SAS and Stata Code

Evidence on the Use and Efficacy of Internal Whistleblowing Systems

By Stephen R. Stubben and Kyle T. Welch

March 2020

OVERVIEW

Our study uses four programs to access data and prepare the tables that appear in the published paper.

1. getdata.sas

This SAS program accesses WRDS to create many of the variables that we use in the study. It combines accounting data from Compustat, market data from CRSP, institutional ownership data from Thomson Financial, director data from BoardEx, and litigation and internal control data from AuditAnalytics. It saves the final dataset as a Stata .dta file, which we merge to the internal whistleblowing data using "prepare.do".

2. prepare.do

This Stata program accesses the internal whistleblowing data provided by NAVEX Global and creates a number of variables that we use in the study. It merges the whistleblowing data with all our other data, then creates two output files: "navdata_reports.dta" and "navdata_firmyear.dta". We use the former file in our report-level analyses presented in Tables 1 and 2, and we use the latter file in our firm-year analyses presented in Tables 3 through 5.

3. tables12.do

The Stata program uses "navdata_reports.dta" to create Tables 1 and 2 of the published paper.

4. tables345.do

The Stata program uses "navdata_firmyear.dta" to create Tables 3 through 5 of the published paper.

```
/*************************************************************************
***** getdata.sas
*****
***** SAS program to access data from WRDS
***** Last revised 8/3/2019
*************************************************************************/


%include 'winsorize.sas';
libname seg '/wrds/comp/sasdata/seghist';
libname aa  ('/wrds/audit/sasdata/corp_legal' '/wrds/audit/sasdata/audit_comp');
libname tfn '/wrds/tfn/sasdata/s34';
libname boardex '/wrds/boardex/sasdata/na';


*** Compustat data;

data swdata;
    set comp.funda(keep=cusip cik gvkey datadate fyear sich at sale csho emp scf ibc
                   indfmt datafmt popsrc consol curcd);
    where (datadate between '01JAN2002'd and '31DEC2018'd) and (indfmt='INDL') and (datafmt='STD')
           and (popsrc='D') and (consol='C') and (curcd='USD') and (at > 0) and (sale > 0) and (scf = 7);
    cusip6 = substr(cusip,1,6); * create six-digit cusip for merge with Thomson Financial;
    ngvkey = 1 * gvkey;         * create numeric gvkey for merge with whistleblowing data;
    ncik = 1 * cik;             * create numeric cik for merge with Boardex;
    roa = ibc / at;             * previous draft used accrual measure, still using earnings from SCF;

    drop indfmt datafmt popsrc consol curcd cusip ibc scf;
    rename sich = sic;
run;

* Add GICS industry codes;
proc sql undo_policy=none;
    create table swdata as select a.*, b.ggroup as ind
    from swdata a left join comp.company b
    on (a.gvkey = b.gvkey)
    order by gvkey, datadate;
quit;

* Calculate lags and changes;
data swdata;
    set swdata;
    if gvkey = lag(gvkey) and intck('MONTH', lag(datadate), datadate) = 12 then m1 = 1; else m1 = .;
    atm1   = lag(at)   * m1;
    saled1 = dif(sale) * m1;
    salem1 = lag(sale) * m1;
    if (salem1 > 0) then growth = saled1 / salem1; else growth = .;

    * Inputs for KS (2012) litigation risk;
    if (atm1 > 0) then ks_growth = saled1 / atm1; else ks_growth = .;
    cshom1 = lag(csho) * m1;
    ks_size = log(at);

    drop salem1 saled1 atm1 m1;
run;


*** Segment data to calculate geographic dispersion;

proc sql undo_policy=none;
    create table seg as select gvkey, datadate, sid, stype, sales
    from seg.seg_annfund(where=(datadate >= '01JAN2002'd))
    group by gvkey, datadate, stype
