SELECT, EXTRACT, SETUP:

A SET OF COMPUTER PROGRAMS FOR SEARCHING AND MODIFYING LOCAL EARTHQUAKE DATA

By J. ALAN STEPPE

U.S. GEOLOGICAL SURVEY

Open-File Report No. 76-342

This report is preliminary and has not been edited or reviewed for conformity with Geological Survey standards and nomenclature.

Menlo Park, California
January 1976

CONTENTS

SELECT, EXTRACT, SETUP: A SET OF COMPUTER PROGRAMS FOR SEARCHING AND MODIFYING LOCAL EARTHQUAKE DATA

By J. ALAN STEPPE

INTRODUCTION

The system of three computer programs described in this report is intended for processing local earthquake data. The programs are designed to be used either separately or in various combinations within a single computer run. SELECT is a program for searching a list of hypocenters and selecting a subset of that list. EXTRACT is used for extracting the phase data for particular events from a larger collection of phase data. SETUP is a program for modifying lists of phase data, and/or for organizing data for input to an earthquake location program.

This report is intended primarily as a manual for users; the emphasis is on how to use the programs. It assumes that the user has a working knowledge of at least one of the earthquake location programs HYPO71 and HYPOELLIPSE.

The programs SELECT, EXTRACT, and SETUP are written in FORTRAN IV. They have been compiled and executed on the computer system at Lawrence Berkeley Laboratory (also known as BKY). The examples in this writeup pertain to the BKY system.

Users experiencing problems with these programs should contact the author. Comments and suggestions for improvements are also welcome and should be addressed to:

J. Alan Steppe
U.S. Geological Survey
345 Middlefield Road
Menlo Park, California 94025

How to use SELECT

Introduction

SELECT is a program for searching a list of hypocenter summary cards and selecting a subset of that list. The subset selected is defined by intervals of time, magnitude, depth, quality (A, B, C, or D), RMS time residual, and number of readings, and by a polygon in latitude and longitude.

The summary cards to be searched may be in either HYPO71 format or HYPOELLIPSE "error ellipse data" format. SELECT runs on the CDC 7600 computer at BKY (Lawrence Berkeley Laboratory).

Files Used

SELECT expects to find the hypocenter summary card list on a file named QUAKES. It reads search parameters from INPUT and writes them, together with the number of events selected, on OUTPUT. The summary cards selected are written to a file named ORIGNS. The selected events are not automatically printed or punched, but this can easily be done with control cards. Having the selected summary cards on a disk file (ORIGNS) makes it easy to use them in a variety of ways. They may be written on magnetic tape, stored on the data cell (PSS), or read by another program within the same job, as well as printed or punched.

Input Data

The input cards needed to specify the search parameters are described below. Each of the intervals (time, magnitude, depth, quality, RMS time residual, and number of readings) is defined by a lower limit and an upper limit. The intervals are closed intervals. Any or all of the fields defining the intervals (Cards 1, 2, and 3 below) may be left blank and default values will be used. The defaults are:

time:	beginning of Jan. 1, 1900	thru	beginning of Dec. 32, 1999	
magnitude:	-1000.	thru	9.9	
depth:	0.	thru	6371.	(km)
quality:	D	thru	A	
RMS time residual:	0.	thru	100.	(sec)
number of readings:	0	thru	999	

The label (on Card 4 below) is optional; it is printed on the output and is conveniently used to give a name to the epicentral region. The polygon in latitude and longitude which specifies the epicentral region is defined by its vertices, given in order, either clockwise or counterclockwise.

In the examples shown below a lower case letter b denotes a blank space.

Card 1: Time Range Card

Column	Format	Entry	Example
1 - 6	3I2	Year, month, and day of lower time limit	720224
8 - 11	2I2	Hour and minute of lower time limit	1556
13 - 14	I2	second of lower time limit	55
19 - 24	3I2	Year, month, and day of upper time limit	720904
26 - 29	2I2	Hour and minute of upper time limit	1804
31 - 32	I2	second of upper time limit	30

Card 2: Magnitude Range and Depth Range Card

1 - 6	F6.2	lower magnitude limit	bb2.00
7 - 12	F6.2	upper magnitude limit	bb5.00
17 - 23	F7.2	shallower depth limit	bbb5.70
24 - 30	F7.2	deeper depth limit	bbb7.50

Card 3: Quality Range Card

1	A1	worse quality limit	C
2	A1	better quality limit	A
7 - 11	F5.2	lower limit of RMS time residual	b0.00
12 - 16	F5.2	upper limit of RMS time residual	b1.00
21 - 24	I4	lower limit of number of readings	bb15

Card 3: Quality Range Card (cont'd)

Column	Format	Entry	Example
25 - 28	I4	upper limit of number of readings	b100

Card 4: Number of Vertices Card

1 - 2	I2	number of vertices of epicentral polygon	b4
3 - 80	A8, 7A10	any label	bbBEARbVALLEY

Card 5+: Vertex of epicentral region card
one card per vertex (in order)

1 - 3	I3	degrees of latitude	b36
5 - 9	F5.2	minutes of latitude	30.00
10 - 13	I4	degrees of longitude	b121
15 - 19	F5.2	minutes of longitude	15.00

Multiple Regions

A second subset of the hypocenter list can be selected by simply placing another set of cards (1 thru 5+) describing the second subset after those for the first subset. A third set of cards will give a third subset, and so forth. The summary cards for the different subsets are separated on the output file (ORIGNS) by file marks, one file mark after each subset.

Format Control

SELECT normally expects the hypocenter summary card list to be searched to be in chronological order. However, this is not necessary if the time interval given on the time range card covers the times of all the events to be searched; i.e., if there is to be no selection based on time. If the events on QUAKES are in chronological order, then the events in each subset on ORIGNS will be in chronological order.

The summary cards are normally in HYPO71 format, but HYPOELLIPSE "error ellipse data" format summary cards can be used by punching an E in column 50 of the first card; i.e., the time range card for the first subset.

The summary card lists on QUAKES and ORIGNS may have either of two structures. One is an ordinary card image file (packed display code). The other is a "Blocked" form consisting of a series of logical records each of which contains 34 card images of 80 columns each (2720 characters per logical record). This Blocked form is often used for storing summary cards on the data cell since it is somewhat more economical of data cell space than the ordinary card image form.

SELECT normally expects the data on QUAKES to be in the Blocked form, and puts the selected events onto ORIGNS in ordinary card image form. The form expected on QUAKES can be changed to card images by punching the word CARDS in columns 41 - 45 of the first card (the time range card for the first subset). The form written on ORIGNS can be changed to Blocked by punching the word BLOCKED in columns 55 - 61 of the first card.

There should be no file marks within the hypocenter list to be searched (on QUAKES). A standard HYPO71 summary card list header card, DATE ORIGIN etc., will be ignored by SELECT if it appears at the beginning of the hypocenter list and if QUAKES has the card image form. This makes it possible to run SELECT on summary cards (images) produced by HYPO71 in the same job.

SELECT rewinds the file QUAKES before beginning to search, and leaves ORIGNS in a rewound condition at the end of execution.

Common Errors

The most common errors in the use of SELECT are (1) incorrectly determining the vertices of the epicentral region, and (2) misspelling the file name ORIGNS (it has only one letter I). Other common errors are failure to account for file positioning and for the number of files on ORIGNS.

The Lawrence Berkeley Laboratory computer center documentation stored in PSS library HANDBOOK, subset COPY, sections 1.1 thru 1.3, is recommended reading.

Getting the Program

The program SELECT (in the form of relocatable object code) is stored on the data cell (PSS) at LBL, in library QUAKEPRCSR, subset SELECT.

Examples

To illustrate the use of SELECT it is assumed in the following examples that hypocenter summary card data for earthquake locations in central California have been stored on the data cell in a library named USGSEQ subsets NCAL70, NCAL71, and NCAL72, with one year's data in each subset. These are HYPO71 summary cards stored in the Blocked form.

The first example will list all earthquakes in the Hollister quad (36°45' to 37° and 121°15' to 121°30') in 1972.

```
nam,7,63,70000.account,name
FETCHPS(QUAKEPRCSR,SELECT,SELECT)
FETCHPS(USGSEQ,QUAKES,NCAL72)
LINK(F=SELECT,P=BKYIO,L=0,X)
COPYSBF(ORIGNS,OUTPUT)
7/8/9 card
blank card
blank card
blank card
b4           HOLLISTER QUAD
b36b45.00b121b15.00
b36b45.00b121b30.00
b37b00.00b121b30.00
b37b00.00b121b15.00
6/7/8/9   card
```

The next example will list all A quality events with at least 20 readings in the Hollister quad for November 1970 thru February 1971, and pass their summary cards to a user's program.

```
nam,7,200,170000.account,name

MNF.                                     compile user's program

FETCHPS(QUAKEPRCSR,SELECT,SELECT)

FETCHPS(USGSEQ,QUAKES,NCAL70/F,NCAL71)

LINK(F=SELECT,P=BKYIO,FL=0,L=0,X)        execute select

COPYSBF(ORIGNS,OUTPUT)

REWIND(ORIGNS)

LINK,X.                                  execute user's program

7/8/9   card

      PROGRAM USERS(INPUT,OUTPUT,ORIGNS,TAPE1=ORIGNS)
         .
         .
         .
10    READ (1,1000) ···                  read a summary card

1000  FORMAT ( ··· )

      IF( EOF (1) ) 30,20,30             test for end of file on ORIGNS

20    ···                                come here if no end of file
       .
       .
       .

      GO  TO  10

30    ···                                come here if end of file was read
       .
       .
       .
      STOP

      END

7/8/9   card
```

```
701101b0000b00bbbb710228b2400

blank card

AAbbbbbbbbbbbbbbbbbbbb20

b4          HOLLISTER  QUAD

b36b45.00b121b15.00

b36b45.00b121b30.00

b37b00.00b121b30.00

b37b00.00b121b15.00

7/8/9   card

                                        input data for user's program

6/7/8/9   card
```

The next example will list and punch copies of those summary cards from an input deck which have quality A or B and magnitude at least 2.45 in the time period December 15, 1971 thru January 15, 1972. Output is produced at BKY.

```
nam,7,63,70000.account,name
*LOCAL
*NOSTAGE
FETCHPS(QUAKEPRCSR,SELECT,SELECT)
COPY,INPUT,1R,QUAKES.
LINK(F=SELECT,P=BKYIO,L=0,X)
COPYSBF(ORIGNS,OUTPUT)
COPY,ORIGNS/RR,PUNCH.
7/8/9   card
summary cards to be searched, in chronological order
7/8/9   card
711215b0000b00bbbb720115b2400b00bbbbbbbCARDS
bb2.45
BA
b4
b00b00.00b000b00.00
b00b00.00b180b00.00
b90b00.00b180b00.00
b90b00.00b000b00.00
6/7/8/9   card
```

The next example will list all central California earthquakes not in the Hollister quad on January 2 and 3, 1972.

```
nam,7,63,70000.account,name

FETCHPS(QUAKEPRCSR,SELECT,SELECT)

FETCHPS(USGSEQ,QUAKES,NCAL72)

LINK(F=SELECT,P=BKYIO,L=0,X)

COPYSBF(ORIGNS,OUTPUT)

7/8/9   card

720102b0000b00bbbb720103b2400

blank card

blank card

10               NOT IN HOLLISTER QUAD

b00b00.00b000b00.00

b00b00.00b180b00.00

b90b00.00b180b00.00

b90b00.00b000b00.00

b37b00.00b121b15.00

b37b00.00b121b30.00

b36b45.00b121b30.00

b36b45.00b121b15.00

b37b00.00b121b15.00

b90b00.00b000b00.00

6/7/8/9   card
```

Technical Notes

When a field on a summary card that normally contains numeric data is left blank, it will be treated as having the value zero. Thus, for example, an event with no magnitude given will be treated as if it had a magnitude of 0.00. A blank quality will not be included in the range D thru A.

In any one run, all points (both vertices and epicenters) must be in the same quadrant of the earth.

The number of vertices of the epicentral region must be between 3 and 99 inclusive.

The epicentral region may be either a convex or a non-convex polygon. If it is non-convex, SELECT will print a message to that effect; this does not indicate an error, unless the region was intended to be convex.

It is possible to give a list of vertices which does not define a polygon (the boundary of the region may cross over itself). SELECT detects some but not all such errors. It detects those cases for which the total turning angle of the tangent vector is not $\pm 360^{\circ}$, and prints the message WARNING, THE BOUNDARY OF THIS REGION IS NOT A SIMPLE CLOSED CURVE.

The names of the files expected by SELECT appear in its PROGRAM card in the following order: INPUT, OUTPUT, QUAKES, ORIGNS.

SELECT uses subroutines from the system subroutine library BKYIO.

If SELECT prints the message ERROR ITIMEL TOO LARGE OR BLANK CARD ENDS BLOCK, it means that either the lower time limit is later than the time of the last event on QUAKES or there is an extraneous blank card on QUAKES (at the end of a block if QUAKES has the Blocked form).

