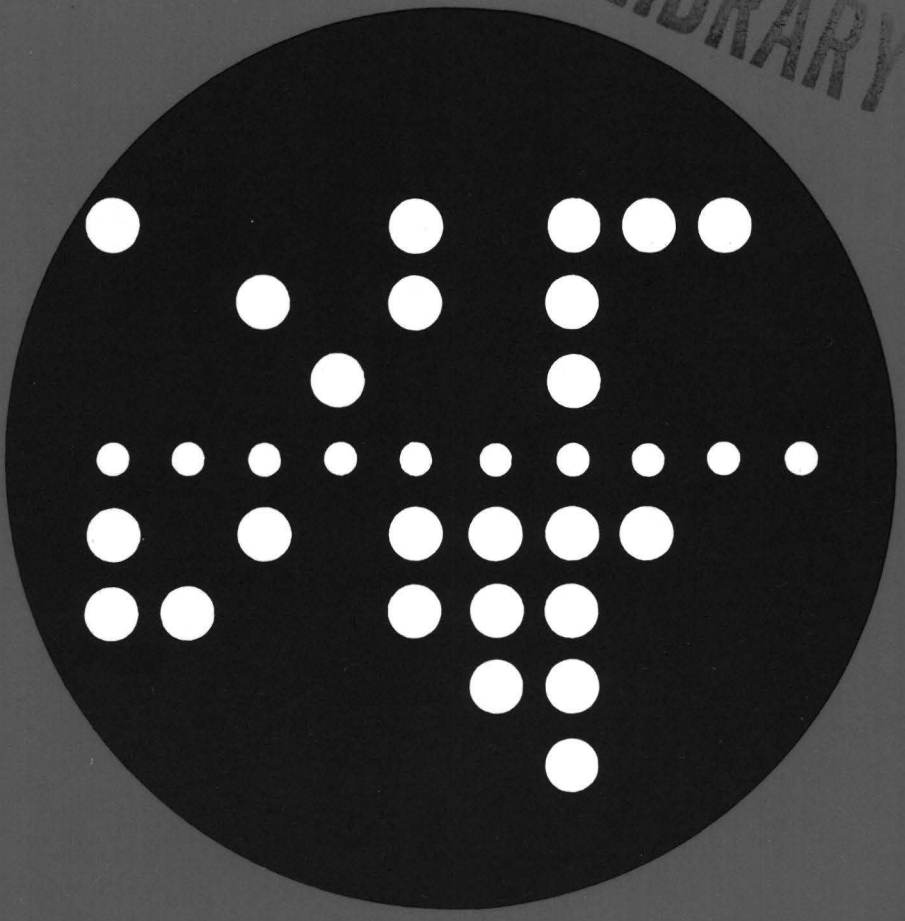
COMPUTING CENTRE NEWSLETTER

October 1980 - N. **45**



Commission of the European Communities

JOINT
RESEARCH
CENTRE

Ispra Establishment

CONTENTS

EDITORIAL NOTE

The Computing Centre Newsletter is published monthly except for August and December.

It describes developments, modifications and specific topics in relation to the use of the computing installations of the Joint Research Centre, Ispra Establishment.

The aim of the Newsletter is to provide information of importance to the users of the computing installations, in a form which is both interesting and readable.

The Newsletter also includes articles which are of intellectual and educational value in order to keep the users informed of new advances in computer science topics.

Administration and contact address:

Ms. A. Cambon (tel. 730)
Support to Computing
Building 36
J.R.C. Ispra Establishment
21020-ISPRA (Varese)

IMSL and NAG Libraries on TSO

M. Dowell

Introduction

In two previous articles we have explained how to use the NAG Library (Newsletter N. 38, page 6) and the IMSL Library (Newsletter N. 42, page 3) of mathematical subroutines in batch FORTRAN jobs. In this article we describe the use of these libraries for test compilation and execution of FORTRAN programs using the TSO system.

Library Names

The names of the data sets and volumes containing the libraries are given in the following table.

