc(tranwrd(&instruments,*,%str( )));
```

```
%let endogenous_c = %sysfunc(tranwrd(&endogenous,*,_x_));
%let endogenous_d = %sysfunc(tranwrd(&endogenous,*,%str( )));


%if &fe ne %str() %then %do;
    data &data._tmp1;
    %let feinfo=fixed effects (by &fe) and;
    %let noint=noint;
%end;

%else %do;
    data &data.&demean;
    %let feinfo=%str();
    %let noint=%str();

%end;
        set &data (keep=&fe &cluster &dependent &var_d &instruments_d &weight &endogenous_d);
        if &dependent ne .;
        %let i=1;
        %do %while(%scan(&var_c,&i,%str( )) ne);
            %let v_a=%scan(&var_c,&i,%str( ));
            %put %str(&v_a);
            %if %index(&v_a,_x_)>0 %then %do;
                %let v_b=%sysfunc(tranwrd(&v_a,_x_,*));
                &v_a=&v_b;
                %put %str(&v_a);
                %put %str(&v_b);
                if missing(&v_a)=0;
            %end;
            %else %do;
                if missing(%scan(&var,&i))=0;
            %end;
            %if &weight ne %str() %then %do;
                if missing(&weight)=0;
            %end;
            %let i=%eval(&i+1);
        %end;

        %let i=1;
        %do %while(%scan(&instruments_c,&i,%str( )) ne);
            %let v_a=%scan(&instruments_c,&i,%str( ));
            %put %str(&v_a);
            %if %index(&v_a,_x_)>0 %then %do;
                %let v_b=%sysfunc(tranwrd(&v_a,_x_,*));
                &v_a=&v_b;
                %put %str(&v_a);
                %put %str(&v_b);
                if missing(&v_a)=0;
            %end;
            %else %do;
                if missing(%scan(&instruments,&i))=0;
            %end;
            %let i=%eval(&i+1);
        %end;
```

```
        %let i=1;
        %do %while(%scan(&endogenous_c,&i,%str( )) ne);
            %let v_a=%scan(&endogenous_c,&i,%str( ));
            %put %str(&v_a);
            %if %index(&v_a,_x_)>0 %then %do;
                %let v_b=%sysfunc(tranwrd(&v_a,_x_,*));
                &v_a=&v_b;
                %put %str(&v_a);
                %put %str(&v_b);
                if missing(&v_a)=0;
            %end;
            %else %do;
                if missing(%scan(&endogenous,&i))=0;
            %end;
            %let i=%eval(&i+1);
        %end;

    %if &fe ne %str() %then %do;
        %let i=1;
        %do %while(%scan(&fe,&i) ne);
            if missing(%scan(&fe,&i))=0;
            %let i=%eval(&i+1);
        %end;
    %end;

    %if &cluster ne %str() %then %do;
        %let i=1;
        %do %while(%scan(&cluster,&i) ne);
            if missing(%scan(&cluster,&i))=0;
            %let i=%eval(&i+1);
        %end;
    %end;
        mykey=_n_;
    run;

%let var=&var_c;
%let instruments=&instruments_c;
%let endogenous=&endogenous_c;

    title "descriptives variables (rough)";
    proc means ;
        var &dependent &var &endogenous &instruments &weight;
        output out=&data._des;
        %if &weight ne %str() %then %do;
            weight &weight;
        %end;
    run;

*multiway demeaning with method of alternate projections;
%if &fe ne %str() %then %do;
    %let i=1;
    %do %while(%scan(&fe,&i) ne);
        proc sort data=&data._tmp&i;
            by %scan(&fe,&i);
        run;
```

```
        %let j=%eval(&i+1);

        proc standard data=&data._tmp&i out=&data._tmp&j mean=0;
            var &dependent &var &endogenous &instruments ;
            by %scan(&fe,&i);
        run;
        %let i=%eval(&i+1);
    %end;


    %if &i>2 %then %do;
        options nonotes;
        %let gap=100;
        %let round=0;
        %let gap=%eval(&gap);

        %do %while( &gap ne 0 and &round<&maxiter);

            data &data._tmp1; set &data._tmp&j;
            run;
            %let i=1;
            %do %while(%scan(&fe,&i) ne);
                proc sort data=&data._tmp&i;
                    by %scan(&fe,&i);
                run;

                %let j=%eval(&i+1);
                proc standard data=&data._tmp&i out=&data._tmp&j mean=0;
                    var &dependent &var &endogenous &instruments ;
                    by %scan(&fe,&i);
                run;
                %let i=%eval(&i+1);
            %end;

            proc sort data=&data._tmp&j;
                by %scan(&fe,1);
            run;

            proc compare base=&data._tmp&j compare=&data._tmp1 out=&data._tmp_comp noprint;
                var &dependent &var &endogenous &instruments  ;
            run;

            proc means data=&data._tmp_comp noprint;
                var &dependent &var &endogenous &instruments ;
                output out=_doctmp_comp2;
            run;

            data _null_;
                set _doctmp_comp2 (keep=&dependent &var
&instruments _stat_ where=(_stat_ in ("max"))) end=last;
                compmax=max(of &dependent &var &endogenous &instruments ) ;
                if last then do;
                call symput ('compmax',compmax);
                end;
            run;
```

```
                data _null_;
                    set _doctmp_comp2 (keep=&dependent &var
&instruments _stat_ where=(_stat_ in ("min"))) end=last;
                    compmin=min(of &dependent &var &endogenous &instruments );
                    if last then do;
                    call symput ('compmin',compmin);
                    end;
                run;

                data _null_; set _doctmp_comp2 (obs=1);
                    gap=round(%sysevalf(&compmax)-%sysevalf(&compmin),&threshold) ;
                    call symput ('gap',gap);
                run;


