WRANGL
DATA WI
DUCKDB

Will Angel

2024-10-29

KEY TAKEAWAYS

1. DuckDB is a *very* fast in-process SQL da
2. Duckplyr is a package for using Dplyer
3. You can use Duckplyr as a *drop in repla*
   2-30x speedup for dplyr code on large d
4. Big data is smaller than it used to be.

AGENDA

- What is DuckDB
- Why should you care about DuckDB
- When should you use DuckDB
- How can you use DuckDB
- What is Duckplyr
- Data processing performance and pro
- A brief tangent on the shrinking of big

DUCKDB

Duc

WHAT IS DUCKDB?

DuckDB is an open source fast in-process analytical databa

- Open Source: Free & Open
- Fast: Performant. **Quickly and efficiently runs analyti**
- In-process: **Runs locally** without a server
- Analytical: DuckDB is optimized for **aggregations** and a
  support online analytical processing (OLAP). DuckDB su
  but is not as fast for online transaction processing (OLTP
- Database: DuckDB can be used to efficiently store relati

WHY SHOULD YOU CAR

- DuckDB is a great tool for local SQL an
  and you can do SQL locally!
- DuckDB is starting to power the next g
  embedded analytical tools, so expect b
  data filtering tools to get more power

WHY SHOULD YOU CAR
TECHNICAL)

- DuckDB is like SQLite but for data proc
  crunch significant amounts of data loc
  spinning up a full database / data ware
  which may create significant time/cost
  simplify system design
- DuckDB is versatile and fast for data p
  Competitive with spark/polars in bench
- DuckDB is portable with zero depende

WHEN SHOULD YOU US

- You should use DuckDB for SQL data p
- You have medium-largish data
- You don’t want the hassle of procuring resources.

HOW CAN YOU USE DU

- Directly use the DuckDB package
- use DBplyr to connect to a local DuckD
- use Duckplyr!

WHAT IS DUCKPLYR

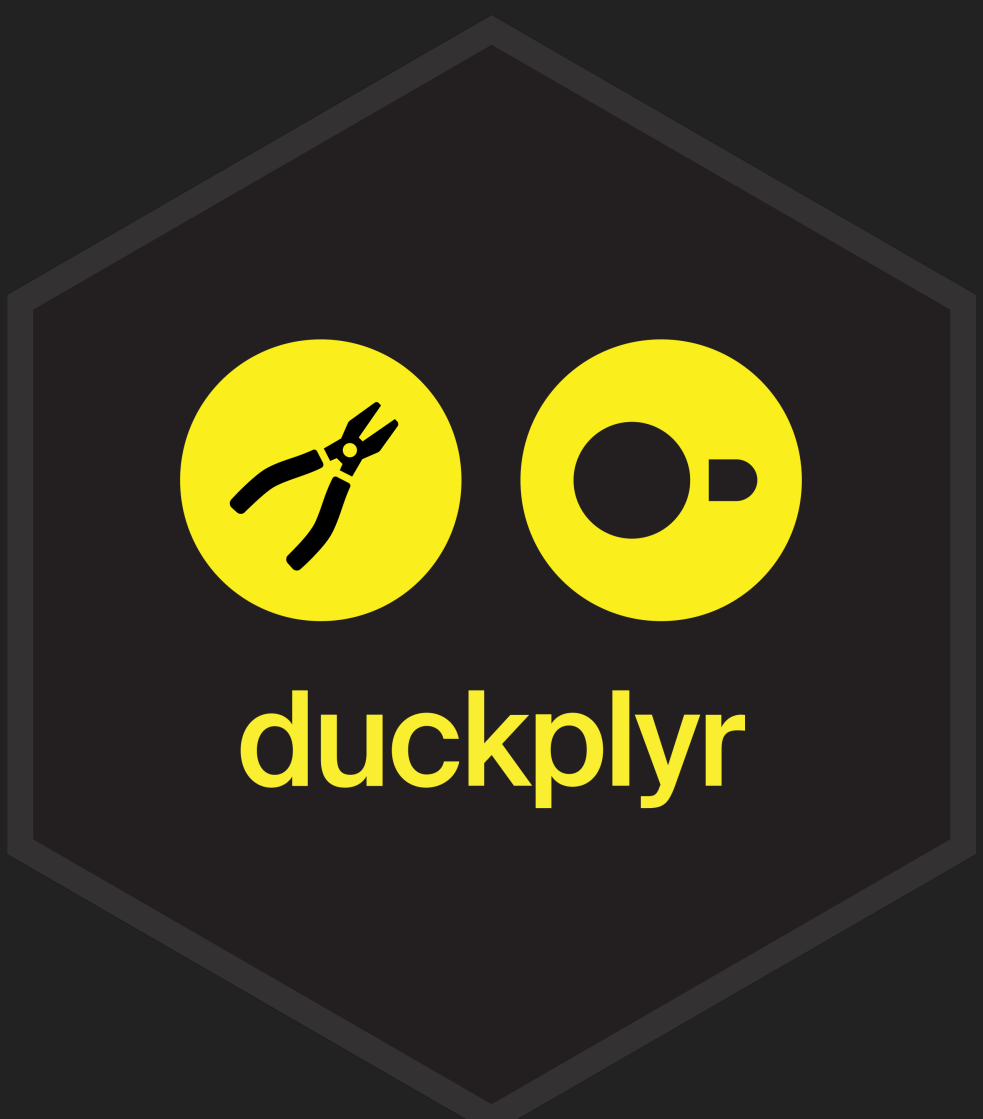
- Duckplyr is a drop in replacement for D
- Duckplyr uses DuckDB's "relational" AP
  and directly construct logical query pla
- This means you can speed up your Dply
  by simply changing the package!
- Unsupported operations will fall back t
  code will always run!
- Duckplyr *overwrites* Dplyr methods, so
  loading the package.

```
library("duckplyr")
```

