PharmaSUG China 2021 - Paper DV-005

Review Response Data with an R Shiny App

Hengwei Liu, Hengrui USA

ABSTRACT

In solid tumor studies the overall response at each cycle is based on the target response, non-target response and new lesion progression. The target response is determined from the sum of diameters for target lesions. In this paper we propose to create an R shiny app which requires the users to input the response data, the tumor data and a CSV file that contains the conditions to be used to get a subset of the response and tumor data. The app produces a plot for the target response, non-target response, new lesion progression and overall response over time. It also produces line plots for the sum of diameters, the percent change from baseline, the change from nadir and the percent change from nadir for sum of diameters for the target lesions. The users of the app can review the response and tumor data shown in the figures and identify data issues based on the RECIST 1.1.

INTRODUCTION

During an oncology study the study team often needs to review the tumor and response data of a particular patient to ensure the data are correct. Instead of printing out the data we can use an R shiny app to facilitate the data review. The advantage of an R shiny app is that the users do not need to do any programming or know any programming language. They only need to modify a CSV file and upload it together with the response and tumor data to the app. The app will produce the figures for review.

The data review often needs to answer two questions:

1. Is the overall response correct based on the target response, non-target response and new lesion progression?
2. Is the target response correct based on the sum of diameters for target lesions?

To answer the first question, we use the app to produce a plot for target response, non-target response and new lesion progression. The plot is reviewed based on the rules in the table 1. This table is copied from the Table 1 in reference [1].

Target lesions	Non-target lesions	New lesions	Overall response
CR	CR	No	CR
CR	Non-CR/non-PD	No	PR
CR	Not evaluated	No	PR
PR	Non-PD or not all evaluated	No	PR
SD	Non-PD or not all evaluated	No	SD
Not all evaluated	Non-PD	No	NE
PD	Any	Yes or No	PD
Any	PD	Yes or No	PD
Any	Any	Yes	PD

Table 1. Time Point Response

To answer the second question, we use the app to produce for target lesions the line plots for sum of diameters, percent change from baseline, change from nadir and percent change from nadir in sum of diameters. The line plots are reviewed based on the following rules with the assumption that the baseline assessment is not missing:

If the sum of diameters is 0, let response=CR.

Else if percent change from baseline in sum of diameters <= -30, let response=PR.

Else if sum of diameters is not missing, let response=SD.

Else let response=NE.

If percent change from nadir in sum of diameters >=20 and the absolute change from nadir >= 5 mm, let response=PD.

Note that the target response is CR when sum of diameters is not 0, but the non-zero result is contributed by pathological lymph nodes with short axis < 10 mm. This scenario is not evident in the output of the R shiny app.

