Excerpts of Code for
"Capital Commitment and Illiquidity in Corporate Bonds"
Journal of Finance

Table of Contents

SECTION I: SAS CODE TO Read Enhanced Academic TRACE data

```
**** Read the Corp Trade File *************************;

* Some fields in the Enhanced TRACE dataset used by Bessembinder,
Maxwell, Jacobsen and Venkataraman (2018, JF) differ from those
received by academic researchers under Academic TRACE subscription;

* local directory;
libname cblib 'folder with text data';

%macro readcbtrd(a,b);

    data WORK.testx;
      %let _EFIERR_ = 0; /* set the ERROR detection macro variable */
      infile "folder information/&a"
                delimiter = '|' MISSOVER DSD lrecl=32767
     firstobs=2 ;
      informat FINRA_SCRTY_ID $264. ;
       informat MSG_SEQ_NB best32. ;
       informat MSG_TYPE_CD $1. ;
       informat REC_TYPE_NM $16. ;
       informat TRD_RPT_TS anydtdtm40. ;
       informat TRD_EXCTN_TS anydtdtm40. ;
       informat TRD_STLMT_DT yymmdd10. ;
       informat ISSUE_SYM_ID $8. ;
       informat CUSIP_ID $9. ;
       informat RPT_SIDE_CD $1. ;
       informat ENTRD_VOL_QT best32. ;
       informat ENTRD_PR best32. ;
       informat PR_OVRRD_CD $1. ;
       informat CALCD_YLD_PT best32. ;
       informat SBMTG_FIRM_ID $1. ;
       informat RPTG_PARTY_ID $40. ;
       informat RPTG_PARTY_GVP_ID $40. ;
       informat CNTRA_PARTY_ID $40. ;
       informat CNTRA_PARTY_GVP_ID $40. ;
       informat RPTG_CPCTY_CD $1. ;
       informat CNTRA_CPCTY_CD $1. ;
       informat LCKD_IN_FL $1. ;
       informat TRD_MDFR_LATE_CD $1. ;
       informat TRD_MDFR_SRO_CD $1. ;
       informat PBLSH_FL $1. ;
       informat SLLR_CMSN_AMT best32. ;
       informat BUYER_CMSN_AMT best32. ;
       informat SLLR_FEES_AMT $1. ;
       informat BUYER_FEES_AMT $1. ;
       informat ASOF_FL $1. ;
       informat RVRSL_FL $1. ;
       informat SPCL_PR_FL $1. ;
       informat SPCL_PR_MEMO_TX $19. ;
```

```
informat SYSTM_CNTRL_NB best32. ;
informat SYSTM_CNTRL_DT yymmdd10. ;
informat PREV_TRD_CNTRL_NB best32. ;
informat PREV_TRD_CNTRL_DT yymmdd10. ;
informat FIRST_TRD_CNTRL_NB best32. ;
informat FIRST_TRD_CNTRL_DT yymmdd10. ;
informat TRD_ST_CD $1. ;
informat HIGH_PR $1. ;
informat HIGH_CNTRL_NB $1. ;
informat LOW_PR $1. ;
informat LOW_CNTRL_NB $1. ;
informat LSALE_PR $1. ;
informat LSALE_CNTRL_NB $1. ;
informat TRDG_MKT_CD $2. ;
informat SPCL_PRCSG_CD $1. ;
informat ATS_FL $1.;
informat NO_RMNRN_CD $1.;
format FINRA_SCRTY_ID $264. ;
format MSG_SEQ_NB best12. ;
format MSG_TYPE_CD $1. ;
format REC_TYPE_NM $16. ;
format TRD_RPT_TS datetime. ;
format TRD_EXCTN_TS datetime. ;
format TRD_STLMT_DT yymmdd10. ;
format ISSUE_SYM_ID $8. ;
format CUSIP_ID $9. ;
format RPT_SIDE_CD $1. ;
format ENTRD_VOL_QT best12. ;
format ENTRD_PR best12. ;
format PR_OVRRD_CD $1. ;
format CALCD_YLD_PT best12. ;
format SBMTG_FIRM_ID $1. ;
format RPTG_PARTY_ID $40. ;
format RPTG_PARTY_GVP_ID $40. ;
format CNTRA_PARTY_ID $40. ;
format CNTRA_PARTY_GVP_ID $40. ;
format RPTG_CPCTY_CD $1. ;
format CNTRA_CPCTY_CD $1. ;
format LCKD_IN_FL $1. ;
format TRD_MDFR_LATE_CD $1. ;
format TRD_MDFR_SRO_CD $1. ;
format PBLSH_FL $1. ;
format SLLR_CMSN_AMT best12. ;
format BUYER_CMSN_AMT best12. ;
format SLLR_FEES_AMT $1. ;
format BUYER_FEES_AMT $1. ;
format ASOF_FL $1. ;
format RVRSL_FL $1. ;
format SPCL_PR_FL $1. ;
format SPCL_PR_MEMO_TX $19. ;
format SYSTM_CNTRL_NB best12. ;
format SYSTM_CNTRL_DT yymmdd10. ;
```

```
   format PREV_TRD_CNTRL_NB best12. ;
   format PREV_TRD_CNTRL_DT yymmdd10. ;
   format FIRST_TRD_CNTRL_NB best12. ;
   format FIRST_TRD_CNTRL_DT yymmdd10. ;
   format TRD_ST_CD $1. ;
   format HIGH_PR $1. ;
   format HIGH_CNTRL_NB $1. ;
   format LOW_PR $1. ;
   format LOW_CNTRL_NB $1. ;
   format LSALE_PR $1. ;
   format LSALE_CNTRL_NB $1. ;
   format TRDG_MKT_CD $2. ;
   format SPCL_PRCSG_CD $1. ;
   format ATS_FL $1.;
   format NO_RMNRN_CD $1.;
input
            FINRA_SCRTY_ID
            MSG_SEQ_NB
            MSG_TYPE_CD $
            REC_TYPE_NM $
            TRD_RPT_TS
            TRD_EXCTN_TS
            TRD_STLMT_DT
            ISSUE_SYM_ID $
            CUSIP_ID $
            RPT_SIDE_CD $
            ENTRD_VOL_QT
            ENTRD_PR
            PR_OVRRD_CD $
            CALCD_YLD_PT
            SBMTG_FIRM_ID $
            RPTG_PARTY_ID $
            RPTG_PARTY_GVP_ID $
            CNTRA_PARTY_ID $
            CNTRA_PARTY_GVP_ID $
            RPTG_CPCTY_CD $
            CNTRA_CPCTY_CD $
            LCKD_IN_FL $
            TRD_MDFR_LATE_CD $
            TRD_MDFR_SRO_CD $
            PBLSH_FL $
            SLLR_CMSN_AMT
            BUYER_CMSN_AMT
            SLLR_FEES_AMT $
            BUYER_FEES_AMT $
            ASOF_FL $
            RVRSL_FL $
            SPCL_PR_FL $
            SPCL_PR_MEMO_TX $
            SYSTM_CNTRL_NB
            SYSTM_CNTRL_DT
            PREV_TRD_CNTRL_NB
```

```
                PREV_TRD_CNTRL_DT
                FIRST_TRD_CNTRL_NB
                FIRST_TRD_CNTRL_DT
                TRD_ST_CD $
                HIGH_PR $
                HIGH_CNTRL_NB $
                LOW_PR $
                LOW_CNTRL_NB $
                LSALE_PR $
                LSALE_CNTRL_NB $
                TRDG_MKT_CD $
                SPCL_PRCSG_CD $
                ATS_FL $
                NO_RMNRN_CD $
    ;

     if FINRA_SCRTY_ID='Confidential Treatment Requested by FINRA'
then do;
     %put "Footer found";
     end;
     else output;
    if _ERROR_ then call symputx('_EFIERR_',1);  /* set ERROR
detection macro variable */
    run;

     proc append base=cbtrad data=testx force; run;
     proc delete data=testx; run;

%mend;

*macro reads each text file and appends to the base dataset;
%macro readdata1;
%let filrf=mydir;
%let rc=%sysfunc(filename(filrf,"  folder info ")); /* assign dir name
*/
%let did=%sysfunc(dopen(&filrf)); /* open directory */
%let lstname=; /* clear filename macro var */
%let memcount=%sysfunc(dnum(&did)); /* get # files in directory */
%if &memcount > 0 %then /* check for blank directory */
     %do i=1 %to &memcount; /* start loop for files */
     %let lstname=%sysfunc(dread(&did,&i)); /* get file name to
process */

           %readcbtrd(&lstname,&i);
    %put &memcount &i &lstname;

     %end;
%let rc=%sysfunc(dclose(&did)); /* close directory */
%mend readdata1;
%readdata1;
```

- **** Clean the Corp Trade File *************************;

```
data trade;
     set cbfl3.cbtrades_2017_18_g1new;
         format ATS_MP_ID $4.0;
         ATS_MP_ID="    ";
     TRD_EXCTN_DT = datepart(TRD_EXCTN_TS);
     TRD_EXCTN_TM = timepart(TRD_EXCTN_TS);
     TRD_RPT_DT = datepart(TRD_RPT_TS);
     TRD_RPT_TM = timepart(TRD_RPT_TS);
     rename ISSUE_SYM_ID = BOND_SYM_ID CALCD_YLD_PT = YLD_CALC_PT
RPTG_CPCTY_CD=MM_CPCTY_CD LCKD_IN_FL=AGU_TRD_ID ASOF_FL=ASOF_CD
RVRSL_FL=RVRSL_CD SPCL_PR_FL=SPCL_TRD_CD SPCL_PR_MEMO_TX=SPCL_MEMO_TX
TRD_ST_CD=TRC_ST;
run;

proc sort data=trade; by cusip_id trd_exctn_dt trd_exctn_tm; run;

data trade;
     set trade;
     if cusip_id=' ' then delete; *remove trades without valid cusip;
run;

proc freq data=trade; tables trc_st; run;


/** corrected trades **/

* remove cancelled trades;
data trade_correctsd;
     set trade;
     if trc_st="C" and rvrsl_cd=" ";
     keep cusip_id systm_cntrl_nb systm_cntrl_dt;
run;
proc sort data=trade_correctsd nodupkey; by cusip_id systm_cntrl_nb
systm_cntrl_dt; run;

data trade_correct1;
     set trade;
     if trc_st="C" and rvrsl_cd=" " then delete; *remove message;
run;

proc sort data=trade_correctsd nodupkey;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;
proc sort data=trade_correct1;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;

data trade_correct;
     merge trade_correct1 (in=a) trade_correctsd (in=b);
     by cusip_id systm_cntrl_nb systm_cntrl_dt;
```

```
     if b then delete; *remove original trade that is corrected;
run;

proc delete data=trade_correctsd; run;
proc delete data=trade_correct1; run;
proc delete data=trade; run;


/** cancelled trades **/
data trade_cancelsd;
     set trade_correct;
     if trc_st="X" and rvrsl_cd=" ";
     keep cusip_id systm_cntrl_nb systm_cntrl_dt;
run;
proc sort data=trade_cancelsd nodupkey; by cusip_id systm_cntrl_nb
systm_cntrl_dt; run;

data trade_correct2;
     set trade_correct;
     if trc_st="X" and rvrsl_cd=" " then delete;  *remove message;
run;

proc sort data=trade_cancelsd nodupkey;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;
proc sort data=trade_correct2;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;

data trade_correct;
     merge trade_correct2 (in=a) trade_cancelsd (in=b);
     by cusip_id systm_cntrl_nb systm_cntrl_dt;
     if b then delete; *remove cancelled trade;
run;

proc delete data=trade_cancelsd; run;
proc delete data=trade_correct2; run;


/** cancelling reversals **/
data trade_cancelrv;
     set trade_correct;
     if trc_st="X" and rvrsl_cd="R" and ASOF_CD="R";
          *message canceling the reversal;
     keep cusip_id systm_cntrl_nb systm_cntrl_dt;
run;
proc sort data=trade_cancelrv nodupkey; by cusip_id systm_cntrl_nb
systm_cntrl_dt; run;

data trade_correct3;
     set trade_correct;
     if trc_st="X" and rvrsl_cd="R" and ASOF_CD="R" then delete;
        *remove message;
run;
```

```
proc sort data=trade_cancelrv nodupkey;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;
proc sort data=trade_correct3;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;

data trade_correct;
     merge trade_correct3 (in=a) trade_cancelrv (in=b);
     by cusip_id systm_cntrl_nb systm_cntrl_dt;
     if b then delete;   *remove reversal that is cancelled;
run;
proc delete data=trade_cancelrv; run;
proc delete data=trade_correct3; run;


/** group 8 - color code dark blue - cancelling reversals **/
data trade_cancelrv;
     set trade_correct;
     if trc_st="C" and rvrsl_cd="R" and ASOF_CD="R";  *message
canceling the reversal;
     keep cusip_id systm_cntrl_nb systm_cntrl_dt;
run;
proc sort data=trade_cancelrv nodupkey; by cusip_id systm_cntrl_nb
systm_cntrl_dt; run;

data trade_correct4;
     set trade_correct;
     if trc_st="C" and rvrsl_cd="R" and ASOF_CD="R" then delete;
          *remove message;
run;

proc sort data=trade_cancelrv nodupkey;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;
proc sort data=trade_correct4;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;

data trade_correct;
     merge trade_correct4 (in=a) trade_cancelrv (in=b);
     by cusip_id systm_cntrl_nb systm_cntrl_dt;
     if b then delete; *remove reversal that is cancelled;
run;

proc delete data=trade_cancelrv; run;
proc delete data=trade_correct4; run;



/** cancelling trades after 20 business days **/
data trade_cancelrv;
     set trade_correct;
     if trc_st="Y" and rvrsl_cd="R" and ASOF_CD="R";     *message
cancelling the trade;
```

```
     rename prev_trd_cntrl_nb=systm_cntrl_nb
prev_trd_cntrl_dt=systm_cntrl_dt;
     keep cusip_id prev_trd_cntrl_nb prev_trd_cntrl_dt;
run;

proc sort data=trade_cancelrv nodupkey; by cusip_id systm_cntrl_nb
systm_cntrl_dt; run;

data trade_correct4;
     set trade_correct;
     if trc_st="Y" and rvrsl_cd="R" and ASOF_CD="R" then delete;
*remove message;
run;

proc sort data=trade_cancelrv nodupkey;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;
proc sort data=trade_correct4;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;

data trade_correct;
     merge trade_correct4 (in=a) trade_cancelrv (in=b);
     by cusip_id systm_cntrl_nb systm_cntrl_dt;
     if b then delete;   *remove trade that is cancelled;
run;

proc delete data=trade_cancelrv; run;
proc delete data=trade_correct4; run;


/** corrected reversals **/

data trade_cancelrv;
     set trade_correct;
     if trc_st="R" and rvrsl_cd="R" and ASOF_CD="R";  *message
reversing the trade;
     rename first_trd_cntrl_nb=systm_cntrl_nb
first_trd_cntrl_dt=systm_cntrl_dt;
     keep cusip_id first_trd_cntrl_nb first_trd_cntrl_dt;
run;
proc sort data=trade_cancelrv nodupkey; by cusip_id systm_cntrl_nb
systm_cntrl_dt; run;

data trade_correct4;
     set trade_correct;
     if trc_st="R" and rvrsl_cd="R" and ASOF_CD="R" then delete;
*remove message;
run;

proc sort data=trade_cancelrv nodupkey;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;
proc sort data=trade_correct4;
     by cusip_id systm_cntrl_nb systm_cntrl_dt; run;
```

```
data trade_correct;
     merge trade_correct4 (in=a) trade_cancelrv (in=b);
     by cusip_id systm_cntrl_nb systm_cntrl_dt;
     if b then delete;   *remove trade that is reversed;
run;
proc delete data=trade_correct4; run;

data trade_reversal; set trade_correct; run;

proc delete data=trade_correct; run;



options nocenter ls=180;
*options nomlogic nosymbolgen dquote nomprint nomacrogen nonotes;
*options nomlogic nosymbolgen dquote nomprint nomacrogen notes;
options mlogic symbolgen dquote mprint macrogen notes;


* local directory;
libname cb '/home/Bonds/CB/T2016to2018';
libname cblib '/backup/Bonds/CB/GlobalDealerCode';

data trade1; set cb.cbtrades_clean_2017_18_g1new; run;

*global dealer codes;
*-------------------;
data bdnet;
     set cblib.dealercode_global_2018_reallynew;
     rptg_new=global_dnum;
     cntra_new=global_dnum;
     rptg_gvp_new=global_dnum;
     cntra_gvp_new=global_dnum;
     drop global_dnum year group academic bdcode;
run;

/* create a GLOBAL dealer code
Global_dnum is a unique dealer code created by the authors. This code
replaces the Masked ID with a simple nuber for each dealer.