On a flat earth where latitude and longitude form a Cartesian coordinate system, a "polygon in latitude and longitude" is simply an ordinary polygon. However, if the region of interest is near the North or South Pole, or is a global rather than a local region, then one must be concerned about the curvature of the earth. On the (nearly spherical) surface of the real earth, the shape of a "polygon in latitude and longitude" is not so easy to interpret. It is, however, a simple matter to specify in a parametric form the portion of the boundary of the epicentral region lying between two successive vertices. Let LATV1 and LATV2 be the latitudes of the two vertices and let LONV1 and LONV2 be their longitudes. Let LAT(t) and LON(t) be the latitude and longitude of a point on the boundary. Then

$$\mathrm{LAT}(t) = (\mathrm{LATV1})(1-t) + (\mathrm{LATV2})\ t$$

$$\mathrm{LON}(t) = (\mathrm{LONV1})(1-t) + (\mathrm{LONV2})\ t$$

where the parameter t assumes values in the interval 0 to 1.

How to use EXTRACT

Introduction

EXTRACT is a program for extracting the phase lists of events with given origin times from a larger group of phase lists. Summary cards from HYPO71 or HYPOELLIPSE may be used to specify the origin times.

EXTRACT runs on the CDC 7600 at BKY (Lawrence Berkeley Laboratory).

Files Used

EXTRACT expects to find the origin times of the desired events on a file named ORIGNS. It reads the larger group of phase lists from a file ALLFAZ, and writes the extracted phase lists for the desired events onto a file SUBFAZ. EXTRACT writes messages describing its progress to the OUTPUT file. It does not read anything from INPUT.

Formats

ORIGNS, ALLFAZ, and SUBFAZ are ordinary card image files (packed display code).

The origin times on ORIGNS should be given one event per card. The Year, Month, Day, Hour, Minute, and Second should be in that order on each card. Normally the time is read in format 3I2, 1X, 2I2, 1X, I2 However, if the first card on ORIGNS has a letter N or a letter S in column 17, the format used will be 6I2. This means that summary cards in either HYPO71 format or HYPOELLIPSE "error ellipse data" format may be used to give the origin times. Within a single run of EXTRACT, the origin times must all have the same format.

The origin times should generally be given in chronological order. However, if ORIGNS is broken up with file marks, the times need be in chronological order only within each (logical) file, i.e., between file marks. EXTRACT will read ORIGNS until it encounters a double-end-of-file-mark (an empty file after the first) or an EOI (an end of information on the named file ORIGNS).

The data on ALLFAZ consists primarily of phase cards (arrival time cards) in the format used by HYPO71 and HYPOELLIPSE. The phase cards for a particular event may be in any order, and should be followed by a single instruction card (blank in columns 1-4). The events on ALLFAZ must be in chronological order. Calibration cards, identified by the letters CAL in columns 63-65, may occur between events. ALLFAZ should consist of a single logical file (no file marks within the data).

EXTRACT writes onto SUBFAZ the phase cards from ALLFAZ corresponding to the origin time on ORIGNS, and the instruction card following those phase cards on ALLFAZ. The events will appear on SUBFAZ in the same order that the origin times appear on ORIGNS. SUBFAZ will be a single logical file.

EXTRACT transfers from ALLFAZ to SUBFAZ any calibration cards it encounters in its search procedure. All calibration cards preceding a particular event on ALLFAZ will appear at least once, before that event on SUBFAZ. There is another version of EXTRACT, called NXTRACT, which deletes all calibration cards.

EXTRACT rewinds ORIGNS before beginning its processing, and leaves SUBFAZ in a rewound condition at the end of execution.

Design Limitations

EXTRACT works by matching the arrival times on the phase cards against the given origin times. It extracts the first event on ALLFAZ which has an arrival time on its first (not 4-weighted) phase card that is slightly later than the given origin time. Thus the given origin time must be earlier than the arrival times of the desired event. It should not be more than about 60 seconds earlier. Of course, if there is an event on ALLFAZ with its arrival times between the given origin time and the arrival times of the desired event, it will be extracted rather than the desired event. In other words, if there are events on ALLFAZ within seconds of each other, EXTRACT may choose the wrong event; the likelihood of this problem occurring depends on the accuracy of the origin times. In the author's experience with local earthquake data, this problem has been extremely rare. EXTRACT is not intended for use on teleseismic data. The user is advised to check the output carefully.

When EXTRACT encounters a file mark on ORIGNS, it reads one more origin time and compares that time with the time of the last event that it read on ALLFAZ. If the origin time is later than this "current position on ALLFAZ time" EXTRACT looks on down ALLFAZ for the desired event; if not, it rewinds ALLFAZ and begins again. When this happens, calibration cards on the first part of ALLFAZ will get copied to SUBFAZ again. Thus the calibration cards on SUBFAZ will be in the same order that they appear on ALLFAZ (normally chronologically) within groups corresponding to the logical files on ORIGNS.

Error Messages

When EXTRACT is operating normally, it prints a one line message for each event found giving part of the origin time card (summary card) and one phase card from the event extracted. EXTRACT checks for a number of error conditions; when one is detected it prints a self-explanatory error message.

EXTRACT attempts to produce correct results in spite of infrequent random errors in the phase cards on ALLFAZ. In particular, it should never abort. However, EXTRACT cannot detect all possible errors, and so may give erroneous results in the presence of errors in the input data. The user should carefully check the input, output, and especially the printed messages from EXTRACT.

Getting the Program

The program EXTRACT (in the form of relocatable object code) is stored on the data cell (PSS) at LBL, in library QUAKEPRCSR, subset EXTRACT. The program NXTRACT is in subset NXTRACT.

File Substitution

The names of the files expected by EXTRACT appear in its PROGRAM card in the following order: ORIGNS, ALLFAZ, OUTPUT, SUBFAZ. The names of the files can be changed by file substitution.

Examples

To illustrate the use of EXTRACT it is assumed in the following examples that there is a list of phase data for many events on tape number 12345 (in packed display code; i.e., a binary copy of an ordinary card image file). It is assumed that the particular events desired in the examples are among those on this tape.

In these examples a lower case letter b denotes a blank space. The control card "FBSIZE,ALLFAZ=176." is included to reduce the computer charges due to LCM bufferloads associated with the file ALLFAZ.

The first example will list and punch phase cards for two events with origin times 720927 1017 14 and 721103 1945 36. Calibration cards will not be included.

```
nam,7,200,70000.account,name
*LOCAL
FBSIZE,ALLFAZ=176.
STAGE,ALLFAZ,12345.
FETCHPS,QUAKEPRCSR,EXTRACT,NXTRACT.
COPY,INPUT,1R,ORIGNS.
LINK,F=EXTRACT,L=0,X.
COPYSBF,SUBFAZ,OUTPUT.
COPY,SUBFAZ/RR,PUNCH.
7/8/9   card
720927b1017b14
721103b1945b36
6/7/8/9 card
```

The next example will save on tape number 67890 the phase cards corresponding to a group of summary cards. Calibration cards will be included.

```
nam,7,200,70000.account,name
FBSIZE,ALLFAZ=176.
STAGE,ALLFAZ,12345.
FETCHPS,QUAKEPRCSR,EXTRACT,EXTRACT.
COPY,INPUT,1R,ORIGNS.
LINK,F=EXTRACT,L=0,X.
STAGE,SUBFAZ,W,67890.
7/8/9  card
       summary cards in chronological order
6/7/8/9  card
```

How to use SETUP

Introduction

SETUP is a program for modifying phase lists and/or setting up an input deck for HYPO71 or HYPOELLIPSE, from input cards and a separate file of phase lists. Several kinds of modifications to the phase lists can be performed under the control of special directive cards included among the input cards.

SETUP runs on the CDC 7600 computer at BKY (Lawrence Berkeley Laboratory).

Files Used

SETUP reads special directives and other input cards from the INPUT file. It may read one or more station lists from a file named STATNS. Phase lists are taken from a file named SUBFAZ. The HYPO71 or HYPOELLIPSE input deck constructed by SETUP is written to a file HYPOIN. SETUP writes messages describing its progress to the OUTPUT file.

Formats

INPUT, SUBFAZ, STATNS, and HYPOIN are all ordinary card image files (packed display code).

The data on SUBFAZ consists primarily of phase cards (arrival time cards) in the format used by HYPO71 and HYPOELLIPSE. The phase cards for a particular event may be in any order, and should be followed by a single instructioa card (blank in columns 10-11). The events on SUBFAZ may be in any order. Calibration cards, identified by the letters CAL in columns 63-65, may occur between events. SUBFAZ should consist of a single logical file (no file marks within the data).

The file named STATNS is not always required. If it is used, it may contain a number of logical files. Each of these should contain one selection card and a station list. The first card in each logical file should be the selection card (a blank card, a card with a 1 in column 1 and otherwise blank, or a card reading BEGIN STATION LIST). It should correspond to the following station list in the manner required by HYPO71 or HYPOELLIPSE. Thus each station list will be followed by an end-of-file mark or an end-of-information on STATNS. The blank card which must follow the station list in an input deck for HYPO71, or the END card for HYPOELLIPSE, should not be put on STATNS; it will generally be an input card to SETUP.

The phase lists copied from SUBFAZ to HYPOIN will occur on HYPOIN in the same order that they had on SUBFAZ. HYPOIN will be a single logical file, and is left in a rewound condition at the end of execution.

Directives

Basically what SETUP does is copy cards from INPUT to HYPOIN, taking other actions whenever a special directive card is read. These actions may include copying phase lists, possibly modified, from SUBFAZ to HYPOIN, or copying a station list from STATNS to HYPOIN.

Roughly speaking, the deck read from INPUT by SETUP is copied to HYPOIN with each directive replaced by a block of data cards taken from SUBFAZ or STATNS. The words which constitute the directive roughly describe the block of data which will replace it. Thus the input deck to SETUP is a "nearly-plain-English" description of what will be written on HYPOIN.

Again roughly speaking, any input deck to SETUP will be a valid input deck if it seems to make sense, given the semantics of the directives as described in the next section. A precise description of what constitutes a valid SETUP input deck is given below under the heading "Syntax of the SETUP Input Deck".

There are eight SETUP directives. In the following list a lower case letter b denotes a blank space and nn and mmmm denote positive integers in formats I2 and I4 respectively. All the directives begin in column 1.

STATIONbLIST

PHASEbLIST

PHASEbLIST,bREPLACEbnn

PHASEbLIST,bREPEATbbmmmm

ALLbPHASES

ALLbSTATIONS

ADDbSTATIONSbnn

INSTRUCTIONbCARD

Semantics of the SETUP Directives

STATION LIST

One logical file, consisting of a selection card and a station list, is copied from the file STATNS to HYPOIN. The end-of-file mark is not transferred.

PHASE LIST

The phase list for one event is copied from SUBFAZ to HYPOIN; any calibration cards preceding the phase list are also copied. The instruction card following the phase list on SUBFAZ is not copied.

PHASE LIST, REPLACE nn

Exactly nn "replacement" cards are copied from INPUT to HYPOIN. Then the next phast list on SUBFAZ is copied to HYPOIN, deleting each phase list card having the same station name as any of the "replacement" cards. Columns 1-4 on each "replacement" card are taken as a station name. Calibration cards preceding the phase list are copied, and the following instruction card is not.

PHASE LIST, REPEAT mmmm

The phase lists of the next mmmm events on SUBFAZ are copied to HYPOIN. Calibration cards preceding any of these events are also copied. The instruction cards following these events are also copied if the PHASE LIST, REPEAT directive is followed by an INSTRUCTION CARD directive (after the ALL STATIONS directive or ADD STATIONS directive and names, if one is used.) Otherwise, each phase list will be followed on HYPOIN by a copy of the next card after the PHASE LIST, REPEAT directive on INPUT (and, again, after the ALL STATIONS or ADD STATIONS directive and names, if used.)

ALL PHASES

All the phase lists remaining on SUBFAZ are copied to HYPOIN. Any calibration cards preceding each phase list are copied. The following instruction card is also copied if the ALL PHASES directive is followed by an INSTRUCTION CARD directive (after the ALL STATIONS or ADD STATIONS directive and names, if used.) Otherwise, all cards on INPUT after the ALL PHASES directive (and after any ALL STATIONS or ADD STATIONS directive and names) (and before an end-of-file mark) are considered to be an instruction list, and this list is placed after each phase list.

ALL STATIONS

Each phase list copied to HYPOIN by the preceding PHASE LIST, REPEAT or ALL PHASES directive, is compared with the station list most recently copied to HYPOIN with a STATION LIST directive. Any phase list card for a station which is not on that station list is deleted. For each station for which there is no phase card in the original phase list, a phase card is inserted with a P- weight (column 8) of 4 and time (to minutes) equal to that of the first card in the phase list.

