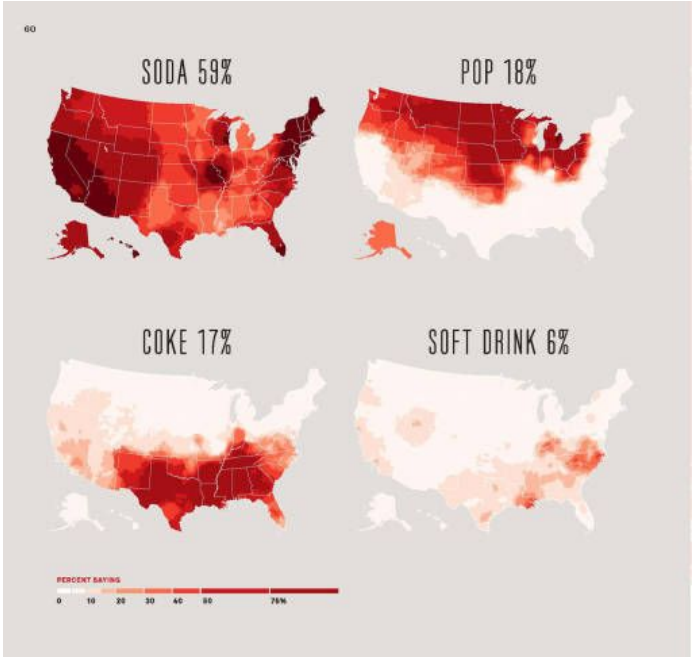
Scaling Shiny

10,000 User App

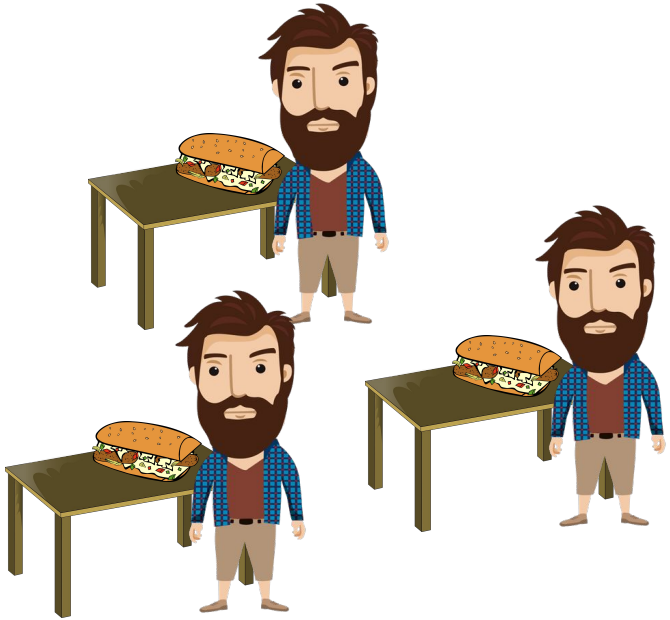
When Shiny was just a glimmer…

Josh Katz, NY Times: “How Y’all, Youse, and You Guys Talk”

