Expressing yourself in R

Hadley Wickham

@hadleywickham

Chief Scientist, RStudio

February 2014

Data analysis is the process by which data becomes understanding, knowledge and insight

Data analysis is the process by which data becomes understanding, knowledge and insight

Data analysis is the process by which data becomes understanding, knowledge and insight

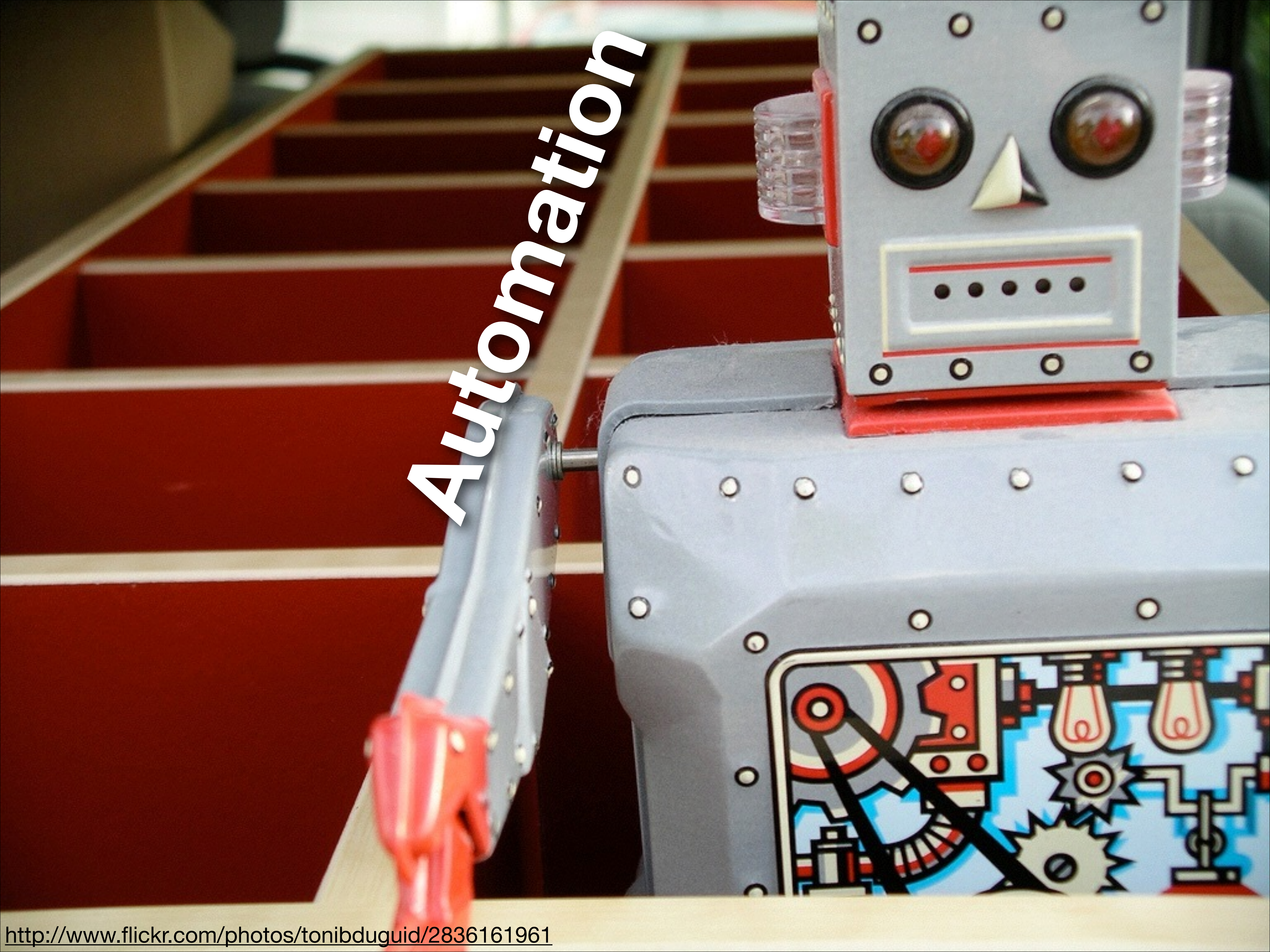
1. Why program?
2. Why R?
3. Data manipulation with dplyr
4. Data visualisation with ggvis

Why program?

