Package 'farr'

June 30, 2022

Title Data and Code for Financial Accounting Research

Version 0.2.27

Description Provides handy functions and data to support a course book for accounting research. Gow, Ian and Tongqing Ding (2022) ``Accounting Research: An Introductory Course" <https://iangow.github.io/far_book/>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

Imports dbplyr (>= 2.2.0), dplyr, magrittr, rlang, tidyr, tibble, readr, stringr, DBI, lubridate

Depends R (>= 3.5.0)

Suggests RPostgres, knitr, rmarkdown, testthat (>= 3.0.0), spelling

VignetteBuilder knitr

Config/testthat/edition 3

Language en-US

NeedsCompilation no

Author Ian Gow [aut, cre] (<https://orcid.org/0000-0002-6243-8409>)

Maintainer Ian Gow <iandgow@gmail.com>

Repository CRAN

Date/Publication 2022-06-30 06:10:02 UTC

R topics documented:

`aaer_dates` *AAER dates from SEC*

Description

A data set containing dates and descriptions for AAERs

Usage

```
aaer_dates
```

Format

A tibble with 40,518 rows and 4 variables:

aaer_num AAER number

aaer_date Date

aaer_desc Description

year Year of AAER ...

`apple_events` *Dates for Apple Events*

Description

A data set containing the dates of Apple media events since 2005.

Usage

```
apple_events
```

Format

A tibble with 47 rows and 3 variables:

event Description of event

event_date First date of event

end_event_date Last date of event ...

Source

https://en.wikipedia.org/wiki/List_of_Apple_Inc._media_events

---

`bloomfield_2021` *Firm-years in RDD analysis of Bloomfield (2021).*

Description

Firm-years in RDD analysis of Bloomfield (2021).

Usage

```
bloomfield_2021
```

Format

A tibble with 1,855 rows and 2 variables:

fyear Fiscal year

permco CRSP firm identifier (PERMCO)

by_tag_year *Tags on StackOverflow*

Description

A data set containing data on tagged questions on StackOverflow

Usage

```
by_tag_year
```

Format

A tibble with 40,518 rows and 4 variables:

year Year

tag Tag

number Number of questions with tag during year

year_total Total number of questions with tag during year ...

comp *Data on accruals and auditor choice*

Description

A data set containing data about accruals for 2,000 firms.

Usage

```
comp
```

Format

A tibble with 16,237 rows and 14 variables:

gvkey GVKEY (firm identifier)

datadate Fiscal year-end

fyear Fiscal year

big_n Indicator for Big Four auditor

ta Total accruals (scaled by assets)

roa Return on assets

cfo Cash flow from operating activities (scaled by assets)

size Size

lev Leverage

mtb Market-to-book ratio

inv_at 1/Total assets

d_sale Change in revenue

d_ar Change in accounts receivable

ppe Property, plant & equipment (scaled by assets) ...

`fhk_firm_years` *Firm-years for replication of Fang, Huang and Karpoff (2016)*

Description

A data set containing the GVKEYs and datadates for firm-years used in Fang, Huang and Karpoff (2016).

Usage

```
fhk_firm_years
```

Format

A tibble with 60,272 rows × 2 variables.

gvkey GVKEY (firm identifier)

datadate Fiscal year-end

`fhk_pilot` *Treatment indicators for SHO pilot firms*

Description

A data set containing the tickers, GVKEYs, and treatment indicator for SHO pilot program.

Usage

```
fhk_pilot
```

Format

A tibble with 3,030 rows × 4 variables.

ticker Ticker

gvkey GVKEY (firm identifier)

permno PERMNO (CRSP security identifier)

pilot SHO pilot program treatment indicator

form_deciles *Form deciles*

Description

Calculate deciles for a variable.

Usage

```
form_deciles(x)
```

Arguments

x	A vector for which deciles are to be calculated.

