

Automation of Standard ADaM Variables Through an R Package – *GenAdam*

Mahesh Divakaran, Genpro Research, Trivandrum, India
Mohin K Mohanan, Genpro Research, Trivandrum, India
Denny Sebastian, Genpro Research, Trivandrum, India

ABSTRACT

For the last couple of years, R software applications in clinical research have been a hot topic in the pharma industry. It offers a variety of user-developed packages containing functions which efficiently manipulate complex data sets and create tables, figures, and listings, perfect for biostatisticians and programmers in the pharmaceutical and biotech industry. But the development of CDISC compliant datasets using specifications or define as the base is a time-consuming manual effort be it through SAS or R. In this paper, we introduce an R package that can be an effective alternative in creating clinical trial ADaM datasets with a single function call followed by visualizing the created dataset. The entire concept spans across two phases. The first phase, where the focus of this paper lies, creates ADaM datasets with standard variables & predecessors presented. The second phase takes as input the study specification and maps variables to corresponding datasets.

Keywords: R, data visualization, clinical trial data, ADaM, R shiny, and clinical analysis.

INTRODUCTION

The world has seen a decrease in mortality and morbidity as a result of improvements in basic clinical research from healthcare organizations and institutes, medical professionals, and clinical trial participants. A new drug's safety and effectiveness are evaluated in clinical trials. The most adaptable and practical data to be gathered is clinical data. Clinicians and statisticians continuously assess the clinical outcomes throughout the course of a study to find the most effective course of care.

In order to increase the effectiveness of clinical research, the Clinical Data Interchange Standards Consortium (CDISC), a nonprofit organization created in 1997, is working to develop data standards for collecting, evaluating, and communicating clinical data. The Study Data Tabulation Model (SDTM) and the ADaM (Analysis Data Model) are two of the principal standards that CDISC supports. Even though CDISC standards implementation can increase development costs, you can reduce the costs by choosing the appropriate technology and implementation process. The adoption of CDISC standards also stimulates innovation, increases efficiency, enhances data quality, lowers costs, and increases predictability, in addition to streamlining operations.

The Analysis Data Model (ADaM) is one of the most crucial requirements for clinical trial submission. The creation of analysis datasets and related metadata is described in ADaM standards. This ensures traceability and makes it easier for a statistical programmer to produce figures, listings, and tables. As a result, reviewers can evaluate and approve a submission more rapidly.

With the help of the R package Shiny, it's simple to create interactive web applications straight from R. It offers the framework needed to study and analyze the results of clinical trials. Listings, tables, and figures can be dynamically presented and customized to make filtering and data visualization easier. These features are compatible with other programs (such as DT, plot.ly). We could deploy the R shiny apps under both the local and public domains after the app finished the setup process. This paper discusses creating standard ADaM variables using an R package-GenAdam; that visualizes the ADaM datasets using R shiny, and we can directly download these datasets in sas7bdat format.

ADAM: ADSL AND BDS BACKGROUND

The ADaM specifies dataset and metadata standards that facilitate the ability to quickly build, replicate, and evaluate clinical trial statistical analyses and trace such investigations' outcomes across SDTM data. ADaM is one of the requirements for submitting data to the FDA (U.S.) and PMDA (Japan). The ADaM Implementation Guide (ADaMIG) v1.3 defines three classes of datasets: analysis datasets, ADaM datasets, and non-ADaM analysis datasets. The categories of Analysis Datasets and their relationships are shown in Figure 1. An analysis dataset is either an ADaM dataset or a non-ADaM analysis dataset. ADaM datasets conform to three standard structural classes:

- ADSL (Subject-Level Analysis Dataset)
- BDS (Basic Data Structure)
- OCCDS (Occurrence Data Structure)

Apart from the above mentioned three classes, we have ADaM Other (OTHER), which is an analysis set that follows general ADaM principles and conventions but does not fit inside any of the 3 defined standard structural classes.

