

D.1 "investment_rate_bea.do"

```

/*This program construct the investment rate in Clementi and Palazzo*/
clear
clear matrix

set more off
cd "/Users/administrator/Dropbox/Data_for_Lu"

/*Load Dataset: Quarterly CRSP-Compustat merged*/
use investment_data_may_2017_ccm2

/*****/
/*Fiscal and Calendar Dates*****/
/*****/
gen month=month(datadate)
gen year=year(datadate)

gen quarter=1 if month<4
replace quarter =2 if month>=4&month<7
replace quarter =3 if month>=7&month<10
replace quarter =4 if month>=10&month<=12

gen date = yq(year, quarter)
gen time = date
format time %tq

*ONLY USA INCORPORATED COMPANIES
*FIC: Incorporation Code - Foreign indicates the country in which a company is
incorporated.
drop if fic~="USA"

/*****/
/*Drop if observations largerg than 2016q4*/
/*****/
drop if date>=228

/*****/
/*Drop financials, utilities, and govt*/
/*****/
destring sic, replace force
drop if sic>=4900 & sic<=4999
drop if sic>=6000 & sic<=6999
drop if sic>=9000

/*****/
/*Unique PERMCO within GVKEY*/
/*****/
sort gvkey lpermco

```

```

by gvkey: gen aux=1 if lpermco~=lpermco[_n-1]&gvkey==gvkey[_n-1]
by gvkey: egen aux_max=max(aux)
drop if aux_max==1
drop aux*

```

```

/*****/
/*Eliminate observations that have a double entry for the same GVKEY*/
/*****/

```

*Each gvkey has multiple securities outstanding (i.e., multiple permno)
 *permno is the issue identifier in CRSP, so we need to avoid double-counting observations
 *We count how many observations for each permno at the GVKEY level and then we keep the
 *accounting data relative to the permno with the largest numbers of available observatons

*GVKEY is a unique six-digit number key assigned to each company

*PERMCO is a unique permanent company identification number assigned by CRSP to all companies
 *with issues on a CRSP File. This number is permanent for all securities issued by this company regardless of name changes.

*PERMNO is a unique permanent security identification number assigned by CRSP to each security.

```

*Count permno by gvkey
sort gvkey lpermno
by gvkey lpermno : gen number_obs=_N
sort gvkey datadate
*Drop all permno less than max permno
by gvkey : egen max_obs=max(number_obs )
drop if number_obs <max_obs

```

```

*What if we have a tie? Keep the permno with the lowest value (security issued first)
sort gvkey datadate lpermno
by gvkey datadate: drop if lpermno~=lpermno[_n-1] &gvkey==gvkey[_n-1]

```

```

*If we still have multiple permno, we eliminate these obs
sort gvkey lpermno datadate
by gvkey: gen check=1 if lpermno ~=lpermno[_n-1]&gvkey ==gvkey[_n-1]
by gvkey: egen check_max=max(check)
drop if check_max==1
drop check*

```

*We can still have multiple identical entries that we eliminate

```
sort gvkey date
by gvkey: drop if date==date[_n-1]
```

*We end up with a dataset that has one security issue for each firm, we will
*match this security issue with CRSP data to perform the portfolio analysis

```
/******  
/*Generate an indicator variable for fiscal year lags*/  
/******  
gen fdate = yq(fyearq, fqtr)  
sort gvkey fdate  
by gvkey: gen fdate_lag = fdate-fdate[_n-1]  
/*Drop multiple observations within a fiscal quarter*/  
by gvkey fdate, sort: drop if fdate==fdate[_n-1]
```

```
/******  
/*Check that we have one observation per gvkey per fiscal quarter*/  
/******  
destring gvkey, replace  
sort gvkey fdate  
by gvkey fdate: egen check=count(gvkey)  
tab check  
drop check
```

```
/******  
//Create book value of equity*/  
/******
```

```
//Definition from French's website:  
http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/Data\_Library/variable\_definitions.html
```

```
//Book equity is constructed from Compustat data or collected from the Moody's  
Industrial, Financial, and Utilities manuals.
```

```
//BE is the book value of stockholders' equity, plus balance sheet deferred taxes  
and investment tax credit (if available), minus the book value of preferred stock.  
//Depending on availability, we use the redemption, liquidation, or par value (in  
that order) to estimate the book value of preferred stock.
```

```
//Stockholders' equity is the value reported by Moody's or Compustat, if it is  
available. If not, we measure stockholders' equity as the book value of common  
equity
```

```
//plus the par value of preferred stock, or the book value of assets minus total  
liabilities (in that order).
```

```
//See Davis, Fama, and French, 2000, "Characteristics, Covariances, and Average  
Returns: 1929-1997," Journal of Finance, for more details.
```

```
//Definition from Frazzini
```

```
//To obtain shareholders' equity we use we use Stockholders' Equity (SEQ) but if
```

```

not available,
//we use the sum of Common Equity (CEQ) and Preferred Stocks (PSTK). If both SEQ
and CEQ are unavailable,
//we proxy shareholders' equity by Total Assets (AT) minus the sum of Total
Liability (LT) and Minority Interest (MIB). To obtain book equity (BE),
//we subtract from shareholders' equity the preferred stock value (PSTKRV, PSTKL or
PSTK depending on availability).

