Using R Shiny to Explore Clinical Trial Data

Yaoxian Yuan, Jiaying Wu, CStone Pharmaceuticals (Suzhou) Co., Ltd

ABSTRACT

Shiny is an R package that makes it easy to build interactive web apps straight from R. It provides the framework to explore and analysis the clinical trial data in a more efficient way. Compatible with other packages(e.g. DT, echarts4r), listing, table and figure can be displayed dynamically which is customized to simplify filtering and visualizing the data. After the set-up process completed, R shiny app could be deployed under both local disk and public disk so that the whole team could review the results. In this paper, an example will show how to develop the R shiny app by statistical programmers who have no experience on the frontend website development.

Key Words: R shiny, echarts4r, deployment, data visualization, clinical trial data

BACKGROUND

R is a programming language and free software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing. In clinical trial industry, the R language is widely used among statisticians. As of July 2021, R ranks 12th in the TIOBE index, a measure of popularity of programming languages [1]. Shiny is an R package that helps to build interactive web apps directly from R. It provides the framework to explore and analysis the clinical data in a more efficient way. Apache ECharts is a free, powerful charting and visualization library offering an easy way of adding intuitive, interactive, and highly customizable charts to your commercial products [2]. Package echarts4r is the R package converted from ECharts. With “ggplot2” knowledge, echarts4r is easy to learn.

With the increasing need for exploring data in a timely and visualized manner, developing the R shiny app is a smart choice.

R SHINY APP STRUCTURE

A simple R Shiny app is divided into two parts: the User Interface (UI) and the Server. The UI is responsible for the app presentation, while the server is responsible for the app logic. Here is the sample code of a simple R shiny app.

```
library(shiny)
ui <- fluidPage()
server <- function(input, output){}
shinyApp(ui = ui, server = server)
```

The UI controls what is being displayed on the application page and how the components are laid out, including defining the webpage layout, defining static HTML display information, defining HTML controls that receive user input, and defining HTML controls that output information when responding to users.

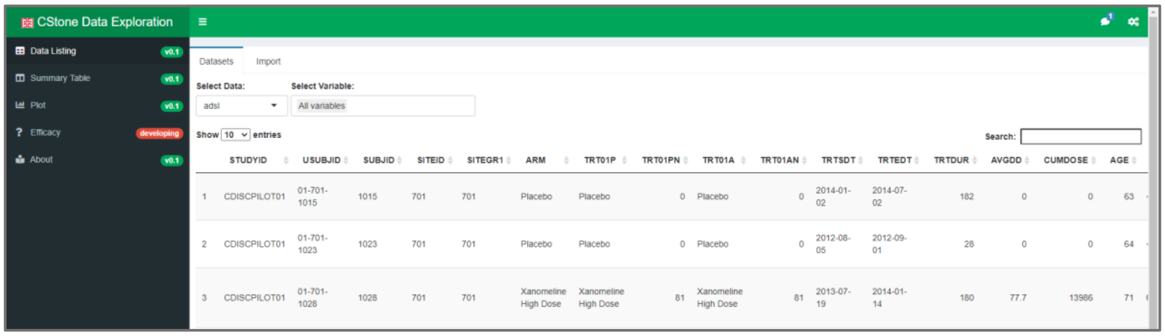
The Server controls the data that will be displayed through the UI. The server will be where you load and analyze data, then define the outputs using input from the UI.

Both of two parts can be defined in one file, but it is a good practice to separate these into three files to simplify any future changes or maintenance on the app. It means you need to set up three files: *ui.R* to define the app presentation, *server.R* to define the app logic, and *app.R* to combine the former two files.

R SHINY APP EXAMPLE

Following is the introduction on a Shiny app in clinical trial data. You can click the URL to browse the webpage, the app was deployed to the shiny cloud. For the sample code of the shiny app, please refer to the *github* or *gitee*.

