```r
#-----------------------------------------------------------------------
#Title: Analyst Coverage Overlaps and Interfirm Information Spillovers
#Authors: Tim Martens and Christoph J. Sextroh
#Responsible for code: Tim Martens
#Date: 15.04.2021
#-----------------------------------------------------------------------

#introduction-------------------------------------------------------------------------
#this file sets the parameters for all following r files
#directories are masked/deleted

#working directory-------------------------------------------------------------------------
base::setwd()

#parameters-------------------------------------------------------------------------
base::set.seed(42) ##set seed

getwd()#packages-------------------------------------------------------------------------
usepackage_function<-function(p){
  if(!base::is.element(p,installed.packages()[,1]))
    utils::install.packages(p,dep=TRUE,repos="http://cran.r-project.org")
  base::require(p,character.only=TRUE)
}

packages<-c("tidyr","dplyr","haven","readxl","lfe","multiwayvcov","lmtest","zoo","dummies","reshape","stargazer","openxlsx","pscl","ggplot2","readr","fixest","DescTools")
for(i in packages){
  usepackage_function(i) ##load package
}
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares crsp header data for further processing

#prepare data----------------------------------------------------------------------
crsp_header<-haven::read_dta("crsp_stock_header_information_20180307.dta") ##import data
crsp_header<-base::as.data.frame(crsp_header) ##change to data frame

crsp_header[,"permno"]<-base::as.character(crsp_header[,"permno"]) ##transform variable to character

crsp_header<-crsp_header[,c("permno","hsiccd")] ##delete unnecessary variables

consistencycheck<-base::nrow(crsp_header) ##security break
crsp_header<-crsp_header[!base::duplicated(base::paste(crsp_header[,"permno"],sep=""))
,] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(crsp_header)) ##security break

base::save(crsp_header,file="crsp_header_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares patents data for further processing

#prepare data----------------------------------------------------------------------
patents<-utils::read.csv(file="patents_20180112.csv",stringsAsFactors=FALSE) ##import data

for(i in c("fdate","idate")){
  patents[,i]<-base::as.Date(patents[,i],"%m/%d/%Y") ##transform variable to date
  patents<-patents[!base::is.na(patents[,i]),] ##delete observations with missing variables
}

for(i in c("permno","patnum")){
  patents[,i]<-base::as.character(patents[,i]) ##transform variable to character
}

for(i in c("permno","patnum")){
  patents<-patents[!base::is.na(patents[,i]),] ##delete observations with missing variables
  patents<-patents[!patents[,i]==""," ] ##delete observations with missing variables
}

patents[,"fyear"]<-base::gsub(".{6}$","",patents[,"fdate"]) ##generate year variable
patents[,"iyear"]<-base::gsub(".{6}$","",patents[,"idate"]) ##generate year variable

patents<-patents[patents[,"fdate"]>="1920-01-01",] ##reduce data
patents<-patents[patents[,"fdate"]<="2010-12-31",] ##reduce data

patents[,"numpatents"]<-1 ##generate variable

consistencycheck<-base::nrow(patents) ##security break
patents<-patents[!base::duplicated(base::paste(patents[,"permno"],"_",patents[,"patnum"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(patents)) ##security break

base::save(patents,file="patents_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares cites data for further processing

#prepare data----------------------------------------------------------------------
cites<-utils::read.csv(file="cites_20180112.csv",stringsAsFactors=FALSE) ##import data

for(i in c("citing","cited")){
  cites[,i]<-base::as.character(cites[,i]) ##transform variable to character
}

for(i in c("citing","cited")){
  cites<-cites[!base::is.na(cites[,i]),] ##delete observations with missing variables
  cites<-cites[!cites[,i]==""] ##delete observations with missing variables
}

cites[,"numcit"]<-1 ##generate variable

consistencycheck<-base::nrow(cites) ##security break
cites<-cites[!base::duplicated(base::paste(cites[,"citing"],"_",cites[,"cited"],sep=""
)),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(cites)) ##security break

base::save(cites,file="cites_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares total patents data for further processing

#prepare data-----------------------------------------------------------------------
base::load(file="patents_processed.Rdata") ##import data

total_patents<-stats::aggregate(patents[c("numpatents")],patents[c("permno","iyear")],base::sum,na.rm=TRUE) ##aggregate data

permno<-base::data.frame("permno"=base::unique(total_patents[,"permno"]),"merge"=1) ##create temporary data frame
year<-base::data.frame("iyear"=c(base::min(patents[,"fyear"],na.rm=TRUE):base::max(patents[,"fyear"],na.rm=TRUE)),"merge"=1) ##create temporary data frame
permno_year<-base::merge(permno,year) ##merge

total_patents<-base::merge(total_patents,permno_year,all.x=TRUE,all.y=TRUE) ##merge
total_patents[,"cited_numpatents"]<-base::ifelse(base::is.na(total_patents[,"numpatents"]),0,total_patents[,"numpatents"]) ##fill nas with zero
total_patents[,"citing_fyear"]<-total_patents[,"iyear"] ##generate variable
total_patents[,"cited_permno"]<-total_patents[,"permno"] ##generate variable
total_patents[,"numpatents"]<-NULL ##delete variable
total_patents[,"merge"]<-NULL ##delete variable
total_patents[,"iyear"]<-NULL ##delete variable
total_patents[,"permno"]<-NULL ##delete variable

total_patents<-total_patents[base::order(total_patents[,"cited_permno"],total_patents[,"citing_fyear"]),] ##order data

total_patents<-total_patents%>%dplyr::group_by(cited_permno) ##group data
total_patents<-total_patents%>%dplyr::mutate(lag3_cited_numpatents=dplyr::lag(cited_numpatents,n=3,default=0)) ##calculate lag
total_patents<-total_patents%>%dplyr::mutate(cited_patentstocktotal=base::cumsum(cited_numpatents)) ##calculate sum
total_patents<-total_patents%>%dplyr::mutate(cited_patentstockunder3y=zoo::rollapplyr(cited_numpatents,3,sum,partial=TRUE)) ##calculate sum
total_patents<-total_patents%>%dplyr::mutate(cited_patentstockbetween3and7y=zoo::rollapplyr(lag3_cited_numpatents,5,sum,partial=TRUE)) ##calculate sum
total_patents<-base::as.data.frame(total_patents) ##ungroup data

total_patents[,"cited_patentstockunder3yperc"]<-total_patents[,"cited_patentstockunder3y"]/total_patents[,"cited_patentstocktotal"] ##generate variable
total_patents[,"cited_patentstockbetween3and7yperc"]<-total_patents[,"cited_patentstockbetween3and7y"]/total_patents[,"cited_patentstocktotal"] ##generate variable

total_patents<-total_patents[,c("cited_permno","citing_fyear","cited_patentstocktotal","cited_patentstockunder3yperc","cited_patentstockbetween3and7yperc")] ##reduce data

consistencycheck<-base::nrow(total_patents) ##security break
total_patents<-total_patents[!base::duplicated(base::paste(total_patents[,"cited_permno"],"_",total_patents[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(total_patents)) ##security break

base::save(total_patents,file="total_patents_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end--------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares total citations data for further processing

#prepare data-----------------------------------------------------------------------
base::load(file="patents_processed.Rdata") ##import data

patents<-patents[,c("permno","patnum","fyear","iyear")] ##reduce data
base::colnames(patents)<-base::paste("citing_",base::colnames(patents),sep="") ##add prefix

base::load(file="cites_processed.Rdata") ##import data

patents_cites<-base::merge(patents,cites,by.x=c("citing_patnum"),by.y=c("citing")) ##merge

total_citations<-stats::aggregate(patents_cites[c("numcit")],patents_cites[c("citing_permno","citing_fyear")],base::sum,na.rm=TRUE) ##aggregate data
total_citations[,"citing_totalcites"]<-total_citations[,"numcit"] ##generate variable
total_citations[,"numcit"]<-NULL ##delete variable

consistencycheck<-base::nrow(total_citations) ##security break
total_citations<-total_citations[!base::duplicated(base::paste(total_citations[,"citing_permno"],"_",total_citations[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(total_citations)) ##security break

base::save(total_citations,file="total_citations_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares the uspatentcitation data for further processing

#start-----------------------------------------------------------------------
uspatentcitation<-readr::read_delim("uspatentcitation_20190306.tsv","\t",escape_double=FALSE,trim_w=TRUE,col_types="cccccccd",progress=FALSE) ##import data
uspatentcitation<-base::as.data.frame(uspatentcitation) ##change to data frame

uspatentcitation[,"citation_id"]<-base::as.character(uspatentcitation[,"citation_id"]) ##transform variable to character
uspatentcitation[,"patent_id"]<-base::as.character(uspatentcitation[,"patent_id"]) ##transform variable to character

uspatentcitation<-uspatentcitation[!base::duplicated(base::paste(uspatentcitation[,"patent_id"],"_",uspatentcitation[,"citation_id"],sep="")),] ##delete duplicates

base::colnames(uspatentcitation)<-base::paste("suspatentcitation_",base::colnames(uspatentcitation),sep="") ##add prefix

consistencycheck<-base::nrow(uspatentcitation) ##security break
uspatentcitation<-uspatentcitation[!base::duplicated(base::paste(uspatentcitation[,"suspatentcitation_patent_id"],"_",uspatentcitation[,"suspatentcitation_citation_id"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(uspatentcitation)) ##security break

base::save(uspatentcitation,file="uspatentcitation_processed.Rdata") ##export data

base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares citation flows data for further processing

#prepare data-----------------------------------------------------------------------
base::load(file="patents_processed.Rdata") ##import data

tempa<-patents[,c("permno","patnum","class","subclass","xi","ncites","fyear","fdate","iyear","idate")] ##reduce data
tempb<-patents[,c("permno","patnum","class","subclass","xi","ncites","fyear","fdate","iyear","idate")] ##reduce data
base::colnames(tempa)<-base::paste("citing_",base::colnames(tempa),sep="") ##add prefix
base::colnames(tempb)<-base::paste("cited_",base::colnames(tempb),sep="") ##add prefix
base::rm(patents) ##clean
base::gc() ##clean

base::load(file="cites_processed.Rdata") ##import data

cites<-base::merge(cites,tempa,by.x=c("citing"),by.y=c("citing_patnum")) ##merge
cites<-base::merge(cites,tempb,by.x=c("cited"),by.y=c("cited_patnum")) ##merge

base::load(file="uspatentcitation_processed.Rdata") ##import data

uspatentcitation<-uspatentcitation[,c("suspatentcitation_patent_id","suspatentcitation_citation_id","suspatentcitation_category")] ##reduce data

consistencycheck<-base::nrow(cites) ##security break
cites<-base::merge(cites,uspatentcitation,by.x=c("citing","cited"),by.y=c("suspatentcitation_patent_id","suspatentcitation_citation_id"),all.x=TRUE,all.y=FALSE) ##merge
base::stopifnot(consistencycheck==base::nrow(cites)) ##security break

cites[,"examinernumcit"]<-base::ifelse(cites[,"suspatentcitation_category"]=="cited by examiner",1,0) ##generate variable
cites[,"examinernumcit"]<-base::ifelse(base::is.na(cites[,"examinernumcit"]),0,cites[,"examinernumcit"]) ##adjust variable
cites[,"othernumcit"]<-base::ifelse(!cites[,"suspatentcitation_category"]=="cited by examiner",1,0) ##generate variable
cites[,"othernumcitadjusted"]<-base::ifelse(base::is.na(cites[,"othernumcit"]),0,cites[,"othernumcit"]) ##adjust variable
cites[,"othernumcit"]<-base::ifelse(base::is.na(cites[,"othernumcit"]),1,cites[,"othernumcit"]) ##adjust variable

cites[,"inclassnumcit"]<-base::ifelse(cites[,"citing_class"]==cites[,"cited_class"],1,0) ##generate variable
cites[,"outclassnumcit"]<-base::ifelse(cites[,"citing_class"]!=cites[,"cited_class"],1,0) ##generate variable
cites[,"insubclassnumcit"]<-base::ifelse(cites[,"citing_subclass"]==cites[,"cited_subclass"]&cites[,"citing_class"]==cites[,"cited_class"],1,0) ##generate variable
cites[,"outsubclassnumcit"]<-base::ifelse(cites[,"citing_subclass"]!=cites[,"cited_subclass"]|cites[,"citing_class"]!=cites[,"cited_class"],1,0) ##generate variable

cites_aggr<-stats::aggregate(cites[c("numcit","examinernumcit","othernumcit","othernumcitadjusted","inclassnumcit","outclassnumcit","insubclassnumcit","outsubclassnumcit")],cites[c("citing_permno","cited_permno","citing_fyear")],base::sum,na.rm=TRUE) ##aggregate data

stopifnot(cites_aggr[,"numcit"]==cites_aggr[,"inclassnumcit"]+cites_aggr[,"outclassnumcit"]) ##security break
stopifnot(cites_aggr[,"numcit"]==cites_aggr[,"insubclassnumcit"]+cites_aggr[,"outsubclassnumcit"]) ##security break
stopifnot(cites_aggr[,"numcit"]==cites_aggr[,"examinernumcit"]+cites_aggr[,"othernumcit"]) ##security break

cites_aggr[,"citing_cited_pair"]<-base::paste(cites_aggr[,"citing_permno"],"_",cites_aggr[,"cited_permno"],sep="") ##generate variable
base::rm(tempa,tempb) ##clean
base::gc() ##clean

citing_cited_pair<-base::data.frame("citing_cited_pair"=base::unique(cites_aggr[,"citing_cited_pair"]),"merge"=1) ##generate data frame
year<-base::data.frame("citing_fyear"=c(base::min(cites_aggr[,"citing_fyear"],na.rm=TRUE):base::max(cites_aggr[,"citing_fyear"],na.rm=TRUE)),"merge"=1) ##generate data frame
```

```r

citing_cited_pair_year<-base::merge(citing_cited_pair,year) ##merge

citing_cited_pair_year[,"citing_permno"]<-base::gsub(".{6}$","",citing_cited_pair_year[,"citing_cited_pair"]) ##reconstruct permno
citing_cited_pair_year[,"cited_permno"]<-base::gsub("^.{6}","",citing_cited_pair_year[,"citing_cited_pair"]) ##reconstruct permno
citing_cited_pair_year[,"merge"]<-NULL ##delete variable

citation_flows<-base::merge(cites_aggr,citing_cited_pair_year,all.x=TRUE,all.y=TRUE) ##merge

citation_flows[,"citing_numcit"]<-base::ifelse(base::is.na(citation_flows[,"numcit"]),0,citation_flows[,"numcit"]) ##fill nas with zero
citation_flows[,"citing_inclassnumcit"]<-base::ifelse(base::is.na(citation_flows[,"inclassnumcit"]),0,citation_flows[,"inclassnumcit"]) ##fill nas with zero
citation_flows[,"citing_outclassnumcit"]<-base::ifelse(base::is.na(citation_flows[,"outclassnumcit"]),0,citation_flows[,"outclassnumcit"]) ##fill nas with zero
citation_flows[,"citing_insubclassnumcit"]<-base::ifelse(base::is.na(citation_flows[,"insubclassnumcit"]),0,citation_flows[,"insubclassnumcit"]) ##fill nas with zero
citation_flows[,"citing_outsubclassnumcit"]<-base::ifelse(base::is.na(citation_flows[,"outsubclassnumcit"]),0,citation_flows[,"outsubclassnumcit"]) ##fill nas with zero
citation_flows[,"citing_examinernumcit"]<-base::ifelse(base::is.na(citation_flows[,"examinernumcit"]),0,citation_flows[,"examinernumcit"]) ##fill nas with zero
citation_flows[,"citing_othernumcit"]<-base::ifelse(base::is.na(citation_flows[,"othernumcit"]),0,citation_flows[,"othernumcit"]) ##fill nas with zero
citation_flows[,"citing_othernumcitadjusted"]<-base::ifelse(base::is.na(citation_flows[,"othernumcitadjusted"]),0,citation_flows[,"othernumcitadjusted"]) ##fill nas with zero

citation_flows[,"citing_self_citation"]<-base::ifelse(citation_flows[,"citing_permno"]==citation_flows[,"cited_permno"],1,0) ##generate variable

citation_flows[,"numcit"]<-NULL ##delete variable
citation_flows[,"inclassnumcit"]<-NULL ##delete variable
citation_flows[,"outclassnumcit"]<-NULL ##delete variable
citation_flows[,"insubclassnumcit"]<-NULL ##delete variable
citation_flows[,"outsubclassnumcit"]<-NULL ##delete variable
citation_flows[,"examinernumcit"]<-NULL ##delete variable
citation_flows[,"othernumcit"]<-NULL ##delete variable
citation_flows[,"othernumcitadjusted"]<-NULL ##delete variable

consistencycheck<-base::nrow(citation_flows) ##security break
citation_flows<-citation_flows[!base::duplicated(base::paste(citation_flows[,"citing_permno"],"_",citation_flows[,"cited_permno"],"_",citation_flows[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows)) ##security break

base::save(citation_flows,file="citation_flows_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end----------------------------------------------------------------------

```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes data for further processing

#prepare data----------------------------------------------------------------------
broker_closure<-readxl::read_excel("broker_closure_20180706cs.xlsx") ##import data
broker_closure<-base::as.data.frame(broker_closure) ##change to data frame

broker_closure[,"datenorm"]<-base::paste(broker_closure[,"datenorm"],"-01",sep="")
##adjust variable

broker_closure[,"closure_year"]<-base::as.numeric(base::gsub(".{6}$","",broker_closure
[,"datenorm"])) ##generate variable

base::save(broker_closure,file="broker_closure_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end--------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes data for further processing

#prepare data----------------------------------------------------------------------
broker_merger<-readxl::read_excel("broker_merger.xlsx") ##import data
broker_merger<-base::as.data.frame(broker_merger) ##change to data frame

broker_merger[,"datenorm"]<-base::paste(broker_merger[,"datenorm"],"-01",sep="")
##adjust variable

broker_merger[,"closure_year"]<-base::as.numeric(base::gsub(".{6}$","",broker_merger[,
"datenorm"])) ##generate variable

base::save(broker_merger,file="broker_merger_processed.Rdata") ##export data

base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end--------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes crsp link data for further processing

#prepare data-----------------------------------------------------------------------
ibes_crsp_link<-haven::read_dta("ibes_crsp_link_20180516.dta") ##import data
ibes_crsp_link<-base::as.data.frame(ibes_crsp_link) ##change to data frame

ibes_crsp_link[,"permno"]<-base::as.character(ibes_crsp_link[,"permno"]) ##transform variable to character

ibes_crsp_link[,"edate"]<-base::as.character(ibes_crsp_link[,"edate"]) ##adjust variable
ibes_crsp_link[,"sdate"]<-base::as.character(ibes_crsp_link[,"sdate"]) ##adjust variable

ibes_crsp_link[,"edate"]<-base::ifelse(base::is.na(ibes_crsp_link[,"edate"]),base::as.character(base::Sys.Date()),ibes_crsp_link[,"edate"]) ##adjust variable

base::save(ibes_crsp_link,file="ibes_crsp_link_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end--------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes data for further processing

#prepare data----------------------------------------------------------------------
ibes<-haven::read_dta("detail_history_detail_file_with_actuals_20180315.dta") ##import data
ibes<-base::as.data.frame(ibes) ##change to data frame

ibes<-ibes[,c("ticker","anndats","analys","estimator")] ##reduce data
base::gc() ##clean

ibes[,"year"]<-base::gsub(".{6}$","",ibes[,"anndats"]) ##generate variable
ibes<-ibes[ibes[,"year"]<=2010,] ##reduce data

ibes<-ibes[!base::is.na(ibes[,"estimator"]),] ##delete observations with missing data
ibes<-ibes[!base::is.na(ibes[,"analys"]),] ##delete observations with missing data
ibes<-ibes[!base::is.na(ibes[,"ticker"]),] ##delete observations with missing data
ibes<-ibes[!base::is.na(ibes[,"year"]),] ##delete observations with missing data
ibes<-ibes[!ibes[,"estimator"]=="",] ##delete observations with missing data
ibes<-ibes[!ibes[,"analys"]=="",] ##delete observations with missing data
ibes<-ibes[!ibes[,"ticker"]=="",] ##delete observations with missing data
ibes<-ibes[!ibes[,"year"]=="",] ##delete observations with missing data

ibes<-ibes[ibes[,"analys"]!=0,] ##reduce data

ibes<-ibes[!base::duplicated(base::paste(ibes[,"ticker"],"_",ibes[,"anndats"],"_",ibes[,"year"],"_",ibes[,"analys"],sep="")),] ##delete duplicates

base::save(ibes,file="ibes_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end--------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares segments data for further processing

#prepare data-----------------------------------------------------------------------
segments<-haven::read_dta("crsp_compustat_merged_database_historical_segments_20180613.dta") ##import data
segments<-base::as.data.frame(segments) ##change to data frame

segments[,"count"]<-1 ###generate variable

segmentsgeoseg<-segments[segments[,"stype"]=="GEOSEG",] ##reduce data
segmentsgeoseg<-stats::aggregate(segmentsgeoseg[c("count")],segmentsgeoseg[c("datadate","gvkey")],base::sum,na.rm=TRUE) ##aggregate
segmentsgeoseg[,"geosegcount"]<-segmentsgeoseg[,"count"] ##generate variable
segmentsgeoseg[,"count"]<-NULL ##delete variable

segmentsbusseg<-segments[segments[,"stype"]=="BUSSEG",] ##reduce data
segmentsbusseg<-stats::aggregate(segmentsbusseg[c("count")],segmentsbusseg[c("datadate","gvkey")],base::sum,na.rm=TRUE) ##aggregate
segmentsbusseg[,"bussegcount"]<-segmentsbusseg[,"count"] ##generate variable
segmentsbusseg[,"count"]<-NULL ##delete variable

segments<-base::merge(segmentsgeoseg,segmentsbusseg,all=TRUE) ##merge

segments[,"year"]<-base::gsub(".{6}$","",segments[,"datadate"]) ##generate variable
segments[,"datadate"]<-NULL ##delete variable

segments<-segments[!base::duplicated(base::paste(segments[,"gvkey"],"_",segments[,"year"],sep="")),] ##delete duplicates

consistencycheck<-base::nrow(segments) ##security break
segments<-segments[!base::duplicated(base::paste(segments[,"gvkey"],"_",segments[,"year"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(segments)) ##security break

base::save(segments,file="segments_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end---------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges ibes data and ibes crsp link data