LIBRARY	DATA SET NAME	VOLUME
IMSL SINGLE PRECISION	SYS1.LIBMASXS	COPICB
IMSL DOUBLE PRECISION	SYS1.LIBMASXD	COPICB
NAG SINGLE PRECISION	SYS1.LIBNAGS	COPICB
NAG DOUBLE PRECISION	SYS1.LIBNAGD	COPICB

ALL OF THE DATA SETS IN THE ABOVE TABLE ARE CATALOGUED

Use of the Libraries

There are two different ways in which subroutines from the mathematical libraries may be used in TSO test FORTRAN compilations.

a) Using the FG1CLG TSO command procedure to perform the compilation, link edit and execution. With this procedure it is possible to obtain the necessary subroutines from one of the libraries by including a specific parameter.

b) Using the FORT TSO command followed by either LOADGO or LINK it is possible to include subroutines from one or more of the libraries.

Note. It is also possible to make use of the mathematical libraries by using the CONCAT TSO command. For further details see the IBM Manual TSO Command Reference Manual (GC28-6732).

a) Using FG1CLG

The TSO command procedure FG1CLG has a parameter PRN(~~). This, together with the VLB(~~) parameter, is equivalent to the use of the PRN= ,VLB= and ULB= parameters of FG1CLG in a batch mode. This allows the inclusion of subroutines from one of the mathematical libraries.
The PRN(~~) parameter has as operand the xxxx part of the library name SYS1.LIBxxxx (e.g. for SYS1.LIBNAGS the user should include the parameter PRN(NAGS)).
The VLB(~~) parameter has as operand the volume serial number of the disk on which the library resides (i.e. COPICB for all of the mathematical libraries).
An example of the use of this technique is given in the appendix to this article (example 1).

b) Use of FORT followed by either LOADGO or LINK

Using this method the compilation of the FORTRAN program is split into two separate phases:

1) The compilation to produce an object module.

2) The use of either the link-editor or the loader to take the object module together with any appropriate subroutine from the various subroutine libraries and produce an executable program.

   Note. If the link editor procedure is used then a load module version of the program is created. This must be executed using the TSO CALL command.

In the second stage of this operation it is possible to include subroutines from one or more of the mathematical libraries by using the LIB(~~) parameter.
Examples of the use of these techniques are given in the appendix to this article (examples 2, 3, 4).

Note for NAG Users

Users are reminded that in the case of NAG library routines the routine names are different for single and double precision libraries.

E04CAF on the double precision library becomes:
E04CAE on the single precision library (i.e. the final character is chaged from F to E)

However, for the IMSL library the routine names are the same for the single and double precision versions. Therefore, in general, it will not be possible to use subroutines from the single & double precision IMSL libraries in the same program.

Annex

In the following examples lines typed by the user are shown in lower case.
The carriage return/ENTER character at the end of each input line is marked by a (CR).

Example 1

This example shows the use of the FG1CLG TSO command procedure for a program which uses the IMSL single precision library.

A

```
qed newcomp1 fortgi new (CR)
INPUT
00010c  example of imsl single precision library (CR)
00020c  analysis of two-way classification design data(CR)
00030       integer i,ndf(5),ier(CR)
00040       real y(6),em(11),gm,s(5)(CR)
00050       data y/73.,90.,98.,107.,94.,49./(CR)
00060       call arcban(y,1,3,2,em,gm,s,ndf,ier)(CR)
00070       write(6,99999) (em(i),i=1,3)(CR)
00080       write(6,99998) (em(i),i=4,5)(CR)
00090       stop (CR)
0010099999 format(18h block means are : ,11f7.2)(CR)
0011099998 format(22h treatment means are : ,11f7.2)(CR)
00120       end (CR)
00130(CR)
QED
scan (CR)
QED
end save (CR)
SAVED
READY
```

B

```
fg1clg newcomp1 prn(masxs) vlb(copicb) (CR)
DATA SET NEWCOMP1.DUMP NOT IN CATALOG
UTILITY DATA SET NOT FREED, IS NOT ALLOCATED
UTILITY DATA SET NOT FREED, IS NOT ALLOCATED
UTILITY DATA SET INVALID, FILENAME RESTRICTED
ENTER CONTROL STATEMENTS-
(CR)
END OF CONTROL STATEMENTS

BLOCK MEANS ARE :  81.50 102.50  71.50
TREATMENT MEANS ARE :  88.33  82.00
```

In stage A a FORTRAN program is typed by the user (using the QED editor). Note the use of the SCAN subcommand of QED to check the validity of the FORTRAN program.

