Quarterly Meeting – 18 May 2021

24 Attendees

Juliane Manitz
Mark Penniston
Nicholas Masel
Matthew Montero
Satish Murthy
Jan Stiers Pieter
Soren Klim
Per Arne Stahl
Lyn Taylor
Andy Nicholls
Paulo Bargo
Bella Fang
Doug Kelkoff
Eli Miller
Emma Martin
Stephen Glavin
Steven Haesendonckx
Jennifer Bradford
Joseph Rickert
Susanna Marquez Gargallo
Tilo Blenk
Matthias Trampisch
Jenny Wissmar
Yilong Zhang

Agenda

- Infrastructure team kick off: Doug Kelhoff
Discussion

Testing steam update (GSK progress): Tilo Blenk provided the following slides summarizing the work GSK have done to date on R package testing, which will act as a basis to kick start activities of a testing stream.

GSK have built a system designed to satisfy the questions that the FDA would ask with respect to adequate testing.
library(readr)
library(dplyr)

hers <- read_csv("data/hersdata.csv")

hers %>%
  group_by(diabetes, exercise) %>%
  summarise(n = n(), mean_glc = mean(glucose))
# diabetes exercise   mean_glc
# 1 no    no           1191  97.4
# 2 no    yes          841   95.7
# 3 yes   no           384   155
# 4 yes   yes           227   155

fit <- lm(glucose ~ exercise, data = hers, subset = (diabetes == "no"))

summary(fit)
# Coefficients:
# Estimate Std. Error t value Pr(>|t|)
# (Intercept) 97.3610  0.2815  345.048  < 2e-16
# exerciseyes -1.6928   0.4376  -3.868 0.0000133
#
# Residual standard error: 9.715 on 2830 degrees of freedom
# Multiple R-squared:  0.00318,  Adjusted R-squared:  0.00313
# F-statistic: 54.97 on 1 and 2830 DF,  p-value: 8.80e-13

---

Components of R testing/verifying

- **package assessment**
  Assessing packages to decide if they can be considered as sufficiently tested/verified as they are.

- **resource assessment**
  Assessing resources like the R Foundation and RStudio to decide if the products they provide, ie R base distribution from R Foundation or tidyverse, r-libs, etc from RStudio, can be considered as sufficiently tested/verified.

- **testing**
  Testing packages: (1) qualification tests for packages considered in package assessment as sufficiently tested and (2) verification tests (reliability/correctness) for packages considered as insufficiently tested.

- **frozen R installations**
  R installations with R base distribution and selected R packages which users cannot change, ie no package installation or update is possible for users.

- **controlled execution**
  When executing a R script for a GxP process (1) frozen installation is used, (2) checking that only tested/verified functions/packages are used, (3) executing as background process, and (4) capturing/saving R script, context information, and standard out/error.
The above works good for simple tests, but not for statistical modelling testing.
Testing statistical models (external reference)

test that(“linear regression models”,
  
  # table 6.3 page 96
  d <- data.frame(
    carbohydrate = c(33, 48, 37, 27, 30, 43, 34, 48, 38, 38, 50, 61, 30, 36, 41, 42, 46, 24, 15, 37),
    age = c(33, 47, 49, 35, 46, 52, 63, 32, 42, 31, 61, 63, 40, 56, 46, 58, 61, 48, 28),
    weight = c(100, 92, 135, 144, 140, 101, 95, 101, 98, 108, 85, 130, 127, 109, 107, 117, 100, 518, 102),
    protein = c(14, 15, 18, 12, 15, 15, 14, 17, 15, 14, 17, 19, 19, 20, 15, 16, 18, 13, 18, 14)
  )

  fit <- lm(carbohydrate ~ age + weight + protein, data = d)
  # table 6.4 page 97
  expect_equivalent(  
    round(coefficients(fit), 3),
    c(36.968, -0.114, -0.228, 1.958)
  )
  expect_equivalent(  
    round(summary(fit)$coefficients[,”Std. Error”], 3),
    c(13.071, 0.109, 0.683, 0.635)
  )
)