This portion of the code is not included below.
*/


* 1. Customer Trades;
*----------------;
data trade_cust; *final;  *CHECK;
     set trade;
     if cust=1;
     * a unique dealer code for all customers is reported on TRACE;
     if RPT_SIDE_CD="B" then do;
          BuyDC=rptg;
```

```
            BuyRSC=RPTG_CPCTY_CD;
            SellDC=cntra;
            SellRSC=CNTRA_CPCTY_CD;
      end;
      if RPT_SIDE_CD="S" then do;
            SellDC=rptg;
            SellRSC=RPTG_CPCTY_CD;
            BuyDC=cntra;
            BuyRSC=CNTRA_CPCTY_CD;
      end;
run;


* 2. INTERDEALER TRADES;
*-------------------;
data trade_dealer;
      set trade;
      if cust=0;
run;


* 2.1 when combinations of rptg_gvp and cntra_gvp are missing;
*-------------------------------------------------------------;
data test1_1; *final;
      set trade_dealer;
      if rptg_gvp=. and cntra_gvp=.;
      if RPT_SIDE_CD="B" then do;
            BuyDC=rptg;
            BuyRSC=RPTG_CPCTY_CD;
            SellDC=cntra;
            SellRSC=CNTRA_CPCTY_CD;
      end;
      if RPT_SIDE_CD="S" then do;
            SellDC=rptg;
            SellRSC=RPTG_CPCTY_CD;
            BuyDC=cntra;
            BuyRSC=CNTRA_CPCTY_CD;
      end;
run;

data test1_2; *final;
      set trade_dealer;
      if rptg_gvp^=. and cntra_gvp^=.;
      if RPT_SIDE_CD="B" then do;
            BuyDC=rptg_gvp;
            BuyRSC=RPTG_CPCTY_CD;
            SellDC=cntra_gvp;
            SellRSC=CNTRA_CPCTY_CD;
      end;
      if RPT_SIDE_CD="S" then do;
            SellDC=rptg_gvp;
            SellRSC=RPTG_CPCTY_CD;
```

```
            BuyDC=cntra_gvp;
            BuyRSC=CNTRA_CPCTY_CD;
      end;
run;

data trade2;
      set trade_dealer;
      if rptg_gvp=. and cntra_gvp=. then delete;
      if rptg_gvp^=. and cntra_gvp^=. then delete;
run;

data test1_3;
      set trade2;
      if cntra_gvp^=. and rptg_gvp=.;
      if RPT_SIDE_CD="B" then do;
            BuyDC=rptg;
            BuyRSC=RPTG_CPCTY_CD;
            SellDC=cntra_gvp;
            SellRSC=CNTRA_CPCTY_CD;
      end;
      if RPT_SIDE_CD="S" then do;
            SellDC=rptg;
            SellRSC=RPTG_CPCTY_CD;
            BuyDC=cntra_gvp;
            BuyRSC=CNTRA_CPCTY_CD;
      end;
run;

data test1_4;
      set trade2;
      if cntra_gvp=. and rptg_gvp^=.;
      if RPT_SIDE_CD="B" then do;
            BuyDC=rptg_gvp;
            BuyRSC=RPTG_CPCTY_CD;
            SellDC=cntra;
            SellRSC=CNTRA_CPCTY_CD;
      end;
      if RPT_SIDE_CD="S" then do;
            SellDC=rptg_gvp;
            SellRSC=RPTG_CPCTY_CD;
            BuyDC=cntra;
            BuyRSC=CNTRA_CPCTY_CD;
      end;
run;


data trade_all; set trade_cust test1_1 test1_2 test1_3 test1_4; run;
```

```
/* double counting of inter-dealer trades & Affiliate Trades*/

*identify affiliate trades - interdealer trades with dc=3763. These
trades are identified in TRACE by a unique dealer code;

data trade_all;
     set trade_all;
     if selldc=3763 or buydc=3763 then aff=1; else aff=0;
run;

* adjust for double reporting of ID trades that are not affiliate
trades;
data trade_all;
     set trade_all;
     if cust=0 and aff=0 and rpt_side_cd="B" then delete;
run;

*some affiliate trades are simply transfer of positions to non-member
affiliates, reported by broker dealer with spcl_prcsg_cd=A. These
trades are not desseminated on TRACE and should be deleted;

data trade_all;
     set trade_all;
     if aff=1 and spcl_prcsg_cd="A" then delete;
run;

*some affiliate trades are legitimate trades with non-member
affiliates, reported by broker dealer with special code=" ". These
trades are  disseminated on TRACE and can be treated as customer
trades (global_dnum=544);

data trade_all;
     set trade_all;
     if aff=1 and spcl_prcsg_cd^="A" then cust=1;
     if aff=1 and buydc=3763 then buydc=544;
     if aff=1 and selldc=3763 then selldc=544;
run;

data cb.trades_w_dealercodes_2018_rnew; set trade_all; run;
```

SECTION II: SAS CODE TO MATCH TRADES TO FISD, TO CREATE 'AGGREGATE MARKET' SAMPLE OF TRADES, AND TO IDENTIFY DEALERS IN THE TOP 70% SAMPLE AND CONSTANT DEALER SAMPLE

```
/*/Part A: Match Screened CUSIPS to FISD Data, Delete if Missing Offering or
Maturity Date, Create File of Dates w/ Changes to Amount Outstanding
Unrelated to Maturity*/;

/*1A) Start with initial file of screened CUSIPS*/;
data t;set bond.cleancusip_character_2006_2016;drop cusip_id;run;
data t;set t;cusip_id=substr(complete_cusip,1,9);run;
proc sort data=t nodupkey;by cusip_id;run;

/*2A) Stack two FISD data sets. Revised to extend sample to 2016.
(fisd_data0190to0416 includes issues between 1990 and April 2016,Fisd_data
includes 1950 to October 2014)*/
data temp;set bond.fisd_data0190to0416;new=1;run;
data FISD;set bond.Fisd_data temp;F=1;run;

/*3A) Merge screened cusips to FISD data by issue_id*/
proc sort data=t;by issue_id;run;
proc sort data=FISD;by issue_id;run;
data three;merge t (in=a) FISD (in=b);by issue_id;if a;run;
proc sort data=three;by issue_id descending new;run;
proc sort data=three nodupkey;by issue_id;run;

/*4A) Delete if missing offering or maturity date*/
data four;set three;if offering_date=. then delete;if maturity=. then
delete;run;
proc sort data=four out=test nodupkey;by issue_id;run;

/*5A)Save final sample of cusips for input into aggregate data sas code.*/
data bond.fincusip121316JF;set four;keep cusip_id issue_id issuer_id
offering_amt offering_date maturity industry_group r144a;run;
proc sort data=bond.fincusip121316JF nodupkey;by issue_id;run;

/*6A) Create a file that identifies dates of changes to amount outstanding
unrelated to the date the issue matures, use action_type in FISD*/
data bond.amtOSJF;set four;if (offering_amt ne amount_outstanding) and
action_type not in ('IM');keep issue_id effective_date
amount_outstanding;run;
/*ACTION_TYPE IM = Issue Matured*/

/*Part B: Create Aggregate Market Sample*/;

/*1B) Pull in main sample created in Part A*/
data inv1;set bond.fincusip121316JF;keep cusip_id;run;
proc sort data=inv1;by cusip_id;run;

/*2B) Pull in trade files from 2006 to 2010. Note the format changed in 2010.
Create volume and time variables*/

data top1a;set
bondnew.Trades_w_dealercodes_2006_new bondnew.Trades_w_dealercodes_2007_new
bondnew.Trades_w_dealercodes_2008_new bondnew.Trades_w_dealercodes_2009_new
bondnew.Trades_w_dealercodes_2010_new;format TRD_EXCTN_DT MMDDYY10.;
```

```
/*Create volume and 9-digit CUSIP*/
volume_dol=(entrd_vol_qt*1000*(entrd_pr/100));cusip_id=substr(cusip_id,1,9);f
ormat cusip_id $9.;length cusip_id $9;
keep TRD_EXCTN_DT EXCTN_TM cusip_id volume_dol buydc selldc entrd_vol_qt
entrd_pr cust buyrsc sellrsc;run;
/*Create time variables based on execution date and time*/
data top1a;set top1a;trdyrmonth = year(TRD_EXCTN_DT)*12 +
month(TRD_EXCTN_DT);
      trdsec = EXCTN_TM/1000000;
      trdhr = int(trdsec/10000);
      trdmin = int(trdsec/100) - trdhr*100;
      trdsec = trdsec - trdhr*10000 - trdmin*100;
      trdday = day(TRD_EXCTN_DT);
      trdmonth = month(TRD_EXCTN_DT);
      trdyr = year(TRD_EXCTN_DT);
      trdtime = ((((trdyr*12 + trdmonth)*30 + trdday)*24 + trdhr)*60 +
trdmin)*60 + trdsec;run;

/*3B) Pull in trades files from 2010 to 2016. Note the format changed in
2010. Create volume and time variables*/
data top1b;set bondnew.Trades_w_dealercodes_2010_11_new
bondnew.Trades_w_dealercodes_2012_g2_new
bondnew.Trades_w_dealercodes_2013_g1_new
bondnew.Trades_w_dealercodes_2014_g1_new
bondnew.Trades_w_dealercodes_2015_g1_new;
format TRD_EXCTN_DT MMDDYY10.;
/*Create volume and 9-digit CUSIP*/
volume_dol=(entrd_vol_qt*(entrd_pr/100));
EXCTN_TM=trd_EXCTN_TM;format EXCTN_TM time.;
keep TRD_EXCTN_DT EXCTN_TM cusip_id volume_dol buydc selldc entrd_vol_qt
entrd_pr cust buyrsc sellrsc TRDG_MKT_CD;run;
/*Create time variables based on execution date and time*/
data top1b;set top1b;trdyrmonth = year(TRD_EXCTN_DT)*12 +
month(TRD_EXCTN_DT);
      trdsec = second(EXCTN_TM);
      trdhr = hour(EXCTN_TM);
      trdmin = minute(EXCTN_TM);
      trdday = day(TRD_EXCTN_DT);
      trdmonth = month(TRD_EXCTN_DT);
      trdyr = year(TRD_EXCTN_DT);
      trdtime = ((((trdyr*12 + trdmonth)*30 + trdday)*24 + trdhr)*60 +
trdmin)*60 + trdsec;run;

/*4B) Combine 2006-2010 and 2010-2016 trade samples. Create a unique trade
identifier 'trid'. An interdealer trade will later have the same trid.*/
data AGGtrades0215;set top1a top1b;trid=_N_;run;

/*5B) Keep only cusips in main sample with trade data*/
proc sort data=AGGtrades0215;by cusip_id;run;
data trades2;merge AGGtrades0215 (in=a) inv1 (in=b);by cusip_id;if a and
b;run;

/*6B) Create a duplicate for each trade. For interdealer trades, we can
construct capital for each dealer. Create 'txn' equal to 1 for dealer buys
and -1 for dealer sells.
