

25th ANNUAL NATIONAL CONFERENCE ON MANAGING ENVIRONMENTAL QUALITY SYSTEMS

APRIL 24-27, 2006

Marriott Renaissance, Austin, Texas

Technical Papers

Recent Applications of SAS in Health and Environmental Analyses

- D.Mintz, Air Explorer - Powerful Web-Driven Tools - 3:00 PM
- S.Fleming, Quality Improvement through the Use of Compiled SAS Macros for Environmental Risk Assessment - 3:30 PM
- J.Stultz, Strategies for Statistical Analysis with SAS System 9 - 4:00 PM

TECHNICAL SESSION: Recent Applications of SAS in Health and Environmental Analyses

AirExplorer – Powerful Web-driven Analytical Tools

David Mintz, US EPA/Air Quality Analysis Group, RTP, NC

Last year EPA launched AirExplorer (<http://www.epa.gov/airexplorer>), a collection of web-based visualization tools for air quality analysts. The tools enable analysts to explore air quality data using interactive maps and graphics. I will demo some of the tools and discuss the underlying SAS® technology and the efficiencies gained from using these tools.

Identifying the Need for Analytical Tools

EPA's Air Quality Analysis Group has long been the agency's source of statistical support on air quality issues. We are charged with supporting the development of national air quality standards as well as tracking progress towards meeting those standards. To that end, we routinely communicate the latest findings regarding the nation's air quality via reports and updates available on-line at <http://www.epa.gov/airtrends>.

In the process of analyzing air quality data, we typically start with some basic questions. Of course, there are always more complex questions that arise, but we typically start with some basic questions, like the following:

- Did air quality improve or worsen? Where?
- Were pollutant concentrations high all year long or were there just a few high days that influenced an annual average?
- To what extent did meteorology play a role?
- How many unhealthy days did my city have last year? How does that compare with previous years? How does that compare with other areas?

For years now, we have answered these questions by retrieving the raw data, processing the data, and then figuring out the best way to summarize the data to answer the questions. Depending on the question, this process could take days or weeks. It often required updating our SAS code to accommodate another year of data or a different parameter. This process was somewhat efficient, but we thought it could be improved.

We started asking ourselves, "What if we could automate some of the basic information? What if we could link our summary charts directly to the database so the information is dynamically updated when the data are updated?" We thought it would be a good idea to develop such a system that could help us quickly answer some of the basic, routine questions. We wanted to create an environment that helped us spend less time generating

information and more time analyzing and interpreting the results. The intent was not to automate everything we do, but rather to automate the summaries that answer the more routine questions. In addition, we thought it would be great to extend this capability to other analysts who are interested in the same issues and have many of the same questions, generally from a more local perspective for their own regional, state, local or tribal area.

Designing and Building the Tools

Our primary goal was to create a suite of interactive, dynamic tools that transform data into relevant information. In addition, we wanted an application that required only a Web browser. We wanted to be able to use our existing SAS code. We wanted users to be able to save the results into a presentation or spreadsheet. We also wanted the ability to add or modify tools to meet our changing needs.

SAS/Intrnet® software allowed us to accomplish these goals. Our existing SAS code required relatively little modification to implement in the Web environment. Most importantly, we wanted to make the powerful features of SAS available to the users of these tools.

Initially, we developed seven tools. Two generate data maps. Two generate data tables. Three generate time series plots. Users can specify a number of parameters, including geographic regions, time periods, and pollutants. The tools query the database and generate results dynamically based on the user-specified parameters. Some of the results are interactive, providing the ability to zoom, rotate, and pan. This ability is especially useful for maps.

