PharmaSUG China 2026 - Paper AP-134

R in Study Validation

Liang Han, INNOVENT

ABSTRACT

In recent years, the pharmaceutical industry has increasingly begun exploring the use of R to replicate and perform tasks traditionally handled by SAS, driven by the rapid growth of the open-source ecosystem and the demand for analytical flexibility. While R offers great flexibility, it brings two main challenges: maintaining strict regulatory compliance in clinical studies and overcoming the steep learning curve for SAS programmers. To solve these issues. This presentation introduces a hybrid programming strategy that integrates SAS and R to leverage the strengths of both languages: SAS for core study execution to guarantee data accuracy and regulatory submission readiness, and R for quality control (QC), exploratory analysis, and enhanced workflow flexibility. This hybrid approach balances the regulatory robustness of SAS with the flexibility and analytical power of R, providing a practical, scalable solution for clinical study validation that enhances efficiency while maintaining compliance.Beyond proposing this integration strategy, a core objective of this work is to help SAS programmers overcome the learning curve of R, enabling them to quickly master its application and seamlessly apply R within real-world clinical projects. To facilitate the transition for SAS programmers, a structured training framework is proposed, encompassing knowledge acquisition, team upskilling, real-world project practice, internal knowledge sharing, and targeted guidance for SAS-to-R adoption. We developed an LLM-based application that automatically converts SAS code into R code. This tool allows programmers to instantly see R equivalents of their SAS logic, helping them learn R through practical use in real projects.

INTRODUCTION

Aimed at supporting SAS programmers in adopting R for clinical programming, this paper details the core components and practical implementation of R-based study validation, covering reproducibility, package management, environment setup, dynamic workflow control, QC workflows, and targeted training frameworks. The core components of R validation are structured around five key pillars: reproducibility environment management, package version control, input/output (I/O) and script management, and tailored training. For reproducibility and package version control, the renv package is utilized to create isolated, reproducible R environments, with snapshot dates specified for CRAN repositories to lock package versions and eliminate dependency conflicts—critical for regulatory-compliant clinical programming. For I/O and script management, the envsetup package is employed to manage project file system locations via YAML configuration files, enabling dynamic path management and automatic sourcing of R scripts, which streamlines workflow standardization and reduces manual errors. The practical programming approach follows a step-by-step workflow: specifying CRAN repositories and installing open-source clinical programming packages (e.g., haven, dplyr, hms), defining the working directory, setting up the R environment with renv, implementing dynamic path management and automatic script sourcing with envsetup, and executing standardized programs (e.g., SDTM template workflows). For QC, lightweight R tools are highlighted: diffdf as an efficient alternative to SAS's PROC COMPARE for dataset comparison, logr for structured log recording, and functionality to save comparison differences—enabling consistent, traceable QC processes aligned with regulatory requirements.

Additionally, we have implemented targeted training initiatives to facilitate a smooth transition, equipping SAS programmers with the practical knowledge needed to quickly and confidently apply R in their daily clinical programming tasks.

PRACTICAL IMPLEMENTATION OF R VALIDATION

Figure 1 show a structured process for clinical data quality control with R.

