
IPF ADVANCED User's Guide

*User Documentation
Interactive Powerflow Program*

Prepared by:

Planning Methods Section
System Planning
Bonneville Power Administration
P.O. Box 3621 — TEOS
Portland, OR 97208

PLANNING METHODS SECTION
SYSTEM PLANNING
BONNEVILLE POWER ADMINISTRATION
P.O. BOX 3621, M/S - TEOS
PORTLAND, OR 97208

NOTICE TO NON-BPA USERS

The Bonneville Power Administration (BPA) releases BPA-developed computer programs under the following conditions:

1. BPA does not charge for program development costs; however, a fee to cover costs incurred in answering inquiries is assessed against the organization receiving the material. This fee typically includes costs for personnel and computer resources, reproduction, shipping, and postage.
2. BPA cannot provide assistance with conversion to other computers or consulting services to the program users.
3. In consideration of receipt and acceptance of these programs — or portions thereof — if sold, assigned, or transferred to another organization, you and your organization agree to advise any third-party recipient in writing that the program(s) and/or documentation are in the public domain and available from BPA. The intent of this agreement is to ensure that BPA-developed or supplied programs, and/or documentation, whether in whole or in part, that are in the public domain, are identified as such to recipients.

“LEGAL NOTICE”

Neither BPA nor any person acting on behalf of BPA:

1. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe upon privately owned rights; or
2. Assumes any liability with respect to the use of, or for damages resulting from the use of any information, apparatus, method or process disclosed in this report.

9/13/95

WORKING COPY ONLY - NOT FOR DISTRIBUTION

INTRODUCTION	1 - 1
OVERVIEW	1 - 1
PROGRAM AND FEATURES	1 - 1
CHAPTER SUMMARIES	1 - 2
AUDIENCE	1 - 2
IPF DOCUMENTATION	1 - 2
A NOTE ON TYPOGRAPHICAL CONVENTIONS	1 - 3
IPF OVERVIEW	2 - 1
IPF: A POWER FLOW ENGINEERING APPLICATION	2 - 1
IPF FAMILY OF PROGRAMS	2 - 2
IPF INTERACTION MODEL	2 - 3
The GUI Approach	2 - 4
The BPF Approach	2 - 4
The CFLOW Approach	2 - 4
The IPFBAT Approach	2 - 4
PCL COMMANDS	3 - 1
INTRODUCTION	3 - 1
GENERAL	3 - 6
FILE OPENING AND SAVING	3 - 8
PROCESSES	3 - 10
GET_DATA	3 - 12
PUT_DATA	3 - 42
REPORT GENERATION	3 - 43
CUSTOMIZING IPF	4 - 1
INTRODUCTION	4 - 1
XGUI RESOURCES	4 - 1
CHANGING IPF RESOURCES	4 - 3
Changing Window Position and Size	4 - 3
Changing Fonts	4 - 4
Changing Colors	4 - 4
Changing Default File Names	4 - 5
THE XGUI FILE	4 - 6
IPF NETWORK DIAGRAMS	5 - 1
OVERVIEW	5 - 1
INPUT REQUIREMENTS, OUTPUT, AND OPERATION	5 - 2
Input Requirements	5 - 2
Output	5 - 2
Plot Program Operation	5 - 2
COORDINATE FILE	5 - 3
File Identification Record — [ID COORD	5 - 3
Options Record — O	5 - 4
PostScript Records	5 - 9
>Define Records	5 - 10
Comment Records	5 - 10
Draw Records	5 - 11
Bus Coordinate Data	5 - 11

Branch Coordinate Data 5 - 13
Area Coordinate Data 5 - 14
Intertie Coordinate Data 5 - 16
Annotation Record 5 - 17
Trailer Record 5 - 18
POSTSCRIPT PROCEDURES 5 - 19
Coordinate Data Within the Postscript Procedures File 5 - 19
Diagram Identification Data 5 - 19
Legend 5 - 20
Line Pattern Data 5 - 21
Bus Overvoltage/Undervoltage Range Values 5 - 21
Bus Symbols 5 - 22
Area Symbols and Bubble Plots 5 - 22
DIAGRAM COMPONENTS 5 - 24
Supportive Diagram Components 5 - 24
Primary Diagram Components: Bus/Branch Diagrams 5 - 24
Primary Diagram Components: Interchange Diagram 5 - 28
Primary Diagram Components: Difference Diagram 5 - 28
BATCH MODE PLOTTING WITH IPFBAT 5 - 32
Example 1 5 - 32
Example 2 5 - 33
Example 3 5 - 33
Example 4 5 - 33
Example 5 5 - 35
MISCELLANEOUS TOPICS 6 - 1
DIRECTORIES 6 - 1
Data and Execution Directories 6 - 1
Setting Paths 6 - 2
Using ipf_setup 6 - 2
GUI EDITORS 6 - 3
Internal Editor 6 - 3
External Editor 6 - 4

Page Options Dialog Box 5 - 6
GUI Diagram Options 5 - 7
Bus Coordinate Data Record 5 - 13
Branch Coordinate Data Record 5 - 14
Area Coordinate Data Record 5 - 15
Intertie Coordinate Data Record 5 - 17
Diagram Example 5 - 25
Bus/Branch Diagram with Inset 5 - 30
Area Interchange Diagram 5 - 31

IPC Commands Quick Descriptions 3 - 2
Area A Record 3 - 24
Intertie I Record 3 - 24
AC Buses (Types B, BE, BS, BC, BD, BV, BQ, BG, BT, BX) 3 - 25
DC Buses (Type BD and BM) 3 - 26
X-bus Output Data 3 - 26
Continuation Bus (+) 3 - 28
All Lines (Types L, LD, LM, E, T, TP, and R) 3 - 28
Area A Record 3 - 32
Intertie I Record 3 - 33
AC Buses (Types B, BE, BS, BC, BD, BV, BQ, BG, BT, BX) 3 - 33
DC Buses (Type BD and BM) 3 - 34
X-bus Output Data 3 - 35
Continuation Bus (+) 3 - 36
All Lines (Types L, LD, LM, E, T, TP, and R) 3 - 37
File Identification Record Format 5 - 4
Option Record Format 5 - 4
Page Options 5 - 5
Powerflow Values Options 5 - 8
File Management Option 5 - 9
PostScript Record Format 5 - 10
Define Record Format 5 - 10
Comment Record Format 5 - 10
Draw Record Format 5 - 11
Bus Coordinate Data Format 5 - 12
Branch Coordinate Data Format 5 - 13
Area Coordinate Data Format 5 - 15
Intertie Coordinate Data Format 5 - 16
Annotation Record Format 5 - 17
Trailer Record Format 5 - 18

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The *IPF Advanced User's Guide* covers topics which may be of interest to experienced IPF users who want to take advantage of all features and capabilities of the program set, and those who may want to customize the appearance and accessibility of the programs.

1.2 PROGRAM AND FEATURES

The Interactive Powerflow (IPF) program package consists of the Powerflow “engine,” IPFSRV, and various other “front end” programs which are joined by interprocess communication (IPC) routines that shuttle data and instruction messages back and forth. The Powerflow program serves primarily as a solution and database engine that sends and receives data when requested by the client program.

The most common client is GUI, which presents the system data in a graphic, push-button interface. GUI communicates with IPFSRV via buffers and commands, which form the PCL syntax. IPFBAT is a non-graphic client which sends and receives the IPC buffers just as GUI does. The user may also employ the PCL syntax, in several different ways:

- by entering the commands in the Command Dialog of GUI,
- by putting the commands in a file and loading it as a Command File in GUI,
- by entering the commands interactively in the IPFBAT program,
- by putting a complete set of commands, including initialization and exit, in a file and sending the file to IPFSRV via IPFBAT.

See Chapter 3 for details on PCL.

Another client is CFLOW, which offers a C language “interface” to IPFSRV. The user writes a program in ANSI C, which includes the header file `cflowlib.h` and calls various CFLOW library routines, then compiles and links this program against the `libcflownew.a` library.

The batch IPF program, BPF, uses the original batch command file language, PFC. The interpreter for this language is built into the `ipfmain` executable, and is accessed with the name “`bpf.`” This name is passed to `ipfmain` as `argv[0]`. Other subsidiary programs, such as IPFCUT and IPFNET, work similarly. These do not require IPC, but do require a properly named link, or other mechanism to pass the name of the program desired to `ipfmain`. This philosophy has been adopted

to avoid having two complete separate copies of the Powerflow program, one for BPF and another for IPFSRV, as well as a plethora of smaller executables for the subsidiary programs.

1.3 CHAPTER SUMMARIES

- | | |
|-------------------|--|
| <i>Chapter 1</i> | Summarizes the IPF program set philosophy, documentation and chapters in the Advanced Guide. |
| <i>Chapter 2</i> | Gives an overview of the complete IPF program set. |
| <i>Chapter 3</i> | Describes the use of IPF Program Control Language (PCL) from the GUI Command Dialog and the IPFBAT program. Lists all PCL commands, with syntax and examples. |
| <i>Chapter 4</i> | Tells how to change the XGUI and pfmaster.post files to customize IPF. |
| <i>Chapter 5</i> | Provides details on coordinate data and advanced usage of the network diagram plotting facility. |
| <i>Chapter 6</i> | Covers a variety of advanced topics and techniques, such as setting up directory structure for IPF programs and data, using the internal editor and setting up an external editor. |
| <i>Appendix A</i> | Contains sample studies comparing the use of the batch and the GUI approach, or a combination. |

1.4 AUDIENCE

This guide is designed for experienced users of IPF, who want to explore the full extent of its features and capabilities, and customize its appearance for their own or other's use.

1.5 IPF DOCUMENTATION

The IPF documentation consists of four manuals:

- The *IPF Basic User's Guide* provides an introduction to the graphical user interface with special emphasis on the X Window System and OSF/Motif. A task-oriented chapter explains how to accomplish common IPF tasks. A reference-oriented chapter describes all the GUI commands available, along with some background conceptual information.
- The *IPF Batch User's Guide* is based on the previous *Powerflow Program User's Manual*. The treatment is mainly conceptual and reference-oriented. These topics are treated in detail: program input and database files, the PFC program control language, and record formats.

- The *IPF Advanced User's Guide* covers use of the PCL language from the GUI Command Dialog and the IPFBAT program, customization of IPF, and also detailed information on hardcopy plotting of maps.
- The *CFLOW User's Guide* serves the needs of advanced IPF users who want to write special purpose programs in C, using the CFLOW library of C functions. The manual consists of a section about how to write CFLOW programs, and a reference-oriented section describing the C library functions in alphabetical order. There are also example programs to get you started.

1.6 A NOTE ON TYPOGRAPHICAL CONVENTIONS

Operating system commands, parameters, file names, etc., and any information that you type or that might appear on your screen all appear in the Courier plain font, for example, `ls -sF`.

CHAPTER 2

IPF OVERVIEW

2.1 IPF: A POWER FLOW ENGINEERING APPLICATION

Electric power system network design encompasses the following tasks:

- Determination of load centers (points) and generation patterns as well as sizes of loads and generation.
- Determination of available transmission corridors (rights-of-way) and assessment of the capacity of these corridors to accommodate transmission lines.
- Evaluation of existing or planned networks with regard to adequate power-carrying capability, voltage regulation, reliability of service, and operating economics.
- Determination of size and routing of new transmission lines, and size and location of terminal equipment for achieving efficient and economical reinforcements when needed.
- Evaluation of proposed reinforcements in light of power flow capability, ability to withstand transient disturbances, reliability of overall service, economics, impact on regional economy, environment, energy conservation and operational constraints such as construction lead times, coordination of various facility ownership interests, flexibility for future growth and compatibility with other long-range plans.

The dynamic nature of load growth, load distribution, and generation patterns make the problem of network design one of planning. To plan for the future, a design must look at the past and present. This makes the Interactive Powerflow program a key tool for the network design engineer. Thanks to a comprehensive, structured database, it permits a complex network structure to be modeled and evaluated at various points in time. IPF also incorporates most modern modeling and analysis concepts. In addition, IPF assists the engineer in documenting major design decisions and changes.

2.2 IPF FAMILY OF PROGRAMS

IPF can be thought of as a family of programs. BPF is the batch form of the powerflow program. When the editing and displaying of buses and branches is being handled by GUI, the work of calculating solution voltages for a given power system network is done by IPFSRV, which is just the batch program in a different guise. Auxiliary programs allow you to do plots in batch mode, do a save of network data in batch mode, perform a “cut” of a solved base case, etc.

Note: There are two different program control languages, the “new style” PCL, and the “old style” PFC. These two sets of commands are *not* completely compatible even though the “new style” command set and syntax is closely modeled on the “old style.” This manual describes PCL only. For PFC, refer to the *IPF Batch User's Guide*.

GUI The X-based (X Window System) push button and menu-driven Graphical User Interface program that works in conjunction with the Powerflow server, IPFSRV. See the *IPF Basic User's Guide* for information on GUI operations.

IPFSRV The Powerflow server to the GUI. It executes Powerflow commands through Powerflow Control Language (PCL) scripts dispatched from the GUI.

IPFBAT The batch version of IPFSRV. It accepts a “new style” Powerflow Control Language (PCL) script file. Plotting can be done with a control file; however, for most plots IPFPLOT is easier to use. Example of use: `ipfbat test.pcl`.

The “new style” PCL commands used with IPFSRV and IPFBAT (pseudo standard of `.pcl`) are described in the *IPF Advanced User's Guide*. Many of the PFC commands are supported, but not all.

BPF The updated version of the old BPA batch Powerflow program. It executes using the commands from an “old style” PowerFlow Control (PFC) script file. Example of use: `bpf test.pfc`.

The PFC commands (pseudo standard of `.pfc`) used with BPF are scripts for a complete Powerflow run. The *IPF Batch User's Guide* describes the commands available.

IPFPLOT Batch plotting program to produce printed maps. The program accepts a coordinate file and a base case file on the command line, as well as an optional second base case file. When the second base case file is specified, a difference plot is produced. You can also use IPFPLOT to produce bubble diagrams. The same coordinate files are used for both GUI and IPFPLOT, but not all capabilities are available in GUI. Documentation is in Chapter 5 of the *IPF Advanced User's Guide*.

- IPFNET** The batch version of the “save netdata file” function built into the GUI/IPFSRV. This program generates a WSCC-formatted network data file in any of the following dialects: BPA, WSCC, or PTI. “Dialects” means that the file is still WSCC, but the data is generated with special processing or restrictions and is destined for use with other programs. In the case of the PTI dialect, that data is preprocessed by the PTI-proprietary conversion program WSCFOR. Documentation is in Appendix F of the *IPF Batch User's Guide*.
- IPFCUT** The stand-alone program that cuts out a subsystem from a solved base case file. Flows at the cut branches are converted into equivalent generation or load on specially formatted +A continuation bus records. An ensuing Powerflow run should solve with internal branch flows and bus voltages which are identical to those quantities in the original base case. Documentation is in Appendix F of the *IPF Batch User's Guide*.
- Several methods are available to define the cut system: bus names, zones, base kVs, and individual branches.
 - A pi-back feature replaces selected buses with a passive-node sequence (lines consisting of sections) with the original loads pi-backed in proportion to the line admittances.
- IPS2IPF** The program that converts a network data file from IPS to IPF. Duplicate buses are renamed; LTC steps are converted to taps, shunt susceptance on slack and BQ buses are transferred to +A records; sectionalized lines containing a section 0 are renumbered 1, 2, . . . ; BX, X, and remote controlled bus data are converted to IPF format, etc. Documentation is in Appendix G of the *IPF Batch User's Guide*, and Appendix D of the *IPF Basic User's Guide*.

2.3 IPF INTERACTION MODEL

The conceptual model of IPF is quite simple. You load power system network data into the database and solution “engine” (PF), which performs the calculations for the solution, and then returns the answers to GUI, CFLOW, or a file.

IPF offers several different approaches to accomplish power system solutions:

- The Graphical User Interface (GUI) approach.
- The traditional batch (BPF) approach.
- The programmed CFLOW approach.
- The hands-on PCL approach.

2.3.1 The GUI Approach

When you use the GUI approach, you use the dialog boxes, menus, windows, etc., of the GUI. This makes data input, output, and manipulation easy. In addition to allowing basic case solution tasks to be accomplished, certain specialized tasks such as line impedance calculations are available. However, for more involved tasks, you need to use the BPF approach. For information about how to work with the GUI dialog boxes, menus, windows, etc., see the *IPF Basic User's Guide*. This guide also has a tutorial to show you how to solve straightforward power system cases.

2.3.2 The BPF Approach

When you use BPF, you must first create a PFC file with the appropriate commands to accomplish the solution task at hand. At runtime these commands are accepted by BPF and executed according to a logical processing order determined by the program. Hence you need not be concerned with the ordering of commands in your PFC file. Input commands will be processed first, and a solution done automatically before any output is produced. Finally, a new base file will be created, if you have requested one. See the *IPF Batch User's Guide* for information on this approach.

2.3.3 The CFLOW Approach

Many system planning studies entail a large number of similar runs. IPS users have encoded these standard operations in the COPE language; to do the same sort of thing with IPF, you will use the CFLOW approach. Unlike COPE, CFLOW is not a complete language which is interpreted by the IPF program itself. Instead, CFLOW consists of a library of C language functions, callable from either C or Fortran. To use CFLOW, you write your program (at least the main must be in C), including the header file cflowlib.h, which defines all the structures and unions which allow access to the powerflow input and solution values. To retrieve these values, you call various CFLOW routines. You can also pass modifications to IPFSRV, ask for a new solution, etc. See the *IPF CFLOW User's Guide* for information on writing these programs.

2.3.4 The IPFBAT Approach

IPFBAT allows you fine control over the database and solution “engine” (IPFSRV). When you use the PCL approach, you first create a PCL file with the appropriate commands to accomplish the solution task at hand. At runtime these commands are interpreted by IPFBAT. The PCL file commands are processed sequentially. Additional PCL command files may be specified by name, so that a “chain” of PCL files may be processed in one run. See the *IPF Advanced User's Guide* for details.

CHAPTER 3

PCL COMMANDS

3.1 INTRODUCTION

The commands listed below are those currently available in the Powerflow Command Language (PCL). This language was developed to meet the needs of communication between the GUI and the Powerflow server (`ipfsrv`), but is also available for direct user entry via the Command Dialog or loading of a PCL command file in the GUI, and via the IPFBAT program, by invoking the same command file. However, since `ipfsrv` is expecting these commands to have been generated by another program, it expects them to be syntactically perfect and legally ordered. There is no checking of input, or confirmation procedure, or chance to edit and re-enter commands. A typo can easily crash the program.

The GUI process sends these commands in response to various button pushes, and all checking, etc. is done before a command is ever sent. The generated text strings are placed in a buffer which is sent over an interprocess communications channel open to the Powerflow process (`ipfsrv`). The Powerflow process on its end interprets these commands and responds by changing its memory resident case data, querying the case data, sending data back to the GUI, etc. CFLOW library routines also send and receive these commands in the same manner.

PCL has some overlap with the PFC command language used in the batch Powerflow (`bpf`) program. For those commands which do the same things (e.g. `OLD_BASE`, `CHANGES`, etc.) the syntax is identical. However, many of the batch commands are not available in PCL, and of course PCL contains many commands which are not needed by BPF. The same conventions on case and spacing apply to both languages.

- Upper, lower, or mixed case are legal: `OLD_BASE`, `Old_Base`, `oldbase`.
- Spaces between command elements are ignored: `/oldbase,file=std.bse` and `/ OLD_base , File = std.bse` are equivalent.
- Underbars may be used within command words if desired: `oldbase`, `old_base`, `O_L_D_B_A_S_E`.

Table 3-1 IPC Commands Quick Descriptions

Command	Description	Page
(END)	Terminates a data stream following a command.	3-6
*[EOM]	Used in the Command Dialog to launch a command stream.	3-6
CFLOW	Launches a CFLOW program.	3-10
CHANGES	Introduces system data change records.	3-10
GET_DATA	Fetches data from the Powerflow process.	3-12
GET_DATA, TYPE = A_DATA	Retrieves all type A input data records in WSCC format.	3-12
GET_DATA, TYPE = AREA_DATA	Initializes the user analysis arrays.	3-13
GET_DATA, TYPE = AREA_LIST	Loads the area list dialog into the Network Data Edit Dialog and the Reports Dialog.	3-13
GET_DATA, TYPE = BSEKV_LIST	Loads the base kV list dialog into the Network Data Edit Dialog and Reports Dialog.	3-14
GET_DATA, TYPE = BUS_EXISTS	Inquires whether a given bus exists.	3-15
GET_DATA, TYPE = BUS_LIST	Loads the bus list dialog in the Alpha Search, Network Data Edit, and Reports Dialogs.	3-15
GET_DATA, TYPE = BUS_VOLTAGES	Returns a list of all bus voltages.	3-16
GET_DATA, TYPE = CONNECTION	Retrieves the network connection diagram for a given set of buses.	3-17
GET_DATA, TYPE = COUNT	Computes the number of network data records satisfying the filter criteria in a NETWORK_DATA command.	3-18
GET_DATA, TYPE = FILE_EXISTS	Inquires whether a named files exists.	3-19
GET_DATA, TYPE = INITIALIZE_DEF	Initializes the user analysis arrays.	3-19
GET_DATA, TYPE = INPUT	Retrieves the full network data for a bus.	3-20
GET_DATA, TYPE = I_DATA	Retrieves all I type data in WSCC format.	3-19
GET_DATA, TYPE = LINE_IMPEDANCE_CALCULATION	Computes transmission line impedance data given tower geometry and conductor characteristics.	3-20
GET_DATA, TYPE = LOAD_AREA	Initializes arrays associated with area interchange data.	3-21

Table 3-1 IPC Commands Quick Descriptions (Continued)

Command	Description	Page
GET_DATA, TYPE = SUB_DEFINE	Loads the user analysis arrays.	3-39
GET_DATA, TYPE = LOAD_REF_AREA	Initializes arrays associated with area interchange data.	3-22
GET_DATA, TYPE = LOAD_REF_BASE	Loads a reference base case history file for the purpose of base case comparison or plot comparisons.	3-22
GET_DATA, TYPE = NETWORK_DATA	Filters network data records.	3-22
GET_DATA, TYPE = OUTAGES	Retrieves the list of outaged data for the Report Dialog.	3-23
GET_DATA, TYPE = OUTPUT	General purpose command for accessing virtually the entire network data base.	3-23
GET_DATA, TYPE = OWNER_LIST	Loads the ownership list dialog into the Reports Dialog and the Network Data Edit Dialog.	3-29
GET_DATA, TYPE = RECORD_LIST	Loads the record type list dialog into the Reports Dialog and the Network Data Edit Dialog.	3-30
GET_DATA, TYPE = COMMENTS	Obtains the area interchange output data from the WSCC-formatted input area records.	3-17
GET_DATA, TYPE = REF_OUTPUT	General purpose command for accessing virtually the entire network data base.	3-32
GET_DATA, TYPE = STATUS	Retrieves the case description.	3-39
GET_DATA, TYPE = SUB_DEFINE	Performs character string substitution using computed base case quantities.	3-39
GET_DATA, TYPE = COMMENTS	Retrieves all IPF system parameters describing the case in residence.	3-17
GET_DATA, TYPE = ZONE_LIST	Loads the zone list dialog into the Reports Dialog and the Network Data Edit Dialog.	3-40
INITIALIZE	Starts up the Powerflow process.	3-6
NETWORK_DATA	Specifies that a network data file is to be loaded into the Powerflow process.	3-8
NEW_BASE	Saves the solved, resident base case.	3-8
OLD_BASE	Loads a previously solved Powerflow case file.	3-8
PLOT	Plots a Powerflow coordinate file.	3-10

Table 3-1 IPC Commands Quick Descriptions (Continued)

Command	Description	Page
PUT_DATA, TYPE = COMMENTS	Defines IPF system parameters which either describe the case in residence or modify parameters which will influence certain processes (solution, debugging).	3-42
QUIT, EXIT	Executes closing procedures and exits the Powerflow process.	3-6
REPORTS, SELECT AI_SUMMARY	Retrieves filtered area interchange output data.	3-44
REPORTS, SELECT BUS_BR_INPUT	Retrieves filtered WSCC-formatted bus and branch input data records.	3-46
REPORTS, SELECT BUS_BR_OUTPUT	Retrieves filtered bus and branch output records.	3-47
REPORTS, SELECT BUS_INPUT	Retrieves filtered WSCC-formatted bus input data records.	3-45
REPORTS, SELECT BUS_UVOV	Retrieves filtered under/over voltage bus output data.	3-48
REPORTS, SELECT LINE_COMPARISON	Retrieves filtered line loading differences between the base case in residence and a selected base case history data file.	3-49
REPORTS, SELECT NETWORK_CHANGES	Retrieves the list of all accumulated changes performed on the base case in residence.	3-50
REPORTS, SELECT NETWORK_DELETIONS	Retrieves the list of all deleted network data in WSCC format.	3-51
REPORTS, SELECT OVERLOADED_LINES	Retrieves filtered overloaded branch output data.	3-52
REPORTS, SELECT OVERLOADED_TXS	Retrieves filtered overloaded transformer output data.	3-53
REPORTS, SELECT PHASE_SHIFTER	Retrieves the phase shifter report.	3-54
REPORTS, SELECT TIE_LINE_SUMMARY	Retrieves filtered area tie line flows.	3-55
REPORTS, SELECT VOLTAGE_COMPARISON	Retrieves filtered voltage differences between the resident base case and a selected base case history data file.	3-56
SAVE_FILE	Saves the solved resident base case in a named file.	3-9
SOLUTION	Causes the Powerflow process to solve the currently resident base case.	3-11

Table 3-1 IPC Commands Quick Descriptions (Continued)

Command	Description	Page
SYSCAL	Passes a system command to the operating system.	3-7

3.2 GENERAL

(END)

Use this command to terminate a data stream following a command in a PCL command file. Some commands which would be followed by data are /CHANGES, FILE=* and /PLOT.