Reaping the Benefits

In February 2005, we launched AirExplorer, a collection of web-based visualization tools linked directly to air quality data. Recently, our group sat down and, in 30 minutes, answered many of the previously mentioned basic questions using AirExplorer. A year ago, this same exercise might have taken us several days or weeks. Here are a few other examples of how AirExplorer has been used to support major activities:

- Provide baseline air quality information for comparison with post-Katrina air quality results
- Compare ozone levels in 2005 with levels in previous years
- Inform EPA's PM2.5 designation process
- Examine air quality in areas affected by fires and other episodic events
- Provide air quality information to USA Today
- Answer ad-hoc requests and provide context for many other analyses

Not only is this system helping *us*, it is helping analysts outside our office and outside EPA. The Web site has been accessed over 10,000 times since it was launched just over

a year ago. We have gotten positive feedback from many State and local analysts. Here are a couple of examples.

- “This web-based application is exactly the type of tool Wisconsin is looking for in order to integrate environmental data with our health outcomes.” -*Marni Bekkedal, Ph.D. Wisconsin Bureau of Environmental Health*
- “My initial impression is that this is going to be a fantastic resource! Right now, we're working on a program to identify sources of small particles in our county, and the access to speciated PM2.5 data was fantastic!” -*Scott Johnson, Ventura County APCD*

Summary

Now, we have the ability to generate graphics and detailed data reports on demand and discuss the results immediately. And we have extended these tools to other analysts. There is a huge savings in time and resources that were previously required to generate this basic information. Consequently, more time can be spent assessing, interpreting, and communicating the results.

In the coming year, we plan to develop additional tools that help examine multi-pollutant issues. We also plan to incorporate meteorology and emissions information in order to gain a better understanding of “why” and “how” air quality is changing. In addition, we are continuing to communicate the site’s availability through various forums, including national conferences and workshops largely attended by air quality analysts.

Disclaimer

This paper represents the views of the author and not necessarily those of the U.S. Environmental Protection Agency. Mention of trade names or commercial products does not constitute endorsement or recommendation for use.

Acknowledgements

I would like to acknowledge my group (EPA’s Air Quality Analysis Group) for development and design ideas, Michael Celii and Ahsan Ullah of SAS Institute for enhancements during our pilot project, Kristen Bremer and Jeff Levy for Web design guidance, Chap Gleason and Kamau Njuguna for technical advice, and Tim Richards, Timothy O’Boyle, and Justin LaRose for application deployment and support.

Quality Improvement through the use of Stored, Compiled SAS[®] Macros for Environmental Risk Assessment

*Steve Fleming, Teresa Shaw, and Sandy Donlon
URS Corporation, PO Box 201088, Austin, TX 78720-1088
E-mail: steve_fleming@urscorp.com*

1. Abstract

Stored, compiled SAS[®] macros have improved the quality of our human and ecological risk assessment work based on environmental data from a former air base with a multitude of sites. We began with a series of screening programs written in SAS[®] to identify constituents of potential concern for the risk assessment, calculate summary statistics, and perform a screening level assessment of the cumulative human health risk associated with various constituents in a specific matrix (e.g., soils, sediment, groundwater, or surface water) for a given pathway (e.g., ingestion, inhalation, or dermal). As the code was applied to new sites we found ourselves maintaining a version of the screening for each site. Maintenance of this code became a very complex task as comments requiring changes were received from our clients and regulators over a period of years. Although each site had its quirks that had to be addressed separately, we were able to identify portions of code that could be converted to stored, compiled SAS[®] macros for use across all sites. The advantages of this approach are that:

- Required changes (e.g. errors found in code, new project rules, further modularization of the code) can be made in one place,*
- Those changes are applied consistently across all sites,*
- Thorough internal review and testing of macros becomes more feasible and less time-consuming, and*
- Document control is more manageable with a small set of core programs insuring everyone is using the correct version of code.*

The disadvantages of stored, compiled SAS[®] macros are:

- An increase in the difficulty of troubleshooting errors within macros, and*
- All users need to quit SAS[®] before updates to macros can be made.*

2. Introduction

SAS[®] (SAS Institute) allows users to define macros that can be called to perform specific tasks. For a beginner's guide to SAS[®] macro programming see Burlew. Carpenter describes and discusses the merits of various ways of creating a macro library.

