SAS AND APL: THE SIMULTANEOUS USE OF BOTH IN DATA ANALYSIS

Richard LaValley and Eugene Mertz, Satellite Business Systems

I. Introduction

With the modernization of data collection and processing hardware, most businesses are facing an enormous quantity of useful data. Analysts have the challenging task of reducing this data into useful and concise information which can be used by management to make well informed and timely decisions. Many software tools have been developed to assist analysts in their efforts. Two of the more powerful tools are SAS and APL.

SAS offers business analysts standard statistical and graphical techniques which can be easily used in a relatively user-friendly environment. Simple data manipulation can be easily coded within SAS. Complex data manipulation, however, is not as easily accomplished and the code is complex.

APL is an analytically structured programming language which provides the analyst with the capability to do complex data manipulations easily. Additionally, a vast number of statistical procedures are available for an installation's public libraries. Data selection and data manipulation are extremely easy with the use of APLDI, one of the standard software packages, which can be obtained from IBM. This package is extremely useful in processing large data sets because it is easy to request complex multiple criteria selections from the data. In addition, this package has a series of standard data manipulation programs which can be used on the selected data (programs such as sums, averages, crosstabs, etc.) as well as any program that the analyst can develop.

This paper attempts to show examples of the use of both APL and SAS in data analysis and data modeling. It attempts to show both "tools" being used to perform the necessary job in a quick and easy way.

Satellite Business Systems (SBS), is a telecommunications firm which provides voice, data and image communication service to commercial and residential users. SBS is currently engaged in an extensive program of data reduction in order to provide the information necessary for assessing operations performance, maintenance effectiveness and networking optimization.

The Operations Evaluation Group (OEG) within SBS under the direction Dr. William C. Hardy has been given the task of providing upper management with the necessary information and tools to monitor, characterize and diagnose operational performance. This program is an attempt to provide management with the pertinent information without exposure to the high volume (over 500,000 phone calls daily are processed) of data. This group has been developing new measurements and upgrading industry standard measurements to SBS operations. The development phase of these measurements requires a vast amount of ad hoc mathematical modeling and data processing.

This data processing is currently being completed by a combination of assembly language (ASM), APL, APLDI and SAS programs. The assembly language programs are being used to do the large scale processing and sorting of the data. APL and APLDI are basically being used for data manipulation, development and testing of new measurements, producing preliminary analysis of the data and development of special analytical techniques. SAS is being used for more sophisticated analytical techniques such as multivariate regression and ANOVA and for its excellent graphic capability for presentation of data.

II. Computer Center Environment

Currently software includes MVS/SP1 operating system with JES2 for batch job execution control, TSO for interactive work using the products VSAPL for TSO release 4.0, APLDI-2, and SAS release 79.6

The I/O hardware pertinent to our analytic requirements includes IBM3279-2B terminals, IBM3287-2C quad-color printers and an IBM3800 with APL character sets installed. The choice of I/O devices selected for output depends, of course, on the analytic application. The SAS graphic outputs, are currently being routed to the quad-color printers.

The batch job system allows for execution of many applications as if they were TSO sessions but with terminal inputs being supplied from an "alternate input source", usually a data set which has been edited to contain the necessary keyboard inputs to supply, in sequence, the entries to each open keyboard request. This flexibility permits repetitive job submission with varying data input provided by looping program in APLDI or a single job submission having multiple steps each one of which provides for analysis and/or graphical display of data.

VSAPL for TSO has the capability, through auxiliary processors which provide I/O services to APL, to read and write TSO data sets. This capability is key to the success of using APL based applications in a total system sense which tends to be transparent to the user.

III. Analytic Requirements at SBS

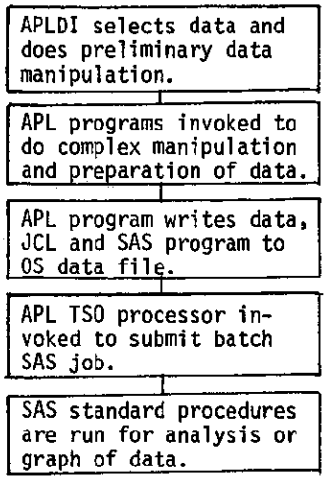
The main requirements for data analysis fall in the areas of ad hoc queries, modeling and production reporting. APLDI and SAS are two of the tools which augment each other very well. Since it is not always clear whether SAS alone, APLDI alone or a combination of the two is the best method to use in the interest of saving computer resources, many of the production reporting systems are first modeled using SAS and APLDI in combination and an optimum approach selected once the detail of the production system can be specified with accuracy. The end system may not use SAS or APL/APLDI at all but may be written in ASM or PL/I instead. However, for a quick implementation with minimum manpower investment to evaluate the usefulness of a particular production reporting system the SAS/APLDI combination is usually selected. Section IV of this paper presents several such implementations.