#merge-----------------------------------------------------------------------
base::load(file="ibes_processed.Rdata") ##import data
base::load(file="ibes_crsp_link_processed.Rdata") ##import data

base::stopifnot(base::inherits(ibes[,"ticker"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(ibes_crsp_link[,"ticker"],"character")==TRUE) ##assure that variables have the same format

ibes_ibes_crsp_link<-base::merge(ibes,ibes_crsp_link,by.x=c("ticker"),by.y=c("ticker"),all.x=FALSE,all.y=FALSE) ##merge

ibes_ibes_crsp_link<-ibes_ibes_crsp_link[ibes_ibes_crsp_link[,"anndats"]>=ibes_ibes_crsp_link[,"sdate"]&ibes_ibes_crsp_link[,"anndats"]<=ibes_ibes_crsp_link[,"edate"],] ##assure clean data

ibes_ibes_crsp_link<-ibes_ibes_crsp_link[ibes_ibes_crsp_link[,"score"] %in% c(1:4),] ##assure clean data

consistencycheck<-base::nrow(ibes_ibes_crsp_link) ##security break
ibes_ibes_crsp_link<-ibes_ibes_crsp_link[!base::duplicated(base::paste(ibes_ibes_crsp_link[,"ticker"],"_",ibes_ibes_crsp_link[,"anndats"],"_",ibes_ibes_crsp_link[,"year"],"_",ibes_ibes_crsp_link[,"analys"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(ibes_ibes_crsp_link)) ##security break

base::save(ibes_ibes_crsp_link,file="ibes_ibes_crsp_link_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes experience data for further processing

#prepare data-----------------------------------------------------------------------
base::load("ibes_ibes_crsp_link_processed.Rdata") ##import data

ibes_ibes_crsp_link<-ibes_ibes_crsp_link[!base::duplicated(base::paste(ibes_ibes_crsp_link[,"permno"],"_",ibes_ibes_crsp_link[,"year"],"_",ibes_ibes_crsp_link[,"analys"],sep="")),] ##delete duplicates

ibes_ibes_crsp_link<-ibes_ibes_crsp_link%>%dplyr::group_by(analys) ##group data
ibes_ibes_crsp_link<-ibes_ibes_crsp_link%>%dplyr::mutate(minyear=base::min(year)) ##calculate min
ibes_ibes_crsp_link<-base::as.data.frame(ibes_ibes_crsp_link) ##ungroup data

ibes_ibes_crsp_link[,"expy"]<-base::as.numeric(ibes_ibes_crsp_link[,"year"])-base::as.numeric(ibes_ibes_crsp_link[,"minyear"]) ##generate variable

ibes_experience<-ibes_ibes_crsp_link[,c("permno","analys","year","expy")] ##reduce data

ibes_experience<-ibes_experience%>%dplyr::group_by(year) ##group data
ibes_experience<-ibes_experience%>%dplyr::mutate(expquant090=stats::quantile(expy,probs=0.90)) ##calculate quantile
ibes_experience<-base::as.data.frame(ibes_experience) ##ungroup data

ibes_experience[,"expyo090"]<-base::ifelse(ibes_experience[,"expy"]>ibes_experience[,"expquant090"],1,0) ##generate variable
ibes_experience[,"expyu090"]<-base::ifelse(ibes_experience[,"expy"]<=ibes_experience[,"expquant090"],1,0) ##generate variable

for(i in c("expy","expquant090")){
  ibes_experience[,i]<-NULL ##reduce data
}

base::save(ibes_experience,file="ibes_experience_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes expertise data for further processing

#prepare data-----------------------------------------------------------------------
base::load(".ibes_ibes_crsp_link_processed.Rdata") ##import data

ibes_ibes_crsp_link<-ibes_ibes_crsp_link[!base::duplicated(base::paste(ibes_ibes_crsp_link[,"permno"],"_",ibes_ibes_crsp_link[,"year"],"_",ibes_ibes_crsp_link[,"analys"],sep="")),] ##delete duplicates
ibes_ibes_crsp_link[,"count"]<-1 ##generate variable

base::load(file="crsp_header_processed.Rdata") ##import

crsp_header[,"sic2"]<-base::gsub(".{2}$","",crsp_header[,"hsiccd"]) ##generate variable

ibes_expertise<-base::merge(ibes_ibes_crsp_link,crsp_header,by=c("permno"),all.x=FALSE,all.y=FALSE) ##merge

tempind<-stats::aggregate(ibes_expertise[c("count")],ibes_expertise[c("analys","year","sic2")],base::sum,na.rm=TRUE) ##aggregate
tempind[,"indcount"]<-tempind[,"count"] ##generate variable
tempind[,"count"]<-NULL ##delete variable

tempanalyst<-stats::aggregate(ibes_expertise[c("count")],ibes_expertise[c("analys","year")],base::sum,na.rm=TRUE) ##aggregate
tempanalyst[,"totalcount"]<-tempanalyst[,"count"] ##generate variable
tempanalyst[,"count"]<-NULL ##delete variable

temp<-base::merge(tempanalyst,tempind,by=c("analys","year"),all.x=FALSE,all.y=FALSE) ##merge

temp[,"hiind"]<-(temp[,"indcount"]/temp[,"totalcount"])^2 ##generate variable

ibes_expertise<-stats::aggregate(temp[c("hiind")],temp[c("analys","year")],base::sum,na.rm=TRUE) ##aggregate

ibes_expertise[,"hiindo90"]<-base::ifelse(ibes_expertise[,"hiind"]>0.9,1,0) ##generate variable
ibes_expertise[,"hiindu90"]<-base::ifelse(ibes_expertise[,"hiind"]<=0.9,1,0) ##generate variable

for(i in c("hiind")){
  ibes_expertise[,i]<-NULL ##reduce data
}

consistencycheck<-base::nrow(ibes_expertise) ##security break
ibes_expertise<-ibes_expertise[!base::duplicated(base::paste(ibes_expertise[,"analys"],"_",ibes_expertise[,"year"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(ibes_expertise)) ##security break

base::save(ibes_expertise,file="ibes_expertise_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes portfolio data for further processing

#prepare data-----------------------------------------------------------------------
base::load("ibes_ibes_crsp_link_processed.Rdata") ##import data

ibes_ibes_crsp_link<-ibes_ibes_crsp_link[!base::duplicated(base::paste(ibes_ibes_crsp_link[,"permno"],"_",ibes_ibes_crsp_link[,"year"],"_",ibes_ibes_crsp_link[,"analys"],sep="")),] ##delete duplicates

ibes_ibes_crsp_link[,"numfirms"]<-1 ##generate variable

ibes_portfolio<-stats::aggregate(ibes_ibes_crsp_link[c("numfirms")],ibes_ibes_crsp_link[c("analys","year")],base::sum,na.rm=TRUE) ##aggregate data

ibes_portfolio<-ibes_portfolio%>%dplyr::group_by(year) ##group data
ibes_portfolio<-ibes_portfolio%>%dplyr::mutate(portquant090=stats::quantile(numfirms,probs=0.90)) ##calculate quantile
ibes_portfolio<-base::as.data.frame(ibes_portfolio) ##ungroup data

ibes_portfolio[,"porto090"]<-base::ifelse(ibes_portfolio[,"numfirms"]>ibes_portfolio[,"portquant090"],1,0) ##generate variable
ibes_portfolio[,"portu090"]<-base::ifelse(ibes_portfolio[,"numfirms"]<=ibes_portfolio[,"portquant090"],1,0) ##generate variable

for(i in c("numfirms","portquant090")){
  ibes_portfolio[,i]<-NULL ##reduce data
}

base::save(ibes_portfolio,file="ibes_portfolio_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes actions data for further processing

#prepare data-----------------------------------------------------------------------
base::load("ibes_ibes_crsp_link_processed.Rdata") ##import data

ibes_ibes_crsp_link[,"analys_actions"]<-1 ##generate variable

ibes_actions<-stats::aggregate(ibes_ibes_crsp_link[c("analys_actions")],ibes_ibes_crsp_link[c("analys","permno","year")],base::sum,na.rm=TRUE) ##aggregate data

ibes_actions<-ibes_actions%>%dplyr::group_by(year) ##group data
ibes_actions<-ibes_actions%>%dplyr::mutate(actionsquant050=stats::quantile(analys_actions,probs=0.50)) ##calculate quantile
ibes_actions<-base::as.data.frame(ibes_actions) ##ungroup data

ibes_actions[,"actionso050"]<-base::ifelse(ibes_actions[,"analys_actions"]>ibes_actions[,"actionsquant050"],1,0) ##generate variable
ibes_actions[,"actionsu050"]<-base::ifelse(ibes_actions[,"analys_actions"]<=ibes_actions[,"actionsquant050"],1,0) ##generate variable

for(i in c("analys_actions","actionsquant050")){
  ibes_actions[,i]<-NULL ##reduce data
}

base::save(ibes_actions,file="ibes_actions_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes common data for further processing

#prepare data-----------------------------------------------------------------------
base::load("ibes_ibes_crsp_link_processed.Rdata") ##import data

ibes_ibes_crsp_link<-ibes_ibes_crsp_link[!base::duplicated(base::paste(ibes_ibes_crsp_link[,"permno"],"_",ibes_ibes_crsp_link[,"year"],"_",ibes_ibes_crsp_link[,"analys"],sep="")),] ##delete duplicates

base::load("ibes_experience_processed.Rdata") ##import data

ibes<-base::merge(ibes_ibes_crsp_link,ibes_experience,all=FALSE) ##merge

stopifnot(base::nrow(ibes_ibes_crsp_link)==base::nrow(ibes)) ##security break

base::load("ibes_portfolio_processed.Rdata") ##import data

ibes<-base::merge(ibes,ibes_portfolio,all=FALSE) ##merge

stopifnot(base::nrow(ibes_ibes_crsp_link)==base::nrow(ibes)) ##security break

base::load("ibes_actions_processed.Rdata") ##import data

ibes<-base::merge(ibes,ibes_actions,all=FALSE) ##merge

stopifnot(base::nrow(ibes_ibes_crsp_link)==base::nrow(ibes)) ##security break

base::load("ibes_expertise_processed.Rdata") ##import data

ibes<-base::merge(ibes,ibes_expertise,all=FALSE) ##merge

stopifnot(base::nrow(ibes_ibes_crsp_link)==base::nrow(ibes)) ##security break

list<-base::ls() ##clean
list<-list[!list %in% c("ibes")] ##clean
base::rm(list=list) ##clean

ibes_a<-ibes ##create temp data
ibes_b<-ibes ##create temp data

base::colnames(ibes_a)<-base::paste("a_",base::colnames(ibes_a),sep="") ##add prefix
base::colnames(ibes_b)<-base::paste("b_",base::colnames(ibes_b),sep="") ##add prefix

ibes_common<-base::merge(ibes_a,ibes_b,by.x=c("a_analys","a_year"),by.y=c("b_analys","b_year"),all.x=FALSE,all.y=FALSE) ##merge

ibes_common<-ibes_common[ibes_common[,"a_permno"]!=ibes_common[,"b_permno"],] ##reduce data

ibes_common[,"common_analys"]<-1 ##generate variable

ibes_common[,"common_actionso050"]<-base::ifelse(ibes_common[,"a_actionso050"]==1&ibes_common[,"b_actionso050"]==1,1,0) ##generate variable
ibes_common[,"common_actionsu050"]<-base::ifelse(ibes_common[,"a_actionso050"]==0&ibes_common[,"b_actionso050"]==0,1,0) ##generate variable
ibes_common[,"only_a_actionso050"]<-base::ifelse(ibes_common[,"a_actionso050"]==1&ibes_common[,"b_actionso050"]==0,1,0) ##generate variable
ibes_common[,"only_b_actionso050"]<-base::ifelse(ibes_common[,"a_actionso050"]==0&ibes_common[,"b_actionso050"]==1,1,0) ##generate variable

dummiesacommonanalys<-c("common_analys") ##create vector of dummies
dummiesactionsone<-base::grep("common_actions",base::colnames(ibes_common),value=TRUE) ##create vector of dummies
dummiesactionstwo<-base::grep("only_a_actions",base::colnames(ibes_common),value=TRUE) ##create vector of dummies
dummiesactionsthree<-base::grep("only_b_actions",base::colnames(ibes_common),value=TRUE) ##create vector of dummies
dummiesexperience<-base::grep("a_expy",base::colnames(ibes_common),value=TRUE) ##create vector of dummies
dummiesportfolio<-base::grep("a_port",base::colnames(ibes_common),value=TRUE) ##create vector of dummies
dummiesexpertise<-base::grep("a_hiind",base::colnames(ibes_common),value=TRUE)
```

```
##create vector of dummies

ibes_common<-stats::aggregate(ibes_common[c(dummiesacommonanalys,dummiesactionsone,dummiesactionstwo,dummiesactionsthree,dummiesexperience,dummiesportfolio,dummiesexpertise)],ibes_common[c("a_permno","b_permno","a_year")],base::sum,na.rm=TRUE) ##aggregate data

consistencycheck<-base::nrow(ibes_common) ##security break
ibes_common<-ibes_common[!base::duplicated(base::paste(ibes_common[,"a_permno"],"_",ibes_common[,"b_permno"],"_",ibes_common[,"a_year"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(ibes_common)) ##security break

base::save(ibes_common,file="ibes_common_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares ibes total data for further processing

#prepare data-----------------------------------------------------------------------
base::load("ibes_ibes_crsp_link_processed.Rdata") ##import data

ibes_ibes_crsp_link<-ibes_ibes_crsp_link[!base::duplicated(base::paste(ibes_ibes_crsp_link[,"permno"],"_",ibes_ibes_crsp_link[,"year"],"_",ibes_ibes_crsp_link[,"analys"],sep="")),] ##delete duplicates
ibes_ibes_crsp_link[,"total_analys"]<-1 ##generate variable

ibes_total<-stats::aggregate(ibes_ibes_crsp_link["total_analys"],ibes_ibes_crsp_link[c("permno","year")],base::sum,na.rm=TRUE) ##aggregate data

base::save(ibes_total,file="ibes_total_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end--------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares funda data for further processing

#prepare data----------------------------------------------------------------------
funda<-haven::read_dta("crsp_compustat_merged_fundamentals_annual_20180318.dta") ##import data
funda<-base::as.data.frame(funda) ##change to data frame

funda[,"lpermno"]<-base::as.character(funda[,"lpermno"]) ##transform variable to character
funda[,"fyear"]<-base::as.character(funda[,"fyear"]) ##transform variable to character

funda<-funda[!base::is.na(funda[,"fyear"]),] ##delete observations with missing data
funda<-funda[!base::is.na(funda[,"lpermno"]),] ##delete observations with missing data

funda<-funda[!base::is.na(funda[,"at"]),] ##delete observations with missing data

funda<-funda[funda[,"indfmt"]=="INDL",] ##assure clean data ##http://www.kaikaichen.com/?p=387
funda<-funda[funda[,"datafmt"]=="STD",] ##assure clean data
funda<-funda[funda[,"popsrc"]=="D",] ##assure clean data
funda<-funda[funda[,"consol"]=="C",] ##assure clean data

funda<-funda[!base::duplicated(base::paste(funda[,"lpermno"],"_",funda[,"fyear"],sep=" ")),] ##delete duplicates

funda[,"xrd_adj"]<-base::ifelse(base::is.na(funda[,"xrd"]),0,funda[,"xrd"])
funda[,"randdassets"]<-funda[,"xrd_adj"]/funda[,"at"]
funda[,"roa"]<-funda[,"oibdp"]/funda[,"at"]
funda[,"capexassets"]<-funda[,"capx"]/funda[,"at"]
funda[,"ppeassets"]<-funda[,"ppent"]/funda[,"at"]
funda[,"leverage"]<-(funda[,"dltt"]+funda[,"dlc"])/funda[,"at"]

funda[,"txdb_adj"]<-base::ifelse(base::is.na(funda[,"txdb"]),0,funda[,"txdb"])
funda[,"tobinq"]<-((funda[,"csho"]*funda[,"prcc_f"])+funda[,"at"]-funda[,"ceq"]-funda[,"txdb_adj"])/funda["at"]

funda<-funda[,c("lpermno","gvkey","fyear","datadate","cusip","tic","state","linkdt","linkenddt","at","sale","xrd","randdassets","roa","capexassets","ppeassets","leverage","tobinq")] ##delete unnecessary variables

consistencycheck<-base::nrow(funda) ##security break
funda<-funda[!base::duplicated(base::paste(funda[,"lpermno"],"_",funda[,"fyear"],sep=" ")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(funda)) ##security break

base::save(funda,file="funda_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares hpsimilarity data for further processing

#prepare data-----------------------------------------------------------------------
hpsimilarity<-utils::read.delim("tnic2_allyears_extend_scores_HIGRAN_20180701.txt") ##import data
hpsimilarity<-base::as.data.frame(hpsimilarity) ##change to data frame

hpsimilarity[,"year"]<-base::as.character(hpsimilarity[,"year"]) ##transform variable to character
hpsimilarity[,"gvkey1"]<-base::as.character(hpsimilarity[,"gvkey1"]) ##transform variable to character
hpsimilarity[,"gvkey2"]<-base::as.character(hpsimilarity[,"gvkey2"]) ##transform variable to character

hpsimilarity[,"hp_score"]<-hpsimilarity[,"score"] ##rename variable
hpsimilarity[,"score"]<-NULL ##delete variable

consistencycheck<-base::nrow(hpsimilarity) ##security break
hpsimilarity<-hpsimilarity[!base::duplicated(base::paste(hpsimilarity[,"year"],"_",hpsimilarity[,"gvkey1"],"_",hpsimilarity[,"gvkey2"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(hpsimilarity)) ##security break

base::save(hpsimilarity,file="hpsimilarity_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares sic data for further processing

#prepare data----------------------------------------------------------------------
sic2<-readr::read_delim("sic2_20190326.csv",";",escape_double=FALSE,trim_ws=TRUE,col_types= cols(sic2=col_double(),descriptions=col_character())) ##import data
sic2<-base::as.data.frame(sic2) ##transform to data frame

base::save(sic2,file="sic2_processed.rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares the inventors common data for further processing

#start-----------------------------------------------------------------------
patent_inventor_link<-readr::read_delim("patent_inventor_20190306.tsv","\t",escape_double=FALSE,trim_w=TRUE,col_types="cc",progress=FALSE) ##import data
patent_inventor_link<-base::as.data.frame(patent_inventor_link) ##change to data frame

##see notes 2_example_for_duplicates_in_inventor_patent_link
patent_inventor_link<-patent_inventor_link[!base::duplicated(base::paste(patent_inventor_link[,"patent_id"],"_",patent_inventor_link[,"inventor_id"])),] ##delete duplicates
##see notes 2_example_for_duplicates_in_inventor_patent_link

consistencycheck<-base::nrow(patent_inventor_link) ##security break
patent_inventor_link<-patent_inventor_link[!base::duplicated(base::paste(patent_inventor_link[,"patent_id"],"_",patent_inventor_link[,"inventor_id"])),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(patent_inventor_link)) ##security break

base::load(file="patents_processed.Rdata") ##import data

patents_patent_inventor_link<-base::merge(patents,patent_inventor_link,by.x=c("patnum"),by.y=c("patent_id"),all=FALSE)

patents_patent_inventor_link<-patents_patent_inventor_link[,c("permno","fyear","inventor_id")]

patents_patent_inventor_link<-patents_patent_inventor_link[!base::duplicated(base::paste(patents_patent_inventor_link[,"permno"],"_",patents_patent_inventor_link[,"fyear"],"_",patents_patent_inventor_link[,"inventor_id"])),] ##delete duplicates

patents_patent_inventor_link[,"fyearm1"]<-base::as.numeric(patents_patent_inventor_link[,"fyear"])-1
patents_patent_inventor_link[,"fyearm2"]<-base::as.numeric(patents_patent_inventor_link[,"fyear"])-2
patents_patent_inventor_link[,"fyearm3"]<-base::as.numeric(patents_patent_inventor_link[,"fyear"])-3

patents_patent_inventor_link_a<-patents_patent_inventor_link ##create temp data
patents_patent_inventor_link_b<-patents_patent_inventor_link ##create temp data

base::colnames(patents_patent_inventor_link_a)<-base::paste("a_",base::colnames(patents_patent_inventor_link_a),sep="") ##add prefix
base::colnames(patents_patent_inventor_link_b)<-base::paste("b_",base::colnames(patents_patent_inventor_link_b),sep="") ##add prefix

patents_patent_inventor_link_common_part0<-base::merge(patents_patent_inventor_link_a,patents_patent_inventor_link_b,by.x=c("a_inventor_id","a_fyear"),by.y=c("b_inventor_id","b_fyear"),all.x=FALSE,all.y=FALSE) ##merge
patents_patent_inventor_link_common_part1<-base::merge(patents_patent_inventor_link_a,patents_patent_inventor_link_b,by.x=c("a_inventor_id","a_fyearm1"),by.y=c("b_inventor_id","b_fyear"),all.x=FALSE,all.y=FALSE) ##merge
patents_patent_inventor_link_common_part2<-base::merge(patents_patent_inventor_link_a,patents_patent_inventor_link_b,by.x=c("a_inventor_id","a_fyearm2"),by.y=c("b_inventor_id","b_fyear"),all.x=FALSE,all.y=FALSE) ##merge
patents_patent_inventor_link_common_part3<-base::merge(patents_patent_inventor_link_a,patents_patent_inventor_link_b,by.x=c("a_inventor_id","a_fyearm3"),by.y=c("b_inventor_id","b_fyear"),all.x=FALSE,all.y=FALSE) ##merge

patents_patent_inventor_link_common<-base::rbind(patents_patent_inventor_link_common_part0,patents_patent_inventor_link_common_part1,patents_patent_inventor_link_common_part2,patents_patent_inventor_link_common_part3)

patents_patent_inventor_link_common<-patents_patent_inventor_link_common[patents_patent_inventor_link_common[,"a_permno"]!=patents_patent_inventor_link_common[,"b_permno"],] ##reduce data

patents_patent_inventor_link_common<-patents_patent_inventor_link_common[!base::duplicated(base::paste(patents_patent_inventor_link_common[,"a_permno"],"_",patents_patent_inventor_link_common[,"b_permno"],"_",patents_patent_inventor_link_common[,"a_inventor_id"],"_",patents_patent_inventor_link_common[,"a_fyear"])),] ##delete duplicates

patents_patent_inventor_link_common[,"common_inventors"]<-1 ##generate variable

```

```r
inventors_common<-stats::aggregate(patents_patent_inventor_link_common[c("common_inventors")],patents_patent_inventor_link_common[c("a_permno","b_permno","a_fyear")],base::sum,na.rm=TRUE) ##aggregate data

consistencycheck<-base::nrow(inventors_common) ##security break
inventors_common<-inventors_common[!base::duplicated(base::paste(inventors_common[,"a_permno"],"_",inventors_common[,"b_permno"],"_",inventors_common[,"a_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(inventors_common)) ##security break

base::save(inventors_common,file="inventors_common_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares investors common data for further processing

#prepare data-----------------------------------------------------------------------
investors_common<-haven::read_dta("ggl_measures_of_common_ownership_bi-directional_pair_year_data_20200330.dta")
investors_common<-base::as.data.frame(investors_common)

investors_common[,"gvkey_a"]<-base::as.character(base::as.numeric(investors_common[,"gvkey_a"]))
investors_common[,"gvkey_b"]<-base::as.character(base::as.numeric(investors_common[,"gvkey_b"]))
investors_common[,"year"]<-base::as.character(investors_common[,"year"])

base::save(investors_common,file="investors_common_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end--------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares alliance common data for further processing

#prepare data-----------------------------------------------------------------------
alliances<-readxl::read_excel("SDC_1962_2019_20200312.xlsx",col_names=TRUE,skip=1)
alliances<-base::as.data.frame(alliances)

alliances[,"year_announced"]<-base::gsub(".{6}$","",alliances[,"date_announced"])

alliances2firms<-dplyr::filter(alliances,base::is.na(firm3)&!base::is.na(firm2))