In stage B the FG1CLG TSO command procedure is used to compile link edit and execute the program. Note, that link editor control statements may be input. Typing a (CR) without any other information ends these control statements.

Example 2

This example shows the use of the NAG single precision library using FORT followed by LOADGO.

```
list newcomp2.fort ↵
 NEWCOMP2.FORT
00010 C      EXAMPLE OF NAG DOUBLE PRECISION LIBRARY
00020 C      USES F03AAF - MATRIX DETERMINANT CALCULATION
00030        DOUBLE PRECISION DETERM,A(4,4),WKSPCE(18)
00040        INTEGER I,N,J,IA,IFAIL
00050        READ(5,99999) (WKSPCE(I),I=1,7)
00060        N=3
00070        READ(5,99998) ((A(I,J),J=1,N),I=1,N)
00080        IA=4
00090        IFAIL=1
00100        WRITE(6,99997) (WKSPCE(I),I=1,6)
00110        CALL F03AAF(A,IA,N,DETERM,WKSPCE,IFAIL)
00120        IF(IFAIL.EQ.0) GOTO 20
00130        WRITE(6,99996)IFAIL
00140        STOP
00150 20     WRITE(6,99995)DETERM
00160        STOP
00170 99999 FORMAT(6A4,A3)
00180 99998 FORMAT(3F5.0)
00190 99997 FORMAT(4(1X/),1H ,5A4,A3,7HRESULTS/1X)
00200 99996 FORMAT(25H0ERROR IN F03AAF IFAIL = ,I2)
00210 99995 FORMAT(24H0VALUE OF DETERMINANT = ,F4.1)
00220        END
READY
fort newcomp2 ↵
G1 COMPILER ENTERED
SOURCE ANALYZED
PROGRAM NAME = MAIN
*  NO DIAGNOSTICS GENERATED
READY
loadgo newcomp2.obj lib('sys1.libnagd') fortlib ↵
fo3aaf example program data ↵
    33    16    72 ↵
   -24   -10   -57 ↵
    -8    -4   -17 ↵

 FO3AAF EXAMPLE PROGRAM RESULTS

 VALUE OF DETERMINANT =  6.0
```

In stage A a previously created data set is listed.
In stage B the program stored in the data set is compiled.
In stage C the load and execution of the program is performed.
In C1 the data is input. In C2 the output is produced.

Example 3

This example shows the compilation link edit and execution of a program which uses the NAG simple precision library.

```
list newcomp3.fort (CR)
 NEWCOMP3.FORT
00010        INTEGER MAXDIV,IFAIL,NOFUN
00020        REAL A,B,EPS,ACC,ANS,ERROR,FUN
00030        EXTERNAL FUN
00040        A=0.0
00050        B=1.0
00060        MAXDIV=20
00070        EPS=1.0E-8
00090        IFAIL=1
00100        CALL D01AGE(0.0,1.0,FUN,MAXDIV,EPS,0.0,ANS,ERROR,NOFUN,
A 00110     * IFAIL)
00120        WRITE(6,99998)ANS,ERROR,NOFUN
00130        IF(IFAIL)20,40,20
00140 20     WRITE(6,99997)
00150 40     STOP
00160 99998 FORMAT(/12H INTEGRAL = ,F11.4,3X,9H ERROR = ,E11.4,3X,
00170       * 20H NUMBER OF POINTS = ,I3)
00180 99997 FORMAT(43H METHOD WAS UNABLE TO EVALUATE THE INTEGRAL)
00190        END
00200        REAL FUNCTION FUN(X)
00210        REAL X
00220        FUN=4.0/(1.0+X*X)
00230        RETURN
00240        END
READY
fort newcomp3 (CR)
G1 COMPILER ENTERED
SOURCE ANALYZED
PROGRAM NAME = MAIN
B * NO DIAGNOSTICS GENERATED
SOURCE ANALYZED
PROGRAM NAME = FUN
*  NO DIAGNOSTICS GENERATED
    *STATISTICS*  NO DIAGNOSTICS THIS STEP
READY
link newcomp3.obj lib('sys1.libnags') fortlib (CR)
C
READY
call newcomp3 (CR)
TEMPNAME ASSUMED AS A MEMBER NAME
D
INTEGRAL =      3.1416    ERROR =  0.3052E-04 NUMBER OF POINTS =   9
```

In stage A a previously created data set is listed.
In stage B the program stored in the data set is compiled.
In stage C the link edit of the program is performed.
The output load module of the program is stored in NEWCOMP3.LOAD(TEMPNAME).
In stage D the library program is executed.