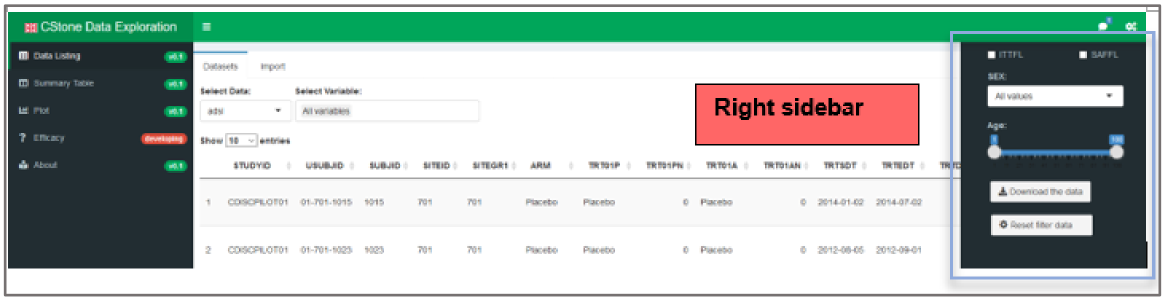
The shiny app contains five main pages: *Data Listing*, *Summary Table*, *Plot*, *Efficacy* and *About*, which is shown on Figure 1. Specially, the sidebar on the right is developed to filter and download data. This sidebar automatically works on all pages by default. Generally, core variables or baseline character variables are implemented in the right sidebar, Figure 2 shows the example of the sidebar.



Select Data: adsl
Select Variable: All variables
Show 10 entries
Search:

	STUDYID	USUBJID	SUBJID	SITEID	SITEGR1	ARM	TRT01P	TRT01PN	TRT01A	TRT01AN	TRTSDT	TRTEDT	TRTDUR	AVGDD	CUMDOSE	AGE
1	CDISCPILOT01	01-701-1015	1015	701	701	Placebo	Placebo	0	Placebo	0	2014-01-02	2014-07-02	182	0	0	63
2	CDISCPILOT01	01-701-1023	1023	701	701	Placebo	Placebo	0	Placebo	0	2012-08-05	2012-09-01	28	0	0	64
3	CDISCPILOT01	01-701-1028	1028	701	701	Xanomeline High Dose	Xanomeline High Dose	81	Xanomeline High Dose	81	2013-07-19	2014-01-14	180	77.7	13986	71

Figure1: The whole page of shiny app



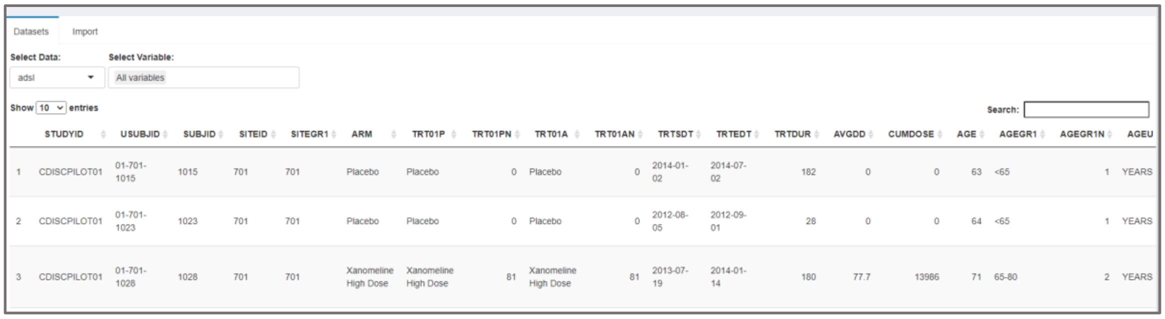
Select Data: adsl
Select Variable: All variables
Show 10 entries

	STUDYID	USUBJID	SUBJID	SITEID	SITEGR1	ARM	TRT01P	TRT01PN	TRT01A	TRT01AN	TRTSDT	TRTEDT
1	CDISCPILOT01	01-701-1015	1015	701	701	Placebo	Placebo	0	Placebo	0	2014-01-02	2014-07-02
2	CDISCPILOT01	01-701-1023	1023	701	701	Placebo	Placebo	0	Placebo	0	2012-08-05	2012-09-01

ITTFL SAFFL
SEX: All values
Age:

Figure2: Right sidebar

Data Listing page is designed to display patient line listings which is popularly used in clinical trials as Figure 3 shows. Multiple datasets can be selected in the selectinput. The variables in selectinput will be changed dynamically according to the dataset selected. A search box also can be used to search the data. As mentioned above, the data table will be responded immediately if the right sidebar worked.