ADD STATIONS nn

The next nn cards are read from INPUT and the first 4 characters (columns) of each are taken as the name of a station to be added. For each of these stations, a phase card with a P-weight of 4 and time (to minutes) equal to that of the first card in the original phase list, is inserted at the beginning of each phase list copied to HYPOIN by the preceding PHASE LIST, REPEAT or ALL PHASES directive.

INSTRUCTION CARD

The instruction card most recently read from SUBFAZ is written to HYPOIN. If no instruction card has yet been read from SUBFAZ, a blank card will be written. When used after a PHASE LIST, REPEAT or ALL PHASES directive, each phase list copied is followed on HYPOIN by the instruction card which followed that phase list on SUBFAZ.

Getting the Program

The program SETUP (in the form of relocatable object code) is stored on the data cell (PSS) at LBL, in library QUAKEPRCSR, subset SETUP.

Examples

Suppose there is a list of phase data for many events on tape number 67890 (in packed display code; i.e., a binary copy of an ordinary card image file). The following example will replace the instruction card following each event with a blank card and save the result on tape number 54321.

```
nam,7,63,70000.account,name
STAGE,SUBFAZ,67890.
FETCHPS,QUAKEPRCSR,SETUP,SETUP.
LINK,F=SETUP,L=0,X.
STAGE,HYPOIN,W,54321.
7/8/9     card
ALLbPHASES
blank card
6/7/8/9   card
```

The following example will use HYPO71 to locate the events with phase lists on tape number 67890. For the first 27 events phase cards will be added for the stations bBVL and bEKH. For the rest of the events the phase list will be modified to have exactly one phase card for each station on the station list (assuming at most one in the original list).

```
nam,7,200,145000.account,name
STAGE,SUBFAZ,67890.
FETCHPS,QUAKEPRCSR,SETUP,SETUP.
COPY,INPUT,1R,STATNS/RR.
LINK,F=SETUP,L=0,X.
RETURN,SUBFAZ,SETUP,STATNS.
FETCHPS,UPGEO,BIN,RHP71.
LINK,F=BIN,L=0,B.
RETURN,BIN.
LGOB,HYPOIN.
7/8/9     card
selection card
station list
7/8/9     card
STATIONbLIST
blank card
bb4.0bbbb0.0
bb5.9bbbb3.5
bb6.8bbb15.0
bb8.05bb25.0
blank card
bbb5.bb50.b100.b1.78bbbb2
```

```
PHASEbLIST,bREPEATbb0027
ADDbSTATIONSb02
bBVL
bEKH
INSTRUCTIONbCARD
ALLbPHASES
ALLbSTATIONS
INSTRUCTIONbCARD
6/7/8/9    card
```

Technical Notes

The names of the files expected by SETUP appear in its PROGRAM card in the following order: INPUT, OUTPUT, SUBFAZ, HYPOIN, STATNS. The names of the files can be changed by file substitution.

The year number (columns 10-11) on phase cards read from SUBFAZ should not be zero.

The length of an instruction list following an ALL PHASES directive is limited to a maximum of 25 cards. The list of stations to be added following an ADD STATIONS directive is limited to 99 cards. The maximum number of replacement phase cards following a PHASE LIST, REPLACE directive is 99. Each station list read from STATNS is limited to a maximum of 301 stations.

Syntax of the SETUP Input Deck

This section gives a precise description of what constitutes a valid SETUP input deck. The description is given in a special "language" (or "meta-language") which is commonly used for specifying computer programming languages. This meta-language is essentially Backus Normal Form (or BNF), which was devised by Backus (1959, 1963) for describing the ALGOL language. BNF was originally applied to strings of characters; the form used here is a variant of BNF which applies to strings of card images.

The sequence of cards in a SETUP input deck is described below in a metalinguistic form. The interpretation of this form is according to the following rules:

1. The symbol ≡ has the meaning is defined as.

2. The symbol | has the meaning or.

3. A phrase enclosed in angle brackets is a <metalinguistic variable> which is defined by the enclosed phrase or is explicitly defined elsewhere, and is the name of a class of strings of SETUP input cards.

4. Juxtaposition of metalinguistic variables or constants (SETUP directives) in a formula signifies that the specified cards follow one another in a string of cards.

Note the following consequences of these rules:

a. A statement such as <list> ≡ <free card>|<list><free card> can be read as "a <list> is defined as a <free card>, or a <list> followed by a <free card>."

b. Definitions may be recursive. In the preceding example, one can break off a <free card> from the end of a <list> and have either a <list> or the empty string left; hence a <list> is just a non-empty string of <free card>'s.

The Syntax of the SETUP input deck is as follows:

<input deck> ≡ <deck> | <deck> <all clause>

<deck> ≡ <empty deck> | <transfer clause> | <deck> <transfer clause>

<transfer clause> ≡ STATION LIST | INSTRUCTION CARD | PHASE LIST |
<replace clause> | <repeat clause> | <free card>

<replace clause> ≡ PHASE LIST, REPLACE nn <nn replacement cards>

<repeat clause> ≡ PHASE LIST, REPEAT mmmm <add clause> <instruction phrase>

<add clause> ≡ <empty deck> | ALL STATIONS |
ADD STATIONS nn <nn station name cards>

<instruction phrase> ≡ INSTRUCTION CARD | <free card>

<all clause> ≡ ALL PHASES <add clause> <instruction clause>

<instruction clause> ≡ INSTRUCTION CARD | <list>

<list> ≡ <free card> | <list> <free card>

<free card> ≡ <any one card not a directive>

Combining the Programs

It is possible to run more than one of the programs SELECT, EXTRACT, and SETUP in a single job. In fact, they were written with this use in mind. When several programs run in the same job they can communicate through disk files. As each program runs, it writes results onto a disk file which is read by other programs later in the job. These files can also be manipulated by using system routines.

The programs can be combined in many different ways, depending on the job control cards used. Thus to effectively use the programs in combination, the user needs to have a fairly good understanding of control cards. At BKY (Lawrence Berkeley Laboratory) the documentation on PSS library HANDBOOK, subsets COPY and STAGING is particularly recommended; subsets CONTROL, LOADING, and STORAGE may also be helpful.

The file names expected by the programs were chosen for convenience in writing the control cards needed for certain commonly used combinations. For other combinations it may be necessary to change the file names, say by file substitution or by renaming. Note that each of the programs leaves its results file in a rewound condition when it finishes execution.

When these programs are used on a large volume of data, it is often convenient to store the data on magnetic tape, particularly the phase data. If the data on tape is subdivided into smaller segments by record and file marks, it is possible to select only those segments which are actually needed, when the data is transferred from the tape to the computer. This leads to greater efficiency in the operation of the programs, particularly EXTRACT, since they will then not process large amounts of irrelevant data. The control card used at BKY to access certain segments of the data on a tape is discussed in subset STAGE of the PSS library WRITEUPS.

Example

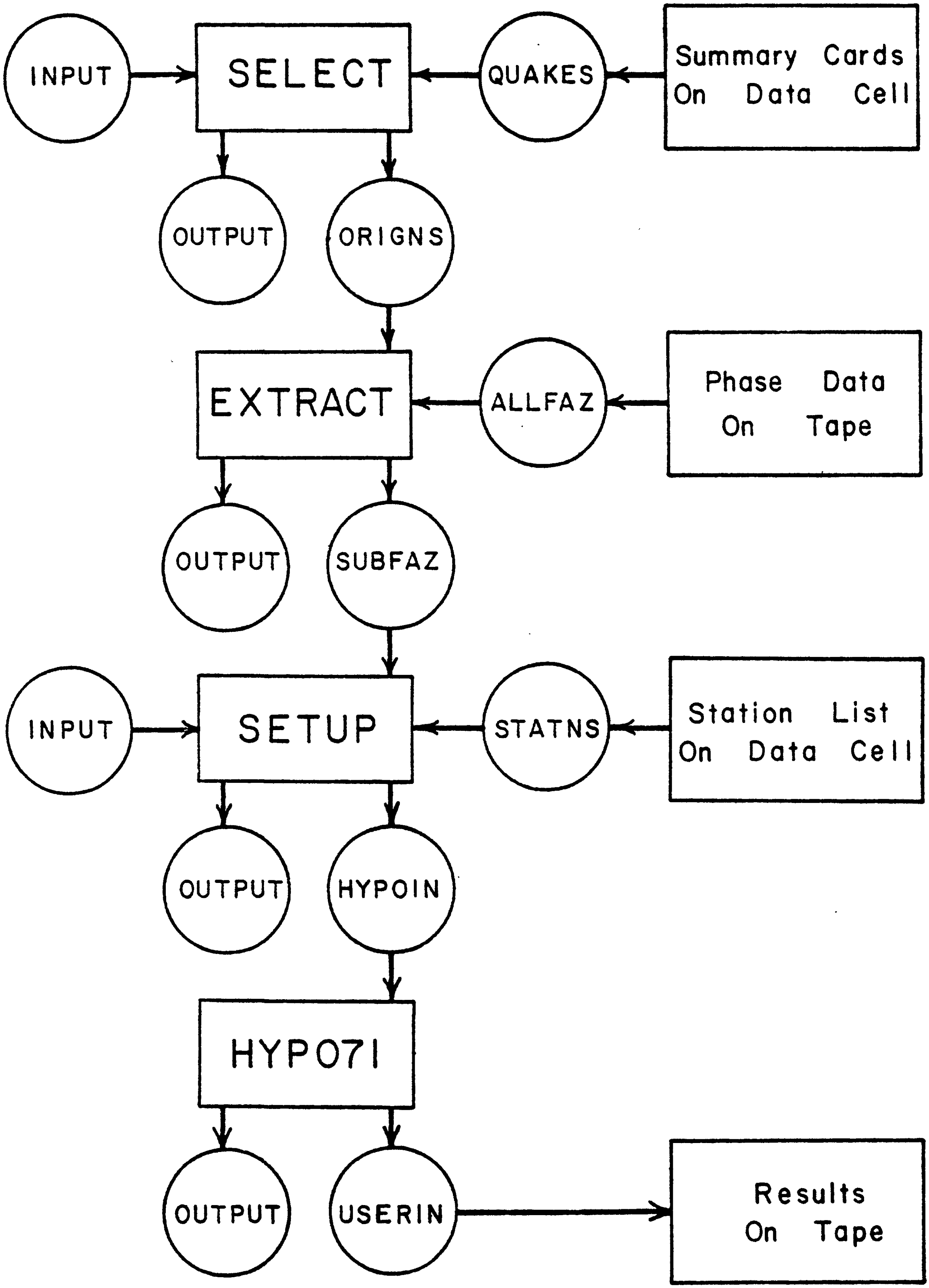
The type of job considered when the file names expected by the programs were chosen is illustrated by the following example. The accompanying flow chart shows how the data, programs, and files relate to each other in this example.

Suppose the phase data for a large number of events in 1972 are stored on tape number 54321, with one file mark after the data for each quarter of the year. Suppose these events have been located and the resulting summary cards stored on the data cell in library USGSEQ subset NCAL72. Suppose further that an appropriate station list and its preceding selection card are stored on the data cell in library YOURLIB subset STALIST. The following example shows how the subset of these events which fall in a particular geographic region and time interval can be relocated with a different crustal model from that originally used. The results of the relocations which would normally be punched out on cards will instead be stored on tape number 98765.

```
nam,4,400,145000.account,name
*LOCAL
FETCHPS,QUAKEPRCSR,SELECT,SELECT.
FETCHPS,USGSEQ,QUAKES,NCAL72.
LINK,F=SELECT,P=BKYIO,L=0,X.
RETURN,QUAKES,SELECT.
COPYSBF,ORIGNS,OUTPUT.
```

```
FETCHPS,QUAKEPRCSR,EXTRACT,EXTRACT.

FBSIZE,ALLFAZ=176.

STAGE,ALLFAZ,54321,2FS,2FXF.

LINK,F=EXTRACT,L=0,X.

RETURN,ORIGNS,ALLFAZ,EXTRACT.

FETCHPS,QUAKEPRCSR,SETUP,SETUP.

FETCHPS,YOURLIB,STATNS,STALIST.

LINK,F=SETUP,L=0,X.

RETURN,SUBFAZ,STATNS,SETUP.

FETCHPS,UPGEO,BIN,RHP71.

LINK,F=BIN,L=0,B.

RETURN,BIN.

LGOB,LC=20000,HYPOIN,,USERIN.

RETURN,HYPOIN,LGOB.

STAGE,USERIN,W,98765.

7/8/9  card

720905b0000b00bbbb721115

blank card

blank card

b4    HOLLISTER  QUAD

b36b45.00b121b15.00

b36b45.00b121b30.00

b37b00.00b121b30.00

b37b00.00b121b15.00
```

```
7/8/9  card

STATIONbLIST

blank card

bb3.5bbbb0.0

bb5.8bbbb3.5

bb6.8bbb15.0

bb8.05bb25.0

blank card

bbb5.bb50.b100.b1.78bbbb2bbbbbbbbbbbbbb2

ALLbPHASES

blank card

6/7/8/9  card
```



Flow Chart for Example of Combining Programs

Program Notes for SELECT

The program SELECT consists of a main program, called SELECT, and 6 subroutines: SETREG, TESTREG, SETRGN, TESTRGN, READIN, and BLKWRT.

