PharmaSUG 2024 - Paper SD-426

Shift gears with 'gt': Finely tuned clinical reporting in R using "gt" and "gt summary" packages

Raj Chennamaneni, Sudhir Kedare and Jagan Mohan Achi, Jazz Pharmaceuticals Inc

ABSTRACT

Efforts are underway to use open-source technologies like R and Python for FDA regulatory submission. Based on ongoing initiatives with FDA, there is a strong likelihood that FDA will embrace regulatory submissions utilizing open-source technologies along with SAS®. There are numerus existing and emerging packages in R for presenting data in tabular format. This paper will focus on {gt} and its extension {gt summary} package. "gt" package framework uses table header, stub, column labels, spanner column labels, table body and table footer components to create summary reports. We will use these components to demonstrate the ease and flexibility of developing clinical reports. Furthermore, we will leverage "gt summary package" to generate descriptive statistic tables, efficacy outputs, inline tables, and a few custom tables.

INTRODUCTION

With the growing popularity of open-source technologies, there is an increasing interest in utilizing R for clinical reporting. This shift extends from static to dynamic reporting, enabling early detection of trends in clinical results during study conduct. Several R packages are available for clinical reporting, including {kable}, {kableExtra}, {formatable}, {gt}, and {gtsummary}. These packages offer varying levels of development simplicity, customization options, and output formats. In this article, we focus on the {gt} and {gtsummary} packages with the assumption that readers have fundamental understanding of R programming and familiarity with the {dplyr} and {tidyverse} packages. Additional information about these packages can be found in the referenced sources. Mastery of the principles underlying the {gt} and {gtsummary} packages empowers users to construct interactive and dynamic reports in R.

{gt} Package

The {gt} package design revolves around five primary table components: Table header, Stub head, Stub, Footnotes, and Source notes. Each of these components comprises subcomponents that offer customization options. Constructing tables with the {gt} package has a three-step approach as outlined below in Table 1. First, generate a table object containing summary statistics. Second, utilize various formatting options to craft a "gt" object. Finally, export the formatted "gt" object to the desired output.

Reporting in gt		
Table Object	“gt” Object	HMTL RTF LaTeX Docx
Input		*Output*

Table 1. Input and Output in {gt} package

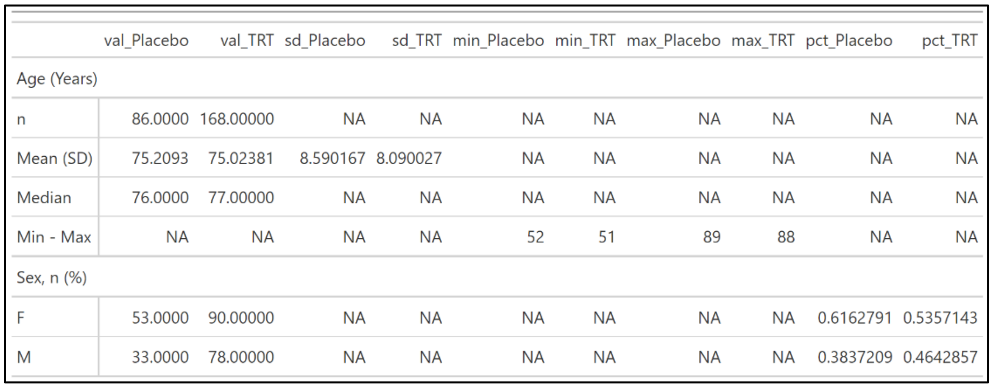
STEP 1: CREATING A TABLE OBJECT

This step involves creating a data frame with desired summary statistics/display values. This can be easily done with {dplyr} and {tidyerse} packages in combination with base R functions. We can use advanced concepts in R like metaprogramming to automate. Examples of creating table objects are found in reference material.

STEP 2: CREATING A “GT” OBJECT

To generate a "gt" object, we utilize the gt() function. This function requires input "data", alongside a few additional optional parameters. Among these, the "rowname_col" and "groupname_col" options facilitate the conversion of row and column names into row and column labels, respectively. Table 2 illustrates a fundamental "gt" object produced from a dataset containing summary statistics obtained in Step 1.

```
gt(
  data,
  rowname_col =
  groupname_col =
…..
)
```



	val_Placebo	val_TRT	sd_Placebo	sd_TRT	min_Placebo	min_TRT	max_Placebo	max_TRT	pct_Placebo	pct_TRT
Age (Years)										
n	86.0000	168.00000	NA	NA	NA	NA	NA	NA	NA	NA
Mean (SD)	75.2093	75.02381	8.590167	8.090027	NA	NA	NA	NA	NA	NA
Median	76.0000	77.00000	NA	NA	NA	NA	NA	NA	NA	NA
Min - Max	NA	NA	NA	NA	52	51	89	88	NA	NA
Sex, n (%)										
F	53.0000	90.00000	NA	NA	NA	NA	NA	NA	0.6162791	0.5357143
M	33.0000	78.00000	NA	NA	NA	NA	NA	NA	0.3837209	0.4642857