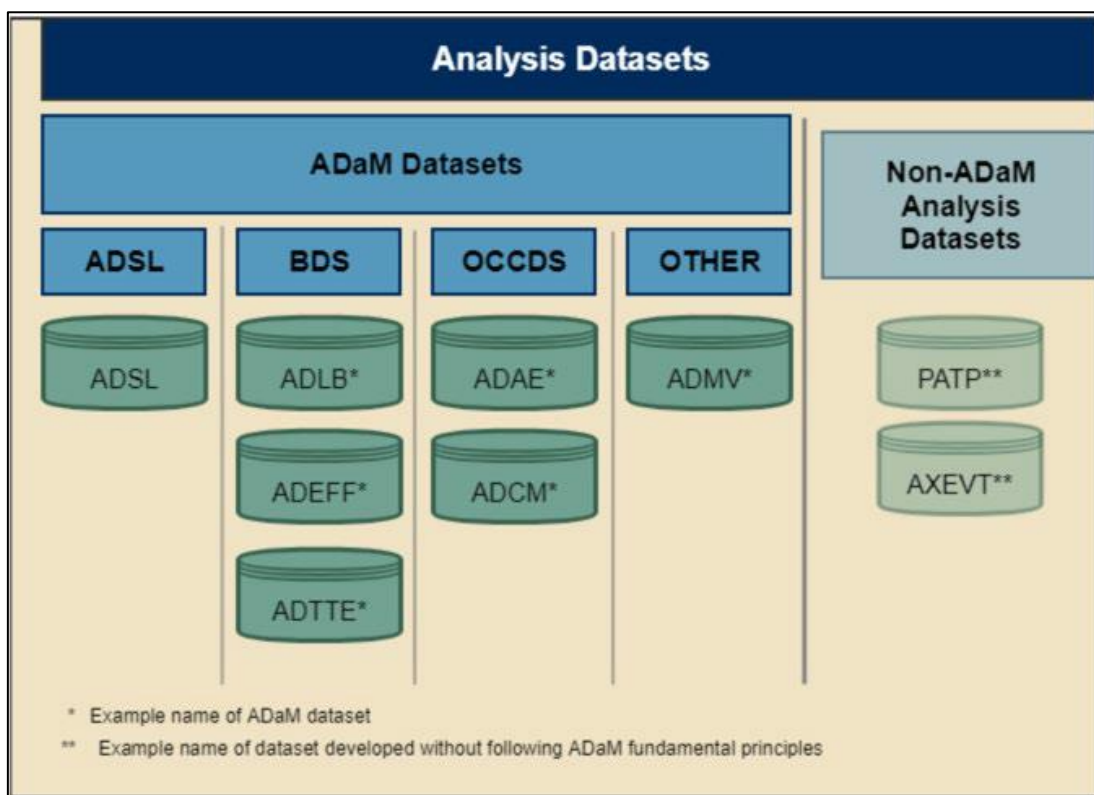


Figure 1: Categories of Analysis Datasets (Source: ADaMIG v1.3)

THE SUBJECT-LEVEL ANALYSIS DATASET (ADSL)

ADSL contains subject-level data organized into "one record per subject," including population flags at the subject level, treatment variables (planned and actual), demographic data, stratification and subgrouping factors, and significant dates. Even if no other ADaM datasets are supplied, CDISC-based data submission in accordance with FDA SDTCG (Study Data Technical Conformance Guide) must include an ADSL and its associated metadata information. ADSL serves as the source for all other ADaM datasets' subject-level variables, including population flags, treatment variables, subgrouping variables, and statistical model covariates. Common variables in ADSL are:

- Identifier Variables: STUDYID, USUBJID, SUBJID, SITEID,
- Subject Demographics Variables: AGE, AGEU, SEX, RACE
- Treatment Variables: ARM, TRTxxP

THE BASIC DATA STRUCTURE (BDS)

BDS contains the information of various parameters, its results and other dependent variables. "One or more records per subject, per analysis parameter, and per analysis timepoint." Common variables in BDS datasets are:

- Identifier Variables: STUDYID, USUBJID
- Analysis Parameter Variables: PARAM, PARAMCD, AVAL, BASE, CHG

ADOPTION OF R FOR ADAM DATASETS

R software's use in clinical research has been a hot topic in the pharmaceutical business for a while. R is utilized for exploratory analysis, routine data reading, and interpretation tasks but not commonly for creating TFL or SDTM/ADaM. In order to do so, we need to consider the following issues.