SELECT. The main program assumes that the input summary cards on QUAKES have the "Blocked" form, and that the output summary cards are to be written on ORIGNS in ordinary card image form. This is the default case; subroutines are called to handle the other cases. The selection process is performed on groups of 34 cards at a time. Selection on the basis of origin time is treated differently from other search parameters. For greater efficiency, SELECT takes advantage of the chronological ordering of events to avoid checking each event against the time limits.

For the purpose of defining the epicentral region and for testing whether an epicenter lies in the epicentral region, latitude and longitude are treated as if they were Cartesian coordinates.

SETREG. This subroutine sets up the common block /AAB/ (which is used by TESTREG) to contain, for each side of a given polygon, a vector perpendicular to that side and the negative of the dot product of that vector with a vector from the origin to a point on the side. If the polygon is convex, each vector in /AAB/ points from the side toward the interior of the polygon. SETREG also determines whether the given list of vertices is a convex polygon by testing whether each vertex is interior with respect to every side.

TESTREG. This subroutine determines whether a given point lies within a convex polygon. For each side of the polygon it tests whether the given point lies on the same side of the infinite straight line through the two vertices defining that side as does the interior of the polygon. This could be done by comparing the projections, onto a vector perpendicular to the side and directed toward the interior, of the vector from the origin to the given point and of a vector from the origin to the side. However, it is equivalent to compare the corresponding dot products rather than the projections.

SETRGN. This subroutine will, given the vertices of a polygon in order either clockwise or counterclockwise, set up the common block /VXVY/ (which is used by TESTRGN) to contain the vertices in counterclockwise order. Whether the original order is clockwise or counterclockwise depends on whether the total turning angle of the tangent vector is -360° or +360°, respectively. This is determined by applying the method of TESTRGN to the vectors between consecutive vertices (rather than from a given point to consecutive vertices). An error message is printed when the total turning angle is not ±360°, since the given list of vertices is then not a polygon.

TESTRGN. This subroutine is used by SELECT to test whether an epicenter lies within a non-convex epicentral region. TESTRGN determines whether a given point lies within an arbitrary polygon. The polygon is specified by its vertices in counterclockwise order, that is, with the interior to the left of the directed boundary. The algorithm used in TESTRGN is based on the fact that the total angle swept out by a line from a given point to a point on the boundary of a simple closed curve, as the boundary point is moved around the figure, is 0° if the given point is exterior to the figure and is 360° if the given point is interior. Counterclockwise angles are taken as positive. For a polygon, the total angle swept out is equal to the sum of the angles between consecutive vertices as seen from the given point.

The polygon will be taken to be a closed subset of the plane so that a point on the boundary (on a side) is interior. For a boundary taken counterclockwise, the interior is to the left of the boundary, so a point on a side may be thought of as lying just to the left of the directed line segment between the consecutive vertices determining the side. Thus the angle between the vertices as seen from a point on a side is +180°. Therefore, the angle swept out as the boundary point moves along the side between two consecutive vertices lies in the range $-180° < \theta \leq +180°$.

Let V_1 and V_2 respectively be the vectors from the given point to two consecutive vertices. Let $V_1 = a_1 i + b_1 j + c_1 k$ and $V_2 = a_2 i + b_2 j + c_2 k$ where i j k are unit vectors of a right handed coordinate system in 3-dimensional space, with i and j in the plane of the polygon and j 90° counterclockwise from i. Note that $c_1 = c_2 = 0$. Let Θ be the angle from V_1 to V_2. Then

$$V_1 \cdot V_2 = |V_1|\ |V_2| \cos \Theta = a_1 a_2 + b_1 b_2$$

$$V_1 \times V_2 = \begin{vmatrix} i & j & k \\ a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \end{vmatrix} = (a_1 b_2 - a_2 b_1) k$$

$$V_1 \times V_2 = |V_1|\ |V_2|\ (\sin|\Theta|)\ell$$

where ℓ is a unit vector perpendicular to V_1 and V_2 and so directed that a right-hand screw driven in the direction of ℓ would carry V_1 into V_2. Now if Θ is positive then $\ell = k$, and if Θ is negative then $\ell = -k$, so that

$$V_1 \times V_2 = |V_1|\ |V_2|\ (\sin\Theta)k = (a_1b_2 - a_2b_1)k \quad .$$

Let $\quad N(\Theta) = (a_1b_2-a_2b_1)$ and $D(\Theta) = (a_1a_2 + b_1b_2) \quad .$

$$\tan\Theta = \sin\Theta/\cos\Theta = N(\Theta)/D(\Theta)$$

$$\Theta = \arctan\,(N(\Theta)/D(\Theta))$$

The quadrant of Θ can be determined from the signs of $\sin\Theta$ and $\cos\Theta$ which are the same as the signs of $N(\Theta)$ and $D(\Theta)$ respectively.

It would be possible to compute the sum of the angles between consecutive vertices by using this expression to determine each angle and then summing. However, this would involve computing an arctangent once for each side of the polygon. This is undesirable because computing inverse trigonometric functions is relatively expensive in terms of computer time.

It is possible to determine whether the sum of the angles between consecutive vertices is 0° or 360° without computing any inverse trigonometric functions (or any other transcendental functions, or even any divisions). This is done by accumulating the quadrant number of the sum of the angles between successive vertices.

Since $-180° < \Theta \leq 180°$ there is an integer $n \,\epsilon\{-2, -1, 0, +1\}$ such that $n(90°) \leq \Theta \leq (n + 1)(90°)$. If n is taken as $n = \begin{cases} -2 & -180° < \Theta < -90° \\ -1 & -90° \leq \Theta < 0° \\ 0 & 0° \leq \Theta < 90° \\ 1 & 90° \leq \Theta \leq 180° \end{cases}$

then n can be determined from $N(\theta)$ and $D(\theta)$ as

$$n = \begin{cases} -2 & N(\Theta)<0 \text{ and } D(\Theta)<0 \\ -1 & N(\Theta)<0 \text{ and } D(\Theta)\geq 0 \\ 0 & N(\Theta)\geq 0 \text{ and } D(\Theta)>0 \\ 1 & N(\Theta)\geq 0 \text{ and } D(\Theta)\leq 0 \end{cases}$$

Let S be an angle and let m be the integer such that $m(90°)\leq S<(m+1)(90°)$.

Then $(m+n)(90°)\leq S+\Theta<(m+n+2)(90°)$.

Let m' be the integer such that $m'(90°)\leq S+\Theta<(m'+1)(90°)$.

Then either $m'=m+n$ or $m'=m+n+1$.

Which of these holds, and thus the value of m', can be determined from the parity, even or odd, of m'. Now

$$m' \text{ is even if } 0\leq \tan(S+\Theta)<\infty$$

and $$m' \text{ is odd if } -\infty< \tan(S+\Theta)<0$$

$$\text{or if } \tan(S+\Theta) = \pm\infty \quad .$$

Suppose N(S) and D(S) are such that $\tan S = N(S)/D(S)$.

Then the trigonometric identity

$$\tan(x + y) = (\tan x + \tan y)/(1-\tan x \tan y)$$

gives $$\tan(S+\Theta) = (N(S)D(\Theta) + N(\Theta)D(S))/(D(S)D(\Theta) - N(S)N(\Theta)).$$

Let $$N(S+\Theta) = N(S)D(\Theta) + N(\Theta)D(S)$$

and $$D(S+\Theta) = D(S)D(\Theta) - N(S)N(\Theta)$$

then $$\tan(S+\Theta) = N(S+\Theta)/D(S+\Theta)$$

Note that $\tan(S+\Theta) = \pm\infty$ if and only if $D(S+\Theta) = 0$ and otherwise sign $(\tan(S+\Theta))=$ sign $(N(S+\Theta)D(S+\Theta))$. Thus the parity of m' can be found without computing $\tan(S+\Theta)$.

Now let Θ_p be the angle from vertex p-1 to vertex p . Let

$S_p = \Sigma_{q=1}^{p-1} \Theta_q$ and m_p be such that $m_p (90°) \leq S_p < (m_p+1)(90°)$. Substituting Θ_p, S_p, and m_p for Θ, S, and m above, it is easy to see that $m_{p+1} = m'_p$ can be computed recursively.

READIN. Summary cards in card image form (packed display code) are read from the file QUAKES in groups of 34 cards.