Example 4

This example shows the compilation, load and execution of a program which uses both the NAG and IMSL single precision libraries.

```
list newcomp4.fort ↵
 NEWCOMP4.FORT
00010 C      EXAMPLE OF THE USE OF TWO LIBRARIES
00020 C      NAG & IMSL (BOTH SINGLE PRECISION)
00030 C      THE PROGRAM FINDS THE ROOT OF A FUNCTION
00040        INTEGER MAXFN,IER
00050        REAL A,B,FUN
00060        EXTERNAL FUN
00070        A=-10.
00080        B=10.
00090        MAXFN=100
00100        CALL ZBRENT(FUN,0.0,8,A,B,MAXFN,IER)
00110        IF(IER.EQ.0)GOTO 20
00120        WRITE(6,99998)IER
00130        STOP
00140 20     WRITE(6,99999)B
00150        STOP
00160 99999  FORMAT(20H0ESTIMATE OF ROOT = ,E10.3)
00170 99998  FORMAT(25H0ERROR IN ZBRENT IER   = ,I2)
00180        END
00190        REAL FUNCTION FUN(X)
00200        REAL X
00210        IFAIL=0
00220        FUN=S15ACE(X,IFAIL)-0.75
00230        RETURN
00240        END
READY
fort newcomp4 ↵
G1 COMPILER ENTERED
SOURCE ANALYZED
PROGRAM NAME = MAIN
%  NO DIAGNOSTICS GENERATED
SOURCE ANALYZED
PROGRAM NAME = FUN
%  NO DIAGNOSTICS GENERATED
    %STATISTICS%  NO DIAGNOSTICS THIS STEP
READY
loadgo newcomp4.obj lib('sys1.libnags' 'sys1.libmaaxs') fortlib ↵
ESTIMATE OF ROOT = -0.674E+00
READY
```

A

B

C

In stage A a previously created data set is listed.
In stage B the program stored in the data set is compiled.
In stage C the load and execution of the program is performed.
(Note, in particular, the use of the LIB parameter with two libraries).

TSO HELP

M.Dowell

In Newsletter No. 42 (June 1980) we gave details of newly introduced TSO enhancements. From the beginning of October full details of these changes have been included in the batch version of the HELP information.
Users who maintain a copy of this listing are advised to obtain (if they have not already done so) a new version to replace their existing one.

A new version may be obtained by using the TSO command procedure LSTHELP.

An example of its use is given below:

```
lsthelp (CR)
DATA SET SLSSSTSH.SSSCSRSA.CNTL NOT IN CATALOG
SAVED
UTILITY DATA SET NOT FREED. IS NOT ALLOCATED
ACCOUNTING NUMBER = XXXXYYYY:
SPECIFY BOX NO. AND PROGRAMMER'S NAME.
..... (3 NUMERICS AND MAX. 16 ALPHANUMERICS)
aaabbbbbbbbbbbbbbb (CR)
YOUR JOB IS NAMED 'TSO ABCD ' AND HAS BEEN PASSED TO HASP.
READY
```

Note
All lines typed by the user are shown in small characters and terminated by the (CR) character to denote carriage return/enter.

The user must replace:

aaa - by the appropriate box number (3 numerics)

bbbbbbbbbbbbbbb - by some textual identification for the output listing of up to 16 alphanumeric characters

QED NOTES(2)

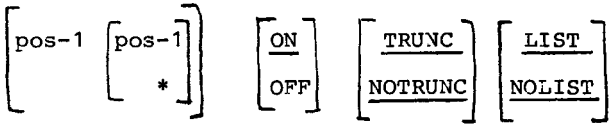
M. Dowell

Introduction

The FIELD subcommand of the QED Editor provides facilities which are not available in the normal EDIT system. It allows the user to specify that subsequent CHANGE, FIND and LIST subcommands will have their scope restricted to a specific field within the data records analysed. After a FIELD subcommand has been given the following points should be noted:

* Only character strings which lie entirely within the positions specified may be located and/or modified with CHANGE or FIND subcommands.
* Any position number specification in CHANGE or FIND subcommands will be regarded as relative to the start of the field (and not the entire record).
* Any truncation caused by a CHANGE subcommand will be at the end of the specified field and not at the end of the record (see the TRUNC/NOTRUNC operand of FIELD described in the following section for further details).
* The function of the LIST subcommand (including lines listed indirectly due to the VERIFY subcommand) depends on whether or not the LIST operand of the FIELD subcommand was given (see following section).

