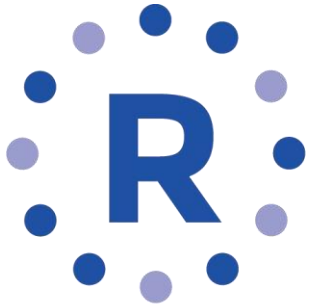


consortium

Part of the new
R/Insurance
Webinar Series

R performance culture

24 January 2024

Welcome to the webinar!

R/insurance webinar series

1) From Excel to programming in R

2) From programming in R to putting R into production

3) R performance culture

4) High performance programming in R

Delivered on behalf of the R Consortium by Georgios Bakoloukas and Benedikt Schamberger, Actuarial Control, Group Risk Management, Swiss Re

Background to Swiss Re's R community

Large actuarial R programming, Atelier, community

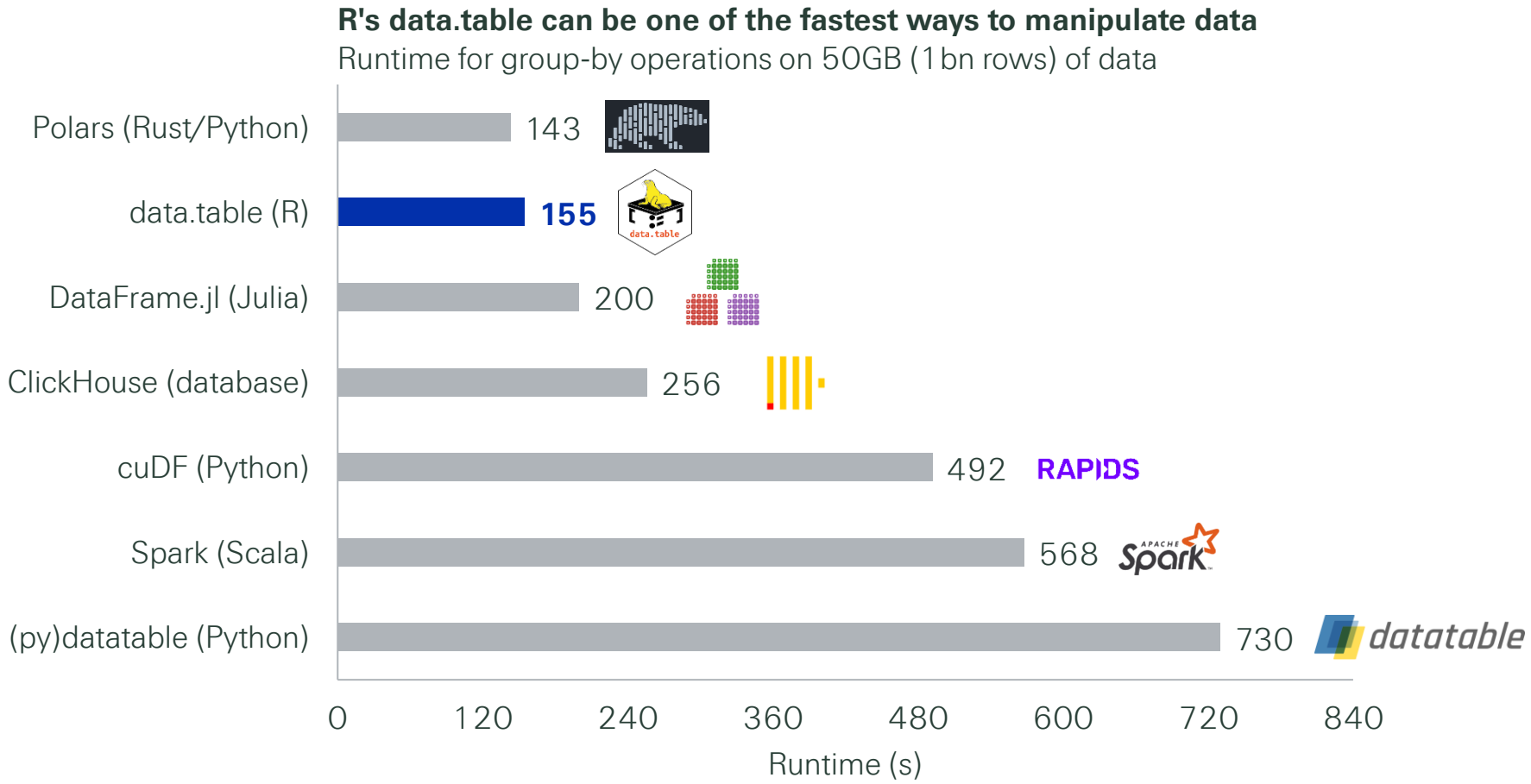
- Swiss Re internal R community sponsored by our Group Chief Actuary Philip Long (Atelier programme)
- 2000+ community with 500+ regular coders who also support each other
- The case we see today relates to code optimisations we did for experience study work, ie comparing how an insurance portfolio performed to initial expectations
- Views expressed belong solely to the speakers and not necessarily to the speaker's employer

