PhUSE US Connect 2019

Paper

Proof-of-concept (POC) Development of Safety Visual Analytics (SVA) Using R-shiny and SAS

Rinki Jajoo, Sammy Yuan, Nathan Li, Yong Zhu,
Daniel Woodie Merck & Co., Inc., Kenilworth, NJ, USA

ABSTRACT:

To alleviate the cumbersome burdens that safety physicians suffer every day during various reviews of ongoing clinical trial data, Proof of Concept(POC) development of an SVA system (in house named as VACS) was launched in Clinical Safety Statistics (CSS). In this paper, we will share the strategic thinking behind the design of the SVA system using SAS and R-shiny, in the hope to meet regulatory compliance, maintain confidentiality of ongoing clinical trial data and perform computing efficiently for large scale applications. We will also illustrate how these requirements are handled via an automated, real-time computing infrastructure using a divide-and-conquer mechanism between data visualization and core statistical computing undertaken by R and SAS, respectively. The good practice of implementing the SVA framework to adapt for the balance between requirements for highly customized solutions and re-usefulness of the programs will also be summarized system.

INTRODUCTION

Safety Visual Analytic tool is customized system designed to provide end user capabilities to visualize aggregated data, analyze and dynamically drill down without additional programming efforts. The SVA tool also provide developers and statisticians to apply customized statistical methods to display graphics while using data enriched for analysis and reporting (ADAM data). SVA tool was designed to support Clinical Safety Risk Management Physician in review of ongoing aggregated safety data. With shift in understanding safety data, there is critical need for interactive assessment rather than prespecified reporting results of the data. Also, SVA will assist safety physician to draw conclusion based on the data displayed, the system required to be compliant as per regulatory guidance.

Design of Safety Visual Analytic Tool (SVA)

SVA is a data visualization platform which shall enable end users to review the clinical safety data. Safety physicians or other end users submit the requests from R-shiny User interface (LINUX) which will trigger the execution of corresponding SAS macros stored in the CPI platform (UNIX) and return the output in visual format for analysis requested by end users. The reason for creating the SVA System are to overcome following challenge with the off shelf interactive tools 1) does not provide enough flexibility to modify the underlying statistical methods 2) requires tool specific data structures 3) lack reproducibility and validation of the tool and 4) pre-canned reports methods cannot be modified.

Hence, SVA system was designed as three tier systems 1) Data Visual layer 2) Computational layer and 3) Data source layer. The three-tier system provides flexibility to update one layer without much impact to other two layers. In SVA the data visual layer was developed using R-Shiny web interface. The computational layer used traditional SAS for any derivations or computations and the data layer is a UNIX superdome, where submission related activities occurred.

Architecture of SVA system

Data visualization layer

R-Shiny is used to serve out interactive visualizations for our analysts. This framework was preferred because it is native to the web and can be viewed in the browser, it has an extensive set of options for powerful visualizations, is written in R and can easily access the powerful set of statistical packages in the open-source and can easily be ported to communicate with our other architectural layer using standard communication protocols. Additionally, the

user's information is captured at the visit to the application and is then used for setting access permissions for both the computational layer and data source layer.

Computational layer

As much of our workflows are currently supported with extensive SAS code-bases, our computational layer needed to be able to leverage the existing macros. Moreover, there is no current guidance on how to use R in regulated environment created challenge to use R computation for SVA displays. Hence the SVA system is using SAS computing for any derivation required on source data. The design allowed validity of the plots while maintaining traceability and reproducibility. SAS macro computation on the source data is triggered when user pass parameter from R-shiny interface. SAS program initialize and clears any previous run output. The SAS macro executes, and the results are passed to R-Shiny through functionality of rCPI package.

Data source layer

The data source layer can be accessed directly from both our computational layer and data visualization layer. The data source layer resides on Unix Platform (CPI) which is also used as regulatory/submission workbench. The data source stored in standard folder structure, in ADaM or SDTM format. The design of the system allows to use the same source data as used for analysis and reporting. The design provide efficiency in programming and validation by reducing the resource requirement for recreating dataset.

