Paper CT13

Introducing PROC R

Jim Box, SAS Institute Inc.

ABSTRACT

PROC R is a new procedure in SAS® Viya® that allows programmers to freely switch between R and SAS code in the same program. It makes it easy to move data between R dataframes and SAS datasets, produce R graphics in SAS output, utilize SAS commands in the R code, and much more. This new PROC makes it easy to use the right coding language for the specific part of the problem you are trying to solve, all in the SAS environment

INTRODUCTION

When installing SAS Viya, the administrator can use the SAS Configurator for Open Source to install an R environment with a specific list of packages. This environment should be thought of as a Production R environment, meaning that individual users cannot make changes to the list or versions of packages.

PROC R works much like PROC PYTHON in terms of features and syntax. The key capabilities are

- Moving SAS datasets in and out of R data frames
  - **df2ds()** and **ds2df()**
- Moving macro variables and values between SAS and R
  - **symget()** and **symput()**
- Using SAS functions inside the R code block
  - **sasfnc()**
- Executing SAS code from the R block
  - **submit()**
- Having R output display in the SAS Results Window
  - **show**() and **rplot**()

The easiest way to illustrate this is with examples.

EXAMPLE ONE: GET PACAKGE LIST

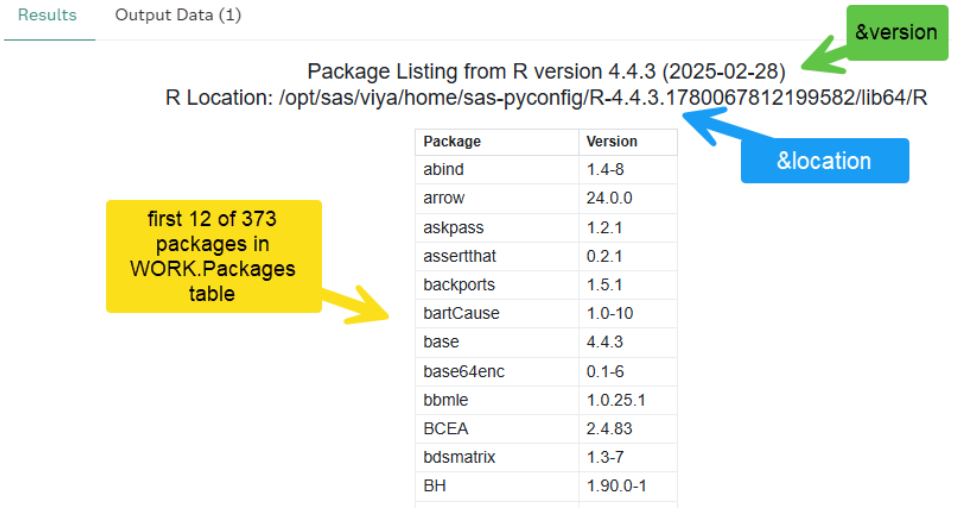
Code

```
1   PROC R;
2   SUBMIT;
3
4   ver = R.version.string
5   location = R.home()
6   pkg = as.data.frame(installed.packages()[,c(1,3)])
7
8   df2sd(pkg, "WORK.Packages")  # Move dataframe to SAS dataset
9   symput("version",ver )
10  symput("location",location )
11
12  ENDSUBMIT;
13  RUN;
14
15  TITLE "Package Listing from &version";
16  TITLE2 "R Location: &location";
17  PROC SQL;
18    Select Package, version
19    from packages
20    order by upcase(Package);
21  quit;
22  TITLE;
23  TITLE2;
24
```

Program 1: Create a dataset with the R Package list and display it in SAS

Code Line	Description
1-2	Open up a PROC R Block
4 -6	Regular R code to get the R version, location, and package list
8	Move the R dataframe called pkg into the SAS dataset WORK.Packages
9	Create a SAS macro variable called &version that has the value of R.version.string
10	Create the SAS macro variable called &location that has the value of R.home
11-12	Close the PROC R block and run the code.
15	Utilize &version macro variable
16	Utilize &location macro variable
19	Utilize the WORK.Packages dataset

Output 1 shows the first part of this SQL report with the values captured in R showing up as standard SAS objects.