Select Data: adsl

Select Variable: All variables

Show 10 entries

Search:

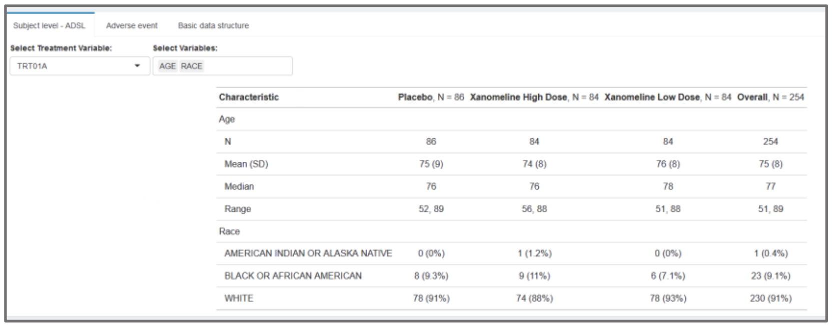
	STUDYID	USUBJID	SUBJID	SITEID	SITEGR1	ARM	TRT01P	TRT01PN	TRT01A	TRT01AN	TRTSDT	TRTEDT	TRTDUR	AVGDD	CUMDOSE	AGE	AGEGR1	AGEGR1N	AGEU
1	CDISCPILOT01	01-701-1015	1015	701	701	Placebo	Placebo	0	Placebo	0	2014-01-02	2014-07-02	182	0	0	63	<65	1	YEARS
2	CDISCPILOT01	01-701-1023	1023	701	701	Placebo	Placebo	0	Placebo	0	2012-08-05	2012-09-01	28	0	0	64	<65	1	YEARS
3	CDISCPILOT01	01-701-1028	1028	701	701	Xanomeline High Dose	Xanomeline High Dose	81	Xanomeline High Dose	81	2013-07-19	2014-01-14	180	77.7	13986	71	65-80	2	YEARS

Figure3: Example of data listing

The R package “DT” provides an R interface to the JavaScript library “DataTables”. R data object can be displayed as tables on HTML pages, it provides filtering, pagination, sorting, and many other features in the table [3]. *Output() and render*() functions work together to add R output to the UI. *Output() is used in *ui.R* and render*() is used in *server.R*. For example, to output the data listing, package “DT” provides function DTOuput() and renderDT().

```
ui <- fluidPage(
    DTOutput("listing1")
)
server <- function(input, output){
    output$listing1 <- renderDT(

    )
}
```

Summary Table page is designed to display basic statistical summaries such as frequency count and descriptive statistics. Figure 4 shows the summary of data from subject level analysis dataset ADSL. The package “gtsummary” is used to generate this table. Function gt_output() is used in *ui.R*, correspondingly, render_gt() is used in *server.R*. The “gtsummary” package provides an elegant and flexible way to create publication-ready analytical and summary tables using the R programming language [4]. Figure 5 shows the summary of adverse events and Figure 6 is the table generated from datasets with CDISC basic data structure implemented. The package “arsenal” is applied to develop the summary table of this kind of data. tableOutput() in *ui.R* and renderTable() in *server.R* are used to output summary tables in Figure 5 and Figure 6.

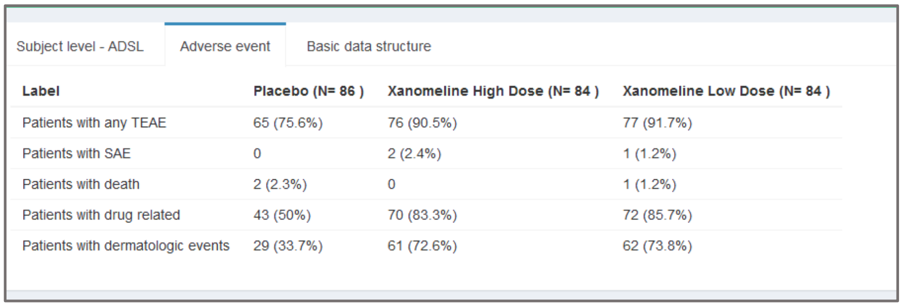


Select Treatment Variable: TRT01A

Select Variables: AGE RACE

Characteristic	**Placebo**, N = 86	**Xanomeline High Dose**, N = 84	**Xanomeline Low Dose**, N = 84	**Overall**, N = 254
Age				
N	86	84	84	254
Mean (SD)	75 (9)	74 (8)	76 (8)	75 (8)
Median	76	76	78	77
Range	52, 89	56, 88	51, 88	51, 89
Race				
AMERICAN INDIAN OR ALASKA NATIVE	0 (0%)	1 (1.2%)	0 (0%)	1 (0.4%)
BLACK OR AFRICAN AMERICAN	8 (9.3%)	9 (11%)	6 (7.1%)	23 (9.1%)
WHITE	78 (91%)	74 (88%)	78 (93%)	230 (91%)