Reproducibility

http://www.flickr.com/photos/tonibduguid/2836161961/sizes/l/

Automation

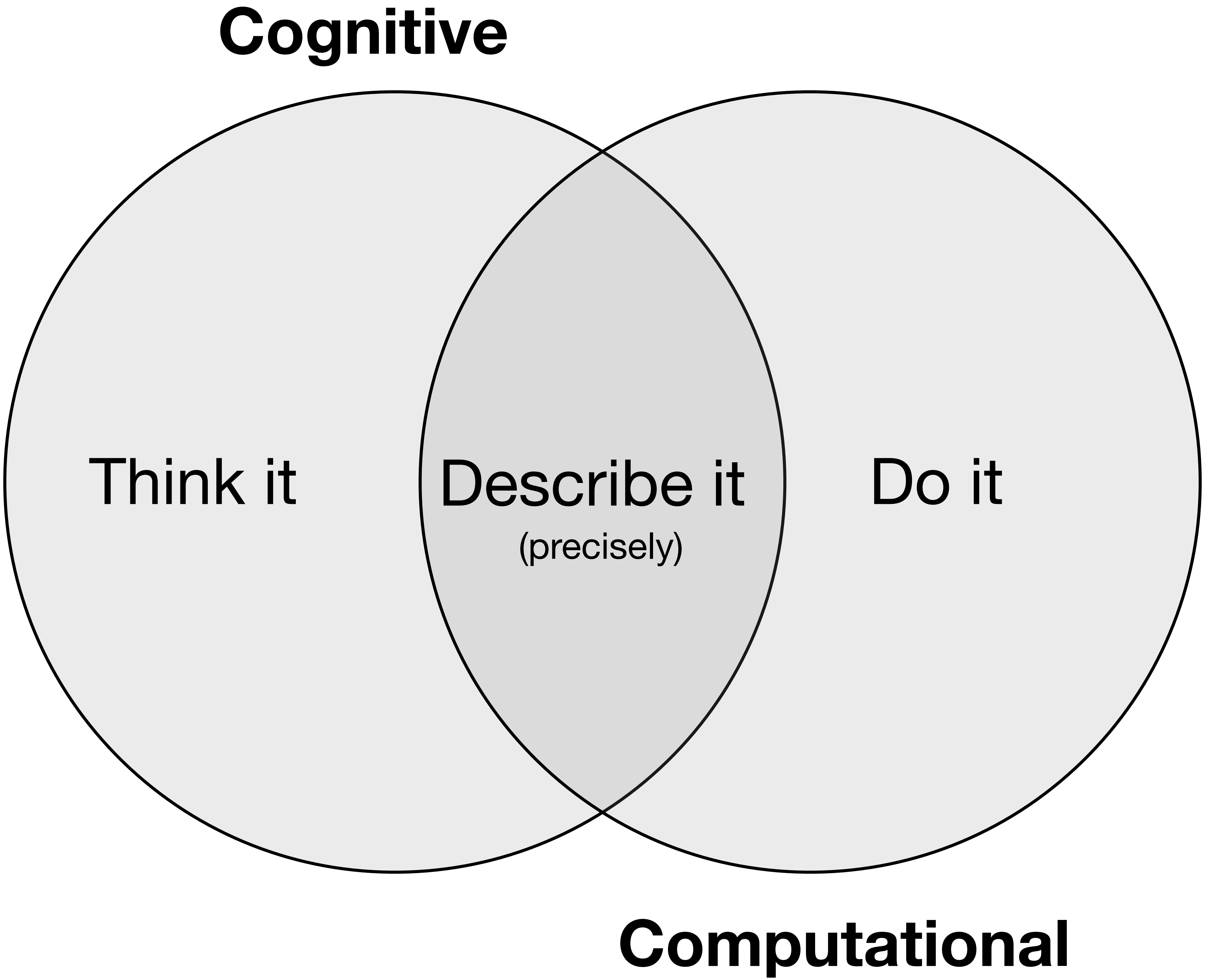


http://www.flickr.com/photos/tonibduguid/2836161961

Communication

http://www.flickr.com/photos/altemark/337248947/sizes/l/

Why R?



