Contract NAS8-21120

MCR-67-441

INTERPLANETARY TRAJECTORY ERROR ANALYSIS:

COMPUTER PROGRAM DESIGN VOLUME II - FINAL REPORT

APRIL 1968

(NASA-CR-61794) INTERPLANETARY TRAJECTORY ERROR ANALYSIS: COMPUTER PROGRAM DESIGN, VOLUME 2 Final Report (Martin Marietta Corp.) 192 p

N73-72582

Unclas
00/99 69451

Prepared for

National Aeronautics and Space Administration
George C. Marshall Space Flight Center
Huntsville, Alabama 35812

Prepared by

Gentry Lee
Ralph Falce
Fred Hopper

MARTIN MARIETTA CORPORATION
DENVER DIVISION
P.O. Box 179
Denver, Colorado 80201

MCR 67-441

INTERPLANETARY TRAJECTORY ERROR ANALYSIS:

COMPUTER PROGRAM DESIGN

by

Gentry Lee
Ralph Falce
Fred Hopper

Martin Marietta Corporation
Denver Division
P. O. Box 179
Denver, Colorado 80201

CONTRACT NO. NAS8-21120

Volume II - Final Report

April 1968

Prepared
for

National Aeronautics and Space Administration

George C. Marshall Space Flight Center

Huntsville, Alabama 35812

FOREWORD

This document represents the second and final volume of the final report on Contract NAS8-21120 entitled "Mars Transfer Trajectory Determination and Error Analysis." It is a companion to Volume I, which was published in December 1967.

TABLE OF CONTENTS

Chapter	Title	Page
	FOREWORD	i
	TABLE OF CONTENTS	ii
	ABSTRACT	vi
I.	INTRODUCTION	1
II.	ULTIMATE PROGRAM DESIGN	9
A.	The Executive Group	12
B.	The Supporting Group	15
C.	The Dynamics Group	17
1.	Dynamics Module, DYNM	18
2.	Observation Module, OBSM	19
3.	Navigation Module, NAVM	20
4.	Guidance Module, GUIDM	20
5.	Control Module, CONTM	21
D.	Auxiliary Group	22
1.	Policy Module, POLM	22
2.	Information Module, INFM	23
3.	Auxiliary Module	24
III.	TUTORIAL PROGRAM	25
A.	Executive Group	26
1.	EXM1	26
2.	EXM2	26
3.	EXM3	28
B.	Supporting Group	28
1.	NTM Model 1	30
2.	PSIM Model 1	32
3.	HM Model 1	35
C.	Dynamics Group	38
1.	Series 1 Models	38
a.	DYNM Model 1	39
b.	OBSM Model 1	41
c.	NAVM Model 1	42

TABLE OF CONTENTS (Continued)

Chapter		Title	Page
	d.	GUIDM Model 1	44
	e.	CONTM Model 1	46
D.		Auxiliary Group	49
	1.	POLM Model 1	49
	2.	INFM Model 1	52
E.		Subroutines and Functions	53
	1.	DATIN	53
	2.	MATIN	53
	3.	RNUM	53
		ADDENDUM TO PART III	54
	1.	Derivation of Equations for NTM, Model 1	54
	2.	Derivation of Equations for PSIM, Model 1	55
	3.	Derivation of Equations for HM, Model 1	60
	4.	Derivation of Guidance Matrix for POLM, Model 1	63
	5.	Instructions for Use of Tutorial Program	64
	a.	Run Designation	64
	b.	Event Designation	64
	c.	Time Registers	64
	d.	Print Control	65
	e.	Initial Conditions and Other Input	65
	f.	Input Deck Assembly	65
	g.	Printout of Input Data	67
IV.		NUMERICAL RESULTS	68
A.		Problem Description	68
B.		Numerical Results	70
APPENDIX 1		Flow Charts for the Ultimate Program	A-1
		EXM1	A-2
		EXM2	A-3
		EXM3	A-4
		NTM	A-6

TABLE OF CONTENTS (Continued)

Chapter	Title	Page
	PSIM	A-7
	HM	A-8
	DYNM	A-9
	OBSM	A-10
	NAVM	A-11
	GUIDM	A-12
	CONTM	A-13
	POLM	A-14
	INFM	A-15
APPENDIX 2	Flow Charts and Listings for the Tutorial Program	A-17
	EXM1 INITIALIZATION	A-18
	EXM1	A-19
	EXM2 INITIALIZATION	A-20
	EXM2	A-21
	EXM3 INITIALIZATION	A-23
	EXM3	A-24
	NTM	A-27
	PSIM	A-28
	TM (Tutorial version of HM)	A-32
	DYNM	A-33
	OBSM	A-34
	NAVM	A-35
	GUIDM	A-38
	CONTM	A-39
	POLM	A-41
	INFM	A-42
	DATIN	A-45
	MATIN	A-46
	RNUM	A-51

TABLE OF CONTENTS (Continued)

Chapter	Title	Page
	Listings	A-53
	EXM1, EXM2, EXM3	A-53
	NTM	A-56
	PSIM	A-57
	TM	A-60
	DYNM	A-61
	OBSM	A-62
	NAVM	A-63
	GUIDM	A-66
	CONTM	A-67
	POLM	A-69
	INFM	A-71
	DATIN	A-74
	MATIN	A-76
	FUNCTION RNUM	A-78
	SUBROUTINE EXTRA	A-79

ABSTRACT

This report is the second of two dealing with the subject of interplanetary error analysis. Chapter I, the Introduction, presents the philosophy underlying the conceptual design of an "ultimate" digital computer program for performing accuracy studies on interplanetary missions. Great flexibility and ease of use has been sought, with emphasis being placed on the navigation, guidance, and control aspects of the problem. Chapter II describes the logical structure envisioned for this ultimate program, and in Chapter III a greatly simplified tutorial version is presented in detail. This latter program has been coded and checked out, and is ready for use.

I. INTRODUCTION

This report is concerned with the design and implementation of a digital computer program to handle various interplanetary trajectory analysis tasks ranging from pre-flight mission analysis to trajectory determination and estimation based upon real data taken from an actual spacecraft. The current document is a companion to Volume I of the final report under NASA contract NAS8-21120 [1], which contains an analytical summary and literature survey of the technical state-of-the-art with respect to the many problems involved in an interplanetary trajectory analysis.

The emphasis in this report is on the modular design of an "ultimate", comprehensive computer program for interplanetary missions. The program under consideration would have three basic operating modes. These modes could be roughly described as follows:

1) Mission Analysis and Error Propagation
2) Simulation
3) Trajectory Estimation for an Actual Mission.

When operating in the mission analysis mode, primarily as a pre-flight tool, the program would have the capability of assisting in parametric error studies of all kinds. Due to the large number of computational options that could be included within the design, the analyst would be able to perform these mission analyses with any desired technical sophistication. For example, if computation time were a constraint and only approximate accuracy were needed for a specific study, the simpler computational options could be exercised. On the other hand, if a highly

accurate error study were needed, the most detailed computational schemes could be chosen to perform the study.

One of the primary advantages of the computer program design advanced in this report is its ability to operate in a simulation mode. Careful attention has been paid to setting up, within the overall modular structure, a module that permits the analyst to assume he knows exactly what is occurring during the flight and to check his estimation routine's capability of reproducing this trajectory. This simulation capability has been omitted in many trajectory determination programs. By using the simulation capability, the analyst can also determine the validity or non-validity of statistical accuracy statements resulting from preflight error propagation studies. These accuracy statements have a tendency to be overly-optimistic and can influence actual trajectory estimation computations.

The major change from the simulation mode to the actual trajectory estimation mode involves a change in the input measurement data. In the simulation mode, it is assumed that the trajectory is known exactly and then, based upon that known trajectory, the correct measurement data are corrupted by random and bias errors. The estimation routine next attempts to reproduce the known trajectory. During an actual flight, however, the measurement data are recorded from Earth-based or on-board sensors and are processed immediately by the estimation module. For obvious reasons, the module that knows the trajectory exactly within the simulation scheme is by-passed altogether when an actual trajectory is being estimated.

The computer program design discussed in this report attempts to satisfy simultaneously the three operating mode requirements. Since many of the computational options that are used in a mission analysis task are also applicable to the simulation and actual trajectory estimation problems, much unnecessary and redundant programming effort can be omitted by constructing such a flexible program.

Before discussing in detail the essential philosophy behind the computer program design, it is worthwhile to motivate the discussion by recalling the problems that the program is constructed to solve. Although the modular design could be extended, by making some necessary modifications, to handle boost and re-entry phases of interplanetary flight, in its current state the designed program constrains its attention to the transfer phase of the interplanetary mission. This transfer phase begins at injection or boost cut-off and extends to some pre-determined target conditions in the neighborhood of a target gravitational body. Within the transfer phase are periods during which the spacecraft is tracked by Earth-based or on-board sensors and the position and velocity of the spacecraft is subsequently estimated, based upon the tracking information. In addition, during the transfer phase and as a result of the estimation process, impulsive midcourse guidance corrections may be imparted to the spacecraft so that the trajectory more nearly meets the prescribed target conditions.

At least five different "critical problem areas" can be singled out with respect to an analysis of the interplanetary transfer. The first of these concerns the computation of accurate nominal or reference

trajectories. Another consideration deals with the calculation of state transition matrices that relate linear perturbations about the nominal trajectory from one time to another. A third problem area is the midcourse guidance policy -- when should corrections be made and how should they be determined. Related to these midcourse corrections is the fourth problem. One needs to know how accurately the spacecraft can follow its commands and what is the affect on the mission of errors in midcourse guidance execution. The fifth major problem concerns the entire filtering or estimation apparatus that is designed to yield position and velocity estimates from the measurement data. All of these problems are discussed in the companion analytic report and are simply listed here to indicate the complexity of the interplanetary transfer analysis problem.

With these problem areas in mind, one can now define more accurately the role that the designed analysis program will play. As a particular example, suppose that the program is operating in its mission analysis mode. In that mode, the program would determine answers to such questions as the following:

(1). For a specified transfer trajectory to Mars, what is the functional dependence of the accuracy of the estimated position near the target upon error sources such as errors at injection, both random and bias errors in the tracking instruments, errors in midcourse correction execution, and errors in gravitational or astronomical constants?

(2). For a specific mission, what nominal trajectories are least sensitive to the aforementioned error sources?

(3). For a particular interplanetary transfer with prescribed position and velocity accuracy on the target conditions, what combinations of numerical error from each known error source are permissible?

All of these questions are concerned with preliminary mission specification. The parametric studies that answer the above questions would doubtless be important both in selection of the final mission design and in the numerous trade-off studies undergone in an engineering process.

Equally important questions can be answered by exercising the program in each of its other two modes. When operating in its simulation mode, the program's most important responsibility is determining whether or not, for various approximating assumptions, the estimation procedure will indeed converge to the true transfer trajectory. When an actual flight is underway, the program operates in its third mode, estimating the transfer trajectory from real measurement data and determining midcourse corrections based upon some specified and automated guidance policy.

Some definition of the special terms to be used in this report will probably facilitate its reading. With respect to the designed program, the expressions mode, module, and model will be used quite frequently. The term <u>mode</u> has been chosen to represent any of the three basic purposes for which the program has been constructed. The three operating modes for the program have already been defined. A <u>module</u>, on the other hand, is a structural block within the program that contains control and computational logic. These modules, which permit variations in operating modes and computational schemes with a minimum of change, form the

sub-structure of the program and are the source of program flexibility. The third term, model, is used to refer to specific computational schemes needed by the program. For example, one can speak of several different models for computing state transition matrices in the interplanetary problem. Models generally appear within modules and can be changed without affecting the basic sub-structure of the program.

The next chapter of this report deals exclusively with the modular structure of the "ultimate" program for interplanetary analysis. In the next chapter, the specific modules comprising the structure are identified in detail and the flexibility of the design is demonstrated in a heuristic fashion. The philosophy behind the designed structure will become more apparent as the particular modules are defined and their interlocking duties are enumerated. An appendix referenced in the next chapter, Appendix I, contains logical flow charts for these modules. These flow charts, representing in some cases already coded control statements, should be valid for all the tasks the "ultimate" program might perform.

The third chapter explains a tutorial program that has been actually coded, within the basic modular structure, to demonstrate the way in which the modules tie together. Due to financial limitations under the current contract, the computational models for the tutorial program are extremely simple and are severely limited in their real-world application. Nevertheless, they are valuable because they demonstrate an actual implementation of the designed modular structure. In addition, since more sophisticated models could easily be inserted without too much difficulty, the tutorial program represents the first step toward development of the

ultimate program. The flow charts and listings for this tutorial program appear in Appendix II.

In chapter IV, numerical results generated by the tutorial program are presented. Again it should be stressed that since the results are based upon overly-simple computational models, little significance should be assigned to the numbers themselves. What is most important in the chapter on results is the nature or quantitative aspect of the numbers. In other words, the results show what kind of questions can be answered by the program rather than particular numerical values of any special significance.

It is thought that the program design presented in this document represents the first step toward a truly comprehensive interplanetary analysis tool. Special care was taken in designing the modular structure to insure that the program would indeed be flexible and would permit changes with a minimum of effort. Thus, for example, if a more sophisticated guidance policy were developed, it could be integrated into the existing program without undue expenditure of effort. In addition, by developing a modular program to handle such a variety of interplanetary tasks, it seems as if much of the customary duplication of work has been circumvented.

What has already been coded and developed within the program design is not extremely useful. An ideal final program, based upon the modular structure presented in this document, would include a multitude of computational models covering all ranges of technical sophistication. These models or options would be located, conceptually, within the dotted blocks of the design flow charts given in the first Appendix. To use the program,

the analyst would simply read in an input statement specifying the operational mode and concomitant computational models needed to handle a particular problem. It is suggested that such a comprehensive analysis program is realizable within the framework of the modular design advanced in this report.

II. ULTIMATE PROGRAM DESIGN

The purpose of this chapter of the report is to explain in detail the conceptual design of a computer program that would perform all the tasks discussed in the introduction. More specifically, thirteen modules that would form the sub-structure of the ultimate program will be presented together with a discussion of the manner in which these modules would function.

It is important to review briefly the major considerations that led to the modular structure to be presented. Past experience with large computer programs emphasized that flexibility is a characteristic of all good programs. With respect to the designed program for interplanetary analysis, two specific kinds of flexibility were sought. The first kind could be called computational flexibility. By computational flexibility we mean the ability to change or replace actual portions of the computational process without seriously altering the overall program. This factor is vitally important in any interplanetary analysis program. As an example of the importance of computational flexibility in the program under consideration, one need only cite the numerous new and potentially powerful midcourse guidance policies that are currently being developed. To prevent the program from becoming obsolete, it must be designed so that new guidance laws can easily be incorporated into the existing logical flow.

The second kind of flexibility included in the design criteria was operational flexibility. This facet of the designed program was briefly touched upon in the introduction and refers to the program's ability to

operate on several different classes of problems. Since the computational sophistication required for a real-time trajectory estimation program is, in general, an order of magnitude greater than that needed for pre-flight parametric studies of various kinds, it is current practice to use separate programs for the two tasks. Nevertheless, there is enough overlap in the two classes of problems to justify creating a single program with a single modular structure to handle both jobs. By making computational models or options of differing levels of technical sophistication available to the program, the analyst can then choose the computational complexity needed to satisfy the requirements of any particular study. Thus, by varying an initial input statement, the computation of state transition matrices along the transfer trajectory might be changed from simple numerical differencing based upon two-body approximations to fourth order Runge-Kutta integration of the full variational equations representing deviations about the nominal trajectory.

An added inducement to making the program operationally flexible was the realization that the program should have a simulation capability. The application of advanced estimation theory to trajectory problems is not new; however, there remain some computational difficulties associated with the application that have not yet been satisfactorily rectified. If a simulation capability is included in an interplanetary program, the analyst has the opportunity, after the mission has been defined, to check the validity of his estimation process as well as the degree of realism of his pre-flight error propagation studies. When an actual interplanetary mission is about to be flown, the simulation capability can be used to

determine such things as the quantitative error induced in the estimation procedure due to failure to model correctly all the numerous error sources. In addition, by exercising the program in its simulation mode, during which one portion of the program is assumed omniscient and knows the trajectory exactly, a certain degree of familiarity with the program's operation can be obtained that could prove to be invaluable if peculiarities occur during the actual flight.

Two other concerns during the design process were program efficiency and simplicity. With respect to efficiency, an attempt was made to write the program with a minimum of Fortran statements and computational models. The use of machine language, particularly in those modules whose primary function is program control, would probably improve even more the efficiency of the program. Simplicity was considered very important for use of the program. In particular, the input requirements have been defined in such a way as to make the program readily accessible to someone not familiar with all the program's inner workings. The program outputs have also been clearly labeled for easy recognition and interpretation.

The result of the design process is a modular structure for the overall computer program that is given schematically in Figure II-1. The different arrows in the figure indicate paths that the information flow might follow for different classes of problems. Basically, there are thirteen identifiable modules, to be discussed below, that can be broken into four convenient groups according to the type of function they perform. The first group, called The Executive Group, constitutes what might generally be called the "main program" and is concerned with the control of the

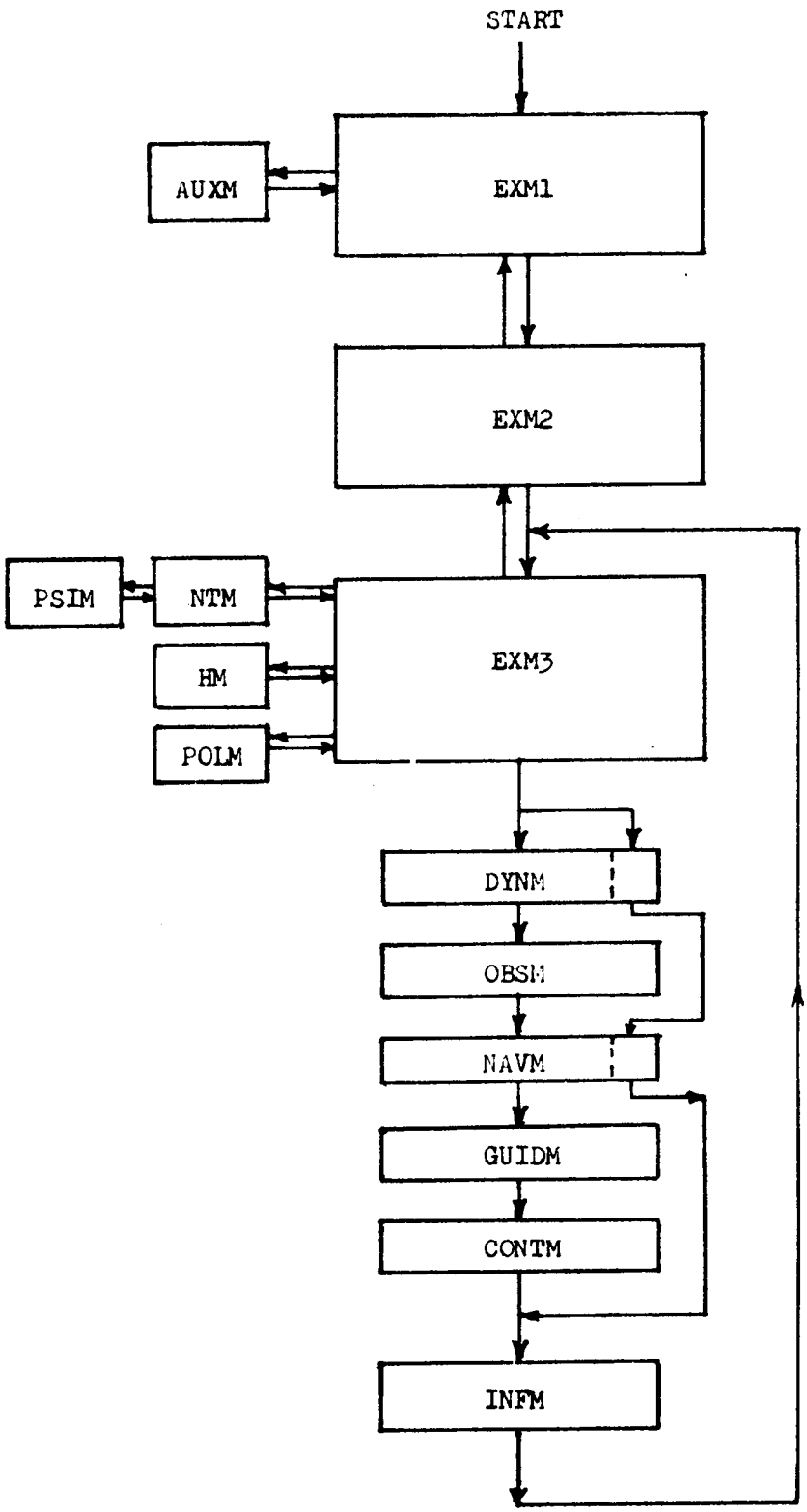

START
AUXM
EXM1
EXM2
PSIM
NTM
HM
POLM
EXM3
DYNM
OBSM
NAVM
GUIDM
CONTM
INFM

FIGURE II-1

MODULE DESCRIPTION

Symbol	Name	Function
EXM1	Executive Module 1	Control analysis on a trajectory pass basis.
EXM2	Executive Module 2	Control a single trajectory pass on an Event basis.
EXM3	Executive Module 3	Control analysis on a computation cycle and navigation cycle basis.
	* * * * * * *	
NTM	Nominal Trajectory Module	Generate nominal trajectory.
PSIM	PSI Module	Generate state transition matrices as needed.
HM	H-Matrix Module	Compute observation matrix at navigation epochs.
	* * * * * * *	
AUXM	Auxiliary Module	Preprocessing of input data, post processing of output data.
POLM	Policy Module	Generate midcourse correction matrices in accordance with guidance policy model.
INFM	Information Module	Control all data printout.
	* * * * * * *	
DYNM	Dynamics Module	Propagate true state data "along" nominal trajectory.
OBSM	Observation Module	Compute observation data from state data.
NAVM	Navigation Module	Compute state estimate and related data.
GUIDM	Guidance Module	Compute midcourse guidance correction and related data.
CONTM	Control Module	Compute execution errors and related data.

program's logical flow. A second group of modules has been designated as The Supporting Group. This group consists of trajectory dependent modules whose purpose is to supply computations of the nominal trajectory, state transition matrices, and observation matrices.

The Dynamics Group of modules, which comprise the third grouping, is involved primarily in the cycle-by-cycle estimation process and the computation of associated covariance matrices. Included within this group are one module that performs recursive or non-recursive filtering and another that computes actual state vector information to be used when the program is being exercised in its simulation mode. The final group, called The Auxiliary Group, consists of the remaining three modules that cannot be legitimately lumped into one of the other categories. These diverse modules deal with such items as processing input-output data, controlling data printout, and generating midcourse correction matrices in accordance with a specified guidance policy.

A. The Executive Group

The three modules that comprise the Executive Group are designated in Figure II-1 as EXM1, EXM2, and EXM3. The purpose of these modules is to control the logical sequence of the entire program. At the present time, EXM1 has not been designed in detail although its functions are enumerated in the schematic flow chart found in Appendix I. The same statement holds true for those portions of EXM2 that were not needed for the tutorial program. It is believed that parts of EXM2 and virtually all of EXM1 should eventually be written in machine language to increase the program's overall efficiency.

Module EXM1 is at the highest executive level. It controls the program on a trajectory run basis, making available all pertinent data as well as starting and terminating each run. Its other purposes include controlling access to individual run inputs in a multiple run operation and communicating with AUXM, an input-output module, on those runs that are being generated by random variables for Monte Carlo processing.

The second executive module, EXM2, exercises program control on an event basis. The event concept is of fundamental importance in the design of the ultimate program and is worth defining in some detail with respect to EXM2. An arbitrary number of discrete points along the nominal trajectory are designated as events. These may arise naturally from mission considerations (for example, a pre-determined time of midcourse correction) or may be chosen by the analyst for some computational reason, such as a point where the dynamic model can be conveniently changed. At all events, through the control of EXM2, the capability exists to change parameter values, to specify different computational models, and to vary the printed output.

Events where midcourse guidance corrections are computed are termed "Guidance Events". Since a great many observation-navigation points may be involved, they are referred to as "Navigation Epochs" and are not generally given event status.

Every event has an associated ESN (Event Sequence Number) and a corresponding TEV (Time of Event). These quantities are input as two arrays of like dimension, with the fixed point ESN's necessarily increasing as the corresponding TEV's increase. Events that are also guidance events are identified by the array (GEV) of their Event Sequence Numbers.

At each event, EXM2 is signalled and comes into the process to exercise control. EXM2 reads a deck of cards that contains all the input necessary to call the desired computational models as sub-routines, perform the requisite initialization processes, and introduce new data. Eventually EXM2 would be programmed in machine language for greater efficiency.

The third executive module, EXM3, controls the program on a computation and navigation cycle basis. Its internal structure has been completed and is given in the flow chart in Appendix I. One of the tasks of EXM3 is to keep track of all the different times needed by the program. The current version of EXM3 contains two time registers, TD denoting Dynamics Time and TN denoting Navigation Time. It is conceivable that a more sophisticated tracking model would require additional time registers in EXM3.

The module EXM3 also sets various decision flags, which are indicated in the flow chart by symbols ending in F. The two most important of these decision flags are IPRNTF and IEVF. The first of these controls the frequency of data printout while the second flag directs the logical flow back up into EXM2 after any computation cycle terminating at an Event Time.

