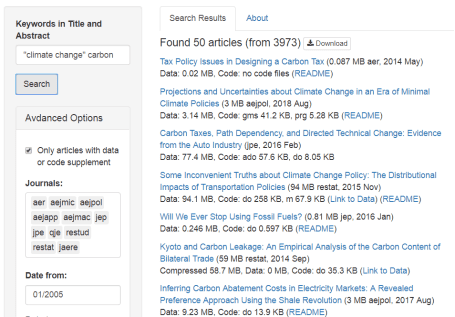
In my view, one of the greatest developments during the last decade in economics is that the [Journals of the American Economic Association](https://www.aeaweb.org/journals/) and some other leading journals require authors to upload the replication code and data sets of accepted articles.

I came across a Shiny app that allows to search currently among more than 3000 articles that have an accessible data. Just click here to use it:

One can perform a keyword search among the abstract and title. The screenshot shows an example:



One gets some information about the size of the data files.

The main idea is that such a search function could be helpful for teaching economics and data science. For example, my students can use the app to find an interesting topic for a Bachelor or Master Thesis in form of an interactive analysis with RTutor. You could also generate a topic list for a seminar, in which students shall replicate some key findings of a resarch article.

While the app performs well for a single user, I have not tested the performance for many simultaneous users. If it is too sluggish or you don’t get connected there are perhaps currently too many users. Then just try it out a bit later.

If you want to analyse yourself the collected data underlying the search app, you can download the zipped SQLite databases using the following links:

* Main database (should suffice for most analyses):  
  <http://econ.mathematik.uni-ulm.de/ejd/articles.zip>

I try to update the databases regularly.

Below is an example, for a simple analysis based on that databases. First make sure that you download and extract articles.zip into your working directory.

We first open a database connection

library(RSQLite)

db = dbConnect(RSQLite::SQLite(),"articles.sqlite")

File type conversion between databases and R can sometimes be a bit tedious. For example, SQLite knows no native Date or logical type. For this reason, I typically use package dbmisc when working with SQLite databases. It allows to specify a database schema as simple yaml file and has a lot of convenience function to retrieve or modify data that automatically use the provided schema. The following code sets the database schema that is provided in the package EconJournalData:

library(dbmisc)

db = set.db.schemas(db,schema.file=

system.file("schema/articles.yaml", package="EconJournalData"))

Of course, for a simple analysis as ours below just using the standard function in the DBI package without schemata suffices. But I am just used to working with the dbmisc package.

The main information about articles is stored in the table article

# Get the first 4 entries of articles as data frame

dbGet(db, "article",n = 4)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **id** | **year** | **date** | **journ** | **title** | **vol** | **issue** | **artnum** | **article\_url** | **has\_data** | **data\_url** | **size** | **unit** | **files\_txt** | **downloaded\_file** | **num\_authors** | **file\_info\_stored** | **file\_info\_summarized** | **abstract** | **readme\_file** |
| aer\_108\_11\_1 | 2018 | 2018-11-01 | aer | Firm Sorting and Agglomeration | 108 | 11 | 1 | https://www.aeaweb.org/articles?id=10.1257/aer.20150361 | TRUE | https://www.aeaweb.org/doi/10.1257/aer.20150361.data | 0.05339 | MB | NA | aer\_vol\_108\_issue\_11\_article\_1.zip | NA | TRUE | NA | Abstract To account for the uneven distribution of economic activity in space, I propose a theory of the location choices of heterogeneous firms in a variety of sectors across cities. In equilibrium, the distribution of city sizes and the sorting patterns of firms are uniquely determined and affect aggregate TFP and welfare. I estimate the model using French firm-level data and find that nearly half of the productivity advantage of large cities is due to firm sorting, the rest coming from agglomeration economies. I quantify the general equilibrium effects of place-based policies: policies that subsidize smaller cities have negative aggregate effects. | aer/2018/aer\_108\_11\_1/READ\_ME.pdf |
| aer\_108\_11\_2 | 2018 | 2018-11-01 | aer | Near-Feasible Stable Matchings with Couples | 108 | 11 | 2 | https://www.aeaweb.org/articles?id=10.1257/aer.20141188 | TRUE | https://www.aeaweb.org/doi/10.1257/aer.20141188.data | 0.07286 | MB | NA | aer\_vol\_108\_issue\_11\_article\_2.zip | NA | TRUE | NA | Abstract The National Resident Matching program seeks a stable matching of medical students to teaching hospitals. With couples, stable matchings need not exist. Nevertheless, for any student preferences, we show that each instance of a matching problem has a “nearby” instance with a stable matching. The nearby instance is obtained by perturbing the capacities of the hospitals. In this perturbation, aggregate capacity is never reduced and can increase by at most four. The capacity of each hospital never changes by more than two. | aer/2018/aer\_108\_11\_2/Readme.pdf |
| aer\_108\_11\_3 | 2018 | 2018-11-01 | aer | The Costs of Patronage: Evidence from the British Empire | 108 | 11 | 3 | https://www.aeaweb.org/articles?id=10.1257/aer.20171339 | TRUE | https://www.aeaweb.org/doi/10.1257/aer.20171339.data | 0.44938 | MB | NA | aer\_vol\_108\_issue\_11\_article\_3.zip | NA | TRUE | NA | Abstract I combine newly digitized personnel and public finance data from the British colonial administration for the period 1854-1966 to study how patronage affects the promotion and incentives of governors. Governors are more likely to be promoted to higher salaried colonies when connected to their superior during the period of patronage. Once allocated, they provide more tax exemptions, raise less revenue, and invest less. The promotion and performance gaps disappear after the abolition of patronage appointments. Patronage therefore distorts the allocation of public sector positions and reduces the incentives of favored bureaucrats to perform. | aer/2018/aer\_108\_11\_3/Readme.pdf |
| aer\_108\_11\_4 | 2018 | 2018-11-01 | aer | The Logic of Insurgent Electoral Violence | 108 | 11 | 4 | https://www.aeaweb.org/articles?id=10.1257/aer.20170416 | TRUE | https://www.aeaweb.org/doi/10.1257/aer.20170416.data | 56 | MB | NA | aer\_vol\_108\_issue\_11\_article\_4.zip | NA | TRUE | NA | Abstract Competitive elections are essential to establishing the political legitimacy of democratizing regimes. We argue that insurgents undermine the state’s mandate through electoral violence. We study insurgent violence during elections using newly declassified microdata on the conflict in Afghanistan. Our data track insurgent activity by hour to within meters of attack locations. Our results  suggest that insurgents carefully calibrate their production of violence during elections to avoid harming civilians. Leveraging a novel instrumental variables approach, we find that violence depresses voting. Collectively, the results suggest insurgents try to depress turnout while avoiding backlash from harming civilians. Counterfactual exercises provide potentially actionable insights  for safeguarding at-risk elections and enhancing electoral legitimacy in emerging democracies. | aer/2018/aer\_108\_11\_4/READ\_ME.pdf |