What are the difficulties in producing submission artefacts in R?

Are there any developments in conquering these obstacles?

What advantages does R have over SAS®?

THE CHALLENGES:

Validation: R is a free and open-source program, and anybody can create and publish R packages to CRAN. In 2015, the US FDA published a clarification statement on statistical software. The FDA does not mandate the use of any particular statistical analysis program. Software program used for statistical analysis, though, must be fully disclosed in

the application, together with their version and build numbers. In addition, documentation of the proper software testing techniques should exist. It necessitates the validation of R packages and the preservation of appropriate records for audit. However, SAS is a licensed piece of software, and the SAS Institute is in charge of validation.

Legacy Data: SAS has been utilized for many years in the pharmaceutical business, and the majority of the data from legacy research is saved in the .sas7bdat format. R or any other computer language, such as Python, is unable to read this data in its entirety, including all formats, informats, and other information. I could only read this data without knowing its format. The difficulty is what motivates SAS to be able to read legacy data.

R Support: Unless a user purchases the commercial editions of R studio or R Shiny from the service providers, there is no support available to assist customers with installation or package issues with R.

BENEFITS OF R OVER SAS:

Open Source: SAS is not free, whereas R is. The price of R's enterprise-level validation and documentation, however, has not yet been estimated. It will likely cost far less than the price of SAS license.

Powerful Tool: R is an excellent tool for data analysis because it is open source, can be compiled on a variety of systems, offers a vast array of statistical and graphical techniques, and generates plots of publishable quality. With R-shiny, the majority of daily tasks can be automated.

SETUP R ENVIRONMENT

The procedures for setting up the environment needed to run the R packages are listed below. Installing these four items is necessary:

- **R:** An environment for statistical computation. It includes a straightforward user interface that functions largely as a command-line interface.
- **Rtools:** A collection of applications needed to create R packages from source on Windows.
- **Rstudio:** R is made simpler to use by rstudio, an IDE (Integrated Development Environment). It has a code editor as well as visualisation and debugging tools. Please utilise it to have a pleasant R experience.
- **Java:** Some of the OHDSI R package components, such as those required to connect to a database, require the Java computing environment.

INSTALLING R

Go to <https://cran.r-project.org/>, then click the Download link compactable to your operating system (OS). After the download has completed, run the installer and accept all of the default options. You should see a screen indicating that the installation was successful.

INSTALLING RTOOLS

Go to <https://cran.r-project.org/>, then go to 'Rtools' download page and select the very latest version of RTools compactable to your operating system (OS) to download. After the download has completed, run the installer and accept all of the default options. You should see a screen indicating that the installation was successful. Since R v4.0.0 you also need to add the path to your .Renv file, as described on the RTools page.

INSTALLING RSTUDIO

Go to <https://www.rstudio.com> or <https://posit.co/downloads/>, then select 'Download RStudio' (or the 'Download' button under 'RStudio'), opt for the free version, and download the installer compactable to your operating system (OS). After the download has completed, run the installer and accept all of the default options. You should see a screen indicating that the installation was successful.

INSTALLING JAVA

Go to <https://www.oracle.com/java/technologies/javase-jdk15-downloads.html>, and select the installer for the Oracle JDK for your operating system (OS). After downloading just run the installer. Check that java is installed by opening the terminal and running the command *java*. If you see some helpful output about usage then the installation was successful.

GENADAM PACKAGE

SAS offers a potent programming tool called *Macro* that lets us avoid writing repetitive code and reuse it repeatedly when required. Additionally, it aids in the creation of dynamic variables within the code that can accept various values depending on the run instances of the same code. Similarly, R has dynamic functions, which makes the programmer's life easier. An R package collects R libraries, datasets, and documentation. R is an open-source programming language allowing users to create and publish their packages. Many packages are available today to derive parts of clinical data reporting, which helps reduce programming time and effort. Here, we introduce the GenAdam package which allows us to generate the standard ADaM variables by just a function call.

The GenAdam R package can be a helpful substitute for constructing clinical trial ADaM datasets by calling a single function along with visualizing the resulting dataset in a shiny app interface. There are two phases to the whole package development.

Phase 1 - We create ADaM datasets with standard variables and predecessors.