DUCKPLYR CAVEATS

- Unsupported operations will fall back t
  code will always run!
  - This may mean you don’t get a perfo
  - If you’re doing out of memory proce
    cause issues
- Duckplyr is still under active developm
  some surprising unsupported operatio
  - “message”:“No relational implement
    group_by().”

BENCHMARKING & SPE



DPLYR VERSUS DUCKPL

To compare Dplyr and Duckplyr performa
the Global Lake area, Climate, and Popul
(GLCP). This includes almost 80 million re
temperature data for lakes, for a 3.9GB C

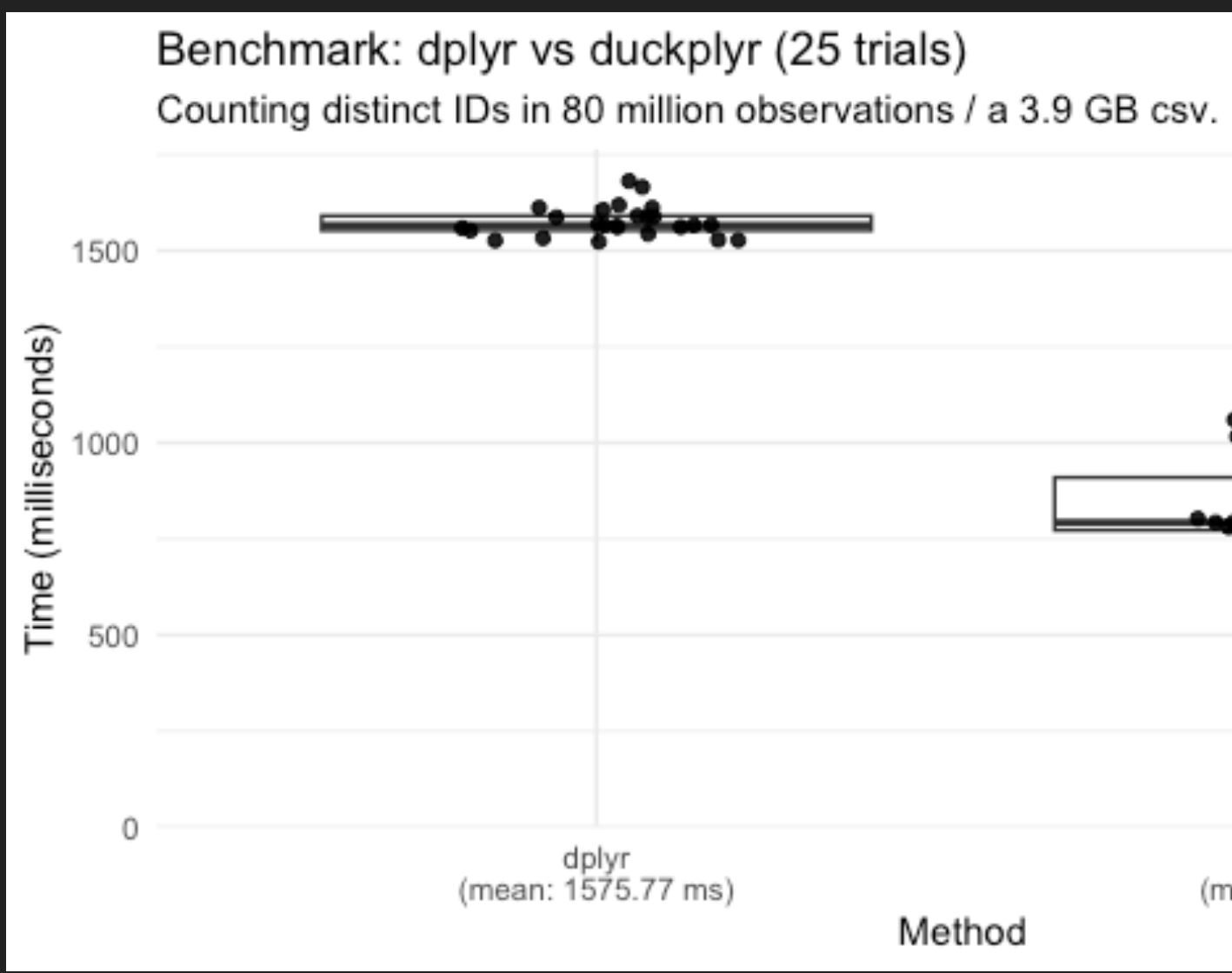
Dplyr:

```
data |>
  distinct(HYBAS_ID) |>
  nrow()
```

