Replication Instructions for: Common Ownership in America: 1980-2017

Backus, Conlon and Sinkinson (2020) AEJMicro-2019-0389 openicpsr-120083 A copy of the paper is here: https://chrisconlon.github.io/site/common_owner.pdf

Open ICPSR Install Instructions

1. Download and unzip the repository.
2. All required files are included or are downloaded programatically from WRDS (see notes below).

Github Install Instructions

To download the repo simply type:

```
git clone https://github.com/chrisconlon/CommonOwnerReplication
```

You will need to have the git large file storage extension installed. (Which you probably do not).

To install this extension follow the directions at: https://git-lfs.github.com

Dataset Size and Memory

1. We recommend that you have at least 64GB of RAM available.
2. All of the datasets saved will take up about 14 GB of drive space.
3. NumPy is used extensively for the calculations and is multithreaded (so more cores will help).
4. The computation of the κ_{fg} terms is parallelized quarter by quarter explicitly (so cores will help a lot here).
5. But most of the time spent is in merging and filtering data in pandas (more cores don't help much).
6. Total runtime on a 2015 iMac with 64GB of RAM is around 3 hours.
7. WRDS download time is about an hour (Depends on internet speed) and total download is $>$ 10GB.

Downloading from WRDS

User must provide their own WRDS account. User will be prompted for WRDS username and password in file 1_Download_WRDS_Data.py.

To request an account, please visit: https://wrds-www.wharton.upenn.edu/register/

If you do not have API access, you will need to consult the wrds_constituents.pdf document for instructions on using the WRDS web interface. This is strongly NOT RECOMMENDED. Because you cannot apply complex filters to the SQL queries as we do programatically, you will also need much more disk space (on the order of a Terabyte to save the entire Thomson-Reuters s34 13f database.)

If you are running this on a batch job (not interactively) such as on a HPC cluster you will need to pre-enter your WRDS password by creating a pgpass file.

As an example:

```
import wrds
db = wrds.Connection(wrds_username='joe')
db.create_pgpass_file()
```

If you encounter a problem, it might be that your pgpass file is not accessible by your batch job.

For more information please see: https://wrds-www.wharton.upenn.edu/pages/support/programming-wrds/programming-python/python-from-your-computer/ for more details.

Python dependencies

Our run_all.sh bash script should install all of the required python dependencies (assuming python itself is installed correctly and you have necessary acces to install packages).

To install those dependencies manually (such as on a shared server) you may need to do the following.

Python (version 3.8 or above) - install dependencies with

```
pip3 install -r requirements.txt
```

```
numpy, pandas, matplotlib, pyarrow, brotli, seaborn, wrds, scikit-learn, pyhdfe, pyblp, stat
```

We anticipate most users will be running this replication package from within an Anaconda environment. To avoid making changes to your base environment you will want to create a separate environment for this replication package. To do that

```
conda create --name common_owner --file requirements.txt
conda activate common_owner
```

How to run the code

Change to the directory containing this file and run "./run_all.sh" on the terminal. The code should take approximately 3-10 hours to run. Tables and figures will be produced as described below.

```
cd code
./runall.sh
```

Windows Warning

Windows Users: instead use "run_all.bat" from the command prompt.

There are known conflicts between Windows 10 and core Python DLL's in versions $< 3.7.3$. If you are running on Windows 10, all Python programs will run best with Python 3.8 or later (see: https://bugs.python.org/issue35797).

File of origin for tables and figures

Table/Figure Number	Generating File
Table 1	(by hand)
Table 2	(by hand)
Table 3	table3_variance_decomp.py
Table 4	table4_kappa_correlations.py
Figure 1	plots2_kappa_official.py
Figure 2	plots1_basic_descriptives.py
Figure 3	plots1_basic_descriptives.py
Figure 4	plots1_basic_descriptives.py
Figure 5	plots3_big_three_four.py
Figure 6	plots2_kappa_official.py
Figure 7	plots2_kappa_official.py
Figure 8	plots5_investor_similarity.py
Figure 9	plots2_kappa_official.py
Figure 10	plots11_profit_simulations.py
Figure 11	plots11_profit_simulations.py
Figure 12	plots9_blackrock_vanguard.py
Figure 13	plots2_kappa_official.py
Figure 14	plots2_kappa_official.py
Figure 15	plots2_kappa_official.py
Figure 16	plots5_airlines_cereal.py
Figure 17	plots6_sole_vs_shared.py
Figure A1	plots1_basic_descriptives.py
Figure A2	plots8_individual_firm_coverage.py
Figure A3	plots10_kappa_comparison_appendix.py
Figure A4	plots7_short_interest_coverage.py
Figure A5	plots7_short_interest_coverage.py
Figure A6	plots2_kappa_official.py
Figure A7	plots2_kappa_official.py
Figure A8	plots5_investor_similarity.py