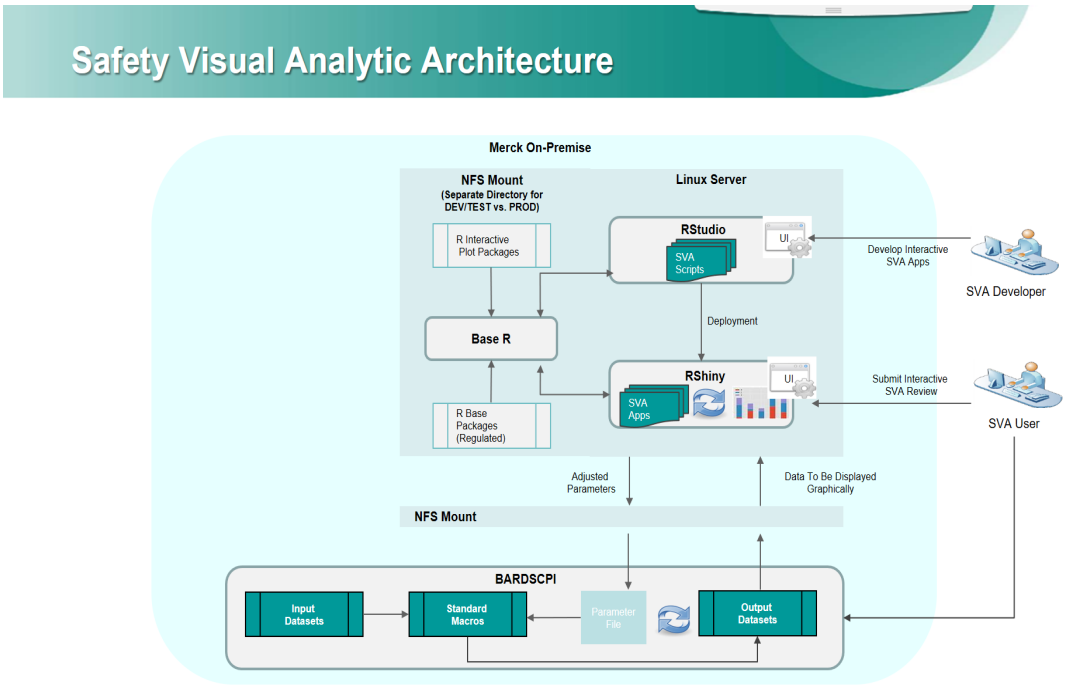

Figure 1: SVA architecture diagram. The three-layer architecture was built using Linux/Unix environment.

Regulated Environment

In order to meet the regulatory compliance, maintain confidentiality of ongoing clinical trial data and perform computing efficiently for large scale applications, the system was created in a controlled environment. R is an open source software, freely available made it easy to use. Shiny is an R package that makes it easy to build interactive web apps straight from R. Shiny combines the computational power of R with the interactivity of the modern web. AS R tool is open software, it created challenges in creating the validated system. To overcome the challenge of regulatory compliance yet leveraging R functionalities, proposed to use enrichments and computation to create analysis datasets in SAS.

SAS has been used by pharmaceutical industries and regulators for data analyses for a long time and was deemed as a validated and compliant statistical tool for submission. In order to take advantage of the already validated

statistical methodologies and widely accept format of the outputs from SAS, the SVA system uses SAS for all the statistical analyses and saves the output data of analyses done by SAS in CPI folders. The output data will then be fetched by R/R-shiny to create visual displays. The usage of SAS as core analysis tool avoids the necessity to validate the R functions for similar analyses, and solve the concern of not being compliant, if using other software.

The communication between three layers are done using a R package developed internally (rCPI by Daniel Woodie). The main functionality of this package is to enable 'read' and 'run' capabilities. Specifically, a statistician or programmer may want to read a file from CPI into their R session. Additionally, they may want to run a SAS macro from R. This modular functionality aims to enable the basic workflow of analyzing data within CPI.

One special note here is that this package leverages OpenSSH -- a low-level program to enable communication between two hosts. This package abstracts away the required understanding of using OpenSSH and provides a simplified style more similar to the standard workflow of statisticians. One requirement for this package to work, though, is for an SSH key pair to be shared between the two machines communicating with one another. For more information on this, please look into SSH key-pairs.