##1 and 2##
alliances2firms[,"singleindentifiera"]<-base::ifelse(alliances2firms[,"cusip_firm1"]>alliances2firms[,"cusip_firm2"],alliances2firms[,"cusip_firm1"],alliances2firms[,"cusip_firm2"]) ##create undirectional identifier
alliances2firms[,"singleindentifierb"]<-base::ifelse(alliances2firms[,"cusip_firm1"]>alliances2firms[,"cusip_firm2"],alliances2firms[,"cusip_firm2"],alliances2firms[,"cusip_firm1"]) ##create undirectional identifier
alliances2firms[,"singleindentifier"]<-base::paste(alliances2firms[,"singleindentifiera"],"_",alliances2firms[,"singleindentifierb"],sep="") ##create undirectional identifier

alliances3firms<-dplyr::filter(alliances,base::is.na(firm4)&!base::is.na(firm3))

alliances3firmspart1<-alliances3firms
alliances3firmspart2<-alliances3firms
alliances3firmspart3<-alliances3firms

##1 and 2##
alliances3firmspart1[,"singleindentifiera"]<-base::ifelse(alliances3firmspart1[,"cusip_firm1"]>alliances3firmspart1[,"cusip_firm2"],alliances3firmspart1[,"cusip_firm1"],alliances3firmspart1[,"cusip_firm2"]) ##create undirectional identifier
alliances3firmspart1[,"singleindentifierb"]<-base::ifelse(alliances3firmspart1[,"cusip_firm1"]>alliances3firmspart1[,"cusip_firm2"],alliances3firmspart1[,"cusip_firm2"],alliances3firmspart1[,"cusip_firm1"]) ##create undirectional identifier
alliances3firmspart1[,"singleindentifier"]<-base::paste(alliances3firmspart1[,"singleindentifiera"],"_",alliances3firmspart1[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 3##
alliances3firmspart2[,"singleindentifiera"]<-base::ifelse(alliances3firmspart2[,"cusip_firm1"]>alliances3firmspart2[,"cusip_firm3"],alliances3firmspart2[,"cusip_firm1"],alliances3firmspart2[,"cusip_firm3"]) ##create undirectional identifier
alliances3firmspart2[,"singleindentifierb"]<-base::ifelse(alliances3firmspart2[,"cusip_firm1"]>alliances3firmspart2[,"cusip_firm3"],alliances3firmspart2[,"cusip_firm3"],alliances3firmspart2[,"cusip_firm1"]) ##create undirectional identifier
alliances3firmspart2[,"singleindentifier"]<-base::paste(alliances3firmspart2[,"singleindentifiera"],"_",alliances3firmspart2[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 3##
alliances3firmspart3[,"singleindentifiera"]<-base::ifelse(alliances3firmspart3[,"cusip_firm2"]>alliances3firmspart3[,"cusip_firm3"],alliances3firmspart3[,"cusip_firm2"],alliances3firmspart3[,"cusip_firm3"]) ##create undirectional identifier
alliances3firmspart3[,"singleindentifierb"]<-base::ifelse(alliances3firmspart3[,"cusip_firm2"]>alliances3firmspart3[,"cusip_firm3"],alliances3firmspart3[,"cusip_firm3"],alliances3firmspart3[,"cusip_firm2"]) ##create undirectional identifier
alliances3firmspart3[,"singleindentifier"]<-base::paste(alliances3firmspart3[,"singleindentifiera"],"_",alliances3firmspart3[,"singleindentifierb"],sep="") ##create undirectional identifier

alliances3firms<-base::rbind(alliances3firmspart1,alliances3firmspart2,alliances3firmspart3)

base::rm(alliances3firmspart1)
base::rm(alliances3firmspart2)
base::rm(alliances3firmspart3)

alliances4firms<-dplyr::filter(alliances,base::is.na(firm5)&!base::is.na(firm4))

alliances4firmspart1<-alliances4firms
alliances4firmspart2<-alliances4firms
alliances4firmspart3<-alliances4firms
alliances4firmspart4<-alliances4firms
alliances4firmspart5<-alliances4firms
```

```r
alliances4firmspart6<-alliances4firms

##1 and 2##
alliances4firmspart1[,"singleindentifiera"]<-base::ifelse(alliances4firmspart1[,"cusip_firm1"]>alliances4firmspart1[,"cusip_firm2"],alliances4firmspart1[,"cusip_firm1"],alliances4firmspart1[,"cusip_firm2"]) ##create undirectional identifier
alliances4firmspart1[,"singleindentifierb"]<-base::ifelse(alliances4firmspart1[,"cusip_firm1"]>alliances4firmspart1[,"cusip_firm2"],alliances4firmspart1[,"cusip_firm2"],alliances4firmspart1[,"cusip_firm1"]) ##create undirectional identifier
alliances4firmspart1[,"singleindentifier"]<-base::paste(alliances4firmspart1[,"singleindentifiera"],"_",alliances4firmspart1[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 3##
alliances4firmspart2[,"singleindentifiera"]<-base::ifelse(alliances4firmspart2[,"cusip_firm1"]>alliances4firmspart2[,"cusip_firm3"],alliances4firmspart2[,"cusip_firm1"],alliances4firmspart2[,"cusip_firm3"]) ##create undirectional identifier
alliances4firmspart2[,"singleindentifierb"]<-base::ifelse(alliances4firmspart2[,"cusip_firm1"]>alliances4firmspart2[,"cusip_firm3"],alliances4firmspart2[,"cusip_firm3"],alliances4firmspart2[,"cusip_firm1"]) ##create undirectional identifier
alliances4firmspart2[,"singleindentifier"]<-base::paste(alliances4firmspart2[,"singleindentifiera"],"_",alliances4firmspart2[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 4##
alliances4firmspart3[,"singleindentifiera"]<-base::ifelse(alliances4firmspart3[,"cusip_firm1"]>alliances4firmspart3[,"cusip_firm4"],alliances4firmspart3[,"cusip_firm1"],alliances4firmspart3[,"cusip_firm4"]) ##create undirectional identifier
alliances4firmspart3[,"singleindentifierb"]<-base::ifelse(alliances4firmspart3[,"cusip_firm1"]>alliances4firmspart3[,"cusip_firm4"],alliances4firmspart3[,"cusip_firm4"],alliances4firmspart3[,"cusip_firm1"]) ##create undirectional identifier
alliances4firmspart3[,"singleindentifier"]<-base::paste(alliances4firmspart3[,"singleindentifiera"],"_",alliances4firmspart3[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 3##
alliances4firmspart4[,"singleindentifiera"]<-base::ifelse(alliances4firmspart4[,"cusip_firm2"]>alliances4firmspart4[,"cusip_firm3"],alliances4firmspart4[,"cusip_firm2"],alliances4firmspart4[,"cusip_firm3"]) ##create undirectional identifier
alliances4firmspart4[,"singleindentifierb"]<-base::ifelse(alliances4firmspart4[,"cusip_firm2"]>alliances4firmspart4[,"cusip_firm3"],alliances4firmspart4[,"cusip_firm3"],alliances4firmspart4[,"cusip_firm2"]) ##create undirectional identifier
alliances4firmspart4[,"singleindentifier"]<-base::paste(alliances4firmspart4[,"singleindentifiera"],"_",alliances4firmspart4[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 4##
alliances4firmspart5[,"singleindentifiera"]<-base::ifelse(alliances4firmspart5[,"cusip_firm2"]>alliances4firmspart5[,"cusip_firm4"],alliances4firmspart5[,"cusip_firm2"],alliances4firmspart5[,"cusip_firm4"]) ##create undirectional identifier
alliances4firmspart5[,"singleindentifierb"]<-base::ifelse(alliances4firmspart5[,"cusip_firm2"]>alliances4firmspart5[,"cusip_firm4"],alliances4firmspart5[,"cusip_firm4"],alliances4firmspart5[,"cusip_firm2"]) ##create undirectional identifier
alliances4firmspart5[,"singleindentifier"]<-base::paste(alliances4firmspart5[,"singleindentifiera"],"_",alliances4firmspart5[,"singleindentifierb"],sep="") ##create undirectional identifier
##3 and 4##
alliances4firmspart6[,"singleindentifiera"]<-base::ifelse(alliances4firmspart6[,"cusip_firm3"]>alliances4firmspart6[,"cusip_firm4"],alliances4firmspart6[,"cusip_firm3"],alliances4firmspart6[,"cusip_firm4"]) ##create undirectional identifier
alliances4firmspart6[,"singleindentifierb"]<-base::ifelse(alliances4firmspart6[,"cusip_firm3"]>alliances4firmspart6[,"cusip_firm4"],alliances4firmspart6[,"cusip_firm4"],alliances4firmspart6[,"cusip_firm3"]) ##create undirectional identifier
alliances4firmspart6[,"singleindentifier"]<-base::paste(alliances4firmspart6[,"singleindentifiera"],"_",alliances4firmspart6[,"singleindentifierb"],sep="") ##create undirectional identifier

alliances4firms<-base::rbind(alliances4firmspart1,alliances4firmspart2,alliances4firmspart3,alliances4firmspart4,alliances4firmspart5,alliances4firmspart6)

base::rm(alliances4firmspart1)
base::rm(alliances4firmspart2)
base::rm(alliances4firmspart3)
base::rm(alliances4firmspart4)
base::rm(alliances4firmspart5)
base::rm(alliances4firmspart6)
```

```
alliances5firms<-dplyr::filter(alliances,base::is.na(firm6)&!base::is.na(firm5))

alliances5firmspart1<-alliances5firms
alliances5firmspart2<-alliances5firms
alliances5firmspart3<-alliances5firms
alliances5firmspart4<-alliances5firms
alliances5firmspart5<-alliances5firms
alliances5firmspart6<-alliances5firms
alliances5firmspart7<-alliances5firms
alliances5firmspart8<-alliances5firms
alliances5firmspart9<-alliances5firms
alliances5firmspart10<-alliances5firms

##1 and 2##
alliances5firmspart1[,"singleindentifiera"]<-base::ifelse(alliances5firmspart1[,"cusip_firm1"]>alliances5firmspart1[,"cusip_firm2"],alliances5firmspart1[,"cusip_firm1"],alliances5firmspart1[,"cusip_firm2"]) ##create undirectional identifier
alliances5firmspart1[,"singleindentifierb"]<-base::ifelse(alliances5firmspart1[,"cusip_firm1"]>alliances5firmspart1[,"cusip_firm2"],alliances5firmspart1[,"cusip_firm2"],alliances5firmspart1[,"cusip_firm1"]) ##create undirectional identifier
alliances5firmspart1[,"singleindentifier"]<-base::paste(alliances5firmspart1[,"singleindentifiera"],"_",alliances5firmspart1[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 3##
alliances5firmspart2[,"singleindentifiera"]<-base::ifelse(alliances5firmspart2[,"cusip_firm1"]>alliances5firmspart2[,"cusip_firm3"],alliances5firmspart2[,"cusip_firm1"],alliances5firmspart2[,"cusip_firm3"]) ##create undirectional identifier
alliances5firmspart2[,"singleindentifierb"]<-base::ifelse(alliances5firmspart2[,"cusip_firm1"]>alliances5firmspart2[,"cusip_firm3"],alliances5firmspart2[,"cusip_firm3"],alliances5firmspart2[,"cusip_firm1"]) ##create undirectional identifier
alliances5firmspart2[,"singleindentifier"]<-base::paste(alliances5firmspart2[,"singleindentifiera"],"_",alliances5firmspart2[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 4##
alliances5firmspart3[,"singleindentifiera"]<-base::ifelse(alliances5firmspart3[,"cusip_firm1"]>alliances5firmspart3[,"cusip_firm4"],alliances5firmspart3[,"cusip_firm1"],alliances5firmspart3[,"cusip_firm4"]) ##create undirectional identifier
alliances5firmspart3[,"singleindentifierb"]<-base::ifelse(alliances5firmspart3[,"cusip_firm1"]>alliances5firmspart3[,"cusip_firm4"],alliances5firmspart3[,"cusip_firm4"],alliances5firmspart3[,"cusip_firm1"]) ##create undirectional identifier
alliances5firmspart3[,"singleindentifier"]<-base::paste(alliances5firmspart3[,"singleindentifiera"],"_",alliances5firmspart3[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 5##
alliances5firmspart4[,"singleindentifiera"]<-base::ifelse(alliances5firmspart4[,"cusip_firm1"]>alliances5firmspart4[,"cusip_firm5"],alliances5firmspart4[,"cusip_firm1"],alliances5firmspart4[,"cusip_firm5"]) ##create undirectional identifier
alliances5firmspart4[,"singleindentifierb"]<-base::ifelse(alliances5firmspart4[,"cusip_firm1"]>alliances5firmspart4[,"cusip_firm5"],alliances5firmspart4[,"cusip_firm5"],alliances5firmspart4[,"cusip_firm1"]) ##create undirectional identifier
alliances5firmspart4[,"singleindentifier"]<-base::paste(alliances5firmspart4[,"singleindentifiera"],"_",alliances5firmspart4[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 3##
alliances5firmspart5[,"singleindentifiera"]<-base::ifelse(alliances5firmspart5[,"cusip_firm2"]>alliances5firmspart5[,"cusip_firm3"],alliances5firmspart5[,"cusip_firm2"],alliances5firmspart5[,"cusip_firm3"]) ##create undirectional identifier
alliances5firmspart5[,"singleindentifierb"]<-base::ifelse(alliances5firmspart5[,"cusip_firm2"]>alliances5firmspart5[,"cusip_firm3"],alliances5firmspart5[,"cusip_firm3"],alliances5firmspart5[,"cusip_firm2"]) ##create undirectional identifier
alliances5firmspart5[,"singleindentifier"]<-base::paste(alliances5firmspart5[,"singleindentifiera"],"_",alliances5firmspart5[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 4##
alliances5firmspart6[,"singleindentifiera"]<-base::ifelse(alliances5firmspart6[,"cusip_firm2"]>alliances5firmspart6[,"cusip_firm4"],alliances5firmspart6[,"cusip_firm2"],alliances5firmspart6[,"cusip_firm4"]) ##create undirectional identifier
alliances5firmspart6[,"singleindentifierb"]<-base::ifelse(alliances5firmspart6[,"cusip_firm2"]>alliances5firmspart6[,"cusip_firm4"],alliances5firmspart6[,"cusip_firm4"],alliances5firmspart6[,"cusip_firm2"]) ##create undirectional identifier
alliances5firmspart6[,"singleindentifier"]<-base::paste(alliances5firmspart6[,"singleindentifiera"],"_",alliances5firmspart6[,"singleindentifierb"],sep="") ##create undirectional identifier
```

```r
##2 and 5##
alliances5firmspart7[,"singleindentifiera"]<-base::ifelse(alliances5firmspart7[,"cusip_firm2"]>alliances5firmspart7[,"cusip_firm5"],alliances5firmspart7[,"cusip_firm2"],alliances5firmspart7[,"cusip_firm5"]) ##create undirectional identifier
alliances5firmspart7[,"singleindentifierb"]<-base::ifelse(alliances5firmspart7[,"cusip_firm2"]>alliances5firmspart7[,"cusip_firm5"],alliances5firmspart7[,"cusip_firm5"],alliances5firmspart7[,"cusip_firm2"]) ##create undirectional identifier
alliances5firmspart7[,"singleindentifier"]<-base::paste(alliances5firmspart7[,"singleindentifiera"],"_",alliances5firmspart7[,"singleindentifierb"],sep="") ##create undirectional identifier
##3 and 4##
alliances5firmspart8[,"singleindentifiera"]<-base::ifelse(alliances5firmspart8[,"cusip_firm3"]>alliances5firmspart8[,"cusip_firm4"],alliances5firmspart8[,"cusip_firm3"],alliances5firmspart8[,"cusip_firm4"]) ##create undirectional identifier
alliances5firmspart8[,"singleindentifierb"]<-base::ifelse(alliances5firmspart8[,"cusip_firm3"]>alliances5firmspart8[,"cusip_firm4"],alliances5firmspart8[,"cusip_firm4"],alliances5firmspart8[,"cusip_firm3"]) ##create undirectional identifier
alliances5firmspart8[,"singleindentifier"]<-base::paste(alliances5firmspart8[,"singleindentifiera"],"_",alliances5firmspart8[,"singleindentifierb"],sep="") ##create undirectional identifier
##3 and 5##
alliances5firmspart9[,"singleindentifiera"]<-base::ifelse(alliances5firmspart9[,"cusip_firm3"]>alliances5firmspart9[,"cusip_firm5"],alliances5firmspart9[,"cusip_firm3"],alliances5firmspart9[,"cusip_firm5"]) ##create undirectional identifier
alliances5firmspart9[,"singleindentifierb"]<-base::ifelse(alliances5firmspart9[,"cusip_firm3"]>alliances5firmspart9[,"cusip_firm5"],alliances5firmspart9[,"cusip_firm5"],alliances5firmspart9[,"cusip_firm3"]) ##create undirectional identifier
alliances5firmspart9[,"singleindentifier"]<-base::paste(alliances5firmspart9[,"singleindentifiera"],"_",alliances5firmspart9[,"singleindentifierb"],sep="") ##create undirectional identifier
##4 and 5##
alliances5firmspart10[,"singleindentifiera"]<-base::ifelse(alliances5firmspart10[,"cusip_firm4"]>alliances5firmspart10[,"cusip_firm5"],alliances5firmspart10[,"cusip_firm4"],alliances5firmspart10[,"cusip_firm5"]) ##create undirectional identifier
alliances5firmspart10[,"singleindentifierb"]<-base::ifelse(alliances5firmspart10[,"cusip_firm4"]>alliances5firmspart10[,"cusip_firm5"],alliances5firmspart10[,"cusip_firm5"],alliances5firmspart10[,"cusip_firm4"]) ##create undirectional identifier
alliances5firmspart10[,"singleindentifier"]<-base::paste(alliances5firmspart10[,"singleindentifiera"],"_",alliances5firmspart10[,"singleindentifierb"],sep="") ##create undirectional identifier

alliances5firms<-base::rbind(alliances5firmspart1,alliances5firmspart2,alliances5firmspart3,alliances5firmspart4,alliances5firmspart5,alliances5firmspart6,alliances5firmspart7,alliances5firmspart8,alliances5firmspart9,alliances5firmspart10)

base::rm(alliances5firmspart1)
base::rm(alliances5firmspart2)
base::rm(alliances5firmspart3)
base::rm(alliances5firmspart4)
base::rm(alliances5firmspart5)
base::rm(alliances5firmspart6)
base::rm(alliances5firmspart7)
base::rm(alliances5firmspart8)
base::rm(alliances5firmspart9)
base::rm(alliances5firmspart10)

alliances6firms<-dplyr::filter(alliances,base::is.na(firm7)&!base::is.na(firm6))

alliances6firmspart1<-alliances6firms
alliances6firmspart2<-alliances6firms
alliances6firmspart3<-alliances6firms
alliances6firmspart4<-alliances6firms
alliances6firmspart5<-alliances6firms
alliances6firmspart6<-alliances6firms
alliances6firmspart7<-alliances6firms
alliances6firmspart8<-alliances6firms
alliances6firmspart9<-alliances6firms
alliances6firmspart10<-alliances6firms
alliances6firmspart11<-alliances6firms
alliances6firmspart12<-alliances6firms
alliances6firmspart13<-alliances6firms
alliances6firmspart14<-alliances6firms
alliances6firmspart15<-alliances6firms
```

```
##1 and 2##
alliances6firmspart1[,"singleindentifiera"]<-base::ifelse(alliances6firmspart1[,"cusip_firm1"]>alliances6firmspart1[,"cusip_firm2"],alliances6firmspart1[,"cusip_firm1"],alliances6firmspart1[,"cusip_firm2"]) ##create undirectional identifier
alliances6firmspart1[,"singleindentifierb"]<-base::ifelse(alliances6firmspart1[,"cusip_firm1"]>alliances6firmspart1[,"cusip_firm2"],alliances6firmspart1[,"cusip_firm2"],alliances6firmspart1[,"cusip_firm1"]) ##create undirectional identifier
alliances6firmspart1[,"singleindentifier"]<-base::paste(alliances6firmspart1[,"singleindentifiera"],"_",alliances6firmspart1[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 3##
alliances6firmspart2[,"singleindentifiera"]<-base::ifelse(alliances6firmspart2[,"cusip_firm1"]>alliances6firmspart2[,"cusip_firm3"],alliances6firmspart2[,"cusip_firm1"],alliances6firmspart2[,"cusip_firm3"]) ##create undirectional identifier
alliances6firmspart2[,"singleindentifierb"]<-base::ifelse(alliances6firmspart2[,"cusip_firm1"]>alliances6firmspart2[,"cusip_firm3"],alliances6firmspart2[,"cusip_firm3"],alliances6firmspart2[,"cusip_firm1"]) ##create undirectional identifier
alliances6firmspart2[,"singleindentifier"]<-base::paste(alliances6firmspart2[,"singleindentifiera"],"_",alliances6firmspart2[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 4##
alliances6firmspart3[,"singleindentifiera"]<-base::ifelse(alliances6firmspart3[,"cusip_firm1"]>alliances6firmspart3[,"cusip_firm4"],alliances6firmspart3[,"cusip_firm1"],alliances6firmspart3[,"cusip_firm4"]) ##create undirectional identifier
alliances6firmspart3[,"singleindentifierb"]<-base::ifelse(alliances6firmspart3[,"cusip_firm1"]>alliances6firmspart3[,"cusip_firm4"],alliances6firmspart3[,"cusip_firm4"],alliances6firmspart3[,"cusip_firm1"]) ##create undirectional identifier
alliances6firmspart3[,"singleindentifier"]<-base::paste(alliances6firmspart3[,"singleindentifiera"],"_",alliances6firmspart3[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 5##
alliances6firmspart4[,"singleindentifiera"]<-base::ifelse(alliances6firmspart4[,"cusip_firm1"]>alliances6firmspart4[,"cusip_firm5"],alliances6firmspart4[,"cusip_firm1"],alliances6firmspart4[,"cusip_firm5"]) ##create undirectional identifier
alliances6firmspart4[,"singleindentifierb"]<-base::ifelse(alliances6firmspart4[,"cusip_firm1"]>alliances6firmspart4[,"cusip_firm5"],alliances6firmspart4[,"cusip_firm5"],alliances6firmspart4[,"cusip_firm1"]) ##create undirectional identifier
alliances6firmspart4[,"singleindentifier"]<-base::paste(alliances6firmspart4[,"singleindentifiera"],"_",alliances6firmspart4[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 6##
alliances6firmspart5[,"singleindentifiera"]<-base::ifelse(alliances6firmspart5[,"cusip_firm1"]>alliances6firmspart5[,"cusip_firm6"],alliances6firmspart5[,"cusip_firm1"],alliances6firmspart5[,"cusip_firm6"]) ##create undirectional identifier
alliances6firmspart5[,"singleindentifierb"]<-base::ifelse(alliances6firmspart5[,"cusip_firm1"]>alliances6firmspart5[,"cusip_firm6"],alliances6firmspart5[,"cusip_firm6"],alliances6firmspart5[,"cusip_firm1"]) ##create undirectional identifier
alliances6firmspart5[,"singleindentifier"]<-base::paste(alliances6firmspart5[,"singleindentifiera"],"_",alliances6firmspart5[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 3##
alliances6firmspart6[,"singleindentifiera"]<-base::ifelse(alliances6firmspart6[,"cusip_firm2"]>alliances6firmspart6[,"cusip_firm3"],alliances6firmspart6[,"cusip_firm2"],alliances6firmspart6[,"cusip_firm3"]) ##create undirectional identifier
alliances6firmspart6[,"singleindentifierb"]<-base::ifelse(alliances6firmspart6[,"cusip_firm2"]>alliances6firmspart6[,"cusip_firm3"],alliances6firmspart6[,"cusip_firm3"],alliances6firmspart6[,"cusip_firm2"]) ##create undirectional identifier
alliances6firmspart6[,"singleindentifier"]<-base::paste(alliances6firmspart6[,"singleindentifiera"],"_",alliances6firmspart6[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 4##
alliances6firmspart7[,"singleindentifiera"]<-base::ifelse(alliances6firmspart7[,"cusip_firm2"]>alliances6firmspart7[,"cusip_firm4"],alliances6firmspart7[,"cusip_firm2"],alliances6firmspart7[,"cusip_firm4"]) ##create undirectional identifier
alliances6firmspart7[,"singleindentifierb"]<-base::ifelse(alliances6firmspart7[,"cusip_firm2"]>alliances6firmspart7[,"cusip_firm4"],alliances6firmspart7[,"cusip_firm4"],alliances6firmspart7[,"cusip_firm2"]) ##create undirectional identifier
alliances6firmspart7[,"singleindentifier"]<-base::paste(alliances6firmspart7[,"singleindentifiera"],"_",alliances6firmspart7[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 5##
alliances6firmspart8[,"singleindentifiera"]<-base::ifelse(alliances6firmspart8[,"cusip
```