Create a dealer code 'dc' for each trade side.*/
data trades2a;set trades2;txn=1;run;
```

```
data trades2b; set trades2a trades2;run;
data trades2b;set trades2b;
      if txn=. then txn=-1;
      if txn=1 then dc=buydc;else dc=selldc;
      if txn=1 then rsc=buyrsc;else rsc=sellrsc;
      drop buydc selldc buyrsc sellrsc;run;

/*7B) Delete customer trades with dc=544 */
data trades2b;set trades2b;if dc=544 then delete;run;

/*8B) Delete bonds with less than 5 trades over entire sample*/
proc sort data=trades2b;by cusip_id;run;
/*Count interdealer trades only once*/
proc sort data=trades2b out=temp nodupkey;by cusip_id trid;run;
proc means data=temp noprint;by cusip_id;output out=counts n=numtrd;run;
data counts;set counts;keep cusip_id _FREQ_;run;
proc sort data=counts;by cusip_id;run;
data trades3;merge trades2b counts;by cusip_id;run;
data trades3;set trades3;if _FREQ_ ge 5;run;
/*How many cusips and unique trades after this screen?*/
proc sort data=trades3 out=temp nodupkey;by cusip_id;run;
proc sort data=trades3 out=temp nodupkey;by trid;run;

/*9B) Now that we have trades, match to issue characteristics by 9-digit
cusip*/
data mat3;set
bond.cleancusip_character_2006_2016;cusip_id=substr(complete_cusip,1,9);run;
proc sort data=mat3 nodupkey;by cusip_id;run;
proc sort data=trades3;by cusip_id;run;
data mat4;merge trades3 (in=a) mat3 (in=b);by cusip_id;if a;run;
/*How many cusips and unique trades after this screen?*/
proc sort data=mat4 out=temp nodupkey;by cusip_id;run;
proc sort data=mat4 out=temp nodupkey;by trid;run;

/*10B) Get daily ratings data and create high yield indicator. Merge by
issue_id and day*/
proc sort data=mat4;by issue_id trd_exctn_dt;run;
data Ratingsbydate12616;set bond.Ratingsbydate;drop spratenumber
moodyratenumber;format date mmddyy10.;rename date=trd_exctn_dt;run;
proc sort data=Ratingsbydate12616;by issue_id trd_exctn_dt;run;
data trades4;merge mat4 (in=a) Ratingsbydate12616 (in=b);by issue_id
trd_exctn_dt;if a;run;
/*If missing daily rating retain previous day rating*/
data trades4;set trades4;by issue_id;retain rating2;if rating ne . then
rating2=rating;
else rating2=rating2;run;
/*Set highyield=. for unrated bonds*/
data trades4;set trades4;if rating2 ge 5 then highyield=1;else highyield=0;
if rating2=99 then highyield=.;run;
data trades3;set _null_;run; data trades2;set _null_;run;data trades2b;set
_null_;run;data trades2a;set _null_;run;

/*11B) Get dissemination date. File provided by FINRA*/
proc sort data=trades4;by cusip_id;run;
proc sort data=bond.dess;by cusip_id;run;
data trades6;merge trades4 (in=a) bond.dess (in=b);by cusip_id;if a;run;
```

```
/*12B) If any single transaction has volume that exceeds issue size by 1x,
then delete all trades from that cusip*/;
data OK2;set trades6;if volume_dol gt (offering_amt*1000) then X=1;run;
proc sort data=OK2;by cusip_id descending X;run;
proc sort data=OK2 nodupkey out=ok2 (keep=cusip_id X);by cusip_id;run;
proc sort data=trades6;by cusip_id;run;
data oneb;merge trades6 (in=a) ok2 (in=b);by cusip_id;if a;run;
data trades7;set oneb;if X ne 1;run;
/*How many cusips and unique trades after this screen?*/
proc sort data=trades7 nodupkey out=test;by cusip_id;run;
proc sort data=trades7 nodupkey out=test;by trid;run;
data trades6;set _null_;run;

/*13B) Delete primary market trades*/
data trades8;set trades7;if TRDG_MKT_CD eq 'P1' then delete;run;
/*How many cusips and unique trades after this screen?*/
proc sort data=trades8 out=temp nodupkey;by cusip_id;run;
proc sort data=trades8 out=temp nodupkey;by trid;run;

/*14B) Determine changes to amount outstanding*/

proc sort data=trades8;by issue_id;run;
/*Create unique identifier for each trade observation'tempid'*/
data trades8;set trades8;tempid=_N_;run;
data trades9;set trades8;keep tempid issue_id TRD_EXCTN_DT;run;
/*Merge trade data with 'amtOSJF' file by issue_id. Keep only issues with
change to amount outstanding*/
proc sort data=bond.amtOSJF;by issue_id;run;
data trades10;merge trades9 (in=a) bond.amtOSJF (in=b);by issue_id;if a and
b;run;

/*effective_date is 'The date on which the change to the issue's amount
outstanding became effective' in FISD.*/
data trades10;set trades10;if TRD_EXCTN_DT lt effective_date then delete;run;
proc sort data=trades10;by tempid issue_id TRD_EXCTN_DT descending
effective_date;run;
proc sort data=trades10 nodupkey;by tempid issue_id TRD_EXCTN_DT;run;
proc sort data=trades8;by tempid;run;
data trades11;merge trades8 (in=a) trades10 (in=b);by tempid;if a;run;
/*If change to 'amount_outstanding' then the final amount outstanding finOS
is set to amount_outstanding*/
/*If no change to 'amount_outstanding' then the final amount outstanding
finOS is set to the offering amount*/
/*Create 'enddt' variable. If change to 'amount_outstanding' that results in
0 outstanding, enddt is the effective_date, otherwise enddt is the maturity
date.*/
data trades11;set trades11;if amount_outstanding ne . then
finOS=amount_outstanding;else finOS=offering_amt;
if finOS=0 then enddt=effective_date;else enddt=maturity;format enddt
YYMMDDN8.;run;

/*15B) Delete trades after end date*/
data trades11;set trades11;if TRD_EXCTN_DT gt enddt then delete;run;
/*How many cusips and unique trades after this screen?*/
proc sort data=trades11 out=temp nodupkey;by cusip_id;run;
proc sort data=trades11 out=temp nodupkey;by trid;run;
```

```
/*16B) Update ratings downgrades/upgrades for a few months beyond FISD
availability*/
data trades11;set trades11;cus6=substr(cusip_id,1,6);
if cus6='00507V' and TRD_EXCTN_DT ge '17MAY2016'd then highyield=0;
if cus6='017363' and TRD_EXCTN_DT ge '29JUL2016'd then highyield=1;
if cus6='05463D' and TRD_EXCTN_DT ge '02SEP2016'd then highyield=0;
if cus6='086516' and TRD_EXCTN_DT ge '26JUL2016'd then highyield=0;
if cus6='109641' and TRD_EXCTN_DT ge '13SEP2016'd then highyield=1;
if cus6='117043' and TRD_EXCTN_DT ge '22AUG2016'd then highyield=0;
if cus6='23331A' and TRD_EXCTN_DT ge '26JUN2016'd then highyield=0;
if cus6='364760' and TRD_EXCTN_DT ge '20MAY2016'd then highyield=1;
if cus6='462613' and TRD_EXCTN_DT ge '14APR2016'd then highyield=0;
if cus6='481165' and TRD_EXCTN_DT ge '21APR2016'd then highyield=1;
if cus6='529043' and TRD_EXCTN_DT ge '03JUN2016'd then highyield=0;
if cus6='67066G' and TRD_EXCTN_DT ge '07SEP2016'd then highyield=0;
if cus6='761655' and TRD_EXCTN_DT ge '05JUL2016'd then highyield=1;
if cus6='776249' and TRD_EXCTN_DT ge '20JUL2016'd then highyield=0;
if cus6='84860W' and TRD_EXCTN_DT ge '22APR2016'd then highyield=0;
if cus6='868157' and TRD_EXCTN_DT ge '28JUN2016'd then highyield=1;
if cus6='87161C' and TRD_EXCTN_DT ge '17JUN2016'd then highyield=0;
if cus6='929160' and TRD_EXCTN_DT ge '09MAR2016'd then highyield=0;run;

/*17B) All bonds are disseminated after June 2014. Delete if missing
dissemination date before*/
data trades11;set trades11;if (dessdate=. or dessdate=' ') and
year(TRD_EXCTN_DT) ge 2015 then dessdate=offering_date;
if (dessdate=. or dessdate=' ') and year(TRD_EXCTN_DT) eq 2014 and
month(TRD_EXCTN_DT) gt 6 then dessdate=offering_date;run;
data trades12;set trades11;if dessdate=' ' then delete;if dessdate=. then
delete;run;
/*How many cusips and unique trades after this screen?*/
proc sort data=trades12 nodupkey out=test;by cusip_id;run;
proc sort data=trades12 nodupkey out=test;by trid;run;

/*18B) Update with 144A flag and industry group variables from FISD in file
'small_2006_2016'*/
proc sort data=trades12;by cusip_id;run;
proc sort data=bond.small_2006_2016;by cusip_id;run;
data trades13;merge trades12 (in=a) bond.small_2006_2016 (in=b);if a;by
cusip_id;run;

/*19B) Delete trades executed prior to sample start at January 2006*/
data trades13;set trades13;if year(trd_exctn_dt) lt 2006 then delete;run;
/*How many cusips and unique trades after this screen?*/
proc sort data=trades13 nodupkey out=test;by cusip_id;run;
proc sort data=trades13 nodupkey out=test;by trid;run;

/*20B) Save final Aggregate Market data sample*/
data bond.aggtradedata121516JF;set trades13;run;

/*21B) Create alternate Aggregate Market data sample that excludes one
relatively large dealer that, during 2014, began to report an immediately
offsetting transaction for the large majority of its principal trades.