IV. Implementation

In our experimentation of this technique, we have used two different approaches to using SAS and APL and we feel a third is possible. The first technique is using APL/APLDI to do the complex data selection and data manipulation from our database. We then invoke an APL program which writes a SAS program to an OS data set and then submits the job in batch. The second technique is to have a pre-stored skeleton SAS program in an OS data set. We use APL/APLDI to do the complex data retrieval and manipulation. However, in this approach, we use APL to read the SAS skeleton program, insert the data in the proper place, write the SAS program to an OS data set and submit the batch job. The third technique which has not been tested but we feel is appropriate to mention is using interactive SAS to invoke APL, use APL to do the data manipulation and selection, use APL to write the data to an OS data set and then use SAS to retrieve the data for analysis.

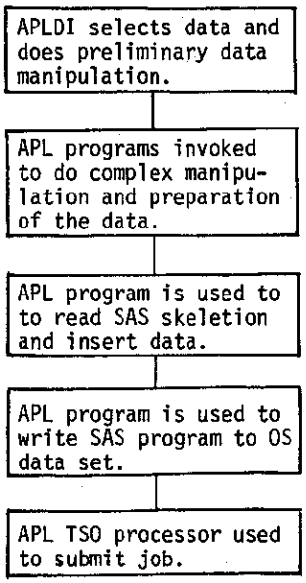
The first technique is simplified in diagram A and the basic example programs are documented in Appendix I. The first step in this technique is the use of APLDI to retrieve, select and do preliminary manipulation of the data from our large data base. The type of complex selections that we are using are quite easy to do in APLDI. In addition, we use APLDI to automatically call up the next step in the process. This next step is the further manipulation of the reduced data from APLDI. This step includes such things as developing models of traffic flow and pattern, spline fitting of data, integration of pattern models, and development of cumulative distributions. This step is used to prepare the data for a SAS standard procedure such as a Proc GPLOT, Proc REG, Proc GLM, etc. The next step in this process is to invoke the standard APLDI program FILE to write the modified data to an OS data set. The final step invokes an APL program which writes the JCL and SAS code to an OS data set and submits the batch job.

DIAGRAM A



The second technique is simplified in Diagram B and the basic example programs are documented in in Appendix II. As in the first technique, APL and APLDI programs are utilized to do the data selection and retrieval, data manipulation, and data preparation for SAS procedures. This technique uses APL to read pre-stored JCL and SAS program from an OS data set, insert data in the appropriate place in the SAS program, write the program to an OS data set, and submit the SAS batch program. This technique has an advantage of maintenance of the skeleton programs being done using either the TSO or SPF editor and requires no knowledge of APL for this maintenance.

DIAGRAM B



The two techniques have been used at SBS by the Operations Evaluation Group. These techniques have been used in a production type environment with the use of APL batch programs. OEG presently uses these techniques to conduct specialized analysis on each of its offered operational services monthly.

V. Conclusion

This paper shows an interface of APL and SAS that is currently being used at SBS. We feel there are many ways that this type of interface can be accomplished. In OEG, we are continually developing and investigating ways to use APL and its applications. This interface with SAS represents one technique that has allowed us to tap the analytical and graphical presentation resources of SAS. It has enabled us to do hundreds of SAS jobs with one APL batch submission.

References

1. APL Data Interface - II/TSO Program Description and Operations Manual, IBM #SH20-2515

2. VSAPL for TSO Terminal User's Guide, IBM #SH20-9180

3. SAS User's Guide: Basics 1982 edition, SAS Institute, Inc.

4. SAS User's Guide: Statistics 1982 edition, SAS Institute, Inc,

5. SAS/Graph Users Guide, 1981 edition, SAS Institute, Inc.

Persons to contact for further information:

Richard LaValley or Eugene Mertz
Satellite Business Systems
8003 Westpark Drive
McLean, VA 22102