```
_firm2"]>alliances6firmspart8[,"cusip_firm5"],alliances6firmspart8[,"cusip_firm2"],all
iances6firmspart8[,"cusip_firm5"]) ##create undirectional identifier
alliances6firmspart8[,"singleindentifierb"]<-base::ifelse(alliances6firmspart8[,"cusip
_firm2"]>alliances6firmspart8[,"cusip_firm5"],alliances6firmspart8[,"cusip_firm5"],all
iances6firmspart8[,"cusip_firm2"]) ##create undirectional identifier
alliances6firmspart8[,"singleindentifier"]<-base::paste(alliances6firmspart8[,"singlei
ndentifiera"],"_",alliances6firmspart8[,"singleindentifierb"],sep="") ##create
undirectional identifier
##2 and 6##
alliances6firmspart9[,"singleindentifiera"]<-base::ifelse(alliances6firmspart9[,"cusip
_firm2"]>alliances6firmspart9[,"cusip_firm6"],alliances6firmspart9[,"cusip_firm2"],all
iances6firmspart9[,"cusip_firm6"]) ##create undirectional identifier
alliances6firmspart9[,"singleindentifierb"]<-base::ifelse(alliances6firmspart9[,"cusip
_firm2"]>alliances6firmspart9[,"cusip_firm6"],alliances6firmspart9[,"cusip_firm6"],all
iances6firmspart9[,"cusip_firm2"]) ##create undirectional identifier
alliances6firmspart9[,"singleindentifier"]<-base::paste(alliances6firmspart9[,"singlei
ndentifiera"],"_",alliances6firmspart9[,"singleindentifierb"],sep="") ##create
undirectional identifier
##3 and 4##
alliances6firmspart10[,"singleindentifiera"]<-base::ifelse(alliances6firmspart10[,"cus
ip_firm3"]>alliances6firmspart10[,"cusip_firm4"],alliances6firmspart10[,"cusip_firm3"]
,alliances6firmspart10[,"cusip_firm4"]) ##create undirectional identifier
alliances6firmspart10[,"singleindentifierb"]<-base::ifelse(alliances6firmspart10[,"cus
ip_firm3"]>alliances6firmspart10[,"cusip_firm4"],alliances6firmspart10[,"cusip_firm4"]
,alliances6firmspart10[,"cusip_firm3"]) ##create undirectional identifier
alliances6firmspart10[,"singleindentifier"]<-base::paste(alliances6firmspart10[,"singl
eindentifiera"],"_",alliances6firmspart10[,"singleindentifierb"],sep="") ##create
undirectional identifier
##3 and 5##
alliances6firmspart11[,"singleindentifiera"]<-base::ifelse(alliances6firmspart11[,"cus
ip_firm3"]>alliances6firmspart11[,"cusip_firm5"],alliances6firmspart11[,"cusip_firm3"]
,alliances6firmspart11[,"cusip_firm5"]) ##create undirectional identifier
alliances6firmspart11[,"singleindentifierb"]<-base::ifelse(alliances6firmspart11[,"cus
ip_firm3"]>alliances6firmspart11[,"cusip_firm5"],alliances6firmspart11[,"cusip_firm5"]
,alliances6firmspart11[,"cusip_firm3"]) ##create undirectional identifier
alliances6firmspart11[,"singleindentifier"]<-base::paste(alliances6firmspart11[,"singl
eindentifiera"],"_",alliances6firmspart11[,"singleindentifierb"],sep="") ##create
undirectional identifier
##3 and 6##
alliances6firmspart12[,"singleindentifiera"]<-base::ifelse(alliances6firmspart12[,"cus
ip_firm3"]>alliances6firmspart12[,"cusip_firm6"],alliances6firmspart12[,"cusip_firm3"]
,alliances6firmspart12[,"cusip_firm6"]) ##create undirectional identifier
alliances6firmspart12[,"singleindentifierb"]<-base::ifelse(alliances6firmspart12[,"cus
ip_firm3"]>alliances6firmspart12[,"cusip_firm6"],alliances6firmspart12[,"cusip_firm6"]
,alliances6firmspart12[,"cusip_firm3"]) ##create undirectional identifier
alliances6firmspart12[,"singleindentifier"]<-base::paste(alliances6firmspart12[,"singl
eindentifiera"],"_",alliances6firmspart12[,"singleindentifierb"],sep="") ##create
undirectional identifier
##4 and 5##
alliances6firmspart13[,"singleindentifiera"]<-base::ifelse(alliances6firmspart13[,"cus
ip_firm4"]>alliances6firmspart13[,"cusip_firm5"],alliances6firmspart13[,"cusip_firm4"]
,alliances6firmspart13[,"cusip_firm5"]) ##create undirectional identifier
alliances6firmspart13[,"singleindentifierb"]<-base::ifelse(alliances6firmspart13[,"cus
ip_firm4"]>alliances6firmspart13[,"cusip_firm5"],alliances6firmspart13[,"cusip_firm5"]
,alliances6firmspart13[,"cusip_firm4"]) ##create undirectional identifier
alliances6firmspart13[,"singleindentifier"]<-base::paste(alliances6firmspart13[,"singl
eindentifiera"],"_",alliances6firmspart13[,"singleindentifierb"],sep="") ##create
undirectional identifier
##4 and 6##
alliances6firmspart14[,"singleindentifiera"]<-base::ifelse(alliances6firmspart14[,"cus
ip_firm4"]>alliances6firmspart14[,"cusip_firm6"],alliances6firmspart14[,"cusip_firm4"]
,alliances6firmspart14[,"cusip_firm6"]) ##create undirectional identifier
alliances6firmspart14[,"singleindentifierb"]<-base::ifelse(alliances6firmspart14[,"cus
ip_firm4"]>alliances6firmspart14[,"cusip_firm6"],alliances6firmspart14[,"cusip_firm6"]
,alliances6firmspart14[,"cusip_firm4"]) ##create undirectional identifier
alliances6firmspart14[,"singleindentifier"]<-base::paste(alliances6firmspart14[,"singl
eindentifiera"],"_",alliances6firmspart14[,"singleindentifierb"],sep="") ##create
undirectional identifier
##5 and 6##
alliances6firmspart15[,"singleindentifiera"]<-base::ifelse(alliances6firmspart15[,"cus
ip_firm5"]>alliances6firmspart15[,"cusip_firm6"],alliances6firmspart15[,"cusip_firm5"]
,alliances6firmspart15[,"cusip_firm6"]) ##create undirectional identifier
alliances6firmspart15[,"singleindentifierb"]<-base::ifelse(alliances6firmspart15[,"cus
```

```
ip_firm5"]>alliances6firmspart15[,"cusip_firm6"],alliances6firmspart15[,"cusip_firm6"]
,alliances6firmspart15[,"cusip_firm5"]) ##create undirectional identifier
alliances6firmspart15[,"singleindentifier"]<-base::paste(alliances6firmspart15[,"singl
eindentifiera"],"_",alliances6firmspart15[,"singleindentifierb"],sep="") ##create
undirectional identifier

alliances6firms<-base::rbind(alliances6firmspart1,alliances6firmspart2,alliances6firms
part3,alliances6firmspart4,alliances6firmspart5,alliances6firmspart6,alliances6firmspa
rt7,alliances6firmspart8,alliances6firmspart9,alliances6firmspart10,alliances6firmspar
t11,alliances6firmspart12,alliances6firmspart13,alliances6firmspart14,alliances6firmsp
art15)

base::rm(alliances6firmspart1)
base::rm(alliances6firmspart2)
base::rm(alliances6firmspart3)
base::rm(alliances6firmspart4)
base::rm(alliances6firmspart5)
base::rm(alliances6firmspart6)
base::rm(alliances6firmspart7)
base::rm(alliances6firmspart8)
base::rm(alliances6firmspart9)
base::rm(alliances6firmspart10)
base::rm(alliances6firmspart11)
base::rm(alliances6firmspart12)
base::rm(alliances6firmspart13)
base::rm(alliances6firmspart14)
base::rm(alliances6firmspart15)

alliances7firms<-dplyr::filter(alliances,base::is.na(firm8)&!base::is.na(firm7))

alliances7firmspart1<-alliances7firms
alliances7firmspart2<-alliances7firms
alliances7firmspart3<-alliances7firms
alliances7firmspart4<-alliances7firms
alliances7firmspart5<-alliances7firms
alliances7firmspart6<-alliances7firms
alliances7firmspart7<-alliances7firms
alliances7firmspart8<-alliances7firms
alliances7firmspart9<-alliances7firms
alliances7firmspart10<-alliances7firms
alliances7firmspart11<-alliances7firms
alliances7firmspart12<-alliances7firms
alliances7firmspart13<-alliances7firms
alliances7firmspart14<-alliances7firms
alliances7firmspart15<-alliances7firms
alliances7firmspart16<-alliances7firms
alliances7firmspart17<-alliances7firms
alliances7firmspart18<-alliances7firms
alliances7firmspart19<-alliances7firms
alliances7firmspart20<-alliances7firms
alliances7firmspart21<-alliances7firms

##1 and 2##
alliances7firmspart1[,"singleindentifiera"]<-base::ifelse(alliances7firmspart1[,"cusip
_firm1"]>alliances7firmspart1[,"cusip_firm2"],alliances7firmspart1[,"cusip_firm1"],all
iances7firmspart1[,"cusip_firm2"]) ##create undirectional identifier
alliances7firmspart1[,"singleindentifierb"]<-base::ifelse(alliances7firmspart1[,"cusip
_firm1"]>alliances7firmspart1[,"cusip_firm2"],alliances7firmspart1[,"cusip_firm2"],all
iances7firmspart1[,"cusip_firm1"]) ##create undirectional identifier
alliances7firmspart1[,"singleindentifier"]<-base::paste(alliances7firmspart1[,"singlei
ndentifiera"],"_",alliances7firmspart1[,"singleindentifierb"],sep="") ##create
undirectional identifier
##1 and 3##
alliances7firmspart2[,"singleindentifiera"]<-base::ifelse(alliances7firmspart2[,"cusip
_firm1"]>alliances7firmspart2[,"cusip_firm3"],alliances7firmspart2[,"cusip_firm1"],all
iances7firmspart2[,"cusip_firm3"]) ##create undirectional identifier
alliances7firmspart2[,"singleindentifierb"]<-base::ifelse(alliances7firmspart2[,"cusip
_firm1"]>alliances7firmspart2[,"cusip_firm3"],alliances7firmspart2[,"cusip_firm3"],all
iances7firmspart2[,"cusip_firm1"]) ##create undirectional identifier
alliances7firmspart2[,"singleindentifier"]<-base::paste(alliances7firmspart2[,"singlei
ndentifiera"],"_",alliances7firmspart2[,"singleindentifierb"],sep="") ##create
undirectional identifier
##1 and 4##
```

```
alliances7firmspart3[,"singleindentifiera"]<-base::ifelse(alliances7firmspart3[,"cusip_firm1"]>alliances7firmspart3[,"cusip_firm4"],alliances7firmspart3[,"cusip_firm1"],alliances7firmspart3[,"cusip_firm4"]) ##create undirectional identifier
alliances7firmspart3[,"singleindentifierb"]<-base::ifelse(alliances7firmspart3[,"cusip_firm1"]>alliances7firmspart3[,"cusip_firm4"],alliances7firmspart3[,"cusip_firm4"],alliances7firmspart3[,"cusip_firm1"]) ##create undirectional identifier
alliances7firmspart3[,"singleindentifier"]<-base::paste(alliances7firmspart3[,"singleindentifiera"],"_",alliances7firmspart3[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 5##
alliances7firmspart4[,"singleindentifiera"]<-base::ifelse(alliances7firmspart4[,"cusip_firm1"]>alliances7firmspart4[,"cusip_firm5"],alliances7firmspart4[,"cusip_firm1"],alliances7firmspart4[,"cusip_firm5"]) ##create undirectional identifier
alliances7firmspart4[,"singleindentifierb"]<-base::ifelse(alliances7firmspart4[,"cusip_firm1"]>alliances7firmspart4[,"cusip_firm5"],alliances7firmspart4[,"cusip_firm5"],alliances7firmspart4[,"cusip_firm1"]) ##create undirectional identifier
alliances7firmspart4[,"singleindentifier"]<-base::paste(alliances7firmspart4[,"singleindentifiera"],"_",alliances7firmspart4[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 6##
alliances7firmspart5[,"singleindentifiera"]<-base::ifelse(alliances7firmspart5[,"cusip_firm1"]>alliances7firmspart5[,"cusip_firm6"],alliances7firmspart5[,"cusip_firm1"],alliances7firmspart5[,"cusip_firm6"]) ##create undirectional identifier
alliances7firmspart5[,"singleindentifierb"]<-base::ifelse(alliances7firmspart5[,"cusip_firm1"]>alliances7firmspart5[,"cusip_firm6"],alliances7firmspart5[,"cusip_firm6"],alliances7firmspart5[,"cusip_firm1"]) ##create undirectional identifier
alliances7firmspart5[,"singleindentifier"]<-base::paste(alliances7firmspart5[,"singleindentifiera"],"_",alliances7firmspart5[,"singleindentifierb"],sep="") ##create undirectional identifier
##1 and 7##
alliances7firmspart6[,"singleindentifiera"]<-base::ifelse(alliances7firmspart6[,"cusip_firm1"]>alliances7firmspart6[,"cusip_firm7"],alliances7firmspart6[,"cusip_firm1"],alliances7firmspart6[,"cusip_firm7"]) ##create undirectional identifier
alliances7firmspart6[,"singleindentifierb"]<-base::ifelse(alliances7firmspart6[,"cusip_firm1"]>alliances7firmspart6[,"cusip_firm7"],alliances7firmspart6[,"cusip_firm7"],alliances7firmspart6[,"cusip_firm1"]) ##create undirectional identifier
alliances7firmspart6[,"singleindentifier"]<-base::paste(alliances7firmspart6[,"singleindentifiera"],"_",alliances7firmspart6[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 3##
alliances7firmspart7[,"singleindentifiera"]<-base::ifelse(alliances7firmspart7[,"cusip_firm2"]>alliances7firmspart7[,"cusip_firm3"],alliances7firmspart7[,"cusip_firm2"],alliances7firmspart7[,"cusip_firm3"]) ##create undirectional identifier
alliances7firmspart7[,"singleindentifierb"]<-base::ifelse(alliances7firmspart7[,"cusip_firm2"]>alliances7firmspart7[,"cusip_firm3"],alliances7firmspart7[,"cusip_firm3"],alliances7firmspart7[,"cusip_firm2"]) ##create undirectional identifier
alliances7firmspart7[,"singleindentifier"]<-base::paste(alliances7firmspart7[,"singleindentifiera"],"_",alliances7firmspart7[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 4##
alliances7firmspart8[,"singleindentifiera"]<-base::ifelse(alliances7firmspart8[,"cusip_firm2"]>alliances7firmspart8[,"cusip_firm4"],alliances7firmspart8[,"cusip_firm2"],alliances7firmspart8[,"cusip_firm4"]) ##create undirectional identifier
alliances7firmspart8[,"singleindentifierb"]<-base::ifelse(alliances7firmspart8[,"cusip_firm2"]>alliances7firmspart8[,"cusip_firm4"],alliances7firmspart8[,"cusip_firm4"],alliances7firmspart8[,"cusip_firm2"]) ##create undirectional identifier
alliances7firmspart8[,"singleindentifier"]<-base::paste(alliances7firmspart8[,"singleindentifiera"],"_",alliances7firmspart8[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 5##
alliances7firmspart9[,"singleindentifiera"]<-base::ifelse(alliances7firmspart9[,"cusip_firm2"]>alliances7firmspart9[,"cusip_firm5"],alliances7firmspart9[,"cusip_firm2"],alliances7firmspart9[,"cusip_firm5"]) ##create undirectional identifier
alliances7firmspart9[,"singleindentifierb"]<-base::ifelse(alliances7firmspart9[,"cusip_firm2"]>alliances7firmspart9[,"cusip_firm5"],alliances7firmspart9[,"cusip_firm5"],alliances7firmspart9[,"cusip_firm2"]) ##create undirectional identifier
alliances7firmspart9[,"singleindentifier"]<-base::paste(alliances7firmspart9[,"singleindentifiera"],"_",alliances7firmspart9[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 6##
alliances7firmspart10[,"singleindentifiera"]<-base::ifelse(alliances7firmspart10[,"cusip_firm2"]>alliances7firmspart10[,"cusip_firm6"],alliances7firmspart10[,"cusip_firm2"],alliances7firmspart10[,"cusip_firm6"]) ##create undirectional identifier
```

```
alliances7firmspart10[,"singleindentifierb"]<-base::ifelse(alliances7firmspart10[,"cusip_firm2"]>alliances7firmspart10[,"cusip_firm6"],alliances7firmspart10[,"cusip_firm6"],alliances7firmspart10[,"cusip_firm2"]) ##create undirectional identifier
alliances7firmspart10[,"singleindentifier"]<-base::paste(alliances7firmspart10[,"singleindentifiera"],"_",alliances7firmspart10[,"singleindentifierb"],sep="") ##create undirectional identifier
##2 and 7##
alliances7firmspart11[,"singleindentifiera"]<-base::ifelse(alliances7firmspart11[,"cusip_firm2"]>alliances7firmspart11[,"cusip_firm7"],alliances7firmspart11[,"cusip_firm2"],alliances7firmspart11[,"cusip_firm7"]) ##create undirectional identifier
alliances7firmspart11[,"singleindentifierb"]<-base::ifelse(alliances7firmspart11[,"cusip_firm2"]>alliances7firmspart11[,"cusip_firm7"],alliances7firmspart11[,"cusip_firm7"],alliances7firmspart11[,"cusip_firm2"]) ##create undirectional identifier
alliances7firmspart11[,"singleindentifier"]<-base::paste(alliances7firmspart11[,"singleindentifiera"],"_",alliances7firmspart11[,"singleindentifierb"],sep="") ##create undirectional identifier
##3 and 4##
alliances7firmspart12[,"singleindentifiera"]<-base::ifelse(alliances7firmspart12[,"cusip_firm3"]>alliances7firmspart12[,"cusip_firm4"],alliances7firmspart12[,"cusip_firm3"],alliances7firmspart12[,"cusip_firm4"]) ##create undirectional identifier
alliances7firmspart12[,"singleindentifierb"]<-base::ifelse(alliances7firmspart12[,"cusip_firm3"]>alliances7firmspart12[,"cusip_firm4"],alliances7firmspart12[,"cusip_firm4"],alliances7firmspart12[,"cusip_firm3"]) ##create undirectional identifier
alliances7firmspart12[,"singleindentifier"]<-base::paste(alliances7firmspart12[,"singleindentifiera"],"_",alliances7firmspart12[,"singleindentifierb"],sep="") ##create undirectional identifier
##3 and 5##
alliances7firmspart13[,"singleindentifiera"]<-base::ifelse(alliances7firmspart13[,"cusip_firm3"]>alliances7firmspart13[,"cusip_firm5"],alliances7firmspart13[,"cusip_firm3"],alliances7firmspart13[,"cusip_firm5"]) ##create undirectional identifier
alliances7firmspart13[,"singleindentifierb"]<-base::ifelse(alliances7firmspart13[,"cusip_firm3"]>alliances7firmspart13[,"cusip_firm5"],alliances7firmspart13[,"cusip_firm5"],alliances7firmspart13[,"cusip_firm3"]) ##create undirectional identifier
alliances7firmspart13[,"singleindentifier"]<-base::paste(alliances7firmspart13[,"singleindentifiera"],"_",alliances7firmspart13[,"singleindentifierb"],sep="") ##create undirectional identifier
##3 and 6##
alliances7firmspart14[,"singleindentifiera"]<-base::ifelse(alliances7firmspart14[,"cusip_firm3"]>alliances7firmspart14[,"cusip_firm6"],alliances7firmspart14[,"cusip_firm3"],alliances7firmspart14[,"cusip_firm6"]) ##create undirectional identifier
alliances7firmspart14[,"singleindentifierb"]<-base::ifelse(alliances7firmspart14[,"cusip_firm3"]>alliances7firmspart14[,"cusip_firm6"],alliances7firmspart14[,"cusip_firm6"],alliances7firmspart14[,"cusip_firm3"]) ##create undirectional identifier
alliances7firmspart14[,"singleindentifier"]<-base::paste(alliances7firmspart14[,"singleindentifiera"],"_",alliances7firmspart14[,"singleindentifierb"],sep="") ##create undirectional identifier
##3 and 7##
alliances7firmspart15[,"singleindentifiera"]<-base::ifelse(alliances7firmspart15[,"cusip_firm3"]>alliances7firmspart15[,"cusip_firm7"],alliances7firmspart15[,"cusip_firm3"],alliances7firmspart15[,"cusip_firm7"]) ##create undirectional identifier
alliances7firmspart15[,"singleindentifierb"]<-base::ifelse(alliances7firmspart15[,"cusip_firm3"]>alliances7firmspart15[,"cusip_firm7"],alliances7firmspart15[,"cusip_firm7"],alliances7firmspart15[,"cusip_firm3"]) ##create undirectional identifier
alliances7firmspart15[,"singleindentifier"]<-base::paste(alliances7firmspart15[,"singleindentifiera"],"_",alliances7firmspart15[,"singleindentifierb"],sep="") ##create undirectional identifier
##4 and 5##
alliances7firmspart16[,"singleindentifiera"]<-base::ifelse(alliances7firmspart16[,"cusip_firm4"]>alliances7firmspart16[,"cusip_firm5"],alliances7firmspart16[,"cusip_firm4"],alliances7firmspart16[,"cusip_firm5"]) ##create undirectional identifier
alliances7firmspart16[,"singleindentifierb"]<-base::ifelse(alliances7firmspart16[,"cusip_firm4"]>alliances7firmspart16[,"cusip_firm5"],alliances7firmspart16[,"cusip_firm5"],alliances7firmspart16[,"cusip_firm4"]) ##create undirectional identifier
alliances7firmspart16[,"singleindentifier"]<-base::paste(alliances7firmspart16[,"singleindentifiera"],"_",alliances7firmspart16[,"singleindentifierb"],sep="") ##create undirectional identifier
##4 and 6##
alliances7firmspart17[,"singleindentifiera"]<-base::ifelse(alliances7firmspart17[,"cusip_firm4"]>alliances7firmspart17[,"cusip_firm6"],alliances7firmspart17[,"cusip_firm4"],alliances7firmspart17[,"cusip_firm6"]) ##create undirectional identifier
alliances7firmspart17[,"singleindentifierb"]<-base::ifelse(alliances7firmspart17[,"cusip_firm4"]>alliances7firmspart17[,"cusip_firm6"],alliances7firmspart17[,"cusip_firm6"],alliances7firmspart17[,"cusip_firm4"]) ##create undirectional identifier
```

