Paper 10420-2016

On The Fly SAS® Reports

Ronald R. Palanca
Mathematica Policy Research, Princeton, NJ

Abstract

In today's fast paced work environment, time management is crucial to the success of the project. Sending requests every time to SAS® programmers to run reports to get the most current data can be a stretch sometimes on an already strained schedule. Why bother to contact the programmer? Why not built-in the execution of the SAS® program in the report so that upon launching of the report, real time data will be retrieved and report will show most recent data.

This paper will demonstrate that by opening an existing SAS® report in Microsoft Word or Excel, the data in the report will refresh automatically. Simple Visual Basic for Applications (VBA) codes are written in MS Word or Excel that upon opening of an existing SAS report, the SAS® programs that creates the report are called from within the MSOffice product and will overwrite the existing report data with the most current data.

This feature will ease the burden on the SAS® programmer by eliminating the need for requests to run SAS programs and can therefore save time and money for the project.

Introduction

SAS® reports are normally refreshed by running the SAS® program. This paper will illustrate that not only can SAS® programs be run from within Microsoft Excel and Word but it also automatically refreshes the data upon launching of the reports. By the time the report opens, real time data will be reflected on the report.

VBA is used to execute SAS® programs but no extensive knowledge of VBA codes is needed to be able to add this feature to SAS® reports created in MS Word or Excel.

The real time approach will work for Excel reports regardless if it has single or multiple sheets.

How does it work?

First, we need a SAS® Program that will output an MS Excel or Word Report. To make this dynamic or real time, a good source of data is a database from an SQL server. For the purpose of outputting report into Excel or Word, the paper used ODS Tagsets in the SAS® program.

Example of SAS® Program for MS Excel output

```
ODS Tagsets.ExcelXP File="C:\Report\Proj_Reports.xls" STYLE=minimal options(embedded_titles='yes');
ODS Tagsets.ExcelXP Options(sheet_name = "Sheet1");
        Proc Tabulate ……..;
          Statements …;
        Run;
ODS Tagsets.ExcelXP Options(sheet_name = "Sheet2");
        Proc Tabulate ……..;
          Statements …;
        Run;
ODS Tagsets.MSoffice2k close    ;
```

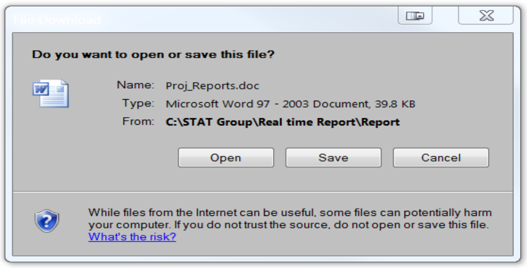
Figure 1

Example of SAS® Program for MS Word output

```
ODS TAGSETS.msoffice2k file=" Proj_Reports.doc" STYLE=minimal OPTIONS ( Orientation = 'landscape' FitToPage = 'yes'
Pages_FitWidth = '1' Pages_FitHeight = '100' );
        Proc Tabulate ……..;
          Statements …;
        Run;
ODS Tagsets.MSoffice2k close    ;
```

Figure 2

As of this writing, SAS® ODS Tagset does not output an MS Excel file format of "xlsx" or the MS Word file format of "docx". The output can only be in ".xls" or ".doc" file format. However, to view the '.xls' or '.doc' output, a warning message (Fig. 3) would always pop up before the report opens.


Do you want to open or save this file?

Name: Proj_Reports.doc
Type: Microsoft Word 97 - 2003 Document, 39.8 KB
From: C:\STAT Group\Real time Report\Report

While files from the Internet can be useful, some files can potentially harm your computer. If you do not trust the source, do not open or save this file. What's the risk?

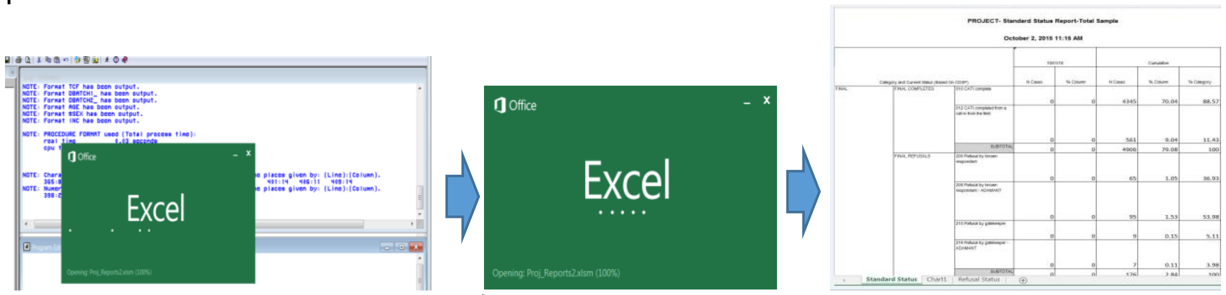
Figure 3

In the latest version of MSOffice which is Office 2013, VBA (Visual Basic for Applications) codes can be written in macro driven Excel (.xlsm) or Word (.docm) files only. This paper will be using VBA so the MS Office file format to be used is either '.xlsm' or '.docm'.