Phase 2 - Involves mapping variables to appropriate datasets using the research definition and metadata as input.

In this paper we focus only on the first phase specifically on the development of ADSL and BDS dataset structures using the example of a single period study.

The GenAdam package is focused to run one line to generate and visualize the ADaM dataset. We use the R package *haven* to read the SDTM input datasets in sas format(.sas7bdat). Most of the data manipulation part are done using the *tidyverse* family of packages and *data.table*. The *Hmisc* package was used to add the labels to the datasets and add attributes. We have used the SDTM datasets to generate the standard variables and derived specific formats for each variable. Some of the variables are derived using the CDISC code lists. We pass the codelist as metadata, which is similar to a codelist tab in our ADaM specifications, to derive these variables.

You can install the Genadam package in R by running the below command.

```
install.packages("<source_package_path>/GenAdam_0.1.0.tar.gz", repos = NULL, type = "source")
```

STRUCTURE OF GENADAM

This package phase mainly has five functions, which perform the visualization of input SDTM datasets and the creation of 4 ADaM datasets (ADSL, ADVS, ADLB, ADQS). Let's look into each of them.

1. SDTM_DATA FUNCTION

The `sdtm_data` function displays the dataset in R shiny. We will be able to filter, search and explore the data in the interactive visualization feature of R shiny and datatable (DT).

The structure of the `sdtm_data` function is:

```
sdtm_data(folder_path, file_name) .
```

Here the function takes two input arguments;

`folder_path` : Folder path of input datasets (Required)

`file_name` : File name of the dataset (Required)

Example of the DM dataset visualization using the `sdtm_data(<folder_path>, "DM")` is shown below.

	STUDYID	DOMAIN	USUBJID	SUBJID	RFSTDTC	RFENDTC	RFXSTDTC	RFXENDTC	RFIDTC	RFPENDTC	DTHDTC	DTHFL	
1	TRID-02	DM	TRID-02-801-390	801-390	2020-07-02T09:35	2020-07-28	2020-07-02T09:35	2020-07-28		AI		80	
2	TRID-02	DM	TRID-02-801-077	801-077							2020-06-22	2020-07-06	80
3	TRID-02	DM	TRID-02-801-125	801-125	2020-07-15T10:00	2020-08-11	2020-07-15T10:00	2020-08-11			2020-07-06	2020-09-10	80
4	TRID-02	DM	TRID-02-801-814	801-814	2020-08-05T08:07	2020-09-01	2020-08-05T08:07	2020-09-01			2020-07-17	2020-10-02	80
5	TRID-02	DM	TRID-02-801-670	801-670							2020-07-20	2020-08-28	80
6	TRID-02	DM	TRID-02-801-451	801-451	2020-09-15	2020-09-15	2020-09-15	2020-09-15			2020-08-03	2020-09-22	80

Figure 2: DM dataset visualization using the `sdtm_data(<folder_path>, "DM")`

2. ADSL FUNCTION

The `adsl()` function creates the ADSL dataset with the standard variables. This function will work only with single-period simple to moderate studies, which is a package limitation. But the developing phase II will be able to overcome these limitations.

The structure of the `adsl()` function is:

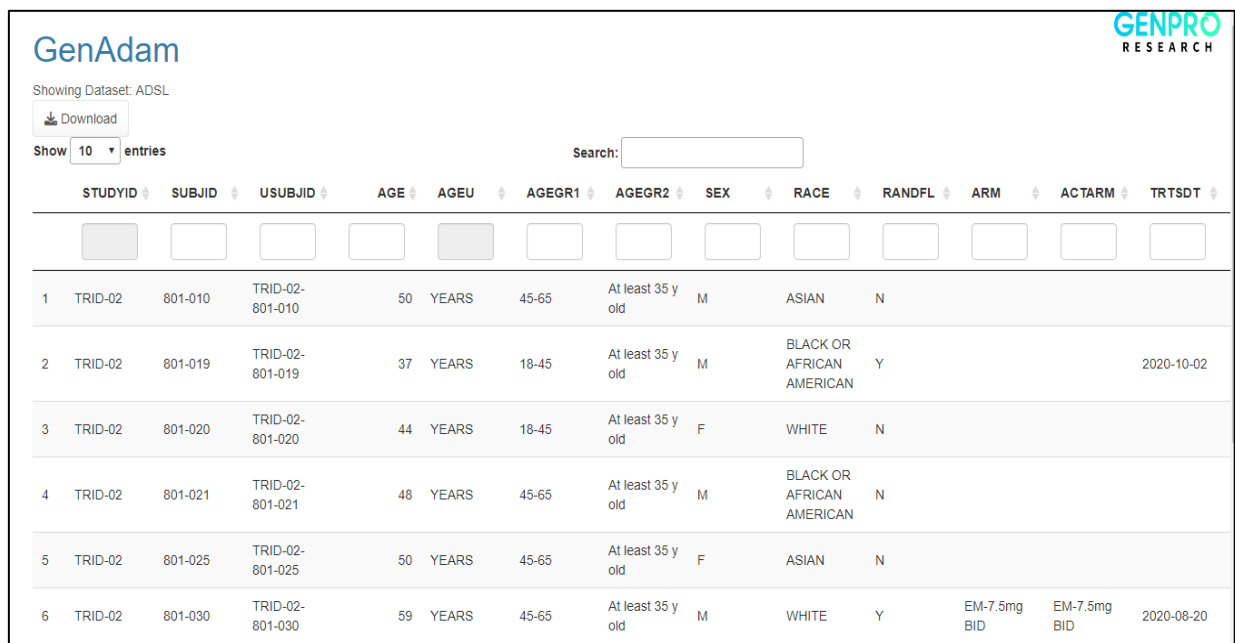
```
adsl(folder_path, varlist, metadata)
```

Here the function takes three input arguments;

`folder_path` : Folder path of input datasets (Required)
`varlist` : list of variables to be output (Optional)
`metadata` : Metadata* filename and Path (Optional)

* Metadata is similar to the codelist sheet of the ADaM Specification.

Example of the ADSL dataset visualization using the `adsl(<SDTM_folder_path>, "metadata.xlsx")` is shown in Figure 3.



GenAdam
Showing Dataset: ADSL
Download
Show 10 entries
Search:

	STUDYID	SUBJID	USUBJID	AGE	AGEU	AGEGR1	AGEGR2	SEX	RACE	RANDFL	ARM	ACTARM	TRTSDT
1	TRID-02	801-010	TRID-02-801-010	50	YEARS	45-65	At least 35 y old	M	ASIAN	N			
2	TRID-02	801-019	TRID-02-801-019	37	YEARS	18-45	At least 35 y old	M	BLACK OR AFRICAN AMERICAN	Y			2020-10-02
3	TRID-02	801-020	TRID-02-801-020	44	YEARS	18-45	At least 35 y old	F	WHITE	N			
4	TRID-02	801-021	TRID-02-801-021	48	YEARS	45-65	At least 35 y old	M	BLACK OR AFRICAN AMERICAN	N			
5	TRID-02	801-025	TRID-02-801-025	50	YEARS	45-65	At least 35 y old	F	ASIAN	N			
6	TRID-02	801-030	TRID-02-801-030	59	YEARS	45-65	At least 35 y old	M	WHITE	Y	EM-7.5mg BID	EM-7.5mg BID	2020-08-20

Figure 3: Output of ADSL from the `adsl()` function

You can access the ADSL dataset in `.sas7bdat` format by clicking the download button. We can also sort the variables and filter the values in each column. We have a search option that allows you to search the table's values. We derived `AGEGR1` and `AGEGR2` values using the code list/metadata. Additionally, we have a `varlist` parameter to specify the output variables. All standard variables are assumed to be defined by default.

3. ADVS FUNCTION

The `advs()` function creates the ADVS dataset with the standard variables and predecessor variables from ADSL. Here we develop a list of standard variables associated with the ADVS dataset.

The structure of the `advs()` function is:

```
advs(folder_path, advs_var, metadata)
```

Here the function takes three input arguments;

`folder_path` : Folder path of input datasets (Required)
`advs_var` : list of variables to be output (Optional)
`metadata` : Metadata/ Codelist filename (Optional)

Example of the ADVS dataset visualization using the `advs(<SDTM_folder_path>)` is shown in Figure 4.

