PHUSE EU Connect 2025

OS17: End to End R: Where to Start?

Luke Thompson
Statistical Programmer
Astellas

11th October 2025

Agenda

I Background

II Development of an R Environment

III Utility macros

IV Useful R resources

V SDTM, ADaM & TLG Programming

VI Conclusion

The speaker is a paid employee of Astellas. This presentation is intended for informational purposes only and does not replace independent professional judgment. This presentation is not intended to provide medical or legal advice. Statements of fact, positions taken and opinions expressed are those of the speaker individually and, unless expressly stated to the contrary, do not necessarily reflect the opinion or position of the speaker's employer, Astellas, or any of its subsidiaries and/or related entities.

Background

Background

- Astellas in June 2024
- 5502-CL-0001 study selected for End to End (E2E) R:

A Phase 1 Combined Single and Multiple Ascending Oral Dose Study to Evaluate the Safety, Tolerability, Pharmacokinetics, Food Effect and Biomarker Response of ASP5502 in Healthy Adults and Participants With Primary Sjögren's Syndrome

- E2E R – SDTM, ADaM, TLGs & eSub
- Why E2E R? Because there isn’t a better time. Pharmaverse packages. Many companies have adopted it already. More and more programmers have R skills.

R environment

Development of the infrastructure

Development of an R Environment

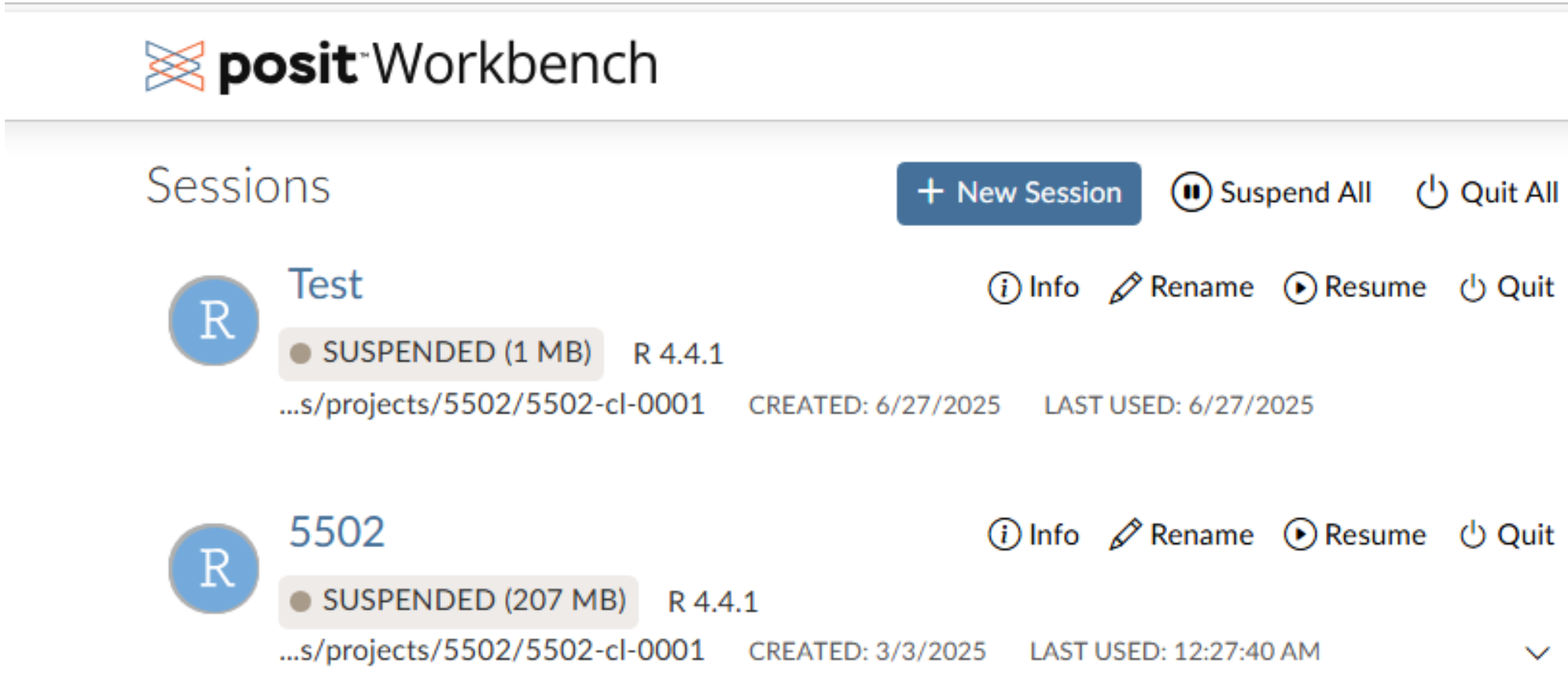
R Environment

Posit Workbench®

- Vendor supplied tool
- Kubernetes system
- User friendly interface

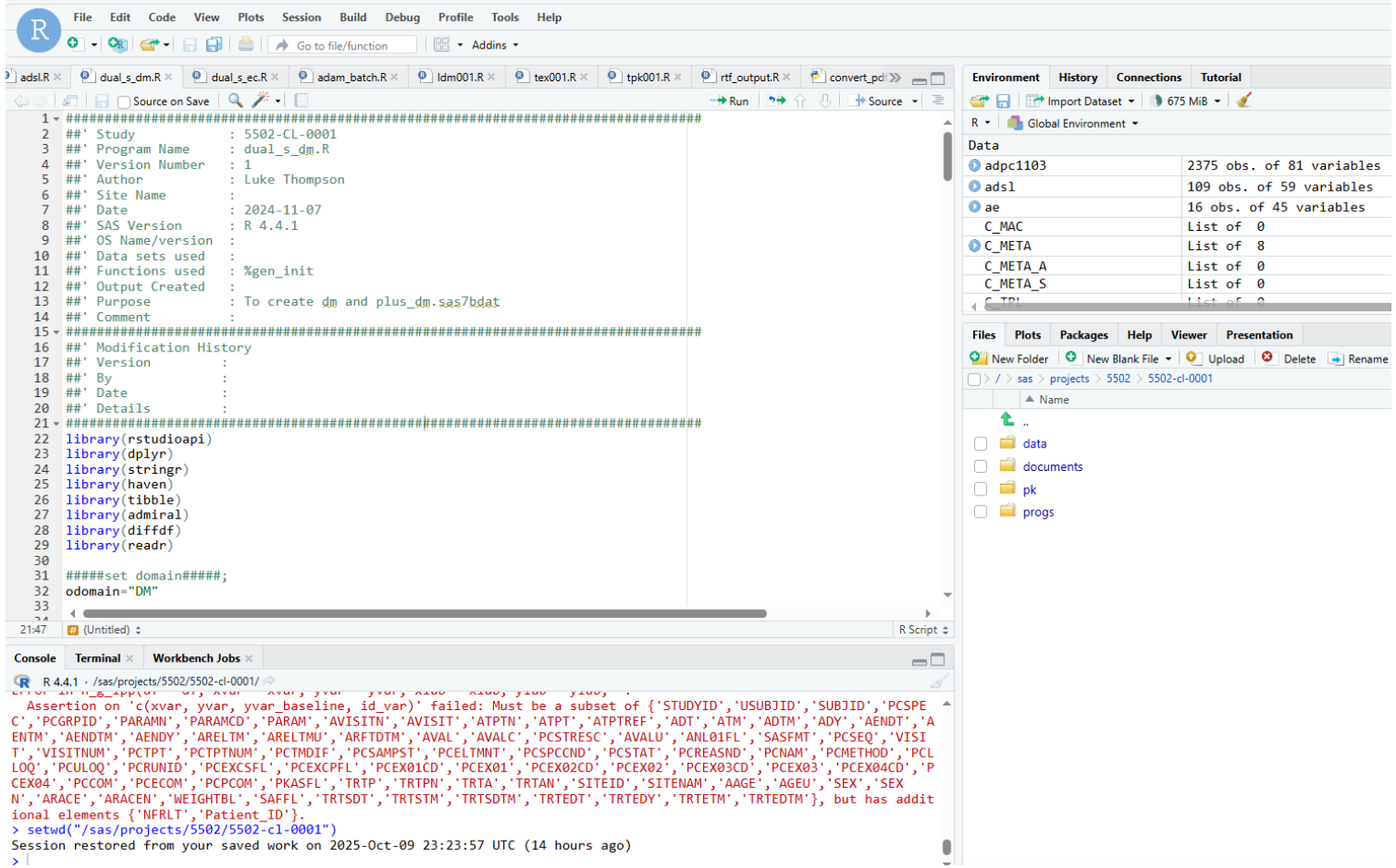
RStudio® (IDE)

- Vendor supplied tool
- Internal Astellas website
- Connects to the Astellas Statistical Computing Environment (SCE)



R Environment

Figure: a screenshot of 2 suspended sessions in Posit Workbench.