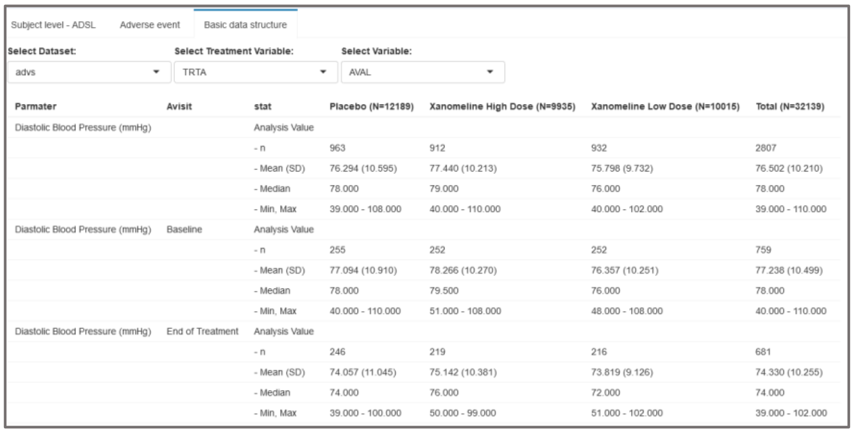
Figure4: Table of subject level data



Subject level - ADSL | Adverse event | Basic data structure

Label	Placebo (N= 86)	Xanomeline High Dose (N= 84)	Xanomeline Low Dose (N= 84)
Patients with any TEAE	65 (75.6%)	76 (90.5%)	77 (91.7%)
Patients with SAE	0	2 (2.4%)	1 (1.2%)
Patients with death	2 (2.3%)	0	1 (1.2%)
Patients with drug related	43 (50%)	70 (83.3%)	72 (85.7%)
Patients with dermatologic events	29 (33.7%)	61 (72.6%)	62 (73.8%)

Figure5: Table of adverse events



Subject level - ADSL | Adverse event | Basic data structure

Select Dataset: advs

Select Treatment Variable: TRTA

Select Variable: AVAL

Parmater	Avisit	stat	Placebo (N=12189)	Xanomeline High Dose (N=9935)	Xanomeline Low Dose (N=10015)	Total (N=32139)
Diastolic Blood Pressure (mmHg)		Analysis Value				
		- n	963	912	932	2807
		- Mean (SD)	76.294 (10.595)	77.440 (10.213)	75.798 (9.732)	76.502 (10.210)
		- Median	78.000	79.000	76.000	78.000
		- Min, Max	39.000 - 108.000	40.000 - 110.000	40.000 - 102.000	39.000 - 110.000
Diastolic Blood Pressure (mmHg)	Baseline	Analysis Value				
		- n	255	252	252	759
		- Mean (SD)	77.094 (10.910)	78.266 (10.270)	76.357 (10.251)	77.238 (10.499)
		- Median	78.000	79.500	76.000	78.000
		- Min, Max	40.000 - 110.000	51.000 - 108.000	48.000 - 108.000	40.000 - 110.000
Diastolic Blood Pressure (mmHg)	End of Treatment	Analysis Value				
		- n	246	219	216	681
		- Mean (SD)	74.057 (11.045)	75.142 (10.381)	73.819 (9.126)	74.330 (10.255)
		- Median	74.000	76.000	72.000	74.000
		- Min, Max	39.000 - 100.000	50.000 - 99.000	51.000 - 102.000	39.000 - 102.000

Figure6: Table generated from dataset with CDISC basic data structure

Plot page is designed to display statistical charts. Package “echarts4r” is a powerful data visualization package in R. Figure 7 shows the bar plot using “echarts4r”, with a message box showing the count number. On the upper right corner, there are five click buttons to support saving image, zooming in/out of the data, viewing the data and restoring figure from left to right respectively. Varieties of plots could be generated by using “echarts4r”, such as, line chart, pie chart, scatter chart, box plot, heatmap, treemap and 3D map [5],etc. Here is the sample code for generating bar chart by using “echarts4r”.