The rCPI package created to interact with data layer (CPI) will enable to

1) Run SAS macros on CPI from the Shiny server
2) Read data from CPI
3) Create log files of every event

Sample Code

```
# Call the package into your session
library(rcpi)

# Set your parameters to initialize activity on the host you're connecting to
id <- "myid"
path <- "/path/to/file/"
host <- "@host1.merck.com"
identity = TRUE
identity_file = "~/.ssh/identity_file"

# Run a macro on CPI
macro_name <- "test.sas"
parameters <- list(x = "testx", y = "testy", z = "testz")
run_cpi(myid,
        host = host,
        path = path_for_macro,
        macro = macro_name,
        params = parameters)

# Read in an dataset from CPI
file_name <- "test.sas7bdat"
temp <- read_cpi(myid,
                 path = path,
                 host = host,
                 identity = identity,
                 identity_file = identity_file,
                 file = file_name)
```

The programming of all component of SVA were performed using a standard process. The process entitled to validate the computational programs using in-house analysis & reporting validation process. The standard SAS macro were leveraging to retain the re-usability/reproducibility and the data visuals was created using R script. An automated standard folder structure framework was developed with collaboration with IT to reduce to overhead for

creating folders and modifying programs. The standard folder structure provided a consistent way to program both R and SAS programs.

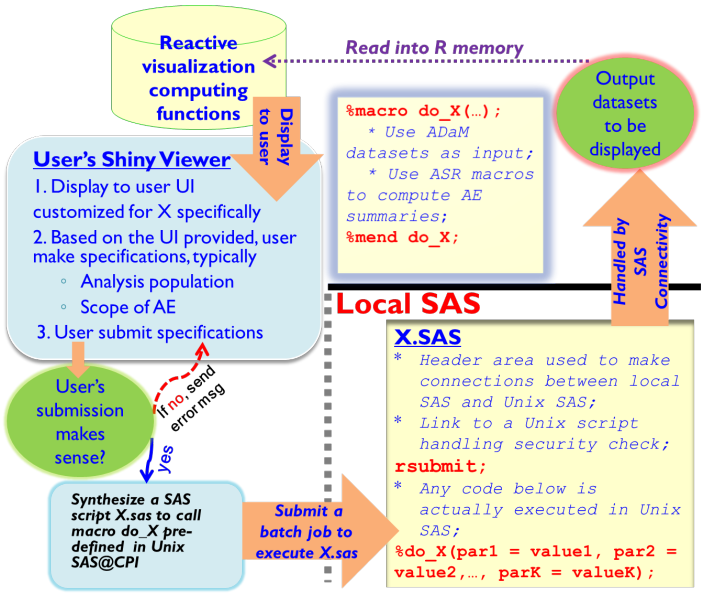

Figure 2: Automated computing flow for analysis module

The SVA system is a regulated environment and all user access are controlled. A user access provisioning process has been developed with IT to control the access to the system as well as data. The request is passed through system and require an approval before granting access. As per CRF 21 part 11 software regulations, the user access will be reviewed every six months for validity. The SVA system is validated/tested as per software development life cycle.

An automated computing has been developed as illustrated in figure 2. User will trigger a R script based on the web interface. User submits a specification using predefined parameters on the UI for the analysis module. To avoid ambiguity each analysis module is developed with range of parameters and hence the error to select an inaccurate combination is avoided.

Landing page (Figure 3) provides introduction to the tool and team of developers. The web interface of the tool is designed to provide easy access to the reporting module using R-shiny script. Shiny is an R package that makes it easy to build interactive web apps straight from R. You can host standalone apps on a webpage or embed them in R Markdown documents or build dashboards. You can also extend your Shiny apps with CSS themes, html, widgets, and JavaScript actions. (www.r-studio.org). Shiny web app is easy to use, and since it is open source, the script can be submitted to regulatory agencies, to be used on top of analysis datasets. For SVA project, a customized page was developed to enable both blinded and unblinded analysis