The table files\_summary contains information about code, data and archive files for each article

dbGet(db, "files\_summary",n = 6)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **id** | **file\_type** | **num\_files** | **mb** | **is\_code** | **is\_data** |
| aejapp\_1\_1\_10 | do | 9 | 0.009427 | TRUE | FALSE |
| aejapp\_1\_1\_10 | dta | 2 | 0.100694 | FALSE | TRUE |
| aejapp\_1\_1\_3 | do | 19 | 0.103628 | TRUE | FALSE |
| aejapp\_1\_1\_4 | csv | 1 | 0.024872 | FALSE | TRUE |
| aejapp\_1\_1\_4 | dat | 1 | 7.15491 | FALSE | TRUE |
| aejapp\_1\_1\_4 | do | 9 | 0.121618 | TRUE | FALSE |

Let us now analyse which share of articles uses Stata, R, Python, Matlab or Julia and how the usage has developed over time.

Since our datasets are small, we can just download the two tables and work with dplyr in memory. Alternatively, you could use some SQL commands or work with dplyr on the database connection.

articles = dbGet(db,"article")

fs = dbGet(db,"files\_summary")

Let us now compute the shares of articles that have one of the file types, we are interested in

# Number of articles with analyes data & code supplementary

n\_art = n\_distinct(fs$id)

# Count articles by file types and compute shares

fs %>% group\_by(file\_type) %>%

summarize(count = n(), share=round((count / n\_art)\*100,2)) %>%

# note that all file extensions are stored in lower case

filter(file\_type %in% c("do","r","py","jl","m")) %>%

arrange(desc(share))

|  |  |  |
| --- | --- | --- |
| **file\_type** | **count** | **share** |
| do | 2606 | 70.55 |
| m | 852 | 23.06 |
| r | 105 | 2.84 |
| py | 32 | 0.87 |
| jl | 2 | 0.05 |

Roughly 70% of the articles have Stata do files and almost a quarter Matlab m files. Using Open Source statistical Software seems not yet very popular among economists, less than 3% of articles have R code files, Python is below 1% and only 2 articles have Julia code.

This dominance of Stata in economics never ceases to surprise me, in particular when for some reason I just happened to open the Stata do file editor and compare it with RStudio… But then, I am not an expert in writing empirical economic research papers – I just like R programming and rather passively consume empirical research. For writing empirical papers it probably *is* convenient that in Stata you can add a robust or robust cluster option to almost every type of regression in order to quickly get the economists’ standard standard errors…

For a teaching empirical economics with R the dominance of Stata is not neccessarily bad news. It means that there are a lot of studies which students can replicate in R. Such replication would be considerably less interesting if the original code of the articles would already be given in R.

Let us finish by having a look at the development over time…

sum\_dat = fs %>%

left\_join(select(articles, year, id), by="id") %>%

group\_by(year) %>%

mutate(n\_art\_year = n()) %>%

group\_by(year, file\_type) %>%

summarize(

count = n(),

share=round((count / first(n\_art\_year))\*100,2)

) %>%

filter(file\_type %in% c("do","r","py","jl","m")) %>%

arrange(year,desc(share))

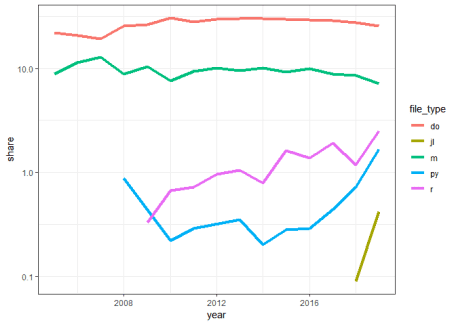
head(sum\_dat)

|  |  |  |  |
| --- | --- | --- | --- |
| **year** | **file\_type** | **count** | **share** |
| 2005 | do | 25 | 22.12 |
| 2005 | m | 10 | 8.85 |
| 2006 | do | 24 | 20.87 |
| 2006 | m | 13 | 11.3 |
| 2007 | do | 24 | 19.35 |
| 2007 | m | 16 | 12.9 |

library(ggplot2)

ggplot(sum\_dat, aes(x=year, y=share, color=file\_type))

+ geom\_line(size=1.5) + scale\_y\_log10() + theme\_bw()



Well, maybe there is a little upward trend for the open source languages, but not too much seems to have happened over time so far…