                %let round=%eval(&round+1);
                %let gap=%sysevalf(&gap);
                %let gap2=%sysevalf(&compmax-&compmin);
                 %put %str(round &round. gap:  &gap2);
            %end;
            options notes;
            %if &round=&maxiter %then %let warning=!alternating projections demeaning
did not converge in &round iterations. beware!;
            %put %str(&warning);



        %end;

        proc datasets library=work nolist nodetails;
            delete &data.&demean;
        quit;

        proc datasets library=work nolist;
            change &data._tmp&j=&data.&demean;
        quit;

        title "descriptives variables (demeaned by &fe)";
        proc means data=&data.&demean;
            var &dependent &var &endogenous &instruments ;
            output out=&data._des_within;
            %if &weight ne %str() %then %do;
                weight &weight;
            %end;
        run;

        *nestedness of fe in clusters;

        %let i=1;
        %do %while (%scan(&fe,&i) ne %str());
            proc freq data=&data.&demean noprint;
                tables %scan(&fe,&i) / out=&data._tmp_n_fe&i;
            run;

            data _null_;
                set &data._tmp_n_fe&i;
```

```
            count=_n_;
            call symput ("n_fe&i",count);
        run;


        %if %scan(&cluster,1) ne %str() %then %do;
            proc freq data=&data.&demean noprint;
                tables %scan(&fe,&i)*%scan(&cluster,1) / out=&data._tmp_n_fe&i._cl1 list;
            run;

            data _null_;
                set &data._tmp_n_fe&i._cl1;
                count=_n_;
                call symput ("n_fe&i._cl1",count);
            run;
            %if %scan(&cluster,2) ne %str() %then %do;
                proc freq data=&data.&demean noprint;
                    tables %scan(&fe,&i)*%scan(&cluster,2) / out=&data._tmp_n_fe&i._cl2 list;
                run;

                data _null_;
                    set &data._tmp_n_fe&i._cl2;
                    count=_n_;
                    call symput ("n_fe&i._cl2",count);
                run;
                %if %scan(&cluster,3) ne %str() %then %do;
                    proc freq data=&data.&demean noprint;
                        tables %scan(&fe,&i)*%scan(&cluster,3) /
out=&data._tmp_n_fe&i._cl3 list;
                    run;

                    data _null_;
                        set &data._tmp_n_fe&i._cl3;
                        count=_n_;
                        call symput ("n_fe&i._cl3",count);
                    run;
                %end;
            %end;
        %end;


        %let df_fe_all=%eval(&df_fe_all-1+&n_fe&i);
        %let i=%eval(&i+1);
    %end;
    %let df_fe_all=%eval(&df_fe_all+1);
%end;

%else %do;
    proc sort data=&data.&demean;
        by %scan(&cluster,1);
    run;
%end;

%if &instruments =%str() %then %do;
    %let print=no;
    %clust3(data=&data.&demean,dependent=&dependent, var=&var,
```

```
                    cluster=&cluster,weight=&weight,
                    multi=&multi, output=&data._est_cl,noint=&noint,print=&print);
        data &data._est_cl;
            set &data._est_cl;

            n_fe1=&n_fe1;
            n_fe2=&n_fe2;
            n_fe3=&n_fe3;

            df_fe_all=&df_fe_all;

            n_fe1_cl1=&n_fe1_cl1;
            n_fe1_cl2=&n_fe1_cl2;
            n_fe1_cl3=&n_fe1_cl3;

            n_fe2_cl1=&n_fe2_cl1;
            n_fe2_cl2=&n_fe2_cl2;
            n_fe2_cl3=&n_fe2_cl3;

            n_fe3_cl1=&n_fe3_cl1;
            n_fe3_cl2=&n_fe3_cl2;
            n_fe3_cl3=&n_fe3_cl3;

            *method 1 : robust cluster standard error corrected for fixed effect nestedness;
            stderr_cl1=stderr_or_cl1*(((n-k)/(n-k-df_fe_all+(n_fe1_cl1=n_fe1)*max(n_fe1-1,0)
+(n_fe2_cl1=n_fe2)*max(n_fe2-1,0)+(n_fe3_cl1=n_fe3)*max(n_fe3-1,0)))**0.5);
            %if %scan(&cluster,2) ne %str() %then %do;
                stderr_cl2=stderr_or_cl2*(((n-k)/(n-k-df_fe_all+(n_fe1_cl2=n_fe1)*max(n_fe1-1,0)
+(n_fe2_cl2=n_fe2)*max(n_fe2-1,0)+(n_fe3_cl2=n_fe3)*max(n_fe3-1,0)))**0.5);
            %end;

            %if %scan(&cluster,3) ne %str() %then %do;
                stderr_cl3=stderr_or_cl3*(((n-k)/(n-k-df_fe_all+(n_fe1_cl3=n_fe1)*max(n_fe1-1,0)
+(n_fe2_cl3=n_fe2)*max(n_fe2-1,0)+(n_fe3_cl3=n_fe3)*max(n_fe3-1,0)))**0.5);

                stderr_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
                stderr_cl1_cl3=stderr_or_cl1_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
                stderr_cl2_cl3=stderr_or_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
                stderr_cl1_cl2_cl3=stderr_or_cl1_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);

                stderr=(stderr_cl1**2+stderr_cl2**2+stderr_cl3**2-stderr_cl1_cl2**2
-stderr_cl2_cl3**2-stderr_cl1_cl3**2+stderr_cl1_cl2_cl3**2)**0.5;
                dendf=min(dendf_cl1,dendf_cl2,dendf_cl3);
            %end;
            %else %if %scan(&cluster,2) ne %str() %then %do;
                stderr_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
                stderr=(stderr_cl1**2+stderr_cl2**2-stderr_cl1_cl2**2)**0.5;
                dendf=min(dendf_cl1,dendf_cl2);
            %end;
            %else %do;
                stderr=stderr_cl1;
                dendf=dendf_cl1;
            %end;

            t=estimate/stderr;
            pvalue=(1-probt(abs(t),dendf))*2;
```

```
        if pvalue ne . then do;
                if pvalue <0.001    then stars="***";
                else if pvalue <0.01 then stars="** ";
                else if pvalue <0.05 then stars="*  ";
                else if pvalue <0.10 then stars=".  ";
        end;

