



# Lawrence Berkeley Laboratory

UNIVERSITY OF CALIFORNIA

RECEIVED

## ENERGY & ENVIRONMENT DIVISION

LAWRENCE  
BERKELEY LABORATORY

APR 1 1982

LIBRARY AND  
DOCUMENTS SECTION

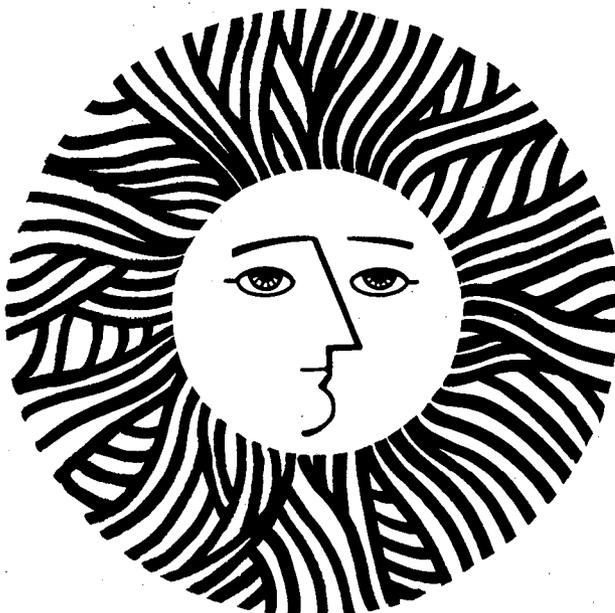
IDEPP, AN INTERACTIVE DATA EDITING AND PLOTTING  
PROGRAM

Stephen R. Brown

September 1981

**For Reference**

Not to be taken from this room.



LBID-448  
c.1

## **DISCLAIMER**

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

IDEPP, AN INTERACTIVE DATA  
EDITING AND PLOTTING PROGRAM

Stephen R. Brown

Building Ventilation and Indoor Air Quality Program  
Energy and Environment Division  
Lawrence Berkeley Laboratory  
University of California  
Berkeley, CA 94720

September 1981

This work was supported by the Assistant Secretary for Conservation and Renewable Energy, Office of Buildings and Community Systems, Building Division of the U.S. Department of Energy under Contract No. W-7405-ENG-48.

# IDEPP, AN INTERACTIVE DATA EDITING AND PLOTTING PROGRAM

## Table of Contents

1.00	Introduction
1.10	An Overview of IDEPP Capabilities and Command Structure
1.11	G and E, Bridges to Edit and Graphics Mode
1.12	B and X, Beginner Mode Commands
1.13	EOLD and ESAV, Wrapping Up an IDEPP Session And What to Do if the Computer Crashes
1.14	D - Accessing This Documentation
1.20	Using IDEPP
1.21	How to Specify an IDEPP Task
1.22	Time Entry Specifications
1.23	Site Selection Entry
1.24	Other Entries and Task Confirmation
2.00	Using IDEPP in Edit Mode
2.10	An Example of an Editing Operation
2.20	Editing Tasks in Detail
2.21	DE - Deletion of Data
2.22	AD - Addition of Constant to Data
2.23	RA - Ratios; Site to Site Comparison
2.24	CF - Curve Fitting / Linear Transform
2.25	ER - Edit Recovery
2.30	Utility Tasks in Edit Mode in Detail
2.31	ET - Viewing Histories of Previous Edits
2.32	TT - Viewing Edits in Progress
2.33	PR - Printer Output of Data
2.34	EC - Exiting Edit Mode to IDEPP Command Mode
2.35	TY - TTY Data Output
2.36	SC - Scan For Data Values Within Given Ranges
2.37	AV - Generate Means and Standard Deviations
3.00	Plotting Data With IDEPP
3.11	G1 - Plot One Day of Data on the CRT
3.12	GN - Plot Next Day's Data
3.13	HP - Graph Histogram for a Given Period
3.14	RT - Repeat CRT Plotting Task for a Different Parameter
3.15	RS - Repeat CRT Plotting Task With New Site Selection
3.16	SR - Change/View the Plotting Constants
3.17	PG - Plot One Day of Data on the Pen-plotter
3.18	PH - Plot a Histogram on the Pen-plotter

4.00	IDEPP Program Description
4.10	The Modular Structure of IDEPP
4.20	Input/Output Details
4.30	IDEPP Structure and Control Flow
4.31	IDEPP Subroutine Map
4.32	Description of Some IDEPP Variables
4.33	Description of IDEPP Routines
4.40	Modifying IDEPP to Accept New Data Formats
4.50	Sample IDEPP Session

# IDEPP, An Interactive Data Editing and Plotting Program

## 1.00 Introduction

IDEPP is an interactive data editing and plotting program written in FORTRAN to execute on a DEC10 system computer. Oriented toward the non-programmer, IDEPP has a number of features to aid the inexperienced user and to protect the data file from being accidentally altered. The command syntax is simple and the program response frequent, facilitating the dialogue between the user and the program.

The data format expected by IDEPP is derived from the format of air quality data recorded by the EEB Mobile Laboratory data-logger. The data-logger polls some thirty inputs once each minute of the day and writes a single record on a floppy disk containing the time and the parametric values. These raw data are averaged, calibrated, and converted to physical units for manipulation by IDEPP. The data format expected by IDEPP is a random-access disk file with each identically sized record containing some number of parametric values and a chronologically ordered time marker. Gaps in time as well as in individual parameters cause no problems.

Although IDEPP was written to support the EEB field-monitoring project, it was intended to be useful to any experiment generating data in the same general format. Modifying IDEPP to work with a different experiment is relatively straightforward and is described in some detail in paragraph 4.40 under Program Description.

To protect the data, the program when the session is initiated creates a working copy of the original file. It is this copy that is altered by editing operations; the only way to alter the original file is with the edit command to end the session, saving all edits. In addition, each edit operation saves the unchanged data records in a scratch file and generates bookkeeping and history around that operation so that the edit can be recovered even in subsequent sessions.

The command syntax is simple and direct. All requests for edit operations, and for plotting operations if the beginner switch is set, are echoed back to the user for confirmation. Thus typos and other errors can be spotted before an operation is begun. Any individual input line can be nullified and started over. The typical command consists of a two-character verb, followed by a 3-character parameter specification, and a carriage return. IDEPP prompts for time and site entries and then echoes back the request. Improper commands, parameters, or times cause a message to be typed out indicating the problem and then generate a prompt for a new command. The echo of any request is detailed enough to avoid ambiguity. In the case of user error, IDEPP gives the user the chance to abort the edit task (or plotting task if in beginner mode) and start over at the command level.

In addition, during editing operations, the user can elect to see each record in its altered form. After the edit is finished, a detailed history of the edit is typed out, including when it occurred, and pointers to the records in the scratch file containing the original data.

These features may lead to a verbose dialogue, but one intended to assure clarity. For plotting operations the "expert" mode is available, which skips most feedback. In the editing mode, which changes the data file, the more conversational mode is retained.

At any time during an editing/plotting session, two levels of help are available from IDEPP: The lower level presents responses given to an input "h" as a command verb, or when an unrecognizable input is read by the program. While these responses are brief, consecutive requests for help receive somewhat more detailed information. If this information is not enough, this program description is available for scanning or print-out, either section by section or in its entirety. The table of contents can be displayed and the section of particular interest chosen by inputting the command, "D" which places the user into the documentation routine. (This program description also exists as a separate ASCII file named IDEPP.DOC and can thus be accessed directly by anyone familiar with the DEC10 system commands.)

#### 1.1 An Overview of IDEPP Capabilities

IDEPP edits and plots data, offering five distinct editing tasks and four distinct plotting tasks. In addition, there are a number of utility tasks to aid the user. Since IDEPP is modular in structure, it is quite easy to add or modify any of these tasks.

IDEPP executes in one of three modes; command mode, edit mode, or graphics mode. "Command" is the mode the user is in at the outset and it acts as a bridge between the edit and graphics modes. Each mode has its own commands which can be invoked only when the user is in that mode. (Switching from one mode to the other is possible at any time by invoking the proper commands--EC, G, or E.)

