CBCS
http://agrobiol.sggw.waw.pl/cbcs

Communications in Biometry and Crop Science
Vol. 6, No. 1, 2011, pp. 1–3

International Journal of the Faculty of Agriculture and Biology,
Warsaw University of Life Sciences, Poland

SOFTWARE TRICKS AND TIPS

SAS Functions

My previous contributions (van Santen, 2008; 2009a,b; 2010a,b) introduced various time saving tools and how it might be employed in data analysis. Now I will describe useful functions I use. SAS offers a myriad of built-in functions and call routines, listed in the online documentation (http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a000245852.htm). The online documentation not only provides the basic syntax but also offers some timesaving tips. I will discuss examples from several function groups and trust the reader will use this as a starting point.

ARITHMETIC FUNCTIONS

Arithmetic functions may be used to generate a new variable from an original variable on a row-by-row basis, e.g., a log-transformation. They may also be used to extract the block number from a plot number. Readers who use spreadsheets on a regular basis to perform the task below are warned that the LOG function in EXCEL refers to base 10, whereas the LN function refers to base e log.

```
DATA Data_transformed;
   SET Data_raw;
   LogE_response      = LOG(response); *natural log transformation;
   LogTEN_response  = LOG10(response); *base10 log transformation;
   Block                = INT(plot/100); *block encoded as the first digit of a 3-digit plot number;
RUN;
```

Some arithmetic functions are very useful to create a new variable from a series of original variables in a row-by-row fashion. A plant scientist might sample five plants per plot for a particular trait such as tiller number with plot being the experimental unit. For ease of data recording the datasheet would have the plot number in the first column and contain the tiller count in the five adjacent columns.

```
DATA Data_means;
   SET Data_raw;
   Plot_mean   = MEAN(Plant1, Plant2, Plant3, Plant4, Plant5);
   Plot_mean   = MEAN (of Plant1- Plant5); *equivalent shorthand method;
   Plant_count = COUNT (of Plant1- Plant5); *counts the number of non-missing plants per plot;
RUN;
```

CHARACTER OR STRING FUNCTIONS

String functions are very useful to extract specific information from longer string either to create a new variable or to be used in a conditional statement. PROC MIXED, e.g., provides several fit statistics in its Fit Statistics table but we are interested in using the AICC to compare alternative covariance models. The SUBSTR function can help us to extract just the information we want. Notice that we simply used the SUBSTR function in a conditional argument to limit the printed output to the information desired. What is required for this print statement to work is that ODS OUTPUT is used to create the Fit dataset and we know that DESCR is the variable name SAS assigned to the description of the fit statistics.

```
PROC PRINT DATA = Fit;
  WHERE SUBSTR(DESCR,1,4) =”AICC”; * IF will NOT work in this situation;
RUN;
```

We may also wish to simply look at a particular level of interaction when doing a Mixed Models analysis of several response variables on a linearized dataset as described in van Santen (2009b), where ODS OUTPUT is used to create the dataset ANOVA from the output of significance tests. We can use the COUNT function to count the number of occurrence of the “*” in each line of the ANOVA table.

```
PROC PRINT DATA = ANOVA;
  WHERE COUNT(EFFECT,”*”) = 1; * this would be a two-way interaction term;
RUN;
```

Differences in capitalization may be an issue in collaborative research efforts or if different people enter data. Knowing that capitalization affects sorting behavior and string recognition in conditional statements leads us to string functions that change the capitalization of strings. I use it as a standard operating procedure to avoid potential problems with strings.

```
DATA Data_New;
  SET Data_raw;
  Trt_name  = UPCASE(Trt_name); *Trt_name now in UPPER CASE;
  Trt_name  = LOWCASE(Trt_name); *Trt_name now in lower case;
RUN;
```

We may also wish to combine (concatenate) several strings into a new string variable, as might be the case when performing discriminant analysis, where the original dataset contains several class variables. SAS offers various CAT functions or CALL CAT routines for this purpose. The various routines differ in the manner they treat internal spaces or trailing blanks.

```
DATA Data_CDA;
  SET Data_raw;
  Separator = ‘-‘; *defines a string that separates Old_Class variables in New_Class;
  New_Class=  CATS(Old_Class1,’_’,Old_class2); *removes leading and trailing blanks ;
  New_Class=  CATX(Separator, Old_Class1,Old_class2); *result same as previous;
  Drop Separator; *unneeded variable dropped from new dataset;
RUN;
```

DATE FUNCTIONS

Anyone who has used dates in spreadsheets and SAS has experienced problems; problems even occur when using the same spreadsheet software on different operating systems accessing the same file. My advice to clients is not to be heroes but record dates with year, month, and day in separate columns and use a SAS function to create the proper SAS serial date, displayed in a format of the client liking. Not only will the dates be correct and thus sort correctly but it also makes spreadsheet data entry easier.

```
DATA Data_Date;
  SET Data_raw;
  Format Datum date9.; *Variable Datum will be displayed in 01JAN1900 format;
  Datum= MDY(Month,Day,Year); *entire Trt_name in upper case;
  Drop Month Day Year; *unneeded variables dropped from new dataset;
RUN;
```

The online documentation referenced above may serve as a starting point. I have also found the UCLA: Academic Technology Services, Statistical Consulting Group website very useful (http://www.ats.ucla.edu/stat/sas/modules/).

References

van Santen E. (2008). Make a project folder home base for SAS. *Communications in Biometry and Crop Science Crop Science* 3 (1), 1–2.

van Santen E. (2009a). SAS Macro Variables. *Communications in Biometry and Crop Science Crop Science* 4 (1), 1–2.
van Santen E. (2009b). SAS Macro Variables and ARRAY Processing. *Communications in Biometry and Crop Science Crop Science* 4 (2), 40–41.
van Santen E. (2010a). Data checking with SAS PROC TABULATE. *Communications in Biometry and Crop Science Crop Science* 5 (1), 1–3.
van Santen E. (2010b). PROC SQL vs. DATA Step or Anything you can do, I can do better. *Communications in Biometry and Crop Science Crop Science* 5 (2), 66–68.

contributed by Edzard van Santen

Forage Breeding and Genetics, Dept. of Agronomy and Soils,
Auburn University, AL 36849-5412.
E-mail: vanedza@auburn.edu

Published online: 5 April 2011