Value

vector

Examples

```
library(farr)
library(dplyr, warn.conflicts = FALSE)

df <-
    tibble(x = rnorm(100)) %>%
    mutate(dec_x = form_deciles(x))
df
```

get_annc_dates *Produce a table mapping announcements to trading dates*

Description

Produce a table mapping announcements to trading dates. See `vignette("wrds-conn", package = "farr")` for more on using this function.

Usage

```
get_annc_dates(conn)
```

Arguments

conn	connection to a PostgreSQL database

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
library(RPostgres)
pg <- dbConnect(Postgres())
get_annc_dates(pg)

## End(Not run)
## End(Not run)
```

---

get_event_cum_rets *Produce a table of cumulative event returns*

Description

Produce a table of event returns from CRSP. See `vignette("wrds-conn", package = "farr")` for more on using this function.

Usage

```
get_event_cum_rets(
  data,
  conn,
  permno = "permno",
  event_date = "event_date",
  win_start = 0,
  win_end = 0,
  end_event_date = NULL,
  suffix = ""
)
```

Arguments

data	data frame containing data on events
conn	connection to a PostgreSQL database
permno	string representing column containing PERMNOs for events
event_date	string representing column containing dates for events
win_start	integer representing start of trading window (e.g., -1)
win_end	integer representing start of trading window (e.g., 1)
end_event_date	string representing column containing ending dates for events
suffix	Text to be appended after "ret" in variable names.

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
library(RPostgres)
pg <- dbConnect(Postgres())
events <- tibble(permno = c(14593L, 10107L),
                 event_date = as.Date(c("2019-01-31", "2019-01-31")))
get_event_cum_rets(events, pg)

## End(Not run)
## End(Not run)
```

`get_event_cum_rets_mth` *Produce a table of cumulative event returns using monthly data*

Description

Produce a table of event returns from CRSP See `vignette("wrds-conn", package = "farr")` for more on using this function.

Usage

```
get_event_cum_rets_mth(
  data,
  conn,
  permno = "permno",
  event_date = "event_date",
  win_start = 0,
  win_end = 0,
  end_event_date = NULL,
  suffix = ""
)
```

Arguments

`data`	data frame containing data on events
`conn`	connection to a PostgreSQL database
`permno`	string representing column containing PERMNOs for events
`event_date`	string representing column containing dates for events
`win_start`	integer representing start of trading window (e.g., -1) in months

win_end	integer representing start of trading window (e.g., 1) in months
end_event_date	string representing column containing ending dates for events
suffix	Text to be appended after "ret" in variable names.

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
library(RPostgres)
pg <- dbConnect(Postgres())
events <- tibble(permno = c(14593L, 10107L),
                 event_date = as.Date(c("2019-01-31", "2019-01-31")))
get_event_cum_rets_mth(events, pg)

## End(Not run)
## End(Not run)
```

---

get_event_dates *Produce a table mapping announcements to trading dates*

Description

Produce a table of event dates for linking with CRSP. See `vignette("wrds-conn", package = "farr")` for more on using this function.

Usage

```
get_event_dates(
  data,
  conn,
  permno = "permno",
  event_date = "event_date",
  win_start = 0,
  win_end = 0,
  end_event_date = NULL
)
```

Arguments

data	data frame containing data on events
conn	connection to a PostgreSQL database
permno	string representing column containing PERMNOs for events
event_date	string representing column containing dates for events
win_start	integer representing start of trading window (e.g., -1)
win_end	integer representing start of trading window (e.g., 1)
end_event_date	string representing column containing ending dates for events

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
pg <- dbConnect(RPostgres::Postgres())
events <- tibble(permno = c(14593L, 10107L),
                 event_date = as.Date(c("2019-01-31", "2019-01-31")))
get_event_dates(events, pg, win_start = -3, win_end = + 3)

## End(Not run)
## End(Not run)
```

---

get_event_rets *Produce a table of event returns*

Description

Produce a table of event returns from CRSP. See `vignette("wrds-conn", package = "farr")` for more on using this function.

