Paper 041-31

Processing Large Environmental Datasets With SAS® : Writing Robust and Dynamic SAS Macro Code

Peter Mooney and Adam Winstanley, National Center of Geocomputation,John Hume Building, National University of Ireland Maynooth, Co. Kildare. Ireland.

Conor McDonagh, Dept of Computer Science, University College Dublin, Belfield, Dublin. Ireland.

ABSTRACT

We discuss the application of SAS to a number of environmental datasets. Usually reports, graphical visualizations, summary statistics, and subset or aggregate datasets must be generated from one or more of these datasets. A library of SAS macros was developed while building a reporting system for several large and complex environmental datasets. This paper discusses a number of SAS code segments from these macros that may be easily extended to other similar projects. The intended audience is the SAS macro programmer and concentrates on several key areas:

- Using the DATASTEP to generate CSV and XML file output corresponding to data used by GPLOT and GCHART;
- Augmenting basic GOPTIONS and CHART options to produce graphical output (JPEG) for several different variables in time-series datasets;
- Using local macro variables to generate dynamic AXIS statements (for graphics) and WHERE statements (for DATASTEP).

The SAS code is contained within macro definitions and offers maximum opportunity for code reuse

INTRODUCTION

There are a number of key drivers behind the dissemination and display of environmental information on the Internet. Members of the public will look to the Internet for easily understandable information about the environment while the scientific community will look to the Internet for the underlying core datasets, derived statistics and analysis (Mooney et al, 2004). In the last 18 months of this environmental data management project several different environmental datasets have been assigned. These datasets originate from different thematic areas (air, water, climate, waste, etc.). Thematic differences present an inhomogeneous computing environment. Datasets are of different size, complexity (in structure and semantics), and file formatting. The majority of these environmental datasets are stored in MS Excel-based datasets with others stored in MS Access databases and CSV (Comma Separated Value) formats. Despite the lack of thematic overlap all the datasets are subject to similar reporting requirements. This paper reports on the development of a library of SAS macros to streamline the process involved in web-based dissemination of graphical representations of these datasets (based on parameterised input from users via HTML forms) and download services. Download services will provide users with the entire dataset (in raw or converted formats) or subset aggregate forms of the original dataset (also in raw or converted formats). The core dataset used in this paper for purposes of example is a water chemistry dataset. It has 12 columns *(River Code, Time of Sample, Dissolved Oxygen, BOD, Temperature, pH level, Ammonia, Chloride, Alkalinity, Nitrate, Total Hardness,* and *Conductivity)*. This is shown in Figure 1 below.

The key requirements are as follows:

1. The SAS program must be handed over to non-SAS users. It must be easy for these users to change input parameters and regenerate output;
2. For each measurement parameter in the core dataset a time-series plot is required in JPEG format;
3. All dataset contain a CODE column – a character string that uniquely identifies the geographical location of the data record;
4. The data used to plot the time-series graphic in step 2 must be made available in CSV and XML format;
5. A consistent file naming scheme must be adopted for all graphics and output data files. These files will all be linked to a web-based query tool using their filenames as unique identifiers;
6. The parameters for the macros must be easily configurable;

7. The PROC UNIVARIATE procedure must be available in a macro definition and the output from this procedure must be in HTML format with a graphical representation of the statistical distribution;
8. The entire SAS program should easily re-run if the underlying core SAS dataset changes;

DATA TRANSFORMATION

The original datasets are maintained in their current format but also transform/convert into equivalent SAS Datasets. This allows the original data producers to continually manage their datasets within their own software applications. Each dataset is converted on a case-by-case basis using Java and exported to a series of CSV files. These CSV files are imported into SAS®. Software development effort can now be directed toward a library of SAS macros capable of dealing with these environmental datasets.

To summarize the components of our reporting system:

1. Data providers prepare their datasets (usually in non-SAS formats). They check their final submission for various data quality issues;
2. In the case of complex-structured Excel sheets and CSV files a Java program is used to convert the data from these formats into a more accessible set of CSV files. This is a key step as the data providers are secure in the knowledge that their original data is not changed or edited in any way.
3. SAS is used to import the data in the CSV files into SAS Datasets. There is usually a separate SAS import program for each dataset. The INFILE statement is used.
4. *Data Reporting Step:* This step contains a number of sub-steps;
    1. If basis statistical analysis or tabulated summaries of various characteristics of the dataset is the only requirement these are generated using SAS ODS functionality.
    2. If graphical output is required these are usually generated with PROC GPLOT and PROC GCHART and output in JPEG format (following a naming convention for files)
    3. Usually the data used to generate the charts in the previous step must also be made available. This is generated by transforming SAS datasets into both CSV and XML formats (and also following the same naming convention)
5. The data providers are informed of the conclusion of the reporting generation process. In the cases of many of the core datasets an online resource will be made available from our website.