The Edit mode includes the editing tasks, which are:

- (1) DE - Delete data, either the whole record or a specific parameter
- (2) AD - Add some positive or negative constant to a given parameter
- (3) CF - Perform a specified transform on a parameter
- (4) RA - Compute ratios between "sites" for a specific parameter

Utility tasks associated with the edit mode are:

- (1) TT - View on the terminal each edited record in its altered form as the edit proceeds
- (2) TY - Output to the terminal selected portions of the data
- (3) ET - View the history of any or all of the edit operations
- (4) PR - Create a disk file of selected portions of the data
- (5) EC - Return to IDEPP command mode from edit mode
- (6) ER - Recover an edit operation
- (7) SC - Scan for data values within given ranges
- (8) AV - Generate means, standard deviations, and ranges for selected parameters

The plotting tasks available are:

- (1) G1 - Plot the data vs. time for a specific day and parameter
- (2) GN - Plot the next day's data similarly (expects to be preceded by a G1 operation)
- (3) HP - Develop and plot a histogram of the data for a specific parameter and selected site(s) and times
- (4) RT - Repeat the previous task (either G1, GN, HP, or RS) for a different parameter
- (5) RS - Repeat the previous task (either G1, GN, HP, or RT) for a different site(s)

All the above tasks utilize either the Techtronix 4010-4014 series graphics terminals, or the ADM-3A terminal with the Retrographics RG-512 card that gives the ADM-3A graphics capability.

The following two tasks utilize the Hewlett-Packard 7225A pen-plotter. (Both these tasks assume that the appropriate plotting task has just been completed.)

- (1) PG - Dump the current time vs. value plot to the pen plotter
- (2) PH - Dump the current histogram to the pen plotter

The graphics-mode utility tasks are:

- (1) SR - Change/View the default plotting constants (range, etc.)
- (2) EC - Exit graphics mode to IDEPP command mode

As noted, the command mode acts as a bridge between editing and graphing operations and it is the mode the user is in when IDEPP begins. It is also the mode via which the user finishes an editing/plotting session and ends IDEPP execution. Tasks that can be invoked in the command mode:

- (1) G - Enter graphics mode
- (2) E - Enter edit mode
- (3) B - Invoke beginner prompts accompanying various commands  
(mainly in graphics mode)
- (4) X - Discontinue beginner prompts
- (5) EOLD - End the session without saving any of the edits
- (6) ESAV - End the session, saving the edits
- (7) D - Display portions of this program description  
on the terminal

All of these tasks are described in detail in the following sections.

#### 1.11 G and E, Bridges to Edit and Graphics Mode

When IDEPP begins execution, the user is in IDEPP command mode, which may be pictured as a bridge between edit and graphics modes. Entering G or E and <cr> will put the user in one of the two other modes, whereas EC<cr> in either edit or graphics mode will bring the user back to command mode. The only path between edit and graphics modes is the command mode.

#### 1.12 B and X, Beginner Mode Commands

IDEPP always outputs some sort of prompt when it expects user input. In many cases, particularly in graphics mode, the extent and detail of the prompt will depend upon whether the "beginner switch" is set. At the start of a session, IDEPP prompts for whether or not "beginner" mode is to be invoked. If it is, the more detailed set of prompts is used. Once either "beginner" or "expert" mode is set, it will remain that way until changed by the user with either the B command to go from "expert" to "beginner", or the X command to go from "beginner" to "expert" mode.

IDEPP also asks for confirmation of a command sequence in most cases. In edit mode, these are usually retained whether or not beginner mode is invoked. In graphics expert mode, most confirmations are skipped. (Beginner mode is recommended for the first few sessions with IDEPP.)

#### 1.13 EOLD and ESAV, Wrapping Up an IDEPP Session And What to Do if the Computer Crashes

At the completion of a session, the user returns to command mode and invokes either EOLD or ESAV to end IDEPP execution. As the mnemonics suggest, EOLD retains the data in its original form, i.e., before the current session. Bookkeeping is altered to reflect this, and an entry is made in the edit history file recording the time and that no data was changed. ESAV rewrites the original data file from the working-copy file (this is the only time IDEPP changes the original data file), and updates the bookkeeping to reflect the current session. Of course, if none of the data has been changed (for instance, only plotting has been done) it's immaterial which exit is used, although invoking EOLD, which does not rewrite the original file, is faster and cheaper.

In the event that IDEPP doesn't complete execution normally through EOLD or ESAV (say the DEC10 crashes), it is possible to recover the affected edit session. Because the recovery procedure is somewhat tricky, it's recommended that IDEPP's author, Steve Brown, X4031, be contacted to do the task. Usually, the working-copy file is copied to the original version. Unless a lot of changes were made to the data, it's probably easier to re-start IDEPP and redo the affected edits. The bookkeeping will look as if they were done twice, but otherwise there will be no effect.

#### 1.14 D - Accessing This Documentation

All of this documentation is available on-line while IDEPP is executing via the D command. When invoked, IDEPP asks whether or not the user wishes to see a table of contents of the documentation. This table contains the titles for the various sections preceded by a reference number; these numbers are input to retrieve the applicable sections. Up to ten separate sections can be retrieved in a single command, and a range can be asked for by separating the beginning and ending paragraphs with a 0 entry (a range counts as three entries).

An alternative way to retrieve sections is to concatenate the mode and task about which information is desired and enter it in response to the D\* prompt (only one entry allowed in this format). This mode facilitates information retrieval without resort to the table of contents. An example would be "EDE" for the deletion task in edit mode.

To retrieve this whole document, simply enter 0<cr> after a D\* prompt. It would be well, of course, to have some sort of hard-copy device either attached to or as your terminal. Actually a better way is to simply dispose the document to a line-printer. It currently resides in DOE:IDEPP.DOC[3300,101]. To print it on the B90 IAQ line-printer enter the following print command from DEC10 monitor mode (where the period "." prompt appears):

```
"."PRINT LPT02:=DOE:IDEPP.DOC[3300,101]
```

When IDEPP has output a screenfull of documentation, it pauses for the user to read and decide whether more is wanted. A <cr> means more, while Q<cr> means no more. A Q<cr> after the D\* prompt returns the user to command mode.

## 1.20

### Using IDEPP

Editing and plotting of data using IDEPP is done in dialogue fashion, with commands being entered by the user and the program responding with prompts, error messages, confirmation messages, and completion messages. Typically, the user invokes a discrete task by typing in a command verb and a parameter mnemonic on one line, and the times, sites, and other inputs on following lines.

## 1.21

### How to Specify an IDEPP Task

IDEPP always prompts for user input. The primary task prompt in the command mode is I\*, E\* for edit mode, and G\* for graphics mode. Prompts for other inputs, such as time, site, data source, etc. are similarly formatted, with a letter followed by an asterisk. The task input required is always a one- or two-character verb, a separator (comma or blank), and a two- or three-character parameter entry, such as CO2 for carbon dioxide, or O3 for ozone. In most cases, the command format is identical for edit and graphics mode. This is particularly true for task, parameter, time and site entries.

For example, when the edit mode of operation is entered, a prompt, "E\*", will appear as an indicator of program readiness to accept an editing task. Readiness to accept subsequent editing tasks will be similarly prompted. The user then responds with a one or two-character task verb indicating the type of edit to be done, a separator (either blank or comma), and the parameter to be worked on, if relevant. An example follows.

```
E*DE,CO2
```

In this example, DE means delete a specific parameter, which in this case is CO2 (note that upper-case characters are used, which is true throughout IDEPP). Another example would be

```
E*TY
```

which means type out to the terminal some data. No parameter is entered as it's irrelevant. There should be no imbedded blanks other than the separator in this input. Some other examples:

```
G*G1,NOX  
G*HP,T3  
E*TT,1  
G*PH
```

## 1.22

### Time Entry Specifications

Following entry of the task and relevant parameter, IDEPP will prompt with "T\*" to indicate readiness to accept entry of inclusive time intervals in the data during which the task is to be performed. Each interval is entered as six integers, comma-separated, indicating

beginning and ending month, day, and time (24-hour clock). Up to 50 such intervals can be entered for a single task, and a simple carriage return after the prompt indicates to IDEPP that no further time intervals are to be entered. By way of example:

T\*8,25,1025, 8,31,2330

means August 25th at 10:25 AM to August 31st at 23:30 PM inclusive (23:30 would be 11:30 PM on a 12-hour clock). It is also possible to invoke a specific time interval during the day for a number of consecutive days by entering the month + 100 as the beginning month entry. Thus

T\*112,2,59, 1,5,830

means operate on each day from December 12 thru January 5th, between the times of 12:59 AM and 8:30 AM inclusively. Note two things, 1) this particular experiment overlaps a year-boundary and thus the ending time is "before" the start time. Don't worry, the program will handle it. 2) In this input, spaces can be used freely to clarify the input, so long as the integers are separated by commas. Note also that sequential intervals need not be chronologically arranged.

