



47467-H010-UX-00

REL BET 4.0
USER'S GUIDE

CR171973

23 DECEMBER 1986

PREPARED FOR

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

CONTRACT NAS9-17554

(NASA-CR-171973) RELEET 4.0 USER'S GUIDE
(TRW Defense Systems Group) 286 p CSDL 09B

N87-18333

Unclas
G3/61 43518

PREPARED BY

F. C. CERBINS
B. P. HUYSMAN
J. K. KNOEDLER
P. S. KWONG
L. A. PIENIAZEK
S. W. STROM

SYSTEM DEVELOPMENT DIVISION
DEFENSE SYSTEMS GROUP
HOUSTON, TEXAS



47467-H010-UX-00

RELBET 4.0
USER'S GUIDE

23 DECEMBER 1986

PREPARED FOR
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

CONTRACT NAS9-17554

PREPARED BY

F. C. CERBINS
B. P. HUYSMAN
J. K. KNOEDLER
P. S. KWONG
L. A. PIENIAZEK
S. W. STROM

SYSTEM DEVELOPMENT DIVISION
DEFENSE SYSTEMS GROUP
HOUSTON, TEXAS




47467-H010-UX-00

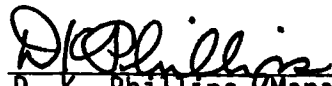
RELBET 4.0
USER'S GUIDE

December 1986

Prepared for
National Aeronautics and Space Administration
Lyndon B. Johnson Space Center
Houston, Texas

Contract NAS9-17554

Approved by 
L. A. Pieniazek, Head
Navigation Analysis Section

Approved by 
D. K. Phillips, Manager
Systems Engineering and
Analysis Department

System Development Division
TRW Defense Systems Group
Houston, Texas

TABLE OF CONTENTS

	Page
1.0 INTRODUCTION.....	1-1
1.1 IDENTIFICATION.....	1-2
1.2 REQUIRED BACKGROUND.....	1-3
1.3 HARDWARE/SOFTWARE ENVIRONMENT.....	1-4
1.4 TERMINOLOGY AND CONVENTIONS.....	1-4
2.0 REFERENCES.....	2-1
3.0 FUNCTIONAL DESCRIPTION.....	3-1
3.1 GENERAL OVERVIEW.....	3-1
3.2 CCT DATA FORMATTING.....	3-3
3.3 DATA PREPARATION.....	3-6
3.4 TRAJECTORY ESTIMATION.....	3-9
3.5 PRODUCTS GENERATION.....	3-11
4.0 PROGRAM INPUT AND EXECUTION.....	4-1
4.1 COMMAND LINE ARGUMENTS.....	4-1
4.2 MENU DRIVEN INPUT.....	4-1
4.3 LINPUT INPUT.....	4-2
4.3.1 <u>Simple Example</u>	4-2
4.3.2 <u>Guidelines</u>	4-4
4.3.3 <u>Not So Simple Example</u>	4-7
5.0 STEP-BY-STEP PROCEDURAL EXAMPLE.....	5-1
5.1 DATA COLLECTION.....	5-8
5.1.1 <u>Identify CCTs</u>	5-8
5.1.2 <u>Obtain L bin Copies of CCTs</u>	5-8
5.2 DATA FORMATTING.....	5-9
5.2.1 <u>Strip Data From CCTs</u>	5-9
5.2.2 <u>Combine Gff Files</u>	5-11
5.2.3 <u>Edit the OBS Per Onboard Data Good Flags</u>	5-12

TABLE OF CONTENTS (Continued)

	Page
5.2.4 <u>Convert Radar Angles</u>	5-12
5.3 DATA PREPARATION.....	5-12
5.3.1 <u>Check For Data Continuity and Completeness</u>	5-13
5.3.2 <u>Identify Significant Events</u>	5-13
5.3.3 <u>Evaluate Noise on Observation Data</u>	5-14
5.3.4 <u>Create Events Data Base</u>	5-16
5.3.5 <u>Evaluate The Recommended Edits</u>	5-16
5.3.6 <u>Create Time Intervals</u>	5-17
5.3.7 <u>Edit The OBS File</u>	5-17
5.4 BEST ESTIMATE OF TRAJECTORY GENERATION PROCEDURES.....	5-17
5.4.1 <u>Obtain Initial State For Filtering</u>	5-18
5.4.1 <u>Adjust The Linput Input Files to Reflect</u> <u>Mission Characteristics</u>	5-18
5.4.3 <u>Run Kalman Filter</u>	5-19
5.4.4 <u>QA Kalman Filter Output</u>	5-19
5.4.4.1 <u>Gross QA (Evaluate OUTSFILT)</u>	5-19
5.4.4.2 <u>Evaluate The Filter Updates</u>	5-20
5.4.4.3 <u>Evaluate The Filter Edits</u>	5-20
5.4.5 <u>Smooth The Kalman Estimate</u>	5-21
5.4.5.1 <u>QA Smoothed Estimate</u>	5-22
5.5 OUTPUT PRODUCTS GENERATION.....	5-23
5.5.1 <u>Prepare Linput Inputs For Prodx</u>	5-23
5.5.2 <u>Ancillary Products Processor</u>	5-24
5.5.3 <u>QA Output Product Tape</u>	5-24
5.5.4 <u>Generate Microfiche</u>	5-24
5.5.5 <u>Sensor Tape Output</u>	5-25
5.5.5.1 <u>Generate SIT and SET</u>	5-28
5.5.5.2 <u>QA Sit/SET Files</u>	5-28
APPENDIX I - PROGRAM MANUALS.....	A-1
APPENDIX II - INPUT BLOCKS.....	AII
APPENDIX III - ARTICLES.....	AIII

1.0 INTRODUCTION AND SCOPE

This manual describes the operation and use of RELBET 4.0 implemented on the Hewlett Packard model 9000. It serves both as a reference and as a training guide. Appendices provide experienced users with details and full explanations of program usage. The body of the manual introduces new users to the system by leading them through a step by step example of a "typical" production. Hopefully this will equip the new user both to execute a typical production process and to understand the most significant variables in that process. It is recommended that a new user read the body of this manual in the order presented and work through the step by step example in section 5. He should refer to the appendices, especially Appendix III, to further develop his needs.

This manual is divided into the following sections:

1. **INTRODUCTION:** Identifies the RELBET System and provides an overview of the organization and use of the manual. Tells you what to expect.
2. **REFERENCES:** Provides references to associated documentation. Tells you where to find additional information.
3. **FUNCTIONAL OVERVIEW:** Provides a basic functional overview of the operation of the RELBET System. Explains what programs are supposed to do.
4. **PROGRAM INPUT AND EXECUTION:** Provides an overview of inputs and options and describes the associated appendices which provide complete user information. Explains how to set up and run a program.
5. **STEP BY STEP PROCEDURAL EXAMPLE:** Provides a basic tutorial by running through an actual production process. Leads you through the details.

The main body is primarily tutorial and emphasizes nominal procedure. The appendices provide further details and additional information concerning the more complicated processors. The appendices are as follows:

- I. **PROGRAM MANUALS:** Contains details on how to execute each program.
- II. **INPUT BLOCKS:** Provides descriptions of the various input parameters accessed with **linput**.

III. **ARTICLES:** Presents a collection of articles about those subjects which merit more discussion than provided in the Program Manuals, Appendix I. These include:

- o The **linput** input language used by most processors
- o The **downlist** formatter program
- o The sequential Kalman filter processor
- o The graphics display process
- o The output products generation program, **prodx**
- o The noise analysis technique

1.1 IDENTIFICATION

The RELBET System is an integrated collection of computer programs that support the analysis and post-flight reconstruction of vehicle to vehicle relative trajectories of two on-orbit free-flying vehicles: the Space Shuttle Orbiter and some other free-flyer. The UNIVAC 1100 version of the system, RELBET 2.0, realizes the full production and analysis capability. The HP9000 version, RELBET 4.0, provides a basic post-flight data production capability and is a partial implementation of the full analysis version. RELBET 4.0 was created by carefully tailoring the RELBET analysis software to fit the production problem and reflects a streamlined production-oriented version that supports the post-flight reconstruction of relative trajectories and the generation of standard data products.

In particular the RELBET 4.0 System accepts Orbiter downlist telemetry input in the form of Computer Compatible Tapes (CCT's) and produces the following outputs:

- o RELBET Ancillary Data Product Tape
- o RELBET Ancillary Data Fiche Tape
- o SENSOR Program Input Data Tapes
- o SENSOR Program Environmental Data Tape
- o Tables for RELBET Report

It incorporates the following features not available in RELBET 2.0:

- o Organization of standard runstreams into shell scripts
- o Enhanced user input scheme via linput
- o Increased data QA programs
- o A text file QA data base

It lacks the following features available in RELBET 2.0:

- o Data and trajectory simulation capabilities
- o General purpose automatic editor
- o Least Squares Filter
- o Residual computation programs
- o Binary Data Base Editor
- o General interactive control of display processors
- o Miscellaneous display processors

The delivered version of RELBET 4.0 is available in magnetic tape media and consists of the following elements.

- o Source code (configured in SCCS format)
- o Relocatable subroutine code
- o Executable programs
- o Program creation directives (makefile's)
- o Program and subroutine documentation (for nroff formatting)

Supplementary tapes for the step-by-step example in Section 5.0 of this manual and sample products are also provided in addition to the software tape.

1.2 REQUIRED BACKGROUND

This manual assumes a basic familiarity with the UNIX system and the vi editor. In addition particular areas, say Kalman filtering, may require additional expertise. For information on these areas, consult the references in Section 2.0.

1.3 HARDWARE/SOFTWARE ENVIRONMENT

RELBET 4.0 assumes the following hardware configuration:

- o Hewlett Packard 9000 Computer model 540
- o Hewlett Packard 7935 440 Megabyte disk drives
- o Hewlett Packard 150 computers as terminals
- o Hewlett Packard 9872B plotter for plots
- o Hewlett Packard 7970 9 track 1600 bpi tape drive.

RELBET 4.0 assumes the following software environment:

- o RELBET 4.0
- o HPUX operating system version 5.1
- o DISSPLA Graphics Library.

1.4 TERMINOLOGY AND CONVENTIONS

Terms are generally used in the sense of UNIX. In the following, **program** refers to an executable object. When a distinction between an interpreted shell program and a compiled and linked program is needed, the terms **shell program** or **script** and **binary program** are used respectively. The term **processor** is also used to mean a program. The term **routine** means a C function or a FORTRAN routine. Routines correspond to source and relocatable code. They are separately compiled but not linked. Thus there is a distinction between driver **routines** and executable **programs** that results from linking. The terms **directory** and **file** have the same meaning as in UNIX. The term **module** is used in a descriptive sense to indicate a set of related routines. The term **package** is also used descriptively. It indicates a set of related routines, data structures, or even programs. The term **context** is used to indicate information related to variables or subroutines defined elsewhere, as well as sizing, format, and defined type information.

The following lexical conventions are used:

Program and File Names: **Bold font** denotes program and file names within text discussion.

Terminal Display: Terminal screen print examples are always indented and separated from the discussion. **Bold font** denotes user input and regular font denotes program output. Two lines consisting of a single indented colon indicate an omission of several lines of text. The elipsis (...) indicates a omission of text on a given line. For example

```
  who
  dick
  :
  :
  ziggy
```

indicates a response to a command: the user typed "who" and the program printed a list of names beginning with "dick" and ending with "ziggy". The intervening names have been omitted.

Flow charts use the following symbols:



Processor Activity



Binary File



Text File



Display, e.g., Printer Plot



User Knowledge

2.0 REFERENCES

The following documentation is applicable to the RELBET 4.0. and the RELBET System in general.

1. "Space Transportation System Post-flight On-orbit Relative Trajectory and Ancillary Data Products," TRW Report 39107-H011-UX-00, B. P. Huysman, L. A. Pieniazek, 1983.
2. "RELBET Product Description (Update/May 1985)", TRW Memo 85:W482.1-57, B. P. Huysman, L. A. Pieniazek, 15 May 1985.
3. "Programming Standards for FORTRAN Deliverables," TRW Memo 82:W582.1-16, D. K. Phillips, 10 February 1982.
4. "RELBET Software Level A Requirements," TRW Report 39107-H003-UX-00, L. A. Pieniazek, 18 May 1982.
5. "RELBET Software Level B Requirements," TRW Report 39107-H005-UX-00, B. P. Huysman, L. A. Pieniazek, 8 November 1982.
6. "Rendezvous BET Program, LRBET3," JSC-18638, W. M. Lear, December 1982.
7. "STAR Version 1.6 User's Guide," TRW Memo 83:W482.8-13, S. W. Strom, 3 May 1984.
8. "Preliminary RELBET User's Manual," TRW Memo 83:W482.8-24, L. A. Pieniazek, 21 July 1983.
9. "RELBET Engineering Manual," TRW Memo 84:W482.8-125, L. D. Erdman, 15 October 1984.
10. "RELBET Programmer's Manual," TRW Memo 84:482.8-130, L. Morris, 26 October 1984.
11. "RELBET User's Manual Update," TRW Memo 85:W482.8-24, L. A. Pieniazek, 13 March 1985.
12. "HP General File Format (GFF) User's Guide," TRW Report 39107-H021-UX-00, D. G. Campbell, 14 December 1984.
13. "RELBET User's Manual Change Pages," TRW Memo 85:W482.8-106, P. S. Kwong, 12 November 1985.
14. "RELBET Programmer's Manual Change Pages," TRW Memo 85:W482.8-107, P. S. Kwong, 12 November 1985.
15. "BOX System User's Guide," TRW Report 47467-H002-UX-00, W. Pace and D. Poritz, February 1986.

16. "Preliminary HP RELBET Streamlining Level A Description," TRW Memo 86:W482.8-19, L. A. Pieniazek, 21 May 1986.
17. "SENSOR Tape Production Manual," TRW Memo 85.W482.1-60, J. Knoedler, 3 June 1986.
18. "RELBET 4.0 User's Guide," TRW Report 47467-H010-UX-00, B. P. Huysman et al., December 1986.
19. "RELBET 4.0 Programmer's Manual," TRW Report 47467-H011-UX-00, P. S. Kwong and L. A. Pieniazek, December 1986.
20. "HP-UX Reference," Hewlett-Packard Company, 1985.
21. "HP-UX Concepts and Tutorials," Hewlett-Packard Company, 1985.
22. "The C Programming Language," B. W. Kernighan and D. M. Ritchie, Prentice-Hall, Inc., 1978.
23. "Automated Code Generator for C Header Files," TRW Memo 86:W482.8-25, L. A. Pieniazek, 8 October 1986.
24. "Automated Software Documentation Utility," TRW Memo 86:W482.8-28, L. A. Pieniazek, 27 October 1986.
25. "Interface Control Document for ORDC Computer Compatible Tapes," CSC-1921, G. E. Habacek/CSC, February 1984.
26. "HP General File Format (GFF) User's Guide," TRW 39107-H021-UX-00, D. Campbell, 14 December 1984.
27. "DISSPLA User's Manual," Integrated Software Systems Corporation, 1981.

3.0 FUNCTIONAL DESCRIPTION

This section presents a functional description of the execution of RELBET 4.0. The description is at the level of major processing functions and will identify the various programs and "shell scripts" or run streams which one executes to accomplish each function. The overall process is first described and then the major steps examined in more detail. Section 5.0 of the User's Manual provides a detailed example of the process. The following discussion makes parenthetical references to the appropriate subsection of Section 5.0.

3.1 GENERAL OVERVIEW

The RELBET production procedure consists of five basic steps:

- o Data Collection
- o Input Data Formatting
- o Input Data Preparation
- o Trajectory Estimation
- o Generation and QA of Output Products.

Figure 3.1.1 summarizes this activity.

The activity begins with the collection of pertinent mission requirements and parameters. These are used to request post-flight telemetry products and to update a data base of standard program inputs. This preparation activity is primarily manual and does not utilize any of the RELBET programs.

The first actual processing phase begins with the receipt of telemetry data. The downlist telemetry comes in the form of Computer Compatible Tapes (CCT's). Various RELBET programs are used to reformat the CCT into files accessible to the various RELBET processors. These files contain information such as vehicle attitude data, relative observation data, and sensed velocity data. This step also transforms observation data into the RELBET reference frames and edits data according to the onboard data good flags.

Various specialized quality assurance (QA) programs support the next processing phase, data preparation. During this activity one checks the

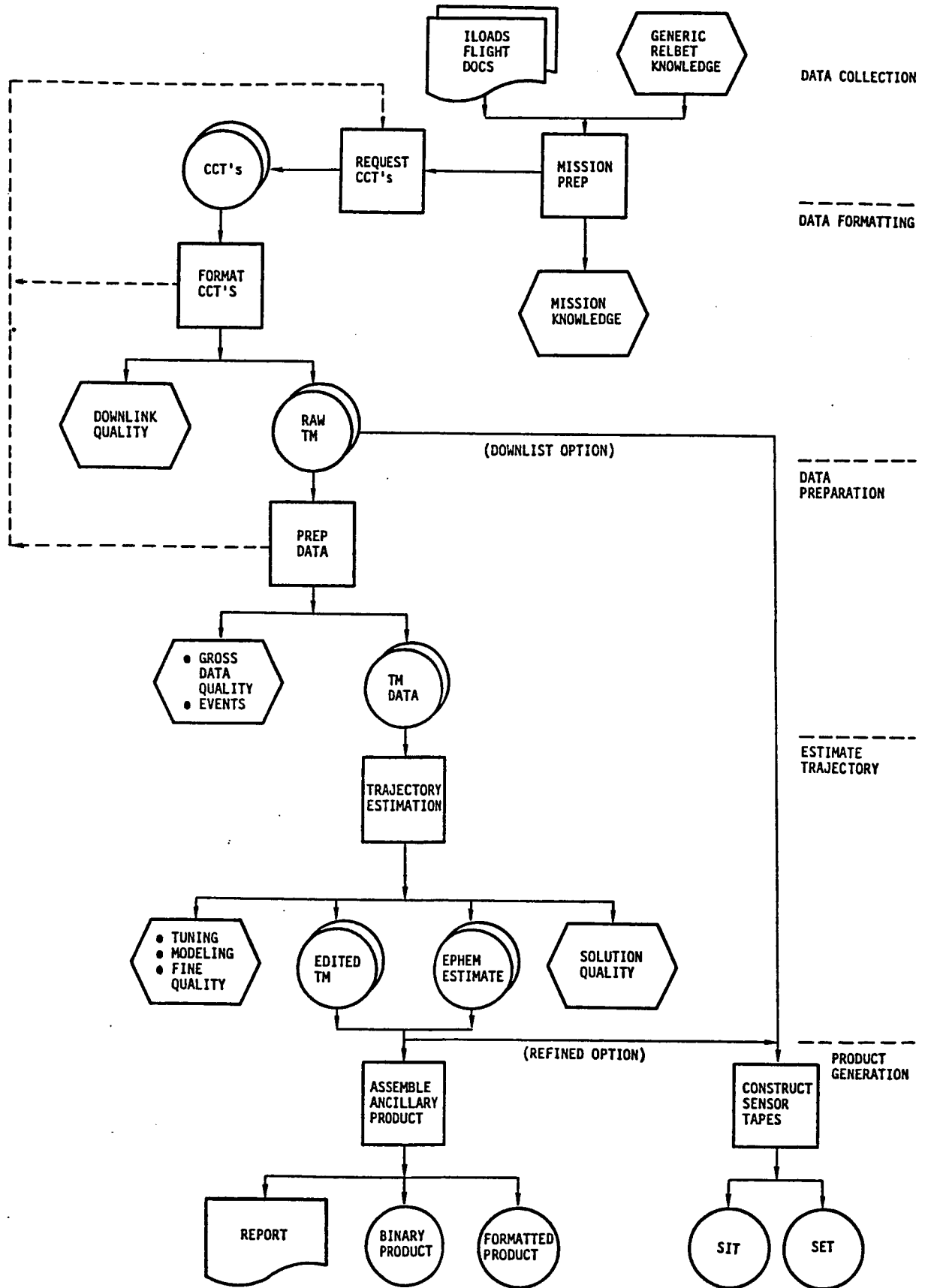


Figure 3.1.1. RELBET Production Overview

overall quality of the downlist for basic telemetry difficulties such as dropouts or invalid data. If the number of difficulties are excessive, a request for replacement CCT's may be made. This phase further examines the "raw" telemetry data and edits gross outliers. Precise times for pertinent events such as attitude maneuvers or tracking intervals are also identified. Such information is useful in resolving anomalies that may occur in subsequent processing. At the completion of this phase, one has determined that the input data are usable and has generated information for later review and analysis.

The next phase deals with estimating the relative trajectory. One also gives the quality and consistency of the data greater scrutiny during this phase. During this process a Kalman filter processes the telemetry. The output of the filter is reviewed to detect various anomalies. These anomalies are then either corrected or accounted for by correlating them with specific events that are known to cause problems. When satisfactory performance with the Kalman filter is obtained, its solution estimates are processed with a "smoother" program to obtain a "Best Estimate of the Relative Trajectory" (RELBET).

During the final processing phase the required output products are assembled from the various refined estimates and downlist telemetry. This activity is supported by a variety of computational, formatting and quality assurance programs.

3.2 CCT DATA FORMATTING

Before it can be used, the input telemetry data must be checked and reformatted into a format accessible by the RELBET processors.

The major source of input data for the RELBET process is the downlist telemetry contained in the CG011X CCT (Computer Compatible Tape). (CG011X is a particular shuttle data product generated for each mission.) In general the periods of interest are long enough that several CCT's are required. These must be reformatted and merged into files with the format used by the RELBET processors. Figure 3.2.1 depicts this process. The **dwnfmt** program strips

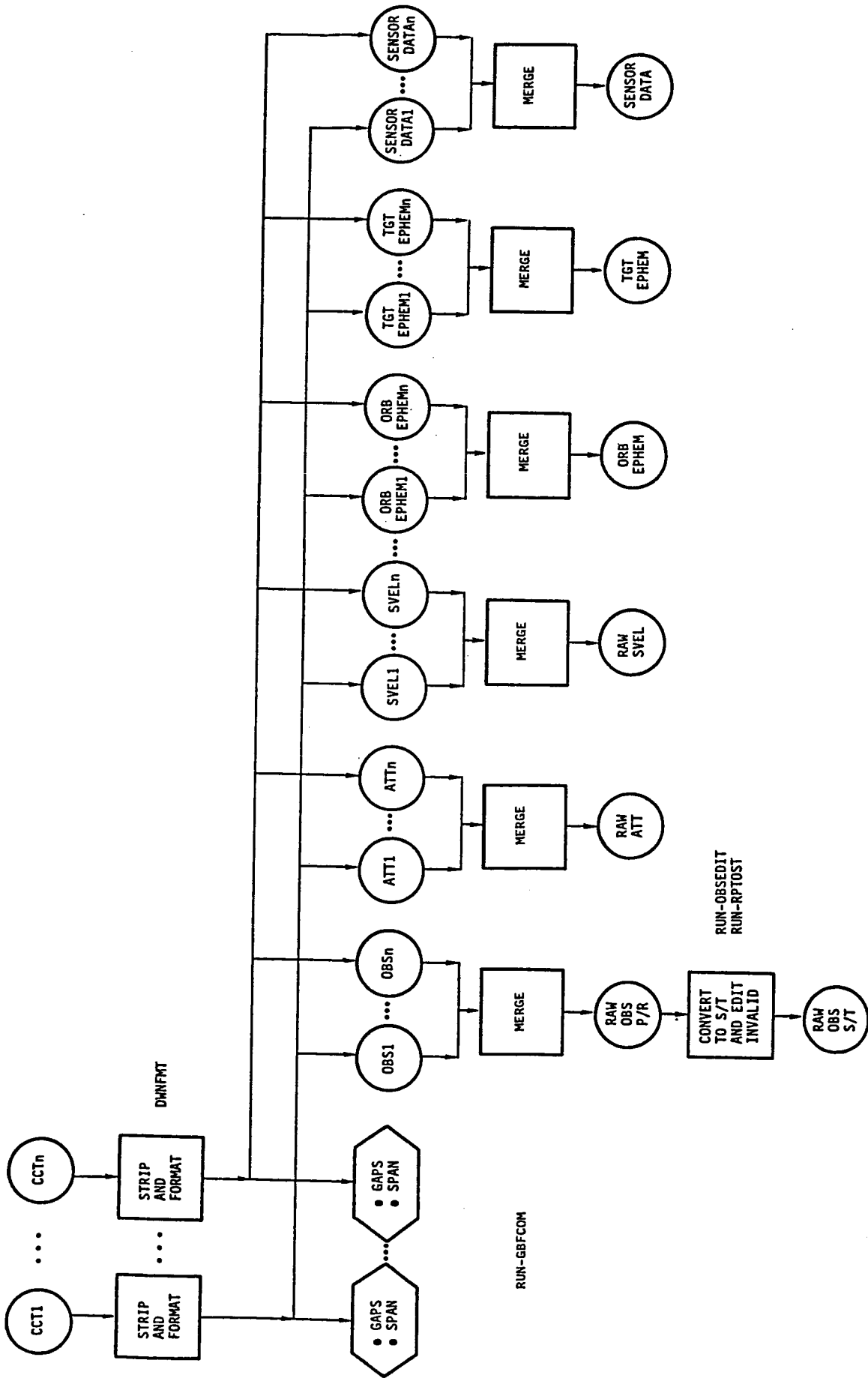


Figure 3.2.1. Format CCT's

required parameters from the CCT, checks them for validity, and generates time ordered binary files containing properly scaled parameters (5.2.1). In doing this it identifies and extracts the homogeneous data buffer created by the onboard rendezvous navigation program. This buffer is generated at the navigation cycle frequency (about a quarter of a Hertz) and downlisted asynchronously at a frequency of about one Hertz. Note that several files may be produced from single CCT, each file with its particular contents. The following files are required for later processing:

- o Attitude file: contains selected quaternions and is used to determine orbiter attitude. The following discussion uses the abbreviation "ATT" for this file.
- o Observation file: contains relative observations. The following discussion uses the abbreviation "OBS" for this file.
- o Sensed velocity file: contains selected sensed velocities for use in propagating the orbiter through burns. The following discussion uses the abbreviation "SVEL" for this file.
- o Vehicle ephemeris files: contain on-board state estimates that are used primarily for analysis and to obtain initial estimates. The following discussion uses the abbreviations "OEPH" and "TEPH" respectively for the orbiter and target ephemerides.
- o SENSOR Information File: contains miscellaneous data required for generating the SENSOR tapes.

Several CCT's are usually needed to encompass an entire mission segment and `dwnfmt` handles only one CCT at a time. Thus the telemetry for a mission segment usually involves multiple telemetry files for each of the above types. Subsequent processors expect only one file of each type. Thus it is necessary to merge the various files together. The shell program `run_gbfcom` will merge any number of files together (5.2.2).

The onboard systems associate a data good flag with each observation. If this flag indicates that the current observation is bad then the onboard filter does not process the datum. The shell program `run_obsedit` edits the OBS file for such data (5.2.3). This prevents the RELBET filters from processing bad data. The radar angles as downlisted are in the roll-pitch frame used for

crew display. The shell program **run_rptost** converts the roll-pitch angles to shaft-trunnion angles (5.2.4). The latter are preferred for filtering because they theoretically exhibit statistically independent biases.

At the conclusion of this processing phase the CCT's have been formatted into files required by the subsequent processors.

3.3 DATA PREPARATION

The input data formatting process gets the telemetry contained in the CCT's into the RELBET system. Before processing the raw telemetry with the filters or generating output products further processing is necessary. Figure 3.3.1 depicts this process. It involves assessing the data coverage, editing bad data points, and identifying events and anomalies that are not detected while stripping the data. One then saves all this information into the event data base.

Several shell programs help in this process. The shell script **run_qacover** summarizes data coverage of all the data types on the downlisted observation, attitude, and sensed velocity files (5.3.1). These files are analyzed to determine the quality of the CCT's. If coverage is not complete, the Input Data Formatting must be repeated with a newly requested CCT. If it is complete, all problems associated with the downlink of the CCT's have been identified and handled.

The execution of the shell script **run_qadata** identifies events in the data to help in assessing filter performance later in the processing steps by providing text files and some data files in gff format which can be analyzed or plotted (5.3.2). The shell program **run_noise** runs the noise analysis procedure. It computes noise values for each observation type and enters this value into the proper slot of the observation file (5.3.3). The Kalman filter needs the noise value for its processing. The program **run_noise** also generates text files that summarize the noise information.

One then uses the text files output from **run_qadata** to edit the attitude, sensed velocity, and observation files for bad data. Several shell programs facilitate this editing process. The UNIX **sort** program merges the text files

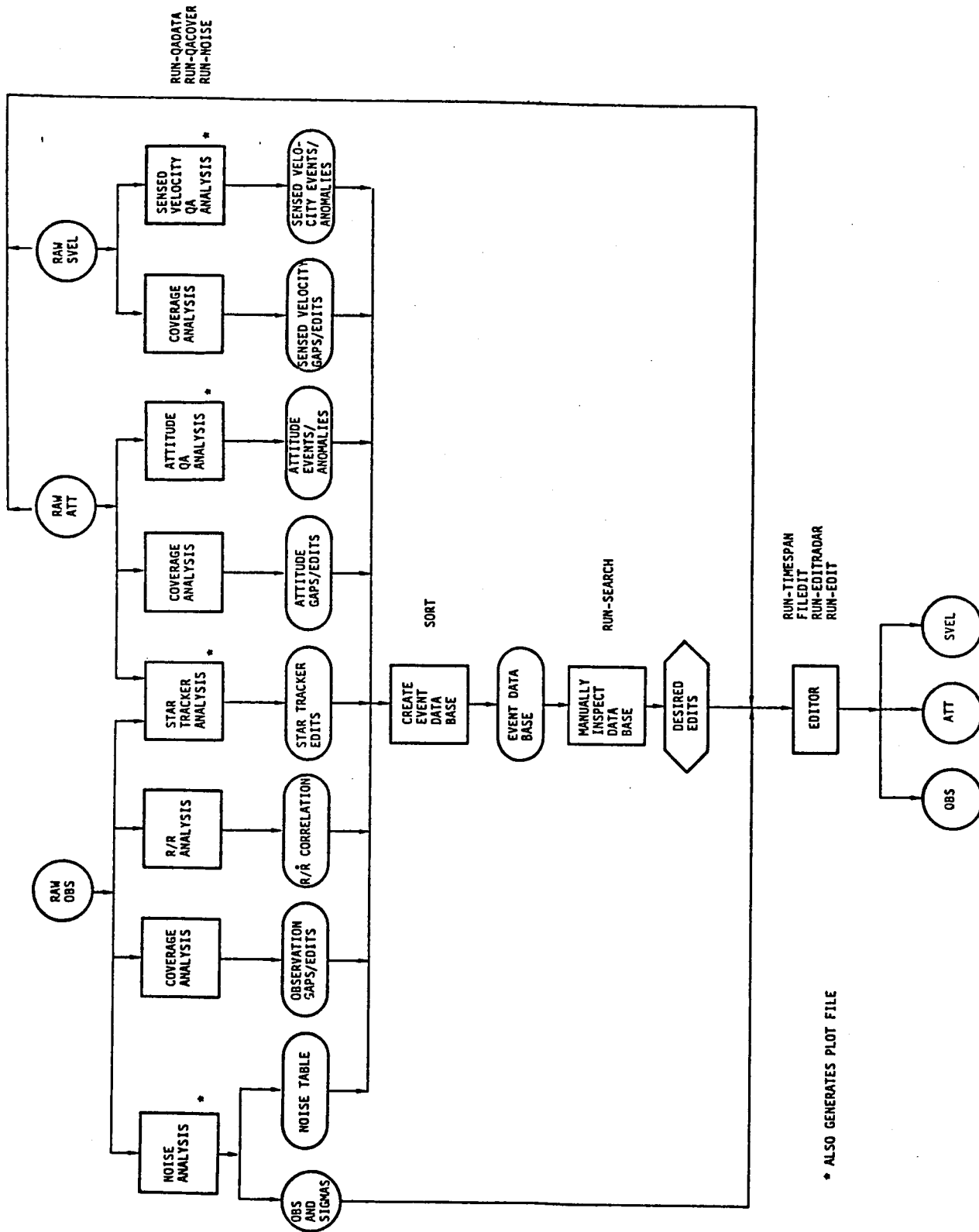


Figure 3.3.1. Data Preparation

output by these programs into the event data base. From the event data base, the shell program `run_search` (5.3.5) extracts information relating to possible difficulties. A knowledgeable engineer can review this information and identify time intervals during which files should be edited. The shell program `run_timespan` aids in preparing the time intervals (5.3.6). One uses the program `fileedit` to edit the desired files. Two shell programs, `run editradar` and `run_editst` help in editing the radar and star tracker data.

At the conclusion of the data preparation phase, one has edited the input data and created a data base which will be useful for correlating problems during a particular time period. The various gff files may be plotted for more detailed analysis and review (5.3.7).

Table 3.3.1 summarizes the various QA programs.

Table 3.3.1. QA Programs

<u>Name</u>	<u>Text File</u>	<u>Gff File</u>
qaatt	identifies attitude manuevers and bad points on the attitude file	angular momenta and magnitudes between consecutive attitudes
qaranjmp	identifies inconsistencies in range and range rate to detect sudden biases or equipment reconfiguration	
qastar	identifies false targets that the star tracker may have locked onto	star file containing azimuth, elevation, and angle for each star tracker observation
qasv	identifies burns as jumps in the magnitudes of the sensed acceleration computed from the sensed velocity and bad data	sensed acceleration file
qanois	documents the noise for each observation type	noise files for each observation

3.4 TRAJECTORY ESTIMATION

Up until now the different input items have been looked at more or less individually. During the next processing phase they are combined together to obtain a best Kalman estimate. Figure 3.4.1 depicts this process.

The program `sfilt` uses a Kalman filter to process the various observation information and the quality of the estimate is assessed (5.4.3). The programs `sln2rl`, `cmp2sg`, and `rlvsrl` aid this assessment. The display programs `qplot`, `xqdisp`, and `gdisp` help to investigate the contents of gff files generated by the Kalman filter. In general there will be various spikes and violations of different success criteria (e.g., 3-sigma residuals, excessive Kalman edits, etc.). These are manually correlated with possible causes and corrective adjustments are effected when possible. This filter-evaluate-adjust process continues until the user decides that no further refinements are merited.

At the conclusion of this phase, the best Kalman estimate of relative trajectory has been obtained.

Kalman estimates tend to contain unrealistic spikes and also reflect uncertainties due to looking at only one side of the data arc. A smoother algorithm is used to remove the spikes and improve estimates by considering both sides of the data arc. The smoother program `smooth` processes the results of the best Kalman estimate. The results of this process are compared with the Kalman estimate with the aid of the programs `sln2rl`, `cmp2sg`, and `rlvsrl` and the various display programs. If there is a large discrepancy between the smoothed and Kalman estimates further adjustments may be necessary. When a satisfactory smoothed estimate is obtained the program `sln2rl` extracts the relative trajectory information from the smoothed solution.

At the conclusion of this processing phase the best estimate of relative trajectory has been obtained.

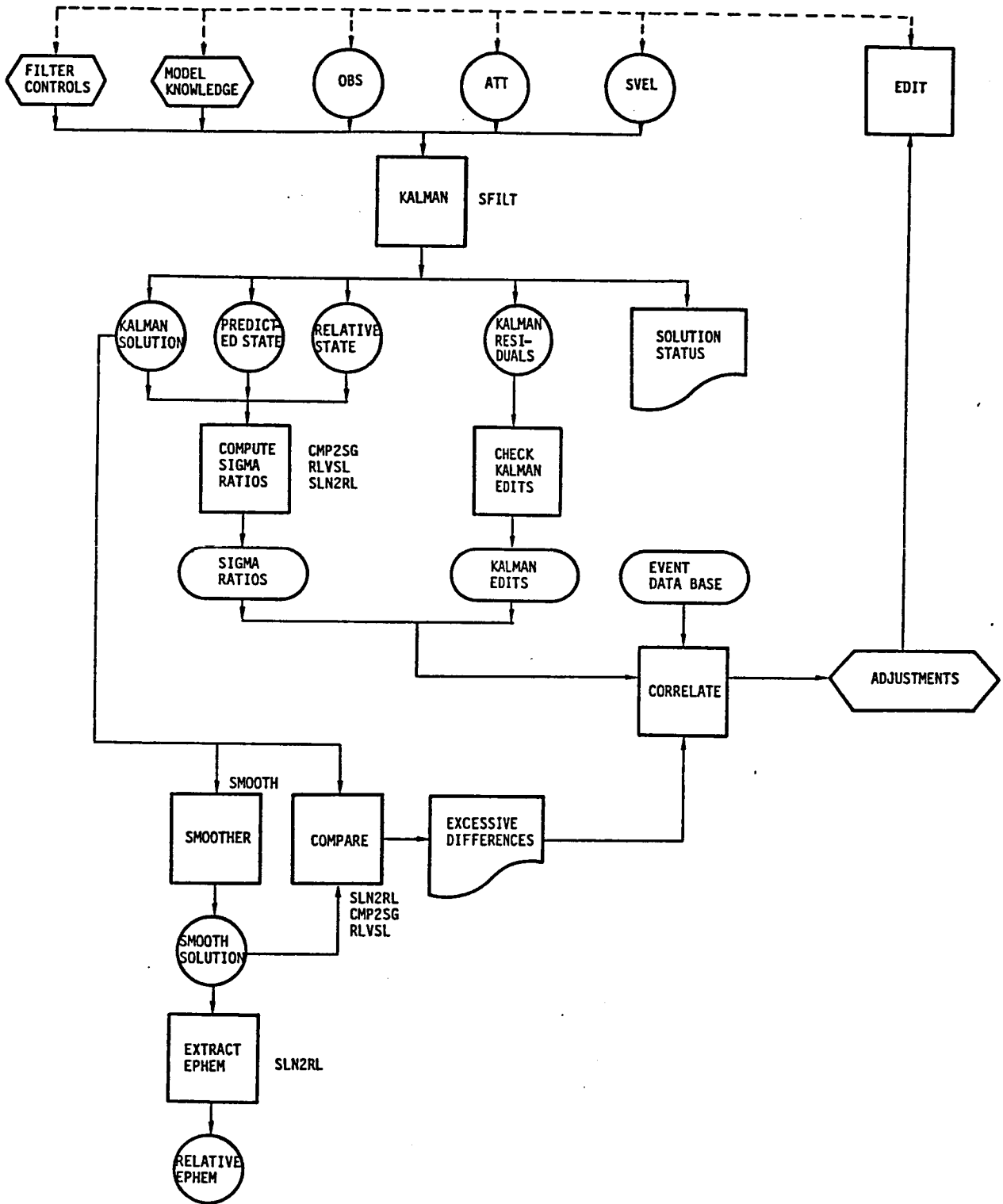


Figure 3.4.1. Trajectory Estimation

3.5 PRODUCTS GENERATION

When suitable quality telemetry data and trajectory estimates have been obtained, the output ancillary products are generated. Figure 3.5.1 depicts the process for the Ancillary Data Products. The program **prodx** is used to compute the required parameters (5.5.2). The program **qatape** is used to read the product tape and perform a check of its contents (5.5.3).

