PharmaSUG 2020 - Paper DV-164

Using R Markdown to Generate Clinical Trials Summary Reports

Radhika Etikala, Xuehan Zhang (Emily), SCHARP at Fred Hutch, Seattle, Washington

ABSTRACT

The scope of the paper is to show how to produce a statistical summary report along with explanatory text using R Markdown in RStudio. Programmers write a lot of reports that describe the results of data analyses. There should be a clear and automatic path from data and code to the final report. R Markdown is ideal for this as it is a system for combining code and text into a single document. It is also an efficient, user-friendly tool for producing reports that do not need constant updating. RStudio is often used in the Pharmaceutical and Healthcare industries for analysis and data visualization, and the R Markdown tool can also be leveraged for creating reports and datasets for submission to regulatory agencies.

This paper presents an RStudio program that demonstrates how to use R Markdown to generate a statistical table showing adverse events (AE) by system organ class (or preferred term) and severity grade along with text that explains the table. Collecting AE data and performing analysis of AEs is a common and critical part of Clinical Trials. A well-developed reporting system such as one generated with R Markdown, provides a solid foundation and an efficient approach towards a better understanding of what the data represent.

INTRODUCTION

R Markdown is a powerful tool within RStudio and is very flexible. It allows you to write documents that combine written text with analytical code. R Markdown consolidates code and document into a single file. This is important in order to ensure reproducibility. An added benefit is figures and tables get automatically updated if the underlying parameters in the code change.

One can use a single R Markdown file to save and execute code as well as generate high-quality reports. The text in the document can be fully formatted in a report style. If necessary, code can be made visible or not, and documents can be output as PDFs, Word Documents, HTML, and other formats. It provides an authoring framework for data science, which is widely used in generating reports with narration. R Markdown is easy to use and easy to learn; moreover, it is a free and open source. Training materials of all levels are available online.

The purpose of this paper is to demonstrate an approach to create summary reports that can have a large impact on programmer's daily work. It shows some common R Markdown techniques and how the documentation could be used. In addition, this paper presents an R Markdown program that describes the step-by-step process, packages and functions to generate the summary report containing narrative details.

R MARKDOWN PAKAGES AND FUNCTIONS

Here is how you can use R Markdown to create a summary report to include the list of Packages, functions, and options used in the program with the description.

PACKAGES

PACKAGE	Description
knitr	Package for dynamic report generation R
kableExtra	To build tables and manipulate table styles
sas7bdat	SAS Database Reader

PACKAGE	Description
markdown	Markdown rendering for R
rmarkdown	R Markdown Document Conversion
haven	Import and export 'SPSS', 'STATA' and 'SAS' files
readxl	Reads Excel files
devtools	Tools to make developing R packages Easier
xtable	Export tables to LaTex or HTML
data.table	Expression of 'data.frame'
plyr	Tools for Splitting, Applying and Combining data
tidyverse	Makes data science faster, easier and more fun
glue	Interpreted string litterals
janitor	Simple tools for examining and cleaning the dirty data

FUNCTIONS AND OPTIONS

FUNCTION/ OPTION	Description
include = FALSE	prevents code and results from appearing in the finished file.
echo = FALSE	prevents code but not the results from appearing in the finished file.
message = FALSE	prevents messages that are generated by code from appearing in the finished file.
warning = FALSE	prevents warnings that are generated by code from appearing in the finished.
fig.cap = "..."	adds a caption to graphical results.
message = FALSE	prevents messages that are generated by code from appearing in the finished file.
results="hide"	hide the results/output (but here the code would still be displayed).

STEP-BY-STEP PROCESSING

The AE summary report is generated using a derived adverse events dataset, which is an organization-specific standard dataset configuration to include CDISC standards.

STEP 1: FILL IN THE HEADER

The R Markdown file contains a header section. Below shows an example of a header. The date can be generated dynamically by quoting the inline R expression. If you have a problem installing the 'tinytex', use "header-includes:" as shown below in the YAML header.

A YAML header embedded by ---s.

```
---
title: "Table 1"
Author: <optional>
Date: <optional>
output: pdf_document: default
          latex_engine: xelatex
header-includes:
   - \usepackage{booktabs}
```

```
   - \usepackage{float}
   - \usepackage{colortbl}
   - \usepackage[table]{xcolor}
   - \usepackage[justification=centering,font=bf]{caption} # this will bold
the captions
   - \captionsetup[table]{labelsep=space}
   - \usepackage{sectsty} \sectionfont{\centering}
   - \usepackage{fontspec}
---
```

STEP 2: READING THE DATA

Next steps include setting up the libraries, options, installing packages, and reading the data. Executable code should be placed between the chunk delimiters ``` and ```.