    having srcdate = max(srcdate);

    * Drop eliminations (sid=99) and corporate segment (sales=0);
    create table geoseg as select gvkey, datadate, sum(sales) as tsales, sales / calculated tsales as pct
    from seg(where=(stype='GEOSEG' and sid ~= 99 and sales > 0))
    group by gvkey, datadate;

    * Calculate HH index based on revenues across geographic segments;
    create table geoseg as select distinct gvkey, datadate, n(pct) as n, sum(pct * pct) as hhi
    from geoseg
    group by gvkey, datadate;

    create table swdata as select a.*, b.n as n_geo, -1 * b.hhi as geodisp
    from swdata a left join geoseg b
    on (a.gvkey = b.gvkey) and (a.datadate = b.datadate);
quit;
```

```
*** CRSP data;

proc sql undo_policy=none;
    * Add permno;
    create table swdata as select a.*, b.lpermno as permno, b.linkdt as crsp_firstdt
    from swdata a left join crsp.ccmxpf_linktable(
     where=(linktype in ("LC", "LU", "LX", "LD", "LN", "LS") and linkprim in ("P" "C"))) b
    on (a.gvkey = b.gvkey) and
       (b.linkdt <= a.datadate or b.linkdt = .B) and (a.datadate <= b.linkenddt or b.linkenddt =.E)
    order by gvkey, datadate, crsp_firstdt;
quit;

* Keep only one permno per firm (check to be safe);
proc sort data=swdata nodupkey;
    by gvkey datadate;
run;

proc sql undo_policy=none;
     * Add stock returns and trading volume;
    create table temp as select a.gvkey, a.datadate, b.date, b.ret, b.vol * (b.vol >= 0) as vol
    from swdata a left join crsp.msf b
    on (a.permno = b.permno) and (intck('MONTH', a.datadate, b.date) between -11 and 0)
    and (a.crsp_firstdt <= b.date)
    where (ret >= -1);

    * Add market returns;
    create table temp as select a.*, b.vwretd as mret
    from temp a left join crsp.msi b
    on (a.date = b.date);
quit;

proc fedsql;
    * Calculate returns, volatility, skewness, etc.;
    create table temp2 as select distinct gvkey, datadate, exp(sum(log(1+ret)))-exp(sum(log(1+mret))) as ks_ret,
                 std(log(1+ret)) as vole, std(ret) as ks_std, sum(vol) as ks_vol, skewness(ret) as ks_skew
    from temp group by gvkey, datadate
    having n(ret) = 12;
quit;

proc sql undo_policy=none;
    * Merge to original data;
    create table swdata as select a.*, b.vole, b.ks_ret, b.ks_std, b.ks_vol, ks_skew
    from swdata a left join temp2 b
    on (a.gvkey = b.gvkey) and (a.datadate = b.datadate);

    * Add firm age;
    create table ages as select distinct gvkey, min(fyear) as firstyr from comp.funda(where=(&testc (at > 0)))
    group by gvkey;

    create table swdata as select a.*, log(1 + a.fyear - b.firstyr) as age
    from swdata a left join ages b
    on (a.gvkey = b.gvkey);

    * Add CRSP industry code;
    create table swdata as select a.*, b.siccd as sic_alt
    from swdata a left join crsp.stocknames b
    on (a.permno = b.permno) and (b.namedt <= a.datadate <= b.nameenddt);
quit;

* Check for duplicate observations to be safe;
proc sort data=swdata nodupkey;
    by gvkey datadate;
run;


* Calculate KS (2012) litigation risk;
data swdata;
    set swdata;
    if missing(sic) then sic = sic_alt; * Replace Compustat SIC with CRSP SIC if missing;

    * Litigious industries (FPS 1994);
    if ((sic >= 2833 and sic <= 2836) or
        (sic >= 8731 and sic <= 8734) or
        (sic >= 3570 and sic <= 3577) or
        (sic >= 7370 and sic <= 7374) or
        (sic >= 3600 and sic <= 3674) or
        (sic >= 5200 and sic <= 5961)) then fps_risk = 1; else fps_risk = 0;
```

```
    if cshom1 > 0 then ks_turn = ks_vol / cshom1 / 10000; else ks_turn = .; * Compustat in $mm, CRSP in $100;
    drop sic sic_alt crsp_firstdt cshom1 ks_vol;
run;

* Winsorize variables;