        *method 2 : robust cluster standard error uncorrected for fixed effect nestedness;
        stderr2_cl1=stderr_or_cl1*(((n-k)/(n-k-df_fe_all))**0.5);
        %if %scan(&cluster,2) ne %str() %then %do;
            stderr2_cl2=stderr_or_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
        %end;

        %if %scan(&cluster,3) ne %str() %then %do;
            stderr2_cl3=stderr_or_cl3*(((n-k)/(n-k-df_fe_all))**0.5);

            stderr2_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
            stderr2_cl1_cl3=stderr_or_cl1_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
            stderr2_cl2_cl3=stderr_or_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
            stderr2_cl1_cl2_cl3=stderr_or_cl1_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);

            stderr2=(stderr2_cl1**2+stderr2_cl2**2+stderr2_cl3**2-stderr2_cl1_cl2**2
-stderr2_cl2_cl3**2-stderr2_cl1_cl3**2+stderr2_cl1_cl2_cl3**2)**0.5;
        %end;
        %else %if %scan(&cluster,2) ne %str() %then %do;
            stderr2_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
            stderr2=(stderr2_cl1**2+stderr2_cl2**2-stderr2_cl1_cl2**2)**0.5;
        %end;
        %else %do;
            stderr2=stderr2_cl1;
        %end;

        t2=estimate/stderr2;
        dendf2=n-k-df_fe_all;
        pvalue2=(1-probt(abs(t2),dendf2))*2;
        if pvalue2 ne . then do;
                if pvalue2 <0.001   then stars2="***";
                else if pvalue2 <0.01 then stars2="** ";
                else if pvalue2 <0.05 then stars2="*  ";
                else if pvalue2 <0.10 then stars2=".  ";
        end;
        call symput ('dendf2',dendf2);

        call symput ('dendf',dendf);
        call symput ('n',n);
        call symput ('g_cl1',dendf_cl1+1);
        call symput ('g_cl2',dendf_cl2+1);
        call symput ('g_cl3',dendf_cl3+1);

    run;

    %if &n_fe1>0 %then %let fe1_det=(n1=%scan(&n_fe1,1));
    %if &n_fe2>0 %then %let fe2_det=(n2=%scan(&n_fe2,1));
    %if &n_fe3>0 %then %let fe3_det=(n3=%scan(&n_fe3,1));
```

```
        %if &g_cl1>0 %then %let cluster1_det=(g1=%scan(&g_cl1,1));
        %if &g_cl2>0 %then %let cluster2_det=(g2=%scan(&g_cl2,1));
        %if &g_cl3>0 %then %let cluster3_det=(g3=%scan(&g_cl3,1));

        data _null_; set &data.&demean._tmp_fit_cl1 (obs=1);
            call symput ('r2',cvalue1);
        run;

        %if &nestedness ne no %then %do;
            proc print data=&data._est_cl;
                var parameter--estimate stderr stars pvalue ;
            run;
        %end;

        %if &nestedness ne yes %then %do;

            proc print data=&data._est_cl;
                var parameter--estimate stderr2 stars2 pvalue2 ;
            run;
        %end;

    %end;
    %else %if &instruments ne %str()  %then %do;
        %let j=1;
        %do %while (%scan(&endogenous,&j) ne %str());
            %let print=no;
            %clust3(data=&data.&demean,dependent=%scan(&endogenous,&j), var=&instruments &var,
cluster=&cluster, weight=&weight,
                multi=&multi, output=&data._est_cl_step1_&j,noint=&noint,print=&print);
            data &data._est_cl_step1_&j;
                set &data._est_cl_step1_&j;
                n_fe1=&n_fe1;
                n_fe2=&n_fe2;
                n_fe3=&n_fe3;

                df_fe_all=&df_fe_all;

                n_fe1_cl1=&n_fe1_cl1;
                n_fe1_cl2=&n_fe1_cl2;
                n_fe1_cl3=&n_fe1_cl3;

                n_fe2_cl1=&n_fe2_cl1;
                n_fe2_cl2=&n_fe2_cl2;
                n_fe2_cl3=&n_fe2_cl3;

                n_fe3_cl1=&n_fe3_cl1;
                n_fe3_cl2=&n_fe3_cl2;
                n_fe3_cl3=&n_fe3_cl3;

                *method 1 : robust cluster standard error corrected for fixed effect nestedness;
                stderr_cl1=stderr_or_cl1*(((n-k)/(n-k-df_fe_all+(n_fe1_cl1=n_fe1)*max(n_fe1-1,0)
+(n_fe2_cl1=n_fe2)*max(n_fe2-1,0)+(n_fe3_cl1=n_fe3)*max(n_fe3-1,0)))**0.5);
                %if %scan(&cluster,2) ne %str() %then %do;
                    stderr_cl2=stderr_or_cl2*(((n-k)/(n-k-df_fe_all+(n_fe1_cl2=n_fe1)*max(n_fe1-1,0)
+(n_fe2_cl2=n_fe2)*max(n_fe2-1,0)+(n_fe3_cl2=n_fe3)*max(n_fe3-1,0)))**0.5);
                %end;
```