For ease of entry, IDEPP interprets month and day null entries for the end-time of a particular time interval as identical with the start-time entries. That is, the entry

T\*8,10,1200,,1330                    is equivalent to  
T\*8,10,1200, 8,10,1330

and

T\*12,13,830,,14,1030                is equivalent to  
T\*12,13,830, 12,14,1030

in addition

T\*9,5                                    is equivalent to  
T\*9,5,0,9,5,2400

### 1.23                    Site Selection Entry

The data is multi-plexed between a number of different sampling sites (four in the case of EEB data). IDEPP therefore prompts with "S\*" after the last time entry to indicate readiness to accept site selection entry. The EEB version of IDEPP expects 0-4 entries consisting of 0, 1, 2, or 3 to select the one outdoor and three indoor sampling sites. No entry, just a carriage-return, is interpreted to mean selection of all sites. A few examples:

S\*  
S\*0,1  
S\*1,4,0

The first example selects all sites, the second selects sites 0 and 1, and the third 1, 4, and 0. Note that the entries are comma-separated, and that order is inconsequential.

#### 1.24 Other Entries and Task Confirmation

In several cases, the specific task will need more input than simply parameter, times, and sites. If so, there will be self-explanatory prompts to obtain the necessary input. After all the inputs for a particular task have been entered, IDEPP will usually prompt for confirmation of the task as entered before proceeding to execute the task. Exceptions to this procedure are a few of the editing tasks where the data are not changed by the task, and graphics tasks when in "expert" mode. In requesting confirmation IDEPP will describe the task as it interprets it with enough detail so that there's no ambiguity as to what IDEPP will do if the task is confirmed. This request for confirmation gives the user a chance to review the task for input errors IDEPP didn't trap, particularly incorrect time entries, and to start over (by entering "n<cr>").

## 2.0

### Using IDEPP in Edit Mode

Edit mode is where actual alteration of the data takes place. The various editing commands are used to change data in the working copy of the original file, while saving the original version of the altered records. The utility tasks aid the user in keeping track of what is going on, as well as allowing selected portions of the data to be output to the TTY or a lineprinter.

The sections following, 2.1 - 2.36, describe the edit-mode tasks in detail. The command sequences are as similar as is practical, with fewer inputs required where irrelevant or unnecessary. Verb, parameter, time and site entries are all as described in paragraphs 1.2 - 1.24. Other entries are prompted for in a self-explanatory way when needed.

#### 2.10 An Example of an Editing Operation

The following is an example of an edit operation, namely deleting some CO2 data from the EEB trailer.

E\*DE,CO2

User

T\*4,25,1200, 4,25,1300

Input

S\*2

DELETE CO2 FROM DATA WITHIN  
TIME INTERVALS:  
4/25/1200 TO 4/25/1300  
AT SITES 2 , , , , RIGHT? Y/N Y

IDEPP  
Confirmation

EDIT 3 FINISHED

EDIT: 3 COMPLETED: 24-Feb-81 AT 11:43

CO2 WAS EDITED PARAMETER

DE WAS EDITING TASK

SITES AFFECTED WERE 2 , , , ,

SCRATCH FILE RECORDS= 19 TO 22

TIME INTERVALS WERE:

4/25/1200 TO 4/25/1300

E\*

IDEPP  
History  
of Completed  
Edit

#### 2.20 Editing Tasks in Detail

In general, editing tasks fall into two groups: those which alter the data, such as DE and AD, and those which are utility tasks, such as TT and PR. The first group alters the working-copy and scratch files and therefore the order in which they're done is significant, whereas the utility tasks don't change either of these files and the order of their execution is irrelevant.

## 2.21 Deletion of Data

The DE task is invoked to remove from further consideration either the specific parameter named after the DE entry, or, if "ALL" is entered, the entire data record. In neither case does the record physically disappear from the data; since the data files are all random-access files, it would be costly and inefficient to eliminate the data. In addition, recovery of a specific edit would be vastly more difficult. Instead, a flag is used to indicate that either a specific parameter or record has been "deleted". In the case of a specific parameter, the data entry in the record for that parameter is replaced by 9999.0, which was chosen as a number that could be read and written just as any other value but was highly unlikely to be a valid value. If an entire record is to be henceforth ignored, then a separate flag, usually used to indicate large standard deviations, is set to "DELET". In output versions of the data, this appears in place of the "STAND" in "STANDARD DEV."

The primary effect of a DE operation is to skip the affected data during any subsequent plotting operations and during a number of editing operations. Whether or not (and how) a particular editing task is affected is covered in sections 2.22 to 2.37.

## 2.22 AD - Addition of a Constant to a Parameter

This editing task makes it possible to do simple algebraic addition involving any parameter. After the site entry, IDEPP prompts for the constant to be entered, which can be either positive or negative and integer or real.

## 2.23 RA - Ratios: Site-to-Site Comparisons

This task makes it possible to compare values at different sampling sites for the same parameter. Most often this would involve a comparison between outdoor or ambient, and the indoor sites.

Several additional inputs are needed for this task and IDEPP prompts for them after the site entry. After first asking which site is to be the divisor site, IDEPP asks for a minimum acceptable value for both the divisor and any dividend. This is helpful in ignoring data that may be invalid for some reason and in preventing too large quotients being generated, say by a zero value for the divisor site. Finally a maximum time-lapse between a divisor-site record and a dividend-site record is requested so that widely separated data points will be ignored.

The quotients generated by the ratio operation are stored in the same record as the relevant dividend as a parameter labelled RA1 to RA4, depending on how many separate ratio operations have been done in the current editing session. If more than four, then the previous results begin to be overwritten. The slot in which a particular ratio stores its results is re-initialized to RA1 with the beginning of each IDEPP session. The results can be plotted as RA1 to RA4, but it's up to the user to keep track of which ratio operations are represented by RA1, 2, etc.

Where the divisor is too small or was deleted, the previous divisor is used in its place, as long as the time-lapse isn't exceeded. If the dividend is too small, a deletion flag is inserted for the quotient in the record. If the current dividend is too much later than the divisor in time, thus exceeding the time-lapse, a deletion flag is inserted as the quotient. If either the record or the dividend was previously deleted, the quotient is flagged as before.

Another aspect of RA tasks is that the original record is not backed up into the scratch file, meaning that it's impossible to recover an RA operation and, since no entry is made in the edit history file, IDEPP won't even try (This makes sense because it isn't the actual data that's being affected).

To view the calculated ratios, first invoke either the "TT,3" task to display the results on the terminal, or the "PR" task to generate a file to dump to the line-printer. Viewing ratios calculated earlier in the session or during a previous, saved session can be done via the "TY" command.

## 2.24 CF - Curve-fitting/Linear Transform

This task makes it possible to perform any transform of the data represented by a curve in two-dimensional space that maps the old parametric values to some new set of values in a one-to-one fashion. IDEPP accepts the transform as a series of pairs of numbers representing the x-y coordinates of points on a curve generated by plotting the old values on the x-axis against the new values on the y-axis. In entering the transform, the user first enters the number of coordinate pairs to be used, then the pairs of numbers themselves. If a large number of points is to be entered, it's fine to go from one line to another and, in fact, IDEPP will wait until the indicated number of coordinate pairs has been input.

