ABSTRACT


Printed output from SAS PROCS might sometimes be more useful in reports if additional information could be added to the standard output. Conversely, information not included in output data sets may be useful for further work. Heretofore, the information had to be typed on the output or copied and repunched for subsequent processing. The new features, PROC PRINTTO and substring handling, allow considerable capability in modifying and using SAS printout. Portions of output may also be used an input for further processing where there is now no output capability. Simple examples show the augmentation of PROC FREQ and PROC ANOVA and the retrieval of information from PROC GLM.

The solution to a user's needs may require as much as the writing of a complete PROC or may be limited to a fairly simple calculation involving sums of squares that already appear in a PROC. Many users have written new PROCs such as the ones presented at this meeting but the work involved in actually changing an existing PROC is more than most of us want to undertake, especially if the changes are relatively minor. PROC MATRIX has offered another solution to those who wish to enhance SAS. PROC PRINTTO has offered another solution to those who wish to enhance SAS. It allows considerable capability in modifying and using SAS printout. Portions of output may also be used as input for further processing where there is now no output capability. Simple examples show the augmentation of PROC FREQ and PROC ANOVA and the retrieval of information from PROC GLM.

Two of the examples I will show augment existing PROCs; The third illustrates outputting data from GLM.

PROC PRINTTO functions to direct SAS page output to some device other than the printer. For example, the following code causes PROC ANOVA output to be stored on disk.

```
PRINTTO UNIT = 20;
ANNOVA; CLASSES DOSE; MODEL RCOUNT = DOSE;
PROC PRINTTO;
```

The second PROC PRINTTO sends the output back to the page printer. The ANOVA output may now be read from the disk as 133 byte records and treated as ordinary data.

In the first case I want to produce a frequency table of wind speed by wind direction. A separate table will be produced for each value of stability, a variable related to temperature. PROC FREQ produces the table nicely, (Figure 1) but I want to add two lines to the bottom of the table to give the number of cases of no wind (calm) or missing, (Figure 2). The first two pages of the program are not illustrated. They input the data, assign class values to speed, directions, and stability, and produce a dummy data set of one observation for each speed and stability. This set is used to insure that each table includes all speed classes. Figure 3 and Figure 4 show the rest of the code. Once the printout has been routed to disk by PROC PRINTTO, the tables are produced and the printer is enabled again. The tables are then read as data in lines 98 through 123. Several operations take place: the "by" variable, stability, is found and retained; the table heading is changed from numeric values to more descriptive values; and the dummy line is eliminated. Finally lines 124 to 132 print the table and add the additional information.

The second example shows the modification of PROC ANOVA to produce a Kruskal-Wallis test. (This is already an option in PROC ANOVA.) To calculate the Kruskal-Wallis H we first rank the data and then apply PROC ANOVA to the ranks (Figures 5 and 6). Sall, 1977, shows the arithmetic necessary to convert the treatment sum of squares to the H statistic. The conversion is

\[ H = \frac{12}{N+1} \left( \frac{N^2}{SST} \right) \]
Once the ANOVA is copied to disk, it is re-read and the treatment sums of squares (SST), df, and N are captured and retained. The printout is also modified to appear on one page. The probability associated with H is approximately chi-square and is computed using the chi-square function. Figure 7 shows the data set retrieved from disk and Figure 8 shows the completed table.

The last example involves using the error sums of squares (ESS) and df from GLM. Once the ESS and df are read from disk they are merged with a dummy data set which contains the "by" variables for each ESS and df, (Figure 9, lines 93-98). This data set was produced in lines 73 through 76. Substring operations could have been used to find the "by" variables but the code would have been much longer.

In conclusion, these simple examples show several operations available to us now.

1) On a limited scale SAS may be augmented within the existing framework of its PROCs.
2) Information may be added to printout that could not appear in a title line including information related to variables being processed.
3) Title lines for PROC FREQ may be written in a more descriptive manner than is now generally possible and with the columns in a desired order (see also SAS Communications III (2) for another method).
4) Outputting of variables may be used more often than current options allow.

REFERENCES


Figure 1. Standard output of PROC FREQ.

Figure 2. Output of PROC FREQ modified and augmented. The column headings have been rewritten and two lines have been added below the table.
62 PROC SORT; BY STAB;
63 PROC MEANS NOPRINT; VAR MISS NO_CALM; * FIND THE NUMBER OF MISSING AND
64 OUTPUT OUT=X N=MISSING NO_CALM; * CALM FOR EACH STABILITY.
65 BY STAB;
66 DATA D; SET D Dummy;
67 PROC SORT; BY STAB;
68 PROC FREQ DATA=D; TABLES WDR*WSPD/ NOROW NOCOL NOPERCENT MISSING; BY STAB;
69 DATA D; SET D;
70 IF MISS=.: * ELIMINATE UNUSABLE DATA.
71 TITLE SURRY UPPER LEVEL WIND FREQUENCY DISTRIBUTION;
72 PROC PRINTTO UNIT=2U; * ROUTE TO DISK.
73 PROC FREQ DATA=D;
74 TABLES WDR*WSPD/ NOROW NOCOL NOPERCENT;
75 BY STAB;
76 PROC PRINTTO ; * RETURN TO PRINTER.
77 DATA ONE; INFILE FT20FDOI ; INPUT IMAGE 1-132;
78 *
79 MISSING=. CALM=.
80 * SET FOR THE INTERLEAVING.
81 FILE PRINT;
82 IF FIRST.STAB THEN PUT _PAGE_ / 9 63 'STAB=' STAB;
83 IF LAST.STAB = 0 AND MISSING=. THEN PUT & 21 IMAGE; * PRINT THE PROC FREQ
84 * TABLE.
85 IF LAST.STAB THEN PUT / &Z1 MISSING=5.0 / &Z1 NO_CALM=5.0;
86 * PRINT THE TWO EXTRA
87 * LINES.