GenAdam
Showing Dataset: ADVS
Download
Show 10 entries
Search:

	STUDYID	SUBJID	USUBJID	ADT	ATM	ADTM	PARAMCD	PARAM	ABLFL	BASE	AVAL	CHG	PCHG	TRT01P
11	TRID-02	801-019	TRID-02-801-019	2020-10-01	13:28	2020-10-01T13:28	DIABP	Diastolic Blood Pressure		74	79	5	6.75675675675676	
12	TRID-02	801-019	TRID-02-801-019	2020-10-01	13:42	2020-10-01T13:42	DIABP	Diastolic Blood Pressure		74	79	5	6.75675675675676	
13	TRID-02	801-019	TRID-02-801-019	2020-10-01	13:56	2020-10-01T13:56	DIABP	Diastolic Blood Pressure		74	77	3	4.05405405405405	
14	TRID-02	801-019	TRID-02-801-019	2020-10-01		2020-10-01	DIABP	Diastolic Blood Pressure		74	78	4	5.40540540540541	
15	TRID-02	801-019	TRID-02-801-019	2020-10-01	13:30	2020-10-01T13:30	DIABP	Diastolic Blood Pressure		74	83	9	12.1621621621622	
16	TRID-02	801-019	TRID-02-801-019	2020-10-01	13:44	2020-10-01T13:44	DIABP	Diastolic Blood Pressure		74	78	4	5.40540540540541	

Figure 4 Output of ADVS from the advs() function

We have an option to download the data in .sas7bdat format using the download button and it also enables the options to search and filter values directly from the app itself.

4. ADLB FUNCTION

The adlb() function creates the ADLB dataset with the standard variables and predecessor variables from ADSL. Here we develop a list of standard variables associated with the ADLB dataset.

The structure of the adlb() function is:

```
adlb(folder_path, adlb_var, metadata)
```

Here the function takes three input arguments;

folder_path : Folder path of input datasets (Required)

adlb_var : list of variables to be output (Optional)

metadata : Metadata/ Codelist filename (Optional)

Example of the ADLB dataset visualization using the adlb(<SDTM_folder_path>) is shown in Figure 5.

GenAdam
Showing Dataset: ADLB
Download
Show 10 entries
Search:

	STUDYID	SUBJID	USUBJID	ADT	ATM	ADTM	PARAMCD	PARAM	ABLFL	BASE	AVAL	CHG	PCHG	TRT01P
71	TRID-02	801-019	TRID-02-801-019	2020-09-18	12:30	2020-09-18T12:30	AST	Aspartate Aminotransferase	Y	13	13	0	0	
72	TRID-02	801-019	TRID-02-801-019	2020-09-18	12:35	2020-09-18T12:35	BARB	Barbiturates						
73	TRID-02	801-019	TRID-02-801-019	2020-10-01	14:18	2020-10-01T14:18	BARB	Barbiturates						
74	TRID-02	801-019	TRID-02-801-019	2020-09-18	12:30	2020-09-18T12:30	BASO	Basophils	Y	0	0	0		
75	TRID-02	801-019	TRID-02-801-019	2020-09-18	12:30	2020-09-18T12:30	BASOLE	Basophils/Leukocytes	Y	0	0	0		
76	TRID-02	801-019	TRID-02-801-019	2020-09-18	12:30	2020-09-18T12:30	BICARB	Bicarbonate	Y	26	26	0	0	
77	TRID-02	801-019	TRID-02-801-019	2020-09-18	12:30	2020-09-18T12:30	BILI	Bilirubin	Y	5.13	5.13	0	0	
78	TRID-02	801-019	TRID-02-801-019	2020-09-18	12:35	2020-09-18T12:35	BILI	Bilirubin		5.13				
79	TRID-02	801-019	TRID-02-801-019	2020-09-18	12:35	2020-09-18T12:35	BNZDZPN	Benzodiazepine						

Figure 5: Output of ADLB from the adlb() function

5. ADQS FUNCTION

The `adqs()` function creates the ADQS dataset with the standard variables and predecessor variables from ADSL. Here we develop a list of standard variables associated with the ADQS dataset.