The Executive Group of modules is currently the group that is designed in the least detail. In working upwards in sophistication from the tutorial program, considerable effort should be expended on the Executive Group. In addition, it must be pointed out that no computational model options are available to these control modules and hence careful attention should be paid in their eventual implementation, to building a control flexibility that is commensurate with the flexibility of the rest of the program.

B. The Supporting Group

The three modules that make up The Supporting Group all provide computational data to EXM3. As shown in Figure II-1, these three modules are PSIM, NTM, and HM. The fourth module feeding into EXM3 is discussed in The Auxiliary Group.

Module NTM is responsible for nominal trajectory computations. Its core is a computational model that advances the values for the nominal trajectory from the latest time to a next time defined by an increment Δt. Provision has been made for the increment Δt to be computed internally where desired.

In Appendix I, a brief flow chart of NTM as it would appear in the ultimate program is given. The importance of this flow chart is the dotted black box entitled model insert. In the ultimate program, several computational models would be stored here, to be called upon at the analyst's prerogative. As an example, three of these options might be two-body approximation, virtual mass trajectory generation, and Encke integration. By reading in an input statement, the analyst would activate the computational technique he desired to use for computing the nominal trajectory.

Module PSIM, which is responsible for the state transition matrix computations, also shows a dotted black box in which several available computational models might be stored. Ideally, differing levels of computational sophistication in computing the state transition matrices would be accessible to the program at this point in the flow. Model options might range from state transition matrices based upon conic section trajectories to fourth order Runge Kutta integration of the variational equations.

The third module in The Supporting Group has been denoted by HM. Its task is to take the nominal trajectory information and compute a nominal set of observational data. In addition, it generates the so-called H matrix, which is needed in the estimation process, that relates linear perturbations about the nominal observations to linear perturbations in the nominal trajectory. The flow chart for HM is also given in Appendix I and contains a dotted black box for the computational model insert. In the ultimate program, a series of options, representing different kinds of observational readings, would be stored in this location to be activated at will by the analyst. Possible model options would range from Earth-based Doppler data to on-board sensor information and could conceivably include atmospheric measurements if the program were extended to include the re-entry phase of the interplanetary mission.

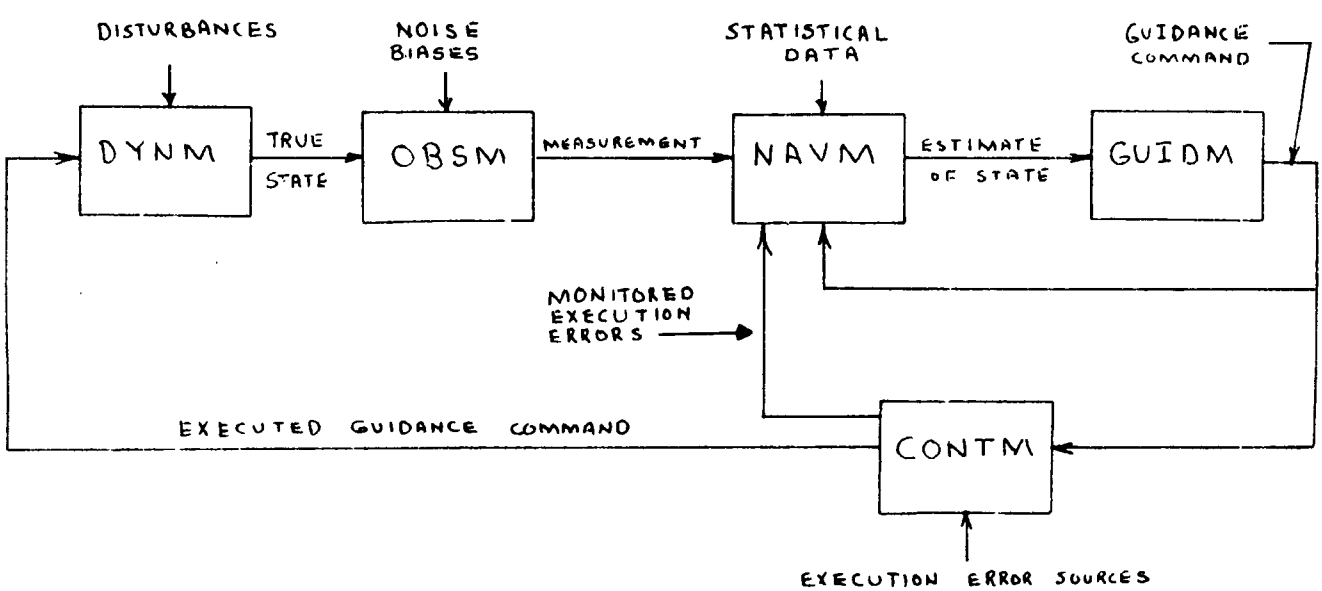
C. The Dynamics Group

This group, which comprises DYNM, OBSM, NAVM, GUIDM, and CONTM, is in many ways the central part of the program and dictates the entire logical structure thereof. As computation progresses from point to point along a nominal trajectory this group performs two distinct but related functions.

1) The computation of data pertaining to the actual trajectory being flown by an off-nominal vehicle and to contaminated measurements available to that vehicle (or a ground tracking network).

2) Computation of data that can be regarded as pertaining to a simulation of the vehicle's navigation, guidance, and control systems.

The accompanying block diagram describes the basic information flow through modules of the Dynamics Group, which is entered only at Navigation Epochs.



Following are descriptions of individual modules within the group.

1. Dynamics Module, DYNM

The duty of this module is to keep track of all necessary data pertaining to the actual flight of an off-nominal vehicle, in a manner separate from the estimation procedure being simulated in NAVM. Disturbances to the system, resulting in the off-nominal behavior being described include dynamic noise, systematic forcing functions, variations from nominal of systematic forcing functions involved in generation of the nominal trajectory (e.g. variations from nominal of a micro-thrust program or of programmed impulsive maneuvers), and variations from nominal of parameters in the dynamic model (e.g. parameters of the gravitational field). Since the Dynamics Group is only entered At Navigation epochs, the difference equations employed here must involve the cumulative effects over the navigation cycle of the effects just described. Currently these cumulative effects are evaluated in PSIM over the smaller computation interval Δt_D used by that module and the necessary data transferred at each DYNM entry. However, in the ultimate program it may prove advantageous to alter this procedure. A special pass is made thru DYNM after every midcourse correction for the purpose of updating dynamics information in accordance with the correction actually executed.

Two types of models will be used within this module, depending on the mode of analysis being employed. One will be strictly for propagating the covariance matrix of the true state, and will not involve numerical values of any random variables entering the picture.

The other, however, will propagate the true state vector as it is affected by various numerically input error sources. This second class of models can be employed in various studies utilizing the simulation mode, and also in Monte Carlo analyses.

The module structure is basically as given by the block diagram in Appendix I. Two models are provided, Model A being entered on a normal Computation when MCCF is 1 for updating after a midcourse correction.

2. Observation Module, OBSM

This module has the function of providing contaminated observation data to the Navigation Module. On the basis of the Observation Matrix H, computed in HM, and the true state, it computes data related to what might be termed a true observation. This is then modified by the introduction of measurement noise and biases and sent on to NAVM. Flexibility is gained by computing H in a separate module, since the form of the equations employed in OBSM are not in general heavily dependent on choice of coordinate system used for computing the trajectory related data.

In the simulation mode the models employed in OBSM are expected to be quite similar to that described in Part III for the Tutorial program. For covariance matrix propagation usage, the models will probably be "do nothing" models.

In Appendix I the block diagram for OBSM is given, and is self-explanatory.

3. Navigation Module, NAVM

This module has the function of processing contaminated observation data from OBSM to generate data pertaining to the state estimate (which may be augmented). The module structure is very simple and is presented as a block diagram in Appendix I. As is the case in DYNM, provision is made for entry after a midcourse correction for the purpose of updating estimates in accordance with whatever information is actually available to the navigation system.

Little will be said here of model structure, since this is highly dependent on individual problem formulation and is expected to vary widely. A simple example of a model for usage in the simulation mode is that described in the tutorial example. For studies involving covariance matrix propagation, a rather sophisticated model is envisioned involving an augmented state of arbitrary dimension. It can readily include propagation of a covariance matrix of the actual error in the estimate for comparison with the matrix P being recursively estimated.

4. Guidance Module, GUIDM

In GUIDM the state estimate from NAVM is operated on by means of matrices generated in POLM during a Guidance Preset Pass. In the case of models such as those described in Part III for the tutorial program, the result is a linear impulsive velocity correction computed for each midcourse guidance event. Future versions of models to be inserted in GUIDM are expected to also include decision processes, which make possible two things. The first, and simplest, is the provision of a lower bound on any velocity correction made based on

criteria involving statistics of the execution error sources. The second, and much more interesting, is that by specifying a great many "candidate" midcourse correction points and appropriate decision processes, some of the newer so-called adaptive guidance laws can be simulated.

Another area in which this module is expected to play a major role is that involving continuous guidance (e.g. a long-term micro thrust program). This would require guidance equations similar in certain respects to those employed in present day booster flight. It is probable that simulation of such schemes would still involve pre-computation in POLM of certain data based on a nominal trajectory, with GUIDM employing this data during normal runs.

The block diagram for GUIDM is presented in Appendix I. This is designed to accept a wide variety of models involving linear impulsive guidance schemes and other more sophisticated schemes that can be modeled on that basis. Upon first entry at a Guidance event, MCCF will be zero and GUIDF one. Except at the last guidance event (when K = KL, at the target), the velocity correction will be computed. The module does nothing during a second updating pass that follows execution of the computed correction.

5. Control Module, CONTM

The task assined to this model is the execution of the guidance command computed in GUIDM, subject to a variety of control errors. These include such things as pointing errors, scale factor errors, and bias errors of several kinds. Models of wide range in complexity

will be needed for this module, depending on the emphasis in an analysis. A good example of a simplified model is that described in Part III for the tutorial program. It operates in the simulation mode, with numerical values being input for all random error sources. It also computes statistical data for NAVM, using probabilities conditioned on knowledge of the commanded velocity correction.

Models for the covariance matrix propagation mode of analysis will be simpler and highly idealized, since this portion of the problem is highly non-linear. At the other extreme, models can readily be generated for handling in a realistic manner the multitude of errors that can affect the execution of guidance commands.

The CONTM block diagram in Appendix I accepts models involving impulsive velocity corrections. During a first pass through the module at a guidance event (with MCCF = 0 and GUIDF = 1) MCCF is set to 1. This controls the logical flow through the entire Dynamics group during the second pass following a guidance correction. During the second pass, MCCF is reset to zero.

D. Auxiliary Group

This group includes models that are auxiliary to the basic program and, unlike those of the supporting group, do not fall into some well defined class. Included here are the Policy Module POLM, Information Module INFM, and Auxiliary Module AUXM.

1. Policy Module POLM

The Policy Module functions during a Guidance Preset run to generate data to be used, during subsequent runs, by the guidance

system. It does this on the basis of the nominal trajectory computed in NTM. An important analytical capability exists here in that the nominal trajectory used by POLM does not have to agree with that used in later runs. Thus it is possible to include in an analysis the effects of using inaccurate precomputed guidance data. The version of POLM described by the flow chart in Appendix I accepts models that operate on state transition matrices computed in PSIM to generate matrices for linear impulsive guidance laws. (A good example of such a model is that described in Part III for the tutorial program, which utilizes a fixed time-of-arrival guidance constraint.)

During the Kth entry into POLM (which will be at the Kth guidance event), for all $K > 1$ the prevailing value of the state transition matrix from PSIM is stored as Φ_{k-1}. When $K = KL$, denoting arrival at some target situation, the model insert is entered for computation of all guidance data pertaining to the entire trajectory.

2. <u>Information Module INFM</u>

This module will handle all data output functions. At each entry, on the basis of flag settings, various blocks of data are caused to be printed out. This and print frequency are discussed in Part III. Here a few words are in order concerning the structure of the module. Because of the diverse nature of output data desired at different times and types of events, it appears desirable to incorporate what may appear to be an excessively large number of model inserts. It is important, however, to maintain great flexibility of output, and this can be done in such a manner. It is quite probable,

however, that well defined groups of models can be formed (with some obvious duplication involved), each of which can be used with a given compatible set of models. This nebulous nature of INFM results directly from the fact that the modular structure of the Ultimate Program is aimed at creating programs that are "tailor made" for specific problems.

3. Auxiliary Module AUXM

This module is to be used for special processing of input and output data. It works with EXM1, where entry is effected only at the beginning and end of a run. Although many of its duties are undefined at this time, the part it will play in a Monte Carlo analysis represents, perhaps, an extreme case. For such an analysis it will accept covariance matrices describing multivariate Gaussian distributions (or data describing other distributions), properly sample these distributions to select initial values of all random variables appearing in the simulation, and after the run process results so as to yield statistical data on the output random variables.

Because of the limitations imposed by the magnitude of the present dtudy, no work has been done to further define AUXM.

III. TUTORIAL PROGRAM

This section describes in all necessary detail the Tutorial Program that has been coded and checked out. The description in Part II of the Ultimate Program applies here for the most part, and will not be repeated as such. Instead, the simplifications made in producing the tutorial version of that program will be discussed, together with detailed descriptions of the models provided.

The major area of simplification has been at the executive level - in particular, EXM1 and EXM2. Most of the functions identified for EXM1 and many of those for EXM2 require a high level of programming effort involving a great deal of machine language. Because of the limited nature of the present contract, therefore, most of EXM1 and part of EXM2 have not been designed. The need for these as yet unavailable parts has been avoided in the tutorial version by assembling program and input decks in a conventional manner. This allows verification of the major concepts of the program, but does not provide many of the desirable features such as automatic run and data processing control for Monte Carlo analyses, or extreme flexibility and simplicity of model change and re-initialization at event times. In this respect, the input-output procedures of the tutorial version bear little resemblance to those forseen for the ultimate program. A related simplification lies with the coding of all modules and contained models belonging to the Supporting, Dynamics, and Auxiliary groups. For each of these the tutorial version has a single subroutine containing all the modular logic together with the model, coded in such a manner that their individual identities are partially obscured.

A. Executive Group

The discussion of the functions of the members of the Executive Group presented in Part II is, in general, applicable here too. Reference should be made to that material wherever necessary to understand the abbreviated descriptions that follow. Diagrams of these modules are to be found with the Main Program listing of Appendix II.

1. EXM1

The tutorial version of EXM1 has the simple function of initiating a series of runs, keeping a count of those completed, and commanding STOP when a preset number has been reached. It also sets IGPRF, the Guidance Present Flag, by reading an input card. The initialization model EXM1 IN1 sets the cumulative run register IRUN to zero and reads a card for IRUNX, the total number of runs desired.

	Variable	Program Symbol	Description	Source
Internal Input	none			
External Input	RUNX	IRUNX	Number of Runs in sequence	
Output	GPRF	IGPRF	Guidance Preset Flag	

2. EXM2

The tutorial version of EXM2, as with the ultimate program, exercises control during individual runs on an Event basis. It updates the ESN and TEV registers at each entry, and if the previous event was a Guidance Event, also updates the GEV register. The initialization flag INIF, however, is active only at the first event of a run, calling

the Input subroutine which performs all initialization for modules of hierarchy below the executive level.

	Variable	Program Symbol	Description	Source
Internal Input				
	GPRF	IGPRF	Guidance Preset Flag	EXM1
External Input				
	ESM( )	IESN( )	Event Sequence No. Array	
	TEV( )	TEV( )	Time of Event array	
	GEV( )	IGEV( )	Guidance Event array	
	ESNX	IESNX	final ESN	
	JL	JEVL	final ESN index	
	KL	KL	final GEV index	
Output				
	J	JEV	current ESN index	
	ESN(J)	IIESN	current ESN	
	TEV(J)	TTEV	current Time of Event	
	K	K	current GEV index	
	GEV(K)	IIGEV	current GEV	

3. EXM3

This module exercises control on the basis of Dynamics Time and Navigation Time cycles. It is identical for both the ultimate program and the tutorial version. For a verbal description of its operation, see Part II, and for a block diagram see Appendix II.

	Variable	Program Symbol	Description	Source
Internal Input				
	GPRF	IGPRF	Guidance Preset Flag	EXM1
	TEV	TTEV	Time of Next Event	EXM2
	IIESN	ESN	Next Event No.	EXM2
	IIGEV	GEV	Next Guidance Event	EXM2
	MCCF	MCCF	Midcourse Correction Flag	CONTM
External Input				
	None			
Output				
	t_D	TD	Dynamics Time	
	t_N	TN	Navigation Time	
	NAVF	NAVF	Navigation Flag	
	PRNTF	IPRNTF	Print Flag	
	EVF	IEVF	Event Flag	
	GUIDF	IGUIDF	Guidance Flag	

B. Supporting Group

This group includes NTM, PSIM, and HM. Their functions, in the order given, are to generate the nominal trajectory, compute state transition matrices, and compute associated observation matrices; all in support of

the Dynamics Group computations pertaining to an off-nominal vehicle. NTM and PSIM are entered each Dynamics Time cycle, whereas HM is entered only on a Navigation Time cycle basis. Each of these modules contains the simple logic necessary to link their imbedded models to EXM3, and are unchanged from the descriptions in Part II. As mentioned before, however, for the tutorial version both module and model are coded together as a single subroutine. These are listed in Appendix II, the contained models being categorized as Series 1. This series of subroutines is described in detail after the following few words on the manner in which they "play together".

At each incremental updating of dynamics time t_D the following computational sequence is performed:

1) In NTM the nominal trajectory is advanced by means of difference equations based on the planar differential equations of motion in a central force field.

2) In PSIM perturbation equations derived from the NTM Model 1 equations are used to update the state transition matrix which maps the state (herein defined as a 6-vector of perturbations from nominal of position and velocity) at the previous navigation epoch forward to the new t_D. Also matrices are computed which give the cumulative effect of one parameter (the gravitational constant of the sun), and the cumulative effect of white acceleration noise; both over the same time interval.

3) If the updated dynamics time is <u>also</u> a Navigation epoch, HM is entered. Here an idealized earth-based tracking system is modeled,

and the observation matrix H is generated relating perturbations from nominal of tracking measurements to the off-nominal vehicle state.

1. NTM Nominal Trajectory Module, Model 1

This is a simplified planar model which generates a nominal trajectory given initial conditions on the polar coordinates r, θ, $\dot{r}$, $\dot{\theta}$. The approximate procedure used to solve the differential equations of motion is described in the Addendum to Part III, and should be sufficiently accurate to form the basis of a perturbation study if the integration interval Δt is chosen such $\frac{\dot{r}\Delta t}{r} < .01$. (Note, however, that PSIM Model 1 imposes a lower bound on Δt_D.)

At each entry into the model, Δt is computed by subtracting the previous value of t_D from the new value given it in EXM3. The equations are

$$\Delta t = t_D - t_{D(save)} \tag{1}$$

$$t_{D(save)} = t_D \tag{2}$$

The incremental changes in the nominal state variables r, $\dot{r}$, and θ over the interval Δt are computed by means of the following equations (not numbered as in the Part III Addendum):

$$D_3 = 1 - \left(\frac{\mu}{r^3} - \frac{3}{2}\frac{h^2}{r^4}\right)\frac{\Delta t^2}{2} \tag{3}$$

$$\Delta r = \frac{1}{D_3}\left[\dot{r}\Delta t + \left(\frac{h^2}{r^3} - \frac{\mu}{r^2}\right)\frac{\Delta t^2}{2}\right] \tag{4}$$

$$\Delta \dot{r} = \frac{1}{D_3}\left[\left(\frac{h^2}{r^3} - \frac{\mu}{r^2}\right)\Delta t + \left(\frac{\mu}{r^3} - \frac{3}{2}\frac{h^2}{r^4}\right)\dot{r}\Delta t^2\right] \tag{5}$$

$$\Delta\theta = \frac{h}{r^2}\left(1 - \frac{\Delta r}{r}\right)\Delta t \tag{6}$$

Then r, $\dot{r}$, and θ are updated with these increments, following which the new value of $\dot{\theta}$ is obtained from the expression

$$\dot{\theta} = \frac{h}{r^2} \quad . \tag{7}$$

Internal Input

				Source
	t_D	TD	Dynamics Time	EXM3

External Input

	Math Symbol	Program Symbol	Description	Units
Initial values	r	R	radius vector	km
	θ	THETA	central angle	radians
	$\dot{r}$	RDOT	radial velocity	km/ks
	$\dot{\theta}$	THEDT	angular rate	rad/ks
	θ_E	THETAE	central angle, earth	rad
	t_D(save)	TDSV	previous value of t_D	ks
	μ	MU	Gravitational constant of sun	km^3/ks^2
	r_E	RE	radius vector, earth	km
	$\dot{\theta}_E$	THEDTE	central angle rate, earth	rad/ks

Output

r	R	see above
θ	THETA	"
$\dot{r}$	RDOT	"
$\dot{\theta}$	THEDT	"
θ_E	THETAE	"
r_E	RE	"
Δt	DELT	Dynamics Time Increment
h	H	Unit angular momentum

NTM Initialization Model 1

All External Input listed for NTM Model 1 is entered by means of this Initialization Model. In addition it computes from the initial conditions the unit angular momentum for the nominal trajectory:

$$h = r^2 \dot{\theta} .$$

2. PSIM Model 1

This is a simplified model which generates the state transition matrix $\Psi = \dfrac{\partial(x_{t_i} + \Delta t)}{\partial(x_{t_i})}$ by solving variational equations based on planar equations of motion in a central force field. It is a companion to NTM Model 1 which computes the underlying nominal trajectory in the same coordinate system. Model 1 is intended for use in preliminary analyses where a large integration step size is desired (e.g. a Δt corresponding to 1° in central angle). With this in mind, at each computation step the cumulative $\Psi' \stackrel{\text{def}}{=} \Psi_{t_i, t_o}$ is updated by premultiplication by Ψ. Thus a small Δt (and hence a great many computation steps) may result in excessive roundoff errors. This difficulty is avoided by placing a lower bound on the Δt employed, depending on the problem being studied. If many (e.g. several thousand) observation points are to be considered, a different model should probably be used).

The equations include a perturbation of the gravitational constant μ of the central force field. This has been done to facilitate a simple study involving augmentation of the state vector with one parameter, and involves generation of the matrix $\Gamma = \dfrac{\partial(x_{t_i} + \Delta t)}{\partial \mu}$

over the interval Δt, where the variation of μ is held constant. At each computation step the cumulative $\Gamma' \stackrel{def}{=} \Gamma_{t_i, t_o}$ is updated by premultiplying by Ψ and adding Γ to the resulting product.

In a similar manner Model 1 also processes an off-nominal acceleration disturbance q over each interval Δt. Currently this takes the form of a white noise acceleration input obtained by means of a random number generator. The cumulative q'_{t_i, t_o} is updated by premultiplying by Ψ and adding $q\Delta t$ to the velocity components of the resulting product. Eventually it is planned that, through the use of sub-models, q can include other types of random or systematic disturbances.

The computations performed by Model 1 are given below.

<u>Reset Pass</u> - This is the first pass following a Navigation Epoch, and is controlled by the Reset Flag RESF. Ψ', Γ', and q' are reset as follows:

$$\Psi' = I$$

$$\Gamma' = 0$$

$$q' = 0$$

<u>Matrix Computation and Updating</u>

At each entry the elements of Ψ are computed in accordance with Equations (1) thru (7) of the Part III Addendum. In computing the elements of A, mean values over the interval Δt_D of all quantities evaluated in Equations (2) are used.

Internal Input

Math Symbol	Program Symbol	Description	Source
μ	MU	Gravitational Constant of sum	NTM
r	R	radius vector	NTM
$\dot{\theta}$	THEDT	central angular velocity	NTM
$\dot{r}$	RDOT	radial velocity	NTM
$\Delta t, \Delta t_D$	DELT	dynamics time increment	NTM
NAVF	NAVF	Navigation Flag	EXM3
RESF	IRESF	Matrix Reset Flag	PSIM

External Input

σ_q(3x1)	SIGMQ(I)	Second moments of Gaussian distributions from which the elements of q are randomly selected.

Output

Ψ'(6x6)	PSIP(I,J)	Cumulative State Transition Matrix
Γ'(6x1)	GAMMAP(I,1)	Cumulative parameter matrix
q (3x1)	Q(I)	White noise acceleration disturbance
q'(6x1)	QP(I)	Cumulative dynamic noise vector

Subroutines Called

Function RNUM

PSIM Initialization Model

This model computes initial values of the a_1, a_2, a_3, b_1, b_2, b_3, and c of PSIM Model 1 Appendix, Eqn. (2). It also initializes the Matrix Reset Flag RESF to 1.

3. HM Model 1

Model 1 is a simplified model of an earth-based tracking system intended for use in error analyses where this aspect of the problem is of secondary importance. It assumes that at each observation time a tracking station is available which can look directly overhead at the nominal vehicle being studied. The present version of Model 1 assumes further that the vehicle trajectory lies in the ecliptic plane, although it should not be difficult to modify the model to handle less restrictive cases.

