PharmaSUG China 2023 - Paper AA-132

A Shiny App for Clinical Data Check

Luwei Pang, MSD China, Shanghai

Ben Gao, MSD China, Shanghai

ABSTRACT

Global-multiple-center enrollment, advanced study design, and various technical measurement in clinical trials improve the accuracy and quality of human health outcomes. However, they also result in an exponential increase in the amount of clinical data. Every functional department is responsible for ensuring data quality to comply with Good Clinical Practice (GCP) and achieve precise analysis results. As statistical programmers, it is important to perform data validity and correlation check among multiple datasets during the analysis and reporting processes. This paper discusses the criteria for routine data issue checks in oncology studies and the utilization of Shiny to automatically generate data check reports and periodically monitor data issues. This Shiny application provides a user-friendly interface and enhances the flexibility to integrate and execute data check programs written in SAS. Examples for building data check criteria library, implementing Shiny application and relevant R functions are presented and explained in this paper.

INTRODUCTION

Data issues, such as inconsistencies, errors, or missing data, can arise during clinical trials due to various reasons. These issues significantly impact different aspects of the trial, particularly in the analysis and reporting stages. It is the responsibility of every functional department to identify, document and address data issues as soon as possible. In practice, the data management (DM) team utilizes many types of data check to clean the data. However, most of the checks focus on the variables in the same domain. Many crucial checks for data consistency and study-specific logic checks that cross different domains are highly in demand to ensure the accuracy of the data and the validity of statistical analysis results.

In addition to identifying the data issues, it is also essential to categorize, document and monitor the data issues on an ongoing basis. Clear categorization of data issues based on their severity, domain, and impact, as well as detailed documentation, are necessary to facilitate the DM team to query and correct the issues. Regular monitoring and tracking of the status of data issues are also crucial for effective detection and management (Su et.al., 2018). Although Excel is a commonly used tool to document data issues, it still requires time-consuming manual effort for formatting and tracking the reports. Moreover, users need to operate on several different software platforms to complete the data checking processes. An efficient tool is desired to streamline the data checking process and automate the reporting and tracking of data issues.

R is a powerful programming language and environment that enables efficient data analysis, manipulation, and interactive graphical presentation. Shiny, an R package, simplifies the process of building web application frameworks and creating user-friendly interfaces using web-based widgets like sliders, dropdowns menus, and buttons. After deploying the app centrally and hosting it on a sever, users with minimal knowledge of R can easily interact with the app and obtain results.

This paper demonstrates the process and rationales for constructing a data check criteria library, which covers the commonly used data consistency and validity checks in oncology clinical trials from the perspective of a statistical programmer. The paper also explores the concepts of developing a Shiny app to automate and streamline the process of data checking, including identifying, reporting, and tracking issues during the analysis and reporting stages. The paper provides the detailed information of the app's implementation and relevant R packages/functions.

DATA CHECK LIBRARY

Achieving data integrity in oncology studies is a challenging task due to the large amount of data collected or transmitted through various systems (such as EDC, IRT, central lab, tumor imaging) and the

diverse SDTM domains to which the collected data is mapped. Good news is that pharmaceutical companies have built standardized CRF libraries and thereafter normalized SDTM mapping methodologies, which opens opportunity to explore and generalize data checking criteria applicable to most studies across different cancer indications.

Pinnacle 21 (P21) can be used for data check, but its general check logic can sometimes result in false-positive issues. Addressing these issues requires excessive resources to review and provide rationales in cSDRG. Therefore, customized programs with stricter rules would be more efficient in outputting genuine issues.

To achieve these goals, a fine-grained summary of data checking criteria is necessary. This involves selecting source data, categorizing data issues, and implementing corresponding programs, which will be explained in more detail below.