We should forget about small efficiencies, say about 97% of the time: **premature optimization is the root of all evil**. Yet we should not pass up our opportunities in that critical 3%.

Donald Knuth
Author of *The Art of Computer Programming,*
creator of TeX, ACM Turing Award recipient

R is designed for flexibility, but can have high performance

R's data.table can be one of the fastest ways to manipulate data

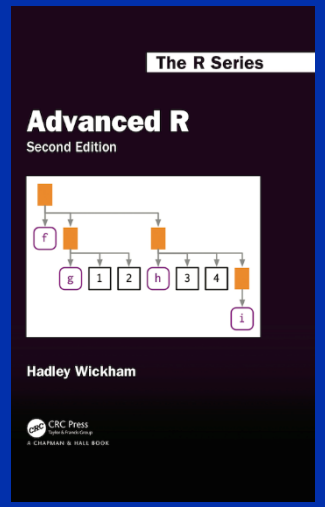


Database-like ops benchmark developed by Matt Dowle. More information at https://h2oai.github.io/db-benchmark/. Results as of middle 2021.

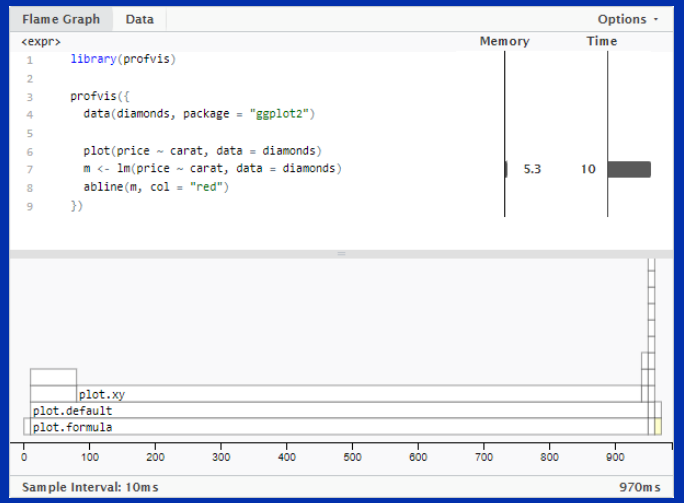
The R ecosystem provides guidance and tools to tune performance

There are tools ranging from general guidelines, to scripts and single lines of code