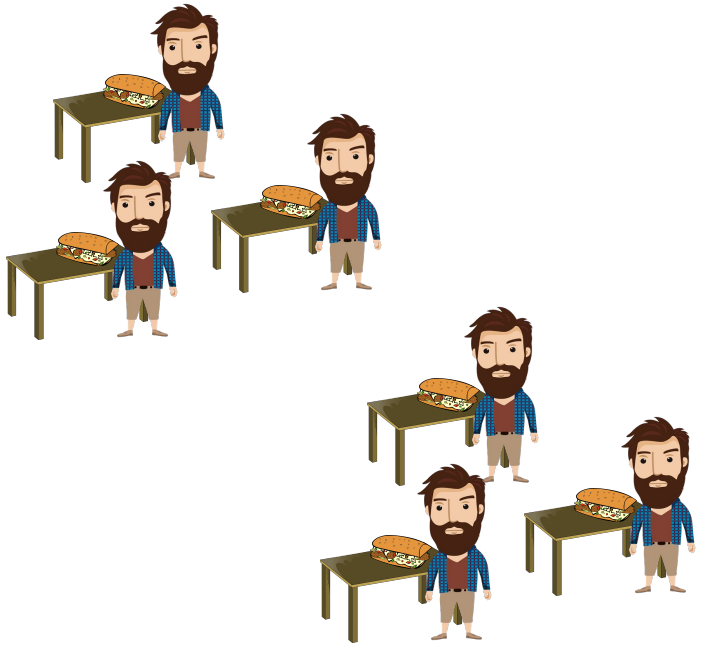


Scaling 101

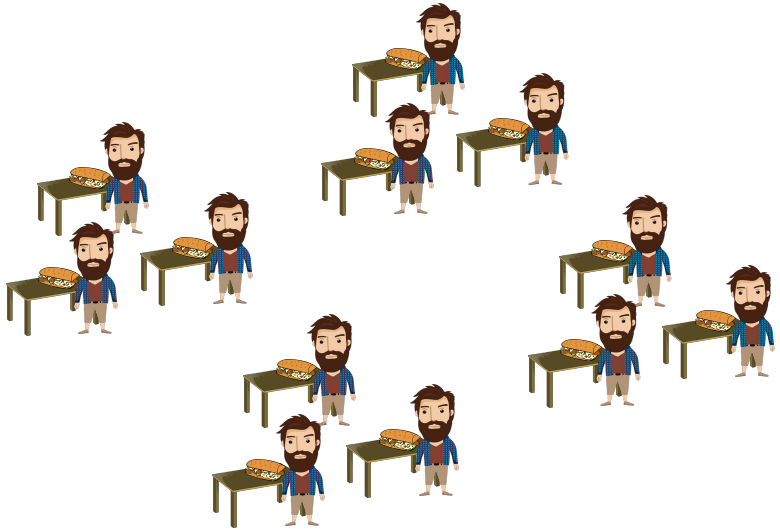
Scaling 101



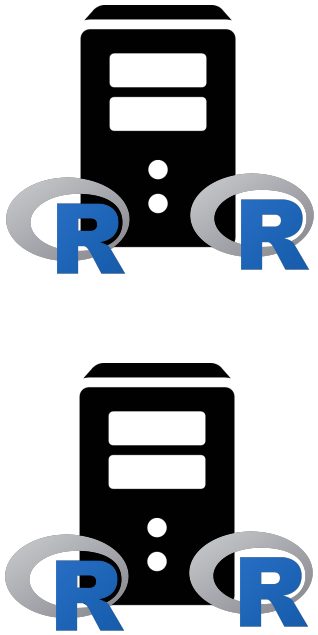
Scaling 101

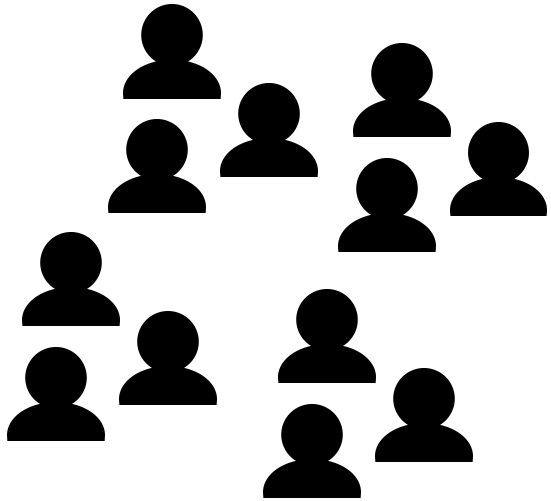


Scaling 101



Scaling 101





Our Stack

PostgresSQL DB (RDS)

Shared File System (EFS)

RStudio Connect Cluster (ASG)

Sticky Load Balancer (ALB)

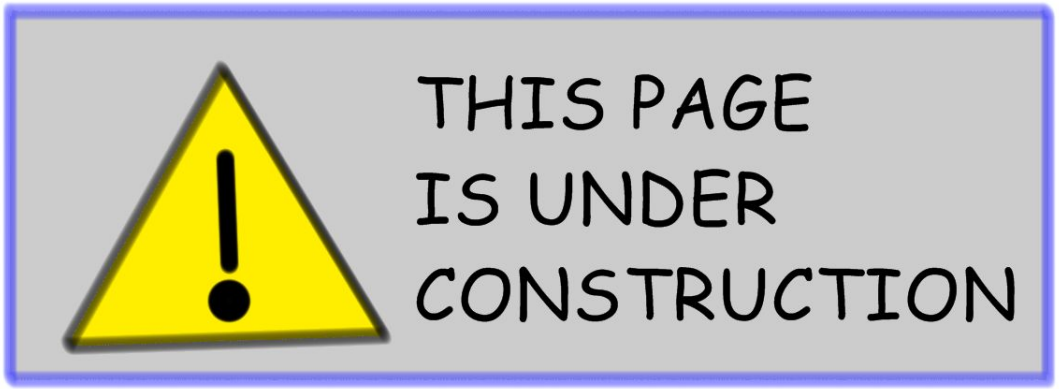
Tools for Load Testing

Goals:

1. Easy, interactive test recording
2. Playback recording, with an average machine simulating 1K users
3. **Support for websockets and Shiny** ✨

Demo

Where can I get these tools?



Writing a “Real” Load Test

```
library(beepr)

start <- Sys.time()
end <- start + lubridate::dminutes(30)

while(Sys.time() < end) {
  Sys.sleep(rnorm(1,25))
  beep() #change an input
}

beep(3)
```

Our Testing Stack

PostgresSQL DB (RDS)

Shared File System (EFS)

RStudio Connect Cluster (ASG)

Metrics Server (Prometheus & Grafana)

Sticky Load Balancer (ALB)

Load Testing Client (ASG)

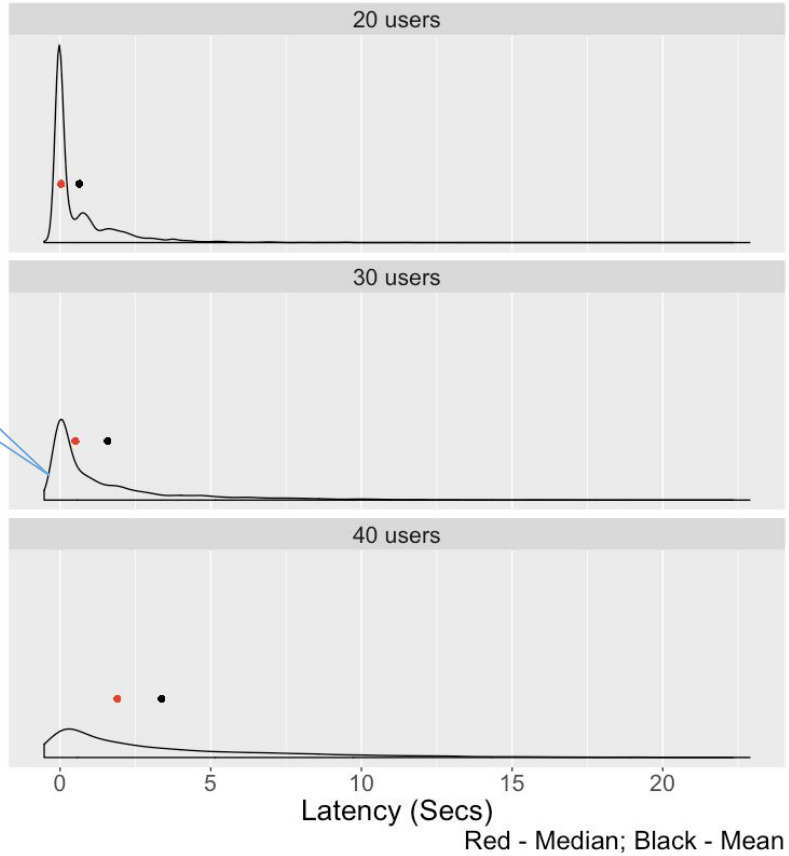
Me! Orchestrating the Stack (Python's Fabric)