```

```

//We follow Palazzo (2012) to construct BE at quarterly frequency

```

```

//BE = Shareholders' equity + deferred taxes and investment tax credits - book
value of preferred stock
rename ceqq equity_common
rename seqq equity
rename txditcq tax_credit
rename pstkq preferred_par
rename pstkrq preferred_redemption
replace tax_credit=0 if tax_credit==.

```

```

//First we build shareholders' equity
gen shareholders_equity = equity
replace shareholders_equity= equity_common+preferred_par if
shareholders_equity==.&equity_common~=.&preferred_par~=.
replace shareholders_equity= atq-ltq if shareholders_equity==.&atq~=.&ltq~=.

```

```

//Then we add balance sheet deferred taxes and investment tax credit
replace shareholders_equity=shareholders_equity+ tax_credit

```

```

//In the last step we subtract the preferred stock value
gen be= shareholders_equity -preferred_redemption if
shareholders_equity~=.&preferred_redemption~=.
replace be =shareholders_equity -preferred_par if
shareholders_equity~=.&preferred_redemption==.&preferred_par~=.

```

```

keep datadate date fdate fdate_lag gvkey capxy sppey ppegqtq ppentq prccq cshoq
aqcy atq fqtr fyearq sic dp* be rdq lperm* /*
*/ cusip exchg tax_credit fyearq fqtr cheq preferred_par dlcq dlittq actq invtq

```

```

/*****
/*****CREATE THE NET INVESTMENT FLOW VARIABLE*****/
/*****

```

```

/*****
/*Convert year-to-date (cumulative) data into quarterly flows*/
/*****

```

```
*MERGE DEFLATOR DATA
sort date
merge date using deflator
drop if _merge~=3
drop _merge
```

```
//PPENT AND PPEGT IN REAL TERMS USING NON RESIDENTIAL FIXED ASSET DEFLATOR
```

```
replace ppent=100*ppent/deflator
replace ppegt=100*ppegt/deflator
replace capxy=100*capxy/deflator
replace sppey=100*sppey/deflator
```

```
sort gvkey fdate
```

```
/*Replace missing observations between ppen_{t-1} and ppen_{t+1}, where ppen_{t-1}
and ppen_{t+1} are non missing*/
```

```
*before replacement 585,871 non-missing values
gen dummy=0
by gvkey: replace dummy=1 if ppent==.&ppent[_n-1]~=.
by gvkey: replace dummy=0 if ppent==.&ppent[_n+1]>=.
```

```
replace ppent=(ppent[_n-1]+ppent[_n+1])/2 if dummy==1
```

```
*before replacement 587,248 non-missing values. We added 1,377 observations
(0.235%)
```

```
sort gvkey fdate
/*Gross investment: INV */
gen capx_gross=ppegt-ppegt[_n-1] if fdate_lag==1
```

```
/*NET investment: INV - DEP */
gen capx_net=ppent-ppent[_n-1] if fdate_lag==1
```

```
/******
/* Generate the acquisition to assets ratio*/
/******
sort gvkey fyearq fqtr
gen aqc = aqcy
replace aqc = aqcy - aqcy[_n-1] if fqtr~=1&fdate_lag==1
replace aqc = 0 if aqc==.
gen aqc_ratio=aqc/atq[_n-1] if fdate_lag==1
replace aqc_ratio=0 if aqc_ratio==.
```

```
replace aqcy = 0 if aqcy==.
sort gvkey fdate
gen aqc_ratio_annual=aqcy/atq[_n-4] if fdate==fdate[_n-4]+4&fqtr==4
replace aqc_ratio_annual=0 if aqc_ratio_annual==.&fdate==fdate[_n-4]+4&fqtr==4
```

```

//Generate the investment flow using CAPEX and SPPE for comparison purposes
sort gvkey fyearq fqtr
gen capxq = capxy
by gvkey: replace capxq = capxy - capxy[_n-1] if fqtr~=1&fdate_lag==1
gen sppeq = sppey
by gvkey: replace sppeq = sppey - sppey[_n-1] if fqtr~=1&fdate_lag==1

keep datadate date deflator fdate fdate_lag gvkey sic ppentq ppegtq capx_net
capx_gross/*
*/ capxq sppeq capxy sppey prccq cshoq aqc_ratio* dp* be rdq lperm* cusip exchg
fyearq fqtr tax_credit cheq preferred_par invtq dlcq dlttq actq
sort gvkey date

drop if capx_net==.
save temp, replace

/*****
/*****CREATE THE CAPITAL STOCK AT THE FIRM LEVEL*****/
/*****/

gen k_stock=ppentq
drop if k_stock==.

sort gvkey date
replace k_stock=. if gvkey==gvkey[_n-1]

gen dummy=1 if k_stock~=.
drop if k_stock==.

keep gvkey date k_stock dummy
sort gvkey date

merge gvkey date using temp
sort gvkey date

drop _merge

drop fdate_lag
sort gvkey fdate
by gvkey: gen fdate_lag=fdate-fdate[_n-1]

//sort gvkey fdate
//Generate the capital stock recursively
by gvkey: replace k_stock=k_stock[_n-1] +capx_net if dummy~=1&fdate_lag==1