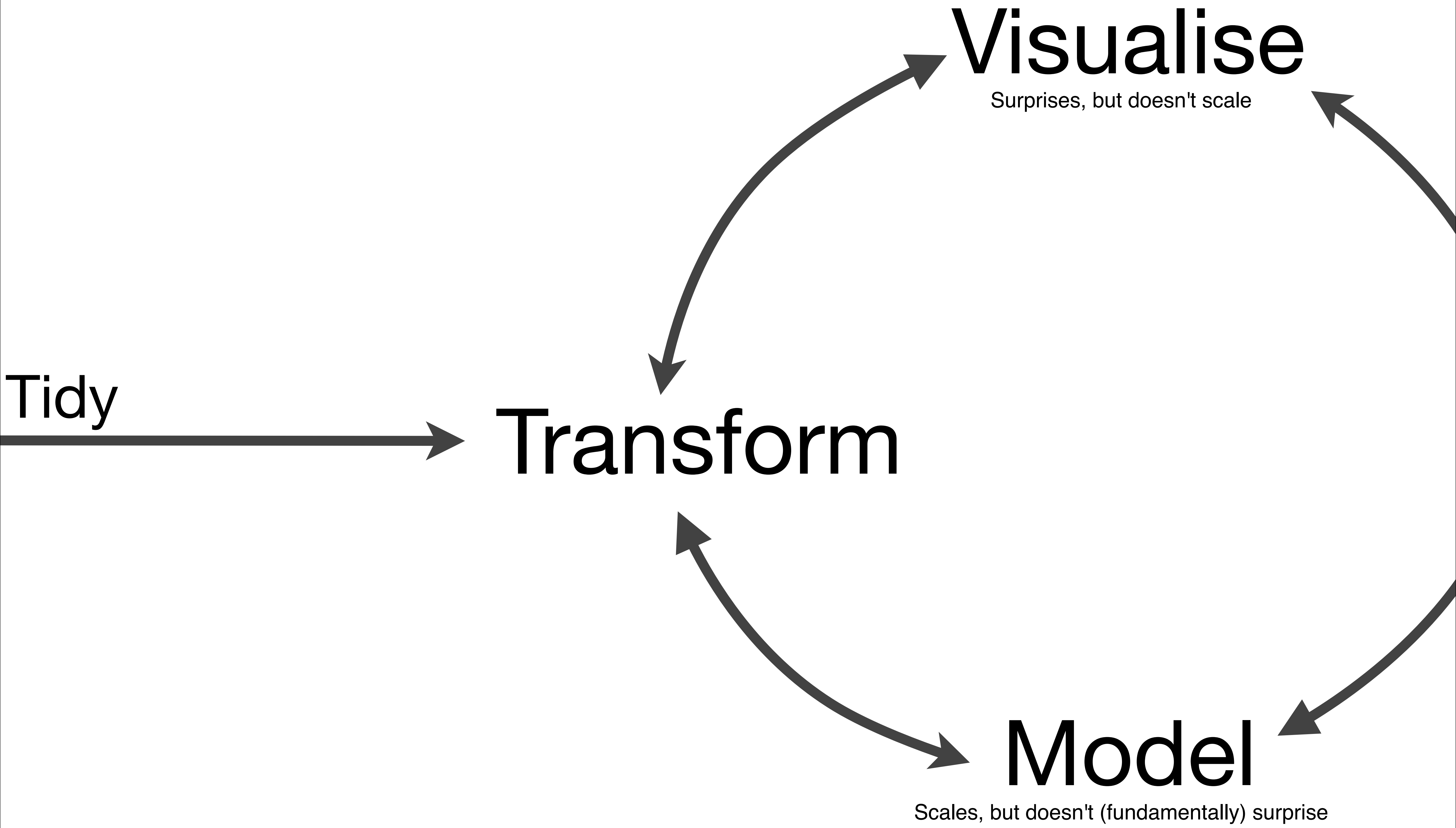
Cognitive

Think it

Describe it
(precisely)