R Environment

Figure: a screenshot of the Rstudio environment

R language and Packages

Importance of Validation

- Required for regulatory compliance, data integrity, quality assurance, standardisation, & traceability
- SAS has built-in validation – Due to reputation, pre validated by vendor.
- R requires validation from individual or companies, because it is open source. E.g. package quality may differ (documentation, validation levels) and it is ever-changing.
- Possible to do inhouse, but workload too demanding

R language and Packages

OpenVal® by Atorus Research

- Comprehensive suite of validated packages
- Release every 6 months and packages can be added, updated or removed
- Negotiation possible
- Contained many of the packages we required including some Pharmaverse packages

Utility Macros

Utility Macros

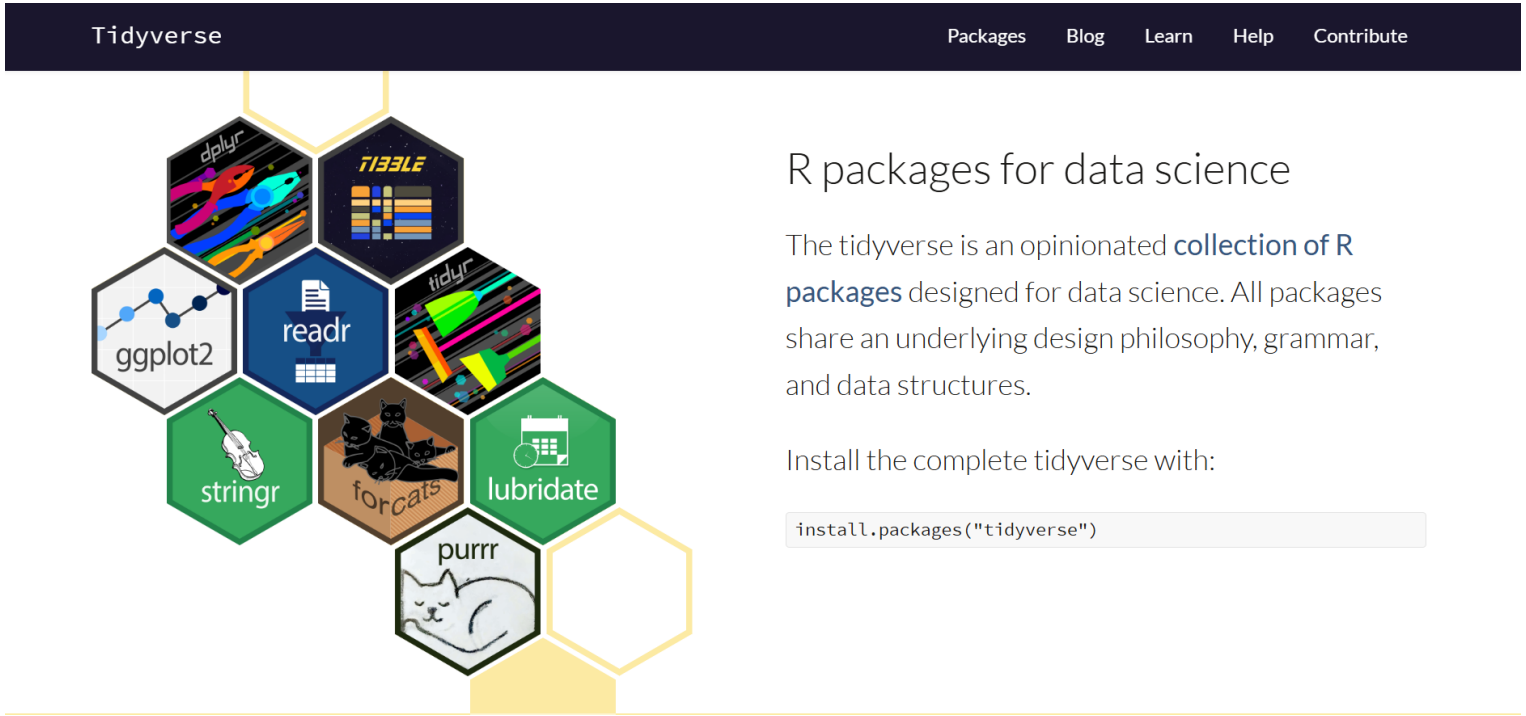
Setup

- Maintain the current workflow we had for SAS
- Utilise code and templates we already had and convert them into SAS.
- We replicated utility SAS Macros into R functions
- Functions to initialise program names, load libraries, read in deliverable files, create formats, ect.
- Tidyverse – dplyr, stringr, haven & magnitr