Usage

```
get_event_rets(
  data,
  conn,
  permno = "permno",
  event_date = "event_date",
  win_start = 0,
  win_end = 0,
  end_event_date = NULL
)
```

Arguments

data	data frame containing data on events
conn	connection to a PostgreSQL database
permno	string representing column containing PERMNOs for events
event_date	string representing column containing dates for events
win_start	integer representing start of trading window (e.g., -1)
win_end	integer representing start of trading window (e.g., 1)
end_event_date	string representing column containing ending dates for events

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
pg <- dbConnect(RPostgres::Postgres())
events <- tibble(permno = c(14593L, 10107L),
                 event_date = as.Date(c("2019-01-31", "2019-01-31")))
get_event_rets(events, pg, win_start = -3, win_end = +3) %>%
  select(permno, event_date, date, ret)

## End(Not run)
## End(Not run)
```

---

get_ff_ind *Fetch Fama-French industry grouping.*

Description

Fetch Fama-French industry grouping from Ken French's website.

Usage

```
get_ff_ind(ind)
```

Arguments

ind	Fama-French industry grouping (e.g., 11, 48)

Value

tbl_df

Examples

```
## Not run:
get_ff_ind(5)
## End(Not run)
```

`get_got_data` *Generate simulated data as described in Gow, Ormazabal and Taylor (2010).*

Description

Function to generate simulated panel data as described in Gow, Ormazabal and Taylor (2010).

Usage

```
get_got_data(N = 400, T = 20, Xvol, Evol, rho_X, rho_E)
```

Arguments

`N`	Number of firms
`T`	Number of years
`Xvol`	Cross-sectional correlation of *X*
`Evol`	Cross-sectional correlation of errors
`rho_X`	Autocorrelation coefficient for firm-effect portion of *X*
`rho_E`	Autocorrelation coefficient for firm-effect portion of epsilon

Value

tibble

Examples

```
set.seed(2021)
test <- get_got_data(N = 500, T = 10, Xvol = 0.75,
                     Evol = 0.75, rho_X = 0.5, rho_E = 0.5)
test
```

get_idd_periods *Period for Inevitable Disclosure Doctrine (IDD)*

Description

Periods defined by precedent-setting legal cases adopting or rejecting the Inevitable Disclosure Doctrine (IDD) by state.

Usage

```
get_idd_periods(min_date, max_date)
```

Arguments

min_date	First date of sample period
max_date	Last date of sample period

Details

Three kinds of period by state:

- Pre-adoption
- Post-adoption
- Post-rejection

Value

tibble with four columns: state, period_type, start_date, end_date

Examples

```
idd_periods <- get_idd_periods(min_date = "1994-01-01",
                               max_date = "2010-12-31")
idd_periods
```

get_me_breakpoints *Create a table of with cut-offs for size portfolios*

Description

Create a table of with cut-offs for size portfolios

Usage

```
get_me_breakpoints()
```

Value

tbl_df

Examples

```
library(dplyr, warn.conflicts = FALSE)
get_me_breakpoints() %>% filter(month == '2022-04-01')
```

---

get_size_rets_monthly *Create a table of monthly returns for size portfolios*

Description

Create a table of monthly returns for size portfolios

Usage

```
get_size_rets_monthly()
```

Value

tbl_df

Examples

```
library(dplyr, warn.conflicts = FALSE)
get_size_rets_monthly() %>% filter(month == "2022-04-01")
```

---

get_test_scores *A function returning data on test_scores.*

Description

A function returning simulated data on test_scores.

Usage

```
get_test_scores(
  effect_size = 15,
  n_students = 1000L,
  n_grades = 4L,
  include_unobservables = FALSE,
  random_assignment = FALSE
)
```

Arguments

effect_size	Effect of attending camp on subsequent test scores.
n_students	Number of students in simulated data set.
n_grades	Number of grades in simulated data set.
include_unobservables	Include talent in returned data (TRUE or FALSE)
random_assignment	Is assignment to treatment completely random? (TRUE or FALSE)

Value

tbl_df

Examples