Duckplyr:

```
data |>
  duckplyr::as_duckplyr_tibble() |>
  distinct(HYBAS_ID) |>
  nrow()
```

DPLYR VERSUS DUCKPL



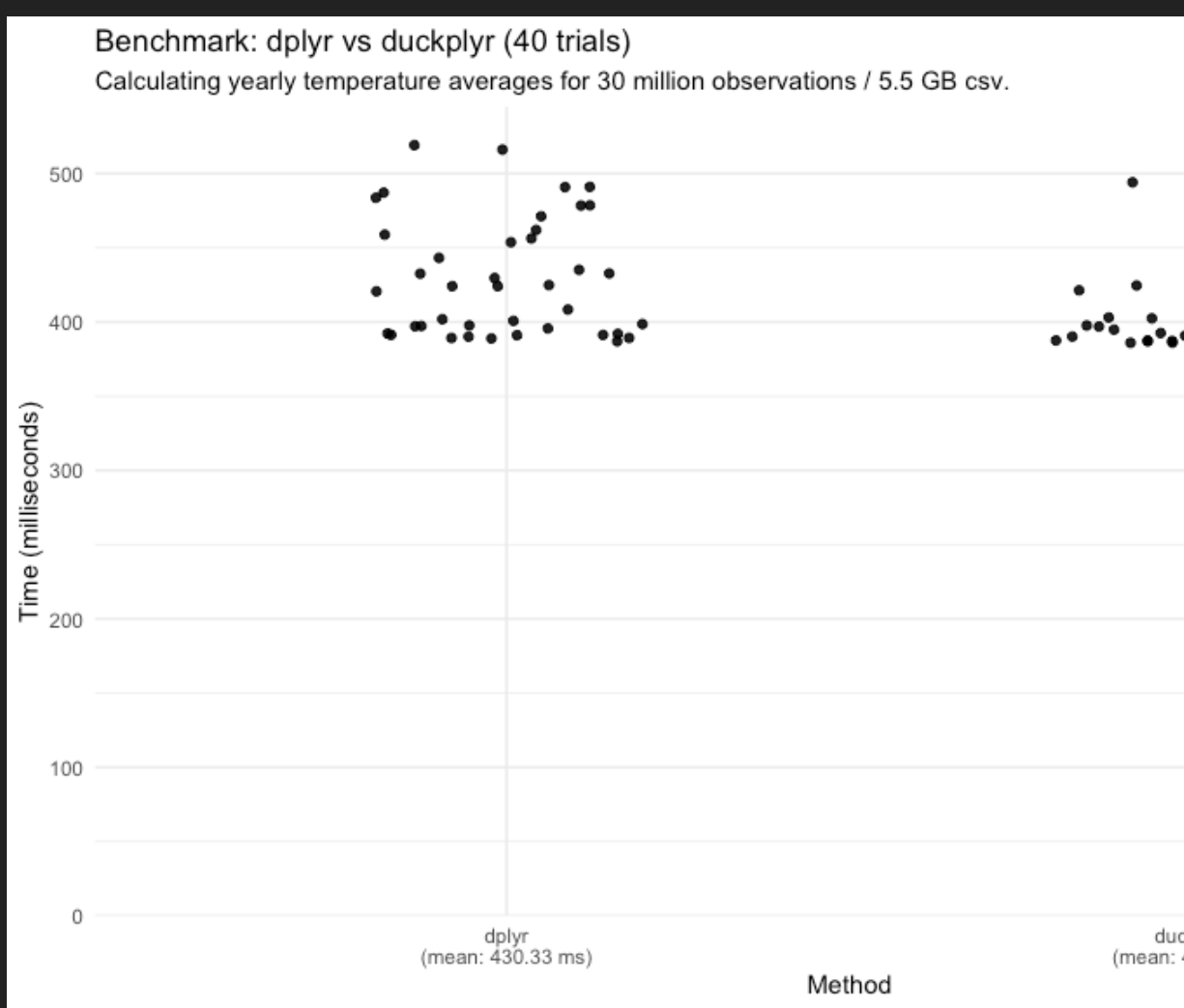
86% speed increase counting distinct IDs in 80 million row