Example:

```
/Network_Data, File = *
B      ARAPAHOA115.
BS     ARAP1    13.8    3.5                45.        1.05
B      SHERIDAN115.  41.5   11.8
T      ARAPAHOA115. ARAP1    13.8        .00672.14684.00064-00313115.5 13.2
L      ARAPAHOA115. SHERIDAN115.        .00261.00657        .0004
(end)
```

*[EOM]

This is used when entering commands in the Command Dialog of the GUI, to indicated that the line or lines you have entered into the buffer should be send to ipfsrv. When the GUI generates this "End of Message" string, it has all six characters; however, the first two characters are all that are required to be entered.

For Command Dialog usage, this also serves in place of the (END) command above. But unlike it, this must be entered after *every* command, when using the Command Dialog.

INITIALIZE

This command starts up the Powerflow process, and therefore will never be entered by a GUI user, but is required at the beginning of every IPFBAT command file. It calls `p_pfnit.f` with the following parameters.

```
integer function p_pfnit (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/INITIALIZE
```

QUIT, EXIT

This command in either form executes closing procedures and exits the Powerflow process. It calls `p_pfexit.f` with the following parameters.

```
integer function p_pfexit (in_buffer, out_buffer)
```



```
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains either of the following lines of information.

```
/QUIT
/EXIT
```

SYSCAL

This command passes a string to the operating system for execution. It is used by the GUI to send a plot to the currently selected print destination, but it will pass any command string.

Example:

```
/SYSCAL
lp -d COMPAQ20 -T ps 0102hw1.ps
(END)
```

3.3 FILE OPENING AND SAVING

NETWORK_DATA

This command specifies that a network data file is to be loaded into the Powerflow process. It calls `p_gtnetdat.f` with the following parameters.

```
integer function p_gtnetdat (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

Example:

```
/NETWORK_DATA, FILE=0102hw1.net
```

For a full description of the command `/NETWORK_DATA`, see the *IPF Batch User's Guide*. Successful execution should return an IPF state of 2.

NEW_BASE

This command saves the solved base case in residence in the named file. It calls `p_newbse.f` with the following parameters.

```
integer function p_newbse (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information

```
/NEW_BASE, FILE = <filename>
C < case comments - three records maximum >
C < case comments - three records maximum >
C < case comments - three records maximum >
```

The comment records in the above command are optional. Successful execution should return an IPF state of 6.

OLD_BASE

This command specifies that a previously solved power flow case is to be loaded from the specified file and used as the base system for the current request. It calls `p_gtbase.f` with the following parameters.

```
integer function p_gtbase (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
```

```
character out_buffer * (MAXBUFFER)
```

Example;

```
/OLD_BASE, FILE = 97hsla.bse
```

Successful execution should return an IPF state of 6.

SAVE_FILE

This command saves the solved base case in residence in the named file in one of four forms. One of these forms is NEW_BASE, described previously, and admits a redundancy into the command procedure.

This command calls `p_svfile.f` with the following parameters.

```
integer function p_svfile (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains any of following information.

```
/SAVE_FILE, TYPE = SYSTEM_CHANGES, FILE = <filename>

/SAVE_FILE, TYPE = NEW_BASE, FILE = <filename>

/SAVE_FILE, TYPE = NETWORK_DATA, FILE = <filename>, ...

/SAVE_FILE, TYPE = WSCC_BINARY_STABILITY, FILE = <filename>
                WSCC_ASCII_STABILITY

/SAVE_FILE, TYPE = NETWORK_DATA,
FILE = < filename >,
DIALECT = < value >,      BPA | WSCC | WSCC1 | PTI
RATINGS = < value >,      EXTENDED | NOMINAL | MINIMUM
SIZE = < value >,         120 | 80
```

See the *IPF Batch User's Guide*, Appendix F, for a complete description of the Dialect, Ratings, and Size options when saving a network data file.

3.4 PROCESSES

CFLOW

This command launches a CFLOW program if it is in your directory search path. The following works on a VAX VMS system if <program name> is defined as a foreign command:

```
/CFLOW
PROGRAM = my_cflow_program

<program name>:= $dev:[dir]file.exe
```

ARGS = is required only if the CFLOW program requires command line arguments other than the socket number. Any I/O that the CFLOW program does to standard input or standard output will be to and from the same terminal window that the ipfbat program is run from (intermixed with any I/O from the ipfbat program).

CHANGES

This command introduces system data change records. It calls p_change.f with the following parameters.

```
integer function p_change (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

Examples:

```
/CHANGES, FILE = 0102hw1.chg

/changes, file = *
B D  ARAPAHOA115.
BEM  ARAP1  13.8                                1.05
T M  ARAPAHO 115. ARAP1  13.8                    .00672.
(end)
```

PLOT

This command creates a PostScript diagram (map) file, using a coordinate file and the currently loaded base case. It calls plot_load.f, which is the main routine for the batch program IPF-PLOT. The batch program takes three parameters, as shown below, and uses the second case, if provided, to produce difference plots.

```
subroutine plot_load (coord_file, base1_file, base2_file)
character * 60 coord-file, base1_file, base2_file
```

Difference plots are not available from the GUI or IPFBAT. The parameters required in the PCL formulation are a coordinate file and the name of a PostScript output file. Records which follow these two file names are interpreted as comments to be placed on the map, following any comments ("C" records) which occur in the coordinate file. However, if a comment begins with an ampersand (&) or an "at" symbol (@), it will not be printed. The & precedes the name of at most one auxiliary coordinate file to be included on the map. The @ signals the presence of a plot option which will override that option in the coordinate file. See Chapter 5, IPF Network Diagrams, for complete information on plotting diagrams, including use of the IPFPLOT program.

Example:

```
/plot
hvmap.cor
98hw3_study.ps
Transfer Study for PAST
Prepared by A. P. Planner
&transfer.cor
@OP Offset=12.7 18.9
(end)
```

SOLUTION

This command causes the Powerflow process to solve the currently resident base case. It calls p_solton.f with the following parameters.

```
integer function p_solton (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array in_buffer contains the following information. Normal defaults are shown; optional items are in [].

```
/SOLUTION
> BASE_SOLUTION ]
> DEBUG, TX=OFF, BUS=OFF, AI=OFF, DCMODEL=OFF [ON]
> LTC = ON [OFF, ON_NV, ON_NPS, ON_DCONLY]
> AI_CONTROL = CONTROL [MON, OFF]
> MISC_CNTRL, -
    X_BUS = BPA, - [WSCC]
    PHASE_SHIFTER_BIAS = BPA, - [WSCC]
    DCLP = ON, - [OFF]
    VFLATSTART = ON, - [OFF]
    ITER_SUM = OFF, - [ON]
    TSTART = 0.5, -
    NUMVSTEPS = 3
> SOL_ITER, DECOUPLED = 2, NEWTON = 30
> LIMITS, QRES=0.01, PHA=45.0, DEL_ANG=1.0, DEL_VOLT=0.15
> TOLERANCE, BUSV = 0.005, AIPOWER = 0.001, TX = 0.001, Q = 0.005
```

3.5 GET_DATA

This command with its many different forms fetches data from the Powerflow process. It calls `p_gtdata.f` with the following parameters.

```
integer function p_gtdata (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains any of the following commands.

```
/GET_DATA, TYPE = A_DATA
/GET_DATA, TYPE = AREA_DATA
/GET_DATA, TYPE = AREA_LIST
/GET_DATA, TYPE = BSEKV_LIST
/GET_DATA, TYPE = BUS_EXISTS
/GET_DATA, TYPE = BUS_LIST
/GET_DATA, TYPE = BUS_VOLTAGES
/GET_DATA, TYPE = COMMENTS
/GET_DATA, TYPE = CONNECTION
/GET_DATA, TYPE = COUNT
/GET_DATA, TYPE = FILE_EXISTS
/GET_DATA, TYPE = I_DATA
/GET_DATA, TYPE = INITIALIZE_DEF
/GET_DATA, TYPE = INPUT
/GET_DATA, TYPE = LINE_IMPEDANCE_CALCULATION
/GET_DATA, TYPE = LOAD_AREA
/GET_DATA, TYPE = LOAD_DEFINE
/GET_DATA, TYPE = LOAD_REF_AREA
/GET_DATA, TYPE = LOAD_REF_BASE
/GET_DATA, TYPE = NETWORK_DATA
/GET_DATA, TYPE = OUTAGES
/GET_DATA, TYPE = OUTPUT
/GET_DATA, TYPE = OWNER_LIST
/GET_DATA, TYPE = RECORD_LIST
/GET_DATA, TYPE = REF_AREA_DATA
/GET_DATA, TYPE = REF_OUTPUT
/GET_DATA, TYPE = SOL_PAR
/GET_DATA, TYPE = STATUS
/GET_DATA, TYPE = SUB_DEFINE
/GET_DATA, TYPE = SYSTEM
/GET_DATA, TYPE = ZONE_LIST
```

The routine `p_gtdata.f` parses these commands and calls a subroutine to perform the specific task, according to the type of data indicated.

GET_DATA, TYPE = A_DATA

This command retrieves in `out_buffer` all type A input data records in WSCC format. It calls

a_data.f with the following parameters.

```
integer function a_data (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array in_buffer contains the following information.

```
/GET_DATA, TYPE = A_DATA
```

Note that no areas are specified in in_buffer; this command just gets a list of the areas in the case.

GET_DATA, TYPE = AREA_DATA

This command obtains the area interchange output data from the WSCC-formatted input area records. This command should be preceded with a prior command GET_AREA, TYPE=LOAD_AREA. It calls p_gtbase.f with the following parameters.

```
integer function p_gtdata (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array in_buffer contains the following information.

```
/GET_DATA, TYPE = AREA_DATA
A  <areaname>
```

GET_DATA, TYPE = AREA_LIST

This command loads the area list filter window in the Network Data Edit Dialog and the Report Dialog of the GUI. It returns in out_buffer the list of area names in the following format.

```
<areaname> LINEFEED
```

where <areaname> is the area name in A10 format.

It calls area_list.f with the following parameters.

```
integer function area_list (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array in_buffer contains optional filter data in the following command.

```
/GET_DATA, TYPE = AREA_LIST [ FROM BUS_DATA ]
                           WHERE AREAS = <areal>, <area2>, etc AND
```

```

ZONES = <zone1>, <zone2>, etc AND
OWNERS = <own1>, <own2>, etc AND
BASEKV = base1
        < base  ( example < 115.0 means all base
                  kv's less than or equal to 115.0)
        > base  ( example > 115.0 means all base
                  kv's greater than or equal to 115.0)
base1 < base2  (all bases between base1 and
                base 2)
                base2 > base1  (same as above)
TYPE = '*' , 'A*', 'A?', 'I' , 'B*', 'L*', 'B?',
        'B' , 'BE', 'BS', 'BC', 'BD', 'BV', 'BQ',
        'BG', 'BT', 'BX', 'BM', 'BF', '+', 'X' ,
        'Q' , 'LD', 'LM', 'E' , 'T' , 'TP', 'R' ,
        'RZ'
ALL

```

Details of the filter are found in the *IPF Basic User's Guide*, under the section on “Dynamic Filters”.

GET_DATA, TYPE = BSEKV_LIST

This command loads the base kV filter window in the Network Data Edit Dialog and the Reports Dialog of the GUI. It returns in `out_buffer` the list of filtered base kVs in the following format.

```
<basekv> LINEFEED
```

where <basekv> is the base kV in F6.1 format.

It calls `bsekvltt.f` with the following parameters.

```

integer function bsekvlst (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)

```

The character array `in_buffer` contains optional filter data in the following command.

```

/GET_DATA, TYPE = BSEKV_LIST [ FROM BUS_DATA ]
WHERE AREAS = <area1>, <area2>, etc AND
ZONES = <zone1>, <zone2>, etc AND
OWNERS = <own1>, <own2>, etc AND
BASEKV = base1
        < base  ( example < 115.0 means all base
                  kv's less than or equal to 115.0)
        > base  ( example > 115.0 means all base
                  kv's greater than or equal to 115.0)
base1 < base2  (all bases between base1 and
                base 2)
                base2 > base1  (same as above)

```



```

TYPE = ' * ', 'A*', 'A?', 'I ', 'B*', 'L*', 'B?',
       'B ', 'BE', 'BS', 'BC', 'BD', 'BV', 'BQ',
       'BG', 'BT', 'BX', 'BM', 'BF', '+ ', 'X ',
       'Q ', 'LD', 'LM', 'E ', 'T ', 'TP', 'R ',
       'RZ'
BUS = "<busname>" (quotes are necessary)
ALL
LOADING = (<min> <max>)

```

Details of the filter are found in the *IPF Basic User's Guide* under the section on "Dynamic Filters."

GET_DATA, TYPE = BUS_EXISTS

This command inquires whether a given bus exists. It calls `ex_bus.f` with the following parameters.

```

integer function ex_bus (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)

```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = BUS_EXISTS, BUS = bus_name base_kv
```

`bus_name` is an eight character name (blank filled to eight characters) followed by a Fortran F4.0 or F6.1 base kV.

The return status is 0 if the bus exists in the current case or 1 if it does not exist.

GET_DATA, TYPE = BUS_LIST

This command loads the bus list dialog in the Alpha Search Dialog, the Network Data Edit Dialog, and the Reports Dialog of the GUI. It returns in `out_buffer` the list of filtered bus names and base kVs in the following format.

```
<busname><base kv> LINEFEED
```

where `<busname>` is the bus name in A8 format ; `<basekv>` is the base kV in F6.1 format.

It calls `bus_list.f` with the following parameters.

```

integer function bus_list (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)

```

The character array `in_buffer` contains optional filter data in the following command.

```
/GET_DATA, TYPE = BUS_LIST [ FROM BUS_DATA ]
      WHERE AREAS = <area1>, <area2>, etc AND
            ZONES = <zone1>, <zone2>, etc AND
            OWNERS = <own1>, <own2>, etc AND
            BASEKV = base1
                  < base ( example < 115.0 means all base
                        kv's less than or equal to 115.0)
                  > base ( example > 115.0 means all base
                        kv's greater than or equal to 115.0)
            base1 < base2 (all bases between base1 and
                        base 2)
            base2 > base1 (same as above)
      TYPE = '*' , 'A?' , 'I' , 'B*' , 'L*' , 'B?' ,
            'B' , 'BE' , 'BS' , 'BC' , 'BD' , 'BV' , 'BQ' ,
            'BG' , 'BT' , 'BX' , 'BM' , 'BF' , '+' , 'X' ,
            'Q' , 'LD' , 'LM' , 'E' , 'T' , 'TP' , 'R' ,
            'RZ'
      BUS = "<busname>" (quotes are necessary)
      AFTER_BUS = "<busname>"
      ALL
      LOADING = (<min> <max>)
```

Details of the filter are found in the *IPF Basic User's Guide* under the section on "Dynamic Filters."

GET_DATA, TYPE = BUS_VOLTAGES

This command returns in `out_buffer` the list of all bus voltages in the following format.

```
<busname><base kv><voltage><angle> LINEFEED
```

```
where <busname> is the bus name in (A8),
      <basekv> is the base kV in (F4.0),
      <voltage> is the actual voltage magnitude in kV in (I4), and
      <angle> is the angle in degrees in (I4).
```

It calls `gtbsvolt.f` with the following parameters.

```
integer function gtbsvolt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = BUS_VOLTAGES
```

GET_DATA, TYPE = COMMENTS

This command obtains case comments, along with case ID, project name, and headers.

There is a related command

```
/PUT_DATA, TYPE = COMMENTS
```

which modifies the corresponding data except for header 1 which is not modifiable. Header 1 is formatted to include case name, case description, program version, date, etc. Up to 20 comments are returned. The three header records are always returned.

The returned values are encoded in the character array `out_buffer` in free field, C-formatted strings. The quantities enclosed in angle brackets "<...>" denote variables returned. Headers may be up to 130 characters; comments may be up to 120 characters, not including the "H" or "C" in column 1.

```
/GET_DATA, TYPE = COMMENTS
CASE_ID = "< case name >"           10 chars
CASE_DS = "< case description >"    20 chars
H< header 1 information >
H< header 2 information >
H< header 3 information >
C< comment text >
...
C< comment text >

return status: status = 0 : success
                  1 : error
```

GET_DATA, TYPE = CONNECTION

This command retrieves network connection information for given buses. Its main usage is to draw the display network diagram. It calls `gtconnect.f` with the following parameters.

```
integer function gtconnect (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = CONNECTION
B      <busname, etc> returns all connection data associated with bus
B      <busname, etc> returns all connection data associated with bus
```

The connection data is returned in `out_buffer` in the following format.

```
B <busname><basekv> LINEFEED
```

```
L <busname><basekv> <busname><basekv> LINEFEED
T <busname><basekv> <busname><basekv> LINEFEED
```

Specifically,

```
"B" records: (1:2) = "B "
              (3:14) = <bus1><base1>.
"L" records: (1:2) = "L "
              (3:14) = <bus1><basekv1>
              (16:27) = <bus2><base2>.
              (28:29) = <number of parallel circuits>
"T" records: (1:2) = "T "
              (3:14) = <bus1><basekv1>
              (16:27) = <bus2><base2>.
              (28:29) = <number of parallel circuits>
```

GET_DATA, TYPE = COUNT

This command computes the number of network data records that would be retrieved using a subsequent /GET_DATA,TYPE=NETWORK_DATA command using the same filter which is defined with this command. The output appears in a dialog field in the Network Data Edit Dialog of the GUI. It returns the count in out_buffer in the following format.

```
count = ccccc
```

cccccc is the count in I6 format. This command calls gtcount.f with the following parameters.

```
integer function gtcount (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array in_buffer contains filter data in the following command.

```
/GET_DATA, TYPE = COUNT [ FROM BUS_DATA ]
      WHERE AREAS = <area1>, <area2>, etc AND
            ZONES = <zone1>, <zone2>, etc AND
            OWNERS = <own1>, <own2>, etc AND
            BASEKV = base1
                  < base ( example < 115.0 means all base
                           kv's less than or equal to 115.0)
                  > base ( example > 115.0 means all base
                           kv's greater than or equal to 115.0)
            base1 < base2 (all bases between base1 and
                           base 2)
                  base2 > base1 (same as above)
      TYPE = '*' ', 'A*', 'A?', 'I ', 'B*', 'L*', 'B?',
            'B ', 'BE', 'BS', 'BC', 'BD', 'BV', 'BQ',
            'BG', 'BT', 'BX', 'BM', 'BF', '+ ', 'X ',
            'Q ', 'LD', 'LM', 'E ', 'T ', 'TP', 'R ',
```

```
'RZ'  
BUS = "<busname>" (quotes are necessary)  
AFTER_BUS = "<busname>"  
ALL  
LOADING = (<min> <max>)
```

GET_DATA, TYPE = FILE_EXISTS

This command inquires whether a named file exists on the platform where `ipvsrv` is running. It calls `ex_file.f` with the following parameters.

```
integer function ex_file (in_buffer, out_buffer)  
parameter (MAXBUFFER = 6600)  
character in_buffer * (MAXBUFFER)  
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = FILE_EXISTS, FILE = <file_name>
```

The return status is 0 if the bus exists, or 1 if it does not exist.

GET_DATA, TYPE = I_DATA

This command retrieves in `out_buffer` all type I records in WSCC format. It calls `i_data.f` with the following parameters.

```
integer function i_data (in_buffer, out_buffer)  
parameter (MAXBUFFER = 6600)  
character in_buffer * (MAXBUFFER)  
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information

```
/GET_DATA, TYPE = I_DATA
```

GET_DATA, TYPE = INITIALIZE_DEF

This command initializes the user analysis arrays. It should be called prior to a `USER_ANALYSIS` command. It calls `p_initdef.f` with the following parameters.

```
integer function p_initdef (in_buffer, out_buffer)  
parameter (MAXBUFFER = 6600)  
character in_buffer * (MAXBUFFER)  
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = INITIALIZE_DEF
```

GET_DATA, TYPE = INPUT

This command retrieves the full network data given the identification of that record in WSCC format. If the record is type B, all data relevant to that bus is retrieved. It calls `gtinput.f` with the following parameters.

```
integer function gtinput (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains WSCC-formatted records following the command `/GET_DATA`.

```
/GET_DATA, TYPE = INPUT
A      <areaname>
I      <areal area2>
B      <busname, etc> returns all data associated with bus
+      <busname, etc> returns all data if id fields have wild cards
              (type - column 2, owner, columns 3-5, and code-year
              columns 20-21)
X      <busname, etc>
L      <bus1 bus2, etc> returns all parallels if id is wild card (*)
              returns all sections if section is 0
T      <bus1 bus2, etc>
R      <bus1 bus2, etc>
E      <bus1 bus2, etc>
```

The character array `in_buffer` is passed through `p_gtdata.f` to `gtinput.f`.

GET_DATA, TYPE = LINE_IMPEDANCE_CALCULATION

This command computes transmission line impedance data given tower geometry and conductor characteristics. It calls a stand-alone module `p_lic.f` and associated routines, which are completely separate from the Powerflow data base. (It was added to `ipfsrv` to keep the GUI free from any FORTRAN modules.)

```
integer function p_lic (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information

```
/GET_DATA, TYPE = LINE_IMPEDANCE_CALCULATION
      UNITS = < ENGLISH | METRIC >,
      DISTANCE = < miles | km >
      BASEKV = <basekv>,
      BASEMVA = <basemva>,
```

```

        FREQUENCY = <freq>
CONDUCTOR = 1 .3636 .05215 1.602 -20.75 50. 50. 0.0 0.0 0
CONDUCTOR = 1 .3636 .05215 1.602 -19.25 50. 50. 0.0 0.0 0
CONDUCTOR = 2 .3636 .05215 1.602 -0.75 77.5 77.5 0.0 0.0 0
CONDUCTOR = 2 .3636 .05215 1.602 0.75 77.5 77.5 0.0 0.0 0
CONDUCTOR = 3 .3636 .05215 1.602 19.25 50. 50. 0.0 0.0 0
CONDUCTOR = 3 .3636 .05215 1.602 20.75 50. 50. 0.0 0.0 0
CONDUCTOR = 0 .5 2.61 0.386 -12.9 98.5 98.5 0.0 0.0 0
CONDUCTOR = 0 .5 2.61 0.386 12.9 98.5 98.5 0.0 0.0 0

```

The returned line impedance data in character array `out_buffer` has the following format.

```
LIC = <r> <x> <g/2> <b/2>
```

`<r>`, `<x>`, `<g/2>`, and `<b/2>` are the corresponding per unit line quantities encoded as F14.8 fields.

GET_DATA, TYPE = LOAD_AREA

This command initializes arrays associated with area interchange data. It should be called prior to any requests for area output. It calls `ldardata.f` with the following parameters.