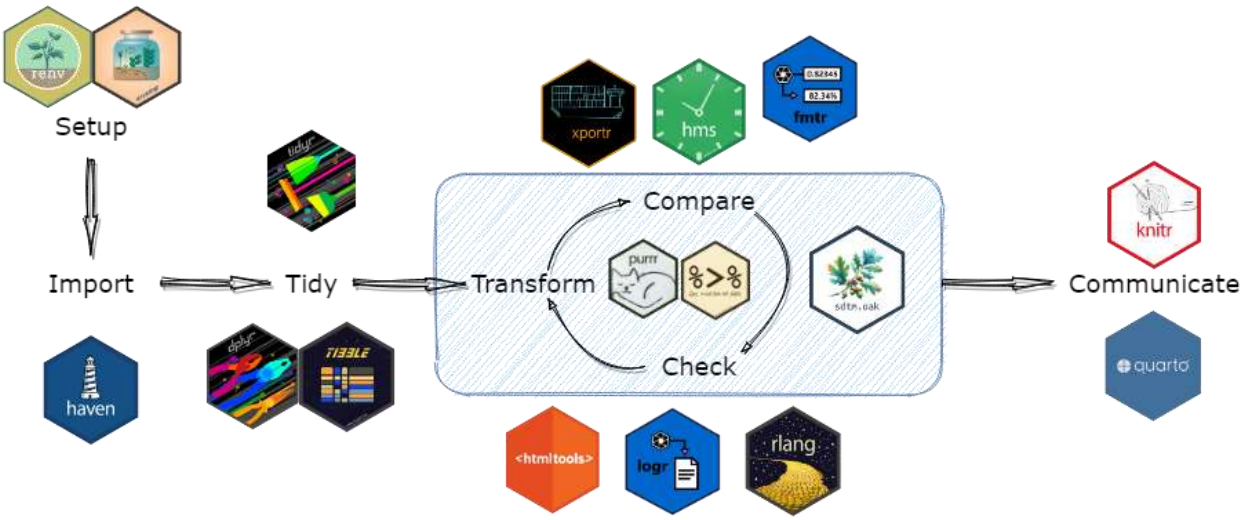

Figure 1. R Validation Workflow

The diagram illustrates the R validation workflow, a structured process designed to support SAS programmers in transitioning to R for clinical data quality control (QC). It begins with Setup (using renv for reproducible environment management), followed by Import (via haven for data ingestion), Tidy (with tidyr for data structuring), Transform (leveraging dplyr and hms for data manipulation), and Compare/Check (utilizing diffdf and logr for dataset comparison and logging). The workflow concludes with Communicate (through knitr and quarto for reporting), balancing regulatory compliance with analytical flexibility to streamline QC tasks in clinical studies.

PROGRAMMING STRATEGY

Hybrid approach. First line in SAS, QC in R. SAS ensure accuracy and submission R provide further flexibility. Help SAS programmers pick up R.

R VALIDATION: CORE COMPONENTS

- Package version control: renv lock package version.
- Reproducibility environment: renv.
- I/O & Scripts manage: envsetup manage path and source script.

Renv, a powerful R package designed to help you create reproducible environments for your R projects. Use renv to make your R projects more isolated, portable and reproducible. The envsetup package helps you manage R project environments by providing a flexible configuration system that adapts to different deployment stages (development, testing, production) without requiring code changes.

PROGRAMMING APPROACH

- Specify R package repository and Install open-source R packages.
- Define Working Directory.
- Setup R environment.
- Dynamic Path Management and Automatic Script Sourcing.
- Execute programs.

Specify R package repository and Install open-source R packages

The date-based snapshot played a vital part in the submission because, in the world of clinical trials, reproducibility and transparency are paramount. Researchers and regulators must ensure they can

precisely recreate the computational environments used to analyze trial data. Repository snapshots ensure this reproducibility and provide complete transparency. By including a frozen URL(https://packagemanager.posit.co/) that references a date-based snapshot, researchers make it transparent which software was used, while guaranteeing that regulators can access the exact package versions used in their analyses to replicate their work, eliminating concerns about compatibility issues or unexpected changes in package behavior.

```
options(repos = "https://packagemanager.posit.co/cran/2025-12-17")
renv::install(c('hms', 'haven', 'dplyr'))
renv::snapshot(type="all")
```

This is useful because you can then share the lockfile and other people or other computers can easily reproduce your current environment by running restore(), which uses the metadata from the lockfile to install exactly the same version of every package.

Define Working Directory

Under the File menu, click on New Project, choose Existing Directory, then choose your study path as the location, click on Create Project.