CHOOSING SOURCE DATA

The source data that statistical programmers retrieve for analysis and reporting can differ due to distinct processes and platforms adopted at each company, ranging from pure SDTM datasets, SDTM-like datasets or even data directly from EDC systems. The data fed into this Shiny APP is SDTM-like datasets, i.e., SDTM domains with both standard SDTM variables and non-standard protocol specific variables that should have been directed to SUPP domains. This selection is based on several considerations:

1. It represents the most original data received without additional derivation or manipulation, allowing the DM team to quickly identify and pinpoint any reported issues.
2. Data is continuously refreshed, enabling real-time data monitoring.
3. The inherent SDTM-like structure facilitates P21 checks.

In short, the basic principle is to select input data that is generated in the early stage of the data flow, constantly refreshed, and bearing SDTM structure.

BUILDING CRITERIA LIBRARY

Establishing a library of common data check criteria is a continuous and incremental process. We identify two types of data issues: the ones impact analysis table results directly and the ones compromise CDISC data compliance. A feasible approach is to start from baseline and safety domains covering safety DMC reporting needs (e.g., disposition, adverse events, exposure, concomitant medications, and death in Table 1), and gradually expand the library to efficacy domains, e.g., imaging, lab etc.

The criteria should be reviewed by different study teams, optimally in different cancer indications, to ensure their applicability. Once validated, the criteria can be installed to ‘standard library’ and maintained regularly.

Data Patterns

Data completeness, consistency and correctness are the most important factors in cleaning clinical trial data. Data in each single domain should be completed without missing values, while data with intrinsic linkage or dependency across different domains should maintain consistency. Going through the data entry guidelines (DEG), Statistical Analysis Plan (SAP) and mock-up shells will help identify the critical data in Tables, Listings and Graphs (TLGs) and interrelationship among different domains. In addition to collected data, derived variables in SDTM, e.g., EPOCH, RFSTDTC, should also be checked to guarantee their correctness.

Pinnacle 21

Leveraging existing Pinnacle 21 rules contributes greatly to improving critical data quality. Pinnacle 21 has a series of death-related rules that can identify missing death information or incorrect death date, such as SD1252, which reports the records where death is collected during survival follow-up, but death form is not filled out by the site. Building standard death checking macros and running regularly before data freeze, in addition to relying solely on P21 process, can be more efficient.

Pinnacle 21 lacks tools to customize checks, so it might overreport issues. For example, SD0065 reports the unique combination of USUBJID, VISIT, VISITNUM that exist in non-SV domains but missing in SV. But the number of issues diminish greatly when erroneously entered visits in SV can be excluded. Developing a tailor-made macro that incorporates this change can report real issues, which allows the DM team to spend much less time reviewing and querying cases.

Category	Domain(s)	P21 Rule ID	Issue Description
AE	AE		AETOXGR='5' and AEOUT='FATAL' and AESDTH='Y' - should be consistent.
AE	AE	SD1332	AEOUT='NOT RECOVERED/NOT RESOLVED' but AEENDTC is populated.
AE	AE	SD1333	AEOUT='RECOVERED/RESOLVED', but both AEENDTC and AEDUR are missing.
AE	AE, CF		AE and CF cross-check (merge by USUBJID, AESPID/CFSPID): · AE records not in CF; CF records not in AE. · Minimum date of CFDTC not equal to AESTDTC; maximum date of CFENDTC not equal to AEENDTC.
Death	AE, CE	SD1347	· AEENDTC for fatal AE is not the same as DTHDTC. · AEENDTC where AETOXGR='5' or AEOUT='FATAL' or AESDTH='Y' is NOT equal to CESTDTC where CEGRPID='DEATH' and CESTDTC ^=''.
Death	CE, CF	SD1252	CFGRPID in ('SFU', 'VST') and CFSTRESC='DEAD', but not records in CE domain where CEGRPID='DEATH' and CESTDTC ^= ''.
Death	CE, DS	SD1256	DSDECOD='DEATH', but not records exist in CE domain where CEGRPID='DEATH' and CESTDTC ^= ''.
Death	CF, DS	SD1316	CFGRPID in ('SFU', 'VST') and CFSTRESC='DEAD', but not records in DS domain where DSDECOD='DEATH'.
Disposition	DS, EX		Subject has treatment records in EX and discontinues from trial, but no treatment discontinuation records exist in DS.
Exposure	EX		· Unblinded: missing any of EXTRT, EXDOSTOT, EXNUMDOS, EXDOSE, EXSTDTC. · Blinded: missing any of EXNUMDOS, EXSTDTC.
Exposure	EX		For combination therapy, check drug component level disposition status versus overall disposition status.
IE Criteria	IE, DS, DM		Non-randomized subjects with no I/E criteria recorded in IE domain.
Epoch	SE		SESTDTC is less than previous element's SEENDTC.
Visits	SV	SD0065	Unique combination of (USUBJID, VISIT, VISITNUM) exists in non-SV domain, but missing in SV domain (remove SV records where ERRVISIT='ERRONEOUS VISIT').