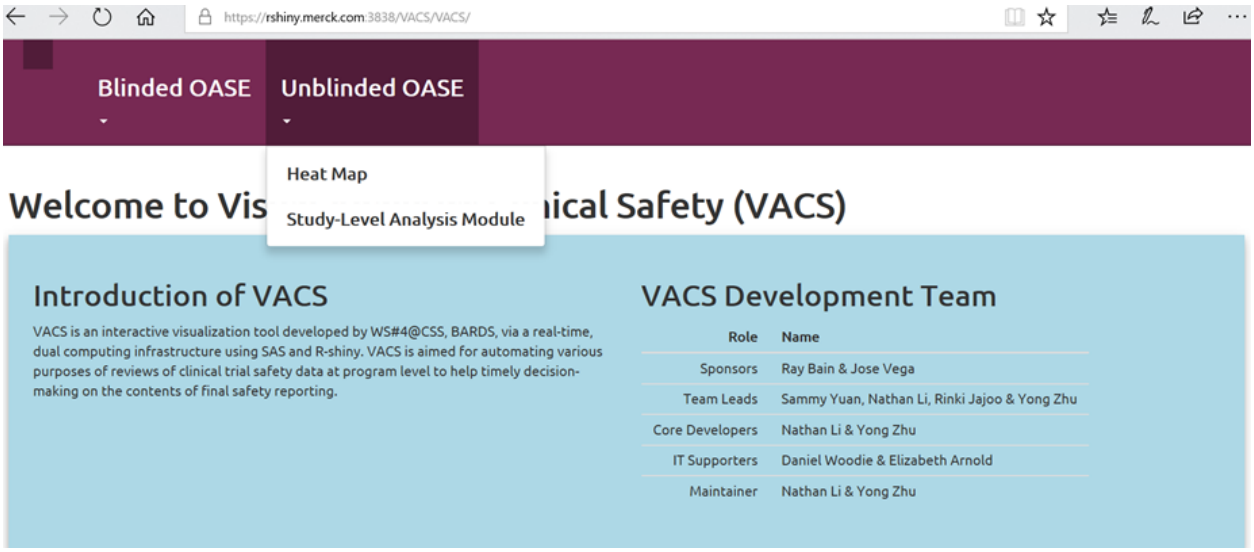

Figure 3: Landing page for Visual Analytic for Clinical Safey. The tab provides user access to the reports.

In this paper we are going to illustrate few study-level analysis module like Kaplan-Meier plot, AE profile plot and AE bar plot. SVA application is easy to navigate and only required options provided for end users. The annotation on the page provides detailed instruction for end user to navigate through system.

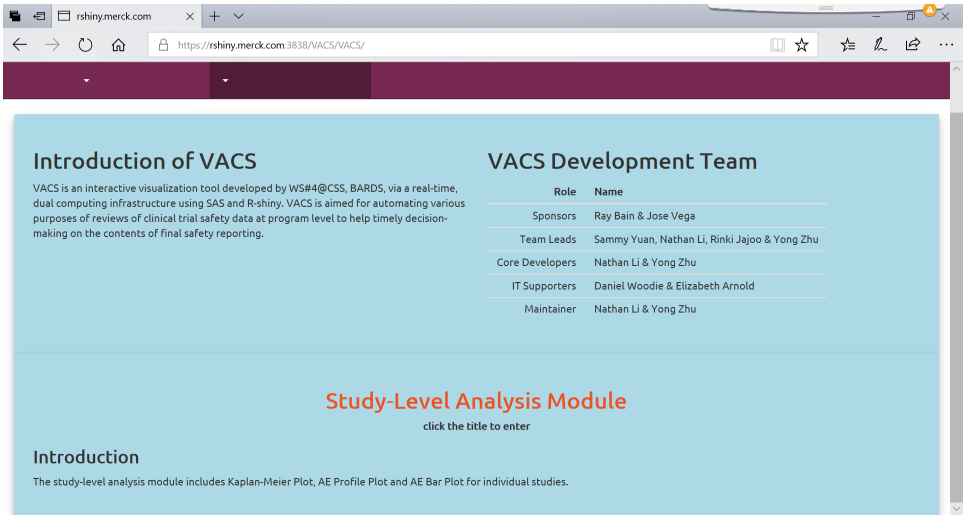

Figure 4 Study level analysis module page for user to enter. The introduction provides the scope for Study level analysis module.

AE Profile Plot often refers to Volcano Plot and Rainfall Plot, which displays between-treatment comparison results for all AEs under investigation in one plot. Volcano plot is a scatter plot of transformed p-values against the point estimate of risk difference between 2 treatment groups from a randomized study, in the sense that a safety concern should not be determined the size of p-values alone but taken the size of effort into consideration as well. Rainfall plot is a horizontal bar plot displaying the confidence intervals ordered by the size of risk difference to show potentially the most harmful and protective effects of the investigational product.