```

```

//Calculate number of observations
by gvkey : gen obs= _n

//Merge using BEA deflators
gen sic3=floor(sic/10)
sort sic3
merge sic3 using delta_bea

drop if _merge==2
drop _merge

//If delta at the 3-digit SIC code is missing we calculate the average at the
2-digit level
gen sic2=floor(sic/100)
bysort sic2: egen average_delta_sic2=mean(delta_bea)

replace delta_bea=average_delta_sic2 if delta_bea==.

//If delta at the 2-digit SIC code is missing we calculate the average at the
1-digit level
gen sic1=floor(sic/1000)
bysort sic1: egen average_delta_sic1=mean(delta_bea)

//If delta still missing, we use the unconditional mean
egen average_delta=mean(delta_bea)
replace delta_bea=average_delta if delta_bea==.

//Accounting depreciation for comparison purposes
sort gvkey fdate
by gvkey: gen delta_accounting=(capx_gross-capx_net)/k_stock[_n-1] if fdate_lag==1
egen upperbound= pctlile(delta_accounting) , p(99.5)
egen lowerbound= pctlile(delta_accounting) , p(0.5)
replace delta_accounting=. if delta_accounting<lowerbound
replace delta_accounting=. if delta_accounting>upperbound
drop upperbound lowerbound

drop average_delta* sic1 sic2 sic3

//Investment rate
drop capx_gross
sort gvkey fdate
by gvkey: gen capx_gross=capx_net+delta_bea*k_stock[_n-1] if fdate_lag==1
by gvkey: gen inv_rate_recursive = capx_gross/k_stock[_n-1] if fdate_lag==1
by gvkey: gen inv_rate_annual = (capxy-sppey)/ppentq if fdate==fdate[_n-4]+4
by gvkey: gen inv_rate_accounting = (capxq-sppeq)/ppentq[_n-1] if fdate_lag==1
by gvkey: gen inv_rate_accounting_annual = (capxy-sppey)/ppentq[_n-4] if
fdate==fdate[_n-4]+4

```

```

gen time = date
format time %tq

save investment_rate_data_bea, replace

/*****
/*****
/*****
/*****COMPUTE STATISTICS*****/
/*****
/*****
/*****

cd
"/Users/administrator/Desktop/work/Research/Investment_equity_returns/data/quarterl
y_analysis/May_2017/inv_rate"

use investment_rate_data_bea, clear

/*****
/*Drop top and bottop 0.5% for investmet rates, now we work wiht truncated sample*/
/*****

//Drop first 12 quarters
replace inv_rate_recursive=. if obs <=12

//Set sample from 1978q1 to 2016q4 (168 quarters)
drop if date<72
drop if date>227

//Drop negative k_stock and investment rate less tha 100%
replace inv_rate_recursive=. if k_stock<=0
replace inv_rate_recursive=. if inv_rate_recursive<=-1

//Drop if acquisition larger than 5% of assets in a given quarter
drop if aqc_ratio >0.05
drop if aqc_ratio <-0.05

//Eliminate top and bottom 0.5% to dampen impact of outliers
egen upperbound= pctlile(inv_rate_recursive) , p(99.5)
egen lowerbound= pctlile(inv_rate_recursive) , p(0.5)
replace inv_rate_recursive=. if inv_rate_recursive>upperbound
replace inv_rate_recursive=. if inv_rate_recursive<lowerbound
drop upperbound lowerbound
gen inv_rate= inv_rate_recursive
drop if inv_rate==.

egen upperbound= pctlile(inv_rate_accounting) , p(99.5)