Output 1: Results of SQL Report on R Location, version, and Packages

EXAMPLE TWO: CREATE AND DISPLAY DATASETS, EXECUTE SAS CODE, AND USE SAS FUNCTIONS

Code

```
1  %let inputTable = SASHELP.HEART;
2
3  proc R;
4  submit;
5
6  # Set up libraries and read in data
7  library(dplyr)
8
9  df= sd2df(symget("inputTable"))         # Move data to dataframe
10 print(head(df))                         # Print will output to log
11 show(df,paste("Dataset",symget("inputTable")),count=11)  # Show will output to Results window
12
13 dead <- df %>% filter(Status == "Dead")  # Make a new dataframe with a filter
14 df2sd(dead,"Dead")                       # Move dataframe to SAS dataset
15 symput("numDead",nrow(dead) )            # Put the number of rows into a SAS macro variable
16
17 submit("%put &=numDead;")                # Execute SAS Code on exit
18 hello = sasfnc("propcase","HELLo WORld") # Execute a SAS function
19 print(hello)
20
21 endsubmit;
22 RUN;
23
```

Program 2: Data Manipulations

Code Line	Description
1	Define a SAS Macro variable &inputTable and assign to SASHELP.HEART
3-4	Open the PROC R session
7	Initialize the dplyr package
9	Move the SASHELP.HEART dataset into the R dataframe df by resolving the &inputTable macro set in line 1
10	Print the head of the dataframe. When using the print function, the results are sent to the SAS Log
11	Show the first 11 records of the df, using the value of the &inputTable concatenated with the word "Dataset" as the title. When using show(), the output will be sent to the Results tab
13	Create a new dataframe called dead that uses dplyr functions to filter the df to just include records with Status = "Dead"
14	Create a new SAS dataset called Dead from this new dataframe. Note that not providing a libname reference means the dataset will be written to the WORK library
15	Make a new macro variable called &numDead that resolves to the count of the number of records in the dead dataframe
17	Specify SAS code to be run on exiting the R session
18 - 19	Use the SAS function propcase with the argument "HELLo WORld" and assign it to the R variable hello, then print that to the log
21-22	End the R session

```
  Status DeathCause AgeCHDdiag    Sex AgeAtStart Height Weight Diastolic
1   Dead      Other         NA Female          29  62.50    140        78
2   Dead     Cancer         NA Female          41  59.75    194        92
3  Alive                    NA Female          57  62.25    132        90
4  Alive                    NA Female          39  65.75    158        80
5  Alive                    NA   Male          42  66.00    156        76
6  Alive                    NA Female          58  61.75    131        92
  Systolic MRW Smoking AgeAtDeath Cholesterol Chol_Status BP_Status
1      124 121       0         55          NA                Normal
2      144 183       0         57         181   Desirable      High
3      170 114      10         NA         250        High      High
4      128 123       0         NA         242        High    Normal
5      110 116      20         NA         281        High   Optimal
6      176 117       0         NA         196   Desirable      High
```

Output 2A: Results of line 10, printing head(df) to the SAS Log

Log | Results | Output Data (1)

Dataset SASHELP.HEART

Obs	Status	DeathCause	Sex	Chol_Status	BP_Status	Weight_Status	Smoking_Status	AgeCHDdiag	AgeAtStart	Height	Weight	Diastolic
1	Dead	Other	Female		Normal	Overweight	Non-smoker	.	29	62.50	140	78
2	Dead	Cancer	Female	Desirable	High	Overweight	Non-smoker	.	41	59.75	194	92
3	Alive		Female	High	High	Overweight	Moderate (6-15)	.	57	62.25	132	90
4	Alive		Female	High	Normal	Overweight	Non-smoker	.	39	65.75	158	80
5	Alive		Male	High	Optimal	Overweight	Heavy (16-25)	.	42	66.00	156	76
6	Alive		Female	Desirable	High	Overweight	Non-smoker	.	58	61.75	131	92
7	Alive		Female	Desirable	Normal	Overweight	Moderate (6-15)	.	36	64.75	136	80
8	Dead	Other	Male	High	Normal	Normal	Non-smoker	.	53	65.50	130	80
9	Alive		Male	Borderline	Normal	Overweight	Non-smoker	.	35	71.00	194	68
10	Dead	Cerebral Vascular Disease	Male	High	Normal	Normal	Light (1-5)	.	52	62.50	129	78
11	Alive		Male	Borderline	Normal	Overweight	Very Heavy (> 25)	.	39	66.25	179	76