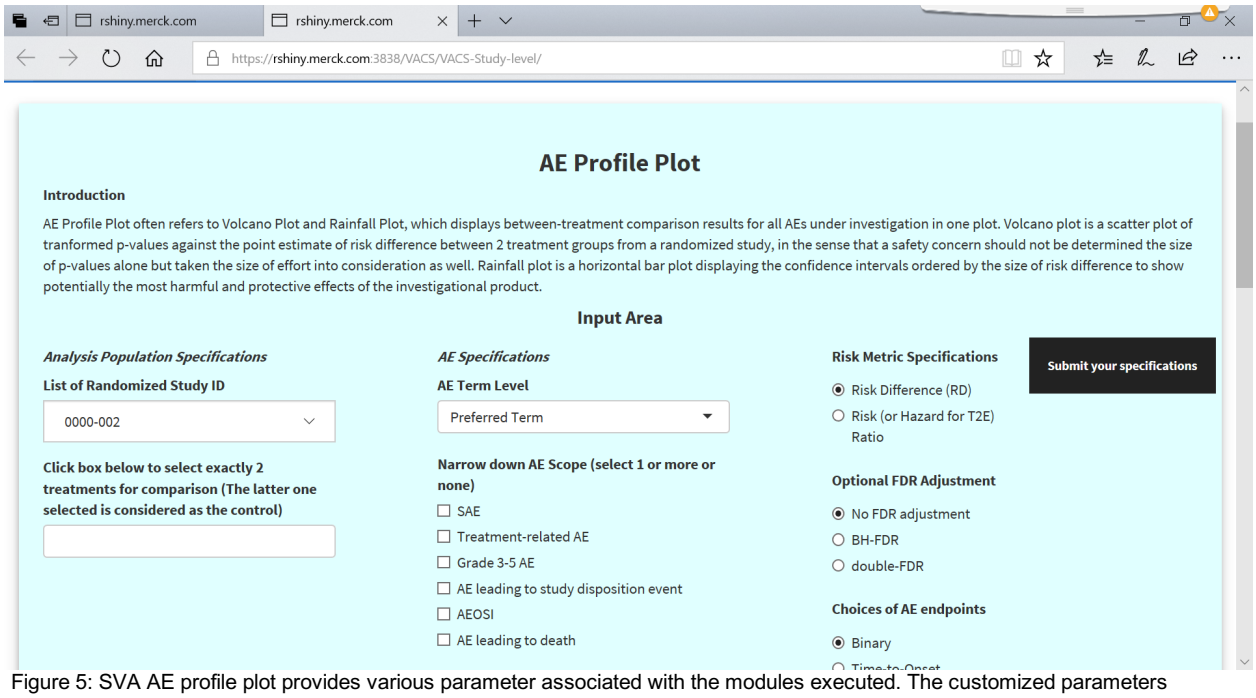

Figure 5: SVA AE profile plot provides various parameter associated with the modules executed. The customized parameters provide

In the R Shiny user interface, users can choose multiple parameters including study ID, treatments for comparison and AE term level. Besides, users can narrow down AE scope including serious AE, grade 3-5 AE, AEOSI and so on. After entering parameters, uses can submit specific parameters, and execute a SAS macro. Chosen parameters are presented in the screen shoot.

Real Time Computing

A SAS macro uses derived datasets such as ADAE and ADASL to generate datasets on CPI Unix server. As mentioned above in this paper, rCPI package provides functionality to read datasets from SAS to R and trigger the execution of SAS macros by passing parameters. The process will yield the output datasets to draw the tplots. Volcano plot and Rainfall plot are generated and presented, according to the chosen parameters.

