ABSTRACT

The macro presented in this paper is intended for end-users who wish to use "PUT" statements to generate reports, but do not want to spend a lot of time counting columns to make sure everything fits nicely within the available space. The macro takes the variables that the user supplies for the report and calculates the BESTFIT within the 132 available columns. Henceforth, macro %BESTFIT was created.

Macro BESTFIT calls several other macros in determining the appropriate column locations of each variable. This paper will discuss each of the sub-macros and how they interrelate to produce a nice formatted report.

INTRODUCTION

Generating reports using PUT statements can be very tedious and time consuming. Often users plan what their reports look like on some kind of spreadsheet that contains 132 columns so as to be sure everything fits within the available space. If a user has a report encompassing many variables, more than one page may be necessary. If more than one page is generated, 10 variables are a good idea so each observation may be easily identified on each page.

Macro %BESTFIT solves all these problems. It calculates the number of pages depending on how many variables are in the report and in what order they are to appear. Moreover, it enables the user to specify which variables are ID variables and on which page they are to be placed.

SUB-MACROS

%BESTFIT consists of several sub-macros that determine the page on which to put the variables, what columns to put them in, and how much extra space is available.

%INITIZE initializes the following macro variables:

<table>
<thead>
<tr>
<th>Macro Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSN</td>
<td>Data Set Name</td>
</tr>
<tr>
<td>SEARCH</td>
<td>Upper bound LINESIZE</td>
</tr>
<tr>
<td>STRTID</td>
<td>10 Variables</td>
</tr>
<tr>
<td>ENDid</td>
<td>Other Variables</td>
</tr>
<tr>
<td>STRT</td>
<td>End of Variables</td>
</tr>
<tr>
<td>END</td>
<td>Labels associated with each SAS variable in VARLIST above.</td>
</tr>
<tr>
<td>VARLIST</td>
<td>List of the SAS variables that are to be in the report</td>
</tr>
<tr>
<td>TITLES</td>
<td>List of the SAS variables from the GRADE data set. Consequently, TITLES contains labels for each variable. Each label spans 2 rows with an asterisk (*) denoting the split much like a split character of PROC PRINT.</td>
</tr>
</tbody>
</table>

Notice that the value of VARLIST contains 5 SAS variables from the GRADE data set. Consequently, TITLES contains labels for each variable. Each label spans 2 rows with an asterisk (*) denoting the split much like a split character of PROC PRINT.

The variables in VARLIST appear in the order in which they will appear in the report. The values of START and ENDI indicate there are two ID variables; in this case, SOCIAL and NAME respectively. STRT and END refers to the remaining variables.

The value of SEARCH informs BESTFIT of the number of columns in which the variables in VARLIST must fit. In this case, 132 columns.
MACRO-BASED REPORT GENERATOR
by ANTHONY B. KREAMER

**SETUP** takes the values of VARLIST and TITLEs and scans them to create macro variables VAR1, VAR2, ..., VARN and TITLE1, TITLE2, ..., TITLEN respectively. In the aforementioned example, VAR1=SOCNUM and TITLE1=SOCIAL SECURITY NUMBER. Thus far the variables in the report have been defined as have the order in which they will appear and their labels.

**HEADINGS** takes the values of TITLE1, TITLE2, ..., TITLEN and scans for the asterisk which serves as a split character. Thus, two labels can be generated for a given variable. For example, HEADINGS takes the value of TITLE1 above and creates macro variables TITLE11 and TITLE12 which have the value of SOCIAL SECURITY and NUMBER, respectively. If no split character is found, then TITLEN.1 is blank and TITLEN.2 has the value.