Our Stack

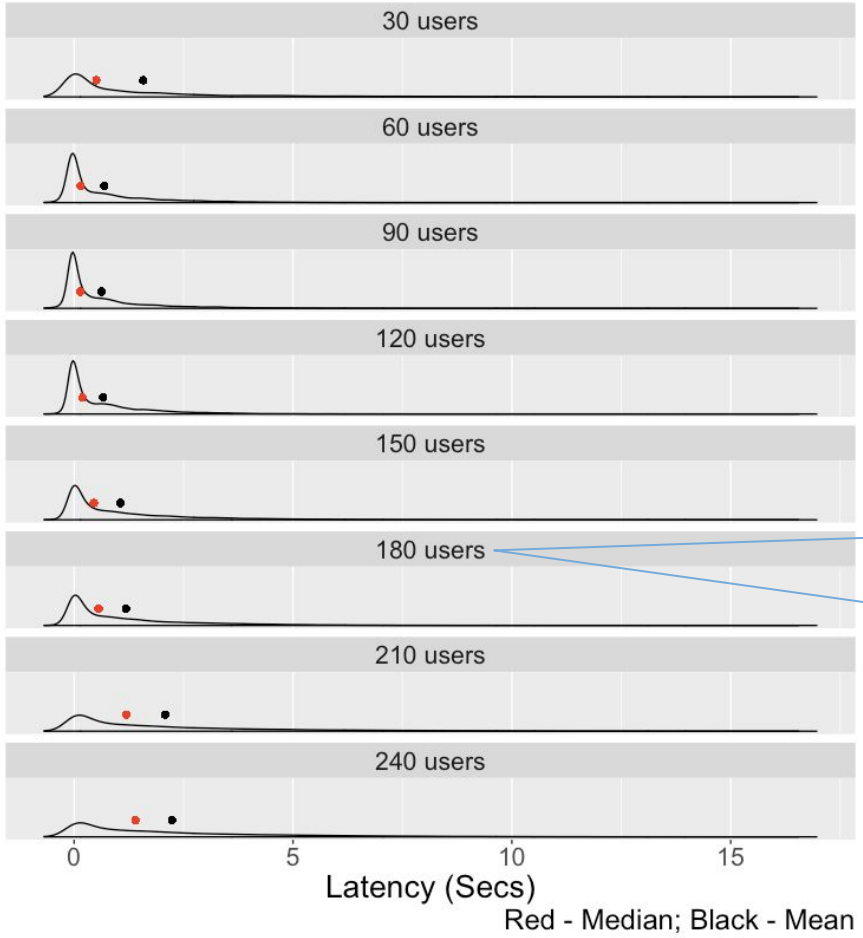
How many users per R process?

30 Users. 1 R process.

< 1 second median latency



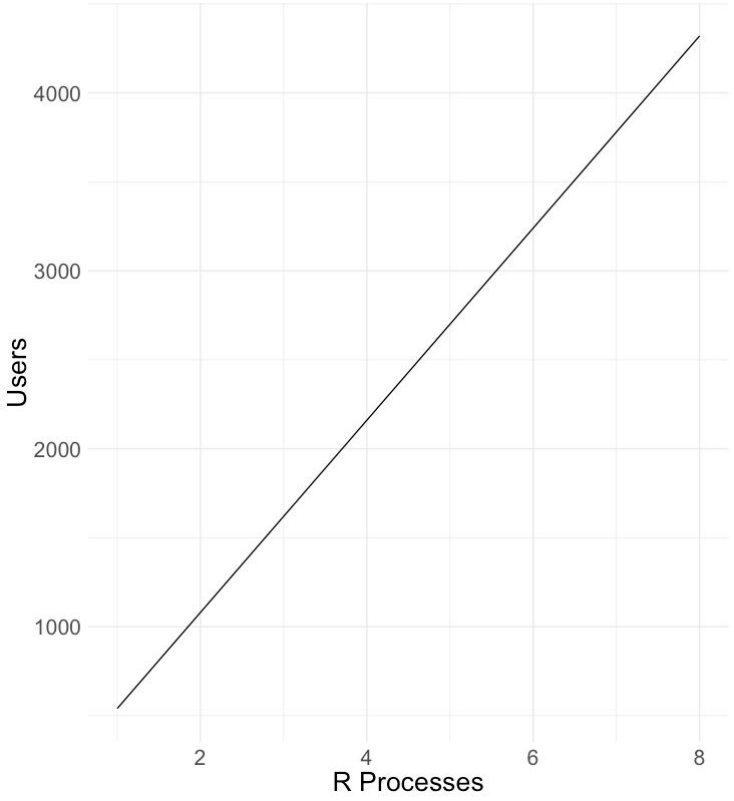
How many R processes per node?

