In a computer system that is based primarily on applications written in a high level language (e.g., FORTRAN), the integration of SAS Software may present some problems for both programmers and users. These problems include data accessibility, program and file maintenance, and user interface. This paper will present a FORTRAN program, running under UNIX, that addresses some of these problems. Using both user inputs and data retrieved from specific data files, this program will generate a SAS program to process and plot toxicology data such as animal body weights and food consumption. This FORTRAN program also creates a BCL command procedure that invokes the SRS program. Control is passed to the command procedure at the end of the FORTRAN program. The SRS program uses PROC GPLOT to plot, display, group, delete, and save graphs without any user intervention.

INTRODUCTION

In the Dept. of Preclinical Safety Assessment at Sandoz, toxicology data are stored in direct access files not easily accessible to SAS Software. Application programs written in FORTRAN are being used to access these data files. Users run specific programs to display data in various tabular formats.

Since the installation of SRS and SAS/GRAPH Software, there has been an increasing user demand to view these data in graphic forms. Usually, a SRS program has to be written for each study (i.e., project), with minor changes such as different variables, labels, and BY values. Tables generated by existing FORTRAN programs have to be modified to allow access by SRS Software. Furthermore, different logical assignments have to be made to access different data files.

A method has been developed to replace the above process. It tries to achieve the following goals:

1) It should be easy to use (no knowledge of SRS Software required),
2) It should be efficient,
3) It should be flexible,
4) It should be highly automated.

We will now look at a FORTRAN program called GBSR that was designed to meet these goals.

PROGRAM DESCRIPTION

GBSR plots animal body weight mean values or food consumption mean values over time. Appendix 1 contains a partial listing of the source code. The program may be divided roughly into six logical code segments, which are discussed below. The first sample session in Appendix 5 will serve as an example. (References to this session are enclosed in parentheses.)

Segment 1: At different points in the program, user inputs are necessary to determine what data files to access. The user is first asked to enter a study number that he or she wishes to access. The user then chooses the kind of data to be plotted (body weight) and the time period to be plotted (1-26). These user inputs instruct GBSR where to fetch different pieces of data such as titles, labels, and most importantly, the data to be plotted.

Segment 2: This segment processes the data to be plotted. Mean values are calculated and written to a data file (XXXX.dat, Appendix 2), together with the corresponding sex, test period number, and dose group number. This file will later be accessed by PROC GPLOT in the SAS program. Note that GBSR will delete the existing data file if the user decides to regenerate the data. GBSR also displays the message "Generating SAS data ..." and the test period number being processed.

Segment 3: When data generation is completed, the user is asked to choose a graphic output device for the plot. The user may plot the graph on the Zeta 8 plotter or display it on a Tektronix 4020 graphics terminal. Other device types can be supported when the need arises. GBSR always saves the graph to a graphics catalog, even if the user is running it on an unsupported device. The graph can always be replayed later by running GBSR on a supported device.

Segment 4: The BCL command procedure GBSR.COM is built (Appendix 3). This procedure assigns the correct names for SRS INPUT and BCL. It then invokes GBSR.SAS, which will be created later in Segment 5. The log file created by GBSR.SAS is automatically deleted at the end of the procedure. Note that GBSR will delete the old copy of GBSR.COM if one exists, so that only one version will be kept on the system at any one time.

Segment 5: GBSR now begins to build the SRS program called GBSR.SAS (Appendix 4). Again, the old copy of GBSR.SAS is deleted if one exists. Several options have to be considered when generating this SRS program. First, GBSR has to decide if the SRS program should process new (or regenerated) data before plotting, or to simply replay an existing graph. If regenerated data are being processed, the SRS program should first delete the old graph from the graphics catalog before saving the new graph. PROC GPLOT is needed only when plotting a new graph. Different SYMBOL statements are used, depending on the number of dose groups in the study. One aspect of PROC GPLOT that does not change is its graphic output settings. The OPTIONS EEN-TK2010 and MODISPLAY are always used with PROC GPLOT, and its graphic output is always stored in the temporary catalog TEMP. Also, graphs generated by PROC GPLOT are always grouped together so that PROC GREPLAY can easily replay more than one graph with only one REPLAT command. The final part of GBSR.SAS actually outputs the graph to a graphic device using a combination of GPLOT and PROC GREPLAY. Note that plotting on the Zeta 8 requires the ROTATE option and the use of color mapping.

Segment 6: The last segment of GBSR is essentially a LIBNAME.COM system call with the argument "e:GBS", which passes control from the program to GBSR.COM. The message "Invoking SAS ..." is displayed to signal the end of GBSR.