Table 1 Data Issue Examples

Extensibility

Each study may have its special endpoints that brings new or non-standard domains, so developing protocol-specific checking macros to investigate and check those domains are usually needed. There are no strict requirements on the macro implementations so long as its output naming can be recognized by the APP, e.g., prefixed uniformly.

SET UP SAS PROGRAMS

Building on top of the summarized standard criteria, the next step is to create 'standard' SAS programs invoked by the Shiny app. The SAS programs mainly consist of core macros identifying data issues and macro calling programs.

Standard SAS macro

Optimally, each macro will focus on a single issue category with individual domain(s), e.g., 4 SAS macros were created to address the above 4 different death criteria, each with its own set of domains. This approach enhances the macro's readability and maintainability, making it easier to add analogous checks to the same macro in the future. Additional parameters can be added to boost the macro's compatibility. For example, there is no point in checking masked variables like EXTRT, EXDOSE, EXDOSTOT under a blinded environment. Users can set macro parameter *blinded* to N to bypass checks on those variables.

The macro outputs should follow a consistent naming convention, specified in parameter *rename_outdata*, to enable downstream utility programs to identify the datasets without ambiguity. For example, prefix 's0' can be used for standard macros, while 'p0' can be used for user-crated specific checking macros.

DM team always prepares a workbook for the study team to enter data issues, demanding identifier information like USUBJID, DOMAIN, XXGRPID (form name), XXSPID. It is convenient to have macros render columns in the same order as that workbook.

```
%macro s0chk0ex001(
  outdata_libref=work
 ,rename_outdata=s0chk0ex001
 ,blinded=Y
 ,exvars_plus=
 ,timespan=
 ,debug=N);
```

Calling programs

Calling programs are used to group macro calls classified under the same category and enable parameter adjustment for customized checks (Table 2). SAS program logs and lists will be specified in this call, which will be looked up and displayed by APP's log checker.

Category	Call Program	Macro
Death	call0s0chk0death.sas	s0chk0death001.sas
		s0chk0death002.sas
		s0chk0death003.sas
		s0chk0death004.sas

Table 2 Macro and Calling Program

DATA CHECK SHINY APP

The Shiny app's user interface (Figure 1) comprises three main components:

1. The Inputs component provides users the flexibility to select inherent standard data issue check criteria and project-specific data check programs written in SAS based on their business needs.
2. The Log Checker component verifies log files and enables users to detect any errors or warnings in the log file.
3. The Output component displays the data issue check reports, tracks the status of the data issues (e.g., 'NEW', 'UNSOLVED', 'SOLVED') and allows users to filter and sort the results easily.