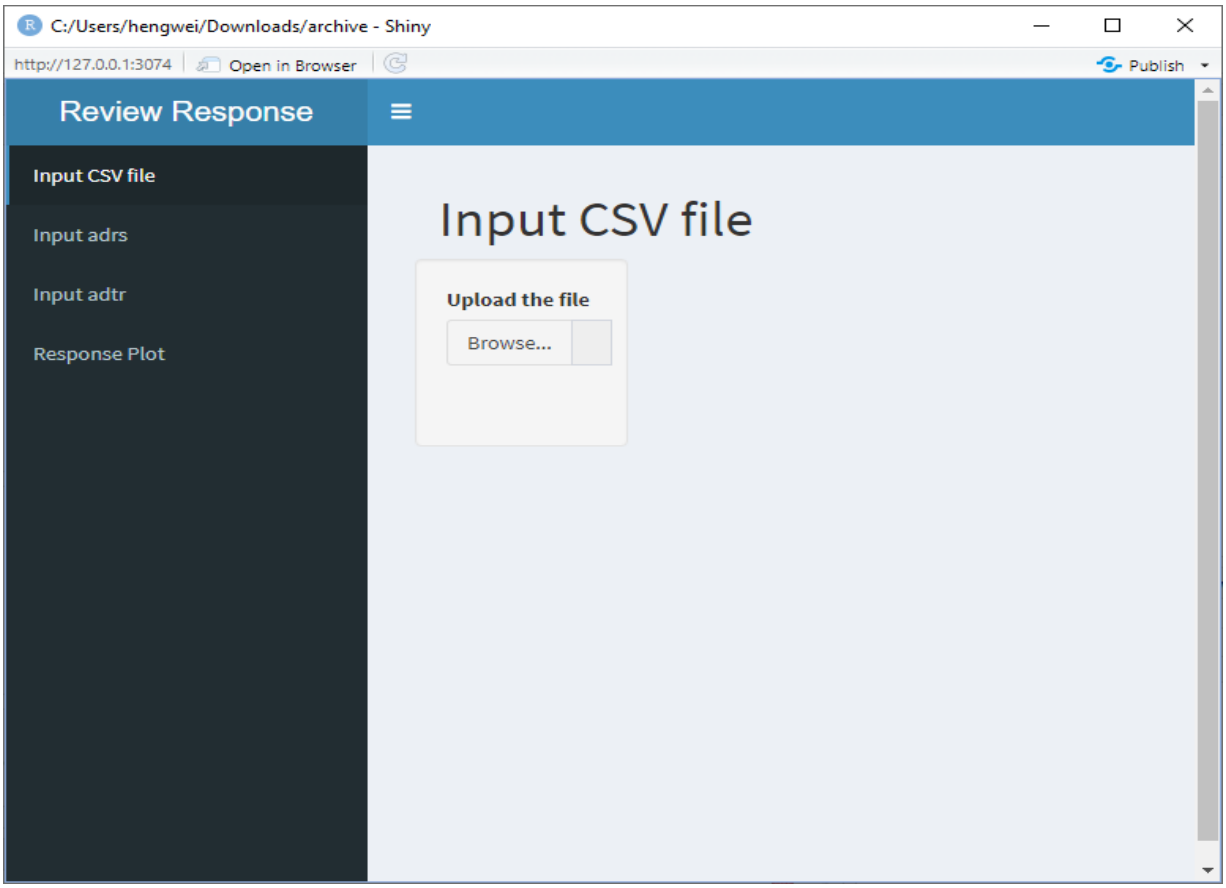
The R function ggplot is used in this app. The R used in the development is version 4.1.0.

THE DETAILS

THE DASHBOARD

In the R shiny app, we create a dashboard where the user can input a CSV file that contains the conditions used to get a subset of the input datasets. The figures are produced based on the information in the CSV file.

Display 1 shows the R shiny app dashboard. First tab is used to input the CSV file. The next two tabs are used to input the SAS datasets ADRS (Disease Response Analysis Dataset) and ADTR (Tumor Analysis Dataset). The fourth tab is used to produce the plot.



Display 1. Dashboard for the R Shiny App

THE WORKFLOW

1. User fills out the CSV file. In the CSV file the user specifies the conditions used to get a subset of the datasets ADRS and ADTR.
2. User uploads the CSV file, the SAS datasets ADTR and ADRS to the app.
3. User clicks on the tab for figure in the shiny app and creates the figure.

THE STRUCTURE OF THE CSV FILE

Table 2 shows the structure of the CSV file.

ADRS	The conditions used to get a subset of ADRS
ADTR	The conditions used to get a subset of ADTR

Table 2. The Input CSV File for the R Shiny App

Table 3 shows an example of the CSV file in actual use.

ADRS	adrs$PARQUAL=='CENTRAL' & adrs$RSACPTFL=='Y' & (adrs$PARAMCD=='OVRLRESP' \| adrs$PARAMCD=='TRGRESP' | adrs$PARAMCD=='NTRGRESP' \| adrs$PARAMCD=='NEWLPROG')
ADTR	adtr$PARQUAL=='CENTRAL' & adtr$ANL01FL=='Y' & adtr$PARAMCD=='SUMDIAM' &adtr$TRACPTFL=='Y'

Table 3. A Sample CSV File for the R Shiny App

VARIABLES REQUIRED BY THE R SHINY APP

The SAS datasets used in this app follow the ADaM data structure in ADaM IG 1.1.

Table 4 shows the variables required by the shiny app in the dataset ADTR and ADRS.