```
                %if %scan(&cluster,3) ne %str() %then %do;
                    stderr_cl3=stderr_or_cl3*(((n-k)/(n-k-df_fe_all+(n_fe1_cl3=n_fe1)*max(n_fe1-1,0)
+(n_fe2_cl3=n_fe2)*max(n_fe2-1,0)+(n_fe3_cl3=n_fe3)*max(n_fe3-1,0)))**0.5);

                    stderr_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
                    stderr_cl1_cl3=stderr_or_cl1_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
                    stderr_cl2_cl3=stderr_or_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
                    stderr_cl1_cl2_cl3=stderr_or_cl1_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);

                    stderr=(stderr_cl1**2+stderr_cl2**2+stderr_cl3**2-stderr_cl1_cl2**2
-stderr_cl2_cl3**2-stderr_cl1_cl3**2+stderr_cl1_cl2_cl3**2)**0.5;
                    dendf=min(dendf_cl1,dendf_cl2,dendf_cl3);
                %end;
                %else %if %scan(&cluster,2) ne %str() %then %do;
                    stderr_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
                    stderr=(stderr_cl1**2+stderr_cl2**2-stderr_cl1_cl2**2)**0.5;
                    dendf=min(dendf_cl1,dendf_cl2);
                %end;
                %else %do;
                    stderr=stderr_cl1;
                    dendf=dendf_cl1;
                %end;

                t=estimate/stderr;
                pvalue=(1-probt(abs(t),dendf))*2;

                if pvalue ne . then do;
                        if pvalue <0.001    then stars="***";
                        else if pvalue <0.01 then stars="** ";
                        else if pvalue <0.05 then stars="*  ";
                        else if pvalue <0.10 then stars=".  ";
                end;

                *method 2 : robust cluster standard error uncorrected for fixed effect nestedness;
                stderr2_cl1=stderr_or_cl1*(((n-k)/(n-k-df_fe_all))**0.5);
                %if %scan(&cluster,2) ne %str() %then %do;
                    stderr2_cl2=stderr_or_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
                %end;

                %if %scan(&cluster,3) ne %str() %then %do;
                    stderr2_cl3=stderr_or_cl3*(((n-k)/(n-k-df_fe_all))**0.5);

                    stderr2_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
                    stderr2_cl1_cl3=stderr_or_cl1_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
                    stderr2_cl2_cl3=stderr_or_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
                    stderr2_cl1_cl2_cl3=stderr_or_cl1_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);

                    stderr2=(stderr2_cl1**2+stderr2_cl2**2+stderr2_cl3**2-stderr2_cl1_cl2**2
-stderr2_cl2_cl3**2-stderr2_cl1_cl3**2+stderr2_cl1_cl2_cl3**2)**0.5;
                %end;
                %else %if %scan(&cluster,2) ne %str() %then %do;
                    stderr2_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
                    stderr2=(stderr2_cl1**2+stderr2_cl2**2-stderr2_cl1_cl2**2)**0.5;
                %end;
                %else %do;
```

```
        stderr2=stderr2_cl1;
    %end;

    t2=estimate/stderr2;
    dendf2=n-k-df_fe_all;
    pvalue2=(1-probt(abs(t2),dendf2))*2;
    if pvalue2 ne . then do;
            if pvalue2 <0.001   then stars2="***";
            else if pvalue2 <0.01 then stars2="** ";
            else if pvalue2 <0.05 then stars2="*  ";
            else if pvalue2 <0.10 then stars2=".  ";
    end;
    call symput ('dendf2',dendf2);

    call symput ('dendf',dendf);
    call symput ('n',n);
    call symput ('g_cl1',dendf_cl1+1);
    call symput ('g_cl2',dendf_cl2+1);
    call symput ('g_cl3',dendf_cl3+1);
run;


%if &n_fe1>0 %then %let fe1_det=(n1=%scan(&n_fe1,1));
%if &n_fe2>0 %then %let fe2_det=(n2=%scan(&n_fe2,1));
%if &n_fe3>0 %then %let fe3_det=(n3=%scan(&n_fe3,1));

%if &g_cl1>0 %then %let cluster1_det=(g1=%scan(&g_cl1,1));
%if &g_cl2>0 %then %let cluster2_det=(g2=%scan(&g_cl2,1));
%if &g_cl3>0 %then %let cluster3_det=(g3=%scan(&g_cl3,1));

data _null_; set &data.&demean._tmp_fit_cl1 (obs=1);
    call symput ('r2',cvalue1);
run;
%if &nestedness ne no %then %do;
                    proc print data=&data._est_cl_step1_&j;
        var parameter--estimate stderr stars pvalue ;
    run;
    title;
%end;
%if &nestedness ne yes %then %do;

    proc print data=&data._est_cl_step1_&j;
        var parameter--estimate stderr2 stars2 pvalue2 ;
    run;
%end;

*1.2 reestimate the first stage using prog reg to save the predicted values
(since surveyreg can`t do it);
proc reg data=&data.&demean noprint;
    model %scan(&endogenous,&j)= &instruments   &var   ;
    output out=&data._tmp_s1_&j (keep=mykey pred)   predicted = pred;
    %if &weight ne %str() %then %do;
        weight &weight;
    %end;
run;
```

```
    %let j=%eval(&j+1);
    %end;

    * 2.1 run syslin to get the structural estimates and save the tsls residuals ;
    proc syslin data=&data.&demean 2sls out=&data._tmp_s2_r ;
        title "intermediary step 2: second stage results &feinfo unclustered";
        %if &fe ne %str() %then %do;
title2 "ps: unclustered ses need to be multiplied by ((n-k)/(n-k-(g-1)))**0.5";
title3  " to take into account the demeaning of the fixed effect &fe. (g-1: df of &fe)";
        %end;
        endogenous      &endogenous ;
        instruments     &instruments  &var;
        model       &dependent =  &endogenous  &var / &noint ;
        output residual = iv_resid ;
        ods graphics off;
        ods output
        parameterestimates=&data._tmp_iv_est
        fitstatistics=&data._iv_fit;
        %if &weight ne %str() %then %do;
            weight &weight;
        %end;
    run;