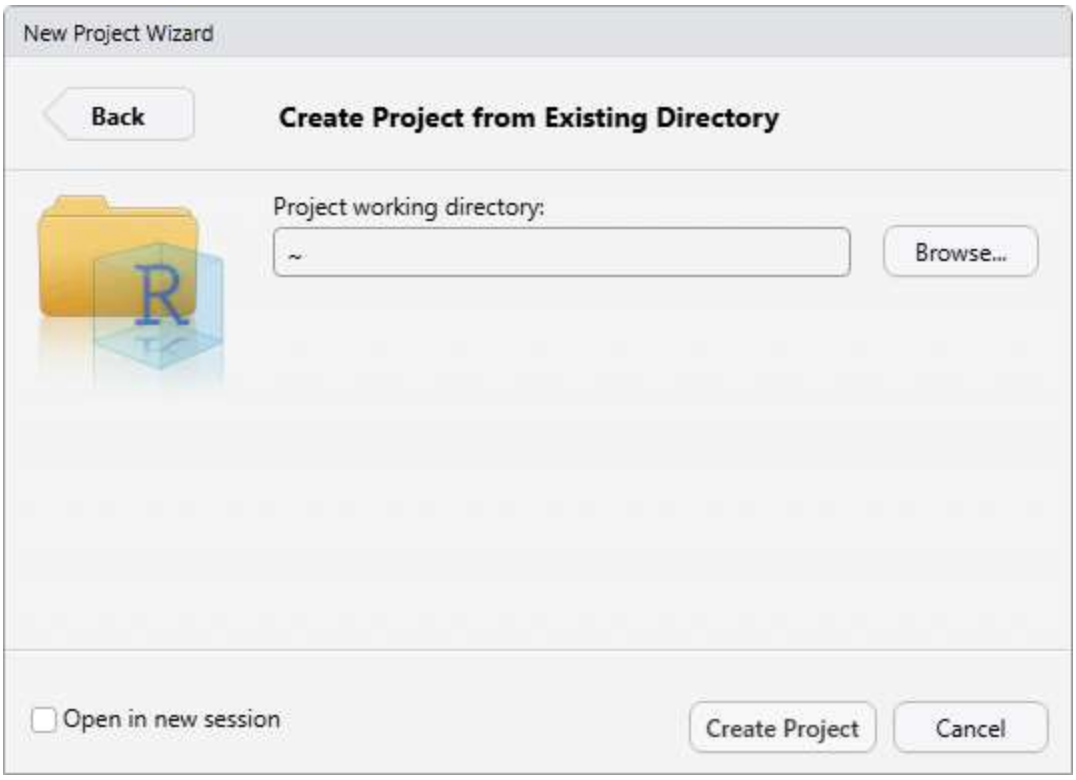

Figure 2 Creating Project from Existing Directory

Initializing project

renv::init() set up project infrastructure including the project library and the .Rprofile that ensures renv will be used in all future sessions.

renv::restore() restore the project's dependencies from the lockfile(renv.lock), as previously generated by renv:snapshot().

renv::status() report inconsistencies between lockfile, library, and dependencies.

```
> renv::init()
The following package(s) will be updated in the lockfile:

# CRAN ------------------------------------------------------------------------
- renv   [* -> 1.1.7]

The version of R recorded in the lockfile will be updated:
- R      [* -> 4.5.3]

- Lockfile written to "[redacted]/pharma SUG2026/renv.lock".
```

Figure 3 Initializing renv in an existing project

```
> renv::restore()
The following package(s) will be updated:

# CRAN ------------------------------------------------------------------------
- boot          [1.3-32 -> 1.3-31]
- cluster       [2.1.8.2 -> 2.1.8.1]
- foreign       [0.8-91 -> 0.8-90]
- lattice       [0.22-9 -> 0.22-6]
- Matrix        [1.7-4 -> 1.7-3]
- mgcv          [1.9-4 -> 1.9-1]
- renv          [1.1.7 -> 1.1.5]
- survival      [3.8-6 -> 3.8-3]
- conflicted    [* -> 1.2.0]

# RSPM ------------------------------------------------------------------------
- admiraldev    [* -> 1.3.1]
- askpass       [* -> 1.2.1]
- assertthat    [* -> 0.2.1]
```

Figure 4 Restore project library from a lockfile

Verify the environment. Run renv::status() to confirm all packages match the lock file.

Dynamic Path Management and Automatic Script Sourcing

The envsetup package uses a YAML configuration file to manage project file system locations and automatically sources R scripts from specified directories. Instead of hardcoding paths or manually changing configurations, envsetup uses YAML configuration files to manage these differences automatically.

envsetup.yml*

```yaml
default:
  project_path:
    !expr Sys.setenv(ENVSETUP_PROJECT_PATH = getwd()); getwd()
  paths:
    pdev_code_library: !expr file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'MACRO')
    .rdbs: !expr file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'RDBS', 'DATA')
    .adam: !expr  file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'ADAM', 'DATA')
    .sdtm: !expr  file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'SDTM', 'DATA')
    .expl: !expr  file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'EXPL', 'DATA')
    .qsdtm: !expr file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'QC', 'SDTM', 'DATA')
    .qadam: !expr file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'QC', 'ADAM', 'DATA')
    .qrept: !expr  file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'QC', 'REPORT', 'DATA')
    .qr: !expr file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'QC', 'R', 'DATA')
    .rept: !expr file.path(Sys.getenv('ENVSETUP_PROJECT_PATH'), 'REPORT', 'DATA')
```

Figure 5 YAML configuration file example

The YAML configuration file helps to manages file system locations (data, output, programs).

```
Sourcing file:  '[redacted]/generate_adam_template.R'

 The following objects are added to .GlobalEnv:

    'generate_adam_template'

Sourcing file:  '[redacted]load_data.R'

 The following objects are added to .GlobalEnv:

    'load_sas_from_path'
```