****

APPENDIX I

```
**********************************************************************    01/13/1983  08:58   SPLINEJCL.
    ∇ Z←SPLINEJCL;TSO;REC;CTL;X;I;DSN;M;DATADSN;TIT2
[1]   ⍝DEFINE OUTPUT DATA SET NAME.
[2]    DSN←''''USRID.PROGRAM.CNTL'''
[3]    DATADSN←BAREQ 'NAME OF DATASET CONTAINING DATA TO BE PLOTTED'
[4]   ⍝SET UP SAS TITLE2 INTO VARIABLE TIT2
[5]    TIT2←BAREQ 'TITLE2'
[6]   ⍝INITIALIZE MATRIX M TO NULL ROWS OF LENGTH 80
[7]    M←⍬ 80⍴''
[8]   ⍝BUILD JCL INTO MATRIX M
[9]    M←M,[⎕IO] 80↑'//USRIDX JOB (ACC,BN),USRID,CLASS=A,MSGCLASS=A,NOTIFY=USRID'
[10]   M←M,[⎕IO] 80↑'/*JOBPARM  T=0'
[11]   M←M,[⎕IO] 80↑'// EXEC SAS,REGION=1000K'
[12]  ⍝SET NAME OF DATA SET CONTAINING DATA INPUT INTO JCL
[13]   M←M,[⎕IO] 80↑'//TEST DD DSN=',DATADSN,',DISP=OLD'
[14]   M←M,[⎕IO] 80↑'//SAS3287 DD SYSOUT=(S,SASWTR)'
[15]  ⍝BUILD SAS PROGRAM INTO MATRIX M
[16]   M←M,[⎕IO] 80↑'GOPTIONS DEVICE=IBM3287F HSIZE=8.5;'
[17]   M←M,[⎕IO] 80↑'OPTIONS DEVADDR=(SAS,,,P3);'
[18]   M←M,[⎕IO] 80↑'DATA;'
[19]   M←M,[⎕IO] 80↑'INFILE TEST;'
[20]   M←M,[⎕IO] 80↑'INPUT Y X;'
[21]   M←M,[⎕IO] 80↑'PROC GPLOT;'
[22]   M←M,[⎕IO] 80↑'PLOT Y*X/CVREF=BLACK AREAS=1 VREF=',(⍕PORTS),';'
[23]   M←M,[⎕IO] 80↑'SYMBOL C=BLACK I=JOIN V=E;'
[24]   M←M,[⎕IO] 80↑'TITLE .C=RED POTENTIAL UTILIZATION CURVE;'
[25]  ⍝SET SAS TITLE INTO SAS PROGRAM
[26]   M←M,[⎕IO] 80↑'TITLE2 ',TIT2,';'
[27]   M←M,[⎕IO] 80↑'LABEL X=TIME IN G.M.T.;'
[28]   M←M,[⎕IO] 80↑'LABEL Y=DEMAND ON GIVEN PORTS;'
[29]  ⍝SETUP TSO COMMAND AUXILIARY PROCESSOR FOR TSO FILE SERVICES FOR APL
[30]   X←100 ⎕SVO 'TSO'
[31]   →(0≠X←TSO)/NOSHARE
[32]   TSO←'FREE FI(SYSOUTB) ATTRLIST(FB)'
[33]   TSO←'ATTR FB BLKSIZE(800) LRECL(80) RECFM(F B S)'
[34]  ⍝TSO←'ALLOC FI(SYSOUTB) NEW SP(1 1) TRACKS USING(FB) DA(',DSN,')'
[35]   TSO←'ALLOC FI(SYSOUTB) OLD DA(',DSN,')'
[36]   →(0≠X←TSO)/TSOERR
[37]  ⍝SETUP TSO FILE I/O AUXILIARY PROCESSOR FOR USE BY APL
[38]   REC←'SYSOUTB (256'
[39]   CTL←REC,')(CTL'
[40]   X←111 ⎕SVO 2 3⍴'RECCTL'
[41]   →(0≠X←CTL)/NOSHARE
[42]  ⍝WRITE THE OUTPUT FILE WITH THE CONTENTS OF MATRIX M
[43]  L:REC←80↑M[⎕IO;]
[44]   →(0≠X←CTL)/IOERROR
[45]   →(0≠11⍴M←1 0↓M)/L
[46]   'DONE...'
[47]  ⍝CLOSE AND FREE THE OUTPUT FILE
[48]   Z←⎕SVR 2 3⍴'RECCTL'
[49]   TSO←'FREE FI(SYSOUTB)'
[50]   →0
[51]  ⍝ERROR HANDLING AND ERROR MESSAGE DISPLAY FROM RC(11) MESSAGE MATRIX.
[52]  IOERROR:(RC(11)[1 12 15 17 30 440 441 442 443 444 445 901 913 937 ⍳X;]),' RC=',⍕X
[53]  HOLDERR:SAVSPLINEJCL←RESTART
[54]  RESTART:→L
[55]  NOSHARE:→HOLDERR,0⍴⎕←'SHARE ERROR',⍕X
[56]  TSOERR:→HOLDERR,0⍴⎕←'TSO ERROR ',⍕X
    ∇
```