%winsor(dsetin=swdata, vars=ks_size ks_growth ks_ret ks_skew ks_std ks_turn growth, pctl=2 98);

data swdata;
    set swdata;
    ks_risk = -7.883 + 0.566 * fps_risk + 0.518 * ks_size + 0.982 * ks_growth + 0.379 * ks_ret - 0.108
              * ks_skew + 25.635 * ks_std + 0.00007 * ks_turn;

    drop fps_risk ks_size ks_growth ks_ret ks_skew ks_std ks_turn;
run;


*** Institutional ownership;

proc sql undo_policy=none;
    create table tfn as select cusip, fdate, mgrno, shares, 1000000 * shrout1 as shrout
    from tfn.s34(where=((fdate between '01JAN2002'd and '31DEC2018'd) and (not missing(cusip))));

    create table tfn as select distinct cusip, fdate, sum(shares) / shrout as own
    from tfn group by cusip, fdate;

    create table tfn as select a.*, substr(b.cusip,1,6) as cusip6
    from tfn a left join names b
    on (a.cusip = b.ncusip);
quit;

* Eliminate duplicates;
proc sort data=tfn nodupkey;
    by cusip6 fdate;
run;

proc sql undo_policy=none;
    create table swdata as select a.*, b.own
    from swdata a left join tfn b
    on (a.cusip6 = b.cusip6) and intck('MONTH', b.fdate, a.datadate) between 0 and 2;
quit;


*** Audit Analytics legal data;

data aa;
    set aa.auditlegal(keep=company_fkey case_start_date_s settlement_dollars defendant);
    where not missing(company_fkey) and (case_start_date_s between '01JAN2002'd and '31DEC2018'd);
    year = year(case_start_date_s);
    if defendant ne 1 then delete;
    rename
      company_fkey          = cik
      case_start_date_s     = case_start
      settlement_dollars    = settlement;
run;

proc sort data=aa nodupkey;
    by cik case_start dock_num;
run;

proc sql undo_policy=none;
    create table aa_annual as select distinct cik, year, n(case_start) as nlegal, sum(settlement) as dlegal
    from aa group by cik, year;

    * Merge AA to WRDS data (nlegal = Number of cases, dlegal = Dollars in settlements);
    create table swdata as select a.*, b.nlegal, b.dlegal
    from swdata a left join aa_annual b
    on (a.cik = b.cik) and (a.fyear = (b.year-1));
quit;


*** Audit Analytics 404 internal control data;

proc sort data=aa.auditsox404(keep=company_fkey fye_ic_op ic_is_effective where=(fye_ic_op ge '01JAN2002'd))
out=aa;
    by company_fkey fye_ic_op ic_is_effective;
run;

* Keep one observation per company/year, after previously sorting by ineffective;
```

```
proc sort data=aa nodupkey;
    by company_fkey fye_ic_op;
run;

* Define material weakness indicator;
data aa;
    set aa;
    mwic = 0;
    if ic_is_effective = 'N' then mwic = 1;
run;

* Merge to primary dataset;
proc sql undo_policy=none;
    create table swdata as select a.*, b.mwic as icweak
    from swdata a left join aa b
    on (a.cik = b.company_fkey) and intck('MONTH', a.datadate, b.fye_ic_op) = 0;

    create table swdata as select a.*, b.mwic as icfocus
    from swdata a left join aa b
    on (a.cik = b.company_fkey) and intck('MONTH', a.datadate, b.fye_ic_op) = -12;
quit;

* Check for duplicate observations to be safe;
proc sort nodupkey data=swdata;
    by gvkey fyear;
run;


*** Boardex data;

* Identify boards with director in compliance role;
data bx_roles;
    set boardex.na_wrds_org_composition;
    where not missing(rolename);
    if missing(dateendrole) then dateendrole = '31DEC2019'd;

    brd_compl = 0;
    if find(upcase(rolename), 'COMPLIANCE') > 0 then brd_compl = 1;
    if find(upcase(rolename), 'ETHICS')     > 0 then brd_compl = 1;
    if find(upcase(rolename), 'REGULAT')    > 0 then brd_compl = 1;
    drop directorname companyname;
run;

* Delete duplicates;
proc sort data=bx_roles;
    by companyid directorid datestartrole dateendrole descending brd_compl;
run;