Conversations with FINRA indicated that these transactions actually
represented transfers of inventory to an off-shore subsidiary. */
/*Since November 2015 FINRA has required dealers to specifically flag such
offshore affiliate transactions. See
```

```
http://www.finra.org/industry/notices/15-14. Since the affiliate flag was not
available for the majority of our sample period we could not reliably
identify which trades involved genuine capital commitment by this bank. See
Section II.b of the paper and Footnote 15. */
/*Create new aggregate data that excludes the affiliate code 3763 (which is
first reported in November 2015*/
/*and that excludes dealer 3458 April 2014 to October 2015.*/
data bond.aggtradedata121516ADJJF;set bond.aggtradedata121516JF;
if dc=3763 then delete;
if dc=3458 and trdyr=2014 and trdmonth ge 4 then delete;
if dc=3458 and trdyr in (2015,2016) then delete;run;


/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*Part C: Determine Dealers in the Top 70 and
Constant Dealer samples*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/;

/*1C) Find customer volume for each dealer-year relative to total customer
volume each year*/
/*Exclude dealer 3458 with affiliate trades*/
data top2;set bond.aggtradedata121516JF;keep dc TRD_EXCTN_DT volume_dol;if
cust=1;if dc=3458 then delete;run;
data top2;set top2;year=year(TRD_EXCTN_DT);run;
proc sort data=top2;by year dc;run;
proc means data=top2;by year dc;var volume_dol;output out=year sum=dcvol;run;
proc sort data=top2;by year;run;
proc means data=top2;by year;var volume_dol;output out=year2 sum=totvol;run;
data dcthreshold121516;merge year year2;by year;run;
proc sort data=dcthreshold121516;by year descending dcvol;run;
data dcthreshold121516;set dcthreshold121516;retain pct;by
year;p=dcvol/totvol;if first.year then pct=p;else pct=pct+p;run;
/*how many potential dealers?*/
proc sort data=dcthreshold121516 nodupkey out=test;by dc;run;

/*2C) Dealers in Top 70% Sample*/
proc sort data=dcthreshold121516;by year descending dcvol;run;
data bond.dealerthresh051917JF;set dcthreshold121516;if lag(pct) gt .70 and
lag(year)=year then delete;run;
proc sort data=bond.dealerthresh051917JF nodupkey out=test;by dc;run;
/*25 dealers*/

/*3C) Dealers in Constant Dealer Sample*/
data top2;set bond.aggtradedata121516JF;keep dc TRD_EXCTN_DT volume_dol;if
cust=1;run;
data top2;set top2;year=year(TRD_EXCTN_DT);run;
proc sort data=top2;by year dc;run;
proc means data=top2;by year dc;var volume_dol;output out=dcconstant121516
sum=dcvol;run;
proc sort data=dcconstant121516;by year descending dcvol;run;
data two;set dcconstant121516;retain rank;by year;if first.year then
rank=1;else rank=rank+1;run;
/*Keep if in Top 30 by volume in any given year*/
data two;set two;if rank le 30;run;
proc sort data=two nodupkey;by dc;run;
/*58 unique dealers in top 30 in any year*/

data two;set two;ind=1;keep dc ind;run;
proc sort data= dcconstant121516;by dc;run;
```

```
proc sort data= two; by dc;run;
data three;merge dcconstant121516 two;by dc;run;
data three;set three;if ind=1;run;
proc transpose data=three out=four;by dc;id year;var dcvol;run;
data four;set four;if _2006=. or _2007=. or _2008=. or _2009=. or _2010=. or
_2011=. or _2012=. or _2013=. or _2014=. or _2015=. or _2016=. then
delete;run;
/*Keep if traded in all sample years */
/*36 left*/
data four;set four;tot= _2006+ _2007+ _2008+ _2009+ _2010+ _2011+ _2012+
_2013+ _2014+ _2015 + _2016;run;
proc sort data=four;by descending tot;run;
/*Exclude dealer 3458 with affiliate trades*/
data bond.dealerconstant051917JF;set four;if dc=3458 then delete;run;
```

SECTION III: SAS CODE TO CREATE TABLE III, TRADING COSTS FOR CUSTOMER TRADES

```
*read BARCLAYS index values for benchmark;
data Barclay; set cb2.barclayindex_120116; run;
proc sort data=barclay; by date; run;
data barclay;
      set barclay;
      drop B_144A B_MBS B_CMBS B_ABS B_TShort S_P_500;
      SPINDX=S_P_500;
      B_Tlong_pre = lag1(B_Tlong);
      B_Corp_pre = lag1(B_Corp);
      B_T3M_pre = lag1(B_T3M);
      B_HY_pre = lag1(B_HY);
      SPINDX_pre = lag1(SPINDX);
      rename date=trd_exctn_dt;
run;

*read trade data;
data tradeall;
      set bond.aggtradedata121516adjJF;
      if TRDG_MKT_CD="P1" then delete; *primary market;
      if cust=1;                    *customer trades;
      if rsc="P";                   *dealer acts as principal;
      if offering_amt*1000 lt 500000000 then sz=1;
      if offering_amt*1000 ge 500000000 and offering_amt*1000 lt 1000000000
      then sz=2;
      if offering_amt*1000 ge 1000000000 then sz=3;
      if TRD_EXCTN_DT gt dessdate then trans=1; else trans=0;
      drop trdsec trdhr trdmin trdday trdmonth trdyr trdg_mkt_cd _freq_
      issue_id Issuer_id rating rating2 X tempid effective_date
      amount_outstanding finOS enddt cus6 industry_group quant_sgn year;
run;

proc sort data=tradeall; by trd_exctn_dt; run;
data tradeall;
      merge tradeall (in=a) barclay (in=b);
      by trd_exctn_dt;
      if a and b;
run;

proc sort data=tradeall nodupkey out=tradeall;
      by cusip_id TRD_EXCTN_DT trdtime trid; run;
proc sort data=tradeall; by cusip_id TRD_EXCTN_DT exctn_tm; run;

*create variables for indicator variable regressions following Bessembinder,
Maxwell and Venkataraman (2006, JFE);
data trade1;
      set tradeall;
      by cusip_id trd_exctn_dt;
      *previous trade;
      lcusip_id=lag1(cusip_id);
      ltrd_exctn_dt=lag1(trd_exctn_dt);
      lexctn_tm = lag1(exctn_tm);
      lentrd_pr=lag1(entrd_pr);
```

```
      lentrd_vol_qt=lag1(entrd_vol_qt);
      lcust=lag1(cust);
      ltxn=lag(txn);
      lvolume_dol=lag1(volume_dol);
      lB_Tlong=lag1(B_Tlong);
      lB_Corp=lag1(B_corp);
      lB_T3M=lag1(B_T3m);
      lB_HY=lag1(B_HY);
      lSPINDX=lag1(SPINDX);
   *one-day lag of previous trade;
      lB_Tlong_pre=lag1(B_Tlong_pre);
      lB_Corp_pre=lag1(B_Corp_pre);
      lB_T3M_pre=lag1(B_T3m_pre);
      lB_HY_pre=lag1(B_HY_pre);
      lspindx_pre=lag1(SPINDX_pre);
      *elapsed day;
      if cusip_id =lcusip_id then elapdays = 1 + trd_exctn_dt -
      ltrd_exctn_dt;
      weightvar1 = 1/sqrt(elapdays);
      *new cusip;
      if first.cusip_id then do;
            ltrd_exctn_dt=.;
            lexctn_tm =.;
            lentrd_pr=.;
            lentrd_vol_qt=.;
            lcust=lag1(cust);
            ltxn=.;
            lvolume_dol=.;
            lB_Tlong=.;
            lB_corp=.;
            lB_T3M=.;
            lB_HY=.;
            lspindx=.;
            elapdays=.;
            weightvar1=.;
            lB_Tlong_pre=.;
            lB_Corp_pre=.;
            lB_T3M_pre=.;
            lB_HY_pre=.;
            lspindx_pre=.;
      end;
      format ltrd_exctn_dt date10.;
run;
data trade2;
      set trade1;
      t_ret = ((entrd_pr-lentrd_pr)/lentrd_pr)*100;
      t_chg = (entrd_pr-lentrd_pr);
      *adjacent trades on the same day then use one-day return;
      if (cusip_id=lcusip_id) and (trd_exctn_dt=ltrd_exctn_dt) then do;
            stockret = ((spindx-spindx_pre)/spindx_pre)*100;
            trsyret = ((B_Tlong-B_Tlong_pre)/B_Tlong_pre)*100;
            bondret = ((B_corp-B_corp_pre)/B_corp_pre)*100;
            Tbillret = ((B_T3m-B_T3m_pre)/B_T3m_pre)*100;
            HYret = ((B_HY-B_HY_pre)/B_HY_pre)*100;
      end;
 *adjacent trades are NOT on the same day then multiday return including both
 day s and day t;
```

```
      if (cusip_id=lcusip_id) and (trd_exctn_dt^=ltrd_exctn_dt) then do;
            stockret = ((spindx-lspindx_pre)/lspindx_pre)*100;
            trsyret = ((B_Tlong-lB_Tlong_pre)/lB_Tlong_pre)*100;
            bondret = ((B_corp-lB_corp_pre)/lB_corp_pre)*100;
            Tbillret = ((B_T3m-lB_T3m_pre)/lB_T3m_pre)*100;
            HYret = ((B_HY-lB_HY_pre)/lB_HY_pre)*100;
      end;
      TSret = trsyret - Tbillret;
      CSret = HYret - bondret;
      if (cusip_id^=lcusip_id) then do;
            stockret = .;
            trsyret = .;
            bondret = .;
            Tbillret = .;
            HYret = .;
            t_ret = .;
            t_chg = .;
            TSret = .;
            HYret = .;
      end;
      drop B_Tlong_pre B_Tlong B_Corp_pre B_Corp B_T3M_pre B_T3M B_HY_pre
      B_HY SPINDX_pre SPINDX lB_Tlong lB_corp lB_T3M lB_HY lspindx
      lB_Tlong_pre lB_Corp_pre lB_T3M_pre lB_HY_pre lspindx_pre HYret
      Tbillret;
run;

data trade3;
      set trade2;
      * Q(t);
      *******;
      Q = 0;
      if txn=1 then Q=-1;
            *if (RPT_SIDE_CD="B") and (cust=1) then Q=-1;
      if txn=-1 then Q=1;
            *if (RPT_SIDE_CD="S") and (cust=1) then Q=1;
      preQ = 0;
      if ltxn=1 then preQ=-1;
            *if (lRPT_SIDE_CD="B") and (lcust=1) then preQ=-1;
      if ltxn=-1 then preQ=1;
            *if (lRPT_SIDE_CD="S") and (lcust=1) then preQ=1;
      deltaQ = Q - preQ;
      if (cusip_id^=lcusip_id) then deltaQ=.;
            *deltaQtrace = deltaQ*trace;
      if (cusip_id^=lcusip_id) then preQ=.;
      * I(t);
      ******;
      if cust=0 then D=1; else D=0;
      if lcust=0 then preD=1; else preD=0;
      if (cusip_id^=lcusip_id) then preD=.;
      abslret=abs(t_ret);
      if abslret>10 then delete;
      *price change unlikely to reflect trading cost bounce, follows
         Bessembinder, Maxwell and Venkataraman (2006);
run;

proc means data=trade3 noprint;
      var cust;
```

```
      by cusip_id;
      output out=tradfreq n=numtrad;
run;

data trade4;
      merge trade3 (in=a) tradfreq (in=b);
      by cusip_id;
      if a and b;
      if numtrad>20; *keep bonds with more than 20 trades over sample period;
      keep cusip_id exctn_tm trd_exctn_dt entrd_pr entrd_vol_qt volume_dol
      txn dc rsc cust offering_amt highyield dessdate r144a trdsz sz trans
      stockret t_ret trsyret CSret TSret bondret deltaQ elapdays weightvar1
      age young offering_date;
run;

*create period variables;
data trade5;
      set trade4;
      trdyrmonth = year(TRD_EXCTN_DT)*12 + month(TRD_EXCTN_DT);
      MDY=mdy(month(TRD_EXCTN_DT),1,year(TRD_EXCTN_DT));format MDY MMYYn4.;
      yr=year(MDY);mo=month(MDY);
      *definition;
      if (yr=2006) or (yr=2007 and mo le 6) then pre=1;else pre=0;
      if (yr=2007 and mo ge 7) or (yr=2008) or (yr=2009 and mo le 4) then
      crisis=1;else crisis=0;
      if (yr=2009 and mo ge 5) or (yr=2010 and mo le 6) then post=1; else
      post=0;
      if (yr=2010 and mo ge 7) or (yr=2011) or (yr=2012) or (yr=2013) or
      (yr=2014 and mo le 3) then reg=1; else reg=0;
      if (yr=2014 and mo ge 4) or (yr in (2015,2016)) then volc=1; else
      volc=0;
      *deltaQ interaction;
            deltaq_volume_dol = deltaq*log(volume_dol);
            deltaq_1 = deltaq*pre;
            deltaq_2 = deltaq*crisis;
            deltaq_3 = deltaq*post;
            deltaq_4 = deltaq*reg;
            deltaq_5 = deltaq*volc;
      period=0;
      if (yr=2006) or (yr=2007 and mo le 6) then period=1;
      if (yr=2007 and mo ge 7) or (yr=2008) or (yr=2009 and mo le 4) then
      period=2;
      if (yr=2009 and mo ge 5) or (yr=2010 and mo le 6) then period=3;
      if (yr=2010 and mo ge 7) or (yr=2011) or (yr=2012) or (yr=2013) or
      (yr=2014 and mo le 3) then period=4;
      if (yr=2014 and mo ge 4) or (yr in (2015,2016)) then period=5;
      if period=0 then delete;
run;

*trading cost by period;
proc model data = trade5 noprint;
 parms b0 b2 b3 b4 b5 b6 b7 b8 b9 b10 b11;
 t_ret = b0 + b2*trsyret + b3*stockret + b4*bondret + b5*TSret + b6*CSret +
         b7*deltaq + b8*deltaq_2 + b9*deltaq_3 + b10*deltaq_4 + b11*deltaq_5;
        fit t_ret/ gmm kernel = (bart,1,0) outest = temp1;
        weight weightvar1;
```

```
         instruments trsyret stockret bondret TSret CSret deltaq deltaq_2
        deltaq_3 deltaq_4 deltaq_5;
quit;

*trade size;
data trade5;
      set trade5;
      period=0;
      if volume_dol<100000 then trdsz=1;
      if volume_dol>=100000 and volume_dol<=1000000 then trdsz=2;
      if volume_dol>1000000 and volume_dol<=10000000 then trdsz=3;
      if volume_dol>10000000 then trdsz=4;
      if volume_dol<1000 then delete;
run;
proc sort data=trade5; by trdsz; run;
proc model data = trade5 noprint;
 parms b0 b2 b3 b4 b5 b6 b7 b8 b9 b10 b11;
 t_ret = b0 + b2*trsyret + b3*stockret + b4*bondret + b5*TSret + b6*CSret +
              b7*deltaq + b8*deltaq_2 + b9*deltaq_3 + b10*deltaq_4 +
              b11*deltaq_5;
             fit t_ret/ gmm kernel = (bart,1,0) outest = temp1;
             weight weightvar1;
             instruments trsyret stockret bondret TSret CSret deltaq deltaq_2
                deltaq_3 deltaq_4 deltaq_5;
              by trdsz;
quit;

* volume distribution;
proc sort data=trade5; by period trdsz; run;
proc univariate data=trade5 noprint;
      var volume_dol;
      by period;
      output out=test1 sum=volume_all;
run;
proc univariate data=trade5 noprint;
      var volume_dol;
      by period trdsz;
      output out=test2 mean=trdsz_mn median=trdsz_md sum=volume_trdsz;
run;
proc sort data=test2; by period; run;
proc sort data=test1; by period; run;
data test3;
      merge test2 (in=a) test1 (in=b);
      by period;
      if a and b;
      volume_pct=(volume_trdsz/volume_all)*100;
run;

* By Credit rating - use highyield;
* Issue Size use sz;
*Bond Age - use age;
      age=trd_exctn_dt - offering_date;
      if age <= 365 then young=1; else young=0;
* Click versus call;
      if ((highyield=0) and (sz=3) and (young=1) and (trdsz^=4)) then elec=1;
      else elec=0;
      if highyield=. or sz=. or young=. or trdsz=. then delete;
```

SECTION IV: SAS CODE TO CLASSIFY TRADES AS PRINCIPAL VS. NOT PRINCIPAL (AGENCY OR EFFECTIVELY AGENT TRADES)

```
/*Part A: Prepare data for classification algorithm*/

/*1A) Create new variables and a unique identifier for each trade-dealer
combination*/;

data bond.aggtradedata121516jf;set bond.aggtradedata121516jf;
entrd_vol_qt=(100/entrd_pr)*volume_dol;
quant_sgn=entrd_vol_qt*txn;
triddc=_N_;
year=year(TRD_EXCTN_DT);
run;

/*2A) Keep only dealer-years in Constant or Top 70% sample*/
data A1;set bond.dealerthresh051917jf;thresh=1;keep year dc thresh;run;
proc sort data=A1;by year dc;run;
proc sort data=bond.aggtradedata121516jf;by year dc;run;
data threshsamp;merge bond.aggtradedata121516jf (in=a) A1 (in=b);by year
dc;if a;run;

data A1;set bond.dealerconstant051917jf;const=1;keep dc const;run;
proc sort data=A1;by dc;run;
proc sort data=threshsamp;by dc;run;
data bothsamp;merge threshsamp (in=a) A1 (in=b);by dc;if a;run;
data bothsamp;set bothsamp;if thresh=1 or const=1;drop thresh const;run;

/*how many cusips?*/
proc sort data=bothsamp nodupkey out=test;by cusip_id;run;
/*xxx cusips*/

/*3A) Create a unique cusip identifier for all cusips*/
proc sort data=bothsamp;by cusip_id dc descending cust trdtime;run;
data bothsamp;set bothsamp;by cusip_id;retain cusid;if _N_=1 then cusid=1;if
first.cusip_id and _N_ ne 1 then cusid=cusid+1;else cusid=cusid;run;
proc sort data=bothsamp nodupkey out=test;by cusid;run;

/*4A) Create sample of potential matched trades. Exlude trades already signed
as A=Agency by FINRA.*/
/*These trades cannot be matched to trades coded as P=Principal in the
effectively agent algorithm.