Other formatting programs such as **stop**, **fiche**, and **mkatape** are available for generating other products such as the SENSOR tapes and microfiche print. Figure 3.5.2 depicts the process for creating the SENSOR tapes: the SIT (SENSOR Input Tape) and the SET (SENSOR Environment Tape). The final report is obtained by manually assembling various plots and tables obtained in previous processing (5.5.5).

RELBET processing is done when all identified products have been delivered.

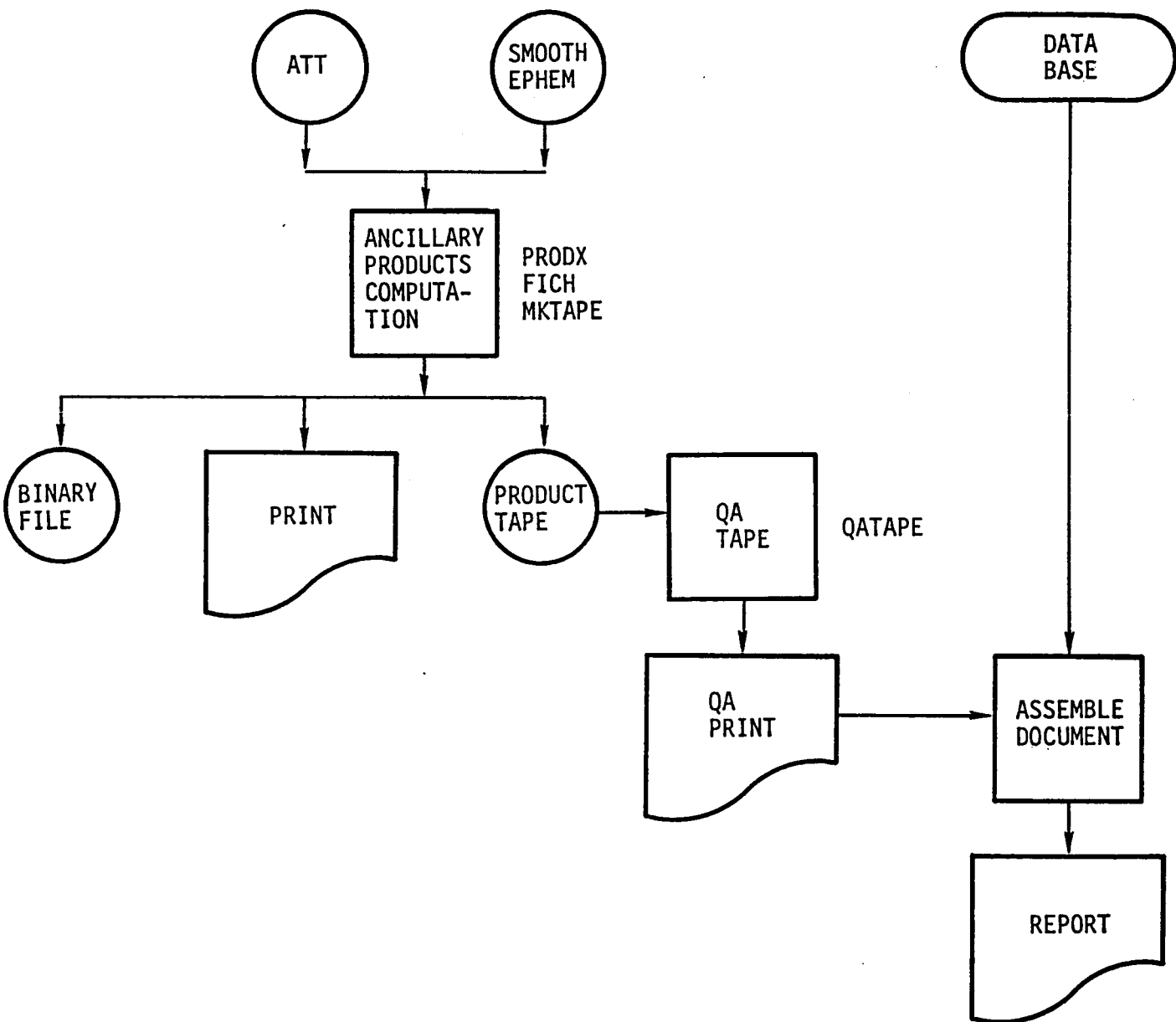


Figure 3.5.1. Ancillary Product Assembly

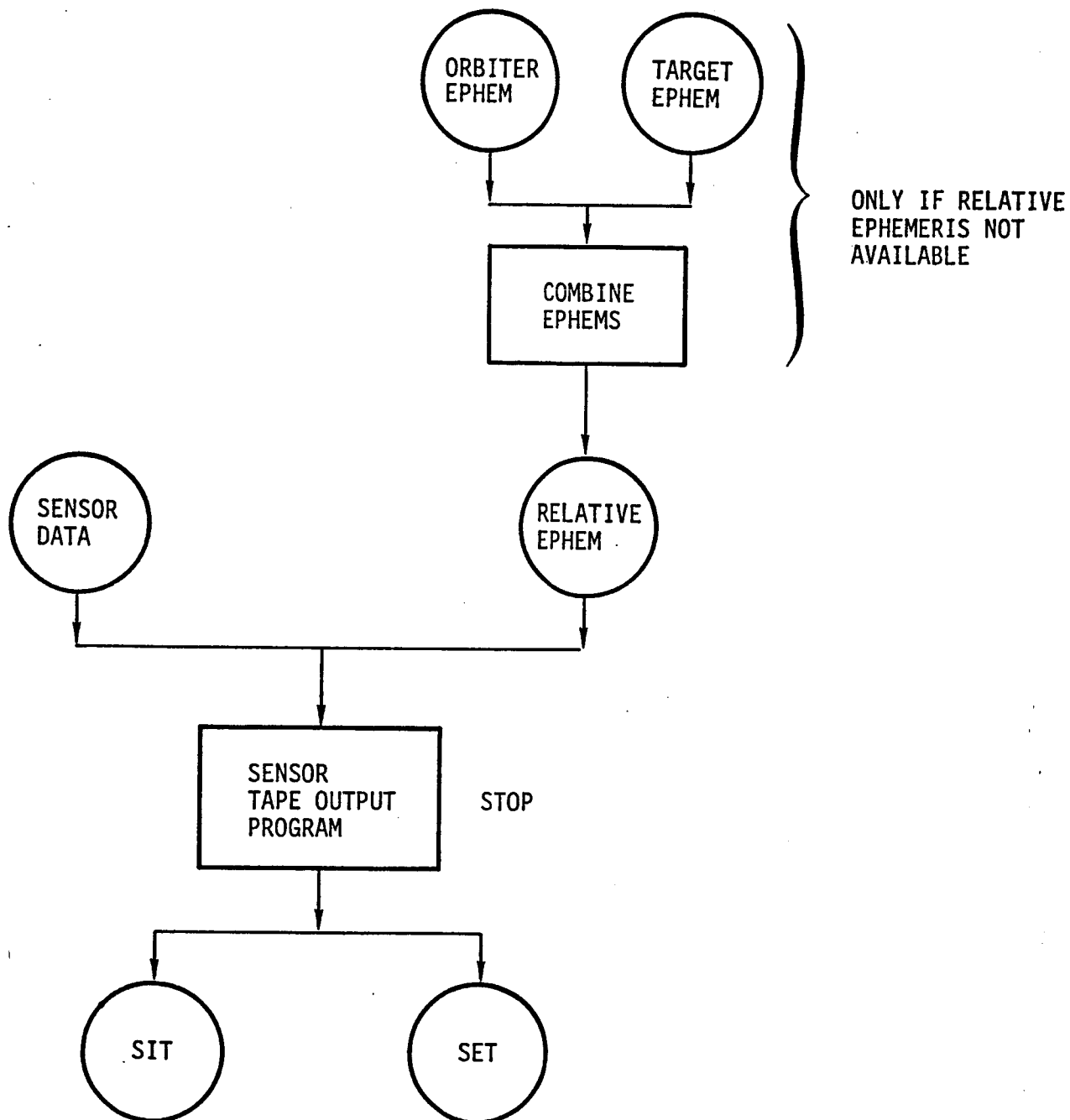


Figure 3.5.2. SENSOR Tape Construction

4.0 PROGRAM INPUT AND EXECUTION

All details concerning input and execution of RELBET processors are presented in the appendices. Appendix I contains the manual entries for each processor which provide complete details of processor function, discusses command line arguments and options available, if any, and provides a list of input blocks, if any, to be piped into the process at execution time.

Appendix II contains manual entries for specified input blocks which provide a definition and information on type, size and default (if any) for each variable in the input block.

Appendix III contains articles which provide the user with more information than may be available in the program manuals found in Appendix I.

4.1 COMMAND LINE ARGUMENTS

The programs which utilize command line arguments for input of user controls work basically as do normal UNIX programs with the exception that when an option is indicated which requires information input then the information appears immediately after the option with no separating space, i.e., the option '-f' indicates that the following characters are to be interpreted as a file name so that '-fname' indicates that the file name is 'name'. Note that the use of the dash '-' preceding the option flag is typical.

4.2 MENU DRIVEN INPUT

Menu driven input is basically self explanatory and is a major factor only in the program **gdisp**. Consult the manual in Appendix I.

4.3 LINPUT INPUT

The following discusses the linput input scheme for those processors which require linput input. This scheme provides a basic capability similar to the popular FORTRAN NAMELIST as well as additional computational capability. The best way to proceed is to read the simple example then to scan the guidelines in order to note the terminology and tables, and then to read the not so simple example using the guidelines for reference.

4.3.1 Simple Example

This example will create an input text file from scratch for the **rptost** processor which reads in the observation gff file and converts the roll and pitch radar measurements to statistically independent shaft and trunnion angles which are required by the **sfilt** processor.

- Step 1: Review the manual entry for the program **rptost** in Appendix I. Note that the input blocks required are **xxgnrl**, **xxtime**, and **xxrpst** which allow user initialization of the internal common blocks **xxgnrl**, **xxtime**, and **xxrpst** respectively.
- Step 2: Review the manual entries for the input blocks **xxgnrl**, **xxtime**, and **xxrpst** in Appendix II.
- Step 3: Use the UNIX editor to create a new text file called 'anything'.
- Step 4: Insert the following comments using the linput syntax to indicate the user is satisfied with utilizing the default values for the **xxgnrl** common block.

```
// gnr1  
// use defaults
```

Remember that these lines are ignored by the linput process because anything following the symbol '//' is ignored.

Step 5: Insert the following:

```
// time
date = 1985 , 1 , 0 , 0 , 0 ;
dates = 0.0;
tbegin = -1.e30;
tend = 1.e30;
```

Notice the line separators ';'.

Note that the exponents are entered in C language notation using the 'e'.

Step 6: Insert the following:

```
// rpst
infil = "OBS";
outfil = "OBS_OUT";
%%
```

Note that characters are entered between single quotes.

Note the end of file marker '%%'.

Step 7: Exit the file 'anything'. The file contents should resemble:

```
// gnr1
// use defaults
// time
date = 1985, 1 , 0 , 0 , 0;
dates = 0.0;
tbegin = -1.e30;
//rpst
infil = "OBS";
outfil = "OBS_OUT";
%%
```

Step 8: The input file is now created and can be provided to the processor **rptost** during execution as follows:

```
rptost < anything
```

4.3.2 Guidelines

RELBET production experience offers the following guidelines in the use of the input input scheme. These guidelines should help you understand the examples in the step by step procedures of Section 5.

- Store one input block per file
- Input blocks are generally associated with internal FORTRAN common blocks. For example the input block xxgnrl or gnrl is the user interface to the initialization of several parameters in the xxgnrl common block. Note that input blocks referred to as dwnfmt.files, dwnfmt.msids, and dwnfmt.cct are exceptions and are only indirectly associated with the common blocks xxfiles, xxmsid, and xxcct respectively.
- Convention dictates that if the common block having user input options has the name "xxname" then the input block may be referred to with or without the xx, i.e., as "name" or "xxname".
- A collection of one or more input block files is called a category of input block files.
- Several categories exist for the inputs required by RELBET processors as described in Table 4.3.2.1.

Table 4.3.2.1. Input Categories

<u>CATEGORY</u>	<u>INPUT BLOCK FILES</u>	<u>PROCESSOR</u>
Cct	cct , dwnfmt.cct	dwnfmt
Compar	gnrl , time , nflz	xcmpar
Kalman	sen , bias , kal scov	sfilt
Master	gnrl , time , misc mast	dwnfmt
Msids	dwnfmt.msids	dwnfmt
Numdis	gnrl , time , con erth , nflz , prnt dprm , usys	xqdisp
Plot	gnrl , time , misc ggen , qprm , qcrv graf , pmmx	qplot , ascale
Product	gnrl , time , misc con , nflz , toff sprm	prodx
Propagation	gnrl , con , misc time , file , sptm erth , grav , atm sun , moon , init prop , max , vcx mas , vnt , svbi rnpX	sfilt , smooth
Rptost	gnrl , time , rpst	rptost
Smooth	bias , scov	smooth

- Note that most categories are identified one to one with a particular processor but some categories are common to more than one process and some processes utilize more than one category.
- In general RELBET input consists of one or more categories separated by an end of file marker %%.
- Within a category the ordering of the input block files and their contents are not significant.
- For processors requiring several categories the order of the categories is important. They must appear in the order indicated in Table 4.3.2.2.

Table 4.3.2.2. Category Order

<u>PROCESSOR</u>	<u>CATEGORY ORDER (First to Last)</u>
dwnfmt	Master , Files , Msids , Cct
sfilt	Propagation , Kalman
smooth	Propagation , Smooth

- Remember that between these categories a %% marker must occur.
- The dwnfmt processor allows for building arrays within common blocks by the repeated use of the same category. The only repeatable categories available in RELBET are Files and Msids. One of the parameters in these categories is called 'end'. When non-zero, it indicates the last inputs for the category. The inputs for these repeatable categories usually appear in one text file along with a %% marker between each repeated category.
- The collection of categories of input block files required by a process may be either 'cat'ed to an interim file which is then redirected to the process via standard input or 'cat'ed and piped directly to the process. Remember that care must be taken to ensure that the proper ordering of categories is preserved as required by a particular process and that the end of file marker %% appears in the appropriate places.

4.3.3 Not So Simple Example

In the following example we construct the inputs for the program `sfilt` which performs the Kalman filter process to generate an estimate of the relative trajectory. As observed in the previous section the program `sfilt` uses the categories of input block files called Propagation and Kalman. In general once the individual input block has been built for one run it is modified rather than rebuilt for the next run so that we assume there exist text files representing the input blocks belonging to the required categories.

Step 1: Review the text files for each input block needed by the Propagation and Kalman categories to reconfigure the values and options for the desired process. Assume the text files are named according to convention discussed above.

Step 1a: Begin with the Propagation category of inputs.

Step 1b: Use the UNIX editor to look at the file called `gnrl`. In this case the `pbug` parameters are likely to need adjustment to set printed and binary output options.

Step 1c: Continue to run through the entire list of input files making adjustments as required. Note that all of the inputs available for processing need not appear in the text files, as defaults are invoked simply by the non-appearance of input parameters in the `linput` syntax. Thus until the user is familiar with the inputs available careful consultation with the input descriptions in Appendix II are necessary. When reviewing the files make sure that no end of file markers `'%%'` appear. Keep in mind that if a parameter is set more than once then the value of the last occurrence is the one used. This is the only case where the order of parameters within a category makes any difference.

Step 1d: Do the same for the Kalman category of inputs.

Step 2: Create a file containing the end of the file marker '%' and call it anything, say EOF. The file contains only one line as follows:

%%

Step 3: Provide the inputs to the **sfilt** program.

Any number of methods can be used to direct the input block files to the processor at execution some allowing more flexibility than others. The simplest is to copy all of the input block files along with the EOF file at the appropriate places to an interim file as in the following shell script.

```
cat /
gnrl con misc time file sptm erth grav /
atm sun moon init prop max vcx /
mas vnt svbi rnp /
EOF /
sen bias kal scov /
EOF /
> anything
```

If the above script is run then the inputs needed by **sfilt** are in file **anything**. Notice the placement of the EOF files after copying each category to **anything**. Note that in UNIX the symbol '/' tells the shell interpreter to ignore the new line, i.e., the '/' is a line continuation symbol.

To execute the program the user then performs:

```
sfilt < anything
```

A better way to feed the inputs into the program is illustrated in the following annotated shell script.

```
# set environmental variables indicating where to find the various
# input block text files belonging to categories required by
# the sfilt program
```

```
# The end of file marker file
EOF=/users/Desired/Directory/EOF
```

```
# The Propagation category files
```

```
P1 =/users/Desired/Directory
```

```
P2 =$P1
```

```
P3=$P1
```

```
P4=$P1
```

```
P5=$P1
```

```
P6=$P1
```

```
P7=$P1
```

```
P8=$P1
```

```
P9=$P1
```

```
P10-$P1
```

```
P11=$P1
```

```
P12=$P1
```

```
P13=$P1
```

```
P14=$P1
```

```
P15=$P1
```

```
P16=$P1
```

```
P17=$P1
```

```
P18=$P1
```

```
P19=$P1
```

```
# The Kalman category files
```

```
K1=/users/Desired/Directory
```

```
K2=$K1
```

```
K3=$K1
```

```
K4=$K1
```

```
K5=$K1
```

```
# display the current file selections
```

```
# this allows for a record of what files were used for this run
```

```
echo $P1/gnr1
```

```
echo $P2/con
```

```
echo $P3/misc
```

```
echo $P4/time
```

```
echo $P5/file
```

```
echo $P6/sptm
```

```
echo $P7/erth
```

```
echo $P8/grav
```

```
echo $P9/atm
```

```
echo $P10/sun
```

```
echo $P11/moon
```

```
echo $P12/init
```

```
echo $P13/prop
```

```
echo $P14/max
```

```
echo $P15/vcx
```

```
echo $P16/mas
```

```
echo $P17/vnt
```

```

echo $P18/svbi
echo $P19/rnpx
echo $K1/sen
echo $K2/bias
echo $K3/kal
echo $K4/scov

# display the contents of input block files
# this allows for a record of user inputs to this run
more
$P1/grn1 $P2/con $P3/misc $P4/time $P5/file $P6/sptm $P7/erth $P8/grav /
$P9/atm $P10/sun $P11/moon $P12/init $P13/prop $P14/max $P15/vcx /
$P16/mas $P17/vnt $P18/svbi $P19/rnpx /
$EOF /
$K1/sen $K2/bias $K3/kal $K4/scov /
$EOF

# cat and pipe categories into sfilt
cat
$P1/grn1 $P2/con $P3/misc $P4/time $P5/file $P6/sptm $P7/erth $P8/grav
$P9/atm $P10/sun $P11.moon $P12/init $P13/prop $P14/max $P15/vcx
$P16/mas $P17/vnt $P18/svbi $P19/rnpx
$EOF /
$K1/sen $K2/bias $K3/kal $K4/scov /
$EOF
... | sfilt

```

Suppose the above shell script is in a file called **cat_sfilt**. Then to run this shell script perform:

```
cat_sfilt > OUT
```

Note that this produces a listing of the input block files used, their contents and the nominal printed output of **sfilt** onto the file called **OUT**.

5.0 STEP-BY-STEP PROCEDURAL EXAMPLE

This example represents one way to perform RELBET analysis. It does not illustrate or allude to variations in the procedures. The illustration is straight forward and follows the description set forth in Section 3.0. There are variations which may be more efficient than those illustrated; however, those are left to be discovered by the user as familiarity with the RELBET functions is gained.

All of the data files and program outputs have been collected on supplementary tapes provided in addition to the software delivery tape. Note that the shell scripts which have a prefix of `cat_` are ad hoc creations for the purpose of running several programs which require linput input. These scripts appear only on the supplementary tape.

This example assumes that all executable programs reside in the HP directory `/users/Relbet/Master/Control` and that this directory belongs to the users `.profile` file. The directory `/data0/Work/Example` is assumed to be the current working directory. All files created and used are in the directory `/data0/Work/Example`. Note that some shell scripts use the directory `/tmp`.

The process of production creates various files which are then processed and modified to create other files. Table 5.1 presents a summary of all files created in the production process. The numbers listed in the Source and Target columns refer to subsections of this article, i.e., Section 2.1 refers to subsection 5.2.1. In this example, three CCTs from mission 51I were processed through a production-like scenario. The 51I rendezvous is a good example in that it illustrates the kind of problems which may arise in a post-flight analysis situation. There appear to be problems with the star tracker data especially during one period. These problems are pointed out, however, analysis of the problems is beyond the intention and scope of this discussion which is primarily to demonstrate how the RELBET processors work together. In other words, there is no claim that the RELBET solution achieved through these examples is in fact the very best estimate of relative trajectory possible for this mission.

Table. 5.1. Files Summary

<u>File</u>	<u>Type</u>	<u>Source</u>	<u>Target</u>	<u>Description</u>
ATT	gff	2.2	3.1,3.2 4.3,4.5 5.2	Attitude data file
ATT1	gff	2.1	2.2	Attitude data file from CCT1
ATT2	gff	2.1	2.2	Attitude data file from CCT2
ATT3	gff	2.1	2.2	Attitude data file from CCT3
ATT_AR	gff	3.2		plot file for significant movements about the orbiter rotation axis
CCT1	tape	1.2	2.1	First CCT from E_bin Library
CCT2	tape	1.2	2.1	Second CCT from E_bin Library
CCT3	tape	1.2	2.1	Third CCT from E_bin Library
CMP_POS_KS	text		4.5	Sigma ratio comparison between relative position elements of KREL and SREL relative trajectory files
CMP_POS_PK	text	4.4	4.4	Sigma ratio comparison between relative position elements of PRED and KREL relative trajectory files
CMP_VEL_KS	text	4.5		Sigma ratio comparison between relative velocity elements of KREL and SREL relative trajectory files
CMP_VEL_PK	text	4.4	4.4	Sigma ratio comparison between relative velocity elements of PRED and KREL rel traj files
KCOV	gff	4.4	4.5	Kalman computed sigmas in UVW
KOBS	gff	4.3	4.4	Kalman pre-edit obs residuals, sigmas, bias solutions, edit status
KOBS_ED	gff	4.4	4.4	Copy of KOBS file where the frames are edited whenever a Kalman edit occurs
KREL	gff	4.3	4.4,4.5	Kalman relative trajectory solution
KREL_RL	gff	4.5	4.5	UVW difference between Kalman and smooth relative trajectory solutions

Table. 5.1. Files Summary (Continued)

<u>File</u>	<u>Type</u>	<u>Source</u>	<u>Target</u>	<u>Description</u>
KSOL	gff	4.3	4.5	Kalman solution file
KTEXT	text	4.4		Kalman analysis text data base
OBS	gff	2.2,2.3	3.1,3.2	Observations data file
		2.4,3.7	3.3,3.8	
OBS_ST	gff	3.2		plot file inertial el and az of Star Tracker angles
OBS_ddnois	gff	3.3	3.3	Master collection of noise values
OBS_ddnois_Na	gff	3.3	3.3	Noise value per time interval for radar range
OBS_ddnois_Nb	gff	3.3	3.3	Noise value per time interval for radar range rate
OBS_ddnois_Nc	gff	3.3	3.3	Noise value per time interval for radar shaft angle
OBS_ddnois_Nd	gff	3.3	3.3	Noise value per time interval for radar trunnion angle
OBS_ddnois_Ni	gff	3.3	3.3	Noise value per time interval for Star Tracker Horizontal angle
OBS_ddnois_Nj	gff	3.3	3.3	Noise value per time interval for Star Tracker Vertical angle
OE	gff	2.2	4.1,5.5	Orbiter Ephemeris data file
OE1	gff	2.1	2.2	Orbiter Ephemeris data file from CCT1
OE2	gff	2.1	2.2	Orbiter Ephemeris data file from CCT2
OE3	gff	2.1	2.2	Orbiter Ephemeris data file from CCT3
OE_REL	gff	5.5	5.5	Onboard relative trajectory
OUTSFILT	text	4.3	4.4	Summary status print of sfillt
OUTSMOOTH	text	4.5		Summary status print of smooth
PRED	gff	4.3	4.4	Kalman predicted state

Table. 5.1. Files Summary (Continued)

<u>File</u>	<u>Type</u>	<u>Source</u>	<u>Target</u>	<u>Description</u>
PRED_RL	gff	4.4	4.4	UVW difference between predicted and Kalman relative trajectories states
QA_SP51I_PRT	text	5.3	5.3	Special Products print file
ROBS1	gff	2.1	2.2	Radar Observations data file from CCT1
ROBS2	gff	2.1	2.2	Radar Observations data file from CCT2
ROBS3	gff	2.1	2.2	Radar Observations data file from CCT3
SBIAS	gff	4.5		Smooth bias solutions and sigmas
SCOV	gff	4.5		Smooth computed sigmas in UVW
SET0	binary	5.5	5.5	SET with downlist data
SETR	binary	5.5	5.5	SET with RELBET data
SIT0	binary	5.5	5.5	SIT with downlist data
SITR	binary	5.5	5.5	SIT with RELBET data
SN	gff	2.2	5.5	SENSOR Information data file
SN1	gff	2.1	2.2	SENSOR Information data file from CCT1
SN2	gff	2.1	2.2	SENSOR Information data file from CCT2
SN3	gff	2.1	2.2	SENSOR Information data file from CCT3
SP51I_BIN	binary	5.2		Special Products binary file
SP51I_PRT	text	5.2	5.3,5.4	Special Products print file
SPtape	tape	5.2	5.3	Special Product Tape on drive 0
SREL	gff	4.5	5.2,5.5	Smooth relative traj solution
SSOL	gff	4.5	4.5	Smooth solution file

Table. 5.1. Files Summary (Continued)

<u>File</u>	<u>Type</u>	<u>Source</u>	<u>Target</u>	<u>Description</u>
STOBS1	gff	2.1	2.2	Star Tracker Observations data file from CCT1
STOBS2	gff	2.1	2.2	Star Tracker Observations data file from CCT2
STOBS3	gff	2.1	2.2	Star Tracker Observations data file from CCT3
SV	gff	2.2	3.1,3.2	Sensed Velocity data file
			4.3,4.5	
			5.2	
SV1	gff	2.1	2.2	Sensed Velocity data file from CCT1
SV2	gff	2.1	2.2	Sensed Velocity data file from CCT2
SV3	gff	2.1	2.2	Sensed Velocity data file from CCT3
SV_SV	gff	3.2		plot file for significant sensed accelerations
TE	gff	2.2	4.1,5.5	Target Ephemeris data file
TE1	gff	2.1	2.2	Target Ephemeris data file from CCT1
TE2	gff	2.1	2.2	Target Ephemeris data file from CCT2
TE3	gff	2.1	2.2	Target Ephemeris data file from CCT3
TEXT	text	3.4	3.5,4.4	Current working text data base
edit.attr	text	3.5	3.5	Times of anomalous attitude data
edit.oatt	text	3.5	3.5	Times of attitude data loss
edit.range	text	3.5	3.5	Times of anomalous radar data
edit.sacc	text	3.5	3.5	Times of anomalous sensed velocity data
edit.star	text	3.5	3.5,3.6	Times of anomalous star tracker data
ktextaran	text	4.4	4.4	radar range information from <u>KOBS_ED</u> file

Table. 5.1. Files Summary (Continued)

<u>File</u>	<u>Type</u>	<u>Source</u>	<u>Target</u>	<u>Description</u>
ktextbrnr	text	4.4	4.4	radar range rate information from <u>KOBS_ED</u> file
ktextcsft	text	4.4	4.4	radar shaft angle info from <u>KOBS_ED</u> file
ktextdtrn	text	4.4	4.4	radar trunnion angle information from <u>KOBS_ED</u> file
ktextizth	text	4.4	4.4	star tracker horizontal angle information from <u>KOBS_ED</u> file
ktextjztv	text	4.4	4.4	star tracker vertical angle information from <u>KOBS_ED</u> file
span.aran	text	3.6	3.7	time spans over which to edit radar data
span.star	text	3.6	3.7	time spans over which to edit star tracker data
textATT	text	3.1	3.4,3.8	attitude infoormation from <u>ATT</u> file
textNaran	text	3.3	3.4,3.8	Noise value per time interval for radar range
textNbrnr	text	3.3	3.4,3.8	Noise value per time interval for radar range rate
textNcsft	text	3.3	3.4,3.8	Noise value per time interval for radar shaft angle
textNdtrn	text	3.3	3.4,3.8	Noise value per time interval for radar trunnion angle
textNizth	text	3.3	3.4,3.8	Noise value per time interval for Star Tracker Hor angle
textNjztv	text	3.3	3.4,3.8	Noise value per time interval for Star Tracker Ver angle
textSV	text	3.1	3.4,3.8	sensed velocity info from <u>SV</u> file
textaran	text	3.1	3.4,3.8	radar range info from <u>OBS</u> file
textattr	text	3.2	3.4,3.8	timeline of significant movements about the orbiter rotation axis
textbrnr	text	3.1	3.4,3.8	radar range rate info from <u>OBS</u> file
textburn	text	3.2	3.4,3.8	timeline of significant sensed accelerations
textcsft	text	3.1	3.4,3.8	radar shaft angle information from <u>OBS</u> file
textdtrn	text	3.1	3.4,3.8	radar trunnion angle information from <u>OBS</u> file

Table 5.1. Files Summary (Continued)

<u>File</u>	<u>Type</u>	<u>Source</u>	<u>Target</u>	<u>Description</u>
textizth	text	3.1	3.4,3.8	star tracker horizontal angle information from OBS file
textjztv	text	3.1	3.4,3.8	star tracker vertical angle information from OBS file
textmrol	text	3.1	3.4,3.8	radar roll angle information from OBS file
textopit	text	3.1	3.4,3.8	radar pitch angle information from OBS file
texttrj	text	3.2	3.4,3.8	timeline of significant inconsistencies between range and range rate radar data
textstar	text	3.2	3.4,3.8	timeline of anomalous star tracker data (stars or bad marks)

5.1 DATA COLLECTION

The first step is to obtain the downlist data and transform it into internal system format.

5.1.1 Identify CCTs

This step begins prior to a flight by identifying specific times of interest in the nominal mission profile. This information is passed to the CCT librarian for pre-mission planning purposes. After the actual times of interest are known subsequent to a flight, those times are relayed to the CCT librarian who will, in turn, supply the necessary tape numbers.

5.1.2 Obtain L-bin Copies of CCTs

AT JSC, the downlist Computer Compatible Tapes (CCT) reside in the "E-bin library" which is currently housed in Building 12. You cannot check these tapes out and must use L-bin copies of the desired CCT's. You can arrange for this transfer or make your own copies by transferring the E-bin tapes to the P-library and copying them to L-bin tapes. The L-bin tapes should be 1600 BPI. In this case you can use the following procedure:

1. Transfer E-Bin Tapes to P-Library in Building 12

Currently (December 1986) tapes are transferred by calling Work Control at 483-3082 and requesting that an E-bin tape (give the tape number) be transferred to the P-Library for a few days.

2. Make an L-Bin Copy of the Tapes with UNIVAC

```
@JSC*CALLUP.TAPES,F BADGE*RUNID., E-binTAPE #  
@ASG,T TPIN.,U9S,E-bin TAPE #  
@REWIND TPIN.  
@JSC*CALLUP.TAPELABEL  
@ASG,TJ $L-TPOUT.,U9V,,56 . tapelable (if desired)  
@REWIND $L-TPOUT.
```

```

@COPY,NMC TPIN.,$L-TPOUT.           (copies header)
@COPY,NMC TPIN.,$L_TPOUT.          (copies data)
@FREE TPIN.
@FREE $L-TPOUT

```

This example assumes that you are a bona fide user of the NASA/JSC UNIVAC 1100 series system located in Building 12 and have access to a terminal connected to that system. It is assumed that the operating system has been accessed and that the proper @RUN cards have been entered.

The command @JSC*CALLUP.TAPELABEL instructs the computer operators to assign a new tape. The return comment will give the L-Library tape number. Prior to this point the printer should be on or the terminal screen copied so that the copies of the E-bin tape number and the corresponding L-bin tape number are acquired.

5.2 DATA FORMATTING

After obtaining the L-bin tape from building 12, the following procedure will create the necessary input files for the RELBET system.

5.2.1 Strip Data From CCTs

```

cat_dwnfmt > OUT#
mv OUTPUT OUTPUT#
head -100 BUGS BUGS#a
tail -100 BUGS BUGS#b
rm BUGS

```

where # represents the number corresponding to the CCT being processed and should also be the same as the suffix of the input parameter, **fname/xxdwnfmt files**, i.e., 1, 2, or 3. This process is run three times and takes about 45 minutes for each CCT. The shell script **cat_dwnfmt** collects the desired input files and pipes them into the **dwnfmt** processor. The script is usually

tailored for this purpose per execution using the UNIX `vi` editor.

The mandatory linut inputs are provided in the following `xxfiles`:

<code>xxgnrl.dwnfmt</code>	- pbug flags
<code>xxtimes</code>	- base date information
<code>xxmisc.dwnfmt</code>	- jobdes information
<code>xxmast</code>	- processing controls
<code>xxdwnfmt_files</code>	- output file descriptions
<code>xxdwnfmt_msids</code>	- desired m/sid descriptions
<code>xxdwnfmt_cct</code>	- skip label
<code>xxcct</code>	- tape drive id

You must mount the CCT tape on the drive indicated by the number assigned to `cct/xxcct`. The output gff file names are specified in `fname/xxdwnfmt_files` and must be changed for each CCT processed. Use a favorite UNIX file editor such as `vi` to modify the linut input files. Notice that the `dbproc/xxdwnfmt_files` flag is set for the output files `SN#` to indicate that frame generation is dependent on the validity of the time-tag only, otherwise the loss of a particular m/sid component of the output frame due to transmission problems would result in loss of the entire frame. Note that the base times on the output gff files specified in `date/xxtime` and `dates/xxtime` must be the same as the reference date of the CCT time-tags. These time-tags are usually in seconds from the beginning of the year.

If no data is found for a particular output gff file, then a error message is returned to the terminal. In this example no data are obtained for the files `COBS1`, `COBS2`, and `COBS3` so that an error message is generated by the file routines. This is not unexpected because COAS observation data (the contents of the `COBS#` files) are rarely available on rendezvous missions. These files are ignored in subsequent processing and may be removed from the system.

The following files are collected for each CCT:

<u>File</u>	<u>Type</u>	<u>Contents</u>
ROBS#	gff	Radar Observations
STOBS#	gff	Star Tracker Observations
ATT#	gff	Attitude Data
SV#	gff	Sensed Velocity Data
OE#	gff	Orbiter Ephemeris (Onboard Filter Solution)
TE#	gff	Target Ephemeris (Onboard Filter Solution)
SN#	gff	SENSOR Program Data
BUGS#a	text	Summary of output file structures and initial processing status
BUGS#b	text	Summary of output file contents and final processing status
OUT#	text	Summary of inputs processed
OUTPUT#	text	Summary of output file header info where # = 1,2,3.

The text files produced are for status information only and are really not required beyond this step except for data control purposes. Removal of these text files will not impact subsequent processing.

5.2.2 Combine Gff Files

```
run_gbfcom OBS ROBS* STOBS*
run_gbfcom ATT ATT*
run_gbfcom SV SV*
run_gbfcom OE OE*
run_gbfcom TE TE*
run_gbfcom SN SN*
```

The shell script `run_gbfcom` invokes the program `gbfcom` which combines two gff files into one. `Run_gbfcom` allows for combining any number of gff files into one. Note that because the SN-files are large `run_gbfcom` takes about 3 minutes to combine them into SN. The following files are created:

<u>File</u>	<u>Type</u>	<u>Contents</u>
OBS	gff	Radar and Star Tracker Observations
ATT	gff	Attitude Data
SV	gff	Sensed Velocity Data
OE	gff	Orbiter Ephemeris (Onboard Filter Solution)
TE	gff	Target Ephemeris (Onboard Filter Solution)
SN	gff	SENSOR Program Data

5.2.3 Edit the OBS Per Onboard Data Good Flags

run_obsedit OBS

The onboard sensor systems assign data good flags for each observation datum downlisted. The values of these flags are stored on the OBS file frames. This shell script runs the RELBET fileedit process to edit the OBS file according to the values of these data good flags. Experience has shown that if the data is not good enough for the onboard system then it is not good enough for RELBET. The process takes about 3 minutes to run.

5.2.4 Convert Radar Angles

run_rptost OBS

The shell script **run_rptost** invokes the **rptost** processor which converts the roll-pitch radar angles to shaft-trunnion angles. The process takes about 4 minutes to run.

5.3 DATA PREPARATION

This section describes the various data preparation procedures. One of the major goals of this activity is the creation of the event data base. This data base is stored as a text file and contains information related to mission environment such as burns, attitude maneuvers, data coverage, and anomalous

data. This information is vitally useful in analyzing the filter results by comparing them to a mission event profile.

5.3.1 Check For Data Continuity and Completeness

run_qacover text OBS SV ATT

The shell script **run_qacover** invokes the **qacover** processor on the given gff files to create output text files whose root name is the first argument, "text", given to **run_qacover**. The information contained in the text files created by this processor includes time intervals of data present, data lost, and data edited. The process takes about 10 minutes and creates the following text files.

<u>File</u>	<u>Type</u>	<u>Contents</u>
textaran	text	radar range info from OBS file
textbrnr	text	radar range rate info from OBS file
textcsft	text	radar shaft angle info from OBS file
textdtrn	text	radar trunnion angle info from OBS file
textmrol	text	radar roll angle info from OBS file
textopit	text	radar pitch angle info from OBS file
textizth	text	star tracker horizontal angle info from OBS file
textjztv	text	star tracker vertical angle info from OBS file
textATT	text	attitude info from ATT file
textSV	text	sensed velocity info from SV file

These text files should be reviewed to assure that enough data were retrieved in the downlist reformatting process to continue. In general data loss would have to exceed 3% of the expected amount before attempts to request more data (new CCT requests) are necessary.

5.3.2 Identify Significant Events

run_qadata text OBS SV ATT

The shell script **run_qadata** invokes the following programs:

- qastar** - process star tracker observations from **OBS** file together with the attitude data from **ATT** file to determine inertially whether a star or anomalous data marks recorded
- qasv** - process the sensed velocity file **SV** to determine a timeline of sensed accelerations (or burns)
- qaranjmp** - process the radar range and range rate measurements from the **OBS** file to determine whether inconsistencies between the measurements occur
- qaatt** - process the attitude file **ATT** to determine a timeline of significant angular accelerations

All of the events identified by this process have been found to cause filter problems at one time or another in the past. The purpose of identifying these events beforehand is to have on hand a record of known problem sources to aid in the subsequent filter analysis. The process creates several text files, whose root name is the first argument "text" given on the command line to **run qadata**, and gff output files and takes about 4 minutes.