Using the first sample session in Appendix 5 as an example, we will now describe briefly what GBSR.SAS does after it is invoked by GBSR.COM. First, data is read from XXXX.DAT and sorted. PROC GPLOT is used to delete the old graph generated from a previous session. PROC GPLOT, together with the GPLOT statement, plots the data using TK2010 as the graphics device driver. The graph is sent to the temporary graphics catalog called TEMP without displaying it on the terminal. Next, PROC GREPLAY is used again to group the graph. Note that this is done even when PROC GPLOT generates only one graph. A group name is assigned, and the group is copied to the permanent graphics catalog GLIB.GXXXX. Finally, PROC GREPLAY plots the graph on the Zeta 8 plotter (Appendix 6). The C- and CMP- options map the colors from TK2010 to Z8800. Due to the way independent graphs are replayed on different devices, all graphs are saved with the TK2010 device driver, even if it is to be plotted on the Zeta 8 plotter. Note that GBSR.SAS is much shorter and executes much faster when the user is simply replaying an existing graph (see second sample session, Appendix 5).

CONCLUSION

We have described a method that enables a user to access SAS Software without having to learn SRS and, in the case of graphic applications, SAS/GRAPH. The user enjoys the benefits of data relations displayed in graphic form, without having to deal directly with graphic catalogs. Other options (choice of graphic device, for example) provide flexibility for the user. The use of graphic catalogs allows graphs to be saved and replayed later. This means that a graph has to be generated only once, which may save a considerable amount of time.
Generation of the SAS program via a FORTRAN program assures accuracy and eliminates the need to make minor modifications, while automatic graph and file manipulations and deletions also minimize maintenance and save disk space.

REFERENCES


SAS and SAS/GRAFP are registered trademarks of SAS Institute Inc., Cary, NC, USA.

SAS is a registered trademark of Digital Equipment Corp., Maynard, MA, USA.

The authors may be contacted at:

Dept. of Preclinical Safety Assessment
Sanofi Research Institute
Route 10
East Hanover, NJ 07936