```

```

egen lowerbound= pctlile(inv_rate_accounting) , p(0.5)
replace inv_rate_accounting=. if inv_rate_accounting>upperbound
replace inv_rate_accounting=. if inv_rate_accounting<lowerbound
drop upperbound lowerbound

//Here we calculate the book-to-market ratio using an alternative approach used in
empirical asset pricing
//The market value of equity is common shares outstanding times the fiscal end
stock price
gen market_value=100*(prccq*cshoq+dlttq+preferred_par-invtq)/deflator
gen bm = (k_stock)/market_value

replace bm=. if bm<=0
egen upperbound= pctlile(bm) , p(99.5)
egen lowerbound= pctlile(bm) , p(0.5)
replace bm=. if bm>upperbound
replace bm=. if bm<lowerbound
drop upperbound lowerbound

sort gvkey date
by gvkey : drop if date==date[_n-1]

/*****
/*Create an histogram for the investment rate*/
*****/
gen hist = inv_rate
replace hist = . if hist >0.2
replace hist = . if hist <-0.2

histogram hist, bin(51) percent xtitle("Investment Rate", size(medlarge))
ytitle("Percentage", size(medlarge)) plotregion(style(none)) ylabel(, angle(0))
graph export ir_distribution_bea.eps, replace

histogram hist, bin(5) percent xtitle("Investment Rate", size(medlarge))
ytitle("Percentage", size(medlarge)) plotregion(style(none)) ylabel(, angle(0))
graph export ir_distribution_bea_large_bins.eps, replace

histogram delta_bea, bin(5) percent xtitle("Depreciation Rate", size(medlarge))
ytitle("Percentage", size(medlarge)) plotregion(style(none)) ylabel(, angle(0))

graph export depreciation_bea.eps, replace

//Clean annual investment rate
//Only consider end of the fiscal year values
replace inv_rate_annual =. if fqtr ~4
replace inv_rate_accounting_annual =. if fqtr ~4

```

```

replace inv_rate_annual =. if aqc_ratio_annual>0.05&aqc_ratio_annual<.
replace inv_rate_annual =. if aqc_ratio_annual<-0.05

replace inv_rate_accounting_annual =. if aqc_ratio_annual>0.05&aqc_ratio_annual<.
replace inv_rate_accounting_annual =. if aqc_ratio_annual<-0.05

egen upperbound= pctlile(inv_rate_annual) , p(99.5)
egen lowerbound= pctlile(inv_rate_annual) , p(0.5)
replace inv_rate_annual=. if inv_rate_annual>upperbound
replace inv_rate_annual=. if inv_rate_annual<lowerbound
drop upperbound lowerbound

egen upperbound= pctlile(inv_rate_accounting_annual) , p(99.5)
egen lowerbound= pctlile(inv_rate_accounting_annual) , p(0.5)
replace inv_rate_accounting_annual=. if inv_rate_accounting_annual>upperbound
replace inv_rate_accounting_annual=. if inv_rate_accounting_annual<lowerbound
drop upperbound lowerbound

sort gvkey date

save data_firm_level_bea, replace

/*****/
/*cross--sectional observations*/
/*****/
egen obs_inv_rate = count(inv_rate) , by(date)

/*****/
/*cross--sectional mean*/
/*****/
by date, sort: egen mean_bm = mean(bm)
by date, sort: egen mean_inv_rate_aux = mean(inv_rate)
by date, sort: egen mean_inv_rate = max(mean_inv_rate_aux)
by date, sort: egen mean_inv_rate_pos_aux = mean(inv_rate) if inv_rate>0.01
by date, sort: egen mean_inv_rate_pos = max(mean_inv_rate_pos_aux)
by date, sort: egen mean_inv_rate_neg_aux = mean(inv_rate) if inv_rate<-0.01
by date, sort: egen mean_inv_rate_neg = max(mean_inv_rate_neg_aux)

/*****/
/*cross--sectional standard deviation*/
/*****/

by date, sort: egen sd_inv_rate_aux = sd(inv_rate)
by date, sort: egen sd_inv_rate = max(sd_inv_rate_aux)

/*****/
/*cross--sectional median*/
/*****/
by date, sort: egen median_inv_rate = median(inv_rate)