```

integer function ldardata (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)

```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = LOAD_AREA
```

GET_DATA, TYPE = LOAD_DEFINE

This command loads the user analysis arrays. It calls `p_loaddef.f` with the following parameters.

```

integer function p_loaddef (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)

```

The character array `in_buffer` contains the following information.

```

/GET_DATA, TYPE = LOAD_DEFINE
> DEFINE ...
> DEFINE ...
> DEFINE ...
C ...
C ...
C ...

```

GET_DATA, TYPE = LOAD_REF_AREA

This command initializes arrays associated with area interchange data using the data from the reference base case. Prior to this call, the reference base case should be loaded. It calls `p_ldxardta.f` with the following parameters.

```
integer function p_ldxardta (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = LOAD_REF_AREA
```

GET_DATA, TYPE = LOAD_REF_BASE

This command loads a reference base case history file for the purpose of base case comparison or plot comparisons. The requested base case must be in the IPF format. It calls `p_gtdata.f` with the following parameters.

```
integer function p_gtdata (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information

```
/GET_DATA, TYPE = LOAD_REF_BASE, FILE = <file_name>
```

GET_DATA, TYPE = NETWORK_DATA

This command gets filtered network data records. The output appears in the scrollable edit list on the Network Data Edit Dialog in the GUI. This command calls `gtndat.f` with the following parameters.

```
integer function gtndat (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains filter data in the following command.

```
/GET_DATA, TYPE = NETWORK_DATA [ FROM BUS_DATA ]
      WHERE AREAS = <area1>, <area2>, etc AND
            ZONES = <zone1>, <zone2>, etc AND
            OWNERS = <own1>, <own2>, etc AND
            BASEKV = base1
                  < base ( example < 115.0 means all base
```



```

                                kv's less than or equal to 115.0)
                                > base  ( example > 115.0 means all base
                                kv's greater than or equal to 115.0)
                                base1 < base2  (all bases between base1 and
                                base 2)
                                base2 > base1  (same as above)
TYPE = '*' , 'A*', 'A?', 'I' , 'B*', 'L*', 'B?',
        'B' , 'BE', 'BS', 'BC', 'BD', 'BV', 'BQ',
        'BG', 'BT', 'BX', 'BM', 'BF', '+', 'X' ,
        'Q' , 'LD', 'LM', 'E' , 'T' , 'TP', 'R' ,
        'RZ'
BUS = "<busname>" (quotes are necessary)
AFTER_BUS = "<busname>"
ALL
LOADING = (<min> <max>)

```

Details of the filter are found in the *IPF Basic User's Guide* under the section on "Dynamic Filters."

GET_DATA, TYPE = OUTAGES

This command retrieves the list of outaged data for the Report Dialog (under Bone Pile). It calls `gtoutage.f` with the following parameters.

```

integer function gtoutage (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)

```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = OUTAGES
```

The returned outage data returned in character array `out_buffer` is WSCC-formatted network data that has been deleted.

GET_DATA, TYPE = OUTPUT

This command can access virtually the entire network data base. It calls `gtoutput.f` with the following parameters.

```

integer function gtoutput (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)

```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = OUTPUT
```

```

A    <areaname>      returns all data associated with area
I    <areal area2>   returns all intertie data between the areas
B    <busname, etc>  returns all data associated with bus
+    <busname, etc>  returns all data if id fields have wild cards
                        (type - column 2, owner, columns 3-5, and code-year
                        columns 20-21)
X    <busname, etc>
L    <bus1 bus2, etc> returns all parallels if id is wild card (*)
                        returns all sections if section is wild card (0)
T    <bus1 bus2, etc>
R    <bus1 bus2, etc>
E    <bus1 bus2, etc>

```

The returned values in `out_buffer` correspond with the input record in `in_buffer`.

Table 3-2 Area A Record

Column	Format	Description
1	A1	A — Area Identifier
2	1X	Blank
3-12	A10	Area name
13	1X	Blank
14-28	E15.7	Total Area Generation (MW)
29-43	E15.7	Total Area Load (MW)
44-58	E15.7	Total Area Losses (MW)
59-73	E15.7	Net Area Export (MW)

Table 3-3 Intertie I Record

Column	Format	Description
1	A1	I — Intertie Area Identifier
2	1X	Blank
3-12	A10	Area 1 name
13	1X	Blank
14-23	A10	Area 2 name
24	1X	Blank
25-39	E15.7	Scheduled Area1-Area2 Export (MW)

Table 3-3 Intertie I Record (Continued)

Column	Format	Description
40-54	E15.7	Actual Area1-Area2 Export (MW)
55-69	E15.7	"Circulating" Flow (MW)
70	1X	Blank
71	I1	0 — No Area1-Area2 I record exists 1 — Area1-Area2 I record exists

Table 3-4 AC Buses (Types B, BE, BS, BC, BD, BV, BQ, BG, BT, BX)

Columns	Format	Description
1	A1	Bus code "B"
2	A1	Bus type
3	1X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus name
15-18	F4.0	Bus base KV
19-20	A2	Zone
21-35	E15.7	P_gen (MW)
36-50	E15.7	Q_gen (MVAR)
51-65	E15.7	Voltage (KV))
66-80	E15.7	Angle (degrees)
81-95	E15.7	P_load (MW)
96-110	E15.7	Q_load (MVAR)
111-125	E15.7	B_shunt used (MVAR)
126-140	E15.7	B_shunt scheduled (MVAR)
141-155	E15.7	B_shunt (capacitors) used (MVAR)
156-170	E15.7	B_shunt (capacitors) scheduled (MVAR)
171-185	E15.7	B_shunt (reactors) used (MVAR)

Table 3-4 AC Buses (Types B, BE, BS, BC, BD, BV, BQ, BG, BT, BX)

Columns	Format	Description
186-200	E15.7	B_shunt (reactors) scheduled (MVAR)
201-215	E15.7	Q unscheduled (MVAR)

Table 3-5 DC Buses (Type BD and BM)

Columns	Format	Description
1	A1	Bus code "B"
2	A1	Bus type ("D" or "M")
3	1X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus name
15-18	F4.0	Bus base KV
19-20	A2	Zone
21-35	E15.7	P_d-c (MW)
36-50	E15.7	Q_d-c (MVAR)
51-65	E15.7	D_C Voltage (KV))
66-80	E15.7	Converter angle (degrees)
81-95	E15.7	P_valve losses (MW)
96-110	E15.7	Q_valve losses (MVAR)

Table 3-6 X-bus Output Data

Columns	Format	Description
1	A1	Record code ("X")
2-6	5X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus name
15-18	F4.0	Bus base KV

Table 3-6 X-bus Output Data (Continued)

Columns	Format	Description
19-20	2X	(not used)
21	I1	Group No. 1 scheduled units
22	I1	Group No. 1 used units
23-37	E15.7	Group No. 1 reactance (MVAR) / unit
38	I1	Group No. 2 scheduled units
39	I1	Group No. 2 used units
40-54	E15.7	Group No. 2 reactance (MVAR) / unit
55	I1	Group No. 3 scheduled units
56	I1	Group No. 3 used units
57-71	E15.7	Group No.3 reactance (MVAR) / unit
72	I1	Group No. 4 scheduled units
73	I1	Group No. 4 used units
74-88	E15.7	Group No. 4 reactance (MVAR) / unit
89	I1	Group No. 5 scheduled units
90	I1	Group No.5 used units
91-105	E15.7	Group No. 5 reactance (MVAR) / unit
106	I1	Group No. 6 scheduled units
107	I1	Group No. 6 used units
108-122	E15.7	Group No. 6 reactance (MVAR) / unit
123	I1	Group No. 7 scheduled units
124	I1	Group No. 7 scheduled units
125-139	E15.7	Group No. 7 reactance (MVAR) / unit
140	I1	Group No. 8 scheduled units
141	I1	Group No. 8 used units
142-156	E15.7	Group No. 8 reactance (MVAR) / unit

Table 3-7 Continuation Bus (+)

Column	Format	Description
1	A1	Continuation bus code “+”
2	A1	Continuation bus subtype (A,C,F,I,N,P,S)
3	1X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus name
15-18	F4.0	Bus base KV
19-20	A2	Classification code year, *I — constant current loads, *Z — constant impedance loads, *P — constant MVA loads,
21-35	E15.7	P_gen (MW)
36-50	E15.7	Q_gen (MVAR)
51-65	E15.7	P_load (MW)
66-80	E15.7	Q_load(MVAR)
81-95	E15.7	G_shunt (MW)
96-110	E15.7	B_shunt (MVAR)

Table 3-8 All Lines (Types L, LD, LM, E, T, TP, and R)

Columns	Format	Description
1	A1	Line code L, E, or T
2	A1	Line subtype (LD, LM, or TP)
3	1X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus1 name
15-18	F4.0	Bus1 base KV
19	I1	Interchange metering point (0, 1, or 2)
20-27	A8	Bus 2 name

Table 3-8 All Lines (Types L, LD, LM, E, T, TP, and R)

Columns	Format	Description
28-31	F4.0	Bus 2 base KV
32	A1	Parallel ID (* (asterisk) means all parallels)
33	I1	Number of circuits
34-48	E15.7	P_in (MW)
49-63	E15.7	Q_in (MVAR)
64-78	E15.7	P_out (MW)
79-93	E15.7	Q_out (MVAR)
94-108	E15.7	P_loss (MW)
109-123	E15.7	Q_loss (MW)
124-138	E15.7	Critical line loading (amps)
139-146	F8.1	Critical line rating (amps)
147	A1	Critical line rating code (N,T,B)
148	I1	Critical line loading terminal (0,1,2)
149-163	E15.7	Critical transformer loading (MVA)
164-171	F8.1	Critical transformer rating (MVA)
172	A1	Critical transformer rating code (N,T,E,B)
173	I1	Critical transformer loading terminal (0,1,2)
174-188	E15.7	Total Line loading (percent)
189-203	E15.7	Total Line loading (amps)
204-218	E15.7	Total Transformer loading (percent)
219-233	E15.7	Total Transformer loading (MVA)
234-241	F8.2	Tap1 in kV (Type T or TP) or %Compensation (L or E.)
242-249	F8.2	Tap2 in kV
250-256	A7	(Reserved for difference plotting)

GET_DATA, TYPE = OWNER_LIST

This command loads the ownership list dialog in the Reports Dialog and in the Network Data Edit

Dialog of the GUI. It returns in `out_buffer` the list of filtered ownership names in the following format.

```
<ownership>LINEFEED
```

`<ownership>` is the ownership name in A3 format. It calls `owner_list.f` with the following parameters.

```
integer function owner_list (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains optional filter data in the following command.

```
/GET_DATA, TYPE = OWNER_LIST [ FROM BUS_DATA ]
      WHERE AREAS = <area1>, <area2>, etc AND
            ZONES = <zone1>, <zone2>, etc AND
            OWNERS = <own1>, <own2>, etc AND
            BASEKV = base1
                      < base ( example < 115.0 means all base
                           kv's less than or equal to 115.0)
                      > base ( example > 115.0 means all base
                           kv's greater than or equal to 115.0)
            base1 < base2 (all bases between base1 and
                           base 2)
            base2 > base1 (same as above)
      TYPE = '*' ', 'A*', 'A?', 'I ', 'B*', 'L*', 'B?',
            'B ', 'BE', 'BS', 'BC', 'BD', 'BV', 'BQ',
            'BG', 'BT', 'BX', 'BM', 'BF', '+ ', 'X ',
            'Q ', 'LD', 'LM', 'E ', 'T ', 'TP', 'R ',
            'RZ'
      BUS = "<busname>" (quotes are necessary)
      AFTER_BUS = "<busname>"
      ALL
      LOADING = (<min> <max>)
```

Details of the filter are found in the *IPF Basic User's Guide* under the section on "Dynamic Filters."

GET_DATA, TYPE = RECORD_LIST

This command loads the record type list dialog in the Reports Dialog and in the Network Data Edit Dialog of the GUI. It returns in `out_buffer` the hard-coded list of various record type codes in the following format.

```
<record_type>LINEFEED
```

`<record_type>` is the record type code in A2 format. This command calls `type_list.f` with the

following parameters.

```
integer function type_list (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains optional filter data in the following command.

```
/GET_DATA, TYPE = RECORD_LIST [ FROM BUS_DATA ]
      WHERE AREAS = <area1>, <area2>, etc AND
      ZONES = <zone1>, <zone2>, etc AND
      OWNERS = <own1>, <own2>, etc AND
      BASEKV = base1
              < base ( example < 115.0 means all base
                    kv's less than or equal to 115.0)
              > base ( example > 115.0 means all base
                    kv's greater than or equal to 115.0)
      base1 < base2 (all bases between base1 and
                    base 2)
      base2 > base1 (same as above)
TYPE = '*' , 'A*', 'A?', 'I' , 'B*', 'L*', 'B?',
      'B' , 'BE', 'BS', 'BC', 'BD', 'BV', 'BQ',
      'BG', 'BT', 'BX', 'BM', 'BF', '+', 'X' ,
      'Q' , 'LD', 'LM', 'E' , 'T' , 'TP', 'R' ,
      'RZ'
BUS = "<busname>" (quotes are necessary)
AFTER_BUS = "<busname>"
ALL
LOADING = (<min> <max>)
```

In this instance only, the filter has no impact upon the contents of the returned data. Details of the filter are found in the *IPF Basic User's Guide* under the section on "Dynamic Filters."

GET_DATA, TYPE = REF_AREA_DATA

This command obtains the area interchange output data from the WSCC-formatted input area records using the reference base case data. This command should be preceded with a prior command `GET_AREA, TYPE=LOAD_REF_AREA`. It calls `p_gtxardta.f` with the following parameters.

```
integer function p_gtxardta (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = AREA_REF_DATA
A  <areaname>
```

GET_DATA, TYPE = REF_OUTPUT

This command can access virtually the entire network data from the reference base. It calls `gtal-topt.f` with the following parameters.

```
integer function gtaltopt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = OUTPUT
A      <areaname>      returns all data associated with area
I      <area1 area2>   returns all interties between the two areas
B      <busname, etc>  returns all data associated with bus
+      <busname, etc>  returns all data if id fields have wild cards
                        (type - column 2, owner, columns 3-5, and code-year
                        columns 20-21)
X      <busname, etc>
L      <bus1 bus2, etc> returns all parallels if id is wild card (*)
                        returns all sections if section is wild card (0)
T      <bus1 bus2, etc>
R      <bus1 bus2, etc>
E      <bus1 bus2, etc>
```

The returned values in `out_buffer` correspond with the input record in `in_buffer`.

Table 3-9 Area A Record

Column	Format	Description
1	A1	A — Area Identifier
2	1X	Blank
3-12	A10	Area name
13	1X	Blank
14-28	E15.7	Total Area Generation (MW)
29-43	E15.7	Total Area Load (MW)
44-58	E15.7	Total Area Losses (MW)
59-73	E15.7	Net Area Export (MW)

Table 3-10 Intertie I Record

Column	Format	Description
1	A1	I — Intertie Area Identifier
2	1X	Blank
3-12	A10	Area 1 name
13	1X	Blank
14-23	A10	Area 2 name
24	1X	Blank
25-39	E15.7	Scheduled Area1-Area2 Export (MW)
40-54	E15.7	Actual Area1-Area2 Export (MW)
55-69	E15.7	“Circulating” Flow (MW)
70	1X	Blank
71	I1	0 — No Area1-Area2 I record exists 1 — Area1-Area2 I record exists

Table 3-11 AC Buses (Types B, BE, BS, BC, BD, BV, BQ, BG, BT, BX)

Columns	Format	Description
1	A1	Bus code “B”
2	A1	Bus type
3	1X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus name
15-18	F4.0	Bus base KV
19-20	A2	Zone
21-35	E15.7	P_gen (MW)
36-50	E15.7	Q_gen (MVAR)
51-65	E15.7	Voltage (KV))
66-80	E15.7	Angle (degrees)

Table 3-11 AC Buses (Types B, BE, BS, BC, BD, BV, BQ, BG, BT, BX)

Columns	Format	Description
81-95	E15.7	P_load (MW)
96-110	E15.7	Q_load (MVAR)
111-125	E15.7	B_shunt used (MVAR)
126-140	E15.7	B_shunt scheduled (MVAR)
141-155	E15.7	B_shunt (capacitors) used (MVAR)
156-170	E15.7	B_shunt (capacitors) scheduled (MVAR)
171-185	E15.7	B_shunt (reactors) used (MVAR)
186-200	E15.7	B_shunt (reactors) scheduled (MVAR)
201-215	E15.7	Q unscheduled (MVAR)

Table 3-12 DC Buses (Type BD and BM)

Columns	Format	Description
1	A1	Bus code "B"
2	A1	Bus type ("D" or "M")
3	1X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus name
15-18	F4.0	Bus base KV
19-20	A2	Zone
21-35	E15.7	P_d-c (MW)
36-50	E15.7	Q_d-c (MVAR)
51-65	E15.7	D_C Voltage (KV))
66-80	E15.7	Converter angle (degrees)
81-95	E15.7	P_valve losses (MW)
96-110	E15.7	Q_valve losses (MVAR)

Table 3-13 X-bus Output Data

Columns	Format	Description
1	A1	Record code (X)
2-6	5X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus name
15-18	F4.0	Bus base kV
19-20	2X	(not used)
21	I1	Group No. 1 scheduled units
22	I1	Group No. 1 used units
23-37	E15.7	Group No. 1 reactance (MVAR) / unit
38	I1	Group No. 2 scheduled units
39	I1	Group No. 2 used units
40-54	E15.7	Group No. 2 reactance (MVAR) / unit
55	I1	Group No. 3 scheduled units
56	I1	Group No. 3 used units
57-71	E15.7	Group No.3 reactance (MVAR) / unit
72	I1	Group No. 4 scheduled units
73	I1	Group No. 4 used units
74-88	E15.7	Group No. 4 reactance (MVAR) / unit
89	I1	Group No. 5 scheduled units
90	I1	Group No.5 used units
91-105	E15.7	Group No. 5 reactance (MVAR) / unit
106	I1	Group No. 6 scheduled units
107	I1	Group No. 6 used units
108-122	E15.7	Group No. 6 reactance (MVAR) / unit
123	I1	Group No. 7 scheduled units
124	I1	Group No. 7 scheduled units
125-139	E15.7	Group No. 7 reactance (MVAR) / unit

Table 3-13 X-bus Output Data (Continued)

Columns	Format	Description
91	I1	Group No. 8 scheduled units
92	I1	Group No. 8 used units
93-100	E15.7	Group No. 8 reactance (MVAR) / unit

Table 3-14 Continuation Bus (+)

Column	Format	Description
1	A1	Continuation bus code (+)
2	A1	Continuation bus subtype (A,C,F,I,N,P,S)
3	1X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus name
15-18	F4.0	Bus base KV
19-20	A2	Classification code year, *I — constant current loads, *Z — constant impedance loads, *P — constant MVA loads,
21-35	E15.7	P_gen (MW)
36-50	E15.7	Q_gen (MVAR)
51-65	E15.7	P_load (MW)
66-80	E15.7	Q_load(MVAR)
81-95	E15.7	G_shunt (MW)
96-110	E15.7	B_shunt (MVAR)

Table 3-15 All Lines (Types L, LD, LM, E, T, TP, and R)

Columns	Format	Description
Columns	Format	Description
1	A1	Line code L, E, or T
2	A1	Line subtype (LD, LM, or TP)
3	1X	(Not used)
4-6	A3	Ownership
7-14	A8	Bus1 name
15-18	F4.0	Bus1 base KV
19	I1	Interchange metering point (0, 1, or 2)
20-27	A8	Bus 2 name
28-31	F4.0	Bus 2 base KV
32	A1	Parallel ID (* (asterisk) means all parallels)
33	I1	Number of circuits
34-48	E15.7	P_in (MW)
49-63	E15.7	Q_in (MVAR)
64-78	E15.7	P_out (MW)
79-93	E15.7	Q_out (MVAR)
94-108	E15.7	P_loss (MW)
109-123	E15.7	Q_loss (MW)
124-138	E15.7	Critical line loading (amps)
139-146	F8.1	Critical line rating (amps)
147	A1	Critical line rating code (N,T,B)
148	I1	Critical line loading terminal (0,1,2)
149-163	E15.7	Critical transformer loading (MVA)
164-171	F8.1	Critical transformer rating (MVA)
172	A1	Critical transformer rating code (N,T,E,B)
173	I1	Critical transformer loading terminal (0,1,2)

Table 3-15 All Lines (Types L, LD, LM, E, T, TP, and R) (Continued)

Columns	Format	Description
Columns	Format	Description
174-188	E15.7	Total Line loading (percent)
189-203	E15.7	Total Line loading (amps)
204-218	E15.7	Total Transformer loading (percent)
219-233	E15.7	Total Transformer loading (MVA)
234-241	F8.2	Tap1 in kV (Type T or TP) or %Compensation (L or E.)
242-249	F8.2	Tap2 in kV
250-256	A7	(Reserved for difference plotting)

GET_DATA, TYPE = SOL_PAR

This command obtains solution tolerances, controls, or switches that influence the processing of the case in residence. The obtained system data is identical to the set of data modified by the related command /SOLUTION.

The returned values are encoded in the character array `in_buffer` in free field, C-formatted strings. The quantities enclosed in angle brackets “< ... >” denote variables quantified; < status > denotes a logical on or off; < value > denotes an integer, floating point, or character quantity.

```

/GET_DATA, TYPE = SOL_PAR,
> AI_CONTROL = < value >          { CON | MOD | OFF }
> BASE_SOLUTION = < status >
> DEBUG_TX = < status >
> DEBUG_BUS = < status >
> DEBUG_AI = < status >
> DEBUG_DC = < status >
> LIMITS_QRES = < value >
> LIMITS_PHA = < value >
> LIMITS_DA = < value >
> LIMITS_DV = < value >
> LTC = < value>                  { ON | ON_NV | ON_NPS | OFF | ON_DCONLY }
> MISC_XBUS = < value >           { BPA | VMAX | WSCC }
> MISC_DCLP = < status >
> MISC_VFLAT = < status >
> MISC_TSTART = < value >
> MISC_ITER_SUM = < status >
> MISC_PHA_SHIFT_BIAS = < value > { BPA | WSCC }
> SOL_ITER_DECOUP = < value >
> SOL_ITER_NEWTON = < value >
> TOL_BUSV = < value >

```



```
> TOL_AIPOWER = < value >
> TOL_TX = < value >
> TOL_Q = < value >

return status: status = 0 : success
                  1 : errors
```

GET_DATA, TYPE = SUB_DEFINE

This command performs character string substitution using computed base case quantities upon the tokens defined with the >DEFINE statement within comment records in the USER_ANALYSIS command. It calls `p_subdef.f` with the following parameters.

```
integer function p_subdef (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = SUB_DEFINE, SOURCE = BASE
                                     ALTERNATE_BASE
```

GET_DATA, TYPE = STATUS

This command retrieves the case description. It calls `gtstatus.f` with the following parameters.

```
integer function gtstatus out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains the following information.

```
/GET_DATA, TYPE = STATUS
```

The character array `out_buffer` contains the following information.

```
c
c Program: <n> version <n> date <n>
c Program size: max buses <n> max branches
c Case: <name> status <n> base file <n>
c <n> buses <n> branches <n> areas <n> d-c lines
c <n> changes
c c comments
c c comments
c c comments
c
```

GET_DATA, TYPE = SYSTEM

This command obtains system-specific information pertaining to parameters which describe general characteristics of the base case in residence.

The returned values are encoded in the character array `out_buffer` in free field, C-formatted strings. The quantities enclosed in angle brackets “< ... >” denote variables returned; < status > denotes a logical on or off and < value > denotes an integer or floating point quantity.