DPLYR VERSUS DUCKPL

Now we'll take a look at the overall GLCP
million observations in a 5.5GB CSV file.

```
data |>
  summarise(
    avg_size = mean(total_km2), n = n(), .by = cou

library(duckplyr)

data |>
  summarise(
    avg_size = mean(total_km2), n = n(), .by = cou
```

DPLYR VERSUS DUCKPL



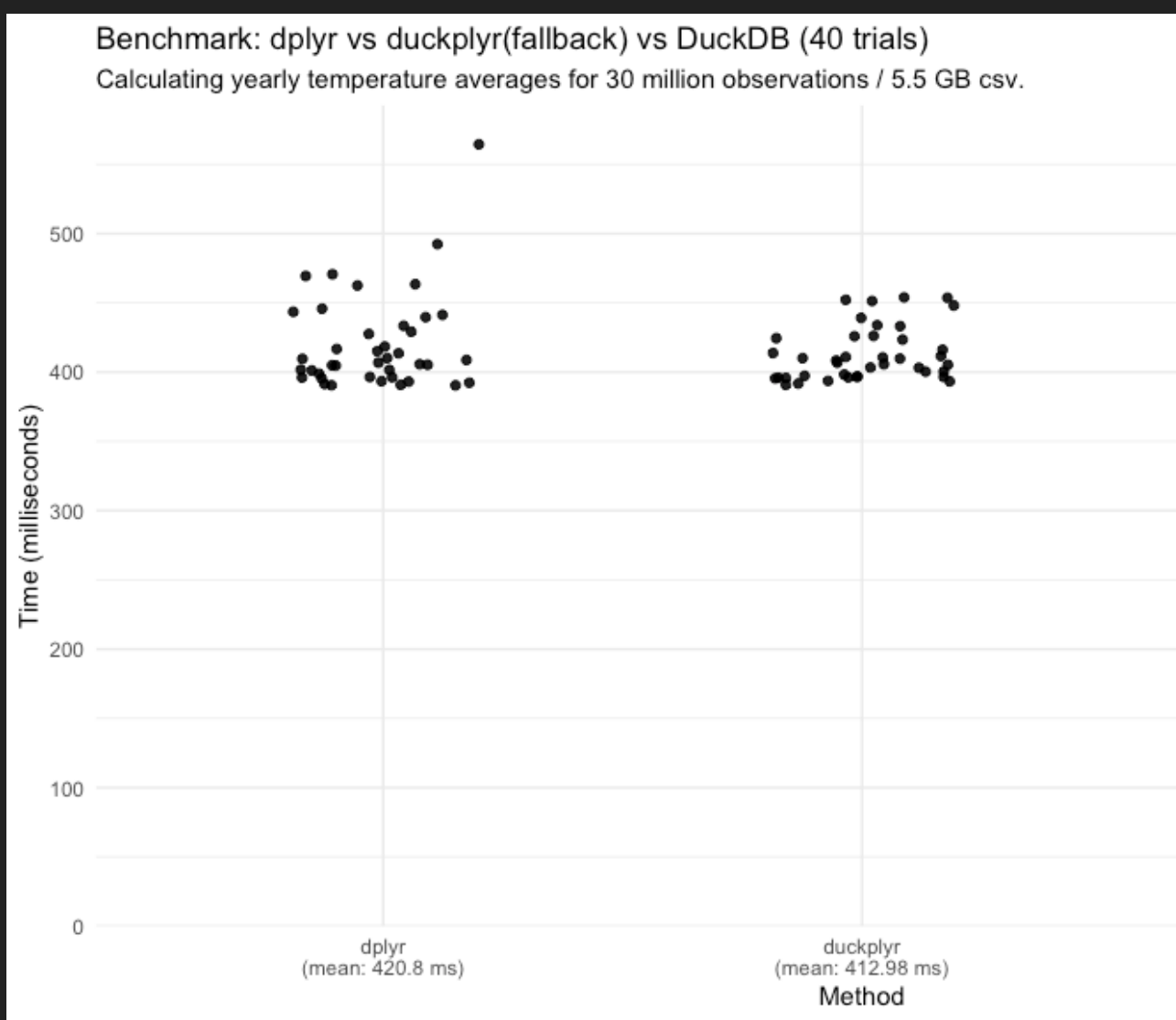
We ran into a limitation of Duckplyr and fell back to dplyr.
“{”version”:“0.4.1”,“message”:“No relational implementatio