APPENDIX I

Partial listing of GBFM.FOR

C Generate body wt/food consumption data for SAS/GRAFP.
C * indicates codes not directly related to this paper.
C
C LOGICAL DECU
BYTE FRM(2),RMS
CHARACTER*4 TITLE(2)
CHARACTER*3 BUF(2)
CHARACTER*1 TKAA(2)
CHARACTER*2 OGE(2)
CHARACTER*15 DES(2)
CHARACTER*6 TITLE,TITL2
INTEGER*2 LGES(2),IHP(26)
DATA BUF/*',?',HT,*'/
DATA DES/*'Body Weight','Food Consumption'/
DATA TLES/(1,15)
DATA TUB/*'DAY','WEEK'/
DATA T UM/*'M','Y'/
C
C Get study * and password from user
WRITE (2,20)
10 FORMAT ('ENTER STUDY #: ') READ (2,20) L,SN
20 FORMAT ('8$') WRITE(2,50)
C
C Determine if user is allowed to access this study
C
C * Generate study data from related data files.
C Data needed for this example include TITLE, TITL2,
C INT, TIP, IMAF, L, MRS, HMT, MP, TITLE, TITL2.
C
C Generate body weight or food consumption data
WRITE (2,40)
40 FORMAT ('GENTRAN DATA FOR 1 B.W. 2 F.C.');*
READ (2,40) JK
C
C OPEN SAS data file
DEU = .FALSE.
NEW = .TRUE.
ENCISE (30,60,40) SWE,BUF,EL(12)
OPEN (5,NINE-FR,TYPE='NEW',CHARINCREF='LIST')
MOZ = .FALSE. ! Old data file exists
READ (2,70) !
70 FORMAT ('I (B.W.,5) RMS DATA ALREADY GENERATED, REGENERATE [N] * ? ' )
READ (2,80) ) LN,MHS
C
C FORMAT (') SWE,FH(30,50,40) RMS = 'H'
IF (NZ(S.EQ."")' THEN
DEU = .TRUE.
DEL = .TRUE. ! Generate new data file
CLOSE (5,DISP='DELETE') ! don't generate new data file
ELSE
WRITE (2,90)
90 FORMAT ('SPLIT DATA [Y] * ? ' )
READ (2,90) LN,MHS
IF (NZ(S.EQ."")' THEN
DEU = .TRUE.
DEL = .TRUE. ! Generate new data file
CLOSE (5,DISP='DELETE') ! don't generate new data file
STOP
END IF
C
100 CONTINUE ! Open new SAS data file
OPEN (5,NINE-FR,TYPE='NEW',CHARINCREF='LIST')
C ! Display scheduled intervals and get beginning and ending weeks
WRITE (2,100) TUES(TU(1),1),MP
100 FORMAT ('SCHEDULED INTERVALS ARE','A6',:')
C WRITE (2,101)
110 FORMAT (13)'mp)
C ! Process and generate SAS data
WRITE (2,150)
110 FORMAT ('Generating SAS data .../)
C ! Compute means and write means to SAS data file.
C A data line consists of the following:
C ! a, text period, dose group, mean
DO JJ = 1,MP
WRITE (2,150) 11M(JM),IHP(JJ)
WRITE (2,150) (* Processing ',A6',/)
150 FORMAT ('$ EXIT')
C ! Delete existing graph from graphics catalog
IF (.EQ.0) THEN STOP
IF (.NOT. DEU) STOP * ! no need to continue execution
WRITE (2,150)
150 FORMAT ('Your graph will still be saved;/')
C ! Please wait ...
C ! Build GBFM.COM
OPEN (5,N,I,M,G2,,'FORTRAN',TYPE='OLD',ERR=1020)
CLOSE (5,DISP='DELETE') ! delete old GBFM.COM
OPEN (5,N,I,M,G2,,'NEW',CHARINCREF='LIST')
WRITE (5,200) (DEC(I),1,1),BUF(12)
C ! Generate and invoke this command procedure/
C 'S ASSIGN(GBM NEWSASDB SRS INPUT)'
C 'S ASSIGN(GBM NEFBA SRS OUTPUT')
C 'S ASSIGN(GBM SRS DATA')
C 'S DELETE/LOG GBFM.LOG/V'
C 'S EXIT'
C ! Close (5)
CLOSE (5)
C ! Build GBFM.SRS
OPEN (5,N,I,M,G2,,'FORTRAN',TYPE='OLD',ERR=1010)
CLOSE (5,DISP='DELETE') ! delete old GBFM.SRS
OPEN (5,N,I,M,G2,,'NEW',CHARINCREF='LIST')
WRITE (5,210) !
210 FORMAT ('GFM Plot body weight/food',
C 'consumption means;',
C ' Run GBFM to generate and invoke',
C 'this program;'/
C OPTIONS (1) /
C 'LIBRARY GLA ""')
C ! Read SAS data file and sort data
IF (NEW) WRITE (5,220)
220 FORMAT ('GETA MEMA;'/
C ' INTEG DAT/'
C ' INPUT SEX $ 1-7 WEEK GROUP MEMA;'/
C 'PROC SORT;'/
C ' BY DESCENDING SEX;'/
C ! Delete existing graph from graphics catalog
IF (.EQ.0) WRITE (5,230) (DEC(I),1,1),BUF(12)
C 230 FORMAT ('PROC GRAPH PLOT=(""="GBFM.CG",GLA,",HMT","/
C 'DELETE,'LOG GBFM,"";'/
C 'QUIT;'/
C ! If (.NOT.NEW) GOTO 1050
C C ! Save new graph in graphics catalog
C WRITE (5,240) (TITLE(I),1,1),BUF(12)
C 240 FORMAT ('GENERATE DEVICE=TEMP,NODISPLAY;'/
C 'PROC PLOT GBOX-TMP;'/
C 'PROC PLOT GOUT=TEMP;'/
C 'PROC PLOT GOUT=GOUT;'/
C 'VARX <VAR1 VAR2 VAR3> GROUPS=125 SDEF;'/
C 'BY DESCENDING SEX;'/
C 'TITLE H=' ','GLA,HMT","/
C 'DELETE,'LOG GBFM,"";'/
C ! Save new graph in graphics catalog
C WRITE (5,250) (DESC(I),1,1),DESC(12)
250 FORMAT ('TITLE H=' '"=",'CLOSES(1),' GLA","/
C 'QUIT;'/
C ! If (.EQ.0) GOTO 1050
C C ! Save new graph in graphics catalog
C WRITE (5,260) (TITLE(I),1,1),DESC(12)
C 260 FORMAT ('TITLE H=' ""=",'CLOSES(1),' GLA","/
C 'QUIT;'/
C ! If (.EQ.0) GOTO 1050
C C ! Save new graph in graphics catalog
C WRITE (5,270) (TITLE(I),1,1),DESC(12)
270 FORMAT ('TITLE H=' '"=",'CLOSES(1),' GLA","/
C 'QUIT;'/
C ! if (NEW) GOTO 1050
C G2