<u>File</u>	<u>Type</u>	<u>Contents</u>
textstar	text	timeline of anomalous star tracker data (stars or bad marks)
textburn	text	timeline of significant sensed accelerations
textrj	text	timeline of significant inconsistencies between range and range rate radar data
textattr	text	timeline of significant movements about the orbiter rotation axis
ATT_AR	gff	plot file for significant movements about the orbiter rotation axis
SV_SV	gff	plot file for significant sensed accelerations
OBS_ST	gff	plot file inertial el and az of Star Tracker angles

5.3.3 Evaluate Noise on Observation Data

run_noise OBS

This shell script invokes the following programs:

- ddnois** - generate noise tables for each observation type over specified time intervals
- qanois** - selects noise values from the output of **ddnois** and generates text summaries and gff files of the selected noise for each observation type
- obsnois** - place the noise values found on the output gff files of **qanois** onto the frames in the **OBS** file for processing by the filter

Note that the time intervals over which the noise is computed is in general arbitrary. The process creates several text and gff output files and takes about 17 minutes. The gff files except for the **OBS** file (updated in this step) are not used in subsequent steps, however, they could be useful for plots.

<u>File</u>	<u>Type</u>	<u>Contents</u>
textNaran	text	Noise value per time interval for radar range
textNbrnr	text	Noise value per time interval for radar range rate
textNcsft	text	Noise value per time interval for radar shaft angle
textNdtrn	text	Noise value per time interval for radar trunnion angle
textNizth	text	Noise value per time interval for Star Tracker Hor angle
textNjztv	text	Noise value per time interval for Star Tracker Ver angle
OBS_ddnois	gff	Master collection of noise values
OBS__ddnois__Na	gff	Noise value per time interval for radar range
OBS__ddnois__Nb	gff	Noise value per time interval for radar range rate
OBS__ddnois__Nc	gff	Noise value per time interval for radar shaft angle
OBS__ddnois__Nd	gff	Noise value per time interval for radar trunnion angle
OBS__ddnois__Ni	gff	Noise value per time interval for Star Tracker Hor angle
OBS__ddnois__Nj	gff	Noise value per time interval for Star Tracker Ver angle

5.3.4 Create Events Data Base

```
sort -n +0 -o TEXT text*
```

This command is a UNIX system command and is used to merge all of the text files having the prefix 'text' into the following file.

<u>File</u>	<u>Type</u>	<u>Contents</u>
TEXT	text	Current working text data base

5.3.5 Evaluate The Recommended Edits

```
run_search TEXT
```

This shell script reviews the data base for particular events where pre-editing the data before filter processing may be useful and outputs text files containing the time intervals identified. The following files are generated:

<u>File</u>	<u>Type</u>	<u>Contents</u>
edit.sacc	text	Times of anomalous sensed velocity data
edit.attr	text	Times of anomalous attitude data
edit.oatt	text	Times of attitude data loss
edit.star	text	Times of anomalous star tracker data
edit.range	text	Times of anomalous radar data

These text files are inspected to determine whether any of the gff files need to be edited before continuing with the filter process. In this example inspection of **edit.range** reveals that an initial interval of radar data must be edited because it probably represents the onboard performance of the radar self-test function rather than actual observations of the target. This is a typical event in a rendezvous.

In addition the times indicated in **edit.star** are generally used to edit the star tracker data.

5.3.6 Create Time Intervals

```
run_timespan edit.star span.star
```

This shell script reads in the entries extracted from the data base and outputs lines of begin and end times determined from the time tag and duration parameters found on each entry.

Using the UNIX **vi** editor create the file **span.aran** to contain the one line whose begin time is the begin time of radar self-test data and whose end time is the begin time of radar contact with the target. The following two files are produced:

<u>File</u>	<u>Type</u>	<u>Contents</u>
span.star	text	time spans over which to edit star tracker data
span.aran	text	time spans over which to edit radar data

5.3.7 Edit The OBS File

```
run_editradar OBS span.aran  
run_editst OBS span.star
```

These shell scripts invoke the **fileedit** process to edit the radar data during the self-test period and the star tracker data during periods of anomalous observations (stars etc...). The **OBS** file is modified by this procedure which take about 4 minutes total to run.

5.4 BEST ESTIMATE OF TRAJECTORY GENERATION PROCEDURES

The data files needed by the filtering process have now been prepared including the **OBS**, **ATT** and **SV** gff files. The input inputs for the filter must now be prepared. See the article in Appendix III of this manual for details on the inputs required by the filter.

5.4.1 Obtain Initial State For Filtering

```
mkinit -t21022252.38 OE TE
mv MKINIT xxinit
```

Assume that an initial time is selected for processing (usually the begin time of the first valid observation). In this case 21022252.38 seconds relative to the base time on the input gff files OE and TE has been chosen. Note that in general the onboard solution vectors have proven adequate for use as the initial estimate of orbiter and target states.

The output file is **MKINIT** which contains the inertial states for the orbiter and target which occur at or just after the desired time. This text file is in the form needed by the linput input scheme to provide the initial states to the filter. Note it is important to ensure that the attitude data precedes the initial time by at least 60 seconds.

5.4.2 Adjust The Linput Input Files To Reflect Mission Characteristics

This step involves the UNIX **vi** editor to adjust the following key input files.

- xxfile** - to specify the input gff files
- xxgnr1** - to set desired pbug flags
- xxkal** - set desired Kalman filter controls
- xxmas** - set the mass table for the orbiter and target
- xxprop** - set the propagation options desired
- xxscov** - set the initial covariance values (default used)
- xxvcx** - set the cross-sectional areas for orbiter and target
- xxtime** - begin and end times of processing (usually the begin time is the same as the time-tag of the initial state)

5.4.3 Run Kalman Filter

`cat_sfilt > OUTSFILT`

The shell script `cat_sfilt` collects the desired linput input files and pipes them to the `sfilt` processor. The `cat_sfilt` script is tailored for per execution using UNIX `vi`. The `sfilt` program sequentially processes onboard observations to provide an estimate of the relative state vector. This process takes about 60 minutes to run and under the requested `pbug` options outputs the following files.

<u>File</u>	<u>Type</u>	<u>Contents</u>
KSOL	gff	Kalman solution file
KOBS	gff	Kalman pre-edit obs residuals, sigmas, bias solutions, edit status
PRED	gff	Kalman predicted state
KREL	gff	Kalman relative traj solution
OUTSFILT	text	Summary status print of <code>sfilt</code>

5.4.4 QA Kalman Filter Output

The quality of the Kalman filter output must be assured by an expert in evaluating sequential filter processing. The expert has the following procedures and tools available.

5.4.4.1 Gross QA (Evaluate OUTSFILT)

Roughly speaking, the ratios of the residual to the standard deviation of the residual are scanned to confirm convergence of the filter. The edit column in the status record print displays whether or not the observations were processed by displaying an F (false, i.e., no edit) or T (true, i.e., edit). Too many T's indicate filter divergence. If an acceptable number of T's are present, the procedure may proceed.

5.4.4.2 Evaluate The Filter Updates

```
sln2r1 -sKSOL -cKCOV -n22
r1vsr1 PRED KREL
cmp2sg -rPRED_RL -cKCOV -t.5
mv CMP_POS CMP_POS_PK
mv CMP_VEL CMP_VEL_PK
```

This process evaluates the updates to the predicted state performed by the filter processing of observations. Large updates relative to the covariance are documented in the output text files **CMP_POS_PK** and **CMP_VEL_PK**. These periods may be correlated to the **TEXT** data base for probable causes if any. Note the correlation process makes use of the processor **search**. The **sln2r1** processor takes about 6 minutes to run. The following files are output by the above procedure.

<u>File</u>	<u>Type</u>	<u>Contents</u>
KCOV	gff	Kalman computed sigmas in UVW
PRED_RL	gff	UVW difference between predicted and Kalman rel traj states
CMP__POS__PK	text	Sigma ratio comparison between relative position elements of PRED and KREL relative trajectory files
CMP__VEL__PK	text	Sigma ratio comparison between relative velocity elements of PRED and KREL relative trajectory files

5.4.4.3 Evaluate The Filter Edits

```
run_kobsedit KOBS
run_qacover ktext KOBS_ED
sort -n +0 -o KTEXT ktext* CMP_POS_PK CMP_VEL_PK
```

This procedure (which takes about 15 minutes) generates a text data base of the form of **TEXT** called **KTEXT** containing intervals of Kalman edits per observation type as well as the areas of large Kalman updates. This data base can be inspected and correlated to the master data base **TEXT** for probable

causes. Note a large number of Kalman edits occurred on the star tracker vertical angle. This can be seen by performing the following:

```
search -fKTEXT -pjztv -sedit
```

No obvious causes are identified by searching the file **TEXT** for correlations to events and, since the filter recovers from these edit periods, the problem must be reserved for further analysis.

The following files are produced by the above procedure.

<u>File</u>	<u>Type</u>	<u>Contents</u>
KOBS_ED	gff	Copy of KOBS file where the frames are edited whenever a Kalman edit occurs
ktextaran	text	radar range info from KOBS_ED file
ktextbrnr	text	radar range rate info from KOBS_ED file
ktextcsft	text	radar shaft angle info from KOBS_ED file
ktextdtrn	text	radar trunnion angle info from KOBS_ED file
ktextizth	text	star tracker horizontal angle info from KOBS_ED file
ktextjztv	text	star tracker vertical angle info from KOBS_ED file
KTEXT	text	Kalman analysis text data base

5.4.5 Smooth The Kalman Estimate

```
cat_smooth KSOL SSOL > OUTSMOOTH
```

Under the assumption that the above Kalman estimate is satisfactory (subject to analysis of star tracker edit problem), the smooth process is invoked by this run specific shell script. Note that the required input files are identical to those used by **sfilt** in Section 5.4.3.

This process requires a lot of time (about 6-8 hours) primarily because of the inverse matrix computations involved. The following files are produced:

<u>File</u>	<u>Type</u>	<u>Contents</u>
OUTSMOOTH	text	Summary status print of smooth
SSOL	gff	Smooth solution file

5.4.5.1 QA Smoothed Estimate

```
sn2r1 -sSSOL -rSREL -bSBIAS -cSCOV
r1vsr1 KREL SREL
cmp2sg -rKREL_RL -cKCOV
mv CMP_POS CMP_POS_KS
mv CMP_VEL CMP_VEL_KS
```

QA of the smoothed solution is performed by comparing the smooth relative trajectory with the Kalman relative trajectory. These gff files together with the Kalman computed sigmas (**KCOV**) may be plotted to analyze the state differences in UVW coordinates. In general, if the difference is within 3-sigma, then the final solution is acceptable.

This process evaluates the differences between the Kalman and Smooth filter solutions for the relative state. Large differences relative to the Kalman computed covariance are documented in the output text files **CMP_POS_KS** and **CMP_VEL_KS**. These text files provide summary information about when the differences have exceeded the 3-sigma threshold. The anomalous intervals (if any) can be analyzed to determine if a Kalman/smooth iteration is necessary. If no information appears in these files, then the differences were within the 3-sigma limit and the production process may continue. The event data base is specifically provided as a quick guide for solving simple problems of this nature.

Events may be correlated to the TEXT and KTEXT data bases for possible anomaly resolution. The above procedure takes about 10 minutes and produces the following files.

<u>File</u>	<u>Type</u>	<u>Contents</u>
SBIAS	gff	Smooth bias solutions and sigmas
SCOV	gff	Smooth computed sigmas in UVW

SREL	gff	Smooth relative traj solution
KREL__RL	gff	UVW difference between Kalman and smooth relative trajectory solutions
CMP_POS_KS	text	Sigma ratio comparison between relative position elements of KREL and SREL rel traj files
CMP_VEL_KS	text	Sigma ratio comparison between relative velocity elements of KREL and SREL rel traj files

5.5 OUTPUT PRODUCTS GENERATION

This section describes the various procedures for generating the output products.

5.5.1 Prepare Linput Inputs For Prodx

```
search -fTEXT -paran izth -slost > LOST
```

This step involves the UNIX **vi** editor to prepare the key inputs below (see Section 4):

```
xxnflz - specify input and output file names
xxtoff - specify intervals of no obs data, and output options
xxsprm - specify file ids and groups desired
```

Note that the other required inputs are either defaulted or the same as used by previous processors.

To determine the intervals of no available observation data the processor **search** is used to write those intervals for which data gaps are identified to the file **LOST**. This file must be cleaned up slightly to remove those gaps of zero duration. Performing the following yields the time spans desired which are then entered into the file **xxtoff** using **vi**.

```
run_timespan LOST > span.LOST
```

5.5.2 Ancillary Products Processor

`cat_prodx > OUTPRODX`

The `cat_prodx` script collects the desired input files and pipes them to the `prodx` processor which takes about 30 minutes to run. The script is tailored to this execution and is created using the UNIX `vi`. The `prodx` program computes the required parameters for the Ancillary Data Products and generates the following files. Note that before executing the script a tape is required on tape drive 0 as specified in `drive/xxtoff`.

<u>File</u>	<u>Type</u>	<u>Contents</u>
SP51I_BIN	binary	Special Products binary file
SP51I_PRT	text	Special Products print file
SPtape	tape	Special Product Tape

5.5.3 QA Output Product Tape

`run_qatape QA_SP51I_PRT 0`

This shell script invokes the `qatape` processor and requires about 30 minutes to run. The SP tape generated above must be mounted onto tape drive 0 which is the second argument to the script. The output file contains the first and last records of the product tape and may be compared to the file `SP51I_PRT` for spot checking the parameter values and number of records written.

<u>File</u>	<u>Type</u>	<u>Contents</u>
QA_SP51I_PRT	text	Special Products print file

5.5.4 Generate Microfiche

`fiche SP51I_PRT 660bph 0`

This program reformats the print file to tape in order to obtain microfiche listings of the product. Six UNIVAC tapes are produced and labelled as shown on Figure 5.5.4.1. and delivered to Building 12 accompanied by the form on Figure 5.5.4.2.

<u>File</u>	<u>Type</u>	<u>Contents</u>
SPfichetape1	tape	Tape 1 of microfiche format of special products
SPfichetape2	tape	Tape 2 of microfiche format of special products
SPfichetape3	tape	Tape 3 of microfiche format of special products
SPfichetape4	tape	Tape 4 of microfiche format of special products
SPfichetape5	tape	Tape 5 of microfiche format of special products
SPfichetape6	tape	Tape 6 of microfiche format of special products

5.5.5 Sensor Tape Output

The RELBET **stop** program has provided the link between the shuttle downlist process and the bench version of the rendezvous nav program called SENSOR. Currently a UNIVAC format SIT/SET tapes may be obtained only by running the appropriate RELBET 2.0 processors on the UNIVAC, however the HP **stop** program provides an HP binary version of the same data in the same format (HP) which could probably be reformatted into UNIVAC format with little difficulty. In the following procedures two different versions of SIT/SET files are produced. One provides only downlisted data and the other replaces the onboard solution state vectors with the RELBET solution states.

```
*****
**TAPE WILL BE RETAINED MAX 21 DAYS**
*****
**TAPE NAME          PROCESSOR**
**FICH1              P      **
**USER NAME          PHONE NO.**
**B. HUYSMAN         333-3133**
**BOX NO. 660       **
**RETURN TO USER DATE: 12 / 6/ 86 **
*****
```

```
*****
**TAPE WILL BE RETAINED MAX 21 DAYS**
*****
**TAPE NAME          PROCESSOR**
**FICH2              P      **
**USER NAME          PHONE NO.**
**B. HUYSMAN         333-3133**
**BOX NO. 660       **
**RETURN TO USER DATE: 12 / 6/ 86 **
*****
```

```
*****
**TAPE WILL BE RETAINED MAX 21 DAYS**
*****
**TAPE NAME          PROCESSOR**
**FICH3              P      **
**USER NAME          PHONE NO.**
**B. HUYSMAN         333-3133**
**BOX NO. 660       **
**RETURN TO USER DATE: 12 / 6/ 86 **
*****
```

```
*****
**TAPE WILL BE RETAINED MAX 21 DAYS**
*****
**TAPE NAME          PROCESSOR**
**FICH4              P      **
**USER NAME          PHONE NO.**
**B. HUYSMAN         333-3133**
**BOX NO. 660       **
**RETURN TO USER DATE: 12 / 6/ 86 **
*****
```

```
*****
**TAPE WILL BE RETAINED MAX 21 DAYS**
*****
**TAPE NAME          PROCESSOR**
**FICH5              P      **
**USER NAME          PHONE NO.**
**B. HUYSMAN         333-3133**
**BOX NO. 660       **
**RETURN TO USER DATE: 12 / 6/ 86 **
*****
```

Figure 5.5.4.1. Labels for Microfiche Tapes

REQUEST FOR PERIPHERAL PROCESSING (Submit in duplicate)																																																							
PROGRAMMER B. Huysman			BOX NUMBER 660	DIV. CODE FM7	PHONE NO. 333-3133	PROJ. NUMBER 1404	DATE 12/5/86	SEQ. NUMBER																																															
REEL NUMBER	MAGNETIC LABEL (YES OR NO)	COPY	DENS	MODE	SAVE	COPY	NO. OF FILES	TRACKS		DUMP	PRINT	PRINT		CARD/TAPE		COMMENTS																																							
								7	8			BCD	Data	SGS	Non		Other																																						
FICH1	No	X	1600				1	X																																															
FICH2	No	X	1600				1	X																																															
FICH3	No	X	1600				1	X																																															
FICH4	No	X	1600				1	X																																															
FICH5	No	X	1600				1	X																																															
<table border="1" style="width:100%; border-collapse: collapse;"> <tr> <td><input type="checkbox"/> PUNCH</td> <td><input type="checkbox"/> PRINTER</td> <td><input type="checkbox"/> FRBD</td> <td><input checked="" type="checkbox"/> AUTO-COM</td> <td><input type="checkbox"/> CAL-COMP</td> <td><input type="checkbox"/> XEROX</td> <td colspan="11" rowspan="6" style="vertical-align: top;">OPERATOR COMMENTS:</td> </tr> <tr> <td>CARD FORM</td> <td>CARRIAGE TAPE <input type="checkbox"/> STANDARD <input type="checkbox"/> SPECIAL</td> <td><input type="checkbox"/> FILM</td> <td><input type="checkbox"/> FILM</td> <td>PEN 02 03 04 05</td> <td>OVERLAY <input type="checkbox"/> NO-OVERLAY <input checked="" type="checkbox"/></td> </tr> <tr> <td>NO. OF CARDS</td> <td>PROGRAM</td> <td>PROGRAM</td> <td>INK BALL POINT</td> <td>PROGRAM <input type="checkbox"/> CYBER <input type="checkbox"/> UNIVAC <input type="checkbox"/> IBM</td> </tr> <tr> <td>SIZE PAPER</td> <td>REDUCTION RATIO 42X <input type="checkbox"/> 24X <input type="checkbox"/></td> <td>OVERLAY <input type="checkbox"/> NO OVERLAY <input type="checkbox"/></td> <td>FORM 00 01 02 OTHER _____</td> <td>SPECIAL FORM NO.</td> </tr> <tr> <td>NO. OF COPIES</td> <td>16MM <input type="checkbox"/> 35MM <input type="checkbox"/> 105MM <input type="checkbox"/></td> <td>REDUCTION RATIO 24X <input type="checkbox"/> 42X <input type="checkbox"/> 48X <input type="checkbox"/></td> <td><input type="checkbox"/> STANDARD SET UP</td> </tr> <tr> <td>NO. OF PAGES</td> <td>FORM SLIDE NUMBER</td> </tr> </table>																	<input type="checkbox"/> PUNCH	<input type="checkbox"/> PRINTER	<input type="checkbox"/> FRBD	<input checked="" type="checkbox"/> AUTO-COM	<input type="checkbox"/> CAL-COMP	<input type="checkbox"/> XEROX	OPERATOR COMMENTS:											CARD FORM	CARRIAGE TAPE <input type="checkbox"/> STANDARD <input type="checkbox"/> SPECIAL	<input type="checkbox"/> FILM	<input type="checkbox"/> FILM	PEN 02 03 04 05	OVERLAY <input type="checkbox"/> NO-OVERLAY <input checked="" type="checkbox"/>	NO. OF CARDS	PROGRAM	PROGRAM	INK BALL POINT	PROGRAM <input type="checkbox"/> CYBER <input type="checkbox"/> UNIVAC <input type="checkbox"/> IBM	SIZE PAPER	REDUCTION RATIO 42X <input type="checkbox"/> 24X <input type="checkbox"/>	OVERLAY <input type="checkbox"/> NO OVERLAY <input type="checkbox"/>	FORM 00 01 02 OTHER _____	SPECIAL FORM NO.	NO. OF COPIES	16MM <input type="checkbox"/> 35MM <input type="checkbox"/> 105MM <input type="checkbox"/>	REDUCTION RATIO 24X <input type="checkbox"/> 42X <input type="checkbox"/> 48X <input type="checkbox"/>	<input type="checkbox"/> STANDARD SET UP	NO. OF PAGES	FORM SLIDE NUMBER
<input type="checkbox"/> PUNCH	<input type="checkbox"/> PRINTER	<input type="checkbox"/> FRBD	<input checked="" type="checkbox"/> AUTO-COM	<input type="checkbox"/> CAL-COMP	<input type="checkbox"/> XEROX	OPERATOR COMMENTS:																																																	
CARD FORM	CARRIAGE TAPE <input type="checkbox"/> STANDARD <input type="checkbox"/> SPECIAL	<input type="checkbox"/> FILM	<input type="checkbox"/> FILM	PEN 02 03 04 05	OVERLAY <input type="checkbox"/> NO-OVERLAY <input checked="" type="checkbox"/>																																																		
NO. OF CARDS	PROGRAM	PROGRAM	INK BALL POINT	PROGRAM <input type="checkbox"/> CYBER <input type="checkbox"/> UNIVAC <input type="checkbox"/> IBM																																																			
SIZE PAPER	REDUCTION RATIO 42X <input type="checkbox"/> 24X <input type="checkbox"/>	OVERLAY <input type="checkbox"/> NO OVERLAY <input type="checkbox"/>	FORM 00 01 02 OTHER _____	SPECIAL FORM NO.																																																			
NO. OF COPIES	16MM <input type="checkbox"/> 35MM <input type="checkbox"/> 105MM <input type="checkbox"/>	REDUCTION RATIO 24X <input type="checkbox"/> 42X <input type="checkbox"/> 48X <input type="checkbox"/>	<input type="checkbox"/> STANDARD SET UP																																																				
NO. OF PAGES	FORM SLIDE NUMBER																																																						
PROGRAMMER INSTRUCTIONS: RUN PROGRAM HAR9TK																																																							
											TIME IN		TIME OUT																																										
OPERATOR																																																							

Figure 5.5.4.2. Form to Request Microfiche

5.5.5.1 Generate SIT and SET

```
eph2rel OE TE
stop -rOE_REL -sSN
mv SIT SITO
mv SET SETO
stop -rSREL -sSN
mv SIT SITR
mv SET SETR
```

The `stop` process requires a relative trajectory file as input so that the `eph2rel` processor is run. Thus, the onboard relative trajectory appears on the file `OE_REL` output by `eph2rel`.

The outputs of `stop` are `SIT` and `SET` and these are saved to avoid losing them to subsequent `stop` processing. Each run of `stop` takes about 2 minutes and the above procedure generates the following files.

<u>File</u>	<u>Type</u>	<u>Contents</u>
<code>OE_REL</code>	<code>gff</code>	Onboard relative trajectory
<code>SITO</code>	<code>binary</code>	<code>SIT</code> with downlist data
<code>SETO</code>	<code>binary</code>	<code>SET</code> with downlist data
<code>SITR</code>	<code>binary</code>	<code>SIT</code> with RELBET data
<code>SETR</code>	<code>binary</code>	<code>SET</code> with RELBET data

5.5.3.2 QA SIT/SET Files

```
read_sit -fSITO > QA_SITO
read_set -fSETO > QA_SETO
read_sit -fSITR > QA_SITR
read_set -fSETR > QA_SETR
```

QA of the `stop` outputs consists of reading the binary files and generating printed output of the files contents at a specified frequency. These executions utilize the default frequency which outputs only the first and last records of the input file. The output is reviewed to verify that the correct number of records have been written and to spot check the parameter values.

APPENDIX I - PROGRAM MANUALS

APPENDIX I - PROGRAM MANUALS

The program manuals presented here describe the user inputs and execution procedures for each processor including shell scripts in the RELBET System. The "manual entries" follow the standard UNIX format. They contain the following sections as appropriate.

NAME: Names of all externally accessible identifiers followed by a brief description of the package.

SYNOPSIS: A quick summary of how to invoke the programs parameters. Includes arguments.

DESCRIPTION: A functional description of what the programs do and what the options are.

OPTIONS: Description of the options when they are not suitable for inclusion in the DESCRIPTION.

FILES: The files used or assumed by the application.

EXAMPLE: Annotated examples of how to use the application.

COMMENTS: Miscellaneous comments. For example, rationales for the design or functions may appear here.

BUGS: Known problems.

DIAGNOSTIC: Warning and error messages, debug options.

SEE ALSO: References to related applications.

AUTHOR: The name of the responsible programmer.

ascale(1)

(RELBET)

ascale(1)

NAME

ascale - automatic scaling processor

SYNOPSIS

ascale< Input_file

DESCRIPTION

The program ascale is part of the Graphic Display Process. Two RELBET processors are involved in this process including plotx for the actual plotting and ascale to perform the automatic scaling function. Input is via linput input blocks. Note that the same input is used for program plotx. Output file is pmmx.

Use the following input blocks; see description in Appendix II

- xxgnr1 (optional)
- xxtime (mandatory)
- xxmisc (optional)
- xxqgen (mandatory)
- xxqprm (mandatory)
- xxpmmx (mandatory)
- xxqcrv (mandatory)
- xxgraf (mandatory)

For more details, see article on GRAPHIC DISPLAY PROCESS (Appendix III)

NAME

cmp2sg - compare relative difference to computed sigmas

SYNOPSIS

cmp2sg [options]

DESCRIPTION

cmp2sg reads in two files including a relative trajectory difference file and a file containing the computed sigmas associated with some filtering process in the same frame format as a relative trajectory file. cmp2sg computes ratios from the magnitude of relative position and velocity differences to the magnitude of the relative position and velocity sigmas and compares the results to the thresholds. If the threshold is exceeded, then an interval of excessive sigma ratios is accumulated and output at the end of the interval (whenever the ratios go below the threshold). The output is one line of text for each interval including start time, "FILTER", either "Pos_Peaks" or "Vel_Peaks", duration, and average magnitude of the position or velocity sigma over the interval. The following list describes the options. Unless the default is specified, the input is mandatory.

-rtrajfile
indicates file name of relative traj file

-ssigfile
indicates file name of sigma data file

-tnumber
is the threshold of comparison (default is 3 indicating that 3-sigma is a good comparison)

FILES

input

trajfile
LVW referenced difference between two relative trajectory files in gff format

sigfile
U/W referenced sigmas associated with the base state which acts as the origin of the relative trajectory difference UVW frame at each time

output

CMP_POS
text file of position information

CMP_VEL
text file of velocity information

ddnois(1)

(RELBET)

ddnois(1)

NAME

ddnois - computes noise characteristics of obs

SYNOPSIS

ddnois < input_file >text_output

DESCRIPTION

Ddnois processes an input observation file to produce time-ordered estimates of the noise associated with each observation type requested. The algorithm used to produce these estimates is based on computing divided differences over a sequence of observations. Associated with each obs type requested is the length of a time interval. Starting at the begin time the data for each obs type is partitioned by these time intervals and the data within each interval make up the sequence over which the noise is computed. Up to 10th order differences are computed for each sequence and in general the noise representing the interval is the value after which the difference between successive differences becomes negligible (usually occurs at 5th or 6th order).

Use the following input blocks; see description in Appendix II

xxgnr1 (optional)
xxtime (mandatory)
xxnnois (mandatory)

For more detail, see article on NOISE ANALYSIS (Appendix III)

FILES

input

input_file
text file containing the input blocks stated above

obsfile

observation file in gff format

output

stdout

records containing time interval, frame id, average value of obs over time interval, number of points in time interval, and ten variate difference noise values for that interval

Noisefile

gff file whose name is specified in xxnnois and its frame contains time-tag (or begin time of interval), obs id (aran, brnr, etc...), edit (or rather number of points in interval), data1 (or end time of noise interval), data2 (or average value of observation in that interval), and data3 - data13 (or the ten variate difference noise values).

dwnfmt(1)

(RELBET)

dwnfmt(1)

NAME

dwnfmt - downlist formatter

SYNOPSIS

dwnfmt < Input_file > Output_file

DESCRIPTION

The Downlist formatter strips required parameters from the standard orbiter downlist Computer Compatible Tape (CCT), performs transmission validity checks and necessary unit conversions, and generates various internal RELBET files.

Files BUGS and OUTPUT are also created at execution level to contain debug and additional summary print.

Use the following input blocks; see description in Appendix II

- xxgnr1 (mandatory)
- xxtime (mandatory)
- xxmisc (optional)
- xxmast (mandatory)
- dwnfmt_files (mandatory)
- dwnfmt_msids (mandatory)
- xxcct (mandatory)
- dwnfmt_cct (mandatory)

Note : Input_file is a text file containing the linut input blocks stated above.

For more details, see article on DOWNLIST FORMATTER (Appendix III)

eph2rel(1)

(RELBET)

eph2rel(1)

NAME

eph2rel - reads two ephemeris files and creates relative trajectory file

SYNOPSIS

eph2rel ephemeris_file1 ephemeris_file2

DESCRIPTION

eph2rel reads two ephemeris files and creates a relative trajectory file as output. The first ephemeris_file is the base file name. The second ephemeris_file is the file to be interpolated for the relative portion of the frame. The output relative trajectory file is assigned the name ephemeris_file1_REL. For more detail refer to the RELBET USERS MANUAL Section 5.5.1.

Input :

ephemeris_file1 - ggf ephemeris file for base state
ephemeris_file2 - ggf ephemeris file for target state

Output :

<ephemeris_file1>_REL - ggf relative trajectory file

NAME

fiche - microfiche print formatter

SYNOPSIS

fiche file run_id tape_drive

DESCRIPTION

The program fiche generates a tape which can be read by the fiche reader. The format of the tape is 9-track, 1600 bpi, ASCII, 132 character fixed length records. It can generate a multi-reel file. However, the fiche processor will treat each tape as an individual file. It requires three user inputs which are put on the command line.

usage : fiche file run_id tape_drive where
file - name of print file to put on the tape
runid - Univac type runid 660???
tape_drive - 0 or 1

The tape can be processed by a setup called HAR9TK. The normal procedure is to fill out a Peripheral Processing Request form and turn it in at the Work Control window in Building 12 along with the tape to be processed (currently Oct 1986).

Note: user needs to check AUTO-COM , and for program instructions
RUN PROGRAM HAR9TK

Note: If more than one tape is needed, the program fiche will prompt user for addition tape.

1. rewind tape manually and take off tape from tape drive
2. mount new tape manually and enter 0 or 1 for tape drive to continue

NAME

fileedit - gff file editor

SYNOPSIS

fileedit [options]

DESCRIPTION

fileedit reads in one of two file sources to be used to change the edit status value of certain frames appearing on a specified gff file (-o option) depending upon certain program dependent criterion. The file sources can be a reference gff file (-f option) or a text file (-t option) with time information. In this discussion to edit a frame means to apply a user specified set of tests to a reference frame and if all of the tests are successful then to place the user specified edit value (either positive or negative) into the edit status portion of the frame to be edited. Thus the user may either edit (declare a frame invalid) or unedit (declare a frame valid) as a result of successful testing. Multiple frame ids can be specified to indicate the frames to be edited. The following list describes the options. Unless a default is indicated, the option is mandatory.

-anumber

number provides the edit value to be applied to the edit field. Zero is the same as default. Negative input indicates edited frames are to be unedited. Positive input indicates unedited frames are to be edited. When used with -c indicates that only frames bearing the specified edit value are effected. Default is 999.

-ttextfile

Indicates file name wherein edit intervals are specified by two text fields. The first field should be begin time; the second field is an end time of edit interval. Frames associated with specified (<p_>) frame_ids are edited within text input time intervals. The default is to process the whole file.

-c

Indicates that the edit process will change the edit status of all frames possessing the desired edit value input using the <-a> option or defaulted to 999. For example in the default case frames with 999 receive -999 if the proper edit criteria are met. In this usage negative values may be used with <-a>. The object is to undo a previous edit operation or redo a previous operation. This option can be used with both <-t> and <-f> options and has the same purpose.

-ogfffile

Indicates file name of gff file to be edited. If unused or same as the reference file name (option <-f>) then the reference file itself is to be edited.

-pframeidlist

Indicates list of frame_ids separated by blanks to be effected. Default is to edit all frame ids.

-frefgfffile

Indicates file name of reference gff file whose frames are used in the edit determination of each input file frame. If the <-o> is unused or

same as the reference file name then the reference file itself is to be edited.

-gframeid

Indicates frame_id source of edit criterion on the reference file to be considered. Only one frame_id is allowed. Ignored if the reference file itself is being edited. See <-f>.

-hnumber

Number indicates threshold to be compared to the reference frame desired data value to determine whether to edit or to unedit. This is necessary only if -i__ option is specified in which case a double precision value is expected which is treated as a sort of threshold

-iinteger

Integer indicates the position on the reference frame where the data value to be compared to the threshold <-h> is to be found. If not specified then the edit status portion of the reference frame is used for edit determination. ie., if the reference frame is edited then the frame is edited.

-jinteger

integer indicates the type of test used to compare input threshold with frame info as criterion for edit. Edit is performed if the following tests are true.

=0: abs(frame info) >= abs(threshold)

>0: frame info >= threshold

<0: frame info <= threshold

ignored if -i option not used

NOTE: if the frame being tested was a valid frame before the test then the results of the test will change the value of the edit status flag in some cases even if the result shows valid data, however in this case the frame will still be valid.

FILES

input

gff gff file to be edited named by -o option

ref_gff

reference gff file used for edit criterion named by -f option

text text file defining edit intervals named by -t option

output

gff_ED

output gff with name of input file suffixed with _ED

EXAMPLE

The following examples demonstrate the execution of filedit. Note that all output gff files are called Obs_ED.

filedit -fObs_ref -oObs

This option simply applies the edit status of Obs_ref to frames in Obs occurring at the same time or just after.

filedit -oObs -a-111

NOTE: default option if no reference file provided. This option changes all edited frames to unedited frames by placing a 111 in the edit status value of all edited frames

filedit -fObs -h0.5 -i3 -j-1

NOTE: default option if no gff file to be edited is provided. This configuration assumes that the specified reference file itself is to be edited, i.e., as if -oObs were specified. This option evaluates the third datum value (d) in each frame and performs the comparison according to the evaluation option -1 (negative) to see whether $d \leq 0.5$.. if true frame is edited if not true frame is unedited.

filedit -fObs_ref -oObs -paran brnr -h50.0 -i2 -garan

This option reevaluates all aran and brnr frames on Obs based on aran frame on Obs_ref irregardless of edit status of either reference or target frames. The range value (rv) of reference frame which is in position 2 on the data portion of frame is checked according to the default evaluation option to see whether $\text{abs}(rv) \geq \text{abs}(50.0)$. If true, frame is edited with -999; if not true, frame is unedited with 999.

filedit -fObs_ref -oObs -paran brnr -h50.0 -i2 -garan -j1

The range value (rv) of reference frame is checked according to the evaluation option 1 (positive) to see whether $rv \geq 50.0$., if true frame is edited; if not true frame is unedited.

filedit -fObs_ref -oObs -paran brnr -h50.0 -i2 -garan -j-1

The range value (rv) of reference frame is checked according to the evaluation option -1 (negative) to see whether $rv \leq 50.0$., if true frame is edited if not true frame is unedited.

filedit -fObs_ref -oObs -paran brnr -h50.0 -i2 -garan -j-1 -a111

The option <-a111> causes only valid reference frames to be considered and applies an edit value of -111 to edited frames and 111 to unedited frames as discussed above.

filedit -oObs

This option edits all valid frames in file Obs_text

filedit -tObs_text -oObs

This option edits all valid frames in intervals specified on file Obs_text

filedit -tObs_text -oObs -pmrol opit

This option edits all valid mrol and opit frames in intervals specified on file Obs_text

filedit -oObs -a-999

This option unedits all frames which are currently edited Output saved

filedit(1)

(RELBET)

filedit(1)

on file Obs_ED.

filedit -oObs_ED -a999 -c

This option edits all frames exhibiting edit values of 999

filedit -oObs_EDtop -a-999 -c -pmr01

This option unedits all mrol frames exhibiting edit values of -999

NAME

gbfcom - merge files

SYNOPSIS

gbfcom output input1 input2

DESCRIPTION

gbfcom reads two files and performs a merge using the first input file as a base file and the second file as a merge file ie., if two frames with identical frame ids occur at the same time then the base file frame is discarded and the merge file frame is used. The new file is created in the first argument.

NAME

gdisp - generic display

SYNOPSIS

gdisp (respond to prompts)

DESCRIPTION

The program gdisp provides a display of any RELBET internal files. The user is provided with a variety of options.

```
***** **
menu generic display
```

```
d)isplay      f)ormat      b)asdate      h)elp      q)uit      #)abort
*****
```

The options are:

d)isplay

The user opens and sets the current file. In addition, the user is prompted for the record and number of frames to display in one input, each separated by a comma. Note that the user may move 'forwards' or for the 'number of frames' entry. The user can implement a 'help' mode at this point by entering a negative number, positive number respectively (e.g. -1,1). One can exit the display option by entering 2*-1, or -1,-1.

f)ormat

Allows the user to choose a preferred output from four different numeric formats, namely: 0 = f18.8, 1 = d25.17, 2 = d15.7 and 3 = f12.4, two print formats, namely: 80 columns or 130 columns, and lines/page if paging is desired with user specified number of lines per page.

b)asdate

Allows the user to change the basedate. The input is in the order of : year, month, day, hour, minutes, seconds. Note that each entry is separated by a comma (e.g. 1984,11,8,5,4,3).

h)elp

Display information about each option.

q)uit

exit the program.

#)abort

similar to quit.

NAME

mkinit - creates xxinit text file from two traj files or a rel traj file

SYNOPSIS

mkinit [options] files

DESCRIPTION

mkinit

mkinit reads two trajectory files or one relative trajectory file and writes out text to be used for xxinit linput input containing base state, target state and a time tag.

Usage: mkinit -t__ file_name target_eph_name

-t optional desired time of output state, if not input assume start time of the file

FILES

input

file_name
relative trajectory file or base trajectory file

target_eph_name
second trajectory file if first file is not a relative trajectory file.

output

MKINIT

text file in format for linput input block xxinit

NAME

mktape - UNIVAC FORTRAN tape formatter

SYNOPSIS

mktape (respond to prompts)

DESCRIPTION

This program prompts user for inputs required to process an hp9000 created binary file with mixed type data words. Output is a univac FORTRAN V readable tape in the same mixed order.