Four measurements are assumed: range R, range rate $\dot{R}$, and angles α and β in and normal to the ecliptic plane.* The model generates the observation (H) matrix as defined below, by means of equations (2) thru (10). See the Part III Addendum for a figure and derivations.

$$H = \frac{\partial(R,\dot{R},\alpha,\beta)}{\partial(r,\Theta,N,\dot{r},\dot{\Theta},\dot{N})} = \begin{bmatrix} E_1 & E_2 x10^{-3} & 0 & 0 & 0 & 0 \\ F_1 & F_2 x10^{-3} & 0 & E_1 & E_2 x10^{-3} & 0 \\ G_1 x10^3 & G_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{R} x10^3 & 0 & 0 & 0 \end{bmatrix} \quad (1)$$

where

$$R = \left[r^2 + r_E^{\,2} - 2\, r\, r_E \cos(\Theta_E - \Theta)\right]^{1/2} \quad (2)$$

$$E_1 = \frac{1}{R}\left[r - r_E \cos(\Theta_E - \Theta)\right] \quad (3)$$

---

* The tracker assumed would measure directly deviations from the nominal values of α and β.

$$E_2 = -\frac{1}{R}\, r\, r_E \sin(\Theta_E - \Theta) \tag{4}$$

$$\dot{R} = E_1 \dot{r} - E_2(\dot{\Theta}_E - \dot{\Theta}) \tag{5}$$

$$F_1 = \frac{1}{R}\left[-\dot{R}E_1 + \dot{r} + r_E \sin(\Theta_E - \Theta)(\dot{\Theta}_E - \dot{\Theta})\right] \tag{6}$$

$$F_2 = -\frac{1}{R}\left[\dot{R}E_2 + \dot{r}\, r_E \sin(\Theta_E - \Theta) + r\, r_E \cos(\Theta_E - \Theta)(\dot{\Theta}_E - \dot{\Theta})\right] \tag{7}$$

$$\cos\alpha = \frac{r^2 - r_E^{\,2} - R^2}{2r_E R} \tag{8}$$

$$G_1 = \frac{\sin(\Theta_E - \Theta)}{R\cos\alpha}\left[1 - \frac{E_1 r}{R}\right] \tag{9}$$

$$G_2 = -\frac{r}{R\cos\alpha}\left[\cos(\Theta_E - \Theta) + \frac{E_2 \sin(\Theta_E - \Theta)}{R}\right] \tag{10}$$

Internal Input

Math Symbol	Program Symbol	Description	Source
r	R	radius vector	NTM
Θ	THETA	central angle	NTM
Θ_E	THETAE	central angle, Earth	NTM
r_E	RE	radius vector, Earth	NTM
$\dot{r}$	RDOT	radial velocity	NTM
$\dot{\Theta}$	THEDT	central angle rate	NTM
$\dot{\Theta}_E$	THEDTE	central angle rate, Earth	NTM

External Input

None

Output

H	TM(4,6)	Observation Matrix
R	RNG	nominal radar range, range rate
$\dot{R}$	RNGDT	nominal radar range, range rate

sin α	SALPHA	sin and cos of nominal elevation angle α (see diagram in Appendix)
cos α	CALPHA	

C. The Dynamics Group of Modules: DYNM, OBSM, NAVM, GUIDM, CONTM

The Dynamics group is entered only at navigation epochs, as determined by the navigation time increment Δt_N and Guidance Events. Entry in controlled in EXM3 by means of the Navigation Flag NAVF. This module group is actually the heart of the program, containing a compatible set of models that computes in parallel 1) data pertaining to the true state (perturbation), with noisy observations based thereon; and 2) data pertaining to the estimation process being employed by the navigation system being simulated. Capability is provided to simulate the guidance system which processes the navigation data in generating midcourse correction commands, and to apply these corrections subject to execution errors. Following a guidance correction a secondary pass is made thru the group, without advancement of the Dynamics Time register, in order to suitably update the state, the estimated state, and other data. Although each module of the tutorial program is coded as a subroutine with its model imbedded therein, the following descriptions apply to the model portions only.

1. Series 1 Models

This is a compatible series of discrete models designed to implement a simple deterministic analysis in which numerical values of the state are propagated from (time) point to point "along" a reference trajectory, a navigator recursively estimates the state, and midcourse velocity corrections based thereon are computed and executed. The end products include numerical values of target miss and total midcourse Δv expended. With the addition of a suitable Auxiliary Module (AUXM) to the tutorial program, the Series 1 Models can be used in Monte Carlo analyses.

a. DYNM Model 1

The plant is described by the difference equation

$$x_k = \Psi'_{k,k-1}\, x_{k-1} + \Gamma'_{k,k-1}\, r_s + q' \qquad (1)$$

where r_s is a vector of parameter errors and q' is a random noise vector. The coded version of this model pertains to an unaugmented state, and allows a single parameter error, $r = \delta\mu$, where μ is the gravitational constant of the acting central force field. Modification to generalize r_s is a relatively simple matter here, but may require significant changes in PSIM, where Γ' is computed. Ψ', the state transition matrix, and Γ' both pertain to the interval $(t_N + \Delta t_N, t_N)$. Similarly, q' is the cumulative effect over that interval of white acceleration noise, and is computed in PSIM on a dynamics time cycle basis. (See the PSIM writeup for additional information.)

Following a midcourse correction, on a secondary pass thru the model, the state is updated in accordance with the expression

$$x_k^+ = x_k + \begin{bmatrix} 0 \\ --- \\ \Delta v' \end{bmatrix} \qquad (2)$$

where $\Delta v'$ is the actual midcourse velocity change executed.

Variable	Program Symbol	Description	Source
Internal Input			
$\Psi'(6x6)$	PSIP(6,6)	State Transition Matrix	PSIM
$\Gamma'(6x1)$	GAMMAP(6,1)	Parameter Sensitivity Matrix	PSIM

$q'(6x1)$	QP(6)	Cumulative Dynamic Noise Vector	PSIM
$\Delta v'(3x1)$	DELTVP(3,1)	Executed velocity correction	CONTM
MCCF	MCCF	Midcourse Correction Flag	CONTM

External Input

$\delta\mu$	RS	Error in gravitational constant
$x_o(6x1)$	X(6,1)	Initial value of state

Output

x(6x1)	X(6,1)	True state

b. OBSM Model 1

This model computes the noisy measurement z at each navigation epoch, based on the observation matrix computed in HM and the true state x. Provision is made to add a measurement bias vector b which is related to z by means of a matrix L. (L is currently initialized in as an identity matrix, but will eventually constitute an external input.) The observation is contaminated by a white noise vector v which is computed in RNUM on the basis of a set of externally input standard deviations applying to the elements of z.

The observation equation at a navigation epoch at t_k is computed as

$$z_k = H_k x_k + L_k b + v_k \tag{1}$$

The current version of this model has been coded in a limited way, with fixed dimensions on matrices and vectors, making it a specific companion to HM Model 1. It is structured, however, so that greater generality can by obtained with little change.

	Variable	Program Symbol	Description	Source
Internal Input				
	H(4x6)	HM(4,6)	observation matrix	HM
	x(6x1)	X(6,1)	true state	DYNM
External Input				
	b(4x1)	B(4,1)	Measurement bias vector	
Output				
	z(4x1)	Z(4,1)	contaminated measurement vector	

c. NAVM Model 1

This model makes use of the observation vector z computed in OBSM in a recursive navigation scheme to compute an estimate $\hat{x}$ of the unaugmented state vector x. Also at each navigation epoch, the covariance matrix P of the error in the estimate is computed. It is to be noted that this model simulates, in effect, the navigator in a space mission, and is "isolated" from those parts of the program that propagate the true state. This makes possible the study of navigation processes based on inaccurate physical models and statistical parameters.

The filtering equations in Model 1 are

$$P_k' = \Psi_{k,k-1}' P_{k-1} \Psi_{k,k-1}'^{T} + Q_k$$

$$K_k = P' H_k^T \left[H_k P_k' H_k^T + P_{v_k} \right]^{-1}$$

$$P_k = \left[I - K_k H_k \right] P_k'$$

$$\hat{x}_k = \Psi_{k,k-1}' \hat{x}_{k-1} + K_k \left[z_k - H_k \Psi_{k,k-1}' \hat{x}_{k-1} \right]$$

$$\tilde{x}_k = x_k - \hat{x}_k$$

On a secondary pass thru the model, following a midcourse correction, $\hat{x}$ and P are updated to reflect changes known to the navigation system

$$\hat{x}_k^{+} = \hat{x}_k + \begin{bmatrix} 0 \\ \hline \Delta v + e_m \end{bmatrix}$$

$$P_k^+ = P_k + \left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & P_{e_u} \end{array}\right] .$$

The actual error in the estimate is also updated.

$$\tilde{x}_k^+ = \tilde{x}_k + \left[\begin{array}{c} 0 \\ \hline e_u \end{array}\right] .$$

Δv is the command computed by guidance, e_u and P_{e_u} are the unmonitored portion of the execution error and its covariance matrix, and e_m is the monitored portion.

	Variable	Program Symbol	Description	Source
Internal Input				
	$\Psi'(6x6)$	PSIP(6,6)	State Transition Matrix	PSIM
	$H(4x6)$	HM(4,6)	Observation Matrix	HM
	$z(4x1)$	Z(4,1)	Measurement Vector	OBSM
	$x(6x1)$	X(6,1)	State vector	DYNM
	MCCF	MCCF	Midcourse Correction Flag	CONTM
	NAVF	NAVF	Navigation Flag	EXM3
External Input				
	$Q(6x6)$	QQ(6,6)	Dynamic Covariance Matrix	
	$P_v(4x4)$	PV(4,4)	Measurement Covariance Matrix	
Output				
	$\hat{x}(6x1)$	XH(6,1)	Estimate of State	
	$\tilde{x}(6x1)$	XAPX(6,1)	Error in the Estimate	
	$P(6x6)$	P(6,6)	Covariance of Estimate Error	

d. GUIDM Model 1

This model is for deterministic analyses involving linear impulsive midcourse guidance laws of the form

$$\Delta v = G_k x_k \tag{1}$$

where G is a (3x6) matrix computed in POLM. This provides great flexibility in that a wide variety of guidance laws can be written in the form of Equation (1).

The error in the Δv command implied by the error in state estimate is computed here, as well as $|\Delta v|$.

$$e_n = G_k \tilde{x}_k \tag{2}$$

$$|\Delta v| = \sqrt{\Delta v_1^2 + \Delta v_2^2 + \Delta v_3^2} \tag{3}$$

	Variable	Program Symbol	Description	Source
Internal Input				
	$G_k(3x6)$	G(K,6,6)	Midcourse Correction Matrix	POLM
	$\hat{x}(6x1)$	XH(6,1)	State Estimate	NAVM
	$\tilde{x}(6x1)$	XAPX(6,1)	Error in Estimate	NAVM
	MCCF	MCCF	Midcourse Correction Flag	CONTM
	GUIDF	IGUIDF	Guidance Flag	EXM3
	K	K	Guidance Event Index	EXM2
	KL	KL	Final value of K	EXM2
External Input				
	None			

Output

$\Delta v(3x1)$	DELTV(3,1)	Midcourse Correction Command
$\lvert \Delta v \rvert$	DVMAG	Magnitude of Δv
$e_n(3x1)$	EN(3,1)	Error in Δv due to navigation error

e. CONTM Model 1

Model 1 computes the errors in execution of a midcourse correction. It is based in part on the approximate methods described by Gates*.

The unmonitored execution error is taken to be the sum of a pointing error, an error proportional to the magnitude of Δv and directed along Δv, and an error independent of the magnitude of Δv but directed along it. In that order,

$$e_u = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} \times \Delta v + e_4 \Delta v + e_5 \frac{\Delta v}{|\Delta v|} \qquad (1)$$

where e_1, e_2, and e_3 are assumed to be independent, Gaussian $(0, \sigma_p)$; and e_4 and e_5 are independent Gaussian $(0, \sigma_4)$ and $(0, \sigma_5)$ respectively. The term in (1) involving e_1, e_2, and e_3 is the vector cross product.

The monitored execution error is taken to be an error along Δv and independent of $|\Delta v|$; an example of which would be an off-nominal thrust decay measured by an axial accelerometer.

$$e_m = e_6 \frac{\Delta v}{|\Delta v|} \qquad (2)$$

---

* C. R. Gates, "A Simplified Model of Midcourse Maneuver Execution Errors", JPL Tech Rpt No. 32-504.

The errors e_1, e_2, ..., e_6 are externally input, and do not have to be consistent with the assumed standard deviations σ_p, σ_4, σ_5, and σ_6. These latter, however, are the basis of the following computation of P_{e_u}, the covariance matrix of unmonitored execution errors. P_{e_u} is computed for the purpose of updating P in NAVM following a midcourse correction. We will define P_{e_u} as the following conditional expectation:

$$P_{e_u} = \mathcal{E}\left\{e_u \, e_u^T \middle| \Delta v\right\} \tag{3}$$

Then, with the aid of the additional simplifying assumption that $e_5 \frac{\Delta v}{|\Delta v|}$ is spherically, Gaussian (0, σ_5) distributed, (3) can be expanded and reduced to

$$P_{e_u} = \sigma_p^2\left[|\Delta v|^2 I - \Delta v \Delta v^T\right] + \sigma_4^2 \Delta v \, \Delta v^T + \sigma_5^2 \frac{\Delta v \Delta v^T}{|\Delta v|^2} \tag{4}$$

Similarly,

$$P_{e_m} = \sigma_6^2 \frac{\Delta v \Delta v^T}{|\Delta v|^2} \tag{5}$$

(The above calculations of P_{e_u} and P_{e_m} rightfully belong in NAVM, and should be moved to that module in future versions of this program.)

Finally, in this model the actual velocity change experienced is given by

$$\Delta v' = \Delta v + e_u + e_m \tag{6}$$

	Variable	Program Symbol	Description	Source
Internal Input				
	$\Delta v(3x1)$	DELTV(3,1)	Midcourse Correction Command	GUIDM
	$\lvert\Delta v\rvert$	DVMAG	Magnitude of Δv	GUIDM
External Input				
	$e(6x1)$	E(6)	Vector of errors $e_1, e_2, \ldots, e_6$	
	σ_p	SIGMAP	Assumed standard deviations of errors $e_1, \ldots, e_6$	
	σ_4	SIGMA4		
	σ_5	SIGMA5		
	σ_6	SIGMA6		
Output				
	$e_u(3x1)$	EU(3,1)	Unmonitored execution error	
	$e_m(3x1)$	EM(3,1)	Monitored execution error	
	$\Delta v'(3x1)$	DELTVP(3,1)	Total velocity change	
	$P_{e_u}(3x3)$	PEU(3,3)	Covariance matrix of e_u	
	$P_{e_m}(3x3)$	PEM(3,3)	" " " e_m	

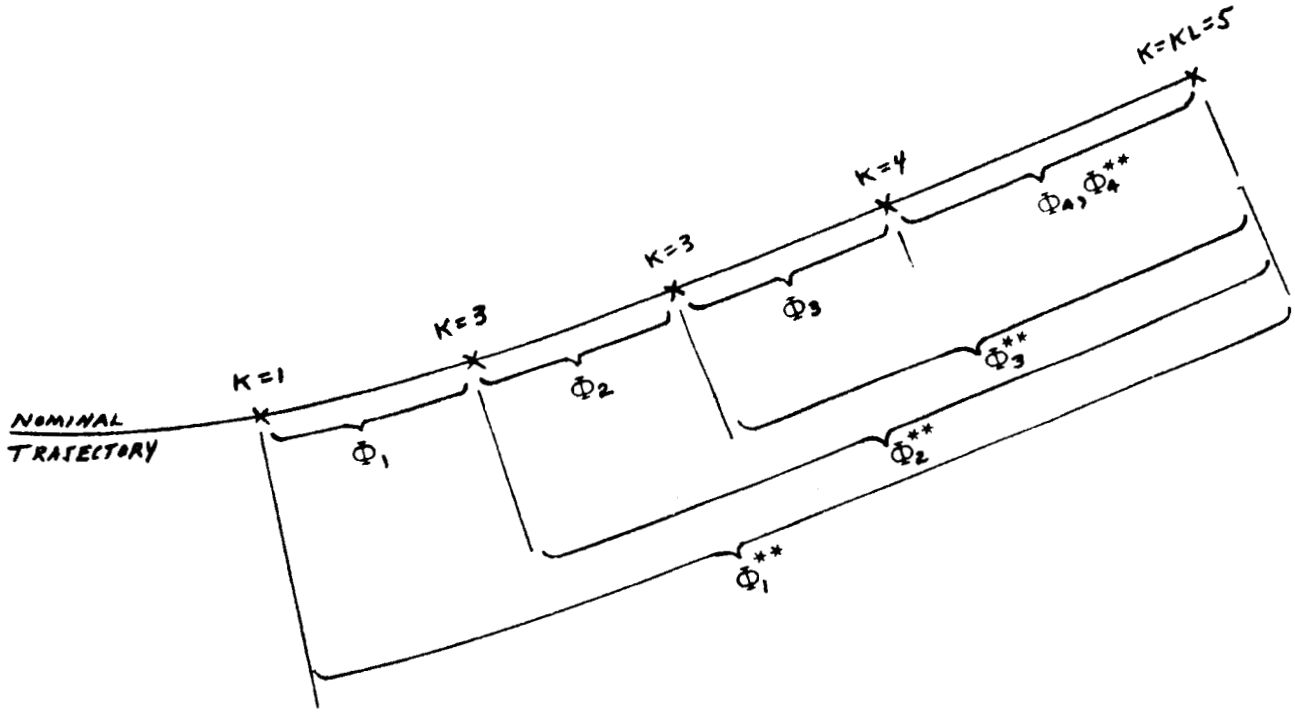
D. Auxiliary Group of Modules

In the tutorial program this group contains only the Policy Module POLM and Information Module INFM. No Auxiliary Module AUXM (which would perform special processing of input and output data for Monte Carlo and other special types of analyses) was needed to satisfy the design goals of the tutorial program.

1. POLM Model 1

The basic function of the Policy Module is to compute matrices required for the generation of midcourse velocity corrections in accordance with some guidance policy governed by model choice. Model 1 is based on a Fixed Time-of-Arrival guidance constraint, and can generate guidance matrices for an arbitrary number of prescribed (on the basis of time) midcourse correction points (Guidance Events) along the nominal trajectory. It does this during a "Guidance Preset Pass" controlled by the flag GPRF. At each Guidance Event, designated by IGEV(K), the state Transition matrix measured from IGEV(K-1) is available in PSIM and stored as $\Phi_{k-1}(6x6)$. It is necessary during this Guidance Preset Pass to use a very large Δt_N such that a Navigation Epoch is never encountered except when automatically designated at a Guidance Event (because the state transition matrix is reset in PSIM to the identity matrix at all Navigation Epochs). When the final event, representing some sort of target encounter, is reached the collection of Φ matrices is processed so as to yield state transition matrices from each Guidance Event to target encounter. These are shown schematically in the following example for a sequence

of four such midcourse correction points. The index K is used throughout the program to designate the Guidance Event, and KL denotes the final or target value of K. Computation of the Φ^{**} matrices progresses backwards, and proceeds as shown below.



$$\Phi_4^{**} = \Phi_4$$

$$\Phi_3^{**} = \Phi_4^{**} \Phi_3$$

$$\Phi_2^{**} = \Phi_3^{**} \Phi_2$$

$$\Phi_1^{**} = \Phi_2^{**} \Phi_1$$

Now the fixed Time-of-Arrival Guidance Matrix G_k is computed for each Guidance Event IGEV(K). The first step is the partitioning of Φ_k^* as $\begin{bmatrix} \varphi_1 & \varphi_2 \\ \varphi_3 & \varphi_4 \end{bmatrix}_k$, where each submatrix is (3x3).

Then G_k is given by the expression

$$G_k = -\left[\varphi_2^{-1}\varphi_1 \;\vdots\; I(3x3)\right]_k$$

These matrices are stored for use by the Guidance Module GUIDM during subsequent runs. (The indices on the above submatrices φ_1 and φ_2 were inadvertently reversed when programming this model.) See the Part III Addendum for a brief derivation of the above guidance law.

	Variable	Program Symbol	Description	Source
Internal Input				
	$\Psi'(6x6)$	PSIP(6,6)	State transition matrix from the previous Guidance Event to the current one.	PSIM
	k	K	Guidance Event index	EXM2
	k_ℓ	KL	Last Guidance Event	EXM2
External Input				
	None			
Output				
	$\Phi_k(6x6)$	PHI(K,6,6)	State transition matrix from Guidance Events K-1 to K.	

$\Phi_k^{**}(6x6)$	PHISS(K,6,6)	State transition matrix from Guidance Events K to KL.
$G_k(6x6)$	G(K,6,6)	Fixed Time-of-Arrival guidance matrix for Kth Guidance Event.

2. INFM Model 1

The Information Module is self-explanatory if one follows the block diagram of Appendix II. Essentially, on the basis of certain flag settings, specific blocks of data are printed at each entry. At a Guidance Event two complete sets of data are printed out, the second reflecting changes in state and updating of the Navigation data following a midcourse correction. This module is obviously subject to much change during future evolution of the program.

E. Subroutines and Functions

Several of these used in the program, but not having module status, are briefly described below.

1. DATIN

This is a data input subroutine performing much of the initialization function that will be handled by EXM2 in the ultimate program. It is designed specifically for the current set of models.

2. MATIN

This is a matrix inversion subroutine used throughout the program.

3. RNUM

This is a random number generating function that selects at random a number from a normal distribution of unity standard deviation and zero mean.

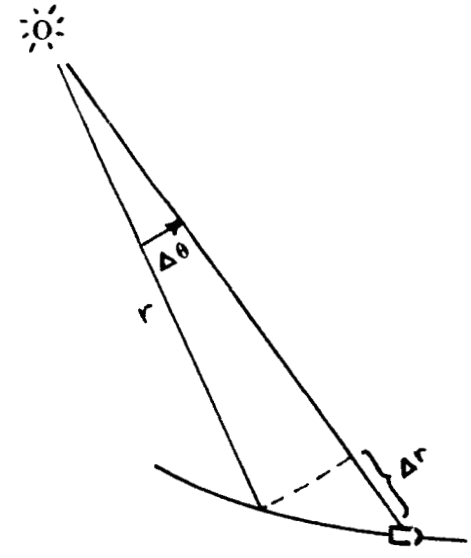
ADDENDUM TO PART III

1. Derivation of Equations for NTM, Model 1

The planar equations of motion in a central force field are used in the form

$$\ddot{r} + \frac{\mu}{r^2} - \frac{h^2}{r^3} = 0 \tag{1}$$

$$r^2\dot{\theta} = h \tag{2}$$



An approximate difference equation representation is obtained as follows.

Over the interval Δt, θ advances by $\Delta\theta$, r by Δr, and $\dot{r}$ by $\Delta\dot{r}$. From (1) and (2),

$$\Delta\dot{r} \approx \left(\overline{-\frac{\mu}{r^2} + \frac{h^2}{r^3}}\right)\Delta t$$

$$\Delta\theta \approx \left(\overline{\frac{h}{r^2}}\right)\Delta t$$

where the bars indicate mean values taken over the interval Δt. It is readily shown that for $\Delta r << r$ these expressions can be written

$$\Delta\dot{r} \approx \left[-\frac{\mu}{r^2}\left(1 - \frac{\Delta r}{r}\right) + \frac{h^2}{r^3}\left(1 - \frac{3}{2}\frac{\Delta r}{r}\right)\right]\Delta t \tag{3}$$

$$\Delta\theta \approx \frac{h}{r^2}\left(1 - \frac{\Delta r}{r}\right)\Delta t \tag{4}$$

The equation for Δr is simply

$$\Delta r \approx \left(\dot{r} + \frac{\Delta\dot{r}}{2}\right)\Delta t \quad . \tag{5}$$

Solving (3) and (5) simultaneously for $\Delta\dot{r}$ and Δr

$$\Delta r = \frac{1}{D_3}\left[\dot{r}\Delta t + \left(\frac{h^2}{r^3} - \frac{\mu}{r^2}\right)\frac{\Delta t^2}{2}\right] \tag{6}$$

$$\Delta\dot{r} = \frac{1}{D_3}\left[\left(\frac{h^2}{r^3} - \frac{\mu}{r^2}\right)\Delta t + \left(\frac{\mu}{r^3} - \frac{3}{2}\frac{h^2}{r^4}\right)\dot{r}\Delta t^2\right] \tag{7}$$

where $$D_3 = 1 - \left(\frac{\mu}{r^3} - \frac{3}{2}\frac{h^2}{r^4}\right)\frac{\Delta t^2}{2} \tag{8}$$

Equation (4) for $\Delta\theta$ is used without change. Note that (5) could have similarly been used to obtain Δr after using (6) and (7) for $\Delta\dot{r}$, but this simplification was overlooked.

In Model 1, r, θ, and $\dot{r}$ are updated by means of (6), (4), and (7). Following this, $\dot{\theta}$ is updated by means of (2). Angular momentum h is computed during initialization from the initial conditions on r and $\dot{\theta}$.

Model 1 also computes the central angle swept by the earth assuming a circular orbit. Inclination of the plane of the space vehicle's orbit with respect to the ecliptic must be accounted for by means of a transformation not identified in this particular model.