proc sort data=bx_roles nodupkey;
    by companyid directorid datestartrole dateendrole;
run;

* Prepare CIK code file;
data bx_names;
    set boardex.na_wrds_company_names;
    cik = 1 * cikcode;
    keep companyid boardid cik;
run;

* Delete duplicates;
proc sort data=bx_names nodupkey;
    by companyid cik;
run;

* Add CIK codes to dataset;
proc sql undo_policy = none;
    create table bx_roles as select a.*, b.cik
    from bx_roles a left join bx_names b
    on (a.companyid = b.companyid)
    order by companyid, directorid;
quit;

* Merge back to primary dataset;
proc sql undo_policy = none;
    * Merge directors by fiscal year end;
    create table temp as select a.gvkey, a.datadate, b.brd_compl
    from swdata a left join bx_roles b
    on (a.ncik = b.cik) and
       (not missing(a.ncik)) and
```

```
       (b.datestartrole <= a.datadate <= b.dateendrole);

    * Check for at least one director in compliance role;
    create table temp as select distinct gvkey, datadate, max(brd_compl) as brd_compl
    from temp group by gvkey, datadate;

    create table swdata as select a.*, b.brd_compl
    from swdata a left join temp b
    on (a.gvkey = b.gvkey) and
       (a.datadate = b.datadate);
quit;

* Check for duplicate observations to be safe;
proc sort data=swdata nodupkey;
    by gvkey datadate;
run;


*** Create variables for analyses;

data swdata;
    set swdata;

    logemp = log(1+emp);
    if missing(own) then own = 0;  * Set missing institutional ownership to zero;
    size = log(at)

    * Reassign changed GIC codes;
    if ind = '4040' then ind = '6010';
    if ind = '2540' then ind = '5020';

    * Clean up legal data, set to missing if future year not yet available;
    if missing(nlegal) then nlegal = 0;
    if fyear >= 2017   then nlegal = .;
    if missing(dlegal) then dlegal = 0;
    if fyear >= 2017   then dlegal = .;

    lognlegal = log(1+nlegal);
    logdlegal = log(1+dlegal);

    if missing(cik) then do;
       lognlegal = .;
       logdlegal = .;
       icweak    = .;
       icfocus   = .;
    end;
run;

* Save output data for Stata;
proc export data=swdata outfile='sw_wrds.dta' replace;
run;
```

```
*****************************************************************
***** prepare.do
*****
***** Start with original csv files from NAVEX, clean up data
*****************************************************************


* Import and prepare file #1

import delimited using "~/PATH/FILE1NAME.csv", clear

* Clean variables
replace client   = strupper(stritrim(strtrim(client)))
replace issue    = strupper(stritrim(strtrim(issue)))
replace outcome  = strupper(stritrim(strtrim(outcome)))
replace source   = strupper(stritrim(strtrim(source)))
replace length   = strupper(stritrim(strtrim(length)))
replace reporter = strupper(stritrim(strtrim(reporter)))
replace intakemethod = strupper(stritrim(strtrim(intakemethod)))

destring clientid, replace force

* Drop missing data
drop if missing(clientid)
drop if missing(client)
drop if missing(reportdate)
drop if missing(issue)

* Re-code dates
gen rdate = date(substr(reportdate,1,10),"YMD")
gen cdate = date(substr(dateclosed,1,10),"YMD")
format rdate %td
format cdate %td
drop reportdate dateclosed
gen year = year(rdate)

* Merge categories for issue type, outcome, and length using NAVEX mapping files
merge m:m issue using issuetypes, force keepusing(issuecat)
drop if _merge==2
drop _merge

merge m:m outcome using outcomes, force keepusing(outcomecat)
drop if _merge==2
drop _merge

merge m:m length using lengths, force keepusing(lengthcat)
drop if _merge==2
drop _merge

merge m:m source using sources, force keepusing(sourcecat)
drop if _merge==2
drop _merge