```

/GET_DATA, TYPE = SYSTEM,
CASE_DT = < case date >
OLD_BASE = < file name >
NEW_BASE = < file name >
OLD_NETD = < file name >
NEW_NETD = < file name >
OLD_CHGF = < file name >
NEW_CHGF = < file name >
PRG_VERS = < program version >
BASE_MVA = < base MVA >
NUM_DC_SYS = < number of DC systems >
NUM_AREA = < number of areas>
NUM_ITIE = < number of interties>
NUM_ZONE = < number of zones>
NUM_OWN = < number of owners>
NUM_BUS = < number of buses >
NUM_AREA_SBUS = < number of area slack buses>
NUM_DC_BUS = < number of DC buses >
NUM_AGC_BUS = < number of AGC buses>
NUM_BX_BUS = < number of BX buses>
NUM_ADJ_BUS = < number of adjustable buses >
NUM_PCT_VAR_BUS = < number of % VAR controlled buses >
NUM_BRN = < number of branch records>
NUM_CKT = < number of circuits >
NUM_DC_LINE = < number of DC lines >
NUM_LTC = < number of LTC transformers >
NUM_PHAS = < number of phase shifters >
SOLN_STATUS = < solution status >
NUM_KV = < number of different KVs >
NUM_REC_TYP = < number of record types >

return status: status = 0 : success
                  1 : errors

```

GET_DATA, TYPE = ZONE_LIST

This command loads the zone list dialog in the Reports Dialog and the Network Data Edit Dialog. See the *IPF Basic User's Guide*. It returns in `out_buffer` the list of filtered zones in the following format.

```
<zone>LINEFEED
```

<zone> is the zone name in A2 format. It calls `zone_list.f` with the following parameters.

```
integer function zone_list (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains optional filter data in the following command.

```
/GET_DATA, TYPE = ZONE_LIST [ FROM BUS_DATA ]
      WHERE AREAS = <area1>, <area2>, etc AND
      ZONES = <zona1>, <zona2>, etc AND
      OWNERS = <own1>, <own2>, etc AND
      BASEKV = base1
              < base ( example < 115.0 means all base
                    kv's less than or equal to 115.0)
              > base ( example > 115.0 means all base
                    kv's greater than or equal to 115.0)
      base1 < base2 (all bases between base1 and
                    base 2)
      base2 > base1 (same as above)
      TYPE = '*' , 'A*', 'A?', 'I' , 'B*', 'L*', 'B?',
              'B' , 'BE', 'BS', 'BC', 'BD', 'BV', 'BQ',
              'BG', 'BT', 'BX', 'BM', 'BF', '+', 'X' ,
              'Q' , 'LD', 'LM', 'E' , 'T' , 'TP', 'R' ,
              'RZ'
      BUS = "<busname>" (quotes are necessary)
      AFTER_BUS = "<busname>"
      ALL
      LOADING = (<min> <max>)
```

Details of the filter are found in the *IPF Basic User's Guide* under the section on "Dynamic Filters."

3.6 PUT_DATA

PUT_DATA, TYPE = COMMENTS

This command replaces case comments, along with caseid, case description, and headers.

There is a related command

```
/GET_DATA, TYPE = COMMENTS
```

which obtains the corresponding data including header 1 which is not modifiable. Header 1 is formatted to include case name, case description, program version, date, etc. Up to 20 comments are allowed. The two header records must be present even if blank. For all blank “H” or “C” records (blank “C” records are accepted, but optional) include at least one blank character after the “H” or “C”.

The sent values are encoded in the character array `out_buffer` in free field, C-formatted strings. The quantities enclosed in angle brackets “<...>” denote variables. Headers may be up to 130 characters; comments may be up to 120 characters, not including the “H” or “C” in column 1.

```
/PUT_DATA, TYPE = COMMENTS
CASE_ID = "< case name >"           10 chars
CASE_DS = "< case description >"      20 chars
H< header 2 information >
H< header 3 information >
C< comment text >
...
C< comment text >
```

3.7 REPORT GENERATION

The /REPORTS command with its many different forms fetches data from the Powerflow process for display in the Reports Dialog. It calls `p_report.f` with the following parameters.

```
integer function p_report (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

The character array `in_buffer` contains any of the following commands.

```
/REPORTS, SELECT AI_SUMMARY
/REPORTS, SELECT BUS_INPUT
/REPORTS, SELECT BUS_BR_INPUT
/REPORTS, SELECT BUS_BR_OUTPUT
/REPORTS, SELECT BUS_UVOV
/REPORTS, SELECT LINE_COMPARISON
/REPORTS, SELECT NETWORK_CHANGES
/REPORTS, SELECT NETWORK_DELETIONS
/REPORTS, SELECT OVERLOADED_LINES
/REPORTS, SELECT OVERLOADED_TXS
/REPORTS, SELECT PHASE_SHIFTER
/REPORTS, SELECT TIE_LINE_SUMMARY
/REPORTS, SELECT VOLTAGE_COMPARISON
```

The routine `p_report.f` parses these command and calls a subroutine to perform the specific task. The modules are listed below.

REPORTS, SELECT AI_SUMMARY

This command retrieves filtered area interchange output data. It calls `areaintrpt.f` with the following parameters.

```
integer function areaintrpt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `AI_SUMMARY`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT AI_SUMMARY
      [ OUTPUT = <filename> ]
```

The output is placed in `out_buffer`. The report dialog is shown below.

AREA INTERCHANGE SUMMARY							
Area interchange summary case				date 6/11/93			
Area name	Slack bus		Slack bus generation				net
	Name	Base	Beginning	Change	Final	maximum	interchange
			MW	MW	MW	mW	mW
AREA 1	G31	13.8	1277.0	0.0	1277.0	1900.0	1105.0
AREA 2	G4	22.0	1641.0	0.0	1641.0	1200.0	-2210.0
AREA 3	G32	13.8	1277.0	0.0	1277.0	1900.0	1105.0

Close

REPORTS, SELECT BUS_INPUT

This command retrieves filtered WSCC-formatted bus input data records. It calls `businrpt.f` with the following parameters.

```
integer function businrpt(in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `BUS_INPUT`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT BUS_INPUT [ FROM BUS_DATA ]
[ OUTPUT = <filename> ]
WHERE (repeat filter from BUS_LIST)
```

The output is placed in `out_buffer`. The report dialog is shown below.

BUS INPUT DATA																				
Btc<O><	Bus	><KV>ZN<Pld><Qld>Psh>Qsh><Pm>Pgen>qmax>qmin><vh><vl><rembus><kv>%q>																		
BQ	G1	13.802										1200	1000	1000	-1000	1010				
BX	G1X	13.802									300						1010	1010	G1X	13.8
BS	G2	13.802										1200	995.6				1000			
B	G2 HIGH	2300	2540.0	100.0																
BS	G31	13.801										1900	1277	1000	-1000	940				
B	G31HIGH	5000	1100.0																	
BQ	G32	13.803										1900	1277	1000	-1000	940				
B	G32HIGH	5000	3100.0																	
BG	G4	22.002										1200	1641	500.0	-500			G4 HIGH	230	
B	G4 500	50002																		
Close																				

Close

REPORTS, SELECT BUS BR INPUT

This command retrieves filtered WSCC-formatted bus and branch input data records. It calls `bus-brinrpt.f` with the following parameters.

```
integer function busbrinrpt(in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

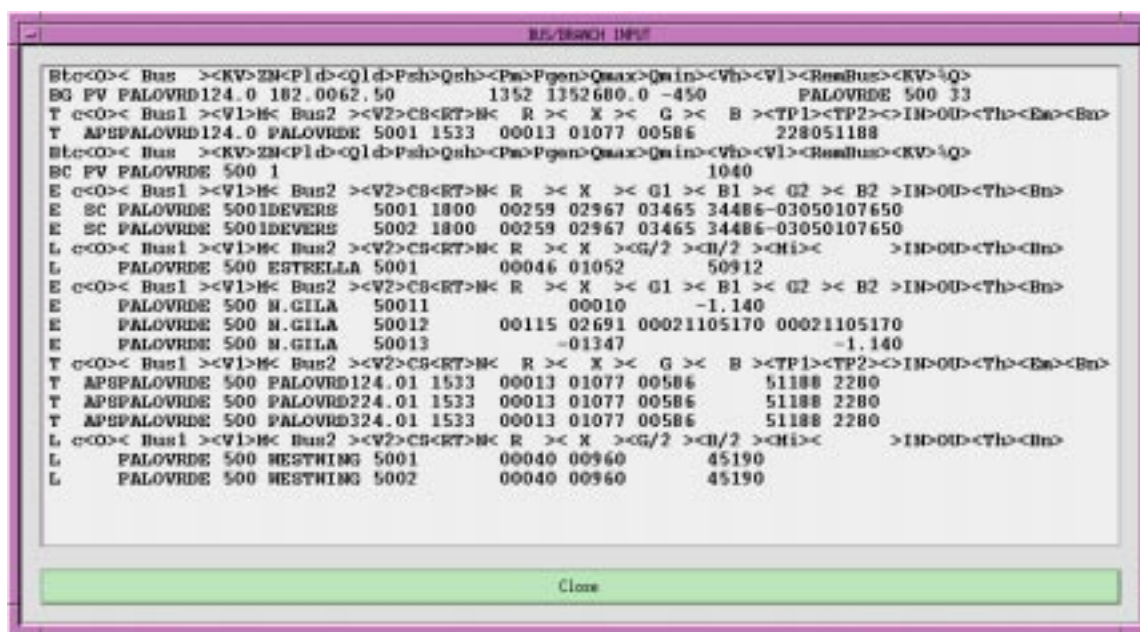
`in_buffer` is positioned to the first character following the string `BUS_BR_INPUT`. The character array `in_buffer` contains the following information.

```

/REPORTS, SELECT BUS_BR_INPUT [ FROM BUS_DATA ]
                [ OUTPUT = <filename> ]
                WHERE (repeat filter from BUS_LIST)

```

The output is placed in `out_buffer`. The report dialog is shown below.



REPORTS, SELECT BUS_BR_OUTPUT

This command retrieves filtered bus and branch output records. It calls `busbrotrpt.f` with the following parameters.

```
integer function busbrotrpt(in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `BUS_BR_OUTPUT`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT BUS_BR_OUTPUT [ FROM BUS_DATA ]
      [ OUTPUT = <filename> ]
      WHERE (repeat filter from BUS_LIST)
```

The output is placed in `out_buffer`. The report dialog is shown below.

BUS/BRANCH OUTPUT										
BUS NAME	VOLTS-PU	...	GENERATION..	...	LOAD.....	...	shunt.....	bus		
	ACTUAL KV/ANGLE		MW	MVAR	MW	MVAR	mw	avail	used	type
G2	13.8 1.000		988.6	0.0	0.0	0.0	0.0	0.0	0.0	BS
	13.8/ 0.0									
-----ID TO BUS NAME										
	13.20/230.00	G2	HIGH	230	988.6	40.8	0.0	121.8	90 N	991 MVA
BUS NAME	VOLTS-PU	...	GENERATION..	...	LOAD.....	...	shunt.....	bus		
	ACTUAL KV/ANGLE		MW	MVAR	MW	MVAR	mw	avail	used	type
G2	HIGH 230 1.048		0.0	0.0	540.0	100.0	0.0	0.0	0.0	B
	241.1/ -7.0									
-----ID TO BUS NAME										
	230.00/ 13.20	G2		13.8	-988.6	81.0	0.0	121.8	90 N	991 MVA
Close										

REPORTS, SELECT BUS_UVOV

This command retrieves filtered under/over voltage bus output data. It calls `busuvovrpt.f` with the following parameters.

```
integer function busuvovrpt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` positioned to the first character following the string `BUS_UVOV`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT BUS_UVOV [ FROM BUS_DATA ]
[ OUTPUT = <filename> ]
WHERE (repeat filter from BUS_LIST)
```

The output is placed in `out_buffer`. The report dialog is shown below.

OVERVOLTAGE/UNDERVOLTAGE										
Own	Zone	Bus name	Base	Type	Voltage		Relative	Voltage range		Violation
					KV	PU	%	minimum	maximum	pu
								pu	pu	
	01	G31	13.8	BS	12.97	0.9400		0.9500	1.0520	-0.0100
	01	RECT1AC	230.0	B	251.21	1.0922		0.9500	1.0700	0.0222
BPA	01	RECTFER1	113.0	BM	104.12	0.9214		0.9500	1.0520	-0.0286
BPA	01	RECTFER2	113.0	BM	104.12	0.9214		0.9500	1.0520	-0.0286
LAP	02	INVERTER	113.0	BM	102.84	0.9101		0.9500	1.0520	-0.0399
	02	MID PS	500.0	B	553.47	1.1069		1.0000	1.1000	0.0069
	03	G32	13.8	BQ	12.97	0.9400		0.9500	1.0520	-0.0100
	03	RECT2AC	230.0	B	251.21	1.0922		0.9500	1.0700	0.0222

Close

REPORTS, SELECT LINE_COMPARISON

This command retrieves filtered line loading differences between the base case in residence and a selected base case history data file. It calls `lfodifcpt.f` with the following parameters. (*Not currently working.*)

```
integer function lfodifcpt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `LINE_COMPARISON`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT LINE_COMPARISON
      [ OUTPUT = <filename> ]
      FILE = <filename>
      WHERE (repeat filter from BUS_LIST)
```

The output is placed in `out_buffer`. The report dialog is shown below.

REPORTS, SELECT NETWORK_CHANGES

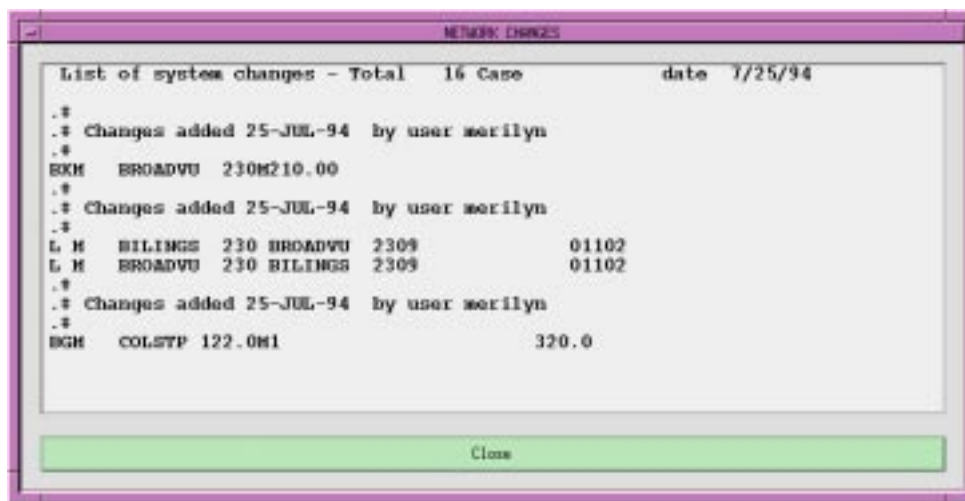
This command retrieves the list of all accumulated changes performed on the base case in residence. It calls `chglilsrpt.f` with the following parameters.

```
integer function chglilsrpt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `NETWORK_CHANGES`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT NETWORK_CHANGES
      [ OUTPUT = <filename> ]
```

The output is placed in `out_buffer`. The report dialog is shown below.



REPORTS, SELECT NETWORK_DELETIONS

This command retrieves the list of all deleted network data in WSCC format. It calls `deleterpt.f` with the following parameters.

```
integer function deleterpt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `NETWORK_DELETIONS`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT NETWORK_DELETIONS
      [ OUTPUT = <filename> ]
```

The output is placed in `out_buffer`. The report dialog is shown below. (This is the report accessed under Bone Pile.)



REPORTS, SELECT OVERLOADED_LINES

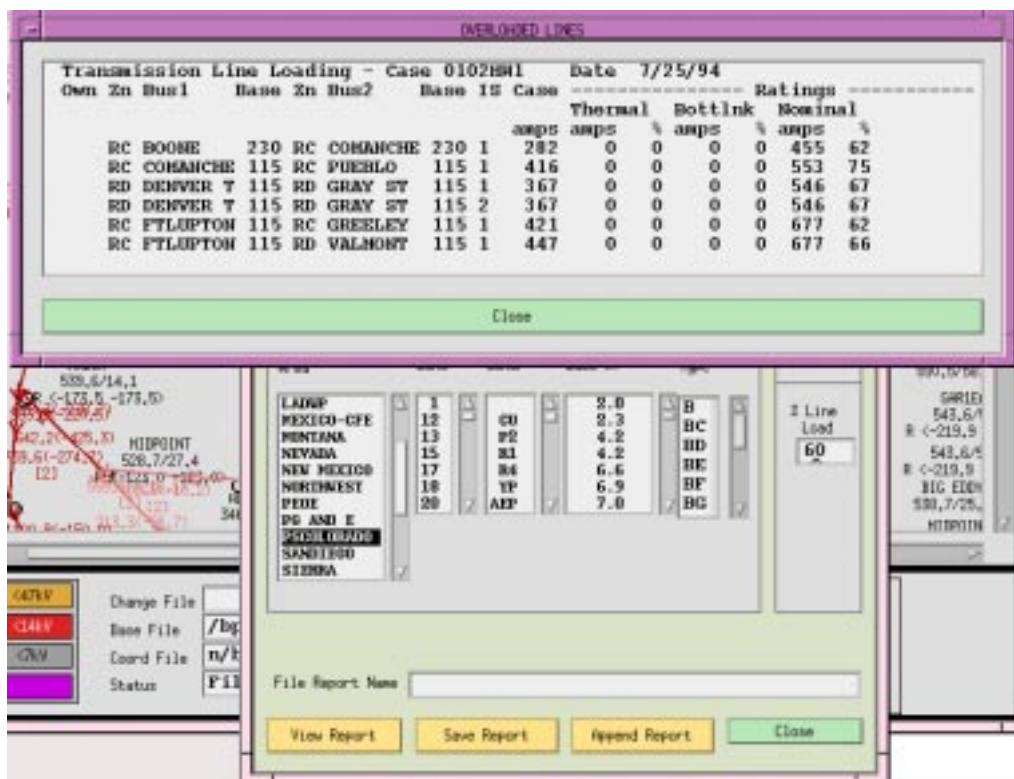
This command retrieves filtered overloaded branch output data. It calls `ovldlnsrpt.f` with the following parameters.

```
integer function ovldlnsrpt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` positioned to the first character following the string `OVERLOADED_LINES`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT OVERLOADED_LINES [ FROM BUS_DATA ]
[ OUTPUT = <filename> ]
WHERE (repeat filter from BUS_LIST)
```

The output is placed in `out_buffer`. The report dialog is shown below.



REPORTS, SELECT OVERLOADED_TXS

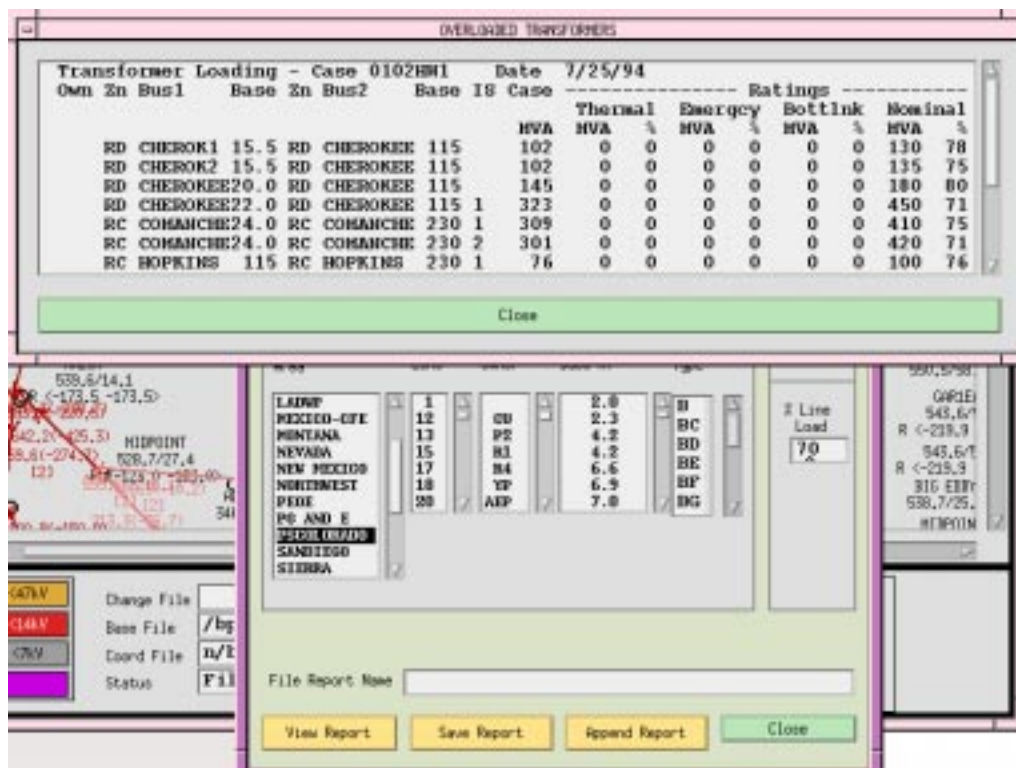
This command retrieves filtered overloaded transformer output data. It calls `ovldtxsrpt.f` with the following parameters.

```
integer function ovldtxsrpt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `OVERLOADED_TXS`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT OVERLOADED_TXS [ FROM BUS_DATA ]
[ OUTPUT = <filename> ]
WHERE (repeat filter from BUS_LIST)
```

The output is placed in `out_buffer`. The report dialog is shown below.



REPORTS, SELECT PHASE_SHIFTER

This command retrieves the phase shifter report. It calls `phshftrpt.f` with the following parameters.

```
integer function phshftrpt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `PHASE_SHIFTER`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT PHASE_SHIFTER
      [ OUTPUT = <filename> ]
```

The output is placed in `out_buffer`. The report dialog is shown below.

PHASE SHIFTER											
Summary of phase shifters - Total 25 Case											
date 1/25/94											
0-----	Phase-shifter	P S	-Phase (degrees)-			Flow (MM)			dP/dT		
			tap	min	max	actual	min	max	(MM/d)		
BILINGS	161.0	BLGS PHA	161.0	.0	-60.0	.0	5.1	-36.0	-34.0	-.879	
BILINGS	230.0	BLGS PHA	230.0	.0	-60.0	.0	44.9	-70.0	-68.0	-.329	
CAL S PS	120.0	CAL SUB	120.0	-.0			-49.3				
CROS PHA	230.0	CROSSPHM	230.0	.0	-.0	75.0	-93.2	-39.0	-35.0	-.469	
DELTAPS	345.0	INTERSTA	345.0	-.0			-389.3				
FT CH PS	120.0	FT CHUR	120.0	-21.1			49.1				
GLECH PS	230.0	GLENCAMY	230.0	.0			142.2				
H ALLEN	345.0	HA PS	345.0	3.4	-60.0	60.0	102.1	95.0	105.0	-.455	
INYO	115.0	INYO PS	115.0	7.8	-30.0	30.0	50.0	50.0	50.0	-1.823	
JEFFERSON	161.0	JFRENPHA	161.0	-.0			-34.6				
LADD	230.0	LAGRANDE	230.0	.0			25.6				
LIBERTY	230.0	LINTYPHS	230.0	-.0			-13.6				
NLY230	230.0	NLYPHS	230.0	0.0	-40.3	40.3	126.0	-400.0	400.0	-.240	

Close

REPORTS, SELECT TIE_LINE_SUMMARY

This command retrieves filtered area tie line flows. It calls `inttierpt.f` with the following parameters.

```
integer function inttierpt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `TIE_LINE_SUMMARY`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT TIE_LINE_SUMMARY
      [ OUTPUT = <filename> ]
      WHERE (repeat filter from BUS_LIST)
```

The output is placed in `out_buffer`. The report dialog is shown below.

TIE_LINE_SUMMARY							
Tie line summary of area interchange case						date 6/11/93	
Area 1	Area 2	/----- Intertie branch -----/					
		Zone Bus 1		Zone Bus 2		line flow	
0AREA 1	AREA 2	01 * G31HIGH	500.0	02 MALINA	230.0	7.9	
		01 G31HIGH	500.0	02 * MID PS	500.0	1163.5	
		01 RECT1AC	230.0	02 * G1	13.8	-500.0	
		01 RECT1AC	230.0	02 * G1X	13.8	0.0	
		01 RECT1AC	230.0	02 * G2 HIGH	230.0	-224.3	
Close							

REPORTS, SELECT VOLTAGE_COMPARISON

This command retrieves filtered voltage differences between the base case in residence and a selected base case history data file. It calls `vltdifprt.f` with the following parameters. (*Not currently working.*)

```
integer function vltdifprt (in_buffer, out_buffer)
parameter (MAXBUFFER = 6600)
character in_buffer * (MAXBUFFER)
character out_buffer * (MAXBUFFER)
```

`in_buffer` is positioned to the first character following the string `VOLTAGE_COMPARISON`. The character array `in_buffer` contains the following information.

```
/REPORTS, SELECT VOLTAGE_COMPARISON
      [ OUTPUT = <filename> ]
      FILE = <filename>
      WHERE (repeat filter from BUS_LIST)
```

The output is placed in `out_buffer`. The report dialog is shown below.