The challenge is how to move the output that is created in 'xls' or 'doc' file type to the latest version of MS Excel and Word and overwrite the current data with real time data.

Upon launching of the Excel or Word report, the following happens: the VBA code will run the SAS® program and move the report that was created by SAS® program to the report that is opening.

As illustrated in Figure 4 below, SAS® is running in the background while the Excel file is opening. After execution of SAS® program, SAS exits and then the updated report opens.


Office
Excel
Opening: Proj_Reports2.xlsm (100%)
Office
Excel
Opening: Proj_Reports2.xlsm (100%)
Standard Status
Chart1
Refusal Status

```
NOTE: Format TCF has been output.
NOTE: Format DBATCH1_ has been output.
NOTE: Format DBATCH2_ has been output.
NOTE: Format AGE has been output.
NOTE: Format $SEX has been output.
NOTE: Format INC has been output.

NOTE: PROCEDURE FORMAT used (Total process time):
      real time           0.03 seconds
      cpu t

NOTE: Chara                                 e places given by: (Line):(Column).
      365:8                                    401:14   406:11   409:14
NOTE: Numer                                 e places given by: (Line):(Column).
      398:2
```

PROJECT- Standard Status Report-Total Sample

October 2, 2015 11:15 AM

Category and Current status (Based On CDSP)			[illegible]		Cumulative		
			N Cases	% Column	N Cases	% Column	% Category
FINAL	FINAL COMPLETED	210 CATI complete	0	0	4345	70.04	88.57
		212 CATI completed from a call-in from the field	0	0	561	9.04	11.43
		SUBTOTAL	0	0	4906	79.08	100
	FINAL REFUSALS	200 Refusal by known respondent	0	0	65	1.05	36.93
		208 Refusal by known respondent - ADAMANT	0	0	95	1.53	53.98
		210 Refusal by gatekeeper	0	0	9	0.15	5.11
		218 Refusal by gatekeeper - ADAMANT	0	0	7	0.11	3.98
		SUBTOTAL	0	0	176	2.84	100

Figure 4

How to set up VBA.

For Excel. Open Excel and save as "*.xlsm "(The name of report as defined from SAS® program). Select 'Developer' (Alt + F11) in menu bar and click on Visual Basic and select 'ThisWorkbook'.
For Word. Open Word and save as "*.docm "(The name of report as defined from SAS® program). Select Developer (Alt + F11) in menu bar and click on Visual Basic and select 'ThisDocument'.

Visual Basic code defines an object and creates an instance of SAS® to associate with that object. Sample VBA code for both Excel and Word can be found at the end of the paper. Here is the sample code in Excel

```
Dim sasobj As Object                                  'Declare a variable that refers to an application/object
Set sasobj = CreateObject("SAS.Application")
sasobj.Visible = True                                 'To make interactive SAS visible
sasobj.submit ("%include 'C: \Proj_Reports_xls.sas';")  ' Run the SAS Program that creates the report
sasobj.submit ("ENDSAS;")                             'To exit interactive SAS

'*******Define variables
Dim wbOpenSource As Workbook
Dim wsSheetSource As Worksheet
Dim rngCopy As Range
Dim


Set wbOpenSource = Workbooks.Open("C:\Proj_Reports2.xls")              'Open Spreadsheet created by SAS
Set wsSheetSource = wbOpenSource.Worksheets("Standard Status")          'Worksheet where to  get data
Set rngCopy = wsSheetSource.Range("a:zz").EntireColumn                  'Set the range of the cells to be copied
rngCopy.Copy                                                            'Copy  data


Set wbOpenDest = Workbooks("Proj_Reports2.xlsm")                        'Activate the Destination workbook
Set wsSheetDest = wbOpenDest.Worksheets("Standard Status")              'Sheet where to paste data
Set rngPaste = wsSheetDest.Range("a1")                                  'Starting cell where to paste data
rngPaste.PasteSpecial                                                   'Paste data to active workbook


'*******Add codes here if there are more sheets to copy
Set wsSheetSource
Set rngCopy


wbOpenSource.Close                                                      'Close the spreadsheet where SAS data came from
Kill "C:\Proj_Reports2.xls"                                             'Delete the spreadsheet where SAS data came from
```

Summary of the whole process

A SAS® program that will output a report in Excel or Word. Proc Tabulate or Proc Report would be a good tool for the presentation of the data.