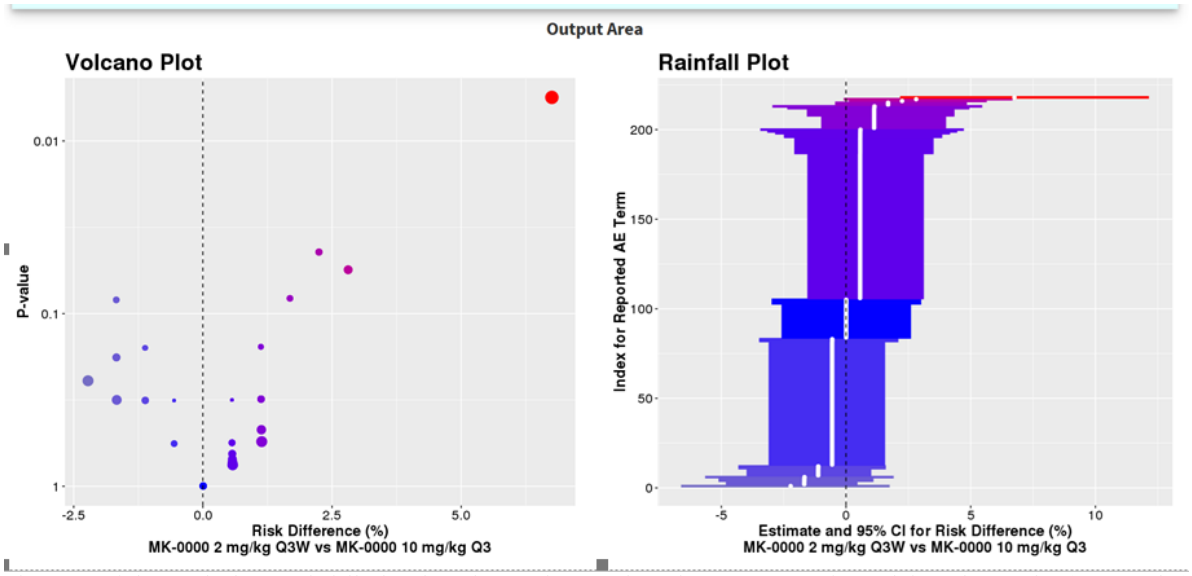

Figure 6: Right graph show Rainfall Plots based on end user selected parmaters; Left graph is Valcano plot (scatter plot) for the same parameters.

Dynamics Visualization:
R Shinny's interactive plotting features are important and useful. When mouse hovers over points in Volcano and Rainfall plots, more information about points are formally presented. We use dehydration as an example. In the table, we can see numbers of events of interest in treatments for comparison, the calculated incidence rates, risk difference and its confidence interval and p-value. So interactive plots can provide more information for users. In Rainfall plots, Kaplan-Meier plot can be generated if the point is double clicked. The Kaplan-Meier plot is generated from the Kaplan-Meier module. So SVA provides drill down features, and modules can be shared within SVA.

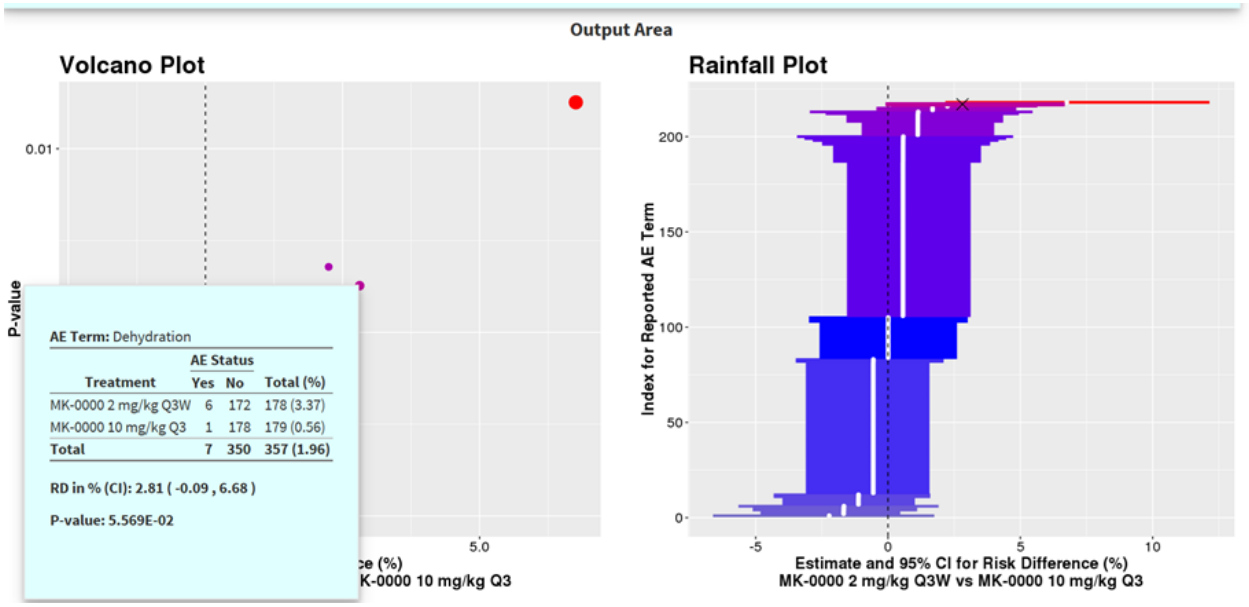

Figure 7: Dynamic features include annotation when a AE of intereset is selected. The right side panel shows Rainfall plot with a X mark, representing AE seclected in left side window Volacano plot.