Installing the necessary libraries, functions is the top benefits of using R Markdown. To successfully generate a file template, we need to install necessary packages. This can be done by the function "Install.packages ()". Description of each installed package is given above in the package section.

````
```{r echo=FALSE, eval=TRUE}

library(knitr)
# Set some knitr options
# This will always generate two figures (pdf and png, which can be handy
for presentations)
opts_chunk$set(tidy = TRUE, cache = FALSE, messages = FALSE, warning =
FALSE, echo = FALSE, dev = c("pdf", "png"), dpi = 200)

```

```{r setup, echo=FALSE}

# Load necessary packages

# More options could be added later in the template

#install.packages("data.table", type="source", dependencies=TRUE)

#install.packages("~/sas7bdat_0.1.tar.gz", repos = NULL, type ="source")

#install.packages('kableExtra')

suppressPackageStartupMessages({
  library(sas7bdat)
  library(devtools)
  library(xtable)
  library(data.table)
  library(plyr)
  library(knitr)
  library(markdown)
  library(rmarkdown)
  library(haven)
  library(readxl)
````
````

```
  library(kableExtra)
  library(tidyverse)
  library(glue)
  library(janitor) })

```

STEP3: CREATING THE USER DEFINED FUNCTION

Rounding function that round X.5 to the higher whole number and -X.5 to the lowest whole number.

```
```{r echo=FALSE}

# Defining the functions.

comcat <- function(X) do.call(paste, c(as.list(X), sep="','"))

# FUNCTION: RoundUp()
# Arguments   :
#           x : numbers to round (vector)
#      digits : digits to round (0 to 11) (scaler)
#  tol_digits : tol_digits for noise to add to x (1 to 12) (scaler)
# Details     :
#         Rounding function that round X.5 to the higher whole number.

RoundUp <- function(x, digits = 0, tolerance_digits = 12){
  if (digits < 0 | digits > 11) {
    stop('digits must between 1 and 11')
  }
  if (tolerance_digits < digits) {
    stop('tolerance_digits must be larger than digits')
  }
  sign(x) * round(abs(x) + 10 ^ -tolerance_digits, digits)
}

```
```

STEP 4: SUMMARIZING THE DATA

The code below is reading in the raw data, sub-setting the data, and counting the number of participants by body system, preferred term, and by severity grade.

Variables used from Legacy ADAE dataset are: AEsoc ,AEmdra, AEseve_txt, ptid, AEseve. These variables are used to filter respective condition to obtain counts. RStudio is case sensitive so pay attention to variable names and datasets names.

```
```{r echo=FALSE}