Useful R Resources

Tidyverse websites

Useful tools to begin R programming



Tidyverse websites

Useful tools to begin R programming

Data transformation with dplyr : : CHEATSHEET

dplyr functions work with pipes and expect **tidy data**. In tidy data:

Each **variable** is in its own **column** & Each **observation**, or **case**, is in its own **row**

pipes
x |> f(y) becomes **f(x, y)**

Summarize Cases

Apply **summary functions** to columns to create a new table of summary statistics. Summary functions take vectors as input and return one value (see back).

summary function

summarize(.data, …)
Compute table of summaries.
mtcars |> summarize(avg = mean(mpg))

count(.data, …, wt = NULL, sort = FALSE, name = NULL) Count number of rows in each group defined by the variables in … Also **tally()**, **add_count()**, **add_tally()**.
mtcars |> count(cyl)

Group Cases

Use **group_by**(.data, …, .add = FALSE, .drop = TRUE) to create a "grouped" copy of a table grouped by columns in ... dplyr functions will manipulate each "group" separately and combine the results.

```
mtcars |>
  group_by(cyl) |>
  summarize(avg = mean(mpg))
```

Alternate grouping syntax with **.by** as an argument:

```
mtcars |>
  summarize(
    avg = mean(mpg), .by = cyl
  )
```

Use **rowwise**(.data, …) to group data into individual rows. dplyr functions will compute results for each row. Also apply functions to list-columns. See tidyr cheat sheet for list-column workflow.

```
starwars |>
  rowwise() |>
  mutate(film_count = length(films))
```

ungroup(x, …) Returns ungrouped copy of table.

```
g_mtcars <- mtcars |> group_by(cyl)
ungroup(g_mtcars)
```

Manipulate Cases

EXTRACT CASES

Row functions return a subset of rows as a new table.

filter(.data, …, .preserve = FALSE) Extract rows that meet logical criteria.
mtcars |> filter(mpg > 20)

distinct(.data, …, .keep_all = FALSE) Remove rows with duplicate values.
mtcars |> distinct(gear)

slice(.data, …, .preserve = FALSE) Select rows by position.
mtcars |> slice(10:15)

slice_sample(.data, …, n, prop, weight_by = NULL, replace = FALSE) Randomly select rows. Use n to select a number of rows and prop to select a fraction of rows.
mtcars |> slice_sample(n = 5, replace = TRUE)

slice_min(.data, order_by, …, n, prop, with_ties = TRUE) and **slice_max()** Select rows with the lowest and highest values.
mtcars |> slice_min(mpg, prop = 0.25)

slice_head(.data, …, n, prop) and **slice_tail()** Select the first or last rows.
mtcars |> slice_head(n = 5)

Logical and boolean operators to use with filter()

==	<	<=	is.na()	%in%	|	xor()
!=	>	>=	!is.na()	!	&	

See **?base::Logic** and **?Comparison** for help.

ARRANGE CASES

arrange(.data, …, .by_group = FALSE) Order rows by values of a column or columns (low to high), use with **desc()** to order from high to low.
mtcars |> arrange(mpg)
mtcars |> arrange(desc(mpg))

ADD CASES

add_row(.data, …, .before = NULL, .after = NULL) Add one or more rows to a table.
cars |> add_row(speed = 1, dist = 1)

Manipulate Variables

EXTRACT VARIABLES

Column functions return a set of columns as a new vector or table.

pull(.data, var = -1, name = NULL, …) Extract column values as a vector, by name or index.
mtcars |> pull(wt)

select(.data, …) Extract columns as a table.
mtcars |> select(mpg, wt)

relocate(.data, …, .before = NULL, .after = NULL) Move columns to new position.
mtcars |> relocate(mpg, cyl, .after = last_col())