2. Derivation of Equations for PSIM Model 1

The variational forms of the planar central force field equations

$$\ddot{r} + \frac{\mu}{r^2} - r\dot{\theta}^2 = 0$$

$$\frac{d}{dt}(r^2\dot{\theta}) = 0$$

can be written

$$\frac{d}{dt}(\delta\dot{r}) = \delta\ddot{r} = \left(\frac{2\mu}{r^3} + \dot{\theta}^2\right)\delta r + 2r\dot{\theta}\,\delta\dot{\theta} - \frac{1}{r^2}\,\delta\mu$$

$$\frac{d}{dt}(\delta\dot{\theta}) = \delta\ddot{\theta} = \frac{2\dot{r}\dot{\theta}}{r^2}\,\delta r - \frac{2\dot{\theta}}{r}\delta\dot{r} - \frac{2\dot{r}}{r}\,\delta\dot{\theta} \quad .$$

Over the interval $\Delta t = t_{i+1} - t_i$

$$\Delta\delta\dot{r} \approx \left\{ \left(\overline{\frac{2\mu}{r^3} + \dot{\theta}^2}\right)\left(\delta r_i + \frac{\Delta\delta r}{2}\right) + \overline{2r\dot{\theta}}\left(\delta\dot{\theta}_i + \frac{\Delta\delta\dot{\theta}}{2}\right) - \overline{\frac{1}{r^2}}\,\delta\mu \right\}\Delta t$$

$$\Delta\delta\theta \approx \left\{ \overline{\frac{2\dot{r}\dot{\theta}}{r^2}}\left(\delta r_i + \frac{\Delta\delta r}{2}\right) - \overline{\frac{2\dot{\theta}}{r}}\left(\delta\dot{r}_i + \frac{\Delta\delta\dot{r}}{2}\right) - \overline{\frac{2\dot{r}}{r}}\left(\delta\dot{\theta}_i + \frac{\Delta\delta\dot{\theta}}{2}\right)\right\}\Delta t$$

$$\Delta\delta r \approx \left(\delta\dot{r}_i + \frac{\Delta\delta\dot{r}}{2}\right)\Delta t$$

$$\Delta\delta\theta \approx \left(\delta\dot{\theta}_i + \frac{\Delta\delta\dot{\theta}}{2}\right)\Delta t$$

where the bars indicate mean values over Δt of the underlying expressions. This set of 4 difference equations is properly rearranged and solved simultaneously for $\Delta\delta r$, $\Delta\delta\theta$, $\Delta\delta\dot{r}$, and $\Delta\delta\dot{\theta}$ in terms of the initial values δr_i, $\delta\theta_i$, $\delta\dot{r}_i$, $\delta\dot{\theta}_i$, and $\delta\mu$.

Similarly the out-of-plane motion (nominally zero) is described by the perturbation equation

$$\frac{d}{dt}(\delta\dot{N}) = \delta\ddot{N} = -\frac{\mu}{r^3}\,\delta N \quad ,$$

which leads to the approximate difference equations

$$\Delta\delta N \approx -\overline{\frac{\mu}{r^3}}\left(\delta N_i + \frac{\Delta\delta N}{2}\right)\Delta t$$

$$\Delta\delta N \approx \left(\delta\dot{N}_i + \frac{\Delta\delta\dot{N}}{2}\right)\Delta t$$

These are solved simultaneously for $\Delta\delta N$ and $\Delta\delta\dot{N}$ in terms of δN_i and $\delta\dot{N}_i$.

The combined results are, in matrix form,

$$\begin{bmatrix} \Delta\delta r \\ \Delta\delta\theta \\ \Delta\delta N \\ \Delta\delta\dot{r} \\ \Delta\delta\dot{\theta} \\ \Delta\delta\dot{N} \end{bmatrix} = \begin{bmatrix} A_1\frac{\Delta t}{2} & 0 & 0 & \left(\frac{A_2}{2}+1\right)\Delta t & A_3\frac{\Delta t}{2}\times 10^{-3} & 0 \\ B_1\frac{\Delta t}{2}\times 10^{3} & 0 & 0 & B_2\frac{\Delta t}{2}\times 10^{3} & \left(\frac{B_3}{2}+1\right)\Delta t & 0 \\ 0 & 0 & C_1 & 0 & 0 & C_2 \\ A_1 & 0 & 0 & A_2 & A_3\times 10^{-3} & 0 \\ B_1\times 10^{3} & 0 & 0 & B_2\times 10^{3} & B_3 & 0 \\ 0 & 0 & C_3 & 0 & 0 & C_1 \end{bmatrix} \begin{bmatrix} \delta r_i \\ \delta\theta_i \\ \delta N_i \\ \delta\dot{r}_i \\ \delta\dot{\theta}_i \\ \delta\dot{N}_i \end{bmatrix}$$

$$+ \begin{bmatrix} A_4\frac{\Delta t}{2} \\ B_4\frac{\Delta t}{2}\times 10^{3} \\ 0 \\ A_4 \\ B_4\times 10^{3} \\ 0 \end{bmatrix} \delta\mu = A\,\vec{\delta r_i} + \Gamma\delta\mu \quad . \qquad (1)$$

The elements of A and Γ are evaluated by means of the expressions

$$a_1 = \frac{2\mu}{r^3} + \dot{\theta}^2$$

$$a_2 = 2r\dot{\theta}$$

$$a_3 = \frac{1}{r^2}$$

$$b_1 = -\frac{2\dot{r}\dot{\theta}}{r^2}$$

$$b_2 = \frac{2\dot{\theta}}{r}$$

$$b_3 = \frac{2\dot{r}}{r}$$

$$D_1 = -\left[1 + \frac{b_3}{2}\Delta t + (b_2 a_2 - a_1)\frac{\Delta t^2}{4} + (b_1 a_2 - a_1 b_3)\frac{\Delta t^3}{8}\right] \qquad (2)$$

$$A_1 = \frac{1}{D_1}\left[-A_1\Delta t + (a_2 b_1 - a_1 b_3)\frac{\Delta t^2}{2}\right]$$

$$A_2 = \frac{1}{D_1}\left[(a_2 b_2 - a_1)\frac{\Delta t^2}{2} + (a_2 b_1 - a_1 b_3)\frac{\Delta t^3}{4}\right]$$

$$A_3 = -\frac{A_2 \Delta t}{D_1}$$

$$A_4 = \frac{1}{D_1}\left[a_3\Delta t + a_3 b_3 \frac{\Delta t^2}{2}\right]$$

$$B_1 = \frac{1}{D_1}\left[b_1\Delta t + a_1 b_2 \frac{\Delta t^2}{2}\right]$$

$$B_2 = \frac{1}{D_1}\left[b_2\Delta t + b_1 \frac{\Delta t^2}{2}\right]$$

$$B_3 = \frac{1}{D_1}\left[b_3\Delta t + a_2 b_2 \frac{\Delta t^2}{2} + (a_2 b_1 - a_1 b_3)\frac{\Delta t^3}{4}\right]$$

$$B_4 = \frac{1}{D_1}\left[- a_3 b_2 \frac{\Delta t^2}{2} - a_3 b_1 \frac{\Delta t^3}{4}\right]$$

$$D_2 = 1 + \frac{\mu}{r^3} \frac{\Delta t^2}{4}$$

$$C_1 = - \frac{\mu}{r^3} \frac{\Delta t^2}{2D_2}$$

$$C_2 = \frac{\Delta t}{D_2}$$

$$C_3 = - \frac{\mu}{r^3} \frac{\Delta t}{D_2}$$

(2) Continued

The desired state transition over the interval Δt is seen to be

$$\Psi \overset{\text{def}}{=} \Psi_{t_{i+1}, t_1} = A + I^{(6x6)} \tag{3}$$

The scheme for updating the cumulative matrices (measured from some reference point t_o*) is

$$\Psi'_{new} = \Psi \Psi'_{old} \tag{4}$$

$$\Gamma'_{new} = \Psi \Gamma'_{old} + \Gamma \tag{5}$$

where $$\Gamma \overset{\text{def}}{=} \Gamma_{i+1,1} \tag{6}$$

Noise Input

A white noise acceleration disturbance is computed at each computation cycle by means of the Function RNUM. This samples from a trivariate normal distribution defined by the input of SIGMQ(I),

---

* In present usage t_o is taken as the previous navigation epoch.

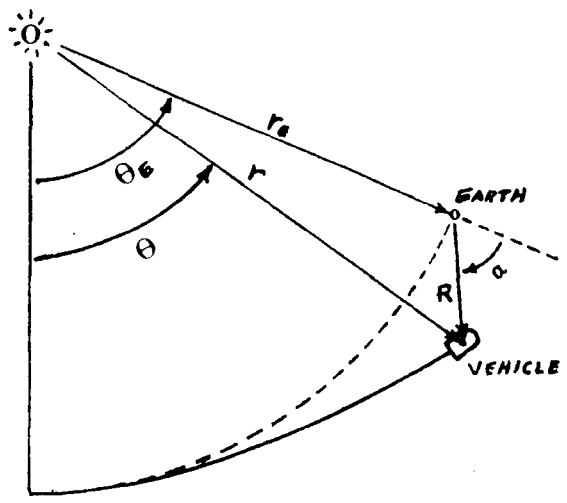
$i = 1, 2, 3$, yielding the (3x1) q vector. The cumulative noise vector will be a (6x1) vector denoted by q'. The updating equation in PSIM Model 1 is

$$q'_{new} = \Psi q'_{old} + \begin{bmatrix} 0 \\ \hline q \end{bmatrix} \Delta t_D \quad . \tag{7}$$

3. Derivation of Equations for HM, Model 1

Assumptions:

1) The vehicle trajectory lies in the ecliptic plane.
2) A tracking station on earth always looks directly overhead at the vehicle.
3) $-\pi \leq \theta_E - \theta \leq \pi$



Nominal Measurements

The nominal range measurement is given by the Law of Cosines:

$$R = \left[r^2 + r_E^2 - 2\, r\, r_E \cos(\theta_E - \theta)\right]^{1/2} \tag{1}$$

This is differentiated to give the range rate:

$$\dot{R} = \frac{1}{R}\left\{\left[r - r_E \cos(\theta_E - \theta)\right]\dot{r} + \left[r\, r_E \sin(\theta_E - \theta)\right]\left[\dot{\theta}_E - \dot{\theta}\right]\right\} \tag{2}$$

Other relations needed are, from the Sine and Cosine Laws:

$$\sin\alpha = \frac{r\sin(\theta_E - \theta)}{R} \tag{3}$$

$$\cos\alpha = \frac{r^2 - r_E^{\,2} - R^2}{2r_E R} \tag{4}$$

Perturbation Equations

Taking the first variations of (1), (2), and (3):

$$\delta R = E_1\,\delta r + E_2\,\delta\theta \tag{5}$$

$$\delta\dot{R} = F_1\,\delta r + F_2\,\delta\theta + E_1\,\delta\dot{r} + E_2\,\delta\dot{\theta} \tag{6}$$

$$\delta\alpha = G_1\,\delta r + G_2\,\delta\theta \tag{7}$$

where

$$E_1 = \frac{\partial R}{\partial r} = \frac{\partial\dot{R}}{\partial\dot{r}} = \frac{1}{R}\left[r - r_E\cos(\theta_E - \theta)\right] \tag{8}$$

$$E_2 = \frac{\partial R}{\partial\theta} = \frac{\partial\dot{R}}{\partial\dot{\theta}} = -\frac{1}{R}\left[r\,r_E\sin(\theta_E - \theta)\right] \tag{9}$$

$$F_1 = \frac{\partial\dot{R}}{\partial r} = \frac{1}{R}\left[-\dot{R}\,E_1 + \dot{r} + r_E\sin(\theta_E - \theta)(\dot{\theta}_E - \dot{\theta})\right] \tag{10}$$

$$F_2 = \frac{\partial\dot{R}}{\partial\theta} = \frac{1}{R}\left[-\dot{R}\,E_2 - \dot{r}\,r_E\sin(\theta_E - \theta) - r\,r_E\cos(\theta_E - \theta)(\dot{\theta}_E - \dot{\theta})\right] \tag{11}$$

The out-of-plane measurement β is given by the approximate expression $\beta = \frac{N}{R}$. (12)

Hence $$\delta\beta = \frac{\partial\beta}{\partial N}\,\delta N = \frac{1}{R}\,\delta N \tag{13}$$

All other first order partial derivatives are zero.

The desired observation matrix H is finally obtained by assembling the above partial derivatives in accordance with the expression

$$H = \frac{\partial(R, \dot{R}, \alpha, \beta)}{\partial(r, \theta, N, \dot{r}, \dot{\theta}, \dot{N})} ,$$

and is Equation (1) of the text.

<u>HM Initialization Model 1</u> (Companion to HM Model 1)

This model presets all elements of the observation matrix HM(I, J) to zero.

4. <u>POLM, Model 1, Derivation of Fixed Time-of-Arrival Guidance Matrix G_k.</u>

Let $\Phi^{**}_{f,o}(6x6) = \left[\begin{array}{c|c} \varphi_1 & \varphi_2 \\ \hline \varphi_3 & \varphi_4 \end{array}\right]$ be the state transition matrix that maps the state $x(6x1) = \left[\begin{array}{c} p \\ \hline v \end{array}\right]$ from the point t_o to some final time t_f, where p nd v denote (3x1) position and velocity vectors respectively, and the submatrices of Φ^{**} are (3x3). Thus

$$\left[\begin{array}{c} p_f \\ \hline v_f \end{array}\right] \quad \left[\begin{array}{c|c} \varphi_1 & \varphi_2 \\ \hline \varphi_3 & \varphi_4 \end{array}\right] \left[\begin{array}{c} p_o \\ \hline v_o \end{array}\right] .$$

The Fixed Time-of-Arrival guidance constraint requires identification of a Δv at t_o which, when added to v_o will cause P_f to be zero. Thus we must solve the expression

$$0 = \left[\varphi_1 \mid \varphi_2\right] \left[\begin{array}{c} p_o \\ \hline v_o + \Delta v \end{array}\right]$$

for Δv.

The result is

$$\Delta v = - \left[\varphi_2^{-1} \varphi_1 \mid I(3x3)\right] \left[\begin{array}{c} p_o \\ \hline v_o \end{array}\right] ,$$

where the coefficient of $\left[\begin{array}{c} p_o \\ \hline v_o \end{array}\right]$ is the desired guidance matrix corresponding to the point t_o.

5. Instructions for Use of Tutorial Program

It is assumed that the nominal trajectory has been designed elsewhere, so that initial conditions and other information pertaining thereto are available.

a. Run Designation

IRUNX is the total number of runs desired. Any run (or set of runs that are identical except for error data) must be preceded by a Guidance Preset run for computing Guidance matrices, unless no midcourse corrections are to be made. The preset run is to be counted in IRUNX, and requires input of IGPRE = 1. It is input as 0 on the remaining runs.

b. Event Designation

The first step is designation of all Events and their time of occurrence along the nominal trajectory. To each event is assigned a fixed point number in ascending order with time. This set of event sequence numbers forms the ESN array, and the corresponding times forms the TEV array. The event sequence numbers for all events which are to be guidance events (points of midcourse correction) are then assembled to form the GEV array. If no midcourse correction is to be made, a single GEV must be input, the final ESN. IESNX is input as the final ESN, and JEVL and KL as the total number of events and guidance events respectively. For examples of event designation and input, see Part IV.

c. Time Registers

DLTT, the reference value of Δt, is chosen to satisfy accuracy

requirements of NTM and PSIM (see those writeups). DELTN is the time between navigation epochs. On a Guidance Preset Run (see POIM writeup) this is input as infinity.

d. Print Control

IPT controls the frequency of printout on a navigation cycle basis. If input as 1, printout will occur at every navigation epoch, if 2 at every second one, etc. In addition, printout will automatically occur at all events.

e. Initial Conditions and Other Input

The current set of models computes in a polar coordinate system, and all angles were originally expressed in radians, with other units being at the discretion of user. However, to solve a last minute scaling problem, angular units in the perturbation relations were changed to milliradians. (This change must be observed for the input of x, $\hat{x}$, P, Q, SIGMQ, P_v, SIGMA, and B.) Angular elements of the time state, Earth state, the error vector E, and SIGMP, however, remain in radians.

f. Input Deck Assembly

The necessary input cards are listed below in proper order with the required Fortran format for each individual card.

Card Order	Variable Names	Fortran Format
1	IRUNX	I5
2	IGPRF	I5
3	IPT, IESNX, KL, JEVL	4I5
4	IESN	10I5
5	TEV	5E13.6
6	IGEV	10I5
7	DLTT, DELTN, TD, TN	4E13.6
8	THEDTE, THETAE, THETA, THEDT	4E13.6
9	MU, R, RDOT	3E13.6
10	RE	E13.6
11	X	6E13.6
12	RS, SIGMQ	4E13.6
13	NM, B	I5, 8X, 4E13.6
14	SIGMA	4E13.6
15	XH	6E13.6
16-21	P	6E13.6
22-27	QQ	6E13.6
28-31	PV	4E13.6
32	E	6E13.6
33	SIGMA4, SIGMA5, SIGMA6, SIGMAP	4E13.6

The first card, IRUNX, is required for only the initial case when a single run contains multiple cases. The input card set for the second and any following cases of a run begins with card number 2, IGPRF.

g. Printout of Input Data (from DATIN subroutine)

Preceding the labeled data printout under the control of INFM is an unlabeled block of data. This records all of the card input data for the run concerned except for that appearing in the normal printout for the first event. The unlabeled data is identified as follows:

Line	Item		
1	IPT	Print Interval	
2	DLTT	Reference value of dynamics time increment	
3	DELTN	Navigation time increment	
4	RE	Radius of Earth's orbit	
5	THEDTE	Angular velocity of Earth about the Sun	
6	MU	Gravitational constant of Sun	
7*	H	Unit angular momentum of nominal trajectory	
8	RS	Error in MU	
9-11	SIGMQ	SIGMQ(3)	(See PSIM)
12-15	B	B(4,1)	(See OBSM)
16-19	SIGMA	SIGMA(4)	(See OBSM)
20-25	QQ	Q(6,6)	(See NAVM)
26-29	PV	PV(4,4)	(See NAVM)
30-35	E	E(6)	(See CONTM)

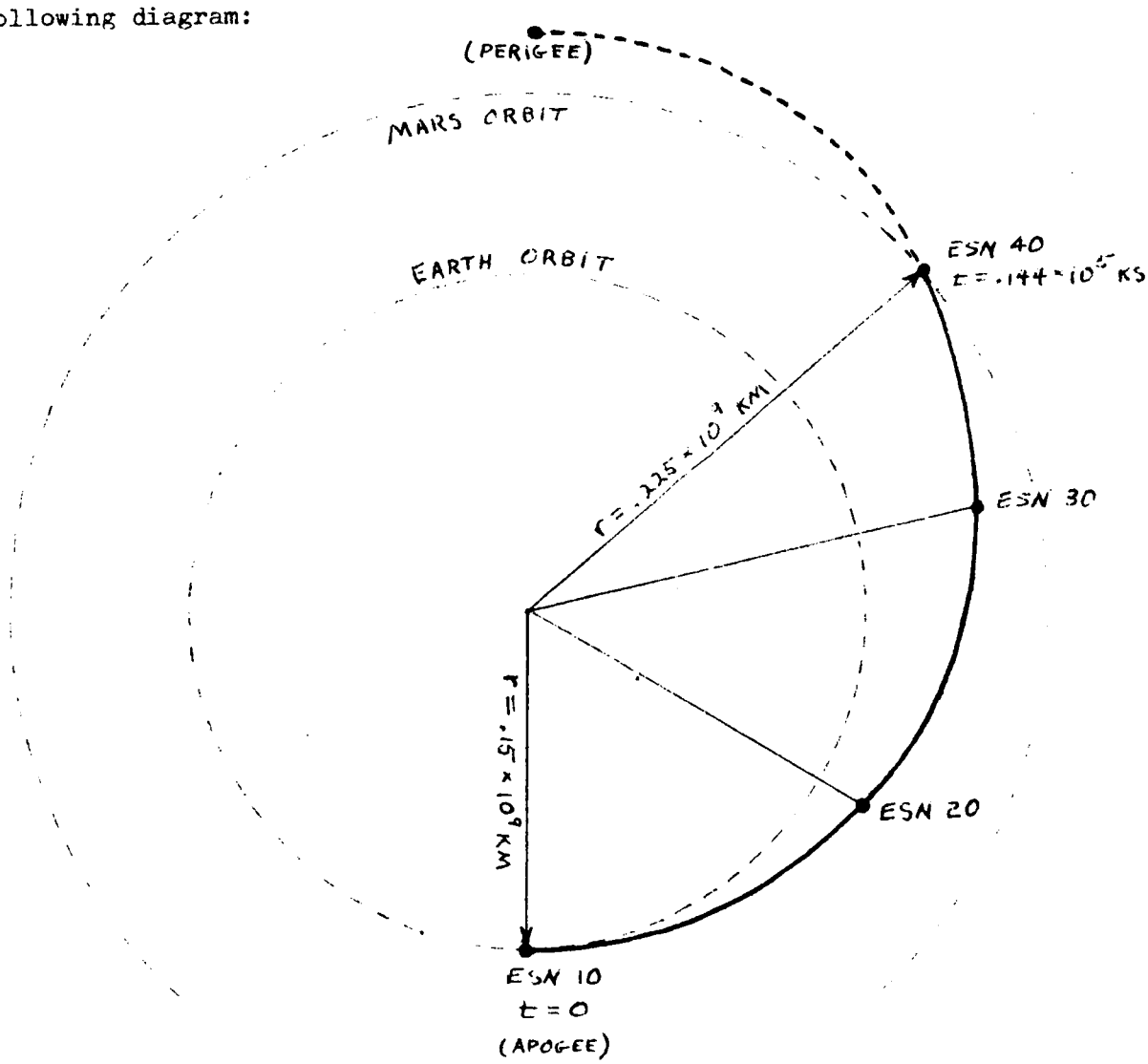
* The first such printout of H in a sequence of runs based on a single nominal trajectory will be in error; since H is not computed until after the present Write statement appears.

IV. NUMERICAL RESULTS

A. Problem Description

In order to establish the validity of results obtained from the tutorial program and to demonstrate its capabilities in different modes of operation, two separate groups of experiments were formulated. Each run within the experiments was based on all or part of a single nominal heliocentric Earth to Mars transfer orbit, which will now be described briefly.

The nominal trajectory was an elliptical arc as described in the following diagram:



This nominal trajectory was nearly tangent to the earth's orbit at time zero and intersected the Mars orbit about 167 days later after passing thru a central angle of about 130°. No significance is to be attached to this trajectory other than that it serves as a reference for program check-out and demonstration purposes. A transformation, unidentified herein, is implied which relates the orbital plane inclination and the polar co-ordinates employed to some inertial reference frame. Initial conditions and orbital parameters for the nominal trajectory are given below.

<u>Initial Conditions</u>

$t = 0$

$r = .15 \times 10^9$ km

$\theta = 0.$ rad

$\dot{r} = 0.$ km/ks

$\dot{\theta} = .2222 \times 10^{-3}$ rad/ks

<u>Orbital Parameters</u>

$r_a = .25 \times 10^9$ km

$r_p = .15 \times 10^9$

Eccentricity = .25

<u>Other Numerical Input</u>

$\mu = .133 \times 10^{18}$ km^3/ks^2 (gravitational constant of Sun)

$r_E = .148 \times 10^9$

$\dot{\theta}_E = .199 \times 10^{-3}$ rad/ks

Problem setup for the covariance propagation runs (Figures IV-1 to IV-9) was typical of missions involving no midcourse corrections. The listed initial conditions and required parameter values were input in the customary way. The ESN and corresponding TEV arrays were input as

ESN(1) = 10	TEV(1) = 0.
ESN(2) = 20	TEV(2) = _____ (varied as required)

A single Guidance Event was input

$$GEV(1) = 30 ,$$

with run termination controlled by ESNX = 20. (Note that defining GEV(1) as 20 would have given the same results.) Although propagation of the state and estimate thereof were not required, the present set of models does this automatically, so initial conditions for those variables and all error sources entering into the deterministic equations must be input. These can be easily input as zero. Since no midcourse corrections were made, a guidance preset run was not required and the Guidance Preset Flag GPRF was input as zero in all cases.

The simulation runs involved complete missions, including midcourse corrections being made at both events 20 and 30. Thus a guidance preset run was required for each set having given ESN, TEV, and GEV arrays in order to compute the required guidance matrices. Since a fixed time-of-arrival guidance scheme is modeled in POLM, the midcourse corrections attempted to derive position errors to zero at the final event, No. 40. Two subgroups of the simulation runs were formed, depending upon whether or not errors in the midcourse execution were permitted. For the first subgroup, midcourse execution error sources were simply input as zeros.

B. Numerical Experiments

In this section of the report, results will be presented of numerical experiments conducted by exercising the tutorial program. As was mentioned in the introduction, no special significance is attached to the actual numbers obtained from the experiments. However, the results are important for two reasons. First, the graphs accompanying this discussion are

indicative of the kinds of information that can be garnered from an interplanetary analysis program. Secondly, although the simple computational models assembled in the tutorial program do have a limited application, two-body analyses are sufficient for initial design studies and, more importantly, many of the comments about the quantitative nature of the results will still hold true when more complex computational models are inserted into the program. It should also be mentioned, before beginning the detailed discussion, that it is assumed that the reader is familiar with the analytic methods presented in Volume I of the current report. A familiarity with the companion document is necessary if any insight into the interplanetary problem is to be gained from surveying the results to follow.