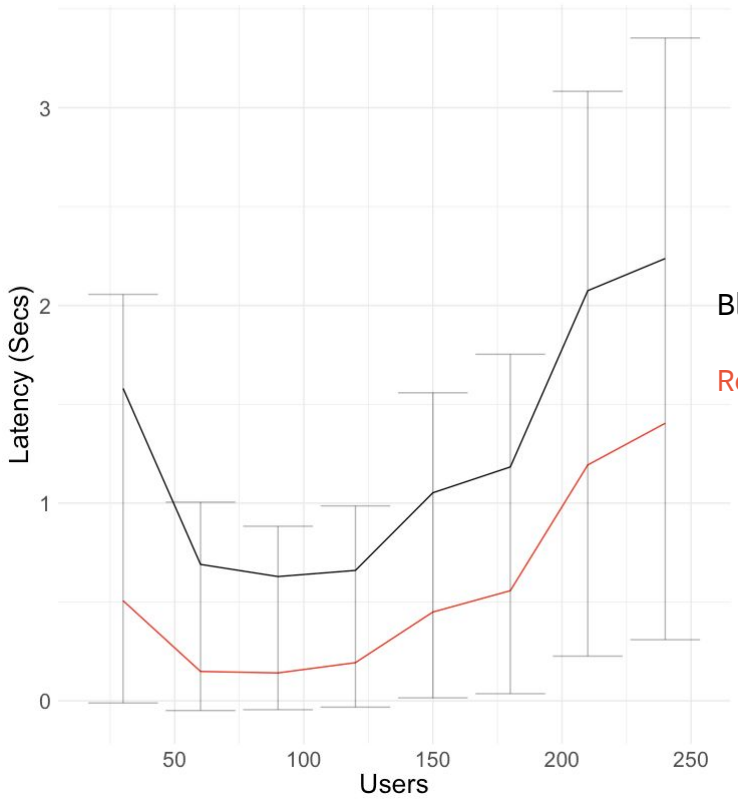


180 Users. 6 R processes.

< 1 second median latency

Linear Scaling



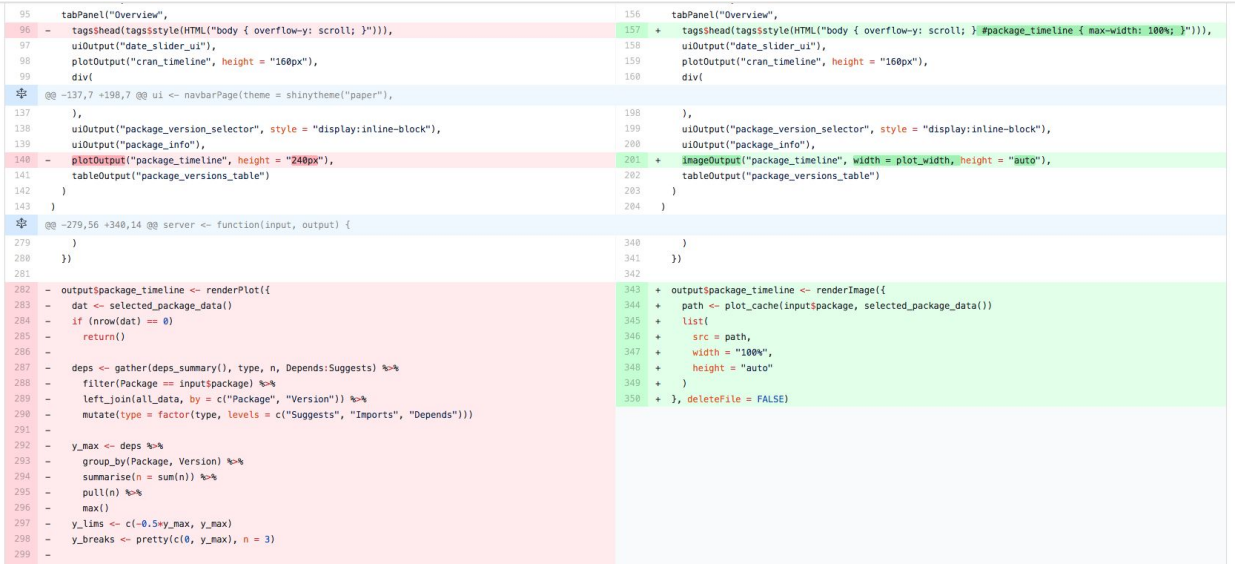


Black - Mean

Red - Median

App Optimization

All about the coefficients



```
 95    tabPanel("Overview",
96 -    tags$head(tags$style(HTML("body { overflow-y: scroll; }"))),
97      uiOutput("date_slider_ui"),
98      plotOutput("cran_timeline", height = "160px"),
99      div(
@@ -137,7 +198,7 @@ ui <- navbarPage(theme = shinytheme("paper"),
137     ),
138     uiOutput("package_version_selector", style = "display:inline-block"),
139     uiOutput("package_info"),
140 -   plotOutput("package_timeline", height = "240px"),
141     tableOutput("package_versions_table")
142   )
143 )
@@ -279,56 +340,14 @@ server <- function(input, output) {
279     )
280   })
281
282 - output$package_timeline <- renderPlot({
283 -   dat <- selected_package_data()
284 -   if (nrow(dat) == 0)
285 -     return()
286 -
287 -   deps <- gather(deps_summary(), type, n, Depends:Suggests) %>%
288 -     filter(Package == input$package) %>%
289 -     left_join(all_data, by = c("Package", "Version")) %>%