	newcode	SASDate_	BOD1	TEMP1	DO1	Conductivity1	pH1
265	03M01-0100	26/01/2000	1.2	.	98	173	8.2
266	03M01-0100	29/05/2000	1.5	10	101	93	8.2
267	03M01-0100	26/07/2000	1.8	17	97	320	8.3
268	03M01-0100	19/10/2000	1.2	.	98	75	7.5
269	03M01-0100	28/11/2000	2.7	.	97	45	6.9
270	03M01-0100	22/02/2001	1.3	.	97	121	7.9
271	03M01-0100	29/05/2001	0.8	17	103	179	8.2
272	03M01-0100	23/10/2001	1.5	11	96	85	7.5
273	03M01-0200	21/03/1978	.	.	84	95	7.1
274	03M01-0200	03/04/1978	1.1	.	84	147	7.5
275	03M01-0200	17/04/1978	1.3	.	96	222	8.2
276	03M01-0200	29/05/1978	1.4	19	100	232	.
277	03M01-0200	22/08/1978	1.2	14	90	18	7.4
278	03M01-0200	05/09/1978	1.4	12	88	314	8.2
279	03M01-0200	21/09/1978	2.3	13	23	202	7.8
280	03M01-0200	04/01/1979	0.6	0.5	92	.	7.6
281	03M01-0200	06/06/1979	1.3	15	108	240	8.5

Figure 1: View of the Core Dataset Used in this Paper

DESCRIPTION OF THE MACRO PROGRAM

Global variables are declared with the full file directory paths to the folders where the various types of output are written. The pixel size of output graphics (width as `&xpixelSize` and height as `&ypixelSize`) are also declared as

global variables. All of these variables are easily changed by the user (even without in-depth SAS knowledge) if the underlying dataset changes or different types of graphical output is required.

```
%let graphicsPath = C:\SASdata\mquinn\full\rivers\output\images\;
%let xmlPath = C:\SASdata\mquinn\full\rivers\output\xml\;
%let csvPath = C:\SASdata\mquinn\full\rivers\output\csv\;
%let htmlPath = C:\SASdata\mquinn\full\rivers\output\html\;
%let xpixelSize = 640;
%let ypixelSize = 480;
```

The macro header line is as follows:

```
%macro drawRiverGraph(
          theDataset=,
          theCode=,
          parameter=,
          supText=,
          Lim=,
          Location=,
          chemtype=);
```

The macro parameters are as follows: `theDataset` is the SAS Dataset containing the River data, `theCode` is the particular river code that the graph must be drawn for; `parameter` is the River water chemical parameter to graph over time; `supText` is the text for the y-axis and TITLE2; `Lim` is the list of exceedance limits for the parameter specified by `parameter`. The parameter `&Location` is used to describe the geographical location of the coverage of the dataset. This parameter is used in the output of XML. Finally, `&chemType` is a description of the chemical parameter also used in the output of XML.

Next the core dataset specified by `theDataset` is subset into the temporary dataset `WORK.temp`; This dataset is sorted by DATE. Only the columns for Date of sampling and the river parameter are kept.

```
data WORK.temp;
       set &dataset;
       where CODE = &theCode;
run;

PROC SORT DATA=WORK.temp(KEEP=SASDate &parameter)
    OUT=WORK.RiverData;
    BY SASDATE;
RUN;
```

For this run of the macro the temporary SAS dataset `WORK.RiverData` is used by PROC GPLOT. This dataset is also used to generate the CSV and XML output files. A number of PROC SQL statements are used to find the extents of both the X (time series) axis and Y (parameter) axis. The results of each PROC SQL statement are formatted and stored in local macro variables using the `putn()` statement. The X-axis is formatted as `date7.` and the Y-axis is formatted as decimal numbers `z6.2`.

```
PROC SQL noprint; select max(SASDate) into: maxDate from WORK.temp; quit;

PROC SQL noprint; select min(SASDate) into: minDate from WORK.temp; quit;

PROC SQL noprint; select max(&parameter) into: maxYAxis from WORK.temp; quit;

PROC SQL noprint; select min(&parameter) into: minYAxis from WORK.temp; quit;

%let theMaxD = %sysfunc(putn(&maxDate,date7.));
%let theMinD = %sysfunc(putn(&minDate,date7.));
%let theMinYAxis = %sysfunc(putn(&minYAxis,z6.2));
%let theMaxYAxis = %sysfunc(putn(&maxYAxis,z6.2));
```