If an old data point lies between input curve points, the new value is generated by a simple linear interpolation. This means that enough coordinate pairs must be input to give accurate results, particularly where the curve changes sharply.

## 2.25 ER - Edit Recovery

With this task it is possible to recover any previous edit operation, even one from an earlier session. Recovery is possible because a history of each edit is kept for viewing, and because all the records changed by a particular edit are stored in a scratch file. The edits are numbered sequentially and the edit number can be used at any time to recover all the original records and re-insert them in the file.

After invoking the task with the verb "ER", IDEPP requests the number of the edit operation to recover. When input, IDEPP responds with the documentation of that edit and asks for confirmation of the command. When the edit has been recovered, IDEPP notes the edit number and time of recovery in the edit history file. Thereafter, whenever the edit history file is viewed, the record of that recovery will appear.

Some confusion can occur if there are overlapping edit operations (that overlap in the data) and only one of them is to be recovered. The reason is that records in the scratch file representing the data before a particular edit will also represent the data before all subsequent edits. If, for example, edit 5 is to be recovered and edit 8 overlapped (timewise), then recovering edit 5 will also recover the overlapping part of edit 8. One way of dealing with this problem is to recover 8 first, then 5, then re-execute edit 8 (which would appear under the current edit number). Another way would be to recover 5, then re-execute edit 8 on just the overlapping portion of the data. If the edit that is to be recovered is in the current editing session, it might be easier (and cheaper) to simply end the session without saving any of the edits done during that session and start afresh from where you left off at the end of the previous session.

### 2.3 Utility Tasks in Detail ET, TT, TY, PR, EC

This group of tasks provides ways to look at the data and output it, either partially or its entirety, and to keep track of what is happening during the editing/plotting session. Utility tasks are for the convenience of the user and in no way alter the data.

#### 2.31 ET - Viewing Histories of Previous Edits

As the title suggests, this task is used to find out the relevant parameters of some previous edit. After invoking the task with "ET<cr>", IDEPP will prompt for entry of the edit number. Upon entry, of this number (0 if all edits are wanted) IDEPP will respond with a history of the edit similar to that displayed at the end of an edit operation. An example follows:

E\*ET

WHICH EDIT DO YOU WANT?

E\*3

YOU WISH HISTORY OF EDIT 3? Y/N Y

EDIT: 3 COMPLETED: 24-Feb-81 AT 11:43  
CO2 WAS EDITED PARAMETER  
DE WAS EDITING TASK  
SITES AFFECTED WERE 2 , , , ,  
SCRATCH FILE RECORDS= 19 TO 22  
TIME INTERVALS WERE:  
4/25/1200 TO 4/25/1300

E\*

The file containing the history of each edit is actually a separate file on the disk and can be output to a line-printer via Monitor commands (not during an editing session). The file name will be DSK:EDTSK.DAT[PPN,EXPMT], where DSK, PPN, and EXPMT are supplied by the user.

If an edit session was scrubbed by exiting IDEPP without saving any of the edits, an entry to that effect is entered in the edit history file and the history of that session's edits deleted from the file.

The edit history file provides an easy way to keep track of data manipulations. (It's a good idea to obtain a hard copy of the file at the end of each editing session.)

### 2.32 TT - Viewing Edits in Progress

This task enables the user to view each data record as it's changed during an edit operation. A line of data is output to the TTY for each edited record, so that the user can see which records are being changed, and how.

For clarity (and in the case of overlapping edits) the output line contains not only the parameter being edited but a subgroup of all the parameters in the record. The EEB data are divided into three subgroups as follows:

- Group 1: CO, CO2, NOX, NO2, NO, O3, SO2, TO, MF1, MF2
- Group 2: T1, T2, T3, T4, RH1, RH2, RH3, RH4, RHO
- Group 3: WSP, WDR, RAD, TEX, TCB, RA1, RA2, RA3, RA4

When invoking the TT task, the group number is entered at the same time, thus: TT,1. IDEPP confirms the command and, until disabled by an "NT" command, will print out at the TTY the group requested. To change to another group, just enter that command--no NT is needed.

After a CRT page of data is output, IDEPP pauses so that the user can examine the data. A <cr> entered at this time will tell IDEPP to simply proceed. A "Q<cr>" will cause typeout to cease, although the edit will proceed.

### 2.33 PR - Printer Output of Data

This task enables the user to create a disk file containing selected portions of the data for subsequent off-line printing. The format includes most, though not all, of the parameters in each record, and has time and site selection identical to other tasks. Other user inputs are prompted for, including the name of the file to be generated and whether it will reflect the edited or original data. Previously deleted records can be included if wanted. If ratios were calculated during the current session, they will appear in place of the standard deviation entry for the affected parameter.

### 2.34 EC - Exiting to IDEPP Command Mode

This command simply returns the user to IDEPP command mode, from whence one can go to graphics mode, or quit, etc. Going from one mode to another can be done at any time, and has no effect on the data or on the TT "switch".

### 2.35 TY - TTY Data Output

With this task the user can output any selected portion of the data on the terminal, either from the original (OR) or the working-copy (WC) file and including or excluding previously deleted records. IDEPP prompts for these inputs when the task is invoked, and asks which output group is wanted. These groups are the same as for the TT task. After each page output, IDEPP pauses for a user response. If this response is <cr>, another page is output; a "Q<cr>" will stop the operation, and "N<cr>" will go to the next time interval, if there is one; otherwise it, also, will stop the operation.

### 2.36 SC - Scan For Data Within Given Ranges

This task enables the user to scan through the data for values of a given parameter lying within the given ranges. Time and site entries are entered in the normal fashion, then IDEPP prompts for the source file (OR for original data file, WC for IDEPP working-copy file). They are entered as FF,DDD<cr> where FF is OR or WC and DDD is ALL or NO (for example, OR,NO or WC,ALL). IDEPP then prompts (R\*) for a range, expecting two floating-point numbers and <cr>. Up to five ranges can be input, with IDEPP prompting with a R\* for each range. An unaccompanied <cr> signals no more ranges and IDEPP proceeds. Output is the same as with the TY command, and offers the same user responses.

### 2.37 AV - Means and Standard Deviations

This task generates the mean, standard deviation, range, and number of data points included for a given parameter, set of time intervals, and sites. IDEPP will also prompt for whether the original or working-copy file is to be referenced. Computations are output for each discrete time interval, and for all the intervals together. No deleted records or values are included. This task can be used in conjunction with the "TT" task to view each data point included in the calculations.

### 3.0 Plotting Data With IDEPP

IDEPP plots data on either one of two types of CRT terminals and one type of pen-plotter. The two CRT devices are the Tektronix 4010-4014 series graphics terminal, and the Lear-Sigler ADM3A terminal with the Retrographics 512 graphics card added to give it graphics capability. The pen-plotter is the Hewlett-Packard 7225A with the 17603A Personality Module. Any upward-compatible HP plotter can also be used, such as the 7220S.

The plots generated on the CRT are for internal use, as are some of the pen-plotter plots; however, high-quality histograms suitable for publication can also be generated on the pen-plotter.

IDEPP generates two types of plots. One is the standard time vs. value or t/v plot. The period covered in the plot is always a single day, from midnight to midnight. The other type of plot is a histogram displaying the range of data on the x-axis, split up into bins, with the percentage of data points lying within the range of each bin plotted on the y-axis in bar-graph fashion. Time and site selections for histograms are as flexible as those used with the various editing commands.

The pen-plotter can be used to generate overlay plots, such as those comparing ambient to indoor levels or one experiment to another.

The command language for graphics mode is similar to that of IDEPP command and edit modes, particularly the task, parameter, time and site entries. The following sections, paragraphs 3.1 - 3.19 describe IDEPP's graphics capabilities in detail.