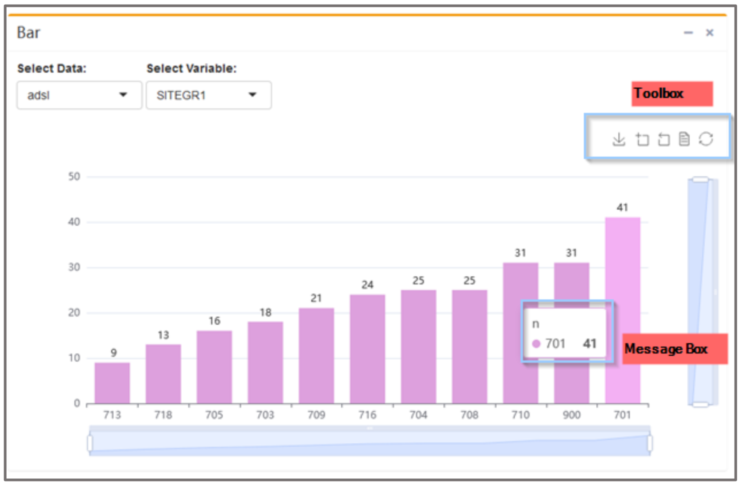


Figure7: Bar chart used echarts4r

```
tmpplotds %>% count(get(input$plotvariables1)) %>% arrange(n) %>%
    e_charts(get(input$plotvariables1)) %>%
    e_bar(n, legend = FALSE) %>%
    e_datazoom(
    x_index=0,
    toolbox = FALSE
    ) %>%
    e_datazoom(
    y_index=0,
    toolbox = FALSE
    ) %>%
    e_tooltip(trigger = "item") %>%
    e_labels(position = "top") %>%
    e_x_axis(splitLine = list(show = FALSE)) %>%
    e_color(c('#dda0dd', '#eeeeee', '#59c4e6', '#edafda')) %>%
    e_toolbox_feature(feature = "saveAsImage") %>%
    e_toolbox_feature(feature = "dataZoom") %>%
    e_toolbox_feature(feature = "dataView") %>%
    e_toolbox_feature(feature = "restore")
```

In addition to basic charts, plots used in oncology studies could also be generated by using “echarts4r”. Here is the example of swimmer plot. A message box shows the overall response of the selected patient on the relative day by the selecting timepoint. Data zoom in x axis or y axis could be used to zoom in/out the plot. Following shows the snapshot of swimmer plot by “echarts4r”, also with the sample code.

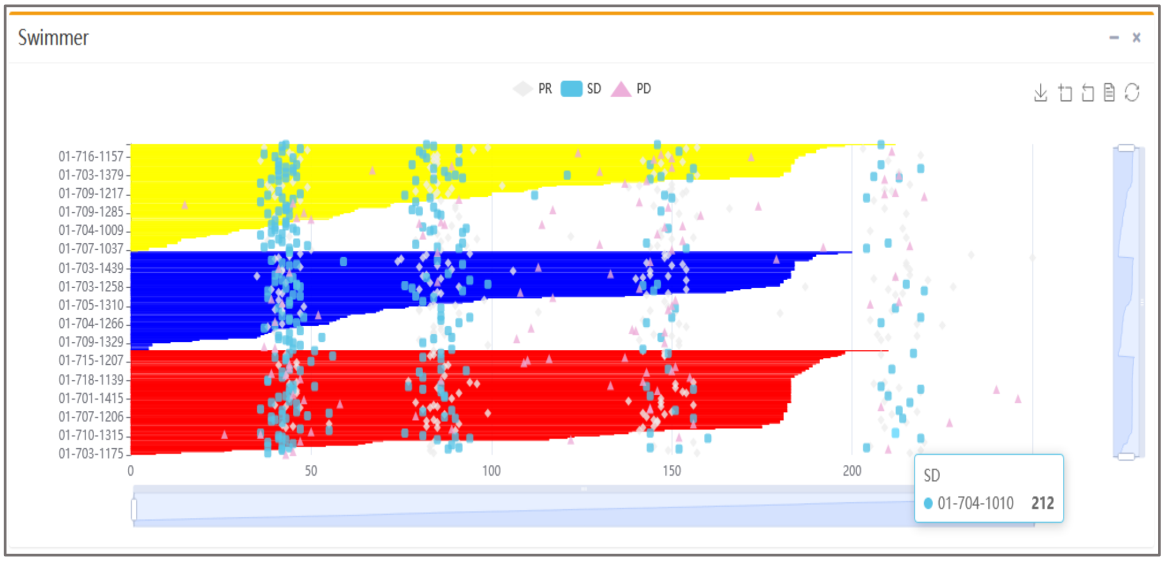
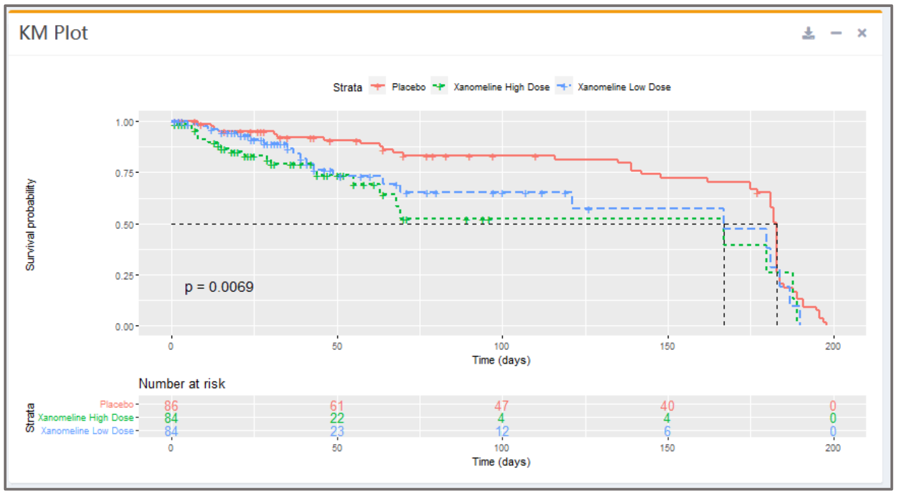