850
APPENDIX 2
Partial listing of XXXXUN.DAT

APPENDIX 3
Listing of sample GBFN.COM

APPENDIX 4
Listing of sample GBFN.SAS

APPENDIX 5
Listing of sample session

$RUN GBFN
ENTER STUDY : XXX
PASSWORD:
GENERATE DATA FOR 1) R. 2) F.: 1
SAS DATA already generated. REGENERATE [Y] [N]
SCHEDULED INTERVALS ARE WEEK:
-4 -3 -2 -1 1 2 3 4 5 6
17 18 19 20 21 22 23 24 25 26
ENTER FIRST AND LAST WEEKS (): 1,26
Generating SAS data...
Processing WEEK 1
Processing WEEK 2
Processing WEEK 3
Processing WEEK 26

APPENDIX 2
Partial listing of XXXXUN.DAT

APPENDIX 3
Listing of sample GBFN.COM

APPENDIX 4
Listing of sample GBFN.SAS

APPENDIX 5
Listing of sample session

$RUN GBFN
ENTER STUDY : XXX
PASSWORD:
GENERATE DATA FOR 1) R. 2) F.: 1
SAS DATA already generated. REGENERATE [Y] [N]
SCHEDULED INTERVALS ARE WEEK:
-4 -3 -2 -1 1 2 3 4 5 6
17 18 19 20 21 22 23 24 25 26
ENTER FIRST AND LAST WEEKS (): 1,26
Generating SAS data...
Processing WEEK 1
Processing WEEK 2
Processing WEEK 3
Processing WEEK 26

APPENDIX 2
Partial listing of XXXXUN.DAT

APPENDIX 3
Listing of sample GBFN.COM

APPENDIX 4
Listing of sample GBFN.SAS

APPENDIX 5
Listing of sample session

$RUN GBFN
ENTER STUDY : XXX
PASSWORD:
GENERATE DATA FOR 1) R. 2) F.: 1
SAS DATA already generated. REGENERATE [Y] [N]
SCHEDULED INTERVALS ARE WEEK:
-4 -3 -2 -1 1 2 3 4 5 6
17 18 19 20 21 22 23 24 25 26
ENTER FIRST AND LAST WEEKS (): 1,26
Generating SAS data...
Processing WEEK 1
Processing WEEK 2
Processing WEEK 3
Processing WEEK 26

APPENDIX 2
Partial listing of XXXXUN.DAT

APPENDIX 3
Listing of sample GBFN.COM

APPENDIX 4
Listing of sample GBFN.SAS

APPENDIX 5
Listing of sample session

$RUN GBFN
ENTER STUDY : XXX
PASSWORD:
GENERATE DATA FOR 1) R. 2) F.: 1
SAS DATA already generated. REGENERATE [Y] [N]
SCHEDULED INTERVALS ARE WEEK:
-4 -3 -2 -1 1 2 3 4 5 6
17 18 19 20 21 22 23 24 25 26
ENTER FIRST AND LAST WEEKS (): 1,26
Generating SAS data...
Processing WEEK 1
Processing WEEK 2
Processing WEEK 3
Processing WEEK 26

APPENDIX 2
Partial listing of XXXXUN.DAT

APPENDIX 3
Listing of sample GBFN.COM

APPENDIX 4
Listing of sample GBFN.SAS

APPENDIX 5
Listing of sample session

$RUN GBFN
ENTER STUDY : XXX
PASSWORD:
GENERATE DATA FOR 1) R. 2) F.: 1
SAS DATA already generated. REGENERATE [Y] [N]
SCHEDULED INTERVALS ARE WEEK:
-4 -3 -2 -1 1 2 3 4 5 6
17 18 19 20 21 22 23 24 25 26
ENTER FIRST AND LAST WEEKS (): 1,26
Generating SAS data...
Processing WEEK 1
Processing WEEK 2
Processing WEEK 3
Processing WEEK 26
### Six Month Oral Toxicity Study in the Dog on a Test Compound

**Sandoz Project T-XXXX**

**Sex = M & F**

#### Body Weight Means

<table>
<thead>
<tr>
<th>Mean (KG)</th>
<th>0</th>
<th>10.2</th>
<th>10.4</th>
<th>10.6</th>
<th>10.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Group

- **1**
- **2**
- **3**
- **4**