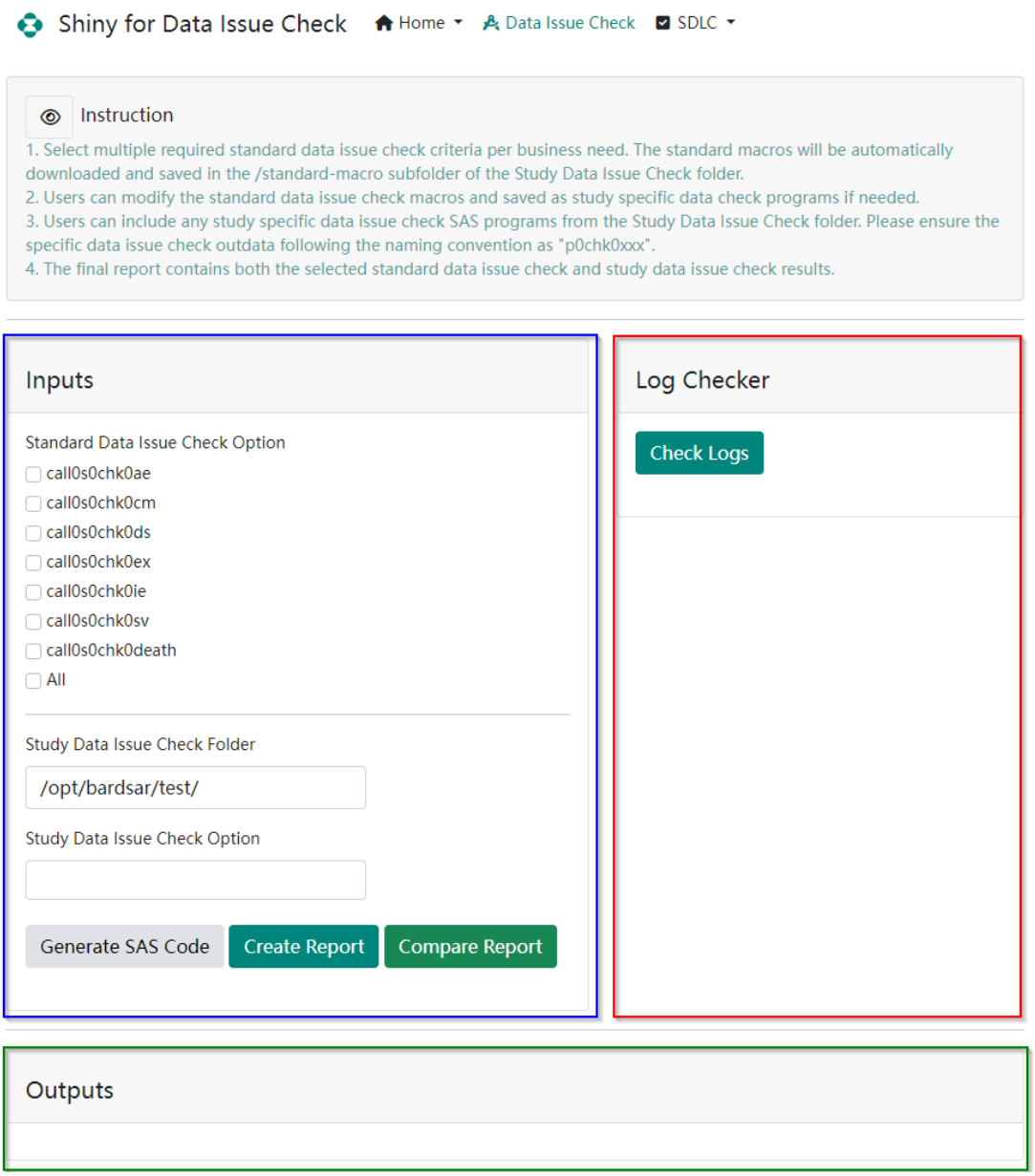

Figure 1. User Interface of the Shiny app

The Shiny app offers several additional benefits to automate and optimize the routine data check process in the data analysis and report stages, including:

- Web-based: The entire operating and reviewing process is web-based, requiring minimal knowledge of R and saving time by eliminating the need to open documents or operate programs in different software.
- Flexibility: The app allows integration and operation of any project-specific data check program written in SAS with selected data issue check criteria to generate reports. Users can monitor the status of data issues by comparing the current report with any previous version.
- Efficiency: In addition to web-based reports, the app also generates well-structured reports in xlsx format, reducing manual efforts and minimizing human errors.
- Verification: The standard data issue check criteria are validated for accuracy and reliability.