Performance tips in Advanced R



Code profiling with profvis



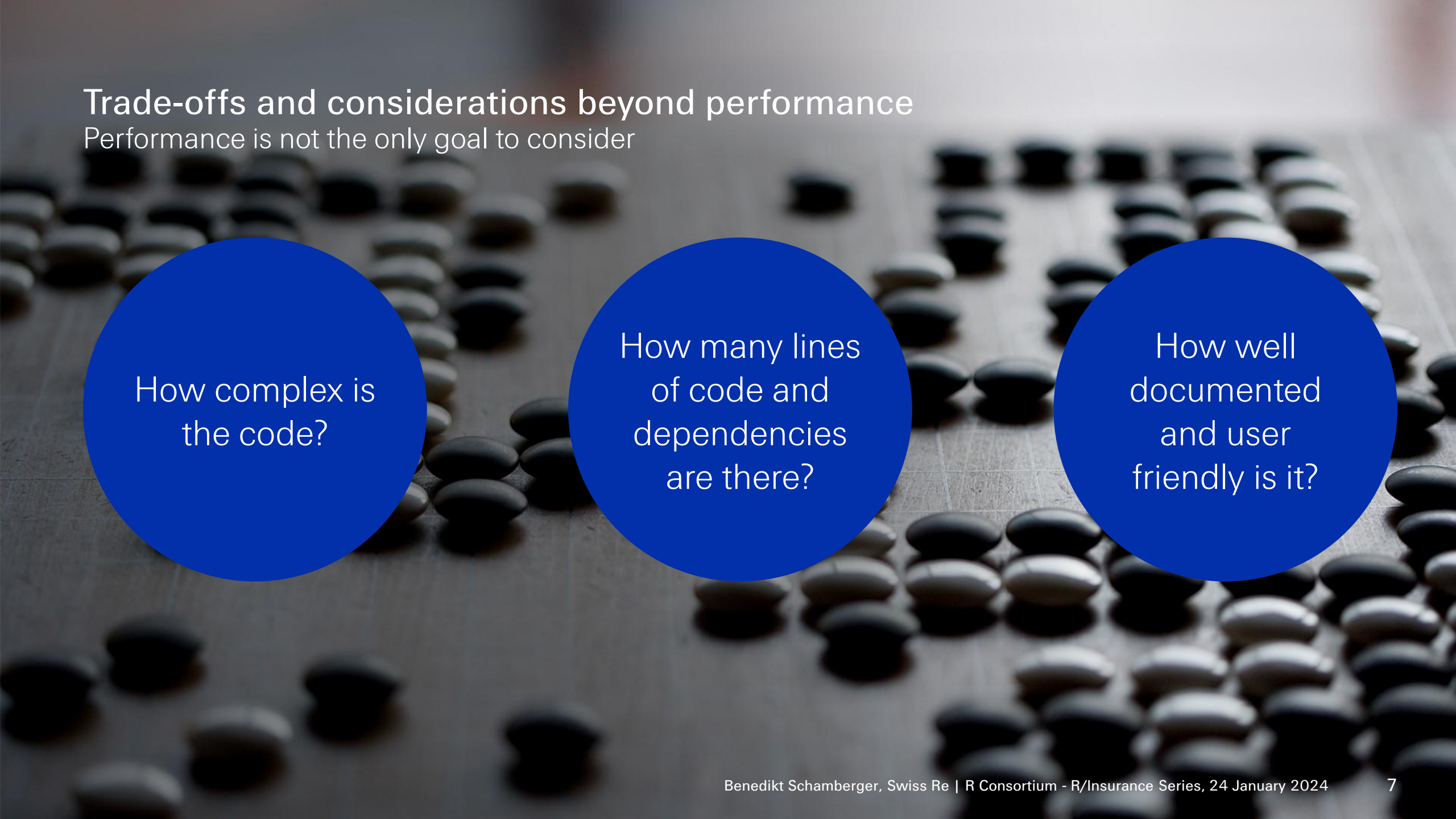
Compare alternatives with bench

```
library(bench)
set.seed(42)
dat <- data.frame(x = runif(10000, 1, 1000), y=runif(10000, 1, 1000))

bnch <- bench::mark(
  dat[dat$x > 500, ],
  dat[which(dat$x > 500), ],
  subset(dat, x > 500)
)

bnch
#> # A tibble: 3 x 6
#>   expression                   min   median `itr/sec` mem_alloc `gc/sec`
#>   <bch:expr>                <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
#> 1 dat[dat$x > 500, ]          358µs    443µs     2184.      377KB     13.5
#> 2 dat[which(dat$x > 500), ]   261µs    338µs     2879.      260KB     13.5
#> 3 subset(dat, x > 500)        486µs    573µs     1696.      509KB     16.1
```

Advanced R Second Edition “Improve performance” section of the e-book at https://adv-r.hadley.nz/perf-improve.html
profvis R package at https://rstudio.github.io/profvis/
bench R package at https://github.com/r-lib/bench

