Paper OS04

Introduction of PharmaForest – Forest of SAS packages

Ryo Nakaya, Takeda, Osaka, Japan,
Hiroki Yamanobe, Maruho, Osaka, Japan,
Yutaka Morioka, EPS Corporation, Osaka, Japan

ABSTRACT

PharmaForest is an open-source repository of SAS packages, which is based on SAS Packages Framework (SPF), that brings together a diverse ecosystem of SAS packages designed to accelerate and standardize programming using SAS. Built to support workflows in clinical development, for instance, CDISC data creation and Reporting TFLs for Clinical Study Report, PharmaForest provides a centralized archive where developers and users can easily access, share, and validate reusable macros and utilities for pharmaceutical clinical development. This initiative not only reduces duplication of effort but also fosters transparency, reproducibility, and innovation across the pharmaceutical programming community. By integrating best practices from individuals and companies, PharmaForest aims to become a trusted hub for collaborative development and aims to establish more vibrant communities.

INTRODUCTION

The open-source package ecosystems in R and Python have significantly advanced scientific computing, fostering innovation through modular, reusable, and community-driven development. On the other hand, Bartosz Jablonski introduced **SAS Packages Framework (SPF)** in 2019[1] while there had been no concept of packages in SAS for a long time. SAS packages framework (SPF) is believed to mark a pivotal step toward establishing a similar culture of sharing and standardized extensions within the SAS environment. Although its recognition remains gradual, the framework has begun to attract attention among developers seeking reproducibility and portability especially in the pharmaceutical industry since open-source software utilization is increasingly expanding recently in this area so that companies have more collaboration to reduce redundancy in developing each own programming codes and tools.

Within this evolving landscape, **PharmaForest** emerged as a small yet enthusiastic collective founded by three members in the PHUSE Japan Open-Source Technologies (OST) working group. The small guild humorously describes itself as a “minor league challenger” with the playful slogan “Challenge Pharmaverse!”—a nod to the influential R-based open-source ecosystem[2] that has set the benchmark for collaborative clinical-data programming and analysis.

While the tone is light-hearted, the group’s ambition is genuine: to explore whether a SAS-centered open-source community can achieve the same level of transparency, accessibility, and technical sophistication as its R counterpart. Whether PharmaForest will one day catch up with its senior peer, Pharmaverse, remains an open question—but the journey itself has already started to contribute meaningfully to the maturation of open-source culture in pharmaceutical programming not only R but also including SAS, which SAS itself is a proprietary software of SAS Institute[3].

SAS PACKAGES FRAMEWORK (SPF)

SAS Packages Framework (SPF) is a set of core macros developed and maintained by Bartosz Jablonski in MIT license that provides a new framework called **“SAS Packages”** for creating and using modular SAS components with simple way like those package framework in R or Python. Since its release in 2019, there have been 11 core macros[4] as of 2025 December listed below while bolded macros are the ones frequently called in generating and using SAS packages.

- %extendpackagesfileref
- **%generatepackage**
- %helppackage
- **%installpackage**
- %ispackagesfilerefok
- %listpackages
- **%loadpackage** (and %loadpackageS)
- %loadpackageaddcnt
- %previewpackage
- %relocatepackage
- %saspackagesframeworknotes
- %splitcodeforpackage
- %unloadpackage
- %verifypackage

It is extremely easy to enable SPF—once SPFinit.sas in the GitHub repository of SPF being installed, simply execute the two lines below at the beginning of a session to enable SPF.

```
filename packages "\path\to\your\packages";
%include packages(SPFinit.sas);
```

Once SPF has been enabled above, the process for installing packages from GitHub or online directories and loading the packages is similar to the one in R or Python, which you are familiar with. In R, `install.packages("packagename")` and `library(packagename)` are used, `pip install packagename` and `import packagename` are used in Python. In SAS (means SPF here), you can install and load packages as shown below:

```
%installPackage(packagename)
%loadPackage(packagename)
```

SAS packages bundle macros, formats, functions, data, and even documentation into one .zip file. Loading package enables users to use these bundled contents in the package as well as support users to see help information attached to each content within package. To see help information of macros in the package, you can do like this:

```
%helpPackage(packagename, macroname)
```

For developers, SPF provides package generating functionality. Developers can call `%generatePackage` to generate package zip file from source package files. The creation of a SAS package, including detailed source folder structures and required files, is described in Jablonski (2021) [5].

```
%generatePackage(
       filesLocation = \path\to\your\source\package,
       markdownDoc   = 1,  /* 1: Generate package documentation(.md) */
       easyArch      = 1   /* 1: Create a ZIP archive ([Package]_[version].zip) */
       testPackage   = Y,  /* Y to test package */
       testResults   = \path\to\your\folder,   /* location to save test results */
       workInTestResults = 1 /* 1: Save work library results in test sessions */
)
```

WHAT IS PHARMAFOREST?