Figure8: Swimmer plot used echarts4r

```
tmpplotds2 %>%
    arrange(TRTA, treat) %>%
    e_charts(x = USUBJID) %>%
    e_bar(treat, legend = FALSE, name = "Treatment duration") %>%
    e_scatter(CR, name = "CR", symbol = "circle", symbol_size = 8) %>%
    e_scatter(PR, name = "PR", symbol = "diamond", symbol_size = 8) %>%
    e_scatter(SD, name = "SD", symbol = "roundRect", symbol_size = 8) %>%
    e_scatter(PD, name = "PD", symbol = "triangle", symbol_size = 8) %>%
    e_datazoom(
    x_index = 0,
    type = "slider",
    toolbox = FALSE
    ) %>%
    e_datazoom(
    y_index = 0,
    type = "slider",
    toolbox = FALSE
    ) %>%
    e_tooltip() %>%
    e_flip_coords() %>%
    e_y_axis(splitLine = list(show = FALSE)) %>%
    e_x_axis(show = TRUE) %>%
    e_add("itemStyle",color) %>%
    e_color(c('#dda0dd' , '#eeeeee', '#59c4e6', '#edafda')) %>%
    e_toolbox_feature(feature = "saveAsImage") %>%
    e_toolbox_feature(feature = "dataZoom") %>%
    e_toolbox_feature(feature = "dataView") %>%
    e_toolbox_feature(feature = "restore")
```

In the similar way, waterfall and spider could also be generated by using “echarts4r”. The syntax of “echarts4r” is similar to “ggplot2” which is more widely used by the users, it is very easy to learn the “echarts4r” if you are familiar with “ggplot2”. echarts4rOutput() is used in *ui.R* and renderEcharts4r() in *server.R*.

Efficacy page is designed to display analysis of efficacy data. Survival analysis is essential when talking about an oncology study, “survminer” is a useful package for the survival analysis, K-M curve could be generated by using the package. Figure 9 shows the km curve by using “survminer”, following is the sample code.