Round quantity variable so to accurately pick up buys that are exactly offset
by sells and vice versa (to avoid SAS rounding errors)
Keep only necessary variables*/
data bond.algoJFv1;set bothsamp;
if rsc eq 'P';
quant_sgn=round(quant_sgn,.0001);
keep TRD_EXCTN_DT cusip_id volume_dol entrd_pr trdtime cust txn dc rsc cusid
quant_sgn entrd_vol_qt triddc;run;
proc sort data=bond.algoJFv1 out=temp nodupkey;by cusip_id;run;


/*Part B: Run all customer trades through the algorithm to identify
"effectively agent" vs. principal trades, defined as: Trades are classified
```

```
as 'principal' if not reported as 'Agency' by FINRA and not 'reversed' within
one minute. Trades are classified as 'reversed' when an exact offsetting
quantity (either a customer or interdealer trade) occurs or a combination of
2-3 trades offsets the customer trade within 60 seconds prior or subsequent
to the trade. */


/*NOTE: This is a computationally intensive run. For large data sets (like
this one), run multiple small data sets at the same time.
Below is an example how to cut the data.*/
data bond.pair0to4000;set bond.algoJFv1;where cusid le 4000;run;
data bond.pair4000to8000;set bond.algoJFv1;where cusid le 8000 and cusid ge
4001;run;
data bond.pair8000to12000;set bond.algoJFv1;where cusid le 12000 and cusid ge
8001;run;
data bond.pair12000to16000;set bond.algoJFv1;where cusid le 16000 and cusid
ge 12001;run;
data bond.pair16000to20000;set bond.algoJFv1;where cusid le 20000 and cusid
ge 16001;run;
data bond.pair20000to21000;set bond.algoJFv1;where cusid ge 20001;run;

/*1B) Effectively agent algorithm*/

proc sort data=bond.algoJFv1;by cusip_id dc TRD_EXCTN_DT trdtime;run;

/*Customer trades to be matched*/
data cust (rename=(quant_sgn=cqty trdtime=ctim));set bond.algoJFv1;
keep triddc cusip_id TRD_EXCTN_DT dc quant_sgn trdtime triddc;where cust=1
and rsc = 'P';;run;
data cust;set cust;custid=_N_;run;

/*All trades as potential matches*/
data pair1 (rename=(triddc=ptriddc trdtime=ptrdtime cust=pcust));set
bond.algoJFv1;keep cusip_id TRD_EXCTN_DT dc quant_sgn trdtime cust
triddc;run;

/*2B) Define the total number of observations to send through the algorithm*/
data _null_;
      set cust end=eof;
      if eof then call symput('numcust',custid);run;%put &numcust;
proc delete data=bond.algoJFv1OUT;run;

/*3B) Begin algorithm*/
dm 'log; autoscroll 0';
proc printto log="/home/Bonds/corp/data/Logs/algoJFv1OUT.log";run;
%macro PAIRA;
%do i=1 %to &numcust;

data cust&i;set cust;where custid=&i;run;

data P&i;merge cust&i (in=a) pair1 (in=b);by cusip_id dc TRD_EXCTN_DT;if a
and b;
if triddc ne ptriddc;if abs(ptrdtime-ctim) le 60;run;

data AP2&i;set P&i;rename quant_sgn=qty2 ptriddc=pid2;keep quant_sgn
ptriddc;run;
```

```
data AP3&i;set P&i;rename quant_sgn=qty3 ptriddc=pid3;keep quant_sgn
ptriddc;run;

data _null_;set P&i end=eof;if eof then call symput('num',_N_);run;%put &num;

proc sql noprint;

select count(*) into :obs_count from P&i;

quit;

%if (&obs_count ne 0) %then %do;

PROC SQL; CREATE TABLE pairv1&i AS SELECT * FROM P&i where cqty+quant_sgn=0
and &num=1;quit;

PROC SQL; CREATE TABLE pairv2&i AS SELECT * FROM P&i,AP2&i where ptriddc ne
pid2 and
(cqty+quant_sgn=0 or cqty+qty2=0 or cqty+quant_sgn+qty2=0) and &num=2;quit;

PROC SQL; CREATE TABLE pairv3&i AS SELECT * FROM P&i,AP2&i,AP3&i where
ptriddc ne pid2 and pid2 ne pid3 and ptriddc ne pid3
and (cqty+quant_sgn=0 or cqty+qty2=0 or cqty+qty3=0 or cqty+quant_sgn+qty2=0
or cqty+quant_sgn+qty2+qty3=0 or cqty+qty2+qty3=0 or cqty+quant_sgn+qty3=0)
and &num ge 3;quit;

data pair2&i;set pairv1&i pairv2&i pairv3&i;if &num=1 then PR=1;if &num=2
then PR=2;if &num ge 3 then PR=3;run;

proc append base=bond.algoJFv1OUT (keep = triddc ptriddc pid2 pid3 pcust PR)
data=pair2&i force;run;
%end;

%end;run;%mend PAIRA;
%PAIRA;
dm 'log; autoscroll 1';
proc printto;run;


/*Part C: Merge with Aggregate data sample then create final Top 70% and
Constant Dealer samples*/

/*1C: Stack data from all runs (Important: see Note at top of Part B)*/;
data allnewpair;set
algoJFv1OUT;
run;

/*2C) Create 'Princ' variable*/
proc sort data=allnewpair (keep=triddc pr) nodupkey;by triddc;run;
proc sort data=bond.aggtradedata121516jf;by triddc;run;
data allnewpair2;merge bond.aggtradedata121516jf (in=a) allnewpair (in=b);by
triddc;if a;run;

data allnewpair2;set allnewpair2;
if pr=1 or pr=2 or pr=3 then rscnewJF='A';
if rscnewJF='A' or rsc='A' then princ=0;else princ=1;run;
```

```
/*3C) Merge with mainthreshsamp010217JF and mainconstsamp010217JF*/

data one;set allnewpair2;keep trid dc princ;run;
proc sort data=one;by trid dc;run;
proc sort data=bond.mainthreshsamp010217JF;by trid dc;run;
data bond.mainthreshsamp010217JFv2;merge bond.mainthreshsamp010217JF (in=a)
one (in=b);by trid dc;if a;run;

data one;set allnewpair2;keep trid dc princ;run;
proc sort data=one;by trid dc;run;
proc sort data=bond.mainconstsamp010217JF;by trid dc;run;
data bond.mainconstsamp010217JFv2;merge bond.mainconstsamp010217JF (in=a) one
(in=b);by trid dc;if a;run;
```

SECTION V: SAS CODE TO CREATE CAPITAL MEASURES AT THE DEALER-PORTFOLIO-DAY/WEEK/MONTH AND THE AGGREGATE-DAY/WEEK/MONTH FOR THE TOP 70% SAMPLE

```
/*Part A: Prepare Data and Create Variables for Analysis*/;

/*1A) Create time variables*/
data bond.mainthreshsamp010217JFv2;set bond.mainthreshsamp010217JFv2;
MDY=mdy(month(TRD_EXCTN_DT),1,year(TRD_EXCTN_DT));format MDY MMYYn4.;
TimeToMaturity= (year(maturity)*12 + month(maturity))-(year(offering_date)*12
+ month(offering_date));
BondAge= (trdyrmonth)-(year(offering_date)*12 + month(offering_date));
entrd_vol_qt=(100/entrd_pr)*volume_dol;
quant_sgn=entrd_vol_qt*txn;
volume_dol_sgn=volume_dol*txn;
run;

/*2A) Exclude first month of trade (if day < 16 then start offering month +1,
else offering month +2)*/
data one;set bond.mainthreshsamp010217JFv2;
dayoff=day(offering_date);
if dayoff le 15 then mostart=(INTNX("month",offering_date,1,"b"));
if dayoff gt 15 then mostart=(INTNX("month",offering_date,2,"b"));format
mostart date9.;
MDYst=mdy(month(mostart),1,year(mostart));format MDYst MMYYn4.;
if MDY lt MDYst then delete;
drop mostart MDYst issue_id industry_group;run;
/*how many cusips?*/
proc sort data=one nodupkey out=test;by cusip_id;run;
/*XXX*/

/*3A) Allocate cusips into portfolios each month*/

/*Some firms move from investment grade (IG) to high yield (HY) (or vice
versa) in the month. Drop these cusip-months.*/
proc sort data=one;by cusip_id trdyrmonth;run;
proc means data=one noprint;by cusip_id trdyrmonth;var highyield;
output out=A mean (highyield)=aveHY;run;
data two;merge one A;by cusip_id trdyrmonth;
drop _TYPE_ _FREQ_;
if aveHY gt .5 then HY=1;if aveHY le .5 then HY=0;if highyield=. then HY=.;
if (aveHY=0 or aveHY=1);run;

/*4a) Create 144A indicator*/
data two;set two;if r144a=. then r144a=0;run;

data two;set two;by cusip_id trdyrmonth;
if offering_amt*1000 lt 500000000 then sz=1;
if offering_amt*1000 ge 500000000 and offering_amt*1000 lt 1000000000 then
sz=2;
if offering_amt*1000 ge 1000000000 then sz=3;

if r144a=0 and sz=1 and HY=0 then port=1;
if r144a=0 and sz=2 and HY=0 then port=2;
if r144a=0 and sz=3 and HY=0 then port=3;
```

```
if r144a=0 and sz=1 and HY=1 then port=4;
if r144a=0 and sz=2 and HY=1 then port=5;
if r144a=0 and sz=3 and HY=1 then port=6;

if r144a=1 and HY=0 then port=7;
if r144a=1 and HY=1 then port=8;

if port=. then delete;run;
/*Note we eliminate unrated bonds in this analysis*/

/*Part B: Find Monthly # trades, trade size, trade volume, # block trades,
and block volume for each dealer-month-portfolio*/;

proc sort data=two;by port dc trdyrmonth;run;
/*Define a block trade as $10M or higher*/
data two;set two;N=1;
if (highyield=0 and volume_dol ge 10000000) or (highyield=1 and volume_dol ge
10000000) then block=1;else block=0;
blockvol=block*volume_dol;run;
proc expand data=two out=three method=none;by port dc trdyrmonth;
convert N=numtrddc/transformout=(sum);
convert volume_dol=avetrdszdc/transformout=(cuave);
convert volume_dol=totvoldc/transformout=(sum);
convert block=numblkdc/transformout=(sum);
convert blockvol=blkvoldc/transformout=(sum);run;
data three;set three;by port dc trdyrmonth;if last.port or last.dc or
last.trdyrmonth;run;

/*Part C: Compute Monthly % principal and principal volume for Customer
Trades for each dealer-month-portfolio*/;
data four;set two;if cust=1;
princvol=princ*volume_dol;run;
proc means data=four noprint;by port dc trdyrmonth;var princ;output out=five
mean (princ)=princdc;run;
proc means data=four noprint;by port dc trdyrmonth;var princvol volume_dol;
output out=six sum (princvol volume_dol)=sumprincvol sumvol;run;
data six;set six;princvoldc=sumprincvol/sumvol;run;
data seven;merge six five;by port dc trdyrmonth;run;


/*Part D: Compute Daily and Weekly Capital Measures*/;

/*) Compute daily capital*/

/*1D) Cumulate inventory*/
data onecap;set two;if princ=1;run;
/*Keep only principal trades*/
proc sort data=onecap;by port dc trdyrmonth TRD_EXCTN_DT trdtime trid;run;
data onecap;set onecap;vols=volume_dol_sgn;run;
data onecap;set onecap;
by port dc trdyrmonth TRD_EXCTN_DT;retain daycap;
if first.port or first.dc or first.TRD_EXCTN_DT then daycap=vols;else
daycap=daycap+vols;run;
data onecap;set onecap;absdaycap=abs(daycap);run;

/*2D) Compute time inforce*/
data b21;set onecap;a=hms(trdhr,trdmin,trdsec);time=put(a, time.);run;
```

```
proc sort data=b21;by port dc trdyrmonth TRD_EXCTN_DT descending a descending
trid;run;
data b21;set b21;by port dc trdyrmonth TRD_EXCTN_DT;inforceA=abs(dif(a));run;
data b21;set b21;by port dc trdyrmonth TRD_EXCTN_DT;enda=hms(24,00,00);
if first.port or first.dc or first.TRD_EXCTN_DT then inforce=max((enda-a),0);
sumtimeA=86400;run;
/*86400 is the number of seconds in a day*/
/*The last trade is inforce until the end of the day*/
data b21;set b21;if inforce=. then inforce=inforceA;run;
/*All other trades are inforce until the next trade*/

/*3D) Compute daily time-weighted capital by time inforce*/
data b21;set b21;
waabsdaycapA=absdaycap*(inforce/sumtimeA);run;
proc means data=b21 noprint;by port dc trdyrmonth TRD_EXCTN_DT;var
waabsdaycapA;
output out=suminv sum(waabsdaycapA)=waabsdaycap;run;

/*4D) Compute overnight capital (ending absolute capital for the day)*/
data on;set onecap;by port dc trdyrmonth TRD_EXCTN_DT;if last.port or last.dc
or last.TRD_EXCTN_DT;
ondaycap=absdaycap;keep port dc trdyrmonth TRD_EXCTN_DT ondaycap;run;
data suminv2;merge suminv on;by port dc trdyrmonth TRD_EXCTN_DT;run;

/*5D) Insert days a dealer-portfolio does not trade. Capital will be set to
0.*/

/*Identify set of dealer-portfolios*/
proc sort data=b21 nodupkey out=dc (keep=port dc);by port dc;run;
data dc;set dc;a=1;run;
/*Identify all trading days in the sample*/
proc sort data=b21 nodupkey out=day (keep=TRD_EXCTN_DT trdyrmonth);by
TRD_EXCTN_DT;run;
data day;set day;a=1;run;
proc sql;create table combo as select * from day as a full join dc as b on
a.a eq b.a;quit;
proc sql;create table dcday2 as select * from combo as a left join suminv2 as
b on
a.TRD_EXCTN_DT eq b.TRD_EXCTN_DT and a.dc eq b.dc and a.port eq b.port;quit;
/*This is the complete set of dealer-portfolios for all trading days*/

/*6D) Do not include days before (after) a dealer enters (exits) the sample
for a particular portfolio (as principal or agent).