Use these helpers with select() and across()
e.g. mtcars |> select(mpg:cyl)

contains(match)
ends_with(match)
starts_with(match)
num_range(prefix, range)
all_of(x)/**any_of**(x, …, vars)
matches(match)
:, e.g., mpg:cyl
!, e.g., !gear
everything()

MANIPULATE MULTIPLE VARIABLES AT ONCE

```
df <- tibble(x_1 = c(1, 2), x_2 = c(3, 4), y = c(4, 5))
```

across(.cols, .funs, …, .names = NULL) Summarize or mutate multiple columns in the same way.
df |> summarize(across(everything(), mean))

c_across(.cols) Compute across columns in row-wise data.

```
df |>
  rowwise() |>
  mutate(x_total = sum(c_across(1:2)))
```

MAKE NEW VARIABLES

Apply **vectorized functions** to columns. Vectorized functions take vectors as input and return vectors of the same length as output (see back).

vectorized function

mutate(.data, …, .keep = "all", .before = NULL, .after = NULL) Compute new column(s). Also **add_column()**.
mtcars |> mutate(gpm = 1 / mpg)
mtcars |> mutate(gpm = 1 / mpg, .keep = "none")

rename(.data, …) Rename columns. Use **rename_with()** to rename with a function.
mtcars |> rename(miles_per_gallon = mpg)

posit®

Online courses

Useful R Resources

Online Resources

Useful R Resources

AI?

Useful R Resources

How would I sort a dataset named data by USUBJID and VISITUM and set a new variable named TEST to 1 when first.usubjid in R as opposed to SAS?

Stella 15:23 Edited

In R, you can sort a dataset and create a new variable based on conditions using the `dplyr` package, which is part of the tidyverse. Here's how you can do it:

1. First, ensure you have the `dplyr` package installed and loaded.
2. Use the `arrange()` function to sort the dataset by `usubjid` and `visitum`.
3. Use the `mutate()` function to create a new variable `test` that is set to 1 for the first occurrence of each `usubjid`.

Here's an example code snippet:

```
# Load the dplyr package
library(dplyr)

# Assuming your dataset is named 'data'
data <- data %>%
  arrange(usubjid, visitum) %>%  # Sort by usubjid and visitum
  group_by(usubjid) %>%           # Group by usubjid
  mutate(test = ifelse(row_number() == 1, 1, 0)) %>%  # Set test to 1 for the first usubjid
  ungroup()                       # Ungroup the data

# View the modified dataset
print(data)
```

In this code:

- `arrange()` sorts the dataset.
- `group_by()` groups the data by `usubjid`.
- `mutate()` creates the new variable `test`, setting it to 1 for the first occurrence of each `usubjid` and 0 otherwise.
- `ungroup()` is used to remove the grouping after the mutation.

This approach should give you the desired result similar to what you would achieve in SAS. Let me know if you have any further questions!

SDTM, ADaM & TLG Programming

SDTM, ADaM & TLG Programming

Pharmaverse

"A connected network of companies and individuals working to promote collaborative development of curated open-source R packages for clinical reporting usage in pharma, in a space where previously we would only ever have worked in silos on our own closed-source and often duplicative solutions."

- There are a wealth of packages available for E2E R
- Highly recommend checking their website: End-to-End Clinical Reporting Packages

SDTM, ADaM & TLG Programming

Pharmaverse

SDTM, ADaM & TLG Programming

SDTM – non-pharmaverse

- Pharmaverse package called OAK, opted not to use it.

“An EDC (Electronic Data Capture systems) and Data Standard agnostic solution that enables the pharmaceutical programming community to develop CDISC (Clinical Data Interchange Standards Consortium) SDTM (Study Data Tabulation Model) datasets in R”

- Other packages used instead, to continue the replication of standard templates.
- Tidyverse mainly used

```
dm <- dm_raw %>%
  dplyr::rename(COUNTRY_ = COUNTRY, SEX_ = SEX, AGEU_ = AGEU ) %>%
  dplyr::mutate(
    STUDYID = project,
    DOMAIN = odomain,
    SUBJID = stringr::str_trim(Subject),
    USUBJID = ifelse(is.na(SUBJIDP),
                     paste0(STUDYID, "-", SUBJID),paste0(STUDYID, "-",
substr(SUBJID,1,5),SUBJIDP)),
    SITEID = stringr::str_trim(as.character(`_SITEID`)),
    SEX = stringr::str_trim(substr(SEX_, 1, 1)),
    COUNTRY = COUNTRY_STD)
```