Trade-offs and considerations beyond performance

Performance is not the only goal to consider



Case study: simplified experience study exposure calculation – 1/4

dplyr version

```
calculate_exposures_simple_dplyr <- function(df, observation_start, observation_end, ...) {
   df |>
    mutate(
      iss_age = if_else(
        !is.na(insured_birthdate) & !is.na(policy_issue_date),
        as.integer((policy_issue_date - insured_birthdate) / 365.25),
        NA_integer_
        )
      ) |>
    mutate(
      iss_month  = month(policy_issue_date),
      iss_day    = day(policy_issue_date),
      iss_year   = year(policy_issue_date),
      start      = pmax(policy_issue_date, observation_start, na.rm = TRUE),
      end        = pmin(expo_end_date, observation_end, na.rm = TRUE),
      start_year = year(start),
      end_year   = year(end)
      )
...
```

Case study: simplified experience study exposure calculation – 2/4

dtplyr version

```
calculate_exposures_simple_dtplyr <- function(df, observation_start, observation_end, ...) {
  setDT(df)
  df |>
    lazy_dt(immutable = FALSE) |>
    mutate(
      iss_age = if_else(
        !is.na(insured_birthdate) & !is.na(policy_issue_date),
        as.integer((policy_issue_date - insured_birthdate) / 365.25),
        NA_integer_
      )
    ) |>
    mutate(
      iss_month  = month(policy_issue_date),
      iss_day    = day(policy_issue_date),
      iss_year   = year(policy_issue_date)
    ) |>
    mutate(
      start      = pmax(policy_issue_date, observation_start, na.rm = TRUE),
      end        = pmin(expo_end_date, observation_end, na.rm = TRUE)
    ) |>
    mutate(
      start_year = year(start),
      end_year   = year(end)
    ) |>
    as.data.frame()
...
```

Case study: simplified experience study exposure calculation – 3/4

data.table version

```
calculate_exposures_simple_dt <- function(dt, observation_start, observation_end, ...) {

  dt[
    !is.na(insured_birthdate) & !is.na(policy_issue_date),
    iss_age := as.integer((policy_issue_date - insured_birthdate) / 365.25)][
    , ":="(iss_month = month(policy_issue_date),
           iss_day   = day(policy_issue_date),
           iss_year  = year(policy_issue_date))][
    , ":="(start = pmax(policy_issue_date, observation_start, na.rm = TRUE),
           end   = pmin(expo_end_date, observation_end, na.rm = TRUE))][
    , ":="(start_year = year(start), end_year = year(end))]

...
```

Case study: simplified experience study exposure calculation – 4/4

C version

R code

```r
calculate_exposures_simple_c <- function(df, observation_start, observation_end, ...) {
  .Call("Ccalculate_exposures_simple", df, observation_start, observation_end, ...)
}
```

C code

```c
#include <R.h>
#include <Rinternals.h>
#include <omp.h>
...

SEXP Ccalculate_exposures_simple(SEXP df, SEXP observation_start, SEXP observation_end, ...) {
  // Multi-threading via OpenMP
  const int n_th = MAX(1,MIN(INTEGER(n_threads)[0], omp_get_num_procs()));
  const int obs_start = (int) REAL(observation_start)[0];
  ...
#pragma omp parallel for num_threads(n_th)
  for (R_xlen_t i = 0; i < n_out; ++i) {
    ...
    iss_agep[i] = (int)((policy_issue_datep[i] - insured_birthdatep[i]) / 365.25);
    endp[i] = MIN(expo_end_datep[i], obs_end);
    ...
  }
...
```

Start exploring how to improve critical code

Choose the right trade-offs with R's toolbox

R can be performant and scalable

R ecosystem offers several tools to improve code

Keep trade-offs in mind

R Consortium Impact

- R Consortium Community **Grants** and Sponsorships Over **USD $1.4 Million**
- Organize large scale **collaborative projects**
  - R Validation Hub
  - R-Ladies
  - Diversity and Inclusion Working Group
- Co-host multidisciplinary **data science forums**
  - Stanford Data Institute
- Direct support for key **R events**
  - R/Medicine, R/Pharma, useR!, LatinR, more
- Direct support for **R User Groups**

Organizations Can Become a Member Today!

Email Joseph Rickert at **director@r-consortium.org** to set up first call