So if a dealer never trades a particular portfolio, eliminate (i.e., do not
set all days to zero).
Once a dealer starts trading a particular portfolio, then if they have an
inactive month it is set to 0*/
/*Once a dealer stops trading a particular portfolio, eliminate.*/
/*We are looking at the most active dealers, so most dealers are active in
all portfolios throughout the sample period.*/
proc sort data=two nodupkey out=dcsamp (keep=port dc TRD_EXCTN_DT);by port dc
TRD_EXCTN_DT;run;
/*When dealer begins to trade a portfolio*/
data first;set dcsamp;by port dc;if first.port or first.dc;rename
TRD_EXCTN_DT=first;run;
/*When dealer stops trading a portfolio*/
```

```
data last;set dcsamp;by port dc;if last.port or last.dc;rename
TRD_EXCTN_DT=last;run;
proc sort data=dcday2;by port dc;run;
data b23;merge dcday2 (in=x) first (in=y) last (in=z);by port dc;if x;run;
data b23;set b23;if TRD_EXCTN_DT gt last then delete;if TRD_EXCTN_DT lt first
then delete;run;
/*Inactive days are set to zero*/
data b23;set b23;if waabsdaycap=. then do;waabsdaycap=0;ondaycap=0;end;drop
_TYPE_ _FREQ_;drop a;run;


/*Compute weekly capital*/

/*7D) Put days into weekly periods*/
data two;set two;
wkdy=weekday(TRD_EXCTN_DT);
if wkdy ne 7 then weekT=(INTNX("week",TRD_EXCTN_DT,0,"e"));
if wkdy eq 7 then weekT=(INTNX("week",TRD_EXCTN_DT,1,"e"));
format weekT date7.;dayT=weekday(weekT);
week=weekT-1;format week date7.;day=weekday(week);run;
/*7=Saturday. If Saturday trade occurs, put in following week*/
/*Capital is accumulated throughout the week (beginning and ending at
midnight Friday night)*/
/*WeekT is the Saturday AFTER the trade, so we set week=weekT-1 to move to
Friday*/

/*8D) Cumulate absolute weekly capital*/
data onecapwk;set two;if princ=1;run;
/*Keep only principal trades*/
proc sort data=onecapwk;by port dc week TRD_EXCTN_DT trdtime trid;run;
data onecapwk;set onecapwk;vols=volume_dol_sgn;run;
data onecapwk;set onecapwk;
by port dc week;retain weekcap;
if first.port or first.dc or first.week then weekcap=vols;else
weekcap=weekcap+vols;run;
data onecapwk;set onecapwk;absweekcap=abs(weekcap);run;

/*9D) Compute weekend capital*/
proc sort data=onecapwk;by port dc week trdtime;run;
data suminvW2;set onecapwk;by port dc week;if last.port or last.dc or
last.week;
onweekcap=absweekcap;keep port dc week onweekcap;run;

/*10D) Insert week if a dealer-portfolio does not trade. Capital will be set
to 0*/
/*Note this step is similar to 5D and 6D*/
proc sort data=onecapwk nodupkey out=week (keep=week);by week;run;
data week;set week;a=1;run;
proc sql;create table comboW as select * from week as a full join dc as b on
a.a eq b.a;quit;
proc sql;create table dcday2W as select * from comboW as a left join suminvW2
as b on
a.week eq b.week and a.dc eq b.dc and a.port eq b.port;quit;

/*11D) Do not include weeks before (after) a dealer enters (exits) the sample
for a particular portfolio (as principal or agent).*/
```

```
proc sort data=two nodupkey out=dcsampW (keep=port dc week);by port dc
week;run;
/*When dealer begins to trade a portfolio*/
data first;set dcsampW;by port dc;if first.port or first.dc;rename
week=first;run;
/*When dealer stops trading a portfolio*/
data last;set dcsampW;by port dc;if last.port or last.dc;rename
week=last;run;
proc sort data=dcday2W;by port dc;run;
data b23W;merge dcday2W (in=x) first (in=y) last (in=z);by port dc;if x;run;
data b23W;set b23W;if week gt last then delete;if week lt first then
delete;run;
/*Inactive weeks are set to zero*/
data b23W;set b23W;if onweekcap=. then do;onweekcap=0;end;drop a;run;


/*Part E: Compute daily/weekly aggregate volume for all cusips and dealers
for each portfolio*/
/*If aggregate volume is zero, then delete observation (capital should not be
set to 0)*/

/*1E) Start with Aggregate market data, eliminate trading around initial
offering, and create time variables*/
data oneB;set bond.aggtradedata121516JF;keep TRD_EXCTN_DT cusip_id dc
trdyrmonth highyield offering_amt offering_date trid volume_dol r144a;
if r144a=. then r144a=0;run;
data oneB;set oneB;MDY=mdy(month(TRD_EXCTN_DT),1,year(TRD_EXCTN_DT));format
MDY MMYYn4.;
dayoff=day(offering_date);
if dayoff le 15 then mostart=(INTNX("month",offering_date,1,"b"));
if dayoff gt 15 then mostart=(INTNX("month",offering_date,2,"b"));format
mostart date9.;
MDYst=mdy(month(mostart),1,year(mostart));format MDYst MMYYn4.;
if MDY lt MDYst then delete; drop mostart MDYst;
wkdy=weekday(TRD_EXCTN_DT);
if wkdy ne 7 then weekT=(INTNX("week",TRD_EXCTN_DT,0,"e"));
if wkdy eq 7 then weekT=(INTNX("week",TRD_EXCTN_DT,1,"e"));
format weekT date7.;week=weekT-1;format week date7.;run;

/*2E) Allocate into portfolios*/
proc sort data=oneB;by cusip_id trdyrmonth;run;
proc means data=oneB noprint;by cusip_id trdyrmonth;var highyield;
output out=Ab mean (highyield)=aveHY;run;
data oneB;merge oneB Ab;by cusip_id trdyrmonth;
drop _TYPE_ _FREQ_;if aveHY gt .5 then HY=1;if aveHY le .5 then HY=0;if
highyield=. then HY=.;
if (aveHY=0 or aveHY=1);run;
data oneB;set oneB;by cusip_id trdyrmonth;
if offering_amt*1000 lt 500000000 then sz=1;
if offering_amt*1000 ge 500000000 and offering_amt*1000 lt 1000000000 then
sz=2;
if offering_amt*1000 ge 1000000000 then sz=3;
if r144a=0 and sz=1 and HY=0 then port=1;
if r144a=0 and sz=2 and HY=0 then port=2;
if r144a=0 and sz=3 and HY=0 then port=3;
if r144a=0 and sz=1 and HY=1 then port=4;
if r144a=0 and sz=2 and HY=1 then port=5;
```

```
if r144a=0 and sz=3 and HY=1 then port=6;
if r144a=1 and HY=0 then port=7;
if r144a=1 and HY=1 then port=8;
if port=. then delete;run;

/*3E) Daily aggregate volume*/
proc sort data=oneB nodupkey;by TRD_EXCTN_DT trid;run;
proc sort data=oneB;by port TRD_EXCTN_DT;run;
proc means data=oneB noprint;by port TRD_EXCTN_DT;output out=aggvol sum
(volume_dol)=aggvol;run;
proc sort data=b23;by port TRD_EXCTN_DT;run;
data b24;merge b23 (in=x) aggvol (in=y);by port TRD_EXCTN_DT;if x;run;
/*If no aggregate volume for a portfolio on that day then delete*/
data b24;set b24;if aggvol=. then delete;run;

/*4E) Weekly aggregate volume*/
proc sort data=oneB nodupkey;by week trid;run;
proc sort data=oneB;by port week;run;
proc means data=oneB noprint;by port week;output out=aggvol sum
(volume_dol)=aggvol;run;
proc sort data=b23W;by port week;run;
data b24W;merge b23W (in=x) aggvol (in=y);by port week;if x;run;
/*If no volume for a portfolio on that week then delete*/
data b24W;set b24W;if aggvol=. then delete;run;


/*Part F: Compute total amount outstanding for each day*/;

/*1F) Observation for each cusip for all dates in sample*/
proc sort data=two nodupkey out=cus (keep=cusip_id offering_date offering_amt
enddt);by cusip_id;run;
data cus;set cus;a=1;run;
proc sort data=two nodupkey out=mo (keep=trdyrmonth TRD_EXCTN_DT mdy);by
TRD_EXCTN_DT;run;
data mo;set mo;a=1;run;
proc sql;create table combo as select * from mo as a full join cus as b on
a.a eq b.a;quit;

/*2F) Get portfolio data for each month*/
data cus2;set two;keep cusip_id trdyrmonth port;run;
proc sort data=cus2 nodupkey;by cusip_id trdyrmonth;run;
proc sql;create table combo2 as select * from combo as a left join cus2 as b
on
a.cusip_id eq b.cusip_id and a.trdyrmonth eq b.trdyrmonth;quit;

/*3F) Days around offering are not in the sample*/
data combo2;set combo2;
dayoff=day(offering_date);
if dayoff le 15 then mostart=(INTNX("month",offering_date,1,"b"));
if dayoff gt 15 then mostart=(INTNX("month",offering_date,2,"b"));format
mostart date9.;
MDYst=mdy(month(mostart),1,year(mostart));format MDYst MMYYn4.;
if MDY lt MDYst then delete;
MDYend=mdy(month(enddt),1,year(enddt));format MDYend MMYYn4.;if MDY gt MDYend
then delete;
drop mostart MDYst MDYend;run;
```

```
/*4F) For no-trade months, retain portfolio from previous month*/
proc sort data=combo2;by cusip_id mdy;run;
data combo2;set combo2;by cusip_id;retain port2;
if first.cusip_id then do;port2=port;end;
else do;
if port ne . then port2=port;
else port2=port2;end;run;
proc sort data=combo2;by cusip_id descending mdy;run;
data combo3;set combo2;by cusip_id;retain port3;
if first.cusip_id then do;port3=port2;end;
else do;
if port2 ne . then port3=port2;
else port3=port3;end;run;
proc sort data=combo3;by cusip_id mdy;run;
data os;set combo3;keep mdy port3 cusip_id offering_date enddt offering_amt
TRD_EXCTN_DT;run;

/*5F) Account for changes to the amount outstanding after the offering date*/
proc sort data=two out=cus;by cusip_id TRD_EXCTN_DT finos;run;
proc sort data=cus (keep=cusip_id finos TRD_EXCTN_DT) nodupkey;by cusip_id
finos;run;
data cus;set cus;rename TRD_EXCTN_DT=dtchg;run;
proc sql;create table os2 as select * from os as a left join cus as b on
a.cusip_id eq b.cusip_id and a.TRD_EXCTN_DT ge b.dtchg;quit;
proc sort data=os2;by cusip_id TRD_EXCTN_DT descending dtchg;run;
proc sort data=os2 nodupkey;by cusip_id TRD_EXCTN_DT;run;
data os2;set os2;if finos=. then finos=offering_amt;run;


/*Part G: Prepare dealer-portfolio-month data from Part B and Part C*/

/*1G) Merge monthly files together and create average portfolio-dealer-month
issue size and bond age variables. These two variables are independent
variables in Table IA.II (Internet Appendix)*/
proc sort data=three;by port dc trdyrmonth;run;
proc sort data=seven;by port dc trdyrmonth;run;
data three;set three;keep port dc trdyrmonth numtrddc avetrdszdc totvoldc
numblkdc blkvoldc;run;
proc sort data=two;by port dc trdyrmonth;run;
proc means data=two noprint;by port dc trdyrmonth;var offering_amt bondage;
output out=aves mean (offering_amt bondage)=aveiss aveage;run;
data monthlyTS;merge three seven aves;by port dc trdyrmonth;if port=. then
delete;run;