R has CRAN and Pharmaverse as integrated repositories of R packages and these initiatives greatly encourage R users to share their knowledge and exchange their programming skills. Furthermore, Pharmaverse is expanding its ecosystem with more than forty packages and four hundred collaborators as of December 27 in 2025 to enhance more utilities for pharmaceutical clinical development.

Building upon this vibrant spirit of collaboration, **PharmaForest** originated from the **PHUSE Japan OST Working Group** after roughly one year of internal investigation by individuals. SAS Packages sub-working group in the PHUSE Japan OST Working Group was formed in April 2025 as a new wave of open-source code initiative using SAS, and PharmaForest was born in GitHub repository on **30 June 2025**, which seeks to foster a similarly dynamic and vibrant community centered around **SAS packages**.

The goal of the initiative is very clear.

- To **actively encourage sharing** of SAS know-how that has often stayed within individuals.
- To build up collective knowledge, boost productivity, ensure quality through standardization, and **energize our community**.

Our priority is based on the above-mentioned goals.

- Our priority is to share openly—and get others to share as well—so that more people can join in.
- On that basis, we will work on improving quality, driving standardization, and creating long-term value.

Technically we already have had a vibrant community of SAS for a long time such as SAS support communities[6] or forums globally held across regions where people can share their knowledge and ask whatever they want to know about. In the meantime, **a standardized format and unified style of exchanging SAS codes** were missing while R or Python have had long time as a shape of “package”. The afore-mentioned SAS Packages Framework (SPF) revolutionized this missing piece in SAS world, and the big trend of introducing open-source utilization nowadays represented by R in the pharmaceutical industry greatly motivated us to create open-source SAS packages repository for the pharmaceutical industry to make our existing community even more vibrant. We have SAS Packages Archive

(SASPAC) as a centralized repository of packages for comprehensive scope not limited to pharmaceutical run by Bartosz Jablonski[7].

Before moving on, let us make a brief digression to share a behind-the-scenes story, as this marks our first appearance on the global stage. Logo of PharmaForest that was adopted symbolizes our aspiration to plant individual “trees” (packages) and let them grow together into a vast forest in the future. It was created with the image of three squirrels nurturing this vision. The mountain, forest, and house motifs incorporated in the design also derive from the family names of the founding members.

The foundational platform of **PharmaForest** is hosted on **GitHub**, where all packages and related resources are maintained.
https://github.com/PharmaForest

In parallel, we also developed a **gallery-style webpage** that allows visitors to explore PharmaForest in a more **visual and interactive** manner. The gallery-style webpage can be also accessed by mobile phones reading the QR code.
https://pharmaforest.github.io/

PACKAGES IN PHARMAFOREST

Figure 1: PharmaForest Logo Map

There are 43 packages in PharmaForest as of December 27, 2025. Rapid growth of the number of packages in these six months is mainly due to our strategy to firstly increase the number of packages, which aims to catch people’s attention to SAS packages and the repository. Recent advancement in generative AI tools such as ChatGPT enabled us to create nice hex logos easily.

These packages have a variety of functionalities including plotting technical graphs, data operation functions, and checkers as well as packages with other features. Users can quickly see overview of packages and its categories (Output/Visualization, Data Utility, Checker, Other) in the gallery page(https://pharmaforest.github.io/).

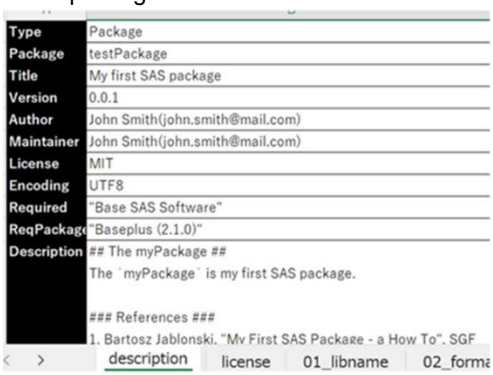
There are two types of packages in PharmaForest in terms of origin. One is PharmaForest original packages which are created and maintained with collaborative effort among members, the other one is mirroring external GitHub repositories which have SAS packages created and maintained by individuals.