Figure 6 Automatic R Script Sourcing

Execute programs

With the environment restored, your analysis code should run exactly as it did for the creator.

```
# ---------------------------[ START PGM HEADER ]----------------------------------------
# *                    Property of Innovent Biologics Co. Ltd.
# *
# * Program Name             : v_ae.R
# * Program Purpose          : validate SDTM domain
# * Compound/Study/Analysis  : XXX/XXXXXXX/PK
# * Program Author           : XXXXXX
# * Date Completed           : 2026-05-20
# * Input Datasets           :
# * Output Datasets          :
# * System                   : R version 4.5.3 Platform: x86_64-w64-mingw32/x64 (64-bit)
# * Revision History
# * ------------------------------------------------------------------------------------
# * Revision Date            :
# * Revised by               :
# * Revision Description     :
# * ---------------------------[ STOP PGM HEADER ]----------------------------------------

#clean environment
xd_start()

# 初始化参数
pgmname <- "v_ae"
domain <- "ae"


#log
options("logr.on" = TRUE)
log_file = log_open(file.path(.file_path$qsdtml, paste0(pgmname,".log"))
                    ,logdir = FALSE)
log_code()

#load SAS dataset
# load_sas_from_path(lib_path=.lib_path[c('rdbs','sdtm','qsdtm','metct','metsdtm')])
# View(.sas_library)
# * ------------------------------------------------------------------------------------


xd_def2R(type = "SDTM",domain=domain)
```

Figure 7 SDTM Qc Program

TRAINING

Successful training strategies combined knowledge acquisition with practical application. It was essential to create opportunities for learners to immediately apply what they've learned in real projects or meaningful tasks. Committing to using new skills regularly was emphasized as critical to lasting skill development despite busy workloads. To accelerate the adoption of R among SAS programmers, we implemented a comprehensive upskilling strategy for our team through three key initiatives:

- Best Practices: Established R-specific Good Programming Practice (GPP) guidelines, modeled after our company's existing SAS GPP SOPs.
- Training: Conducted specialized training programs, including the study of Pharmaverse packages.
- Tooling: Developed an LLM-powered SAS-to-R conversion application that enables programmers to translate SAS code into R code compliant with clinical trial standards.

GPP

Good programming practices in R.

1 编程指南 Programming Guidelines

1.1 程序结构 Program Structure

每个 R 程序文件应包含以下结构（按顺序排列）：

- **程序头注释**（Program Header）
- **包加载**（`library()` 调用）
- **全局选项设置**（Global Options）
- **路径/常量定义**（Paths / Constants）
- **自定义函数定义**（User-Defined Functions）
- **程序主体**（Main Body）

Figure 8 GPP

SAS TO R CONVERT APP

Figure 9 SAS2R APP

UPSKILLING OF TEAM

Community & Knowledge Sharing: Regularly organize knowledge-sharing seminars to disseminate R-related knowledge and trends within the pharmaceutical industry.

CONCLUSION

The adoption of R for clinical programming requires a dual focus on robust technical validation and strategic personnel enablement. By establishing a standardized, reproducible environment through tools like renv and envsetup, alongside streamlined QC workflows utilizing diffdf and logr, we have demonstrated a practical framework for R-based study validation. Equally important, our comprehensive upskilling strategy—anchored in R-specific GPP guidelines, targeted Pharmaverse training, and innovative LLM-powered code conversion tools—has effectively bridged the gap for SAS programmers, accelerating their transition to R. Ultimately, this holistic approach not only ensures strict adherence to regulatory requirements and data traceability but also empowers clinical programming teams to confidently leverage the power of open-source R, maintaining the highest standards of operational efficiency and data quality..

REFERENCES

Ushey K, Wickham H (2026). “renv” Accessed Jun 7, 2026. https://rstudio.github.io/renv/.

Masel N, Stackhouse M, Ceney A (2025. “envsetup” Accessed Jun 7, 2026. https://github.com/pharmaverse/envsetup.

ACKNOWLEDGMENTS

The author thanks his manager, Shiyao Liu, for valuable feedback and continued support.

RECOMMENDED READING

- *R for data science*
- *Pharmaverse*
- *Johnson & Johnson’s Open Source Journey in R*
- *Johnson & Johnson’s Success Story: A Hybrid R/SAS Strategy for FDA Submission*

CONTACT INFORMATION

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

Liang Han
INNOVENT
Liangh.han@innoventbio.com