merge m:m reporter using reporters, force keepusing(reportercat)
drop if _merge==2
drop _merge

merge m:m intakemethod using intakes, force keepusing(intakecat)
drop if _merge==2
drop _merge

* Save dataset
keep clientid client rdate cdate year issue issuecat outcome outcomecat length lengthcat reporter reportercat
source sourcecat anonymous mgtaware mgtinvolved intakemethod intakecat accesscount
save navdata_temp, replace


* Import and prepare file #2

import delimited using "~/PATH/FILE2NAME.csv", clear

* Clean variables
replace client   = strupper(stritrim(strtrim(client)))
replace issue    = strupper(stritrim(strtrim(issue)))
replace source   = strupper(stritrim(strtrim(source)))
replace length   = strupper(stritrim(strtrim(length)))
replace reporter = strupper(stritrim(strtrim(reporter)))
replace intakemethod = strupper(stritrim(strtrim(intakemethod)))
```

```
destring clientid, replace force

* Drop missing data
drop if missing(clientid)
drop if missing(client)
drop if missing(reportdate)
drop if missing(issue)

* Re-code dates
gen rdate = date(substr(reportdate,1,10),"YMD")
gen cdate = date(substr(dateclosed,1,10),"YMD")
format rdate %td
format cdate %td
drop reportdate dateclosed
gen year = year(rdate)

* Merge categories for issue type, outcome, and length using NAVEX mapping files
merge m:m issue using issuetypes, force keepusing(issuecat)
drop if _merge==2
drop _merge

merge m:m resolution using resolutions, force keepusing(outcomecat)
drop if _merge==2
drop _merge

merge m:m length using lengths, force keepusing(lengthcat)
drop if _merge==2
drop _merge

merge m:m source using sources, force keepusing(sourcecat)
drop if _merge==2
drop _merge

merge m:m reporter using reporters, force keepusing(reportercat)
drop if _merge==2
drop _merge

merge m:m intakemethod using intakes, force keepusing(intakecat)
drop if _merge==2
drop _merge


* Combine data
keep clientid client rdate cdate year issue issuecat outcome outcomecat length lengthcat reporter reportercat
source sourcecat anonymous mgtaware mgtinvolved intakemethod intakecat accesscount
append using navdata_temp


* Clean up multiple clientid-client combinations manually, repeated 22 times
replace clientid = XXXX1  if clientid==XXXX2 & client=="CLIENT NAME"
[21 additional lines suppressed]

* Replace issue categories with more user-friendly captions
replace issuecat="AC" if issuecat=="ACCOUNTING, AUDITING AND FINANCE"
replace issuecat="BI" if issuecat=="BUSINESS INTEGRITY"
replace issuecat="HR" if issuecat=="HR, DIVERSITY AND WORKPLACE RESPECT"
replace issuecat="MU" if issuecat=="MISUSE MISAPPROPRIATION OF CORPORATE ASSETS"
replace issuecat="SF" if issuecat=="ENVIRONMENT, HEALTH AND SAFETY"
replace issuecat="UN" if issuecat==""

* Create indicators for issue types
gen byte iss_ac = issuecat=="AC"
gen byte iss_bi = issuecat=="BI"
gen byte iss_hr = issuecat=="HR"
gen byte iss_mu = issuecat=="MU"
gen byte iss_sf = issuecat=="SF"
gen byte iss_un = issuecat=="UN"

* Create indicators for outcomes
gen byte out_sb = outcomecat=="SUBSTANTIATED"
gen byte out_un = outcomecat=="UNSUBSTANTIATED"
gen byte out_na = outcomecat=="NOT APPLICABLE/NEITHER"
gen byte out_ms = outcomecat==""

* Create indicators for reporter
gen byte rpt_em = reportercat=="EMPLOYEE-CURRENT"
replace  rpt_em = 1 if reportercat=="EMPLOYEE-FORMER"
gen byte rpt_bp = reportercat=="BUS-PARTNER"
gen byte rpt_cu = reportercat=="CUSTOMER"
```

```
gen byte rpt_ot = reportercat=="OTHER"
replace  rpt_ot = 1 if reportercat=="FRIEND/RELATIVE"
gen byte rpt_ms = reportercat=="MISSING"
replace  rpt_ms = 1 if missing(reportercat)

* Create indicators for source
gen byte src_fh = sourcecat=="FIRSTHAND"
gen byte src_sh = sourcecat=="SECONDHAND"