INPUT ITEMS

Input to the program mktape consists of the following:

1. Tape drive number
2. Source of HP-9000 filename - up to 32 characters
3. Sequence of type code one by one (eg. hit return key after each input)
4. Number of records to process

EXECUTION PROCEDURE

1. Mount the output tape to the tape drive selected.
2. Type in mktape
3. Answer prompts.
4. Dismount the output tape.

EXAMPLE

The following shows an actual example of user inputs for the program mktape and its prompts for the Special Product. The number of double precision data words in an ADP output record is 84 plus 1(for continue flag). The dictionary has twice this number of character data words. Each character data word contains four bytes(four characters). The prompt is indicated by double quotes (").

"Enter the tape drive number to user(0 or 1)"

0 output tape drive number

"Please enter the source hp file name now"

SP51D input disk filename

"Assume type code .0 for integer, .4 for real*4, .6 for char*4, or .8 for real*8. Input the sequence of #.type and RETURN until the end of sequence is terminated by the entry 0.0"

13.6 number of character variable word(13) in header record,
 decimal point, indicator for character data(6).

13.0 number of integer variable words(13) in header record,
 decimal point, indicator for integer data(0).

0.0 end of record designator.

"Inp no.recs. to process or a negative number to process to end of file or
zero to terminate now."

1 number of header records to read using the format.

"Assume type code .0 for integer, .4 for real*4, .6 for char*4, or .8 for
real*8. Input the sequence of #.type and RETURN until the end of sequence
is terminated by the entry 0.0"

2.0 number of integer words which begins the dictionary
 record(2), decimal point, indicator for integer data(0).

168.6 number of character variable words in the dictionary
 record(168), decimal point, indicator for character data(6).

0.0 end of record designator.

"Inp no. of recs. to process or a negative number to process to end of file
or zero to terminator now."

1 number of dictionary records to read using the above format.

"Assume type code .0 for integer, .4 for real*4, .6 for char*4, or .8 for
real*8. Input the sequence of #.type and RETURN until the end of sequence
is terminated by the entry 0.0"

85.8 number of double precision words in the data record(85),
 decimal point, indicator for double precision data(8).

0.0 end of record designator

"Inp no. of recs. to process or a negative number to process to end of file
or zero to terminator now."

-1000 any negative number will do. It is used to tells the
 program to process all remaining records in this file,
 rather than using a fixed number of records.

MESSAGES

Possible messages are:

1. Error open tape drive
2. Illegal type specified .not 0, 4, 6, 8
3. Error on input; proceed as if ok

The meanings of the above messages are self-explanatory.

NAME

obsnois - merges obs data with noise data

SYNOPSIS

obsnois obsfile noisefile

DESCRIPTION

obsnois merges observation data and noise data into one file. The noise data on the output file is stored in the first location of obs data. The output file is a binary file with _ON attached to the first input file (obsfile). If any of the input files is unavailable, an error message is generated.

Note: User needs to run obsnois repeatedly for each frame id until all of the desirable types are completed. The output file is the input file for the next run. For example:

```
obsnois f8obs Obs_ddnois_Na
```

The output file is f8obs_ON. For the next run, we will have

```
obsnois f8obs_ON Obs_ddnois_Nb
```

Now the output file is f8obs_ON_ON.

FILES

input

obsfile observation file in gff format, name must be first argument

noisefile noise file in gff format, name must be second argument gff file whose frame contains time-tag (or begin time of interval), obs id (aran, prnr, etc...), edit (or rather number of points in interval), data1 (or end time of noise interval), data2 (or average value of observation in that interval), and data3 selected noise representing the current time interval.

output

obsfile_ON observation file in gff format with noise data in the first data slot

plotx(1)

(RELBET)

plotx(1)

NAME

plotx - graphic display

SYNOPSIS

plotx < Input_file

DESCRIPTION

The program plotx is part of the Graphic Display Process. Two RELBET processors are involved in this process including plotx for the actual plotting and ascale to perform the automatic scaling function. The program plotx provides graphic display of data from gff files. Input is via linput input blocks. The user has available a number of formatting options to tailor a display to his/her particular needs as well as a set of canned "default" options. These default provide as plot configuration that may be used as is or else adapted to a particular application.

Use the following input blocks; see description in Appendix II

xxgnr1 (optional)
xxtime (mandatory)
xxmisc (optional)
xxqgen (mandatory)
xxqprm (mandatory)
xxpmmx (mandatory)
xxqcrv (mandatory)
xxgraf (mandatory)

Excution Procedure for Graphic Display Process

1. Run program ascale to obtain pmmx min/max values for x and y axis
2. Cat pmmx Input_file >in_plotx
3. Run program plotx for plotting

plotx < in_plotx

For more details, see article on GRAPHIC DISPLAY PROCESS(Appendix III)

Note: Input_file is a text file containing the linput input blocks stated above.

prodx(1)

(RELBET)

prodx(1)

NAME

prodx - Ancillary Products Processor

SYNOPSIS

prodx < input >output

DESCRIPTION

The Special Products Processor utilizes trajectory and attitude information to produce data product files containing various parameters over a specified time interval. Output includes print describing the input and processing status. Options include both binary (HP and UNIVAC) and formatted data product files. Use the following input blocks; see description in Appendix II.

- xxgnr1 - (optional)
- xxtime - (mandatory)
- xxmisc - (optional)
- xxcon - (optional)
- xxnflz - (mandatory)
- xxtoff - (mandatory)
- xxsprm - (mandatory)

For more details, see article on SPECIAL PRODUCT (Appendix III).

Note: input is a text file containing the input input blocks stated above.

qaatt(1)

(RELBET)

qaatt(1)

NAME

qaatt - Generates a summary of angular acceleration events .

SYNOPSIS

qaatt [flags] file

DESCRIPTION

qaatt provides output to stdout documenting attitude files. The following optional flags allow for the changing of constants.

-enumber is the maximum acceptable angular acceleration magnitude (default .1)

-bnumber is the minimum angular acceleration magnitude (default .0005)

The options are parsed and the appropriate variables are set. If the file is unavailable, an error message is generated. qaatt creates an output gff with the angular acceleration vector and magnitude for each point between the two input angular rates; the frame id for this file is "attr." Also, for any point > e and any point > b, a record is written to stdout containing time, "attr", "valid" or "edit", average magnitude, number of points in the interval, and delta time. accumulated.

Example of printed output:

1000.00	attr	valid	3.8400000	0.001	1
1003.84	attr	edit	3.8400000	2.010	1
1100.00	attr	valid	3.8400000	0.012	1

Interpretation:

A significant change in angular rate occurred at 1000.0 for a duration of 3.84 seconds at a magnitude of .001 and included 1 point. A bad attitude occurred at 1003.84 for a duration of 3.84 seconds at a magnitude of 2.01. Another significant attitude change occurred at 1100.00 for 3.84 seconds with a magnitude of .012.

FILES

input

ATT RELBET attitude file in gff format

output

stdout a record for each interval containing time, "attr", "valid" or "edit", delta time of the interval, average magnitude, and number of points in the interval.

ATT_AR file in gff format containing records with time, frame id "attr", vector angular acceleration vector, and the magnitude of the acceleration, for each point between the thresholds.

qacover(1)

(RELBET)

qacover(1)

NAME

qacover - Generates a coverage history summary

SYNOPSIS

qacover [options] list of files

DESCRIPTION

qacover provides output to stdout summarizing the coverage of a specified data type from a gff file. The following optional flags allow the user to change default values.

-tnumber is the minimum time gap for lost data(default 30seconds)

-fframeid frame id (default no check on frame id)

The options are parsed and the appropriate variables are set. The files are parsed and output text is written until the last file has been processed. If a file is unavailable, an error message is generated and the next file is processed. The output text contains the start time of the interval, the frame id, data indicator (valid,lost or edit), the duration of the interval and the number of points in the interval.

Example:

1000.0	aran	edit	1000.00	300
2000.0	aran	lost	500.00	1
2500.0	aran	valid	5.65	2
2505.65	aran	edit	3.84	1
2509.49	aran	lost	0.00	1

Interpretation:

At 1000.0 the range observation had a edit flag for 300 points with a duration of 1000 seconds. At 2000.0 there was a gap in the data. It picked up at 2500.0 and was good for 2 points, the next point had an edit flag, then the data ended.

FILES

input

file1,.... list of RELBET gff files

output

stdout records contain start time of the interval, frame id, data indicator (valid, lost or edit), the duration of the interval and the number of points in the interval.

NAME

qanois - generate text file for noise statistics and binary file for graphic

SYNOPSIS

qanois [options] file

DESCRIPTION

qanois provides output to stdout documenting noise statistics for obs gff file and binary file for graphic. The input is from ddnois. The following is a list of options, unless a default is provided, the option is mandatory.

-fframeid is the frame id

-inumber number is the index to difference table to use for noise source(default 6th number)

-qnumber number is the quantization value default see above

-mnumber number is the minimum number of points acceptable for noise consideration (default see above)

The options are parsed and the appropriate variables are set. If a file is unavailable, an error message is generated. The output text has the following format: Begin_time,Data_type,"noise",duration,quantized value, average_obs measurement,and number_of points.

Example:

1000.0	aran	noise	384.0	.01	50.0	90
1384.0	aran	noise	768.0	.03	90.0	180

Interpretation: At 1000.0 seconds Range data noise average .01 meters for 384.0 seconds with an average range measurement of 50 meters and a total of 90 observation marks etc...

The output binary file uses the following naming convention : Inputfile_name plus suffix _N plus first letter of the frame id desired. The frame of the output binary file is described as follows: Standard header where edit status indicates number of points in the noise interval; the data portion of frame provides the following information: End time of noise interval, Average value of observation, and Computed noise selected by Index.

Note: User needs to run qanois repeatedly for each frame id until all of the desirable types are finished.

FILES

input

qanois(1)

(RELBET)

qanois(1)

OBS observation file in gff format

output

stdout records containing time, frame id, "noise", duration, quantized value, average obs measurement, and number of points.

OBS_N[a-p] gff file for plotting, containing the end time of noise interval, average value of observation and computed noise selected by index. The suffix [a-p] is determined by the frame id requested.

NAME

qaranjmp - Generates a radar range versus range rate difference history

SYNOPSIS

qaranjmp [options] file

DESCRIPTION

qaranjmp provides output to stdout documenting jumps in range compared with range computed from range rate. The following intervals are documented.

```

If range is <= Rbound; 15 <= diff < 30
                        30 <= diff
> Rbound; 30 <= diff < 70
           70 <= diff < 100
           100 <= diff

```

The following flags are optional inputs to change the values of Rbound and the time gap built in value.

```

-rnumber      number is Rbound (default 6000 meters)
-gnumber      number is maximum allowable gap for computation of jump in
              range (default 50 seconds)

```

The options are parsed and the appropriate variables are set. If a file is unavailable, an error message is generated. An interim gff file is created with the same name as the input file suffixed with _RJ. This file contains a record with the range and range rate in one record and no other observations. This file can be removed after program execution; it has a frame id of "rand". For each point that falls in the previously defined interval a record is written to stdout containing start time of interval, "jump", "range", duration of interval, and magnitude of the jump.

Example:

```

1000.0 range jump 30.0 75.0
1090.0 range jump 66.0 35.0
1200.0 range jump 15.0 101.0

```

Interpretation:

At time 1000.0 radar range measurements are inconsistent with range rate yielding differences averaging 75.0 meters for a span of 30 seconds. From 1030.0 to 1090.0 no measureable inconsistencies occur. At 1090.0 range jumps averaging 35 meters lasted for 66 seconds. No inconsistencies were found between 1156.0 to 1200 seconds; then the jumps averaged 101.0 for 15 seconds.

FILES

```

input
  OBS          observation file in gff format
output

```


qaranjmp(1)

(RELBET)

qaranjmp(1)

OBS_RJ combination of range and range rate observations at the same
 time for interim use, in gff format.

stdout text records for each point that is between the thresholds
 containing start time, "jump", "range", duration of
 interval, and magnitude of the jump.

NAME

qastar - Star Tracker obs history

SYNOPSIS

qastar [options] files

DESCRIPTION

qastar provides output to stdout summarizing the star observations on an observation file. The following options provide ways of changing the default constants. all are optional.

- p recompute and print the text table of suspected stars based on output of a previous run (the input gff for observation is thus a gff _ST file and the attitude file is not needed).
- y use the Y star tracker instead of the Z.
- lnumber defines the delta angle reflecting the low noise threshold setting for spotting stars (default .001)
- unumber defines the delta angle reflecting the high noise threshold setting for spotting stars (default .01)
- tnumber defines maximum time difference allowable between input quaternions and observations (default 60 seconds)

An input observation file(obsfile) and attitude file(attfile) are required inputs (except when using the -p option as noted above). The Z star tracker is assumed unless the -y option is requested. The options are parsed and the appropriate variables are set. Output text is written for each interval where two consecutive observations are fixed on the same inertially fixed object. This interval is accumulated. The text includes the following information: time of beginning, "star","edit",duration of interval, average angle over the interval. An output gff file is created with the name the same as the input obs file plus the suffix _ST. The data on the file is:time of the end of interval, "star",mode,azimuth,elevation and angle.

Example of printed output:

```
1000.0 star edit 15.045625 1.228931
```

Interpretation:

At time 1000.0 the observations are fixed on the same inertially fixed object for 15.045625 seconds with an average angular difference of 1.228931.

FILES

input

obsfile observation file in gff format
attfile attitude file in gff format

output

obsfile_ST output gff file containing the inertially computed coordinates for each mark. The records include the time,

qastar(1)

(RELBET)

qastar(1)

"star", mode, azimuth, elevation and angle.

stdout

text information for each interval that the observations are on a fixed object. The text includes time, "star", "edit", duration and average angle over the interval.

NAME

qasv -Summarizes the sensed velocity file and creates a sensed acceleration file

SYNOPSIS

qasv [options] file

DESCRIPTION

qasv provides output to stdout documenting edited sensed velocities and burn intervals.

-enumber is the maximum acceptable sensed acceleration (default 100)

-bnumber is the minimum burn sensed acceleration (default .00369)

The options are parsed and the appropriate variables are set. If a file is unavailable, an error message is generated. A gff file is generated containing the sensed acceleration vectors and the magnitude of the sensed acceleration at times of events. The frame id for this file is "sacc." For each point >e or >b a record is written to stdout containing start time, "sacc", "valid" or "edit", delta time, and magnitude of the sensed acceleration. The edit intervals are not accumulated; the valid ones are.

Example:

1000.0	sacc	valid	3.84	0.004430	1
1010.0	sacc	edit	3.84	200.000000	1
2000.0	sacc	valid	9.99	0.005000	3

Interpretation:

At time 1000.0 a significant sensed acceleration was measured for a duration of 3.84 seconds at an average magnitude of .004430 m/sec² including 1 point. At 1010 a bad point occurred causing sensed acceleration reading of 200. for 3.34 seconds including 1 point. Again at 2000.0 seconds a significant sensed acceleration was measured for a duration of 9.99 seconds at an average magnitude of .005 including 3 points.

FILES

input

SVEL sensed velocity file in gff format

output

SVEL_SV gff file containing the sensed velocity vectors and the magnitude of the sensed acceleration at the times that fall between the thresholds.

stdout

text information for all times whose sensed acceleration is above the minimum burn input with -b option. Those above the maximum input (-e option) are designated "edit", others are "valid". Each record contains start time, "sacc", mode, delta time, and magnitude of the sensed acceleration.

qatape(1)

(RELBET)

qatape(1)

NAME

qatape - UNIVAC tape Qa processor

SYNOPSIS

qatape (respond to prompts)

DESCRIPTION

This program reads the data product (binary) tape created on Univac. Output depends upon the response to first prompt.

- 0 prints first and last page
- 1 prints every page
- 2 prints every n record
- 3 creates binary disc file in HP format

INPUT/OUTPUT

Input - UNIVAC data product tape

Output - data product print file and data product binary file

EXECUTION PROCEDURE

1. Mount the input tape to the tape drive desired
2. Type in qatape
3. Answer prompts
4. Dismount tape

EXAMPLE

The following is an example of user inputs for the program qatape and its prompts. The prompts are self-explanatory. The prompt is indicated by double quotes (").

qatape

"enter 0 - for first and last page of output

enter 1 - for all pages of output

enter 2 - for every n records of output

enter 3 - for creating the binary file on disk"

2

"enter output data product print file name (72 char)"

prt.qatape2

"enter output record frequency"

50

"enter tape drive, please"

0

rdwt(1)

(RELBET)

rdwt(1)

NAME

rdwt - write a subset of a file with user specified frame id and time

SYNOPSIS

rdwt [options] files

DESCRIPTION

rdwt reads in a file and writes to the file with user specified frame and time. The following optional flags are available:

-fframeid frame id (default no check on frame id)
-bnumber begin time (default to file begin time)
-enumber end time (default to file end time)

The options are parsed and the appropriate variables are set. The files are parsed and output binary files are written. If an input file is unavailable, an error message is generated.

FILES

input
file1,... input gff files

output
file1_COPY,... output gff file that is a subset of the input file

read_set(1)

(RELBET)

read_set(1)

NAME

read_set - provides summary output of a SET file

SYNOPSIS

read_set [options] file

DESCRIPTION

read_set reads in a binary SET file created by the stop processor and provides a printed output option over a specified time span. The following describes the input flags; all are mandatory.

-ffilename indicates file name of SET file

-bnumber number indicates begin time in gmt seconds

-enumber number indicates last time in gmt seconds

-pnumber number indicates print frequency option. Nonpositive means first and last only (default)

FILES

input
file binary file created by the stop processor as described in Appendix I in the Programmers Manual

output
stdout text information about file as described in Appendix I in the Programmers Manual

read_sit(1)

(RELBET)

read_sit(1)

NAME

read_sit - provides summary output of a SIT file

SYNOPSIS

read_sit [options] file

DESCRIPTION

read_sit reads in a binary file created by the stop processor and provides a printed output option over a specified time span. The following describe the mandatory flags.

-filename indicates file name of SIT file

-bnumber number indicates begin time in gmt seconds

-enumber number indicates last time in gmt seconds

-pnumber number indicates print frequency option. Nonpositive means first and last only (default)

FILES

input
file binary SIT file created by the stop processor as described in Appendix 1 in the Programmers Manual

output
stdout text information about the file as described in Appendix I in the Programmers Manual

r1vsr1(1)

(RELBET)

r1vsr1(1)

NAME

r1vsr1

SYNOPSIS

r1vsr1 relative_trajectory_file1 relative_trajectory_file2

DESCRIPTION

r1vsr1 reads two relative trajectory files and determines the difference between states in UVW coordinates. The first input file is the base file of the UVW reference frame. The output file is the first input file name plus _RL.

For example:

r1vsr1 PRED KREL

output is PRED_RL

rptost(1)

(RELBET)

rptost(1)

NAME

rptost

SYNOPSIS

rptost < input_file

DESCRIPTION

Converts roll/pitch to shaft/trunnion reference. Use the following 1input input blocks; see description in Appendix II

xxgnr1 (optional)
xxtime (mandatory)
xxrpst (mandatory)

Note: input_file is a text file containing the 1input input blocks stated above.

run_editradar(1)

(RELBET)

run_editradar(1)

NAME

run_editradar - shell script for editing radar data

SYNOPSIS

run_editradar obsfile span.aran

DESCRIPTION

The shell script run_editradar invokes the program fileedit which edits the radar data during the specified time intervals. For more information, refer to RELBET USER'S MANUAL Section 5.3.7.

Input Files:

obsfile - the gff observation file to be edited

span.aran - the text file containing begin/end times for each interval to be edited

Output File:

obsfile - the gff input file including the edits

run_editst(1)

(RELBET)

run_editst(1)

NAME

run_editst - shell script for editing star tracker data

SYNOPSIS

run_editst obsfile span.star

DESCRIPTION

The shell script run_editst invokes the program fileedit which edits the star tracker data during periods of anomalous observations (stars etc...). For more detail, refer to RELBET USER'S MANUAL Section 5.2.3.

Input Files:

obsfile - the gff observation file to be edited

span.star - the text file containing begin/end times for each interval to be edited

Output File:

obsfile - the gff input file including the edits

run_gbfcom(1)

(RELBET)

run_gbfcom(1)

NAME

run_gbfcom - shell script for combining Gff files

SYNOPSIS

run_gbfcom output input1 input2 ...

DESCRIPTION

The shell script run_gbfcom invokes the program gbfcom which combines two gff files into one. Run_gbfcom allows for combining any number of gff files into one. For more detail, refer to RELBET USER'S MANUAL Section 5.2.2.

run_kobsedit(1)

(RELBET)

run_kobsedit(1)

NAME

run_kobsedit - shell script which utilizes the filter edits to actually edit the frames on the Residual file output by the sfilr process

SYNOPSIS

run_kobsedit KOBS

DESCRIPTION

The shell script run_kobsedit invokes the filedit processor which edit the KOBS file . For more detail, refer to RELBET USER'S MANUAL Section 5.4.4.3.

Input File:

KOBS - a gff file containing the Kalman pre_edit obs residuals, sigmas, bias solutions, sigmas, and the edit status

Output File:

KOBS_ED - the copy of KOBS where each frame exhibiting a Kalman edit status value of -1 is edited

run_noise(1)

(RELBET)

run_noise(1)

NAME

run_noise - shell script for evaluate noise on observation data

SYNOPSIS

```
run_noise input_file_name [year][month][day][hour][minute][second]
```

DESCRIPTION

run_noise automates the evaluation of noise on observation data. This shell script invokes the following programs:

- ddnois - generates noise tables for each observation type specified time intervals
- qanois - selects noise values from the output of ddnois and generates text summaries and gff files of selected noise for each observation type
- obsnois - place the noise values found on the output gff files of qanois onto the frames in the observation file for processing by the filter

Input :

input_file_name - is the gff observation file. This file is actually updated to contain the noise estimate in the first data slot of each frame to be processed by the filter

Input options together with their default values are as follows:

```
year      1985
month     1
day       0
hour      0
minute    0
second    0
```

The following is the default INPUT created by the shell script and used by the ddnois processor. Note that this file is stored in the directory /tmp during the run and is removed afterwards. You can change the shell contents if desired.

```
//xxgnr1 - use default
//xtime inputs
date = 1985,1,0,0,0;
dates = 0.e0;
tbegin = -1.0e20;
tend = 1.0e20;
delta = 0.0;
endopt = 0;
endval = 1.0e20;
//xxnois inputs
fnam = "input_file_name";
fnamo = "input_file_name_ddnois";
nobs = 6;
obarray = "aran", "brnr", "csft", "dtrn", "izth", "jztv";
obdelt = 200 , 200 , 200 , 200 , 400 , 400;
%%
```

Output:

The process creates several text and gff output files listed below:

```
textNaran      text
textNbrnr      text
```

run_noise(1)

(RELBET)

run_noise(1)

textNcsft	text
textNdtrn	text
textNizth	text
textNjztv	text
<input_file_name>_ddnois	gff
<input_file_name>_ddnois_Na	gff
<input_file_name>_ddnois_Nb	gff
<input_file_name>_ddnois_Nc	gff
<input_file_name>_ddnois_Nd	gff
<input_file_name>_ddnois_Ni	gff
<input_file_name>_ddnois_Nj	gff
input_file_name	gff

For more detail, refer to RELBET USER'S MANUAL, Section 5.3.3 and the programs entries for the programs ddnois, qanois, and obsnois.

run_obsedit(1)

(RELBET)

run_obsedit(1)

NAME

run_obsedit - shell script for editing the raw observation file
on the basis of the onboard data good flags

SYNOPSIS

run_obsedit obsfile

DESCRIPTION

The shell script run_obsedit invokes the program filedit to inspect each frame on an observation file which has just been output by the dwnfmt processor and if the onboard data good flag whose value should reside in the first data slot indicates bad data (value is 0) then the frame is edited. Note that a working version of the obsfile is stored in a directory called /tmp during the run. For more detail, refer to RELBET USER'S MANUAL Section 5.2.3.

Input/Output :

obsfile - the observation file which is updated to include edited frames

NAME

run_qacover - shell script for checking data continuity and completeness

SYNOPSIS

```
run_qacover root obs_file <sv_file> <att_file>
```

DESCRIPTION

The shell script run_qacover invokes the qacover processor on the given gff files to create output text files whose root name is the first argument, root. The information contained in the text files created by this processor includes time intervals of data valid, data lost, and data edited. For more detail, refer to RELBET USER'S MANUAL Section 5.3.1.

Input:

root - root name for output text files is mandatory
obs_file - gff observation file is mandatory
sv_file - gff sensed velocity file is optional
att_file - gff attitude file is optional

Output:

rootaran	text
rootbrnr	text
rootcsft	text
rootdtrn	text
rootmrol	text
rootopit	text
rootizth	text
rootjztv	text
<rootsv_file	text>
<rootatt_file	text>

run_qadata(1)

(RELBET)

run_qadata(1)

NAME

run_qadata - shell script for identify significant events

SYNOPSIS

run_qadata root obs_file sv_file att_file

DESCRIPTION

The shell script run_qadata invokes the the following programs:

qastar - process the star tracker observation

qasv - process the sensed velocity file

qaranjmp - process the radar range and range rate measurement

qaatt - process the attitude file The process creates several text

files, whose root name is the first argument, root, and several gff output files. For more detail, refer to RELBET USER'S MANUAL Section 5.3.2 and the programs entries for qastar, qasv, qaranjmp, and qaatt.

Input :

root - the root name for the output text files

obs_file - gff observation file is mandatory

sv_file - gff sensed velocity file is mandatory

att_file - gff attitude file is mandatory

Output:

rootstar text

rootburn text

rootrj text

rootattr text

obs_file_ST gff

sv_file_SV gff

att_file_AR gff

run_qatape(1)

(RELBET)

run_qatape(1)

NAME

run_qatape - shell script for qa UNIVAC product tape

SYNOPSIS

run_qatape output_text tape_drive_number

DESCRIPTION

The shell script run_qatape invokes the program qatape which qa the UNIVAC product tape. The product tape must be mounted onto the tape drive before execution. The output file contains the first and last records of the input product tape. You can change the shell contents if desired. For more detail, refer to RELBET USER'S MANUAL Section 5.5.3.

Input:

output_text - name of output text file tape_drive_number - either 0 or 1

Output:

output_text text contains the first and last records on the tape
as well as the number of records encountered

run_re12eph(1)

(RELBET)

run_re12eph(1)

NAME

run_re12eph - shell script for creating two ephemeris files
from a relative trajectory file

SYNOPSIS

```
run_re12eph rel_traj_file output_eph1 output_eph2 [year] [month] [day]  
[hour] [minutes] [second]
```

DESCRIPTION

The shell script run_re12eph invokes the program xcmpar which takes a relative trajectory file and creates two ephemeris files as output.

The input option defaults values are:

```
year      1985  
month     1  
day        0  
hour       0  
minute    0  
second    0
```

The following is the INPUT created by using the defaults. You can change the shell contents if desired. For more details see the program entries for xcmpar.

```
// xxgnr1 -use default  
// xxtime input  
date = 1985,1,0,0,0;  
dates = 0.e0;  
tbegin = -1.0e20;  
tend = 1.0e20;  
delta = 1.0;  
endopt = 0;  
endval = 1.0e20;  
// xxnflz inputs  
fname = "rel_traj_file",  
"EPH",  
"output_eph1",  
"output_eph2",  
" ",  
"REL";  
bfopt = 1;  
posx = 1,-1;  
print = 0;  
plot = 0;  
trjout = 1,1,0;
```

Input:

rel_traj_file - gff relative trajectory file

Output:

output_eph1 - gff ephemeris file output_eph2 - gff ephemeris file

run_rptost(1)

(RELBET)

run_rptost(1)

NAME

run_rptost - shell script for converting radar angles

SYNOPSIS

run_rptost obsfile [year][month][day][hour][minute][second]

DESCRIPTION

The shell script run_rptost invokes the program rptost which converts the roll-pitch radar angles to shaft-trunnion angles. The input option default values are:

```
year      1985
month     1
day       0
hour      0
minute    0
second    0
```

The following is the INPUT created by the shell script. Note that this file is stored in a directory called /tmp and is removed at the end of the run. You can change the shell contents if desired.

```
//xxgnr1 -use default
//xxtime inputs
date = 1985,1,0,0,0;
dates = 0;
tbegin = 1.0;
tend = 1.0e20;
delta = 0.0;
endopt = 0;
endval = 1.0e20;
//xxrpst inputs
infile = "obsfile";
outfile = "OUTPUT";
antang = 1.169370564e0;
%%
```

Input/Output:

obsfile - the gff observation file is updated to contain the shaft-trunnion angles on frame ids csft and dtrn respectively. Note that during the run a temporary gff file is stored in directory /tmp.

For more detail, refer to RELBET USER'S MANUAL Section 5.2.4.

run_search(1)

(RELBET)

run_search(1)

NAME

run_search - shell script for stripping the recommended edits
from the event data base file

SYNOPSIS

run_search TEXT

DESCRIPTION

The shell script run_search invokes the program search which reviews the
the event data base, TEXT, for particular events where pre-editing the data
before filter processing may be useful and outputs text files containing
the time intervals identified. The following files are generated:

edit.sacc	text
edit.attr	text
edit.oatt	text
edit.star	text
edit.range	text

For more detail, refer to RELBET USER'S MANUAL Section 5.3.4.

run_timespan(1)

(RELBET)

run_timespan(1)

NAME

run_timespan - shell script for creating time intervals

SYNOPSIS

run_timespan edit.file

DESCRIPTION

The shell script run_timespan invokes the UNIX system function awk to read in the entries extracted from the data base and outputs lines of begin and end times determined from the time tag and duration parameters found on each entry to standard out. For more detail, refer to RELBET USER'S MANUAL Section 5.3.6.

search(1)

(RELBET)

search(1)

NAME

search - gff file processor

SYNOPSIS

search [options]

DESCRIPTION

This program extracts intervals of desired information from the text files produced by the relbet qa processors or any source where the frames have the general form as follows:

time-tag(number) ID(string) INFO_TYPE(string) duration(number) number ...
the information used on each frame comes from the first three fields but the complete frame is read in and output if applicable

- h indicates help option which displays primary and dependent labels available for extraction. The use of this option precludes all others.
- ffilename indicates the file name of the input text file (mandatory)
- tnumber is the origin time of interest, use with -r option
- rnumber is the radius about the origin which encloses the time interval of interest, use with -t option
- bnumber is the begin time of the specified interval of interest use with -e option
- enumber is the end time of the specified interval of interest use with -b option
- pframeids indicates the list of frame (primary) IDs desired separated by blanks (default all frame IDs)
- s indicates the list of INFO_TYPES (secondary IDs) desired associated with the list of IDs (default, use all)

sfilt(1)

(RELBET)

sfilt(1)

NAME

sfilt - Sequential Filter

SYNOPSIS

sfilt < input_sfilt_text >output_printfile

DESCRIPTION

The Sequential Filter provides an estimated trajectory for the shuttle and a target vehicle from an a priori estimate and onboard observations with a Kalman filtering technique. The output solution file consists of the estimated trajectory in the standard RELBET solution files format containing state vectors, solution biases and lower triangular covariances matrices at each observation time or a predicted state between obs data intervals.

Use the following input blocks; see description in Appendix II

- xxgnr1 (mandatory)
- xxmisc (optional)
- xxtime (mandatory)
- xxfile (mandatory)
- xxsptm (optional)
- xxerth (optional)
- xxinit (mandatory)
- xxprop (mandatory)
- xxmax (mandatory)
- xxvcx (mandatory)
- xxmas (mandatory)
- xxrnpv (mandatory)
- xxsen (optional)
- xxbias (mandatory)
- xxkal (mandatory)
- xxscov (mandatory)

NOTE: The following pbug flags have special meaning for sfilt

- pbug(1) > 0, turns on nominal print
 < 0, adds detail such as cov, phi
- pbug(2) > 0, turns on print from kalman subroutine m50 update
 = 0, turns on print of UVW target state updates
- pbug(9) > 0, sets the output frequency of status record print
 = 0, no status records printed
- pbug(11) > 0, sets output unit and output print option to KOBS
- pbug(12) > 0, sets output unit and output print option to PRED
- pbug(13) > 0, sets output unit and output print option to KREL

FILES

Input

- input_sfilt_text textfile containing the input input blocks stated above
- obs observation file
- att attitude file
- sv sensed velocity file

Output

- solution file(s) GFF format file

sfilt(1)

(RELBET)

sfilt(1)

Kalman predicted states GFF relative trajectory format file
 Kalman relative states GFF relative trajectory format file
 Kalman obs info file GFF format file whose frame consists
 of time-tag , obs type , edit (0) ,
 value of Kalman edit status (-1 or 1),
 value of observation ,
 value of Kalman residual ,
 value of Kalman residual sigma ,
 value of Kalman computed bias , and
 value of Kalman computed bias sigma

printfile stdout where status records are
 printed at frequency specified by pbug(9)

NOTE: The status records have the form

sfilt: Status - (K-P)/sqrt(KCOV)

t = 2.10264872580679E+007 range = 100669.758579478192000

.316507D-01 .228848D-01-.997817D-03-.195972D-01 .175694D-01-.157791D-01
 -.587952D-01-.457497D-01 .976228D-02 .357978D-01-.385274D-01 .336792D-01

	obs	resid	resid/sigma	bias	bias/sigma
izth F	-.893683D-02	.471989D-04	.170703D-01	.506947D-05	.254375D-01
jztv F	-.452582D-01	.917308D-03	.988347D+00	.358999D-04	.185671D+00

where :

(K-P)/sqrt(KCOV) - indicates that the numbers on line 3 and 4
 represent the difference between the Kalman and predicted
 inertial orbiter and target states divided by the computed
 sigma for each respective state element.

t - indicates the time of current step

range - indicates the relative range between orbiter and target
 table of obs information includes the obs type, the Kalman
 edit status (T - edit, F - no edit), and obs value, Kalman
 residual, the residual ratio, bias, and bias ratio

For more information , see article on SFILT (Appendix III)

sln2r1(1)

(RELBET)

sln2r1(1)

NAME

sln2r1 - reads a solution file and generates rel traj,cov,bias files

SYNOPSIS

sln2r1 [options]

DESCRIPTION

sln2r1 reads an arbitrary sized (nsol) solution file output by either the kalman or smoothing filter and generates

- relative trajectory files for solution states
- covariance file in the form of UVW relative trajectory files where the noise sigmas representing the base state solution and relative state solution in UVW of the base solution state are presented as a 12 - element vectors and the bias sigmas follow
- bias solution file containing the bias solution for as many obs solved for per frame

The following list describes the options; all are optional except the -s option.

- rfilename indicates create the rel traj
- cfilename indicates create the covariance soln file
- bfilename indicates create the bias soln file
- nnumber is the size of the solution vector (default 16)
- sfilename is the input solution file

smooth(1)

(RELBET)

smooth(1)

NAME

smooth - smooth the Kalman estimates

SYNOPSIS

smooth solution_file_from_sfilt output print_freq num_records <input_text>out_text

DESCRIPTION

The smoother acts on a file containing solution type records output by the Kalman sequential filter. Starting at the end of the file and working toward the beginning, it smooths the output estimate of the sequential filter. At any one time all of the data previously processed by the Kalman sequential filter contributes to the estimate of the state at that time. The smoother working from the other direction adjust the Kalman estimate with information derived from data not yet processed by the Kalman. In the absence of state noise, the smoothing filter output is similar to a batch least squares solution.

print_freq - the output status print control specifying the frequency at which to print status records.

number_of_records - the number of records to process is sometimes useful to verify inputs before committing to an overnight process (the smoother runs a long time).

Use the following input blocks; see description in Appendix II

- xxgnr1 (mandatory)
- xxmisc (optional)
- xxtime (mandatory)
- xxfile (mandatory)
- xxsptm (optional)
- xxerth (optional)
- xxinit (mandatory)
- xxprop (mandatory)
- xxmax (mandatory)
- xxvcx (mandatory)
- xxmas (mandatory)
- xxrnp (mandatory)
- xxbias (mandatory)
- xxscov (mandatory)

Note: The above input blocks are usually the same ones used for sfilt.

The input file is the output solution file from sfilt. The output files include a print text file and solution file.

FILES

Input

input_text	textfile containing the 1input input blocks stated above (input_sfilt_text)
att	attitude file
sv	sensed velocity file
solution file(s)	GFF format file (output from sfilt)

Output

printfile	stdout where status records are printed at a frequency specified by the 3rd command line argument. Note that the print_freq input must be given to output status. A summary of inputs
-----------	---

smooth(1)

(RELBET)

smooth(1)

is also printed.

NOTE: The status records have the form

Processing Record 3739 , time 2.10379658205702E+007

Status Ratio (S-K)/sqrt(KCDV)

.161641E-01 .166557E-01 .165393E-01 .166677E-01 .821242E-02 .166226E-01
.207571E+01 .320113E+01 .305750E+01 .165550E+01 .124473E+01 .194436E+01
.215015E+01 .312345E+00 .338971E+00 .596695E+00
.175760E-01 .297317E-01 .000000E+00 .000000E+00 .000000E+00 .000000E+00

where :

(S-K)/sqrt(KCDV) - indicates that the following values represent the difference between the Smooth and Kalman solution states divided by the Kalman computed sigma for each respective state element. The first row compares orbiter state. The second row compares target state. The third and fourth rows compare bias solutions in the order radar range, shaft, trunnion, range rate, z-star tracker horizontal, z-star tracker vertical, COAS horizontal, COAS vertical, y-star tracker horizontal, y-star tracker vertical.

For more information , see article on SEQUENTIAL FILTER (Appendix III)

stop(1)

(RELBET)

stop(1)

NAME

stop - sensor tape output processor

SYNOPSIS

stop [options]

DESCRIPTION

stop reads in two files including a relative trajectory file and a SENSOR data file which is created by the downlist program specifically for the purpose of writing two binary files of SENSOR input data called SIT and SET as described in Appendix 1 of the Programmers Manual. Both of the options are mandatory.

-rfilename indicates file name of relative traj file

-sfilename indicates file name of sensor data file

xcmpar(1)

(RELBET)

xcmpar(1)

NAME

xcmpar

SYNOPSIS

xcmpar < input

DESCRIPTION

The program xcmpar is used for comparison of trajectories. Output may be 2 trajectories or relative trajectory

Use the following input blocks; see description in Appendix II

xxgnr1 (optional)
xxtime (mandatory)
xxnflz (mandatory)

Note: input is a text file containing the input blocks stated above.

NAME

xqdsp

SYNOPSIS

xqdsp < input

DESCRIPTION

This program provides for the display of trajectory and attitude information in various coordinates. This information may be displayed alphanumerically and/or saved for graphic display with the Graphics Display Processor.