Do it

Computational



Visualise

Surprises, but doesn't scale

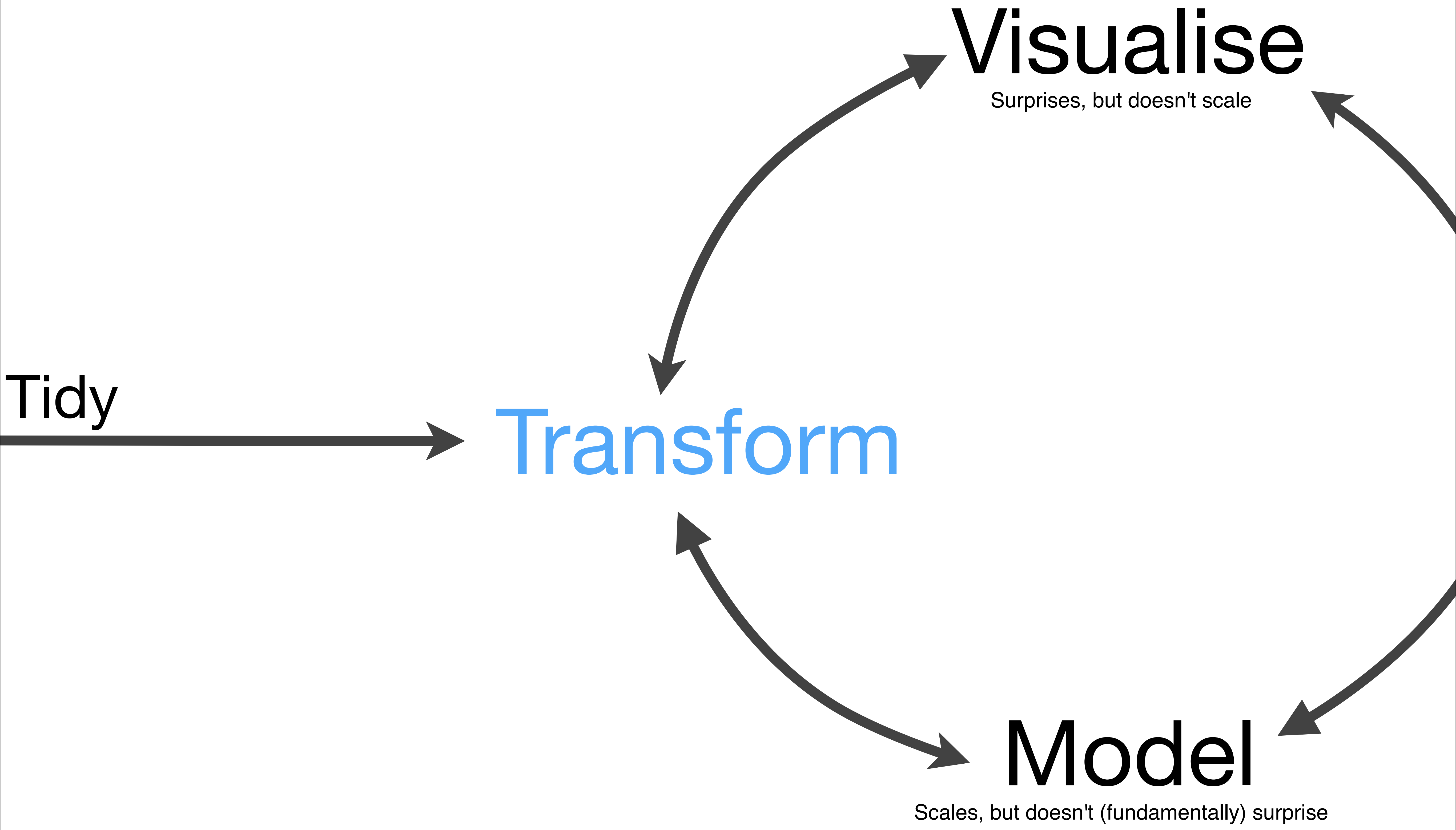
Tidy

Transform

Model

Scales, but doesn't (fundamentally) surprise

dplyr



Visualise

Surprises, but doesn't scale

Tidy

Transform

Model

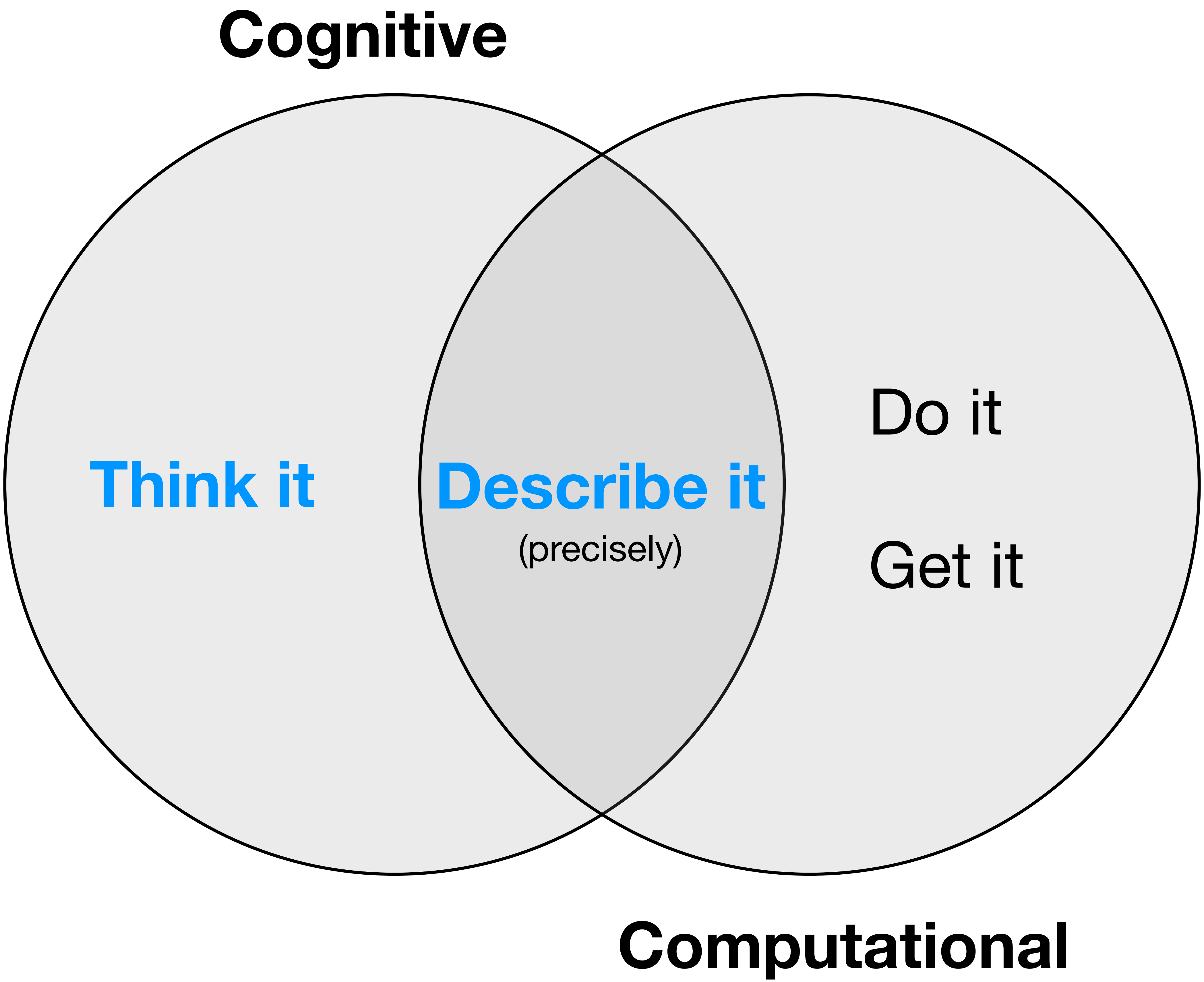
Scales, but doesn't (fundamentally) surprise

```
library(dplyr)
logs <- readRDS("logs.rds") # http://cran-logs.rstudio.com/

print(logs)
#> Source: local data frame [23,454,437 x 10]
#>
#>          date     time    si                r_arch        r_os   package
#> 1  2013-01-01 00:18:22  551371     2.15.2 x86_64 darwin9.8.0     knitr
#> 2  2013-01-01 00:43:47  220277     2.15.2 x86_64     mingw32 R.devices
#> 3  2013-01-01 00:43:51 3505851     2.15.2 x86_64     mingw32     PSCBS
#> 4  2013-01-01 00:43:53  761107     2.15.2 x86_64     mingw32      R.oo
#> 5  2013-01-01 00:31:15  187381     2.15.2   i686   linux-gnu     akima
#> 6  2013-01-01 00:59:46 2388932     2.15.2 x86_64     mingw32 spacetime
#> 7  2013-01-01 00:31:31   34662     2.15.1 x86_64   linux-gnu    mnormt
#>                    00:30:55  873639     2.15.2 x86_64     mingw32      MASS
#>                    00:43:26  607000         NA     NA          NA     tsDyn
#> 10 2013-01-01 00:19:25  402583     2.15.2 x86_64 darwin9.8.0   mvtnorm
#>..        ...      ...     ...        ...    ...         ...       ...
#> Variables not shown: version (chr), country (chr), ip_id (int)

print(object.size(logs), units = "GB")
#> 1.6 Gb
```

Commas helpful

No, I don't want to see 10,000 rows!

Not “big” data, but still big



Cognitive

Think it

Describe it

(precisely)

Do it

Get it

Computational

Key insight

There are only a few data analysis verbs **and** they’re the same regardless of where your data lives