How to install packages from PharmaForest is simple. You can utilize `%installPackage()` after enabling SPF. The block of code below downloads *OncoPlotter* package (.zip) located in GitHub repository in PharmaForest, one of PharmaForest original packages, to your location. Users need to load the installed packages using `%loadPackage()` to use functionality in the packages.

```
%installPackage(oncoplotter, mirror=pharmaforest)
```

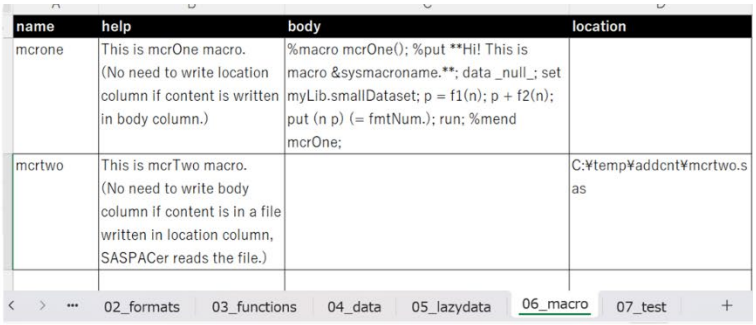
サスパッカー (SASPACER)

SASPACer is a package to help users create SAS packages. Since there is specific format of folder structure and file requirements for source packages, which will be used to build zip package file, *SASPACer* offers an alternative way of constructing these source folder and files. Users can find an excel spread sheet in the repository of *SASPACer*(https://github.com/PharmaForest/SASPACer/tree/main/SASPACer/addcnt) and just fill up package information including macros or functions which users would like to bundle in the package.



Type	Package
Package	testPackage
Title	My first SAS package
Version	0.0.1
Author	John Smith(john.smith@mail.com)
Maintainer	John Smith(john.smith@mail.com)
License	MIT
Encoding	UTF8
Required	"Base SAS Software"
ReqPackag	"Baseplus (2.1.0)"
Description	## The myPackage ##
	The `myPackage` is my first SAS package.
	### References ###
	1. Bartosz Jablonski. "My First SAS Package - a How To". SGF

description | license | 01_libname | 02_forma



name	help	body	location
mcrone	This is mcrOne macro. (No need to write location column if content is written in body column.)	%macro mcrOne(); %put **Hi! This is macro &sysmacroname.**; data _null_; set myLib.smallDataset; p = f1(n); p + f2(n); put (n p) (= fmtNum.); run; %mend mcrOne;	
mcrtwo	This is mcrTwo macro. (No need to write body column if content is in a file written in location column, SASPACer reads the file.)		C:¥temp¥addcnt¥mcrtwo.sas

02_formats | 03_functions | 04_data | 05_lazydata | 06_macro | 07_test

Figure 2: Package information

```
%installPackage(SASPACer, mirror=pharmaforest)
%loadPackage(SASPACer)

%ex2pac(
      excel_file       = \path\to\excel\excelfile.xlsx, /* Path of input excel file */
      package_location = \path\to\output,               /* Output path */
      complete_generation = Y                           /* Y: Run %generatePackage() */
)
```

With the above sample code, users can create source folder structure and required files to `package_location` reading `excel_file`. `Complete_generation=Y` will run `%generatePackage()` to build zip file using the source package folders and files created.
https://github.com/PharmaForest/SASPACer

ONCOPLOTTER

OncoPlotter is a SAS package to **create figures commonly created in oncology studies**. The package prepares Kaplan-Meier plot (`%kaplan_meier_plot`), swimmer plot (`%swimmer_plot`), and waterfall plot (`%waterfall_plot`) as of December 27, 2025. These figures are often created in clinical study reports or publications, but detailed customization are not covered well in normal settings of SAS procedures. *OncoPlotter* standardized these figures with customizability as well as a feature to output SAS codes generated in the macro can let users have further development if needed. In addition, the package generates dummy datasets of ADSL, ADRS, ADTR, and ADTTE to test three of macros.

```
%installPackage(OncoPlotter, mirror=pharmaforest)
%loadPackage(OncoPlotter)
```