File Usage (xxnflz)

Files 1 and 2 are reserved for input trajectory information. These may be either ephemeris or relative trajectory files. File 3 is reserved for input attitude information and must contain an attitude quaternion in the first four data words of a frame. File 4 is reserved for the output file. Output frames contain six data words and are in internal units. The frame ID corresponds to the parameter sets defined below.

Vehicle States (xxnflz)

Up to two vehicles may be considered: One is designated the orbiter and the other the target. The state for a particular vehicle may come from either File 1 or File 2. If the file is a relative trajectory file, the State may also be either trh base state or the relative state. When necessary states are obtained by Lagrangian interpolation from a user specified number of points (max 10).

Timing Control

Output is generated within a user specified time span (tbegin and tend). Output continues until the next time would be greater than the desired stop time.

Output Parameters (xxndpm)

Parameter Group Table below summarizes the various options for output parameters. Here O designates the first vehicle or orbiter state vector, T the second vehicle or target state vector, and A the attitude of the vehicle. All groups consist of six parameters. The coordinate systems are defined in the Engineering Manual. In particular, the output frames use the given ID as frame type and consist of the following parameters in each coordinate system.

M50 Cartesian - x, y, z, x, y, z

UVW - u, v, w, u, v, w

SHELL - x, y, z, x, y, z
 s s s s s sELEMENTS - semimajor axis, eccentricity, inclination, ascending node,
 argument or perigee, argument of latitude

EULER ANGLES - pitch, yaw, roll, x, y, z axis rates

Parameter Group Table

Number	ID	REQUIRED	OBJECT COORDINATES	INPUTS
1	sxyz	0	M50 Cartesian	0
2	selt	0	M50 Elements	0
3	suvw	0	Target UVW 0, T	0
4	slvh	0	Target LVLH 0, T	0
5	ssh1	0	Target Shell	0, T
6	txyz	T	M50 Cartesian	T
7	telt	T	M50 Elements	T
8	tuvw	T	Orbiter UVW, 0, T	0, T
9	tlvh	T	Orbiter LVLH	0, T
10	tsh1	T	Orbiter Shell	0, T
11	tbod	T	Orbiter Spherical	0, T, A
12	ire1	T-0	M50 Cartesian	0, T
13	ere1	T-0	M50 Elements	0, T
14	apyr	A	M50 Euler Angles	A
15	auvw	A	Orbiter UVW Euler Angles	0, A
16	rpvm	T-0	Range, veh1 and veh2 position magnitudes, range rate, veh1 and veh2 velocity magnitudes	0, T

Use the following input blocks; see description in Appendix II

xxgnr1 (optional)
 xxtime (mandatory)
 xxcon (optional)
 xxerth (optional)
 xxnflz (mandatory)
 xxprnt (optional)
 xxdprm (mandatory)
 xxusys (optional)

Note: input is a text file containing the input input blocks stated above.

APPENDIX II - INPUT BLOCKS

APPENDIX II - INPUT BLOCKS

This Appendix describes the standard input blocks required by the major RELBET processors. The format is similar to that of the UNIX type manual entries in Appendix I. The description section provides the details in the following format

```
input_id
  brief description :: type
  default values
  additional information
```

The id is the name that linput expects and the type is the FORTRAN type and dimension that linput uses for the parameter.

INPUT PARAMETERS

NAME	BLOCK	DESCRIPTION
a62rh	xxatm	a62 ref dens
a62rr	xxatm	a62 rr param
a62sc	xxatm	a62 c param
a62sh	xxatm	a62 scale ht
a76a	xxatm	a76 a param
a76a1	xxatm	a76 a1 param
a76a2	xxatm	a76 a2 param
a76a3	xxatm	a76 a3 param
a76b	xxatm	1976 std atmos. empirical factor b
a76f10	xxatm	a76 f10 flux
a76rh1	xxatm	a76 ref dens
abm0	xxatm	bm params 0
abm1	xxatm	bm fit params
abmalt	xxatm	bm ref alt
abmcb1	xxatm	bm cbd1
abmcb2	xxatm	bm cbd2
abmcda	xxatm	bm c-densea
abmc1g	xxatm	bm cos lag
abmcm2	xxatm	bm cbm2
abmgde	xxatm	bm gdi
abmrrf	xxatm	bm rref
abms1g	xxatm	bm sin lag
aeroc	xxvcx	drag numbers
antang	xxrpst	antenna angle
bfopt	xxnflz	base file flag
bias	xxbias	initial biases
bugs	xxgnr1	debug file
c	xxvcx	cylindrical fit parameters
cct	xxcct	
cd	xxvcx	drag coefficient
cdetut	xxrnpv	et offset
cfrmid	dwnfmt_files	frame ID,s for output gff RELBET file
chgdef	dwnfmt_files	change the default values
chord	xxvcx	vehicle chord

INPUT PARAMETERS (cont'd)

NAME	BLOCK	DESCRIPTION
clight	xxcon	speed of light
col80	xxprnt	column width option
cterms	xxgrav	c harms
curves	xxgraf	curves to plot
d	xxvcx	zutek fit parameters
datbuf	xxsprm	data buffer
date	xxtime	base date
dates	xxtime	base sec
dbflag	dwnfmt_files	process frames with bad data
delta	xxtime	time step
dict	xxsprm	dictionary record
drgfac	xxvcx	aerodynamic drag factor
drive	xxtoff	tape drive
dtable	xxtoff	data dropout table
dtbias	xxtoff	time bias and rate
dtdate	xxtoff	time bias date
dtmax	xxkal	max step
dtmoon	xxmoon	moon dt
dtnom	xxprop	nom step
dtsun	xxsun	sun eph time step
dtxcld	xxdata	data x id table
edcrit	xxkal	edit criterion
end	dwnfmt_files	end flag
end	dwnfmt_msids	
endopt	xxtime	term opt
endval	xxtime	term val
files	xxwrit	available Datain files to write to
filid	dwnfmt_files	
filids	xxfile	file ids for vehicles
fnam	nnois	
fname	dwnfmt_files	
fname	xxnflz	file names
fnamo	nnois	
frmthk	xxgraf	frame thickness
frsize	dwnfmt_files	frame size

INPUT PARAMETERS (cont'd)

NAME	BLOCK	DESCRIPTION
gmargin	xxgraf	grace margin
gmsg	xxsprm	generic message
gwbtabs	xxdata	wieghts and bias table
hdgree	xxvcx	maximum degree of harmonics
hdrid	dwnfmt_files	
header	xxprnt	header print option
hifile	xxmast	(unused) max unit allowed for internal files
hlunit	xxmast	(unused) highest unit
horder	xxvcx	max order of harmonics
hpbin	xxtoff	hp binary file
icoas	xxsen	coas id
id_nav	dwnfmt_msids	
idseq	dwnfmt_files	id sequence
imrk	xxqcrv	line option of curves
in	xxgnr1	input file unit designation
infil	xxrpst	input obs file name
iopt	xxscov	input option on state covariances UVW
iradar	xxsen	rendevous radar id
isen1	xxsen	1st extra sensor id
isen2	xxsen	2nd extra sensor id
isen3	xxsen	3rd extra sensor id
itrky	xxsen	y star tracker id
itrkz	xxsen	z star tracker id
jobdes	xxmisc	job description
jterms	xxgrav	j harms
kd	xxvcx	drag multiplier
kealb	xxsun	albedo
kear	xxsun	earth reflect
keedit	xxqcrv	edit plot options
kfile	xxqcrv	curve file
kflux	xxsun	flux
kline	xxqcrv	line type code
kname	xxqcrv	curve names
kparms	xxqcrv	parameter id's
ksmbol	xxqcrv	curve plot symbol code

INPUT PARAMETERS (cont'd)

NAME	BLOCK	DESCRIPTION
kspan	xxqcrv	begin,end
kstep	xxqcrv	step size
ksun	xxsun	1 au solar force
lblwd	xxgraf	label width option
lcynum	xxcct	
ldate	xxtoff	launch date
lgnpos	xxgraf	legend position
lordx	xxnflz	interpolation flag
lpage	xxprnt	lines/page
lrmmdl	xxkal	noise model option
mastab	xxmas	mass tables
maxfil	xxmast	maximum number of files
maxobs	xxsen	maximum number of obs
maxsen	xxsen	maximum number of sensors
maxtyp	xxmast	maximum number of m/sid's
msid	dwnfmt_msids	
muarth	xxerth	earth mu
moon	xxmoon	moon mu
musun	xxsun	sun's mu
nama1	xxfile	att 1 name
nama2	xxfile	att2 name
namd1	xxfile	data 1 name
namd2	xxfile	data 2 name
name1	xxfile	eph 1 name
name2	xxfile	eph 2 name
namr	xxfile	rtrj name
nams	xxfile	sol name
namv1	xxfile	svel 1 name
namv2	xxfile	svel 2 name
ncons	xxkal	lrbet3 noise model constants
nfiles	xxwrit	number of available Datain files
nformat	xxprnt	format id
nlabel	xxqcrv	point label options
nmord	xxmoon	moon intrp order
nobs	nnois	

INPUT PARAMETERS (cont'd)

NAME	BLOCK	DESCRIPTION
nsclz	xxusys	scale factor names
nsord	xxsun	sun intrp order
nvars	xxwrit	number of variables in each file
nveh	xxmax	number of vehicles
o4000	xxcct	
obaray	nnois	
obdelt	nnois	
obsrvr	xxmisc	id of observer vehicle
off_sc_conv_lo_hi	dwnfmt_msids	
option	xxgraf	main option
out	xxgnrl	unit for output print
outfil	xxrpst	output obs file name
paero	xxprop	aero force
pbug	xxgnrl	debug flags
pcb	xxprop	central body force
pdrag	xxprop	drag force
pfid	xxqprm	frame id's
pframe	xxgraf	frame flag
pfreq	xxsptm	print frequency
pharm	xxprop	harmonics force
pi	xxcon	pi
plegnd	xxgraf	legend size
plot	xxnflz	plot flag
pmmx	xxpmmx	min/max values
pmoon	xxprop	moon force
pmxflg	xxqprm	min/max options
pname	xxqprm	parameter names
pofset	xxqprm	offset
porigin	xxgraf	physical origin location
posx	xxnflz	location for data
prad	xxprop	solar radiation flag
print	xxnflz	print flag
pscale	xxqprm	scale factors
psize	xxqcrv	plot symbol size
psun	xxprop	sun force

INPUT PARAMETERS (cont'd)

NAME	BLOCK	DESCRIPTION
psvb	xxsvbi	bias flag for file 1
psvel	xxprop	sensed velocity flag
punits	xxqprm	units names
pvent	xxprop	vent force
pword	xxqprm	index to data word
qafreq	xttoff	qa rec print freq
qbs	xxsen	sensor attitude quaternions m50 to sensor
qpdev	xxqgen	device for plot
rbarna	xxvcx	actual cg pos rel to nominal cg
recpnt	xxwrit	pointers to record numbers in a file
req	xxerth	earth eq rad
rgrav	xxgrav	grav radius
rnp0	xxrnpX	rnp matrix
rpol	xxerth	earth pol rad
rsob	xxsen	sensor offsets
rvcov	xxscov	position, velocity covariances
rvmoo0	xxmoon	init moon state
rvopt	xxprop	prop option
rvsun0	xxsun	init sun state
sagate	xxvcx	accel gate
sarea	xxvcx	area for solar rad
seq	xxsprm	file sequence id
sgbias	xxbias	bias sigmas
skip	dwnfmt_cct	skip flag
sncon	xxscov	constants in the state noise computation
solfac	xxvcx	solar radiation factor
spg	xxdprm	display group flags
spg	xxsprm	display group flags
sptime	xxsptm	special times
sptol	xxsptm	tolerance
sref	xxvcx	vehicle reference area
srflc	xxvcx	vehicle reflectivity
sterms	xxgrav	s harms
sthite	xxqgen	standard symbol height
svbtb	xxsvbi	bias table for file 1

INPUT PARAMETERS (cont'd)

NAME	BLOCK	DESCRIPTION
sxcld	xxka1	state exclusion flags
tape	xxtoff	tape option
target	xxmisc	id of target vehicle
tbegin	xxtime	beg time
tcrflt	xxmast	time current file
tend	xxtime	end time
term	xxgnr1	unit for short print
timcon	xxbias	bias time constants
timoff	xxmast	time off
title	xxgraf	main title
title2	xxgraf	title2
title3	xxgraf	title3
title4	xxgraf	title4
tlapse	xxtoff	summary frequency (min)
tmoo0	xxmoon	moon state time
trjout	xxnflz	traj flags
trnp0	xxrnpv	rnp time
trvint	xxinit	init t,r,v
tsun0	xxsun	sun base time
ttlmu1	xxgraf	title scale
ttlwd	xxgraf	title width option
undrwt	xxka1	underweighting options
unta1	xxfile	att 1 unit
unta2	xxfile	att2 unit
untd1	xxfile	data 1 unit
untd2	xxfile	data 2 unit
unte1	xxfile	eph 1 unit
unte2	xxfile	eph 2 unit
untr	xxfile	rtrj unit
unts	xxfile	sol unit
untv1	xxfile	svel 1 unit
untv2	xxfile	svel 2 unit
usclz	xxusys	scale factors
usysex	xxusys	display scale flags
usysin	xxusys	input unit system id

INPUT PARAMETERS (cont'd)

NAME	BLOCK	DESCRIPTION
usyst	xxusys	unit system names
uzea1	xxfile	att 1 use
uzea2	xxfile	att 2 use
uzed1	xxfile	data 1 use
uzed2	xxfile	data 2 use
uzee1	xxfile	eph 1 use
uzee2	xxfile	eph 2 use
uzer	xxfile	rtrj use
uzes	xxfile	sol use
uzev1	xxfile	svel 1 use
uzev2	xxfile	svel 2 use
var	xxkal	observation variances
vars	xxwrit	names of variables in the files
vnntab	xxvnt	vent timeline
wei	xxerth	earth inrl rot rate
xlabel	xxgraf	x axis label
xmmx	xxgraf	x axis min/max
xyang	xxgraf	xyz label angles
xyarea	xxgraf	subplot area
xyaxs	xxgraf	xy axes option
xygrid	xxgraf	xy grid flags
xyhite	xxgraf	xy label height fact
xypage	xxgraf	page size
xyzln	xxgraf	line options
kyznch	xxgraf	axes scale per inch
kyzstp	xxgraf	axes units per tic
ylabel	xxgraf	y axis label
ymmx	xxgraf	y axis min/max
zlabel	xxgraf	z axis label
zmmx	xxgraf	z axis min/max

xxatm(2)

(RELBET)

xxatm(2)

NAME

xxatm - input parameters for atmospheric parameters

SYNOPSIS

DOUBLE PRECISION a62sc
DOUBLE PRECISION a62sh
DOUBLE PRECISION a62rr
DOUBLE PRECISION a62rh
DOUBLE PRECISION a76a
DOUBLE PRECISION a76b
DOUBLE PRECISION a76f10
DOUBLE PRECISION a76rh1
DOUBLE PRECISION a76a1
DOUBLE PRECISION a76a2
DOUBLE PRECISION a76a3
DOUBLE PRECISION abmc1g
DOUBLE PRECISION abms1g
DOUBLE PRECISION abmgde
DOUBLE PRECISION abmal1t
DOUBLE PRECISION abm1(3,2)
DOUBLE PRECISION abm0(3)
DOUBLE PRECISION abmcda
DOUBLE PRECISION abmcb1
DOUBLE PRECISION abmcb2
DOUBLE PRECISION abmcm2
DOUBLE PRECISION abmrrf

DESCRIPTION

a62sc

a62 c param :: DOUBLE PRECISION a62sc
.2590478000d-03
atmospheric curve fit parameter

a62sh

a62 scale ht :: DOUBLE PRECISION a62sh
.1206650000d+06
atmospheric curve fit scale height

a62rr

a62 rr param :: DOUBLE PRECISION a62rr
.1294000000d-07
atmospheric curve fit parameter

a62rh

a62 ref dens :: DOUBLE PRECISION a62rh
.1225053499d+01
atmospheric density at sea level

a76a

a76 a param :: DOUBLE PRECISION a76a
.2046000000d-01

xxatm(2)

(RELBET)

xxatm(2)

1976 std atmos. empirical factor a

a76b

1976 std atmos. empirical factor b :: DOUBLE PRECISION a76b

.8402100000d+00

a76f10

a76 f10 flux :: DOUBLE PRECISION a76f10

.1010000000d+03

1976 std atmos. solar flux

a76rh1

a76 ref dens :: DOUBLE PRECISION a76rh1

.5299400000d-10

1976 std atmos. ref. density at scale height

a76a1

a76 a1 param :: DOUBLE PRECISION a76a1

.5299400000d-10

1976 std atmos. curve fit param

a76a2

a76 a2 param :: DOUBLE PRECISION a76a2

.7949100000d+05

1976 std atmos. curve fit param a2

a76a3

a76 a3 param :: DOUBLE PRECISION a76a3

.8332520000d+06

1976 std atmos. curve fit param a3

abmclg

bm cos lag :: DOUBLE PRECISION abmclg

.7986355100d+00

babb-mueller cosine of lag angle

abmslg

bm sin lag :: DOUBLE PRECISION abmslg

.6018150230d+00

babb-mueller sine of lag angle

abmgde

xxatm(2)

(RELBET)

xxatm(2)

bm gdi :: DOUBLE PRECISION abmgde

.1375000000d+01

babb-mueller gdi exponent

abmalt

bm ref alt :: DOUBLE PRECISION abmalt

.2222395555d+06

babb-mueller ref. altitude

abm1

bm fit params :: DOUBLE PRECISION abm1(3,2)

-.2515818000d+02, -.1166693668d-04, .9922058998d+04 ,
-.2321397100d+02 , -.1625013451d-04 , .7868452230d+06

babb-mueller curve fit params

abm0

bm params 0 :: DOUBLE PRECISION abm0(3)

-.1986449600d+00 , .4152999344d-05 , -.3926329360d+05

babb-mueller curve fit params

abmcda

bm c-densea :: DOUBLE PRECISION abmcda

.8999972860d+05

babb-mueller parameter c-densea

abmcb1

bm cbd1 :: DOUBLE PRECISION abmcb1

.4604986877d-04

babb-mueller parameter cbd1

abmcb2

bm cbd2 :: DOUBLE PRECISION abmcb2

-.4500000000d-04

babb-mueller parameter cbd2

abmcm2

bm cbm2 :: DOUBLE PRECISION abmcm2

-.9896459900d+00

babb-mueller parameter cbm2

abmrrf

bm rref :: DOUBLE PRECISION abmrrf

xxatm(2)

(RELBET)

xxatm(2)

.1225004738d+01

babb-mueller parameter rref

xxbias(2)

(RELBET)

xxbias(2)

NAME

xxbias - input parameters for bias inputs

SYNOPSIS

DOUBLE PRECISION bias(10)
DOUBLE PRECISION sgbias(10)
DOUBLE PRECISION timcon(10)

DESCRIPTION

bias

initial biases :: DOUBLE PRECISION bias(10)

at the beginning of the filter run, the observation biases are initialized to the values in sbias

sgbias

bias sigmas :: DOUBLE PRECISION sgbias(10)

50.d0,.01745d0,.01745d0,.1d0,
.0002d0,.0002d0,
.0002d0,.0002d0,
.00074d0,.00074d0

at the beginning of the filter run, the observation bias sigmas are initialized to the values in sgbias

timcon

bias time constants :: DOUBLE PRECISION timcon(10)

400.d0,10000.d0,10000.d0,400.d0,
10000.d0,10000.d0,
10000.d0,10000.d0,
400.d0,400.d0

the time constants used in the propagation of the observation biases (affects the decay of each computed bias)

xxcct(2)

(RELBET)

xxcct(2)

NAME

xxcct - input parameters for cct info

SYNOPSIS

INTEGER cct
INTEGER lcyum
INTEGER o4000

DESCRIPTION

cct

indicates which tape drive the input cct is on

lcyum

number of samples per cycle to process cct

o4000

the indicator of header record present on cct header records

xxcon(2)

(RELBET)

xxcon(2)

NAME

xxcon - input parameters for constants

SYNOPSIS

DOUBLE PRECISION c1ight
DOUBLE PRECISION pi

DESCRIPTION

specifies physical and mathematical
constants

c1ight

speed of light :: DOUBLE PRECISION c1ight
0.2997925d9

pi

pi :: DOUBLE PRECISION pi
.314159265358979324d1
value of math constant pi

xxdata(2)

(RELBET)

xxdata(2)

NAME

xxdata - input parameters for obs infor

SYNOPSIS

INTEGER dtxcld(25)
DOUBLE PRECISION gwbtabs(2,25)

DESCRIPTION

dtxcld

data x id table :: INTEGER dtxcld(25)

ids of data that are excluded from processing

gwbtabs

wieghts and bias table :: DOUBLE PRECISION gwbtabs(2,25)

observation weights and bias information. on input

gwbtabs(1,*)= obs sigma in internal units ,

gwbtabs(2,*)= obs bias in internal units

NAME

xxdprm - input parameters for numerical display

SYNOPSIS

INTEGER spg(20)

DESCRIPTION

spg

display group flags :: INTEGER spg(20)

positive value specifies parameter groups to display

options ids

1. sxyz vehicle 1 m50 state
2. selt vehicle 1 m50 elements (a,e,i,node,perigee,tra)
3. suvw vehicle 1 in vehicle 2 uvw
4. slvh vehicle 1 in vehicle 2 lvlh
5. sshl vehicle 1 in vehicle 2 shell
6. txyz vehicle 2 m50 state
7. teit vehicle 2 m50 elements (a,e,i,node,perigee,tra)
8. tuvw vehicle 2 in vehicle 1 uvw
9. tlvh vehicle 2 in vehicle 1 lvlh
10. tshl vehicle 2 in vehicle 1 shell
11. tbod look angle to veh 2 from veh 1 in veh 1 body frame
12. irel m50 state of veh2 relative to veh1
13. erel m50 element of veh2 minus m50 elements of veh1
14. apyr inertial pitch-yaw-roll (321 angles of input att quat)
15. auvw uvw pitch-yaw-roll (321 angles of quaternion obtained by multiplying uvw -> m50 quat quat by the input attitude conjugate q, i.e. quat * con(q))
16. rpvm range, veh1 and veh2 position magnitudes, range rate, veh1 and veh2 velocity magnitudes

Note: If a relative trajectories is used to generate any of the above options both the vehicle and the target option must be set.

dwnfmt_cct(2)

(RELBET)

dwnfmt_cct(2)

NAME

dwnfmt_cct - input parameters for additional cct processing option for
downlist formatter

SYNOPSIS

INTEGER skip(1)

DESCRIPTION

skip

skip flag :: INTEGER skip(1)

indicates that first record on the input cct tape is to be skipped when
set to 1

dwnfmt_files(2)

(RELBET)

dwnfmt_files(2)

NAME

dwnfmt_files - input parameters for output gff file options for downlist formatter, for local interface between user and common block xxfiles in subroutine dfnput

SYNOPSIS

```
CHARACTER*72 fname( 1 )
CHARACTER*4  hdrid( 1 )
CHARACTER*4  cfrmid( 10 )
INTEGER  filid( 1 )
INTEGER  frsize( 3 )
INTEGER  idseq( 200 )
INTEGER  dbflag( 1 )
INTEGER  chgdef( 1 )
INTEGER  end( 1 )
```

DESCRIPTION

fname
hdrid
cfrmid

frame ID,s for output gff RELBET file :: CHARACTER*4 cfrmid(10)

up to 10 ID's may be input associated in the order in which they occur to the presence of numbers larger than 200 in the id_sequence. ie., whenever a number greater than 200 occurs in the id_sequence then associate the next frame ID on the list with the successive information until a new frame ID is indicated.

filid
frsize

frame size :: INTEGER frsize(3)

the integer word size of the different portions of the frame

(1) the number of integer words including the time-tag which make up the header portion of the frame

(2) the number of integer words in data portion of the frame

(3) the number of dummy places to fill after the data portion of the frame in integer words

idseq

id sequence :: INTEGER idseq(200)

the id_sequence is set up to resemble as closely as possible the frame building process. The internal datum ids of the msids used to build the frames are arranged in a way which reflects their use in building the output frame as follows:

- the first number must be the internal id of the msid representing a time value which is used as the time-tag of the output frame. this value is further negated to emphasize that this parameter is destined for the time-tag position of the header portion of the frame

- the next number must be a number greater than 200 representing the frame id to be placed on the frame_id position of the header portion of the frame. as mentioned above the actual frame id is input in the list cfrmid.

- the remaining numbers represent the internal ids of the m/sid's whose values will appear on the data portion of the output frame in the order in which they are herein specified.

dwnfmt_files(2)

(RELBET)

dwnfmt_files(2)

EX: idseq = -74,1000,56,57,58;

dbflag

process frames with bad data :: INTEGER dbflag(1)

when set non-zero indicates that file frames are to be output even if some data bad (note:if time-tag is bad no data output)

chgdef

change the default values :: INTEGER chgdef(1)

non_zero indicates that the inputs for frsize , cfrmid and idseq are to be used instead of the defaults. Note: if chgdef is set then all settings of frsize , cfrmid and idseq must be set.

end

end flag :: INTEGER end(1)

indicates that current input block is the last files input block and the next input block is an msids input block

dwnfmt_msids(2)

(RELBET)

dwnfmt_msids(2)

NAME

dwnfmt_msids - input parameters for m/sid processing options for downlist formatter, for local interface between user and common block xxmsid in subroutine dfnpu

SYNOPSIS

CHARACTER*10 msid(1)
INTEGER id_nav(2)
DOUBLE PRECISION off_sc_conv_lo_hi(5)
INTEGER end(1)

DESCRIPTION

msid
:: CHARACTER*10 msid(1)
alphanumeric m/sid number associated with downlisted parameter on cct

id_nav
the internal datum id and flag indicating nav buffer parameter (1), data id has range 1 to 200 (2), =0, not part of nav buffer: =1, part of nav buffer

off_sc_conv_lo_hi
the double precision data for converting and checking m/sid values (1), offset term added to value (2), scale factor multiplied on value before validation check (3), conversion factor to achieve internal RELBET units (4), minimum expected value (5), maximum expected value default: = 0.0 , 1.0 , 1.0 , -1.e30 , 1.e30

end
indicates that current input block is the last msids input block and the next input block is a cct input block

xxerth(2)

(RELBET)

xxerth(2)

NAME

xxerth - input parameters for earth constants

SYNOPSIS

DOUBLE PRECISION muerth
DOUBLE PRECISION req
DOUBLE PRECISION rpo1
DOUBLE PRECISION wei

DESCRIPTION

specifies parameters and constants associated with the earth's shape, mass, and rotation rate.

muerth

earth mu :: DOUBLE PRECISION muerth
.3986012d15
earth gravitational parameter

req

earth eq rad :: DOUBLE PRECISION req
.6378166d7
equatorial radius of earth

rpo1

earth po1 rad :: DOUBLE PRECISION rpo1
.6356784283607107d7

wei

earth inrl rot rate :: DOUBLE PRECISION wei
.729211514645921d-4
inertial rotation rate of the earth

NAME

xxfile - input parameters for file info

SYNOPSIS

```

CHARACTER*72 nama1
INTEGER unta1
INTEGER uzea1
CHARACTER*72 nama2
INTEGER unta2
INTEGER uzea2
CHARACTER*72 name1
INTEGER unte1
INTEGER uzee1
CHARACTER*72 name2
INTEGER unte2
INTEGER uzee2
CHARACTER*72 namd1
INTEGER untd1
INTEGER uzed1
CHARACTER*72 namd2
INTEGER untd2
INTEGER uzed2
CHARACTER*72 namv1
INTEGER untv1
INTEGER uzev1
CHARACTER*72 namv2
INTEGER untv2
INTEGER uzev2
CHARACTER*72 namr
INTEGER untr
INTEGER uzer
CHARACTER*72 nams
INTEGER unts
INTEGER uses
INTEGER fillds( 3,2 )

```

DESCRIPTION

nama1
att 1 name :: CHARACTER*72 nama1
name for first attitude file

unta1
att 1 unit :: INTEGER unta1
unit for first attitude file

uzea1
att 1 use :: INTEGER uzea1
usage code for first attitude file

nama2
att2 name :: CHARACTER*72 nama2
name for second attitude file

unta2
att2 unit :: INTEGER unta2

xxfile(2)

(RELBET)

xxfile(2)

unit for second attitude file

uzea2

att 2 use :: INTEGER uzea2

usage code for second attitude file

name1

eph 1 name :: CHARACTER*72 name1

name for first ephemeris file

unte1

eph 1 unit :: INTEGER unte1

unit for first ephemeris file

uzee1

eph 1 use :: INTEGER uzee1

usage code for first ephemeris file

name2

eph 2 name :: CHARACTER*72 name2

name for second ephemeris file

unte2

eph 2 unit :: INTEGER unte2

unit for second ephemeris file

uzee2

eph 2 use :: INTEGER uzee2

usage code for second ephemeris file

namd1

data 1 name :: CHARACTER*72 namd1

name for data file 1

untd1

data 1 unit :: INTEGER untd1

unit for data file 1

uzed1

data 1 use :: INTEGER uzed1

usage code for data file 1

namd2

data 2 name :: CHARACTER*72 namd2

xxfile(2)

(RELBET)

xxfile(2)

name for data file 2

untd2

data 2 unit :: INTEGER unt2

unit for data file 2

uzed2

data 2 use :: INTEGER uzed2

usage code for data file 2

namv1

svel 1 name :: CHARACTER*72 namv1

name for sensed velocity file 1

untv1

svel 1 unit :: INTEGER untv1

unit for sensed velocity file 1

uzev1

svel 1 use :: INTEGER uzev1

usage code for sensed velocity file 1

namv2

svel 2 name :: CHARACTER*72 namv2

name for sensed velocity file 2

untv2

svel 2 unit :: INTEGER untv2

unit for sensed velocity file 2

uzev2

svel 2 use :: INTEGER uzev2

usage code for sensed velocity file 2

namr

rtrj name :: CHARACTER*72 namr

name for relative trajectory file

untr

rtrj unit :: INTEGER untr

unit for relative trajectory file

uzer

rtrj use :: INTEGER uzer

xxfile(2)

(RELBET)

xxfile(2)

usage code for relative trajectory file

nams

sol name :: CHARACTER*72 nams

name for solution file

unts

sol unit :: INTEGER unts

unit for solution file

uzes

sol use :: INTEGER uzes

usage code for solution file

filids

file ids for vehicles :: INTEGER filids(3,2)

xxgnr1(2)

(RELBET)

xxgnr1(2)

NAME

xxgnr1 - input parameters for general I/O info

SYNOPSIS

INTEGER bugs
INTEGER in
INTEGER term
INTEGER out
INTEGER pbug(19)

DESCRIPTION

bugs

debug file :: INTEGER bugs

6

in

input file unit designation :: INTEGER in

69

term

unit for short print :: INTEGER term

6

out

unit for output print :: INTEGER out

6

pbug

debug flags :: INTEGER pbug(19)

19*0

NAME

xxgraf - input parameters for plot controls

SYNOPSIS

```

INTEGER option
CHARACTER*60 xlabel
CHARACTER*60 ylabel
CHARACTER*60 zlabel
CHARACTER*60 title
CHARACTER*60 title2
CHARACTER*60 title3
CHARACTER*60 title4
REAL t1mul( 4 )
CHARACTER*8 curves( 20 )
INTEGER pframe
INTEGER xygrid( 3 )
CHARACTER*4 xyaxs( 2 )
CHARACTER*4 ttlwd
CHARACTER*4 lblwd
CHARACTER*4 xyzln( 3 )
REAL xypage( 2 )
REAL porgin( 2 )
REAL xyarea( 2 )
REAL frmthk
REAL gmargin
REAL xyhite
REAL xyznch( 3 )
REAL xyzstp( 3 )
REAL xmmx( 2 )
REAL ymmx( 2 )
REAL zmmx( 2 )
REAL xyang( 3 )
REAL plegend
REAL lgnpos( 2 )

```

DESCRIPTION

option

main option :: INTEGER option

1

designate main action:

```

neg      stop
non neg  plot
pos      use automatic scale of axes
zero     no automatic scale of axes. User must give min,max and
         steps for x and y axes.(xmmx,ymmx,xyzstp)

```

xlabel

x axis label :: CHARACTER*60 xlabel

specifies the label for x axis. The first character cues special features. '!' label and tics on opposite side of axis. '\$' No axis and tick mark are drawn. Tick mark can be drawn if blank in quote is used.

ylabel

y axis label :: CHARACTER*60 ylabel

specifies the label for y axis. See xlabel for detail.

zlabel

z axis label :: CHARACTER*60 zlabel

specifies the label for z axis . See xlabel for detail.

title

main title :: CHARACTER*60 title

up to 4 titles may be specified. each title may be up to 60 characters long. each title should start with and end with single quote. each title has a scale factor associated with this factor specifies what factor of nominal character size the title is to be displayed with . a negative scale factor results in the title being underlined.

title2

title2 :: CHARACTER*60 title2

. see title for description

title3

title3 :: CHARACTER*60 title3

see title for description

title4

title4 :: CHARACTER*60 title4

see title for description

ttlmu1

title scale :: REAL ttlmu1(4)

2.0 , 1.5 , 0.0 , 0.0

title scale: specifies the scale for title . A negative values cause the corresponding title to be underlined.

curves

curves to plot :: CHARACTER*8 curves(20)

curves to plot. Up to 20 curves may be defined. Up to 7 characters can be used for curves name. The last character is reserved for dollar sign.

pframe

frame flag :: INTEGER pframe

Plot frame flag : specifies whether a frame is to be drawn around plot.
 > 0 Draw frame
 else No frame

xygrid

xy grid flags :: INTEGER xygrid(3)

2*1,0

xygrid flag : specifies the grid options. The first two entries specify the grid frequency k for the x and y axes respectively.
 K > 0 k grid line per tic mark
 K = 0 No grid lines

xxgraf(2)

(RELBET)

xxgraf(2)

K < 0 k tic marks per grid line The third entry specifies the line type. Options are same as for curve line types.

xyaxs

xy axes option :: CHARACTER*4 xyaxs(2)

2*'II'

specifies the character version of axes options. the options are: S for linear display scale, L for log, I for absolute scale, P for Polar. Only the first two characters matter, e.g., 'IL', is linear in X and Log in Y.

ttlwd

title width option :: CHARACTER*4 ttlwd

lblwd

label width option :: CHARACTER*4 lblwd

xyzln

line options :: CHARACTER*4 xyzln(3)

xypage

page size :: REAL xypage(2)

11.0 , 8.5

the page size is the area which is taken up by the entire graphic display --both label and plot. it is specified in nominal inches, the first entry corresponding to the horizontal (x) dimension and the second to the vertical (y) dimension. the term nominal is used since the actual size depends upon the particular graphic device. the actual size may be smaller for hp . a typical page size is 11 by 8.5

porigin

physical origin location :: REAL porigin(2)

2*-1.0

physical origin :: specifies the location of plot origin in nominal inches from the lower left hand corner of the page . A negative value results in the program setting the plot origin.

xyarea

subplot area :: REAL xyarea(2)

8.5 , 6.0

this is the area encompassed by the x and y axes. thus the value specifies the length of the x axis and the second value specifies the length of the y axis. it is specifies in nominal inches. the default value for subplot area is 6 by 6. a typical subplot area is 8.5 by 6.0 for hp screen.

frmthk

frame thickness :: REAL frmthk

Specifies the scale for the frame thickness in nominal inches. If a value greater than zero is specified, the frame around the plot will be drawn with the approximate thickness. Values greater than 0.125 are reset to this value.

gmargin

grace margin :: REAL gmargin

specifies a distance in nominal inches from the plot area border within which points will be plotted. Positive values allow for points being plotted outside the plot area. Negative values ensure that point will lie within the plot area.

xyhite

xy label height fact :: REAL xyhite

1.2

axis label scale : specifies for axis labels in multiple of .14 inches.

xyznch

axes scale per inch :: REAL xyznch(3)

specifies absolute scale of axes in axis units per inch .

xyzstp

axes units per tic :: REAL xyzstp(3)

specifies axis units per tic. this parameter is specified if and only if option = 0 is used (no scaling is done by program)

xmx

x axis min/max :: REAL xmx(2)

this parameter is specified if and only if option = 0 is used. this implies no scaling is done by program.

ymmx

y axis min/max :: REAL ymx(2)

see xmx for detail

zmx

z axis min/max :: REAL zmx(2)

see xmx for detail

xyang

xyz label angles :: REAL xyang(3)

45.0 , 2*0.0

specifies angle relative to horizontal for tic numbers. The default is 0.

xxgraf(2)

(RELBET)

xxgraf(2)

plegnd

legend size :: REAL plegnd

1.0

Legend scale : specifies scale factor for legend display as a multiple of standard character height of 0.14 inches. Options are :
> 0 Display legend width specified scale
else No legend

lgnpos

legend position :: REAL lgnpos(2)

8.47 , 3.5

Specified the position of the legend's upper hand corner in nominal inches form the plot origin.

xxgrav(2)

(RELBET)

xxgrav(2)

NAME

xxgrav - input parameters for earth grav param

SYNOPSIS

DOUBLE PRECISION rgrav
DOUBLE PRECISION cterms(35)
DOUBLE PRECISION sterms(35)
DOUBLE PRECISION jterms(2:8)

DESCRIPTION

rgrav

grav radius :: DOUBLE PRECISION rgrav

6378160.0d0

geopotential model earth radius

cterms

c harms :: DOUBLE PRECISION cterms(35)

0.d0, .155752d-05, .212763d-05, .304690d-06, .957d-07,
-.502698d-06, .738439d-07, .591298d-07, -.16838d-08,
-.460853d-07, .99182d-07, -.142322d-07, -.207839d-08,
.310069d-09, -.778802d-07, .682041d-08, .577916d-09,
-.152714d-11, -.170638d-09, .206637d-10, .17670d-06,
.281935d-07, .285733d-08, -.418909d-09, -.307997d-12,
-.179403d-10, .29525d-11, .214773d-07, .395567d-08,
-.58076d-09, -.200713d-09, -.102636d-10, -.150544d-11,
.175356d-12, -.986208d-13

earth harmonics c22 thru c88

sterms

s harms :: DOUBLE PRECISION sterms(35)

0.d0, -.880523d-06, .280994d-06, -.216784d-06, .19946d-06,
-.462625d-06, .157940d-06, -.92433d-08, .71686d-08,
-.838408d-07, -.567829d-07, -.286286d-08, .646339d-09,
-.147513d-08, .296244d-07, -.437589d-07, .925271d-09,
-.152888d-08, -.421805d-09, -.174166d-10, .846618d-07,
.155828d-07, -.330286d-08, -.24187d-09, .348475d-10,
.708604d-11, -.125606d-11, .176579d-07, .691077d-08,
.18285d-09, .982345d-10, .11156d-10, .72419d-11,
.454672d-12, .861829d-13

earth harmonics s22 thru s88

jterms

j harms :: DOUBLE PRECISION jterms(2:8)

1.082637d-03, -2.539d-06, -1.617d-06, -2.34d-07, 5.55d-07,
-3.48d-07, -2.09d-07

earth j harmonic terms

xxinit(2)

(RELBET)

xxinit(2)

NAME

xxinit - input parameters for initial time and state

SYNOPSIS

DOUBLE PRECISION trvint(7,2)

DESCRIPTION

trvint

init t,r,v :: DOUBLE PRECISION trvint(7,2)

initial time and state. time is seconds since base time and occurs in entry 1,i. state is m50 cartesian and occurs as position, velocity in terms 2-7,i. here i refers to the vehicle

xxka1(2)

(RELBET)

xxka1(2)

NAME

xxka1 - input parameters for inputs for kalman filter

SYNOPSIS

DOUBLE PRECISION dtmax
DOUBLE PRECISION edcrit(25)
DOUBLE PRECISION var(20)
INTEGER lrmmdl
DOUBLE PRECISION ncons(10)
DOUBLE PRECISION undrwt(2)
INTEGER sxcld(24)

DESCRIPTION

dtmax

max step :: DOUBLE PRECISION dtmax

5.0d0

maximum allowed time step for kalman filter

edcrit

edit criterion :: DOUBLE PRECISION edcrit(25)

if abs(resid) > sigma*edcrit for a given observation, then that observation is edited

var

observation variances :: DOUBLE PRECISION var(20)

30.d0, .0027d0,
.0027d0, .1d0,
.5d-3, .5d-3,
.5d-4, .5d-4,
.45d-3, .45d-3,
10*0.d0

These are the default observation sigmas reset to variances on initialization process.

lrmmdl

noise model option :: INTEGER lrmmdl

This option implements the lrbet3 range, range rate, and range bias noise models :=0,do not use;>0,use

ncons

lrbet3 noise model constants :: DOUBLE PRECISION ncons(10)

2400.d0,9000.d0,36000.d0,8.0d0,
15000.d0,93750.d0,300000.d0,8.0d0,
.1d0,.01d0

These are the noise model constants: (1-4),range ;(5-8),range bias;(9,10),range rate

undrwt

underweighting options :: DOUBLE PRECISION undrwt(2)

undrwt(1) >0, implements the underweighting of the update by adjusting the

xxka1(2)

(RELBET)

xxka1(2)

computed residual variance by a factor of (1-undrwt).