Below is a layout of what has been defined thus far.

```
SEARCH
1

* TITLE11 TITLE12 : TITLEN.1
* TITLE12 TITLE12 : TITLEN.2
* VAR1 VAR2 : VARN

**MAXTITLE** calculates how much space is required for TITLEN.1 and TITLEN.2 and places that with the most characters in macro variables MAXTITLE.

**MAXVALUE** calculates the maximum space required for SAS variable VARN and places the result in macro variable MAXVALN.

**MAX** compares the length of MAXTITLE and MAXVALN and places the longer in MAXN. MAXN, therefore, determines which takes up more space, the label for a given SAS variable or the value of the variable.

At this point, all the necessary macro variables for BESTFIT are created. Pagination, as well as column locations, are determined by **EXTRA**. Also, if the maximum space that is used on a particular page is less than **SEARCH** (132), **EXTRA** will attempt to use all of **SEARCH** spaces by inserting "extra" spaces between variables. Let us take a closer look at **EXTRA**.

**EXTRA** assumes one page will be generated. If more are needed, the macro increments the macro variable PAGE. The order of **VARLIST** is crucial in determining the order of the variables on the report and in turn what page they appear on. For example, let us assume that the maximum space required to fit all five variables is 150. **EXTRA** will know that **SEARCH** is less than 150 and tell us two pages are required. Here is where the use of 10 variables come into play. Two variables had been previously designated as 10 variables (SOCNUM and NAME). They will appear on both pages so as to easily identified each observation. The remaining three variables will be placed in the appropriate columns of their respective pages.

If the variables SOCNUM, NAME, and GRADE take up a total of 120 spaces and the next variable (COURSE in our case) takes up 20 spaces, **EXTRA** will put COURSE on the next page along with SOCNUM and NAME. Since the first three variables take up 120 spaces and there are **SEARCH** (132) available spaces as defined by **SEARCH**, twelve spaces are left to be dispersed within the columns of the report.

```
3 SPACES 3 SPACES
1 120 132

* TITLE11 TITLE12 TITLE13 12 SPACES*
* TITLE12 TITLE12 TITLE12 LEFT OVER*
* TO BE *
* VAR1 VAR2 VAR3 DISPERSSED* IN REPORT*
* SOCNUM NAME GRADE *