CHAPTER 4

CUSTOMIZING IPF

4.1 INTRODUCTION

X resources are X system components managed in common by the X server. Examples of resources are colors, fonts and their characteristics, default size and position of windows, and default file names for dialog boxes. In IPF, you can change many of these resource values. (Some cannot be changed without affecting adversely the system consistency.)

You customize IPF, as you do other X clients, by changing the client's X resources file. This is an ASCII text file, so you can edit it with any text editor. For some X clients, like your window manager, a special X client allows you to customize client resources with a windows/menus interface. However, for IPF you need to edit IPF's X resources file, XGUI, with a text editor. Because of this, it is recommended that only advanced users edit XGUI. This file is in your home directory.

Note: After you make changes to the XGUI file, you may *not* see changes in your client (IPF) till you exit the X Window System itself. Exiting and restarting IPF is not sufficient! You must exit IPF and the X Window System and restart both.

This appendix is meant only to get you started with customizing IPF. Though making changes to X resource values is generally easy, you may cause problems that you won't know how to fix without doing some further study of X resources. For example, if you accidentally change some branch colors to the main window's background display color, you will not see those branches, even though they are still there! Caution is urged when you modify X resource values and specifications.

Refer to the following X Window System user's book (or any of the many other X user's books) for more information:

- Quercia, Valerie and Tim O'Reilly. *X Window System User's Guide OSF/Motif Edition*. O'Reilly & Associates, Inc., 1991. This is a good, general introduction to X and OSF/Motif.

4.2 XGUI RESOURCES

As you look at the following XGUI file excerpts, note these characteristics:

- For the most part, each line specifies a resource and value for the resource.

- Each resource specification consists of a descriptive resource name followed by a colon (:) followed by a value.
- Values may be numeric or alpha.
- Comment lines are preceded by an exclamation point (!).
- Categories of resources are grouped for reading convenience.
- The IPF resources you can change — the ones in the XGUI file — are window size and position, fonts, colors, and file names.

4.3 CHANGING IPF RESOURCES

Most values for resources are easy to figure out. Others, like colors, are more difficult. Here are a few hints to get you started. Be sure to read further in Quercia's book, or a comparable book, for more information.

Note: If you make a spelling mistake or the value is not correct for the resource, there are no error messages. The resource is ignored. And, there are no X Window System facilities to help you find your error, so be careful!

A.3.1 Changing Window Position and Size

Windows are positioned on the screen by specifying pixel locations. X is the horizontal dimension. Y is the vertical dimension. The position $x, y = (0, 0)$ is the upper left corner of the screen. (XGUI.x, XGUI.y) represents the upper left-hand corner of the main window of IPF.

To change the main window default position:

The default in the XGUI file is:

```
XGUI.x: 127
XGUI.y: 0
```

Change the 0 to a 127:

```
XGUI.x: 127
XGUI.y: 127
```

Now the upper left-hand corner of the main window appears 127 pixels down and to the right of the 0,0 screen pixel position in the extreme upper left-hand corner of the screen.

To change the main window default size:

The default in the XGUI file is:

```
XGUI.width: 600
XGUI.height: 550
```

Change the 550 to a 600:

```
XGUI.width: 600
XGUI.height: 600
```

Now the IPF main window will be square.

Other windows and dialog boxes in XGUI work similarly.

A.3.2 Changing Fonts

The X Window System for your computer comes with a standard set of fonts. Commonly used fonts such as Courier (a typewriter-like font), Times (a serif, “newspaper” font), and Helvetica (a common, sans serif font) are included. Ask your system administrator about the X fonts available on your computer. Each size, shape, and kind of font has a unique name. Here is the name of one of IPF's default fonts:

```
--Courier-Bold-R-----100-----ISO8859-1
```

This X font name specifies Courier bold with a normal slant (R) of size 10 points. Its International Standards Organization character set registry identification is ISO8851-1. The asterisks denote “don't care” states for the other font parameters such as, for example, foundry (who made the font) and pixel size.

To change a default font:

For example, the default font in the XGUI file for descriptive IPF text in the windows and dialog boxes is:

```
XGUI*XmText.fontList:--Courier-Bold-R-----100-----ISO8859-1
```

Change the Courier to Times:

```
XGUI*XmText.fontList:--Times-Bold-R-----100-----ISO8859-1
```

Now the descriptive text is a 10 point bold Times of a regular slant.

A.3.3 Changing Colors

X color values can be specified as regular names such as blue, red, yellow, magenta, slate blue, sky blue, navy blue, etc., or as hexadecimal digits. Ask your system administrator for a list of all named standard colors because this is much easier to deal with than figuring out hexadecimal color values. However, here is a quick explanation of the hexadecimal color specification system.

Colors are specified in the RGB (Red-Green-Blue) color system. The RGB hex numbers have 12 digits. The first four stand for the Red component. The middle four stand for the Green component. And the last four stand for the Blue component. (Reading left to right, of course.) ffff hex stands for fully saturated red if it is in the first position. 8000 hex stands for a half way saturated red (called “red4” in X). 0000 hex stands for no red at all, which would be black. The green and blue work analogously. Thus, ffffffffffff hex stands for pure white, and 000000000000 hex stands for pure black. To get the color you want, you need to play with different values of hex numbers in the appropriate Red-Green-Blue positions since the RGB intensities are mixed to render one color.

To change a default color:

For example, the default color in the XGUI file for the 500 kV branches as shown in the branch color key of the IPF main window is:

```
XGUI*kv_500_label.background: #ffffcccc0000
```

Note the hexadecimal color value specification. This default color is a gold-looking color.

Use a color name specification and change the #ffffcccc0000 to yellow:

```
XGUI*kv_500_label.background: yellow
```

Now the 500kV branches and the branch color key shows up as pure yellow.

A.3.4 Changing Default File Names

When you first execute IPF and select the Open command from the main window menu, some default file names fill some of the file text boxes. You can specify your own valid file names or file name masks in the XGUI file.

To change a default file mask:

For example, the default coordinate file mask in the XGUI file for the Open dialog box is:

```
XGUI*open_dia_coord_dir_text.value: /shr5/all/ipf/dat/*.cor
```

The *.cor selects just the coordinate files in the given directory (if they all end with .cor, of course). (The asterisk (*) is a UNIX wildcard character meaning “any arbitrary length string of characters.”)

Change the /shr5/all to /archive/year1992:

```
XGUI*open_dia_coord_dir_text.value: /archive/year1992/ipf/dat/*.cor
```

Now the coordinate files selected are in the /archive/year1992/ipf/dat directory.

You can change any part of the file name, of course, just so long as the file name is a valid file name for your operating system.

4.4 THE XGUI FILE

The entire XGUI file follows. This may not precisely match the copy you have, of course.

```
#####
#
#  THIS IS A RESOURCE FILE - ALSO CALLED THE 'XGUI' FILE.
#( This file contains some DEFAULTS for the GUI program. )
#
#####
# below are the user specified print strings
# WARNING: the listItemCount must be LESS THAN OR EQUAL TO
# the number of comma separated listItems or program will crash!!!
#####

XGUI*printer_selection_box*textString: print
XGUI*printer_selection_box*listItems: lpr, \
lpr -Pps, \
print/queue=myprinter, \
lp -d COMPAQ20 -T ps
XGUI*printer_selection_box*listItemCount: 4

! Sizes & positions:

XGUI*cflow_socket_number_text.value: 2001
XGUI*cflow_sel_text.value: 2002

XGUI.x: 0
XGUI.y: 0
!XGUI.width: 825
!XGUI.height: 672

XGUI*highlightThickness: 0
XGUI*XmText.marginWidth: 2
XGUI*XmText.marginHeight: 2

XGUI*ipc_command_board.x: 793
XGUI*ipc_command_board.y: 0
XGUI*ipc_command_board.width: 475
XGUI*ipc_command_board.height: 703

XGUI*ipf_report_list_dialog.x: 0
XGUI*ipf_report_list_dialog.y: 650
XGUI*ipf_report_list_dialog.width: 740
XGUI*ipf_report_list_dialog.height: 330

XGUI*bus_front_box.x: 0
XGUI*bus_front_box.y: 0
XGUI*bus_front_box.width: 370
```



```

XGUI*bus_front_box.height: 600

XGUI*bus_output_dialog.x: 500
XGUI*bus_output_dialog.y: 665

XGUI*bus_elec_info_dialog.x: 0
XGUI*bus_elec_info_dialog.y: 665

XGUI*switched_reactance_dialog.x: 650
XGUI*switched_reactance_dialog.y: 100

XGUI*error_message_dialog.x: 0
XGUI*error_message_dialog.y: 500

XGUI*area_selection_box.busVisibleItemCount: 7
XGUI*area_selection_box.branchVisibleItemCount: 7
XGUI*area_selection_box.listVisibleItemCount: 7

!STANDARD GUI PLOT DEFAULTS
XGUI*print_size_x_text.value: 21.59
XGUI*print_size_y_text.value: 27.94
XGUI*print_border_xpos_text.value: 21.59
XGUI*print_border_ypos_text.value: 27.94
XGUI*print_cmnt_xpos_text.value: 12.7
XGUI*print_cmnt_ypos_text.value: 3.7
XGUI*print_x_offset_text.value: 0.0
XGUI*print_y_offset_text.value: 0.0
XGUI*print_case_xpos_text.value: 12.7
XGUI*print_case_ypos_text.value: 4.0
XGUI*print_x_scale_text.value: 1.0
XGUI*print_y_scale_text.value: 1.0

XGUI*plot_comments_text: UserComments
XGUI*label_box_x.value: 12.0
XGUI*label_box_y.value: 4.5
XGUI*coord_name_x.value: 10.5
XGUI*coord_name_y.value: 4.0
XGUI*pf_data_x.value: 10.5
XGUI*pf_data_y.value: 4.0
XGUI*legend_x_text.value: 5.0
XGUI*legend_y_text.value: 0.5

!XGUI*unitType: 4

! this will shrink font for on screen graphics ???????????????
! Nothing works here for UnixWare - "Cannot convert string to type FontList"
!XGUI*drawwindow*fontList: *-HELVETICA*-R-***-10-***-***-ISO8859-1

! USE THIS FOR DECSTATIONS
!XGUI*XmText.fontList: *-Courier-Bold-R-***-100-***-***-ISO8859-1

```

```
!XGUI*XmlList.fontList:--*Courier-Bold-R-***-100-***--ISO8859-1
```

```
! USE THIS FOR VAXSTATIONS
```

```
XGUI*XmlText.fontList:--*Courier-Bold-R-***-140-***--ISO8859-1
```

```
XGUI*XmlList.fontList:--*Courier-Bold-R-***-140-***--ISO8859-1
```

```
! Colors:  R    G    B
```

```
!                                     |--| |--| |--|
```

```
XGUI*foreground: #000000000000
```

```
XGUI*background: #e000e000e000
```

```
XGUI*XmlText.background: #f000f000f000
```

```
XGUI*XmlList.background: #f000f000f000
```

```
!XGUI*ac_E_line_info_form.background:#d000d000d000
```

```
!XGUI*ac_L_line_info_form.background:#d000d000d000
```

```
!XGUI*area_selection_dialog.background:#d000d000d000
```

```
!XGUI*branch_selection_dialog.background:#d000d000d000
```

```
!XGUI*bus_selection_dialog.background:#d000d000d000
```

```
!XGUI*continuation_bus_edit_dialog.background:#d000d000d000
```

```
!XGUI*exit_warning_dialog.background:#d000d000d000
```

```
!XGUI*file_menu_open_dialog.background:#d000d000d000
```

```
!XGUI*ipc_command_board.background:#d000d000d000
```

```
!XGUI*status_dialog.background:#d000d000d000
```

```
!XGUI*tool_dialog.background:#d000d000d000
```

```
!XGUI*transformer_dialog.background:#d000d000d000
```

```
XGUI*overload_high_label.background:      #ffff38e838e8
```

```
XGUI*overload_high_label.foreground:      #ffffffffffff
```

```
XGUI*overload_moderate_label.background:   #fb82ed900000
```

```
XGUI*overload_moderate_label.foreground:   #000000000000
```

```
XGUI*overload_low_label.background:        #aaa603660000
```

```
XGUI*overload_low_label.foreground:        #ffffffffffff
```

```
XGUI*overload_none_label.background:       #000000000000
```

```
XGUI*overload_none_label.foreground:       #ffffffffffff
```

```
XGUI*overload_extreme_text.value:          110
```

```
XGUI*overload_moderate_text.value:         100
```

```
XGUI*overload_mild_text.value:             90
```

```
XGUI*kv_500_label.background:#ffffcccc0000
```

```
XGUI*kv_500_label.foreground:#000000000000
```

```
XGUI*kv_345_label.background:#e08074007200
```

```
XGUI*kv_345_label.foreground:#000000000000
```

```
XGUI*kv_300_label.background:#9c0078000300
```

```
XGUI*kv_300_label.foreground:#ffffffffffff
```

```
XGUI*kv_230_label.background:#20002000c000
```

```
XGUI*kv_230_label.foreground:#ffffffffffff
```

```
XGUI*kv_161_label.background:#0700a700ac00
```

```

XGUI*kv_161_label.foreground:#ffffffffffff
XGUI*kv_138_label.background:#0e8095002700
XGUI*kv_138_label.foreground:#ffffffffffff
XGUI*kv_115_label.background:#000000000000
XGUI*kv_115_label.foreground:#ffffffffffff
XGUI*kv_69_label.background:#3800bd00ffff
XGUI*kv_69_label.foreground:#000000000000
XGUI*kv_46_label.background:#e180aac03d00
XGUI*kv_46_label.foreground:#000000000000
XGUI*kv_14_label.background:#dc0023d023d0
XGUI*kv_14_label.foreground:#ffffffffffff
XGUI*kv_7_label.background:#a000a000a000
XGUI*kv_7_label.foreground:#000000000000
XGUI*kv_dc_label.background:#c7000000e000
XGUI*kv_dc_label.foreground:#ffffffffffff

XGUI*active_pb_color:#ffffe4848a3c
!
! interactive command window params
!
XGUI*main_command_window.historyVisibleItemCount: 4
XGUI*ipc_scroll_cmds_pftogui.height: 200
XGUI*ipc_scroll_cmds_guitopf.height: 200

! This is the dir mask when the open file dialog is first brought up.

XGUI*file_selection_box_open.dirpf
Mask: ../dat/*

! This is the dir mask when the open linez dialog is first brought up.

XGUI*line_z_filesel.directory: ../dat
XGUI*line_z_filesel.pattern: *.lcd

! These are the default dir masks in the open file menu.

XGUI*open_dia_command_dir_text.value: ../dat/*.pcl
XGUI*open_dia_change_dir_text.value: ../dat/*.chg
XGUI*open_dia_base_dir_text.value: ../dat/*.bse
XGUI*open_dia_network_dir_text.value: ../dat/*.net
XGUI*open_dia_coord_dir_text.value: ../dat/*.cor

! These are the default filepaths in the open file menu.
! Leave blank where you don't want a default.
! Don't set defaults for both base and network!!!

XGUI*file_select_dia_command_text.value:
XGUI*file_select_dia_change_text.value:
XGUI*file_select_dia_base_text.value:
XGUI*file_select_dia_network_text.value:

```

```

XGUI*file_select_dia_coord_text.value:

#####
#
# Notes to users:
#The following tutorial explains how to set your own values for
#1) Dialog window default location
#2) Dialog window default size
#3) Dialog window default background color
#4) Text box default values
#
#####
#
# To set a default window location, size or background color, first the
# internal name of the dialog window must be known. The following list is
# a partial list of some important dialog windows:
#
# XGUI      (main)
# XGUI*ipc_command_board      XGUI*ipc_report_dialog
# XGUI*bus_front_box          XGUI*bus_output_box
# XGUI*error_message_dialog   XGUI*print_opt_display_dialog
# XGUI*print_dialog           XGUI*reports_form_1
# XGUI*reports_form_2         XGUI*reports_form_3
# XGUI*reports_form_4         XGUI*ipc_scroll_cmds_pftogui
# XGUI*area_selection_dialog  XGUI*area_interchange_box
# XGUI*bus_sect_dialog        XGUI*bus_edit_dialog (network editor)
# XGUI*modify_bus_coord_dia   XGUI*cflow_selection_dialog
# XGUI*open_file_dialog       XGUI*save_file_dialog
# XGUI*help_dialog            XGUI*error_message_dialog
# XGUI*ipc_command_window     XGUI*ipf_alpha_bus_list_dialog
# XGUI*line_tap_dialog        XGUI*linetap2
# XGUI*line_Z_calc_dialog     XGUI*line_z_filesel
# XGUI*line_z_save_dialog     XGUI*print_opt_page_dialog
# XGUI*plot_options_dialog    XGUI*user_comment_dialog
# XGUI*printer_select_dialog  XGUI*select_reports_dialog
# XGUI*pf_report_dialog       XGUI*reports_file_select_dia
# XGUI*reports_overwrite_dialog XGUI*solve_dialog
# XGUI*bus_help_dialog        XGUI*pf_descp_form
#
#
#       For example if it is desired that the error box should appear
# 600 pixels from left edge of screen, 250 pixels from top edge and
# the dialog is supposed to be 333 pixels wide, 488 pixels high, then the
# following four lines should be edited into the XGUI file:
#
# XGUI*error_message_dialog*x:      600
# XGUI*error_message_dialog*y:      250
# XGUI*error_message_dialog*width:  333
# XGUI*error_message_dialog*height: 488
#

```

```
#####
#
# Setting background and foreground colors is a more complicated, but can be
# achieved if one recalls the format for the 500kv label color above:
#
# ! Colors:
# !
# XGUI*foreground:
# XGUI*background:
#
# XGUI*kv_500_label.background:
# XGUI*kv_500_label.foreground:
#
# What the above is saying, is the screen is composed of 3 colors, Red, Green
# and Blue. Each color may have a value from zero (0000) to
# (ffff)Hexidecimal.
# (000000000000) is black, (ffffffffffff) is white, (ffff00000000) is red,
# (0000ffff0000) is green and (00000000ffff) is blue.
#
# The default background and foreground values for any screen which did not
# have a background or foreground color will be 00000000 (black) on
# d000d000d00
# (almost white).
#
# The example for the kv_500_label background is ffff red, cccc green and
# 0000 black which comes out a shade of yellow. This will be the color for
# the 500 kv transmission lines and the 500kV legend background.
#
# If one wishes to alter the colors for overloaded lines, then the following
# lines should be changed to suit:
#
# XGUI*overload_high_label.background:
# XGUI*overload_high_label.foreground:
# XGUI*overload_moderate_label.background:
# XGUI*overload_moderate_label.foreground:
# XGUI*overload_low_label.background:
# XGUI*overload_low_label.foreground:
#
#####
#
# ANOTHER USE FOR THE RESOURCE FILE IS TO SPECIFY SOME DEFAULT VALUES
# FOR VARIOUS TEXT BOXES. For example is one wants to have a default name
# appear in the coordinate file selection box, then the following line might
# be useful:
#
# XGUI*file_select_dia_coord_text.value: newtestdc3.cor
#
# As of April 1994, there are literally hundreds of potential text boxes
# available in GUI which can be preset. New text boxes are continually being
# added and old text boxes are occasionally deleted. At this time it would be
```

```
# very difficult to maintain an accurate list of all available text values.
#
# Listing them all is beyond the scope of this report.  However a partial list
# is available for some of the plot options.  If the text box is not listed
# below, then they might be looked up by peeking inside of one of the *.u
# files.
#
#
##### List of all known text boxes as of APR/29/1994 #####
#
# ***** acbusform.u *****      ( jacket for bus_front_box )
#       bus_owner bus_zone
#       bus_v_max bus_v_hold
#       bus_v_min bus_remote_name
#       bus_remote_pcs bus_phase_xx
#       bus_remote_base bus_p_load
#       bus_q_load bus_p_shunt
#       bus_q_shunt bus_p_max
#       bus_p_gen bus_q_sched
#       bus_q_min bus_q_min_2
#       bus_p_max_2 bus_p_gen_2
#       bus_q_max bus_q_min_3
#       bus_p_max_3 bus_phase
#       bus_q_sched_3
#
# ***** aclineform.u *****      ( jacket for bus_front_box )
#       line_meter line_owner
#       line_circuit_id line_section
#       line_R line_G
#       line_X line_B
#       line_no_parallel line_miles
#       line_rating_emergency line_rating_thermal
#       line_rating_bottleneck_text
#
# ***** areaselect.u *****      ( area_interchange_box )
#       area_scheduled_export_text area_base_kv_text
#       area_max_volt_text area_min_volt_text
#       area_slack_bus_name_text area_name_text
#       area_zone_1_text area_zone_2_text
#       area_zone_3_text area_zone_4_text
#       area_zone_5_text area_zone_6_text
#       area_zone_7_text area_zone_8_text
#       area_zone_9_text area_zone_10_text
#       interchange_area_1_text interchange_area_2_text
#       interchange_export_text
#
# ***** buscoord.u *****
#       full_name_text diag_name_text
#       bus_x_pos_text bus_y_pos_text
```

```

#      name_x_pos_text name_y_pos_text
#      nom_volts_text generator_angle_text
#      reactor_angle_text
#
# ***** buseditn.u ***** ( bus_edit_dialog ) - network editor
#      edit_bus_id1_text edit_column_num_text
#      sect_bus_id1_text sect_bus_id2_text
#      bus_sect_dia_tie_text
#
# ***** cflow.u ***** ( cflow_selection_dialog,
cflow_socket_request_dialog )
#      cflow_command_text cflow_sel_text
#      cflow_socket_number_text
#
# ***** contform.u ***** ( jacket for bus_front_box )
#      cont_gen_qmin cont_gen_qmax
#      cont_gen_p cont_shunt_q
#      cont_load_q cont_shunt_p
#      cont_load_p cont_owner_text
#      code_year_text
#
# ***** dcbusform.u ***** ( jacket for bus_front_box )
#      dc_bus_owner_text dc_bus_zone_text
#      dc_bus_bridges_text dc_bus_smooth_reac_text
#      dc_bus_max_angle_text dc_bus_value_drop_text
#      dc_bus_bridge_rate_text dc_bus_min_angle_text
#      dc_bus_commutate_text dc_bus_conv_type_text
#      dc_bus_ign_del_text dc_bus_min_ext_text
#      dc_bus_power_text dc_bus_voltage_text
#
# ***** dclineform.u ***** ( jacket for bus_front_box )
#      dc_line_owner_text dc_line_length_text
#      dc_line_resistance_text dc_line_inductance_text
#      dc_line_nominal_text dc_line_thermal_text
#      dc_line_bottleneck_text dc_line_capacitance_text
#      dc_line_controls_text dc_line_ign_text
#      dc_line_ext_text dc_line_power_text
#      dc_line_voltage_text dc_line_in_month_text
#      dc_line_in_year_text dc_line_out_month_text
#      dc_line_out_year_text dc_line_metering_text
#
# ***** equivform.u ***** ( jacket for bus_front_box )
#      equiv_meter_text equiv_owner_text
#      equiv_circuit_id_text equiv_section_text
#      equiv_R_text equiv_G1_text
#      equiv_X_text equiv_B1_text
#      equiv_parallel_text equiv_G2_text
#      equiv_B2_text equiv_rating_nominal_text
#      equiv_rating_thermal_text equiv_rating_bottleneck_text
#