# Summarizing the data.
# Reading the data

file_name <- "adae"
```
```

```
adata_path_test <- "/H:/R/PharmaSUG/"
adae <- data.table(read.sas7bdat(paste0(adata_path_test, "adae.sas7bdat")))

# counting the enrolled PTIDS
enr <- read_sas("/H:/R/PharmaSUG/enr.sas7bdat")
enr_num <- length(unique(enr$ptid))

# Checking for 0 row dataset
if (nrow(adae) == 0) {
  # Writing a warning if file is missing

  warning(paste0('No observations in',file_name,' dataset'))

} else {

    # Creating AEseve_txt variable labels

  adae[, AEseve_txt := factor(AEseve, levels = 1:5, labels = c('Mild',
'Moderate', 'Severe', 'Life-threatening', 'Death'))]

  # Getting Total severity Category

  adae_with_total <- rbindlist(list(adae, adae))
  adae_with_total[1:nrow(adae), AEseve_txt := 'Total']

  # resetting factor order

  adae[, `:=`(AEsoc = factor(as.vector(AEsoc)), AEmdra =
factor(as.vector(AEmdra)))]

 # counting one or more Adverse Events per severity Category

  results_ptid_level <- adae_with_total[, .(AEsoc = 'Participants with one
or more AEs',
    Info = paste0(length(unique(ptid)) , '  (', formatC(RoundUp(100 *
length(unique(ptid)) / unique(enr_num), 1), digits = 1 , format = "f"),
'\\%)')
  ), by = .(AEseve_txt)]

  # Note there may be multiple people per severity Category

  results_AEsoc_level <- adae_with_total[, .(
    Info = paste0(length(unique(ptid)) , ' (', formatC(RoundUp(100 *
length(unique(ptid)) / unique(enr_num), 1), digits = 1 , format = "f"),
'\\%)')
  ), by = .(AEsoc,AEseve_txt)]

  results_AEmdra_level <- adae_with_total[, .(
    Info = paste0(length(unique(ptid)) , ' (', formatC(RoundUp(100 *
length(unique(ptid)) / unique(enr_num), 1), digits = 1 , format = "f"),
'\\%)')
  ), by = .(AEsoc ,AEmdra, AEseve_txt)]

  results_long <- rbindlist(list(results_ptid_level, results_AEsoc_level,
results_AEmdra_level), use.names = TRUE, fill = TRUE)
```

```
  results_long <- melt.data.table(results_long, id.vars = c('AEsoc',
'AEmdra', 'AEseve_txt'))

  # Creating Results for each severity Category and total severity Category

  results <- dcast.data.table(results_long, AEsoc + AEmdra ~ AEseve_txt,
value.var = 'value',)
  results[, AEmdra_soc := AEmdra]
  results[is.na(AEmdra), AEmdra_soc := AEsoc]
  results[is.na(AEmdra), AEmdra := '']

  if (all(names(results) != 'Mild')) results[, `Mild` := NA_character_]
  if (all(names(results) != 'Moderate')) results[, `Moderate` :=
NA_character_]
  if (all(names(results) != 'Severe')) results[, `Severe` := NA_character_]
  if (all(names(results) != 'Life-threatening')) results[, `Life-
threatening` := NA_character_]
  if (all(names(results) != 'Death')) results[, `Death` := NA_character_]

  # Replacing NA results with 0 (0.0\\%)

  results[is.na(`Mild`), `Mild` := '0 (0.0\\%)']
  results[is.na(`Moderate`), `Moderate` := '0 (0.0\\%)']
  results[is.na(`Severe`), `Severe` := '0 (0.0\\%)']
  results[is.na(`Life-threatening`), `Life-threatening` := '0 (0.0\\%)']
  results[is.na(`Death`), `Death` := '0 (0.0\\%)']
  results[is.na(`Total`),     `Total` := '0 (0.0\\%)']

  # Sorting the data by SOC and PT

  (sorted <- order(results$AEsoc, results$AEmdra, na.last = FALSE));
results[sorted, ]

  results_sorted <- setDT(results)[, indx := AEsoc][, .SD[1:(.N+1)],
indx][,indx := NULL][!.N]

  results_sorted[, AEsoc := NULL][, AEmdra := NULL]

  # replacing "NA" with spaces

  results_sorted[is.na(`AEmdra_soc`), `AEmdra_soc` := ' ']
  results_sorted[is.na(`Mild`), `Mild` := ' ']
  results_sorted[is.na(`Moderate`), `Moderate` := ' ']
  results_sorted[is.na(`Severe`), `Severe` := ' ']
  results_sorted[is.na(`Life-threatening`), `Life-threatening` := ' ']
  results_sorted[is.na(`Death`), `Death` := ' ']
  results_sorted[is.na(`Total`),     `Total` := ' ']

  #re-ordering the columns using 'setcolorder'

  setcolorder(results_sorted, c('AEmdra_soc', 'Mild', 'Moderate', 'Severe',
'Life-threatening', 'Death', 'Total'))

  write.csv(results_sorted, file = paste0(adata_path_test,
't_ae_sev_grade_test.csv'), row.names = FALSE)
  }
```

STEP 5: INSERTING THE NARRATIVES

To make it more reproducible, it is much better to combine both code and explanations. Displaying a block of text (Explanation of Tables) at the beginning of the report.

```
## EXPLANATION OF IND ANNUAL REPORT TABLES

Table 1:   Adverse Experiences (AEs) by Body System/Preferred Term and
Severity

* This table shows the number of participants reporting AEs by MedDRA
preferred term, body system, and severity grade.
* The first row of the table, 'Participants with one or more AE,' shows the
number of participants reporting at least one AE categorized by the highest
severity experienced.
* Body systems are sorted alphabetically and MedDRA preferred terms are
sorted alphabetically within a body system.
* If a participant reports more than one AE for the row, the participant is
counted in the row once at the highest grade reported.
* Participants with multiple AEs within a body system are counted once in
the body system row.
* All percentages use the number of enrolled participants as the
denominator.
* Data are from the adverse experience CRF.
```