Single table verbs

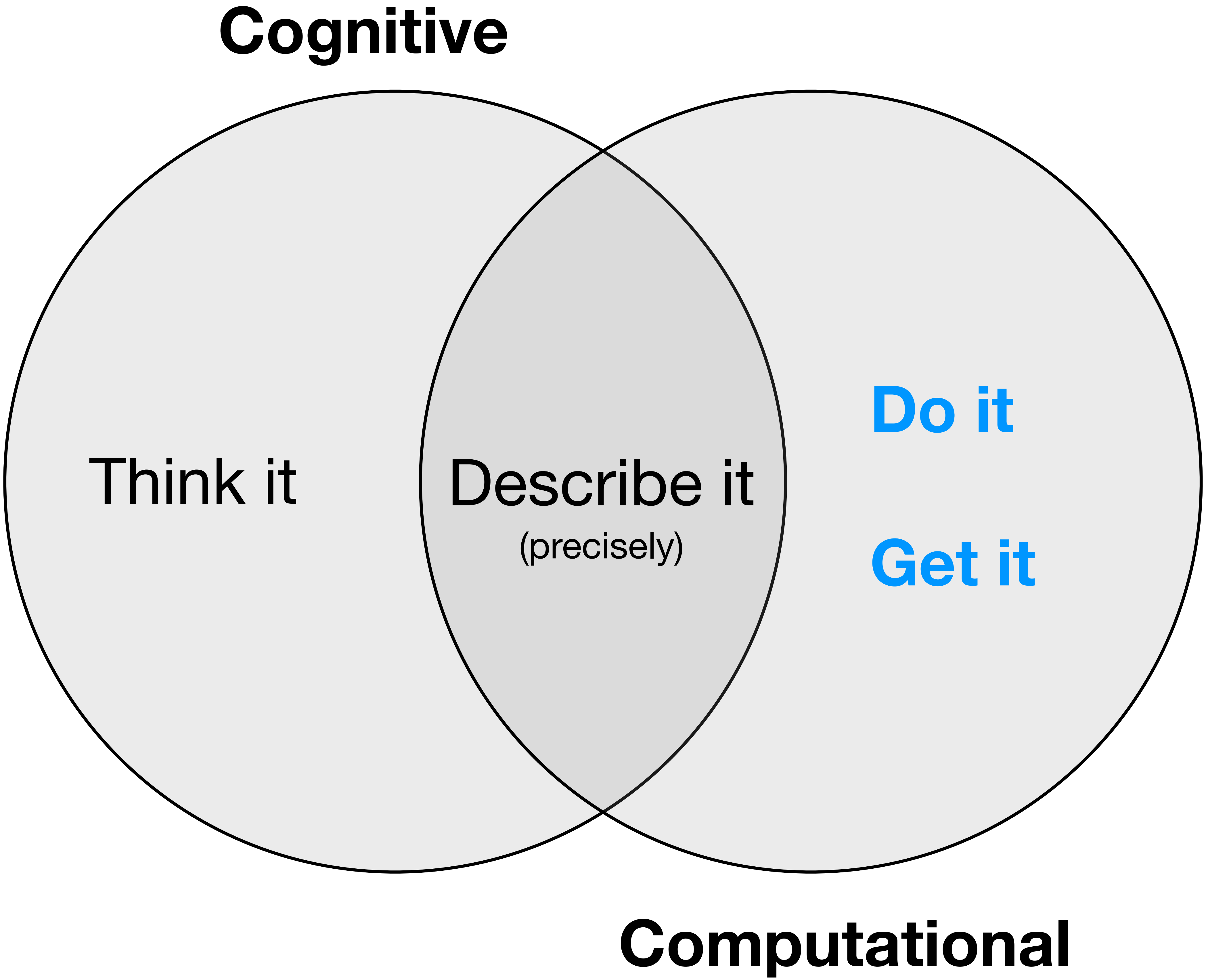
+ group by

- **select**: subset variables
- **filter**: subset rows
- **mutate**: add new columns
- **summarise**: reduce to a single row
- **arrange**: re-order the rows

```r
# What packages are most downloaded
packages <- group_by(logs, package)
counts <- summarise(packages, n = n())
head(arrange(counts, desc(n)), 20)

# Takes ~2s (mostly to build index)
# All functions are pure (no side-effects) -> easy to
# reason about. But function composition is hard to read.
# Solution: x %.% f(y) -> f(x, y)

logs %.%
  group_by(package) %.%
  summarise(n = n()) %.%
  arrange(desc(n)) %.%
  head(20)
```

Multi-table verbs

- **left join**: all x + matching y
- **inner join**: matching x + y
- **semi join**: all x with match in y
- **anti join**: all x without match in y



Cognitive

Think it

Describe it

(precisely)

Do it

Get it

Computational

Local data frames

- High-performance C++. Avoid copies. Avoid R function call overhead with custom interpreter for simple R expressions.
- Thanks to Romain Francois
- (Currently working on automatic parallelisation)

Key insight

Move the computation to the data

dplyr sources

- Local data frame
- Local data table
- Local data cube (experimental)
- RDMS: Postgres, MySQL, SQLite, Oracle, MS SQL
- BigQuery

Translate R to SQL

High-level data manip verbs correspond to high-level component of SQL grammar.

Automatically translate small expressions from R to SQL.