The information to be presented in this section divides into two convenient sets. The first set, of which Figures IV-1 through IV-9 are a part, consists of experiments conducted only on the navigation and estimation processes of the program. No midcourse corrections are applied so that the entire focus is on different facets of the estimation procedure. The second set deals with a mission consisting of three separate navigation periods that are punctuated by two midcourse corrections at predetermined times. The midcourse guidance policy used is very simple -- not only have the correction times been arbitrarily determined, but also the simplest linear impulsive guidance scheme (fixed-time-of-arrival) has been employed. The first three graphs in the second set assume no error in the execution of the guidance correction, while the remainder of the Figures deal with the execution error problem.

Again it should be stressed that the numbers themselves obtained in these experiments are not of major importance. In fact, since general comments pertaining to methods and techniques are the meat of this discussion, actual numbers will be omitted whenever possible. Thus instead of saying, with respect to certain graphs, that "the 1σ accuracy value for the measuring devices in this experiment was" such and such, comments such as "highly accurate measurements were assumed" will be used. Comparisons will often be expressed numerically as ratios rather than defining the exact numbers related to the quantities being compared.

The first three graphs are concerned with determining the influence of observation frequency on propagated covariance matrices. A spacecraft in a heliocentric two-body orbit, in transit from Earth to Mars, undergoes a 3200 kilosecond coasting period during which time it is tracked and measured from Earth. Earth-based observations, reasonably accurate, are made of range, range rate, azimuth, and elevation. Covariance matrices reflecting the accuracy of the measuring devices and the initial accuracy of the 6x1 state vector of position and velocity are available. In addition, a specific nominal trajectory has been postulated and its associated state transition matrices have been calculated. If each observation vector recorded during the 3200 kilosecond coast period is processed using Kalman recursive filtering to obtain the state vector estimates, what is the accuracy of that estimate at the end of the coast and how does it depend on the number of observations?

This problem is subsumed into the class of problems dealing with optimum observation schedules. Generally, what is sought in an interplanetary

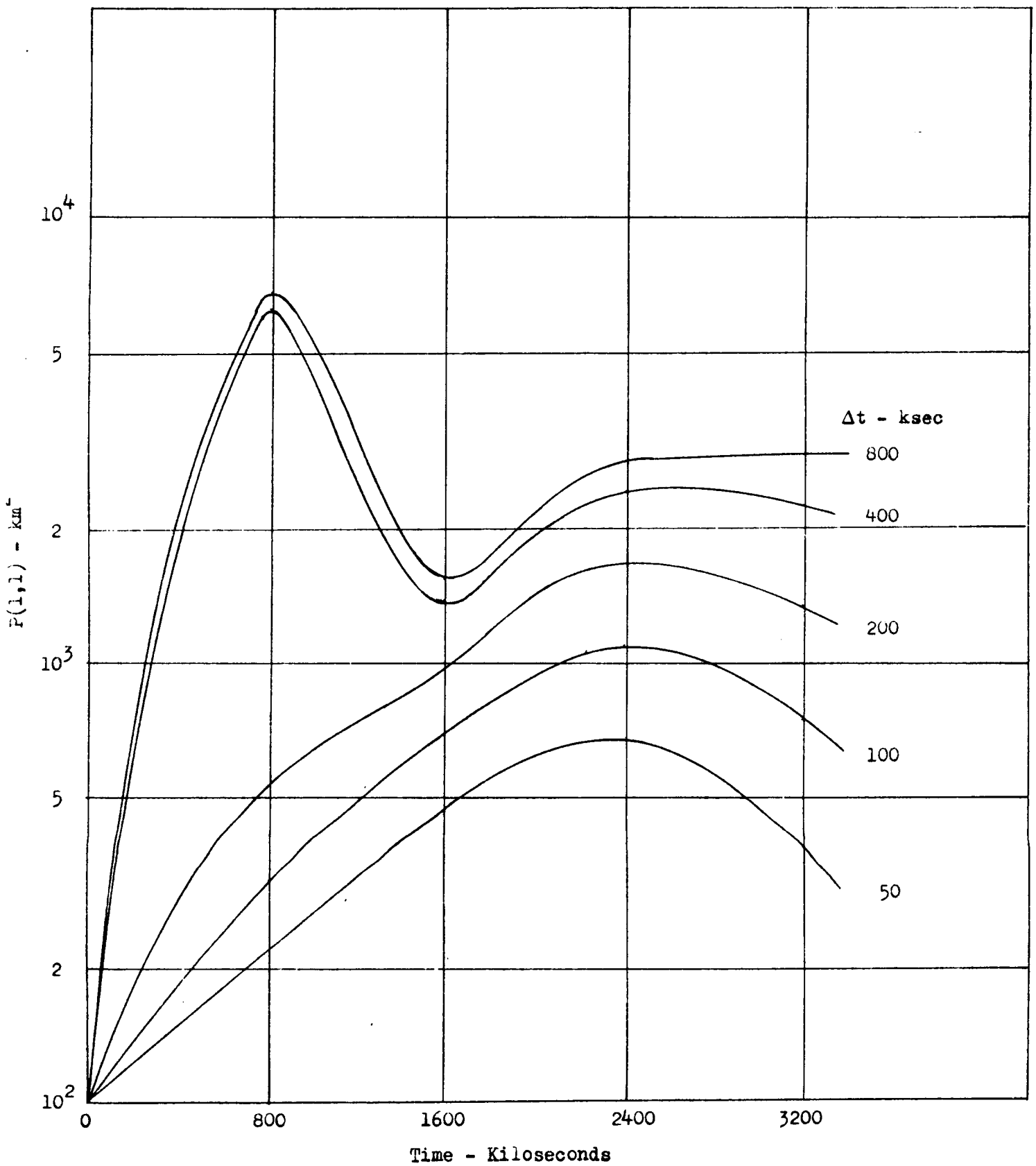

10^4
5
2
10^3
5
2
10^2
F(1,1) - km²
0
800
1600
2400
3200
Time - Kiloseconds
Δt - ksec
800
400
200
100
50

Fig. IV-1

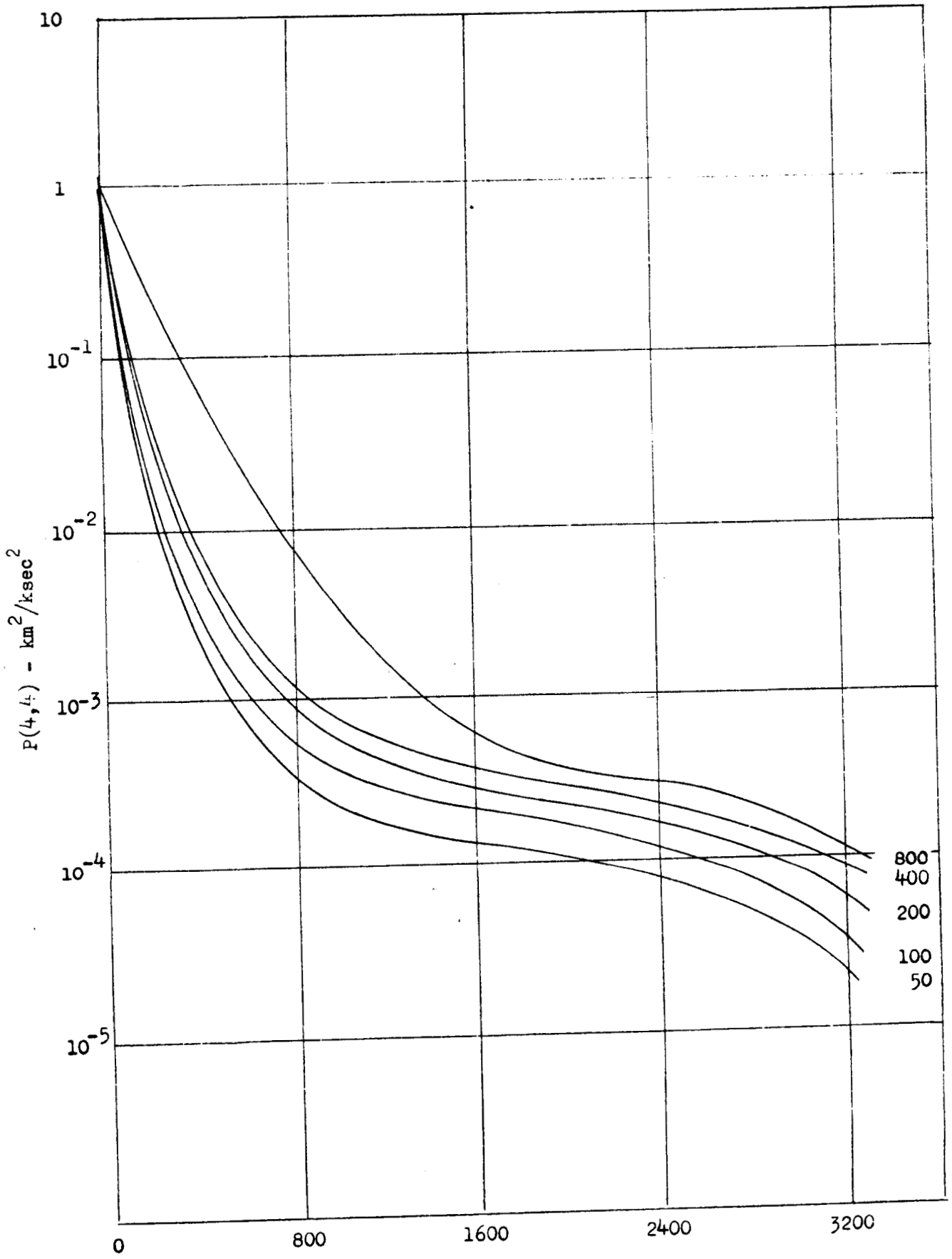

10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
$P(4,4)$ - $km^2/ksec^2$
0
800
1600
2400
3200
800
400
200
100
50

Fig. IV-2

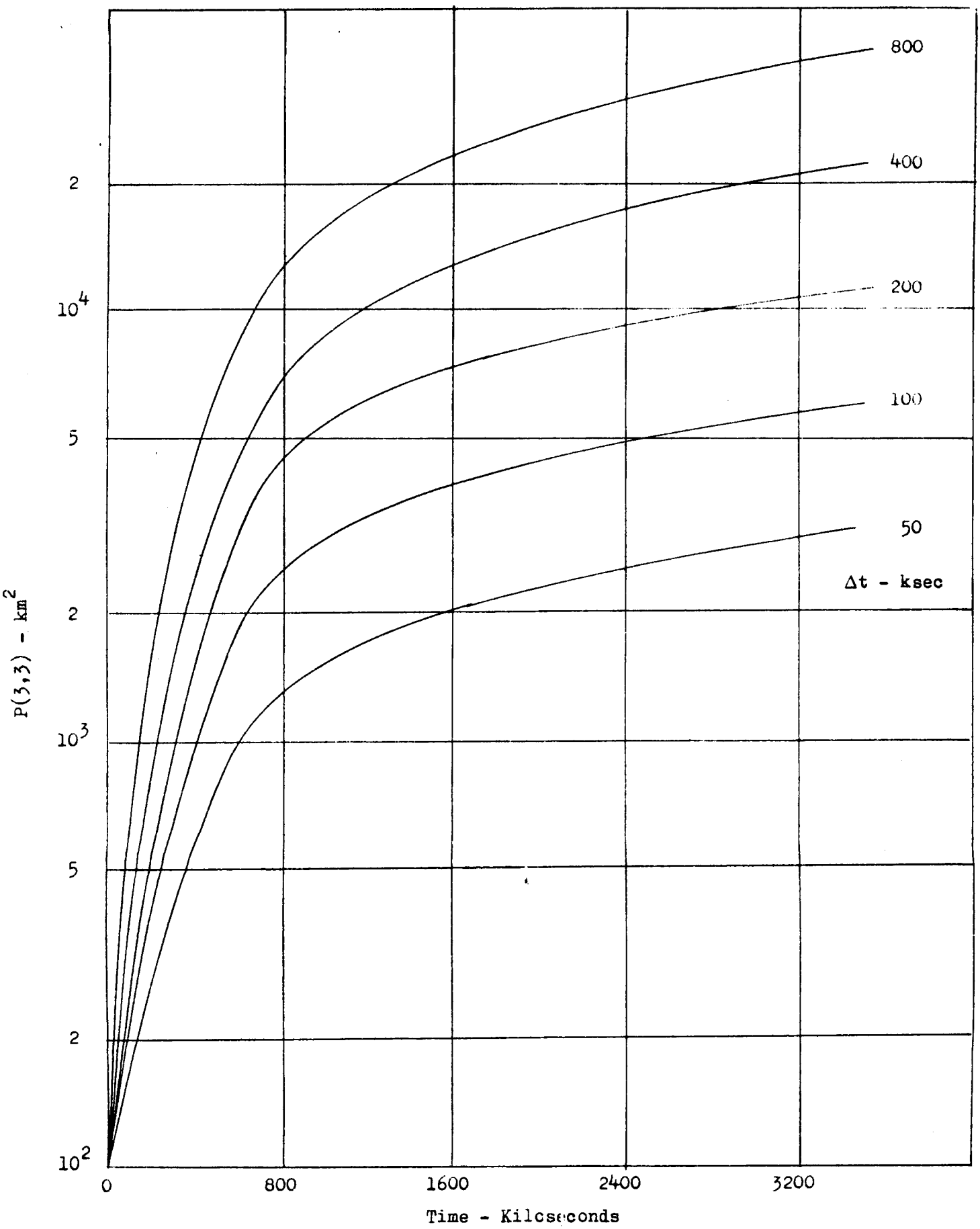

800
400
200
100
50
Δt - ksec
2
10^4
5
2
10^3
5
2
10^2
P(3,3) - km^2
0
800
1600
2400
3200
Time - Kilcseconds

Fig. IV-3

mission using recursive filtering to estimate the trajectory is an observation schedule that is frequent enough to keep perturbations from becoming unbounded but, at the same time, is not so frequent that it places unreasonable demands upon tracking facilities or violates the estimation theory assumption of uncorrelated random noise between separate observation vectors. From the experiments conducted with the tutorial program, it has become apparent that determining a "stable" observation schedule (that is, a schedule that probabilistically, prevents the difference between the estimated and actual state vectors from growing) is very much trajectory dependent. Thus one is led to the rather intriguing thought that one of the criteria for trajectory design for an interplanetary mission should be that the nominal trajectory has, as one of its properties, a rather broad class of "stable" observation schedules.

The sample problem whose results are shown in Figures IV-1, IV-2, and IV-3 illustrates the kind of propagated error differences that can be anticipated when observation schedules are changed. Figure IV-1 shows the propagation of a diagonal element (P_{11}) of the covariance matrix -- this diagonal element reflects uncertainties in the estimate of the range component of the state vector. In a sense, this component is the most "observable" of the position components, since all the elements of the measurement vector yield some information about it. The five continuous plots connect values of P_{11} computed <u>after</u> each observation has been processed. The number of observation vectors recorded during the coast varies from 4 to 64 on the five plots, depending upon the length of time Δt between observations.

The initial variance in the estimate of range was assumed to be 100 km^2, which corresponds to saying that the 1σ value was $\pm$ 10 km. If only four observations are taken during the coast, then the uncertainty (1σ value) in the range grows by almost a factor of ten and demonstrates marked unstable behavior. Due to the initial uncertainties assumed, four observations are not enough to keep the range uncertainty from growing. If observations are taken every 50 kiloseconds, however, the uncertainty in range is kept at more or less the same level. The remaining curves show the affect of choosing time intervals between observations that fall between the two extremes. Each separate plot represents a doubling of the number of observations and it should be noted that, due to the specific trajectory being analyzed, the reduction in P_{11} that results from doubling the observations is not necessarily uniform.

Figure IV-2 is a similar plot of covariance matrix element P_{44}, which corresponds to the variance in the estimate of range rate. All of the observation schedules are stable with respect to the uncertainty in range rate. One might legitimately ask why range rate is so much more "observable" than range for this particular problem. The answer reinforces the high degree of trajectory dependence implicit in determining "stable" observation schedules. For this particular nominal trajectory and the specified coast period, range rate perturbations about the nominal values grow very slowly. By contrast, the state transition matrices relating range perturbations about the nominal, for the same problem, have large elements. An uncertainty in range grows very fast as a result and must be observed more frequently. For another nominal trajectory -- for example,

a near circular orbit -- perturbations and hence uncertainties in range might demonstrate the same behavior as that shown for range rate in Figure IV-2.

The third graph illustrates still another phenomenon. Figure IV-3 plots the covariance element P_{33} associated with the uncertainty in the "out of plane" component of position. The plane, as was mentioned in Chapter III, is the plane of the nominal two-body trajectory. Due to the nature of the measurement vector assumed, this component of the state vector is the least "observable". And as demonstrated by Figure IV-3, not even 64 observations taken during the coast period can keep the uncertainty in the out of plane component from growing rapidly.

The purpose of these first three graphs is to demonstrate the role that observation frequency plays in the propagation of covariance matrices along a coasting arc. In addition, it is hoped that the high dependence of propagated uncertainties upon both the nominal trajectory being used and the kind of measurement data available has been sufficiently underscored.

The next two graphs, Figures IV-4 and IV-5, deal with the problem of initial covariance matrices. Persons learning about the theory and application of Kalman filtering are often troubled by the amount of assumably a priori knowledge required to implement the process. The point that is demonstrated by the figures is that if several observations are going to be processed, the affect of the initial covariance matrix becomes very small after a period provided that it has been chosen in a conservative way. The underlying problem used to demonstrate this fact is the same

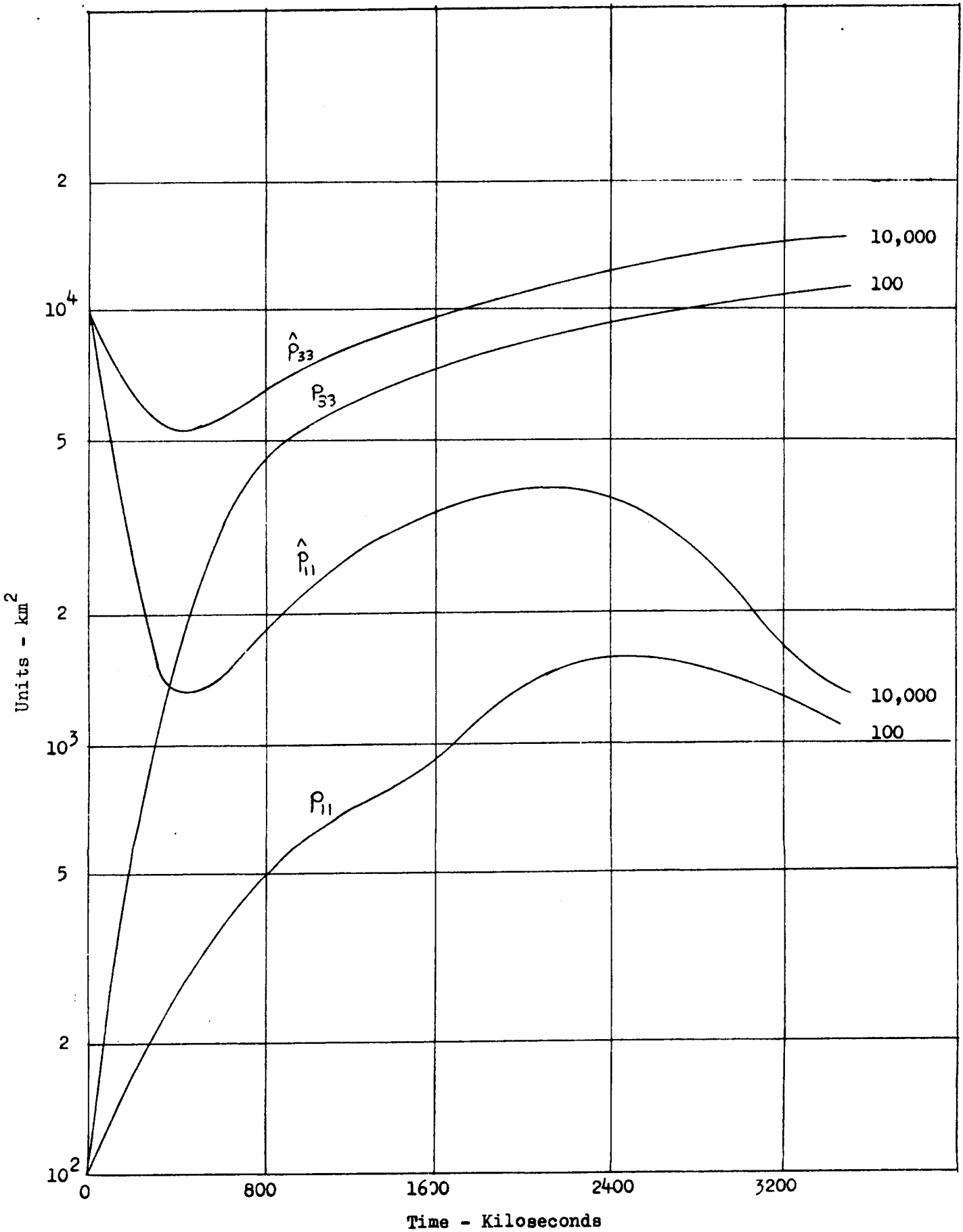

2
10^4
5
2
10^3
5
2
10^2
0
800
1600
2400
3200
Units - km^2
Time - Kiloseconds
$\hat{P}_{33}$
P_{33}
$\hat{P}_{11}$
P_{11}
10,000
100
10,000
100

Fig. IV-4

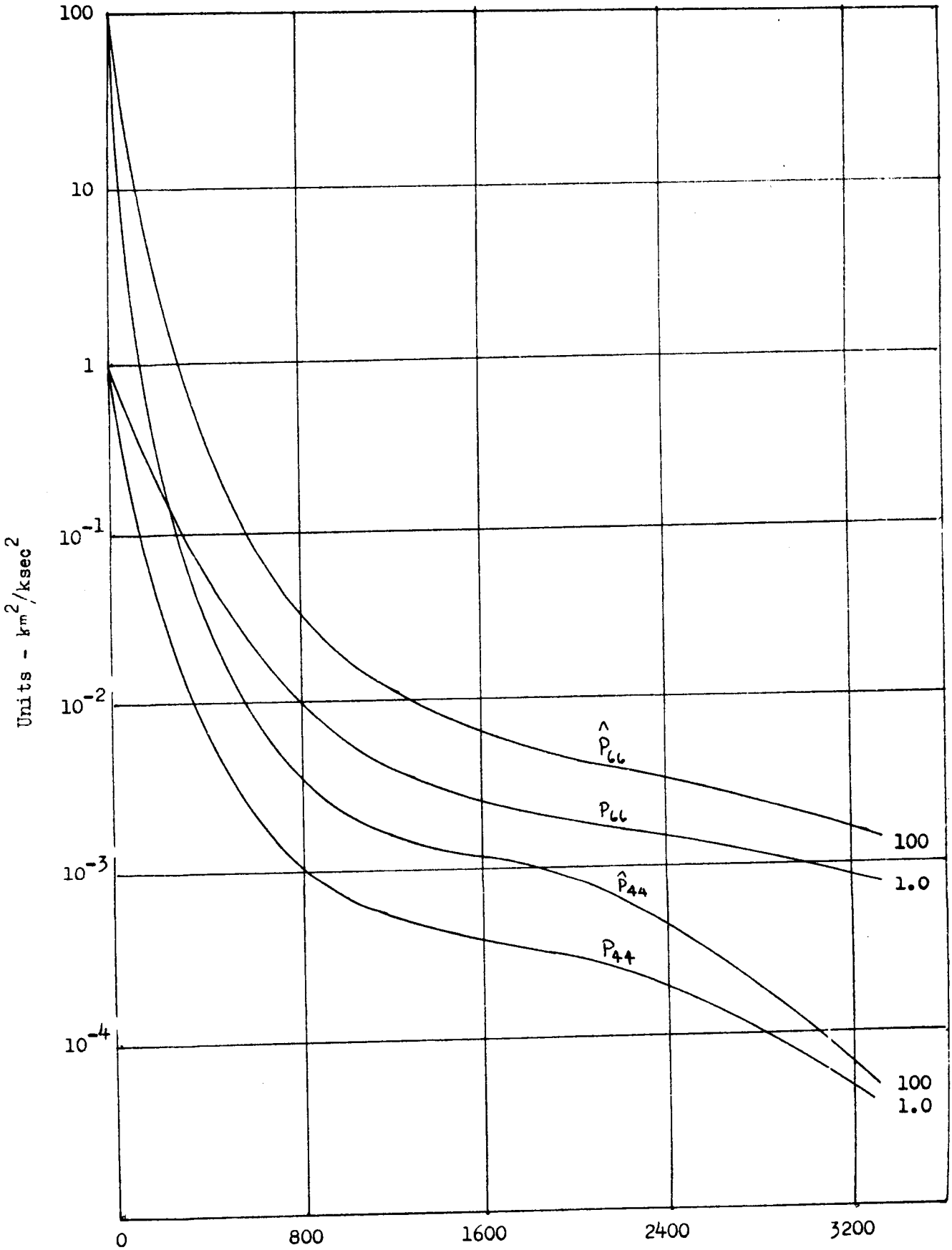

100
10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
Units - $km^2/ksec^2$
0
800
1600
2400
3200
$\hat{P}_{66}$
P_{66}
$\hat{P}_{44}$
P_{44}
100
1.0
100
1.0