In Figure 7 the AE is selected by hovering over the scatter plot (Volcano plot). The AE will dnynamically selected in the other window in this case Rainfall plot. The interactive features allow end user to drill down to specific are of intrestest. The tool use R-shiny features like annotations to display the required information. End user have capability to further link to other modules to aid in decision making. In figure 8, the AE of dehydration is further investigated using KM plot. The dynamic features are key part of SVA tool. The linkage created bewteen two different graphics are one of the key highlight of SVA.

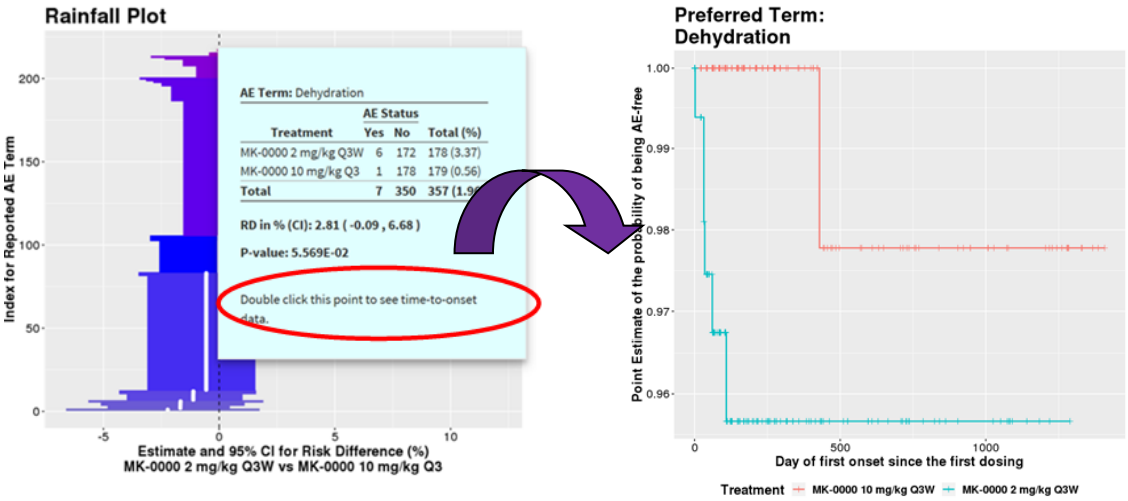

Figure 8: The selection of an AE will highlight and provide information on other graphical windows. The windows are dynamically linked to display same information at different level.

An innovative module, heat map (Fig 9) is included in SVA. The heat map module can be used to compare AEs, AEOSIs and other safety events across multiple indications, treatments, protocols and so on. In the screen sheet, we present a heat map for AEOSI by indications. Y axis stands for a list of indications, and x axis stands for a list of AEOS categories. The area of bubbles stands for incidence rates. Color stands for relative risk. In the heat map, the indication 18 is set as the comparison, and relative risk is calculated by the incidence rate in a specific indication over the indication 18. All the graphics can be downloaded in various image format (*.png or *.jpeg) or pdf