Dataset	Variable	Label	Note
ADRS	USUBJID	Unique Subject identifier	
ADRS	PARAMCD	Parameter Code	We need the parameter codes for target response, non-target response, new lesion progression and overall response in the CSV file.
ADRS	PARAM	Parameter	
ADRS	AVALC	Analysis Value (C)	

Dataset	Variable	Label	Note
ADRS	ADY	Analysis Relative Day	
ADRS	PARQUAL	Evaluator	The evaluator can be independent central review (ICR) or Investigator.
ADRS	RSACPTFL	Acceptance Flag	This flag indicates which record is accepted from multiple evaluators in ICR.
ADTR	USUBJID	Unique Subject Identifier	
ADTR	PARAMCD	Parameter Code	We need the parameter code for the sum of diameters in the CSV file.
ADTR	AVAL	Analysis Value	
ADTR	ADY	Analysis Relative Day	
ADTR	ANL01FL	Analysis Flag 01	If the number of target lesions with non-missing result at a post-baseline visit is less than that at baseline, the post-baseline record will not be flagged for analysis.
ADTR	PCHG	Percent Change from Baseline	
ADTR	NADIR	Nadir	Nadir for a visit is defined as the smallest sum of diameters prior to the visit.
ADTR	PCNSD	Percent Change from Nadir	
ADTR	ABLFL	Baseline Record Flag	
ADTR	PARQUAL	Evaluator	The evaluator can be independent central review (ICR) or Investigator.
ADTR	TRACPTFL	Acceptance Flag	This flag indicates which record is accepted from multiple evaluators in ICR.

Table 4. The Variables Required in the Datasets ADRS and ADTR

SOME INTERESTING POINTS

This app is designed to create some plots for each patient. We use a dropdown menu to provide the list of patients for the users to choose from.

The R packages sas7bdat and haven can both read SAS datasets, but the package sas7bdat cannot read compressed datasets. If the input SAS datasets are compressed, we need to use the haven package to read them.

In the CSV file there is the condition to get a subset of the ADRS dataset. To read that condition into the R shiny program, we use this trick:

```
eval(parse(text= paste0('subset(adrs,' , vr,')' ))).
```

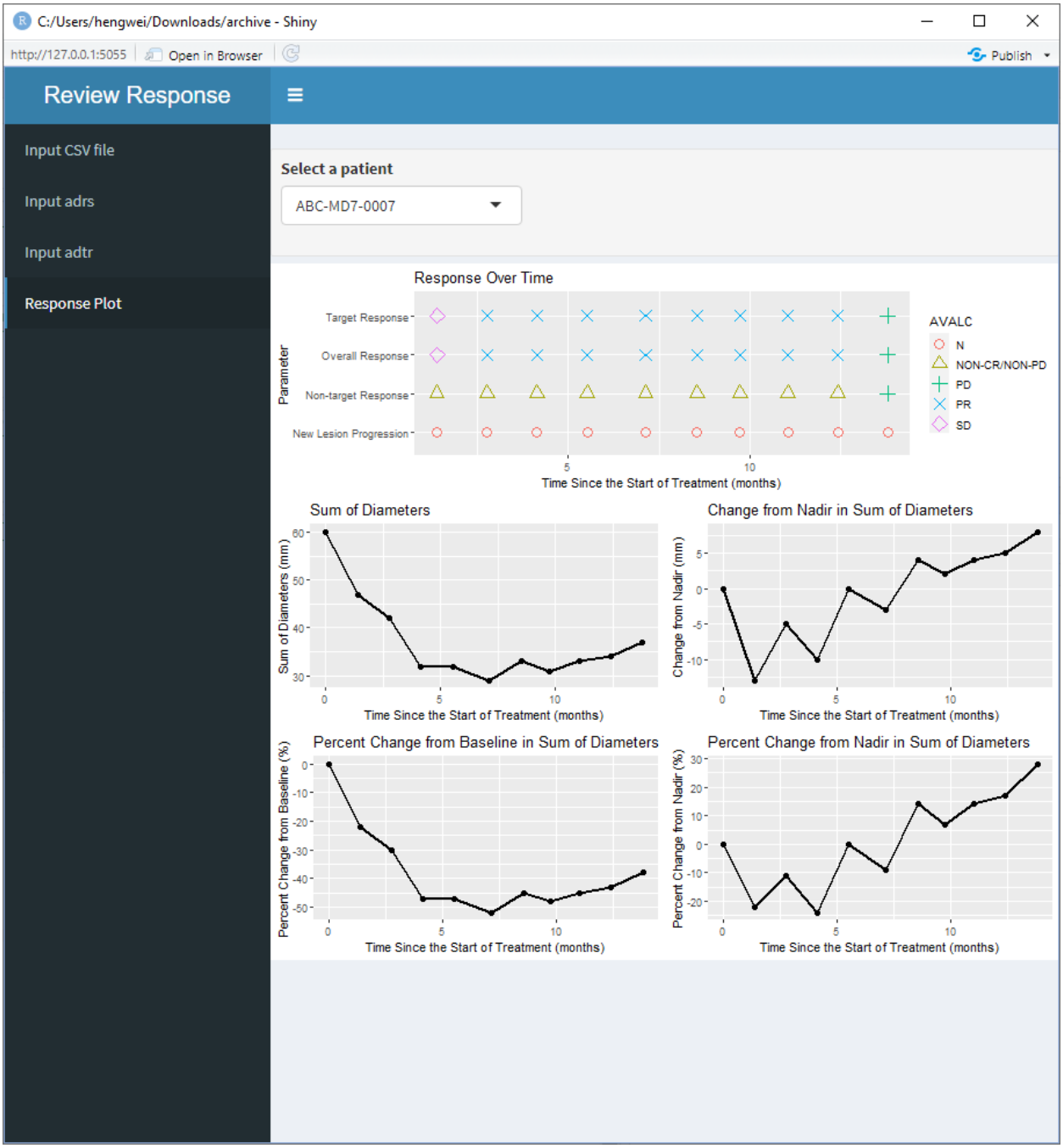
The condition to get a subset of ADTR is handled in the same way.