Table 2. “gt” Object with summary statistics

The "gt" object can be customized using various built-in functions. Utilizing "tab_*" functions allows for making general modifications.

a. Titles and footnotes can be added using “tab_header: and “tab_footnote” functions. Table 3 shows the output with titles and footnotes.

```
tab_header(
  title = …,
  subtitle = …,
    …
)

tab_footnote(
  footnote = …,
  locations = NULL,
  placement = c("auto"),
    …
)
```

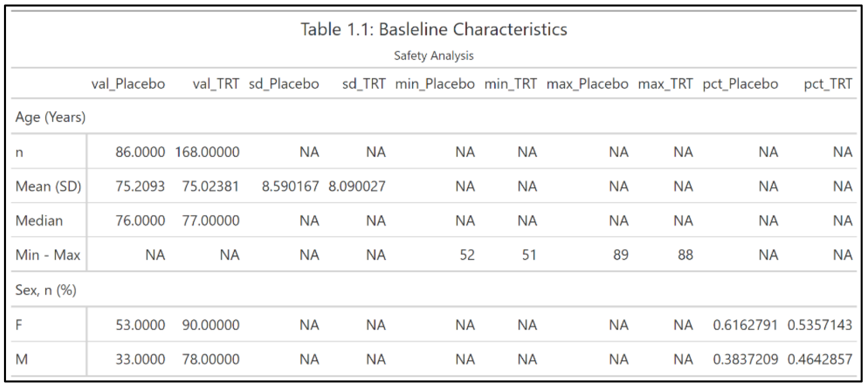


Table 1.1: Basleline Characteristics

Safety Analysis

	val_Placebo	val_TRT	sd_Placebo	sd_TRT	min_Placebo	min_TRT	max_Placebo	max_TRT	pct_Placebo	pct_TRT
Age (Years)										
n	86.0000	168.00000	NA	NA	NA	NA	NA	NA	NA	NA
Mean (SD)	75.2093	75.02381	8.590167	8.090027	NA	NA	NA	NA	NA	NA
Median	76.0000	77.00000	NA	NA	NA	NA	NA	NA	NA	NA
Min - Max	NA	NA	NA	NA	52	51	89	88	NA	NA
Sex, n (%)										
F	53.0000	90.00000	NA	NA	NA	NA	NA	NA	0.6162791	0.5357143
M	33.0000	78.00000	NA	NA	NA	NA	NA	NA	0.3837209	0.4642857

Table 3. "gt" Object with tab headers

b. The columns can be further customized using "fmt *" functions and "col_*" functions. Table 4 shows the resulting output after utilizing "fmt_percent", "fmt_number", "col_width", "cols_align" functions.

```
fmt_integer(
    columns = ...,
    rows    = ...,
  …
  )


  fmt_percent(
    columns  = ...,
    decimals = ...,
  …

  )

  cols_merge(
    columns = ...,
    pattern = ...,
  …

  )
```

cols_label(.list = ...)

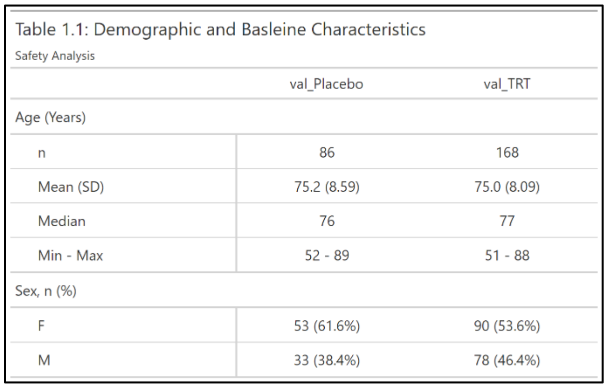


Table 1.1: Demographic and Basleline Characteristics

Safety Analysis

	val_Placebo	val_TRT
Age (Years)		
n	86	168
Mean (SD)	75.2 (8.59)	75.0 (8.09)
Median	76	77
Min - Max	52 - 89	51 - 88
Sex, n (%)		
F	53 (61.6%)	90 (53.6%)
M	33 (38.4%)	78 (46.4%)

Table 4. "gt" Object with fmt_* and col_* functions

c. We can save the created object to a file using “gtsave” function. “gt” supports HTML, PDF, PNG, LaTex and RTF formats.

gt::gtsave(filename = "....ext").

{gtsummary} Package

“gtsummary” package builds on {gt} , {broom} and {labelled} packages and generates publication ready tables on the fly. Rather than building from individual table components, {gt summary} provides integrated functions that obviates the need to build tables from scratch. We will cover summary tables, efficacy tables and inline and custom tables.