```
set.seed(2021)
library(dplyr, warn.conflicts = FALSE)
get_test_scores() %>% head()
```

---

get_trading_dates *Produce a table mapping dates on CRSP to "trading days"*

Description

Produce a table mapping dates on CRSP to "trading days". Returned table has two columns: date, a trading date on CRSP; td, a sequence of integers ordered by date. See `vignette("wrds-conn", package = "farr")` for more on using this function.

Usage

```
get_trading_dates(conn)
```

Arguments

conn	connection to a PostgreSQL database

Value

tbl_df

Examples

```
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
pg <- dbConnect(RPostgres::Postgres())
get_trading_dates(pg) %>%
  filter(between(date, as.Date("2022-03-18"), as.Date("2022-03-31")))

## End(Not run)
```

---

idd_dates *Dates for Inevitable Disclosure Doctrine (IDD)*

Description

Dates of precedent-setting legal cases adopting or reject the Inevitable Disclosure Doctrine (IDD) by state.

Usage

```
idd_dates
```

Format

A tibble with 24 rows and 3 variables:

state Two-letter state abbreviation

idd_date Date of precedent-setting legal case

idd_type Either "Adopt" or "Reject"

Source

doi:10.1016/j.jfineco.2018.02.008

---

iliev_2010 *Data on public float.*

Description

Data on public float of listed companies from Iliev (2010).

Usage

```
iliev_2010
```

Format

A tibble with 7,213 and 9 variables:

gvkey Compustat firm identifier (GVKEY)

fyear Fiscal year

fdate Date of end of fiscal year

pfdate Date for public float value

pfyear Year for public float value

publicfloat Public float in $ million

mr Indicator for filing of a management report

af Indicator for accelerator filer

cik SEC firm identifier (CIK)

`llz_2018` *GVKEYs used in Li, Lin and Zhang (2018)*

Description

GVKEYs used in Li, Lin and Zhang (2018)

Usage

```
llz_2018
```

Format

A tibble with 5,830 rows and 1 variable:

gvkey GVKEY

Source

https://research.chicagobooth.edu/-/media/research/arc/docs/journal/online-supplements/llz-datasheet-and-code.zip

michels_2017 *Data on firms suffering natural disasters.*

Description

Data on firms suffering natural disasters based on the sample in Michels (2017).

Usage

```
michels_2017
```

Format

A tibble with 423 rows and 12 variables:

cusip CUSIP supplied by Michels (2017)
eventdate Date of relevant natural disaster supplied by Michels (2017)
cik Matched CIK (SEC firm identifier)
permno Matched PERMNO (CRSP security identifier)
gvkey Matched GVKEY (Compustat firm identifier)
date_filed Date of next filing of type 10-Q, 10-K, 10QSB, 10-K405 after event
form_types List of relevant form types filed on date_filed
next_period_end Next fiscal period-end after event date
next_fqtr Fiscal quarter of next period-end after event date
prev_period_end Last fiscal period-end before event date
prev_fqtr Fiscal quarter of last period-end before event date
recognize Indicator for event being recognized (next_period_end before date_filed)

sho_r3000 *Russell 3000 stocks at time of SEC Reg SHO sample formation.*

Description

A data set containing the tickers and company names for Russell 3000 at time SEC created the pilot sample. Data are created from sample supplied by FHK.

Usage

```
sho_r3000
```

Format

A tibble with 3000 rows × 2 variables.

russell_ticker Ticker
russell_name Company name

sho_r3000_gvkeys *Russell 3000 sample used by SEC with GVKEYs*

Description

A data set containing the tickers, PERMNOs, GVKEYs, and treatment assignments for Russell 3000 sample used by SEC.

Usage

```
sho_r3000_gvkeys
```

Format

A tibble with 2,951 rows × 3 variables.

ticker Ticker

permno PERMNO (CRSP security identifier)

gvkey GVKEY (Compustat firm identifier)

pilot Indicator for stock being part of Reg SHO pilot program

Source

http://iangow.me/far_book/natural-revisited.html#the-sho-pilot-sample

sho_r3000_sample *Russell 3000 sample used by SEC*

Description

A data set containing the tickers, PERMNOs, and treatment assignments for Russell 3000 sample used by SEC.

Usage

