

# Basic programming for accounting researchers: Assignment #1 (2014)

Ian D. Gow

15 September 2024

## Assignment

You are interested in performing an analysis similar to Fairfield and Yohn (2001) with alternative variable specifications. Essentially, you are interested in whether disaggregation helps in the prediction of ROA. To prepare for data analysis, you have decided to gather all of the necessary variables for fiscal years from 2000–2010. Based on the result in Fairfield and Yohn (2001), you want to compare the results of the following regressions:<sup>1</sup>

$$(1) ROA_{t+1} = ROA_t + \Delta ROA_t$$

$$(2) ROA_{t+1} = ROA_t + \Delta TURNOVER_t + \Delta MARGIN_t$$

Definitions are as follows:

- $ROA_t$ : Net income in year  $t$  divided by total assets at year  $t - 1$
- $TURNOVER_t$ : Sales in year  $t$  divided by total assets at year  $t - 1$
- $MARGIN_t$ : Net income in year  $t$  divided by sales in year  $t$ .

**Your task:** Create the data set that has the variables in the above regressions using annual Compustat data.

---

<sup>1</sup>Note: these regression specifications differ from Fairfield and Yohn (2001). They are set up to facilitate programming training as opposed to testing the research question. Refer to Fairfield and Yohn (2001) for proper specification and variable definitions.

## Response

In writing this answer, I used the packages listed below.<sup>2</sup> This note was written using [Quarto](#) and compiled with [RStudio](#), an integrated development environment (IDE) for working with R. The source code for this note is available [here](#) and the latest version of this PDF is [here](#).

```
library(tidyverse)
library(modelsummary)
library(DBI)
library(dbplyr)
library(farr)
```

I start by connecting with the WRDS PostgreSQL database. Note that you should use `Sys.setenv()` as described [here](#) to set up *your* WRDS connection parameters before running this code.<sup>3</sup>

```
db <- dbConnect(RPostgres::Postgres())
```

I then create a remote data frame for Compustat North American annual data (`comp.funda`).

```
funda <- tbl(db, I("comp.funda"))
```

From this I create the “standard” version of annual Compustat data (see [here](#)).

```
funda_mod <-
  funda |>
  filter(indfmt == "INDL", datafmt == "STD",
         consol == "C", popsrc == "D")
```

In the code below, I exclude observations with either non-positive assets (`at`) or non-positive sales (`sale`). This filter also eliminates observations with missing `at` or missing `sale`. I include `system_time()` at the end to time the code. For me it takes between 7 and 12 seconds to run, as the final data have to be downloaded from the WRDS server.<sup>4</sup> Note that I keep `lag(datadate)`, as it’s important to consider whether “stub” reporting periods or gaps in the time series need to be addressed.

---

<sup>2</sup>Execute `install.packages(c("tidyverse", "DBI", "modelsummary", "dbplyr", "duckdb", "RPostgres", "farr"))` within R to install all the packages you need to run the code in this note.

<sup>3</sup>You may also need to confirm using the Duo app that WRDS uses for 2FA.

<sup>4</sup>It is not clear to me why the run time is so variable. Perhaps varying usage of the server or blockages in getting the data from somewhere in the US to Australia.

```

fy_2001_data <-
  funda_mod |>
  group_by(gvkey) |>
  window_order(datadate) |>
  filter(at > 0, sale > 0) |>
  mutate(lag_datadate = lag(datadate),
         roa = ni / lag(at),
         turnover = sale / lag(at),
         margin = ni / sale,
         d_roa = roa - lag(roa),
         d_turnover = turnover - lag(turnover),
         d_margin = margin - lag(margin)) |>
  ungroup() |>
  select(gvkey, datadate, lag_datadate, fyear,
         roa, turnover, margin, d_roa, d_turnover, d_margin) |>
  filter(between(fyear, 2000, 2010)) |>
  arrange(gvkey, datadate) |>
  collect() |>
  system_time()

```

```

user  system elapsed
0.597  0.101  13.880

```

Here is a quick snapshot of the data:








```
fy_2001_data
```

```

# A tibble: 97,664 x 10
   gvkey  datadate  lag_datadate fyear    roa turnover  margin  d_roa
   <chr>  <date>      <date>      <int>  <dbl>  <dbl>    <dbl>  <dbl>
1 001004 2001-05-31 2000-05-31   2000  0.0250    1.18  0.0212 -0.0234
2 001004 2002-05-31 2001-05-31   2001 -0.0840    0.910 -0.0923 -0.109
3 001004 2003-05-31 2002-05-31   2002 -0.0175    0.854 -0.0205  0.0665
4 001004 2004-05-31 2003-05-31   2003  0.00510    0.950  0.00537  0.0226
5 001004 2005-05-31 2004-05-31   2004  0.0218    1.05  0.0207  0.0167
6 001004 2006-05-31 2005-05-31   2005  0.0480    1.23  0.0392  0.0262
7 001004 2007-05-31 2006-05-31   2006  0.0599    1.08  0.0553  0.0119
8 001004 2008-05-31 2007-05-31   2007  0.0704    1.30  0.0543  0.0105
9 001004 2009-05-31 2008-05-31   2008  0.0577    1.05  0.0552 -0.0126
10 001004 2010-05-31 2009-05-31   2009  0.0324    0.982  0.0330 -0.0253
# i 97,654 more rows
# i 2 more variables: d_turnover <dbl>, d_margin <dbl>