#### 3.11 G1 - Plot One Day of Data on the CRT

This command enables the user to plot one day of data for a specific parameter on the CRT screen. Since the routine simply plots one day, the time entry need only be the two integers indicating the month and day. After the required inputs are entered and confirmation of the command received by IDEPP (if in beginner mode), the screen is cleared, the grid and plot displayed, and the time and value of all the first 40 points in the plot are output below the plot. The titles include the date, site, experiment name, parameter, and parametric units. An example of the command sequence follows:

```
G*G1,C02
T*8,15
T*<CR>
S*2
PLOT C02 SITE 2 FOR AUG 15 RIGHT? Y/N Y<CR>
```

How fast the plot is generated depends on how busy the computer is, but typically the plot is returned within a minute.

### 3.12 GN - Graph Next Day's Data

This task is for the convenience of the user. Entering GN after a G1 task has been completed will simply plot the next day's data for the same parameter and site. Consecutive GN tasks can be invoked and will retain the original parameter and site entered under the G1 task.

### 3.13 HP - Graph Histogram for a Given Period

This task plots a histogram for the parameter, site(s), and time intervals input. The time and site entries are quite flexible and are described in detail in paragraphs 1.22 and 1.23. Depending on the particular experiment, histogram, and how busy the computer is, it can take as long as several minutes to generate the plot. There is a default range for histograms for each parameter. If a particular value lies outside that range, the value and time is displayed on the TTY and the maximum or minimum bin incremented.

### 3.14 RT - Repeat CRT Plotting Task for Different Parameter

This is another convenience task which enables the user to easily go from parameter to parameter over the same time intervals, plotting either t/v graphs or histograms. All that is required is the task and the new parameter. If the beginner switch is set, then each new parameter in a histogram plot will have to be confirmed; otherwise the plot will simply proceed.

### 3.15 RS - Repeat CRT Plotting Task with New Site Selection

This convenience task allows the user to repeat a t/v or histogram plot with different site(s). After the RS<cr> entry, IDEPP returns the S\* prompt for site selection exactly as in any other operation requiring it.

### 3.16 SR - Change/View the Plotting Constants

Within IDEPP there is a table of default plotting constants. For each parameter there are entries for t/v maximum, t/v major grid divisions, histogram maximum, histogram major grid divisions, histogram bin-size, and t/v-histogram minimum. Any of these can be viewed and/or changed via the SR command. To invoke the command, simply enter the verb, the parameter of interest, and <cr>. IDEPP will display the current constants for that parameter and prompt for and confirm any changes the user makes. Note that since the plotting constants are initialized to their default values when IDEPP execution begins, any user changes via the SR command don't persist from session to session.

### 3.17 PG - Plot One Day of Data on the Pen-Plotter

This task essentially replicates a CRT plot of t/y data on the HP7225A pen-plotter (see Figure 1). Since the routine expects a G1 task to have been just previously completed, there's no need for any user

input other than PG<cr>. Note, however, that the TTY baud rate must be set at 2400 (the HP7225A maximum).

The plotter used with the EEB data has been the HP7225A with the 17603A Personality Module. The plotter is connected to the computer via the auxiliary RS232 connector of the ADM3A terminal.

Since the plotter is initialized by IDEPP each time a pen-plotter is asked for, it can be left off except when it's to be used.

Any user of the HP7225A should become familiar with its care and operation via the documentation supplied with the plotter.

### 3.18 PH - Plot a Histogram on the Pen-Plotter

This task, like the PG task, essentially replots the current CRT plot on the pen-plotter, in this case a histogram. It must, therefore, be immediately preceded by a CRT histogram task (invoked by the HP, RT, or RS command).

Two types of histograms can be plotted: one for "internal" use and the other for "publication". The differences are in format and size, as Figures 2 and 3 illustrate. IDEPP prompts for the maximum percent frequency and, with publishable lots, the relative size. This information facilitates customizing the histogram plot for a particular report.

In addition, overlays of more than one plot are possible. After an initial plot another CRT plot (with the same parameter and plotting constants) can be generated, and a second pen-plot asked for. IDEPP asks whether or not the current plot is to overlay a previous one and, if so, skips the grid plot and appropriately relocates the current titling (see Figure 4).

## 4.0

## IDEPP Program Description

This section is directed to the programmer who wants to understand more about how IDEPP is structured, and/or wants to debug or modify the program, to either add tasks or to handle different data sources. Included in the section is a control tree relating the various routines, and a description of some of the more relevant variables. The major source of program description, however, is the source code itself, where an effort was made to include enough commentary to facilitate understanding and modification by programmers other than the author. The following sections are a complement and introduction to the source code.

### 4.1

### The Modular Structure of IDEPP

IDEPP was written in a top-down, modular fashion. Top-down means that the overall architecture and command/information channels were configured before any of the routines were actually written. Modular fashion means that each distinct program task was accorded its own routine. While this structure produces a relatively large number of sub-routines (approximately 45) debugging and modification are more easily accomplished.

In particular, IDEPP's source code is in three segments, ILINK.FOR, ELINK.FOR, and GLINK.FOR. ILINK.FOR is the root link and always must be in core. ELINK.FOR and GLINK.FOR, however, need only be in core when the respective mode of operation (edit or graphics) is in execution. ELINK.FOR contains all the routines specific to editing operations, and GLINK.FOR contains those specific to graphing operations. ILINK.FOR contains the routines used by both of the other segments for data retrieval and storage, along with job initialization and termination. The advantage of this setup is that it isn't swapped in and out of core as often in a busy time-sharing environment, and that it can be easily set up as a segmented program with overlays.

A major benefit of the modular approach is that it allows for easy addition of new editing or plotting tasks. In the case of editing tasks, the direct operation on the data is a defined subroutine, with a small associated code sequence for each task in the editing executive routine, ECOMND, where the commands are decoded, and in the task executive, EDATA, where the data is retrieved and most of the book-keeping operations performed. Adding a new editing task usually requires inserting a task routine to do the desired operation on a single data record, and the associated code sequences in ECOMND and EDATA.

For graphing operations, each plotting routine receives via COMMON the array of points to plot, the plotting constants (such as range, etc.), and titling information. It then does the actual plotting, whether on the CRT or on the HP pen-plotter. This device-dependent routine, plus associated code in the graphics executive routine GCOMND and the data acquisition routine DATAQ, are usually the only additions necessary to bring up a new plotting task.

Since IDEPP first started being used, five new plotting tasks and one new editing task have been added without any major changes to the overall structure of IDEPP.

#### 4.2 Input/Output Details

IDEPP is set up to work with random-access data files residing in a first-level subdirectory on a de-mountable disk-pack on the DEC10 system in Cory Hall on the UC Berkeley campus. When starting up, IDEPP asks for the subdirectory name and the symbolic disk structure name pointing to the data. IDEPP expects a random-access ASCII file named CDATE.DAT with a record length of 480 characters or 96 words containing the data to be worked on. IDEPP creates a copy of this file in the same directory named EDIT.DAT and it is this file that is altered by editing operations. At the end of an IDEPP session EDIT.DAT can be copied to CDATE.DAT, thus preserving the changes made in the current session, or not copied--effectively scrubbing the session.

One other random-access file named SCRACH.DAT is created by IDEPP to contain the original version of altered records and the bookkeeping to reinsert them in EDIT.DAT if wanted.

This provision demands that there be enough file space for CDATE.DAT and its copy and for SCRACH.DAT. In fact, it's quite possible that SCRACH.DAT will grow to the size of the other two files if a lot of editing is done. If space becomes a real problem, SCRACH.DAT could be deleted to provide new space, although recovery of any edits before the deletion would be impossible, of course, and the numbering of the edits would start over at one.

More detail on the format of the data files can be found in the routines DATARD, RECOVR and OUTONE. These routines are called to do all reading and writing of the three data files. The routine DINIT, which initializes an IDEPP session, opens the files and contains information about the file structure. The routine PARCK links the numbers in the data to the mnemonics assigned each parameter, such as C02, 03, etc.

#### 4.30 IDEPP Structure and Control Flow

As mentioned earlier, IDEPP is divided into three segments or links. One link is for overall control and session initialization and termination, and the other two are for editing and graphics, respectively. In this section the flow of control and what the various routines do is described in moderate detail. A subroutine map and a list of definitions of variables is included in this section.