3. Project Description

For the past couple of years we have been responsible for statistical support of human and ecological risk assessments of a former Air Force Base in Illinois. The areas of concern encompassing over 130 sites grouped into about 40 report groups have yielded about 500,000 analytical results thus far. For each site an assessment of the carcinogenic

and non-carcinogenic risks associated with the constituents found at the site are estimated using input from soil, groundwater, surface water, and sediment results. The effect on various receptors (resident, industrial worker, etc.) is evaluated for the ingestion, inhalation, and dermal pathways.

To meet the needs of the project risk assessors a set of programs written in SAS[®] was initially maintained for each report group. Modifications to the code were added as needed for each report group. The myriad quirks of each site prevented the consolidation of the SAS[®] programs into one set for all report groups.

4. Issues

The maintenance of a separate set of SAS[®] programs for each report group led to frequent code duplication. As each report group evolved, inconsistent results became possible across report groups. We feared hearing “I though you fixed that already” from our client.

Reporting is not a one-time process. As comments from internal, client, and regulator reviews came in, the SAS[®] code had to be adapted. The noted code duplication made this a difficult task to keep straight, leading to inconsistent quality in our reports.

5. Solution

Caught between the need to maintain separate code to handle the intricacies of each report group and the need to provide more consistent quality in our results, we chose to identify areas of the programs that matched across report groups for conversion to stored, compiled SAS[®] macros. This allowed each report group to call the same macro to perform certain functions. An example of the code for a stored, compiled SAS[®] macro is shown in Appendix A. The key to making the macro stored and compiled are the `mstored` and `sasmstore=` options, the `libref` pointing to the folder where the compiled version will be stored, and the `store` option at the end of the `%macro` definition.

We have found it best practice to name the source macro file with the same name as the macro and to save the macro source file in the same folder as the stored, compiled version. As each macro is compiled, it is added to a catalog of stored macros called `sasmacr.sas7bcacat`. A SAS[®] session is pointed to the macro catalog by the `autoexec.sas` code in Exhibit 4.1 that is executed any time SAS[®] is opened.

Exhibit 4.1. Autoexec.sas code that points to stored, compiled macros

```
options sasmstore=macros mstored;

/* List locations where cataloged macros may be found, SAS looks for */
/* the called macro in the order listed */
libname macros ('K:/project/macro_source'
               'K:/macro_source'
/* Specifying read only allows multiple people to access macros at */
/* the same time */
               ) access=readonly;
```

6. Advantages

Required changes to the code can be made in one place and apply to all report groups simultaneously. This leads to more consistent quality among the reported results. It is quite satisfying to make the change in one place and be able to assure the risk assessor that the change will be in affect from here forward.

Decreased code duplication allows for easier review and testing of code for internal documentation of quality assurance. We now store electronic copies of our QC reviews with the macros.

Document control is more manageable with a small set of core programs insuring everyone is using the correct version of code. Statisticians new to the project can be trained more easily.

7. Disadvantages

Debugging within macros is more complex than in open code as the macro becomes a “blackbox.” Determining why and where an error occurs is more difficult. Two approaches help in this regard. One is to add a testing parameter to the macro. When this switch is turned on, the macro is directed to create output that helps in debugging. The other option is to pull the macro into the open code, debug, and correct the stored version when the error has been identified.

Everyone has to be out of SAS[®] in order to update the stored, compiled version of a macro. Because SAS[®] reads in the stored version on macro execution, an update to that version is not allowed until all holds on the macro catalog have been released. This can be quite annoying when only a small change is needed. The approach we have tried to alleviate this disadvantage is to perform all updates to macros at the end of the day. In the mean time, each statistician can maintain his or her own test version of the changed macro to complete any work that is required that day.

8. Conclusions

The use of stored, compiled SAS[®] macros for risk assessment work has greatly improved the quality of our reports. Any drawbacks of this approach are far outweighed by tangible quality improvements.

9. References

Burlew, Michele M. 1998, SAS[®] Macro Programming Made Easy, Cary, NC: SAS Institute, Inc., 280pp.

SAS Institute Inc., Cary, NC, USA, www.sas.com.