DPLYR VERSUS ~~DUCKPL~~ DUCKDB

Let’s switch to DuckDB and write some S

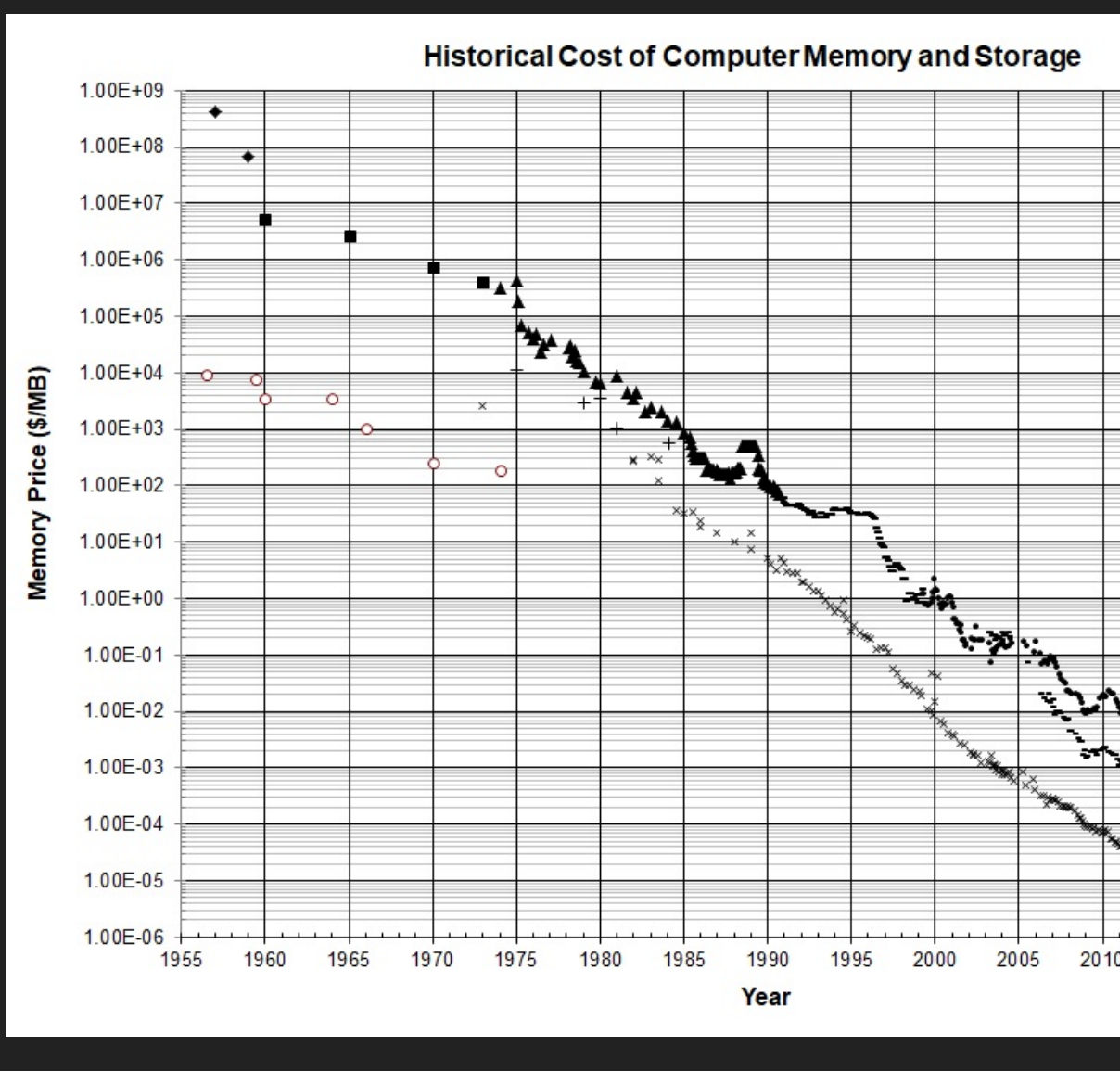
```
library(duckdb)

con <- dbConnect(duckdb())
duckdb_register(con, "data", data)


dbGetQuery(con, "
      SELECT
        country,
        AVG(total_km2) as avg_size,
        COUNT(*) as n
      FROM data
      GROUP BY country
    ")
```