SDTM, ADaM & TLG Programming

Figure: a snippet of programming of SDTM DM

SDTM, ADaM & TLG Programming

Figure: a snippet of programming of SDTM DM

```
# Deriving dthdtc
ddthdtc1 <-
  # Combine with ae_1 where conditions are met
  dplyr::bind_rows(
    S_RAW[['ae_1']] %>%
      dplyr::mutate(across(where(is.character), ~ na_if(., ""))) %>%
      dplyr::filter(toupper(AESDTH) == "YES") %>%
      rename_ymd(prefin = "AEENDT", prefout = "DTHDT")
  ) %>%
  # Combine with ds_3 where conditions are met
  dplyr::bind_rows(
    S_RAW[['ds_3']] %>%
      dplyr::mutate(across(where(is.character), ~ na_if(., ""))) %>%
      dplyr::filter(toupper(DSDECOD) == "DEATH") %>%
      rename_ymd(prefin = "DSSTDT", prefout = "DTHDT")
  ) %>%
  # Combine with ds_4 where conditions are met
  dplyr::bind_rows(
    S_RAW[['ds_4']] %>%
      dplyr::mutate(across(where(is.character), ~ na_if(., ""))) %>%
      dplyr::filter(toupper(DSDECOD) == "DEATH") %>%
```

SDTM, ADaM & TLG Programming

ADaM – pharmaverse (Admiral)

- *“(ADaM In R Asset Library) - Modular framework to generate ADaM via R functions relying on community contributions”*

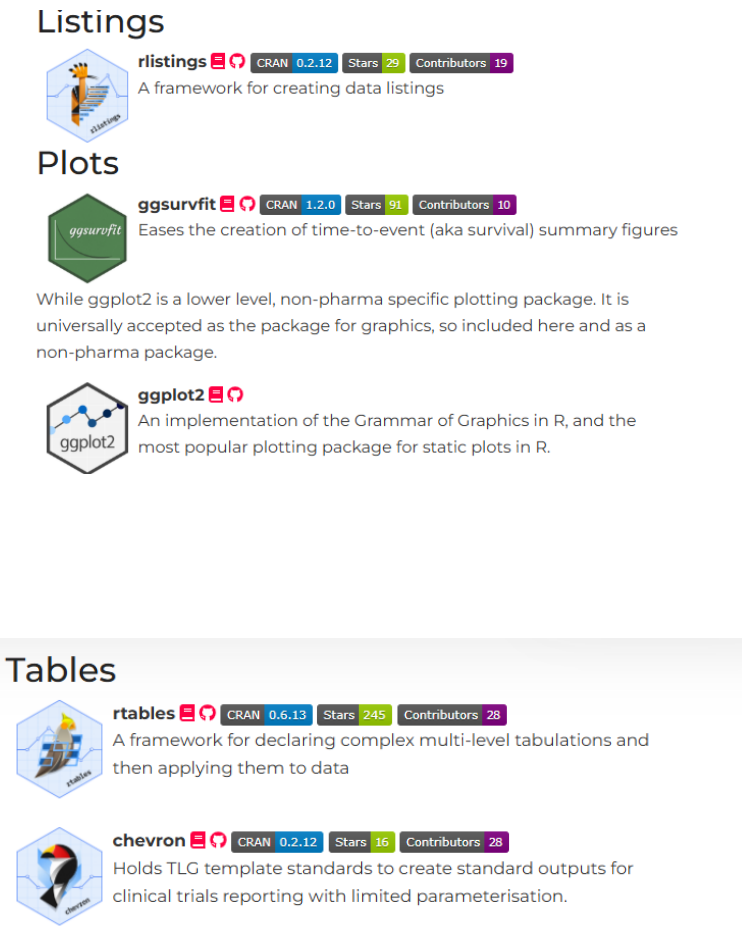
```
    new_vars_prefix = "EXST"
  ) %>%
  derive_vars_dtm(
    dtc = EXENDTC,
    new_vars_prefix = "EXEN",
    time_imputation = "last"
  )
#> The default value of `ignore_seconds_flag` will change to "TRUE" in admiral
#> 1.4.0.

adsl <- adsl %>%
  derive_vars_merged(
    dataset_add = ex_ext,
    filter_add = (EXDOSE > 0 |
      (EXDOSE == 0 &
        str_detect(EXTRT, "PLACEBO"))) & !is.na(EXSTDTM),
    new_vars = exprs(TRTSDTM = EXSTDTM, TRTSTMF = EXSTTMF),
    order = exprs(EXSTDTM, EXSEQ),
    mode = "first",
    by_vars = exprs(STUDYID, USUBJID)
  ) %>%
  derive_vars_merged(
    dataset_add = ex_ext,
    filter_add = (EXDOSE > 0 |
      (EXDOSE == 0 &
        str_detect(EXTRT, "PLACEBO"))) & !is.na(EXENDTM),
    new_vars = exprs(TRTEDTM = EXENDTM, TRTETMF = EXENTMF),
    order = exprs(EXENDTM, EXSEQ),
    mode = "last",
    by_vars = exprs(STUDYID, USUBJID)
  )
```