    * 3.1 merge the tsls resids with the first stage preds;
    data &data._tmp_s1;
        merge &data._tmp_s1_1 (rename=(pred=%scan(&endogenous,1)) drop=%scan(&endogenous,&j))
        %let j=2;
        %do %while (%scan(&endogenous,&j) ne %str());
            &data._tmp_s1_&j (rename=(pred=%scan(&endogenous,&j)) drop=mykey )
        %let j=%eval(&j+1);
        %end;
    ;run;

    proc sql;
        create table &data._tmp_iv_r as select * from &data._tmp_s2_r(drop=&endogenous) as aa
        inner join &data._tmp_s1 as bb on aa.mykey=bb.mykey;
    quit;

    %let print=no;
    %clust3(data=&data._tmp_iv_r,dependent=iv_resid, var=&endogenous &var, cluster=&cluster,
weight=&weight, multi=&multi, output=&data._est_cl_step2,noint=&noint,print=&print);

    data &data._est_cl_step2;
        merge &data._tmp_iv_est (keep=variable estimate) &data._est_cl_step2 (drop=estimate);

        n_fe1=&n_fe1;
        n_fe2=&n_fe2;
        n_fe3=&n_fe3;

        df_fe_all=&df_fe_all;

        n_fe1_cl1=&n_fe1_cl1;
        n_fe1_cl2=&n_fe1_cl2;
        n_fe1_cl3=&n_fe1_cl3;
```

```
            n_fe2_cl1=&n_fe2_cl1;
            n_fe2_cl2=&n_fe2_cl2;
            n_fe2_cl3=&n_fe2_cl3;

            n_fe3_cl1=&n_fe3_cl1;
            n_fe3_cl2=&n_fe3_cl2;
            n_fe3_cl3=&n_fe3_cl3;

            *method 1 : robust cluster standard error corrected for fixed effect nestedness;
            stderr_cl1=stderr_or_cl1*(((n-k)/(n-k-df_fe_all+(n_fe1_cl1=n_fe1)*max(n_fe1-1,0)
+(n_fe2_cl1=n_fe2)*max(n_fe2-1,0)+(n_fe3_cl1=n_fe3)*max(n_fe3-1,0)))**0.5);
            %if %scan(&cluster,2) ne %str() %then %do;
                stderr_cl2=stderr_or_cl2*(((n-k)/(n-k-df_fe_all+(n_fe1_cl2=n_fe1)*max(n_fe1-1,0)
+(n_fe2_cl2=n_fe2)*max(n_fe2-1,0)+(n_fe3_cl2=n_fe3)*max(n_fe3-1,0)))**0.5);
            %end;

            %if %scan(&cluster,3) ne %str() %then %do;
                stderr_cl3=stderr_or_cl3*(((n-k)/(n-k-df_fe_all+(n_fe1_cl3=n_fe1)*max(n_fe1-1,0)
+(n_fe2_cl3=n_fe2)*max(n_fe2-1,0)+(n_fe3_cl3=n_fe3)*max(n_fe3-1,0)))**0.5);

                stderr_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
                stderr_cl1_cl3=stderr_or_cl1_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
                stderr_cl2_cl3=stderr_or_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
                stderr_cl1_cl2_cl3=stderr_or_cl1_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);

                stderr=(stderr_cl1**2+stderr_cl2**2+stderr_cl3**2-stderr_cl1_cl2**2
-stderr_cl2_cl3**2-stderr_cl1_cl3**2+stderr_cl1_cl2_cl3**2)**0.5;
                dendf=min(dendf_cl1,dendf_cl2,dendf_cl3);
            %end;
            %else %if %scan(&cluster,2) ne %str() %then %do;
                stderr_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
                stderr=(stderr_cl1**2+stderr_cl2**2-stderr_cl1_cl2**2)**0.5;
                dendf=min(dendf_cl1,dendf_cl2);
            %end;
            %else %do;
                stderr=stderr_cl1;
                dendf=dendf_cl1;
            %end;

            t=estimate/stderr;
            pvalue=(1-probt(abs(t),dendf))*2;

            if pvalue ne . then do;
                    if pvalue <0.001    then stars="***";
                    else if pvalue <0.01 then stars="** ";
                    else if pvalue <0.05 then stars="*  ";
                    else if pvalue <0.10 then stars=".  ";
            end;

            *method 2 : robust cluster standard error uncorrected for fixed effect nestedness;
            stderr2_cl1=stderr_or_cl1*(((n-k)/(n-k-df_fe_all))**0.5);
            %if %scan(&cluster,2) ne %str() %then %do;
                stderr2_cl2=stderr_or_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
            %end;

            %if %scan(&cluster,3) ne %str() %then %do;
```

```
            stderr2_cl3=stderr_or_cl3*(((n-k)/(n-k-df_fe_all))**0.5);

            stderr2_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
            stderr2_cl1_cl3=stderr_or_cl1_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
            stderr2_cl2_cl3=stderr_or_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);
            stderr2_cl1_cl2_cl3=stderr_or_cl1_cl2_cl3*(((n-k)/(n-k-df_fe_all))**0.5);

            stderr2=(stderr2_cl1**2+stderr2_cl2**2+stderr2_cl3**2-stderr2_cl1_cl2**2
-stderr2_cl2_cl3**2-stderr2_cl1_cl3**2+stderr2_cl1_cl2_cl3**2)**0.5;
        %end;
        %else %if %scan(&cluster,2) ne %str() %then %do;
            stderr2_cl1_cl2=stderr_or_cl1_cl2*(((n-k)/(n-k-df_fe_all))**0.5);
            stderr2=(stderr2_cl1**2+stderr2_cl2**2-stderr2_cl1_cl2**2)**0.5;
        %end;
        %else %do;
            stderr2=stderr2_cl1;
        %end;

        t2=estimate/stderr2;
        dendf2=n-k-df_fe_all;
        pvalue2=(1-probt(abs(t2),dendf2))*2;
        if pvalue2 ne . then do;
                if pvalue2 <0.001   then stars2="***";
                else if pvalue2 <0.01 then stars2="** ";
                else if pvalue2 <0.05 then stars2="*  ";
                else if pvalue2 <0.10 then stars2=".  ";
        end;
        call symput ('dendf2',dendf2);

        call symput ('dendf',dendf);
        call symput ('n',n);
        call symput ('g_cl1',dendf_cl1+1);
        call symput ('g_cl2',dendf_cl2+1);
        call symput ('g_cl3',dendf_cl3+1);
    run;