APPENDIX II

```
********************************************************************    01/13/1983  08:58   WRITECARDS
    ▼ Z←DSN WRITECARDS M;TSO;REC;CTL;X
[1]  ⍝ARGUMENT DSN IS THE DATA SET NAME FOR OUTPUT
[2]  ⍝ARGUMENT M IS THE MATRIX OF SAS-READY DATA
[3]  ⍝SETUP TSO COMMAND AUXILIARY PROCESSOR FOR TSO FILE SERVICES FOR APL
[4]   X←100 ⎕SVO 'TSO'
[5]   →(0≠X←TSO)/NOSHARE
[6]   TSO←'DELETE ',DSN
[7]   TSO←'FREE FI(SYSOUTB) ATTRL(ST(FB)'
[8]   TSO←'ATTR FB BLKSIZE(3840) LRECL(80) RECFM(F B S)'
[9]   TSO←'ALLOC FI(SYSOUTB) NEW SP(1 1) TRACKS USING(FB) DA(',DSN,')'
[10]  →(0≠X←TSO)/TSOERR
[11] ⍝SETUP TSO FILE I/O AUXILIARY PROCESSOR FOR USE BY APL
[12]  REC←'SYSOUTB (256'
[13]  CTL←REC,')(CTL'
[14]  X←111 ⎕SVO 2 3 ⍴'RECCTL'
[15]  →(0≠X←CTL)/NOSHARE
[16] ⍝WRITE THE OUTPUT FILE WITH THE CONTENTS OF MATRIX M
[17] L:REC←80↑M[⎕IO;]
[18]  →(0≠X←CTL)/IOERROR
[19]  →(0≠1↑⍴M←1 0 ↓M)/L
[20]  'DONE ...'
[21] ⍝CLOSE AND FREE THE OUTPUT FILE
[22]  Z←⎕SVR 2 3 ⍴'RECCTL'
[23]  TSO←'FREE FI(SYSOUTB)'
[24]  →0
[25] ⍝ERROR HANDLING AND ERROR MESSAGE DISPLAY FROM RC111 MESSAGE MATRIX
[26] IOERROR:(RC111[1 12 15 17 30 440 441 442 443 444 445 901 913 937 ↓X;]),' RC=',⍕X
[27] HOLDERR:⎕SWRITECARDS←RESTART
[28] RESTART:→L
[29] NOSHARE:'SHARE ERROR',⍕X
[30]  →HOLDERR
[31] TSOERR:'TSO ERROR ',⍕X
[32]  →HOLDERR
```

```
********************************************************************    01/13/1983  08:58   READCARDS.
    ▼ Z←W READCARDS DSN;TSO;REC;CTL;W;X;Y;Y3;Y23
[1]  ⍝ARGUMENT DSN IS THE DATA SET NAME FOR INPUT
[2]  ⍝ARGUMENT W IS THE NUMBER OF COLUMNS IN THE INPUT DATA SET
[3]  ⍝INITIALIZE MATRIX Z TO NULL ROWS OF LENGTH W
[4]   Z←(0,W)⍴''
[5]  ⍝SETUP TSO COMMAND AUXILIARY PROCESSOR FOR TSO FILE SERVICES FOR APL
[6]   X←100 ⎕SVO 'TSO'
[7]   →(0≠X←TSO)/NOSHARE
[8]   REC←'CARDS(192'
[9]   CTL←REC,')(CTL'
[10]  TSO←'ALLOC FI(CARDS) DA(',DSN,') SHR'
[11]  →(0≠X←TSO)/TSOERR
[12] ⍝SETUP TSO FILE I/O AUXILIARY PROCESSOR FOR USE BY APL
[13]  X←111 ⎕SVO 2 3 ⍴'RECCTL'
[14]  →(0≠X←REC)/NOSHARE
[15] ⍝ AND EDIT SAS-READY DATA INTO DATA STREAM WHEN KEYWORDS "CARDS"
[16] ⍝ AND "TITLE2" ARE FOUND
[17] L:Y←REC
[18]  →((0=⍴Y)∨∧/'NO RECORDS FOUND'=16↑Y)/END
[19]  →(0≠X←CTL)/IOERROR
[20] LD:→(∧/'CARDS'=6↑Y)/GO
[21]  →(∧/'TITLE2'=6↑Y)/GO2
[22]  →GO1
[23] ⍝ADD IN "CARDS" DATA
[24] ⍝ ORD, MTBOSIOBJ, AND BASELINE ARE GLOBAL VARIABLES SET BY USER
[25] ⍝ Y3 IS A LOCAL VARIABLE SET BY PREVIOUS APLDI QUERY AND APL PROCESSING OF GLOBAL VARIABLES Q1
AND Q4
[26] GO:Z←Z,[⎕IO](1,W)⍴W↑Y
[27]  Y3← 10 2 ⍕(((⍴Q1),1)⍴Q1),Q4
[28]  Y23←(1↓⍴Y3)↑¯1+⍴MTBOSIOBJ
[29]  Y1←((Y23,22)↑ 3 0 10 2 10 2 ⍕(((ORD),[0.5] 23↑MTBOSIOBJ),23↑BASELINE)),(Y23,20)↑Y3
[30]  Z←Z,[⎕IO]((1↑⍴Y1),W)↑Y1
[31]  →L
[32] ⍝ADD IN "TITLE2" DATA
[33] GO2:Y←((-+/∧\' '=⌽Y)↓Y),' ',TIT2,''''
[34] GO1:Z←Z,[⎕IO](1,W)⍴W↑Y
[35]  →L
[36] END:'DONE ...',⍕⍴Z
[37]  →EXIT
[38] ⍝ERROR HANDLING AND ERROR MESSAGE DISPLAY FROM RC111 MESSAGE MATRIX
[39] TSOERR:'TSO ERROR ',⍕X
[40] EXIT:
[41]  ⎕SVR 3 3 ⍴'RECCTLTSO'
[42]  TSO←'FREE FI(CARDS)'
[43]  →0
[44] IOERROR:(RC111[1 12 15 17 30 440 441 442 443 444 445 901 913 937 ↓X;]),'RC=',⍕X
[45]  ⎕SREADCARDS←HOLDERROR
[46] HOLDERROR:→L
[47]  →EXIT
[48] NOSHARE:'SHARE ERROR ',⍕X
[49]  →IOERROR
```