SDTM, ADaM & TLG Programming

TLG

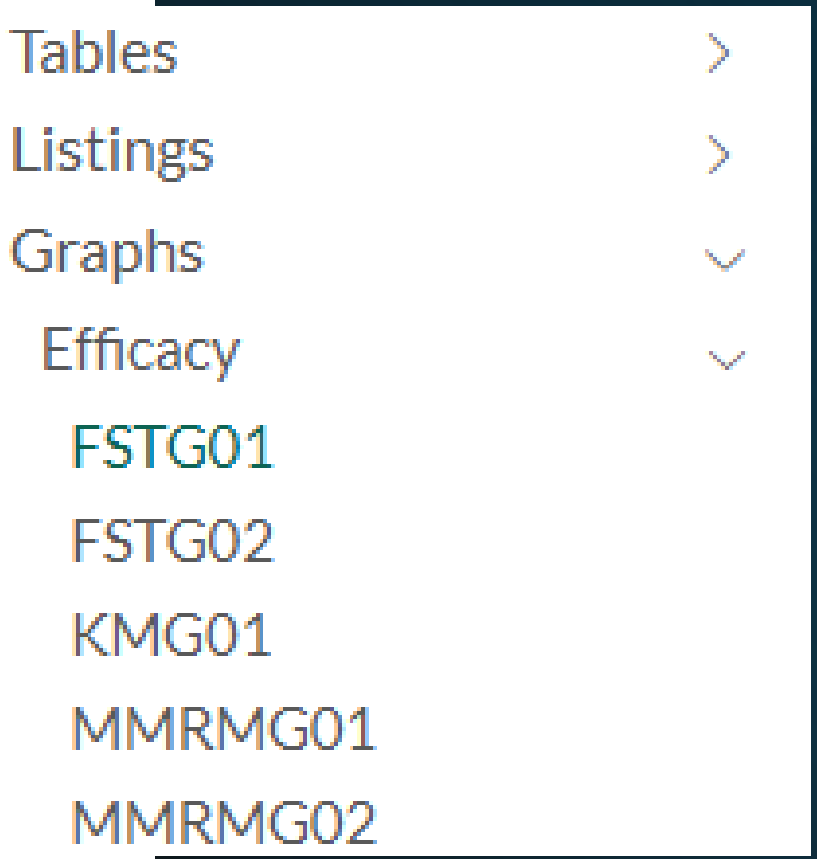
- TLGs, especially graphs, are where R excels (ggplot2)
- A multitude of packages are available.
- Pharmaverse lists some of these packages that can be used. Image right is non-extensive
- There may be packages not listed such as specific statistical analysis or specific formatting



SDTM, ADaM & TLG Programming

TLG

- TLG Catalog provides many template outputs.
- A combination of the TLG catalog, SAS templates, pharmaverse examples & R templates were used as a basis.
- All were modified to be as close to Astellas standards as possible.