BLKWRT. This subroutine collects summary cards to be output in the blocked form. When 34 cards have been collected they are written out as a single logical record on the file ORIGNS. The entry points BLKSTR and BLKEND respectively initialize and terminate the processing of BLKWRT.

```
      PROGRAM SELECT (INPUT,OUTPUT,QUAKES,ORIGNS,TAPE1=QUAKES,TAPE9=
     *ORIGNS,TAPE99=INPUT)
C ------- SELECT IS A PROGRAM FOR SEARCHING A LIST OF HYPOCENTERS ------
C ------- ON THE FILE QUAKES AND WRITING A SUBSET ONTO THE FILE ORIGNS -
C ------- WRITTEN BY J. ALAN STEPPE -------------------------------------
C ------- VERSION OF DECEMBER 14, 1974 ----------------------------------
      INTEGER A(272)
      DIMENSION VLAT(99),VLON(99)
      DIMENSION LABEL(8)
      DIMENSION FMT116(1),FMT117(3),FMT118(1),FMT119(1),FMT120(1),FMT121
     *(1)
      CALL SECOND(TS)
      PRINT 150
  150 FORMAT(1H1,14HPROGRAM SELECT)
      REWIND 1
      ICARDS=0
      READ 127,KDATEL,IHRMNL,ISECL,KDATEU,IHRMNU,ISECU,NCARDS,NELLIPS
     *,NBLOKED
  127 FORMAT(I6,1X,I4,1X,I2,4X,I6,1X,I4,1X,I2,8X,A5,4X,A1,4X,A7)
      IF(EOF(99))40,2,40
    2 IF(NCARDS .EQ. 5HCARDS) ICARDS=1
      IF(NELLIPS .EQ. 1HE) GO TO 4
C ------- SET UP FORMATS FOR DECODING HYPO71 SUMMARY CARDS -------------
      FMT116(1)=10H(45X,F5.2)
      FMT117(1)=10H(18X,F2,1X
      FMT117(2)=10H,F5.2,1X,F
      FMT117(3)=10H3,1X,F5.2)
      FMT118(1)=8H(78X,R1)
      FMT119(1)=8H(51X,I2)
      FMT120(1)=10H(62X,F5.2)
      FMT121(1)=10H(37X,F6.2)
      GO TO 3
C ------- SET UP FORMATS FOR DECODING HYPOELLIPSE SUMMARY CARDS --------
    4 FMT116(1)=10H(34X,F2.1)
      FMT117(1)=10H(14X,F2,1X
      FMT117(2)=10H,F4.2,F3,1
      FMT117(3)=7HX,F4.2)
      FMT118(1)=8H(76X,R1)
      FMT119(1)=8H(36X,I3)
      FMT120(1)=10H(45X,F4.2)
      FMT121(1)=10H(29X,F5.2)
C ------- READ SEARCH PARAMETERS AND SET DEFAULTS ---------------------
    3 IF(KDATEU .LT. 101) KDATEU=991232
      IF(KDATEL .LT. 101) KDATEL=101
      IF(IHRMNL .EQ. 0) IHRMNL=0
      IF(ISECL .EQ. 0) ISECL=0
      IF(IHRMNU .EQ. 0) IHRMNU=0
      IF(ISECU .EQ. 0) ISECU=0
      PRINT 140
      PRINT 102,KDATEL,IHRMNL,ISECL,KDATEU,IHRMNU,ISECU
  102 FORMAT(1H0,11X,10HTIME RANGE,I7,1X,I4,1X,I2,6H THRU ,I7,1X,I4,1X,
     *I2)
C ------- CONVERT TIME LIMITS TO FORMAT I14 ---------------------------
      IF(NELLIPS .EQ. 1HE) GO TO 5
      ITIMEL=ISECL+1000*IHRMNL+10000000*KDATEL
      ITIMEU=ISECU+1000*IHRMNU+10000000*KDATEU
      GO TO 6
    5 ITIMEL=100*ISECL+10000*IHRMNL+100000000*KDATEL
      ITIMEU=100*ISECU+10000*IHRMNU+100000000*KDATEU
    6 IBLANK=6H
```

```
      READ 103,MALPHAL,MALPHAU,HL,HU
  103 FORMAT(2A6,4X,2F7.2)
      DECODE(10,130,MALPHAL)RMAGL
  130 FORMAT(F6.2)
      DECODE(10,130,MALPHAU)RMAGU
      IF(MALPHAU .EQ. IBLANK) RMAGU=9.9
      IF(MALPHAL .EQ. IBLANK) RMAGL=-1000.
      PRINT 104,RMAGL,RMAGU
  104 FORMAT(1H0,6X,15HMAGNITUDE RANGE,F12.2,3X,6H THRU ,F7.2)
      IF(HU .EQ. 0.0) HU=6371.
      IF(HL .EQ. 0.0) HL=0.0
      PRINT 105,HL,HU
  105 FORMAT(1H0,10X,11HDEPTH RANGE,F12.2,3X,6H THRU ,F7.2)
      READ 106 ,IQL,IQU,RMSL,RMSU,NOL,NOU
  106 FORMAT(2R1,4X,2F5.2,4X,2I4)
C ------- INSIDE THE CDC 7600, A=1, B=2, C=3, D=4 --------------------
      IF(IQL .GT. 4) IQL=4
      IF(IQU .GT.4) IQU=1
      PRINT 107,IQL,IQU
  107 FORMAT(1H0,8X,13HQUALITY RANGE,8X,R1,6X,6H THRU ,3X,R1)
      IF(RMSU .LE. 0.0) RMSU=100.
      IF(RMSL .EQ. 0.0) RMSL=0.0
      PRINT 108,RMSL,RMSU
  108 FORMAT(1H0,3X,18HRMS RESIDUAL RANGE,F12.2,3X,6H THRU ,F7.2)
      IF(NOU .LE. 0) NOU=999
      IF(NOL .EQ. 0) NOL=0
      PRINT 109,NOL,NOU
  109 FORMAT(1H0,21HNUMBER READINGS RANGE,I9,6X,6H THRU ,I4)
C ------- EPICENTRAL REGION MUST BE A POLYGON ------------------------
      READ 110,NVRTCS,LABEL
  110 FORMAT(I2,A8,7A10)
      PRINT 111,NVRTCS,LABEL
  111 FORMAT(1H0,I3,30H VERTICES OF EPICENTRAL REGION,10X,A8,7A10)
C ------- VERTICES MUST BE GIVEN IN ORDER,         --------------------
C ------- EITHER CLOCKWISE OR COUNTERCLOCKWISE ------------------------
      DO 10 I=1,NVRTCS
      READ 112,LATD,VLATM,LOND,VLONM
  112 FORMAT(I3,1X,F5.2,I4,1X,F5.2)
      IF(VLATM .EQ. 0.0) VLATM=0.0
      IF(VLONM .EQ. 0.0) VLONM=0.0
      PRINT 113,LATD,VLATM,LOND,VLONM
  113 FORMAT(5X,I3,1X,F5.2,4X,I4,1X,F5.2)
      VLAT(I)=60.*LATD+VLATM
      VLON(I)=60.*LOND+VLONM
   10 CONTINUE
      CALL SETREG(NVRTCS,VLAT,VLON,KONVEX)
      PRINT 140
  140 FORMAT(1H0)
      NSELECT=0
      IF(ICARDS .EQ. 0) GO TO 15
      IEND=-1
      READ (1,122) (A(J),J=1,8)
      IF(EOF(1))35,11,35
   11 IF(A(1) .EQ. 10H DATE     0) GO TO 16
      IEND =1
      GO TO 16
C ------- SKIP THRU HYPOCENTER LIST TO LOWER TIME LIMIT ---------------
   15 IF(ICARDS .NE. 0) GO TO 16
      READ (1) A
      IF(EOF(1)) 35,17,35
   16 CALL READIN(IEND,A)
      IF(IEND .GT. 0) GO TO 35
   17 IF(A(265) .EQ. IBLANK) GO TO 20
      DECODE(20,114,A(265)) ITIME
  114 FORMAT(I14)
      IF(ITIME .LT. ITIMEL) GO TO 15
   20 DO 22 I=1,272,8
      DECODE(20,114,A(I)) ITIME
      IF(ITIME .GE. ITIMEL) GO TO 25
   22 CONTINUE
      PRINT 115,(A(J),J=1,8)
  115 FORMAT(1H0,50HERROR  ITIMEL TOO LARGE OR BLANK CARD ENDS BLOCK  ,
     *8A10)
      GO TO 15
   25 IA=I
      CALL BLKSTR(I,A)
C ------- BEGIN SEARCHING HYPOCENTER LIST -----------------------------
      IF(KONVEX .GT. 0) GO TO 27
C ------- NON-CONVEX REGION -------------------------------------------
   67 DO 70 I=IA,272,8
      DECODE(80,FMT116,A(I)) RMAG
      IF(RMAG .LT. RMAGL) GO TO 70
      IF(RMAG .GT. RMAGU) GO TO 70
      DECODE(80,FMT117,A(I)) EQLATD,EQLATM,EQLOND,EQLONM
      EQLAT=60.*EQLATD+EQLATM
      EQLON=60.*EQLOND+EQLONM
      CALL TESTRGN(EQLAT,EQLON,IN,NVRTCS)
      IF(IN .EQ. 0) GO TO 70
      DECODE(80,FMT118,A(I)) IQ
      IF(IQ .GT. IQL) GO TO 70
      IF(IQ .LT. IQU) GO TO 70
      DECODE(80,FMT119,A(I)) NO
      IF(NO .LT. NOL) GO TO 70
      IF(NO .GT. NOU) GO TO 70
      DECODE(80,FMT120,A(I)) RMS
      IF(RMS .LT. RMSL) GO TO 70
      IF(RMS .GT.RMSU) GO TO 70
      DECODE(80,FMT121,A(I)) H
      IF(H .LT. HL) GO TO 70
      IF(H .GT. HU) GO TO 70
      DECODE(20,114,A(I)) ITIME
      IF(ITIME .GT. ITIMEU) GO TO 36
      NSELECT=NSELECT+1
      IF(NBLOKED .EQ. 7HBLOCKED) GO TO 68
      IL=I+7
      WRITE (9,122) (A(J),J=I,IL)
      GO TO 70
   68 CALL BLKWRT(I,A)
   70 CONTINUE
      IF(ICARDS .NE. 0) GO TO 71
      READ (1) A
      IF(EOF(1)) 35,72,35
   71 CALL READIN(IEND,A)
      IF(IEND .GT. 0) GO TO 35
   72 DECODE(20,114,A(1)) ITIME
      IF(ITIME .GT. ITIMEU) GO TO 36
      IA=1
      GO TO 67
C ------- CONVEX REGION -----------------------------------------------
   27 DO 30 I=IA,272,8
```

```
      DECODE(80,FMT116,A(I)) RMAG
      IF(RMAG .LT. RMAGL) GO TO 30
      IF(RMAG .GT. RMAGU) GO TO 30
      DECODE(80,FMT117,A(I)) EQLATD,EQLATM,EQLOND,EQLONM
      EQLAT=60.*EQLATD+EQLATM
      EQLON=60.*EQLOND+EQLONM
      CALL TESTREG(EQLAT,EQLON,IN,NVRTCS)
      IF(IN .EQ. 0) GO TO 30
      DECODE(80,FMT118,A(I)) IQ
      IF(IQ .GT. IQL) GO TO 30
      IF(IQ .LT. IQU) GO TO 30
      DECODE(80,FMT119,A(I)) NO
      IF(NO .LT. NOL) GO TO 30
      IF(NO .GT. NOU) GO TO 30
      DECODE(80,FMT120,A(I)) RMS
      IF(RMS .LT. RMSL) GO TO 30
      IF(RMS .GT.RMSU) GO TO 30
      DECODE(80,FMT121,A(I)) H
      IF(H .LT. HL) GO TO 30
      IF(H .GT. HU) GO TO 30
      DECODE(20,114,A(I)) ITIME
      IF(ITIME .GT. ITIMEU) GO TO 36
      NSELECT=NSELECT+1
      IF(NBLOKED .EQ. 7HBLOCKED) GO TO 28
      IL=I+7
      WRITE (9,122) (A(J),J=I,IL)
  122 FORMAT(8A10)
      GO TO 30
   28 CALL BLKWRT(I,A)
   30 CONTINUE
      IF(ICARDS .NE. 0) GO TO 31
      READ (1) A
      IF(EOF(1)) 35,32,35
   31 CALL READIN(IEND,A)
      IF(IEND .GT. 0) GO TO 35
C ------- UPPER TIME LIMIT IS CHECKED ON ONLY EVERY 34TH CARD WHEN    ---
C ------- ALL QUAKES ARE BEING REJECTED ---------------------------------
   32 DECODE(20,114,A(1)) ITIME
      IF(ITIME .GT. ITIMEU) GO TO 36
      IA=1
      GO TO 27
   35 PRINT 123
  123 FORMAT(1H0,22HEND OF HYPOCENTER LIST)
      GO TO 37
   36 PRINT 124
  124 FORMAT(1H0,17HEND OF TIME RANGE)
   37 PRINT 125,NSELECT
  125 FORMAT(1H0,I7,4X,20HEARTHQUAKES SELECTED)
      PRINT 140
      PRINT 140
      IF(NBLOKED .EQ. 7HBLOCKED) CALL BLKEND(I,A)
C ------- WRITE END OF FILE MARK BETWEEN DIFFERENT GROUPS OF QUAKES ----
      ENDFILE 9
      REWIND 1
C ------- READ NEW SET OF SEARCH PARAMETERS -------------------------
    1 READ 101,KDATEL,IHRMNL,ISECL,KDATEU,IHRMNU,ISECU
  101 FORMAT(I6,1X,I4,1X,I2,4X,I6,1X,I4,1X,I2)
      IF(EOF(99)) 40,3,40
   40 ENDFILE 9
      REWIND 9
      PRINT 126,NCARDS,NELLIPS,NBLOKED
  126 FORMAT(1H0,10HEND OF RUN,9X,18HFORMAT CONTROL WAS,3X,A5,4X,A1,4X,A
     *7)
      CALL SECOND(TE)
      PRINT 1000,TS,TE
 1000 FORMAT(1X,3HTS=,F20.5,20X,3HTE=,F20.5)
      STOP
      END
      SUBROUTINE READIN(IEND,A)
      INTEGER A(272)
      IF(IEND)1,2,3
    1 READ (1,122) (A(J),J=1,8)
      IF(EOF(1))2,4,2
    2 IEND=1
      RETURN
    3 IEND=-1
    4 DO 5 I=9,272,8
      IL=I+7
      READ (1,122) (A(J),J=I,IL)
  122 FORMAT(8A10)
      IF(EOF(1))6,5,6
    5 CONTINUE
      RETURN
    6 IEND=0
      DO 7 J=I,272
      A(J)=10H
    7 CONTINUE
      RETURN
      END
      SUBROUTINE SETREG(K,V1,V2,KONVEX)
      DIMENSION V1(99),V2(99)
      COMMON /AAB/ A1(100),A2(100),B(100)
      DO 20 I=1,K
      I2=I+1
      IF(I .EQ. K) I2=1
      A1(I)=V2(I2)-V2(I)
      A2(I)=V1(I)-V1(I2)
      B(I)=V2(I)*(-A2(I))-V1(I)*A1(I)
   20 CONTINUE
      KP1=K+1
      A1(KP1)=0.0
      A2(KP1)=0.0
      B(KP1)=-1.E300
      TEST=A1(1)*V1(3)+A2(1)*V2(3)+B(1)
      IF(TEST.GE.0.0) GO TO 30
      DO 25 I=1,K
      A1(I)=-A1(I)
      A2(I)=-A2(I)
      B(I)=-B(I)
   25 CONTINUE
C ------- TEST FOR CONVEXITY ------------------------------------------
   30 IM1=K
      DO 35 I=1,K
      DO 33 J=1,K
      IF(J .EQ. I) GO TO 33
      IF(J .EQ. IM1) GO TO 33
      IF(A1(J)*V1(I)+A2(J)*V2(I)+B(J) .LT. 0.0) GO TO 40
   33 CONTINUE
      IM1=I
   35 CONTINUE
      KONVEX=1
      RETURN
   40 PRINT 100
  100 FORMAT(1H0,25HTHIS REGION IS NOT CONVEX)
      KONVEX=0
      CALL SETRGN(K,V1,V2)
      RETURN
      END
      SUBROUTINE TESTREG(X,Y,IN,K)
      COMMON /AAB/ A1(100),A2(100),B(100)
      IF(A1(1)*X+A2(1)*Y+B(1)  .LT.  0.0)  GO  TO  2
      IF(A1(2)*X+A2(2)*Y+B(2)  .LT.  0.0)  GO  TO  2
      IF(A1(3)*X+A2(3)*Y+B(3)  .LT.  0.0)  GO  TO  2
      I=3
    1 I=I+1
      IF(A1(I)*X+A2(I)*Y+B(I)  .GE.  0.0)  GO  TO  1
      IF(I  .LE.  K)  GO  TO  2
      IN=1
      RETURN
    2 IN=0
      RETURN
      END
      SUBROUTINE SETRGN(K,V1,V2)
C ------- THIS SUBROUTINE PUTS THE VERTICES INTO COUNTERCLOCKWISE ORDER
C ------- BY EXAMINING THE TOTAL TURNING ANGLE OF THE TANGENT VECTOR ---
      DIMENSION V1(99),V2(99)
      COMMON /VXVY/ VX(99),VY(99)
      DO 30 I=1,K
      VX(I)=V1(I)
      VY(I)=V2(I)
   30 CONTINUE
      DXOLD=VX(K)-VX(K-1)
      DYOLD=VY(K)-VY(K-1)
      X=VX(K)
      Y=VY(K)
      TANNUM=0.0
      TANDEN=1.0
      MQUAD=0
      I=1
    2 DXNEW=VX(I)-X
      X=VX(I)
      DYNEW=VY(I)-Y
      Y=VY(I)
      DOT=DXOLD*DXNEW+DYOLD*DYNEW
      CROSS=DXOLD*DYNEW-DXNEW*DYOLD
      DXOLD=DXNEW
      DYOLD=DYNEW
      TEMP=TANNUM*DOT+CROSS*TANDEN
      TANDEN=TANDEN*DOT-TANNUM*CROSS
      TANNUM=TEMP
      IF(CROSS .GE. 0.0) GO TO 6
      IF(DOT .GE. 0.0) GO TO 4
      MQUAD=MQUAD-2
      GO TO 16
    4 MQUAD=MQUAD-1
      GO TO 8
    6 IF(DOT .GT. 0.0) GO TO 16
      MQUAD=MQUAD+1
    8 IF(TANNUM*TANDEN .LT. 0.0) GO TO 10
      IF(TANDEN .EQ. 0.0) GO TO 10
      MQUAD=MQUAD+1
    9 I=I+1
      IF(I .LE. K) GO TO 2
      GO TO 50
   11 DXNEW=VX(I)-X
      X=VX(I)
      DYNEW=VY(I)-Y
      Y=VY(I)
      DOT=DXOLD*DXNEW+DYOLD*DYNEW
      CROSS=DXOLD*DYNEW-DXNEW*DYOLD
      DXOLD=DXNEW
      DYOLD=DYNEW
      TEMP=TANNUM*DOT+CROSS*TANDEN
      TANDEN=TANDEN*DOT-TANNUM*CROSS
      TANNUM=TEMP
      IF(CROSS .GE. 0.0) GO TO 14
      IF(DOT .GE. 0.0) GO TO 12
      MQUAD=MQUAD-2
      GO TO 8
   12 MQUAD=MQUAD-1
      GO TO 16
   14 IF(DOT .GT. 0.0) GO TO 8
```

```
      MQUAD=MQUAD+1
   16 IF(TANNUM*TANDEN .LT. 0.0) GO TO 18
      IF(TANDEN .EQ. 0.0) GO TO 18
      GO TO 9
   18 MQUAD=MQUAD+1
   10 I=I+1
      IF(I .LE. K) GO TO 11
   50 IF(MQUAD .GT. -4) GO TO 60
      IF(MQUAD .LT. -5) GO TO 70
      DO 40 I=1,K
      VX(I)=V1(K+1-I)
      VY(I)=V2(K+1-I)
   40 CONTINUE
      RETURN
   60 IF(MQUAD .LT. 3) GO TO 70
      IF(MQUAD .LE. 4) RETURN
   70 PRINT 101
  101 FORMAT(1H0,65HWARNING, THE BOUNDARY OF THIS REGION IS NOT A SIMPLE
     * CLOSED CURVE)
      RETURN
      END
      SUBROUTINE TESTRGN(X,Y,IN,K)
C ------- THIS SUBROUTINE DETERMINES WHETHER THE POINT X,Y LIES WITHIN -
C ------- THE ARBITRARY POLYGON WITH VERTICES VX,VY -------------------
C ------- IN COUNTERCLOCKWISE ORDER, BY DETERMINING WHETHER ------------
C ------- THE TOTAL ANGLE ABOUT X,Y IS 0 OR 360 DEGREES. ---------------
C ------- TANNUM AND TANDEN ARE THE NUMERATOR AND DENOMINATOR ----------
C ------- OF THE TANGENT OF THE SUM OF THE ANGLES. --------------------
C ------- WRITTEN BY J. ALAN STEPPE -----------------------------------
C ------- VERSION OF OCTOBER 4, 1974 ----------------------------------
      COMMON /VXVY/ VX(99),VY(99)
      DXOLD=VX(K)-X
      DYOLD=VY(K)-Y
      TANNUM=0.0
      TANDEN=1.0
      MQUAD=0
      I=1
C ------- MQUAD IS EVEN -----------------------------------------------
    2 DXNEW=VX(I)-X
      DYNEW=VY(I)-Y
      DOT=DXOLD*DXNEW+DYOLD*DYNEW
      CROSS=DXOLD*DYNEW-DXNEW*DYOLD
      DXOLD=DXNEW
      DYOLD=DYNEW
      TEMP=TANNUM*DOT+CROSS*TANDEN
      TANDEN=TANDEN*DOT-TANNUM*CROSS
      TANNUM=TEMP
      IF(CROSS .GE. 0.0) GO TO 6
      IF(DOT .GE. 0.0) GO TO 4
      MQUAD=MQUAD-2
      GO TO 16
    4 MQUAD=MQUAD-1
      GO TO 8
    6 IF(DOT .GT. 0.0) GO TO 16
      MQUAD=MQUAD+1
    8 IF(TANNUM*TANDEN .LT. 0.0) GO TO 10
      IF(TANDEN .EQ. 0.0) GO TO 10
      MQUAD=MQUAD+1
   10 I=I+1
      IF(I .LE. K) GO TO 2
      GO TO 50
C ------- MQUAD IS ODD ------------------------------------------------
   11 DXNEW=VX(I)-X
      DYNEW=VY(I)-Y
      DOT=DXOLD*DXNEW+DYOLD*DYNEW
      CROSS=DXOLD*DYNEW-DXNEW*DYOLD
      DXOLD=DXNEW
      DYOLD=DYNEW
      TEMP=TANNUM*DOT+CROSS*TANDEN
      TANDEN=TANDEN*DOT-TANNUM*CROSS
      TANNUM=TEMP
      IF(CROSS .GE. 0.0) GO TO 14
      IF(DOT .GE. 0.0) GO TO 12
      MQUAD=MQUAD-2
      GO TO 8
   12 MQUAD=MQUAD-1
      GO TO 16
   14 IF(DOT .GT. 0.0) GO TO 8
      MQUAD=MQUAD+1
   16 IF(TANNUM*TANDEN .LT. 0.0) GO TO 18
      IF(TANDEN .EQ. 0.0) GO TO 18
      GO TO 9
   18 MQUAD=MQUAD+1
   10 I=I+1
      IF(I .LE. K) GO TO 11
   50 IF(MQUAD .GT. 0) GO TO 99
      IN=0
      RETURN
   99 IN=1
      RETURN
      END
      SUBROUTINE BLKWRT(I,A)
C ------- SUMMARY CARDS TO BE OUTPUT ARE BLOCKED ----------------------
C ------- INTO LOGICAL RECORDS OF 34 CARD IMAGES EACH -----------------
      INTEGER A(272),BLKOUT(272)
      IL=I+7
      DO 1 J=I,IL
      K=K+1
      BLKOUT(K)=A(J)
    1 CONTINUE
      IF(K .GE. 272) GO TO 3
      RETURN
      ENTRY BLKEND
      IF(K .LE. 0) GO TO 4
      KP1=K+1
      DO 2 J=KP1,272
      BLKOUT(J)=10H
    2 CONTINUE
    3 WRITE (9) BLKOUT
      ENTRY BLKSTR
      K=0
    4 RETURN
      END
```