gen byte src_ms = sourcecat=="MISSING"
replace  src_ms = 1 if missing(sourcecat)

* Create indicators for length
gen byte lng_00_01 = lengthcat=="LESS THAN 1 MONTH"
replace  lng_00_01 = 1 if lengthcat=="ONCE"
gen byte lng_01_03 = lengthcat=="LESS THAN 3 MONTHS"
gen byte lng_03_12 = lengthcat=="LESS THAN 12 MONTHS"
gen byte lng_12_99 = lengthcat=="MORE THAN 12 MONTHS"
gen byte lng_ms = lengthcat=="MISSING"
replace  lng_ms = 1 if missing(lengthcat)

* Create indicators for management aware
gen byte maw_yes = upper(substr(mgtaware,1,3))=="YES"
gen byte maw_no  = upper(substr(mgtaware,1,2))=="NO"
replace  maw_no  = 0 if upper(substr(mgtaware,1,3))=="NOT"
gen byte maw_ms  = 1 - maw_yes - maw_no
replace  maw_ms  = 1 if missing(mgtaware)

* Create indicators for management involved
gen byte min_yes = upper(substr(mgtinvolved,1,3))=="YES"
gen byte min_no  = upper(substr(mgtinvolved,1,2))=="NO"
replace  min_no  = 0 if upper(substr(mgtinvolved,1,3))=="NOT"
gen byte min_ms  = 1 - min_yes - min_no
replace  min_ms  = 1 if missing(mgtinvolved)

* Replace T/F variables as 0/1
gen byte anon = upper(anonymous)=="TRUE"
replace  anon = . if missing(anonymous)
drop anonymous

* Calculate time to close
gen byte cdate_ms = 0
replace  cdate_ms = 1 if missing(cdate)
gen time = cdate - rdate
replace time = . if missing(cdate)
replace time = . if (cdate < rdate)
replace time = . if time > 1000

* Create indicator for issues involving retaliation
gen retaliat = strpos(upper(issue), "RETALIAT") > 0

* Create helpline/hotline indicator
gen helpline = strpos(upper(intakemethod), "HELP") > 0

* Create intake method indicators
gen byte int_dir = intakecat=="PERSON"
replace  int_dir = 1 if intakecat=="PHONE"

* Delete test cases
drop if upper(issue)=="HOTLINE TEST"
drop if upper(issue)=="PENETRATION TEST"
drop if upper(issue)=="OTHER - TEST"
drop if upper(issue)=="TEST CASE"
drop if upper(issue)=="TEST"
drop if upper(issue)=="TELECOM TESTING"
drop if upper(issue)=="REPORT FORM TEST"
drop if upper(issue)=="TC TEST"
drop if strpos(upper(issue), "TEST CALL") > 0
drop if strpos(upper(issue), "TEST ISSUE") > 0
drop if upper(intakemethod)=="TEST"


* Merge identifiers
merge m:m clientid using navlink, force keepusing(ngvkey)
keep if _merge==3
drop _merge
order clientid year ngvkey, first
rename year fyear

drop clientid client issue outcome length reporter source intakemethod
```

```
save navdata_reports, replace


* Collapse to firm-year dataset
collapse (count) ncases=iss_bi (sum) iss_* (mean) out_* lng_* maw_* min_* src_* rpt_* int_dir anon accesscount
time helpline retaliat, by(ngvkey fyear)
save navdata_temp, replace

* Merge NAVEX data and WRDS data
use "~/PATH/sw_wrds.dta", clear
merge m:m ngvkey fyear using navdata_temp, force
keep if _merge == 3
drop _merge

* Add penalty data
destring cik, replace
merge m:m cik fyear using penalties, force keepusing(lognpen logdpen)
drop if _merge == 2
drop _merge

* Set to missing if future years not yet available
foreach v of varlist lognpen logdpen {
    replace `v' = 0 if missing(`v')
        replace `v' = . if fyear > 2015
}

* Add KLD data
merge m:m cusip6 fyear using kldscores, force keepusing(env_* com_* hum_* emp_* div_* pro_* cgov_*)
drop if _merge == 2
drop _merge

* Delete all identifying information
gen masked_id = [calculation suppressed]
drop ngvkey gvkey cik ncik cusip6 permno
save navdata_firmyear, replace

use navdata_reports, clear
gen masked_id = [calculation suppressed]
drop ngvkey
save navdata_reports, replace
```

```
*****************************************************************
***** tables12.do
*****
***** Performs report-level analyses in Tables 1 and 2
*****************************************************************


global reqvars "size logemp age growth roa VOLE ks_risk own"