The root link of IDEPP (in files ILINK.FOR and ILINK.REL for the source and relocatable binary code, respectively) is responsible for program initialization and termination and acts as a bridge between the graphics and editing links. When IDEPP begins execution, the user is asked where the data file resides and what sort of terminal is being used. Given this information, the routine DINIT is called to initialize the session. IDEPP ends a session from this link by calling the routine FINIS. The inclusion of a number of utility routines used by two or more

of the links reduces swapping and facilitates overlays.

The routines that edit data are in the edit link (ELINK.FOR and ELINK.REL). The entry routine from ILINK is the routine ECOMND, which conducts the dialogue with the user as to which edit operation is to be performed and the times and sites involved, etc. ECOMND then calls EDATA to coordinate the actual operation. EDATA calls the utility routines to retrieve and writes the data and the particular task routines to actually modify a data record. When the task is completed, it returns to ECOMND for the next user request.

The graphics routines are organized quite similarly to the edit routines. They reside in the graphics link (ILINK.FOR and ILINK.REL) and are entered from ILINK. The entry routine is GCOMND and it conducts the user dialogue and obtains the inputs necessary to do a plotting operation. DATAQ is then called to retrieve the data and move them to plotting arrays, which it does through the same utility routines used by ILINK and ELINK. Then GCOMND calls the appropriate plotting routine which generates the plots. For CRT plots, this means calling routines from the PLOT10 software on the DEC10 System. For HP7225A pen-plotter plots, this means actually driving the plotter directly with formatted write operations. A plotting operation control then returns to GCOMND and awaits the next user request.

#### 4.31 IDEPP Subroutine Map

The map below diagrams the interrelationships of all the various routines in IDEPP. Each routine is called by a routine to the left, or calls a routine to the right, or both. The lines indicate the calling trees. Some routines have alternate entry points, which are listed in parentheses. Some routines are listed more than once to avoid too many line crossings. Such entries have an \* or # suffix, with the \* indicating that the routine also calls other routines and that the rest of the tree for that routine appears at some other entry on the map. (Calls to routines external to IDEPP are not included.)

```

IDEPP ----- ECOMND ----- TTYR* ----- HELPS#
|----- EDTYPE
|----- RECOVER ----- OUTONE#
|----- PARCK#
|----- PRINIT*
|----- RAINIT*
|----- TYINIT*
|----- AVINIT*
|----- EDATA ----- EHIST (UHIST)
|----- ADDONE -----
|----- DELALL -----
|----- DELONE -----
|----- FITONE -----
|----- PRNTIT
|----- (PRINIT, PRTFIN)
|----- RATIOS ----- |----- OUTONE
|----- (RAINIT) |----- (OUTSCR)
|----- AVGIT----- |----- KYRHMN
|----- (AVINIT,AVFIN)
|----- SCANIT |
|----- |----- TYINIT ----- TTYR*
|----- (TYPIT, TYRINIT)
|----- TIMREC ----- DATARD#
|----- |----- KYRHMN (KYRMIN)
|-----
|----- GCOMND ----- DATAQ ----- |----- DATARD
|----- |----- |----- TIMECK (TIMCK1)
|----- |----- |----- TIMREC
|----- |----- |----- KYRHMN#
|----- |----- HISTOR
|----- GRFIDY ----- CODIT
|----- GRFNXD
|----- HSTPER
|----- PLTIDY
|----- PLTHST ----- TTYR*
|----- |----- NROUND
|----- |----- LABX
|----- TTYR*
|----- FINIS |----- HINIT ----- AZERO
|----- HELPS# |----- TTYR*
|----- IDOC (IDOCID) |----- PARCK#
|----- DINIT ----- DATARD* |----- HELPS#
|----- |----- COPYIT |----- AZERO
|----- |----- KYRMIN
|----- |----- GINIT

```

Description of Significant Variables  
in IDEPP

Below is a list of the more significant variables used in IDEPP, including all those in any COMMON statements, along with a brief description of how and where the variable is used. Most of the variables aren't listed here, such as DO indices, temporary storage variables, and others. These variables are either explained in comments in the routine where they appear, or are self-evident.

The following group of variables are those in common GCOM.

- ADALY - The number of points in a daily time vs. value plot.
- AFIT - An array containing user-input values describing a transform to be performed on the given parameter. Consists of pairs of values indicating mapping of old to new data.
- BINRNG - An array containing plot constants for all the parameters IDEPP works with. Set initially to default values in routine GINIT and changed, if wanted, in routine GCOMND
- DBARH - Running sum of parametric values for calculating mean and standard deviation on a one-day time vs. value plot.
- DBARH2 - Running sum of squares of values for calculating standard deviation on a one-day time vs. value plot
- DVNDMN - Minimal acceptable dividend in a ratio operation (user input)
- DVSRMN - Minimal acceptable divisor in a ratio operation (also user-input)
- GDATA - Array where current data record is stored.
- GSTOR - Array containing points to be plotted in a time vs. value plot.
- HSTOR - Array of value frequencies for histogram plots.
- ISTOR - Array used to temporarily store current-task parameters for possible later use
- IEDEX - Core image of first ten records of the scratch file, SCRACH.DAT, where scratch-file pointers for each edit are stored. Each edit is accorded three entries: the edit number, and the first and last record number in SCRACH.DAT containing the originals of records affected by the edit.

- IPEAR - Array relating physical units of each parameter to its location in the data array GDATA
- ITEND - Pointer to last entry in table KTIMX
- ITYR - Pointer to end-of-year entry in KTIMX where the data wraps around to a new year
- KEDT - Number of current edit operation; updated from edit to edit and session to session and adjusted for scrubbed sessions.
- KEDT1 - Number of first edit in current IDEPP session; used to adjust edit history and index if session scrubbed
- KFIRST - Year-minute of first data record in data files CDATA.DAT and EDIT.DAT
- KLAST - Year-minute of last data record in CDATA.DAT and EDIT.DAT
- KEND - Eight-digit integer concatenated from the mo/day/hr/min of the end-time for a time interval in an editing or plotting operation
- KLAG - Maximum time in minutes allowed between a divisor and a dividend data record to calculate a ratio in a ratioing operation
- KOPRAT - Array relating location of computed ratio to the parameter ratioed in the data array GDATA; used in PRNTIT routine to output ratios
- KTYEAR - Year-minute of last time in old year of a data file that overlaps a year-boundary
- KYREC - Record number of last time in old year of a data file that overlaps a year-boundary
- KTIMX - Array that serves as a table to facilitate looking for a specific time in the data file; entries are recorded pointing to the first and last record in the data file, a year-boundary record (if encountered), and every fortieth record.
- KSITS - Integer array indicating sites selected for a specific editing or plotting operation
- KWIND - Array containing time-interval entries for the current editing or plotting operation
- KERR - Contains alphanumeric flag indicating large standard deviations or IDEPP-deleted records

- LWIND - Array containing time intervals for current task (in a compressed format for comparison with data times). format is an eight-digit integer concatenated from the month, day, hour, and minute; thus 102378 means 10/23/78, and 050280 means 5/2/80
- LSTART - Start time, in 0.01 hrs on a 24-hr clock, of the time interval for a task to be worked on during a certain interval each day
- LEND - End time related to LSTART.
- LNUM - Used to pass current record number in various operations
- ND - Number of floppy-disks on which data were originally recorded in the EEB mobile laboratory
- NDIVSR - Number of divisor site in ratio operations.
- NBSWCH - Switch indicating whether user wants to operate in beginner or expert mode
- NRPT - Flag indicating whether current time interval is "continuous" (C) or "interval" (I)
- NLAST - Number of last record in data file
- NFITPR - Pointer to last entry in array AFIT used in FITONE routine to perform transforms on a given parameter
- NRAT - Integer to keep track of how many ratios have been performed in a given IDEPP session
- NLIN - Alphanumeric array indicating site selection for a particular task
- XBIN - Array used to plot X-axis component of histograms
- PARNUM - Pointer to GDATA array indicating current parameter of interest
- TSTOR - Array containing times of points in daily time-vs.-value plots (in units of 0.01 hours)
- TDISP - Array containing times of points in daily time-vs-value plots in a format for printing out.