Translation can’t be perfect; aiming for semantic equivalency.

```
hflights <- hflights_postgres("hflights")
hflights <- hflights_postgres() %.% tbl("hflights")
ranked <- hflights %.%
  group_by(TailNum) %.%
  mutate(Rank = rank(desc(ArrDelay))) %.%
  select(TailNum, ArrDelay, Rank)

ranked$query
# SELECT
#   *,
#   RANK() OVER (PARTITION BY "TailNum"
#     ORDER BY "ArrDelay" DESC) AS "rank"
#  FROM "hflights"
```

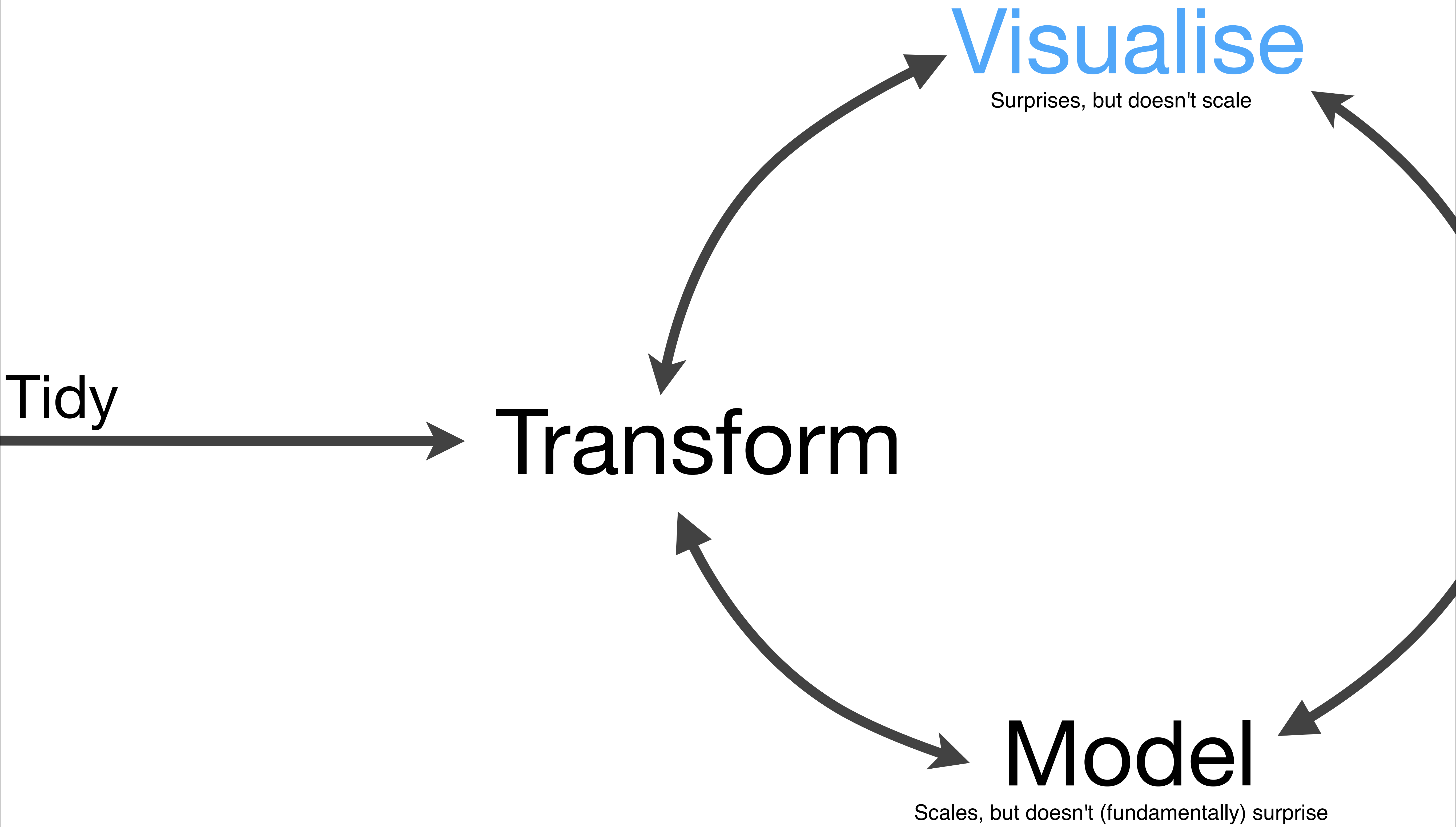
```
worst <- hflights %.%
  group_by(TailNum) %.%
  filter(ArrDelay == max(ArrDelay)) %.%
  select(TailNum, ArrDelay)

worst$query
# SELECT ”TailNum”, ”ArrDelay”
# FROM (
#   SELECT ”TailNum”, ”ArrDelay”, max(”ArrDelay”)
#      OVER (PARTITION BY ”TailNum”) AS ”_W5”
#   FROM ”hflights”
# ) AS ”_W6”
# WHERE ”ArrDelay” = ”_W5”
```