```

```
# ***** fopendia.u *****          ( open_file_dialog )
#     file_select_dia_base_text open_dia_base_dir_text
#     file_select_dia_change_text file_select_dia_command_text
#     file_select_dia_coord_text open_dia_coord_dir_text
#     file_select_dia_network_text open_dia_network_dir_text
#open_dia_command_dir_text          open_dia_change_dir_text
#
# ***** frontdia.u *****          ( bus_front_box )
#     bus_front_name bus_front_volts
#     bus_front_line_name bus_front_volts_2
#     bus_front_name_2
#
# ***** fsavedia.u *****          ( save_file_dialog )
#     save_change_text save_base_text
#     save_net_text save_coord_text
#     save_stab_text
#
# ***** help.u *****
#     help_annotate_text error_dia_identifier_text
#     error_dia_line_text error_dia_message_text
#
# ***** ipcwindow.u *****
#     ipc_text_guitopf ipf_alpha_search_text
#
# ***** linetap.u *****
#     line_tap_name_text line_tap_bus_1_text
#     line_tap_bus_2_text
#
# ***** linetap2.u *****
#     line_tap_section_2 line_tap_section_1
#     line_tap_bus_1 line_tap_name
#     line_tap_bus_2
#
# ***** linezcalc.u *****
#     line_z_distance_text line_z_basekv_text
#     line_z_z1_R_text line_z_y_G_text
#     line_z_z1_X_text line_z_y_B_text
#     line_z_z0_R_text line_z_z0_X_text
#     line_z_edit_number_text line_z_edit_od_text
#     line_z_edit_phase_text line_z_edit_resist_text
#     line_z_edit_od_text line_z_edit_skin_text
#     line_z_edit_horiz_text line_z_edit_vtower_text
#     line_z_edit_vmid_text line_z_edit_separ_text
#     line_z_edit_angle_text line_z_save_text
#
# ***** main.u *****          ( gui_main )
#     overload_extreme_text overload_moderate_text
#     overload_mild_text status_change_file_text
#     status_base_file_text status_coord_file_text
#     status_status_text tools_zoom_factor_text
```



```

#
# ***** pf_descrip.u *****          ( pf_descp_form )
#     pf_case_id_text pf_header_1_text
#     pf_header_2_text pf_case_descrip_text
#     pf_header_3_text pf_comments_text
#
# ***** pqcrvform.u *****
#     pq_curve_p_gen_text pq_curve_q_max_text
#     pq_curve_q_min_text pq_curve_p_gen_text
#
# ***** printopt.u *****
#     print_cmnt_xpos_text print_cmnt_ypos_text
#     print_border_xpos_text print_border_ypos_text
#     print_case_xpos_text print_case_ypos_text
#     print_size_x_text print_size_y_text
#     print_x_offset_text print_y_offset_text
#     print_x_scale_text print_y_scale_text
#     label_box_x label_box_y
#     label_box_se_x label_box_se_y
#     legend_x_text legend_y_text
#
# ***** recorddia.u *****
#     recorder_comand_number recorder_command_line
#
# ***** regxfmr.u *****              ( jacket for bus_front_box )
#     regxfmr_owner_text regxfmr_remote_name_text
#     regxfmr_remote_kv_text regxfmr_max_tap_text
#     regxfmr_taps_text regxfmr_maxMVAR_text
#     regxfmr_schedMVAR_text regxfmr_minMVAR_text
#     regxfmr_min_tap_text regxfmr_max_shift_text
#     regxfmr_min_shift_text regxfmr_shifts_text
#     regxfmr_maxflow_text regxfmr_schedflow_text
#     regxfmr_minflow_text
#
# ***** reportdia.u *****
#     reports_file_name_text reports_alpha_entry_text
#     reports_loading_max_text reports_loading_min_text
#
# ***** solve.u *****
#     solve_decoupled_text solve_newton_text
#
# ***** swrreac.u *****              ( jacket for bus_front_box )
#     swr_reac_remote_name_text swr_reac_remote_kv_text
#     swr_reac_owner_text swr_reac_steps_1_text
#     swr_reac_steps_2_text swr_reac_steps_3_text
#     swr_reac_steps_4_text swr_reac_steps_5_text
#     swr_reac_steps_6_text swr_reac_steps_7_text
#     swr_reac_steps_8_text swr_reac_mvar_1_text
#     swr_reac_mvar_2_text swr_reac_mvar_3_text
#     swr_reac_mvar_4_text swr_reac_mvar_5_text

```

```
#      swr_reac_mvar_6_text swr_reac_mvar_7_text
#      swr_reac_mvar_8_text
#
# ***** tools.u *****
#      tools_zoom_factor_text
#
# ***** xfmr dia.u *****      ( jacket for bus_front_box )
#      xfmr_dialog_X_text xfmr_dialog_R_text
#      xfmr_dialog_G_text xfmr_dialog_B_text
#      xfmr_dialog_id_text xfmr_dialog_section_text
#      xfmr_dialog_owner_text xfmr_dialog_metering_text
#      xfmr_dialog_parallels_text xfmr_dialog_tap1_kv_text
#      xfmr_dialog_tap2_kv_text xfmr_dialog_phase_text
#      xfmr_dialog_nominal_text xfmr_dialog_thermal_text
#      xfmr_dialog_bottleneck_text xfmr_dialog_emerg_text
#
#####
#
#  STANDARD HARDCOPY PLOT DEFAULTS EXPLAINED IN MORE DETAIL:
#
#####
#
# XGUI*print_size_x_text.value: 21.59      This is the metric equivalent of
# XGUI*print_size_y_text.value: 27.94 8.5 x 11 inch paper
#
# XGUI*print_border_xpos_text.value:      The border option puts a border on
the
# XGUI*print_border_ypos_text.value:      diagram in addition to the border
that
# is drawn as part of the diagram in BOX
# option. This option is generally not
# specified.
#
# XGUI*print_cmnt_xpos_text.value:      Comments entered by the user or from
# XGUI*print_cmnt_ypos_text.value:      the powerflow are located relative to
# the BOX option unless another location
# is specified. This option is generally
# not specified.
#
# XGUI*print_x_offset_text.value: 0.0      The origin for the diagram is
generally
# XGUI*print_y_offset_text.value: 0.0      the lower left corner of the paper.
# This option can relocate the origin and
# is generally set to 0.0
#
# XGUI*print_case_xpos_text.value:      The case name is located relative to
# XGUI*print_case_ypos_text.value:      the BOX option unless another
location
# is specified. This option is generally
# not specified.
```

```

#
# XGUI*print_x_scale_text.value: 1.0      The scale factor is generally
specified
# XGUI*print_y_scale_text.value: 1.0      as 1.0 in the xgui.dat file.
However,
# it is often changed through the GUI.
# By making the scale factor less than
# 1.0, the user may specify coordinates
# greater than the paper size and have
# the diagram scaled to fit on the
#
# standard paper.
#
# XGUI*plot_comments_text: UserComments  The location for user comments was
once
# controllable, but is now fixed in the
# label box below the case name and
# description.
#
# XGUI*label_box_x.value: 15.0            BPA users identify diagrams within an
# XGUI*label_box_y.value: 04.0            identification box (usually) in the
# lower right corner of the diagram.
#
# These coordinates are good for a
# portrait orientation diagram with an
# 'x' and 'y' scale of 1.0,
#
# XGUI*coord_name_x.value:                The coordinate name is generally
# XGUI*coord_name_y.value:                located just outside the lower border
# of the diagram. This option is
#
# generally not specified.
#
# XGUI*pf_data_x.value:                   The location for powerflow comments was
# XGUI*pf_data_y.value:                   once controllable, but is now fixed in
# the label box below the user comments.
#
# XGUI*legend_x_text.value:               When the upper left corner of a legend
# XGUI*legend_y_text.value:               box is specified, a legend will be
# displayed in the box. The legend
# identifies each line pattern used in
#
# the hard copy diagram. The legend is
# Postscript code and can be modified by
# the user. This option is generally not# specified.
#####

```


CHAPTER 5

IPF NETWORK DIAGRAMS

5.1 OVERVIEW

IPF has two separate network diagram (map) presentations. One is the display you see in the GUI graphics, and the other is the hard copy map. This chapter addresses the hard copy diagram which is designed for reports, documentation, and analysis. It can be generated as a report from the graphic display or produced in a batch environment.

Both presentations use the same coordinate file format. The most important coordinate data, like bus icon and name locations, and line bending points, can be edited graphically from the GUI by moving things around and saving the altered coordinate file. Some editing, such as entering alpha-numeric data, is done through a text editing window.

The diagram shows power system components modeled in a power flow study. It shows essential bus and branch solution data along with identification data, such as date, case identification, program version, and the options used to generate the diagram. The diagram can be enhanced by adding to the coordinate file such items as:

- A legend.
- A border.
- A control block for case identification, signatures, etc.
- A case title heading.
- Selected tie line flows and loss summary.
- An inset showing detail in a selected area.
- Any PostScript language objects.

These items will show up on a map plotted from the GUI, even though some of them cannot be added, edited, or displayed from the GUI.

5.2 INPUT REQUIREMENTS, OUTPUT, AND OPERATION

The Plot program is a set of FORTRAN subroutines within `ipfmain`, which build a dynamic PostScript objects file of references to a static PostScript objects file (`pfmaster.post`). The same routines are called by the Print Plot command issued from the GUI, a command file entered via IPFBAT or the GUI, and by the batch program IPFPLOT. The dynamic file, which is built by “anding” a coordinate file with Powerflow data, defines *which data* will appear on a diagram. The static ASCII PostScript file describes *how data* will appear on a diagram.

5.2.1 Input Requirements

The input requirements are:

- A coordinate file, built via the GUI and/or an ASCII text editor.
- A solved system network, either from a base case file or currently loaded in IPF.
- The static PostScript file (`pfmaster.post`) defining how data will be shown on the diagram.

5.2.2 Output

A dynamic PostScript file, which is built by the Plot program, is appended to the static PostScript file and sent to a PostScript interpreter (printer or computer display) to produce a diagram.

5.2.3 Plot Program Operation

GUI: Select Print Plot from the File pull-down menu. You can also change the options for the particular plot, by selecting Diagram Options and/or Page Options. These override the options which may be specified in the coordinate file.

IPFBAT: See section 5.6 in this chapter.

IPFPLOT: Enter `ipfplot coordinate_file base_file_1 [base_file_2]`. This is a strictly batch process which does not require the GUI. A coordinate file name and one solved base file must be provided. The second base file is required only for difference maps.

However it is invoked, the Plot program determines which information should appear on the diagram by examining the coordinate file. It then searches the Powerflow data for bus, branch, area, and intertie data that are identified in the coordinate file. When a match is found, the Powerflow and coordinate data are combined and formatted into a dynamic PostScript file to activate procedures on the static PostScript file. In addition to bus, branch, transformer, area, and intertie records, all other coordinate file records — options, draw, >define, comment, and PostScript — are processed by the Plot program and formatted to invoke procedures on the static PostScript file.

5.3 COORDINATE FILE

The coordinate file used by the diagram program is the same file that is generated and used by the GUI display. Since the coordinate file is an ASCII file, it can be generated by any ASCII text editor. The records in that file are described in this section.

The Coordinate file consists of primary coordinate data records and supportive coordinate data records. All coordinates are first quadrant positive Cartesian coordinate values in centimeters. The lower left corner of the diagram is coordinate (0,0). The primary coordinate data records are the focus of diagrams since the records specify the bus and branch locations. These records are:

Bus Record	B
Line Record	L
Transformer Record	T
Area Record	A
Intertie Record	I

The Coordinate file may also include supportive record types to produce a diagram suitable for long term documentation. These records are:

File Identification Record	[ID COORD
Options Record	O
Define Record	>DEFINE_TYPE
Comment Record	C
Draw Record	D
PostScript Record	P
Trailer Record	9 or (*EOR)

Coordinate data records are described in the order they typically appear in the coordinate file.

5.3.1 File Identification Record — [ID COORD

The file identification record signifies the beginning of a set of coordinate records. See Table 5-1. Options are set to their default condition when this record is encountered. One such record is required at the beginning of the coordinate file, but it may optionally contain several [ID COORD records and their associated coordinate record sets. Multiple [ID COORD records may be used to insert a control block, legend, or an inset of a detailed section of the diagram.

Table 5-1 File Identification Record Format

Column	Format	Description
1-9	A9	[ID COORD
9-90		Not used by Plot program, available for user notes.

5.3.2 Options Record — O

Options affect the general appearance of a diagram and determine which Powerflow data will be displayed for each bus and branch on a diagram.

The default option values can be overridden by options specified in the coordinate file. If a coordinate file is currently loaded in the GUI, and the options are changed via the Page Options or Diagram Options menus, these will override (replace) those originally specified in the file. If the file is saved, the new options will be saved in it.

The “=” and “,” and blank field delimiters may be used interchangeably. Option names may be upper or lower case. Upper case characters shown in the tables indicate the minimum mandatory characters for identifying the option.

Table 5-2 Option Record Format

Column	Format	Description
1-2	A 2	OP — identifies an option record.
3-90	A 88	Free field description of option, see tables below for details.

Here is an example of how to specify options in the coordinate file:

```
OPtions OR=L
Options SScale_factor=0.9,0.9
OP BUs_detail=Bus_name,Powerflow_name
```

Page Options

These are general appearance options. Several of them are interdependent. They are applied in the following order:

Orientation
 Scale
 Offset
 Border with transparency flag
 Box for identification
 Case location
 Comment location
 Coordinate file name location
 Size
 Legend location

Table 5-3 Page Options

Option	Description
Size=XX.XX,YY.YY	DEFAULT = 21.59 by 27.94 cm (8.5 by 11 inches).
ORientation=Landscape ORientation=Portrait	DEFAULT=Portrait.
Offset=XX.XX,YY.YY	Lower left of diagram relative to lower left of page. DEFAULT = 0.0, 0.0.
TRansparency=Transparent TRansparency=Opaque	DEFAULT for insets is Opaque. Main diagram is always opaque.
SCale_factor=X.XX,Y.YY	DEFAULT = 1.0, 1.0.
BOrder=XX.XX,YY.YY	Locates upper right corner of border. DEFAULT = no border.
BX=XX.UL,YY.UL,XX.LR, YY.LR BX= 0.0 0.0 selects the WSCC border with case title at the top.	Locates an identification box. XX.UL and YY.UL locate the upper left corner of the box. If XX.LR and YY.LR are zero (0), the box is positioned in the lower right corner of the diagram. The BX option establishes default coordinates for CR (coordinate file), CAs_e_name, and COmments. These locations can be individually overridden . A standard border is drawn as near the edge of the paper as most PostScript printers will allow. DEFAULT = no identification box.
CAse_name=XX.XX,YY.YY	Locates case name from Powerflow program. DEFAULT = no case name.
COmments=XX.XX,YY.YY	Locates comments from user entry and Powerflow program. DEFAULT = no Powerflow comments.
CR=XX.XX,YY.YY	Locates coordinate file name DEFAULT = relative to BX.

Table 5-3 Page Options (Continued)

Option	Description
LG=XX.XX,YY.YY	Locates upper left corner of legend box. DEFAULT = no legend.

The dialog box is titled "Page Options" and contains the following sections:

- Orientation:** Radio buttons for Portrait (selected) and Landscape.
- Transparency:** Radio buttons for Opaque and Transparent (selected).
- Paper Size:** Width: 21.59, Height: 27.94. Preset options: 21.6 x 28 cm (selected), 28 x 43 cm.
- Border Top Right Corner:** X cm: 21.59, Y cm: 27.94.
- Case Name Position:** X cm: 12.7, Y cm: 4.0.
- Comments Position:** X cm: 12.7, Y cm: 3.7.
- Offset:** X cm: 0.0, Y cm: 0.0.
- Scale Factor:** X: 1.0, Y: 1.0.
- Label Box Coordinates (Centimeters):** TOP LEFT CORNER: 12.0, 4.5; BOTTOM RIGHT CORNER: empty.
- Legend Position:** X cm: 5.0, Y cm: 0.5.
- Buttons:** OK (yellow), Close (green).

Figure 5-1 Page Options Dialog Box

Diagram Options

The options described in Table 5-4 determine which Powerflow values will be displayed on a diagram.

Those selections that are ON by DEFAULT may be turned off in one of two ways. Some, such as the bus name selection, may be toggled to the abbreviations on the bus coordinate records, full bus name, or full bus name and base kv. Others, such as generation, may be turned off by preceding

the value of interest with NO_. For example,

to *not* show generation:

```
Option BUs_detail=NO_Generation
or
O BU=NO_G
```

You can also specify on the bus coordinate record whether or not the generation and shunt reactance at this bus should be displayed. See bus coordinate records in Table 5-10 and Figure 5-3 for details. The *no display* indicator on the coordinate records can be overridden with a *draw more* or *always draw* option for generators or shunt reactors. For example,

```
Op BUs_detail=AL_Generators
Op BUs_detail=MO_Generators
Option BUs_detail=AL_Shunt
```

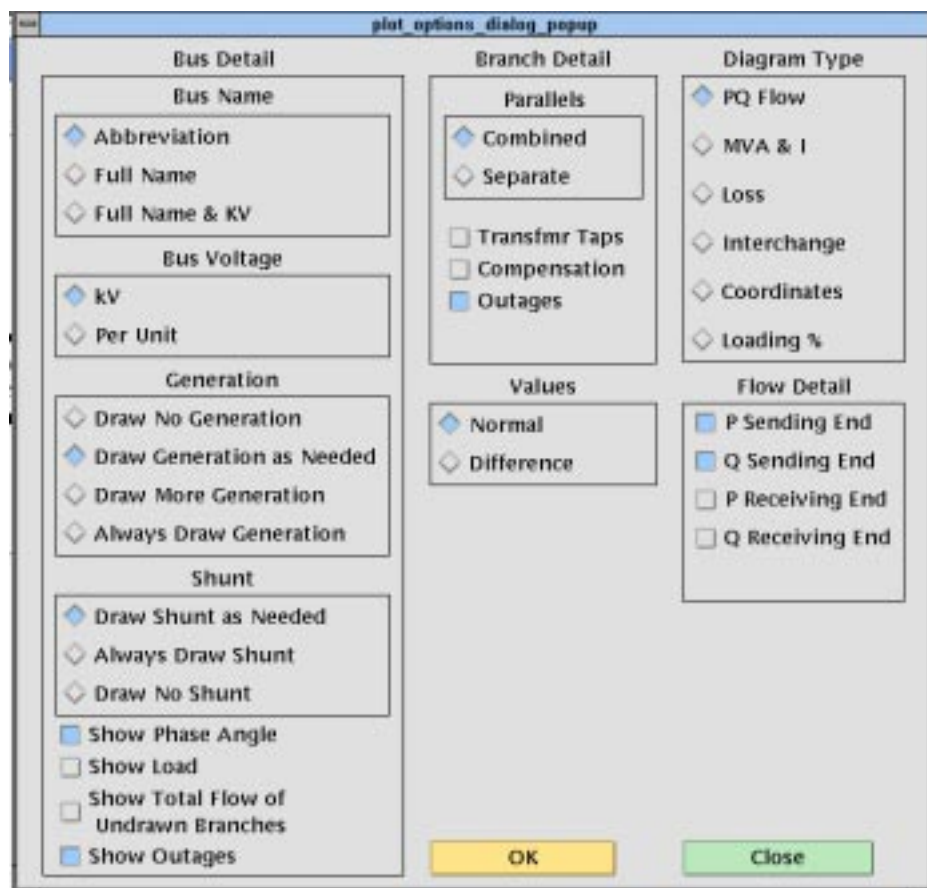


Figure 5-2 GUI Diagram Options

Some of the diagram options are independent; others are mutually exclusive. Look at the GUI menu above to quickly determine which is which. Those with *diamond* buttons are exclusive; those with *square* buttons can be turned on or off in any combination. This applies no matter whether the options are specified from the menu or in the coordinate file.

Table 5-4 Powerflow Values Options

Option	Description
Diagram_type=Pq_flow Diagram_type=Mva/I Diagram_type=Loss Diagram_type=Interchange Diagram_type=Coordinates Diagram_type>Loading %	DEFAULT; determined by FLOW options Maximum values P and/or Q, set by FLOW options chosen Bubble diagram Plot all coord data, no powerflow data % current or % MVA rating
VALUES=Normal VALUES=Difference	DEFAULT Difference plot; case1 - case2
FLOW_detail=P_Sending_end FLOW_detail=Q_Sending_end FLOW_detail=P_Receiving FLOW_detail=Q_Receiving	DEFAULT DEFAULT
BUS_detail=Bus_name,Abbreviation BUS_detail=Bus_name,Name_from Powerflow BUS_detail=Bus_name,Powerflow_name (and kV) BUS_detail=Voltage,kV BUS_detail=Voltage,Per Unit BUS_detail=Angle BUS_detail=Generation BUS_detail=Shunt BUS_detail=Load BUS_detail=Total_flow of undrawn branches BUS_detail=Outages	DEFAULT DEFAULT DEFAULT DEFAULT DEFAULT Not yet implemented.
BRANCH_detail=Trans_taps BRANCH_detail=Compensation BRANCH_detail=Parallels,Combined BRANCH_detail=Parallels,Separate BRANCH_detail=Outages	DEFAULT Not yet implemented.

File Management

This option is used to 'include' a separate file of coordinate data into the current file that you are plotting. CFLOW reports can be incorporated into the diagram by this mechanism. Multiple files may be included, if desired. This option is not presently available from the GUI. That is, if in the GUI you load a coordinate file which includes "Option File" records, these will be ignored. However, there is a means of including a single auxiliary coordinate file in a GUI plot. Under the File pull down menu, select Plot Options, then User Comments. Include a comment which is the name of the file, preceded by an ampersand (&). See section 5.6 for more information.

The ABERDEEN INSET in Figure 5.8 was created with an "&" record.

The included file should *not* be a complete, independent coordinate file. For example, if it has a [ID COORD record, none of the file will be included. If it contains any options which have already been defined, these will produce warning messages and will have no effect. Options which are not defined in the main file can be defined in the included file; they will be applied to the entire diagram.

The included file *must* be terminated with a (*EOR) record, or its last line will be lost.

Table 5-5 File Management Option

Option	Description
File=filename	Coordinate data from the specified file will be inserted into the current file. (This feature has not yet been implemented in the GUI.)

5.3.3 PostScript Records

PostScript, >Define, Comment, and Draw records are processed in the order specified by the user. For example, a PostScript record can be used to change the font for comment records. A comment record may or may not define coordinates. If coordinates are not defined, the comment is printed below the previous comment record. A series of Draw records or a series of PostScript records is often required to accomplish a specific task.

PostScript commands, columns 3 - 82 of the PostScript records, are sent directly to the PostScript file. They have the complete versatility of the PostScript language (PSL). A typical use is to change fonts within a series of comment records or to add simple graphics to a diagram. These records could also be generated, inserted, or edited by a CFLOW program (see also Comment Records, below).

Table 5-6 PostScript Record Format

Column	Format	Description
1	A 1	P — identifies a PostScript record.
3-82	A 80	Any valid PostScript command including commands defined in pfmaster.post.

5.3.4 >Define Records

The define records associate solution values to variables that can be manipulated and printed within comment records. The *IPF Batch User's Guide* provides an in-depth discussion of this feature. See Table 5-7 for the format of >Define records.

Table 5-7 Define Record Format

Column	Format	Description
1-7	A 7	>DEFINE — identifies a define record.
1-90	A 90	See the <i>IPF Batch User's Guide</i> , /USER- ANALYSIS section, Symbol Definitions.

5.3.5 Comment Records

Comment records may display simple text or may be used in conjunction with >define records to display Powerflow values.

They may also be used in conjunction with CFLOW, to create mini-reports on the printed map. Your CFLOW program can edit a coordinate file directly, or create/edit an auxiliary file containing comment records, which is included in the main file(s) with an "OPTION = FILE" record (see also PostScript records, above).

Table 5-8 Comment Record Format

Column	Format	Description
1	A 1	C — identifies a comment record.

Table 5-8 Comment Record Format

Column	Format	Description
3-14	2F 6.2	X, Y coordinates of comment. DEFAULT — after previous comment.
15-90	A 76	Comments.

5.3.6 Draw Records

Draw records are used to draw straight lines such as borders and boxes on a diagram.

Table 5-9 Draw Record Format

Column	Format	Description
1	A 1	D — identifies a draw record.
3-14	2F 6.2	X, Y coordinates.
15	I 1	1 = draw or 2 = move to specified coordinates.
16-27	2F 6.2	X, Y coordinates.
28	I 1	1 = draw or 2 = move to specified coordinates.
29-40	2F 6.2	X, Y coordinates.
41	I 1	1 = draw or 2 = move to specified coordinates.
42-53	2F 6.2	X, Y coordinates.
54	I 1	1 = draw or 2 = move to specified coordinates.
55-66	2F 6.2	X, Y coordinates.
67	I 1	1 = draw or 2 = move to specified coordinates.
68-79	2F 6.2	X, Y coordinates.
80	I 1	1 = draw or 2 = move to specified coordinates.