Textbooks as external references

| Table 6.3 (Continued)...
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Carbohydrates</td>
<td>Age</td>
<td>Weight</td>
<td>Protein</td>
</tr>
<tr>
<td>33</td>
<td>48</td>
<td>37</td>
<td>27</td>
</tr>
<tr>
<td>30</td>
<td>43</td>
<td>34</td>
<td>48</td>
</tr>
<tr>
<td>38</td>
<td>38</td>
<td>50</td>
<td>61</td>
</tr>
<tr>
<td>30</td>
<td>36</td>
<td>41</td>
<td>42</td>
</tr>
<tr>
<td>46</td>
<td>24</td>
<td>15</td>
<td>37</td>
</tr>
</tbody>
</table>

The expected mean for a 50-year-old person with a weight of 70 kg and a protein intake of 20 g is:

\[ \text{Expected Mean} = 36.968 - (0.114 \times 50) - (0.228 \times 70) + (1.958 \times 20) \]

The coefficient for age is significant, indicating that the response does not depend on age. The coefficient for weight is not significant, suggesting that weight has no effect on the response. The coefficient for protein is also not significant, indicating that protein intake has no effect on the response.

Table 6.4 (Continued)...

| Model | Estimate | Std. Error | t value | Pr(>|t|) |
|-------|----------|------------|---------|----------|
| Intercept | 36.968 | 13.071 | 2.834 | 0.005 |
| Age | -0.114 | 0.109 | -1.043 | 0.303 |
| Weight | -0.228 | 0.683 | -0.334 | 0.739 |
| Protein | 1.958 | 0.635 | 3.081 | 0.002 |

The final model is:

\[ \text{Expected Mean} = 36.968 - (0.114 \times \text{Age}) - (0.228 \times \text{Weight}) + (1.958 \times \text{Protein}) \]
Real world data, example above has 2763 observations, so realistic testing.

**Correct results have to be known**

```r
> set.seed(123)
> x <- rnorm(100)  # create numeric vector x with 100 random numbers
> x
[1] -0.560475647 -0.230177489
[3] 1.558708314 0.070508391
[5] 0.229287735 1.715064987
...  
[97] 2.287332993 1.532610626
[99] -0.235700359 -1.626429050
> mean(x)  # calculate arithmetic mean of vector
[1] 0.09948591  # DO WE REALLY KNOW THAT 0.09948591 IS CORRECT?
> expect_equal(mean(x), 0.09948591)
```

To get the known results, you need to use external references or obvious results that are known.
How detailed need the tests to be?

> x <- sample(1:10)  # create numeric vector x with numbers 1 to 10 in random order
> expect_equal(mean(x), 5.5)  # test mean(x) against expected value of 5.5

> x <- sample(1:1e6)  # bigger input, vector with numbers 1 to 1 million
> expect_equal(mean(x), 500000.5)

> x <- c(1e6, 1e6, 0.01, 0.01)  # very big and small elements
> expect_equal(mean(x), 500000.005)

> x <- c(1, 2, 3, NA)  # handling of NA values
> expect_true(is.na(mean(x)))
> expect_true(is.na(mean(x, na.rm = TRUE)), 2)

> x <- numeric(0)  # numeric vector without elements
> expect_true(is.na(mean(x)))

> x <- c(-10, -2:9, 500)  # more function arguments
> expect_equal(mean(x, trim = 0.1), 5.5)

Running tests interactively in RStudio IDE

> test_dir("rtests")
✓ | OK | W | S | Context
✓ | 36 | 4 | base [8.6 s]
✓ | 54 | 4 | dplyr-reg [8.5 s]
✓ | 140 | 4 | purrr [0.3 s]
✓ | 30 | 4 | forcats
✓ | 10 | 4 | haven
✓ | 15 | 4 | jsonlite
✓ | 16 | 4 | lubridate
✓ | 27 | 4 | magrittr
✓ | 20 | 4 | reader
✓ | 123 | 4 | rlang [0.2 s]
✓ | 44 | 4 | stats
✓ | 26 | 4 | stringr
✓ | 13 | 4 | tibble
✓ | 19 | 4 | tidyr
✓ | 20 | 4 | xml2
✓ | 6 | 4 | yaml