DPLYR VERSUS DUCKDB

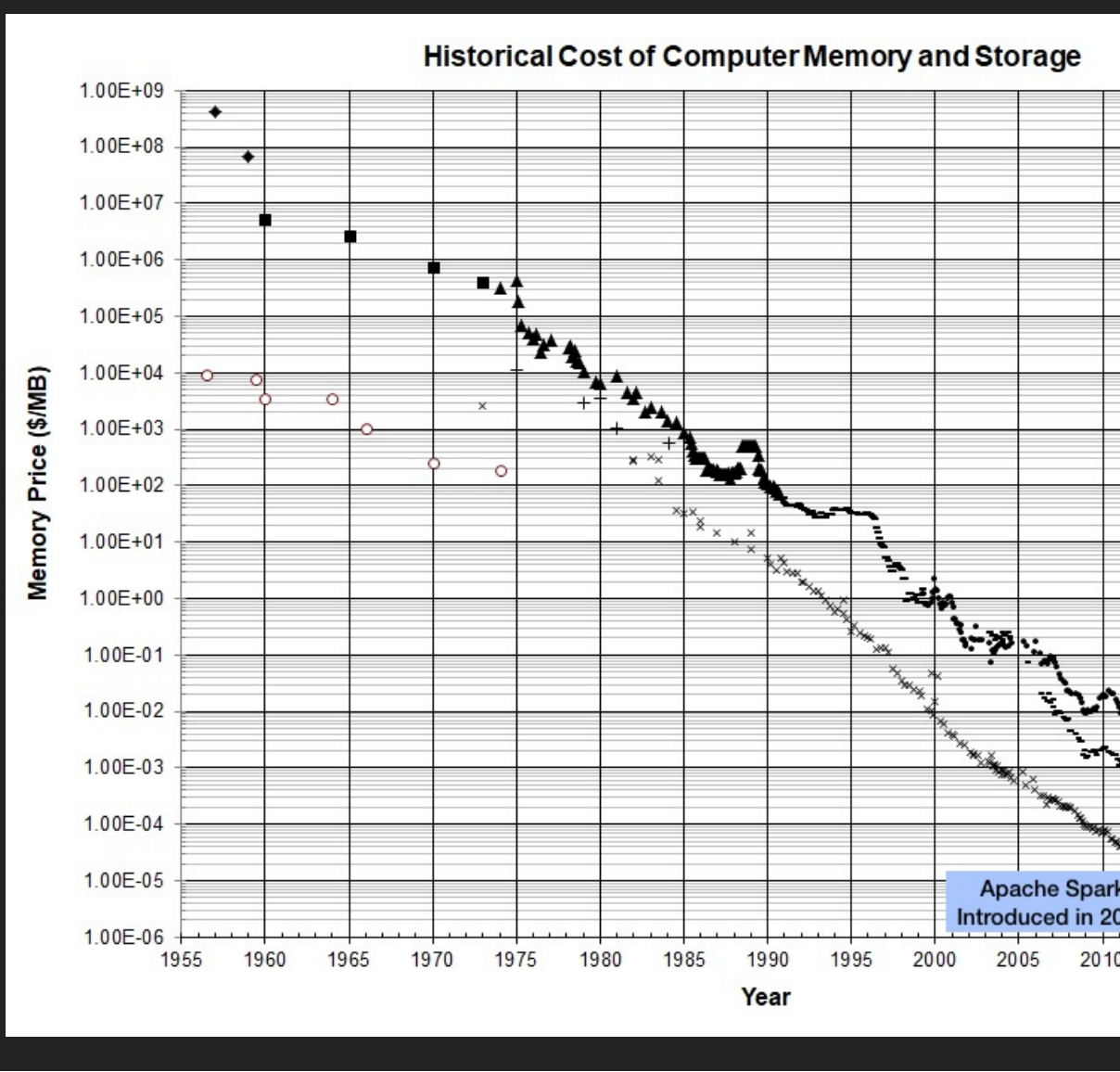


BIG DATA SMALLER THAN IT U TO BE

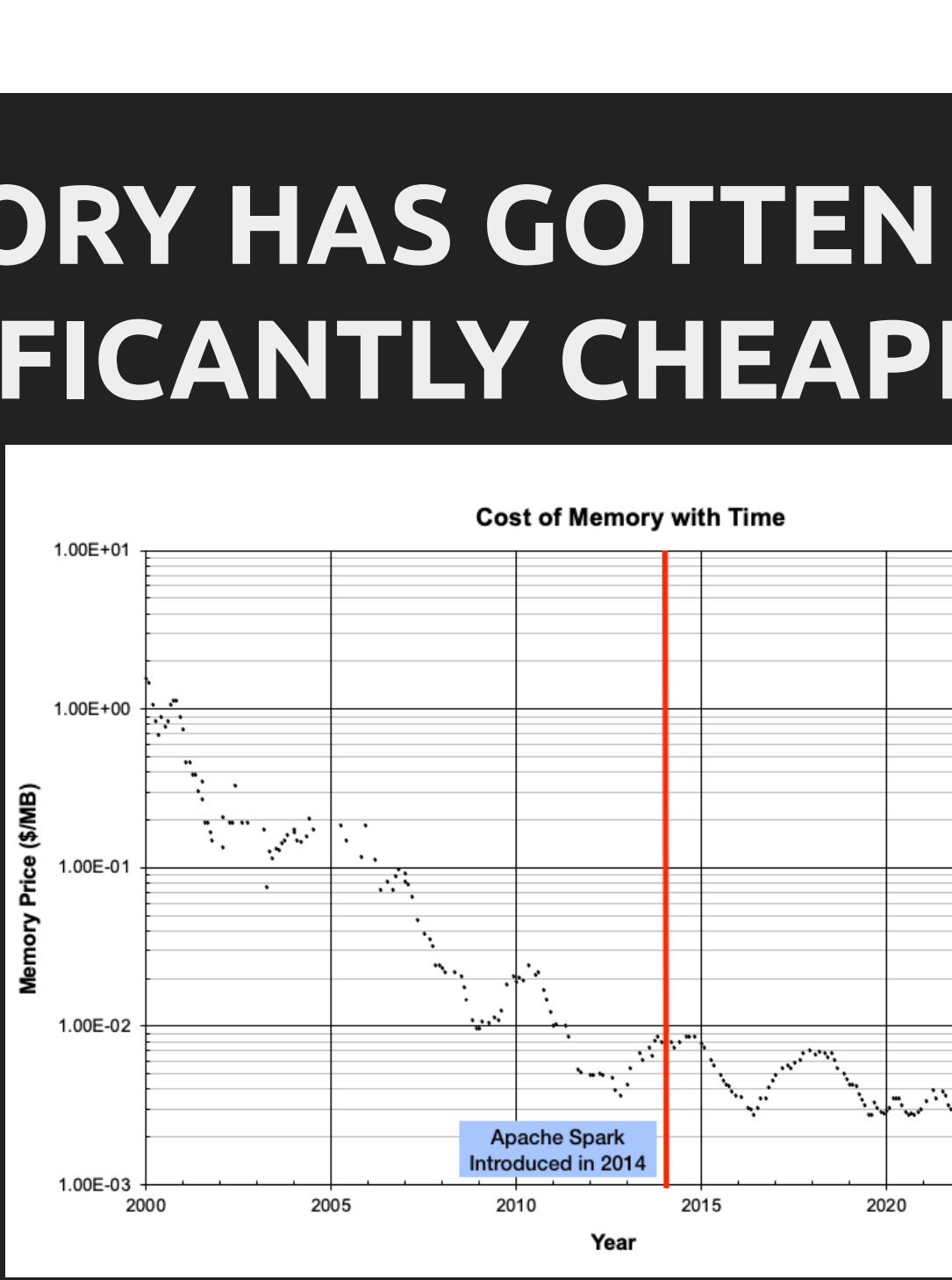
COMPUTERS ARE GETT BIGGER



COMPUTERS ARE GETT BIGGER

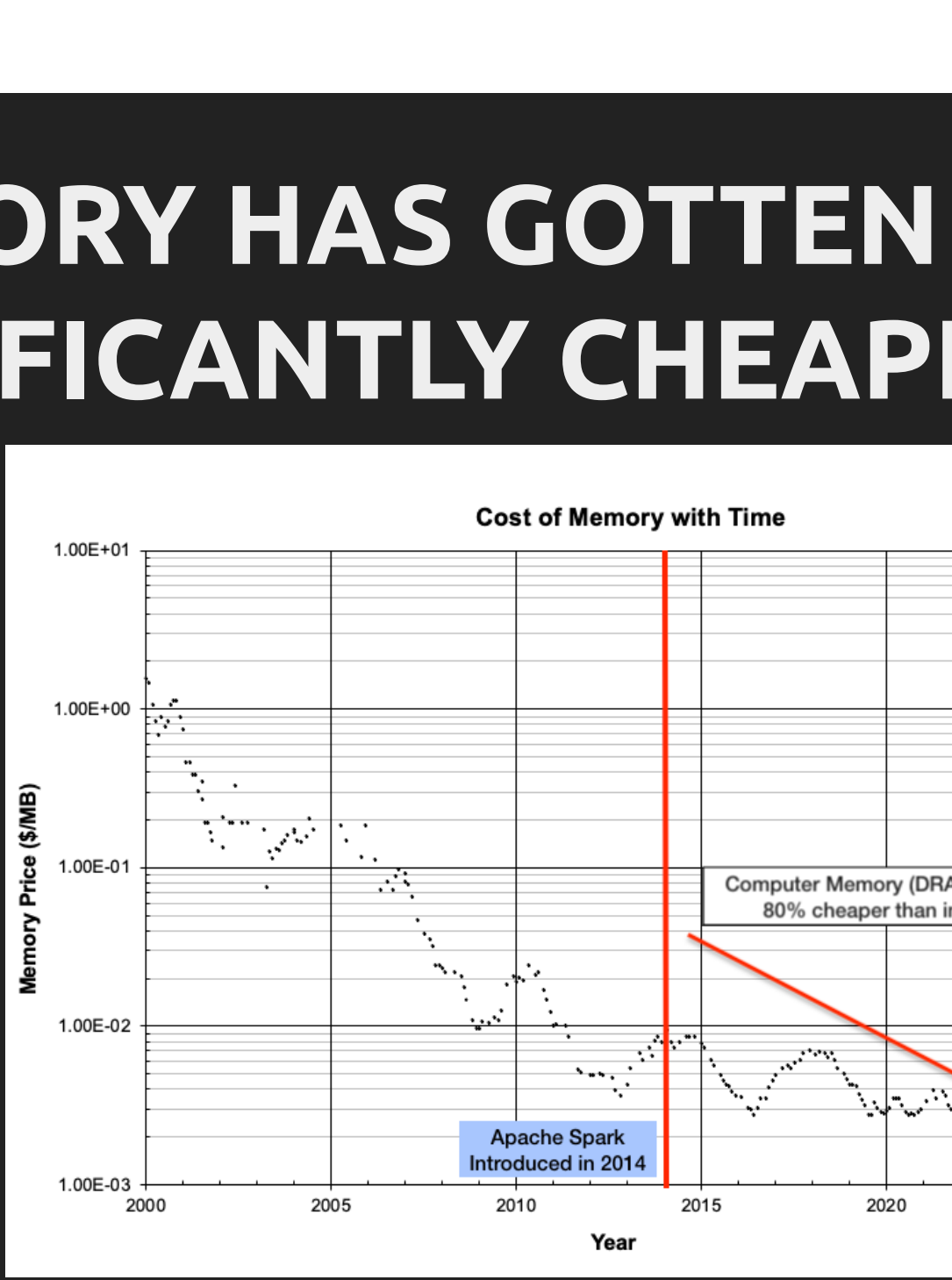


MEMORY HAS GOTTEN
SIGNIFICANTLY CHEAPE