Next the graphical output options are set. The output filename is a character string created from the concatenation of the global variable `&graphicsPath` (holding the output graphics folder path), the code of the river, and the specific parameter depicted in the time-series graphic. JPEG graphics are used.

```
filename out "&graphicsPath.RiverSampleGroup_&thecode._&parameter..jpg";
GOPTIONS RESET=ALL;
GOPTIONS xpixels=&xsize ypixels=&ysize dev = jpeg
gsfname=out gsfmode=replace cback=WHITE;
```

The axis statements are constructed with the TITLE statements. The statements are fully parameterized and do not need to be altered for different parameters or river codes. The timescale of the data time series is approximately 30

years so that the X-axis (AXIS2) is order in 5-year time slices. The PROC GPLOT statement is straightforward with the plot statement generated using the chemical parameter and the DATE column. Horizontal reference lines are plotted as stipulated by VREF.

```
AXIS1 LABEL=(FONT='Sans Serif' a=90 HEIGHT=14pt JUSTIFY=RIGHT "&supText" )
MINOR=none ORDER=("&theMinYAxis" to "&theMaxYAxis" by 5.0);
AXIS2 label=('Measurement Date') ORDER=("&theMinD"d to "&theMaxD"d by year5);
TITLE "River Sample Group - Code (&theCode)" COLOR =blue height=2;
TITLE2 "&supText";

PROC GPLOT DATA=WORK.RiverData;
    plot &parameter*SASDATE /
    VAXIS=AXIS1
    HAXIS=AXIS2
    VREF=&lim CVREF=CX008000 LVREF=1
    FRAME;
RUN;QUIT;
```

At this point the JPEG image is created and stored in the image output directory.

GENERATING CSV AND XML OUTPUT

The data used to draw the graphical plot in PROC GPLOT must be written out to CSV and XML format so that other users may retrieve the data and analyse it for themselves on their own machines. The aim of this section of the macro is to automatically create the output file path and file path name from the global variables and parameters passed to the macro. This ensures that a consistent file naming scheme is maintained.

For example, if the river code as **03B01-0100** and the river parameter **Phosphate** then the output file path and filename for the XML output would be **C:\SASdata\mquinn\full\rivers\output\xml\03B01-0100 - Phosphate.xml**. The corresponding SAS code to create this path string is `(&xmlPath.&riverCode-.&parameter..xml)`

To create a valid XML output file (W3C, 2004) we need to write out the opening XML header tags, then the body of the XML data file, and finally the closing XML file tags. This is achieved using NULL DATASTEP and a number of PUT statements.

```
        data _null_; /* Write out XML header information */
        FILE   "%sysfunc(compress(&xmlPath.&riverCode-.&parameter..xml))" ;
        PUT  '<?xml version="1.0" encoding="UTF-8"?>';
            PUT  '<WFDArchive location = "&Location">';
        run;
        data _null_; /* modify the file and write out the temporary data */
              set  WORK.RiverData;
              FILE "%sysfunc(compress(&xmlOutput.&riverCode-.&parameter..xml))" mod
;
              PUT  '<WaterSampling><SampleDate>' SASDate_
'</SampleDate><RiverCode>';
           PUT  &riverCode '</RiverCode>';
           PUT '<Chemical type = "&chemType" unit = "&supText">' &parameter;
           PUT '</Chemical></WaterSampling>';
    run;
       data _null_; /* finally write closing XML tags */
       FILE  "%sysfunc(compress(&xmlOutput.&riverCode-.&parameter..xml))" mod;
       PUT  '</WFDArchive>';
       run;
```

This produces XML records with the following format:

```
<WaterSampling>
    <SampleDate>14/02/1978</SampleDate>
    <RiverCode>03B01-0100</RiverCode>
    <Chemical type = "Ortho-Phosphate" unit = "mg P 1-1">0.005</Chemical>
</WaterSampling>
```

A similar series of PUT statements are used to create the output CSV file.