For Excel: The VBA code will execute the SAS® program. The VBA code will then open the output spreadsheet from SAS® program, go to worksheet specified in VBA and copy everything in that worksheet. The copied worksheet will then be pasted to the corresponding worksheet of the current spreadsheet. If you have multiple worksheets, it will open the next worksheet specified in VBA code and copy report to the corresponding worksheet in the current spreadsheet.

For Word, Same as in Excel, everything from the SAS® output is copied to the current document. An extra step of deleting the table in current document has to be made first before pasting updated data in the current document. The reason for the deletion is that Word report is created in table or static format and it can't just be overwritten.

After execution of the paste command, the Excel or Word file created by the SAS® program is closed and then deleted.

Conclusion

This paper creates an automated way of updating the values of an existing MS Excel or Word report upon opening of the report.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Please feel free to contact the author:

Ronald R. Palanca: RPalanca@mathematica-mpr.com

Mathematica Policy Research
600 Alexander Park
Princeton, NJ 08543
(609) 799-3535

Example VBA code for Excel Report

```
Private Sub Workbook_open()
Dim sasobj As Object                                    'Declare a variable that refers to an application/object
Set sasobj = CreateObject("SAS.Application")
Application.DisplayAlerts = False
sasobj.Visible = True                                   'To make interactive SAS visible
sasobj.submit ("%include 'C: \Proj_Reports_xls.sas';")  ' Run the SAS Program that creates the report
sasobj.submit ("ENDSAS;")                               'To exit interactive SAS

Application.Wait (Now + TimeValue("0:00:20"))           'Pause spreadsheet opening to let SAS finish running
  Dim wbOpenSource As Workbook
  Dim wsSheetSource As Worksheet
  Dim rngCopy As Range
  Dim wbOpenDest As Workbook
  Dim wsSheetDest As Worksheet
  Dim rngPaste As Range

'---------------------------------------------------------------------------
'--The code below will copy the data from the xls file to the macro driven EXCEL spreadsheet
'---------------------------------------------------------------------------
  Set wbOpenSource = Workbooks.Open("C:\Proj_Reports.xls")          'Open Spreadsheet created by SAS Program
  Set wsSheetSource = wbOpenSource.Worksheets("Sheet1")             'Worksheet where to  get data
  Set rngCopy = wsSheetSource.Range("a:zz").EntireColumn            'Set the range of the cells to be copied
  rngCopy.Copy                                                      'copy  data from where SAS data was written

  Set wbOpenDest = Workbooks("Proj_Reports.xlsm")                   'Activate the Destination workbook
  Set wsSheetDest = wbOpenDest.Worksheets("Sheet1")                 'Sheet where to paste data
  Set rngPaste = wsSheetDest.Range("a1")                            'Starting cell where to paste data
  rngPaste.PasteSpecial                                             'Paste data to active workbook

  wbOpenSource.Close                                                'close the spreadsheet created by SAS program
  Kill "C:\ Report\Proj_Reports.xls"                                'Delete the spreadsheet
End Sub
```

Example VBA code for Word Report

```
Private Sub Document_open()
 Dim sasobj As Object                                     'Declare a variable that refers to an application/object
 Set sasobj = CreateObject("SAS.Application")
 Application.DisplayAlerts = False
 sasobj.Visible = True                                    'To make interactive SAS visible
 sasobj.submit ("%include 'C: \Proj_Reports_Word.sas';")  ' Run the SAS Program that creates the report
 sasobj.submit ("ENDSAS;")                                 'To exit interactive SAS
 Application.OnTime When:=Now + TimeValue("00:00:20"), Name:="MyDelayMacro"   'Delays the opening of Word
End Sub
Public Sub MyDelayMacro()
  Dim oWD As Word.Document
  Dim wOpenSource As Document
'---------------------------------------------------------------------------
'--The code below will copy the data from the doc file to the macro driven WORD document
'---------------------------------------------------------------------------
  Set oWD = ActiveDocument
  If oWD.Tables.Count <> 0 Then                                  'Check if there are any table in the document
      oWD.Tables(1).Delete                                       'Delete all tables in the document
  End If
  Selection.WholeStory                                           'Select the rest of the Text from the document
  Selection.Text = ""                                            'Delete all the Text
'-----Get the Document created by the SAS Program
  Set wOpenSource = Documents.Open("C:\Proj_Reports.doc")        'Open the document created by SAS
  Selection.WholeStory                                           'selects the whole document
  Selection.Copy                                                 'Copy the whole document
'-----Copy new data to the Reports document
  Documents("Proj_Reports.docm").Activate
  Selection.EndKey wdStory                                       'It moves the data to the end of the current line
  Selection.PasteAndFormat Datatype = wdPasteText  'wdPasteDefault   'Paste the data to the current document
 wOpenSource.Close                                               'Close the document created by SAS
 Kill "C:\Proj_Reports.doc"                                      'Delete the document created by SAS
 ThisDocument.Save                                               'Save the current document
End Sub
```