```r
alliances7firmspart17[,"singleindentifier"]<-base::paste(alliances7firmspart17[,"singleindentifiera"],"_",alliances7firmspart17[,"singleindentifierb"],sep="") ##create undirectional identifier
##4 and 7##
alliances7firmspart18[,"singleindentifiera"]<-base::ifelse(alliances7firmspart18[,"cusip_firm4"]>alliances7firmspart18[,"cusip_firm7"],alliances7firmspart18[,"cusip_firm4"],alliances7firmspart18[,"cusip_firm7"]) ##create undirectional identifier
alliances7firmspart18[,"singleindentifierb"]<-base::ifelse(alliances7firmspart18[,"cusip_firm4"]>alliances7firmspart18[,"cusip_firm7"],alliances7firmspart18[,"cusip_firm7"],alliances7firmspart18[,"cusip_firm4"]) ##create undirectional identifier
alliances7firmspart18[,"singleindentifier"]<-base::paste(alliances7firmspart18[,"singleindentifiera"],"_",alliances7firmspart18[,"singleindentifierb"],sep="") ##create undirectional identifier
##5 and 6##
alliances7firmspart19[,"singleindentifiera"]<-base::ifelse(alliances7firmspart19[,"cusip_firm5"]>alliances7firmspart19[,"cusip_firm6"],alliances7firmspart19[,"cusip_firm5"],alliances7firmspart19[,"cusip_firm6"]) ##create undirectional identifier
alliances7firmspart19[,"singleindentifierb"]<-base::ifelse(alliances7firmspart19[,"cusip_firm5"]>alliances7firmspart19[,"cusip_firm6"],alliances7firmspart19[,"cusip_firm6"],alliances7firmspart19[,"cusip_firm5"]) ##create undirectional identifier
alliances7firmspart19[,"singleindentifier"]<-base::paste(alliances7firmspart19[,"singleindentifiera"],"_",alliances7firmspart19[,"singleindentifierb"],sep="") ##create undirectional identifier
##5 and 7##
alliances7firmspart20[,"singleindentifiera"]<-base::ifelse(alliances7firmspart20[,"cusip_firm5"]>alliances7firmspart20[,"cusip_firm7"],alliances7firmspart20[,"cusip_firm5"],alliances7firmspart20[,"cusip_firm7"]) ##create undirectional identifier
alliances7firmspart20[,"singleindentifierb"]<-base::ifelse(alliances7firmspart20[,"cusip_firm5"]>alliances7firmspart20[,"cusip_firm7"],alliances7firmspart20[,"cusip_firm7"],alliances7firmspart20[,"cusip_firm5"]) ##create undirectional identifier
alliances7firmspart20[,"singleindentifier"]<-base::paste(alliances7firmspart20[,"singleindentifiera"],"_",alliances7firmspart20[,"singleindentifierb"],sep="") ##create undirectional identifier
##6 and 7##
alliances7firmspart21[,"singleindentifiera"]<-base::ifelse(alliances7firmspart21[,"cusip_firm6"]>alliances7firmspart21[,"cusip_firm7"],alliances7firmspart21[,"cusip_firm6"],alliances7firmspart21[,"cusip_firm7"]) ##create undirectional identifier
alliances7firmspart21[,"singleindentifierb"]<-base::ifelse(alliances7firmspart21[,"cusip_firm6"]>alliances7firmspart21[,"cusip_firm7"],alliances7firmspart21[,"cusip_firm7"],alliances7firmspart21[,"cusip_firm6"]) ##create undirectional identifier
alliances7firmspart21[,"singleindentifier"]<-base::paste(alliances7firmspart21[,"singleindentifiera"],"_",alliances7firmspart21[,"singleindentifierb"],sep="") ##create undirectional identifier

alliances7firms<-base::rbind(alliances7firmspart1,alliances7firmspart2,alliances7firmspart3,alliances7firmspart4,alliances7firmspart5,alliances7firmspart6,alliances7firmspart7,alliances7firmspart8,alliances7firmspart9,alliances7firmspart10,alliances7firmspart11,alliances7firmspart12,alliances7firmspart13,alliances7firmspart14,alliances7firmspart15,alliances7firmspart16,alliances7firmspart17,alliances7firmspart18,alliances7firmspart19,alliances7firmspart20,alliances7firmspart21)

base::rm(alliances7firmspart1)
base::rm(alliances7firmspart2)
base::rm(alliances7firmspart3)
base::rm(alliances7firmspart4)
base::rm(alliances7firmspart5)
base::rm(alliances7firmspart6)
base::rm(alliances7firmspart7)
base::rm(alliances7firmspart8)
base::rm(alliances7firmspart9)
base::rm(alliances7firmspart10)
base::rm(alliances7firmspart11)
base::rm(alliances7firmspart12)
base::rm(alliances7firmspart13)
base::rm(alliances7firmspart14)
base::rm(alliances7firmspart15)
base::rm(alliances7firmspart16)
base::rm(alliances7firmspart17)
base::rm(alliances7firmspart18)
base::rm(alliances7firmspart19)
base::rm(alliances7firmspart20)
base::rm(alliances7firmspart21)
```

```r
alliances_full<-base::rbind(alliances2firms,alliances3firms,alliances4firms,alliances5firms,alliances6firms,alliances7firms)

alliances_fullp0<-alliances_full
alliances_fullp1<-alliances_full
alliances_fullp2<-alliances_full
alliances_fullp3<-alliances_full

alliances_fullp1[,"year_announced"]<-base::as.numeric(alliances_fullp1[,"year_announced"])+1
alliances_fullp2[,"year_announced"]<-base::as.numeric(alliances_fullp1[,"year_announced"])+2
alliances_fullp3[,"year_announced"]<-base::as.numeric(alliances_fullp1[,"year_announced"])+3

alliances<-base::rbind(alliances_fullp0,alliances_fullp1,alliances_fullp2,alliances_fullp3)

base::rm(alliances_full,alliances_fullp0,alliances_fullp1,alliances_fullp2,alliances_fullp3)

alliances[,"common_alliance_sum"]<-1

alliances[,"singleindentifier_cusip"]<-alliances[,"singleindentifier"]

alliances<-stats::aggregate(alliances[c("common_alliance_sum")],alliances[c("singleindentifier_cusip","year_announced")],base::sum,na.rm=TRUE) ##aggregate data

alliances[,"common_alliance"]<-1

consistencycheck<-base::nrow(alliances) ##security break
alliances<-alliances[!base::duplicated(base::paste(alliances[,"singleindentifier_cusip"],"_",alliances[,"year_announced"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(alliances)) ##security break

alliances_common<-alliances

base::save(alliances_common,file="alliances_common_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges funda data and segments data

#merge-----------------------------------------------------------------------
base::load(file="segments_processed.Rdata") ##import data
base::load(file="funda_processed.Rdata") ##import

base::stopifnot(base::inherits(segments[,"gvkey"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(funda[,"gvkey"],"character")==TRUE) ##assure that variables have the same format

funda_segments<-base::merge(funda,segments,by.x=c("gvkey","fyear"),by.y=c("gvkey","year"),all.x=TRUE,all.y=FALSE) ##merge

consistencycheck<-base::nrow(funda_segments) ##security break
funda_segments<-funda_segments[!base::duplicated(base::paste(funda_segments[,"lpermno"],"_",funda_segments[,"fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(funda_segments)) ##security break

base::save(funda_segments,file="funda_segments_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges funda segments data and crsp header data

#merge-----------------------------------------------------------------------
base::load(file="funda_segments_processed.Rdata") ##import
base::load(file="crsp_header_processed.Rdata") ##import data

base::stopifnot(base::inherits(funda_segments[,"lpermno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(crsp_header[,"permno"],"character")==TRUE) ##assure that variables have the same format

funda_segments_crsp_header<-base::merge(funda_segments,crsp_header,by.x=c("lpermno"),by.y=c("permno"),all.x=FALSE,all.y=FALSE) ##merge

consistencycheck<-base::nrow(funda_segments_crsp_header) ##security break
funda_segments_crsp_header<-funda_segments_crsp_header[!base::duplicated(base::paste(funda_segments_crsp_header[,"lpermno"],"_",funda_segments_crsp_header[,"fyear"],sep=""))] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(funda_segments_crsp_header)) ##security break

base::save(funda_segments_crsp_header,file="funda_segments_crsp_header_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges citation flows data and total citations data

#merge-----------------------------------------------------------------------
base::load(file="citation_flows_processed.Rdata") ##import data
base::load(file="total_citations_processed.Rdata") ##import

base::stopifnot(base::inherits(citation_flows[,"citing_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(total_citations[,"citing_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(total_citations[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format

citation_flows_total_citations<-base::merge(citation_flows,total_citations,by.x=c("citing_permno","citing_fyear"),by.y=c("citing_permno","citing_fyear"),all.x=TRUE,all.y=FALSE) ##merge

base::stopifnot(base::nrow(citation_flows)==base::nrow(citation_flows_total_citations)) ##security break

total_citations[,"cited_permno"]<-total_citations[,"citing_permno"] ##rename variable
total_citations[,"citing_permno"]<-NULL ##delete old variable
total_citations[,"cited_totalcites"]<-total_citations[,"citing_totalcites"] ##rename variable
total_citations[,"citing_totalcites"]<-NULL ##delete old variable

citation_flows_total_citations<-base::merge(citation_flows_total_citations,total_citations,by.x=c("cited_permno","citing_fyear"),by.y=c("cited_permno","citing_fyear"),all.x=TRUE,all.y=FALSE) ##merge

base::stopifnot(base::nrow(citation_flows)==base::nrow(citation_flows_total_citations)) ##security break

consistencycheck<-base::nrow(citation_flows_total_citations) ##security break
citation_flows_total_citations<-citation_flows_total_citations[!base::duplicated(base::paste(citation_flows_total_citations[,"citing_permno"],"_",citation_flows_total_citations[,"cited_permno"],"_",citation_flows_total_citations[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations)) ##security break

base::save(citation_flows_total_citations,file="citation_flows_total_citations_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-------------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges citation flows total citations data and total patents data

#merge-----------------------------------------------------------------------
base::load(file="citation_flows_total_citations_processed.Rdata") ##import data
base::load(file="total_patents_processed.Rdata") ##import

base::stopifnot(base::inherits(citation_flows_total_citations[,"cited_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(total_patents[,"cited_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(total_patents[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format

citation_flows_total_citations_total_patents<-base::merge(citation_flows_total_citations,total_patents,by.x=c("cited_permno","citing_fyear"),by.y=c("cited_permno","citing_fyear"),all.x=FALSE,all.y=FALSE) ##merge

base::stopifnot(base::nrow(citation_flows_total_citations)==base::nrow(citation_flows_total_citations_total_patents)) ##security break

total_patents[,"citing_permno"]<-total_patents[,"cited_permno"] ##rename variable
total_patents[,"cited_permno"]<-NULL ##delete old variable
total_patents[,"citing_patentstocktotal"]<-total_patents[,"cited_patentstocktotal"] ##rename variable
total_patents[,"cited_patentstocktotal"]<-NULL ##delete old variable
total_patents[,"citing_patentstockunder3yperc"]<-total_patents[,"cited_patentstockunder3yperc"] ##rename variable
total_patents[,"cited_patentstockunder3yperc"]<-NULL ##delete old variable
total_patents[,"citing_patentstockbetween3and7yperc"]<-total_patents[,"cited_patentstockbetween3and7yperc"] ##rename variable
total_patents[,"cited_patentstockbetween3and7yperc"]<-NULL ##delete old variable

citation_flows_total_citations_total_patents<-base::merge(citation_flows_total_citations_total_patents,total_patents,by.x=c("citing_permno","citing_fyear"),by.y=c("citing_permno","citing_fyear"),all.x=FALSE,all.y=FALSE) ##merge

base::stopifnot(base::nrow(citation_flows_total_citations)==base::nrow(citation_flows_total_citations_total_patents)) ##security break

consistencycheck<-base::nrow(citation_flows_total_citations_total_patents) ##security break
citation_flows_total_citations_total_patents<-citation_flows_total_citations_total_patents[!base::duplicated(base::paste(citation_flows_total_citations_total_patents[,"citing_permno"],"_",citation_flows_total_citations_total_patents[,"cited_permno"],"_",citation_flows_total_citations_total_patents[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations_total_patents)) ##security break

base::save(citation_flows_total_citations_total_patents,file="citation_flows_total_citations_total_patents_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges citation flows total citations total patents data and ibes common data

#merge-----------------------------------------------------------------------
base::load(file="citation_flows_total_citations_total_patents_processed.Rdata") ##import data
base::load(file="ibes_common_processed.Rdata") ##import

base::stopifnot(base::inherits(citation_flows_total_citations_total_patents[,"cited_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents[,"citing_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(ibes_common[,"a_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(ibes_common[,"b_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(ibes_common[,"a_year"],"character")==TRUE) ##assure that variables have the same format

citation_flows_total_citations_total_patents_ibes_common<-base::merge(citation_flows_total_citations_total_patents,ibes_common,by.x=c("citing_permno","cited_permno","citing_fyear"),by.y=c("a_permno","b_permno","a_year"),all.x=TRUE,all.y=FALSE) ##merge

citation_flows_total_citations_total_patents_ibes_common<-citation_flows_total_citations_total_patents_ibes_common[citation_flows_total_citations_total_patents_ibes_common[,"citing_fyear"]>=base::min(ibes_common[,"a_year"])&citation_flows_total_citations_total_patents_ibes_common[,"citing_fyear"]<=base::max(ibes_common[,"a_year"]),] ##assure clean data

consistencycheck<-base::nrow(citation_flows_total_citations_total_patents_ibes_common) ##security break
citation_flows_total_citations_total_patents_ibes_common<-citation_flows_total_citations_total_patents_ibes_common[!base::duplicated(base::paste(citation_flows_total_citations_total_patents_ibes_common[,"citing_permno"],"_",citation_flows_total_citations_total_patents_ibes_common[,"cited_permno"],"_",citation_flows_total_citations_total_patents_ibes_common[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations_total_patents_ibes_common)) ##security break

base::save(citation_flows_total_citations_total_patents_ibes_common,file="citation_flows_total_citations_total_patents_ibes_common_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges citation flows total citations total patents ibes common data and ibes total data

#merge-----------------------------------------------------------------------
base::load(file="citation_flows_total_citations_total_patents_ibes_common_processed.Rdata") ##import data
base::load(file="ibes_total_processed.Rdata") ##import

base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common[,"cited_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(ibes_total[,"permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(ibes_total[,"year"],"character")==TRUE) ##assure that variables have the same format

ibes_total_cited<-ibes_total ##create temp data
base::colnames(ibes_total_cited)<-base::paste("cited_",base::colnames(ibes_total_cited),sep="") ##adjust prefix
ibes_total_citing<-ibes_total ##create temp data
base::colnames(ibes_total_citing)<-base::paste("citing_",base::colnames(ibes_total_citing),sep="") ##adjust prefix

citation_flows_total_citations_total_patents_ibes_common_ibes_total_cited<-base::merge(citation_flows_total_citations_total_patents_ibes_common,ibes_total_cited,by.x=c("cited_permno","citing_fyear"),by.y=c("cited_permno","cited_year"),all.x=TRUE,all.y=FALSE) ##merge
citation_flows_total_citations_total_patents_ibes_common_ibes_total_cited_citing<-base::merge(citation_flows_total_citations_total_patents_ibes_common_ibes_total_cited,ibes_total_citing,by.x=c("citing_permno","citing_fyear"),by.y=c("citing_permno","citing_year"),all.x=TRUE,all.y=FALSE) ##merge

citation_flows_total_citations_total_patents_ibes_common_ibes_total<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_cited_citing ##rename data frame

citation_flows_total_citations_total_patents_ibes_common_ibes_total<-citation_flows_total_citations_total_patents_ibes_common_ibes_total[citation_flows_total_citations_total_patents_ibes_common_ibes_total[,"citing_fyear"]>=base::min(ibes_total[,"year"])&citation_flows_total_citations_total_patents_ibes_common_ibes_total[,"citing_fyear"]<=base::max(ibes_total[,"year"]),] ##assure clean data

consistencycheck<-base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total) ##security break
citation_flows_total_citations_total_patents_ibes_common_ibes_total<-citation_flows_total_citations_total_patents_ibes_common_ibes_total[!base::duplicated(base::paste(citation_flows_total_citations_total_patents_ibes_common_ibes_total[,"citing_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total[,"cited_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total)) ##security break

base::save(citation_flows_total_citations_total_patents_ibes_common_ibes_total,file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges citation flows total citations total patents ibes common ibes total data and funda segments crsp header data

#merge-----------------------------------------------------------------------
base::load(file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_processed.Rdata") ##import data
base::load(file="funda_segments_crsp_header_processed.Rdata") ##import

base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total[,"citing_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total[,"cited_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(funda_segments_crsp_header[,"lpermno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(funda_segments_crsp_header[,"fyear"],"character")==TRUE) ##assure that variables have the same format

funda_segments_crsp_header_cited<-funda_segments_crsp_header ##create temp data
base::colnames(funda_segments_crsp_header_cited)<-base::paste("cited_",base::colnames(funda_segments_crsp_header_cited),sep="") ##add prefix
funda_segments_crsp_header_citing<-funda_segments_crsp_header ##create temp data
base::colnames(funda_segments_crsp_header_citing)<-base::paste("citing_",base::colnames(funda_segments_crsp_header_citing),sep="") ##add prefix

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_cited<-base::merge(citation_flows_total_citations_total_patents_ibes_common_ibes_total,funda_segments_crsp_header_cited,by.x=c("cited_permno","citing_fyear"),by.y=c("cited_lpermno","cited_fyear"),all.x=FALSE,all.y=FALSE) ##merge
citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_cited_funda_segments_crsp_header_citing<-base::merge(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_cited,funda_segments_crsp_header_citing,by.x=c("citing_permno","citing_fyear"),by.y=c("citing_lpermno","citing_fyear"),all.x=FALSE,all.y=FALSE) ##merge

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_cited_funda_segments_crsp_header_citing ##rename data

consistencycheck<-base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header) ##security break
citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[!base::duplicated(base::paste(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"citing_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"cited_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header)) ##security break

base::save(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header,file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction----------------------------------------------------------------------
#this file merges citation flows total citations total patents ibes common ibes total funda segments crsp header data and hpsimilarity data

#merge----------------------------------------------------------------------
base::load(file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_processed.Rdata") ##import data
base::load(file="hpsimilarity_processed.Rdata") ##import data

base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"citing_gvkey"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"cited_gvkey"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(hpsimilarity[,"gvkey1"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(hpsimilarity[,"gvkey2"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(hpsimilarity[,"year"],"character")==TRUE) ##assure that variables have the same format

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"gvkey1"]<-base::as.character(base::as.numeric(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"citing_gvkey"])) ##generate variable
citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"gvkey2"]<-base::as.character(base::as.numeric(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header[,"cited_gvkey"])) ##generate variable

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity<-base::merge(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header,hpsimilarity,by.x=c("gvkey1","gvkey2","citing_fyear"),by.y=c("gvkey1","gvkey2","year"),all.x=TRUE,all.y=FALSE) ##merge

consistencycheck<-base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity) ##security break
citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[!base::duplicated(base::paste(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[,"citing_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[,"cited_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity)) ##security break

base::save(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity,file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares technology proximity data for further processing

#prepare data-----------------------------------------------------------------------
base::load(file="patents_processed.Rdata") ##import data
base::load(file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_processed.Rdata") ##import data

filla<-base::data.frame("iyear"=base::unique(patents[,"iyear"]),"merge"=1)
fillb<-base::data.frame("permno"=base::unique(patents[,"permno"]),"merge"=1)
fill<-base::merge(filla,fillb,by.x=c("merge"),by.y=c("merge"),all.x=FALSE,all.y=FALSE) ##merge

patents<-patents[,c("permno","class","iyear")] ##reduce data
patents<-patents[patents[,"class"]!="",] ##eliminate patents without patent class

ind<-base::as.data.frame(dummies::dummy(x="class",data=patents)) ##generate dummies for patent classes
classnames<-base::colnames(ind) ##get vector with class names
patents<-base::cbind(patents,ind) ##merge data
num_patents<-stats::aggregate(patents[base::colnames(ind)],patents[c("permno","iyear")],base::sum,na.rm=TRUE) ##aggregate data

num_patents<-base::merge(num_patents,fill,by.x=c("permno","iyear"),by.y=c("permno","iyear"),all.x=FALSE,all.y=TRUE) ##merge

for(i in base::colnames(ind)){
  num_patents[,i]<-base::ifelse(base::is.na(num_patents[,i]),0,num_patents[,i]) ##fill nas with zero
}

full<-NULL ##create empty data frame

for(i in base::unique(num_patents[,"iyear"])){
  base::print(i) ##print
  temp<-num_patents[num_patents[,"iyear"]<=i,] ##reduce data

  cum_num_patents<-stats::aggregate(temp[base::colnames(ind)],temp[c("permno")],base::sum,na.rm=TRUE) ##aggregate data
  cum_num_patents[,"cumiyear"]<-i ##generate variable
  full<-base::rbind(full,cum_num_patents) ##merge
}

patents<-full ##rename data frame

for(i in base::colnames(ind)){
  num_patents[,i]<-base::ifelse(base::is.na(num_patents[,i]),0,num_patents[,i]) ##fill nas with zero
}

pairdata<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity ##rename data frame

measure<-NULL ##create empty data frame

for(i in base::sort(base::unique(pairdata[,"citing_fyear"]))){
  base::print(i) ##print

  patents_temp<-patents[patents[,"cumiyear"]==i,] ##reduce data
  pairdata_temp<-pairdata[pairdata[,"citing_fyear"]==i,] ##reduce data

  patents_a<-patents_temp ##create temp data
  patents_b<-patents_temp  ##create temp data

  base::colnames(patents_a)<-base::paste("a_",base::colnames(patents_a),sep="") ##add prefix
  base::colnames(patents_b)<-base::paste("b_",base::colnames(patents_b),sep="") ##add prefix


  patents_merge<-base::merge(patents_a,pairdata_temp,by.x=c("a_permno","a_cumiyear"),by.y=c("citing_permno","citing_fyear"),all.x=FALSE,all.y=FALSE) ##merge

```