APPENDIX II (con't)

```
**** TSO FOREGROUND HARDCOPY ****
                                        (SKEL    )

//USRIDX JOB (ACCOUNT,BIN),'USRID JOBS',CLASS=A,MSGCLASS=Z,NOTIFY=USRID
// EXEC SAS,REGION=1000K
//SAS3287 DD SYSOUT=(S,SASWTR)
//**********************************************************************
//*  THIS PROGRAM IS USED TO GRAPH THE OPERATIONAL CHARACTERISTIC      *
//*  CURVES OF SERVICES, NETWORKS, AND INDIVIDUAL COMPONENTS IN        *
//*  THE SBS SYSTEM.  THIS IS A TYPICAL EXAMPLE OF THE USE OF SKEL     *
//*  IN TECHNIQUE TWO (DIAGRAM B).                                     *
//**********************************************************************
GOPTIONS DEVICE=IBM3287F HSIZE=8;
OPTIONS DEVADDR=(SAS,,,P0);
PROC FORMAT;
VALUE CUMX 3.91='    50'
           4.61='   100'
           5.30='   200'
           5.52='   250'
           5.70='   300'
           5.99='   400'
           6.21='   500'
           6.40='   600'
           6.55='   700'
           6.68='   800'
           6.80='   900'
           6.91='  1000'
           7.31='  1500'
           7.60='  2000'
           7.82='  2500'
           8.01='  3000'
           8.16='  3500'
           8.29='  4000'
           8.52='  5000'
           8.70='  6000'
           8.85='  7000'
           8.99='  8000'
           9.10='  9000'
           9.21=' 10000'
           9.43=' 12500'
           9.62=' 15000'
           10.13=' 25000'
           10.82=' 50000'
           11.51='100000'
           11.92='150000';
VALUE DURX 0=<1
           1.098=<3
           1.609=<5
           2.708=<15
           3.806=<45
           4.787=<120
           5.192=<180
           5.703=<300
           6.040=<420
           6.291=<540
           6.492=<660
           6.866=<960;
DATA;
INPUT @1 DURATION 3. @4 CUM83 10.2 @14 CUM 10.2 @24 DURN 10.2
     @34 MTBOSI 10.2;
CUMLOG=LOG(CUM);
CUMLOG83=LOG(CUM83);
CUMTBOSI=LOG(MTBOSI);
DURATION=LOG(DURATION);
DURN=LOG(DURN);
CARDS;
PROC GPLOT;
FORMAT DURN DURX.;
FORMAT DURATION DURX.;
FORMAT CUMLOG CUMX.;
FORMAT CUMLOG83 CUMX.;
FORMAT CUMTBOSI CUMX.;
PLOT CUMLOG*DURATION CUMLOG83*DURATION CUMTBOSI*DURN/
     VAXIS=3.91 4.61 5.30 5.52 5.70
     5.99 6.21 6.40 6.55 6.68 6.80 6.91 7.31 7.60 7.82 8.01 8.16 8.29