Figure 3. Please see explanation below.

101 IF IMAGE =: 'STAB' THEN GO TO A; * FIND THE 'BY' LINE.
102 IF IMAGE =: 'FREQ' THEN GO TO B; * FIND THE TABLE HEADING.
103 IF IMAGE LT: '0', * ELIMINATE THE OLD TITLES.
104 RETAIN STAB;
105 OUTPUT;
106 RETURN;
107 A:
108 STAB = ' ';
109 N6 = 6; N1 = 11;
110 CALL SUBSTR (STAB,IMAGE,N6,N1); * CAPTURE THE BY GROUP.
111 RETURN; B:
112 IMAGE=
113 'FREQUENCY CALM 0.75-2.5 2.6-3.5 3.6-7.5 7.6-12.5 12.6- 18.6- GT 2
114 4.5 TOTAL'; * REWRITE THE COLUMN HEADINGS. THIS
115 PERMITS THE COLUMNS TO BE PRINTED
116 IN THE DESIRED ORDER.
117 OUTPUT;
118 IMAGE=
119 18.5 24.5';
120 OUTPUT;
121 INPUT; INPUT;
122 TITLE2 TABLE OF WDR BY WSPD;
123 DATA _NULL_; SET ONE X ; BY STAB; * INTERLEAVE THE TWO
124 FILES.
125 FILE PRINT;
126 IF FIRST.STAB THEN PUT _PAGE_ / & 21 'STAB=' STAB;
127 IF LAST.STAB = 0 AND MISSING=. THEN PUT & 21 IMAGE; * PRINT THE PROC FREQ
128 * TABLE.
129 IF LAST.STAB THEN PUT / &Z1 MISSING=5.0 / &Z1 NO_CALM=5.0;
130 * PRINT THE TWO EXTRA
131 * LINES.

Figure 4. Code for producing modified output as shown in Figure 2.
SIX DOGS WERE ASSIGNED TO EACH OF SIX DOSES OF TYPHOID VACCINE.

THIRTY MINUTES AFTER INJECTION THE COUNT OF LEUKOCYTES PER CUBIC MM OF CAPILLARY BLOOD WAS DETERMINED. TEST IF THERE WAS ANY EFFECT ON THE LEUKOCYTE COUNT DUE TO THE VARIOUS DOSES OF THE VACCINE.

DATA FROM SAS SAMPLE DATA SET (ANOVA4). SEE ALSO SAS VIEWS PART II, P. 15

DATA TYPHOID;
INPUT DOSE COUNT;
CARNS;
.25 1950 .25 5350 .25 4475 .25 3700 .25 4550 .25 3700
.50 3800 .50 3750 .50 9550 .50 7700 .50 8565
1.0 4025 1.0 9250 1.0 3725 1.0 6550 1.0 3275 1.0 5350
2.0 5250 2.0 6625 2.0 3325 2.0 6725 2.0 1375 2.0 4630
4.0 7400 4.0 4450 4.0 4545 4.0 4375
8.0 7500 8.0 4500 8.0 5000 8.0 6250 8.0 9650 8.0 7565
20
PROC PRINT;
21 PROC RANK : VAR COUNT: RANKS RCOUNT;
22 TITLE KRUSKAL-WALLIS ONE-WAY NONPARAMETRIC ANOVA;
23 PROC RANK UNIT=20: * ROUTE PRINTOUT TO DISK;
24 PROC ANOVA: CLASSES DOSE; MODEL RCOUNT =DOSE;
25 PROC PRINTTO ; * RETURN TO PRINTER;
26 DATA A: INFILE FT20F001;
27 INPUT a2 IMAGE $CHAR132. IMAGE2 $ 1-70 a:
28 * WITH BLANKS. IMAGE2 ELIMIN-
29 *ATES LEADING BLANKS;
30 IF IMAGE2 GE: 'O' THEN GO TO C;
31 * AND COMPRESS TO ONE PAGE *;
32 IF IMAGE2 =: 'ANAL' THEN GO TO C;
33 IF IMAGE2 =: 'CORR' THEN GO TO A;
34 IF IMAGE2 =: 'DOSE' THEN GO TO B;
35 F: INPUT; OUTPUT;
36 RETURN;
37 A: INPUT 31 DUM1 & DUM2 & DF;
38 N=DF+1;
39 RETAIN N;
40 OUTPUT;
41 RETURN;
42 B: COUNT + 1; IF COUNT=1 THEN GO TO F;
43 INPUT 31 DUM1 & DF & SS;
44 RETAIN DF;
45 H=SS*12/(N*(N+1));
46 RETAIN H;
47 OUTPUT; RETURN;
48 INPUT;
49 C: INPUT;
50 DATA _NULL_; SET;

Figure 5. Please see explanation below.