    %if &n_fe1>0 %then %let fe1_det=(n1=%scan(&n_fe1,1));
    %if &n_fe2>0 %then %let fe2_det=(n2=%scan(&n_fe2,1));
    %if &n_fe3>0 %then %let fe3_det=(n3=%scan(&n_fe3,1));

    %if &g_cl1>0 %then %let cluster1_det=(g1=%scan(&g_cl1,1));
    %if &g_cl2>0 %then %let cluster2_det=(g2=%scan(&g_cl2,1));
    %if &g_cl3>0 %then %let cluster3_det=(g3=%scan(&g_cl3,1));

    data _null_; set &data._iv_fit (obs=1);
        call symput ('r2',cvalue1);
    run;

    %if &nestedness ne no %then %do;
        proc print data=&data._est_cl_step2;
            var parameter estimate stderr stars pvalue ;
        run;
    %end;
    %if &nestedness ne yes %then %do;
```

```
            proc print data=&data._est_cl_step2;
                var parameter estimate stderr2 stars2 pvalue2 ;
            run;
        %end;




    %end;

    *deletings datasets;
    proc datasets lib=work nolist nodetails;
        delete _doctmp:;
        delete &data._tmp:;
        delete &data.&demean._tmp:;
        %if %upcase(&deldata)=yes %then %do;
            delete &data.&demean;
        %end;
    quit;

    title;

    %myend :;
%mend;

%macro clust3(data, dependent, var, class, format, cluster, weight, output, multi,noint,print);

    %let cluster1=%scan(&cluster,1);
    %let cluster2=%scan(&cluster,2);
    %let cluster3=%scan(&cluster,3);

    title;
    title "intermediary step 1.a: cluster with &cluster1";
    proc surveyreg data=&data ;
        cluster &cluster1;
        %if &class ne %str() %then %do;
            class &class;
        %end;
        %if &format ne %str() %then %do;
            format &format;
        %end;
        model &dependent = &var/ covb &noint solution;
        ods output parameterestimates=&data._tmp_est_cl1 fitstatistics=&data._tmp_fit_cl1
datasummary=&data._tmp_n_cl1 covb=&data._tmp_cov_cl1;
        %if &weight ne %str() %then %do;
                weight &weight;
        %end;
    run;
    quit;

    data _null_;
        set &data._tmp_est_cl1;
        count=_n_;
        call symput ('k',count);
    run;
```

```
data _null_; set &data._tmp_n_cl1;
    if _n_=1 then call symput ('n',cvalue1);
run;

%if &cluster2 ne %str() %then %do;
    title "intermediary step 1.b: cluster with &cluster2";
    proc surveyreg data=&data ;
        cluster &cluster2;
        %if &class ne %str() %then %do;
            class &class;
        %end;
        %if &format ne %str() %then %do;
            format &format;
        %end;
        model &dependent = &var/covb &noint solution;
        ods output parameterestimates=&data._tmp_est_cl2 covb=&data._tmp_cov_cl2;
        %if &weight ne %str() %then %do;
            weight &weight;
        %end;
    run;
    quit;

    %if &cluster3 ne %str() %then %do;
        title "intermediary step 1.c: cluster with &cluster3";
        proc surveyreg data=&data ;
            cluster &cluster3;
            %if &class ne %str() %then %do;
                class &class;
            %end;
            %if &format ne %str() %then %do;
                format &format;
            %end;
            model &dependent = &var/covb &noint solution;
            ods output parameterestimates=&data._tmp_est_cl3 covb=&data._tmp_cov_cl3;
            %if &weight ne %str() %then %do;
                weight &weight;
            %end;
        run;
        quit;
    %end;


    %if &multi = %str() %then %do;
        proc freq data=&data;
            %if &cluster3 ne %str() %then %do;
                tables &cluster1*&cluster2*&cluster3/noprint list out=&data._tmp_cell;
            %end;
            %else %do;
                tables &cluster1*&cluster2/noprint list out=&data._tmp_cell;
            %end;
        run;

        proc means data=&data._tmp_cell noprint;
            var count ;
            output out=&data._tmp_cell2;
        run;
```

```
            data &data._tmp_cell2;
                set &data._tmp_cell2;
                where _stat_="max";
                call symput ('maxcell',count);
            run;

            %if %sysevalf(&maxcell)>1 %then %let multi=1;
            %else %let multi=0;
            %put %str(maxcell : &maxcell);
            %put %str(multi: &multi);
        %end;

        proc surveyreg data=&data ;
            %if &multi=1 or &cluster3 ne %str() %then %do;
            title "intermediary step 2.a: cluster with &cluster1*&cluster2";
            cluster &cluster2 &cluster1;
            %end;
            %else %do;
title "intermediary step 2.a: white heteroscedasticity consistant standard errors (cell=1)";
            %end;
            %if &class ne %str() %then %do;
                class &class;
            %end;
            %if &format ne %str() %then %do;
                format &format;
            %end;
            model &dependent = &var/ &noint covb solution;
            ods output parameterestimates=&data._tmp_est_cl1_cl2
covb=&data._tmp_cov_cl1_cl2;
            %if &weight ne %str() %then %do;
                weight &weight;
            %end;
        run;
        quit;