```

```

by date, sort: egen median_inv_rate_pos_aux = median(inv_rate) if inv_rate>0.01
by date, sort: egen median_inv_rate_pos = max(median_inv_rate_pos_aux)
by date, sort: egen median_inv_rate_neg_aux = median(inv_rate) if inv_rate<-0.01
by date, sort: egen median_inv_rate_neg = max(median_inv_rate_neg_aux)

```

```

/*****/
/*cross--sectional inaction rate*/
/*****/
egen inaction_inv_rate = count(inv_rate) if abs(inv_rate)<0.01, by(date)
bysort date: gen inaction_rate_inv_rate_aux = inaction_inv_rate/obs_inv_rate
bysort date: egen inaction_rate_inv_rate = mean(inaction_rate_inv_rate_aux)

```

```

/*****/
/*cross--sectional negative investment*/
/*****/
egen neg_inv_rate = count(inv_rate) if inv_rate<-0.01 , by(date)
bysort date: gen neg_rate_inv_rate_aux = neg_inv_rate/obs_inv_rate
bysort date: egen neg_rate_inv_rate = mean(neg_rate_inv_rate_aux)

```

```

/*****/
/*cross--sectional positive spikes*/
/*****/
egen posspike_inv_rate = count(inv_rate) if inv_rate>=0.2 , by(date)
bysort date: gen posspike_rate_inv_rate_aux = posspike_inv_rate/obs_inv_rate
bysort date: egen posspike_rate_inv_rate = mean(posspike_rate_inv_rate_aux)

```

```

/*****/
/*cross--sectional negative spikes*/
/*****/
egen negspike_inv_rate = count(inv_rate) if inv_rate<=-0.2 , by(date)
bysort date: gen negspike_rate_inv_rate_aux = negspike_inv_rate/obs_inv_rate
bysort date: egen negspike_rate_inv_rate = mean(negspike_rate_inv_rate_aux)

```

```

/*****/
/*Evaluate the investment rate serial correlation*/
/*****/
drop fdate_lag
sort gvkey fdate
by gvkey: gen fdate_lag = fdate-fdate[_n-1]
by gvkey: gen inv_rate_lag = inv_rate[_n-1] if fdate_lag==1

```

```

/*****/
/*Fama-MacBeth*/
/*****/
gen serialcorr_coeff=0
gen serialcorr_const=0
sort date
set more off
forval t=73 (1) 227 {

```

```

    reg inv_rate inv_rate_lag if date==`t'
    replace serialcorr_coeff = _b[inv_rate_lag] if date==`t'
    replace serialcorr_const = _b[_cons] if date==`t'
}

/*****/
/*Pooled OLS*/
/*****/
reg inv_rate inv_rate_lag
gen serialcorr_coeff_OLS=_b[inv_rate_lag]
gen serialcorr_const_OLS = _b[_cons]

/*****/
/*Create quarterly time-series*/
/*****/

sort date
drop if date==date[_n-1]

sort date

replace negspike_rate_inv_rate=0 if negspike_rate_inv_rate==.
replace posspike_rate_inv_rate=0 if posspike_rate_inv_rate==.

keep date serialcorr* obs_inv_rate mean_inv_rate median_inv_rate_pos
median_inv_rate_neg sd_inv_rate median_inv_rate/*
*/ inaction_rate_inv_rate neg_rate_inv_rate posspike_rate_inv_rate /*
*/ negspike_rate_inv_rate time mean_inv_rate_pos mean_inv_rate_neg mean_bm exchg
sic

/*****/
/*Export output*/
/*****/

format %6.0g mean_inv_rate sd_inv_rate median_inv_rate obs_inv_rate/*
*/ inaction_rate_inv_rate neg_rate_inv_rate posspike_rate_inv_rate
mean_inv_rate_pos mean_inv_rate_neg/*
*/ negspike_rate_inv_rate serialcorr_coeff_OLS serialcorr_const_OLS
median_inv_rate_pos median_inv_rate_neg mean_bm

sum

save investment_rate_stats_bea, replace