STEP 6: GENERATING SUMMARY TABLE

The code below takes the summary data created up until now and generates a table with customized styles. This code generates a table using knitr's 'kableExtra' function. If you have problem using "latex" format while generating PDF output, run the prebuilt code below. No need to Install 'tinytex', if you have 'texlive' installed on your computer.

```
```{r echo=FALSE}

# GENERATING SUMMARY TABLE

tinytex::install_tinytex()
tinytex:::install_prebuilt()
AEmdra_v <- as.vector(adae$AEmdra)
AEsoc_v <- as.vector(adae$AEsoc)

date<-format(Sys.time(), '%B %d, %Y')
row_num <- which(results_sorted$AEmdra_soc %in% AEmdra_v)
b_row_num <- which(results_sorted$AEmdra_soc %in% AEsoc_v)
#options(tinytex.verbose = TRUE)
ae_sev_t <- kable(results_sorted, longtable=T, booktabs=T, align='lcccccc',
escape=F,
                  caption=paste("\\\\ADVERSE EXPERIENCES (AEs) BY BODY
SYSTEM/PREFERRED TERM AND SEVERITY\\\\Data as of",
                                format(Sys.time(), '%B %d, %Y'),
                                "\\\\Number of Enrolled Participants = ",
                                enr_num),
                  col.names = linebreak(c("System Organ Class/Preferred
Term",
```
```

```
                                                "n \\%", "n \\%", "n \\%", "n
\\%", "n \\%", "n \\%"),
                                              align = "c")
                    ) %>%
  kableExtra::add_indent(row_num) %>%
  add_header_above(c("", "Mild", "Moderate", "Severe", "Potentially \n
Life- \n Threatening", "Death", "Total"), line=F, bold = T) %>%
  add_header_above(c("", "Maximum Severity Grade"=6), line=F, bold = T) %>%
  kableExtra::kable_styling(latex_options = c("repeat_header"),font_size
=7.5,position = "l") %>%
  column_spec(1, width = "6cm") %>%
  row_spec(b_row_num,bold=TRUE) %>%
  row_spec(0:1,bold=TRUE)

 ae_sev_t

```

STEP 7: RUN THE CODE IN BATCH

RStudio includes pandoc; you just need to add the relevant directory to your PATH.
Mac: /Applications/RStudio.app/Contents/MacOS/pandoc
Windows: "c:\Program Files\RStudio\bin\pandoc"

To use the R Markdown package from the command line, you need access to pandoc. But if you've installed RStudio, you just need to add the relevant directory (listed above) to your PATH. For example in your ~/.bash_profile file. At the command line, type "pandoc" or "pandoc –version" to check that it's available.

Here is the command to run R Markdown code in batch on linux.

```
R -e "rmarkdown::render('script.Rmd',output_file='summary_AEs_grade.pdf')"
```

FINAL AE SUMMARY REPORT BY R MARKDOWN

Attaching the screenshot of AE summary report generated by the R Markdown

EXPLANATION OF REPORT

Table 1: Adverse Experiences (AEs) by Body System/Preferred Term and Severity

- This table shows the number of participants reporting AEs by MedDRA preferred term, body system, and severity grade.
- The first row of the table, Participants with one or more AE, shows the number of participants reporting at least one AE categorized by the highest severity experienced.
- Body systems are sorted alphabetically and MedDRA preferred terms are sorted alphabetically within a body system.
- If a participant reports more than one AE for the row, the participant is counted in the row once at the highest grade reported.
- Participants with multiple AEs within a body system are counted once in the body system row.
- All percentages use the number of enrolled participants as the denominator.
- Data are from the adverse experience CRF.