* Require nonmissing key variables - to match sample with firm-level analysis
use navdata_firmyear, clear
keep masked_id fyear emp $reqvars
keep if 2004 <= fyear & fyear <= 2017
keep if emp > 0
foreach v of varlist $reqvars {
    drop if missing(`v')
}

* Merge NAVEX data and WRDS data
merge m:m masked_id fyear using navdata_reports, force
keep if _merge == 3
drop _merge

gen logaccess = log(1+accesscount)
gen logtime =   log(1+time)
gen nonmiss  = 1 - (maw_ms + min_ms + lng_ms + src_ms + rpt_ms) / 5

drop if missing(accesscount)
drop if missing(anon)

winsor2 accesscount logaccess time logtime, replace cuts(0 98)

keep issuecat intakecat logaccess logtime nonmiss accesscount time iss_* out_* lng_* min_* maw_* rpt_* src_*
int_* anon retaliat $reqvars masked_id fyear ind
save navdata_temp, replace


*** Table 1: Descriptive Statistics;

* Panel A: Totals
collapse (sum) iss_ac iss_bi iss_hr iss_mu iss_sf iss_un
egen ncases = rowtotal(iss_ac iss_bi iss_hr iss_mu iss_sf iss_un)
foreach v of varlist iss_ac iss_bi iss_hr iss_mu iss_sf iss_un {
    replace `v' = `v' / ncases
}
format iss_* %5.3f
list, clean

* Panel A: Report type distribution by year
use navdata_temp, clear
collapse (sum) iss_ac iss_bi iss_hr iss_mu iss_sf iss_un, by(fyear)
egen ncases = rowtotal(iss_ac iss_bi iss_hr iss_mu iss_sf iss_un)
foreach v of varlist iss_ac iss_bi iss_hr iss_mu iss_sf iss_un {
    replace `v' = `v' / ncases
}
list, clean

* Panels B and C: Totals
use navdata_temp, clear
collapse (count) n=iss_ac (mean) nonmiss accesscount time anon retaliat int_dir out_sb rpt_* src_* maw_* min_*
lng_*
list, clean

* Panel B: Means by year (group pre 2010 into one category)
use navdata_temp, clear
gen dispyear = fyear
replace dispyear = 2009 if fyear < 2010

collapse (count) n=iss_ac (mean) nonmiss accesscount time anon retaliat int_dir out_sb rpt_* src_* maw_* min_*
lng_*, by(dispyear)
list, clean

* Panel C: Means by report category
use navdata_temp, clear
collapse (count) n=iss_ac (mean) nonmiss accesscount time anon retaliat int_dir out_sb rpt_* src_* maw_* min_*
lng_*, by(issuecat)
list, clean
```

```
*** Table 2: Regressions

use navdata_temp, clear
global types "iss_ac iss_bi iss_hr iss_mu iss_sf"
global intks "int_dir"
global rpts "rpt_em rpt_bp rpt_cu rpt_ot"
global srcs "src_fh src_sh"
global maws "maw_yes maw_no"
global mins "min_yes min_no"
global lngs "lng_00_01 lng_01_03 lng_03_12 lng_12_99"

* Panel A: Missing data fields
reghdfe nonmiss $types $intks anon retaliat, absorb(masked_id fyear) vce(cluster masked_id)

foreach v of varlist logaccess logtime out_sb {
        reghdfe `v' $types $intks anon retaliat rpt_ms src_ms maw_ms min_ms lng_ms, absorb(masked_id fyear)
vce(cluster masked_id)
}

* Panel B: Data field values
foreach v of varlist logaccess logtime out_sb {
        reghdfe `v' $types $intks anon retaliat $rpts $srcs $maws $mins $lngs, absorb(masked_id fyear)
vce(cluster masked_id)
}
```

```
*****************************************************************
***** tables345.do
*****
***** Performs firm-year-level analyses in Tables 3 to 5
*****************************************************************


global reqvars "hashelp size logemp age growth roa VOLE ks_risk own"
global firmvars "hashelp size logemp growth roa VOLE own"
global legvars "size age growth roa VOLE ks_risk own"


* Set sample and create variables
use navdata_firmyear, clear

keep if 2004 <= fyear & fyear <= 2017
keep if emp > 0

gen casemp = ncases / emp
gen byte hashelp = helpline > 0

gen logcasemp = log(1+casemp)
gen logaccess = log(1+accessco)