PROCESS OVERVIEW

The Shiny app workflow involves three key steps:

1. Input: Users provide information such as the study data issue check folder, the required standard data issue check criteria, and any study-specific data check programs. Additionally, users can select a previous version of the reports to monitor data issue status.
2. Processing: The app processes the input information and establishes a secure shell (SSH) connection from R to a remote sever to execute SAS code.
3. Output: The app generates and presents the reports and log files. Users can verify the log file and review the reports to ensure data quality.

PROCESSING

To ensure data validity, it is essential to include study-specific data issue check criteria in addition to the routine standard data check rules, given the variety of clinical indications and study designs. Once users have selected the necessary standard and study-specific data issue check criteria, the app processes the information and generates a modified SAS program. The program incorporates standard and project-specific SAS programs, sets up the study working path, and creates log files. Meanwhile, the app downloads the standard data check programs to the study folder using `R.utils::copyDirectory()`. The app then provides the flexibility to execute SAS programs, as most analysis and reporting processes are conducted using SAS software. This enables users to customize the data issue check process to meet their specific study needs and ensure a high level of data quality.

Establish SSH connect and execute command

The app uses `SSH::ssh_connect()` function to establish an SSH connection from R to the company's remote sever. Users are required to log in with a password, and once authenticated, the app submits a command to execute the SAS program. `SSH::ssh_exec_wait()` function is used to submit the command and wait for it to finish before closing the connection.

```
# Set the command to execute the SAS program
command <- paste0('sas94 ', sas_file())
# Connect to the remote server via SSH using the ssh package
session <- ssh_connect("remote server", passwd=l$pwd)
# Submit the command to execute the SAS program and wait for it to finish
ssh_exec_wait(session, command = command)
# Disconnect the SSH connection
ssh_disconnect(session)
```

Create data issue reports

After applying the standard and study-specific data check criteria, the dataset frames that exhibit different data issue categories will be displayed as separate tabs in the Shiny app (Figure 2). These tabs will be created using the `tabPanel()`, `tabsetPanel()`, and `DT::dataTableOutput()` functions.

```
# Display each data issue category in tabsetPanel
output$tbl <- renderUI({
    listtab <- list(NULL)
    for (i in (1:nsheets)){
      listtab[[i]] <- tabPanel(sheets()[i],
                               dataTableOutput(paste0("table",i)))
    }
    do.call(tabsetPanel, listtab)
  })
```

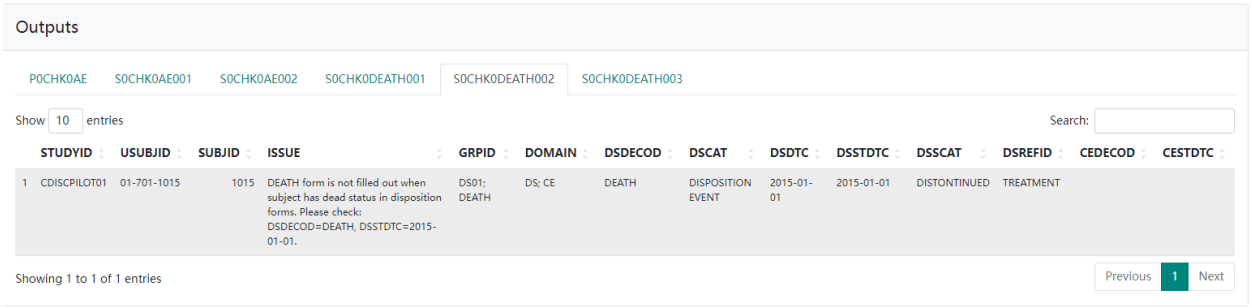

Figure 2. Data issue reports displayed in Shiny app

To periodically monitor the status of data issues and enable the study team to follow up and address them at sites, multiple rounds of data issue checks are required for a long-running trial. The app enables users to select any previous reports stored in the study folder and compare them with the most recent one. To streamline the data issue check process, two additional columns containing data issue status and the date of the previous report version will be added to the data issue follow-up reports (Figure 3). This feature significantly reduces the amount of time and effort required for manual updates and formatting of reports.