When we use ggplot to create the plot for the target response, non-target response, new lesion progression and overall response, we use the function scale_shape_manual to ensure that there are enough symbols in the legend:

```
scale_shape_manual(values=1:length(unique(adrs$AVALC))).
```

The line plots for sum of diameters, percent change from baseline, change from nadir and percent change from nadir are displayed in a 2x2 format. This is achieved through grid.arrange function in the package gridExtra, which can arrange multiple plots in a desirable format.

Display 2 shows a sample output of the R shiny app.



Display 2. Sample Output of the R Shiny App

CONCLUSION

R shiny can be used to create powerful visualization tools for clinical studies. In this paper we have shown how to create an R shiny app which produces figures for the response and tumor data after the user uploads the tumor data, response data and a CSV file to the app. The users can review the figures to identify possible data issues based on the RECIST 1.1. The same idea can be applied to create many other interesting and useful R shiny apps.

REFERENCES

[1] RECIST version 1.1, available at

https://project.eortc.org/recist/wp-content/uploads/sites/4/2015/03/RECISTGuidelines.pdf

CONTACT INFORMATION

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

Hengwei Liu
Hengrui USA
400 Alexander Park
Princeton, NJ 08540
Email: hengwei_liu@yahoo.com

APPENDIX

The R program is provided in the appendix.

```
#app.R
#Shiny App for Response Data
options(shiny.maxRequestSize = 30*1024^2)
library(haven)
library(shinydashboard)
library(shiny)
library(ggplot2)
library(gridExtra)


ui <-
  dashboardPage(


    dashboardHeader(title="Review Response Data"),


    dashboardSidebar(
      sidebarMenu(
        menuItem("Input CSV file", tabName="dashboard"),
```

```
    menuItem("Input adrs", tabName="adrs"),
    menuItem("Input adtr", tabName="adtr"),
    menuItem("Response Plot", tabName="swimmer")
  )
),

dashboardBody(
  tabItems(
    tabItem(tabName="dashboard",
            fluidPage(
              headerPanel(title = "Input CSV file"),
              sidebarLayout(
                sidebarPanel(
                  fileInput("file","Upload the file")
                ),
                mainPanel(
                  tableOutput("rawData")
                )
              )
            )
    ),

    tabItem(tabName="adrs",
            fluidPage(
              headerPanel(title = "Input adrs"),
              sidebarLayout(
                sidebarPanel(
                  fileInput(
                    inputId = "adrs",
                    label = "Choose SAS datasets"
                  )
                ),
                mainPanel(
                  tableOutput("adrs")
                )
              )
            )
```

```
        ),

        tabItem(tabName="adtr",
                fluidPage(
                  headerPanel(title = "Input adtr"),
                  sidebarLayout(
                    sidebarPanel(
                      fileInput(
                        inputId = "adtr",
                        label = "Choose SAS datasets"
                      )
                    ),
                    mainPanel(
                      tableOutput("adtr")
                    )
                  )
                )
        ),

        tabItem(
          tabName="swimmer",
          fluidRow(
            inputPanel( selectInput("USUBJID","Select a patient:", c( ))),
            plotOutput("swPlot", height=200),