Note: undrwt(1) =0, no underweighting

undrwt(2) , sets the criterion for underweighting .ie., whenever
RSS(relative position sigmas)**2>=undrwt(2)**2.

sxcld

state exclusion flags :: INTEGER sxcld(24)

if the sxcld flag corresponding to an element in the filter state vector
is 1, then that element is excluded from consideration in the filter
processing

xxmas(2)

(RELBET)

xxmas(2)

NAME

xxmas - input parameters for mass table

SYNOPSIS

DOUBLE PRECISION mastab(3,10,2)

DESCRIPTION

mastab

mass tables :: DOUBLE PRECISION mastab(3,10,2)

vehicle mass tables... contain up to 10 entries for 2 vehicles.

mastab(1,*,*)= start time of entry in seconds since base

mastab(2,*,*)= mass at start time

mastab(3,*,*)= rate of change of mass

entry is in effect from start time of entry to start time of next entry

xxmast(2)

(RELBET)

xxmast(2)

NAME

xxmast - input parameters for downlist formatter

SYNOPSIS

INTEGER hifile
INTEGER hiunit
INTEGER maxfil
INTEGER maxtyp
INTEGER tcrflt
DOUBLE PRECISION timoff

DESCRIPTION

hifile

(unused) max unit allowed for internal files :: INTEGER hifile
default=10

hiunit

(unused) highest unit :: INTEGER hiunit
the highest unit number used for assigned files

maxfil

maximum number of files :: INTEGER maxfil
the maximum number of files to be written to be used for dimension and looping purposes

maxtyp

maximum number of m/sid's :: INTEGER maxtyp
the maximum number of m/sid's to be processed on any file to be used for dimension and looping purposes

tcrflt

time current file :: INTEGER tcrflt
internal id for t.current.fil and time-homo set processing flag

timoff

time off :: DOUBLE PRECISION timoff
delta difference to check time-tags against the record clock times

xxmax(2)

(RELBET)

xxmax(2)

NAME

xxmax - input parameters for number of vehicles

SYNOPSIS

INTEGER nveh

DESCRIPTION

nveh

number of vehicles :: INTEGER nveh

1

xxmisc(2)

(RELBET)

xxmisc(2)

NAME

xxmisc - input parameters for miscellaneous

SYNOPSIS

CHARACTER*60 jobdes(2)
INTEGER obsrvr
INTEGER target

DESCRIPTION

block containing miscellaneous
bookkeeping and such

jobdes

job description :: CHARACTER*60 jobdes(2)
2*'no job description given'

obsrvr

id of observer vehicle :: INTEGER obsrvr
1

target

id of target vehicle :: INTEGER target
2

xxmoon(2)

(RELBET)

xxmoon(2)

NAME

xxmoon - input parameters for moon constants

SYNOPSIS

DOUBLE PRECISION mumoon
DOUBLE PRECISION dtmoon
DOUBLE PRECISION tmoo0
DOUBLE PRECISION rvmo0(6)
INTEGER nmord

DESCRIPTION

mumoon

moon mu :: DOUBLE PRECISION mumoon

moon gravitational parameter

dtmoon

moon dt :: DOUBLE PRECISION dtmoon

time step for frequency of moon ephemeris

tmoo0

moon state time :: DOUBLE PRECISION tmoo0

base time for rvmo0

rvmo0

init moon state :: DOUBLE PRECISION rvmo0(6)

m50 position of moon relative to earth used to generate moon ephemeris

nmord

moon intrp order :: INTEGER nmord

number of ephemeris points used in interpolation of moon state

xxnflz(2)

(RELBET)

xxnflz(2)

NAME

xxnflz - input parameters for information regarding display files

SYNOPSIS

CHARACTER*72 fname(5)
INTEGER bfopt
INTEGER posx(2)
INTEGER lordx(2)
INTEGER print
INTEGER plot
INTEGER trjout(3)

DESCRIPTION

fname

file names :: CHARACTER*72 fname(5)

specifies names of files used. uses are as follows for all

- 1 input first ephemeris or relative trajectory file
- 2 input second ephemeris or relative trajectory file

**note that posx determine which vehicle corresponds to which file for xqdsp

- 3 input attitude file
- 4 output plot file
- 5 not used for xcmpar
- 3 output ephemeris for vehicle 1
- 4 output ephemeris for vehicle 2
- 5 output relative trajectory of veh 2 with veh 1 base

bfopt

base file flag :: INTEGER bfopt

specifies id of file to use as base. output times correspond to the times of this file and other input information is interpolated to these times. if this option is chosen, then the delta time in time in the times input specifies a minimum number of records between consecutive output times. the following options are valid

- 1 first ephemeris (xqdisp,xcmpar)
- 2 second ephemeris (xqdisp,xcmpar)
- 3 attitude file (xqdisp)
- else error

posx

location for data :: INTEGER posx(2)

absolute value give input file id, either file 1 or 2. the sign specifies whether the state (+) or the relative state (-)

lordx

xxnflz(2)

(RELBET)

xxnflz(2)

interpolation flag :: INTEGER lindx(2)

specifies interpolation option for file 1. options are

-1 from base file (set by program, not user input)

1 use two point position and velocity interpolation

2-10 use lagrangian interpolation with the specified number of points

print

print flag :: INTEGER print

positive value to generate print to unit out

plot

plot flag :: INTEGER plot

positive value to save plot file to file 4

trjout

traj flags :: INTEGER trjout(3)

positive value to generate the specified file. note that this is used by xcmpar and ignored by xqdsp

1 generate ephemeris for vehicle 1

2 generate ephemeris for vehicle 2

3 generate relative trajectory, base =veh1, rel = vehicle 2

nnois(2)

(RELBET)

nnois(2)

NAME

nnois - input parameters for local input for noise analyzer

SYNOPSIS

CHARACTER*72 fnam
CHARACTER*72 fnamo
INTEGER nobs
CHARACTER*4 obaray(30)
DOUBLE PRECISION obdelt(30)

DESCRIPTION

fnam

input file name

fnamo

output file name

nobs

number of observations to process

obaray

array of frame ids

obdelt

array of time interval lengths w/r to obs type to process for noise computation

xypmmx(2)

(RELBET)

xypmmx(2)

NAME

xypmmx - input parameters for minimum and maximum limits on the axes

SYNOPSIS

REAL pmmx(2,21)

DESCRIPTION

pmmx

min/max values :: REAL pmmx(2,21)

min/max values for y axis

xxprnt(2)

(RELBET)

xxprnt(2)

NAME

xxprnt - input parameters for controls for printed output

SYNOPSIS

INTEGER lpage
INTEGER col80
INTEGER header
INTEGER nfrmat

DESCRIPTION

lpage

lines/page :: INTEGER lpage

0

specifies number of lines per page. if negative or zero, then no paging is provided.

col80

column width option :: INTEGER col80

0

positive value sets formats for 80 columns. otherwise, 130 columns.

header

header print option :: INTEGER header

0

positive value causes a header of time and scale information to be printed for each output time

nfrmat

format id :: INTEGER nfrmat

1

specifies output format. options are

1 floating point

else fixed point

xxprop(2)

(RELBET)

xxprop(2)

NAME

xxprop - input parameters for prop options

SYNOPSIS

INTEGER rvopt(2)
INTEGER paero(2)
INTEGER pcb(2)
INTEGER pdrag(2)
INTEGER pharm(2)
INTEGER pmoon(2)
INTEGER prad(2)
INTEGER psvel(2)
INTEGER psun(2)
INTEGER pvent(2)
DOUBLE PRECISION dtnom(2)

DESCRIPTION

rvopt

prop option :: INTEGER rvopt(2)

flag for how vehicle state is obtained options are

<0 interpolate

1 runge kutta integration

2 super g integration

e error

paero

aero force :: INTEGER paero(2)

pcb

central body force :: INTEGER pcb(2)

pdrag

drag force :: INTEGER pdrag(2)

pharm

harmonics force :: INTEGER pharm(2)

pmoon

moon force :: INTEGER pmoon(2)

prad

solar radiation flag :: INTEGER prad(2)

xxprop(2)

(RELBET)

xxprop(2)

psvel

sensed velocity flag :: INTEGER psvel(2)

psun

sun force :: INTEGER psun(2)

pvent

vent force :: INTEGER pvent(2)

dtnom

nom step :: DOUBLE PRECISION dtnom(2)

specifies largest step size in integration step

NAME

xxqcrv - input parameters for curve info

SYNOPSIS

```

CHARACTER*8 kname( 20 )
CHARACTER*8 kparms( 3,20 )
INTEGER imrk( 20 )
INTEGER nlabel( 20 )
INTEGER ksmbol( 20 )
CHARACTER*72 kfile( 20 )
INTEGER kedit( 20 )
INTEGER kline( 20 )
REAL kstep( 20 )
REAL kspan( 2,20 )
REAL psize( 20 )

```

DESCRIPTION

kname

curve names :: CHARACTER*8 kname(20)

20*'\$'

user defined curve names. Up to 7 characters can be used for curve names. The last character is reserved for dollar sign.

kparms

parameter id's :: CHARACTER*8 kparms(3,20)

60*'\$'

parameter id's. It must match exactly in order to compute idxyz

imrk

line option of curves :: INTEGER imrk(20)

symbol frequency : Specify the frequency of plot symbols for the corresponding curve. For a value k :

k > 0 symbol every kth point. Line through each point.

k = 0 no symbols. line through each point.

k < 0 symbol every kth point . No line.

nlabel

point label options :: INTEGER nlabel(20)

label frequency: Specifies the frequency of integer labels for curve points.

N < 0 label every Nth point in the sequence 1,2,3,...

N > 0 label every Nth point in the sequence 1,N,2N,..

N = 0 no labels

ksmbol

curve plot symbol code :: INTEGER ksmbol(20)

Specifies the plotting symbols for corresponding curve. For more detail on the different symbol codes ,see DISSPLA manual for symbols and their corresponding sequence numbers.

kfile

curve file :: CHARACTER*72 kfile(20)

xxqcrv(2)

(RELBET)

xxqcrv(2)

20* '????'

Input file name. It specified where the input datas come from. Different curves can have different input files.

kedit

edit plot options :: INTEGER kedit(20)

kline

line type code :: INTEGER kline(20)

Specifies the line type option for curve. Option are :

- 0 connected
- 1 Dot
- 2 Dash
- 3 Chained dot
- 4 Chained dot dash

kstep

step size :: REAL kstep(20)

curve interval : specifies the plot interval for curve i.e., begin, end, and step size. Times are specified as seconds from base time. If kstep > 0 then use in time seconds, else use counts

kspan

begin,end :: REAL kspan(2,20)

1.,2.d8, 1.,2.d8, 1.,2.d8, 1.,2.d8,
1.,2.d8, 1.,2.d8, 1.,2.d8, 1.,2.d8,
1.,2.d8, 1.,2.d8, 1.,2.d8, 1.,2.d8,
1.,2.d8, 1.,2.d8, 1.,2.d8, 1.,2.d8,
1.,2.d8, 1.,2.d8, 1.,2.d8, 1.,2.d8

begin and end time span (time or count depend)

psize

plot symbol size :: REAL psize(20)

18*1.

specifies the the size of the plot symbols.

xxqgen(2)

(RELBET)

xxqgen(2)

NAME

xxqgen - input parameters for graphic device info

SYNOPSIS

CHARACTER*4 qpdev
REAL sthite

DESCRIPTION

qpdev

device for plot :: CHARACTER*4 qpdev

'hp'

sthite

standard symbol height :: REAL sthite

0.14

NAME

xxqprm - input parameters for parameter spec tables

SYNOPSIS

```
INTEGER pmxflg( 21 )
INTEGER pword( 21 )
CHARACTER*4 pfid( 21 )
CHARACTER*12 punits( 21 )
CHARACTER*8 pname( 21 )
REAL pscale( 21 )
REAL pofset( 21 )
```

DESCRIPTION

pmxflg

min/max options :: INTEGER pmxflg(21)

specifies whether min/max check is performed for parameter. The scaled parameter with offset subtracted is compared against the values specified by xmmx,ymmx. Options are:

- 2 used in Ascale only. Ascale does not scale this parameter.
- 1 plot out range value at extremal values.
- 0 no check
- 1 omit all out range points
- 2 omit points above minimum, plot points below minimum at min values
- 3 omit points above maximum, plot points below maximum at max values

pword

index to data word :: INTEGER pword(21)

1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11 , 12 , 13 , 14 , 15 ,
16 , 17 , 18 , 19 , 20 , 0

This is where the actual data resides. pword(21) is reserved for time and is set to zero.

pfid

frame id's :: CHARACTER*4 pfid(21)

20*'????' , 'time'

specifies the frame type corresponding to the parameter. Only frame with the specified frame type will be used in obtaining the parameter. Special options are:

- '????' wild card: consider all frames.
- 'time' time word wild card. When referenced by a curve, the frame type of the other parameter is assumed and the time word of that frame is plotted.
- else frame id from file must exact match

punits

units names :: CHARACTER*12 punits(21)

21*'ntrnl'

specifies the units names for parameters

pname

parameter names :: CHARACTER*8 pname(21)

'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p',

xxqprm(2)

(RELBET)

xxqprm(2)

'r','s','t','u','time'

user defined parameter names. pname(21) is reserved for TIME only. Up to 7 characters can be used for pname. The last character is reserved for dollar sign.

pscale

scale factors :: REAL pscale(21)

21*1

specifies the scale factor for parameters as internal units per display unit. the default scale is 1.

pofset

offset :: REAL pofset(21)

value that is subtracted from parameter before display. specified in the units determined by pscale. note that min/max are checked after the offset is removed and thus should be relative to the offset

xxrnp0(2)

(RELBET)

xxrnp0(2)

NAME

xxrnp0 - input parameters for rnp parameters

SYNOPSIS

DOUBLE PRECISION trnp0
DOUBLE PRECISION rnp0(3,3)
DOUBLE PRECISION cdetut

DESCRIPTION

parameters related to the true of date to m50 transformation

trnp0

rnp time :: DOUBLE PRECISION trnp0

0.0d0

specifies time in seconds from base time at which the base rnp matrix is anchored.

rnp0

rnp matrix :: DOUBLE PRECISION rnp0(3,3)

1.2d0 , 8*0.d0

true of date to m50 transformation matrix at base time. computed by program if abs(rnp0(1,1))>1.2 else input used

cdetut

et offset :: DOUBLE PRECISION cdetut

0.d0

specifies offset between ephemeris time and universal time

cdetut(1)= et - ut in seconds

cdetut(2)= rate of et - ut increase in seconds/second

xrpst(2)

(RELBET)

xrpst(2)

NAME

xrpst - input parameters for roll-pitch to shaft-trunnion info

SYNOPSIS

CHARACTER*72 infil
CHARACTER*72 outf11
DOUBLE PRECISION antang

DESCRIPTION

infil

input obs file name :: CHARACTER*72 infil

outf11

output obs file name :: CHARACTER*72 outf11

antang

antenna angle :: DOUBLE PRECISION antang

1.169370564

antenna angle for radar entry in radians

xxscov(2)

(RELBET)

xxscov(2)

NAME

xxscov - input parameters for state covariances

SYNOPSIS

```
INTEGER iopt
DOUBLE PRECISION rvcov( 6,6,2 )
DOUBLE PRECISION sncon( 2 )
```

DESCRIPTION

iopt

input option on state covariances UVW :: INTEGER iopt

0

iopt = 1 indicates that the input target UVW covariance is referenced to the base state UVW coordinate frame. iopt = 0 indicates that the input target UVW covariance is referenced to the target inertial frame.(similar to Lear inputs)

rvcov

position, velocity covariances :: DOUBLE PRECISION rvcov(6,6,2)

```
140.d0      ,3*0.d0, -.94d0 ,2*0.d0,
1060.d0     ,0.d0  , -.99d0 ,4*0.d0,
70.d0      ,4*0.d0, -.99d0 ,0.d0  ,
1.22d0     ,2*0.d0, -.94d0 ,3*0.d0,
.13d0      ,6*0.d0,
.08d0
140.d0      ,3*0.d0, -.94d0 ,2*0.d0,
1060.d0     ,0.d0  , -.99d0 ,4*0.d0,
70.d0      ,4*0.d0, -.99d0 ,0.d0  ,
1.22d0     ,2*0.d0, -.94d0 ,3*0.d0,
.13d0      ,6*0.d0,
.08d0
```

rvcov(1,1,1) is the position, velocity covariance for the base state.
rvcov(1,1,2) is the initial position, velocity covariance for the relative state.

sncon

constants in the state noise computation :: DOUBLE PRECISION sncon(2)

1.603d-6, .6412d-6

a value of 2.5648e-7 m**2/sec**3 corresponds to an uncertainty of 400 m downrange in the base state per orbit. a value of relative state per orbit. To increase the downrange error by a factor of n, multiply the corresponding sncon by n**2.

xxsen(2)

(RELBET)

xxsen(2)

NAME

xxsen - input parameters for sensor characteristics

SYNOPSIS

INTEGER maxobs
INTEGER maxsen
INTEGER icoas
INTEGER iradar
INTEGER itrky
INTEGER itrkz
INTEGER isen1
INTEGER isen2
INTEGER isen3
DOUBLE PRECISION rsob(3,7)
DOUBLE PRECISION qbs(4,7)

DESCRIPTION

maxobs

maximum number of obs :: INTEGER maxobs

16

maxsen

maximum number of sensors :: INTEGER maxsen

4

icoas

coas id :: INTEGER icoas

4

iradar

rendevous radar id :: INTEGER iradar

1

itrky

y star tracker id :: INTEGER itrky

3

itrkz

z star tracker id :: INTEGER itrkz

2

xxsen(2)

(RELBET)

xxsen(2)

isen1

1st extra sensor id :: INTEGER isen1

isen2

2nd extra sensor id :: INTEGER isen2

isen3

3rd extra sensor id :: INTEGER isen3

rsob

sensor offsets :: DOUBLE PRECISION rsob(3,7)

.139445122d2, .33933d1, -.176524d1,
.173104302d2, -.28194d0, -.124206d1,
-.1649d2, -.525d1, .29d1,
.1386078d2, -.1408d1, -.322326d1,
3*0.0d0

qbs

sensor attitude quaternions m50 to sensor :: DOUBLE PRECISION qbs(4,7)

.8338858225201d0, 0.d0, 0.d0, .5519369829422d0,
.0264059344345d0, .7065011118937d0, .7072182848987d0, .0010958282010d0,
.0640725060379d0, -.0653106301822d0, .7050333478320d0, -.7032476190239d0,
.0d0, .0d0, .1d1, .0d0,
1.0d0, 3*0.0d0

NAME

xxsprm - input parameters for controls for products contents

SYNOPSIS

CHARACTER*40 gmsg
 CHARACTER*8 seq
 INTEGER spg(20)
 REAL dict(3)
 REAL datbuf(2)

DESCRIPTION

gmsg

generic message :: CHARACTER*40 gmsg

generic message in dpf header

seq

file sequence id :: CHARACTER*8 seq

eight character identifier sequence unique to each product file. contents are 'fnsqsfvr' where

fn flight number
 sq starting sequence number for first file on tape
 sf sequence flag where
 00=sq only one sequence on file
 01=sq start of first sequence
 nn=sq start of sequence nn
 rv revision number

spg

display group flags :: INTEGER spg(20)

positive value specifies parameter groups to display options are

1	orbiter gmt
2	mcc gmt
3	ground elapsed time
4	orbiter m50 state
5	orbiter in target lvlh
6	orbiter in target uvw
7	orbiter euler angles to uvw
8	orbiter m50 attitude matrix
9	orbiter m50 quaternion
10	orbiter attitude rate
11	look angle and rates to target
12	range and range rate
13	simulation flag (used for data drop out info instead)
	1 = data in tracking intervals (default)
	0 = data not in tracking intervals
14	target m50 state
15	target m50 relative state
16	target in orbiter uvw
17	target in orbiter lvlh

dict

dictionary record :: REAL dict(3)

dictionary record format

datbuf

data buffer :: REAL datbuf(2)

xxsprm(2)

(RELBET)

xxsprm(2)

data buffer used for creating Univac tape

xxsp1m(2)

(RELBET)

xxsp1m(2)

NAME

xxsp1m - input parameters for print time info

SYNOPSIS

DOUBLE PRECISION pfreq
DOUBLE PRECISION sptol
DOUBLE PRECISION sptime(20)

DESCRIPTION

pfreq

print frequency :: DOUBLE PRECISION pfreq

4.0d0

nominal print frequency

sptol

tolerance :: DOUBLE PRECISION sptol

tolerance within which special time is printed

sptime

special times :: DOUBLE PRECISION sptime(20)

20*1.0d38

table of special print times as seconds since base time

xxsun(2)

(RELBET)

xxsun(2)

NAME

xxsun - input parameters for sun constants

SYNOPSIS

DOUBLE PRECISION musun
DOUBLE PRECISION dtsun
DOUBLE PRECISION tsun0
DOUBLE PRECISION rvsun0(6)
INTEGER nsord
DOUBLE PRECISION ksun
DOUBLE PRECISION kear
DOUBLE PRECISION kflux
DOUBLE PRECISION kealb

DESCRIPTION

musun

sun's mu :: DOUBLE PRECISION musun

sun gravitational parameter

dtsun

sun eph time step :: DOUBLE PRECISION dtsun

specifies time between consecutive points of sun ephemeris

tsun0

sun base time :: DOUBLE PRECISION tsun0

specifies time tag of rvsun0 as time since base time

rvsun0

init sun state :: DOUBLE PRECISION rvsun0(6)

m50 state of earth relative to sun used to generate
the sun ephemeris

nsord

sun intrp order :: INTEGER nsord

specifies number of points used in interpolation sun state
from ephemeris

ksun

1 au solar force :: DOUBLE PRECISION ksun

solar force on sphere at a distance of 1 au

kear

earth reflect :: DOUBLE PRECISION kear

earth reflection constant

kflux

flux :: DOUBLE PRECISION kflux

solar flux

xxsun(2)

(RELBET)

xxsun(2)

kealb

albedo :: DOUBLE PRECISION kealb

earth albedo

xxsvbi(2)

(RELBET)

xxsvbi(2)

NAME

xxsvbi - input parameters for bias info

SYNOPSIS

INTEGER psvb(2)
DOUBLE PRECISION svbtb(8,10,2)

DESCRIPTION

psvb

bias flag for file 1 :: INTEGER psvb(2)

svbtb

bias table for file 1 :: DOUBLE PRECISION svbtb(8,10,2)

160*1.d20

xxtime(2)

(RELBET)

xxtime(2)

NAME

xxtime - input parameters for time info

SYNOPSIS

INTEGER date(5)
DOUBLE PRECISION dates
DOUBLE PRECISION tbegin
DOUBLE PRECISION tend
DOUBLE PRECISION delta
INTEGER endopt
DOUBLE PRECISION endval

DESCRIPTION

date

base date :: INTEGER date(5)

1985 , 1 , 0 , 0 , 0

base date year,month,day,hour,minute,second

dates

base sec :: DOUBLE PRECISION dates

0.0

seconds in base date

tbegin

beg time :: DOUBLE PRECISION tbegin

-1.d30

begin time as seconds since base time

tend

end time :: DOUBLE PRECISION tend

1.d30

end time as seconds since base time

delta

time step :: DOUBLE PRECISION delta

1

time step

endopt

term opt :: INTEGER endopt

0

termination option

endval

term val :: DOUBLE PRECISION endval

xxtime(2)

(RELBET)

xxtime(2)

1.d30

value for termination

xxtoff(2)

(RELBET)

xxtoff(2)

NAME

xxtoff - input parameters for product time info and misc.

SYNOPSIS

INTEGER ldate(6)
INTEGER dtdate(6)
DOUBLE PRECISION dtbias(2)
INTEGER qafreq
DOUBLE PRECISION tlapse
INTEGER drive
INTEGER tape
INTEGER hpbin
DOUBLE PRECISION dtable(50,2)

DESCRIPTION

ldate

launch date :: INTEGER ldate(6)

launch date as ymdhms

dtdate

time bias date :: INTEGER dtdate(6)

date of mcc/shuttle time bias as ymdhms. dtbias specifies bias and rate

dtbias

time bias and rate :: DOUBLE PRECISION dtbias(2)

time bias and drift rate for mcc/shuttle time bias. the offset is given by

$gt = st + dtbias(1) + dtbias(2)*(st-tb)$ where

gt = ground time

st = shuttle time

tb=time tag of bias as

specified by dtdate (offset from base date)

qafreq

qa rec print freq :: INTEGER qafreq

specifies frequency of print to print unit in terms of records. this is not the same as the microfiche print which is at the same frequency as that specified by delta for the tape.

tlapse

summary frequency (min) :: DOUBLE PRECISION tlapse

specifies the number of minutes between status messages that are displayed to the terminal. this value is converted to seconds internally

drive

tape drive :: INTEGER drive

tape

tape option :: INTEGER tape

xxtoff(2)

(RELBET)

xxtoff(2)

if tape > 0 univac tape is produced

hpbin

hp binary file :: INTEGER hpbin

if hpbin > 0 then hp binary file is produced

dtable

data dropout table :: DOUBLE PRECISION dtable(50,2)

data dropout time intervals table

NAME

xxusys - input parameters for unit system info

SYNOPSIS

```
INTEGER usysex( 5 )
INTEGER usysin
CHARACTER*4 usyst( 5 )
CHARACTER*4 nsclz( 10,5 )
DOUBLE PRECISION usclz( 10,5 )
```

DESCRIPTION

usysex

display scale flags :: INTEGER usysex(5)

1,4*0

positive value indicates that print is to be scaled by corresponding system of units

usysin

input unit system id :: INTEGER usysin

0

id of unit system to scale input with

usyst

unit system names :: CHARACTER*4 usyst(5)

5*' '

user specified abbreviation for corresponding unit system

nsclz

scale factor names :: CHARACTER*4 nsclz(10,5)

50*' '

user specified names for the scale factors in each system of units.

usclz

scale factors :: DOUBLE PRECISION usclz(10,5)

50*1.d0

scale factors as internal units per specified unit details are

1	length (meter default)
3	angle (radian default)
4	time (sec default)

xxvcx(2)

(RELBET)

xxvcx(2)

NAME

xxvcx - input parameters for force info

SYNOPSIS

DOUBLE PRECISION aeroc(2)
DOUBLE PRECISION cd(2)
DOUBLE PRECISION chord(2)
DOUBLE PRECISION c(3,2)
DOUBLE PRECISION d(5,2)
DOUBLE PRECISION kd(2)
DOUBLE PRECISION drgfac(2)
DOUBLE PRECISION rbarna(3,2)
DOUBLE PRECISION sref(2)
DOUBLE PRECISION sarea(2)
DOUBLE PRECISION srflec(2)
DOUBLE PRECISION solfac(2)
INTEGER horder(2)
INTEGER hdgree(2)
DOUBLE PRECISION sagate(2)

DESCRIPTION

aeroc

drag numbers :: DOUBLE PRECISION aeroc(2)

cd

drag coefficient :: DOUBLE PRECISION cd(2)

2.2 , 2.2

chord

vehicle chord :: DOUBLE PRECISION chord(2)

c

cylindrical fit parameters :: DOUBLE PRECISION c(3,2)

d

zutek fit parameters :: DOUBLE PRECISION d(5,2)

kd

drag multiplier :: DOUBLE PRECISION kd(2)

drgfac

aerodynamic drag factor :: DOUBLE PRECISION drgfac(2)

xxvcx(2)

(RELBET)

xxvcx(2)

rbarna

actual cg pos rel to nominal cg :: DOUBLE PRECISION rbarna(3,2)

sref

vehicle reference area :: DOUBLE PRECISION sref(2)

250.0 , 2.926

sarea

area for solar rad :: DOUBLE PRECISION sarea(2)

srfllec

vehicle reflectivity :: DOUBLE PRECISION srfllec(2)

sofac

solar radiation factor :: DOUBLE PRECISION sofac(2)

horder

max order of harmonics :: INTEGER horder(2)

4 , 4

hdgree

maximum degree of harmonics :: INTEGER hdgree(2)

4 , 4

sagate

accel gate :: DOUBLE PRECISION sagate(2)

0.00001369 , 0.0

threshold for sensed accelerations to be included in propagation. if magnitude of the acceleration does not exceed this threshold, then the acceleration is omitted.

xxvnt(2)

(RELBET)

xxvnt(2)

NAME

xxvnt - input parameters for vent info

SYNOPSIS

DOUBLE PRECISION vnttab(4,10,2)

DESCRIPTION

vnttab

vent timeline :: DOUBLE PRECISION vnttab(4,10,2)

80 * 1.0d20

two timeline entries based on last index. max of 10 entries per timeline.
Each entry consists of the following

- 1 start time of vent
- 2 body x force
- 3 body y force
- 4 body z force

xxwrit(2)

(RELBET)

xxwrit(2)

NAME

xxwrit - input parameters for write to file info

SYNOPSIS

CHARACTER*12 files(10)
INTEGER recpnt(2,10,10)
INTEGER nfiles
CHARACTER*12 vars(10,10)
INTEGER nvars(10)

DESCRIPTION

files

available Datain files to write to :: CHARACTER*12 files(10)

'init' ,
'time' , 8* ' '

recpnt

pointers to record numbers in a file :: INTEGER recpnt(2,10,10)

1 , 1 , 2 , 6 , 8 , 1 , 9 , 6 , 12*0,
7 , 1 , 8 , 1 , 11 , 1 , 174*0

nfiles

number of available Datain files :: INTEGER nfiles

2

vars

names of variables in the files :: CHARACTER*12 vars(10,10)

'veh1 time' , 'veh1 state' , 'veh2 time' , 'veh2 state' ,
5* ' ', 'tbegin' , 'tend' , 'endval' , 88 * ' '

nvars

number of variables in each file :: INTEGER nvars(10)

4,3,8*0

APPENDIX III - ARTICLES

APPENDIX III - ARTICLES

The following articles provide additional information on several programs. These processes are in general too complex to be digestable in the manual entry format found in Appendix I. They include

- o The Linput Input Language
- o The Downlist Formatting Processor
- o The Sequential Kalman Filter
- o The Graphic Display Process
- o Special Products Process
- o Noise Analysis Process

LINPUT INPUT LANGUAGE

TABLE OF CONTENTS

	Page
1.0 INTRODUCTION.....	AIII-4
2.0 OVERVIEW	AIII-4
3.0 LEXICAL BASIS.....	AIII-5
3.1 WHITESPACE.....	AIII-5
3.2 IDENTIFIERS.....	AIII-6
3.3 CONSTANTS.....	AIII-6
3.4 OPERATORS.....	AIII-7
3.5 PUNCTUATION.....	AIII-7
4.0 STORAGE	AIII-7
5.0 EXPRESSIONS AND OPERATORS.....	AIII-8
5.1 LIST OPERATORS.....	AIII-8
5.1.1 <u>Assignment Operator</u>	AIII-8
5.1.2 <u>Concatenation Operator</u>	AIII-9
5.1.3 <u>Repeat Operator</u>	AIII-9
5.2 BINARY ARITHMETIC OPERATORS.....	AIII-9
5.3 UNARY ARITHMETIC OPERATORS.....	AIII-10
5.4 OPERATOR PRECEDENCE	
6.0 MISCELLANEOUS REMARKS.....	AIII-11

1.0 INTRODUCTION

The linput input device provides a free format input syntax for simple parameter lists. The syntax is based largely on C and allows for arithmetic operations on input values.

The following provides a description of this input device from a user's point of view. **Bold** font indicates keywords and literals in examples and format specifications.

2.0 OVERVIEW

Linput divides the input stream in specific variable sets called input segments. For example an input segment might consist of the parameters Shuttle_state, Shuttle_time, and Shuttle_mass. Each parameter or variable in the segment has a unique name that is used to reference it. The particular variables in a segment of course depend upon the application of the processor, and in general a given input variable or parameter will be contained in a unique input segment. The actual segment being processed by linput is also determined by the application program. In any event, linput reads from the standard input (stdin). Processing of the segment continues until an end of file or termination token is encountered.

The basic format of an linput segment is a sequence of assignment statements followed by a segment termination token:

```
assignment_statements
%%
```

The simplest assignment statement assigns a given value to a given variable, for example

```
Shuttle_time = 34.3e5;
```

Here one value is assigned to the specified variable. Lists of values may also be assigned to a variable, for example,

```
Shuttle_state = 15.3e5, 2.6e6, 3.56e4,  
                3.2e3, -6.8e2, 6.91e4 ;
```

Note that a semicolon statement terminator ; is required (as in PL/I, C, or Ada).

Once a variable has been assigned it may be used in subsequent expressions. Furthermore, one is free to define one's own variables. For example

```
sec = 1;      min = 60 sec;  hr = 60 min;  
Shuttle_time = (4 hr) + ( 31 min) + (9.235 sec);
```

The last example also illustrates the ability to perform arithmetic operations.

3.0 LEXICAL BASIS

The input text is analyzed into lexical "tokens". The format is free in that whitespace is ignored.

3.1 WHITESPACE

Whitespace is ignored and used to delineate tokens. In addition to normal whitespace characters (space, tabs, newlines), comments are treated as whitespace. Comments can be inserted in either of two ways:

- o Enclose characters between /* and */,
- o Use // to comment all characters to the next newline

Note that comments using /* and */ cannot be nested. However such comments can contain comments using //. Note also that // will comment any /* or /* occurring on the remainder of the line.

3.2 IDENTIFIERS

These provide names for variables and follow the C convention:

Initial character: an upper or lower case letter or underscore (`_`)

Subsequent characters: letters, numerals or underscores.

The following are legal identifiers:

`x`

`_xyz`

`Shuttle_Position`

`vehicle_1`

3.3 CONSTANTS

These may be either strings or numeric. All arithmetic is performed as double with type conversion to int or float occurring on final storage of the variables. Again l-input follows the conventions of C. In particular strings are indicated by enclosing characters in double quotes (`"`). Thus, for example,

`"this is a string"`

`"/users/Ralph/U/clist"`

`5`

`0.32`

`5.6e-5`

are constants.

3.4 OPERATORS

The following operators are recognized:

=	assignment
,	list concatenation
+	binary addition, unary identity
-	binary subtraction, unary negation
*	multiplication
/	division
!	list repeat
^	exponentiation

3.5 PUNCTUATION

The following punctuation tokens are provided.

(left parenthesis
)	right parenthesis
;	statement terminator
%%	segment terminator (must be preceded by a newline)

The parentheses are used to specify the order of expression evaluations. The end of file is also recognized as a segment terminator.

4.0 STORAGE

Storage is allocated dynamically and all values are held in a temporary buffer until the segment terminator is encountered. Temporary values are then discarded and the required variables stored as need. Note that this protects the application program from overwrites if too many values were assigned an identifier.

All variables and expressions are interpreted as lists. Similarly all variables and expressions have values which are the list items. Constants are thus lists with one element.

5.0 EXPRESSIONS AND OPERATORS

The simplest expressions are just identifiers or constants. Complex expressions are formed from these simple expression by the use of operators and the parenthesis punctuation tokens. A statement is an expression followed by the semicolon (;) statement terminator.

5.1 LIST OPERATORS

The list operators are

= , !

They are all binary and obey the syntax

operand₁ operator operand₂

5.1.1 Assignment Operator

The assignment operator has the syntax

identifier = expression

and causes the list elements in the expression to be assigned as elements in the list specified by the identifier. The value of the assignment expression is the value of the second operand. Thus if **list₂** has the value 3,4,5,

list₁ = list₂

has the same value and 3,4,5 is stored for **list₁** as well.

5.1.2 Concatenation Operator

The concatenation operator has the syntax

expression_1 , expression_2

and results in a list with the values in the first operand followed by the values in the second operand. Using the above values, the expression

list_1 , list_2

has the value 3,4,5,3,4,5. As another example, the expression

6, 4, list_1, (list_3 = 3,5),5

has the value 6,4,3,4,5,3,5,5.

5.1.3 Repeat Operator

The repeat operator ! requires that the first operand have an integer value n. It has the syntax

n ! expression

and results in a list that is the second operand repeated n times. Thus

(repeat = 3) ! (sigma = 3.4)

has the value 3.4,3.4,3.4. This is the same as concatenating the second operand with itself n times, where n is the value of the first operand.