https://github.com/PharmaForest/OncoPlotter

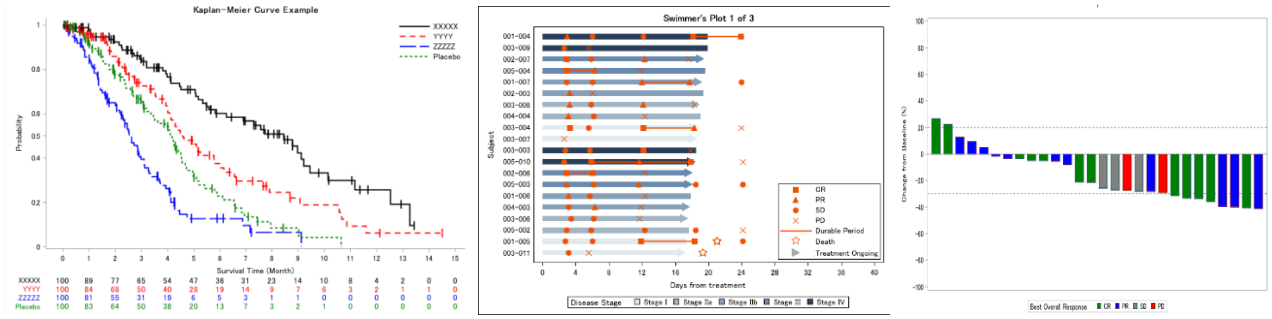


Figure 3: Kaplan-Meier Plot, Swimmer's Plot, Waterfall Plot

ADAMSKI

Adamski is a SAS package **inspired by the R package *admiral***. It aims to bring the same flexible and modular CDISC ADaM derivation framework to the SAS environment. The package follows the *admiral* design principles while adapting to SAS syntax and workflows. It enables consistent, reproducible ADaM dataset creation in compliance with CDISC standards. *Adamski* serves as a bridge between open-source R implementations and traditional SAS programming. Below is an example of `%derive_var_merged_exist_flag`, which is mirror of `derive_var_merged_exist_flag` function in *admiral*.

```
%installPackage(adamski, mirror=pharmaforest)
%loadPackage(adamski)

data have;
  do SEX="F","M"; do AGE=12 to 17; output; end; end;
run;

data want;
  set have;
  /* Single key */
  %derive_var_merged_exist_flag(
       dataset_add = SASHELP.CLASS,
       by_vars        = AGE,
       new_var       = AGE_IN_CLASS,
       true_value    = Y,
       false_value   = N
  );

  /* Multiple keys */
  %derive_var_merged_exist_flag(
       dataset_add = SASHELP.CLASS,
       by_vars        = AGE SEX,
       new_var       = AGE_SEX_IN_CLASS
  );

  /* With condition */
  %derive_var_merged_exist_flag(
       dataset_add = SASHELP.CLASS,
       by_vars        = AGE,
       condition      = %nrbquote( SEX = "F"),
       new_var       = AGE_IN_CLASS_F_ONLY,
       true_value    = Yes,
       false_value   = No
  );
run;
```

https://github.com/PharmaForest/adamski

SAS_DATASET_JSON

Sas_dataset_json is a SAS package designed to support bi-directional conversion between CDISC-compliant Dataset-JSON format and SAS datasets.

- Convert **SAS datasets to Dataset-JSON**
- Convert **Dataset-JSON back to SAS datasets**,

It handles version 1.1 of dataset-JSON and Newline Delimited JSON (NDJSON), which are increasingly getting interest as a potential new format of CDISC data submission to regulatory agencies. The package is recognized by CDISC Open-Source Alliance (COSA) as an open-source project focused on implementing or developing CDISC standards to drive innovation in the CDISC community [8].

In combination with **extended attributes** attached to SAS dataset, the package can convert dataset with metadata into Dataset-JSON format.

```
%installPackage(sas_dataset_json, mirror=pharmaforest)
%loadPackage(sas_dataset_json)
```

```
proc datasets nolist;
   modify adsl;
   xattr add ds originator="X corp."
                    fileOID="www.cdisc.org/StudyMSGv2/1/Define-XML_2.1.0/2024-11-11/"
                    studyOID="XX001-001"
                    metaDataVersionOID="MDV.MSGv2.0.SDTMIG.3.4.SDTM.2.0"
                    sourceSystem_name="SASxxxx"
                    sourceSystem_version="9.4xxxx"
   ;
   xattr add var
                    STUDYID (label="Study Identifier"
                                dataType="string"
                                length=8
                                keySequence=1)
                    USUBJID (label="Unique Subject Identifier"
                                dataType="string"
                                length=7
                                keySequence=2)
                    RFSTDTC (label="Subject Reference Start Date/Time"
                                dataType="date")
                    AGE (label="Age"
                           dataType="integer"
                           length=2)
                    TRTSDT (label="Date of First Exposure to Treatment"
                           dataType="date"
                           targetDataType="integer"
                           displayFormat="E8601DA.") ;
   ;
run;
quit;
%m_sas_to_json1_1(
             outpath = /project/json_out,
             library = WORK,
             dataset = adsl,
             pretty = Y
);
```

https://github.com/PharmaForest/sas_dataset_json

YAML_WRITER

A lightweight SAS macro package to **generate YAML files directly from SAS**. The package leaves you,