/*2G) Add a month for each dealer (even if do not trade in a particular month
in a particular portfolio)*/;
proc sort data=monthlyTS nodupkey out=dc (keep=port dc);by port dc;run;
data dc;set dc;a=1;run;
proc sort data=monthlyTS nodupkey out=day (keep=trdyrmonth);by
trdyrmonth;run;
data day;set day;a=1;run;
proc sql;create table combo as select * from day as a full join dc as b on
a.a eq b.a;quit;
proc sql;create table monthlyTSv2 as select * from combo as a left join
monthlyTS as b on
a.trdyrmonth eq b.trdyrmonth and a.dc eq b.dc and a.port eq b.port;quit;
```

```
/*3G) Do not include months before dealer enters (or after dealer exits) the
sample*/
proc sort data=monthlyTS nodupkey out=dcsamp (keep=port dc trdyrmonth);by
port dc trdyrmonth;run;
data first;set dcsamp;by port dc;if first.port or first.dc;rename
trdyrmonth=first;run;
data last;set dcsamp;by port dc;if last.port or last.dc;rename
trdyrmonth=last;run;
proc sort data=monthlyTSv2;by port dc;run;
data monthlyTSv3;merge monthlyTSv2 (in=x) first (in=y) last (in=z);by port
dc;if x;run;
data monthlyTSv4;set monthlyTSv3;if trdyrmonth gt last then delete;if
trdyrmonth lt first then delete;run;

/*4G) Assign values to no trade months*/
data monthlyTSv4;set monthlyTSv4;
if numtrddc=. then do;
numtrddc=0;
totvoldc=0;
numblkdc=0;
blkvoldc=0;
sumprincvol=0;end;
drop _TYPE_ _FREQ_ first last a;
if sumprincvol=. then do;
sumprincvol=0;end;run;

/*5G) For no-trade months, set average portfolio-dealer-month issue size and
bond age equal to the previous trade month (variables for Internet
Appendix)*/
proc sort data=monthlyTSv4;by port dc trdyrmonth;run;
data monthlyTSv5;set monthlyTSv4;retain aveiss2 aveage2;by port dc;
if first.port or first.dc then do;
aveiss2=aveiss;end;
else do; if aveiss ne . then aveiss2=aveiss;
else aveiss2=aveiss2;end;
if first.port or first.dc then do;
aveage2=aveage;end;
else do; if aveage ne . then aveage2=aveage;
else aveage2=aveage2+1;end;run;
data monthlyTSv5;set monthlyTSv5;
drop sumvol aveiss aveage;run;

/*6G) Compute total amount outstanding for each portfolio-month in order to
compute the variable 'Dollar Volume / Amount Out.'*/
proc sort data=os2 out=os4 nodupkey;by mdy port3 cusip_id;run;
/*Sum over all cusips for each portfolio-month*/
proc means data=os4 noprint;by mdy port3;var finos;
output out=mdyos sum(finos)=totOSmdy;run;
data mdyos;set mdyos;rename port3=port;drop _TYPE_ _FREQ_;run;
data temp;set two;keep mdy trdyrmonth;run;
proc sort data=temp nodupkey;by trdyrmonth mdy;run;
proc sort data=monthlyTSv5;by trdyrmonth;run;
data monthlyTSv5T;merge monthlyTSv5 (in=x) temp (in=y);by trdyrmonth;if
x;run;
proc sort data=monthlyTSv5T;by port mdy;run;
proc sort data=mdyos;by port mdy;run;
data monthlyTSv6;merge monthlyTSv5T (in=x) mdyos (in=y);by port mdy;if x;
```

```
voltoiss=totvoldc/totOSmdy;run;


/*Part H: Prepare dealer-portfolio-day capital data from Part D*/

/*1H) Create average portfolio-dealer-day issue size and bond age variables.
These two variables are independent variables in Table IA.II (Internet
Appendix)*/
data dailyTS1;set b24;drop first last;if port=. then delete;run;
proc sort data=two;by port dc trd_exctn_dt;run;
proc means data=two noprint;by port dc trd_exctn_dt;var offering_amt bondage;
output out=avesDS mean (offering_amt bondage)=aveiss aveage;run;
proc sort data=dailyTS1;by port dc trd_exctn_dt;run;
data dailyTS2;merge dailyTS1 (in=x) avesDS (in=y);by port dc trd_exctn_dt;if
x;run;

/*2H) For no-trade days, set average portfolio-dealer-day issue size and bond
age equal to the previous trade day (variables for Internet Appendix)*/
proc sort data=dailyTS2;by port dc trd_exctn_dt;run;
data dailyTS2;set dailyTS2;retain aveiss2 aveage2;by port dc;
if first.port or first.dc then do;
aveiss2=aveiss;end;
else do; if aveiss ne . then aveiss2=aveiss;
else aveiss2=aveiss2;end;
if first.port or first.dc then do;
aveage2=aveage;end;
else do; if aveage ne . then aveage2=aveage;
else aveage2=aveage2+.04;end;run;
/*Age is in months, so add .04 to average age if retained from previous day*/
data dailyTS2;set dailyTS2;drop aveiss aveage;run;

/*3H) Remove holidays (from SIFMA site) and Sunday from daily data.*/
data dailyTS2;set dailyTS2;yr=year(TRD_EXCTN_DT);
if TRD_EXCTN_DT=holiday('VETERANS',yr) then delete;
if TRD_EXCTN_DT=holiday('NEWYEAR',yr) then delete;
if TRD_EXCTN_DT=holiday('MLK',yr) then delete;
if TRD_EXCTN_DT=holiday('USPRESIDENTS',yr) then delete;
if TRD_EXCTN_DT=holiday('MEMORIAL',yr) then delete;
if TRD_EXCTN_DT=holiday('USINDEPENDENCE',yr) then delete;
if TRD_EXCTN_DT=holiday('LABOR',yr) then delete;
if TRD_EXCTN_DT=holiday('COLUMBUS',yr) then delete;
if TRD_EXCTN_DT=holiday('THANKSGIVING',yr) then delete;
if TRD_EXCTN_DT=holiday('CHRISTMAS',yr) then delete;
if weekday(TRD_EXCTN_DT) = 1 then delete;run;


/*Part I: Prepare dealer-portfolio-week capital data from Part D*/;

/*1I) Create average portfolio-dealer-week issue size and bond age variables.
These two variables are independent variables in Table IA.II (Internet
Appendix)*/
data weeklyTS1;set b24w;drop first last;if port=. then delete;run;
proc sort data=two;by port dc week;run;
proc means data=two noprint;by port dc week;var offering_amt bondage;
output out=aveswk mean (offering_amt bondage)=aveiss aveage;run;
proc sort data=weeklyTS1;by port dc week;run;
data weeklyTS2;merge weeklyTS1 (in=x) aveswk (in=y);by port dc week;if x;run;
```

```
/*2I) For no-trade weeks, set average portfolio-dealer-week issue size and
bond age equal to the previous trade week (variables for Internet Appendix)*/
proc sort data=weeklyTS2;by port dc week;run;
data weeklyTS2;set weeklyTS2;retain aveiss2 aveage2;by port dc;
if first.port or first.dc then do;
aveiss2=aveiss;end;
else do; if aveiss ne . then aveiss2=aveiss;
else aveiss2=aveiss2;end;
if first.port or first.dc then do;
aveage2=aveage;end;
else do; if aveage ne . then aveage2=aveage;
else aveage2=aveage2+.25;end;run;
/*Age is in months, so add .25 to average age if retained from previous
week*/
data weeklyTS2;set weeklyTS2;drop aveiss aveage;run;


/*Part J) Link to monthly control file with Corporate Bond Index Return,
Stock Market Index Return, Chg. in VIX, Chg. in 3-Month Libor, the absolute
value of the aggregate flows into or out of investment grade corporate bond
mutual funds and ETFs scaled by the prior month total net asset value, and %
Retail Volume. Controls are based on the month prior to the observation*/

proc sql;create table monthlyTSv7 as select * from monthlyTSv6 as a left join
bond.controlC as b on
a.mdy eq b.mdyp1;quit;
data dailyTS2;set
dailyTS2;MDY=mdy(month(TRD_EXCTN_DT),1,year(TRD_EXCTN_DT));format MDY
MMYYn4.;run;
proc sql;create table dailyTS3 as select * from dailyTS2 as a left join
bond.controlC as b on
a.mdy eq b.mdyp1;quit;
data weeklyTS2;set weeklyTS2;MDY=mdy(month(week),1,year(week));format MDY
MMYYn4.;
proc sql;create table weeklyTS3 as select * from weeklyTS2 as a left join
bond.controlC as b on
a.mdy eq b.mdyp1;quit;


/*/Part K: Remove years that a dealer is not in the Top 70% for daily,
weekly, and monthly data*/;

/*1K) Daily data*/
data A1;set bond.dealerthresh051917jf;thresh=1;keep year dc thresh;run;
proc sort data=A1;by year dc;run;
data dailyTS3;set dailyTS3;year=year(TRD_EXCTN_DT);run;
proc sort data=dailyTS3;by year dc;run;
data dailyTS4;merge dailyTS3 (in=a) A1 (in=b);by year dc;if a;run;
data dailyTS4;set dailyTS4;if thresh=1;run;

/*2K) Monthly data*/
data monthlyTSv7;set monthlyTSv7;year=year(mdy);run;
proc sort data=monthlyTSv7;by year dc;run;
data monthlyTSv8;merge monthlyTSv7 (in=a) A1 (in=b);by year dc;if a;run;
data monthlyTSv8;set monthlyTSv8;if thresh=1;run;
```

```
/*3K) Weekly data*/
data weeklyTS3;set weeklyTS3;year=year(mdy);run;
proc sort data=weeklyTS3;by year dc;run;
data weeklyTS4;merge weeklyTS3 (in=a) A1 (in=b);by year dc;if a;run;
data weeklyTS4;set weeklyTS4;if thresh=1;run;


/*Part L: Create Appendix Table II and III variable 'ABS (MF+ETF Flows (t-1)
/ Tot. Out. (t-2))'. If high yield portfolio, define by high yield data,
otherwise investment grade data.*/*/*/;
data monthlyTSv8; set monthlyTSv8;
if port in (1,2,3,7) then MF_ETFflow=MF_ETF_FLOW_HG;else
MF_ETFflow=MF_ETF_FLOW_HY;
if port in (1,2,3,7) then MF_ETFflowSC=MF_ETF_DFLOW__HG; else
MF_ETFflowSC=MF_ETF_DFLOW__HY;
if port in (1,2,3,7) then MAshare=MA_HG_MS;else MAshare=MA_HY_MS;
if port in (1,2,3,7) then retshare=pctretig;else retshare=pctrethy;run;
data monthlyTSv8; set monthlyTSv8;absMF_ETFflowSC=abs(MF_ETFflowSC);run;

data dailyTS4; set dailyTS4;
if port in (1,2,3,7) then MF_ETFflow=MF_ETF_FLOW_HG;else
MF_ETFflow=MF_ETF_FLOW_HY;
if port in (1,2,3,7) then MF_ETFflowSC=MF_ETF_DFLOW__HG; else
MF_ETFflowSC=MF_ETF_DFLOW__HY;
if port in (1,2,3,7) then MAshare=MA_HG_MS;else MAshare=MA_HY_MS;
if port in (1,2,3,7) then retshare=pctretig;else retshare=pctrethy;run;
data dailyTS4; set dailyTS4;absMF_ETFflowSC=abs(MF_ETFflowSC);run;

data weeklyTS4; set weeklyTS4;
if port in (1,2,3,7) then MF_ETFflow=MF_ETF_FLOW_HG;else
MF_ETFflow=MF_ETF_FLOW_HY;
if port in (1,2,3,7) then MF_ETFflowSC=MF_ETF_DFLOW__HG; else
MF_ETFflowSC=MF_ETF_DFLOW__HY;
if port in (1,2,3,7) then MAshare=MA_HG_MS;else MAshare=MA_HY_MS;
if port in (1,2,3,7) then retshare=pctretig;else retshare=pctrethy;run;
data weeklyTS4; set weeklyTS4;absMF_ETFflowSC=abs(MF_ETFflowSC);run;


/*Part M: Create time period indicators*/

data monthlyTSv8;set monthlyTSv8;mo=month(mdy);yr=year(mdy);
if (yr=2006) or (yr=2007 and mo le 6) then pre2=1;else pre2=0;
if (yr=2007 and mo ge 7) or (yr=2008) or (yr=2009 and mo le 4) then
crisis2=1;else crisis2=0;
if (yr=2009 and mo ge 5) or (yr=2010 and mo le 6) then post2=1;else post2=0;