Output 2B: Partial results of the show function in line 11. Note the number of records and the title.

Log | Results | Output Data (1)

WORK.DEAD

Columns: 17 Rows: 1,991

Enter expression

#	Status	DeathCause	Sex	Chol_Status	BP_Status	Weight_Status	Smoking_Status
1	Dead	Other	Female		Normal	Overweight	Non-smoker
2	Dead	Cancer	Female	Desirable	High	Overweight	Non-smoker
3	Dead	Other	Male	High	Normal	Normal	Non-smoker
4	Dead	Cerebral Vascular Disease	Male	High	Normal	Normal	Light (1-5)
5	Dead	Cancer	Male	Desirable	Normal	Underweight	Moderate (6-15)
6	Dead	Coronary Heart Disease	Male	Borderline	High	Overweight	Very Heavy (> 25)

Output 2C: Partial look at output dataset created in lines 13-14. 1991 rows in the dataset

```
106  %put &=numDead;
NUMDEAD=1991
```

Output 2D: Results of line 17, submitting the SAS code to show the macro variable created in line 15

```
[1] "Hello World"
```

Output 2E: Results of the SAS function used on line 18 and printed on line 19

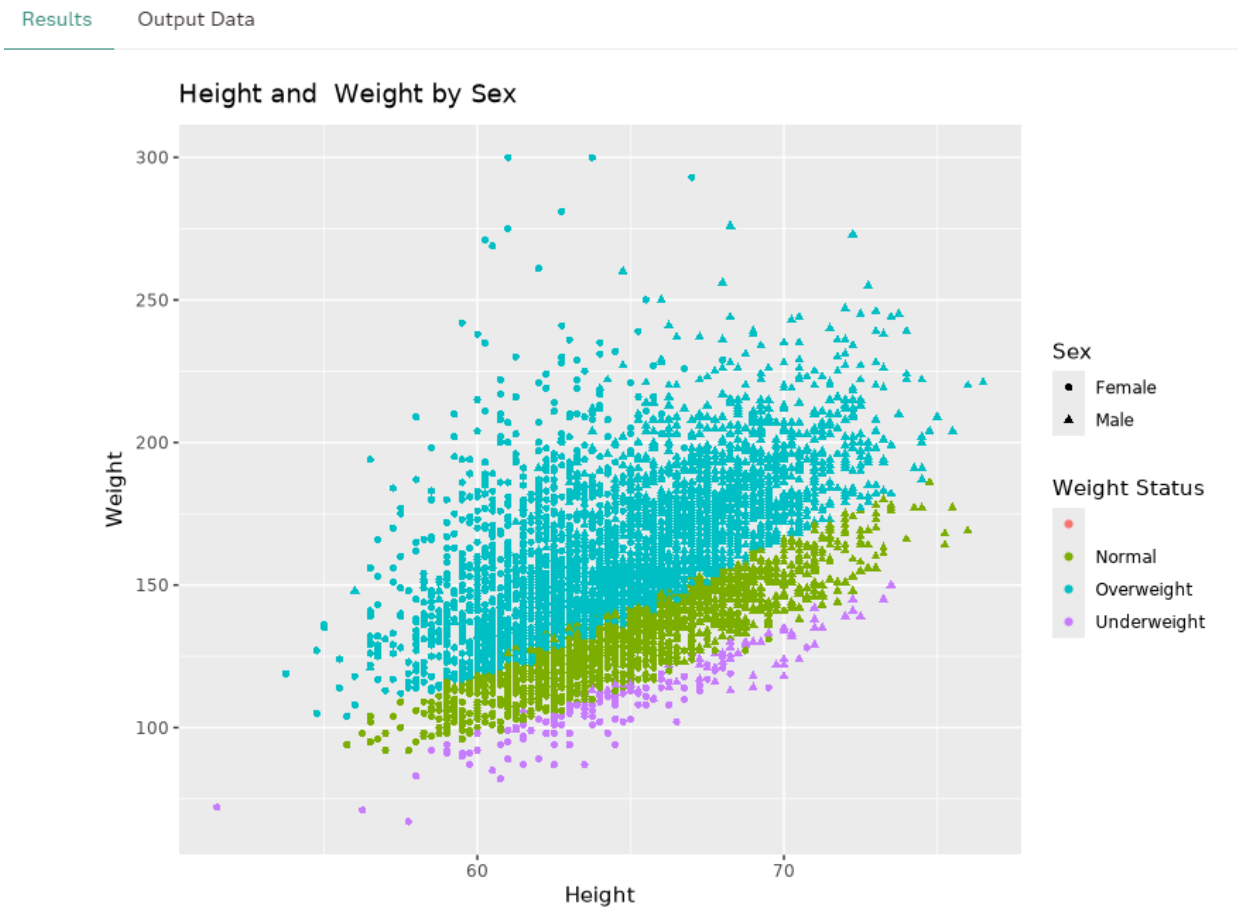
EXAMPLE THREE: R GRAPHICS

Code

```
%let inputTable = SASHELP.HEART;
%let X = Height;
%let Y = Weight;
%let Color = Weight_Status;
%let shape = Sex;
%let _Title = Height and  Weight by Sex;

PROC R;
submit;

library(ggplot2)

df= sd2df(symget("inputTable"))
plotX = sym(symget("X"))  # sym strips the quote envelopes
plotY = sym(symget("Y"))
plotC = sym(symget("Color"))
plotS = sym(symget("shape"))
titletxt = sym(symget("_Title"))

# Function to get either the label or the varaible name
get_label <- function(df, varname) {
  lbl <- attr(df[[varname]], "label")
  if (is.null(lbl) || lbl == "") varname else lbl
}

plt = ggplot(df, aes(x=!!plotX, y = !!plotY, color = !!plotC, shape = !!plotS)) +