```r
    patents_merge<-base::merge(patents_b,patents_merge,by.x=c("b_permno","b_cumiyear"),by.y=c("cited_permno","a_cumiyear"),all.x=FALSE,all.y=FALSE) ##merge

    base::stopifnot(base::nrow(patents_merge)==base::nrow(pairdata_temp)) ##security break

    for(i in classnames){
      base::print(base::paste(i,sep=""))

      patents_merge[,base::paste("sq_a_",i,sep="")]<-patents_merge[,base::paste("a_",i,sep="")]^2 ##calculate the square of the class

      patents_merge[,base::paste("sq_b_",i,sep="")]<-patents_merge[,base::paste("b_",i,sep="")]^2 ##calculate the square of the class

      patents_merge[,base::paste("ab_",i,sep="")]<-patents_merge[,base::paste("a_",i,sep="")]*patents_merge[,base::paste("b_",i,sep="")] ##calculate the product of both classes
    }

    sq_a<-base::paste("sq_a_",classnames,sep="") ##add prefix to class names
    sq_b<-base::paste("sq_b_",classnames,sep="") ##add prefix to class names
    ab<-base::paste("ab_",classnames,sep="") ##add prefix to class names

    patents_merge[,"sum_sq_a"]<-base::rowSums(patents_merge[,sq_a],na.rm=FALSE,dims=1) ##calculate the sum
    patents_merge[,"sum_sq_b"]<-base::rowSums(patents_merge[,sq_b],na.rm=FALSE,dims=1) ##calculate the sum
    patents_merge[,"sum_ab"]<-base::rowSums(patents_merge[,ab],na.rm=FALSE,dims=1) ##calculate the sum


    patents_merge[,"techn_prox"]<-patents_merge[,"sum_ab"]/base::sqrt(patents_merge[,"sum_sq_a"]*patents_merge[,"sum_sq_b"]) ##calculate the proximity measure

    patents_merge<-patents_merge[,c("b_permno","a_permno","b_cumiyear","techn_prox")] ##reduce data

    measure<-base::rbind(measure,patents_merge) ##merge
  }

  techn_prox<-measure ##rename data frame

  techn_prox[,"techn_prox"]<-base::ifelse(base::is.nan(techn_prox[,"techn_prox"]),0,techn_prox[,"techn_prox"]) ##fill NaN with zero

  consistencycheck<-base::nrow(techn_prox) ##security break
  techn_prox<-techn_prox[!base::duplicated(base::paste(techn_prox[,"a_permno"],"_",techn_prox[,"b_permno"],"_",techn_prox[,"b_cumiyear"],sep="")),] ##delete duplicates
  base::stopifnot(consistencycheck==base::nrow(techn_prox)) ##security break

  base::save(techn_prox,file="techn_prox_processed.Rdata") ##export data
  base::rm(list=base::ls()) ##clean
  base::gc() ##clean
  #end-----------------------------------------------------------------------
```

```r
#introduction----------------------------------------------------------------------
#this file merges citation flows total citations total patents ibes common ibes total funda segments crsp header hpsimilarity data and technological proximity data

#merge----------------------------------------------------------------------
base::load(file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_processed.Rdata") ##import data
base::load(file="techn_prox_processed.Rdata") ##import data

base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[,"citing_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[,"cited_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(techn_prox[,"a_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(techn_prox[,"b_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(techn_prox[,"b_cumiyear"],"character")==TRUE) ##assure that variables have the same format

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox<-base::merge(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity,techn_prox,by.x=c("citing_permno","cited_permno","citing_fyear"),by.y=c("a_permno","b_permno","b_cumiyear"),all.x=FALSE,all.y=FALSE) ##merge

consistencycheck<-base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity) ##security break
citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[!base::duplicated(base::paste(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[,"citing_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[,"cited_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity)) ##security break

base::save(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox,file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges citation flows total citations total patents ibes common ibes total funda segments crsp header hpsimilarity technological proximity data and inventor common data

#merge-----------------------------------------------------------------------
base::load(file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_processed.Rdata") ##import data
base::load(file="inventors_common_processed.Rdata") ##import data

base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox[,"citing_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox[,"cited_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(inventors_common[,"a_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(inventors_common[,"b_permno"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(inventors_common[,"a_fyear"],"character")==TRUE) ##assure that variables have the same format

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common<-base::merge(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox,inventors_common,by.x=c("citing_permno","cited_permno","citing_fyear"),by.y=c("a_permno","b_permno","a_fyear"),all.x=TRUE,all.y=FALSE) ##merge

consistencycheck<-base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common) ##security break
citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[!base::duplicated(base::paste(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"citing_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"cited_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common)) ##security break

base::save(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common,file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction----------------------------------------------------------------------
#this file merges citation flows total citations total patents ibes common ibes total funda segments crsp header hpsimilarity technological proximity inventor common data and investors common data

#merge----------------------------------------------------------------------
base::load(file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_processed.Rdata") ##import data
base::load(file="investors_common_processed.Rdata") ##import data

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"citing_gvkey_adj"]<-base::as.character(base::as.numeric(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"citing_gvkey"]))
citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"cited_gvkey_adj"]<-base::as.character(base::as.numeric(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"cited_gvkey"]))

base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"citing_gvkey_adj"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"cited_gvkey_adj"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(investors_common[,"gvkey_a"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(investors_common[,"gvkey_b"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(investors_common[,"year"],"character")==TRUE) ##assure that variables have the same format

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common<-base::merge(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common,investors_common,by.x=c("citing_gvkey_adj","cited_gvkey_adj","citing_fyear"),by.y=c("gvkey_a","gvkey_b","year"),all.x=FALSE,all.y=FALSE) ##merge

consistencycheck<-base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common) ##security break
citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common[!base::duplicated(base::paste(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common[,"citing_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common[,"cited_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common)) ##security break

base::save(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common,file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_processed.Rdata") ##export data
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file merges citation flows total citations total patents ibes common ibes total funda segments crsp header hpsimilarity technological proximity inventor common investors common data and alliances common data

#merge-----------------------------------------------------------------------
base::load(file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_processed.Rdata") ##import data
base::load(file="alliances_common_processed.Rdata") ##import data

temp<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common

temp[,"citing_cusip_adj"]<-base::gsub(".{3}$","",temp[,"citing_cusip"])
temp[,"cited_cusip_adj"]<-base::gsub(".{3}$","",temp[,"cited_cusip"])

temp[,"singleindentifiera"]<-base::ifelse(temp[,"citing_cusip_adj"]>temp[,"cited_cusip_adj"],temp[,"citing_cusip_adj"],temp[,"cited_cusip_adj"]) ##create undirectional identifier
temp[,"singleindentifierb"]<-base::ifelse(temp[,"citing_cusip_adj"]>temp[,"cited_cusip_adj"],temp[,"cited_cusip_adj"],temp[,"citing_cusip_adj"]) ##create undirectional identifier
temp[,"singleindentifier_cusip"]<-base::paste(temp[,"singleindentifiera"],"_",temp[,"singleindentifierb"],sep="") ##create undirectional identifier

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common<-temp

base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common[,"singleindentifier_cusip"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common[,"citing_fyear"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(alliances_common[,"singleindentifier_cusip"],"character")==TRUE) ##assure that variables have the same format
base::stopifnot(base::inherits(alliances_common[,"year_announced"],"character")==TRUE) ##assure that variables have the same format

citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common<-base::merge(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common,alliances_common,by.x=c("singleindentifier_cusip","citing_fyear"),by.y=c("singleindentifier_cusip","year_announced"),all.x=TRUE,all.y=FALSE) ##merge

consistencycheck<-base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common) ##security break
citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common[!base::duplicated(base::paste(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common[,"citing_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common[,"cited_permno"],"_",citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common)) ##security break

base::save(citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common,file="citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common_processed.Rdata") ##export data
```

```
base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares the data for analysis

#prepare data----------------------------------------------------------------------
base::load("citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common_processed.Rdata") ##import data

data<-citation_flows_total_citations_total_patents_ibes_common_ibes_total_funda_segments_crsp_header_hpsimilarity_techn_prox_inventors_common_investors_common_alliances_common ##rename data frame

data[,"common_analys"]<-base::ifelse(base::is.na(data[,"common_analys"]),0,data[,"common_analys"])
data[,"citing_total_analys"]<-base::ifelse(base::is.na(data[,"citing_total_analys"]),0,data[,"citing_total_analys"])
data[,"cited_total_analys"]<-base::ifelse(base::is.na(data[,"cited_total_analys"]),0,data[,"cited_total_analys"])
data[,"citing_totalcites"]<-base::ifelse(base::is.na(data[,"citing_totalcites"]),0,data[,"citing_totalcites"])
data[,"cited_totalcites"]<-base::ifelse(base::is.na(data[,"cited_totalcites"]),0,data[,"cited_totalcites"])
data[,"citing_patentstockunder3yperc"]<-base::ifelse(base::is.na(data[,"citing_patentstockunder3yperc"]),0,data[,"citing_patentstockunder3yperc"])
data[,"citing_patentstockbetween3and7yperc"]<-base::ifelse(base::is.na(data[,"citing_patentstockbetween3and7yperc"]),0,data[,"citing_patentstockbetween3and7yperc"])
data[,"common_inventors"]<-base::ifelse(base::is.na(data[,"common_inventors"]),0,data[,"common_inventors"])
data[,"common_alliance"]<-base::ifelse(base::is.na(data[,"common_alliance"]),0,data[,"common_alliance"])

dummiesacommonanalys<-c("common_analys") ##create vector of dummies
dummiesactionsone<-base::grep("common_actions",base::colnames(data),value=TRUE) ##create vector of dummies
dummiesactionstwo<-base::grep("only_a_actions",base::colnames(data),value=TRUE) ##create vector of dummies
dummiesactionsthree<-base::grep("only_b_actions",base::colnames(data),value=TRUE) ##create vector of dummies
dummiesexperience<-base::grep("a_expy",base::colnames(data),value=TRUE) ##create vector of dummies
dummiesportfolio<-base::grep("a_port",base::colnames(data),value=TRUE) ##create vector of dummies
dummiesexpertise<-base::grep("a_hiind",base::colnames(data),value=TRUE) ##create vector of dummies

for(i in c(dummiesacommonanalys,
           dummiesactionsone,
           dummiesactionstwo,
           dummiesactionsthree,
           dummiesexperience,
           dummiesportfolio,
           dummiesexpertise)){
  data[,i]<-base::ifelse(base::is.na(data[,i]),0,data[,i]) ##fill nas with zeros
}

data[,"common_investors"]<-data[,"ggl_full_attn"]/1000 ##rename variable

data[,"indicator_citing_numcit"]<-base::ifelse(data[,"citing_numcit"]>0,1,0) ##generate variable

for(i in c("citing_numcit",
           "citing_examinernumcit",
           "citing_othernumcit",
           "citing_othernumcitadjusted",
           "citing_inclassnumcit",
           "citing_outclassnumcit",
           "citing_insubclassnumcit",
           "citing_outsubclassnumcit",
           "common_analys",
           "citing_total_analys",
           "cited_total_analys")){
  data[,base::paste("log_",i,sep="")]<-base::log(1+data[,i]) ##calculate logs
}
```

```r

for(i in c("citing_at",
           "cited_at",
           "citing_totalcites",
           "cited_patentstocktotal")){
  data[,base::paste("log_",i,sep="")]<-base::log(data[,i]) ##calculate logs
}

for(i in c("log_citing_totalcites",
           "log_cited_patentstocktotal",
           "common_analys")){
  data[,base::paste(i,"_sq",sep="")]<-(data[,i])^2 ##calculate squares
}

base::stopifnot(data[,"citing_total_analys"]-data[,"common_analys"]>=0) ##security break
base::stopifnot(data[,"a_hiindo90"]+data[,"a_hiindu90"]==data[,"common_analys"]) ##security break
base::stopifnot(data[,"a_expyo090"]+data[,"a_expyu090"]==data[,"common_analys"]) ##security break
base::stopifnot(data[,"a_porto090"]+data[,"a_portu090"]==data[,"common_analys"]) ##security break
base::stopifnot(data[,"common_actionso050"]+data[,"only_a_actionso050"]+data[,"only_b_actionso050"]+data[,"common_actionsu050"]==data[,"common_analys"]) ##security break

for(i in c("citing_at",
           "cited_at",
           "cited_geosegcount",
           "cited_bussegcount")){
  data<-data[!base::is.na(data[,i]),] ##keep only complete observations
}

data<-data[data[,"citing_self_citation"]!=1,] ##reduce data
data<-data[data[,"cited_patentstocktotal"]>0,] ##reduce data
data<-data[data[,"citing_totalcites"]>0,] ##reduce data

data[,"citing_at_adj"]<-data[,"citing_at"]
data[,"cited_at_adj"]<-data[,"cited_at"]

data[,"citing_patentstocktotal_adj"]<-data[,"citing_patentstocktotal"]+1
data[,"cited_patentstocktotal_adj"]<-data[,"cited_patentstocktotal"]+1

data[,"citing_totalcites_adj"]<-data[,"citing_totalcites"]+1
data[,"cited_totalcites_adj"]<-data[,"cited_totalcites"]+1

data[,"citing_total_analys_adj"]<-data[,"citing_total_analys"]+1
data[,"cited_total_analys_adj"]<-data[,"cited_total_analys"]+1

data[,"citing_at_cited_at_ratio"]<-data[,"citing_at_adj"]/data[,"cited_at_adj"]
data[,"citing_patentstocktotal_cited_patentstocktotal_ratio"]<-data[,"citing_patentstocktotal_adj"]/data[,"cited_patentstocktotal_adj"]
data[,"citing_totalcites_cited_totalcites_ratio"]<-data[,"citing_totalcites_adj"]/data[,"cited_totalcites_adj"]
data[,"citing_total_analys_cited_total_analys_ratio"]<-data[,"citing_total_analys_adj"]/data[,"cited_total_analys_adj"]

data[,"citing_at_cited_at_ratio"]<-data[,"citing_at_cited_at_ratio"]/1000
data[,"citing_patentstocktotal_cited_patentstocktotal_ratio"]<-data[,"citing_patentstocktotal_cited_patentstocktotal_ratio"]/1000
data[,"citing_totalcites_cited_totalcites_ratio"]<-data[,"citing_totalcites_cited_totalcites_ratio"]/1000
data[,"citing_total_analys_cited_total_analys_ratio"]<-data[,"citing_total_analys_cited_total_analys_ratio"]/1000

data[,"common_analys_unwin"]<-data[,"common_analys"]
data[,"techn_prox_unwin"]<-data[,"techn_prox"]
data[,"citing_at_cited_at_ratio_unwin"]<-data[,"citing_at_cited_at_ratio"]
data[,"citing_patentstocktotal_cited_patentstocktotal_ratio_unwin"]<-data[,"citing_patentstocktotal_cited_patentstocktotal_ratio"]
data[,"citing_totalcites_cited_totalcites_ratio_unwin"]<-data[,"citing_totalcites_cited_totalcites_ratio"]
data[,"citing_total_analys_cited_total_analys_ratio_unwin"]<-data[,"citing_total_analys_cited_total_analys_ratio"]
```

```r
data[,"common_inventors_unwin"]<-data[,"common_inventors"]
data[,"common_investors_unwin"]<-data[,"common_investors"]

data[,"hp_score_unwin"]<-data[,"hp_score"]

data[,"citing_total_analys_unwin"]<-data[,"citing_total_analys"]
data[,"cited_total_analys_unwin"]<-data[,"cited_total_analys"]
data[,"log_citing_at_unwin"]<-data[,"log_citing_at"]
data[,"log_cited_at_unwin"]<-data[,"log_cited_at"]
data[,"log_citing_totalcites_unwin"]<-data[,"log_citing_totalcites"]
data[,"log_citing_totalcites_sq_unwin"]<-data[,"log_citing_totalcites_sq"]
data[,"log_cited_patentstocktotal_unwin"]<-data[,"log_cited_patentstocktotal"]
data[,"log_cited_patentstocktotal_sq_unwin"]<-data[,"log_cited_patentstocktotal_sq"]
data[,"cited_patentstockunder3yperc_unwin"]<-data[,"cited_patentstockunder3yperc"]
data[,"cited_patentstockbetween3and7yperc_unwin"]<-data[,"cited_patentstockbetween3and7yperc"]

data[,"techn_prox"]<-DescTools::Winsorize(data[,"techn_prox"],probs=c(0.01,0.99))
data[,"citing_at_cited_at_ratio"]<-DescTools::Winsorize(data[,"citing_at_cited_at_ratio"],probs=c(0.01,0.99))
data[,"citing_patentstocktotal_cited_patentstocktotal_ratio"]<-DescTools::Winsorize(data[,"citing_patentstocktotal_cited_patentstocktotal_ratio"],probs=c(0.01,0.99))
data[,"citing_totalcites_cited_totalcites_ratio"]<-DescTools::Winsorize(data[,"citing_totalcites_cited_totalcites_ratio"],probs=c(0.01,0.99))
data[,"citing_total_analys_cited_total_analys_ratio"]<-DescTools::Winsorize(data[,"citing_total_analys_cited_total_analys_ratio"],probs=c(0.01,0.99))
data[,"common_inventors"]<-DescTools::Winsorize(data[,"common_inventors"],probs=c(0.01,0.99))
data[,"common_investors"]<-DescTools::Winsorize(data[,"common_investors"],probs=c(0.01,0.99))

data[,"hp_score"]<-DescTools::Winsorize(data[,"hp_score"],probs=c(0.01,0.99),na.rm=TRUE)

data[,"citing_total_analys"]<-DescTools::Winsorize(data[,"citing_total_analys"],probs=c(0.01,0.99))
data[,"cited_total_analys"]<-DescTools::Winsorize(data[,"cited_total_analys"],probs=c(0.01,0.99))
data[,"log_citing_at"]<-DescTools::Winsorize(data[,"log_citing_at"],probs=c(0.01,0.99))
data[,"log_cited_at"]<-DescTools::Winsorize(data[,"log_cited_at"],probs=c(0.01,0.99))
data[,"log_citing_totalcites"]<-DescTools::Winsorize(data[,"log_citing_totalcites"],probs=c(0.01,0.99))
data[,"log_citing_totalcites_sq"]<-DescTools::Winsorize(data[,"log_citing_totalcites_sq"],probs=c(0.01,0.99))
data[,"log_cited_patentstocktotal"]<-DescTools::Winsorize(data[,"log_cited_patentstocktotal"],probs=c(0.01,0.99))
data[,"log_cited_patentstocktotal_sq"]<-DescTools::Winsorize(data[,"log_cited_patentstocktotal_sq"],probs=c(0.01,0.99))
data[,"cited_patentstockunder3yperc"]<-DescTools::Winsorize(data[,"cited_patentstockunder3yperc"],probs=c(0.01,0.99))
data[,"cited_patentstockbetween3and7yperc"]<-DescTools::Winsorize(data[,"cited_patentstockbetween3and7yperc"],probs=c(0.01,0.99))

data[,"cited_sic2"]<-base::gsub(".{2}$","",data[,"cited_hsiccd"]) ##generate variable
data[,"citing_sic2"]<-base::gsub(".{2}$","",data[,"citing_hsiccd"]) ##generate variable
data[,"same_sic2"]<-base::ifelse(data[,"cited_sic2"]==data[,"citing_sic2"],1,0) ##generate variable
data[,"common_analys_same_sic2"]<-data[,"common_analys"]*data[,"same_sic2"] ##generate variable

data[,"cited_geosegcountquant090"]<-stats::quantile(data[,"cited_geosegcount"],probs=0.9) ##generate variable
data[,"cited_bussegcountquant090"]<-stats::quantile(data[,"cited_bussegcount"],probs=0.9) ##generate variable

data[,"cited_geosego090"]<-base::ifelse(data[,"cited_geosegcount"]>data[,"cited_geosegcountquant090"],1,0) ##generate variable
data[,"cited_bussego090"]<-base::ifelse(data[,"cited_bussegcount"]>data[,"cited_bussegcountquant090"],1,0) ##generate variable

data[,"common_analys_cited_geosego090"]<-data[,"common_analys"]*data[,"cited_geosego09
```

```
0"] ##generate variable
data[,"common_analys_cited_bussego090"]<-data[,"common_analys"]*data[,"cited_bussego090"] ##generate variable

data[,"only_a_actionso050_or_only_b_actionso050"]<-data[,"only_a_actionso050"]+data[,"only_b_actionso050"] ##generate variable

data[,"singleindentifiera"]<-base::ifelse(data[,"citing_permno"]>data[,"cited_permno"],data[,"citing_permno"],data[,"cited_permno"]) ##create undirectional identifier
data[,"singleindentifierb"]<-base::ifelse(data[,"citing_permno"]>data[,"cited_permno"],data[,"cited_permno"],data[,"citing_permno"]) ##create undirectional identifier
data[,"singleindentifier"]<-base::paste(data[,"singleindentifiera"],"_",data[,"singleindentifierb"],sep="") ##create undirectional identifier

data[,"citing_permno_citing_fyear"]<-base::paste(data[,"citing_permno"],"_",data[,"citing_fyear"],sep="") ##generate interaction variable
data[,"cited_permno_citing_fyear"]<-base::paste(data[,"cited_permno"],"_",data[,"citing_fyear"],sep="") ##generate interaction variable

consistencycheck<-base::nrow(data) ##security break
data<-data[!base::duplicated(base::paste(data[,"citing_permno"],"_",data[,"cited_permno"],"_",data[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(data)) ##security break

base::save(data,file="data_processed.Rdata") ##export data
utils::write.table(data,file="data_csv_processed.csv",sep=",",row.names=FALSE) ##export data

base::rm(list=base::ls()) ##clean
base::gc() ##clean
#end-----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares the closure and merger data for analysis

#prepare data-----------------------------------------------------------------------
for(i in c("closures","mergers")){

  base::print(i)

  ##this part makes sure that the analysts and firms are actually treated
  base::load(file="ibes_ibes_crsp_link_processed.Rdata") ##import data

  if(i=="closures"){
    base::load(file="broker_closure_processed.Rdata") ##import data
    events<-broker_closure
    base::rm(broker_closure)
  }

  if(i=="mergers"){
    base::load("broker_merger_processed.Rdata") ##import data
    events<-broker_merger
    base::rm(broker_merger)
  }

  temp_ibes<-ibes_ibes_crsp_link

  temp_events<-events


  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==1886,54,temp_ibes[,"estimator"]) ##adjust estimator for further analysis

  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==345,54,temp_ibes[,"estimator"]) ##adjust estimator for further analysis

  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==909,54,temp_ibes[,"estimator"]) ##adjust estimator for further analysis


  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==748,279,temp_ibes[,"estimator"]) ##adjust estimator for further analysis


  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==2265,112,temp_ibes[,"estimator"]) ##adjust estimator for further analysis