9 Total Spaces
```

**EXTRA** allocated the 12 extra spaces evenly between the columns; i.e., it added 6 extra spaces between columns 1 & 2 and 6 spaces between 2 & 3.
We now know that there are 12 spaces left over. 3 variables on page one of the report which means 2 columns we can add these extra spaces to. This means that we can add an additional 6 spaces to the already 3 spaces that we originally accounted for. This will evenly spaces out the final report by using all the available space.

Once $EXTRA determines how much extra space to add, it will then call macro $STRTEND which calculates the starting and ending columns for every variable. If we know where a variable will start and end, we can calculate the center point between the two. At this time, we already know what the maximum space for the labels and values of a particular variable, so we can now calculate the centers for the titles and values. $STRTEND will create the following macro variables:

- $SAPAGE&II: Start position of variable &II on $PAGE.
- $E&PAGE&II: End position of variable &II on $PAGE.
- $CNTR&PAGE&II: Center location between $SAPAGE&II and $E&PAGE&II.
- $CVAL&PAGE&II: Center location for variable &II. This is calculated based on the variables maximum space. I.e. MAXVAL&II.
- $CTL&PAGE&II.1: Center location of label one for variable &II.
- $CTL&PAGE&II.2: Center location of label two for variable &II.

$BESTFIT now has what it needs to produce the report. It knows the number of pages it needs to generate the report, what variables go on a particular page, the column locations of each variable and of each heading for a given variable. All $BESTFIT does is produce the "PUT" statements for each variable and each heading.

The nice thing about this application is that all a user needs to do to generate a report is change the macro $INITIZE to their specific needs.
MACRO-BASED REPORT GENERATOR
by ANTHONY B. KREMER

MACRO INITLZE;
OPTIONS PRINT LINESIZE=132;
LIBNAME IN 'C:\SAS\SU01';
DM 'LOG'; ZOOM='LOG';
LET DSM=IN.GRAM;
LET SEARCH=132;
LET STRNT=1;
LET ENDT=1;
LET STR=3;
LET ENG=16;
TEST6/TEST7/TEST8/TEST9/TEST10/AVERAGE/FINALGRADE;
LET TITLE=SOCIAL SECURITY NUMBER/STUDENT NAME/GRADE LEVEL/COURSE NAME/
TEST SCORE1/SCORE2/EXAMINATION SCORE/TEST NUMBER 4/
MIDTERM TEST 5/TEST 6/SCORE7/NUMBER 7/TEST 8/TEST 9/
TEST 10/SCORE10/AVERAGE/FINAL GRADE;
SET J=IN;
MACRO INITLZE;

MACRO SETUP;
XDO I=10 TO 100 READ;
SET J=STR10 XTO &ENG;
LET MAX&I=0;
XDO J=1 TO 2;
XLET TITLE&I=SCAN(TITLE&I, &J, *);
XEND;
XELSE XDO;
.XLET TITLE&I=STR&I;
XLET TITLE&I=SCAN(TITLE&I, &J);
XEND;
XEND HEADINGS;

MACRO MAX;
XDO I=1 TO 10 READ;
XIF MAX&I=MAX&I OR MAX&I=MAX&I THEN XLET MAX&I=MAX&I;
LOSE XLET MAX&=MAX&I;
XEND;
XRESULT=VAR/TITLE/MAX&/MAX&/MAX;
XEND MAX;

MACRO MAXTITLE;
XDO I=1 TO 10 READ;
LET MAXTITLE=0;
XDO J=1 TO 2;
XLET TITLE=SCAN(TITLE&I,AJ);
XIF ALEN=GT MAX&I THEN XLET MAXTITLE=MAX&I;
LOSE XLET MAX&=MAX&I;
XEND;
XLET MAXTITLE=MAXTITLE;
XEND;
XRESULT=VAR/TITLE/MAX&/MAX&/MAX;
XEND MAXTIL;

MACRO MAXVALUE;
XDO I=1 TO 10 READ;
LET VALUE(I) I = 6;
DATA TEMP; SET TEMP;
KEEP MAXTITLE LEN;
LEN=LENGTH(TRIM(LEFT(MAXVALUE)));
PROC MEANS NOPRINT;
VAR LEN;
OUTPUT OUT=TEMP MAX=MAXVALUE;