5.2 BINARY ARITHMETIC OPERATORS

The binary arithmetic operators are

+ - * /

They have the usual interpretation as addition, subtraction, multiplication, division and exponentiation. They follow the syntax

expression operator expression

In the case of binary arithmetic operators, the operation is applied pairwise to the elements of the two lists. If one list is longer, the last element of the shorter list continues to be applied to the remaining elements of the longer list. Thus

(1,2,3) + (2,3,4)	results in 3,5,7
3 *(1,2,3)	results in 3,6,9
(3,4)*(1,2,3)	results in 3,8,12

Note that the multiplication will be invoked implicitly if an expression is immediately followed by an identifier without an operator between them.

expression identifier

Thus the statements

ft = 0.3048; x = 3 ft;

will assign 0.9144 to x. This allows for the input of data in units other than internal units in an obvious manner.

5.3 UNARY ARITHMETIC OPERATORS

The unary arithmetic operations are

+ -

They are interpreted as identity and negation respectively. They follow the syntax

operator expression

Unary operators are applied to each of the elements of a list in turn. Thus

-(3,4,5)	results in -3,-4,-5
-(2,+(4,-2))	results in -2,-4,2

5.4 OPERATOR PRECEDENCE

Operators obey the following precedence in descending order.

```
!  
* /  
+ - (unary)  
+ - (binary)  
,  
=
```

Thus since assignment has lower precedence than concatenation the expression

```
x = 3, y = 4
```

will assign the value 4 to the list y and the values 3,4 to the list x. Note the unintended effects of precedence in the following statements.

```
ft = .3048; sec = 1;  
position = 3.2, 6.8, -1.2 ft;  
velocity = 8.9, 3.1, -6.9 ft/sec;
```

This results in the values 3.2, 6.8, -.36576 and 8.9, 3.1, -2.10312. The statements

```
position = ( 3.2, 6.8, -1.2) ft;  
velocity = ( 8.9, 3.1, -6.9) ft /sec;
```

are probably what was intended.

6.0 MISCELLANEOUS REMARKS

The parser is left recursive (it is implemented with yacc); however, no guarantees are made as to the order of expression evaluation. Thus statements like

```
diagonal_matrix = ( row = d,0,0) , 2 !(0,row);
```

should be avoided in favor of

```
row = d,0,0; diagonal_matrix = row , 2!(0,row);
```

The last expression has the value d,0,0, 0,d,0, 0,0,d, 0,0. If the token **diagonal_matrix** is to be stored as a 3 by 3 matrix, the trailing 0's are ignored on final storage.

Note that all lists are simple lists. There are no lists of lists. Thus

```
(3,4),(5,6),7
```

is equivalent to

```
3,4,5,6,7
```

Lists of strings are allowed:

```
friends = "bob", "carol", "ted", "alice";
```

DOWNLIST FORMATTER

AIII-13

TABLE OF CONTENTS

	Page
1.0 INTRODUCTION.....	AIII-15
1.1 FUNCTIONAL DESCRIPTION.....	AIII-15
1.2 READING THE IBM CCT.....	AIII-15
1.3 STRIPPING DATA.....	AIII-15
1.4 VALIDATING AND EDITING DATA.....	AIII-16
1.5 REFORMATTING TO RELBET UNITS.....	AIII-16
1.6 WRITING OUTPUT FILES.....	AIII-16
2.0 OPERATIONAL DESCRIPTION.....	AIII-17
2.1 EQUIPMENT CONFIGURATION.....	AIII-17
2.2 INPUT MATERIALS.....	AIII-17
2.3 OUTPUT MATERIALS.....	AIII-17
2.3.1 <u>Output Files</u>	AIII-17
2.3.2 <u>Printed Output</u>	AIII-18
2.4 EXECUTION PROCEDURES.....	AIII-19
3.0 INPUT DESCRIPTION.....	AIII-19
3.1 MASTER CATEGORY.....	AIII-19
3.2 FILES CATEGORY.....	AIII-23
3.3 MSIDS CATEGORY.....	AIII-23
3.4 CCT CATEGORY.....	AIII-26

1.0 INTRODUCTION

The Downlist Formatter strips required parameters from the standard orbiter downlist Computer Compatible Tape (CCT), performs transmission validity checks and necessary unit conversions, and generates various internal RELBET files.

1.1 FUNCTIONAL DESCRIPTION

The basic functions of the Downlist Formatter are to read IBM CCTs, strip desired data, check for faulty data, convert data into RELBET internal units, and write data to output files in RELBET standard file format.

1.1.1 Reading the IBM CCT

The Formatter uses HP FORTRAN 77 and C language to read the IBM EBCDIC CCT. Input MSID's are matched to MSID's stored on the CCT header and associated information such as data scan locations are obtained. Once a complete set of header records is processed, the data records are read and data scans built to be made available to the stripping functions.

1.1.2 Stripping Data

Data are externally accessible by means of internally recognized datum identification numbers so that the user may strip and process only those particular data desired.

The Formatter incorporates two modes of stripping data from the CCT which may be operated either simultaneously or individually. The nominal mode is to process each downlist data cycle sequentially. The other mode extracts the "time homogeneous" data set consisting of those special parameters buffered by the onboard NAV software at the beginning of each NAV cycle. This NAV buffer changes every NAV cycle during which approximately four downlist cycles have occurred. During these four downlist cycles the NAV buffered parameters should remain constant although the first and last cycles may see changes due

to the asynchronous nature of the downlist and NAV processing cycles. Therefore, the extraction procedure is set up to identify one of the middle downlist cycles for processing.

The extraction method follows a particular user specified parameter, TCRFLT in the MASTER Input block, which must belong to the NAV buffer. As each downlist cycle is read, TCRFLT is checked for a changed value. This should occur only once every NAV cycle or about every four downlist data cycles. The Formatter then extracts and processes the data set which is the second of the three or four duplicate sets as determined by TCRFLT.

1.1.3 Validating and Editing Data

The Formatter checks all data values for CCT error flags. Output frame time-tags are checked for monotonicity and approximate agreement with the data scan time-tag (optional). Data other than time-tags are checked to be within maximum and minimum ranges set by the user for each parameter.

As a result of the above checks, the output data frames containing bad data may be either output or deleted as specified by the user. If bad time-tags are encountered, the associated data frames are always deleted.

1.1.4 Reformatting to RELBET Units

All data types are scaled and converted to double precision. The conversion to internal RELBET units may involve an offset term as well as a conversion factor. These offset terms, scale and conversion factors are associated with the data by means of the internal identification numbers.

1.1.5 Writing Output Files

The data are written in monotonically increasing time ordered frames to files in the RELBET standard file format. Any type of file may be created. Any type of frame may be built using any group of parameters by manipulating the internal datum identification numbers. Specifically the observation data,

orbiter and target ephemerides, sensed velocities, attitudes and Sensor input files are the usual results of CCT processing.

2.0 OPERATIONAL DESCRIPTION

Equipment configuration, input/output materials, and execution procedures are discussed.

2.1 EQUIPMENT CONFIGURATION

The standard RELBET configuration is required.

2.2 INPUT MATERIALS

Input materials consist of input blocks and the Standard Orbiter Downlist CCT.

2.3 OUTPUT MATERIALS

Output materials consist of RELBET Gff files, text files, and printed displays.

2.3.1 Output Files

The user has the option to generate any or all of the following files:

- o Observation data files containing measurement data from each of the on-board tracking sensors including Radar, Star Tracker, and COAS.
- o Orbiter Ephemeris file containing inertial states of the orbiter.
- o Target Ephemeris file containing inertial states of the designated target vehicle.
- o Sensed Velocities file containing the on-board selected velocity data.

- o Attitudes files containing the on-board selected attitude quaternions.
- o SENSOR input file containing all of the information except the orbiter and target states necessary to generate the SIT and SET tapes which are input to the SENSOR program. (Note: the SENSOR program is a NASA utility).

The output files all begin and end at user specified start and stop times. Specification of these times by the user is optional as the processor will default to process data from the CCT starting with the first data record.

2.3.2 Printed Output

There are two basic printed display classes which are designated terminal and debug. The associated displays and their normal print options are as follows:

Terminal Displays (always printed)

- o Status Summaries
- o Error and Warning Messages

The user has the option to route this display to a terminal mass storage file by output redirection techniques.

Debug Display (nominally not printed)

- o Various displays for debugging and verification
- o Input information, address tables, etc.

This display will appear in a file named BUGS located in the directory of execution.

Output Display (always printed)

- o Status Summaries
- o Error warning messages

This display will appear in a file named OUTPUT located in the directory of execution.

2.4 EXECUTION PROCEDURES

The CCT should be mounted on an HP tape drive of which two designations are recognized, 0 and 1. The execution is invoked as follows:

```
dwnfmt < input
```

where **input** is a text file containing the linput input blocks described below.

3.0 INPUT DESCRIPTION

This section discusses the linput inputs available for executing the Downlist Formatter. Four linput categories of input blocks are required. These categories must be made available to the program in the order that they appear here.

- Master - Program controls
- Files - Output file names and information
- Msids - M/sid processing formats
- Cct - CCT processing information.

These blocks are separated from each other by the appearance of the linput end of file symbol **%%**. A different Files category block is required for each output file desired. In fact, a different Msids category is required for each m/sid desired, unless defaults are used. The inputs are summarized below with details referenced to Appendix II whenever applicable.

3.1 MASTER CATEGORY

This group of inputs provides the processing controls. Standard input sets belonging to this block as well as the variables used are:

xxgnrl - bugs, out, pbug
xxtime - date, dates, tbegin, tend
xxmisc - jobdes
xxmast - maxfil, maxtyp, tcrflt, timoff

Notes:

The output text associated with bugs and out are directed to files BUGS and OUTPUT respectively.

The pbug flags are described in Table 3.1.1.

The date and dates base time should reflect the base time of the downlisted GMT times and is usually the beginning of the first day of the year in which the data was created.

The input in jobdes becomes the file description on every output Gff file.

The values for the maximum number of files which can be output, Maxfil, and the number of m/sid's which can be processed, maxtyp, are 20 and 200 respectively. These values should really not change and should be considered constraints on the program.

The tcrflt input controls the "time homogeneous" data set extraction on NAV buffered parameters.

The timoff parameter is used to check the presence of erroneous data records by comparing the data scan time-tag to the output frame time tags occurring within the data scans. In general, these offsets are within 20 seconds, the default. However, SIM CCT data files have been processed which exhibit a much larger difference so that when processing SIM files, it is best to put a very large number in this slot so as to allow every comparison to be satisfactory.

Table 3.1.2 presents a sample input file for the MASTERS Category.

Table 3.1.1. PBUG Processing Options

Special option flags:

PBUG(1):

- < 0, Print detailed processing activities
- > 0, Print summary processing activities

PBUG(2):

- # 0, Print initialized input

PBUG(5):

Used to compute number of discarded frames.

PBUG(9):

- = 1, Write an octal dump to BUGS of each data record encountered at the frequency set by PBUG(10).

PBUG(10):

- 0, Specifies the frequency at which a data record reading status message is sent to terminal.

EX: PBUG(10) = 100, for every 100 data records encountered a message is printed to terminal.

PBUG(14):

- > 0, Specifies the maximum number of records which will be processed before termination. Note that end time termination takes precedence.
- < 0, Will cause termination upon encountering the first data record

Table 3.1.2. Sample Masters Input Category

```
// Masters
bugs      = 6;
out       = 6;
pbug     = 19!0;
date     = 1985, 1, 0, 0, 0;
dates_   = 0.0;
tbegin   = 0.0;
tend     = 1.0e10;
jobdes   = "Downlist Data File";
tcrflt   = 74;
timoff   = 20.0;
```

3.2 FILES INPUT CATEGORY

The purpose of this group is to set parameters associated with generating each output file. A separate input category is created for each output file being generated. The standard input set called `dwnfmt_files` belongs to this group supporting the following inputs:

`dwnfmt_files` - `fname`, `hdrid`, `filid`, `frsize`, `cfrmid`, `idseq`, `chydef`, `end`

Several default configurations set in the `dfdata` block data routine are available to create standard RELBET gff files.).

Table 3.2.1 provides an example of a group of Files input categories. In the first two blocks, the user has requested that the defaults for building files 1 and 2 be changed. In the third block the defaults are used to build file 3. It should be noted that the default replacements are actually the same as the default values themselves.

3.3 MSIDS INPUT CATEGORY

The purpose of this group is to describe the processing characteristics of each M/SID being stripped. A separate MSIDS input category is created for each M/SID being processed. The standard input set called `dwnfmt_msids` belongs to this group supporting the following inputs:

`dwnfmt_msids` - `msid`, `id_nav`, `off_sc_conv_lo_hi`

Most of the m/sid's the user could desire have been defaulted and associated with the internal datum ids in the subroutine `dfdata`. Associated with each m/sid is a conversion factor, scale factor, offset term, and validation range values. Defaults for all of these may be found in subroutine `dfdata` associated with each m/sid. A typical set of MSIDS input blocks is provided in Table 3.3.1.

Figure 3.2.1. Sample Files Input Category

```
// dwnfmt_files inputs
  fname = "ROBS1";
  hdrid = "robs";
  filid = 1;
  chgdef = 1;
  cfrmid = "aran","brnr","mrol","opit";
  idseq = -81,1000,82,192, 1000,83,193, 1000,84,194, 1000,85,194,50!0;
  frsize = 6,4,0;
%%
// dwnfmt_files inputs
  fname = "OE1";
  hdrid = "oeph";
  filid = 2;
  chgdef = 1;
  cfrmid = "oeph";
  idseq = -46,1000,47,48,49,50,51,52,50!0;
  frsize = 4,12,0;
%%
// dwnfmt_files inputs
  fname = "TE1";
  hdrid = "teph";
  filid = 3;
  end = 1;
```

Table 3.3.1. Sample MSIDS Input Categories

```
// Radar range measurement
msid = "V90U44895C";
id_nav = 82, 1;
off_sc_conv_lo_hi = 0.0, 1.0, .3048, 0.0, 1.e10;
%%
// Radar range rate measurement
msid = "V90U4896C";
id_nav = 83, 1;
off_sc_conv_lo_hi = 0.0, 1.0, .3048, -1.e10, 1.e10;
end = 1;
%%
```

3.4 CCT INPUT CATEGORY

The purpose of this group is to set parameters associated with the input block CCT. Parameters include the standard input sets:

xxcct - cct, lcynum, 0400

downfmt_cct - skip

Table 3.4.1 provides an example of a CCT Input Category.

Table 3.4.1. Sample CCT Input Category

```
// CCT inputs
cct = 0;
lcynum = 2;
040000 = 2048;
skip = 1;
%%
```


SEQUENTIAL FILTER

TABLE OF CONTENTS

	Page
1.0 INTRODUCTION.....	AIII-29
2.0 STATE VECTOR.....	AIII-30
3.0 UNITS AND COORDINATES.....	AIII-31
4.0 A PRIORI AND INITIAL ESTIMATES.....	AIII-32
5.0 OBSERVATIONS.....	AIII-32
6.0 FILTERING TECHNIQUE.....	AIII-33

1.0 INTRODUCTION

The Sequential Filter produces an estimated trajectory and associated covariances for the Shuttle and a target vehicle from an a priori estimate and onboard observations with a Kalman filtering technique.

Exactly one shuttle and one target vehicle is considered in each execution of the program.

The Sequential Filter may be executed as follows

```
sfilt < input > output_print
```

The solution file consists of a file in the standard RELBET solution file format containing state vectors and covariance matrices at each distinct observation time, within a time period specified.

For general background on Kalman filters, the user is referred to the following books: Kalman Filtering Techniques, by William M. Lear (unpublished, available from the author); Stochastic Processes and Filtering Theory, by Andrew H. Jazwinski (Academic Press, 1970); Applied Optimal Estimation, edited by Arthur Gelb (M.I.T Press, 1974); Probability, Random Variables, and Stochastic Processes, by Athanasios Papoulis (McGraw-Hill, 1965); Optimal Filtering, by Brian D. O. Anderson and John B. Moore (Prentice-Hall, 1979); and Stochastic Processes and Estimation Theory with Applications, by Touraj Assefi (Wiley, 1979). The first three of these are likely to be the most useful and accessible for users of Kalman filters in aerospace applications. In addition, the original CDC implementation of the LRBET3 filter is dealt with in Bill Lear's documents Rendezvous BET Program, LRBET3 (JSC-18638) and Modifications to LRBET3 (JSC-19022).

2.0 STATE VECTOR

The state vector consists of

- o The position of the Shuttle relative to the center of the earth
- o The inertial velocity of the Shuttle relative to the center of the earth
- o The position of the target relative to the Shuttle
- o The inertial velocity of the target relative to the Shuttle
- o The bias values for observed data.

The Shuttle state relative to the center of the earth forms the first six components of the state vector estimate for the filter.

The target state relative to the center of gravity of the Shuttle forms the second six components of the state vector estimate for the filter.

The data biases are the last ten components of the state vector estimate for the filter.

The elements of the state vector are described below:

1. X_V Mean-of-50 Shuttle vehicle center of gravity (CG) position vector.
2. Y_V
3. Z_V
4. X_V Mean-of-50 Shuttle vehicle CG velocity vector.
5. Y_V
6. Z_V
7. $X_{T/V}$ Position of the target with respect to the Shuttle CG in mean-of-to coordinates.
8. $Y_{T/V}$
9. $Z_{T/V}$
10. $X_{T/V}$ Velocity of the target with respect to the Shuttle CG in mean-of-50 coordinates.
11. $Y_{T/V}$
12. $Z_{T/V}$
13. B_{aran} Rendezvous radar (RR) range bias estimate
14. B_{csft} RR shaft angle bias estimate
15. B_{dtrn} RR trunnion angle bias estimate

- 16. B_{brnr} RR range rate measurement bias estimate
- 17. B_{izth} Z Star tracker (ST) H angle bias estimate
- 18. B_{jztv} Z Star tracker V angle bias estimate
- 19. B_{kyth} Y Star tracker (ST) H angle bias estimate
- 20. B_{lytv} Y Star tracker (ST) V angle bias estimate
- 21. B_{gcoh} COAS H angle bias estimate
- 22. B_{hcov} COAS V angle bias estimate

Units of the state vector are meters, seconds, and radians.

3.0 UNITS AND COORDINATES

The position of the Shuttle is expressed in units of meters and in the mean-of-50 coordinate system.

The velocity of the Shuttle is expressed in units of meters per second and in the mean-of-50 coordinate system.

The position of the target relative to the Shuttle is expressed in units of meters and in the mean-of-50 coordinate system.

The velocity of the target vehicle relative to the Shuttle is expressed in units of meters per second and in the mean-of-50 coordinate system.

The covariance matrix is stored in full, and output to the solution file in lower-triangular form.

The time is in GMT seconds from a user specified base reference time.

The start time and stop time are specified in GMT seconds from the reference time.

4.0 A PRIORI AND INITIAL ESTIMATES

The a priori estimate consists of a position and velocity vector for the Shuttle, a position and velocity vector for the target vehicle, the single time associated with the vectors, and statistical information for the two vectors.

The a priori estimate of the position and velocity vector for the Shuttle is relative to the center of the earth, in units of meters and meters per second, and in the mean-of-50 coordinate system.

The a priori estimate of the position and velocity of the target is relative to the center of the earth (not relative to the Shuttle), in units of meters and meters per second, and in the mean-of-50 coordinate system.

The time for the initial state vectors is expressed in GMT seconds from base reference time, and is at the start time for the output trajectory.

The a priori statistics for the Shuttle and the target vehicle can be expressed in terms of standard deviations and correlation coefficients for the two vehicles separately. No correlations between the two vehicles are allowed. The values for each vehicle are expressed in the orbit-plane coordinate system associated with that vehicle (UVW reference frame). An option exists to express the values for the target relative to the Shuttle UVW frame also.

5.0 OBSERVATIONS

The onboard observations possible are range, range rate, and shaft and trunnion angles from rendezvous radar; COAS angles; and Y and Z star-tracker angles. Pitch and roll angles derived from shaft and trunnion angles are excluded.

Range is expressed in meters, range rate is expressed in meters per second, and angles are expressed in radians (in the range from minus pi through plus pi).

COAS angles are expressed in radians (in the range from minus pi through plus pi).

Star-tracker angles are expressed in radians (in the range from minus pi through plus pi).

6.0 FILTERING TECHNIQUES

The Kalman filtering technique solves for Shuttle state, target vehicle state, and data biases.

The Kalman filtering technique requires the following computational functions:

1. Initialize the estimate and the statistics of the biases in the observed data.
2. Propagate the estimate and the statistics for the data biases.
3. Propagate a trajectory using both gravitational acceleration and sensed acceleration.

The sensed accelerations are computed from an input sensed-velocity file that is written in the standard RELBET internal file format, if it is available for the vehicle being modeled. If the sensed-velocity values are not available, then the sensed accelerations will be assumed to result from drag only.

4. Predict an observation from the estimated state vector and the attitude of the Shuttle at the time of the observation.

The attitude of the Shuttle is interpolated from a file containing attitude data at each IMU time.

The time of the observation is obtained from a file containing the observation times and the corresponding observed values.

5. Revise the estimate by incorporating the actual observation. The actual observation is obtained from a file containing the observation times and the corresponding observed values.

GRAPHIC DISPLAY PROCESS

TABLE OF CONTENTS

	Page
1.0 INTRODUCTION.....	AIII-37
2.0 TERMINOLOGY.....	AIII-37
2.1 AXIS	AIII-37
2.2 PARAMETERS.....	AIII-37
2.3 CURVES.....	AIII-38
2.4 PLOTS.....	AIII-38
2.5 SCALES.....	AIII-39
2.6 TIME SPANS.....	AIII-39
3.0 BASIC OPERATION.....	AIII-40
3.1 PLOT OPTIONS.....	AIII-40
3.1.1 <u>General Controls</u>	AIII-41
3.1.2 <u>Plot Layout</u>	AIII-41
3.1.3 <u>Curve Specifications</u>	AIII-42
3.1.4 <u>Parameter Specifications</u>	AIII-42
3.2 AUTOSCALING.....	AIII-42
3.3 OUTPUT GRAPHICS.....	AIII-43
4.0 OPERATIONAL DESCRIPTION.....	AIII-43
4.1 EQUIPMENT CONFIGURATION.....	AIII-43
4.2 INPUT MATERIALS.....	AIII-43
4.3 OUTPUT MATERIALS.....	AIII-44
4.4 EXECUTION PROCEDURES.....	AIII-44
5.0 SAMPLE INPUT AND OUTPUT DESCRIPTION.....	AIII-45
5.1 SAMPLE INPUTS.....	AIII-45
5.2 SAMPLE OUTPUT.....	AIII-45

1.0 INTRODUCTION

The Graphic Display Process provides graphic display of data from gff files. Two RELBET processors are involved in this process: **plotx** for actually plotting and **ascale** to perform the automatic scaling function. Input is via linput input blocks. The user has available a number of formatting options to tailor a display to his particular needs as well as a set of canned "default" options. These defaults provide a number of plot configuration that may be used as is or else adapted to a particular application.

2.0 TERMINOLOGY

Terms used in the following discussion are defined. The ensuing discussion is based upon these terms and their associations. Thus it is crucial to understand them.

2.1 AXIS

For a two dimensional graphic display data points are plotted against a horizontal dimension and a vertical dimension. The horizontal dimension is referred to as the X axis and the vertical dimension is referred to as the Y axis.

2.2 PARAMETERS

Parameters identify which data values are to be considered as components of the coordinates to be plotted. For example, parameters correspond to a particular data type such as the range, range rate, etc... which correspond to a particular double precision location in a gff file frame. Since edit flags and frame type locations are not double precision, these entries may not be specified for parameters. A parameter is defined by specifying the type of frame (pfid/xxqprm) and the position within the frame (pword/xxqprm) wherein it resides. The index 0 corresponds to the time word (DP word 1 of a frame)

and the indices 1, 2, ..., correspond to the data words (DP words 3, 4, ..., of a frame). Up to 20 parameters may be specified. Parameter 21 (pname(21)/xxqprm) is reserved to the time word for any frame.

2.3 CURVES

Curves identify the parameters with particular coordinate components. Curves are defined by specifying a curve name (kname/xxqcrv) and the (x,y,z) coordinate components to be plotted by giving the corresponding parameter names (pname/xxqprm) for each dimension. They are the basic entities that are displayed. Thus, curves are the basic operational level of the plot. As discussed below, the user is provided with various options such as symbols, line type, labels, etc. Up to 20 curves may be defined, however, processing considers only a user specified subset. These are referred to as "**active**" curves.

2.4 PLOTS

A plot is a particular graphic display. It consists of a display of specified coordinate components (parameters) in user specified curves over a specified time interval. It includes labels, axes, grids, etc., as specified by the user. The area occupied by the plot is called the **page**. The area bounded by the X and Y axes is called the **plot area**. Both areas are user definable. Normally a rectangle called a **frame** is drawn about the page and the axes lines enclose the plot area. The **plot origin** is the lower left hand corner of the plot area and corresponds to the minimum values on the X and Y axes. Normally all curves are plotted within the plot area. However, the user may specify a **grace margin** that defines an offset from the plot area outside of which curves will be scissored.

2.5 SCALES

The following terminology is used when referring to the various dimensions and scale factors encountered. **Internal units** refers to the units associated with parameter values as they occur in the input files. These are consistent with the standard RELBET system of units. **Display units** refer to the units or scales associated with parameters on output. **Absolute units** refer to the actual distances on the output display. They are referred to as **nominal inches** since they are assumed to be inches by the program but the actual size depends upon the particular display device.

2.6 TIME SPANS

All times are specified as time elapsed since a particular **base date**. All processing falls within a **general time span**. The user may also define curve time spans during which curves are to be plotted in two ways. The usual mode is to define a **count span** by begin and end counts. In this mode, processing starts at the beginning of the general time span. Plots start when the number of curve points encountered equals the specified begin count. It continues until either the end of the general time span is exceeded or the number of curve points encountered exceeds the end count. The other mode uses a **time span** and a **minimum time step**. Plotting starts when the time of a curve point exceeds both the curve start time and the general start time. It continues at a frequency governed by the time step: once a point is plotted, another point will not be plotted until its time-tag exceeds the previously plotted point by the given amount. Plotting continues until either the general end time or the curve end time is exceeded.

3.0 BASIC OPERATION

The Graphics Display Process provides for three basic operations: accepting and initializing input, automatic scaling for active curves, and graphic display of active curves.

There are two types of input: linput text input and gff input files. The first type specify the various control options discussed in the next section whereas the gff file input specifies the values that are plotted.

Plots are generated one at time and incorporate all active curves. The graphic output may be routed either directly to an HP 9000 terminal or to an HP 9872 plotter.

3.1 PLOT OPTIONS

The Graphic Display Process options fall in four major categories: plot layout, curve specifications, parameter specifications, and general control.

The relationship of these specifications may be understood by envisioning the parameter and curve specifications as tables with entries for each parameter or curve. The parameter specification table thus has 21 entries each of which contains information specific to that parameter. The curve specification table has 20 entries. Associated parameters are specified by referencing the appropriate entry in the parameter specification. Particular entries of the curve table are designated as active. Only these entries are considered in any operations.

The various options are described below. Those that are mandatory or crucial to the generation of a plot are underlined. Associated linput input blocks are parenthetically referenced.

3.1.1 General Controls

The program provides various general control options:

- o Graphic Output Device: Either HP 2623 terminal or HP 9872 graphic plotter. (qpdev/xxqgen)
- o Input Source Files: Up to 20 may be specified. (kfile/xxqcrv)
- o Processing Options: User may plot and autoscale. (op/xxgraf, pmxflg/xxqprm, pmmx by running program ascale)
- o Base Date: User may specify the base date . (date, dates/xxtime)
- o Define Active Curves: name(s) must match kname/xxqcrv (curves/xxgraf)
- o General Time Span: The user may specify a time span to which all processing is restricted. (tbegin, tend/xxtime)
- o Define Text I/O Units and Print Options: (in, interm, out, term, bugs, pbugs/xxgnr1)

3.1.2 Plot Layout

A plot is a particular graphic display. The general layout of the plot must be defined before any curves can be drawn. This is done automatically by the processor (use of default) or by user input. See Appendix II in the USER'S MANUAL for input input block description and default values.

- o Size and Location: Page size(xypage/xxgraf) plot area (xyarea/xxgraf) frame thickness (pframe, frmthk/xxgraf only) grace margin (gmrgin/xxgraf only), plot origin (porigin/xxgraf)
- o Axes: Type (xyaxes/xxgraf) scale (xmmx, ymmx, zmmx, xyzstp/xxgraf), grid lines (xygrid/xxgraf) labels (xlabel, ylabel, zlabel,xyhite, xyang/xxgraf)
- o Plot Titles: Up to four with scaling (title1, title2, title3, title4, tt1mul/xxgraf)
- o Legend: Scale and position (plegng, lgnpos/xxgraf)

3.1.3 Curve Specifications

Up to 20 curves may be specified.

- o Curve (kname/xxqcrv)
- o Parameters It must be the same lists of name(s) use in pname/xxqprm (kparms/xxqcrv)
- o Source File (kfile/xxqcrv)
- o Plot Symbol (ksmbol/xxqcrv), frequency (imrk/xxqcrv), and scale (psize/xxqcrv)
- o Line Type (kline/xxqcrv) and symbol type (ksymbol/xxqcrv)
- o Display Span (kparms, kspan, kstep/xxqcrv)
- o Count Labels (nlabel/xxqcrv)

3.1.4 Parameter Specifications

Up to 21 parameters may be specified.

- o Frame Type (pfid/xxqprm) and location (pword/xxqprm)
- o Scale Factor for Display (pscale/xxqprm)
- o Min/Max Values The last entry is reserved for time (pmmx)
- o Description of Parameter (pname/xxqprm)

3.2 AUTOSCALING

Scaling of axes for display may be automatic or user input. The autoscaling option provides for setting extremal values of both X and Y axes and for computing parameter extremal values.

In computing parameter extremals the input files are checked to see if any of the active curves use them for a source. If there are such curves then the file is read over the general time interval (defined by the start/stop times) and the extremal values of those parameters referenced by an active curve are computed. The computed extremal values are scaled according to the user defined scale factor and incorporate user defined offsets. The IDs of the parameters being scaled are displayed during this process.

In either event, the maximum and minimum values of all the parameters corresponding to either axis are then computed. These extrema are then rounded and tic step sizes giving roughly 5 to 10 steps per axis are computed. Note that the extrema are initialized to + or -10^{30} : very large values for axes extrema indicate that a parameter was not found in the specified input files.

3.3 OUTPUT GRAPHICS

Output plots may be routed either directly to an HP terminal or to an HP plotter. If an plotter is used, it is best to redirect output to a file, then cat the output file to the plotter for final documentation.

4.0 OPERATIONAL DESCRIPTION

Equipment configuration, input/output materials, and execution procedures are discussed.

4.1 EQUIPMENT CONFIGURATION

The Graphic Display Processor requires the standard RELBET configuration. Graphic display requires an HP graphic terminal and an HP 9872 plotter.

4.2 INPUT MATERIALS

Input materials consist of the appropriate RELBET internal file(s), and linput input blocks. The following is a list of linput input blocks:

xxgnrl (optional)
xxmisc (optional)
xxtime (mandatory)
xxqgen (mandatory)
xxqprm (mandatory)
xxpmmx (mandatory)
xxqcrv (mandatory)
xxgraf (mandatory)

4.3 OUTPUT MATERIALS

Output materials consist of any combination of the following:

- o Display on a graphic terminal (HP 2623) or plotter (HP 9872)
- o Graphic file for user to plot on HP 9872 plotter
- o Status and debug print

4.4 EXECUTION PROCEDURES

1. Run program **ascale** to get the scaling values for X and Y axes (**pmmx**) if desired by executing the command:

```
ascale < input
```

where input is a textfile of linput input blocks.

The output file is **pmmx**.

2. Edit **pmmx** and type in **pmmx =** in the first line
3. Cat the input together

```
cat pmmx input > in-input
```

4. Run **plotx < in_input**

5.0 SAMPLE INPUT AND OUTPUT DESCRIPTION

See MTFCOMMON, MTFDEFAULT in Appendix II, RELBET PROGRAMMER'S MANUAL for more details.

5.1 SAMPLE INPUTS

Figure 5.1.1 is an example of linput input for terminal graphic display.
Figure 5.1.2 is an example of linput input for HP plotter graphic display.

5.2 SAMPLE OUTPUT

Figure 5.2.1 is an example of HP terminal graphic display. Figure 5.2.2 is an example of HP plotter graphic display.

ORIGINAL SOURCE OF POOR QUALITY

```

//xxgnr1 - use default
//xxmisc -use default
//xxtime inputs
date = 1984 ,12,31 , 0 . 0 ;
dates = 0.e0 ;
tbegin = 1.84042754895701e+007 ;
tend = 1.84050890295702e+007 ;
delta = 1.e0 ;
endopt = 0 ;
endval = 1.84050890295702e+007 ; //endval (=tend)
//xxqgen inputs
// qpdev char=4: device for plot
// sthite real: standard symbol height
//xxqprm inputs
pmxflg = 2111 ;
// pmxflg(21):<>0, perform min/max
pword = 412;
// pword(21) index to data word
pfid = "aran","brnr","csft","dtrn";
// pfid(21) char=4:"????",any ;"time",use frmid of other axis
pname = "aran","brnr","csft","dtrn";
// pname(21) char=8: parameter names; pname(21) = "TIME"
pscale = 2111.;
// pscale(21) real: scale factors
// pofset(21) real: offset value subtracted from parameter
// before display in the units determined by pscale.
// note that min/max are checked after
// the offset is removed and thus should be relative to the
// offset
//xxpmmx inputs
pmmx =
.565784873962E+02, .982026748657E+02,
-.914398847388E-01, .198119968176E+00,
.737040266395E-01, .361146003008E+00,
.801412761211E-01, .242091387510E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.184042780000E+08, .184050860000E+08;
//xxqcrv inputs
kname = //
"aran","brnr","csft","dtrn";
// kname(20) char=8:curve names
kparms = //
"time","aran", " " ,
"time","brnr", " " ,
"time","csft", " " ,
"time","dtrn", " " ;
// kparms(3,20) char=8:curve parameter ids - x,y,z - knames
imrk = 410,1210 ;
// imrk(20) ;=0,line(a11);k>0,sym/line(kth);k<0,sym(kth)
ksymbol = 1813 ;
// ksymbol(20) curve plot symbol code
kfile = 41"/Users/Relbet/Test/f8obs";
// kfile(20) char=72 curve file name
kline = 411,810 ;
// kline(20) line type;0,line;1,dot;2,dash;3,chain dot
// ;4,ch dot dash
// psize(20) real:plot symbol size
//xxgraf inputs
xlabel = "time";
// xlabel char=60: specifies the label for x axis.
ylabel = "ARAN VALUES";
// ylabel char=60: specifies the label for y axis.
// zlabel char=60: z axis label
title = "Observations for 51f ";
// title char=60: main title
curves = "aran";
// curves(20) char=8:active curves;
// "aran","brnr","csft","dtrn";

```

Figure 5.1.1. Linput Input Example for Terminal Graphic Display

```

// xxgnr1 - use default
// xxmisc -use default
// xxtime inputs
date = 1984 .12.31 . 0 . 0 ;
dates = 0.e0 ;
tbegin = 1.84042754895701e+007 ;
tend = 1.84050890295702e+007 ;
delta = 1.e0 ;
endopt = 0 ;
endval = 1.84050890295702e+007 ; //endval (=tend)
// xxqgen inputs
qpdev = "dpop"; // qpdev char*4: device for plot
// sthite real: standard symbol height
// xxqprm inputs
pmxflg = 2111 ;
// pmxflg(21):<>0, perform min/max
pword = 412;
// pword(21) index to data word
pfid = "aran","brnr","csft","dtrn";
// pfid(21) char*4:"????",any ;"time",use frmid of other axis
pname = "aran","brnr","csft","dtrn";
// pname(21) char*8: parameter names; pname(21) = "TIME"
pscale = 2111.;
// pscale(21) real: scale factors
// pofset(21) real: offset value subtracted from parameter
// before display in the units determined by pscale.
// note that min/max are checked after
// the offset is removed and thus should be relative to the
// offset
// xxpmmx inputs
pmmx =
.565784873962E+02, .982026748657E+02,
-.914399847388E-01, .198119968176E+00,
.737040266395E-01, .361146003008E+00,
.801412761211E-01, .242091987510E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.000000000000E+00, .000000000000E+00,
.184042780000E+08, .184050860000E+C8;
// xxqcrv inputs
kname = //
"aran","brnr","csft","dtrn";
// kname(20) char*8:curve names
kparms = //
"time","aran", " ",
"time","brnr", " ",
"time","csft", " ",
"time","dtrn", " ";
// kparms(3,20) char*8:curve parameter ids - x,y,z - knames
imrk = 410,1210 ;
// imrk(20) ;=0,line(a11);k>0,sym/line(kth);k<0,sym(kth)
ksymbol = 1813 ;
// ksymbol(20) curve plot symbol code
kfile = 41"/users/Relbet/Test/f8obs";
// kfile(20) char*72 curve file name
kline = 210,1,810 ;
// kline(20) line type;0,line;1,dot;2,dash;3,chain dot
// ;4,ch dot dash
// psize(20) real:plot symbol size
// xxgraf inputs
xypage = 10.0,8.0; // border size
xyarea = 8.0,5.5;
lgnpos = 7.95,3.5; // legend position
xlabel = "time";
// xlabel char*60: specifies the label for x axis.
ylabel = "OBS VALUES";
// ylabel char*60: specifies the label for y axis.
// zlabel char*60: z axis label
title = "Observations for 5if ";
// title char*60: main title
curves = "csft","dtrn";
// curves(20) char*8:active curves;
// "aran","brnr","csft","dtrn";
%%
5