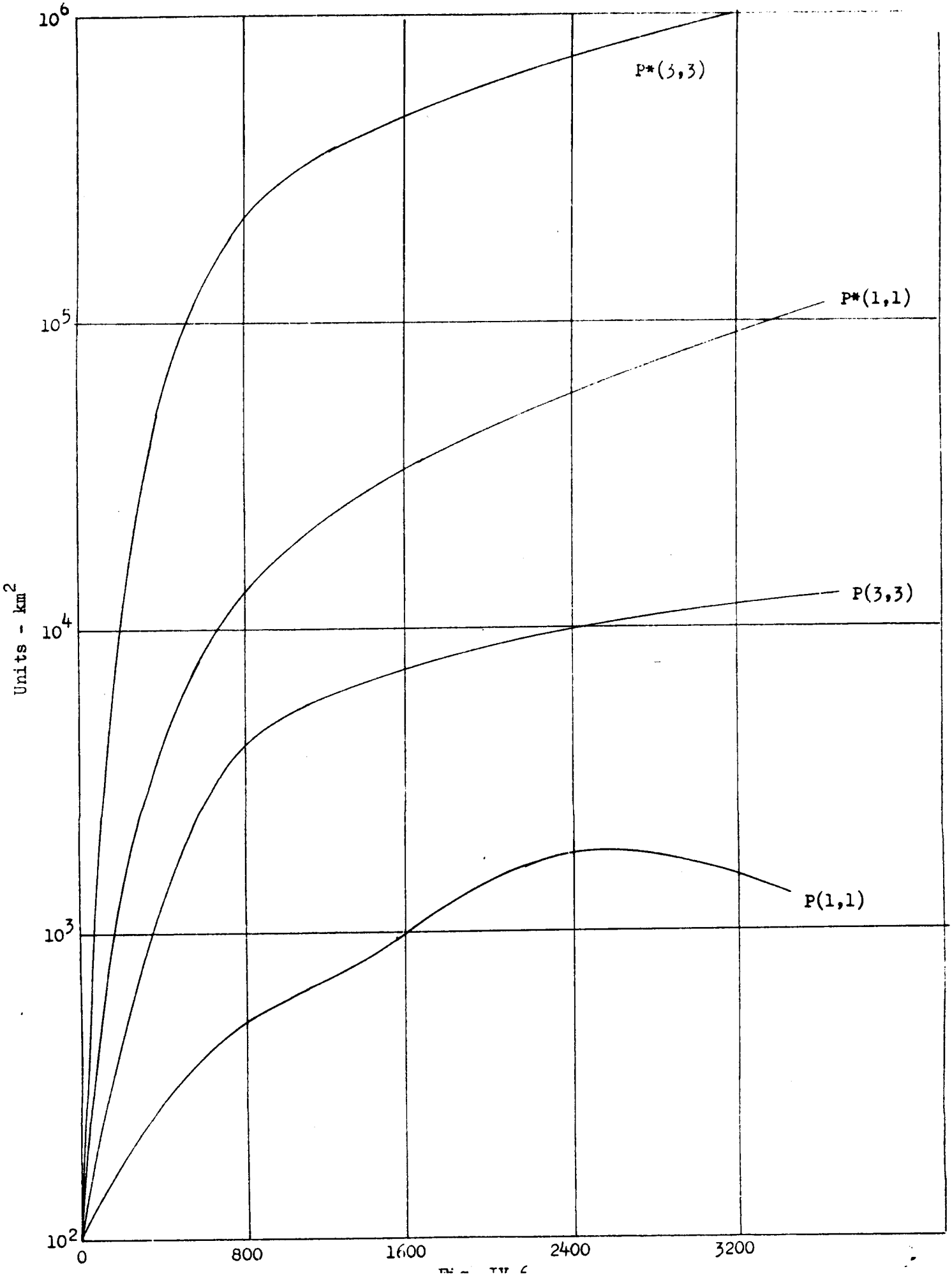
Fig. IV-5

heliocentric, 3200 kilosecond coasting arc used in the first graphs. The same covariance propagation, assuming the same sixteen measurement vectors with known uncertainties, was performed using one initial covariance matrix that was one hundred times as large as another. The elements of the large covariance have been designated in the plots of Figures IV-4 and IV-5 with a circumflex ("hat") symbol. P_{11}, P_{33}, and P_{44} are, as before, the covariance elements associated with range, out of plane position, and range rate; the diagonal covariance element P_{66} is associated with out of plane velocity. After the sixteen measurements have been processed, the ratios $P_{kk}(P_{kk})^{-1}$ have dropped from one hundred to between one and two. The ratios $P_{33}(P_{33})^{-1}$ and $P_{66}(P_{66})^{-1}$ are higher than the other two due to the relative lack of information about the out of plane characteristics being measured by the observations.

Two comments are warranted with respect to initial covariances. The work of Nishimura shows that $P_{kk}(P_{kk})^{-1}$ will never go below unity, which implies that if the initial covariance matrix has been chosen conservatively, then all calculated covariances based upon the initial one will also be conservative. Also, it should be pointed out that choosing an initial covariance matrix more optimistic than actual can lead to serious problems. The gist of the concern is that an overly-optimistic initial covariance matrix propagates rapidly, when measurement accuracies are high, into unrealistic covariances associated with subsequent state vector estimates. Eventually this can lead to filter divergence, a phenomenon discussed in more detail in connection with a later graph.

Of course the most important single factor determining covariance matrix and uncertainty propagation through a recursive filtering process is the accuracy of the measurements involved. This fact is amply demonstrated by Figures IV-6 and IV-7. Again, a 3200 kilosecond coasting arc along a heliocentric transit from Earth to Mars was used for the underlying problem. Thirty-two observations were taken during the coast and these measurement vectors were processed using recursive filtering. The graphs of P_{11}, P_{33}, P_{44}, and P_{66} show the propagation of <u>updated</u> (after each observation) uncertainties in the four elements of the state vector defined earlier. These plots are to be compared with P^*_{11}, P^*_{33}, P^*_{44}, and P^*_{66}, which were obtained from a separate covariance propagation that assumed measurement accuracies an order of magnitude less than the accuracies assumed in propagating the elements P_{kk}.

Looking more closely at the specific plots for P^*_{11} and P_{11} in Figure IV-6, one can see that despite the fact that their initial uncertainties in range were equal, at the end of the coast the variance in range due to the less accurate measurements is about fifty times greater than the variance in range resulting from using the more accurate measurements. In addition, the more accurate measurements appear to have stabilized the growth in range uncertainty while P^*_{11} is still exhibiting unstable behavior. With respect to the accuracy of the out of plane position estimate (P_{33} vs. P^*_{33}), the degradation in the filtering process due to less accurate measurements is more pronounced. After the coasting arc, despite the fact that their initial values were equal, the two covariance elements are separated by roughly two orders of magnitude. This reflects 1σ uncertainties in


10^6
10^5
10^4
10^3
10^2
Units - km^2
P*(3,3)
P*(1,1)
P(3,3)
P(1,1)
0
800
1600
2400
3200

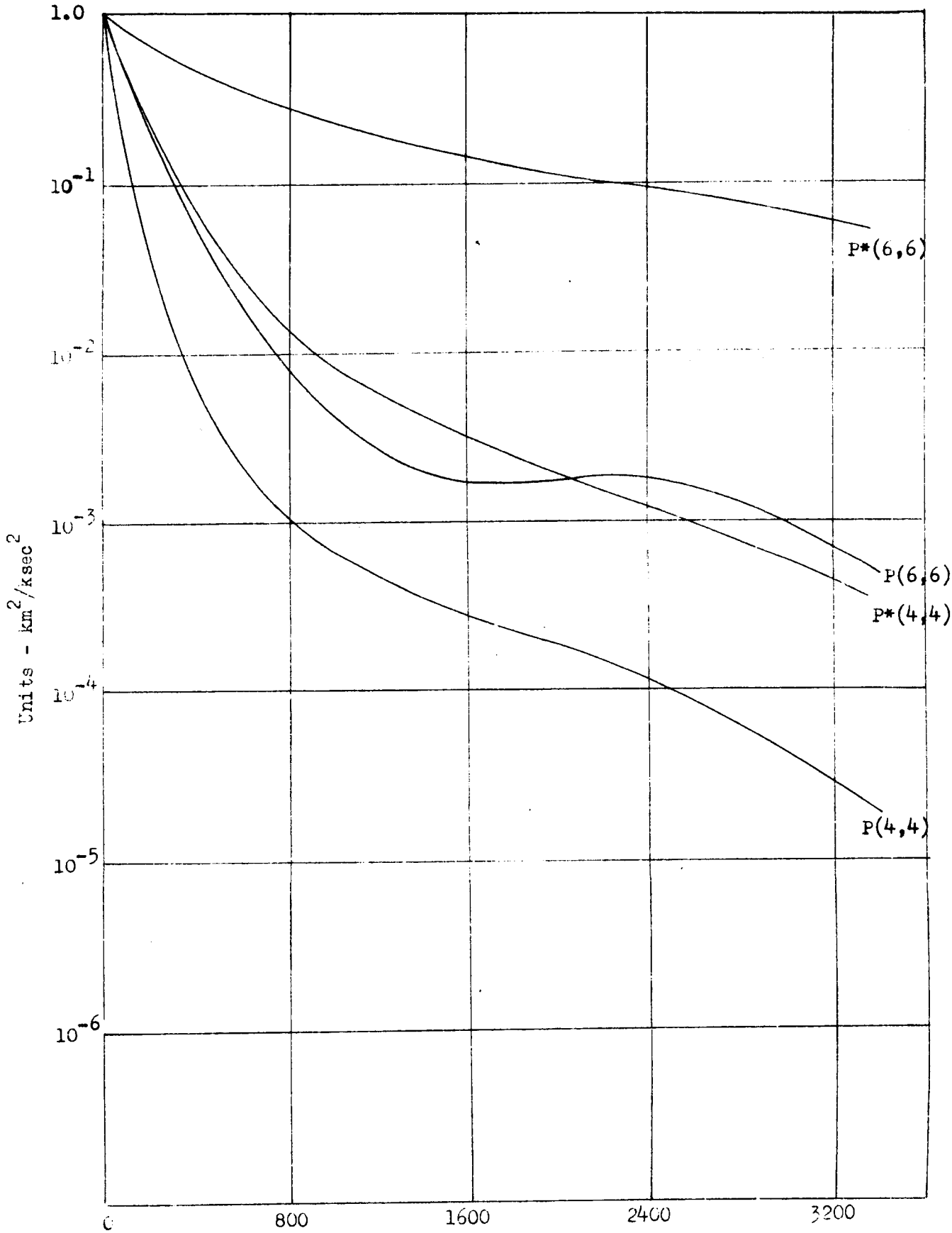

1.0
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
Units - $km^2/ksec^2$
0
800
1600
2400
3200
P*(6,6)
P(6,6)
P*(4,4)
P(4,4)

Fig. IV-7

the out of plane positions that are different by one order of magnitude, almost all of which can be attributed to the difference in accuracy of the measurements used for the two cases.

The covariance elements associated with velocity uncertainty propagation in Figure IV-7 demonstrate another peculiarity. Contrasting P_{44} and P^*_{44}, one sees that both elements are stable and that, in a sense, the extra measurement accuracy has not significantly reduced the propagated range rate uncertainty. In the broad problem of preliminary mission design, one of the major areas of study concerns the required accuracy of the measuring devices. An interesting fact that should affect these studies is that once the propagated uncertainties are highly stable, as shown by P^*_{44}, increasingly accurate measurements will <u>not</u> reduce these uncertainties in a way commensurate with the increased accuracy. Thus, from a cost point of view, there is no reason to buy more precise (and more expensive) instruments when the less precise ones are already accurate enough to force high stability on the propagated uncertainties. For a specific mission, there is some set of measuring devices that will reduce propagated uncertainties enough to be considered cost-optimal.

The next two graphs conclude the results from experiments conducted on a coasting arc without midcourse corrections. Unlike the experiments already discussed, these plots deal with the magnitudes of state vector deviations. The theme of Figures IV-8 and IV-9 could be described as follows: assume a coasting arc along a chosen nominal trajectory has been defined. Assume further that a specified initial state vector disturbance has been selected based realistically upon a covariance matrix

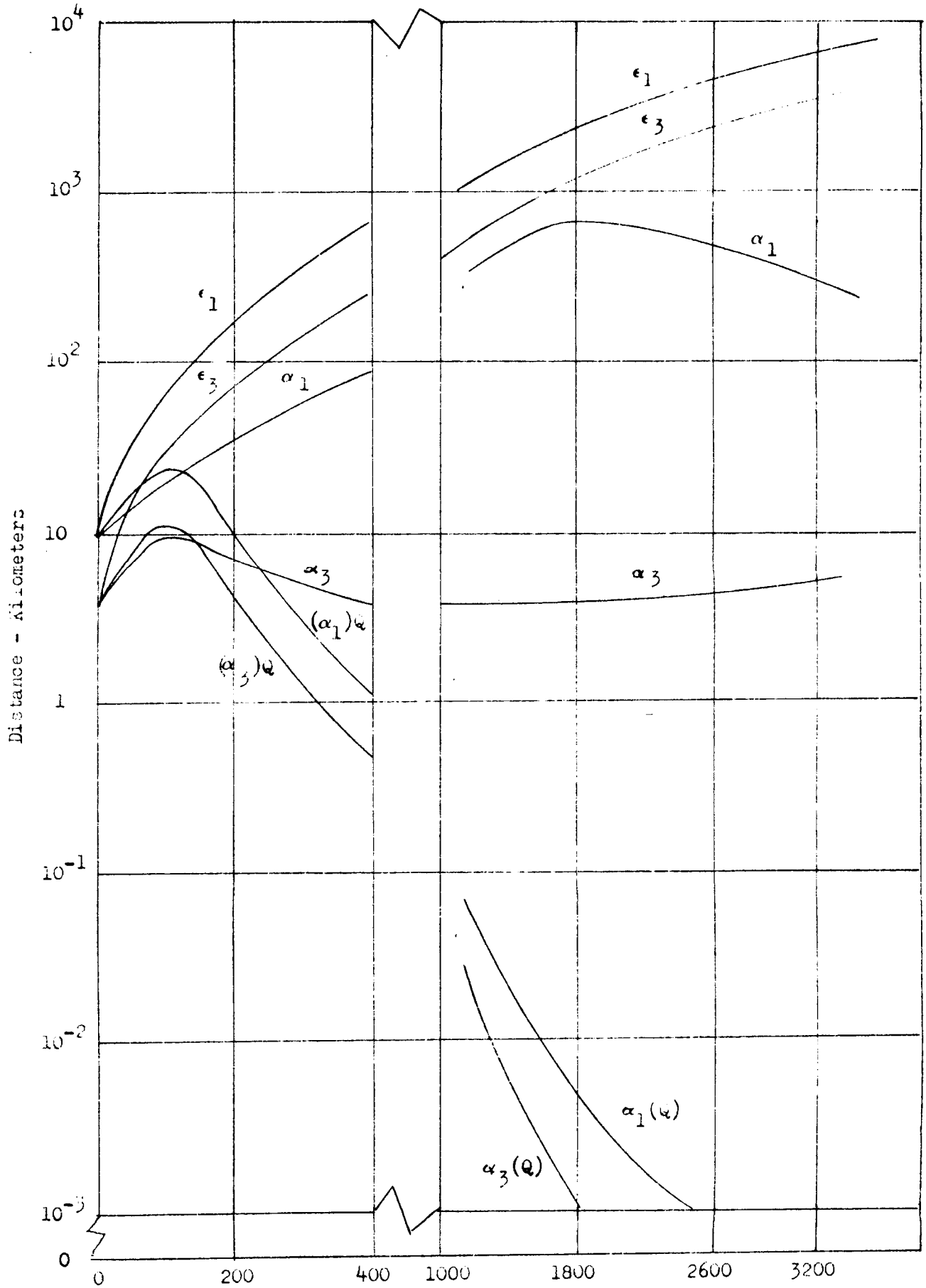

10^4
10^3
10^2
10
1
10^{-1}
10^{-2}
10^{-3}
0
Distance - Kilometers
ϵ_1
ϵ_3
α_1
α_3
$(\alpha_1)_Q$
$(\alpha_3)_Q$
$\alpha_1(Q)$
$\alpha_3(Q)$
0
200
400
1000
1800
2600
3200
Time - Kiloseconds

Fig. IV-8

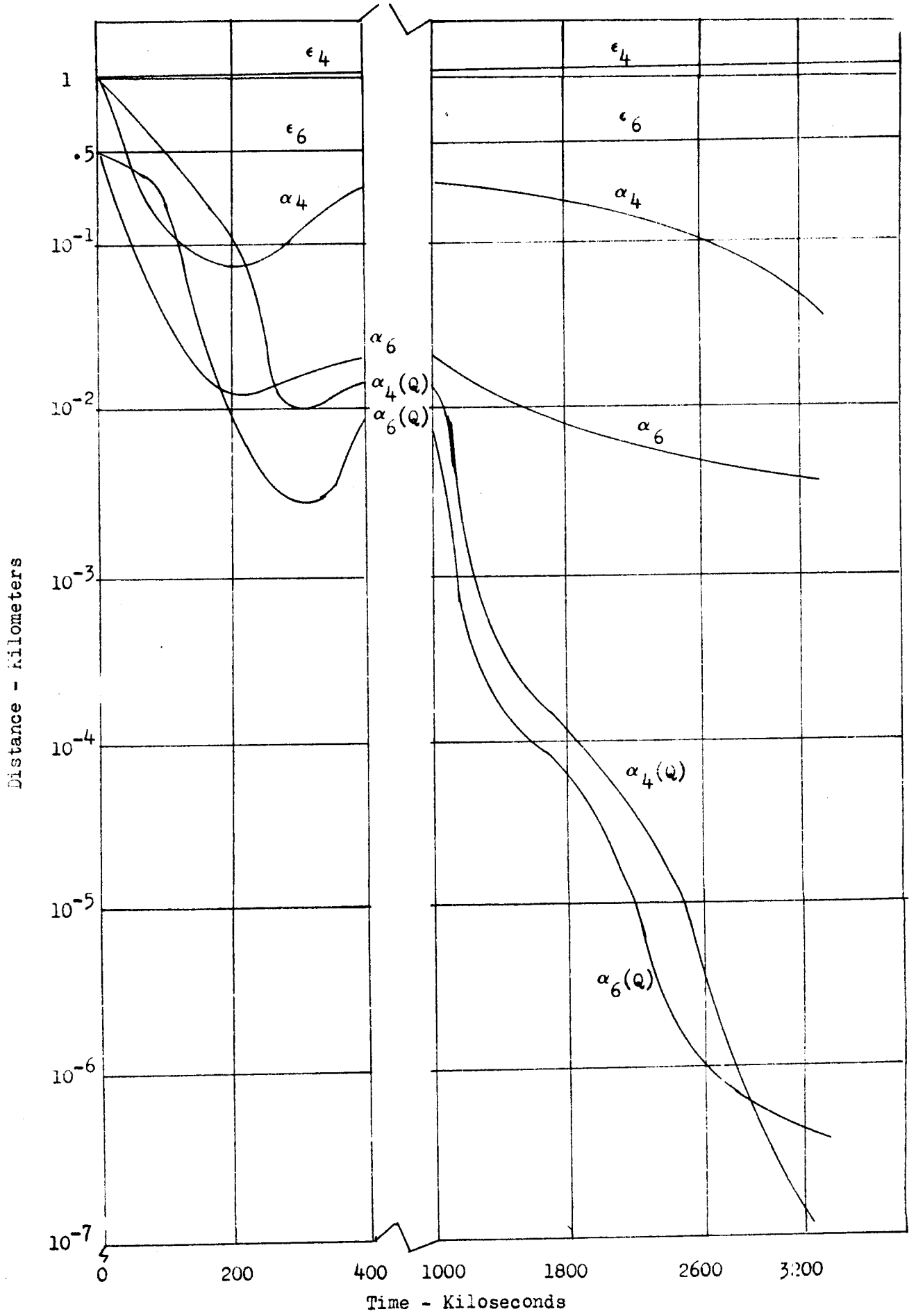

ϵ_4
ϵ_6
α_4
α_6
$\alpha_4(Q)$
$\alpha_6(Q)$
1
.5
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
Distance - Kilometers
0
200
400
1000
1800
2600
3200
Time - Kiloseconds

Fig. IV-9

associated with the initial conditions. If the actual measurement accuracy is one hundred times greater than the accuracy assumed for the observations in the estimation process, what is the resulting convergence of the estimated state vector to the actual state vector? Furthermore, by supposing arbitrary white noise in the dynamic model and slowly removing excess weighting of the a priori information, is it possible to increase the convergence rate?

A comment or two should be made about the so-called Q matrix, a covariance matrix associated with white noise in the dynamic model. This matrix is defined in the companion analytic document as being one way of preventing estimate-associated covariances from becoming too optimistic. By making Q non-zero in the estimation process, the result is a carefully planned increase in the state transition matrix propagated covariance between navigation epochs. This increase, in turn, implies extra uncertainty in the dynamic model information and, consequently, decreases the weight of the dynamics information at each stage of the recursive filtering process. The particular problem that produced Figures IV-8 and IV-9 was chosen with a purpose. For reasons that will become apparent in a later graph, there is justifiable logic behind choosing very conservative measurement accuracies for use in the estimation procedure. If one has indeed been extremely conservative in that choice and has reason to believe that the measurements are actually much better, one can increase convergence and still circumvent the divergence problem by using arbitrary dynamic noise Q.

The plots $\epsilon_1, \epsilon_3, \epsilon_4$, and ϵ_6 in the two figures are simply magnitudes of state vector perturbations propagated along the coasting arc by

using the state transition matrices. Thus, for the nominal trajectory being used, an initial perturbation of 10 km. in range grows to approximately 8000 km. by the end of the coast. The out of plane position perturbation shows a similar growth, although the two velocity perturbations propagated in Figure IV-9 are not increasing rapidly. The program was exercised in its simulation mode for these plots and the quantities α_k show the magnitude of the difference between the actual and estimated state vector components <u>after</u> each of the sixty-four observations were processed. By contrast, the values $(\alpha_k)_Q$ are the magnitudes of actual-estimated differences resulting from using a non-zero value for Q in the estimation process.

This problem was, in a sense, "rigged" for the desired effect. In making the run with $Q = 0$, the assumed covariance associated with the measurements was made more or less commensurate with the initial covariance matrix P_o. Thus, even though the actual measurements were one hundred times more accurate than assumed, the estimation process was reluctant to unweight its <u>a priori</u> information because it had been led to believe, by the assumed covariance statements, that the measurements were no more accurate than the initial information. The result of this was a very slow convergence (decreasing values for α_k in the second figure) in which the initial estimate was very slowly thrown out. From Figure IV-8 it can be seen that the accuracy of the position estimates either worsened or failed to improve. This was due to the rapid growth of position uncertainty and, despite the highly accurate measurements actually recorded, the unwillingness of the estimation procedure to put

too much stock in subsequent measurements.

The use of a non-zero Q causes the propagated covariance matrix associated with successive state vector estimates to grow very rapidly. The validity of the initial estimate becomes more and more in question and, consequently, the estimation process is forced eventually to rely almost completely upon its new measurement. For this "rigged" problem, the measurements were a hundred times more accurate than the estimation process assumed and consequently the estimated state vector converged somewhat uncannily to the actual state vector, as demonstrated by the plots of $(\alpha_k)_Q$ in Figures IV-8 and IV-9.

It should also be noticed that after two observation vectors, one hundred kiloseconds after the beginning of the coast, the estimates using $Q = 0$ were more accurate. This is understandable since using an arbitrary Q does cause the estimates to be non-optimal in the minimum variance sense and, after only two observations, the initial uncertainties have not grown enough to cause the a priori information to be effectively discarded. These two graphs demonstrate that a judicious use of the Q matrix, even when it is arbitrary, can produce more satisfactory results from the estimation procedure.

The remainder of the numerical experiments pertain to missions involving two midcourse corrections and three coasting arcs before and after the corrections. Again a certain nominal trajectory has been specified but this time a midcourse guidance policy has been included in the problem. The guidance law is simple fixed time of arrival, whose purpose is to null the position deviation from the nominal at the

predetermined target time t = 14400 kiloseconds after injection. The two guidance corrections are imparted at times t = 5000 and 10000 kiloseconds after the initiation of the mission. The most important results came from exercising the program in its simulation mode and thus all of the quantities plotted are related to the state vector itself rather than the propagated covariance matrices. After some preliminary trials, it was decided to run all the experiments with a small, non zero dynamic noise Q in the process to prevent divergence of a kind that would obscure the significance of the results.

All of the missions considered had the same basic structure. Ten observation vectors were recorded and processed during each of the three coasting periods and the same nominal trajectory was used for all the missions. Midcourse execution errors, both monitored and unmonitored, were assumed for some of the more complex analyses, although the first three graphs present results of problems in which the midcourse execution errors were zero.

Figures IV-10 and IV-11 pertain to the same experiment. The problem was to determine the result of different assumed and actual measurement accuracies on the performance of the mission. Everything else about the missions was identical. The four trajectories whose significant data is presented in Figures IV-10 and IV-11 are characterized in the table immediately below.