        %if &cluster3 ne %str() %then %do;
            title "intermediary step 2.b: cluster with &cluster2*&cluster3 ";
            proc surveyreg data=&data ;
                %if &class ne %str() %then %do;
                    class &class;
                %end;
                %if &format ne %str() %then %do;
                    format &format;
                %end;
                cluster &cluster2 &cluster3;
                model &dependent = &var/ &noint covb solution;
                ods output parameterestimates=&data._tmp_est_cl2_cl3
covb=&data._tmp_cov_cl2_cl3;
                %if &weight ne %str() %then %do;
                    weight &weight;
                %end;
            run;
            quit;

            title "intermediary step 2.c: cluster with &cluster1*&cluster3 ";
```

```
            proc surveyreg data=&data ;
                %if &class ne %str() %then %do;
                    class &class;
                %end;
                %if &format ne %str() %then %do;
                    format &format;
                %end;
                cluster &cluster1 &cluster3;
                model &dependent = &var/ &noint covb solution;
                ods output parameterestimates=&data._tmp_est_cl1_cl3
covb=&data._tmp_cov_cl1_cl3;
                %if &weight ne %str() %then %do;
                    weight &weight;
                %end;
            run;
            quit;


            proc surveyreg data=&data ;
                %if &multi=1  %then %do;
                    title "intermediary step 2.d: cluster with &cluster1*&cluster2*&cluster3 ";
                    cluster &cluster1 &cluster2 &cluster3;
                %end;
                %else %do;
    title "intermediary step 2.d: white heteroscedasticity consistant standard errors (cell=1) ";
                %end;
                %if &class ne %str() %then %do;
                    class &class;
                %end;
                %if &format ne %str() %then %do;
                    format &format;
                %end;
                model &dependent = &var/ &noint covb solution;
                ods output parameterestimates=&data._tmp_est_cl1_cl2_cl3
covb=&data._tmp_cov_cl1_cl2_cl3;
                %if &weight ne %str() %then %do;
                    weight &weight;
                %end;
            run;
            quit;

        %end;
    %end;

    %if &output=%str() %then %let output=&data._tmp_cl_out;

    data &output;
        merge &data._tmp_est_cl1 (drop=tvalue probt rename=(stderr=stderr_or_cl1 dendf=dendf_cl1))
            %if  &cluster2 ne %str() %then %do;
                  &data._tmp_est_cl2 (drop=parameter estimate tvalue probt
rename=(stderr=stderr_or_cl2 dendf=dendf_cl2))
                  %if  &cluster3 ne %str() %then %do;
                      &data._tmp_est_cl3 (drop=parameter estimate tvalue probt
rename=(stderr=stderr_or_cl3 dendf=dendf_cl3))
                  %end;
                  &data._tmp_est_cl1_cl2 (drop= parameter estimate tvalue probt
```

```
rename=(stderr=stderr_or_cl1_cl2 dendf=dendf_cl1_cl2))
                  %if  &cluster3 ne %str() %then %do;
                      &data._tmp_est_cl2_cl3 (drop= parameter estimate tvalue probt
rename=(stderr=stderr_or_cl2_cl3 dendf=dendf_cl2_cl3))
                      &data._tmp_est_cl1_cl3 (drop= parameter estimate tvalue probt
rename=(stderr=stderr_or_cl1_cl3 dendf=dendf_cl1_cl3))
                      &data._tmp_est_cl1_cl2_cl3 (drop= parameter estimate tvalue probt
rename=(stderr=stderr_or_cl1_cl2_cl3 dendf=dendf_cl1_cl2_cl3))
                  %end;
            %end;
            ;
        k=&k;
        n=&n;
        %if &print ne no %then %do;
            %if  &cluster3 ne %str() %then %do;
                stderr=(stderr_or_cl1**2+stderr_or_cl2**2+stderr_or_cl3**2-stderr_or_cl1_cl2**2
-stderr_or_cl2_cl3**2-stderr_or_cl1_cl3**2+stderr_or_cl1_cl2_cl3**2)**0.5;
                dendf=min(dendf_cl1,dendf_cl2,dendf_cl3);
            %end;
            %else %if &cluster2 ne %str() %then %do;
                stderr=(stderr_or_cl1**2+stderr_or_cl2**2-stderr_or_cl1_cl2**2)**0.5;
                dendf=min(dendf_cl1,dendf_cl2);
            %end;
            %else %do;
                stderr=stderr_or_cl1;
                dendf=dendf_cl1;
            %end;

            t=estimate/stderr;
            pvalue =(1-probt(abs(t),dendf))*2;

            if pvalue ne . then do;
                    if pvalue<0.001     then stars="***";
                    else if pvalue<0.01 then stars="** ";
                    else if pvalue<0.05 then stars="*  ";
                    else if pvalue<0.10 then stars=".  ";
            end;
        %end;

        call symput ('dendf',dendf);
        call symput ('g_cl1',dendf_cl1+1);
        call symput ('g_cl2',dendf_cl2+1);
        call symput ('g_cl3',dendf_cl3+1);
    run;

    data _null_; set &data._tmp_fit_cl1 (obs=1);
        call symput ('r2',cvalue1);
    run;

    %let cluster1_det=%str();
    %let cluster2_det=%str();
    %let cluster3_det=%str();

    %if &g_cl1>0 %then %let cluster1_det=(g1=%scan(&g_cl1,1));
    %if &g_cl2>0 %then %let cluster2_det=(g2=%scan(&g_cl2,1));
    %if &g_cl3>0 %then %let cluster3_det=(g3=%scan(&g_cl3,1));
```