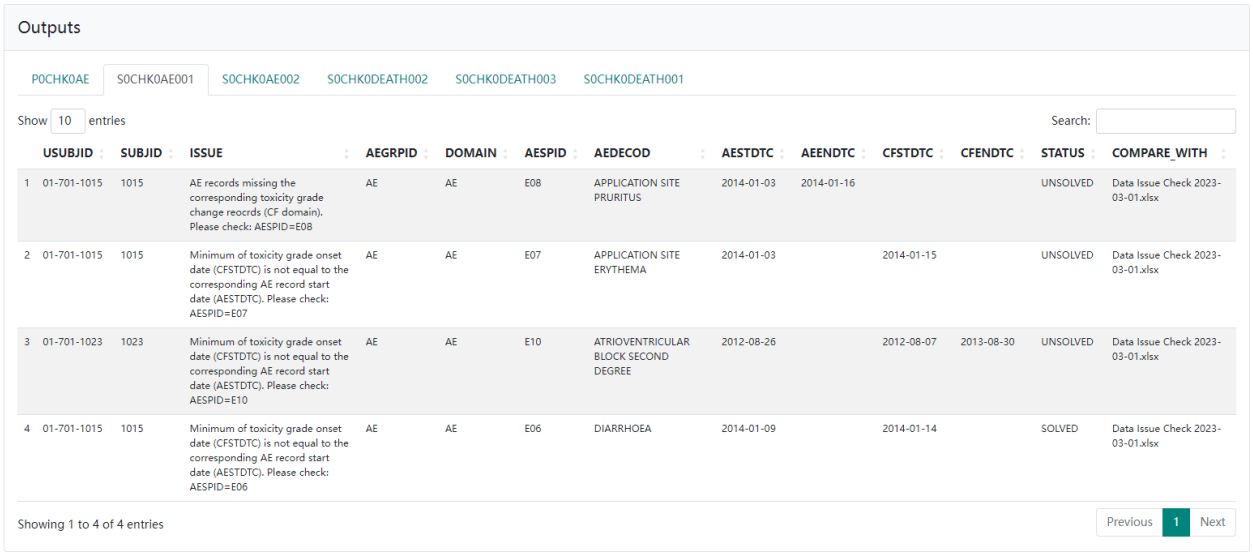

Figure 3. Data issue follow-up reports displayed in Shiny app

In addition to the reports that are displayed within the app, the corresponding Excel reports are also generated and saved in the study folder using the `openxlsx::write.xlsx` function.

```
# Generate the reports in Excel
hs <- createStyle(
  textDecoration = "BOLD", fontColour = "#FFFFFF", fontSize = 12,
  fontName = "Arial Narrow", fgFill = "#4F80BD"
)

openxlsx::write.xlsx(comp_output, file = compare_file(),
                     sheetName=names(comp_output),
                     withFilter=TRUE, headerStyle = hs, colWidths="auto")
```

CONCLUSION

This paper demonstrates the common data issue check criteria utilized in oncology studies and the development of a Shiny app. The app automates and simplifies the clinical data check process during the analysis and reporting stages. It allows the integration and operation of selected standard and study-specific data issue check criteria written in SAS, presents the reports directly and generates corresponding Excel files in the study folder to document and track data issues. The app saves programmers significant time by eliminating the need for manual generation of reports and tracking of data issues. Moreover, it facilitates the entire study team in addressing potential data issues promptly and efficiently, ensuring high quality and validity of clinical data and statistical analysis results.

REFERENCES

Chao Su, Shunbing Zhao, Cynthia He, “An Efficient Tool for Clinical Data Check”, *PharmaSUG 2018*, Paper AD16.

ACKNOWLEDGMENTS

We would like to thank MSD China statistical programming team for the invaluable input of data issue check criteria and testing of the Shiny app. And special thanks to Nicole Zhang and Danfeng Fu for their insightful comments and review of this paper.

CONTACT INFORMATION

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

Luwei Pang
MSD China
lu.wei.pang@merck.com

Ben Gao
MSD China
peng.gao4@merck.com