FILE PRINT;
52 PUT IMAGE @CHAR132.;
53 IF IMAGE2 =: 'DOSE' THEN RETURN;
54 IF COUNT =: 1 THEN RETURN;
55 **** PRINT KRUSKAL-WALLIS LINES ****;
56 PUT / 356 'PR>CHI-SQUARE' / 327 'DF' 343 'H' @ 56'(APPROXIMATELY)'/;
57 PUT ':KRUSKAL-WALLIS' H 324 DF 4.0 @ 38 H 9.4 @;
58 PROB=1-PROBCHI(H,DF);
59 PUT @ 50 PROB 4.3;
Figure 7. Output of PROC ANOVA as read from disk. Note that carriage control characters appear.

Figure 8. Output of PROC ANOVA modified to include Kruskal-Wallis H.
IF FIRST.SP = 0 THEN GO TO A; CSW=0; CSL=0; CP=0; DFF=0; A:
56 RETAIN CSW CP CSL DFF;
57 OUTPUT ABLEI (KEEP = LIST SP DF CSSMT CSSLEN CCP TYPE);
58 CSSW=CSSW+CSSMT; CSL=CSL+CSSLEN; CP=CP+CCP; DFF= DFF+DF;
59 IF LAST.SP = 0 THEN RETURN;
60 TYPE = 'TSS';
61 CSSMT = CSSW; CSSLEN=CSL; CCP = CP; DF=DF;
62 OUTPUT ABLEI (KEEP = LIST SP DF CSSMT CSSLEN CCP TYPE);
63 TYPE = 'REGRESSION SS';
64 CSSWT = CCPMCCP/CCSL; CCP = CCP; DF=DF;
65 OUTPUT ABLEI (KEEP = LIST SP DF CSSMT CSSLEN CCP TYPE);
66 TYPE = 'ERROR SS';
67 CSSMT=CSSW-CSSHT; DF=DF-1;
68 OUTPUT ABLEI (KEEP = LIST SP DF CSSMT CSSLEN CCP TYPE);
69 ESS_ALL = CSSWT; DF_ALL= DF;
70 OUTPUT SIX (KEEP = LIST SP DF_ALL ESS_ALL);
71 PROC PRINT;
72 TITLE= SEPARATE REGRESSIONS THRU A COMMON SLOPE;
73 DATA DESCRIE ; SET A; BY LIST SP STATON;
74 IF FIRST.YEAR + FIRST.MONTH + FIRST.SP + FIRST.STATON GE 1;
75 KEEP PLANT;
76 KEEP YEAR MONTH SP STATON;
77 DATA NAMES; SET DESCRIE ; BY LIST SP;
78 IF FIRST.YEAR + FIRST.MONTH + FIRST.SP GE 1;
79 DATA BLANK; SET;
80 IN TABLE;
81 KEEP ORDER;
82 ORDER = 1.5; OUTPUT;
83 ORDER = 2.5; OUTPUT;
84 ORDER = 3.5; OUTPUT;
85 STOP;
86 PROC GLM DATA=A; MODEL LOGWT = LOGLEN; BY LIST SP STATON;
87 PROC PRTNT TO NEW UNIT=20;
88 PROC GLM DATA=A; MODEL LOGWT = LOGLEN; BY LIST SP STATON;
89 PROC PRTNT TO ;
90 DATA GLM; INFILE FT20FOO1; INPUT A * 2-6 2;
91 IF A = 'ERROR' THEN GO TO ABLE, INPUT; DELETE; ABLE;
92 INPUT DF 23-28 ESS 35-48 ;
93 DATA A1 ; MERGE DESCRIE GLM;
94 KEEP LIST SP DF ESS TYPE;
95 KEEP STATON;
96 TYPE = 'BY AREA';
97 KEEP ORDER;
98 ORDER = 1;
99 DATA TOTAL ; SET;
100 BY SP LIST STATON;
101 KEEP ORDER;
102 IF FIRST.SP = 0 THEN GO TO TWO;
103 SUMDF = 0; SUMESS= 0;
104 TWO:;
105 RETAIN SUMDF SUMESS;
106 SUMDF + DF; SUMESS + ESS;
107 IF LAST.SP = 0 THEN RETURN;
108 DF_SEP = SUMDF; ESS_SEP = SUMESS; MS_SEP = ESS_SEP/DF_SEP; TYPE='TOTAL';

Figure 9. Code for retrieving error sums of squares and degrees of freedom from PROC GLM output.