     8.52 8.70 8.85 8.99 9.10 9.21 9.43 9.62 10.13 10.82 11.51 11.92
     CAXIS=GREEN OVERLAY CTEXT=GREEN HAXIS=0 1.098 1.609 2.708 3.806
     4.787 5.192 5.703 6.040 6.291 6.492 6.866;
TITLE .F=TRIPLEX .C=GREEN .H=4 OPERATING CHARACTERISTIC CURVES;
TITLE2 .F=TRIPLEX .H=2 .C=GREEN
NOTE;
NOTE;
NOTE;
NOTE;
NOTE;
NOTE;
NOTE;
NOTE;
NOTE .J=C .H=2 .C=GREEN 'YREND OBJECTIVE      ';
NOTE .J=R .H=2 .C=GREEN 'YREND BASELINE      ';
SYMBOL1 C=GREEN I=SM50;
SYMBOL2 C=GREEN I=SM50;
SYMBOL3 C=GREEN I=JOIN;
LABEL DURATION=DURATION OF OUTAGE;
LABEL CUMLOG=MTBOSI FOR OUTAGE T MIN OR LONGER;
LABEL CUMLOG83=MTBOSI FOR OUTAGE T MIN OR LONGER;
```

```
**** TSO FOREGROUND HARDCOPY ****
DATA;
INPUT @1 DURATION 3. @4 CUM83 10.2 @14 CUM 10.2 @24 DURN 10.2
     @34 MTBOSI 10.2 @44 CUM82 10.2;
CUMLOG=LOG(CUM);
CUMLOG83=LOG(CUM83);
CUMTBOSI=LOG(MTBOSI);
CUMLOG82=LOG(CUM82);
DURATION=LOG(DURATION);
DURN=LOG(DURN);
CARDS;
  1     196.00     134.4       1.00     134.42       77.0
  2     425.00     249.4       1.02     158.86       94.7
  3     472.00     439.6       1.03     160.75      225.1
  4     521.00     490.3       1.05     162.86      259.8
  5     554.00     556.3       1.07     164.46      276.8
 10     777.80     786.9       1.08     166.19      379.2
 15     965.50     982.2       1.10     167.85      480.0
 30    1846.00    1400.0       1.12     169.35      637.4
 45    2545.00    1618.4       1.13     171.28      782.0
 60    3111.00    1861.7       1.15     172.32      903.0
120   12000.00    3794.0       1.17     173.90     1805.9
150   16800.00    5438.0       1.18     174.86     2298.5
PROC GPLOT;
FORMAT DURN DURX.;
FORMAT DURATION DURX.;
FORMAT CUMLOG CUMX.;
FORMAT CUMLOG83 CUMX.;
FORMAT CUMTBOSI CUMX.;
PLOT CUMLOG*DURATION CUMLOG83*DURATION CUMLOG82*DURATION/
     VAXIS=3.91 4.61 5.30 5.52 5.70
     5.99 6.21 6.40 6.55 6.68 6.80 6.91 7.31 7.60 7.82 8.01 8.16 8.29
     8.52 8.70 8.85 8.99 9.10 9.21 9.43 9.62 10.13 10.82 11.51 11.92
     CAXIS=GREEN OVERLAY CTEXT=GREEN HAXIS=0 1.098 1.609 2.708 3.806
     4.787 5.192 5.703 6.040 6.291 6.492 6.866;
TITLE .F=TRIPLEX .C=GREEN .H=4 OPERATING CHARACTERISTIC CURVES;
TITLE2 .F=TRIPLEX .C=GREEN .H=2 CALCULATED VALUES;
NOTE;
```