  geom_point()+
  labs (title = titletxt,
        x = get_label(df, plotX),
        y = get_label(df, plotY)
        )

rplot(plt)

endsubmit;
RUN;
```

Program 3: Creating R graphics with ggplot

Code Line	Description
1 - 6	Define and populate SAS macro variables
8-9	Open the PROC R session
11	Initialize the ggplot package
13	Create the data frame df from &inputTable
14 - 18	Read the other parameters into R variables, Note the use of sym to strip out any quote envelopes
21-24	This is an R function that puts the column names into the dataframe labels if they were missing
26-31	Basic ggplot R code to create a graph. Note that it is being saved to an object called plt.
33	Display the graphics object in the Results pane
35-36	End the R session



Output 3: ggplot graphical output in the Results window

CONCLUSION

PROC R is a powerful new tool that lets you get the best out of SAS and R together. Combine it with PROC PYTHON, and you have the ability to use whatever code addresses the issue you face, all in the governed SAS ecosystem.

RECOMMENDED READING / VIEWING

- SAS Proc R Documentation
- Introducing PROC R: The Newest Way to Integrate R and SAS (Part 1) – Shelby Taylor
- Introducing PROC R: The Newest Way to Integrate R and SAS (Part 2) – Shelby Taylor
- Introducing PROC R (Part 3): Creating and Calling R Scripts – Shelby Taylor
- SAS Proc R Deep Dive – David Weik

CONTACT INFORMATION

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

Jim Box

Principal Data Scientist, SAS Institute

Jim.Box@sas.com

https://www.linkedin.com/in/jwbox/

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

brand and product names are trademarks of their respective companies.