Full specification of the FIELD Subcommand



(for full details of the syntax notation see the IBM manual: TSO Command Language Reference [GC28-6732]).

Definition of Operands

pos-1
specifies the starting character position number for the field. If is omitted, then the field is assumed to start in the record and be of length pos-1. If neither 2 are specified then the pos-1 and pos-2 from the previous FIELD subcommand will remain in

pos-2
Specifies the ending character position number for the field (this includes pos-2). Note that for variable length records, the length of the records will be temporarily extended to pos-2 (if necessary) by padding with balnks.

*
Specifies that pos-2 should be at the end of the record.
Note that for variable length records the field will be variable in length and will include all characters beginning at (and including) pos-1 in each record. (In this case truncation will only occur if the maximum logical record length is exceeded).

ON
This specifies that this FIELD subcommand is to take effect now. (Note. This is the default if netiher ON or OFF is specified).

OFF
This specifies that this FIELD subcommand is not to take effect now.
A subsequent FIELD ON subcommand may then be used to activate field processing.
A FIELD OFF subcommand will suspend field processing without destroying any previously specified field options. Field processing may again be resumed by a FIELD ON subcommand.

TRUNC
Specifies that truncation of non-blanks (in the specified field) by a subsequent CHANGE subcommand is to be allowed.
Thus, if the specified field is expanded by the CHANGE subcommand, then the field will be truncated on the right. Data outside the field will remain unmodified. A warning will be given if the truncated data contains non-blank characters.
Note that the truncation of non-blanks will always be allowed even if TRUNC is not specified.
TRUNC is the original default value. Subsequent field command will use the previous TRUNC/NOTRUNC specification unless TRUNC or NOTRUNC is specifically given.

NOTRUNC
Specifies that the truncation of non-blanks in the specified field is not allowed.
An error message will be given if any attempt is made to do this.

LIST
Specifies that only positions within the field should be displayed on a subsequent LIST subcommand (or indirectly via a VERIFY subcommand).

NOLIST
Specifies that subsequent LIST commands do not consider any field specifications. This is the default on the first invocation of FIELD. Subsequent FIELD subcommands will leave the existing setting unchanged unless LIST or NOLIST is specifically given.

Examples of the FIELD Subcommand

1. FIELD 10 50

This specifies that subsequent CHANGE and FIND subcommands will only consider characters 10-50 (inclusive) of each record.

2. FIELD 5 *

This specifies that subsequent CHANGE and FIND subcommands will only consider characters from 5 to the end of each record.

3. FIELD 5

This specifies that subsequent CHANGE and FIND subcommands will only consider characters from 1-5 (inclusive) of each record.

4. FIELD 10 50 OFF

This specifies a field as in example 1 which is not yet to take effect.
The field may be put into effect by using a subsequent FIELD ON subcommand.

5. FIELD 5 * LIST

This specifies a field as in example 2. This field will also take effect for LIST (and VERIFY) information.

6. FIELD 5 NOTRUNC

This specifies a field as in example 3. This will not allow truncation of non-blanks in any subsequent CHANGE subcommand.

Use of the FIELD Subcommand

The FIELD subcommand has many possible applications. The following gives a simple (artificial) example of just one.