```
sho_r3000_sample
```

Format

A tibble with 2,954 rows × 3 variables.

ticker Ticker

permno PERMNO (CRSP security identifier)

pilot Indicator for stock being part of Reg SHO pilot program

Source

http://iangow.me/far_book/natural-revisited.html#the-sho-pilot-sample

sho_tickers *Tickers of pilot firms for Reg SHO.*

Description

A data set containing the tickers and company names for pilot firms from Reg SHO pilot. Data are scraped from the SEC's own website.

Usage

```
sho_tickers
```

Format

A tibble with 986 rows × 2 variables.

ticker Ticker

co_name Company name

Source

https://www.sec.gov/rules/other/34-50104.htm

state_hq *Data on firm headquarters based on SEC EDGAR filings.*

Description

Data on firm headquarters based on SEC EDGAR filings. Dates related to SEC filing dates. Rather than provide dates for all filings, data are aggregated into groups of filings by state and CIK and dates are collapsed into windows over which all filings for a given CIK were associated with a given state. For example, CIK 0000037755 has filings with a CA headquarters from 1994-06-02 until 1996-03-25, then filings with an OH headquarters from 1996-05-30 until 1999-04-05, then filings with a CA headquarters from 1999-06-11 onwards. To ensure continuous coverage over the sample period, it is assumed that any change in state occurs the day after the last observed filing for the previous state.

Usage

```
state_hq
```

Format

A tibble with 24 rows and 3 variables:

cik SEC's Central Index Key (CIK)

ba_state Two-letter abbreviation of state

min_date Date of first filing with CIK-state combination in a contiguous series of filings

max_date Date of last filing with CIK-state combination in a contiguous series of filings

Source

https://sraf.nd.edu/data/augmented-10-x-header-data/

---

test_scores *Test scores*

Description

A simulated data set of test scores.

Usage

```
test_scores
```

Format

A tibble with 4000 rows and 5 variables:

id Student identifier

grade School grade at time of test

post Indicator for being in grade 10 or 11

treat Indicator for student attending camp after grade 9

score Test score

truncate *Truncate a vector.*

Description

Truncate a vector at prob and 1 - prob. Extreme values are turned in NA values.

Usage

```
truncate(x, prob = 0.01, p_low = prob, p_high = 1 - prob)
```

Arguments

x	A vector to be winsorized
prob	Level (two-sided) for winsorization (e.g., 0.01 gives 1% and 99%)
p_low	Optional lower level for winsorization (e.g., 0.01 gives 1%)
p_high	Optional upper level for winsorization (e.g., 0.99 gives 99%)

Value

vector

Examples

```
trunced <- truncate(1:100, prob = 0.05)
min(trunced, na.rm = TRUE)
max(trunced, na.rm = TRUE)
```

undisclosed_names *Customer names that represent non-disclosures.*

Description

Data to be combined with data in compsegd.seg_customer to create an indicator for non-disclosure of customer names.

Usage

```
undisclosed_names
```

Format

A tibble with 432 rows and 2 variables:

cnms Matches field in compsegd.seg_customer (WRDS)

disclosure Indicator that name is not disclosed

`winsorize` *Winsorize a vector.*

Description

Winsorize a vector at prob and 1 - prob.

Usage

```
winsorize(x, prob = 0.01, p_low = prob, p_high = 1 - prob)
```

Arguments

`x`	A vector to be winsorized
`prob`	Level (two-sided) for winsorization (e.g., 0.01 gives 1% and 99%)
`p_low`	Optional lower level for winsorization (e.g., 0.01 gives 1%)
`p_high`	Optional upper level for winsorization (e.g., 0.99 gives 99%)

Value

vector

Examples

```
winsorized <- winsorize(1:100, prob = 0.05)
min(winsorized, na.rm = TRUE)
max(winsorized, na.rm = TRUE)
```

Index