Strata	0	50	100	150	200
Placebo	86	61	47	40	0
Xanomeline High Dose	84	22	4	4	0
Xanomeline Low Dose	84	23	12	6	0

Figure9: KM plot used survminer

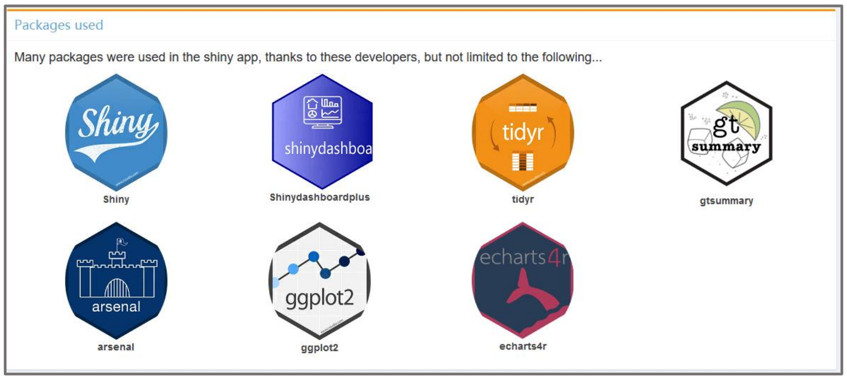
```
fit <- survfit(as.formula(paste("Surv(AVAL, CNSR)~", str1)),
                data = adtte)

ggplot1 <- ggsurvplot(fit,
            pval = TRUE, conf.int = FALSE,
            risk.table = TRUE,
            risk.table.col = "strata",
            linetype = "strata",
            surv.median.line = "hv",
            ggtheme = theme_gray(),
            xlab = "Time (days)",
            axes.offset =  T,
            legend.labs = c("Placebo", "Xanomeline High Dose", "Xanomeline Low Dose")
            )
```

By summarizing all functions mentioned above, Table 1 shows the summary of common output objects in shiny.

Table1: Summary of common output objects in shiny

Object	Function in *ui.R*	Function in *server.R*
Data table	DTOuput()	renderDT()
Summary table(gt)	gt_output()	render_gt()
Summary table(data frame)	tableOutput()	renderTable()
Plot(echarts4r)	echarts4rOutput()	renderEcharts4r()
Plot(ggplot2)	plotOutput()	renderPlot()

About page is designed to provide the information of the packages. There are three parts in About page: the study data, about packages used and about me. This is a pilot study from the CDISC website, source data can be downloaded from the *link*. Since ADTR and ADRS are not available in this data source, the two datasets are dummied by paper author. Many packages are used in the shiny app but not limited to the ones listed in Figure 10.


Packages used

Many packages were used in the shiny app, thanks to these developers, but not limited to the following...

Figure10: Packages used

R SHINY APP DEPLOYMENT

When talking about the deployment of the R shiny app, it could be deployed to the cloud, such as shinyapps.io, but the free accounts are limited, so the better choice is to deploy on local disk or public disk. Here is the method [6] on the deployment of the R shiny app.

1. Create a folder named R_shiny/.
2. Download firefox portable, R portable and install them under the folder R_shiny/.
3. Create a folder under R_shiny/, named shiny3, so the folder is like following

```
R_shiny/

    +- Firefox portable
    +- R-portable
    +- shiny3/
```

4. Modify the R-Portable/App/R-Portable/etc/Rprofile.site, add below codes to the bottom of the file.

```r
.First = function(){
    .libPaths(.Library)
}
```

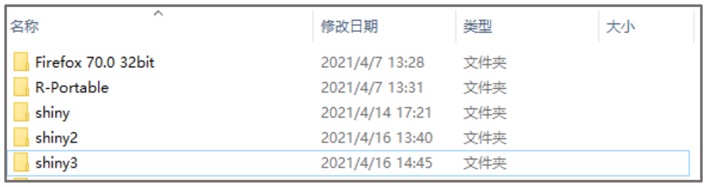
5. Start R-Portable and install core package dependencies.
6. Create a R script, named *runShinyApp.R*, below codes added.

```r
message('library paths:\n', paste('... ', .libPaths(), sep='', collapse='\n'))
#if want to use ie browser
ie=file.path('C:/Program Files (x86)/Internet Explorer/iexplore.exe')
firefox=file.path(getwd(),' ../../Firefox 70.0 32bit/App/Firefox/firefox.exe')
options(browser = firefox)
shiny::runApp('./ ', launch.browser=TRUE)
```