```

Figure 5.1.2. Linput Input Example for Plotter Graphic Display

Observations for 51f

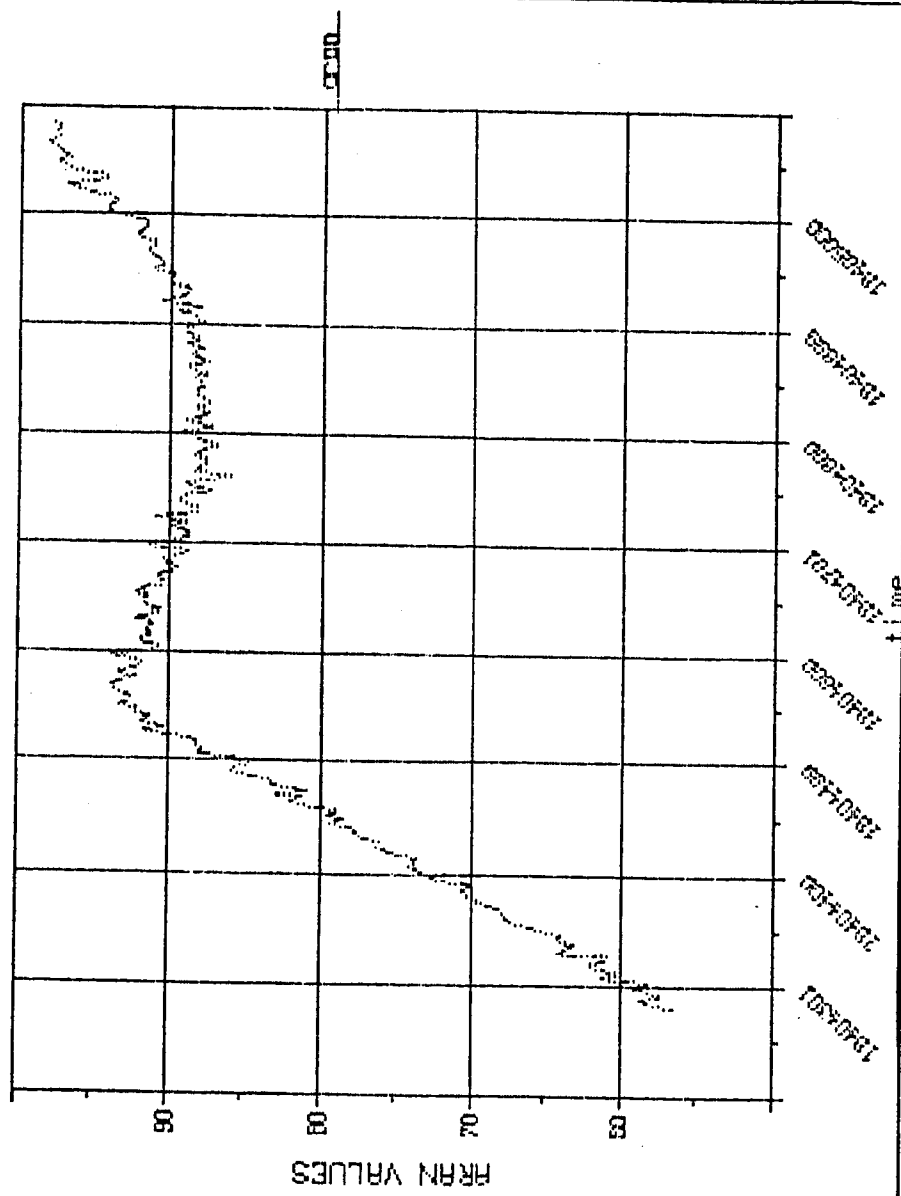


Figure 5.2.1. Example of HP Terminal Graphic Display

Observations for 51f

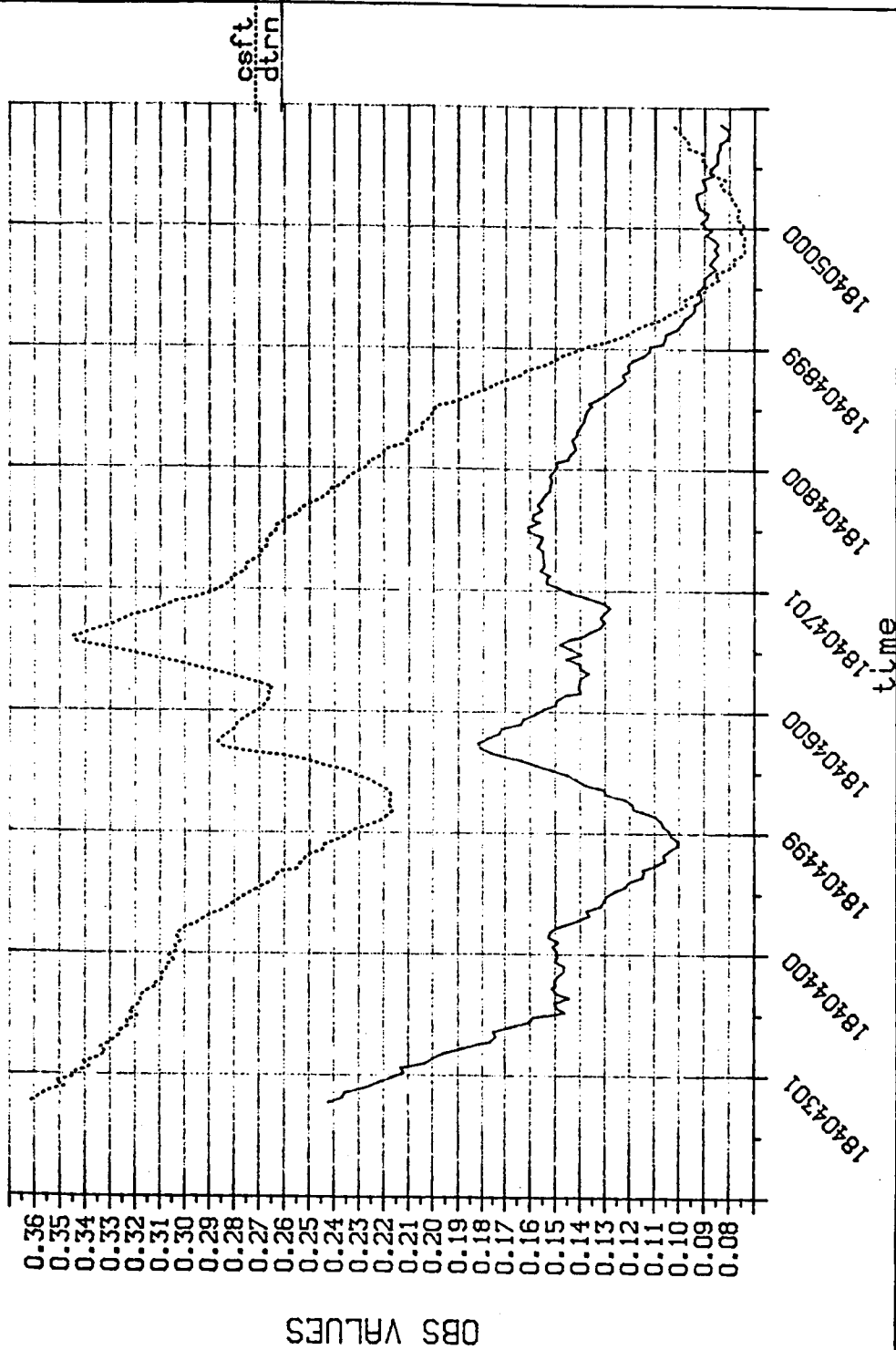


Figure 5.2.2. Example of HP Plotter Graphic Display

SPECIAL PRODUCTS PROCESS

TABLE OF CONTENTS

	Page
1.0 INTRODUCTION.....	AIII-52
2.0 OUTPUT PRODUCTS.....	AIII-52
2.1 BINARY OUTPUT.....	AIII-52
2.2 FORMATTED OUTPUT.....	AIII-55
2.3 PARAMETER GROUPS.....	AIII-55
3.0 INPUT.....	AIII-63
3.1 INPUT FILES.....	AIII-63
3.2 TEXT INPUT.....	AIII-64
3.3 BASE FILE CONTROL OF FREQUENCY AND TIME SPAN.....	AIII-64
4.0 SAMPLE OPERATIONAL DESCRIPTION.....	AIII-64
5.0 ERROR MESSAGES.....	AIII-66

1.0 INTRODUCTION

The Special Products Process utilizes trajectory and attitude information from RELBET standard format files to produce RELBET ancillary data product files containing various parameters over specified time interval. Output also includes text describing the input and processing status. Options include both binary, formatted data product files, and data dropout information. The program **prodx** generates these products.

2.0 OUTPUT PRODUCTS

The Special Products Processor generates both binary and formatted output.

2.1 BINARY OUTPUT

The user has the option to generate a binary data products corresponding to the RELBET Ancillary Data Products described in Reference 1 of Chapter 2.0. This file is tape or mass storage depending on the user's input. This multi-record file has the following general format:

First Record:	Identifier record
Second Record:	Dictionary record
Succeeding Records:	UNIVAC or HP binary data records
END OF FILE	

The first record has a fixed length of 26 (HP or UNIVAC) single precision words. The dictionary and data records all have the same length; however, this length varies depending on the particular parameters desired. Table 2.1.1 depicts the overall file format.

The file identifier record provides the user with information to identify the file contents. The format is shown on Table 2.1.2. The first twelve (12) words constitute a generic identifier message. The thirteenth word is the alphabetic "SPEC" which identifies the data as a special BET product. The

Table 2.1.1. File Format

<u>RECORD</u>	<u>INFORMATION</u>
1	Tape Identifier
2	Parameter Dictionary
3	Data
.	.
.	.
.	Monotonically
.	Increasing
.	Time
.	.
.	.
L	Data

Table 2.1.2. File Identifier Record Format

<u>SINGLE WORD NUMBER</u>	<u>DEFINITION</u>	<u>TYPE</u>	
1	48 character identifier message flight no., data etc.	alphabetic (single)	
12			
13	SPEC	alphabetic (single)	
14	tape start GMT (first data record)	YR	
15		MO	
16		DAY	
17		HR	integer (single)
18		MIN	
19		SEC	
20	tape stop GMT (last data record)	YR	
25		SEC	integer (single)
26	number of parameters in dictionary		integer (single)

next twelve integer words specify the time period in Greenwich Mean Time (GMT) covered by the data records. The last word is an integer defining the number of parameters in the dictionary.

The dictionary record format shown on Table 2.1.3 informs the user of the available parameters and their relative location on the data records. The first word is and their relative location on the data records. The first word is an integer defining the number of parameters in the dictionary. The remaining N alphabetic words identify the relative location of the value of the parameter identified by the corresponding symbol.

The data record format shown on Table 2.1.4 provides the double precision parameters data in a format where the value for a parameter is in the corresponding word location of that parameter symbol in the dictionary record. Note that the first word constitutes a last record flag.

2.2 FORMATTED OUTPUT

A formatted Data Products file will be generated at user option. This file contains the same parameters at the same frequency as the binary data products file. It consists of a dictionary description following by displays of each output record. Figure 2.2.1 and Figure 2.2.2 provide samples of these formats. Note that the dictionary is designated as data record 0.

2.3 PARAMETER GROUPS

The ancillary parameters are divided into 17 groups with the option to include or omit each group (spg/xxsprm). These parameter groups are described in Table 2.3.1 Note that the continuation flag (entry 1, dictionary name CONTINUE) is always included. Although any subset of these 17 groups may be

Table 2.1.3. Dictionary Record Format

<u>DOUBLE WORD NUMBER</u>	<u>VARIABLE DEFINITION</u>	<u>TYPE</u>
1	N = No. of parameters in dictionary	integer (double)
2	Symbol for parameter in location 2 on data record (1st variable)	alphabetic (double)
3	Symbol for parameter in location 3 on data record (2nd variable)	alphabetic (double)
.		
.		
.		
N + 1	Symbol for parameter in location N + 1 on data record (Nth variable)	alphabetic (double)

Note: Value for

Variable no. 1 is the double precision word 2 on data record.

Variable no. 2 is the double precision word 3 on data record.

.

.

Variable no. N is the double precision word N + 1 on data record.

Table 2.1.4. Data Record Format

<u>WORD NUMBER</u>	<u>VARIABLE DEFINITION</u>	<u>TYPE</u>
1	ITYPE 0 = continuing records 1 = last record in file	Double Precision
2	Value of parameter in location 2 on dictionary record (1st variable)	Double Precision
3	Value of parameter in location 3 on dictionary record	Double Precision
.	.	.
.	.	.
.	.	.
.	.	.
N + 1	Value of parameter in location N + 1 on dictionary record	Double Precision

Note: N is number of parameters read from previous dictionary record.

REL BET DATA RECORD O 14513 SPECIAL PRODUCTS
 FLT 13 SEQ 01-00 REV 00

1 *****T	2 SGMTY	3 SGMTMO	4 SGMTD	5 SGMTH
6 SGMTM	7 SGMTS	8 GMTY	9 GMTMO	10 GMTD
11 GMTH	12 GMTM	13 GMTS	14 GETH	15 GETM
16 GETS	17 XM	18 YM	19 ZM	20 XDM
21 YDM	22 ZDM	23 SLVX	24 SLVY	25 SLVZ
26 SLVXD	27 SLVYD	28 SLVZD	29 SU	30 SV
31 SW	32 SUD	33 SVD	34 SWD	35 ALPHU
36 BETAU	37 PHIU	38 A11	39 A21	40 A31
41 A12	42 A22	43 A32	44 A13	45 A23
46 A33	47 Q1	48 Q2	49 Q3	50 Q4
51 RDOTB	52 PDOTB	53 YDOTB	54 TDOTB	55 AZ
56 EL	57 RAZ	58 REL	59 RANGE	60 RRATE
61 PSIM	62 PMX	63 PMY	64 PMZ	65 PMXD
66 PMYD	67 PMZD	68 PRMX	69 PRMY	70 PRMZ
71 PRMXD	72 PRMYD	73 PRMZD	74 PU	75 PV
76 PW	77 PUD	78 PVD	79 PWD	80 PLVX
81 PLVY	82 PLVZ	83 PLVXD	84 PLVYD	85 PLVZD

Figure 2.2.1. Example of Dictionary Record

REL BET DATA RECORD 4635
 FLT 13 SEQ 01-00 REV 00 14S13 SPECIAL PRODUCTS

.1000000000+001 CONTINUE	.1984000000+004 SGMTY	.4000000000+001 SGMTMO	.1000000000+002 SGMTD	.1300000000+002 SGMTH
.5300000000+002 SGMTM	.1644570017+002 SGMTS	.1984000000+004 GMTY	.4000000000+001 GMTMO	.1000000000+002 GMTD
.1300000000+002 GMTH	.5300000000+002 GMTM	.1644570017+002 GMTS	.9500000000+002 GETH	.5400000000+002 GETM
.4644570017+002 GETS	-.2839205994+004 XM	.5339349248+004 YM	-.3269236614+004 ZM	-.6709623168+001 XDM
-.3601182056+001 YDM	-.5322726811-001 ZDM	-.5302148848-002 SLVX	.7552234594-002 SLVY	.4148547608-001 SLVZ
.2675223253-005 SLVXD	.2515077162-004 SLVVD	.9949590197-004 SLVZD	-.4148547608-001 SU	-.5302148848-002 SV
-.7552234594-002 SW	-.9362244604-004 SUD	-.4328031707-004 SVD	-.2515077162-004 SWD	-.2691463728+001 ALPHU
-.5021599890+002 BETAU	.9186476376+002 PHIU	.4184141995+000 A11	.2394892024+000 A21	-.8761132801+000 A31
.8508096467+000 A12	-.4409606539+000 A22	.2857912642+000 A32	-.3178875630+000 A13	-.8649847534+000 A23
-.3882639227+000 A33	.3837934415+000 Q1	.7496063594+000 Q2	.3636237992+000 Q3	-.3982092829+000 Q4
-.2723688132-001 RDOTB	.3113666913-001 PDOTB	.4332770718-001 YDOTB	.5990517572-001 TDOTB	.3384954672+003 AZ
-.5571232346+002 EL	-.2216613264-001 RAZ	.1627672441-001 REL	.4249933830-001 RANGE	.1012579235-003 RRATE
.0000000000 PSIM	-.2839229539+004 PMX	.5339382105+004 PMY	-.3269249737+004 PMZ	-.6709705763+001 PMXD
-.3601119338+001 PMYD	-.5324997160-001 PMZD	-.2354522753-001 PRMX	.3285734771-001 PRMY	-.1312290812-001 PRMZ
-.8259503568-004 PRMXD	.6271778958-004 PRMYD	-.2270349122-004 PRMZD	.4148546370-001 PU	.5302155901-002 PV
.7552297682-002 PW	.9362238503-004 PUD	.4328030621-004 PVD	.2515101742-004 PWD	.5302155901-002 PLVX
-.7552297682-002 PLVY	-.4148546370-001 PLVZ	-.2675236533-005 PLVXD	-.2515101742-004 PLVYD	-.9949585083-004 PLVZD

Figure 2.2.2. RELBET Data Record

Table 2.3.1. Special Products Parameters

<u>VARIABLE SET</u>	<u>PARAMETER GROUP NAME</u>	<u>SYMBOL</u>	<u>PARAMETER</u>	<u>UNITS</u>
Time, Greenwich mean (GMT) Orbiter onboard	1	SGMTY		yr
		SGMTMO		month
		SGMTD		day
		SGMTH		hr
		SGMTM		min
		SGMTS		sec
Time, Greenwich mean (GMT) Ground - MCC UTC	2	GMTY		yr
		GMTMO		month
		GMTD		day
		GMTH		hr
		GMTM		min
		GMTS		sec
Time, ground elapsed (GET)	3	GETH		hr
		GETM		min
		GETS		sec
State Vector, Orbiter, in Aries mean of 1950 Cartesian coordinate System	4	XM	x	km
		YM	y	km
		ZM	z	km
		XDM	\dot{x}	km/sec
		YDM	\dot{y}	km/sec
		ZDM	\dot{z}	km/sec
State Vector, Orbiter, in free flyer LVLH Cartesian coordinates	5	SLVX	x	km
		SLVY	y	km
		SLVZ	z	km
		SLVXD	\dot{x}	km/sec
		SLVYD	\dot{y}	km/sec
		SLVZD	\dot{z}	km/sec
State Vector, Orbiter, in free flyer UVW Cartesian coordinates	6	SU	u	km
		SV	v	km
		SW	w	km
		SUD	\dot{u}	km/sec
		SVD	\dot{v}	km/sec
		SWD	\dot{w}	km/sec
Attitude, Orbiter, Euler angles, body axis with respect to Orbiter UVW coordinates	7	ALPHU	yaw	deg
		BETAU	pitch	deg
		PHIU	roll	deg

Table 2.3.1. Special Products Parameters (Continued)

<u>VARIABLE SET</u>	<u>PARAMETER GROUP NAME</u>	<u>SYMBOL</u>	<u>PARAMETER</u>	<u>UNITS</u>
Attitude, Orbiter, Transformation, Aries mean of 1950 to body matrix	8	A11	matrix elements row ordered	n.d.
		A12		n.d.
		A13		n.d.
		A21		n.d.
		A22		n.d.
		A23		n.d.
		A31		n.d.
		A32		n.d.
Attitude, Orbiter, quaternion, Aries mean of 1950 to body axes	9	Q1	scalar part	n.d.
		Q2	vector	n.d.
		Q3		n.d.
		Q4		n.d.
Attitude, rate, Orbiter, angular velocities and total rate, body about Aries mean of 1950 Cartesian coordinates	10	YDOTB	rate about z	deg/sec
		PDOTB	rate about y	deg/sec
		RDOTB	rate about x	deg/sec
		TDOTB	total rate	deg/sec
Look angle, and angle rates orbiter body to free flyer	11	AZ	yaw	deg
		EL	pitch	deg
		RAZ	yaw rate	deg/sec
		REL	pitch rate	deg/sec
Range, and range rate, between vehicle reference points	12	RANGE	range	km
		RRATE	range rate	km/sec
Simulation Flag	13	PSIM		n.d.
State Vector, free flyer, in Aries mean of 1950 Cartesian coordinates	14	PMX	x	km
		PMY	y	km
		PMZ	z	km
		PMXD	\dot{x}	km/sec
		PMYD	\dot{y}	km/sec
		PMZD	\dot{z}	km/sec
State Vector, free flyer, in Aries mean of 1950 Cartesian coordinates relative to Orbiter	15	PRMX	x	km
		PRMY	y	km
		PRMZ	z	km
		PRMXD	\dot{x}	km/sec
		PRMYD	\dot{y}	km/sec
		PRMZD	\dot{z}	km/sec

Table 2.3.1. Special Products Parameters (Continued)

<u>VARIABLE SET</u>	<u>PARAMETER GROUP NAME</u>	<u>SYMBOL</u>	<u>PARAMETER</u>	<u>UNITS</u>
State Vector, free flyer, in Orbiter UVW Cartesian coordinates	16	PU	u	km
		PV	v	km
		PW	w	km
		PUD	\dot{u}	km/sec
		PVD	\dot{v}	km/sec
		PWD	\dot{w}	km/sec
State Vector, free flyer, in Orbiter LVLH Cartesian coordinates	17	PLVX	x	km
		PLVY	y	km
		PLVZ	z	km
		PLVXD	\dot{x}	km/sec
		PLVYD	\dot{y}	km/sec
		PLVZD	\dot{z}	km/sec

selected, a fatal error will result if none are chosen or sufficient input information to compute parameters is not provided.

Parameter group number 13 is used for data dropout information. If the value is 1, the output reflects processing of tracking data. If -1, the output does not reflect tracking data. The default for data dropout is 1.

3.0 INPUT

The program **prodx** requires both binary input and text input.

3.1 INPUT FILES

The program **prodx** requires three things from input files: the orbiter ephemeris, the target ephemeris, and the orbiter attitude. The parameters `fname` and `posx` in the input block `xxnflz` specifies the files from which to obtain them. The files must be standard RELBET format files with strictly increasing time-tag frames of only one type. There are two possibilities source of the ephemeris information:

- o A separate ephemeris file for each vehicle.
- o A single relative trajectory file for both vehicles.

The attitude source should be a standard attitude file containing Body to M50 quaternions.

In you do not wish to generate all parameter groups, then you may not need all three type of file input. The required input for the parameter groups is summarized below.

<u>Input</u>	<u>Parameter Groups</u>
Orbiter State	4, 5, 12, 15, 16, 17
Target State	11, 12, 14, 15, 16, 17
Attitude	7, 8, 9, 10, 11

3.2 TEXT INPUT

Text input is with the **linput** input language. It consists of the following input blocks.

- xxgnrl (optional)
- xxtime (mandatory)
- xxmisc (optional)
- xxcon (optinal)
- xxnflz (mandatory)
- xxtoff (mandatory)
- xxsprm (mandatory)

3.3 BASE FILE CONTROL OF FREQUENCY AND TIME SPAN

Output parameters are computed at the same frequency as the records in the base file (bfopt/xxnflz). The base file may be any of the input files used for computing the output parameters. (Processing will be terminated if the second ephemeris is designated as the base file but it is not needed to compute any of the output parameters.) A time step of less than .5 seconds is considered a fatal error. The products span an interval extending from a specified begin and end date. Note that this interval is approximate if the frequency is determined by an input file. This span must lie within the spans of all the input files or processing will be terminated.

4.0 SAMPLE OPERATIONAL DESCRIPTION

Figure 4.1 is an example of **linput** inputs wherein two ephemeris files and an attitude file. The relative trajectory file is used as the base file and both the binary and text output are generated.

```

// xxgnrl - use default
// xitime inputs
date = 1985,1,0,0,0;
dates = 0.e0;
tbegin = 9277996.0;
tend = 9279201.0;
delta = 1.e0;
endopt = 0;
endval = 9279201.0;
// xxmisc - use default
// xxcon - use default
// xxnflz inputs
fname = "/users/Relbet/Test/d4eph",
"/users/Relbet/Test/d4teph",
"/users/Relbet/Test/d4att",
"/users/Relbet/Test/SP51D",
"/users/Relbet/Test/SP51D_PRT";
bfopt = 1;
posx = 1,2;
lordx = 8,8;
print = 1;
//trjout
// xtoff inputs
ldate = 1985,1,0,0,0,9277200; //launch date
dtdate = 1985,1,0,0,0,0; //time bias date
//dtbias
qafreq = 5; //rec and print frequency
tlapse = 10.e0; //summary frequency(min)
drive = 0; //tape drive
tape = 1; //univac tape > 0
hpbin = 1; //hp binary file > 0
dtable = 9277220..9277245.,4810., 9277230.,9277260.,4810.;
// data dropout intervals
// xxsprm inputs
gmsg = "51D4SPECIAL PRODUCT"; //generic message in dpf header
seq = "51D1000"; //file sequence id
spg = 1711,310; //display group flags
//dict record format read dict(3) .0 for int .6 for char .8 for real
//datbuf real datbuf(2) data buf for creating Univac tape

```

Figure 4.1. Example of Linput Inputs

Output material consists of the data products files discussed in Sections 2 and job summary prints routed to the user specified output device using the redirected method. Figure 4.2 is an example of text output. For a successful run, these messages are displayed.

- o Name, version, and current time
- o Summary of desired output products (input summary)
- o Summary of file usage (file summary)
- o Dictionary header
- o Last data record in the same format as Formatted Data Product
- o Number of records processed
- o Termination message and error/warning counts

The following debug print is directed to the unit specified by bugs, refer to the RELBET PROGRAMMER'S MANUAL Mtfcommon for further information.

<u>pbug</u>	<u>Value</u>	<u>Print</u>
6	< 0 <-2	Interpolation information for states State interpolation buffers

Invoke the program with the command

```
prodx < input >output
```

or

```
cat input | prodx > output
```

where **input** is a textfile containing the input blocks, and **output** is the file or device desired for job summary print.

5.0 ERROR MESSAGES

In addition to the standard messages involving file manipulation and tape device errors, the following warnings or error messages are issued:


```

*****
** TRU DSG RELBET SYSTEM
** Relbet Products Processor
** baseline: 3 version: 0
** date: 10- 8-86 time: 13:52:06
**
*****

```

Unit Assignments

```

input: 69 output: 6
terminal: 6 debug: 6

```

Print Flags

```

debug: 0 0 0 0 0 0 0 0 0 0 0 0
output: 0 0 0 0 0 0 0 0 0 0 0 0

```

Time Information

```

base time: 1985 yr 1 mo 0 day 0 hr 0 min
           .000000000000000000D+00 sec

```

```

begin time
.927799600000000000D+07
107 days 9 hrs 13 min 16.00000000 sec 107 days 9 hrs 33 min 21.00000000 sec
end time

```

Job Description

```

no job description given
no job description given

```

Display of Mathematical and Physical Constants

```

pi: .314159265358979270D+01 halfpi: .157079632679489630D+0

```

Figure 4.2. Example of Special Product Text Output

```

twopi: .628318530717958530D+01  light speed(m/s): .29979250000000000D+09
1/users/Relbet/Test/d4eph
2/users/Relbet/Test/d4teph
3/users/Relbet/Test/d4att
4/users/Relbet/Test/SP51D
5/users/Relbet/Test/SP51D_PRT

```

```

Base File: 1
Interpolation Options: 8 8
State Source: 1 2
Print Flag: 1 Plot Flag: 0
Ephemeris 1: 0 Ephemeris 2: 0 Rel Traj: 0

```

```

Some inputs for xxtoff
-----

```

```

Tape drive number: 0
Univac compatible tape flag: 1
Hpbm file flag: 1

```

```

Input Summary
-----

```

```

sequence id (seq): 51D1000 message (gmsg): 51D4SPECIAL PRODUCT
10.00 min for status summary (tlapse)

```

```

**parameter group flags**
1 1 1 1 1 1
1 1 1 1 1 1
1 1 1 1 1 1
1 1 0 0 0 0

```

```

Output Files
-----

```

```

data products 0
name: </users/Relbet/Test/SP51D

```

Figure 4.2. Example of Special Product Text Output (Continued)

micro print 1
 name: /users/Relbet/Test/8P5ID_PRT

Data Sources

1st state used: 1 location (posx(1)): 1
 2nd state used: 1 location (posx(2)): 2
 attitude used: 1

Reference Times

472.3842129630 base julian date rel 1/1/1984
 liftoff time(date): 1985, 0, 0, 9277200,
 796.000000 sec from base date
 gnt offset time(date): 1985, 1, 0, 0,
 offset at reference time(date): .000000000000000000 sec
 offset rate(date): .000000000000000000 sec/sec

RELBT DATA RECORD 0
 FLT 51 SEQ 01-00 REV 0 SID4SPECIAL PRODUCT

1	CONTINUE	2	SCRTY	3	SCRTHO	4	SCMTD	5	SCMTH
6	SCMTH	7	SCRTS	8	GMTY	9	GMTD	10	GMTD
11	GMTH	12	GMTS	13	GMTS	14	GETH	15	GETH
16	GETS	17	XM	18	YM	19	ZM	20	XDM
21	YDM	22	ZDM	23	SLVX	24	SLVY	25	SLVZ
26	SLVXD	27	SLVYD	28	SLVZD	29	SU	30	SV
31	SU	32	SUD	33	SVD	34	SUD	35	ALPHU
36	BETAU	37	PHIU	38	A11	39	A21	40	A31

Figure 4.2. Example of Special Product Text Output (Continued)

41	A12	42	A22	43	A32	44	A13	45	A23
46	A33	47	Q1	48	Q2	49	Q3	50	Q4
51	RDOTB	52	PDOTB	53	YDOTB	54	TDOTB	55	AZ
56	EL	57	RAZ	58	REL	59	RANGE	60	RRATE
61	PSIM	62	PMX	63	PMY	64	PMZ	65	PMXD
66	PHYD	67	PHZD	68	PRMX	69	PRMY	70	PRMZ
71	PRXD	72	PRYD	73	PRZD	74	PV	75	PV
76	PV	77	PUD	78	PVD	79	PWD	80	PLVX
81	PLVY	82	PLVZ	83	PLVXD	84	PLVYD	85	PLVZD

```

*****
input file 1
*****
input file 2
*****
altitude file
initialized at base record 220 time= .9277996756D+07 with tbegin=
NTOPEX: channel number 7
>> time = 154633. min 1 data records written
>> time = 154643. min 157 data records written
*****
1
    .927799600D+07
    
```

```

RELSET DATA RECORD 312
FLT 51 SEQ D1-00 REV 0 51D4SPECIAL PRODUCT

.1000000000E+01 CONTINUE .1985000000E+04 SCMTY .4000000000E+01 SCMTHO .1700000000E+02 SCMTD .9000000000E+01 SCMTH
.3300000000E+02 SCMTM .1867570003E+02 SCMTS .1985000000E+04 CMTY .4000000000E+01 CMTMO .1700000000E+02 GMTD
.9000000000E+01 CMTM .3300000000E+02 CMTX .1867570003E+02 CMTS .0000000000E+00 CETH .3300000000E+02 CETM
.1407999992E+02 CETS .6300131050E+04 XH .2316901831E+04 YH .3111982973E+03 ZH -.2124709204E+01 XDM
    
```

Figure 4.2. Example of Special Products Text Output (Continued)

```

.6465122037E+01 YDM      -.3661948276E+01 ZDM      -.7341773552E+02 SLVX      .1213451623E+01 SLVZ
-.2371839919E-02 SLVXD   .2523511164E-03 SLVYD   .2849147897E-02 SLVZD   -.7341773552E+02 SV
.1685014522E+00 SW      .8159768721E-01 SUD      -.3767580994E-02 SVD     .1793192134E+03 ALPHU
-.1579857581E+01 BETAU  -.9139101664E+02 PH1U    -.9460519140E+00 A11    .2761701555E+00 A31
-.3528302637E+00 A12    -.4201449577E+00 A22    -.8480912896E+00 A32    -.8914944719E+00 A23
.4521849287E+00 A33     .1466186013E+00 Q1      .7400695061E-01 Q2     .8394018407E+00 Q4
-.4536304746E-02 RD0TB -.7523836647E-01 PD0TB -.2180912838E-01 YD0TB .7689775102E+02 AZ
.8842861361E+02 EL      -.4328581074E+01 RAZ     .1078356662E-02 REL     .2418014973E-02 RRATE
.1000000000E+01 PSIM    .6279760567E+04 PHX      .2378180696E+04 PHY     -.2202109924E+01 PMXD
.6439483081E+01 PHYD    -.3666868975E+01 PHZD   -.2037048291E+02 PRMX   .6127886447E+02 PRMZ
-.7740072055E-01 PRMXD -.2563895595E-01 PRMYD  -.4920699700E-02 PRMZD .4112506407E+00 PU      .7342661663E+02 PV
-.1660979126E+00 PU     -.8163397335E-01 PUD     .2875853494E-02 PVD     .7342661663E+02 PLVX
.1660979126E+00 PLVY    -.4112506407E+00 PLVZ    .2402923208E-02 PLVXD  -.2805216188E-02 PLVZD

```

```
>>output completed 312 data records
```

```

*****
** TRU DSG RELBET SYSTEM
** Relbet Products Processor
** baseline: 3 version: 0
** date: 10- 8-86 time: 13:55:17
*****
execution terminated 0 warnings 0 errors
*****

```

Figure 4.2. Example of Special Products Text Output (Continued)

TOO FEW PARAMETERS IN OUTPUT

No output parameter group were specified.

BAD FILE ID FOR VEHICLE STATE

A file ID different from ±1, ±2 was specified in posx for a required state.

NOTHING IS WRITTEN ON TAPE

Files or tape error

TIME STEP < $\frac{1}{2}$ SEC

A time step of less than 0.5 seconds was specified.

TERMINATION FOR INPUT ERROR

Input file error

DIVIDED DIFFERENCE NOISE ANALYSIS

TABLE OF CONTENTS

	Page
1.0 INTRODUCTION.....	AIII-75
2.0 DERIVATION OF EQUATIONS.....	AIII-75

1.0 INTRODUCTION

Divided difference noise analysis processes observations to produce time-ordered estimates of the noise associated with the given observation type.

The algorithm used to product these estimates is based on computing divided differences of a sequence of observations. The method is generally superior to Fourier transform methods of noise analysis for the following reasons:

1. In Fourier methods, a large signal can begin to swamp the noise contributions to the higher harmonics, resulting in an over-estimation of the noise (unless special care is taken to remove as much of the signal as possible before Fourier analysis).
2. For divided difference analysis, the data not required to be evenly space.
3. Divided difference analysis generally requires fewer points than Fourier analysis for a meaningful result.

Warning: This technique assumes noise is uncorrelated.

2.0 DERIVATION OF EQUATIONS

Given a set of time-ordered (but not necessarily evenly spaced) data points $y(1)$, $y(2)$, ..., $y(n)$, with corresponding time tags $t(1)$, $t(2)$, ..., $t(n)$, define the first divided difference as

$$\begin{aligned}\Delta_y(i,1) &= [y(i+1) - y(i)]/[t(i+1) - t(i)] & (1) \\ &= - [t(i+1) - t(i)]^{-1} y(i) \\ &\quad + [t(i+1) - t(i)]^{-1} y(i+1) \\ &= W(1,i,1) y(i) + W(2,i,1) y(i+1)\end{aligned}$$

where

$$W(1,i,1) = - [t(i+1) - t(i)]^{-1} \quad (2a)$$

$$W(2,i,1) = - W(1,i,1) \quad (2b)$$

The second divided difference is

$$\begin{aligned} \Delta_y(i,2) &= [\Delta_y(1,i+1) - \Delta_y(1,i)]/[t(i+2) - t(i)] \quad (3) \\ &= [t(i+2) - t(i)]^{-1} [W(1,i+1,1) y(i+1) \\ &\quad + W(2,i+1,1) y(i+2)] \\ &\quad - [t(i+2) - t(i)]^{-1} [W(1,i,1) y(i) \\ &\quad + W(2,i,1) y(i+1)] \\ &= W(1,i,2) y(i) + W(2,i,2) y(i+1) \\ &\quad + W(3,i,2) y(i+2) \end{aligned}$$

where

$$W(1,i,2) = - [t(i+2) - t(i)]^{-1} W(1,i,1) \quad (4a)$$

$$W(2,i,2) = [t(i+2) - t(i)]^{-1} [W(1,i+1,1) - W(2,i,1)] \quad (4b)$$

$$W(3,i,2) = [t(i+2) - t(i)]^{-1} W(2,i+1,1) \quad (4c)$$

In general the p^{th} divided difference can be represented as

$$\Delta_y(i,p) = \sum_{K=1}^{p+1} W(K,i,p) y(i+K-1) \quad (5)$$

where the W's are defined recursively as

$$W(1,i,j) = - [t(i+j) - t(i)]^{-1} W(1,i,j-1) \quad (6a)$$

for $i = 1, 2, \dots, n-j$; $j = 2, 3, \dots, p$

$$W(K,i,j) = [t(i+j) - t(i)]^{-1} [W(K-1,i+1,j-1) - W(K,i,j-1)] \quad (6b)$$

for $i = 1, 2, \dots, n-K$; $j = 2, 3, \dots, p$; $K = 2, 3, \dots, j+1$

$$W(j+1,i,j) = [t(i+j) - t(i)]^{-1} W(j,i+1,j-1) \quad (6c)$$

for $i = 1, 2, \dots, n-p$; $j = 2, 3, \dots, p$

with the starting values

$$W(1,i,1) = - [t(i+1) - t(i)]^{-1} \quad (6d)$$

for $i = 1, 2, \dots, n-1$

$$W(2,i,1) = - W(1,i,1) \quad (6e)$$

for $i = 1, 2, \dots, n-1$

Consider now a sequence of discrete observations contaminated with noise, of the form

$$y^*(i) = y(i) + \epsilon(i) \quad (7)$$

where $\epsilon(i)$ is a sample of a member of an ergodic stochastic process, $\epsilon(t)$. $y(i)$ will be assumed to be a sample of an analytic function $y(t)$. The r^{th} divided difference forms an approximation to the r^{th} derivative, in the sense that

$$\Delta_y(i, r) = \frac{1}{r!} \left. \frac{d^r y}{dt^r} \right|_{t=\tau_r} \quad (8)$$

for some τ_r , $t_i < \tau_r < t_{i+r}$. (This follows from the mean value theorem.)

Taking the divided difference of the noisy data

$$\Delta_{y^*}(i, r) = \frac{1}{r!} \left. \frac{d^r y}{dt^r} \right|_{t=\tau_r} + \Delta_\epsilon(i, r) \quad (9)$$

For some $r > r_c$, the

$$\frac{1}{r!} \left. \frac{d^r y}{dt^r} \right|_{t=\tau_r}$$

term will become negligible compared to $\Delta_\epsilon(i, r)$. Then the r^{th} divided difference of the data becomes an approximation to the r^{th} divided difference of the noise,

$$\Delta_{y^*}(i, r) - \Delta_\epsilon(i, r), \quad r > r_c \quad (10)$$

Taking the expected values of the square of both sides

$$E\{(\Delta_{y^*}(i, r))^2\} - E\{(\Delta_\epsilon(i, r))^2\} \quad (11)$$

$$= E\left\{\left[\sum_{K=1}^{r+1} W(K, i, r) \epsilon_{(i+K-1)}\right]^2\right\}$$

from equation 5.

If now the additional assumption is made that the noise is uncorrelated, so that $E\{\epsilon_m \epsilon_n\} = \sigma^2 \delta_{mn}$, then

$$E\{(\Delta_{y^*}(i, r))^2\} = \sigma^2 \sum_{K=1}^{r+1} [W(K, i, r)]^2 \quad (12)$$

or

$$\sigma^2 = \frac{E\{\Delta_{y^*}(i,r)\}^2}{\sum_{K=1}^{r+1} [W(K,i,r)]^2} \quad (13)$$

Expected value can here be interpreted as an average (for given data), or

$$\sigma^2 = \frac{1}{n-r} \sum_{i=1}^{n-r} [\Delta_{y^*}(i,r)]^2 / \sum_{K=1}^{r+1} [W(K,i,r)]^2 \quad (14)$$

where $r > r_c$, n is the number of data points being considered ($n \gg r$), and the W 's are given by equations 6a-6e.

Reference: Error Analysis and Methods for Estimating Errors in Position, Velocity, and Acceleration Data, Document No. 119-71. Secretariat Range Commander's Council, White Sands Missile Range, New Mexico; dated May 1971.