Given a data set with each record structured as follows:

```
cols  1- 2   House number
cols  4- 8   Colour of house door
cols 10-20   Name of occupant
cols 22-32   Occupation
cols 34-50   Area
```

It is necessary to change, for all of the records with house colour GREEN, that colour to VERDE. It is not possible to use the normal CHANGE facilities to do this as other occurrences of the same string may be found in other parts of the records (viz: MR. GREEN).
Therefore the change may be performed with the aid of a FIELD subcommand to define the part of each record to be examined, followed by the relevant CHANGE command.

```
qed file.data
QED
l
00010 11 GREEN  MRS. BUNN    BAKER      GREENLAND
00020 16 BLUE   MR.  GREEN   BUTCHER    USA
00030 83 RED    MR.  PLOD    POLICEMAN  GREENWICH
00040 21 GREEN  MR.  JONES   CHEMIST    ITALY
END OF DATA
field 4 8
change 10 40 /green/verde/ all
l
00010 11 VERDE  MRS. BUNN    BAKER      GREENLAND
00020 16 BLUE   MR.  GREEN   BUTCHER    USA
00030 83 RED    MR.  PLOD    POLICEMAN  GREENWICH
00040 21 VERDE  MR.  JONES   CHEMIST    ITALY
END OF DATA
```

Note. Only the occurrences of the text GREEN in the appropriate columns has been found.

This trivial example may obviously be expanded to many more practical cases where one wishes to restrict the scope of CHANGE, FIND, or LIST subcommands to specific character positions of each record.

STATISTICS OF COMPUTING INSTALLATION UTILIZATION
REPORT OF COMPUTING INSTALLATION EXPLOITATION
FOR THE MONTH OF SEPTEMBER 1980.

	YEAR 1979	YEAR 1980
General		
Number of working days	20 d	22 d
Work hours from 8.00 to 24.00 for	16.00h	16.00h
Duration of scheduled maintenance	18.18h	26.00h
Duration of unexpected maintenance	18.35h	18.17h
Total maintenance time	36.53h	44.17h
Total exploitation time	283.47h	307.83h
CPU time in problem mode	130.27h	278.63h*
Batch Processing		
Number of jobs	6369	8026
Number of cards input	1154800	1355300
Number of lines printed	18836000	28091000
Number of cards punched	70800	231000
CPU time	114.14h	247.95h*
Number of I/O (Disk)	19876000	25155000
Number of I/O (Magnetic tape)	3236000	3866000
T.S.O		
Number of LOGON's	2487	4403
Number of messages sent by terminals	156243	319500
Number of messages received by terminals	861517	2114500
CPU time	14.11h	29.17h*
Number of I/O (Disk)	2364000	4309250
Connect time	1709.99h	2976.68h
IMS		
Total time service is available	132.14h	119.25h
CPU time	2.02h	1.51h
Number of I/O (Disk)	655900	523800

* Real CPU has been multiplied by a factor of 2 to indicate the increased throughput of the Amdahl.

UTILISATION OF COMPUTING CENTRE BY OBJECTIVES & APPROPRIATION ACCOUNTS FOR THE MONTH OF SEPTEMBER 1980.

		AMDAHL 470/V7A equivalent time in hours
33001	Reactor Safety	300.10
33002	Plutonium Fuel and Actinide Research	1.56
33003	Nuclear Materials	13.12
33004	Safeguards	9.19
33011	Solar Energy	0.11
33012	HYDROGEN	0.27
33013	Design Studies on Thermonuclear Fusion	24.30
33021	Environment and Resources	23.64
33030	METRE	1.33
33041	Informatics	81.38
33044	Training	-
33046	Support to the Commission	3.21
33300	ESSOR	25.64
1.20.1 1.20.2	General Administration - JRC General Services - Administration - Ispra	69.73
1.20.3	General Services - Technical - Ispra	1.27
1.30.3	Central Workshop	1.57
33014/ 33312	High Temperature Materials	0.33
	TOTAL	556.75
1.94.0	Services to External Users	10.68
	TOTAL	567.43