            plotOutput("sdPlot", height=400)
          ))
      )))
server <- function(input, output, session) {
  output$rawData <- renderDataTable({
    file_to_read=input$file
    if(is.null(file_to_read)){
      return()
    }
  })
  output$adrs <- renderDataTable({
    file_to_read=input$adrs
    if(is.null(file_to_read)){
```

```
      return()
    }
  })

  output$adtr <- renderDataTable({
    file_to_read=input$adtr
    if(is.null(file_to_read)){
      return()
    }
  })

  output$swPlot <- renderPlot({
    rawData <- read.csv(input$file$datapath, header=FALSE)
    vr<-paste(rawData[(rawData$V1=='ADRS'),]$V2)

    adrs <- read_sas(input$adrs$datapath)
    data.frame(adrs)

    adrs <- eval(parse(text= paste0('subset(adrs,' , vr,')' )))
    adrs$y <- adrs$PARAM
    adrs$x <- adrs$ADY/(365.25/12)
    adrs <- adrs[c("x","y","AVALC","USUBJID")]

    updateSelectInput(session, "USUBJID", label = "Select a patient",
                      choices = c(unique(adrs$USUBJID)),
select=input$USUBJID)

    dataInput1 <- reactive({
      adrs[(adrs$USUBJID == input$USUBJID),]
    })

    myPlot <- ggplot(NULL, aes( x=x , y=y) ) +
        labs(title = "Response Over Time",
        x = "Time Since the Start of Treatment (months)", y = "Parameter") +
    geom_point(data=dataInput1(), aes( colour=AVALC, shape=AVALC), size=4) +
    scale_shape_manual(values=1:length(unique(adrs$AVALC)))

    print(myPlot) })
```

```
  output$sdPlot <- renderPlot({
    rawData <- read.csv(input$file$datapath, header=FALSE)
    vr<-paste(rawData[(rawData$V1=='ADTR'),]$V2)

    adtr <- read_sas(input$adtr$datapath)
    data.frame(adtr)

    adtr <- eval(parse(text= paste0('subset(adtr,' , vr,')' )))
    adtr$x <- ifelse(adtr$ABLFL=='Y', 0,adtr$ADY/(365.25/12))
    adtr$y1 <- adtr$AVAL
    adtr$y2 <- ifelse(adtr$ABLFL=='Y', 0, adtr$PCHG)
    adtr$y3 <- ifelse(adtr$ABLFL=='Y', 0, (adtr$AVAL-adtr$NADIR))
    adtr$y4 <- ifelse(adtr$ABLFL=='Y', 0, adtr$PCNSD)

    adtr <- adtr[c("x","y1","y2","y3","y4", "USUBJID")]

    updateSelectInput(session, "USUBJID", label = "Select a patient",
                      choices = c(unique(adtr$USUBJID)),
select=input$USUBJID)

    dataInput2 <- reactive({
      adtr[(adtr$USUBJID == input$USUBJID),]
    })

    myPlot1 <- ggplot(dataInput2(), aes(x = x, y = y1)) +
      labs(title = "Sum of Diameters",
           x = "Time Since the Start of Treatment (months)",
           y = "Sum of Diameters (mm)") +
      geom_line(size=1) +
      geom_point( size=2)

    myPlot2 <- ggplot(dataInput2(), aes(x = x, y = y2)) +
      labs(title = "Percent Change from Baseline in Sum of Diameters",
           x = "Time Since the Start of Treatment (months)",
           y = "Percent Change from Baseline (%)") +
      geom_line(size=1) +
      geom_point( size=2)
```

```
    myPlot3 <- ggplot(dataInput2(), aes(x = x, y = y3)) +
      labs(title = "Change from Nadir in Sum of Diameters",
           x = "Time Since the Start of Treatment (months)",
           y = "Change from Nadir (mm)") +
      geom_line(size=1) +
      geom_point( size=2)

    myPlot4 <- ggplot(dataInput2(), aes(x = x, y = y4)) +
      labs(title = "Percent Change from Nadir in Sum of Diameters",
           x = "Time Since the Start of Treatment (months)",
           y = "Percent Change from Nadir (%)") +
      geom_line(size=1) +
      geom_point( size=2)

    ptlist <- list(myPlot1, myPlot3, myPlot2, myPlot4)
    grid.arrange(grobs=ptlist,ncol=2, nrow=2)
     })
}
shinyApp(ui=ui, server=server)
```