290 -     mutate(type = factor(type, levels = c("Suggests", "Imports", "Depends")))
291 -
292 -   y_max <- deps %>%
293 -     group_by(Package, Version) %>%
294 -     summarise(n = sum(n)) %>%
295 -     pull(n) %>%
296 -     max()
297 -   y_lims <- c(-0.5*y_max, y_max)
298 -   y_breaks <- pretty(c(0, y_max), n = 3)
299 -
```

```
156     tabPanel("Overview",
157 +     tags$head(tags$style(HTML("body { overflow-y: scroll; } #package_timeline { max-width: 100%; }"))),
158     uiOutput("date_slider_ui"),
159     plotOutput("cran_timeline", height = "160px"),
160     div(
198     ),
199     uiOutput("package_version_selector", style = "display:inline-block"),
200     uiOutput("package_info"),
201 +   imageOutput("package_timeline", width = plot_width, height = "auto"),
202     tableOutput("package_versions_table")
203   )
204 )
340     )
341   })
342
343 + output$package_timeline <- renderImage({
344 +   path <- plot_cache(input$package, selected_package_data())
345 +   list(
346 +     src = path,
347 +     width = "100%",
348 +     height = "auto"
349 +   )
350 + }, deleteFile = FALSE)
```

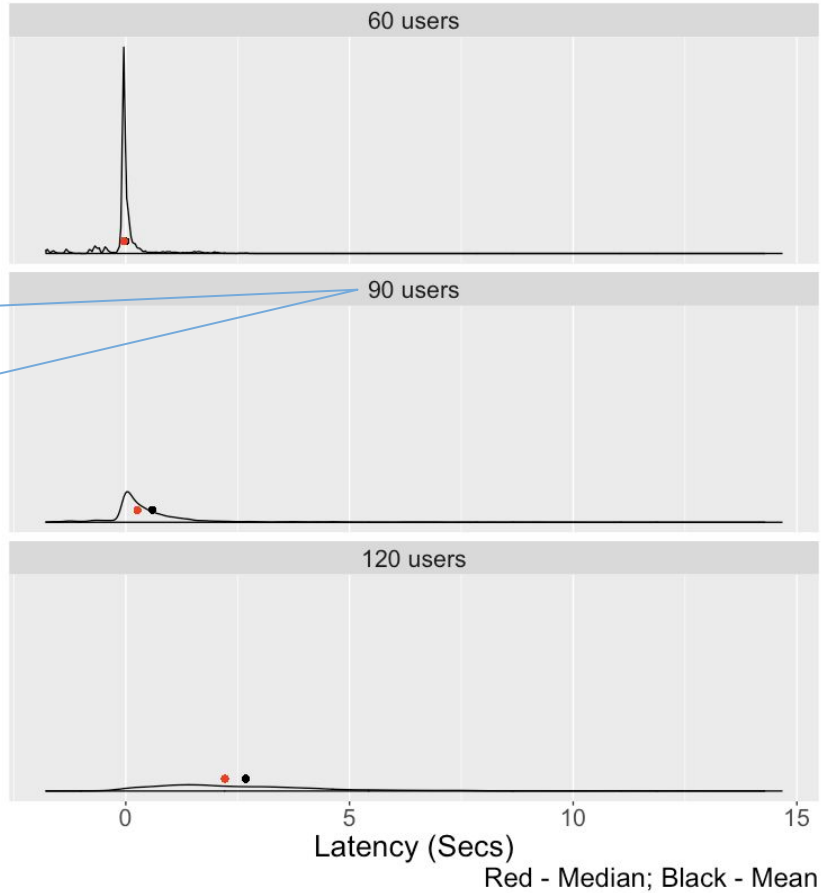
https://github.com/wch/shiny_demo/commit/f0ab6220b17490380e458e2b37f27018fdcbe2c2