Program Notes for EXTRACT

The program EXTRACT consists of a main program, called EXTRACT, and one subroutine: CNVRT.

EXTRACT. Roughly speaking, EXTRACT alternately advances the files ORIGNS and ALLFAZ. It reads an origin time from ORIGNS, looks down ALLFAZ until it finds that event, reads another origin time from ORIGNS, looks further down ALLFAZ until it finds that event, and so on. If EXTRACT encounters an event on ALLFAZ with a time much later than the origin time most recently read from ORIGNS, it will report that the desired event is missing from ALLFAZ and proceed to the next origin time on ORIGNS.

Because of this procedure, a single erroneously late arrival time (a year or month number in error, perhaps) could cause many events on ORIGNS to be missed; i.e., not found on ALLFAZ, although actually there. Therefore EXTRACT makes an effort to catch this error. Erroneously early arrival times are not as serious a problem, since a single error of this type can cause only one event to be missed.

The month numbers read in from data, on both ORIGNS and ALLFAZ, are used as array subscripts in the subroutine CNVRT. It is important to check that they are within the proper range, 1 through 12, since an improper value could cause the program to abort.

CNVRT. This subroutine computes the time in seconds from the beginning of January 1, 1969 to any given time between the beginning of March 1, 1900 and the end of December 31, 1999. It does not know the difference between the Gregorian calendar and the Julian calendar.