The next few variables are in the common HCOM

- HMAX - Maximum value observed in a histogram plot

- HMIN - Minimum value observed in a histogram plot  
IXMIN - Number of negative bins in a histogram plot

The following two variables are in common SCNCOM.

- SCN - Array containing up to five ranges to scan for in a scanning operation  
NRX - Number of ranges in a scanning operation

#### 4.34 Description of IDEPP Routines

The titles and descriptions of each routine in IDEPP are presented below in the order they appear in the respective source codes.

##### PROGRAM IDEPP

This is the main routine. It accomplishes session initialization via routines DINIT and COPYIT and session termination via FINIS. The user dialogue is established and input received to open the correct data file. This routine is the bridge between the edit and graphics modes, with control passing to ECOMND in edit mode or GCOMND in graphics mode. Access to IDEPP documentation is coordinated via the IDOC routine. The beginner switch is set and reset here.

##### SUBROUTINE DATARD (\$,\$,NUMREC,LUNIT)

This routine reads a single record of data from the random-access data files, given the record number via NUMREC and which file to read via LUNIT (either the original or working-copy data file).

##### SUBROUTINE DINIT (INFIL, KDEV, NUMREC)

This routine initializes an IDEPP session. It initializes variables and calls GINIT to set default plotting constants. The routine COPYIT is called to do a cheaper, fast binary copy of the original file to the working-copy file. Then the various disk files used by IDEPP are opened and the edit history file pointer positioned for new edits and the number of the most current edit stored. Finally a table of log-time vs record number for every 40th record is generated to speed data access and some more variables initialized.

##### SUBROUTINE GINIT (RNGBIN)

GINIT copies the table of default plotting constants set by a DATA statement to the table used by the plotting routines.

#### SUBROUTINE HINIT (DUMMY)

HINIT initializes the data collection operation for a histogram plot.

#### SUBROUTINE HELPS (KFOR)

HELPS types an informative message on the TTY in response to a user request or if the beginner switch is set. The message is selected in the call via KFOR.

#### SUBROUTINE KYRHMN (KYMIN, JMON, JDAY, JHOUR, JMIN)

This subroutine calculates the number of minutes elapsed from the beginning of the year (a non-leap year), given the current time in the call as month, day, hour and minute and returns the result as an integer. There are two entries, one for the hour and minute as two integers, and one for the hour and minute concatenated into one integer.

#### SUBROUTINE OUTONE (KUNIT, NOREC)

This routine writes a record to a data file, given which data file via KUNIT and which record via NOREC. A second entry writes a record to the scratch file, given the scratch file record number via NSCREC and including in the record the record number from the working-copy file passed via IOLREC.

#### LOGICAL FUNCTION PARCK(KPARAM)

This function subroutine checks the validity of the parameter mnemonic passed in KPARAM against a table of recognized mnemonics and, if found, establishes the pointer to various arrays. The pointer is PARNUM in common GCOM and points to the parameter value in the data array GDATA and to the mnemonic for the parameter in the IPEAR array and the mnemonic for the physical units in the IUNITS array.

#### SUBROUTINE TIMREC (\$, KMON, KDAY, KHOUR, KERIC, KUNIT)

This routine locates the record in either the working-copy or the original file with a time closest to but after the time passed in the call. It returns the number of that record minus one (for loop convenience in calling routines)

#### SUBROUTINE TIMECK (\$,\$,D)

TIMECK determines whether the time passed it via NTIME in common is within the current user-input time interval being scanned under control of routines EDATA or DATAQ. Alternate returns are taken depending on the result.

SUBROUTINE TTYR (KLEV,KF,\$,TY1,TY2,TY3,TY4,TY5,  
1 TY6,TY7,KY1,KY2,KY3)

This routine reads user input from the TTY according to the format passed in the call via KF. Requests for help are coordinated here, with the initial response indicated by KLEV in the call. User errors in input are trapped and the alternate return taken.

SUBROUTINE COPYIT (\$,LIN,LOUT,NAMIN,NAMOUT,KDEVIC)

COPYIT does a faster, cheaper binary copy of the original data file to the working-copy file and vice-versa.

BLOCK DATA SETIT

This routine sets the arrays most often changed when changing IDEPP to handle a new data format. The common block /SETCOM/ is particularly used for this and only appears in routines where it's needed. These routines are:

GINIT	PARCK	ECOMND	RATIOS
GRF1DY	HSTPER	PLTIDY	PLTHST

SUBROUTINE ECOMND (KDEVIC)

ECOMND coordinates all edit-mode operations. It conducts the user dialogue, calling TTYR for input, checks parameter specifications via PARCK, and decodes the task, time and site entries, and other inputs as needed. EDATA is called to perform the task and record it.

SUBROUTINE EDATA (\$,KTYF,LTSK,LPARM,KINT,KUNIT)

This routine coordinates a particular edit operation. It does any initialization necessary for various types of edits. It locates the records to be edited, calling TIMREC and DATARD and TIMECK. It carries out the edit operation on each record of interest, calling the appropriate routine. It updates and maintains bookkeeping on the scratch file for edit recovery. EHIST is called to record the edit when completed and control returned to ECOMND.

SUBROUTINE EDTYPE (NEDIT,KTASK)

This routine outputs all or parts of the edit history to the TTY. If NEDIT = 0, the history of all the edits is output. If NEDIT = n, the history of the nth edit is displayed, plus all edit-recovery entries.

SUBROUTINE FINIS (\$,KSAVF)

This routine wraps up an IDEPP session. It rewrites the first 10 records of the scratch file (SCRACH.DAT) containing the edit index. It closes all open files. Whether or not the current session's editing results are to be saved are indicated in KSAVF and if the no-save option is chosen, the edit history file is truncated to exclude the current session's edits.

SUBROUTINE RECOVR (LEDT)

This routine uses the edit number passed via LEDT to scan the edit index IEDEX for the range of records in the scratch file containing the original records supplanted in the working-copy file by the particular edit. These records are read from the scratch file and re-inserted into the working-copy file using the working-copy record number inserted at the end of the record when written to the scratch file by OUTSCR. A record of the recovery operation is stored in the edit history file.

SUBROUTINE EHIST (KEDT,LPARM,LTSK,KSITS,NLIN,KWIND,  
1 IHDEX,KINT,ACONST)

This routine writes a synopsis of the just-completed edit operation to the edit history file (EDTSK.DAT) for reference and documentation purposes. It isn't used by IDEPP at all as the first 10 records of the scratch file contain what IDEPP needs. The synopsis includes:

1. The date and time the edit was completed
2. The parameter worked with
3. The time intervals and sites selected
4. The record numbers in the scratch file where the original records are stored
5. Any other relevant parameters, such as the constant in a addition operation

In addition the synopsis is output to the TTY.

SUBROUTINE ADDONE (\$,\$,IADREC,ISCREC,ACONST)

This routine adds the user-input constant ACONST to the selected parameter, first writing the original record to the scratch file, and afterwards writing the modified record to the working-copy file.

SUBROUTINE DELALL (\$,\$,IDAREC,ISCREC)

The variable KERR is set to "DELET" in the current data record, having first saved the original in the scratch file. (Note that a record can't be literally deleted from a random-access file.) This flag is used by other IDEPPP

routines to ignore the record in subsequent editing and plotting operations.

SUBROUTINE DELONE (\$,\$,IDEREC,ISCREC)

DELONE substitutes the skip flag SKIP1 for the chosen parameter in the data record, first saving the original in the scratch file. Other IDEPP routines interpret the flag to mean skip that data point in editing and plotting operations.

SUBROUTINE FITONE (\$,\$, ICFREC, ISCREC)

FITONE utilizes a user-input array of pairs of values mapping old values to new values to transform the parameter of interest. Linear interpolation is used to transform values lying between input old values. The original record is saved in SCRACH.DAT.