DATA TEMP; SET TEMP;
CALL SYMPUT('MAXVALUE', (TRIM(LEFT(MAXVALUE))));
XEND;
XRESULT=VAR/TITLE/MAX&/MAX&/MAX;
XEND MAXVALUE;
MACRO-BASED REPORT GENERATOR
by ANTHONY B. KREMER

MACRO RESULTS(PAR);
XLET LIST=SPAN;
XDO K=1 XTO 5;
XEND(5)
XLET L=SCAN(LIST,AC,7)
XDO J=1 JTO LEND;
XIF BL=TITLE OR BL=CTITLE OR BL=ETITLE
THEN XDO;
XDO J=1 JTO L;
XPUT XSTR( jHALF = 10010101;
XEND;
Xndo;
ELSE XDO;
XPUT XSTR( )HALF = 10010101;
XEND;
XEND;
END(2)
END;
MACRO EXTRU;
XDO J=1 JTO L;
XPUT XSTR( )HALF = 10010101;
XEND;
XEND;
MACRO LINES(NL);
XDO J=1 XTO ANM;
XPUT XSTR( );
XEND;
XEND LINES;
MACRO EXTRA;
XLET PAGE=1;
XLET J=0;
XLET SPACES=3;
XLET T=1;
XLET SUM=0000;
XIF ENDID EQ ETITLE
THEN XLET TOMORROW=
XDO J=REVAL(ENDID+1) XTO LEND;
XLET J=REVAL(J+1);
XLET TSON=SUM;
XPUT XSTR( )SUM = SUM;
XLET SUM=REVAL(SUM + BLMAX1 + ASPACEs);
XIF J = ENDID
THEN XLET SEMINOR=
XIF (BLSUM GT RESEARCH)
XTHEN XDO;
XLET T=0;
XDO J=1 JTO LEND;
XLET EXTRA=REVAL((RESEARCH - BLSUM) / (61 - J));
XPUT XSTR( )PAGE = PAGE;
XPUT XSTR( )EXTRA = EXTRA;
XPUT XSTR( )T = T;
XEND;
XLET T=
XDO J=1 JTO LEND;
XLET PAGE=REVAL(PAGE + 1);
XLET SUM=REVAL(SUM = BLMAX1 + ASPACEs)
XEND;
XIF J = ENDID
XTHEN XDO;
XLET EXTRA=REVAL((RESEARCH - BLSUM) / (61 - J));
XLET J=REVAL(J+1);
XLET F=+;
XLET T=1 JTEST(XA)
XDO J=1 JTO LEND;
XPUT XSTR( )SUM = SUM;
XPUT XSTR( )PAGE = PAGE;
XPUT XSTR( )EXTRA = EXTRA;
XPUT XSTR( )T = T;
XEND;
XEND;
XELSE XLET T=1 JTEST(XA);
XEND;
XEND LINES;
XEND;
XEND EXTRA;
DATA _NULL_; SET &OSN; FILE PRINT HEADER=H;

PUT &STR( )S&PAGE&K • &&S&PAGE&K &PAGE T" &F " &F;

XDO C=1 INTO E; ;

XLET TT=SCANAT(TT, &AC, /);

XPUT XTRC JC = &C &T = &TT CVAPAGEF = &CVALPAGEF VARITY = &VARITY;

XTRC () &VARITY&TRC )

XDO C=1 INTO E; ;

XLET TT=SCANAT(TT, &AC, /);

XPUT XTRC JC = &C &T = &TT CVAPAGEF = &CVALPAGEF TITLE&TRC = &TITLE&TRC;

XTRC () &TITLE&TRC&TRC )

RETURN;

RUN;

END BESTFIT;

XPUT XTRC )START POSITIONS AND ENDING POSITIONS;

XPUT XTRC )-------------------------------;

XDO C=1 INTO E; ;

XPUT XTRC ) &SPANDEK &PAGEDEK &PAGEDEK = &BASEPAGEDEK;

XLET E(1)

XPUT XTRC ) &SPANDEK &PAGEDEK &PAGEDEK = &BASEPAGEDEK;

XLET E(1)

XPUT XTRC ) &SPANDEK &PAGEDEK &PAGEDEK = &BASEPAGEDEK;

XLET E(1)

XPUT XTRC ) &SPANDEK &PAGEDEK &PAGEDEK = &BASEPAGEDEK;

XDO A=1 INTO E; ;

XPUT XTRC ) &SPANDEK &PAGEDEK &PAGEDEK = &BASEPAGEDEK;

XDO A=1 INTO E; ;

XPUT XTRC ) &SPANDEK &PAGEDEK &PAGEDEK = &BASEPAGEDEK;

XLET E(3)

XDO C=1 INTO E; ;

XPUT XTRC )-------------------------------;

XLET E(3)

XDO C=1 INTO E; ;

XPUT XTRC )-------------------------------;

XLET E(3)

XEND BESTFIT;

XEND STRTEND;

MACRO STRTEND;

XLET SPACE,1=1;

XDO E=1 INTO F;

XLET MAX,SCANAT(X, &KL, /);

XLET L=SCANAT(X, &KL, -1);

XLET E,SCANAT(X, &KL, -1);

XLET E,SCANAT(X, &KL, -1);

XLET CVALPAGEF=SCANAT(X, &KL, -1);

XLET CVAPAGEF=SCANAT(X, &KL, -1);

XLET CVAPAGEF=SCANAT(X, &KL, -1);

XDO A=1 INTO F;

XLET ASPANDEK,SPACE = &SPANDEK = &BASEPAGEDEK;

XLET E(1)

XLET E(1)

XLET E(1)

XDO A=1 INTO F;

XLET E(1)

XLET E(1)

XDO A=1 INTO F;

XLET E(1)

XDO A=1 INTO F;

XLET E(1)

XDO A=1 INTO F;

XLET E(1)

XDO A=1 INTO F;

XLET E(1)

XDO A=1 INTO F;

XLET E(1)

XDO A=1 INTO F;

XLET E(1)

XDO A=1 INTO F;

XLET E(1)