```

D.2 "accounting_data_cleaned.do"

```
/*This program prepares the accounting data for portfolio formation*/

clear
clear matrix

cd "/Users/administrator/Dropbox/Data_for_Lu"

use data_firm_level_bea, clear

//Calendar dates
gen calendar_year=year(datadate)
gen calendar_quarter=quarter(datadate)

//Calendar date at quarterly frequency
gen dateq =yq(calendar_year,calendar_quarter)

//We move the quarterly date 2 quarters ahead in time to match with prices in
quarters at t+2
//This allows a six-month lag between CRSP data and accounting data
replace dateq=dateq+2

keep be inv_rate* cusip lpermno lpermco dateq sic bm dlttq dlcq preferred_par
invtq k_stock delta_bea datadate

replace preferred_par=0 if preferred_par==.
replace invtq=0 if invtq==.
replace dlcq =0 if dlcq ==.
replace dlttq=0 if dlttq==.

rename lpermno permno

//We drop missing and negative be observations
drop if be<0
drop if be==.

sort permno dateq

save accounting_data_cleaned.dta, replace
```

D.3 "crsp_cleaning.do"

```

/*This program cleans the raw dataset*/
clear
clear matrix

set mem 2g
set matsize 800
set more off
cd "/Users/administrator/Dropbox/Data_for_Lu"
use CRSP_may_2017

/*****/
/*Generate Time variables*/
/*****/
gen calendar_year=year(date)
gen calendar_month=month(date)
gen calendar_quarter=quarter(date)

/*****/
/*Generate Financial variables*/
/*****/
destring ret, replace force
gen size=abs(prc)*shrout/1000
gen price=abs(prc)
gen shares=shrout/1000
destring ret, replace force

/*****/
/*Drop duplicate rows in CRSP*/
/*****/
sort permno calendar_year calendar_month
drop if date==date[_n-1]

/*****/
/*We only consider ordinary common shares*/
/*****/
drop if shrcd~=10& shrcd~=11

/*****/
/*We exclude observations relative to suspended ,
   halted or non listed shares in NYSE AMEX NASDAQ */
/*****/
sort permno
by permno: egen max_exchcd=max(exchcd)
by permno: egen min_exchcd=min(exchcd)

```

```
drop if min_exchcd<=0
drop if max_exchcd>=4
```

```
destring dlret, replace force
```

```

/*****/
/*Replace delisting return if missing following Shumway*/
/*If poor performance delisting return is missing I use*/
/*an average value of -30\% */
/*****/
replace dlret=-0.30 if dlstcd==500&dlret==.
replace dlret=-0.30 if dlstcd==520&dlret==.
replace dlret=-0.30 if dlstcd==550&dlret==.
replace dlret=-0.30 if dlstcd==551&dlret==.
replace dlret=-0.30 if dlstcd==552&dlret==.
replace dlret=-0.30 if dlstcd==560&dlret==.
replace dlret=-0.30 if dlstcd==561&dlret==.
replace dlret=-0.30 if dlstcd==570&dlret==.
replace dlret=-0.30 if dlstcd==572&dlret==.
replace dlret=-0.30 if dlstcd==574&dlret==.
replace dlret=-0.30 if dlstcd==575&dlret==.
replace dlret=-0.30 if dlstcd==580&dlret==.
replace dlret=-0.30 if dlstcd==581&dlret==.
replace dlret=-0.30 if dlstcd==582&dlret==.
replace dlret=-0.30 if dlstcd==583&dlret==.
replace dlret=-0.30 if dlstcd==584&dlret==.
replace dlret=-0.30 if dlstcd==585&dlret==.
replace dlret=-0.30 if dlstcd==587&dlret==.
replace dlret=-0.30 if dlstcd==589&dlret==.
replace dlret=-0.30 if dlstcd==591&dlret==.

```

```

/*****/
/*Replace delisting return if missing following Shumway*/
/*If NOT poor performance delisting return is missing */
/*I use an average value of -100\% */
/*****/
replace dlret=-1.00 if dlstcd>100&dlret==.&dlstcd~=.
/*****/
/*Assign the delisting return as the last available return*/
/*****/
replace ret= dlret if dlret~=.