Appendix A: Sample source code for stored, compiled SAS® macro

```
/* ***** */
/* Program:      mean_merge_tests.SAS */
/* Date:        15Dec2005 */
/* By:          Steve Fleming */
/* Purpose:     Merges Wilcoxon with other 2 sample group */
/*              comparisons and selects the appropriate */
/*              comparison based on distribution */
/* ***** */

/* The following 2 statements point to the macro catalog */
options mstored sasmsstore=macros;
libname macros 'K:/project/macros_source';

%macro mean_merge_tests(
  /* list of parameters for macro */
  in_wilcox, /* Name of dataset with the Wilcoxon test results */
  in_ttest,  /* Name of dataset with the other test results */
  in_by,     /* Space delimited list of variables */
  out_alltests, /* Name of the dataset to hold the results */
  testing = N /* testing switch */
/* store option tells SAS to save a compiled version */
)/store des="Mean Merge Tests";

/* puts macro name in title of output */
title "&sysmacroname. macro";

/* Audit input parameters */
%if %length(&in_wilcox.) = 0 %then %do;
  %put ***** No Wilcox input dataset found by &sysmacroname. *****;
  %return;
%end;

%if %length(&in_ttest.) = 0 %then %do;
  %put ***** No ttest input dataset found by &sysmacroname. *****;
  %return;
%end;

%if %length(&in_by.) = 0 %then %do;
  %put ***** No by variables found in &sysmacroname. *****;
  %return;
%end;

%if %length(&out_alltests.) = 0 %then %do;
  %put ***** No output dataset found in &sysmacroname. *****;
  %return;
%end;

proc sort data=&in_wilcox.;
  by &in_by.;
proc sort data=&in_ttest.;
  by &in_by.;
run;
```

```

/*-----*/
/* Merge ttest, wilcoxon, and distributional test results */
/*-----*/
data &out_alltests.;
  merge &in_wilcox. &in_ttest.;
  by &in_by.;

  length xsign $15;

/* The following logic was inconsistent across report groups prior */
/* to creation of macro. */
/* If less than 4 samples in either group, no means comparison */
/* if no detects in either group, no means comparison */
  if (s_n < 4 or b_n < 4) or (s_det = 0 and b_det = 0) then do;
    testtype='None';
/* example of using testing logic */
    %if %upcase(&testing) = Y %then %do;
      %put "Not enough samples/detects" &in_by. s_n b_n
          s_det b_det;
    %end;
  end;

  if testtype = 'T-test' xsign = tsign;
  else if testtype = 'Wilcox' xsign = wsign;

/* The following was a new rule applied after macro creation */
/* footnote conclusion if one of the groups is all non-detect */
  if compress(xsign) in ("S","NS")
  and (s_det = 0 or b_det = 0)
  and not (s_det = 0 and b_det = 0)
  then xsign = trim(xsign)||"***";
run;

title;

%mend mean_merge_tests;

/* Check that macro was cataloged correctly */
proc catalog catalog=macros.sasmacr;
  contents;
run;

```

Strategies for Statistical Analysis with SAS System 9

John Stultz

SAS System 9 extends the reach of statistical analysis to more people while simplifying the management of the underlying infrastructure. This demonstration will explore the relationships in a plant species data set from the various end-user environments of SAS System 9. These environments include SAS Insight, SAS Analyst, SAS Enterprise Guide, SAS Add-In for Microsoft Office, SAS Web Report Studio and SAS Information Delivery Portal. Statistical analysis with SAS System 9 takes the familiar SAS analytical interfaces to new levels of usability. SAS has traditionally offered a number of interfaces for analytics such as SAS data step code, SAS Insight, SAS Analyst and SAS Enterprise Guide. SAS System 9 supports these familiar interfaces while moving to the next level, delivering functionality founded on a modern Java framework with new "user-centric" designed interfaces that span infrastructure management; the extraction, transformation and loading (ETL) of data along with subsequent data quality; the creation and management of cubes and data repositories; and the deployment of end-user tools appropriate to the skill level of specific users to access information through a wide array of analytic tools designed to meet specific and specialized requirements.