BATCH PROCESSING DISTRIBUTED BY REQUESTED BY CORE MEMORY SIZE

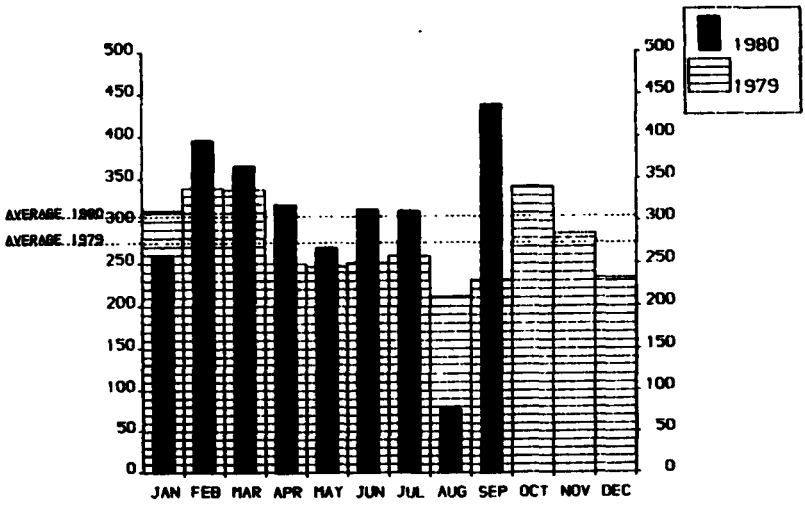
	100	200	300	400	600	800	1000	1200	1400	1400
No. of jobs	2431	2394	944	1039	334	150	119	80	76	-
Elapsed time	64	168	133	237	84	44	53	27	62	-
CPU time	1.7	40.3	30.8	71.5	34.2	9.4	29.1	12.8	16.7	-
"Equiv" time	19	71	57	127	45	17	34	17	29	-
"Turn" time	1.2	1.1	2.0	2.5	2.3	2.1	2.3	1.5	3.6	-
I/O (disk)	1803	4122	3603	7515	1515	1067	756	590	1769	-
I/O (tape)	1607	637	413	847	183	3	68	22	9	-

NOTE.
All times are in hours.
"Equiv" means equivalent.
"Turn" means turn around.
All I/O transfers are measured in 1000's.

PERCENTAGE OF JOBS FINISHED IN LESS THAN:

TIME	15mn	30mn	1hr	2hrs	4hrs	8hrs	1day	2day	3day	6day
%year 1979	35	52	66	80	93	99	100	100	100	100
%year 1980	33	50	65	76	87	94	96	100	100	100

HISTOGRAM OF TOTAL EQUIVALENT TIME(HRS)



Projected total for 1980 = 3662 hours(using average)

Total for 1979 was = 3292 hours

REFERENCES TO THE PERSONNEL/FUNCTIONS OF THE COMPUTING CENTRE

			Room	Tele.
Manager of The Computing Centre		J.Pire		
Responsible for User Registration	Ms.	G.Rambs		
Operations Sector				
Responsible for the Computer Room		A.Binda-Rossetti		
Substituted in case of abscence by:				
Responsible for Peripherals		G.Nocera		
Systems Group				
Responsible for the group		D.König		
Substituted in case of abscence by:		P.A.Moinil		
Responsible for TSO Registration		C.Daolio		
Informatics Support Sector				
Responsible for the Sector	(f.f.)	H.de Wolde	1883	1259
Secretary	Mrs.	G.Hudry	1873	787
Responsible for User Support		H.de Wolde	1883	1259
General Inf./Support Library	Mrs.	A.Cambon	1871	730
Advisory Service/List of Consultants (See Note 1)			1870	730

A.Inzaghi — A.A.Pollicini
H.I. de Wolde
R.Meelhuysen — M.Dowell

NOTE 1. The advisory service is available in the same room as the Computing Support Library (room 1870). Exact details of the advisory service times for a specific week can be found at the head of any output listing (for that week).
Any informatics problem may be raised. However, the service is not designed to help users with problems which are their sole responsibility. For example, debugging of the logic of programs and requests for information which can easily be retrieved from available documentation.
If necessary, other competent personnel from the informatics division may be contacted by the consultant but not directly by the users.
The users should only contact the person who is the consultant for that specific day and only during the specified hours.
Outside the specified hours general information may be requested from Mrs. A. Cambon in the Computing Support Library.

HOW TO OBTAIN COMPUTING CENTRE DOCUMENTATION

Persons interested in receiving copies of the Computing Centre "green books" or in receiving regularly the "Computing Centre Newsletter" are requested to complete the appropriate part of the following form and send it to :-

Ms. A. Cambon

Support To Computing

Building 36

Tel. 730.

Indicate with a (✓) which options are required.

Please add my name to Newsletter mailing list	( )
Please send me copies of the following "green books":	
JRC-TSO Primer	( )
GRAPHIT	( )
Towards a New Programming Style	( )
LIBRARIAN	( )

NAME .....

ADDRESS .....

.....

.....

TELEPHONE .....