

PEACEFUL CO-EXISTENCE: R AND SAS

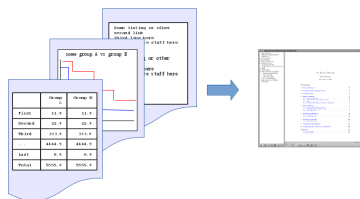
Martin Gregory, Merck
Manuel Cornes, Merck

PhUSE 2016, 9-12 October 2016

MERCK

- 1 Introduction
- 2 lst2pdf overview
- 3 Transfer Data SAS to R
- 4 Execute R from SAS
- 5 Conclusion

Problem statement



Inputs:

- text file: tables
- text file: listings
- graphics/pdf file: figures
- text file specifying titles and location of figures

Output:

- single PDF with tables
listings and figures in order
specified with bookmarks
constructed from the titles

Program must be available from SAS and R

Problem statement: example

Input file:

```
1 : section 1
1.1 : sub-section 1
Figure 1.1.1 : Figure Title

>>FIG>>/path/to/figure/graph.pdf

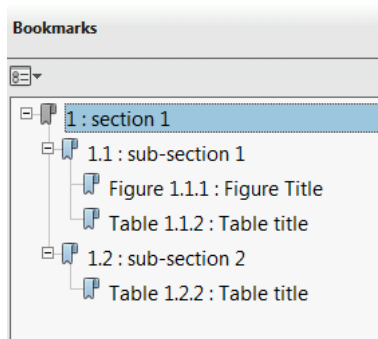
OCx
1 : section 1
1.1 : sub-section 1
Table 1.1.2 : Table title

Table body here ....

OCx
1 : section 1
1.2 : sub-section 2
Table 1.2.2 : Table title

Table body here ....
```

Bookmarks:



Implementation choices

Implementation in both R and SAS too expensive $\Rightarrow$ function in one language , caller function in other language.

We chose R for main implementation

- relative ease with which R can interface with the operating system in a transparent manner
- *pdf* graphics device and *text* function provide fine-grained control for writing text to a PDF
- availability of complex nested data structures required for parsing and checking input and constructing bookmark instructions
- building PDF can be resource intensive. Calculating bookmarks and checking for consistency requires multiple passes of the input files which is avoided using R as it holds the data in memory

R Function overview

```
lst2pdf(input=,           # vector of files
        output=,         # output PDF
        regulatory=FALSE, # check compliance
        watermark="DRAFT" # default to draft
)
```

Processing steps:

- read input files (R)
- parse and check title lines (R)
- build PDF using (R)
- identify graphics files (R)
- convert EPS graphs to PDF if necessary (ghostscript)
- concatenate PDFs if necessary (pdftk)
- build bookmarks if possible (pdftk)

SAS Macro overview

```
%lst2pdf(input=,          * input file or dataset name;  
          intype=LIST|FILE, * input is a list or file;  
          output=,         * output PDF file;  
          regulatory=FALSE, * check compliance;  
          watermark=DRAFT  * default to draft;  
          ) ;
```

Processing steps:

- if necessary
 - convert the input dataset to csv
 - write R program to convert csv to data frame,
- append R code to call *lst2pdf*
- execute the R program
- report the outcome

Transfer data from SAS to R: option 1

R provides two ways to read SAS datasets:

- ① *read.xport* reads SAS V5 transport format datasets
- ② *read.ssd* reads any sas7bdat format dataset, if
 - ① SAS is installed on the same host as R
 - ② the dataset satisfies SAS V5 transport constraints

Transfer data from SAS to R: option 2

SAS/IML can transfer datasets to R if

- ① SAS/IML is licensed
- ② for SAS 9.3 only $R \leq 2.15.3$ is supported
- ③ only supported on Linux and Microsoft Windows
- ④ need a version of R compiled for shared library use $\Rightarrow$
 - ① 10-20% loss in performance
 - ② potential symbol conflicts
 - ③ packages containing C code might not run

Transfer data from SAS to R: option 3

Use a DBMS and avoid transfer!

Transfer data from SAS to R - issues

Data type conversion can be tricky:

MySQL	SAS	R	MySQL	SAS	R
bigint	numeric	double	datetime	numeric	char
mediumint	numeric	integer	timestamp	numeric	char
smallint	numeric	integer	time	numeric	char
tinyint	numeric	integer	year	numeric	char
decimal	numeric	double	char	char	char
int	numeric	integer	varchar	char	char
double	numeric	double	binary	char	char
float	numeric	double	varbinary	char	char
date	char	numeric	enum	char	char
set	char	char			

Transfer data from SAS to R: %lst2pdf

Our input dataset contains a character variable which may > 200
We have written `%sasds2df` which:

- 1 converts any SAS dataset to a CSV containing:
 - a row with the target data type in R
 - a row with the variable names
 - the data

for the `sashelp.class` dataset this would be:

```
'character', 'character', "numeric", "numeric", "numeric"  
"Name", "Sex", "Age", "Height", "Weight"  
"Alfred", "M", 14, 69, 112.5  
"Alice", "F", 13, 56.5, 84  
...  
"William", "M", 15, 66.5, 112
```

- 2 writes an R program to import the CSV

Execute R from SAS: options

SAS/IML Using the *submit /R* statement R code can be submitted from the *IML* procedure. In SAS 9.3, we were able to run R code but not transfer datasets;

pipe file reference This allows a data step to read from the pipe, retrieving the messages returned by the R program execution (the R transcript). It does not provide a return code;

system function This data step function allows us to retrieve the return code but does not directly return the R transcript which needs to be read separately;

x statement The x statement neither retrieves the return code nor the transcript.

Ideally we need return code, transcript or both, so the pipe and system functions are the best candidates.

Execute R from SAS: %lst2pdf

```
filename r1 pipe "R &rdebug --no-save --no-restore <&rpgrm" ;
data _null_ ;
    length rc $8 ;
    do until (eof) ;
        infile r1 end=eof ;
        input ;

        if _infile_ = "WARNING: ignoring environment value of R_HOME"
            then continue ;

        if scan(_infile_,1,":")="R_TO_SAS_RETURN_CODE" then do ;
            %if &debug=Y %then %str(put _infile_ ;) ;
            rc=strip(scan(_infile_,2,":")) ;
            call symput("&rc",rc) ;
            end ;
        else put _infile_ ;
        end ;
    stop ;
run ;
```

Execute R from SAS: issues

- retrieving both return code and transcript
⇒ pipe is only reasonable option
- character values must be quoted:

```
put "lst2pdf(input=lst.files,output='&output'," /  
    "regulatory='&regulatory',watermark='&watermark'", /  
    gs='&gs',pdftk='&pdftk'" /  
    ")" ;
```

Conclusion

transferring data multiple options exist, care must be taken with data type conversion

execution of R code from SAS pipe allows integrating R transcript in SAS log and passing return code

implementing in one language reduces development and validation effort

with relatively small effort we can integrate R applications in SAS