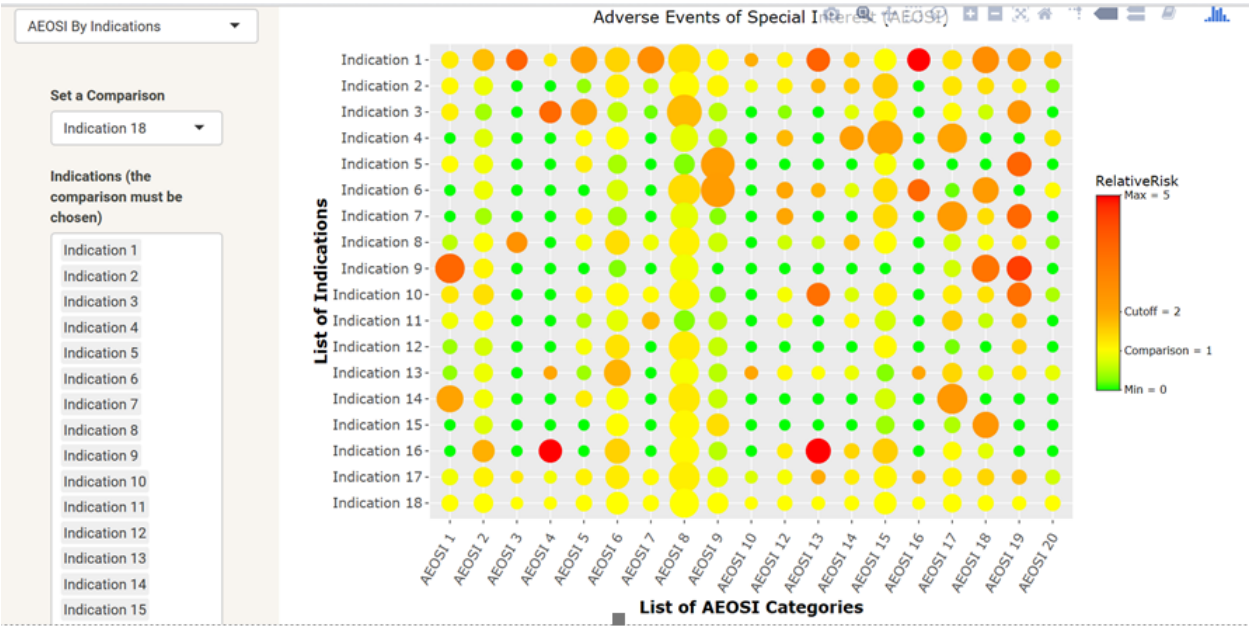

Figure 9: Heatmap for Adverse Event of Special Interset (AEOSI) by Indication

Challenges

The VACS system use R-shiny as data visualization layer. The R is known be one of the powerful tool/language. Though it comes with its own challenges on how to validate the system as well as program written in R? The VACS system is enclosed and only packages which are part of Base R validation as well as the packages has validation evidence are installed on the system. The evidence are testing by unit testing as well as by comparing results with SAS based programming.

Another challenge faced during Proof of concept, is the SAS macro are executed at run time. Every time when R program call SAS macros, the SAS program will start from the beginning of reading in data, not based on previous similar run. In other words, each SAS call is independent to the other call and hence sometimes it takes longer time to execute program and retrieve results. Also, when a SAS program is executing, there is no communication between R and SAS at that point and user may have to wait without knowing if the program is being executing. To overcome this challenge, we have developed a music playlist, which will stop playing if the program finished executing or there is an error encountered during the run. In later, a log file will be created with error for developer to debug. Also we have observed passing special characters crossing different computation platforms (R-Studio server and UNIX CPI sever) can cause problems.

There is no current guidance from regulatory on the use of source system like R in regulated enviornment. R is a language and computing platform widely used by statisiticans for computation and graphical displays. R provide many capabilities and more control to user, which may create potential validation issues for regulatory work. Currently there are mutliple initiatives started to provide guidance for R in regulated envorinment. Till we overcome these challenge of tracebility, reproducibility and validation, the compution for VACS is performed in SAS software.

Conclusion

In this paper, we introduce an innovative safety visual analytical system to aid the safety assessment performed by safety physicians, including the strategical thinking, architecture design and the basic functionalities. The SVA

system combines R-shiny as visualization tool and SAS as the computation tool. The design of SVA system enables us to achieve the goal of interactive graphs and computing on fly, which is based on the three-layer architecture structure. In the SVA system, all the computations are performed by SAS with existing validated macros to guarantee the validity, reproducibility, traceability and compliance of all analyses. However, since the computation layer is separated out from the visualization layer, SAS could be replaced by any other regulated computation tool that is accepted by regulatory agencies. The end users won't be impacted by such replacement. As some researchers pointed out, R is self-sufficient for any analyses. But the lack of fully validated functions and compliance make R not appropriate for analyses that needs full validations. In the ideal case, we hope to use R and shiny for both computation and visualization, which could be a long-term goal. However, it will need much more effort to validate a considerable number of packages needed for the analyses.

The SVA system has reached the POC milestone and been proven to a feasible solution to meet our current needs. To make it more flexible, further development is demanded. For example, we hope to mount both R-studio and SAS onto the same data storage server so that R and SAS can be integrated seamlessly, instead of using SSH key to communicate between different servers.

Acknowledgements

Safety Visual Analytics team would like acknowledge Peter Hu (Statistician, Merck & Co., Inc., Kenilworth, NJ, USA) IT department (Merck & Co., Inc., Kenilworth, NJ, USA) for their support of the project.