```

Table 1: Summary statistics for data to replicate Fairfield and Yohn (2001)

	Unique	Missing Pct.	Mean	SD	Min	Median	Max	Histogram
fyear	11	0	2004.7	3.2	2000.0	2005.0	2010.0	
roa	92183	5	-1.0	57.6	-11122.0	0.0	1031.0	
turnover	92150	5	1.5	32.2	0.0	0.7	7268.3	
margin	97324	0	-6.0	532.3	-29319.0	0.0	150569.0	
d_roa	86670	11	-0.1	72.2	-11068.0	0.0	11120.4	
d_turnover	86677	11	-0.2	36.8	-7263.9	0.0	3101.8	
d_margin	92256	5	1.0	613.0	-27013.5	0.0	173354.2	

From the descriptive statistics in Table 1, it is clear that there are some wild outlier issues.

```
fy_2001_data |> datasummary_skim()
```

Digging deeper, from Figure 1, it seems that there is a long tail to the left for ROA.

```
fy_2001_data |>
  filter(between(roa, -5, 5)) |>
  ggplot(aes(x = roa)) +
  geom_histogram(binwidth = 0.25)
```

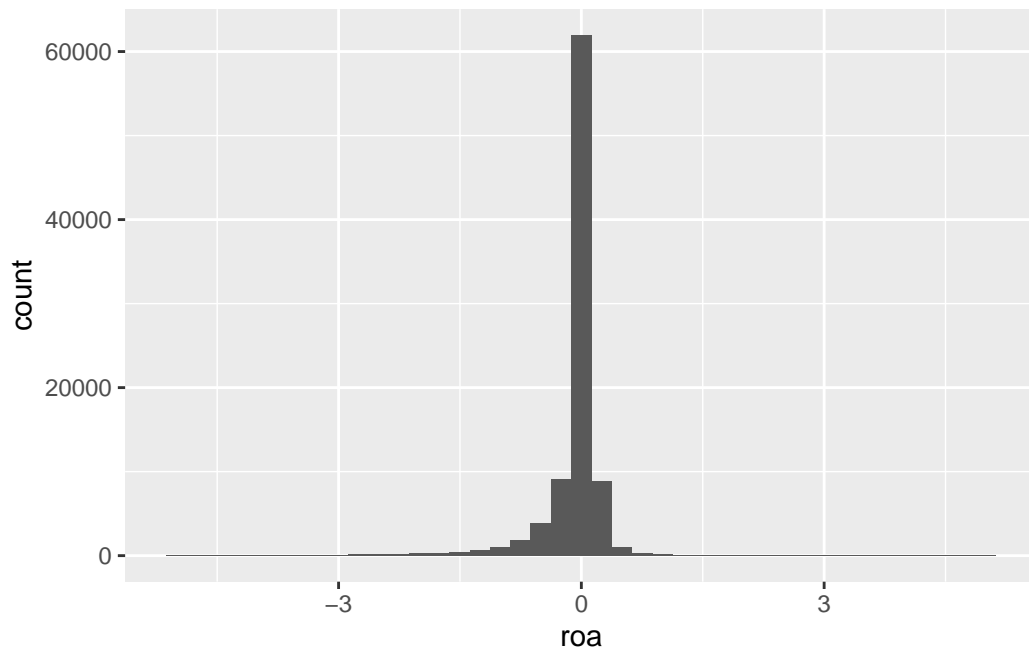


Figure 1: Histogram for ROA (restricted to  $ROA \in (-5, 5)$ )

The histogram in Figure 2 is limited to  $ROA \in (-1, 1)$  (i.e., greater than  $-100\%$  and less than  $100\%$ ) and still a long thick tail to the left is evident.

```
fy_2001_data |>
  filter(between(roa, -1, 1)) |>
  ggplot(aes(x = roa)) +
  geom_histogram(binwidth = 0.05)
```