  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==1863,112,temp_ibes[,"estimator"]) ##adjust estimator for further analysis

  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==1361,112,temp_ibes[,"estimator"]) ##adjust estimator for further analysis

  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==1508,112,temp_ibes[,"estimator"]) ##adjust estimator for further analysis

  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==2091,112,temp_ibes[,"estimator"]) ##adjust estimator for further analysis


  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==283,211,temp_ibes[,"estimator"]) ##adjust estimator for further analysis


  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==265,61,temp_ibes[,"estimator"]) ##adjust estimator for further analysis


  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==949,881,temp_ibes[,"estimator"]) ##adjust estimator for further analysis


  temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==282,147,temp_ibes[,"estimator"]) ##adjust estimator for further analysis

```

```r
temp_ibes[,"estimator"]<-base::ifelse(temp_ibes[,"estimator"]==2038,1425,temp_ibes[,"estimator"]) ##adjust estimator for further analysis


shock<-base::merge(temp_ibes,temp_events,by.x=c("estimator"),by.y=c("estimator"),all.x=FALSE,all.y=FALSE) ##merge

shock[,"datenorm"]<-base::as.numeric(base::gsub(".{6}$","",shock[,"datenorm"]))
shock[,"diff"]<-base::as.numeric(shock[,"year"])-shock[,"datenorm"]

aftereventcoveringa<-dplyr::filter(shock,diff==1)
aftereventcoveringb<-dplyr::filter(shock,diff==2)

aftereventcovering<-base::rbind(aftereventcoveringa,aftereventcoveringb) ##merge

aftereventcovering<-aftereventcovering[,c("analys","permno","estimator","datenorm")]

aftereventcovering<-aftereventcovering[!base::duplicated(base::paste(aftereventcovering[,"analys"],"_",aftereventcovering[,"permno"],sep="")),] ##delete duplicates
aftereventcovering[,"exclude"]<-1

duringeventcoveringa<-dplyr::filter(shock,diff==0)
duringeventcoveringb<-dplyr::filter(shock,diff==-1)
duringeventcoveringc<-dplyr::filter(shock,diff==-2)

duringeventcoveringd<-base::merge(duringeventcoveringa,duringeventcoveringb,by=c("analys","permno","estimator","datenorm"),all=FALSE) ##merge


duringeventcovering<-base::merge(duringeventcoveringc,duringeventcoveringd,by=c("analys","permno","estimator","datenorm"),all=FALSE) ##merge


duringeventcovering<-duringeventcovering[,c("analys","permno","estimator","datenorm")]

duringeventcovering<-duringeventcovering[!base::duplicated(base::paste(duringeventcovering[,"analys"],"_",duringeventcovering[,"permno"],sep="")),] ##delete duplicates
duringeventcovering[,"include"]<-1

treated_analysts_firms<-base::merge(aftereventcovering,duringeventcovering,all=TRUE)


treated_analysts_firms[,"exclude"]<-base::ifelse(base::is.na(treated_analysts_firms[,"exclude"]),0,treated_analysts_firms[,"exclude"])

treated_analysts_firms[,"include"]<-base::ifelse(base::is.na(treated_analysts_firms[,"include"]),0,treated_analysts_firms[,"include"])

treated_analysts_firms<-dplyr::filter(treated_analysts_firms,exclude==0)
treated_analysts_firms<-dplyr::filter(treated_analysts_firms,include==1)


shock<-base::merge(temp_ibes,treated_analysts_firms,by=c("analys","permno","estimator"),all=FALSE) ##merge

shock[,"eventyear"]<-base::as.numeric(base::gsub(".{6}$","",shock[,"datenorm"]))

ibes_a<-shock ##create temp data
ibes_b<-shock ##create temp data

base::colnames(ibes_a)<-base::paste("a_",base::colnames(ibes_a),sep="") ##add prefix
base::colnames(ibes_b)<-base::paste("b_",base::colnames(ibes_b),sep="") ##add prefix


ibes_common<-base::merge(ibes_a,ibes_b,by.x=c("a_analys","a_year"),by.y=c("b_analys","b_year"),all.x=FALSE,all.y=FALSE) ##merge

ibes_common<-ibes_common[ibes_common[,"a_permno"]!=ibes_common[,"b_permno"],] ##reduce data

ibes_common<-ibes_common[ibes_common[,"a_estimator"]==ibes_common[,"b_estimator"],]
```

```
    ##reduce data

    ibes_common[,"common_coverage_drop"]<-1


    shock_common<-ibes_common[!base::duplicated(base::paste(ibes_common[,"a_analys"],"_",ibes_common[,"a_permno"],"_",ibes_common[,"b_permno"],"_",ibes_common[,"a_estimator"],"_",ibes_common[,"a_eventyear"],sep="")),] ##delete duplicates


    shock_common<-stats::aggregate(shock_common[c("common_coverage_drop")],shock_common[c("a_permno","b_permno","a_estimator","a_eventyear")],base::sum,na.rm=TRUE) ##aggregate data


    shock_common[,"common_shock_treatmentindicator"]<-base::ifelse(shock_common[,"common_coverage_drop"]>0,1,0)
    shock_common[,"common_coverage_drop"]<-NULL

    ##this part calculates the number of affected analysts

    shock_common_number_of_affected_analysts<-ibes_common[!base::duplicated(base::paste(ibes_common[,"a_analys"],"_",ibes_common[,"a_permno"],"_",ibes_common[,"b_permno"],"_",ibes_common[,"a_estimator"],"_",ibes_common[,"a_year"],sep="")),] ##delete duplicates

    shock_common_number_of_affected_analysts<-stats::aggregate(shock_common_number_of_affected_analysts[c("common_coverage_drop")],shock_common_number_of_affected_analysts[c("a_permno","b_permno","a_estimator","a_year")],base::sum,na.rm=TRUE) ##aggregate data

    shock_common_number_of_affected_analysts<-dplyr::rename(shock_common_number_of_affected_analysts,common_num_analsts_individual_broker_affected=common_coverage_drop)

    ##this part merges the data

    shock_common_treatment<-shock_common[,c("a_permno","b_permno","a_estimator","a_eventyear","common_shock_treatmentindicator")] ##reduce data

    if(i=="closures"){

      closure_full_shock_common_number_of_affected_analysts<-shock_common_number_of_affected_analysts
      closure_full_treatment<-shock_common_treatment
    }

    if(i=="mergers"){

      merger_full_shock_common_number_of_affected_analysts<-shock_common_number_of_affected_analysts
      merger_full_treatment<-shock_common_treatment
    }
  }

  base::save(closure_full_shock_common_number_of_affected_analysts,file="closure_full_shock_common_number_of_affected_analysts.Rdata") ##export
  base::save(closure_full_treatment,file="closure_full_treatment.Rdata") ##export
  base::save(merger_full_shock_common_number_of_affected_analysts,file="merger_full_shock_common_number_of_affected_analysts.Rdata") ##export
  base::save(merger_full_treatment,file="merger_full_treatment.Rdata") ##export

  base::rm(list=base::ls()) ##clean
  base::gc() ##clean

  base::load(file="closure_full_shock_common_number_of_affected_analysts.Rdata") ##export
  base::load(file="closure_full_treatment.Rdata") ##export
  base::load(file="merger_full_shock_common_number_of_affected_analysts.Rdata") ##export
  base::load(file="merger_full_treatment.Rdata") ##export

  for(i in c("closures","mergers","closures_and_mergers")){

    base::print(i)
```

```
    if(i=="closures"){
      full_shock_common_number_of_affected_analysts<-closure_full_shock_common_number_of_affected_analysts
      full_treatment<-closure_full_treatment
    }

    if(i=="mergers"){
      full_shock_common_number_of_affected_analysts<-merger_full_shock_common_number_of_affected_analysts
      full_treatment<-merger_full_treatment
    }

    if(i=="closures_and_mergers"){
      full_shock_common_number_of_affected_analysts<-base::rbind(closure_full_shock_common_number_of_affected_analysts,merger_full_shock_common_number_of_affected_analysts)
      full_treatment<-base::rbind(closure_full_treatment,merger_full_treatment)
    }

    full_treatment_descriptives<-full_treatment ##create copy of data

    ##this part identifies firms that are treated several times in order to avoid overlapping treatment windows

    full_treatment<-stats::aggregate(full_treatment[c("common_shock_treatmentindicator")],full_treatment[c("a_permno","b_permno","a_eventyear")],base::sum,na.rm=TRUE) ##aggregate data

    full_treatment[,"a_permno_b_permno"]<-base::paste(full_treatment[,"a_permno"],"_",full_treatment[,"b_permno"],sep="") ##generate variable

    full_treatment<-full_treatment%>%group_by(a_permno_b_permno)

    full_treatment<-full_treatment%>%dplyr::mutate(a_eventyear_lag1=dplyr::lag(a_eventyear,1))

    full_treatment<-full_treatment%>%dplyr::mutate(a_eventyear_lead1=dplyr::lead(a_eventyear,1))
    full_treatment<-base::as.data.frame(full_treatment)


    full_treatment[,"diff_actual_lag"]<-full_treatment[,"a_eventyear"]-full_treatment[,"a_eventyear_lag1"]

    full_treatment[,"diff_lead_actual"]<-full_treatment[,"a_eventyear_lead1"]-full_treatment[,"a_eventyear"]


    full_treatment[,"lag_sufficient"]<-base::ifelse(base::is.na(full_treatment[,"diff_actual_lag"])|full_treatment[,"diff_actual_lag"]>=5,1,0)

    full_treatment[,"lead_sufficient"]<-base::ifelse(base::is.na(full_treatment[,"diff_lead_actual"])|full_treatment[,"diff_lead_actual"]>=5,1,0)

    exclude<-dplyr::filter(full_treatment,lag_sufficient==0|lead_sufficient==0)

    exclude[,"excluding_period_start"]<-base::ifelse(exclude[,"lag_sufficient"]==0,exclude[,"a_eventyear_lag1"]-2,exclude[,"a_eventyear"]-2)

    exclude[,"excluding_period_end"]<-base::ifelse(exclude[,"lead_sufficient"]==0,exclude[,"a_eventyear_lead1"]+2,exclude[,"a_eventyear"]+2)

    exclude<-exclude%>%group_by(a_permno_b_permno)

    exclude<-exclude%>%dplyr::mutate(excluding_period_start_min=base::min(excluding_period_start))

    exclude<-exclude%>%dplyr::mutate(excluding_period_end_max=base::max(excluding_period_end))
```

```r
exclude<-base::as.data.frame(exclude)


exclude<-exclude[!base::duplicated(base::paste(exclude[,"a_permno"],"_",exclude[,"b_permno"],sep="")),] ##delete duplicates

exclude<-exclude[,c("a_permno","b_permno","excluding_period_start_min","excluding_period_end_max")]

full_treatment<-dplyr::filter(full_treatment,lag_sufficient==1&lead_sufficient==1)

full_treatment[,"common_shock_treatmentindicator"]<-1

##this part prepares descriptives data

full_treatment_descriptives<-stats::aggregate(full_treatment_descriptives[c("common_shock_treatmentindicator")],full_treatment_descriptives[c("a_permno","b_permno","a_estimator","a_eventyear")],base::sum,na.rm=TRUE) ##aggregate data


full_treatment_descriptives[,"a_permno_b_permno"]<-base::paste(full_treatment_descriptives[,"a_permno"],"_",full_treatment_descriptives[,"b_permno"],sep="") ##generate variable


full_treatment_descriptives<-full_treatment_descriptives%>%group_by(a_permno_b_permno)

full_treatment_descriptives<-full_treatment_descriptives%>%dplyr::mutate(a_eventyear_lag1=dplyr::lag(a_eventyear,1))

full_treatment_descriptives<-full_treatment_descriptives%>%dplyr::mutate(a_eventyear_lead1=dplyr::lead(a_eventyear,1))
full_treatment_descriptives<-base::as.data.frame(full_treatment_descriptives)


full_treatment_descriptives[,"diff_actual_lag"]<-full_treatment_descriptives[,"a_eventyear"]-full_treatment_descriptives[,"a_eventyear_lag1"]

full_treatment_descriptives[,"diff_lead_actual"]<-full_treatment_descriptives[,"a_eventyear_lead1"]-full_treatment_descriptives[,"a_eventyear"]


full_treatment_descriptives[,"lag_sufficient"]<-base::ifelse(base::is.na(full_treatment_descriptives[,"diff_actual_lag"])|full_treatment_descriptives[,"diff_actual_lag"]>=5|full_treatment_descriptives[,"diff_actual_lag"]==0,1,0)

full_treatment_descriptives[,"lead_sufficient"]<-base::ifelse(base::is.na(full_treatment_descriptives[,"diff_lead_actual"])|full_treatment_descriptives[,"diff_lead_actual"]>=5|full_treatment_descriptives[,"diff_lead_actual"]==0,1,0)


full_treatment_descriptives<-dplyr::filter(full_treatment_descriptives,lag_sufficient==1&lead_sufficient==1)

base::load(file="data_processed.Rdata") ##import data


full_treatment_descriptives<-base::merge(data,full_treatment_descriptives,by.x=c("citing_permno","cited_permno","citing_fyear"),by.y=c("a_permno","b_permno","a_eventyear"),all.x=FALSE,all.y=FALSE) ##merge

full_treatment_descriptives[,"obs"]<-1 ##generate variable


full_treatment_descriptives_broker_level<-stats::aggregate(full_treatment_descriptives[c("obs")],full_treatment_descriptives[c("a_estimator")],base::sum,na.rm=TRUE) ##aggregate data


full_treatment_descriptives_broker_level_sum<-base::nrow(full_treatment_descriptives)

```

```
full_treatment_total_temp<-base::merge(data,full_treatment,by.x=c("citing_permno","cited_permno","citing_fyear"),by.y=c("a_permno","b_permno","a_eventyear"),all.x=FALSE,all.y=FALSE) ##merge

full_treatment_descriptives_total_level_sum<-base::nrow(full_treatment_total_temp)

if(i=="closures"){

  full_treatment_descriptives_broker_level_closures<-full_treatment_descriptives_broker_level

  full_treatment_descriptives_broker_level_sum_closures<-full_treatment_descriptives_broker_level_sum

  full_treatment_descriptives_total_level_sum_closures<-full_treatment_descriptives_total_level_sum

  base::save(full_treatment_descriptives_broker_level_closures,file="full_treatment_descriptives_broker_level_closures_processed.Rdata") ##export data

  base::save(full_treatment_descriptives_broker_level_sum_closures,file="full_treatment_descriptives_broker_level_sum_closures.Rdata") ##export data

  base::save(full_treatment_descriptives_total_level_sum_closures,file="full_treatment_descriptives_total_level_sum_closures_processed.Rdata") ##export data
}

if(i=="mergers"){

  full_treatment_descriptives_broker_level_mergers<-full_treatment_descriptives_broker_level

  full_treatment_descriptives_broker_level_sum_mergers<-full_treatment_descriptives_broker_level_sum

  full_treatment_descriptives_total_level_sum_mergers<-full_treatment_descriptives_total_level_sum

  base::save(full_treatment_descriptives_broker_level_mergers,file="full_treatment_descriptives_broker_level_mergers_processed.Rdata") ##export data

  base::save(full_treatment_descriptives_broker_level_sum_mergers,file="full_treatment_descriptives_broker_level_sum_mergers.Rdata") ##export data

  base::save(full_treatment_descriptives_total_level_sum_mergers,file="full_treatment_descriptives_total_level_sum_mergers_processed.Rdata") ##export data
}

if(i=="closures_and_mergers"){

  full_treatment_descriptives_broker_level_closures_and_mergers<-full_treatment_descriptives_broker_level

  full_treatment_descriptives_broker_level_sum_closures_and_mergers<-full_treatment_descriptives_broker_level_sum

  full_treatment_descriptives_total_level_sum_closures_and_mergers<-full_treatment_descriptives_total_level_sum

  base::save(full_treatment_descriptives_broker_level_closures_and_mergers,file="full_treatment_descriptives_broker_level_closures_and_mergers_processed.Rdata") ##export data

  base::save(full_treatment_descriptives_broker_level_sum_closures_and_mergers,file="full_treatment_descriptives_broker_level_sum_closures_and_mergers.Rdata") ##export data

  base::save(full_treatment_descriptives_total_level_sum_closures_and_mergers,file="full_treatment_descriptives_total_level_sum_closures_and_mergers_processed.Rdata") ##export data
}
```

```r
##this part merges the data
base::load(file="data_processed.Rdata") ##import data


shock_common_full<-base::merge(data,full_treatment,by.x=c("citing_permno","cited_permno"),by.y=c("a_permno","b_permno"),all.x=FALSE,all.y=FALSE) ##merge

shock_common_full[,"a_eventyearp2"]<-shock_common_full[,"a_eventyear"]+2
shock_common_full[,"a_eventyearm2"]<-shock_common_full[,"a_eventyear"]-2

shock_common_full<-dplyr::filter(shock_common_full,citing_fyear>=a_eventyearm2) ##reduce data
shock_common_full<-dplyr::filter(shock_common_full,citing_fyear<=a_eventyearp2) ##reduce data

##this part creates the treatment variable

shock_common_full[,"common_shock_treatmentindicator"]<-base::ifelse(base::is.na(shock_common_full[,"common_shock_treatmentindicator"]),0,shock_common_full[,"common_shock_treatmentindicator"])

shock_common_full[,"common_shock_postindicator"]<-base::ifelse(shock_common_full[,"citing_fyear"]>shock_common_full[,"a_eventyear"],1,0)

shock_common_full[,"treatment"]<-shock_common_full[,"common_shock_treatmentindicator"]*shock_common_full[,"common_shock_postindicator"]

consistencycheck<-base::nrow(shock_common_full) ##security break

shock_common_full<-shock_common_full[!base::duplicated(base::paste(shock_common_full[,"citing_permno"],"_",shock_common_full[,"cited_permno"],"_",shock_common_full[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(shock_common_full)) ##security break


shock_common_full[,"diffyear"]<-base::as.numeric(shock_common_full[,"citing_fyear"])-base::as.numeric(shock_common_full[,"a_eventyear"]) ##generate variable


shock_common_full[,"diffyearm1"]<-base::ifelse(shock_common_full[,"diffyear"]==-1,1,0) ##generate variable

shock_common_full[,"diffyearm2"]<-base::ifelse(shock_common_full[,"diffyear"]==-2,1,0) ##generate variable

shock_common_full[,"diffyearp1"]<-base::ifelse(shock_common_full[,"diffyear"]==1,1,0) ##generate variable

shock_common_full[,"diffyearp2"]<-base::ifelse(shock_common_full[,"diffyear"]==2,1,0) ##generate variable

shock_common_full[,"diffyear0"]<-base::ifelse(shock_common_full[,"diffyear"]==0,1,0) ##generate variable


shock_common_full[,"treatmentm1"]<-shock_common_full[,"diffyearm1"]*shock_common_full[,"common_shock_treatmentindicator"] ##generate variable

shock_common_full[,"treatmentm2"]<-shock_common_full[,"diffyearm2"]*shock_common_full[,"common_shock_treatmentindicator"] ##generate variable

shock_common_full[,"treatmentp1"]<-shock_common_full[,"diffyearp1"]*shock_common_full[,"common_shock_treatmentindicator"] ##generate variable

shock_common_full[,"treatmentp2"]<-shock_common_full[,"diffyearp2"]*shock_common_full[,"common_shock_treatmentindicator"] ##generate variable

shock_common_full[,"treatment0"]<-shock_common_full[,"diffyear0"]*shock_common_full[,"common_shock_treatmentindicator"] ##generate variable

#this part creates the analys less shock variable

full_shock_common_number_of_affected_analysts<-stats::aggregate(full_shock_common_nu
```

```
mber_of_affected_analysts[c("common_num_analsts_individual_broker_affected")],full_shock_common_number_of_affected_analysts[c("a_permno","b_permno","a_year")],base::sum,na.rm=TRUE) ##aggregate data


shock_common_full<-base::merge(shock_common_full,full_shock_common_number_of_affected_analysts,by.x=c("citing_permno","cited_permno","citing_fyear"),by.y=c("a_permno","b_permno","a_year"),all.x=TRUE,all.y=FALSE) ##merge

shock_common_full[,"common_num_analsts_individual_broker_affected"]<-base::ifelse(base::is.na(shock_common_full[,"common_num_analsts_individual_broker_affected"]),0,shock_common_full[,"common_num_analsts_individual_broker_affected"])

shock_common_full[,"common_analys_less_shock"]<-shock_common_full[,"common_analys"]-shock_common_full[,"common_num_analsts_individual_broker_affected"]

base::stopifnot(base::min(shock_common_full[,"common_analys_less_shock"])==0)

citing_fyear_vec<-base::unique(shock_common_full[,"citing_fyear"]) ##get a vector of years to reduce the full sample to the relevant years

#this part reduces and exports the data
variables<-c("citing_permno",
             "cited_permno",
             "citing_fyear",
             "treatment",
             "treatmentm2",
             "treatmentm1",
             "treatmentp1",
             "treatmentp2",
             "treatment0",
             "common_analys_less_shock",
             "common_shock_treatmentindicator",
             "common_shock_postindicator")

shock_common_full<-shock_common_full[,variables] ##reduce data

data<-dplyr::filter(data,citing_fyear%in%citing_fyear_vec) ##reduce the full sample to the relevant years


shock_common_full<-base::merge(data,shock_common_full,by.x=c("citing_permno","cited_permno","citing_fyear"),by.y=c("citing_permno","cited_permno","citing_fyear"),all.x=TRUE,all.y=FALSE) ##merge

#this part makes sure to exclude observations with overlapping treatment windows

shock_common_full<-base::merge(shock_common_full,exclude,by.x=c("citing_permno","cited_permno"),by.y=c("a_permno","b_permno"),all.x=TRUE,all.y=FALSE) ##merge


shock_common_full[,"excluding_period_start_min"]<-base::ifelse(base::is.na(shock_common_full[,"excluding_period_start_min"]),2100,shock_common_full[,"excluding_period_start_min"])

shock_common_full[,"excluding_period_end_max"]<-base::ifelse(base::is.na(shock_common_full[,"excluding_period_end_max"]),1900,shock_common_full[,"excluding_period_end_max"])


shock_common_full<-dplyr::filter(shock_common_full,!citing_fyear>excluding_period_start_min)

shock_common_full<-dplyr::filter(shock_common_full,!citing_fyear<excluding_period_end_max)

#this part fills the nas with zeros

shock_common_full[,"treatment"]<-base::ifelse(base::is.na(shock_common_full[,"treatment"]),0,shock_common_full[,"treatment"])

shock_common_full[,"treatmentm2"]<-base::ifelse(base::is.na(shock_common_full[,"treatmentm2"]),0,shock_common_full[,"treatmentm2"])
```