gen logtime = log(1+time)

gen nonmiss = 1 - (maw_ms + min_ms + lng_ms + src_ms + rpt_ms) / 5

foreach c in ac bi hr mu sf un {
        gen log`c'    = log(1 + iss_`c' / emp)
        gen lognot`c' = log(1 + casemp - iss_`c' / emp)
}

gen kld = cgov_str + com_str + div_str + emp_str + env_str + hum_str + pro_str - cgov_con - com_con - div_con -
emp_con - env_con - hum_con - pro_con

* Require nonmissing key variables
keep if (ncases > 0)
foreach v of varlist logcasemp $reqvars {
    drop if missing(`v')
}


* Winsorize data
winsor2 casemp accessco time logcasemp logaccess logtime lognlegal logdlegal lognpen logdpen logac lognotac
logbi lognotbi loghr lognothr logmu lognotmu logsf lognotsf logun lognotun, replace cuts(0 98)
winsor2 $reqvars geodisp, replace cuts(2 98)


*** Table 3: Descriptive statistics

* Panel A: Summary statistics
tabstat logcasemp nonmiss logaccess lognpen logdpen lognlegal logdlegal $reqvars brd_compl icweak icfocus kld
geodisp, stat(n mean sd p25 p50 p75) columns(statistics) format(%5.3f) save

* Panel B: By industry
save navdata_temp, replace
collapse (count) n=casemp (mean) casemp nonmiss accessco time anon retaliat int_dir out_sb
format casemp nonmiss accessco time anon retaliat int_dir out_sb %8.3f
format n %5.0f
list, clean

use navdata_temp, clear
collapse (count) n=casemp (mean) casemp nonmiss accessco time anon retaliat int_dir out_sb, by(ind)
format casemp nonmiss accessco time anon retaliat int_dir out_sb %8.3f
format n %5.0f
list, clean


*** Figure 1: Firm characteristics by quintile
use navdata_temp, clear
foreach v of varlist hashelp brd_compl icweak icfocus {
        tabstat casemp, by(`v') stat(mean) format(%5.3f) save
        matrix output = r(Stat1)\r(Stat2)
}

foreach v of varlist size logemp growth roa VOLE own kld geodisp {
        sort `v'
        gen `v'_5 = group(5)
        tabstat casemp, by(`v'_5) stat(mean) format(%5.3f) save
```

```
        matrix output = r(Stat1)\r(Stat2)\r(Stat3)\r(Stat4)\r(Stat5)
        drop `v'_5
}

* Figure 2: Outcomes by quintile
gen npen   = exp(lognpen) - 1
gen dpen   = exp(logdpen) - 1
gen nlegal = exp(lognlegal) - 1
gen dlegal = exp(logdlegal) - 1

sort logcasemp
gen casemp5 = group(5)
tabstat npen dpen nlegal dlegal, by(casemp5) stat(mean) format(%5.3f) save
matrix output = r(Stat1)\r(Stat2)\r(Stat3)\r(Stat4)\r(Stat5)
drop casemp5 npen dpen nlegal dlegal


*** Table 4: Firm characteristics regressions

* Panel A: Base models
reghdfe logcasemp $firmvars, absorb(ind fyear) vce(cluster masked_id)
reghdfe nonmiss   $firmvars, absorb(ind fyear) vce(cluster masked_id)
reghdfe logaccess $firmvars, absorb(ind fyear) vce(cluster masked_id)

* Panel B: Additional variables
reghdfe logcasemp $firmvars brd_compl,      absorb(ind fyear) vce(cluster masked_id)
reghdfe logcasemp $firmvars icfocus icweak, absorb(ind fyear) vce(cluster masked_id)
reghdfe logcasemp $firmvars kld,            absorb(ind fyear) vce(cluster masked_id)
reghdfe logcasemp $firmvars geodisp,        absorb(ind fyear) vce(cluster masked_id)

* Panel C: By report category
gen byte post2010 = fyear >= 2010
gen byte post2017 = fyear >= 2017

foreach c in ac bi hr mu sf un {
        reghdfe log`c' lognot`c' post2010 post2017 $firmvars, absorb(ind) vce(cluster masked_id)
}


** Table 5: Subsequent outcomes

* Panel A: Base models
foreach v of varlist lognpen logdpen lognlegal logdlegal {
        reghdfe `v' logcasemp $legvars, absorb(masked_id fyear) vce(cluster masked_id)
}

* Panel B: By category
foreach v of varlist lognpen logdpen lognlegal logdlegal {
        reghdfe `v' logac logbi loghr logmu logsf logun $legvars, absorb(masked_id fyear) vce(cluster masked_id)
}
```