if (yr=2010 and mo ge 7) or (yr=2011) or (yr=2012) or (yr=2013) or (yr=2014
and mo le 3) then reg2=1;else reg2=0;
if (yr=2014 and mo ge 4) or (yr in (2015,2016)) then volc2=1;else
volc2=0;run;

data dailyTS4;set dailyTS4;mo=month(mdy);
if (yr=2006) or (yr=2007 and mo le 6) then pre2=1;else pre2=0;
if (yr=2007 and mo ge 7) or (yr=2008) or (yr=2009 and mo le 4) then
crisis2=1;else crisis2=0;
if (yr=2009 and mo ge 5) or (yr=2010 and mo le 6) then post2=1;else post2=0;
if (yr=2010 and mo ge 7) or (yr=2011) or (yr=2012) or (yr=2013) or (yr=2014
and mo le 3) then reg2=1;else reg2=0;
```

```
if (yr=2014 and mo ge 4) or (yr in (2015,2016)) then volc2=1;else
volc2=0;drop _TYPE_ _FREQ_;run;

data weeklyTS4;set weeklyTS4;mo=month(mdy);yr=year(mdy);
if (yr=2006) or (yr=2007 and mo le 6) then pre2=1;else pre2=0;
if (yr=2007 and mo ge 7) or (yr=2008) or (yr=2009 and mo le 4) then
crisis2=1;else crisis2=0;
if (yr=2009 and mo ge 5) or (yr=2010 and mo le 6) then post2=1;else post2=0;
if (yr=2010 and mo ge 7) or (yr=2011) or (yr=2012) or (yr=2013) or (yr=2014
and mo le 3) then reg2=1;else reg2=0;
if (yr=2014 and mo ge 4) or (yr in (2015,2016)) then volc2=1;else
volc2=0;drop _TYPE_ _FREQ_;run;


/*/Part N) Pull total volume for each portfolio-dealer-day and portfolio-
dealer-week, to create a scaled capital measure (capital/volume)*/;

/*1N) Exclude first month of trade (if day < 16 then start offering month +1,
else offering month +2)*/
data one;set bond.mainthreshsamp010217JFv2;dayoff=day(offering_date);
if dayoff le 15 then mostart=(INTNX("month",offering_date,1,"b"));
if dayoff gt 15 then mostart=(INTNX("month",offering_date,2,"b"));format
mostart date9.;
MDYst=mdy(month(mostart),1,year(mostart));format MDYst MMYYn4.;
if MDY lt MDYst then delete;
drop mostart MDYst issue_id industry_group;run;

/*2N) Allocate cusips into groups*/
proc sort data=one;by cusip_id trdyrmonth;run;
proc means data=one noprint;by cusip_id trdyrmonth;var highyield;
output out=A mean (highyield)=aveHY;run;
data two;merge one A;by cusip_id trdyrmonth;
drop _TYPE_ _FREQ_;
if aveHY gt .5 then HY=1;if aveHY le .5 then HY=0;if highyield=. then HY=.;
if (aveHY=0 or aveHY=1);run;
/*Create 144A indicator*/
data two;set two;if r144a=. then r144a=0;run;
data two;set two;by cusip_id trdyrmonth;
if offering_amt*1000 lt 500000000 then sz=1;
if offering_amt*1000 ge 500000000 and offering_amt*1000 lt 1000000000 then
sz=2;
if offering_amt*1000 ge 1000000000 then sz=3;
if r144a=0 and sz=1 and HY=0 then port=1;
if r144a=0 and sz=2 and HY=0 then port=2;
if r144a=0 and sz=3 and HY=0 then port=3;
if r144a=0 and sz=1 and HY=1 then port=4;
if r144a=0 and sz=2 and HY=1 then port=5;
if r144a=0 and sz=3 and HY=1 then port=6;
if r144a=1 and HY=0 then port=7;
if r144a=1 and HY=1 then port=8;
if port=. then delete;run;

/*3N) Find volume for each dealer-day-portfolio*/
proc sort data=two;by port dc TRD_EXCTN_DT;run;
proc expand data=two out=bond.dcportdayvolTHjf method=none;by port dc
TRD_EXCTN_DT;
convert volume_dol=totvoldcportday/transformout=(sum);run;
```

```
data bond.dcportdayvolTHjf;set bond.dcportdayvolTHjf;by port dc
TRD_EXCTN_DT;if last.port or last.dc or last.TRD_EXCTN_DT;keep port dc
TRD_EXCTN_DT totvoldcportday; run;

/*4N) Find volume for each dealer-week-portfolio*/
data two;set two;wkdy=weekday(TRD_EXCTN_DT);
if wkdy ne 7 then weekT=(INTNX("week",TRD_EXCTN_DT,0,"e"));
if wkdy eq 7 then weekT=(INTNX("week",TRD_EXCTN_DT,1,"e"));
format weekT date7.;dayT=weekday(weekT);
week=weekT-1;format week date7.;day=weekday(week);run;
proc sort data=two;by port dc week;run;
proc means data=two noprint;by port dc week;output out=bond.dcportwkvolTHjf
sum (volume_dol)=totvoldcportwk;run;

/*5N) Merge volume file with daily data*/
proc sort data=dailyTS4;by port dc TRD_EXCTN_DT;run;
proc sort data=bond.dcportdayvolTHjf;by port dc TRD_EXCTN_DT;run;
data bond.dailyTS4jf;merge dailyTS4 (in=a) bond.dcportdayvolTHjf (in=b);by
port dc TRD_EXCTN_DT;if a;run;
data bond.dailyTS4jf;set bond.dailyTS4jf;
pctwaabsdaycap=waabsdaycap/totvoldcportday;
pctondaycap=ondaycap/totvoldcportday;
if pctwaabsdaycap=. then pctwaabsdaycap=0;
if pctondaycap=. then pctondaycap=0;
if totvoldcportday=. then totvoldcportday=0;run;

/*6N) Merge volume file with weekly data*/
proc sort data=weeklyTS4;by port dc week;run;
proc sort data=bond.dcportwkvolTHjf;by port dc week;run;
data bond.weeklyTS4jf;merge weeklyTS4 (in=a) bond.dcportwkvolTHjf (in=b);by
port dc week;if a;run;
data bond.weeklyTS4jf;set bond.weeklyTS4jf;
pctonweekcap=onweekcap/totvoldcportwk;
if pctonweekcap=. then pctonweekcap=0;
if totvoldcportwk=. then totvoldcportwk=0;run;

/*/Part O: This is the final step of the creation of the portfolio-dealer-
day, portfolio-dealer-week, portfolio-dealer-month data. This data will be
used in Appendix Table II and III for the portfolio level regressions. This
data will also be used to create the aggregate daily, weekly, and monthly
data used in the main tables.*/;

/*/*/*Take data to Stata. */*/*/;
proc export data=bond.dailyTS4jf outfile="D:\Temp Bond\STATA\dailyTS4jf.csv"
dbms=csv replace;run;
proc export data=bond.weeklyTS4jf outfile="D:\Temp
Bond\STATA\weeklyTS4jf.csv" dbms=csv replace;run;
data bond.monthlyTSv8jf;set monthlyTSv8;pctblock=blkvoldc/totvoldc;run;
proc export data=bond.monthlyTSv8jf outfile="D:\Temp
Bond\STATA\monthlyTSv8jf.csv" dbms=csv replace;run;


/*/*/*Part P: Now we aggregate the portfolio-dealer-day/week data to the
aggregate day, and aggregate week level*/*/*/;

/*1P) Create DAILY aggregated data - one observation for each day*/
data one;set bond.dailyTS4jf;run;
```

```
proc sort data=one;by TRD_EXCTN_DT;run;
/*Sum time-weighted, overnight capital, and total volume*/
proc means data=one noprint;by TRD_EXCTN_DT;
output out=two sum (waabsdaycap ondaycap totvoldcportday)
=waabsdaycap ondaycap totvoldcportday;run;
/*Create scaled time-weighted and overnight variables*/
data two;set two;
waabsdaycapscvol=waabsdaycap/totvoldcportday;
ondaycapscvol=ondaycap/totvoldcportday;run;
/*Link to market control data (index returns, changes to VIX, Libor, mutual
fund/ETF flows, retail market share*/
/*The controls for the aggregated data will be based on investment grade
mutual fund/ETF flows*/
data three;set one;if port in (1,2,3,7);keep TRD_EXCTN_DT pre2 crisis2 post2
reg2 volc2 bondret stockret chgvix pctchglibor absmf_etfflowsc retshare;run;
proc sort data=three nodupkey;by TRD_EXCTN_DT;run;
data bond.dailyagg050117jf;merge two three;by TRD_EXCTN_DT;run;
/*Send to Stata*/
proc export data=bond.dailyagg050117jf outfile="D:\Temp
Bond\STATA\dailyagg050117jf.csv" dbms=csv replace;run;

/* 2P) Create WEEKLY aggregated data - one observation for each week*/
data one;set bond.weeklyTS4jf;run;
proc sort data=one;by week;run;
/*Sum time-weighted, overnight capital, and total volume*/
proc means data=one noprint;by week;
output out=two sum (onweekcap totvoldcportwk)
=onweekcap totvoldcportwk;run;
/*Create scaled time-weighted and overnight variables*/
data two;set two;
onweekcapscvol=onweekcap/totvoldcportwk;run;
/*Link to market control data (index returns, changes to VIX, Libor, mutual
fund/ETF flows, retail market share*/
/*The controls for the aggregated data will be based on investment grade
mutual fund/ETF flows*/
data three;set one;if port in (1,2,3,7);keep week pre2 crisis2 post2 reg2
volc2 bondret stockret chgvix pctchglibor absmf_etfflowsc retshare;run;
proc sort data=three nodupkey;by week;run;
data bond.weeklyagg050117jf;merge two three;by week;run;
/*Send to Stata*/
proc export data=bond.weeklyagg050117jf outfile="D:\Temp
Bond\STATA\weeklyagg050117jf.csv" dbms=csv replace;run;


/*Part Q: Create aggregated monthly data*/;

/*1Q) Exclude first month of trade (if day < 16 then start offering month +1,
else offering month +2)*/
data one;set bond.mainthreshsamp010217JFv2;dayoff=day(offering_date);
if dayoff le 15 then mostart=(INTNX("month",offering_date,1,"b"));
if dayoff gt 15 then mostart=(INTNX("month",offering_date,2,"b"));format
mostart date9.;
MDYst=mdy(month(mostart),1,year(mostart));format MDYst MMYYn4.;
if MDY lt MDYst then delete;
drop mostart MDYst issue_id industry_group;;run;

/*2Q) Find trade volume and average trade size for each month*/
```

```
proc sort data=one nodupkey out=oneadj;by trdyrmonth trid;run;
proc expand data=oneadj out=two method=none;by trdyrmonth;
convert volume_dol=avetrdszdc/transformout=(cuave);
convert volume_dol=totvoldc/transformout=(sum);run;
data two;set two;by trdyrmonth;if last.trdyrmonth;run;

/*3Q) Compute principal % and volume*/
data three;set one;if cust=1;princvol=princ*volume_dol;run;
proc sort data=three;by trdyrmonth;run;
proc means data=three noprint;by trdyrmonth;var princvol volume_dol;
output out=four sum (princvol volume_dol)=sumprincvol sumvol;run;
data four;set four;princvoldc=sumprincvol/sumvol;run;

*4Q) Get total amount outstanding for each month*/;
proc sort data=one out=os1 nodupkey;by trdyrmonth cusip_id;run;
proc means data=os1 noprint;by trdyrmonth;var finos;
output out=os2 sum(finos)=totOSmdy;run;

/*5Q) Merge files*/
data aggmo050117;merge two four os2;by trdyrmonth;run;
data aggmo050117;set aggmo050117;voltoiss=totvoldc/totOSmdy;run;

/*6Q) Get controls*/
data five;set bond.monthlyTSv8jf;if port in (1,2,3,7);keep trdyrmonth pre2
crisis2 post2 reg2 volc2 bondret stockret chgvix pctchglibor absmf_etfflowsc
retshare;run;
proc sort data=five nodupkey;by trdyrmonth;run;
data bond.aggmo050117v2jf;merge five aggmo050117;by trdyrmonth;run;
/*Send to Stata*/
proc export data=bond.aggmo050117v2jf outfile="D:\Temp
Bond\STATA\aggmo050117v2jf.csv" dbms=csv replace;run;
```