NXTRACT. The program NXTRACT is identical to EXTRACT except for the following two statements:

```
102 FORMAT(1H1,15HPROGRAM NXTRACT)                                   NXT  22
 72 CONTINUE                                                         NXT 119
```

```
      PROGRAM EXTRACT (ORIGNS,ALLFAZ,OUTPUT,SUBFAZ,TAPE5=ORIGNS,TAPE8=
     1ALLFAZ,TAPE6=OUTPUT,TAPE7=SUBFAZ)
C ------- EXTRACT -------------------------------------------------------
C ------- A PROGRAM TO EXTRACT PHASE LISTS FOR GIVEN ORIGIN TIMES ------
C ------- BY J. ALAN STEPPE ---------------------------------------------
C ------- VERSION OF SEPTEMBER 16, 1975 ---------------------------------
      INTEGER IQUAKE(8),CARD(800),KARD(8)
      INTEGER FMT(3)
      INTEGER TIME,PTIME
      CALL SECOND (TS)
      IBLANK=4H
      ICAL=3HCAL
      I4=1H4
      IN=1HN
      IS=1HS
      IFMT2=10H6I2)
C ------- SET UP FORMAT FOR READING HYPO71 SUMMARY CARDS ---------------
      FMT(1)=10H(8A10,T1,
      FMT(2)=10H3I2,1X,2I2
      FMT(3)=10H,1X,I2)
      WRITE (6,102)
  102 FORMAT(1H1,15HPROGRAM EXTRACT)
      WRITE (6,104)
  104 FORMAT(1H0)
      PTIME=10000000000
      ASSIGN 28 TO LATE
      MISS=1
C ------- TEST FOR HYPOELLIPSE ERROR ELLIPSE DATA SUMMARY CARDS --------
C ------- AND CHANGE FORMAT IF NECESSARY -------------------------------
      REWIND 5
      READ (5,106) LATNORS
  106 FORMAT(16X,A1)
      IF (EOF(5)) 12,14,12
   12 READ (5,106) LATNORS
      IF (EOF(5)) 92,14,92
   14 IF ((LATNORS.EQ.IN).OR.(LATNORS.EQ.IS)) FMT(2)=IFMT2
      REWIND 5
C ------- READ AN ORIGIN TIME AND CONVERT TO SECONDS -------------------
   16 READ (.ERR.=60,5,FMT) (IQUAKE(J),J=1,8),IYR,IMO,IDY,IHR,IMIN,ISEC
      IF (EOF(5)) 80,18,80
   18 IF ((1.GT.IMO).OR.(IMO.GT.12)) GO TO 60
      CALL CNVRT (IYR,IMO,IDY,IHR,IMIN,ISEC,TIME)
C ------- IF THE LAST QUAKE WAS MISSING, COMPARE THIS ONE --------------
C ------- DIRECTLY WITH THE CURRENT EVENT ON ALLFAZ --------------------
   20 IF (MISS.EQ.0) GO TO 54
C ------- BEGIN SEARCHING ALLFAZ ----------------------------------------
   22 K=1
      N=K+7
   24 READ (.ERR.=64,8,108) (CARD(L),L=K,N),MSTA,IPWT,IRMK
     1,IYR,IMO,IDY,IHR,IMIN,ISEC
  108 FORMAT(8A10,T1,A4,T8,A1,T63,A3,T10,6I2)
      IF (EOF(8)) 90,26,90
   26 IF (IRMK.EQ.ICAL) GO TO 72
      IF (IPWT.EQ.I4) GO TO 70
      IF (MSTA.EQ.IBLANK) GO TO 74
      IF ((1.GT.IMO).OR.(IMO.GT.12)) GO TO 64
C ------- CONVERT ARRIVAL TIME TO SECONDS ------------------------------
      CALL CNVRT (IYR,IMO,IDY,IHR,IMIN,ISEC,PTIME)
      GO TO LATE, (28,52)
C ------- TEST WHETHER ARRIVAL TIME IS BEFORE ORIGIN TIME --------------
```

```
   28 IF (PTIME.GE.TIME) GO TO 40
C ------- SKIP ONE UNWANTED PHASE LIST ----------------------------------
   30 READ (8,110) MSTA
      IF (EOF(8)) 90,32,90
  110 FORMAT(A4)
   32 IF (MSTA.EQ.IBLANK) GO TO 22
      GO TO 30
C ------- TEST CORRESPONDENCE OF ORIGIN TIME AND ARRIVAL TIME ----------
   40 IF (TIME+80.LT.PTIME) GO TO 50
   42 WRITE (6,112) (IQUAKE(M),M=1,4),(CARD(L),L=K,N)
  112 FORMAT(6H EVENT,5X,3A10,A6,5X,8A10)
C ------- WRITE ONE PHASE LIST TO SUBFAZ -------------------------------
      WRITE (7,114) (CARD(L),L=1,N)
  114 FORMAT(8A10)
  116 FORMAT(8A10,T1,A4)
   44 READ (8,116) KARD,MSTA
      IF (EOF(8)) 48,46,48
   46 WRITE (7,116) KARD
      IF (MSTA.EQ.IBLANK) GO TO 16
      GO TO 44
   48 WRITE (7,118)
  118 FORMAT(1H )
      GO TO 16
C ------- CHECK FOR ERRONEOUS LATE ARRIVAL TIME ------------------------
   50 KOLD=K
      ASSIGN 52 TO LATE
      GO TO 70
   52 ASSIGN 28 TO LATE
   54 IF (TIME+80.LT.PTIME) GO TO 58
      IF (MISS.EQ.0) GO TO 56
      NOLD=KOLD+7
      WRITE (6,120) (CARD(L),L=KOLD,NOLD),(CARD(L),L=K,N)
  120 FORMAT(1H0,31HPOSSIBLY BAD TIME ON PHASE CARD, 9X,8A10,/,
     117H IN SAME EVENT AS,24X,8A10,/)
   56 MISS=1
      IF (PTIME.GE.TIME) GO TO 42
      GO TO 30
C ------- WRITE ERROR MESSAGE FOR MISSING QUAKE ------------------------
   58 WRITE (6,122) IQUAKE,(CARD(L),L=K,N)
  122 FORMAT(1H0,30H***** ERROR  --  MISSING QUAKE,10X,8A10,/,17X,
     113H CURRENT CARD,11X,8A10,/,23H SKIPPING TO NEXT EVENT,/)
      MISS=0
      GO TO 16
C ------- ILLEGAL DATA ERROR MESSAGES ----------------------------------
   60 WRITE (6,124) IQUAKE
  124 FORMAT(1H0,40HILLEGAL DATA IN FIELD ON SUMMARY CARD    ,8A10,/)
      GO TO 16
   62 WRITE (6,124) IQUAKE
      GO TO 82
   64 WRITE (6,126) (CARD(L),L=K,N)
  126 FORMAT(1H0,35HILLEGAL DATA IN FIELD ON PHASE CARD,5X,8A10,/)
      IF (IRMK.EQ.ICAL) GO TO 72
      IF (MSTA.EQ.IBLANK) GO TO 74
C ------- STORE PHASE CARDS WITHOUT TIME TO SECONDS -------------------
   70 K=K+8
      N=K+7
      GO TO 24
C ------- COPY CAL CARDS FROM ALLFAZ TO SUBFAZ ------------------------
   72 WRITE (7,116) (CARD(L),L=K,N)
      GO TO 24
```

```
C ------- WRITE ERROR MESSAGE FOR EXTRA INSTRUCTION CARD ---------------
   74 WRITE (6,128) (CARD(L),L=K,N)
  128 FORMAT(1H ,51H*** EXTRA BLANK CARD IN PHASE DATA -- SKIPPED ***   ,
     18A10)
      GO TO 24
C ------- IF ORIGNS HAS ANOTHER NONEMPTY FILE, REWIND ALLFAZ IF --------
C ------- NECESSARY, AND BEGIN AGAIN ------------------------------------
   80 WRITE (6,130)
  130 FORMAT(1H0,21HEND OF FILE ON ORIGNS)
   82 READ (.ERR.=62,5,FMT) (IQUAKE(J),J=1,8),IYR,IMO,IDY,IHR,IMIN,ISEC
      IF (EOF(5)) 92,84,92
   84 IF ((1.GT.IMO).OR.(IMO.GT.12)) GO TO 62
      CALL CNVRT (IYR,IMO,IDY,IHR,IMIN,ISEC,TIME)
      IF (TIME.GT.PTIME) GO TO 20
      MISS=1
      REWIND 8
      GO TO 22
C ------- WRITE END OF RUN MESSAGES -------------------------------------
   90 WRITE (6,132)
  132 FORMAT(//41H ***** ERROR  --  END OF INPUT PHASE DATA)
   92 WRITE (6,134)
  134 FORMAT(//,11H END OF RUN,/)
      END FILE 7
      REWIND 7
      CALL SECOND (TE)
      WRITE (6,136) TS,TE
  136 FORMAT(1X,19HSTARTING CPU TIME =,F15.3,5H  SEC,20X,
     117HENDING CPU TIME =,F15.3,5H  SEC)
      STOP
      END
      SUBROUTINE CNVRT (IYR,IMO,IDY,IHR,IMIN,ISEC,ITIME)
C ------- GIVES TIME IN SECONDS FROM BEGINNING OF JANUARY 1, 1969 ------
      DIMENSION IDA(12)
      DATA IDA/0,31,59,90,120,151,181,212,243,273,304,334/
      IDAYZ=IDA(IMO)+IDY+365*IYR+(IYR/4)-25203
      IF ((IMO.LE.2).AND.(MOD(IYR,4).EQ.0)) IDAYZ=IDAYZ-1
      ITIME=ISEC+60*(IMIN+60*IHR)+IDAYZ*86400
      RETURN
      END
```

Program Notes for SETUP

The program SETUP consists of a main program, called SETUP, and 3 subroutines: ISORT, BINSRCH, and ALLSTA.

SETUP. The main program is fairly straightforward, given an understanding of the syntax and semantics of the SETUP "language".

If alphanumeric character data is read in R format, it can be treated just like integer numbers. Sorting these by increasing value orders the character data in some particular way. This is essentially alphabetical order in the CDC 7600, due to its method of representing characters internally. Thus in SETUP a list of station names is sorted and/or searched just as if it were a list of integer numbers.

ISORT. This subroutine sorts a list of integer numbers into ascending order.

BINSRCH. This subroutine determines whether an integer is included in an ascending ordered list by a binary search technique. The length of a part of the list containing the desired integer is repeatedly reduced by a factor of 2.