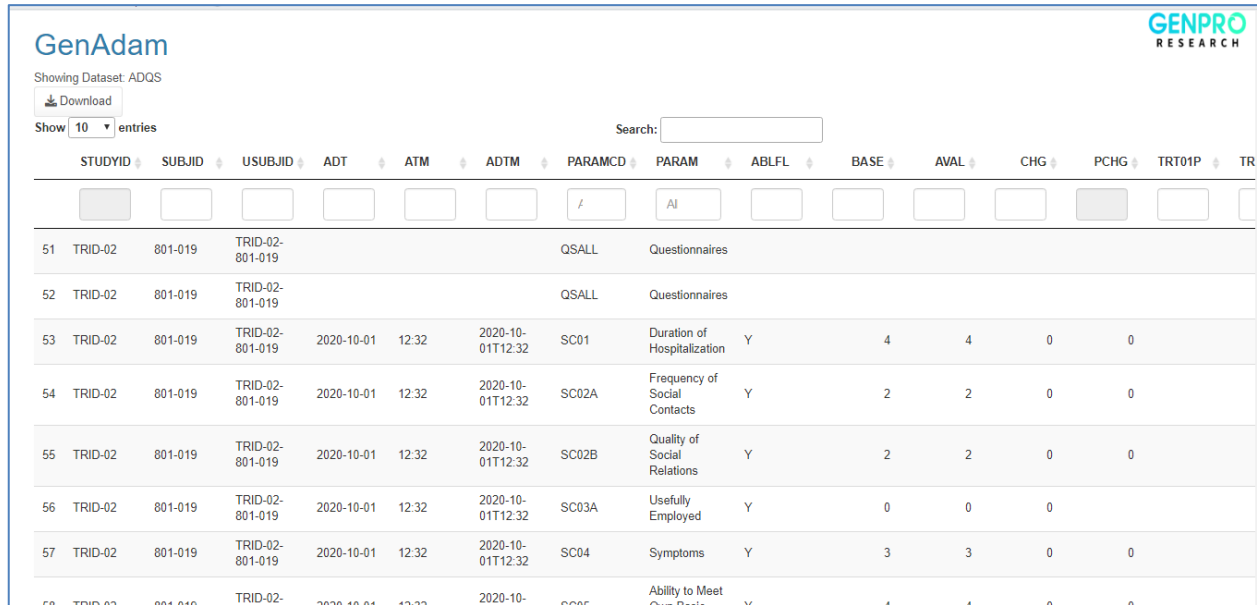
The structure of the `adqs()` function is:

```
adqs(folder_path, adqs_var, metadata)
```

Here the function takes three input arguments;

`folder_path` : Folder path of input datasets (Required)
`adqs_var` : list of variables to be output (Optional)
`metadata` : Metadata/ Codelist filename (Optional)

Example of the ADQS dataset visualization using the `adqs(<SDTM_folder_path>)` is shown in Figure 6.



	STUDYID	SUBJID	USUBJID	ADT	ATM	ADTM	PARAMCD	PARAM	ABLFL	BASE	AVAL	CHG	PCHG	TRT01P	TR
51	TRID-02	801-019	TRID-02-801-019				QSALL	Questionnaires							
52	TRID-02	801-019	TRID-02-801-019				QSALL	Questionnaires							
53	TRID-02	801-019	TRID-02-801-019	2020-10-01	12:32	2020-10-01T12:32	SC01	Duration of Hospitalization	Y	4	4	0	0		
54	TRID-02	801-019	TRID-02-801-019	2020-10-01	12:32	2020-10-01T12:32	SC02A	Frequency of Social Contacts	Y	2	2	0	0		
55	TRID-02	801-019	TRID-02-801-019	2020-10-01	12:32	2020-10-01T12:32	SC02B	Quality of Social Relations	Y	2	2	0	0		
56	TRID-02	801-019	TRID-02-801-019	2020-10-01	12:32	2020-10-01T12:32	SC03A	Usefully Employed	Y	0	0	0	0		
57	TRID-02	801-019	TRID-02-801-019	2020-10-01	12:32	2020-10-01T12:32	SC04	Symptoms	Y	3	3	0	0		
58	TRID-02	801-019	TRID-02-801-019	2020-10-01	12:32	2020-10-01T12:32	SC05	Ability to Meet One's Needs	Y	4	4	0	0		

Figure 6: Output of ADQS from the `adqs()` function

All these functions explained above are used to create the ADSL and BDS domains in a naïve function call. We can update this package with more features, which includes handling the study-specific variables, which will be derived using the specification document and code list.