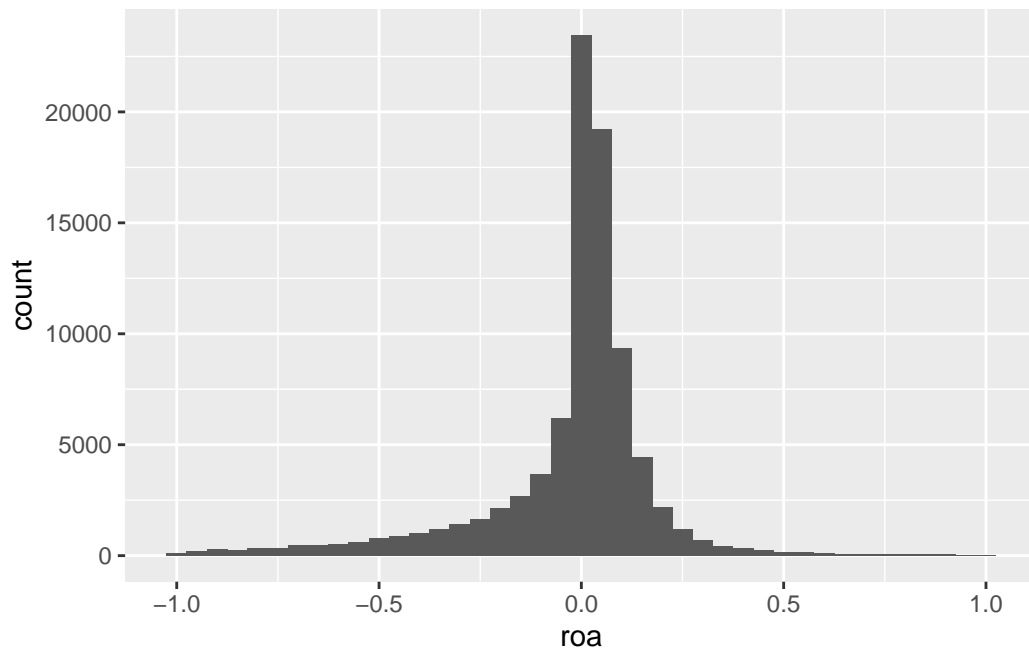


Figure 2: Histogram for ROA (restricted to  $ROA \in (-1, 1)$ )

I run the data step again, but using a local parquet repository (as described [here](#)) and DuckDB. Unsurprisingly, the code runs much faster.

```
db <- dbConnect(duckdb::duckdb())

funda <- load_parquet(db, schema = "comp", table = "funda")

funda_mod <-
  funda |>
  filter(indfmt == "INDL", datafmt == "STD",
         consol == "C", popsrc == "D")

fy_2001_data <-
  funda_mod |>
  group_by(gvkey) |>
  window_order(datadate) |>
  filter(at > 0, sale > 0) |>
  mutate(lag_datadate = lag(datadate),
         roa = ni / lag(at),
         turnover = sale / lag(at),
         margin = ni / sale,
```

```

    d_roa = roa - lag(roa),
    d_turnover = turnover - lag(turnover),
    d_margin = margin - lag(margin)) |>
ungroup() |>
select(gvkey, datadate, lag_datadate, fyear,
       roa, turnover, margin, d_roa, d_turnover, d_margin) |>
filter(between(fyear, 2000, 2010)) |>
arrange(gvkey, datadate) |>
collect() |>
system_time()

```

```

user  system elapsed
0.683  0.048   0.471

```

Yet the data appear to be the same.

fy\_2001\_data

```

# A tibble: 97,664 x 10
  gvkey  datadate  lag_datadate fyear    roa turnover  margin  d_roa
  <chr>  <date>      <date>      <int>   <dbl>   <dbl>   <dbl>  <dbl>
1 001004 2001-05-31 2000-05-31   2000  0.0250    1.18  0.0212 -0.0234
2 001004 2002-05-31 2001-05-31   2001 -0.0840    0.910 -0.0923 -0.109
3 001004 2003-05-31 2002-05-31   2002 -0.0175    0.854 -0.0205  0.0665
4 001004 2004-05-31 2003-05-31   2003  0.00510    0.950  0.00537  0.0226
5 001004 2005-05-31 2004-05-31   2004  0.0218    1.05  0.0207  0.0167
6 001004 2006-05-31 2005-05-31   2005  0.0480    1.23  0.0392  0.0262
7 001004 2007-05-31 2006-05-31   2006  0.0599    1.08  0.0553  0.0119
8 001004 2008-05-31 2007-05-31   2007  0.0704    1.30  0.0543  0.0105
9 001004 2009-05-31 2008-05-31   2008  0.0577    1.05  0.0552 -0.0126
10 001004 2010-05-31 2009-05-31   2009  0.0324    0.982  0.0330 -0.0253
# i 97,654 more rows
# i 2 more variables: d_turnover <dbl>, d_margin <dbl>

```

## References

Fairfield, Patricia M., and Teri Lombardi Yohn. 2001. "Using Asset Turnover and Profit Margin to Forecast Changes in Profitability." *Review of Accounting Studies* 6 (4): 371–85. <https://doi.org/10.1023/a:1012430513430>.