```
shock_common_full[,"treatmentm1"]<-base::ifelse(base::is.na(shock_common_full[,"treatmentm1"]),0,shock_common_full[,"treatmentm1"])
shock_common_full[,"treatmentp1"]<-base::ifelse(base::is.na(shock_common_full[,"treatmentp1"]),0,shock_common_full[,"treatmentp1"])
shock_common_full[,"treatmentp2"]<-base::ifelse(base::is.na(shock_common_full[,"treatmentp2"]),0,shock_common_full[,"treatmentp2"])
shock_common_full[,"treatment0"]<-base::ifelse(base::is.na(shock_common_full[,"treatment0"]),0,shock_common_full[,"treatment0"])
shock_common_full[,"common_analys_less_shock"]<-base::ifelse(base::is.na(shock_common_full[,"common_analys_less_shock"]),shock_common_full[,"common_analys"],shock_common_full[,"common_analys_less_shock"])
#this part eliminates the observations of treatment pairs between the treatments because they bias the results
citing_cited_pair_vec<-dplyr::filter(shock_common_full,common_shock_treatmentindicator==1)
citing_cited_pair_vec<-base::unique(citing_cited_pair_vec[,"citing_cited_pair"])


intermediate_period_treatment_pairs<-dplyr::filter(shock_common_full,citing_cited_pair%in%citing_cited_pair_vec&treatment0==0&treatmentm2==0&treatmentm1==0&treatmentp1==0&treatmentp2==0)
intermediate_period_treatment_pairs[,"intermediate_period_exclude"]<-1
intermediate_period_treatment_pairs<-intermediate_period_treatment_pairs[,c("citing_cited_pair","citing_fyear","intermediate_period_exclude")]


shock_common_full<-base::merge(shock_common_full,intermediate_period_treatment_pairs,by.x=c("citing_cited_pair","citing_fyear"),by.y=c("citing_cited_pair","citing_fyear"),all.x=TRUE,all.y=FALSE)
shock_common_full[,"intermediate_period_exclude"]<-base::ifelse(base::is.na(shock_common_full[,"intermediate_period_exclude"]),0,shock_common_full[,"intermediate_period_exclude"])
shock_common_full<-dplyr::filter(shock_common_full,intermediate_period_exclude==0)

data_shock<-shock_common_full

consistencycheck<-base::nrow(data_shock) ##security break
data_shock<-data_shock[!base::duplicated(base::paste(data_shock[,"citing_permno"],"_",data_shock[,"cited_permno"],"_",data_shock[,"citing_fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(data_shock)) ##security break

if(i=="closures"){
  data_closures<-data_shock
  base::save(data_closures,file="data_closures_processed.Rdata") ##export data
  utils::write.table(data_closures,file="data_closures_processed.csv",sep=",",row.names=FALSE) ##export data
  treatmentobspreandpost<-dplyr::filter(data_shock,common_shock_treatmentindicator==1)
  treatmentobspreandpost<-base::nrow(treatmentobspreandpost)
  base::save(treatmentobspreandpost,file="treatmentobspreandpost_closures_processed.Rdata") ##export data
}

if(i=="mergers"){
  data_mergers<-data_shock
  base::save(data_mergers,file="data_mergers_processed.Rdata") ##export data
  utils::write.table(data_mergers,file="data_mergers_processed.csv",sep=",",row.names=FALSE) ##export data
```

```
      treatmentobspreandpost<-dplyr::filter(data_shock,common_shock_treatmentindicator==1)
      treatmentobspreandpost<-base::nrow(treatmentobspreandpost)

      base::save(treatmentobspreandpost,file="treatmentobspreandpost_mergers_processed.Rdata") ##export data
    }

    if(i=="closures_and_mergers"){
      data_closures_and_mergers<-data_shock

      base::save(data_closures_and_mergers,file="data_closures_and_mergers_processed.Rdata") ##export data

      utils::write.table(data_closures_and_mergers,file="data_closures_and_mergers_processed.csv",sep=",",row.names=FALSE) ##export data

      treatmentobspreandpost<-dplyr::filter(data_shock,common_shock_treatmentindicator==1)
      treatmentobspreandpost<-base::nrow(treatmentobspreandpost)

      base::save(treatmentobspreandpost,file="treatmentobspreandpost_closures_and_mergers_processed.Rdata") ##export data
    }
  }

  base::rm(list=base::ls()) ##clean
  base::gc() ##clean
  #end----------------------------------------------------------------------
```

```r
#introduction-----------------------------------------------------------------------
#this file prepares the data for the analyst-based information analysis

#prepare data-----------------------------------------------------------------------
ibes<-haven::read_dta("summary_history_summary_statistics_20200226.dta")
ibes<-base::as.data.frame(ibes)
ibes[,"year"]<-base::gsub(".{6}$","",ibes[,"statpers"])
ibes<-dplyr::filter(ibes,!oftic==="")
ibes<-dplyr::filter(ibes,!year==="")
ibes<-dplyr::filter(ibes,!base::is.na(oftic))
ibes<-dplyr::filter(ibes,!base::is.na(year))
ibes<-stats::aggregate(ibes[c("numest")],ibes[c("oftic","year")],base::mean,na.rm=TRUE) ##aggregate data

funda<-haven::read_dta("crsp_compustat_merged_fundamentals_annual_20180318.dta") ##import data
funda<-base::as.data.frame(funda) ##change to data frame
funda[,"lpermno"]<-base::as.character(funda[,"lpermno"]) ##transform variable to character
funda[,"fyear"]<-base::as.character(funda[,"fyear"]) ##transform variable to character
funda<-funda[!base::is.na(funda[,"lpermno"]),] ##delete observations with missing data
funda<-funda[!base::is.na(funda[,"fyear"]),] ##delete observations with missing data
funda<-funda[funda[,"indfmt"]=="INDL",] ##assure clean data
funda<-funda[funda[,"datafmt"]=="STD",] ##assure clean data
funda<-funda[funda[,"popsrc"]=="D",] ##assure clean data
funda<-funda[funda[,"consol"]=="C",] ##assure clean data
funda<-funda[!base::duplicated(base::paste(funda[,"lpermno"],"_",funda[,"fyear"],sep=""))
,] ##delete duplicates
funda<-funda%>%group_by(lpermno)
funda<-funda%>%dplyr::arrange(fyear,.by_group=TRUE)
funda<-funda%>%dplyr::mutate(l1_ppent=dplyr::lag(ppent,1))
funda<-funda%>%dplyr::mutate(min_fyear=base::min(fyear))
funda<-base::as.data.frame(funda)

sale_year_sic<-dplyr::filter(funda,sale>0)
sale_year_sic<-stats::aggregate(sale_year_sic[c("sale")],sale_year_sic[c("fyear","sic")],
base::sum,na.rm=TRUE) ##aggregate data
sale_year_sic<-dplyr::rename(sale_year_sic,sic_sale=sale)
sale_year_sic_share<-base::merge(funda,sale_year_sic,by.x=c("sic","fyear"),by.y=c("sic","
fyear"),all.x=FALSE,all.y=FALSE)
sale_year_sic_share<-dplyr::filter(sale_year_sic_share,sale>=0)
sale_year_sic_share[,"sale_year_sic_share"]<-sale_year_sic_share[,"sale"]/sale_year_sic_s
hare[,"sic_sale"]
sale_year_sic_share[,"sale_year_sic_share_sq"]<-sale_year_sic_share[,"sale_year_sic_share
"]^2
hindex<-stats::aggregate(sale_year_sic_share[c("sale_year_sic_share_sq")],sale_year_sic_s
hare[c("fyear","sic")],base::sum,na.rm=TRUE) ##aggregate data
hindex<-dplyr::rename(hindex,hindex=sale_year_sic_share_sq)
hindex[,"hindex_sq"]<-hindex[,"hindex"]^2

base::load("ibes_ibes_crsp_link_processed.Rdata") ##import data
ibes_ibes_crsp_link<-ibes_ibes_crsp_link[!base::duplicated(base::paste(ibes_ibes_crsp_lin
k[,"permno"],"_",ibes_ibes_crsp_link[,"year"],"_",ibes_ibes_crsp_link[,"analys"],sep=""))
,] ##delete duplicates
analyst_coverage<-ibes_ibes_crsp_link[,c("permno","analys","year")]
analyst_coverage_base<-analyst_coverage ##create temp data
analyst_coverage_portfolio<-analyst_coverage ##create temp data
base::colnames(analyst_coverage_base)<-base::paste("base_",base::colnames(analyst_coverag
e_base),sep="") ##add prefix
base::colnames(analyst_coverage_portfolio)<-base::paste("portfolio_",base::colnames(analy
st_coverage_portfolio),sep="") ##add prefix
fullportfolio<-base::merge(analyst_coverage_base,analyst_coverage_portfolio,by.x=c("base_
analys","base_year"),by.y=c("portfolio_analys","portfolio_year"),all=FALSE)
fullportfolio<-fullportfolio[!base::duplicated(base::paste(fullportfolio[,"base_permno"],
"_",fullportfolio[,"base_year"],"_",fullportfolio[,"portfolio_permno"],sep=""))),]
##delete duplicates
fullportfolio<-dplyr::filter(fullportfolio,!base_permno==portfolio_permno)
fullportfolio[,"base_number_of_firms_connected"]<-1
fullportfolio<-stats::aggregate(fullportfolio[c("base_number_of_firms_connected")],fullpo
```

```
rtfolio[c("base_permno","base_year")],base::sum,na.rm=TRUE) ##aggregate data
fullmarket<-ibes_ibes_crsp_link[,c("permno","year")]
fullmarket<-fullmarket[!base::duplicated(base::paste(fullmarket[,"permno"],"_",fullmarket[,"year"],sep="")),] ##delete duplicates
fullmarket[,"total_number_of_covered_firms"]<-1
fullmarket<-stats::aggregate(fullmarket[c("total_number_of_covered_firms")],fullmarket[c("year")],base::sum,na.rm=TRUE) ##aggregate data
connection<-base::merge(fullportfolio,fullmarket,by.x=c("base_year"),by.y=c("year"),all=FALSE)
connection[,"total_number_of_covered_firms"]<-connection[,"total_number_of_covered_firms"]-1
connection[,"connection_perc"]<-connection[,"base_number_of_firms_connected"]/connection[,"total_number_of_covered_firms"]

inst<-haven::read_dta("wrds_thomson_reuters_institutional_13f_holdings_stock_ownership_summary_20200224.dta")
inst<-base::as.data.frame(inst)
inst[,"year"]<-base::gsub(".{6}$","",inst[,"rdate"])
inst<-dplyr::filter(inst,!base::is.na(instown_perc))
inst<-dplyr::filter(inst,!base::is.na(ticker))
inst<-dplyr::filter(inst,!instown_perc==="")
inst<-dplyr::filter(inst,!ticker==="")
inst<-stats::aggregate(inst[c("instown_perc")],inst[c("year","ticker")],base::mean,na.rm=TRUE) ##aggregate data

patents<-utils::read.csv(file="patents_20180112.csv",stringsAsFactors=FALSE) ##import data
for(i in c("fdate")){
  patents[,i]<-base::as.Date(patents[,i],"%m/%d/%Y") ##transform variable to date
  patents<-patents[!base::is.na(patents[,i]),] ##delete observations with missing variables
}
for(i in c("permno")){
  patents[,i]<-base::as.character(patents[,i]) ##transform variable to character
  patents<-patents[!base::is.na(patents[,i]),] ##delete observations with missing variables
  patents<-patents[!patents[,i]==="",] ##delete observations with missing variables
}
patents[,"fyear"]<-base::gsub(".{6}$","",patents[,"fdate"]) ##generate year variable
patents[,"numpatents"]<-1 ##generate variable
patents<-stats::aggregate(patents[c("numpatents","ncites")],patents[c("fyear","permno")],base::sum,na.rm=TRUE) ##aggregate data

##dsf data
dsf<-haven::read_dta("crsp_daily_stock_20200227.dta")
dsf<-base::as.data.frame(dsf)
variables<-c("prc","cfacshr","bidlo","askhi","vol","shrout")
for(i in variables){
  dsf[,i]<-base::as.numeric(dsf[,i]) ##transform variable to numeric
}
variables<-c("permno","permco","cusip")
for(i in variables){
  dsf[,i]<-base::as.character(dsf[,i]) ##transform variable to character
}
dsf[,"dup"]<-base::paste(dsf[,"permno"],"_",dsf[,"date"],"_",dsf[,"ret"],"_",dsf[,"prc"],"_",dsf[,"vol"],sep="") ##delete duplicates ##special events have multiple entries with the same stock information
dsf<-dsf[!base::duplicated(dsf[,"dup"]),] ##delete duplicates
dsf[,"dup"]<-NULL ##delete unnecessary variables
dsf[,"prc"]<-base::abs(dsf[,"prc"]) ##transform variable to absolute value
dsf<-dsf[base::order(dsf[,"permno"],base::as.Date(dsf[,"date"])),] ##order data
dsf<-dsf%>%dplyr::group_by(permno) ##group data
dsf<-dsf%>%dplyr::mutate(cfacshr_lag_1=dplyr::lag(x=cfacshr,n=1)) ##calculate lags
dsf<-dsf%>%dplyr::mutate(prc_lag_1=dplyr::lag(x=prc,n=1)) ##calculate lags
dsf<-dsf%>%dplyr::mutate(askhi_lag_1=dplyr::lag(x=askhi,n=1)) ##calculate lags
dsf<-dsf%>%dplyr::mutate(bidlo_lag_1=dplyr::lag(x=bidlo,n=1)) ##calculate lags
dsf<-base::as.data.frame(dsf) ##ungroup data

dsf[,"adjfac"]<-dsf[,"cfacshr"]/dsf[,"cfacshr_lag_1"] ##define variables
```

```r
dsf[,"lclose"]<- dsf[,"adjfac"]*dsf[,"prc_lag_1"] ##define variables
dsf[,"lo"]<-dsf[,"bidlo"] ##define variables
dsf[,"hi"]<-dsf[,"askhi"] ##define variables
dsf[,"pchg"]<-0 ##adjust for overnight price changes
dsf[,"pchg"]<-base::ifelse(dsf[,"bidlo"]>dsf[,"lclose"],dsf[,"bidlo"]-dsf[,"lclose"],dsf[,"pchg"]) ##adjust for overnight price changes
dsf[,"pchg"]<-base::ifelse(dsf[,"askhi"]<dsf[,"lclose"],dsf[,"askhi"]-dsf[,"lclose"],dsf[,"pchg"]) ##adjust for overnight price changes
dsf[,"lo"]<-dsf[,"lo"]-dsf[,"pchg"] ##adjust for overnight price changes
dsf[,"hi"]<-dsf[,"hi"]-dsf[,"pchg"] ##adjust for overnight price changes
dsf[,"beta"]<-(base::log(dsf[,"hi"]/dsf[,"lo"]))^2+(log(dsf[,"askhi_lag_1"]/dsf[,"bidlo_lag_1"]))^2 ##only use the adjusted returns for the current day
dsf[,"gamma_max"]<-base::ifelse(dsf[,"hi"]>dsf[,"adjfac"]*dsf[,"askhi_lag_1"],dsf[,"hi"],dsf[,"adjfac"]*dsf[,"askhi_lag_1"]) ##only use the adjusted returns for the current day
dsf[,"gamma_min"]<-base::ifelse(dsf[,"lo"]<dsf[,"adjfac"]*dsf[,"bidlo_lag_1"],dsf[,"lo"],dsf[,"adjfac"]*dsf[,"bidlo_lag_1"]) ##only use the adjusted returns for the current day
dsf[,"gamma"]<-base::log(dsf[,"gamma_max"]/dsf[,"gamma_min"])^2 ##only use the adjusted returns for the current day
dsf[,"alpha"]<-base::sqrt(dsf[,"beta"])*(1+base::sqrt(2))-base::sqrt(dsf[,"gamma"]/(3-2*base::sqrt(2))) ##simplified equation 18
dsf[,"spread"]<-2*(base::exp(dsf[,"alpha"])-1)/(1+base::exp(dsf[,"alpha"])) ##simplified equation 18
dsf[,"spread_zero"]<-base::ifelse(dsf[,"spread"]>0,dsf[,"spread"],0) ##set negativ spreads to zero
dsf[,"spread_zero"]<-base::ifelse(base::is.na(dsf[,"spread"]),NA,dsf[,"spread_zero"]) ##set negativ spreads to zero
dsf[,"year"]<-base::gsub(".{6}$","",dsf[,"date"])
dsf<-stats::aggregate(dsf[c("spread_zero")],dsf[c("permno","year")],base::mean,na.rm=TRUE) ##aggregate data
dsf<-dplyr::filter(dsf,!is.nan(spread_zero))
dsf<-dplyr::filter(dsf,!is.na(spread_zero))

merge<-base::merge(funda,ibes,by.x=c("tic","fyear"),by.y=c("oftic","year"),all.x=TRUE,all.y=FALSE)
merge<-base::merge(merge,inst,by.x=c("tic","fyear"),by.y=c("ticker","year"),all.x=TRUE,all.y=FALSE)
merge[,"fyear_p3"]<-base::as.numeric(merge[,"fyear"])+3
merge<-base::merge(merge,patents,by.x=c("lpermno","fyear_p3"),by.y=c("permno","fyear"),all.x=TRUE,all.y=FALSE)
merge<-base::merge(merge,connection,by.x=c("lpermno","fyear"),by.y=c("base_permno","base_year"),all.x=TRUE,all.y=FALSE)
merge<-base::merge(merge,hindex,by.x=c("sic","fyear"),by.y=c("sic","fyear"),all.x=TRUE,all.y=FALSE)
merge<-base::merge(merge,dsf,by.x=c("lpermno","fyear"),by.y=c("permno","year"),all.x=TRUE,all.y=FALSE)

merge<-dplyr::filter(merge,fyear<=2005)
merge<-dplyr::filter(merge,fyear>=1993)

data<-merge

data[,"l3_numpatents"]<-data[,"numpatents"]
data[,"l3_ncites"]<-data[,"ncites"]
data[,"l3_ncites_numpatents"]<-data[,"l3_ncites"]/data[,"l3_numpatents"]

data[,"l3_numpatents"]<-base::ifelse(base::is.na(data[,"l3_numpatents"]),0,data[,"l3_numpatents"])
data[,"l3_ncites"]<-base::ifelse(base::is.na(data[,"l3_ncites"]),0,data[,"l3_ncites"])
data[,"l3_ncites_numpatents"]<-base::ifelse(base::is.na(data[,"l3_ncites_numpatents"]),0,data[,"l3_ncites_numpatents"])

data[,"sic3"]<-base::gsub(".{1}$","",data[,"sic"])
data[,"sic3_fyear"]<-base::paste(data[,"sic3"],data[,"fyear"])

data[,"xrd_adj"]<-base::ifelse(base::is.na(data[,"xrd"]),0,data[,"xrd"])
data[,"randdassets"]<-data[,"xrd_adj"]/data[,"at"]
data[,"roa"]<-data[,"oibdp"]/data[,"at"]
data[,"capexassets"]<-data[,"capx"]/data[,"at"]
data[,"ppeassets"]<-data[,"ppent"]/data[,"at"]
```

```r
data[,"leverage"]<-(data[,"dltt"]+data[,"dlc"])/data[,"at"]

data[,"txdb_adj"]<-base::ifelse(base::is.na(data[,"txdb"]),0,data[,"txdb"])
data[,"tobinq"]<-((data[,"csho"]*data[,"prcc_f"])+data[,"at"]-data[,"ceq"]-data[,"txdb_adj"])/data["at"]

data[,"kzcash_flow"]<-(data[,"ib"]+data[,"dp"])/data[,"ppent"]
data[,"kzq"]<-(data[,"at"]+(data[,"csho"]*data[,"prcc_f"])-data[,"ceq"]-data[,"txdb_adj"])/data[,"at"]
data[,"kzleverage"]<-(data[,"dltt"]+data[,"dlc"])/(data[,"dltt"]+data[,"dlc"]+data[,"seq"])
data[,"kzdividends"]<-(data[,"dvc"]+data[,"dvp"])/data[,"ppent"]
data[,"kzcash_holdings"]<-data[,"che"]/data[,"l1_ppent"]
data[,"kzindex"]<-(-1.002*data[,"kzcash_flow"])+(0.283*data[,"kzq"])+(3.139*data[,"kzleverage"])+(-39.368*data[,"kzdividends"])+(-1.315*data[,"kzcash_holdings"])

data[,"age"]<-base::as.numeric(data[,"fyear"])-base::as.numeric(data[,"min_fyear"])

data[,"l3_numpatents"]<-DescTools::Winsorize(data[,"l3_numpatents"],probs=c(0.00,0.99),na.rm=TRUE)
data[,"l3_ncites_numpatents"]<-DescTools::Winsorize(data[,"l3_ncites_numpatents"],probs=c(0.00,0.99),na.rm=TRUE)

data[,"numest"]<-DescTools::Winsorize(data[,"numest"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"at"]<-DescTools::Winsorize(data[,"at"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"randdassets"]<-DescTools::Winsorize(data[,"randdassets"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"ppeassets"]<-DescTools::Winsorize(data[,"ppeassets"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"capexassets"]<-DescTools::Winsorize(data[,"capexassets"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"leverage"]<-DescTools::Winsorize(data[,"leverage"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"roa"]<-DescTools::Winsorize(data[,"roa"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"tobinq"]<-DescTools::Winsorize(data[,"tobinq"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"kzindex"]<-DescTools::Winsorize(data[,"kzindex"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"instown_perc"]<-DescTools::Winsorize(data[,"instown_perc"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"age"]<-DescTools::Winsorize(data[,"age"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"hindex"]<-DescTools::Winsorize(data[,"hindex"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"hindex_sq"]<-DescTools::Winsorize(data[,"hindex_sq"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"connection_perc"]<-DescTools::Winsorize(data[,"connection_perc"],probs=c(0.01,0.99),na.rm=TRUE)
data[,"spread_zero"]<-DescTools::Winsorize(data[,"spread_zero"],probs=c(0.01,0.99),na.rm=TRUE)

data[,"log_l3_numpatents"]<-base::log(1+data[,"l3_numpatents"])
data[,"log_l3_ncites_numpatents"]<-base::log(1+data[,"l3_ncites_numpatents"])

data[,"log_numest"]<-base::log(1+data[,"numest"])
data[,"log_at"]<-base::log(data[,"at"])
data[,"log_age"]<-base::log(data[,"age"])
data[,"log_spread_zero"]<-base::log(data[,"spread_zero"])

data<-data[!base::is.na(data[,"numest"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"numest"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"age"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"age"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"at"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"at"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"randdassets"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"randdassets"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"roa"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"roa"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"capexassets"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"capexassets"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"ppeassets"]),] ##delete observations with missing data
```

```r
data<-data[!base::is.infinite(data[,"ppeassets"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"leverage"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"leverage"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"tobinq"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"tobinq"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"instown_perc"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"instown_perc"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"kzindex"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"kzindex"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"numest"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"numest"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"connection_perc"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"connection_perc"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"hindex"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"hindex"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"hindex_sq"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"hindex_sq"]),] ##delete observations with missing data
data<-data[!base::is.na(data[,"spread_zero"]),] ##delete observations with missing data
data<-data[!base::is.infinite(data[,"spread_zero"]),] ##delete observations with missing data

consistencycheck<-base::nrow(data) ##security break
data<-data[!base::duplicated(base::paste(data[,"lpermno"],"_",data[,"fyear"],sep="")),] ##delete duplicates
base::stopifnot(consistencycheck==base::nrow(data)) ##security break

base::save(data,file="he_tian_data_processed.Rdata") ##export data
#end----------------------------------------------------------------------
```