How many users per R process (take 2)

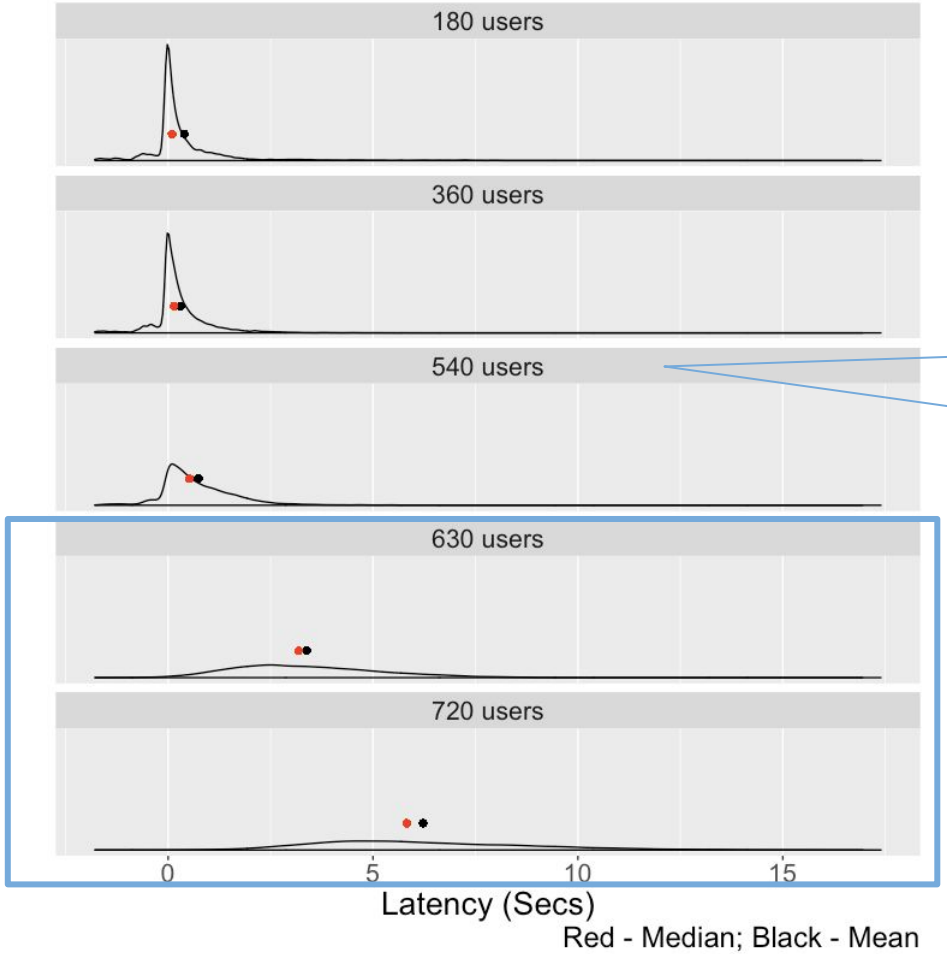
90 Users. 1 R process.

< 1 second median latency

3X Caching Improvement



How many R processes per node?