Table 1
ADVERSE EXPERIENCES (AEs) BY BODY SYSTEM/PREFERRED TERM AND SEVERITY
Data as of April 20, 2020
Number of Enrolled Participants = 42

	Maximum Severity Grade					
	Mild	Moderate	Severe	Potentially Life-Threatening	Death	Total
System Organ Class/Preferred Term	n %	n %	n %	n %	n %	n %
Participants with one or more AEs	**5 (11.9%)**	**16 (38.1%)**	**2 (4.8%)**	**0 (0.0%)**	**0 (0.0%)**	**23 (54.8%)**
Congenital, familial and genetic disorders	**1 (2.4%)**	**0 (0.0%)**	**0 (0.0%)**	**0 (0.0%)**	**0 (0.0%)**	**1 (2.4%)**
Dermoid cyst	1 (2.4%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	1 (2.4%)
Gastrointestinal disorders	**0 (0.0%)**	**2 (4.8%)**	**0 (0.0%)**	**0 (0.0%)**	**0 (0.0%)**	**2 (4.8%)**
Abdominal discomfort	0 (0.0%)	1 (2.4%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	1 (2.4%)
Flatulence	0 (0.0%)	1 (2.4%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	1 (2.4%)
Gastrooesophageal reflux disease	0 (0.0%)	1 (2.4%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	1 (2.4%)
General disorders and administration site conditions	**1 (2.4%)**	**1 (2.4%)**	**0 (0.0%)**	**0 (0.0%)**	**0 (0.0%)**	**2 (4.8%)**
Fatigue	1 (2.4%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	1 (2.4%)
Influenza like illness	0 (0.0%)	1 (2.4%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	1 (2.4%)

SAS PROC REPORT OUTPUT (.PDF)

Below attached is the screenshot of the AE summary report generated by the SAS proc report. It shows that we can achieve similar results using R Markdown, with the advantage of combining results and text into one document.

TABLE 1
ADVERSE EXPERIENCES (AEs) BY BODY SYSTEM/PREFERRED TERM AND SEVERITY
Data as of April 18, 2020

Number of Enrolled Participants = 42

	Maximum Severity Grade					
	Mild	Moderate	Severe	Potentially Life-Threatening	Death	Total
System Organ Class/Preferred Term	n %	n %	n %	n %	n %	n %
Participants with one or more AEs	**5 11.9%**	**16 38.1%**	**2 4.8%**	**0 0.0%**	**0 0.0%**	**23 54.8%**
Congenital, familial and genetic disorders	**1 2.4%**	**0 0.0%**	**0 0.0%**	**0 0.0%**	**0 0.0%**	**1 2.4%**
Dermoid cyst	1 2.4%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 2.4%
Gastrointestinal disorders	**0 0.0%**	**2 4.8%**	**0 0.0%**	**0 0.0%**	**0 0.0%**	**2 4.8%**
Abdominal discomfort	0 0.0%	1 2.4%	0 0.0%	0 0.0%	0 0.0%	1 2.4%
Flatulence	0 0.0%	1 2.4%	0 0.0%	0 0.0%	0 0.0%	1 2.4%
Gastrooesophageal reflux disease	0 0.0%	1 2.4%	0 0.0%	0 0.0%	0 0.0%	1 2.4%
General disorders and administration site conditions	**1 2.4%**	**1 2.4%**	**0 0.0%**	**0 0.0%**	**0 0.0%**	**2 4.8%**
Fatigue	1 2.4%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 2.4%
Influenza like illness	0 0.0%	1 2.4%	0 0.0%	0 0.0%	0 0.0%	1 2.4%

CONCLUSION

The programmatic steps presented in this paper provides a useful way to generate a report with an explanation of table. Using code, like above in R Markdown, can produce consistent, reproducible, efficient, and high-quality summary reports without an increase in cost. And it can save time too. The ability of R Markdown to create clinical trial reports can be looked at as a cost-effective alternative compared to existing methods.

REFERENCES

R Markdown: Definitive Guide: https://bookdown.org/yihui/rmarkdown/word-document.html

R Markdown (R Studio): https://rstudio.com/

RECOMMENDED READING

R Markdown from R Studio

R Markdown Reference Guide

Introduction to summary tools: https://cran.r-project.org/web/packages/summarytools/vignettes/Introduction.html

http://haozhu233.github.io/kableExtra/awesome_table_in_html.html

https://rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf

ACKNOWLEDGMENTS

I would like to thank Anthony Williams, Kobie O'Brian, Julie Stofel and Paul Stutzman for their guidance and review.

CONTACT INFORMATION

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

Author Name: Radhika Etikala
Company: Statistical Center for HIV/AIDS Research & Prevention (SCHARP) at Fed Hutch
Email: retikala@scharp.org