7. Create a vb script, named *run.vbs*, below codes added.

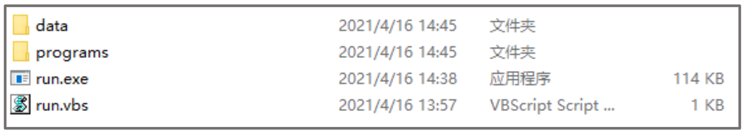
```vbscript
Rexe           = "..\..\R-Portable\App\R-Portable\bin\Rscript.exe"
Ropts          = "--no-save --no-environ --no-init-file --no-restore --no-
Rconsole"
RScriptFile    = "runShinyApp.R"
Outfile        = "ShinyApp.log"
strCommand     = Rexe & " " & Ropts & " " & RScriptFile & " 1> " & Outfile & " 2
>&1"
intWindowStyle = 0
bWaitOnReturn  = False

With CreateObject("Wscript.Shell")
    .CurrentDirectory = "programs"
    .Run strCommand, intWindowStyle, bWaitOnReturn
End With
```

8. Finally, the folder can be created shown as Figure 11, 12 and 13. Double click the run.vbs, finish the deployment.



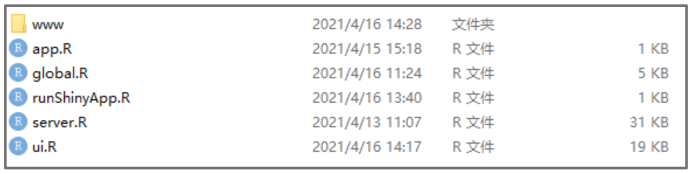
名称	修改日期	类型	大小
Firefox 70.0 32bit	2021/4/7 13:28	文件夹	
R-Portable	2021/4/7 13:31	文件夹	
shiny	2021/4/14 17:21	文件夹	
shiny2	2021/4/16 13:40	文件夹	
shiny3	2021/4/16 14:45	文件夹	

Figure11: Folder under R_shiny



data	2021/4/16 14:45	文件夹	
programs	2021/4/16 14:45	文件夹	
run.exe	2021/4/16 14:38	应用程序	114 KB
run.vbs	2021/4/16 13:57	VBScript Script ...	1 KB

Figure12: Folder under R_shiny/shiny3



www	2021/4/16 14:28	文件夹	
app.R	2021/4/15 15:18	R 文件	1 KB
global.R	2021/4/16 11:24	R 文件	5 KB
runShinyApp.R	2021/4/16 13:40	R 文件	1 KB
server.R	2021/4/13 11:07	R 文件	31 KB
ui.R	2021/4/16 14:17	R 文件	19 KB

Figure13: Folder under R_shiny/shiny3/programs

CONCLUSION

With the help of R shiny, we can quickly develop a visual and interactive web page to explore clinical trial data, after the deployment on public disk, the whole team can review the results. After setting-up the apps, statisticians can quickly validate the study results without developing complex SAS codes and the results can be displayed in a more visualized manner to medical or other functions to illustrate the results. In addition to the basic analysis, R shiny app is also helpful for the exploratory analysis, varieties of plots/tables could be generated to meet the requirements of varieties roles from other functions. For example, heat maps could be used to explore the incidence rate of specified adverse events.

DISCUSSION

During the development, two points can be improved in the future. One is *ui.R*, knowledge about HTML/CSS/JavaScript can be helpful to make the R shiny app look crisp and usable. Another is the big data, the R package “data.table” is powerful when dealing with big data, it provides a high-performance version of base R’s data frame with syntax and feature enhanced for ease of use and with programming efficiency improved [7].

REFFERENCES

[1]: *https://www.tiobe.com/tiobe-index/*

[2]: *https://echarts.apache.org/en/index.html/*

[3]: *https://rstudio.github.io/DT/*

[4]: *https://www.danieldsjoberg.com/gtsummary/*

[5]: *https://echarts.apache.org/examples/en/index.html/*

[6]: *https://www.r-bloggers.com/2014/04/deploying-desktop-apps-with-r/*

[7]: *https://github.com/Rdatatable/data.table/*

ACKNOWLEGEMENTS

I would like to thank my line management for her encouragement and review of this paper, and great appreciation to the R packages developers, without their efforts and sharing sprites, superfluous efforts may be cost by supporting timely data exploring requests.

CONTACT INFORMATION

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

Yaoxian Yuan

CStone Pharmaceuticals (Suzhou) Co., Ltd

21/F, No. 399, West Haiyang Road, New Bund Times Square, Pudong New Area, Shanghai, China, 200126

xiaodaowuzhi@hotmail.com

Jiaying Wu

CStone Pharmaceuticals (Suzhou) Co., Ltd

21/F, No. 399, West Haiyang Road, New Bund Times Square, Pudong New Area, Shanghai, China, 200126

wujiaying@cstonepharma.com