SUMMARY TABLES

In “gtsummary” package, tbl_summary is the core function that generates descriptive statistics for continuous and categorical variables as shown in Table 5. This function has four summary types: continuous, continuos2, categorical and dichotomous. Continuous type shows summary statistics horizontally, while continuous 2 type displays vertically in more than one row.

```
descrp_table <-
  dataset %>%
  select(...) %>%
  tbl_summary(
    by = ...,
    type = all_continuous() ~ "....,
    statistic = all_continuous() ~ c("{N_obs}", "{mean} ({sd})",
"{median}", "{min}, {max}"),
    missing = "no"
    …
  )
```

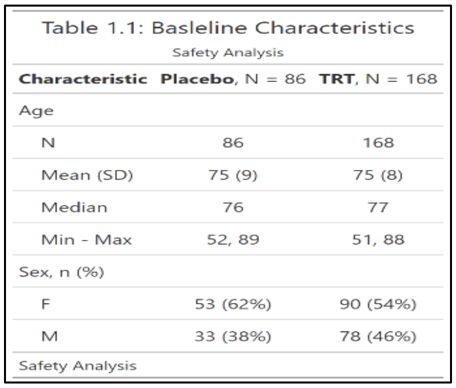


Table 1.1: Basleline Characteristics

Safety Analysis

Characteristic	**Placebo**, N = 86	**TRT**, N = 168
Age		
N	86	168
Mean (SD)	75 (9)	75 (8)
Median	76	77
Min - Max	52, 89	51, 88
Sex, n (%)		
F	53 (62%)	90 (54%)
M	33 (38%)	78 (46%)

Safety Analysis

Table 5. “gtsummary” table report

EFFICACY TABLES

“tbl_survfit” function handles survival analyses data in “gtsummary” packages. “add_p”, “add_n” and “add_nevent” functions can be used to add p-values and columns with number of observations and events as shown in Table 6.

```
eff_table <- tbl_survfit (
  survfit (Surv(..., ...) ~ ...., adtte_xpt),
  times = c (..., ...),
  label_header = "**{time} Years **",
  …
)
```

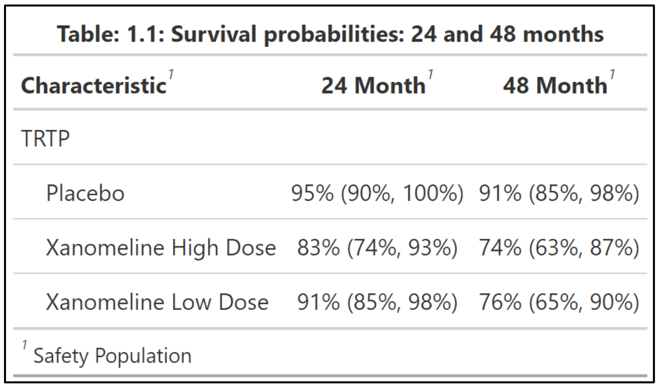


Table: 1.1: Survival probabilities: 24 and 48 months

Characteristic[1]	24 Month[1]	48 Month[1]
TRTP		
Placebo	95% (90%, 100%)	91% (85%, 98%)
Xanomeline High Dose	83% (74%, 93%)	74% (63%, 87%)
Xanomeline Low Dose	91% (85%, 98%)	76% (65%, 90%)

[1] Safety Population

Table 6. "gtsummary" Efficacy Report

IN-TEXT -CUSTOM SUMMARY

{gt summary} provides inline_text function for reporting in documents for summary and regression outputs.

```
        inline_text( … variable = ..., level = "...", column = "...", pattern
      = "....").
```

For more flexible reports, we can use tlbl_custom_summary function where we define our custom function outside the call to tbl_custom_summary and pass the argument to "stat_fins" parameter.

```
        tbl_custom_summary(
          include = c(..., ...),
          by = ....,
          stat_fns = everything() ~ ...,
          statistic = everything() ~ ...,
          digits = everything() ~...,
          overall_row = ...,
          overall_row_label = ...
        )
```

CONCLUSION

We have explored how "gt" and "gtsummary" can be leveraged to generate exceptionally flexible and dynamic reports. The concepts outlined in this paper represent just a portion of the extensive feature sets offered by both packages. By tapping into these additional features, users can develop customized reports tailored to their specific requirements, and seamlessly integrate them with emerging automated solutions.

REFERENCES

Sjoberg DD, Whiting K, Curry M, Lavery JA, Larmarange J. Reproducible summary tables with the gtsum mary package.The R Journal 2021;13:570–80. https://doi.org/10.32614/RJ-2021-053.

Lannone R, Cheng J, Schloerke B, Hughes E, Lauer A, Seo J, Brevoort K (2024). *gt: Easily Create* Present ation-*Ready Display Tables*. R package version 0.10.1.9000, https://github.com/rstudio/gt, https://gt.rstudio. com.

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L., François, R., Grolemund, G., Hayes, A., H enry, L., Hester, J., Kuhn, M., Pedersen, T., Miller, E., Bache, S., Müller, K., Ooms, J., Robinson, D., Seid el, D., Spinu, V.,Yutani, H. (2019). Welcome to the tidyverse. Journal of Open Source Software, 4(43), 16 86.

ACKNOWLEDGMENTS

We thank the Jazz Pharmaceutical Data Science team for their support and guidance in the preparation of this paper.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Raj Chennamaneni
rchennamaneni@jazzpharma.com

Sudhir Kedare
sudhir.kedare@jazzpharma.com