The macro concludes with the resetting of all GOPTIONS, TITLE, and FOOTNOTE.

```
GOPTIONS RESET=ALL;
TITLE;FOOTNOTE;RUN;
%mend drawRiverGraph;
```

RUNNING THE MACRO

The `drawRiverGraph` macro can be executed in two ways depending on the dataset requirements. To call the macro on a single river (say code 100A) for the parameter Biochemical Oxygen Demand (BOD) for Northern Rivers with BOD lower and upper limits of 2.5 and 4.0.

```
%drawRiverGraph(
           theDataset=EPA2005.RiversNorth,
         theCode=100A,
            parameter=BOD,
         supText = Biochemical Oxygen Demand (mg/L),
         lim=2.5 4.0, Location=Northern River Basin District, chemType=River BOD);
```

This type of macro call is suitable for isolated cases. For N = 10 parameters on 163 separate river codes this type of macro call is not the most convenient way to call the macro for every (parameter, river code) pair. The `CALL EXECUTE` Statement is the most appropriate method. The PROC SQL statement extracts all distinct parameter river-code pairings.

```
PROC SQL; CREATE TABLE WORK.tuples AS
       SELECT distinct RiverCode, Parameter FROM EPA2005.RiversNorth
       WHERE Parameter = "BOD";
QUIT;
DATA _NULL_;
       %let P; %let RCode;
       SET WORK.tuples;

       call symput('P',Parameter);
       call symput('RCode',RiverCode);
       call execute(
              '%drawRiverGraph(
                     datasetname=EPA2005.RiversNorth,
                     theCode='|| '&RCode' || ',
                     parameter=' || '&P' || ',
                     supText=Biochemical Oxygen Demand (mg/L),
                     lim=2.5 4.0,
                  Location=Northern River Basin District,
                  ChemType=River BOD);');
RUN;
```

This CALL EXECUTE statement is then modified for the other (N-1) river chemical parameters and their corresponding `supText` and limits of exceedances. The output directories are then manually moved to the web-server. A customized HTML-based query page allows users to download graphics and supporting datasets for any of the parameters on any river.

GENERATING HTML OUTPUT FROM PROC UNIVARIATE

In many cases the data providers are interested in the statistical characteristics of their core datasets supplied to our reporting system. With PROC UNIVARIATE (SAS, 1999) one can retrieve a very detailed statistical analysis including basic statistical measures, moments, basic confidence limits, tests for location, quantiles, and fitted distributions. Another important feature of the output is the ability for users to investigate if their data fits a statistical distribution i.e. normal distribution, logarithmic distribution, etc.

The macro **`generateStatsHTML`** was developed to meet these requirements. Many of the parameters are the same as above: **`dnsName`** is the core SAS dataset, **`Code`** is the river or geographical location under statistical analysis, **`alphaLevel`** is an input to PROC UNIVARIATE representing the confidence interval to compute, **`statsVar`** is the chemical parameter for statistical analysis on **`Code`**. Finally **`supText`** is used for the TITLE string. All global variables are as above.

```
%macro generateStatsHTML(dnsName=,
                         Code=,
                         alphaLevel=,
                         statsVar=,
                         supText=);
```

The macro begins by creating a temporary SAS dataset WORK.TEMP by subsetting the core SAS dataset **`dnsName`** by the `Code`. The GOPTIONS are configured as above. The ODS HTML statement is configured to output all HTML files to the directory specified in the global variable `&htmlPath`. For example, the full output html for River 100A-100 with chemical parameter Temp would resolve to **C:\SASdata\mquinn\full\rivers\output\html\100A-100-Temp-**

stats.html. In a similar fashion the graphical output from PROC UNIVARIATE would be directed to the folder **C:\SASdata\mquinn\full\rivers\output\html\100A-100-Temp-stats.jpeg**

PROC UNIVARIATE is parameterized. The confidence level ALPHA is assigned the value of `&alphaLevel`. The variable for statistical analysis is assigned the column name specified in `&statsVar`. The remainder of PROC UNIVARIATE is hard-coded. These values can be easily changed to meet other reporting aesthetic guidelines if required. The macro closes with the usual closing of the ODS destinations and GOPTIONS reset.

```
        data Work.temp (KEEP = &statsVar);
              set &dnsName;
              where CODE = &Code;
        run;

        ods listing;
        ods html body="&htmlPath.& Code-&statsVar.stats.htm" style=statdoc;
        TITLE1 "&supText";

        GOPTIONS RESET;
        filename out1 "&graphics.& Code.&statsVar.-stats.jpeg";

        GOPTIONS xpixels=&xPixelSize ypixels=&yPixelSize
        cback=WHITE dev = jpeg
        gsfname=out1
        gsfmode=replace;

        PROC UNIVARIATE DATA=Work.temp CIBASIC(ALPHA=&alphaLevel )  MU0= 0;
              VAR &statsVar;
                  HISTOGRAM / Normal(  W= 2 L= 2 COLOR=RED mu=EST sigma=EST )
                  CAXES=BLACK CFRAME=CXE0E0E0 NOVLABEL CBARLINE=BLACK CFILL=BLUE
                  PFILL=SOLID WAXIS=1
              vscale=count;
        RUN;
        QUIT;
        ods html close;
        ods output close; GOPTIONS RESET;
  %mend generateStatsHTML;
```