Graphs > Efficacy > FSTG01

FSTG01

Subgroup Analysis of Best Overall Response

Output

Standard Plot | Plot Specifying Class Variables and Options for the Treatment Variable | Plot Selecting Columns and Changing the Alpha Level | Plot with Fixed Symbol Size | Plot of CR Only, Setting Values Indicating Response | Data Setup

Preview | Try this using WebR

▶ Code

		B: Placebo			A: Drug X				
Baseline Risk Factors	Total n	n	Responders	Response (%)	n	Responders	Response (%)	Odds Ratio	95% CI
All Patients	268	134	84	62.7%	134	100	74.6%	1.76	(1.04, 2.97)
Sex									
F	161	82	48	58.5%	79	61	77.2%	2.45	(1.23, 4.86)
M	107	52	36	69.2%	55	39	70.9%	1.08	(0.48, 2.46)
Categorical Level Biomarker 2									
LOW	95	45	30	66.7%	50	39	78.0%	1.77	(0.71, 4.39)
MEDIUM	93	56	34	60.7%	37	27	73.0%	1.73	(0.71, 4.22)
HIGH	80	33	20	60.6%	47	34	72.3%	1.73	(0.67, 4.44)

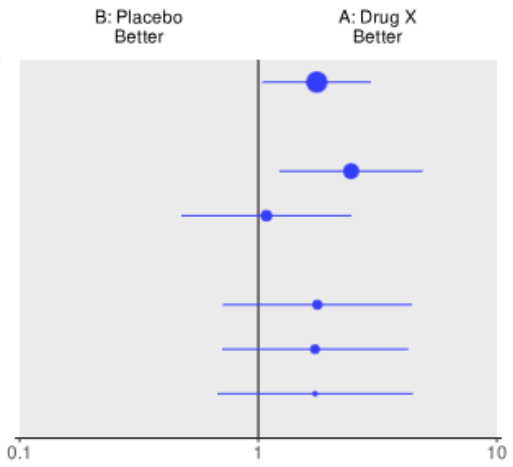


Image: screenshot of a template output in the TLG catalogue using test data.

Preview

Try this using WebR

Code

```r
df <- extract_rsp_subgroups(
  variables = list(
    rsp = "is_rsp",
    arm = "ARM",
    subgroups = c("SEX", "BMRKR2"),
    strata = "STRATA2"
  ),
  data = anl_rsp_arms_ab,
  conf_level = 0.95
)

result <- basic_table() %>%
  tabulate_rsp_subgroups(df, vars = c("n_tot", "n", "n_rsp", "prop", "or", "ci"))

plot <- g_forest(tbl = result)
plot
```

		B: Placebo			A: Drug X				
Baseline Risk Factors	Total n	n	Responders	Response (%)	n	Responders	Response (%)	Odds Ratio	95% CI
All Patients	268	134	84	62.7%	134	100	74.6%	1.76	(1.04, 2.97)
Sex									
F	161	82	48	58.5%	79	61	77.2%	2.45	(1.23, 4.86)
M	107	52	36	69.2%	55	39	70.9%	1.08	(0.48, 2.46)
Categorical Level Biomarker 2									
LOW	95	45	30	66.7%	50	39	78.0%	1.77	(0.71, 4.39)

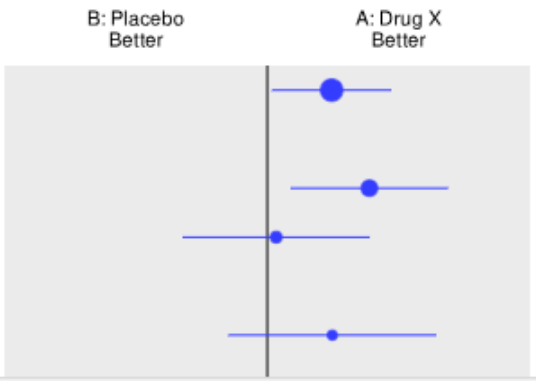


Image: screenshot of a template output in the TLG catalogue using test data.

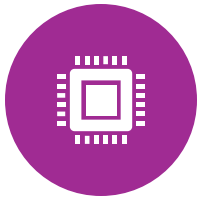
Conclusion

Conclusion

This is the best time to embark on E2E programming R

Plenty of resources available for personal development

Pharmaverse is in a good condition



Setup of infrastructure can challenging,

Outputs successfully delivered on this study.

We now have the flexibility to use both R & SAS.

Thank You

Any questions?

Acknowledgements

- Astellas 5502-CL-0001 statistics/programming team (Anna Spence & Jing Zhu)
- Astellas Clinical Data and Information Strategy team
- My Manager (Mary Beth Blauwet)
- Open Source Stream leads

Contact Information

EMAIL:

LUKE.THOMPSON@ASTELLAS.COM