ALLSTA. This subroutine copies the phase list of one event from SUBFAZ to HYPOIN and does the ALL STATIONS processing on it.

```
      PROGRAM SETUP(INPUT,OUTPUT,SURFAZ,HYPOIN,STATNS,TAPE1=INPUT,TAPE2=
     *OUTPUT,TAPE3=SURFAZ,TAPE4=HYPOIN,TAPE5=STATNS)
C ------- A PROGRAM TO MODIFY PHASE LISTS  AND / OR -------------------
C ------- SETUP AN INPUT DECK FOR HYPO71 OR HYPOELLIPSE ----------------
C ------- BY J. ALAN STEPPE --------------------------------------------
C ------- VERSION OF DECEMBER 17, 1975 ---------------------------------
      INTEGER CARD(8)
      DIMENSION KARD(8),ILIST(8,25)
      DIMENSION LIST(99),LCRD(7)
      DIMENSION NSTAADD(99)
      DIMENSION NSTA(301),IFMT(2)
      DATA NAMEA/*PHASE LIST*/
      DATA NAMEB/*ALL PHASES*/
      DATA NAMEC/*INSTRUCTIO*/
      DATA NAMED/*N CARD    */
      DATA NAMER/*, REPLACE */
      DATA NAMES/*STATION LI*/
      DATA NAMEADD/*ADD STATIO*/
      DATA NAMERPT/*, REPEAT  */
      DATA NAMEALL/*ALL STATIO*/
      CALL SECOND(TS)
      WRITE (2,105)
  105 FORMAT(1H1,13HPROGRAM SETUP)
      WRITE (2,106)
  106 FORMAT(1H0)
      ICAL=3HCAL
      IFMT(2)=2H4)
      DO 20 I=1,8
   20 KARD(I)=10H
      J=0
C ------- COPY FROM INPUT TO HYPOIN AND CHECK FOR SETUP DIRECTIVES -----
    1 READ (1,101) CARD
  101 FORMAT(8A10,T10,I2)
      IF(EOF(1))18,3,18
    3 IF(CARD(1) .EQ. NAMEA) GO TO 21
      IF(CARD(1).EQ.NAMEB) GO TO 8
      IF(CARD(1) .EQ. NAMES) GO TO 22
      IF(CARD(1).NE.NAMEC) GO TO 7
      IF(CARD(2).NE.NAMED) GO TO 7
      WRITE(4,101) KARD
      GO TO 1
    7 WRITE(4,101) CARD
      GO TO 1
C ------- COPY STATION LIST FROM STATNS TO HYPOIN ----------------------
   22 READ (5,115) ISELCRD,LCRD
  115 FORMAT(A10,7A10)
      IF(ISELCRD .EQ. 10H1         ) GO TO 31
      IF(ISELCRD .EQ. 10H          ) GO TO 32
      IF(ISELCRD .EQ. 10HBEGIN STAT) GO TO 31
      WRITE (2,116) ISELCRD
  116 FORMAT(1X,31HERROR , IMPROPER SELECTION CARD,10X,A10)
      GO TO 19
   31 IFMT(1)=10H(8A10,T1,R
      GO TO 33
   32 IFMT(1)=10H(8A10,T3,R
   33 WRITE (4,115) ISELCRD,LCRD
      NS=1
   34 READ (5,IFMT)CARD,NSTA(NS)
      IF(EOF(5))24,23,24
   23 WRITE (4,101) CARD
```

```
      IF(NSTA(NS) .EQ. 4R**** .AND. ISELCRD .EQ. 10HBEGIN STAT) GO TO 34
      NS=NS+1
      GO TO 34
   24 WRITE (2,109)
  109 FORMAT(1X,39HONE FILE FROM STATNS INCLUDED IN HYPOIN)
      NS=NS-1
      KSORT=0
      GO TO 1
C ------- BEGIN PROCESSING FOR  PHASE LIST  DIRECTIVE -----------------
C ------- CHECK FOR REPLACE DIRECTIVE ----------------------------------
   21 IF(CARD(2) .EQ. NAMER) GO TO 25
C ------- CHECK FOR REPEAT DIRECTIVE -----------------------------------
      IF(CARD(2) .EQ. NAMERPT) GO TO 61
C ------- COPY ONE PHASE LIST FROM SUBFAZ TO HYPOIN --------------------
    2 READ(3,101) KARD,IYR
      IF(EOF(3))5,4,5
C ------- IYR.EQ.0 IS USED THROUGHOUT TO TEST FOR AN INSTRUCTION CARD --
    4 IF(IYR.EQ.0) GO TO 6
      WRITE(4,101) KARD
      GO TO 2
C ------- CHECK FOR INSTRUCTION CARD DIRECTIVE -------------------------
    6 READ(1,101) CARD
      GO TO 3
C ------- BEGIN REPLACE ------------------------------------------------
   25 DECODE(30,107,CARD(1))NLIST
  107 FORMAT(20X,I2)
      IF(NLIST .LE. 0) GO TO 2
C ------- COPY CAL CARDS FROM SUBFAZ TO HYPOIN -------------------------
   26 READ(3,108)KARD,ITEM,IYR,IRMK
  108 FORMAT(8A10,T1,R4,T10,I2,T63,A3)
      IF(EOF(3))5,27,5
   27 IF(IRMK .NE. ICAL) GO TO 28
      WRITE(4,101)KARD
      GO TO 26
C ------- COPY REPLACEMENT PHASE CARDS FROM INPUT TO HYPOIN ------------
C ------- AND SAVE STATION NAMES IN *LIST* -----------------------------
   28 DO 29 L=1,NLIST
      READ(1,108)CARD,LIST(L)
      WRITE(4,101)CARD
   29 CONTINUE
C ------- PUT REPLACEMENT STATIONS IN ALPHABETICAL ORDER ---------------
      CALL ISORT(LIST,NLIST)
      MSTART=(NLIST+1)/2
C ------- BEGIN CHECKING PHASE CARDS AGAINST REPLACEMENT LIST ----------
   30 IF(IYR .EQ. 0) GO TO 6
C ------- BEGIN BINARY SEARCH ------------------------------------------
      CALL BINSRCH(ITEM,INCLUDE,MSTART,MIDPT,NLIST,LIST)
      IF(INCLUDE .NE. 0) GO TO 40
      WRITE(4,101)KARD
   40 READ(3,108)KARD,ITEM,IYR
      IF(EOF(3))5,30,5
    5 WRITE(2,102)
  102 FORMAT(1X,29HERROR , PHASE LIST FILE SHORT)
      GO TO 19
C ------- BEGIN REPEAT -------------------------------------------------
   61 DECODE(30,114,CARD(1))NRPT
  114 FORMAT(20X,I4)
      KADD=0
      KALL=0
      READ (1,101) CARD
```

```
C ------- CHECK FOR ALL STATIONS DIRECTIVE -----------------------------
      IF(CAPD(1) .NE. NAMEALL) GO TO 35
      KADD=1
      KALL=1
      READ (1,101) CARD
C ------- SORT STATION LIST INTO ALPHABETICAL ORDER --------------------
C ------- IF IT IS NOT ALREADY SORTED ----------------------------------
      IF(KSORT .EQ. 1) GO TO 63
      CALL ISORT(NSTA,NS)
      NSTART=(NS+1)/2
      KSORT=1
      GO TO 63
C ------- CHECK FOR ADD STATIONS DIRECTIVE -----------------------------
   35 IF(CARD(1) .NE. NAMEADD) GO TO 63
C ------- IF ADD STATIONS DIRECTIVE APPEARS, SET ADD INDICATOR, --------
C ------- READ NAMES OF STATIONS, AND READ ONE MORE INPUT CARD ---------
      KADD=1
      DECODE(20,110,CARD(1))NADD
      DO 62 M=1,NADD
      READ(1,111) NSTAADD(M)
   62 CONTINUE
      READ (1,101) CARD
C ------- TRANSFER NRPT PHASE LISTS ------------------------------------
   63 DO 43 IRPT=1,NRPT
      IF(KADD .EQ. 0) GO TO 69
C ------- COPY CAL CARDS -----------------------------------------------
   64 READ (3,117) KARD,NSKARD,IYR,IRMK,TIME
  117 FORMAT(8A10,T1,R4,T10,I2,T63,A3,T10,A10)
      IF(EOF(3))5,65,5
   65 IF(IRMK .NE. ICAL) GO TO 66
      WRITE (4,101) KARD
      GO TO 64
   66 IF(KALL .EQ. 1) GO TO 39
C ------- ADD THE STATIONS ---------------------------------------------
      DO 67 M=1,NADD
      WRITE (4,113) NSTAADD(M),TIME
   67 CONTINUE
C ------- COPY PHASE LIST ----------------------------------------------
   68 IF(IYR .EQ. 0) GO TO 41
      WRITE (4,101) KARD
   69 READ (3,101) KARD,IYR
      IF(EOF(3))5,68,5
C ------- MAKE PHASE LIST MATCH STATION LIST ---------------------------
   39 CALL ALLSTA(KARD,NSKARD,IYR,TIME,NS,NSTA,NSTART,IERR)
      IF(IERR .NE. 0) GO TO 5
C ------- WRITE INSTRUCTION CARD ---------------------------------------
   41 IF(CARD(1) .NE. NAMEC) GO TO 42
      IF(CARD(2) .NE. NAMED) GO TO 42
      WRITE (4,101) KARD
      GO TO 43
   42 WRITE (4,101) CARD
   43 CONTINUE
      GO TO 1
C ------- BEGIN ALL PHASES RESPONSE ------------------------------------
    8 READ(1,101) CARD
      IF(CARD(1) .NE. NAMEC) GO TO 70
      IF(CARD(2) .NE. NAMED) GO TO 70
    9 READ(3,101) KARD
      IF(EOF(3))17,10,17
   10 WRITE(4,101) KARD
```

```
      GO TO 9
   70 IF(CARD(1) .NE. NAMEADD) GO TO 11
C ------- BEGIN ADD STATIONS -------------------------------------------
      DECODE(20,110,CARD(1))NADD
  110 FORMAT(13X,I2)
C ------- READ NAMES OF STATIONS TO BE ADDED ---------------------------
      DO 71 M=1,NADD
      READ (1,111) NSTAADD(M)
  111 FORMAT(A4)
   71 CONTINUE
      READ (1,101) CARD
      IF(CARD(1) .NE. NAMEC) GO TO 80
      IF(CARD(2) .NE. NAMED) GO TO 80
C ------- COPY CAL CARDS FROM SUBFAZ TO HYPOIN -------------------------
   72 READ (3,112) KARD,IYR,IRMK,TIME
  112 FORMAT(8A10,T10,I2,T63,A3,T10,A10)
      IF(EOF(3))5,73,5
   73 IF(IRMK .NE. ICAL) GO TO 74
      WRITE (4,101) KARD
      GO TO 72
C ------- ADD THE ADDITIONAL STATIONS ----------------------------------
   74 DO 75 M=1,NADD
      WRITE (4,113) NSTAADD(M),TIME
  113 FORMAT(A4,5H P 4 ,A10)
   75 CONTINUE
C ------- COPY ONE PHASE LIST FROM SUBFAZ TO HYPOIN --------------------
   76 WRITE (4,101) KARD
      IF(IYR .EQ. 0) GO TO 77
      READ (3,101) KARD,IYR
      IF(EOF(3)) 5,76,5
   77 READ (3,112) KARD,IYR,IRMK,TIME
      IF(EOF(3))17,73,17
C ------- ADD STATIONS WITH INSTRUCTION LIST FROM INPUT ----------------
C ------- READ INSTRUCTION LIST ----------------------------------------
   80 J=J+1
      DO 81 I=1,8
   81 ILIST(I,J)=CARD(I)
      READ (1,101) CARD
      IF(EOF(1))82,80,82
C ------- COPY CAL CARDS -----------------------------------------------
   82 READ (3,112) KARD,IYR,IRMK,TIME
      IF(EOF(3))5,83,5
   83 IF(IRMK .NE. ICAL) GO TO 84
      WRITE (4,101) KARD
      GO TO 82
C ------- ADD THE ADDITIONAL STATIONS ----------------------------------
   84 DO 85 M=1,NADD
      WRITE (4,113) NSTAADD(M),TIME
   85 CONTINUE
C ------- COPY ONE PHASE LIST FROM SUBFAZ TO HYPOIN --------------------
   86 IF(IYR .EQ. 0) GO TO 87
      WRITE (4,101) KARD
      READ (3,101) KARD,IYR
      IF(EOF(3))5,86,5
C ------ WRITE INSTRUCTION LIST ----------------------------------------
   87 DO 88 K=1,J
      WRITE (4,101) (ILIST(I,K),I=1,8)
   88 CONTINUE
      READ (3,112) KARD,IYR,IRMK,TIME
      IF(EOF(3))17,83,17
```

```
C ------- BEGIN REGULAR ALL PHASES  (NO ADD STATIONS) ----------------
   11 KALL=0
C ------- CHECK FOR ALL STATIONS DIRECTIVE ----------------------------
      IF(CARD(1) .NE. NAMEALL) GO TO 91
      KALL=1
      READ (1,101) CARD
      IF(KSORT .EQ. 1) GO TO 90
      CALL TSORT(NSTA,NS)
      NSTART=(NS+1)/2
      KSORT=1
   90 IF(CARD(1) .NE. NAMFC) GO TO 91
      IF(CARD(2) .NE. NAMED) GO TO 91
      INSCARD=1
      GO TO 92
C ------- READ INSTRUCTION LIST ----------------------------------------
   91 J=J+1
      DO 12 I=1,8
   12 ILIST(I,J)=CARD(I)
      READ(1,101) CARD
      IF(EOF(1))92,91,92
C ------- COPY CAL CARDS FROM SUBFAZ TO HYPOIN ------------------------
   92 READ (3,117) KARD,NSKARD,IYR,IRMK,TIME
      IF(EOF(3))5,93,5
   93 IF(IRMK .NE. ICAL) GO TO 94
      WRITE (4,101) KARD
      GO TO 92
C ------- COPY ONE PHASE LIST FROM SUBFAZ TO HYPOIN -------------------
   94 IF(KALL .EQ. 1) GO TO 95
   14 IF(IYR.EQ.0) GO TO 15
      WRITE(4,101) KARD
      READ(3,101) KARD,IYR
      IF(EOF(3))5,14,5
   95 CALL ALLSTA(KARD,NSKARD,IYR,TIME,NS,NSTA,NSTART,IERR)
      IF(IERR .NE. 0) GO TO 5
      IF(INSCARD .NE. 1) GO TO 15
      WRITE (4,101) KARD
      GO TO 96
C ------- WRITE INSTRUCTION LIST TO HYPOIN ----------------------------
   15 DO 16 K=1,J
      WRITE(4,101)(ILIST(I,K),I=1,8)
   16 CONTINUE
   96 READ (3,117) KARD,NSKARD,IYR,IRMK,TIME
      IF(EOF(3))17,93,17
   17 WRITE(2,103)
  103 FORMAT(1X,22HEND OF PHASE LIST FILE)
      GO TO 19
   18 WRITE(2,104)
  104 FORMAT(1X,17HEND OF INPUT DECK)
   19 ENDFILE 4
      REWIND 4
      CALL SECOND(TE)
      PRINT 1000,TS,TE
 1000 FORMAT(1X,3HTS=,F20.5,20X,3HTE=,F20.5)
      STOP
      END
      SUBROUTINE ISORT(X,NO)
      DIMENSION X(NO)
      INTEGER X,TEMP
      MO=NO
    2 IF(MO-15)21,21,23
   21 IF(MO-1) 9, 9,22
   22 MO=2*(MO/4)+1
      GO TO 24
   23 MO=2*(MO/8)+1
   24 KO=NO-MO
      JO=1
   25 I=JO
   26 IF(X(I)-X(I+MO))28,28,27
   27 TEMP=X(I)
      X(I)=X(I+MO)
      X(I+MO)=TEMP
      I=I-MO
      IF(I)28,28,26
   28 JO=JO+1
      IF(JO-KO)25,25,2
    9 RETURN
      END
      SUBROUTINE BINSRCH(ITEM,INCLUDE,MSTART,MIDPT,NLIST,LIST)
C ------- DETERMINES WHETHER AN ITEM IS INCLUDED IN AN ORDERED LIST ----
C ------- BY A BINARY SEARCH TECHNIQUE ---------------------------------
      DIMENSION LIST(NLIST)
      MIDPT=MSTART
      IF(LIST(MIDPT)-ITEM)52,40,51
   51 LIMITL=1
      GO TO 54
   52 LIMITU=NLIST
   53 IF(MIDPT .GE. LIMITU) GO TO 60
      LIMITL=MIDPT+1
      MIDPT=(LIMITL+LIMITU)/2
      IF(LIST(MIDPT)-ITEM)53,40,54
   54 IF (LIMITL .GE. MIDPT) GO TO 60
      LIMITU=MIDPT-1
      MIDPT=(LIMITL+LIMITU)/2
      IF(LIST(MIDPT)-ITEM)53,40,54
   40 INCLUDE=1
      RETURN
   60 INCLUDE=0
      RETURN
      END
      SUBROUTINE ALLSTA(KARD,NSKARD,IYR,TIME,NS,NSTA,MSTART,IERR)
      DIMENSION KARD(8),NSTA(301),ISTA(301)
      DO 1 I=1,NS
    1 ISTA(I)=0
    2 CALL BINSRCH(NSKARD,INCLUDE,MSTART,MIDPT,NS,NSTA)
C ------- IF THE STATION IS ON THE STATION LIST, ---------------------
C ------- COPY THE PHASE CARD TO HYPOIN --------------------------------
      IF(INCLUDE .EQ. 0) GO TO 4
      ISTA(MIDPT)=1
      WRITE (4,101) KARD
  101 FORMAT(8A10,T1,R4,T10,I2)
    4 READ (3,101) KARD,NSKARD,IYR
      IF(EOF(3))5,6,5
    5 IERR=1
      RETURN
    6 IF(IYR .NE. 0) GO TO 2
      IERR=0
C ------- ADD THE REST OF THE STATION LIST -----------------------------
      DO 7 I=1,NS
      IF(ISTA(I) .NE. 0) GO TO 7
      WRITE (4,102) NSTA(I),TIME
  102 FORMAT(R4,5H P 4 ,A10)
    7 CONTINUE
      RETURN
      END
```

Acknowledgments

I thank Robert L. Wesson for suggesting the development of a system of programs to facilitate relocations and similar types of processing of local earthquake data. I thank William L. Ellsworth for permission to use a program of his as a starting point in the development of EXTRACT. I thank John H. Pfluke for several ideas used in SELECT.

References

Backus, J. W., The syntax and semantics of the proposed international algebraic language of the Zürich ACM-GAMM conference, Proc. Internat. Conf. Inf. Proc., UNESCO, Paris, June 1959.

Backus, J. W., et. al., Revised Report on the Algorithmic Language ALGOL 60, Communications of the Association for Computing Machinery, Vol. 6, No. 1, pp. 1-17, January 1963.

Lahr, J. C. and P. Ward, HYPOELLIPSE: a computer program for determining local earthquake hypocentral parameters, magnitude, and first motion pattern, Open-File Report, U.S. Geological Survey, (in preparation).

Lee, W. H. K. and J. C. Lahr, HYPO71: a computer program for determining hypocenter, magnitude, and first motion pattern of local earthquakes, Open-File Report, U.S. Geological Survey, 100 pp., 1972. Revised edition, 113 pp., 1975.