CONCLUSION

In this paper, several ADaM datasets were generated using the R package *GenAdam*, which presents the ADaM datasets in a shiny app and allows users to filter, sort, search, and download in `.sas7bdat` format. We demonstrate that it is possible to construct clinical trial datasets using R as a suitable substitute. We provided a detailed procedure for configuring the R environment and R code for reading the input dataset and processing the data to generate the ADSL and BDS datasets. Additionally, this package permits us to pass the metadata file and generate variables.

FUTURE WORK

In this paper, we have discussed deriving the standard ADaM variables using the R package *GenAdam*. But it is not sufficient only creating the standard variables, which are easy to map or derive. So as a second phase, we are working on calculating all the variables based on the spec, which is written in a standard template. In the second phase, we will be able to create the complete ADaM package by passing the input SDTM datasets, spec, and metadata.

REFERENCE

1. Minjoe, S. (2017, May). Preparing Analysis Data Model (ADaM) Data Sets and Related Files for FDA Submission. In *PharmaSUG 2017 Conference Proceedings*, Baltimore, Maryland (pp. 1-12).
2. CDISC (<https://www.cdisc.org/standards/foundational/adam>)
3. Sievert, C. (2020). *Interactive web-based data visualization with R, plotly, and shiny*. CRC Press.
4. Wickham H, Miller E, Smith D (2022). *haven: Import and Export 'SPSS', 'Stata' and 'SAS' Files*. <https://haven.tidyverse.org>, <https://github.com/tidyverse/haven>

5. Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R, Golemund G, Hayes A, Henry L, Hester J, Kuhn M, Pedersen TL, Miller E, Bache SM, Müller K, Ooms J, Robinson D, Seidel DP, Spinu V, Takahashi K, Vaughan D, Wilke C, Woo K, Yutani H (2019). "Welcome to the tidyverse." Journal of Open Source Software, 4(43), 1686. doi:10.21105/joss.01686.
6. Wickham H (2016). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York. ISBN 978-3-319-24277-4, <https://ggplot2.tidyverse.org>.
7. Chang W, Cheng J, Allaire J, Sievert C, Schloerke B, Xie Y, Allen J, McPherson J, Dipert A, Borges B (2022). shiny: Web Application Framework for R. R package version 1.7.2.9000, <https://shiny.rstudio.com/>.

ACKNOWLEDGMENTS

We would like to thank the management teams from Genpro Research, who supported this research. This paper and the research behind it would not have been possible without the exceptional support of our managers Roshan Stanly (Manager - Clinical Data Analytics), Ajith B Nair (Senior Manager - Clinical Data Standards), Limna Salim (India Head - Biometrics & Operations), Anoop P Ambika(CTO), and Sachin Marulkar (President & CEO). We would also like to acknowledge our colleagues Anoop Jose and Pranav M Sreejith, who helped us write and review this paper.

CONTACT INFORMATION

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

Author Name : Mahesh Divakaran
 Company : Genpro Research
 Address : Second Floor, Nila Building
 Technopark Campuss
 City / Postcode : Thiruvananthapuram , Kerala – 695581
 Work Phone : +91.4714.026.700
 Email : mahesh.divakaran@genproresearch.com
 Web : www.genproresearch.com

Co-Author Name : Mohin K Mohanan
 Company : Genpro Research
 Address : Second Floor, Nila Building
 Technopark Campus
 City / Postcode : Thiruvananthapuram , Kerala – 695581
 Work Phone : +91.4714.026.700
 Email : mohin.mohanan@genproresearch.com
 Web : www.genproresearch.com

Co-Author Name : Denny Sebastian
 Company : Genpro Research
 Address : Second Floor, Nila Building
 Technopark Campus
 City / Postcode : Thiruvananthapuram , Kerala – 695581
 Work Phone : +91.4714.026.700
 Email : denny.sebastian@genproresearch.com
 Web : www.genproresearch.com

Brand and product names are trademarks of their respective companies.