```

```
//Create date at monthly frequency
drop date
gen date=ym(calendar_year, calendar_month)
```

```
//Create market cap lagged seven months, needed to calculate boof-to-market
sort permno calendar_year calendar_month
by permno: gen lag_size7=size[_n-7] if date==date[_n-7]+7
```

```
//Generate market cap in real values, deflator is CPI
sort date
merge date using CPI

drop if _merge~=3
drop _merge

replace size=100*size/cpi

//Generate one-month lagged market cap, needed for value weighting returns
sort permno date
by permno: gen lag_size=size[_n-1] if date==date[_n-1]+1

replace ret=100*ret

//Create date at quarterly frequency, needed to match with accounting data
gen dateq=yq(calendar_year, calendar_quarter)

sort permno dateq

save CRSP_cleaned, replace
```

D.4 "merge_crsp_compustat.do"

```
/*This program cleans the raw dataset*/
clear
clear matrix

set mem 2g
set matsize 800
set more off

cd "/Users/administrator/Dropbox/Data_for_Lu"
use accounting_data_cleaned.dta

rename cusip cusip_cs

sort permno dateq

merge permno dateq using CRSP_cleaned

tab _merge
drop if _merge~=3
drop _merge

//Calculate excess returns
sort date
merge date using FF_data
drop if _merge~=3
drop _merge

replace ret=ret-100*rf

//Generate market cap in real values, deflator is CPI
sort date
merge date using CPI

drop if _merge~=3
drop _merge

//Calculate book-to-market
sort permno date
by permno: gen be_me=be/lag_size7

drop bm
sort permno date
by permno: gen bm=k_stock/(100*(lag_size7+dlttq+dltcq+preferred_par-invtq)/cpi)

//Set size and inv_rate to missing if there is no available book-to-market
replace lag_size=. if be_me==.
replace inv_rate=. if be_me==.
```

```

replace bm=. if be_me==.
replace delta_bea=. if be_me==.

//This is log of real size
gen log_size=log(lag_size)

//NYSE dummy
gen NYSE=0
replace NYSE=1 if hexcd==1

drop hexcd price

/*****
/*Drop observations that do not match cusip from CS to cusip or ncusip from CRSP*/
*****/

gen cusip6_cs=substr(cusip_cs,1,6)
gen ncusip6_crsp=substr(ncusip,1,6)
gen cusip6_crsp=substr(cusip,1,6)

gen dummy=1 if ncusip6_crsp~=cusip6_cs
gen dummy1=1 if cusip6_crsp~=cusip6_cs
gen dummy2=dummy+dummy1
drop if dummy2~=.
drop dummy*

//Drop negative or zero values for book-market
drop if be_me<=0
winsor be_me, gen(aux) p(0.005)
replace be_me=aux
drop aux

replace bm=. if bm<=0
winsor bm, gen(aux) p(0.005)
replace bm=aux
drop aux

replace be_me=. if
calendar_month~=1&calendar_month~=4&calendar_month~=7&calendar_month~=10
replace inv_rate=. if
calendar_month~=1&calendar_month~=4&calendar_month~=7&calendar_month~=10
replace inv_rate_accounting=. if
calendar_month~=1&calendar_month~=4&calendar_month~=7&calendar_month~=10
replace inv_rate_annual=. if
calendar_month~=1&calendar_month~=4&calendar_month~=7&calendar_month~=10
replace delta_bea=. if
calendar_month~=1&calendar_month~=4&calendar_month~=7&calendar_month~=10
replace bm=. if

```

```
calendar_month~=1&calendar_month~=4&calendar_month~=7&calendar_month~=10  
gen size_at_portfolio=lag_size if be_me~=.
```

```
keep permno lpermco datadate date* calendar_month calendar_year log_size ret  
lag_size be_me NYSE inv_rate inv_rate_a* lag_size sic bm size_at_portfolio  
delta_bea k_stock
```

```
sort permno calendar_year calendar_month  
save merged_crsp_compustat_data, replace
```