- Start and end YAML output streams
- Export datasets into YAML (mapping, sequence, mappingsequence)
- Build nested or inline structures

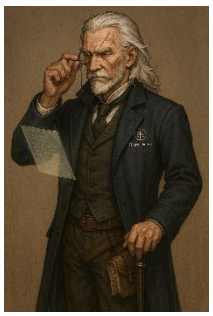
Users can write YAML directly between `%yaml_start` and `%yaml_end`, use `&nw.` for line breaks, and `%dataset_export / %nest / %inline_nest` for structured output.


```
%installPackage(yaml_writer, mirror=pharmaforest)
%loadPackage(yaml_writer)

%yaml_start(outpath=C:/tmp, file=sample);
#This YAML is a sample &nw.;
person: &nw.;
  name: Morio Forest &nw.;
  job: Master Navigator of PharmaForest &nw.;
  age: Unknown &nw.;
  like: hash object &nw.;
%yaml_end();
```

https://github.com/PharmaForest/yaml_writer

FOREST NAVIGATORS



PharmaForest has forest navigators to make users have an additional option to learn about PharmaForest and its packages. The master navigator is *Dr. Forest, Morio*, who is an excellent and diligent GPT assistant who can answer most questions about PharmaForest. He also has first 15 packages(#1-15) information from each README.md. *Dr. Apple* is one of support navigators who is also GPT assistant who has second 15 packages(#16-30) information. Due to the limitation of the number of files customized GPTs can keep, more GPT assistants would be added as we have more packages in PharmaForest.
We also have another GPT assistant to help creating SAS package source folders and files, called "Oba-chan" a SAS package lady. Once you have an idea of package as well as macros or functions to bundle, you can talk to her to get support creating source folder structure and adding draft help information to macros or functions. It should smooth your entry to creation of SAS packages. Please note that they are not speaking on behalf of our organization. You need to sign up to ChatGPT (at least a free user account) to talk to them.

Dr. Forest: https://chatgpt.com/g/g-6881d98193ec8191abb19e4e920cb64c-dr-forest
Dr. Apple: https://chatgpt.com/g/g-68abce4602908191b56d53895bb2e9dc-dr-apple
Oba-chan: https://chatgpt.com/g/g-68be12f679a88191866ef1e9b35be3c4-sas-package-lady-oba-chan

CONCLUSION

The emergence of the SAS Packages Framework (SPF) and initiatives like PharmaForest demonstrates that open-source principles—once thought to be limited to languages like R and Python—can also flourish within proprietary ecosystem of SAS. Although the road toward fully open and mutually complementary environment is still in its early stage, SAS community with open-source minded momentum would grow to the new shape going forward. In order to figure out the direction we would head in this inflection point, more attempts and discussions are necessary. It would be expected that PharmaForest is to be part of this communication with more collaborators.

REFERENCES

[1] Bartosz Jabłonski, "SAS Packages: The Way to Share (a How To)", SAS Global Forum 2020 Proceedings, 4725-2020 https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2020/4725-2020.pdf
[2] Pharmaverse. *Pharmaverse website.* Retrieved from https://pharmaverse.org
[3] SAS Institute Inc. *SAS Institute official website.* Retrieved from https://www.sas.com
[4] Bartosz Jabłonski (2019). *SAS Packages Framework (SPF)*. GitHub. Retrieved from https://github.com/yabwon/SAS_PACKAGES
[5] Bartosz Jabłonski (2021), "My First SAS® Package – A How To", SAS Global Forum 2021 https://github.com/yabwon/SAS_PACKAGES/blob/main/SPF/Documentation/Paper_1079-2021/My%20First%20SAS%20Package%20-%20a%20How%20To.pdf
[6] SAS Institute Inc. *SAS Support Communities.* Retrieved from https://communities.sas.com
[7] SAS Packages Archive. SASPAC GitHub repository. Retrieved from https://github.com/SASPAC
[8] CDISC. (2025). *COSA: CDISC Open-Source Alliance.* Retrieved from https://www.cdisc.org/cosa

CONTACT INFORMATION

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

Ryo Nakaya
Takeda Pharmaceutical Company Limited
Email: ryou.nakaya@takeda.com

Hiroki Yamanobe
Maruho Co., Ltd.
Email: yamanobe_ffa@mii.maruho.co.jp

Yutaka Morioka
EPS Corporation
Email: morioka.yutaka038@eps.co.jp