5.3.7 Bus Coordinate Data

The bus coordinate data describes where and how the Powerflow bus values will be displayed on the diagram. See Table 5-10 and Figure 5-3 for the format of the bus coordinate data record.

Table 5-10 Bus Coordinate Data Format

Column	Format	Description
1	A 1	B — Identifies the Bus coordinate record.
2	I 1	Display flag: 0 or Blank - Display the bus symbol. 1 - Do not display the bus symbol, but print the name.
3-10	A 8	Bus name to match Powerflow data.
11-14	F 4.0	Bus kv to match Powerflow data.
15-22	A 8	Name abbreviation to print on diagram. May be blank to kill bus name print.
23	I 1	Print bus voltage relative to bus name: 1 - above name 2 - right of name 3 - below name 4 - left of name 5 - do not print the voltage 6 - print the voltage, but no name
24-35	2F 6.2	X, Y of center of bus symbol.
36-47	2F 6.2	X, Y of the lower left corner of the bus name (if other than default).
48-50	F 3.0	Angle (in degrees) of generator symbol (0 degrees assigns X > 0, Y = 0 position Angle > 0 moves counter clockwise). An angle of 0 is a flag to not display the generator in the "Draw Generator as Needed" mode.
51-53	F 3.0	Angle of reactance symbol. See the preceding block for Angle characteristics
54-55	A 2	Bus symbol shape identifier: Blank - round symbol HB - horizontal bar, length = radius x 2 VB - Vertical bar User may add other symbols corresponding to symbols added in master PostScript file.
56-59	F 4.2	Bus symbol radius in centimeters.

[illegible]

Figure 5-3 Bus Coordinate Data Record

5.3.8 Branch Coordinate Data

The branch coordinate data describes the bending points in a branch and identifies which segment will show the flow and transformer symbol or compensation symbol. See Table 5-11 and Figure 5-4 for the format of the branch coordinate data record.

Column 27 requires additional explanation. Several alternative routes may be established for printing parallel circuits separately. The most preferred path is 1, next 2, etc. When the option to display parallel circuits separately is on and there are as many or more routings as circuits, the circuits are shown separately.

Table 5-11 Branch Coordinate Data Format

Column	Format	Description
1	A 1	L or T identifies a Line or Transformer.
2		Not used.
3-10	A 8	Bus1 name.
11-14	F 4.0	Bus1 kV.
15-22	A 8	Bus2 name.

Table 5-11 Branch Coordinate Data Format (Continued)

Column	Format	Description
23-26	F 4.0	Bus2 kV.
27	I1	Preference number for routing parallel circuits separately.
28		Not used.
29-30	I 2	Segment for annotation with flow. A negative number means do not show arrow and flow.
31-42	2F 6.2	X, Y coordinates for 1st bending point.
43-54	2F 6.2	X, Y coordinates for 2nd bending point.
55-66	2F 6.2	X, Y coordinates for 3rd bending point.
67-78	2F 6.2	X, Y coordinates for 4th bending point.
79-90	2F 6.2	X, Y coordinates for 5th bending point.

L O R T	BUS NAME 1										BASE kV 1	BUS NAME 2										BASE kV 2	C K T #	A S S E G N M E N T	X-BEND 1	Y-BEND 1	X-BEND 2	Y-BEND 2	X-BEND 3	Y-BEND 3	X-BEND 4	Y-BEND 4	X-BEND 5	Y-BEND 5									
	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2			3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	
L																																											
T																																											

- ☐ = Required; blanks or zero are unacceptable
☐ = Optional; blanks or zeros are acceptable
☐ = Not Applicable

Figure 5-4 Branch Coordinate Data Record

5.3.9 Area Coordinate Data

The area record is used to define the location of an area interchange bubble. See Table 5-12 and Figure 5-5 for the format of the area coordinate data record.

Table 5-12 Area Coordinate Data Format

Column	Format	Description
1	A 1	A identifies an area.
2	I 1	Display flag: 0, blank — Display the area symbol. 1 — Do not display the area symbol, but print the name.
3-12	A 10	Area name — Name of area as shown in IPF.
13		Not used.
14-23	A 10	Abbreviation — A name appearing on the diagram. If blank, no name is printed.
24-35	2F 6.2	X,Y of center of area symbol.
36-37	A 2	Area symbol shape identifier. (The default — blank — is a cartouche. The user may add other symbols corresponding to symbols added in the master PostScript file.)
38-41	F 4.2	Radius — Radius of circular segments of cartouche.
42-45	F 4.2	Width — Width of linear segment of cartouche.
46-90		Not used.

[illegible]

- ☐ = Required; blanks or zero are unacceptable
☐ = Optional; blanks or zeros are acceptable
☐ = Not Applicable

Figure 5-5 Area Coordinate Data Record

5.3.10 Intertie Coordinate Data

The intertie record is used to locate the bending points of an intertie connecting two areas. This record becomes necessary when crowding of interties occurs. See Table 5-13 and Figure 5-6 for the format of the intertie coordinate data record.

Table 5-13 Intertie Coordinate Data Format

Column	Format	Description
1	A 1	I identifies an intertie record.
2		Not used.
3-12	A 10	Area 1 — Name of first area.
13		Not used.
14-23	A10	Area 2 — Name of second area.
24-28		Not used.
29-30	I 2	Segment for annotation with flow. A negative number means do not show arrow and flow.
31-42	2F 6.2	X, Y coordinates for 1st bending point.
43-54	2F 6.2	X, Y coordinates for 2nd bending point.
55-66	2F 6.2	X, Y coordinates for 3rd bending point.
67-78	2F 6.2	X, Y coordinates for 4th bending point.
79-90	2F 6.2	X, Y coordinates for 5th bending point.

I	AREA NAME 1										AREA NAME 2										AS RE G O M E N T	X-BEND 1	Y-BEND 1	X-BEND 2	Y-BEND 2	X-BEND 3	Y-BEND 3	X-BEND 4	Y-BEND 4	X-BEND 5	Y-BEND 5																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0		1	2	3	4	5	6	7	8	9	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					

Figure 5-6 Intertie Coordinate Data Record

5.3.11 Annotation Record

The annotation record can be used to document a data file. The record is ignored by the diagram program. You can also temporarily remove a coordinate record from service by inserting a “!” or “#” before it.

Table 5-14 Annotation Record Format

Column	Format	Description
1	A 1	Exclamation point (!) or pound sign (#) identify an annotation record.
2-90	A	Any ASCII printing characters.

5.3.12 Trailer Record

The trailer record signifies the end of the usable data. Any records beyond the trailer record are ignored by the program. See Table 5-15 for the format of the trailer record.

Table 5-15 Trailer Record Format

Column	Format	Description
1	A 1	9 -or- (*EOR) identify a trailer record.

5.4 POSTSCRIPT PROCEDURES

This section is for the advanced user who wants to customize the appearance of the diagram by modifying PostScript procedures.

The PostScript procedures are stored in ASCII form on the `pfmaster.post` file. The procedures that are most likely to be changed by the user are described below.

5.4.1 Coordinate Data Within the Postscript Procedures File

The PostScript procedures file is used as a data repository for selected coordinate data. A % (percent sign) as the first character of a record instructs the PostScript interpreter to process the record as a comment. The FORTRAN program reads columns 2-79 of these comment records as coordinate data records. These records must be within the first 100 records of the `pfmaster.post` file and contain the following delimiters:

```
%[xx Beginning of a block 'xx' of coordinate format data
%(end) End of block
%[End End of all coordinate data in pfmaster.post repository
```

5.4.2 Diagram Identification Data

Diagram identification data is retrieved when the user requests the label box (BX) option. Typically, the identification includes:

- Powerflow case name.
- Date of Powerflow run.
- Version of Powerflow program used to create case.
- Description of Powerflow case.
- Time of Powerflow run.

The identification data in the `pfmaster.post` file is of the form shown below. Refer to the `pfmaster.post` file for the identification data in your installation.

```
%%Begin data read by psplot
%[BX WSCC will change the first 7 characters to: %[BPABX
%>DEFINE_TYPE OLDBASE LET cAse=CASE
%>DEFINE_TYPE OLDBASE LET dAte=DATE
%>DEFINE_TYPE OLDBASE LET pFvr=PFVER
%>DEFINE_TYPE OLDBASE LET dEsc=DESC
%P /Helvetica-Bold findfont 10 scalefont setfont
.
.
.
```

```
%Option SSpecific Comments=20
%(end)
%[End BPA data block read by psplot --DO NOT REMOVE THIS RECORD

%[WSCCBX WSCC will change the first 4 characters of this record to: %[BX
%! ***** emulated stuff *****
%>DEFINE_TYPE OLDBASE LET tIme=TIME
%P /Helvetica findfont 07 scalefont setfont
.
.
.
%Option SSpecific Comments=03
%(end)

%[End WSCC data block read by psplot --DO NOT REMOVE THIS RECORD
```

5.4.3 Legend

The legend is in PostScript form and is used when the legend option (LG) is selected. The legend identifies line patterns with kV ranges. The user may examine the procedure acLine for a description of the operands passed to that operator.

The legend data in the pfmaster.post file is of the form shown below.

```
/legndId
{
  gsave
  x-legnd y-legnd translate
  0 -4.50 cmtr translate
  newpath
  0 0 moveto 0 4.50 cmtr lineto 2.2 cmtr 4.50 cmtr lineto 2.2 cmtr 0 cmtr lineto
  o
  closepath stroke
  /Helvetica-Bold findfont 10 scalefont setfont
  0.4 cmtr 4.00 cmtr moveto
  (LEGEND) show
  /Helvetica-Bold findfont 08 scalefont setfont
  [ 0.10 cmtr 3.25 cmtr 2.10 cmtr 3.25 cmtr ]
  ( ) ( ) ( ) 1 0 ( 0 - <100 kv) ( ) ( ) 0 099 acLine
  [ 0.10 cmtr 2.50 cmtr 2.10 cmtr 2.50 cmtr ]
  ( ) ( ) ( ) 1 0 (100 - <200 kv) ( ) ( ) 0 199 acLine
  [ 0.10 cmtr 1.75 cmtr 2.10 cmtr 1.75 cmtr ]
  ( ) ( ) ( ) 1 0 (200 - <231 kv) ( ) ( ) 0 230 acLine
  [ 0.10 cmtr 1.00 cmtr 2.10 cmtr 1.00 cmtr ]
  ( ) ( ) ( ) 1 0 (231 - <500 kv) ( ) ( ) 0 499 acLine
  [ 0.10 cmtr 0.25 cmtr 2.10 cmtr 0.25 cmtr ]
  ( ) ( ) ( ) 1 0 (500 kv and up) ( ) ( ) 0 500 acLine
  grestore
```



```
} def
```

5.4.4 Line Pattern Data

Line pattern data is stored in the array `LinePattern`. The length must be defined in `LinePatternArray`. The `put` operator stores data in the array element indicated by the indexes 0, 1, 2, 3, etc.

The first two values in the array are the voltage range for the line pattern. Next is an array with a nested array. The nested array is line pattern, and the other value in the offset as required by the operator `setdash`. The last value in `LinePattern` is the line width.

To be within a voltage range, a voltage must be greater than or equal to the minimum voltage specified and less than the maximum voltage specified.

The line patterns for outages and interchange lines are similar but without the voltage range specification.

The line pattern data in the `pfmaster.post` file is:

```
%***** LINE PATTERN DATA*****

/LinePatternArray 5 def
/LinePattern LinePatternArray array def
/LinePatternCount LinePatternArray 1 sub def

LinePattern 0 [200 231 [ [3 6] 0 ] 1.00 ] put
LinePattern 1 [500 2000 [ [1 1000] 0 ] 1.00 ] put
LinePattern 2 [100 200 [ [5 1] 0 ] 1.00 ] put
LinePattern 3 [231 500 [ [1 5] 0 ] 1.00 ] put
LinePattern 4 [0 99999 [ [1000 1] 0 ] 1.00 ] put

%***** Line Pattern for Outage Line *****

/LinePatternOut [ [ [1 4] 0 ] 1.00 ] def
/curve-offset 8 def

%***** Line Pattern for Interchange Line *****

/LinePatternInt [ [ [1 10000] 0 ] 1.00 ] def
```

5.4.5 Bus Overvoltage/Undervoltage Range Values

Acceptable bus voltage range data is stored in the array `VoltLim`. The length must be defined in `VoltLimArray`. The `put` operator stores data in the array element indicated by the indexes 0, 1, 2, 3, etc.

The first two values in each element of the array are a voltage range. The second two values are the minimum and maximum acceptable per unit voltages for buses in that voltage range.

To be within a voltage range, a voltage must be greater than or equal to the minimum voltage specified and less than the maximum voltage specified.

```
%***** BUS OVER-VOLTAGE/UNDER-VOLTAGE DATA *****

/VoltLimArray 5 def
/VoltLim VoltLimArray array def
/VoltLimCount VoltLimArray 1 sub def
VoltLim 0 [230 500 0.95 1.052] put
VoltLim 1 [500 2000 1.00 1.100] put
VoltLim 2 [115 230 0.95 1.052] put
VoltLim 3 [ 0 115 0.95 1.052] put
VoltLim 4 [ 0 9999 1.00 1.000] put
```

5.4.6 Bus Symbols

The user may add new bus shapes or change the existing ones. The default bus shape is a circle. A vertical bar (VB) and horizontal bar (HB) are also available. See Table 5-10 and Figure 5-3. Bus symbols added in the `pfmaster.post` file can then be associated with bus records through the bus shape and scale factor fields on the bus coordinate record.

A typical bus symbol definition in the `pfmaster.post` file is the vertical bar bus symbol:

```
shapeIndex (VB) eq % begin drawing vertical bar
{
  [] 0 setdash
  obj-line-width setlinewidth
  newpath % clear current point so it doesn't show
  -.075 cmtr -1 cmtr bsScale mul moveto
  -.075 cmtr 1 cmtr bsScale mul lineto
  .075 cmtr 1 cmtr bsScale mul lineto
  .075 cmtr -1 cmtr bsScale mul lineto
  closepath
  gsave
  buscolor aload pop setrgbcolor % set background color
  fill
  grestore
  fgcolor aload pop setrgbcolor % set foreground color
  exit
} if % done drawing vertical bar
```

5.4.7 Area Symbols and Bubble Plots

The user may add new area shapes or change the existing one. The default area shape is a cartou-

che (round-cornered box). See Table 5-12 and Figure 5-5. Area symbols added in the `pmaster.post` file can then be associated with area records through the area shape and scale factor fields on the area coordinate record.

The default area symbol definition in the `pmaster.post` file is the cartouche or bubble:

%----- The enclosed code may be customized at the user's pleasure -----

```

bubFlg 0 ne                                % do not draw cartouche
{ exit } if

shapeIndex ( ) eq                          % begin drawing cartouche
{
bbSz1 0 eq {/bbSz1 0.80 cmtr def} if      % setup default size for bubble
bbSz2 0 eq {/bbSz2 1.50 cmtr def} if      % setup default size for bubble
[] 0 setdash
bub-wall-width setlinewidth
newpath                                    % clear current point so it doesn't show
/bbSz3 bbSz2 2 div def
bbSz3 neg 0 bbSz1 90 270 arc
bbSz3 bbSz1 neg lineto
bbSz3      0 bbSz1 270 90 arc
closepath
gsave
bgcolor aload pop setrgbcolor            % set background color
fill
grestore
fgcolor aload pop setrgbcolor            % set foreground color

/xTxt xTxt bbSz3 sub                      % calculate new x text coordinates
bbSz1 3 div sub def
/yTxt yTxt 0.5 cmtr add def              % calculate new y text coordinates

exit
} if                                      % done drawing cartouche

```

%----- The enclosed code may be customized at the user's pleasure -----

5.5 DIAGRAM COMPONENTS

Example diagrams are shown in 7Figure 5-8 and 5-8.

The hard copy diagram allows buses, branches, area, and interchange symbols to be intermixed on a diagram. Typically, however, bus/branch and area interchange diagram are drawn separately.

The diagram consists of two types of components:

- Supportive components such as borders, diagram identification, legend, and comments.
- Primary diagram components such as buses, branches, areas, and interchange flows.

5.5.1 Supportive Diagram Components

The example diagram in figure 5-8 illustrates the default options assigned when the user selects the identification box option.

A border is drawn as close to the edge of the paper as printers allow.

The information below the border identifies the type of diagram, the date and time the diagram was created, the Powerflow program version that created the diagram, the time that the Powerflow case was created, and the name of the coordinate file.

The case name, the date the case was created, and the Powerflow program version that created the case is shown on the first line within the box. A 20 character description of the case is on the second line. Comments entered by the user at the time the diagram was created follow. Note that the last user-entered comment on the bus/branch diagram begins with an ampersand (&). This comment instructs the program to read additional coordinate data from the file `aberdeeninset.cor`. Additional comments that had been entered with Powerflow data would follow these comments.

Blocks of text such as `LOSSES` on the bus/branch diagram and `INTERTIE SCHEDULED ACTUAL` on the area interchange diagram are the product of `>DEFINE` and `C` records in the coordinate file.

The `LG` (LeGend) option selects and locates the legend shown on the bus/branch diagram.

5.5.2 Primary Diagram Components: Bus/Branch Diagrams

The majority of the options previously discussed in the *Options Record* section refer to bus/branch diagrams. These options, along with the coordinate file, customize the diagram to the user's specifications. In general, the diagram consists of bus and branch symbol groups and Powerflow solution values. Area and intertie information may appear on the same diagram, but there will be no connection between the two graphs, as is shown below.

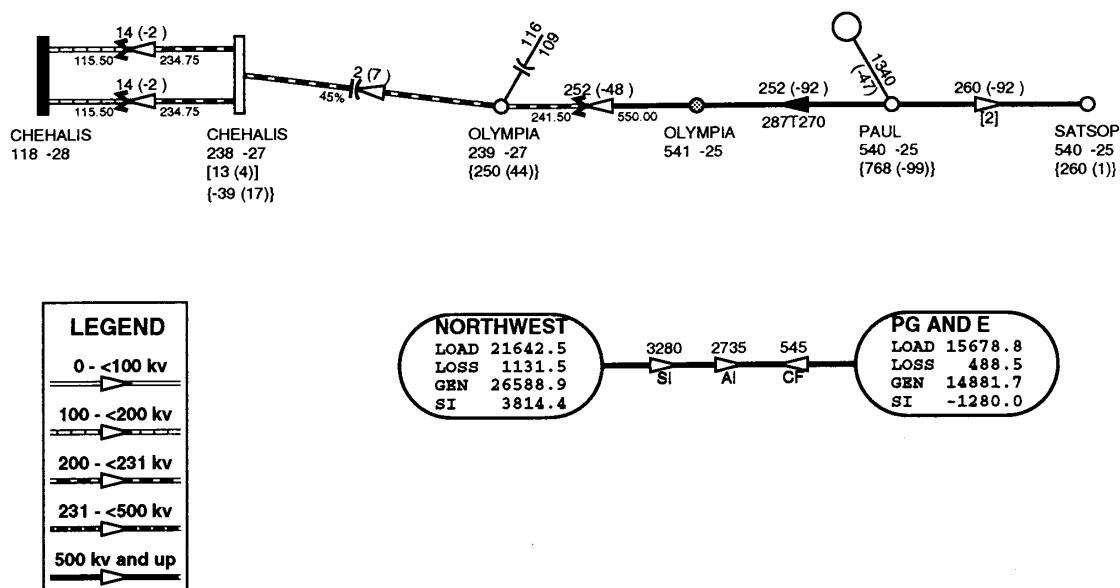


Figure 5-7 Diagram Example

The bus symbol group and values consists of the following:

- Bus symbol — Circle, bar, etc.

In Figure 5-7, the bus symbol is fully-shaded (black) — see CHEHALIS 115 — for undervoltage buses and half-shaded (gray) — see OLYMPIA 500 — for overvoltage buses.

- Identification — Powerflow name and kV or abbreviation.

Figure 5-7 shows buses with the abbreviated form of identification. The CHEHALIS 230 bus is abbreviated CHEHALIS.

- Voltage and angle — Printed above, below, or to the left or right of identification.

In Figure 5-7, the CHEHALIS 230 bus voltage is 238 kV and angle -27 degrees.

- Bus load — Printed above, below, or to the left or right of voltage and angle.

In Figure 5-7, the load at CHEHALIS 230 is 13 MW and 4 Mvar.

- Total flow on branches to buses not shown on diagram — printed above, below, or to the

left or right of load.

In Figure 5-7, the net flow at the CHEHALIS 230 bus on branches that are not shown on the diagram is 39 MW into the bus and 17 Mvar out of the bus.

- Generator symbol — Circle connected to bus with a short line segment.

Power generation in MW is printed above the line and reactive generation is printed below the line.

Figure 5-7 shows the generation at PAUL 500 is 1340 MW and -47 Mvar. A close inspection of the data would show the actual generation is on a low voltage bus at the same location as PAUL 500.

- Shunt Reactive — Capacitor or inductor symbol connected to a bus with a short line segment.

Maximum reactance available at the bus is printed above the line, and actual reactance used is printed below the line.

Figure 5-7 shows that 116 Mvar of capacitive reactance is available and 109 Mvar is used at OLYMPIA 230.

The branch symbol group and values consist of the following:

- Line segment symbol — Voltage-coded line segment representing a line or transformer connection between buses. Note the relationship to the legend.
- Arrow symbol — Indicates direction of real power flow through the branch.

In Figure 5-7, a solid-shaded arrow (black) indicates the branch is loaded at 100% or more of a rated capacity.

- Overload indicator — Flags circuits that are approaching an overload condition.

Branches that are approaching a nominal, thermal, bottleneck, or emergency loading are flagged with an N, T, B, or E under the arrow. The actual current or Mva flow in the line or transformer is before the numeric flag and the rating follows the flag.

Figure 5-7 shows the PAUL 500 to OLYMPIA 500 line carrying 287 amps and is over 90% of the line's thermal rating of 270 amps. The solid arrow indicates the line is over 100% of the rating. (This is contrived data, of course.)

- Branch flow— Real and reactive power (MW and Mvar).

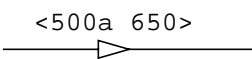
In Figure 5-7, branch flow is shown above the arrow. Real and reactive sending and receiving end flows are differentiated as follows. MW values are simply shown as numbers. Mvar values are shown within parentheses. Receiving end values are shown within square

brackets. Negative values indicate a flow opposite the arrow direction.

The power flowing from PAUL 500 to OLYMPIA 500 is 252 MW and -92 Mvar as measured at the sending end.

- Branch flow — Mva and current.

An alternative to showing MW and Mvar is to show MVA for transformers and current for lines. The value shown is the maximum for any section of the circuit. If the maximum is at a point other than the sending end, an R is appended to the flow. If a circuit is composed of both line sections and transformer sections, the maximum current for the line section and maximum MVA for the transformer section will be shown. The receiving or sending/receiving end flag is eliminated.

Example: 

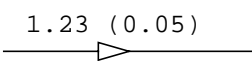
This example indicates 500 amps in a line section and 650 MVA in a transformer.

Example: 

This example indicates a flow of 1000 amps in the line.

- Branch flow — Real and reactive power loss.

Real and reactive losses in MW and Mvar are shown above the arrow.

Example: 

This example indicates a loss of 1.23 MW and 0.05 Mvar in the branch.

- Parallel circuits — Show number of parallel circuits represented by branch (default).

The number of parallel circuits carrying the flow on the diagram is shown in brackets below the line.

Figure 5-7 shows two parallel circuits between PAUL and SATSOP carrying a total of 260 MW and -92 Mvar.

- Parallel circuits — Show flow on each circuit represented (option).

The flow on each of the parallel circuits can be shown separately.

Figure 5-7 shows two parallel circuits between CHEHALIS 230 and CHEHALIS 115. Each circuit is carrying 14 MW and -2 Mvar.

- Transformer symbol (shown at tip of arrow).
- Transformer taps (shown below transformer).

Figure 5-7 shows the CHEHALIS 115/230 transformers with taps of 115.50 and 234.75.

- Series compensation symbol (shown at tip of arrow).
- Series compensation value — Percent of line compensation shown below capacitor symbol.

Figure 5-7 shows the CHEHALIS 230 to OLYMPIA 230 line with 45% compensation.