Assumed → / Actual ↓	Low	High
Low	Mission 2	Mission 4
High	Mission 1	Mission 3

TABLE IV-1

Missions defined according to assumed and actual measurement accuracy.

This problem demonstrates the value of having a simulation capability in an interplanetary analysis program. The actual measurement accuracy statement was used to generate random noise perturbations that corrupted the measurement vectors, where the non-noisy measurement vectors were computed from the actual state vector information. On the other hand, the assumed measurement accuracies were used in the estimation process to determine the weight to be accorded to the new observation.

Figure IV-10 shows the actual state vector deviation in the range component for the four missions as a function of time. The subscripts on ϵ refer to the individual missions defined in the above table. An initial range deviation of three kilometers was assumed and this deviation had increased by about a factor of one hundred prior to the first midcourse correction. Since the state transition matrices that propagate perturbations about the nominal trajectory were the same for all four missions, the plots coincide up until the time of first midcourse correction. At the time t = 5000 kiloseconds, a midcourse correction is imparted, based

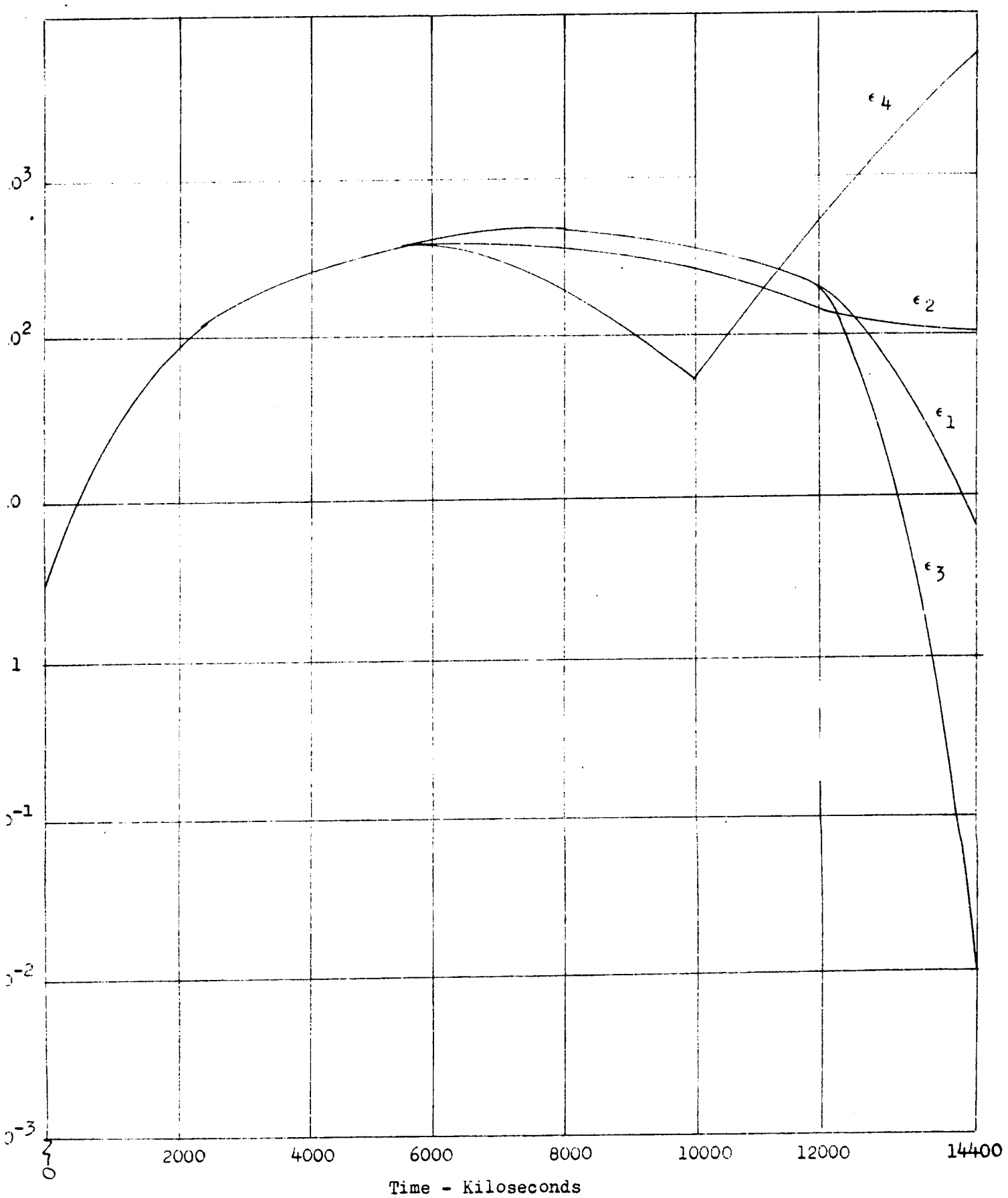

ϵ_4
ϵ_2
ϵ_1
ϵ_3
.0³
.0²
.0
1
-1
-2
-3
0
2000
4000
6000
8000
10000
12000
14400
Time - Kiloseconds

Fig. IV-10

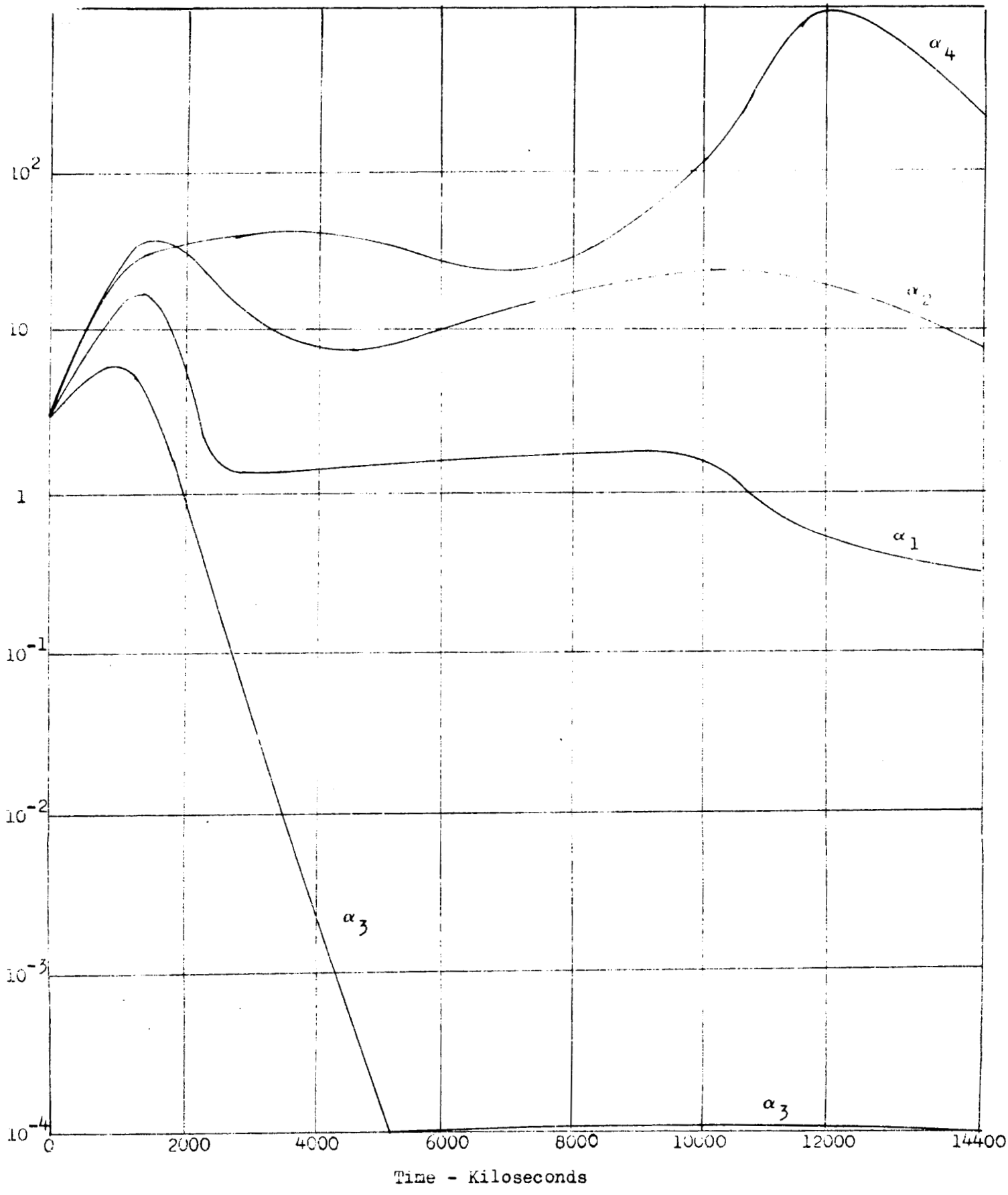

10^2
10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
0
2000
4000
6000
8000
10000
12000
14400
α_4
α_2
α_1
α_3
α_3
Time - Kiloseconds

Fig. IV-11

upon the state vector estimate at that time, that is designed to null the position deviation from the nominal at the target time t = 14400 kiloseconds. The variations in propagated range perturbation between 5000 and 10000 kiloseconds are due to different midcourse corrections which, in turn, imply different state vector estimates at t = 5000.

The imparted correction for missions 1 and 2 was virtually the same and the two propagated range variations are inseparable on the scale of Figure IV-10. At the time t = 10000, another midcourse correction is made, again based upon the state vector estimate at that time. For these runs the correction is perfectly performed and the resulting range deviation at the target is shown as the final point on the plot. Recall that the purpose of the correction was to null the position deviation at the target. Assuming that the guidance law was correctly used, the final range deviations, varying from 8×10^{-3} kilometers for mission 3 to 6×10^{3} kilometers for mission 4, can be attributed to the fact that the final midcourse correction was computed by using an estimate of the state vector that was different from the actual state vector. This is true since the midcourse execution was perfect.

Figure IV-11 plots the magnitude of the difference between the actual and estimated state vectors for the four missions. The plots reveal the reason for the results of the earlier figure. The most accurate estimate of the trajectory, as would be expected, comes from the mission that assumes the measurements are highly accurate when they are. Due to this estimation accuracy, the midcourse corrections effectively achieve their purpose of nulling the target range deviation. Mission 1 is the second most successful.

The measurements are actually the same as in the most effective mission, but due to the assumption of low measurement accuracy, they are given less weight in the estimation process. This causes the estimate to vary from the actual state vector and consequently the midcourse corrections do not completely null the target range deviation.

The second mission correctly assumes measurements of low accuracy. The result is that the estimation procedure generates midcourse correction which do primarily control the range deviation at the target instead of driving it to zero. The fourth mission, for which the measurements were assumed to be highly accurate when they were, in fact, of low accuracy, is a classic example of the divergence problem. After a certain number of observations, the estimation procedure, which has been using the assumed measurement accuracies, supposes that it knows the trajectory much more precisely than it does. As a result new observations, which don't agree with the propagated dynamics information, begin to be neglected after a certain point. The estimated trajectory then diverges from the actual trajectory and the second midcourse correction, acting upon faulty trajectory estimation information, results in a mammoth range deviation at the target.

The relatively low range deviations between 5000 and 10000 kiloseconds for mission 4 are entirely happenstance. Based upon the terminal accuracy of mission 3, it can safely be assumed that if the proper correction had been made at $t = 5000$, the range perturbation between the two corrections would parallel that of mission 3. It should also be pointed out that some wild variations in α_4 at the beginning of the mission have been suppressed

for plotting simplicity -- these variations were due to an occasional observation being by chance very close to precise.

Figure IV-12 is a study of the effect of an extremely large initial perturbation on the estimation process. For the odd number missions, measurements were correctly assumed to be highly accurate; for the even number missions, both the actual and assumed measurement accuracy was low. The ϵ_k, α_k plots are the same quantities as in the previous graphs. Both α_1 and α_3 were small enough to drop immediately off the scale and have not been shown.

The only difference between these missions, other than the accuracy of the measurements, is the initial state vector perturbation. Although the injection accuracy was <u>assumed</u> to be the same in both cases, the larger initial perturbation in range was one member of a six component perturbation taken from the 5σ-hypersurface defined by the initial covariance matrix. Therefore two effects are shown in Figure IV-12. Due to the larger initial perturbation, a larger midcourse correction is required to null the target range deviation. Secondly, since the original perturbation is, probabilistically, highly unlikely, the estimation process will have trouble converging to the actual state vector and determining the proper midcourse correction.

For the highly accurate measurements, despite the large initial perturbation, an accurate state vector estimate is achieved and the target range deviation is effectively nulled. For the less accurate measurements, the estimation process does not successfully reproduce the actual trajectory. Although the percentage deviation of the estimated trajectory from the

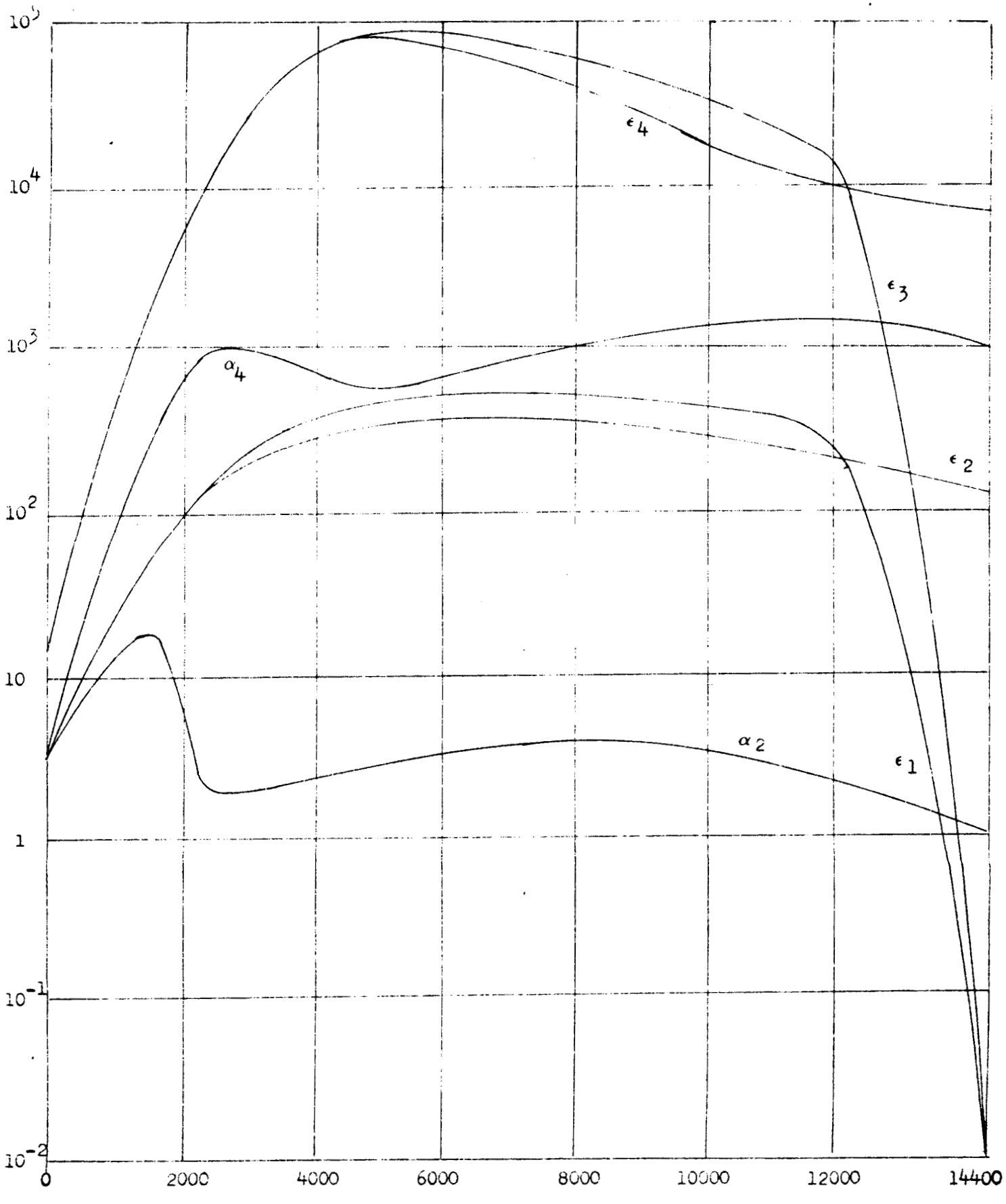

10^5
10^4
10^3
10^2
10
1
10^{-1}
10^{-2}
0
2000
4000
6000
8000
10000
12000
14400
ϵ_4
ϵ_3
α_4
ϵ_2
α_2
ϵ_1

Fig. IV-12

actual trajectory in the second instance is roughly the same, the resulting error in the midcourse correction propagates into a target range deviation that is almost an order of magnitude higher.

The remaining figures are all concerned with a series of six experiments associated with midcourse execution errors. The six missions are defined in the table immediately below.

Mission Number	Initial Uncertainty		Measurement Uncertainty		Execution Uncertainty	
	Assumed	Actual	Assumed	Actual	Assumed	Actual
1	Small	Small	Small	Small	Large	Large
2	S	S	S	S	L	S
3	S	S	L	S	L	L
4	S	S	L	S	L	S
5	S	L	S	S	L	L
6	S	L	S	S	L	S

TABLE IV-2

The missions are classified according to the size of the three major uncertainties associated with the estimation process. The so-called large uncertainties reflect covariance matrices roughly one hundred times the size of the small uncertainties. The above six missions were chosen from all permutations possible in the table to demonstrate certain key points.

Figure IV-13 contains in tabular form the data associated with the six missions involving midcourse execution errors. The columns show the estimated range rate just prior to the correction; the actual range rate at the same time (a difference between these two naturally means that

Section Number	Correction Number	Estimated $\dot{r}$ Meter/Sec.	Actual $\dot{r}$ Meter/Sec.	Computed $\dot{r}$ Correction	Estimated $\dot{r}$ Correction	Actual $\dot{r}$ Correction	Final Deviation in r in Kilometers
1	1	2.50379 10^{-1}	2.50379 10^{-1}	-1.24613 10^{-1}	[illegible]	[illegible] 10^{-1}	
1	2	-4.42019 10^{-1}	-4.42018 10^{-1}	7.28113 10^{-1}	9.16875 10^{-1}	1.[illegible]	1302.54
2	1	2.50379 10^{-1}	2.50379 10^{-1}	-1.24613 10^{-1}	[illegible] 10^{-1}	[illegible] 10^{-1}	
2	2	-4.70922 10^{-2}	-4.70922 10^{-2}	7.36237 10^{-2}	9.25163 10^{-3}	[illegible] 10^{-2}	14.2051
3	1	2.50199 10^{-1}	2.50379 10^{-1}	-1.25402 10^{-1}	-1.99759 10^{-1}	[illegible] 10^{-1}	
3	2	-4.37773 10^{-1}	-4.40669 10^{-1}	9.24655 10^{-1}	9.14025 10^{-1}	1.00902	1309.44
4	1	2.50199 10^{-1}	[illegible] 10^{-1}	-1.25402 10^{-1}	-1.32837 10^{-1}	-1.[illegible] 10^{-1}	
4	2	-4.63977 10^{-2}	-4.50862 10^{-2}	6.73813 10^{-3}	8.73370 10^{-3}	9.[illegible] 10^{-3}	1[illegible].583
5	1	23.5219	23.5324	-33.4524	-33.6406	-33.7[illegible]	
5	2	-7.58726	-7.59317	.932980	1.1955	1.20[illegible]3	937.886
6	1	23.5219	23.5324	-33.4524	-33.4543	-33.4586	
6	2	-7.24187	-7.24187	1.76154 10^{-2}	1.94704 10^{-2}	2.03997 10^{-2}	9.51

Fig. IV-13

the imparted correction is not strictly optimal); the computed impulsive range rate correction based upon the estimated state vector and the fixed-time-of-arrival guidance policy; the estimated range rate correction which contains the monitored execution errors; and the actual range rate correction, which is the estimated correction plus the unmonitored execution errors. The final column contains the final range deviation in kilometers and is a measure of the effectiveness of the correction-estimation process. The data themselves are self-explanatory except for one item. Notice that all the execution errors have the same direction as the computed correction. This can be explained by mentioning that only two sets of random numbers (one for large errors, another for small) were used to generate all the execution errors in an attempt to make the results more nearly uniform.

In Figure IV-14 the actual range deviations from the nominal are plotted for all six missions. What is most significant from this graph is that, at least for the ranges of uncertainties considered, by far the most important contributor to failure to null the target range deviation is the execution error. The three missions with small execution errors are neatly grouped and those with large execution errors also fall together. The failure to model measurement errors properly (consider ϵ_2 vs. ϵ_4 or ϵ_1 vs. ϵ_3) is relatively insignificant, as long as measurement accuracies are conservatively chosen. As was shown earlier, overly-optimistic measurement accuracies are disastrous in themselves. In addition, due to the small actual measurement uncertainties, initial perturbations of a large magnitude, even when not expected, do not hinder the process in an untoward manner.

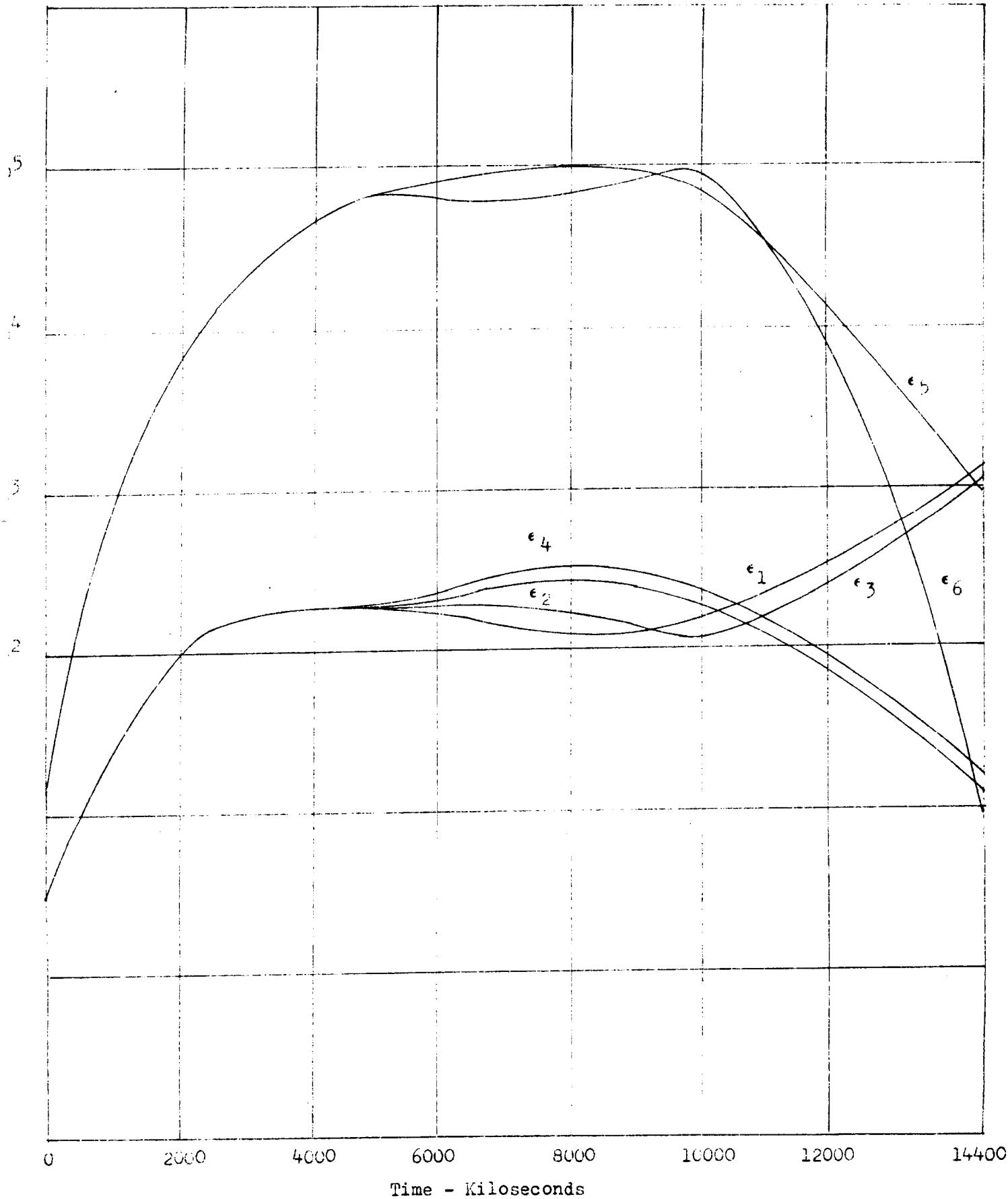

5
4
3
2
ϵ_5
ϵ_4
ϵ_1
ϵ_2
ϵ_3
ϵ_6
0
2000
4000
6000
8000
10000
12000
14400
Time - Kiloseconds

Fig. IV-14

The last two figures (IV-15 and IV-16) are the customary plots of the magnitude difference between the estimated and actual range. The natural separation into two categories, one reflecting large execution errors and the other small, is evident. Figure IV-15 demonstrates that one of the major benefits from small execution errors is that the estimation process (for highly accurate measurements) quickly converges, thus assuming that the next correction will be near-optimal. By contrast, in Figure IV-16 one can see that the presence of a large execution error, even if accounted for by the estimation process as it propagates covariance matrices, delays the convergence of the estimated state vector to the actual state vector and thus causes subsequent computed midcourse corrections to be in error.