Within-File Dependencies:

1_Download_WRDS_Data.py:

`wrds_downloads`

2_Process_WRDS_Data.py

`wrds_cleaning`
`wrds_checks`

3_Calculate_Kappas.py

`kappas`
`investors`
`firminfo`
`utilities/quantiles`

plots3_big_three_four.py:

`kappas`
`investors`

plots5_airlines_cereal.py:

`kappas`

plots9_blackrock_vanguard.py:

`kappas`

plots10_kappa_comparison_appendix.py:

`utilities/matlab_util`

Files Provided and Data Access Statements

WRDS

We use several data sources from WRDS. These are accessed programatically through the WRDS API and we are not able to include individual files in this replication package. (See terms: https://wrds-www.wharton.upenn.edu/users/tou/).

They include: A. CRSP: data on securities prices and shares outstanding; list of S&P 500 constituents. B. Compustat: business fundamentals, short interest, business segment info. C. Thomson-Reuters: s34 database of 13f filings/ownership.

Author Constructed files

data/public:

The below files are publicly available csv's constructed by the authors. These are drops, consolidations, and manager identifiers that are used in our project. They are distributed with this code package.

1. manager_consolidations.csv: lists consolidated manager numbers: several manager actually correspond to one

2. permno_drops.csv: lists dropped permno IDs with reasons why they are dropped
3. big4.csv: lists manager Numbers for Blackrock, Fidelity, State Street, and Vanguard

The markups from from DLEU 2020 can be reproduced by running the replication package:

DeLoecker Eeckhout Unger Markups

4. DLE_markups_fig_v2.csv: markups from Figure 10 of DeLoecker Eeckhout Unger (QJE 2020)

De Loecker, Jan; Eeckhout, Jan; Unger, Gabriel, 2020, "Replication Data for: 'The Rise of Market Power and the Macroeconomic Implications' ", https://doi.org/10.7910/DVN/5GH8XO, Harvard Dataverse, V1

That replication package requires access to WRDS. A subset of the markups (and no additional data) are being made publicly available here.

Scraped 13f filings

The original source data are the publicly available SEC 13f filing data from EDGAR: https://www.sec.gov/edgar/searchedgar/companysearch.html

Most users instead access the Thomson-Reuters S34 database from WRDS (as our script above does). We've also scraped the original source documents from EDGAR and compiled them into an easy to use format. We provide the entire universe of 13f filings as a separate dataset. For the purposes of replicating this paper, we use three smaller extracts as parquet files:

5. cereal.parquet: extract 13F Filings for firms within the cereal industry (includes small cap)
6. airlines.parquet: extract 13F Filings for firms within the airline industry (includes small cap)
7. out_scrape.parquet: extract 13F Filings for LARGE cap firms (a superset of the S&P 500) from 1999-2017 (300MB).

Each file contains: - 13f filings going back to 1999 and end in late 2017 (Data period for this paper).

The full set of scraped 13f filings and a detailed description of how extracts were created are available in two places:

1. The live version of the 13f scraping project is https://sites.google.com/view/msinkinson/research/common-ownership-data?
2. The permanent archived version (including these extracts) is available to the public at Harvard Dataverse (doi:10.7910/DVN/ZRH3EU): https://doi.org/10.7910/DVN/ZRH3EU

Conlon, Christopher T; Sinkinson, Michael; Backus, Matthew, 2020, "Common Ownership Data: Scraped SEC form 13F filings for 1999-2017", https://doi.org/10.7910/DVN/ZRH3EU, Harvard Dataverse, V1.

Description of .parquet file format

We use the parquet format for: - Large data inputs (above) - Most intermediary datasets

Parquet files are compressed columnar storage binaries that are readable by several software packages (R, Python, Stata, Julia, C++, etc.) and platforms. The goal of the parquet project is to maintain good performance for large datasets as well as interoperability.

The storage method is stable and maintained by the Apache Foundation. https://parquet.apache.org/documentation/latest/

We use the python package "pyarrow" to read parquets and the package "brotli" for compression (listed in the requirements.txt).