== Results ==
Duration: 2.2 s

[ FAIL 0 | WARN 0 | SKIP 0 | PASS 907 ]
Documenting tests with R Markdown and testthat

```
library(testthat)

test_that("data frames", {
  i <- 1:10
  d <- data.frame(i = i, f = i + 0.12345, s = letters[i], stringsAsFactors = FALSE)
  expect_true(is.data.frame(d))
  expect_equal(dim(d), c(10, 3))
  expect_equal(nrow(d), 10)
  expect_equal(ncol(d), 3)
  expect_equal(colnames(d), c("i", "f", "s"))
  expect_equal(d[1,], 1)
  expect_equal(d[1,], c(i = 1, f = i + 0.12345, s = "a")
  expect_equal(d[1], data.frame(i = 1, f = 1.12345, s = "a", stringsAsFactors = FALSE))
  expect_equal(d[1,], 1)
  expect_equal(d[3,3], "c")
})
```
Questions for the testing stream

Per Arne Stahl (AZ): AZ are at the same position as GSK and having same discussion about automated testing. One question coming back from QA is how to test the test scripts. Did you have this question at GSK? Tilo’s response: No, they didn’t get that, but for test that, you can test it such that if the TRUE comes out when you know it to be true, then you can show that seems to work.

Andy Nicholls added, that we also have to have faith/trust in the BASE language, such that TRUE is TRUE, FALSE is FALSE and so on.

Per Arne Stahl: stressed that given R has been used for 20 years or so, we all do believe it works as it’s used by academics and they have written new statistical methods using it, however we just need to provide the documentation of this for industry regulators.

Per Arne Stahl – have you asked the regulators if they are happy with the approach you are using. Tilo’s response: No. Andy Nicholls: the idea would be that we take something like this approach and the tests to the R validation Hub testing stream, and release it to the wider R Validation Hub community. This way we can come together and release this to the regulators to request that they do accept this approach and this method of testing for R use in industry.

Doug Kelkhoff: How do you see the collaboration happening in this space. How would companies collaborate to incorporate the tests into the packages. Could we ask the authors to include the tests in their packages? Tilo’s response: as they in academia and not in industry they may not be happy to do this... and also you will need some qualification tests outside of the package tests. Hence, the aim is to write the tests, make them available through the R Validation Hub, to allow free use of the installed systems and packages along with the testing scripts. The intention is to share with the community, and discussions will continue through the testing stream of the R Validation Hub. The intention is to write a white paper on this topic.

Infrastructure: Doug Kelhoff presented the following slides on the newly setup infrastructure team.
Motivations

- **riskmetric:**
  Foundational tools for package assessment

- **Shiny app:**
  Interface for institutions to operationalize riskmetric as part of business process

- **Remainging Outreach Gap:**
  How do we make these tools available for communication, industry consistency, regulator reference?

Seed Ideas

- Database of metrics evaluated against package versions
- Package Repository with Validation reports
- Risk score repo badge
Infrastructure Team Brainstorming Session

Special thanks to attendees: Edgar Manukyan (Roche), Eli Miller (Atorus), Eric Millman (Biogen), Heidi Curinckx (J&J), Marly Cormar (Biogen), Mike Stackhouse (Atorus), Nan Xiao (Merck), Steven Haesendonckx (J&J), Yilong Zhang (Merck)

Infrastructure Team First Steps

Build the Team

- Contributors welcome!
- Lead(s) / organizer role

Infrastructure Team First Steps

Risk score API

- Build an API to run and return risk scores
- Build an endpoint for risk score badges
- Communicate new development needs to riskmetric/shiny app teams
- Estimate infrastructure need to host the api and/or cache risk scores in a database
- Help to write a R Consortium proposal for infrastructure budget
Infrastructure Team Formation

Interested? Join our Slack!

rvalidationhub.slack.com
# infrastructure