SUBROUTINE PRNTIT (\$,NDEL)

PRNTIT generates a sequential file with a subset of all the parameters in a format suitable for line-printer output. (There are too many parameters to fit on one line.) Either the working-copy or original file can be the source. Whether or not "deleted" records are output is indicated by NDEL and the filename is user input. If ratios have been computed in the current IDEPP session, they are output in line-printer format in the standard-deviation slot for the relevant parameter (directly beneath on the next line). This is true, of course, only if the relevant parameter itself appears in this compressed output format.  
NOPFIL = user-input filename  
NDIR = directory of input file

SUBROUTINE SCANIT (\$,\$,\$)

SCANIT scans either the working-copy or original data file within the input time intervals for values of the input parameter lying within the input ranges. If found, the TYPIT routine is called to display the relevant portion of the data record on the TTY.

ENTRY TYPIT (\$,\$,\$)

This routine outputs data from either the original or working file to the TTY. The format is oriented to the 80-character/line limit of an ADM3-A and, so that it all fits on one line, only part of the data record is output. To do this the data is divided into three groups and the relevant group output rather than the whole record. Given the parameter wanted, ECOMND determines which group is wanted and passes the information to TYPIT via NGROUP, causing the routine to branch to the appropriate "write"

statement. Which parameters are in which group is defined by the "write" and "format" statements in this routine. Whether or not to include deleted data is passed via NDEL and acted on.

#### SUBROUTINE AVGIT (\$,NINT,KTF)

This routine generates means and standard deviations for a particular parameter for each user-input time interval, and for all the time intervals together. These are output to the TTY.

#### SUBROUTINE GCOMND (DUMMY)

GCOMND is the command executive for graphics-mode operation. It conducts most user dialogue, decoding commands, checking parameter validity, converting time intervals to integer pairs, initializing site selection, and handling various user errors. When the plotting task has been defined, DATAQ is called to coordinate data collection, and then the appropriate plotting routine called to output the actual plot.

#### SUBROUTINE DATAQ (\$,LTSK,LPARM,KINT,KUNIT)

This routine coordinates the initialization and data retrieval for a plotting operation. TIMECK and TIMREC are called to find and select the data based on user-input time intervals and site selections. HISTOR is called to store data for histograms and develop means and standard deviations for all plots. When done, the routine exits to GCOMND.

#### SUBROUTINE HISTOR (DUMMY)

HISTOR takes the current parameter value in GDATA(PARNUM) and increments the appropriate "bin" in the HSTOR array to develop a histogram. Binsize and range are given via the BINRNG array. In addition calculations for means and standard deviations for both histograms and time vs. value plots are updated here.

#### SUBROUTINE GRFIDY (\$)

GRFIDY takes the arrays GSTOR and TSTOR filled by DATAQ and other variables and plots the values of the selected variable vs time for the single selected date, from midnight to midnight. Use is made of the Tektronix PLOT-10 graphics software for the Tektronix 4014 graphics terminal to drive either the 4014 or an ADM3-A terminal with a RG-512 Retrographics card installed with the PLOT-10 option.

#### SUBROUTINE GRFNXD (\$)

This routine modifies the time-interval array KWIND so that it refers to the day after the day originally input by the user. Care is taken to properly change the array if a month or year boundary is crossed.

#### SUBROUTINE CODIT (NC,NARAY,KARAY)

CODIT accepts an ASCII array NARAY, with NC characters and outputs an array, KARAY, containing decimal equivalent integers for each ASCII character.

#### SUBROUTINE HSTPER (\$,KINT)

HSTPER plots a histogram on either a Tektronix 4014 graphics terminal or an ADM3-A terminal with the RG-512 Retrographics Card and PLOT-10 option. The arrays XBIN and HSTOR developed by DATAQ and HISTOR are used in generating the plot, along with range, mean, and standard-deviation results.

#### SUBROUTINE PLT1DY (\$)

PLT1DY essentially replicates the CRT plot generated by the GRF1DY routine on the Hewlett-Packard 7225A pen plotter. All inputs are the same, and, in fact, this routine only works if the plot is invoked immediately following a time vs value plot on the CRT.

#### SUBROUTINE PLTHST (\$,KSIZ,LPUB,KINT,KOVR,LTYP)

This routine essentially replicates on the Hewlett-Packard 7225A pen plotter the histogram just displayed on the CRT via the HSTPER routine. There are two types of histogram plots possible, a plot for internal use, and a publishable plot. The data displayed is the same; only the format differs to conform to rules for publishing graphs. In addition the routine conducts a dialogue directly with the user to set the physical size of the publishable plot.

#### 4.40 Modifying IDEPP to Accept New Data Formats

IDEPP was written with the intent that it be easily changed to accept a different data format and be useful for data-analysis operations other than those related to the EEB Mobile Laboratory. This section describes in some detail how to accomplish such a modification.

The first thing to consider is whether or not the new data is in a format that is compatible with IDEPP. If not, then some sort of pre-processor has to be written to generate a file of the new data in an acceptable format. The essential characteristics of this format are:

1. The file be a random-access ASCII file made up of data

- records all of the same length.
2. Each record have a time marker indicating when the data in that record was recorded. The marker can include the month, day, hour and minute; the year and second can be there, but they are not considered by IDEPP.
  3. Note that the previous two requirements mean that each data record in the file has a slot for all parameters of interest. Where different parameters are recorded at different intervals, this may mean inserting deletion flags in data records where the parameter wasn't measured, or replicating the measurement in those records.
  4. There be a 5-character slot in the data record for the deletion flag.
  5. The data be arranged in chronologically ascending order.
  6. The data file be named CDATE.DAT and reside in a first-level sub-file directory. The file spec. might then be CDATE.DAT [3300,101,MFD1B].

The preceding are the only data-formatting requirements for the new dataset. In addition, IDEPP can use the following data in each record if desired:

1. An integer (0-99) indicating an experimental state (such as calibration, power-down, etc.) or a sampling site in the case of multiplexed data. IDEPP will use this to plot and edit the data from one or more states/sites.
2. An experiment title of up to five characters. This is used in titling plots and printed output. If not there, no title is used.
3. A one-to-three digit integer that is simply output on some forms of the printed output from IDEPP. For EEB data this is the number of the floppy disk on which the data was originally recorded.

When the data file is in an acceptable format, the next step is to do the necessary modifications to IDEPP. The two major types of modifications usually needed are to the read/write statements and format statements controlling the data I/O operations and the various arrays within IDEPP that reference or point to the data and describe it in user terms.

IDEPP has one routine that reads from either the original or working copy of the data file and one routine that writes to the working-copy file and simultaneously to the scratch file. The copying of one file to another during initialization and termination is done in binary mode in a third routine. The read/write statements and format statements in the first two routines (OUTONE and DATARD) usually need to be modified for a new format, but the binary-mode routine, COPYIT, usually needs no modification. The routine RECOVR, which reads from SCRACH.DAT, also usually needs some modification of the read and format statements.

The opening of the data files done in the routine DINIT and also may need changes, usually just of record size for the random-access files.

The other major area of changes is represented by the various arrays that define the dataset and relate the numbers in the data record to their respective parameters and physical units. These are initialized via data statements and are gathered together in the block data routine SETIT. Comments in that routine describe what each array or variable is for.

Even though considerable thought and effort have been put into making IDEPP adaptable to new data formats, it still requires someone with relevant programming experience to actually make the changes and check out the modified program. As of this writing, IDEPP has been modified to handle data from the Aardvark Infiltration and Radon Monitor, and comparison of the two source files might be instructive in dealing with an IDEPP modification. A list of the differences can be obtained by running the TOPS10 program, FILCOM.

This report was done with support from the Department of Energy. Any conclusions or opinions expressed in this report represent solely those of the author(s) and not necessarily those of The Regents of the University of California, the Lawrence Berkeley Laboratory or the Department of Energy.

Reference to a company or product name does not imply approval or recommendation of the product by the University of California or the U.S. Department of Energy to the exclusion of others that may be suitable.

TECHNICAL INFORMATION DEPARTMENT  
LAWRENCE BERKELEY LABORATORY  
UNIVERSITY OF CALIFORNIA  
BERKELEY, CALIFORNIA 94720