5.5.3 Primary Diagram Components: Interchange Diagram

The area interchange diagrams are quite simple, showing the area data within the area symbol (bubble), and flow data above the lines connecting areas.

The area symbol and data consists of the following:

- Area symbol — Cartouche (bubble).
- LOAD — Summation of all loads within the specified area.
- LOSS — Summation of all losses within the specified area.
- GEN — Summation of all generation within the specified area.
- SI (Scheduled Interchange) — Export of power from the specified area.

The interchange symbol and data consists of the following:

- Interchange symbol — Line with arrows and values for Scheduled Interchange, Actual Interchange, and Circulating Flow.

The values are shown above the arrows.

The example shows that 3280 MW was scheduled from NORTHWEST to PG AND E. The actual interchange was 2735 MW. The circulating flow, defined as actual interchange minus scheduled interchange, is 545 MW from PG AND E to NORTHWEST.

5.5.4 Primary Diagram Components: Difference Diagram

The format of difference diagrams is very similar to the format of the standard diagram. The standard diagram uses only one case, an active case. The difference diagram uses two cases, an active case and an alternate case.

Displayed values are calculated as active case values minus alternate case values. Arrows indicate the direction of power flow in the active case.

If there are a different number of circuits in the active and reference cases, the number of circuits is in each case shown below the circuit.

Example: 

This example shows the active case has 3 circuits and the alternate case has 2 circuits in parallel.

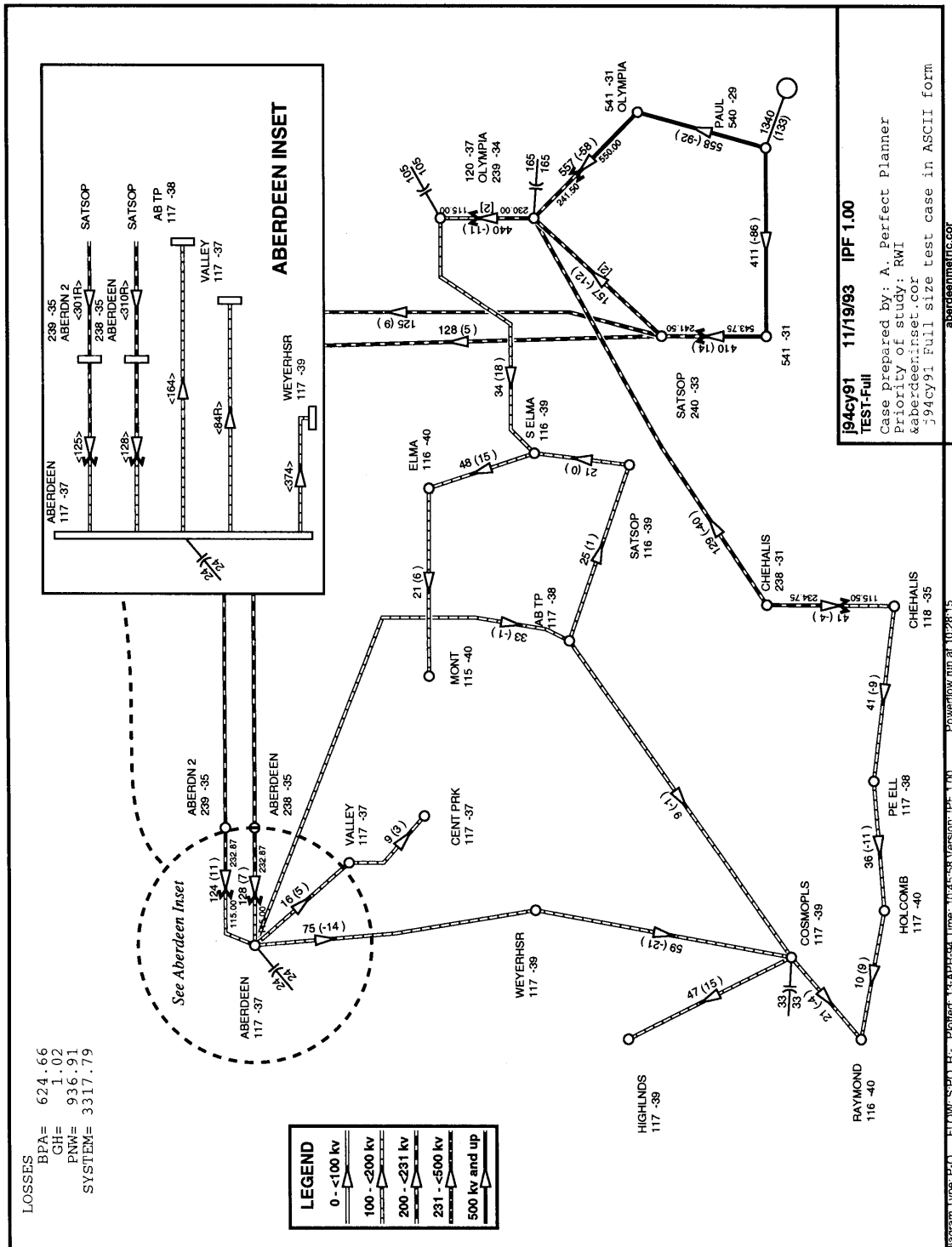


Figure 5-8 Bus/Branch Diagram with Inset

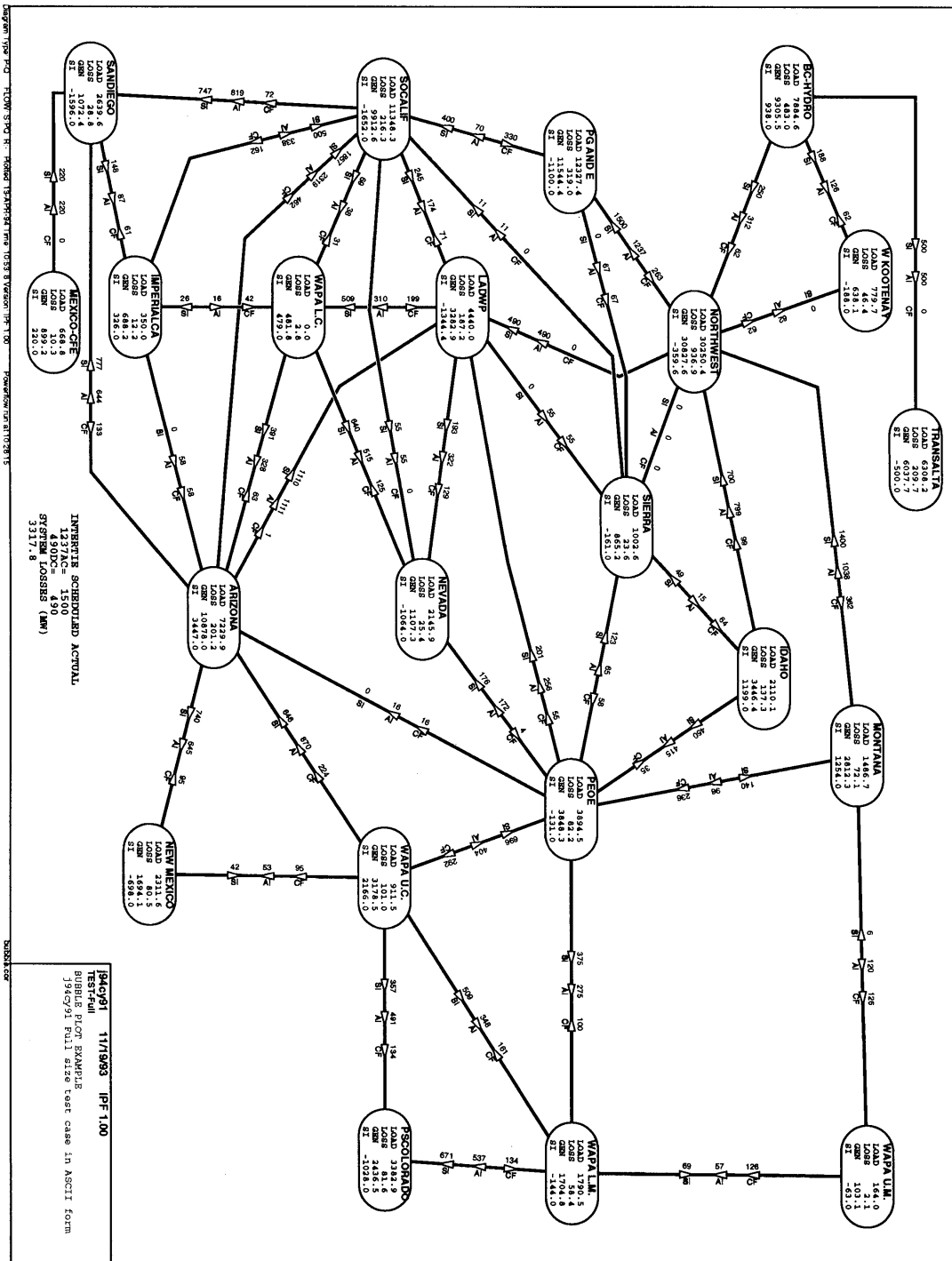


Figure 5-9 Area Interchange Diagram

5.6 BATCH MODE PLOTTING WITH IPFBAT

Batch mode plotting can be used when a coordinate file already exists, and the user simply wants a hard copy diagram based on that file and Powerflow data. If the Powerflow data is on a saved base case (*.bse) file, the simplest method is to use the IPFPLOT program. However, ipfbat offers more flexibility and control. For example, with ipfbat you can load, solve, and plot a netdata file.

This technique can be used to produce diagrams that are generally produced through the GUI or for access to features that have not yet been implemented in the GUI. These features include plotting bubble diagrams, plotting difference diagrams, and plotting diagrams from a master list of coordinate files.

An example of batch mode plotting is accomplished through the IPFBAT program as follows:

```
ipfbat bubble.pcl
```

where the .pcl file is a control file with the IPF commands and data necessary to produce a hard copy diagram.

Commands in the examples are record groups starting with a / (slash) command and ending with the next / (slash) command or (end) for the last command in the file.

Under the command /plot, the first line must name the coordinate file to be used, and the second must name the output PostScript file to be produced. Any subsequent records following, before the next /command, are interpreted as comments, and will be placed in the standard position following the last comment defined in the coordinate file.

Two special uses for comment records must be noted. If the record begins with an ampersand (&), it will be interpreted as an instruction to append the auxiliary coordinate file named on the record. At most one such file may be named. If the record begins with an 'at' symbol (@), it will be interpreted as an option record. Any diagram option indicated on this type of record will override the option specified in the coordinate file. Multiple @ records are allowed and will not be printed on the diagram.

5.6.1 Example 1

Make a "standard" diagram (similar to the GUI operation).

```
/network_data,file=a92cy91.dat      ! Load the powerflow network data
/solution                          ! Solve the powerflow case
/plot                              ! Make a hard copy diagram
aberdeenmetric.cor                ! using this coordinate file
diagram.ps                        ! to build this postscript file.
Case prepared by: A. Perfect Planner ! Include this comment
```

```

Priority of study: RWI          ! and this comment
&aberninset.cor               ! and this additional coordinate file.
@OPTION DIagram_type=Pq_flow   ! Supplement/Override *.cor options.
/syscal                       ! Hello operating system ...
lpr diagram.ps                ! ... send this file to the printer.
/exit                         ! This job is finished.
(end)

```

5.6.2 Example 2

Make a bubble diagram.

```

/old_base,file=j94cy91.bse     ! Load the powerflow saved base case
/plot                         ! Make a hard copy diagram
bubble.cor                    ! using this coordinate file
diagram.ps                    ! to build this postscript file.
BUBBLE PLOT EXAMPLE           ! Include this comment.
/syscal                       ! Hello operating system ...
lpr diagram.ps                ! ... send this file to the printer.
/exit                         ! This job is finished.
(end)

```

5.6.3 Example 3

Make a difference diagram.

```

/old_base,file=9_bus_test.bse ! Load the powerflow saved base case
/get_data,type=load_ref_base,file=bus_alt1.bse ! Load a reference
                                                ! saved base case
/ get_data, type = load_ref_area ! load reference solution data in tables
/plot                         ! Make a hard copy diagram
9bus_metricdif.cor           ! using this coordinate file
diagram.ps                    ! to build this postscript file.
Case prepared by: A. Perfect Planner ! Include this comment
Priority of study: RWI        ! and this comment
Difference plot between two cases ! and this comment.
/syscal                       ! Hello operating system ...
lpr diagram.ps                ! ... send this file to the printer.
/exit                         ! This job is finished.
(end)

```

5.6.4 Example 4

Make a series of diagrams from a list of coordinate files.

```

/old_base,file=/shr5/j96cy89.bse ! Load the powerflow saved base case
/plot                         ! Make a hard copy diagram

```

```
master.cor                                ! using all the coordinate files
                                           ! listed in this file
diagram.ps                               ! to build this postscript file.
Case prepared by: A. Perfect Planner      ! Include this comment
Priority of study: RWI                    ! and this comment on each diagram.
/syscal                                  ! Hello operating system ...
lpr diagram.ps                           ! ... send this file to the printer.
/exit                                    ! This job is finished.
(end)
```

5.6.5 Example 5

Here is an example of a master coordinate file (`master.cor`).

```
master
/home/dave/cor/3rdac.cor
/home/dave/cor/500bus.cor
/home/dave/cor/bubble.cor
/home/dave/cor/sworegon.cor
/home/dave/cor/nwmont.cor
```


CHAPTER 6

MISCELLANEOUS TOPICS

6.1 DIRECTORIES

6.1.1 Data and Execution Directories

Following is a listing of the delivery directory structure. This structure will be built upon installation beneath home/ or usr/, or some similar standard user area on your system.

Contents of user delivery package - Version 9 Programs.

Directories - Files	Description
ipf -	
bin -	
INFO.bin	Information file.
gui	Executable
gui.uid	Executable
ipfmain	Executable (contains ipfsrv,bpf,ipfcut, ipfnet,ipfplot,ipfbat)
ipf_diff	Executable (gui editing utility)
ips2ipf	Executable (data conversion program)
pfmaster.post	PostScript header file
XGUI	User X-resource file
ipf_gui_help.mif	Help file
ipf_setup	Environment variable setup file
editbus	External editor setup script
dat -	
INFO.dat	Information file.
gui	ln -s ../bin/gui
gui.uid	ln -s ../bin/gui.uid
ipfsrv	ln -s ../bin/ipfmain
bpf	ln -s ../bin/ipfmain
ipfplot	ln -s ../bin/ipfmain
ipfbat	ln -s ../bin/ipfmain
ipfcut	ln -s ../bin/ipfmain
ipfnet	ln -s ../bin/ipfmain
ips2ipf	ln -s ../bin/ips2ipf
ipf_diff	ln -s ../bin/ipf_diff
pfmaster.post	ln -s ../bin/pfmaster.post
ipf_gui_help.mif	ln -s ../bin/ipf_gui_help.mif
43bus.net	43-bus test case

43bus.cor	Coordinate file for 43-bus
0102hw1.dat	IPS powerflow data, full-loop case
0102hw1.net	IPF powerflow data, converted from above
ips2ipf.log	Log file from data conversion
0102hw1.pfc	IPF batch (bpf) command file
hvmap.cor	Coordinate file for printed HV system
hvscr.cor	Coordinate file for screen view HV system
WSCCland.cor	Base coordinate file, landscape WSCC form
WSCCport.cor	Base coordinate file, portrait WSCC form
l85x11.cor	Base coordinate file, landscape BPA form
p85x11.cor	Base coordinate file, portrait BPA form
prtcc	Script to print output files w/carr. control
CARRIAGE.PS	PostScript header used by prtcc
lib -	
INFO.lib	Information file on this directory.
ipsasif.info	Information file on IPSASIF program.
*.f	Source files for IPSASIF program
*.h	CFLOW header files
convtbl.c	CFLOW source
pf_rec.c	CFLOW source
cflow_lib_ipc.c	CFLOW source
ft.c	CFLOW source
read.c	CFLOW source
bldcflowlu.m	make file for CFLOW
cflow_progs.m	make file for sample programs
libipcnew.a	IPC library (needed to link CFLOW programs)
libcflownew.a	CFLOW library (needed to link CFLOW programs)
...	CFLOW sample programs and data files

6.1.2 Setting Paths

If you want to be able to execute the IPF programs from any directory in your login, then add the `ipf/dat` directory to your default PATH, in your `.login`, `.profile`, `.kshrc`, or other similar file, where your PATH is currently defined.

Note that the `ipf/dat` directory contains many symbolic links to `ipfmain`, with the names of the various IPF programs. These are required in order to access the programs. If you create your own links, *they must have the same names*, since these names are passed to `ipfmain` as command line argument `argv[0]`. You may want to create such links in a directory you already have on your PATH, rather than adding a new one to it. If so, copy those in `ipf/dat`, using absolute rather than relative pathnames.

6.1.3 Using ipf_setup

A few environment variables are recognized by IPF; you can use these to help find important files, if you don't want (or need) to go so far as to put `ipf/dat` on your standard PATH. The `ipf_setup`

file is in the `ipf/bin` directory. Below is a printout, showing a sample assignment. This text would have to be added to your `.login` file in order to have any effect.

```
# filename: ipf_setup
# purpose : ipf application setup file incorporated into .login file
#
# path to locate the XGUI file (overrides default in home directory)
# setenv XAPPLRESDIR /shrunis/ipf/exe
# path to locate the gui.uid file for application
setenv UIDPATH ./%U:/shrunis/ipf/exe/%U:/usr/lib/X11/uid/%U
# path to locate help file
setenv IPFDIRS "../dat/./shrunis/ipf/dat/"
```

6.2 GUI EDITORS

The Edit Network Data dialog allows you to directly edit the currently loaded case data. This facility might be useful if you are wanting to do the same thing, or same sort of thing, to a large number of records, where opening the individual dialogs for each bus and/or branch would be extremely tedious. However, keep in mind that you can always edit your netdata file directly, outside of IPF, and then reload it.

There are buttons in the Edit Network Data dialog for Internal Editor, External Editor, and Send to PF. The editor choices are described below. After editing the data, you must click the Send to PF button in order to actually generate the change (M) records that alter the loaded system. IPF compares your edited listing to the original listing it retrieved (`editbus.dat`) and puts the changed lines in `editbusc.dat`.

6.2.1 Internal Editor

The internal editor is coded within IPF. It is convenient, but pretty powerless, as editors go.

1. **Entering and Altering Data** - Click in the display window to place an "I-beam" cursor. When you type a character, it will appear to the *right* of the cursor, just as in normal X Windows text entry. However, note that it will replace the current character to the right of the cursor, instead of moving it over. The internal editor is always in overwrite mode, so that you will not disturb the column alignment by changing values. There is no insert mode.
2. **Arrow Keys** - You can use the arrow keys to move around in the display window, and to scroll the display.
3. **Delete and BackSpace** - The action of these keys is controlled by the settings in your `.Xdefaults` file. "Normal" would be that the BackSpace key just moves the cursor back one position, just like the left arrow key, and Delete removed text which has been painted (reverse video) with the left mouse button. The following definitions cause BackSpace to delete the prior character, pulling the right end of the line in after it, and Delete to delete the next character, pulling the rest of the line up (in the absence of any painted text).

```
delCharBakKey: <BackSpace>  
delCharFwdKey: <Delete>
```

4. Insert Blank Character - The Tab key will insert a blank, shoving the rest of the line out to the right.
5. Place Anchor Point (^) - Ctrl-Tab places a point for subsequent operations.
6. Copy and Paste:

Place an anchor point where you want to paste characters in. If you want to paste a whole line, place the point at the beginning of the line you want the new line *above*. In this case, a new line will be created, and the existing text will be pushed to the next line down in the listing. Otherwise, the pasted text will overwrite existing text to the right of the anchor point.

Select the text you want to copy by painting with the *middle* mouse button. This produces an underline instead of reverse video. To select an entire line, press the middle button at the beginning of the line and drag vertically down just slightly. If you keep dragging down, you will get the next line too.

When you release the button, the underlined text will be pasted at the anchor point.

6.2.2 External Editor

In order to use an external editor, the editor you want to use must be set up in the script file `editbus`, in `ipf/bin`. The file currently defines `vi` as the external editor. The retrieved listing is copied to `editbusn.dat`, and a terminal window is opened for `vi`, with that file opened in it.

When you have finished your editing, you must exit and save, just as in a normal `vi` edit session, and then close the window. Again, you must click Send to PF in order to apply the edits.

Other editors can possibly be substituted for `vi`. However, it may be easier to do your editing outside of IPF than to get some other editor working.

A

- A data records 3 - 12
- annotation record 5 - 17
- area coordinate data 5 - 14
- area interchange 3 - 21, 3 - 22, 3 - 31, 3 - 44
- area interchange output 3 - 13
- area list 3 - 13
- area shapes 5 - 22
- area tie line flows 3 - 55
- arguments
 - command line 3 - 10

B

- base kV list 3 - 14
- batch mode plotting 5 - 32
- border 5 - 24
- borders 5 - 11
- boxes 5 - 11
- BPF 2 - 2, 2 - 4
- branch coordinate data 5 - 13
- branch input 3 - 46
- branch output 3 - 47
- bubble diagram 5 - 33
- bubble plots 5 - 14, 5 - 22
- bus coordinate data 5 - 11
- bus exists 3 - 15
- bus input 3 - 45, 3 - 46
- bus list 3 - 15
- bus output 3 - 47
- bus shapes 5 - 22
- bus voltage range 5 - 21
- bus voltages 3 - 16

C

- C language 1 - 1
- case comments 3 - 17
- case description 3 - 39, 3 - 42
- case information 3 - 40
- caseid 3 - 42
- CFLOW 2 - 4
- change records 3 - 10
- changes 3 - 50
- color 4 - 1, 4 - 4
- Command Dialog 3 - 1
- command line arguments 3 - 10
- comments 3 - 17, 3 - 42
- comparison 3 - 22
- conductor characteristics 3 - 20

coordinate file 3 - 10, 5 - 2, 5 - 3

COPE 2 - 4

count records 3 - 18

D

database engine 1 - 1

defaults 4 - 1

deleted data 3 - 51

diagram 5 - 1

diagram file 3 - 10

difference diagram 5 - 33

directory search path 3 - 10

documentation 1 - 2

E

existence of file 3 - 19

exit 3 - 6

external editor 6 - 4

F

fetch data 3 - 12

font 4 - 1, 4 - 4

G

GUI 2 - 2, 2 - 4

H

header 1 3 - 17, 3 - 42

headers 3 - 42

I

I data records 3 - 19

initialize 3 - 19

internal editor 6 - 3

interprocess communications channel 3 - 1

IPC channel 3 - 1

IPFBAT 2 - 2, 2 - 4

IPFCUT 2 - 3

IPFNET 2 - 3

IPFPLOT 2 - 2

IPFSRV 2 - 2

IPS2IPF 2 - 3

L

legend 5 - 24

legend, plot 5 - 20

line impedance 3 - 20

line loading 3 - 49

line pattern 5 - 21

M

map 5 - 1

master coordinate file 5 - 35

N

- network connection 3 - 17
- network data 3 - 20, 3 - 22
- network data file 3 - 8
- network data, output 3 - 23
- network data, reference 3 - 32

O

- outaged data 3 - 23
- output data 3 - 23
- over voltage 3 - 48
- overloaded lines 3 - 52
- overloaded transformer 3 - 53
- ownership 3 - 29

P

- path 3 - 10
- PCL 2 - 2
- PFC 1 - 1, 2 - 2
- pfmaster.post 5 - 2, 5 - 19
- phase shifter 3 - 54

R

- record type 3 - 30
- reference base 3 - 32
- reports 3 - 43

S

- save file 3 - 9
- search path 3 - 10
- socket number 3 - 10
- solution parameters 3 - 11, 3 - 38
- solve case 3 - 11
- solved base case 3 - 8
- standard input 3 - 10
- standard output 3 - 10
- start up 3 - 6

T

- terminal window 3 - 10
- tie line 3 - 55
- tower geometry 3 - 20
- trailer record 5 - 18

U

- under voltage 3 - 48
- UNIX system example 3 - 10
- user analysis 3 - 19, 3 - 21, 3 - 39

V

- VAX VMS system example 3 - 10
- voltage differences 3 - 56
- voltage range 5 - 21

W

windows 4 - 1, 4 - 3

X

X resources 4 - 1

XGUI file 4 - 1, 4 - 6

Z

zone list 3 - 40