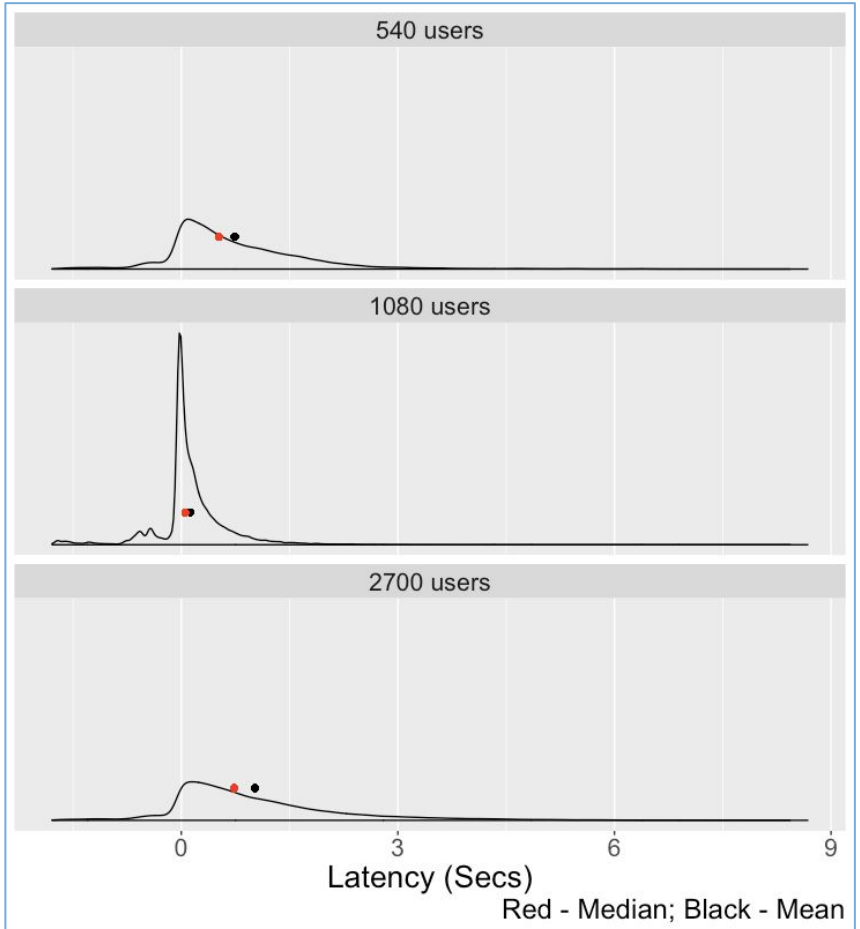


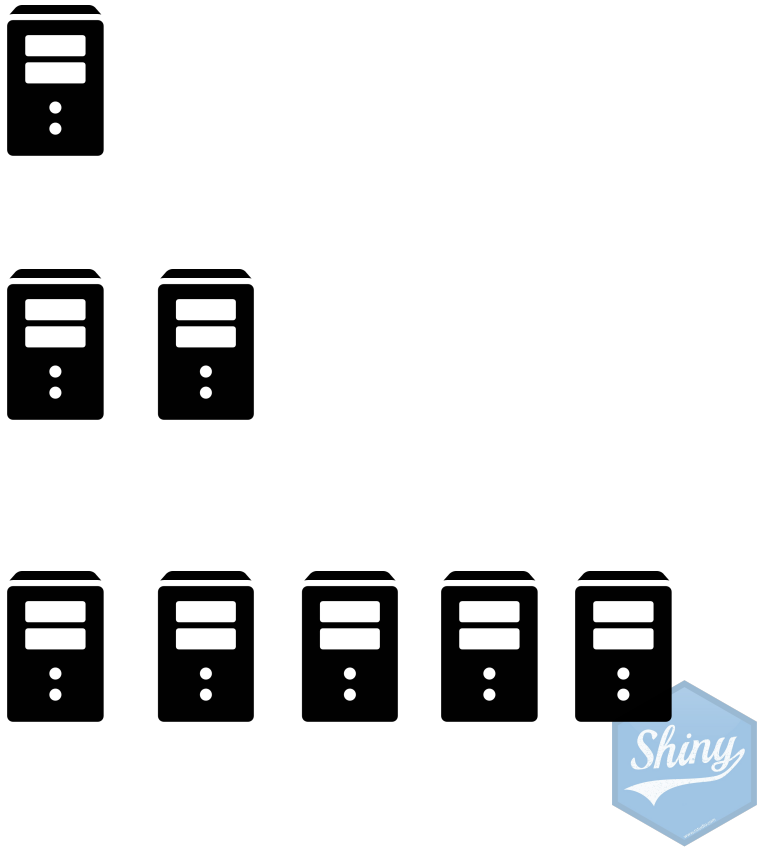
540 Users. 6 R processes.

3x Caching Improvement

CPU utilization over 100%

Can we scale across nodes?





10,000 User Demo

20 node cluster of c4.2xlarge (8 core)

Thanks!

Contact Me: sean@rstudio.com