RUNNING "GENERATESTATSHTML" MACRO

Running the `generateStatsHTML` macro is very similar to the `drawRiverGraph` macro described above.

```
  % generateStatsHTML (
             dnsName=EPA2005.RiversNorth,
           Code=100A-100,
             alphaLevel=0.05,
             statsVar=TEMP
           supText = Water Temperature);
```

To force PROC UNIVARIATE to look at 95% confidence limits the alphaLevel parameter is given the value of 0.05. The user can change this as required. This macro can be integrated into a CALL EXECUTE routine just as described above. This is omitted for brevity. Further details are available in the Recommended Reading section below.

To use the generateStatsHTML macro on another completely different core dataset the macro is called in a similar fashion. The only requirements of this new core dataset is that it contains a time-series variable, at-least one numerical based column, and a CODE column – a unique identifier for the location of measurement. The global variables are changed to the new output locations. Suppose then that the new core dataset contains Air Quality Data.

```
   % generateStatsHTML (
             dnsName=EPA2005.AmbientAirQuality,
           Code=Wicklow:Avondale, alphaLevel=0.05,statsVar=O3,supText = Ambient Ozone);
```

CONCLUSION

We have demonstrated that with the combined use of SAS programmer generated macros, DATA steps, and ODS a powerful WWW-enabled reporting system that can be built. More so the system is not a built to "stove-pipe system"

specifications and can be easily modified to deal with different datasets and different output requirements through various parameterizations in the macro header lines. The expressive power of the SAS language allows the development of a very robust reporting system with only a few hundred Lines of Code (Kaner and Bond 2004)

The macro code presented in this paper has been used on several different environmental datasets originating from different domains – water quality analysis, air quality monitoring, climate change modeling. All of these datasets are characterized by a time-series parameter and several numerical parameters. Only the parameter names in the `CALL EXECUTE` statement must change. Of course the macro programmer is free to extend this macro code to change plot type, plot appearance, etc if they so wish.

In conclusion SAS combines a wide range of tools that other software packages cannot offer in one complete unit. With SAS we have the ability to:

1. Produce output datasets in several different types of formats;
2. Produce graphs and charts in JPEG, PNG, and GIF formats quickly;,
3. Deliver simple and advanced statistics through PROC UNIVARIATE;
4. Generate supporting HTML pages;
5. Deliver a reporting system robust enough to handle both major changes in the contents of the environmental datasets and the higher level reporting requirements.

REFERENCES

Mooney, P., Winstanley, A.C. and McDonagh, C. "Data Quality Issues in an Environmental Data Management System". *GIS-Research UK Conference (GISRUK)*, Glasgow, Scotland, April 2005.

W3C, The XML Markup Language, Official W3C Recommendation", http://www.w3.org/TR/REC-xml/ 2004

SAS Institute Inc, "PROC UNIVARIATE", *The SAS Procedures Guide*, Cary, NC, USA. 1999.

Kaner, C. and Bond, W.P. "Software Engineering Metrics: What Do They Measure and How Do We Know?" *10th International Software Metrics Symposium*, Chicago, IL, USA, Sept 2004

ACKNOWLEDGMENTS

This work is carried out under Ireland's National Development Plan's ERTDI Programme 2002 – 2006 as part of a Postdoctoral Research Fellowship between the Irish Environmental Protection Agency and the National University of Ireland Maynooth (NUIM)

RECOMMENDED READING

The macro described here will be available in SAS program file format on our website from the time of the conference. The URL is http://coe.epa.ie/SAScode/

CONTACT INFORMATION

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

Peter Mooney
Environmental Protection Agency
Richview, Clonskeagh,
Dublin 14. Ireland
Work Phone: 353 1 268 0100
Fax: 353 1 268 0199
E-mail: p.mooney@epa.ie **Web:** http://coe.epa.ie