```
    %if &print ne no %then %do;
        title  "estimates with robust cluster standard errors";
        title2 "dependent variable: &dependent";
        title3 "clusters: &cluster1 &cluster1_det &cluster2 &cluster2_det &cluster3 &cluster3_det";
        title4 "df for student t test: min(g1,g2,g3)-1= %scan(&dendf,1) ";
        title5 "number of observations: %scan(&n,1)";
        title6 "r2: %scan(&r2,1,%str( ))";
        proc print data=&output;
            var parameter--estimate stderr stars pvalue;
        run;

        proc datasets lib=work  /*nolist nodetails*/;
            delete _doctmp:;
            delete &data._tmp:;
        quit;

    %end;
title;
%mend;
%clustr(data=all5, dependent=errors, var= chpi, fe=empcbsa yrqtr, cluster=gvkey analystid);
%clustr(data=all5, dependent=errors, var= chpi fsize ior retvol bm nanalysts,
fe=empcbsa yrqtr, cluster=gvkey analystid);
%clustr(data=all5, dependent=errors, var= chpi fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp,
fe=empcbsa yrqtr, cluster=gvkey analystid);
%clustr(data=all5, dependent=errors, var= chpi nrevs horizon star portsize
ivy tenure exp cohort firstemp, fe=empcbsa firmyrqtr, cluster=gvkey analystid);
%clustr(data=all5, dependent=bias, var= chpi, fe=empcbsa yrqtr, cluster=gvkey analystid);
%clustr(data=all5, dependent=bias, var= chpi fsize ior retvol bm nanalysts,
fe=empcbsa yrqtr, cluster=gvkey analystid);
%clustr(data=all5, dependent=bias, var= chpi fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp, fe=empcbsa yrqtr, cluster=gvkey analystid);
%clustr(data=all5, dependent=bias, var= chpi nrevs horizon star portsize
ivy tenure exp cohort firstemp,
fe=empcbsa firmyrqtr, cluster=gvkey analystid);
data all5_int;
set all5;
problayoff_int=chpi*problayoff;
owner_int=chpi*owner;
cltv90_int=chpi*cltv90;
recourse_int=chpi*recourse;
foreclosure_int=chpi*foreclosure;
run;
%clustr(data=all5_int, dependent=errors, var= problayoff_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp problayoff,
fe=empid zipyrqtr, cluster=gvkey analystid);
%clustr(data=all5_int, dependent=bias, var= problayoff_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp problayoff,
fe=empid zipyrqtr, cluster=gvkey analystid);
%clustr(data=all5_int, dependent=errors, var= owner_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp owner,
fe=empid zipyrqtr, cluster=gvkey analystid);
%clustr(data=all5_int, dependent=bias, var= owner_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp owner,
fe=empid zipyrqtr, cluster=gvkey analystid);
```

```
%clustr(data=all5_int, dependent=errors, var= cltv90_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp cltv90,
fe=empid zipyrqtr, cluster=gvkey analystid);
%clustr(data=all5_int, dependent=bias, var= cltv90_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp cltv90,
fe=empid zipyrqtr, cluster=gvkey analystid);
%clustr(data=all5_int, dependent=errors, var= recourse_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp recourse,
fe=empid zipyrqtr, cluster=gvkey analystid);
%clustr(data=all5_int, dependent=bias, var= recourse_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp recourse,
fe=empid zipyrqtr, cluster=gvkey analystid);
%clustr(data=all5_int, dependent=errors, var= foreclosure_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp foreclosure,
fe=empid zipyrqtr, cluster=gvkey analystid);
%clustr(data=all5_int, dependent=bias, var= foreclosure_int fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp foreclosure,
fe=empid zipyrqtr, cluster=gvkey analystid);
%macro randsamp(n);
%do i = 1 %to &n. ;
data all5&i;
set all5;
original_order+1;
*seed=&sysrandom ;
rand = rand('Uniform');
run;
proc sort data=all5&i(keep=analystzip analystid yrqtr)
out=round&i(rename=(analystid=pseudo_analystid));
  by analystzip rand;
run;
 data nall5&i;
  merge round&i all5&i;
  drop rand;
run;
proc sort data=nall5&i out=nround&i(drop=original_order);
  by original_order;
run;
proc sql;
create table final&i as
select a.analystzip, a.pseudo_analystid, a.analystid, a.yrqtr, a.chpi,
a.fsize,a.ior,a.retvol,a.bm,a.nanalysts,a.nrevs,
a.horizon, a.star, a.portsize, a.ivy, a.tenure, a.exp, a.cohort, a.firstemp, b.errors, b.bias
from nround&i as a, all5&i as b
where a.analystzip=b.analystzip and a.pseudo_analystid=b.analystid;
run;
%end;
%mend randsamp;
%randsamp(2);
%macro regression(n);
%do j = 1 %to &n. ;
%If &j. = 1 %then %do;
%clstr(data=final&j., dependent=errors, var= chpi fsize ior retvol bm nanalysts nrevs
horizon star portsize ivy tenure exp cohort firstemp, fe=empcbsa yrqtr, cluster=gvkey analystid);
%end;
%If &j. ne 1 %then %do;
%clstr(data=final&j.,  dependent=errors, var= chpi fsize ior retvol bm nanalysts nrevs
```

```
horizon star portsize ivy tenure exp cohort firstemp, fe=empcbsa yrqtr, cluster=gvkey analystid);
%end;
%end;
%mend regression;
%regression(2);
%macro append(n);
%do j = 1 %to &n. ;
%If &j.=1 %then %do;
data all; set final&j._est_cl; keep parameter estimate StdErr_or_cl1; run;
%end;
%If &j. ne 1 %then %do;
data final&j._est_cl; set final&j._est_cl; keep parameter estimate StdErr_or_cl1; run;
proc append base=all data=Final&j._est_cl;  run;
%end;
%end;
%mend append;
%append(2);
```