D.5 "inv_rate_moments.do"

```

/*This program cleans the raw dataset*/
clear
clear matrix

set mem 2g
set matsize 800
set more off
cd "/Users/administrator/Dropbox/Data_for_Lu"
use merged_crsp_compustat_data, clear

drop if inv_rate==.
drop if be_me==.

drop date
rename dateq date

/*****/
/*cross--sectional observations*/
/*****/
egen obs_inv_rate = count(inv_rate) , by(date)

/*****/
/*cross--sectional mean*/
/*****/
by date, sort: egen mean_bm = mean(be_me)
by date, sort: egen mean_inv_rate_aux = mean(inv_rate)
by date, sort: egen mean_inv_rate = max(mean_inv_rate_aux)
by date, sort: egen mean_inv_rate_pos_aux = mean(inv_rate) if inv_rate>0.01
by date, sort: egen mean_inv_rate_pos = max(mean_inv_rate_pos_aux)
by date, sort: egen mean_inv_rate_neg_aux = mean(inv_rate) if inv_rate<-0.01
by date, sort: egen mean_inv_rate_neg = max(mean_inv_rate_neg_aux)

/*****/
/*cross--sectional standard deviation*/
/*****/

by date, sort: egen sd_inv_rate_aux = sd(inv_rate)
by date, sort: egen sd_inv_rate = max(sd_inv_rate_aux)

/*****/
/*cross--sectional median*/
/*****/
by date, sort: egen median_inv_rate = median(inv_rate)
by date, sort: egen median_inv_rate_pos_aux = median(inv_rate) if inv_rate>0.01
by date, sort: egen median_inv_rate_pos = max(median_inv_rate_pos_aux)
by date, sort: egen median_inv_rate_neg_aux = median(inv_rate) if inv_rate<-0.01
by date, sort: egen median_inv_rate_neg = max(median_inv_rate_neg_aux)

/*****/
/*cross--sectional inaction rate*/

```

```

/*****/
egen inaction_inv_rate = count(inv_rate) if abs(inv_rate)<0.01, by(date)
bysort date: gen inaction_rate_inv_rate_aux = inaction_inv_rate/obs_inv_rate
bysort date: egen inaction_rate_inv_rate = mean(inaction_rate_inv_rate_aux)

/*****/
/*cross--sectional negative investment*/
/*****/
egen neg_inv_rate = count(inv_rate) if inv_rate<-0.01 , by(date)
bysort date: gen neg_rate_inv_rate_aux = neg_inv_rate/obs_inv_rate
bysort date: egen neg_rate_inv_rate = mean(neg_rate_inv_rate_aux)

/*****/
/*cross--sectional positive spikes*/
/*****/
egen posspike_inv_rate = count(inv_rate) if inv_rate>=0.2 , by(date)
bysort date: gen posspike_rate_inv_rate_aux = posspike_inv_rate/obs_inv_rate
bysort date: egen posspike_rate_inv_rate = mean(posspike_rate_inv_rate_aux)

/*****/
/*cross--sectional negative spikes*/
/*****/
egen negspike_inv_rate = count(inv_rate) if inv_rate<=-0.2 , by(date)
bysort date: gen negspike_rate_inv_rate_aux = negspike_inv_rate/obs_inv_rate
bysort date: egen negspike_rate_inv_rate = mean(negspike_rate_inv_rate_aux)

/*****/
/*Evaluate the investment rate serial correlation*/
/*****/
sort permno date
by permno: gen date_lag =date-date[_n-1]
by permno: gen inv_rate_lag = inv_rate[_n-1] if date_lag==1

/*****/
/*Fama-MacBeth*/
/*****/
gen serialcorr_coeff=0
gen serialcorr_const=0
sort date
set more off
forval t=77 (1) 227 {

    reg inv_rate inv_rate_lag if date==`t'
    replace serialcorr_coeff = _b[inv_rate_lag] if date==`t'
    replace serialcorr_const = _b[_cons] if date==`t'
}

/*****/
/*Pooled OLS*/
/*****/

```

```

reg inv_rate inv_rate_lag
gen serialcorr_coeff_OLS=_b[inv_rate_lag]
gen serialcorr_const_OLS = _b[_cons]

/*****/
/*Create quarterly time-series*/
/*****/

save temp, replace

sort date
drop if date==date[_n-1]

sort date

drop if calendar_year<=1978

replace negspike_rate_inv_rate=0 if negspike_rate_inv_rate==.
replace posspike_rate_inv_rate=0 if posspike_rate_inv_rate==.

keep date serialcorr* obs_inv_rate mean_inv_rate median_inv_rate_pos
median_inv_rate_neg sd_inv_rate median_inv_rate/*
*/ inaction_rate_inv_rate neg_rate_inv_rate posspike_rate_inv_rate /*
*/ negspike_rate_inv_rate mean_inv_rate_pos mean_inv_rate_neg mean_bm

/*****/
/*Export output*/
/*****/

format %6.0g mean_inv_rate sd_inv_rate median_inv_rate obs_inv_rate/*
*/ inaction_rate_inv_rate neg_rate_inv_rate posspike_rate_inv_rate
mean_inv_rate_pos mean_inv_rate_neg/*
*/ negspike_rate_inv_rate serialcorr_coeff_OLS serialcorr_const_OLS
median_inv_rate_pos median_inv_rate_neg mean_bm

sum

save investment_rate_stats_bea, replace

```