Google for

“dplyr”

ggvis

with Winston Chang



Visualise

Surprises, but doesn't scale

Tidy

Transform

Model

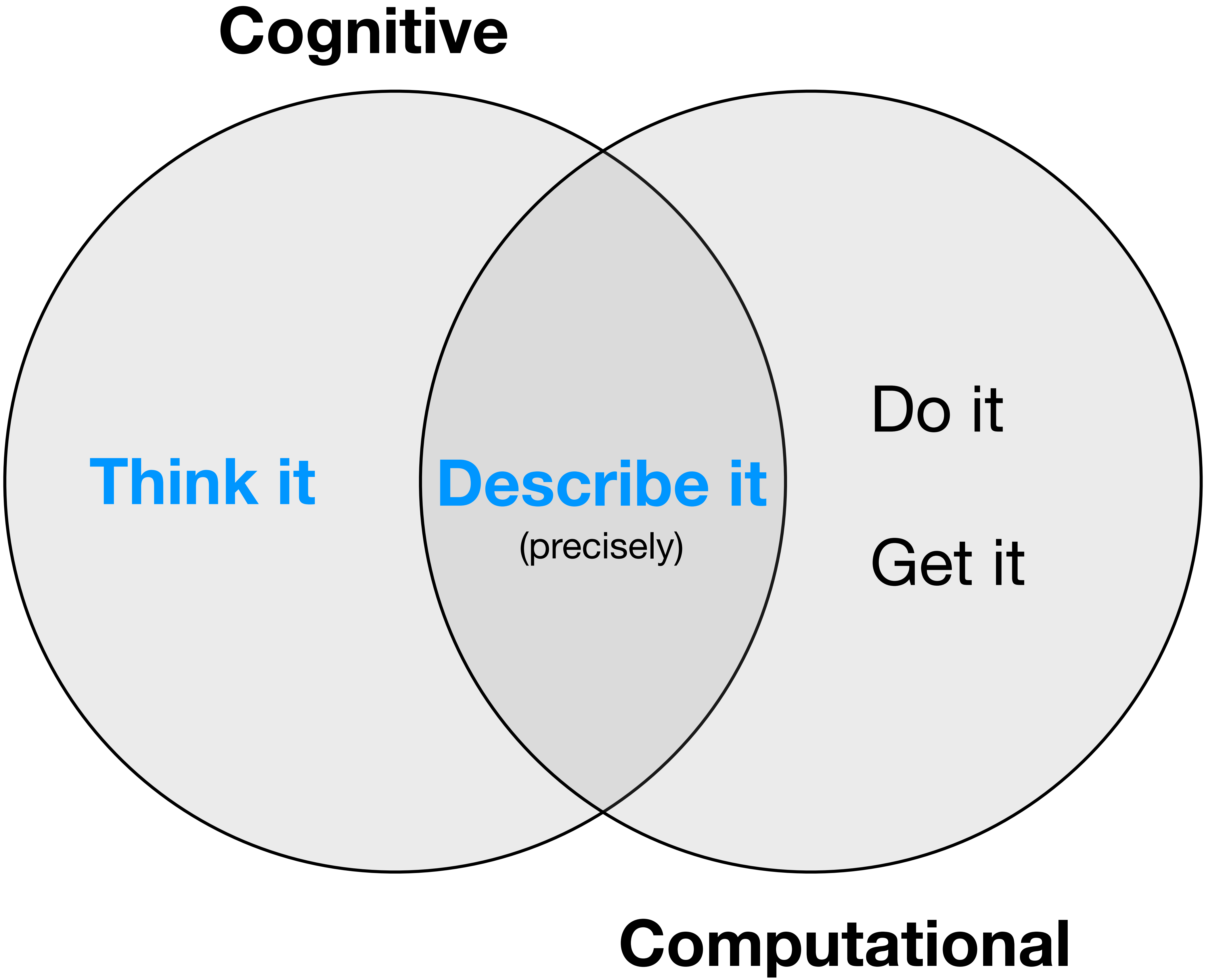
Scales, but doesn't (fundamentally) surprise

Goals

Describe visualisations declaratively
(à la ggplot2).

Graphics not just **on** the web,
but **of** the web.

Built out of reactive components
(interactive and dynamic).



Cognitive

Think it

Describe it

(precisely)

Do it

Get it

Computational

Demo

Google for

“ggvis”

Conclusions

Bottlenecks

Biggest bottleneck in exploration is cognitive.

Need tools that help you define the problem and express solutions programmatically.

R makes it easy to create DSLs for parts of the data analysis process.

Office hour

Thursday 1:40pm • Table A

Google for

“dplyr”,“ggvis”

http://bit.ly/expressive-da2