https://jcmit.net/memoryprice.htm

MEMORY HAS GOTTEN
SIGNIFICANTLY CHEAPE



https://jcmit.net/memoryprice.htm

IN CONCLUSION:

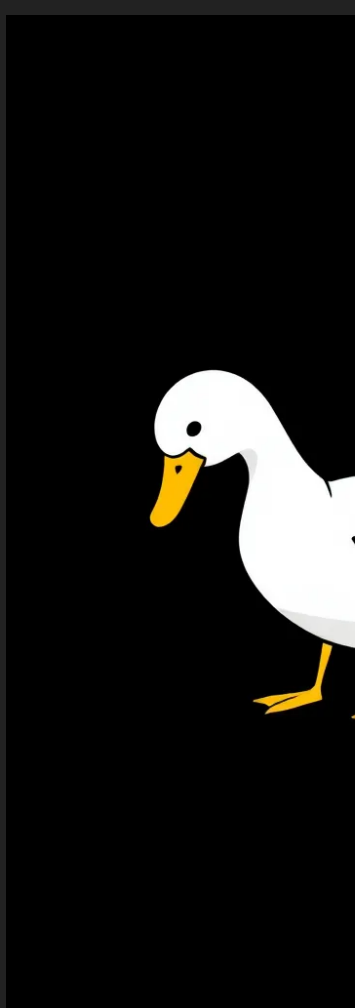
BIG DATA IS SMALLER THAN IT USED

With more access to cheaper computing
tools (DuckDB/Polars(TidyPolars)/Data.Ta
to process data more efficiently:

1. We'll end up with more "Annoyingly M
   big to open in Excel, fits in RAM on a d
2. We can defer investing in large scale d
   like Spark more often, and wait longer
   code.

THANK YOU!

- /in/william-angel/
- @datadrivenangel
- www.williamangel.net
- yt@williamangel.net



AI ducks to