The times t = 5500 and 10500 kiloseconds represent the first observation times after the midcourse corrections. The jumps in the magnitudes of the differences between actual and estimated range at those points can be explained by the unmonitored execution errors.

Many different experiments associated with the midcourse execution errors can be conceived. For certain missions, for example, one could prove the conjecture that, if a probabilistic terminal accuracy cost function were used, one might not want to impart a correction even when the propagated range deviation is not nulled at the target. Another legitimate and valuable parametric study would seek to determine the accuracy gains made by monitoring the execution process -- perhaps a mission might even by designed for which there are no major penalties for failing to monitor any of the execution errors.

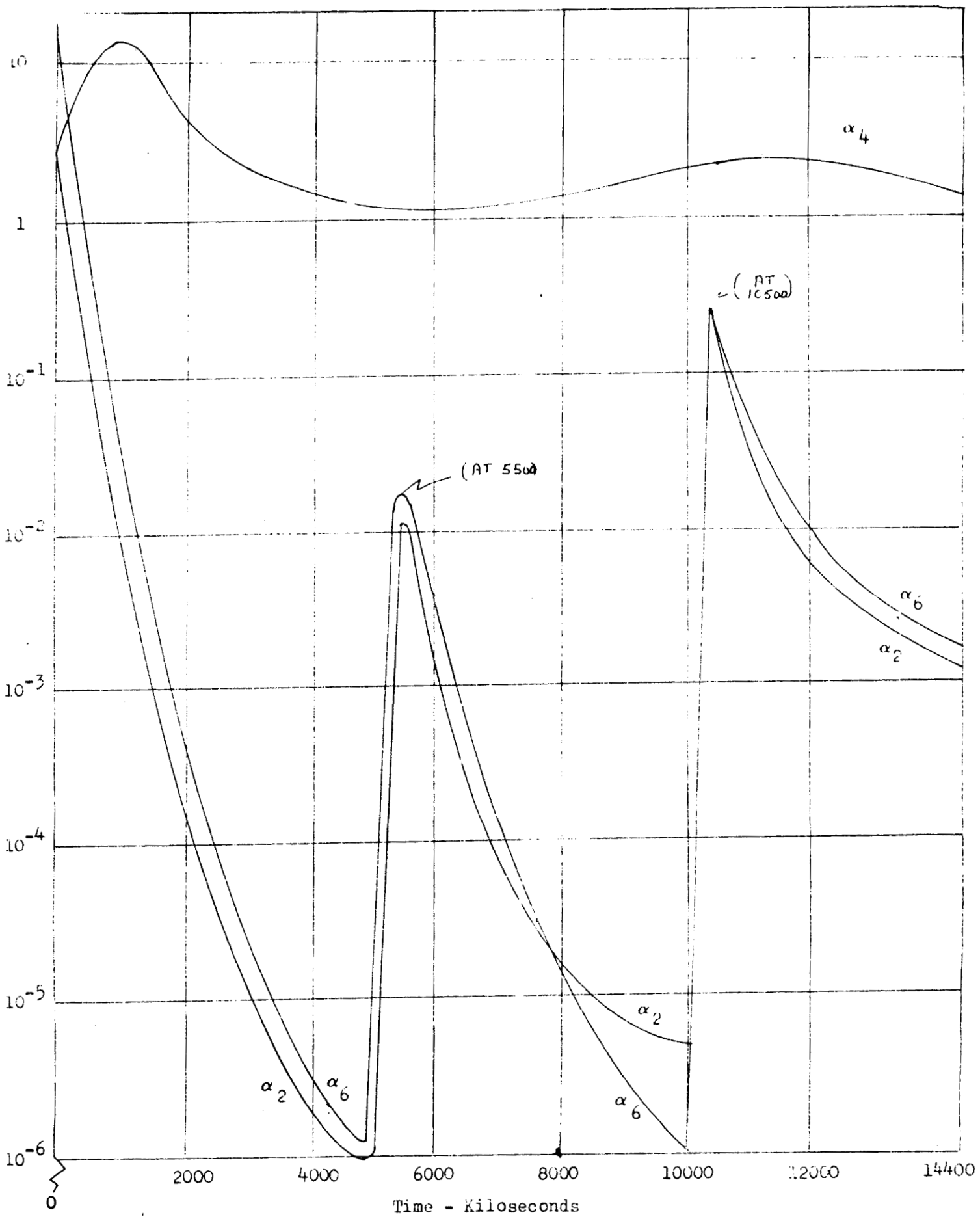

10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
0
2000
4000
6000
8000
10000
12000
14400
Time - Kiloseconds
α_4
(AT 10500)
(AT 5500)
α_6
α_2
α_2
α_6
α_2
α_6

Fig. IV-15

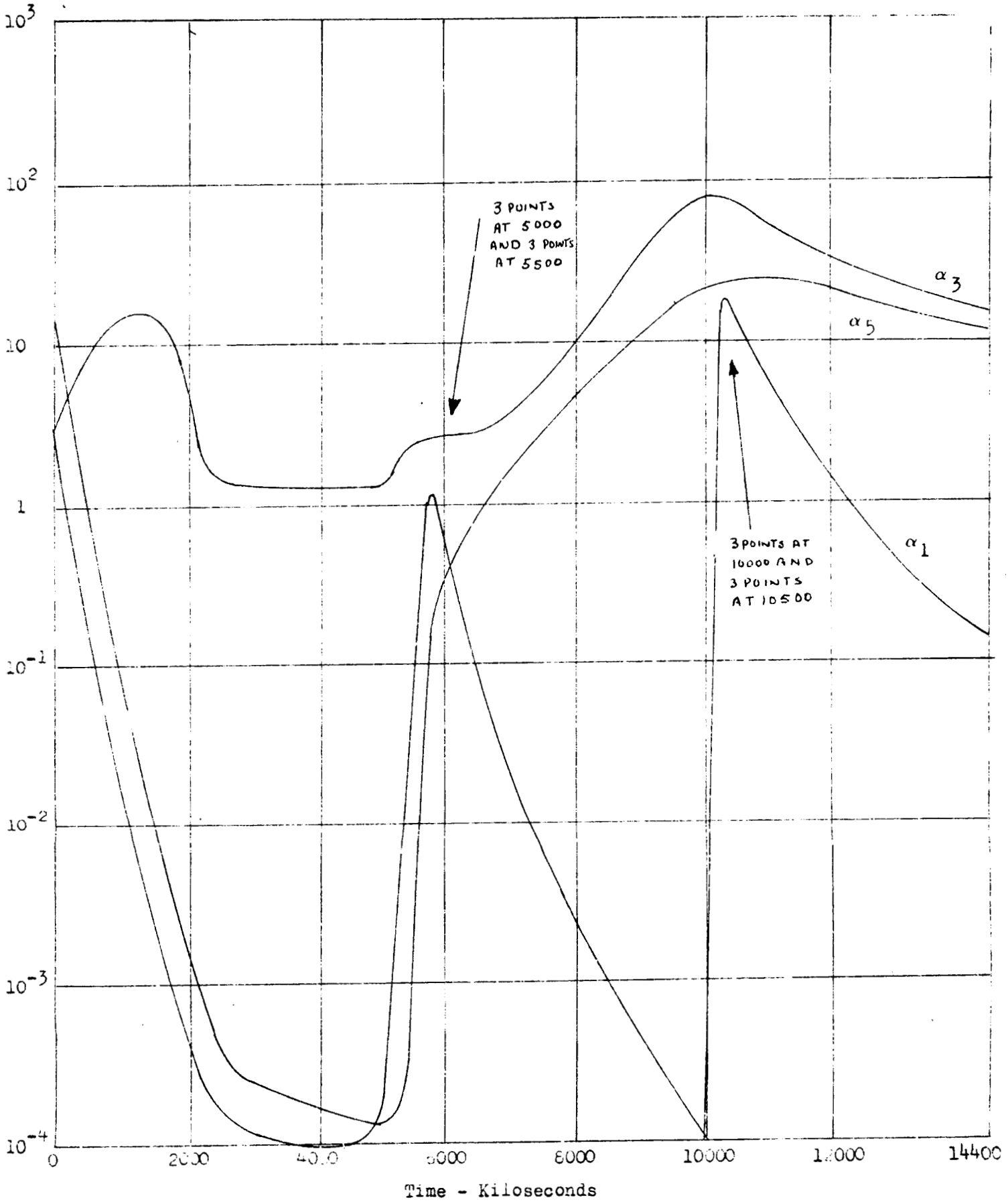

10^3
10^2
10
1
10^{-1}
10^{-2}
10^{-3}
10^{-4}
0
2000
4000
6000
8000
10000
12000
14400
Time - Kiloseconds
3 POINTS AT 5000 AND 3 POINTS AT 5500
3 POINTS AT 10000 AND 3 POINTS AT 10500
α_3
α_5
α_1

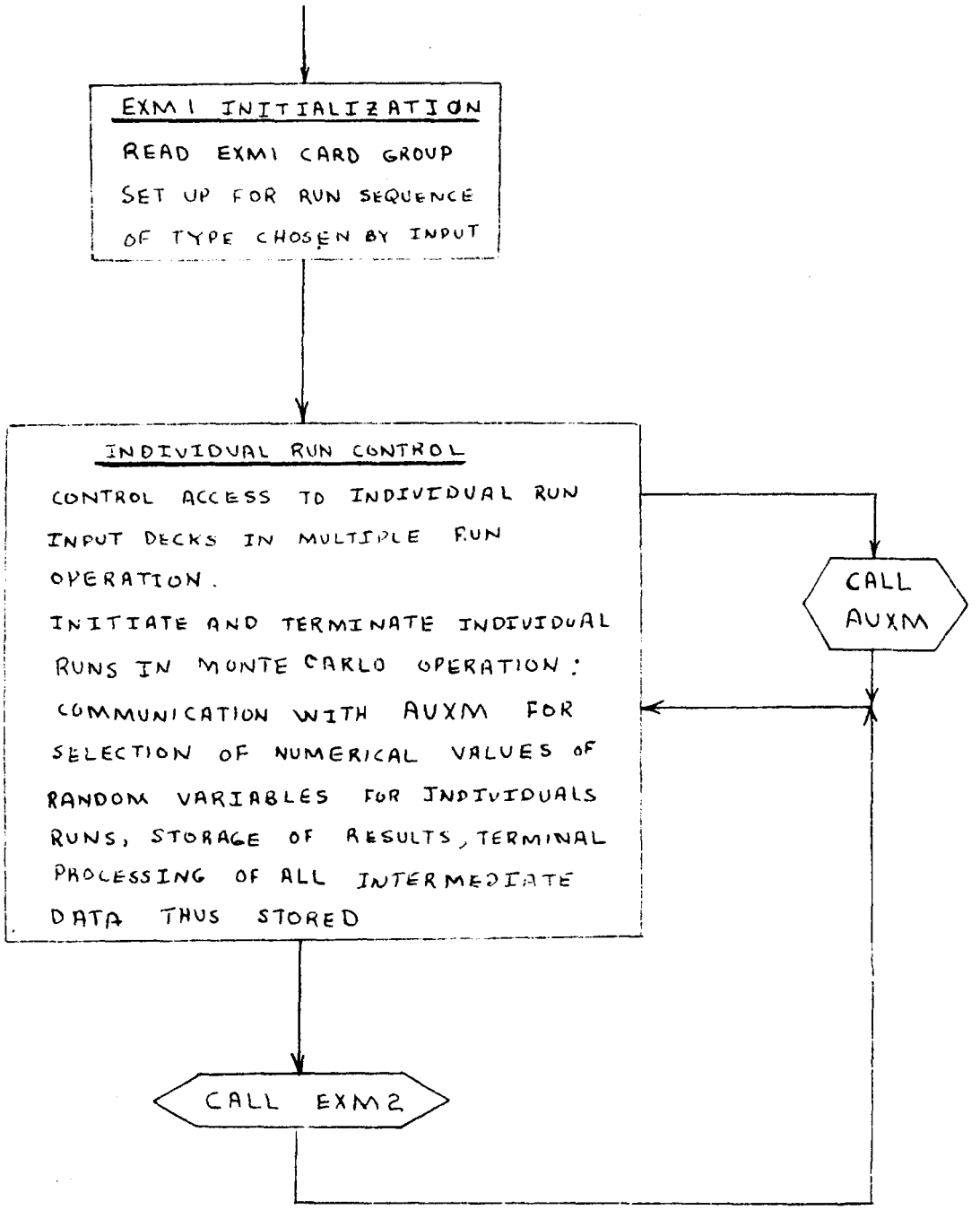
Fig. IV-16

No attempt was made in setting up these numerical experiments to exhaust all the possible studies. A good analysis tool always suggests many more studies than can be run in a set time period. The purpose of these experiments was simply to demonstrate the operation of the tutorial program and exhibit the problem solving flexibility that one might expect from the ultimate program. In addition, it should be stressed that by exercising the program as was done in these numerical experiments, increased familiarity with the overall interplanetary problem will certainly result.

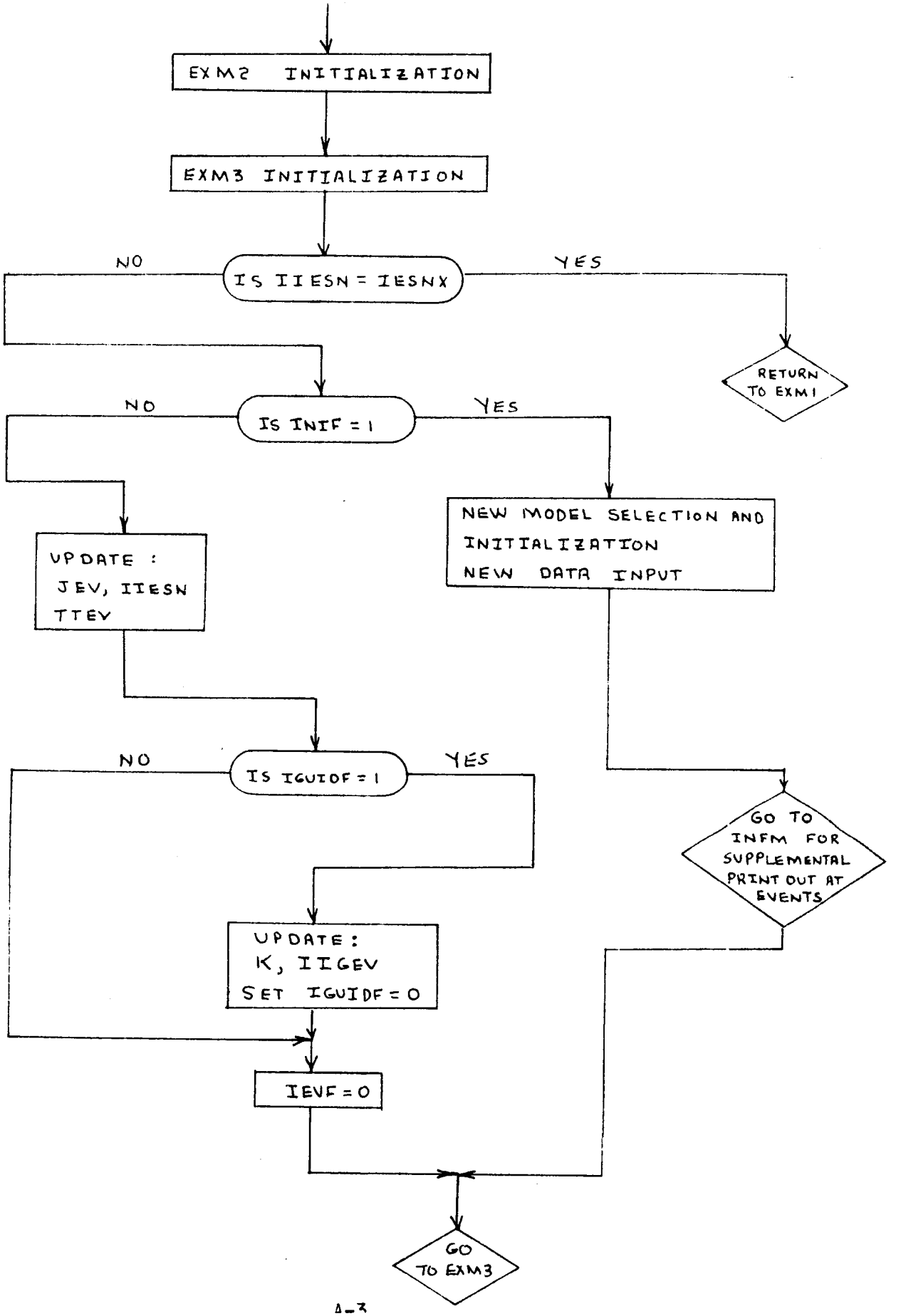
APPENDIX 1

Flow Charts for the Ultimate Program

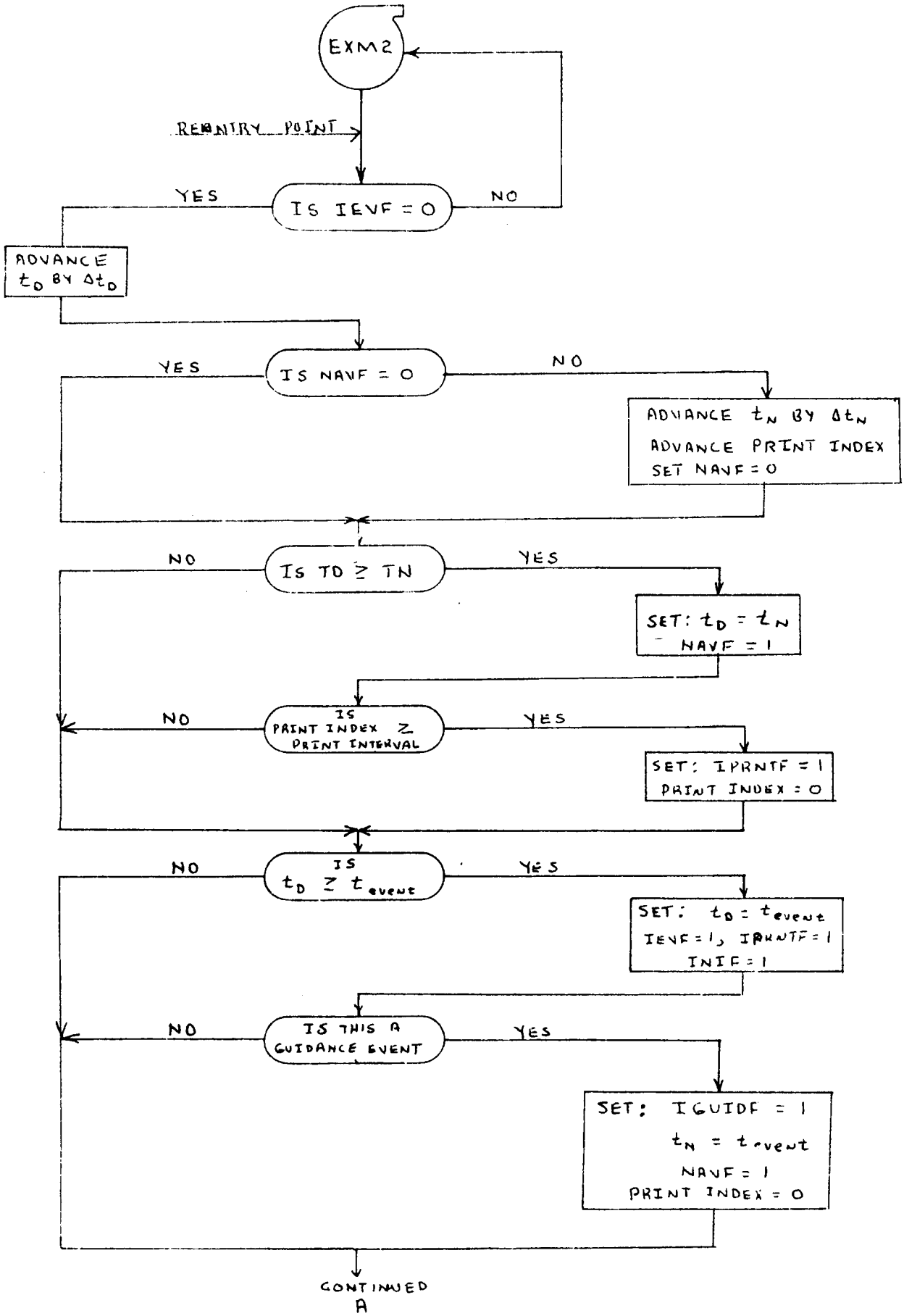
EXM1

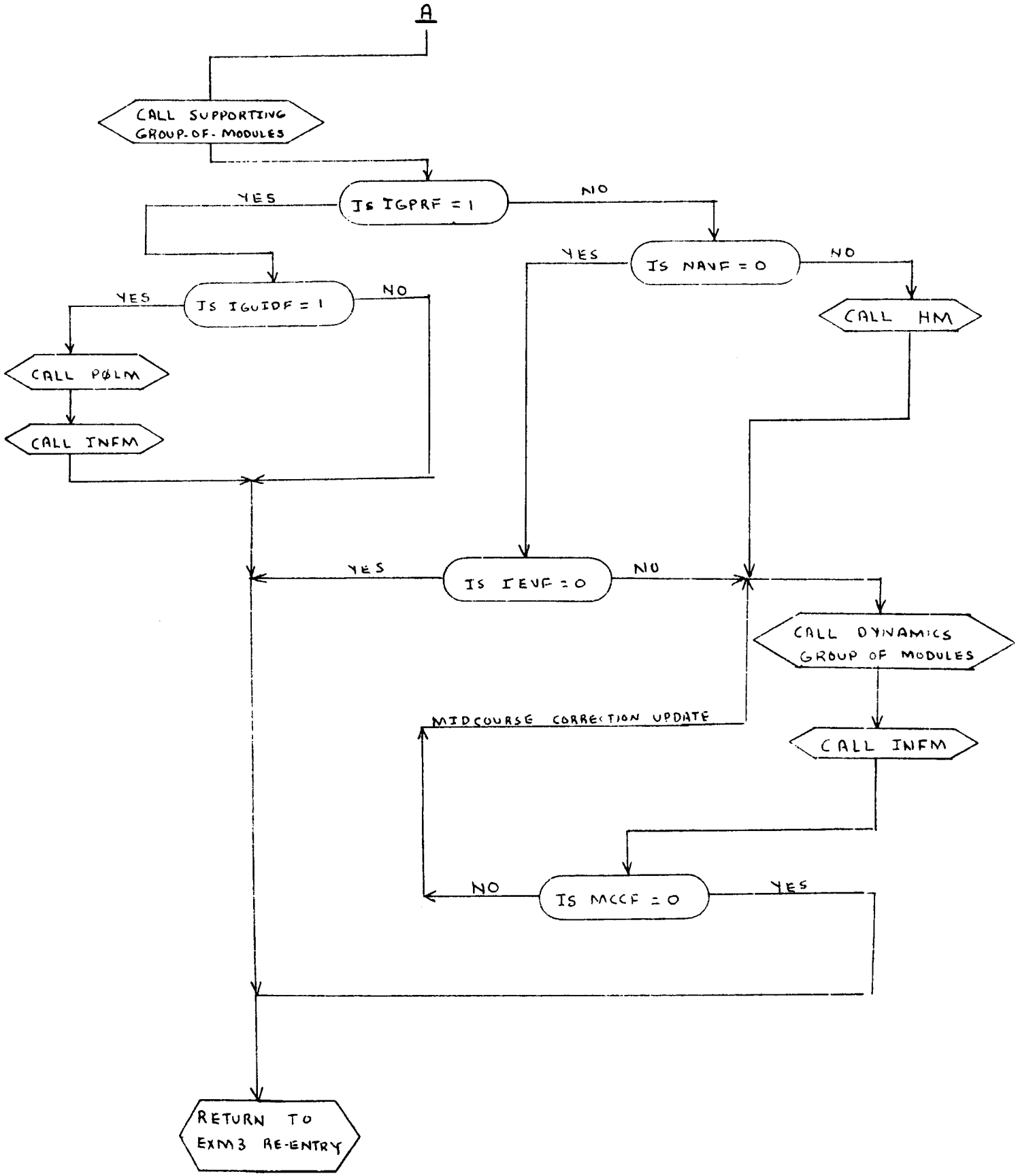

EXM1 INITIALIZATION
READ EXM1 CARD GROUP
SET UP FOR RUN SEQUENCE
OF TYPE CHOSEN BY INPUT
INDIVIDUAL RUN CONTROL
CONTROL ACCESS TO INDIVIDUAL RUN
INPUT DECKS IN MULTIPLE RUN
OPERATION.
INITIATE AND TERMINATE INDIVIDUAL
RUNS IN MONTE CARLO OPERATION:
COMMUNICATION WITH AUXM FOR
SELECTION OF NUMERICAL VALUES OF
RANDOM VARIABLES FOR INDIVIDUALS
RUNS, STORAGE OF RESULTS, TERMINAL
PROCESSING OF ALL INTERMEDIATE
DATA THUS STORED
CALL AUXM
CALL EXM2

EXM2

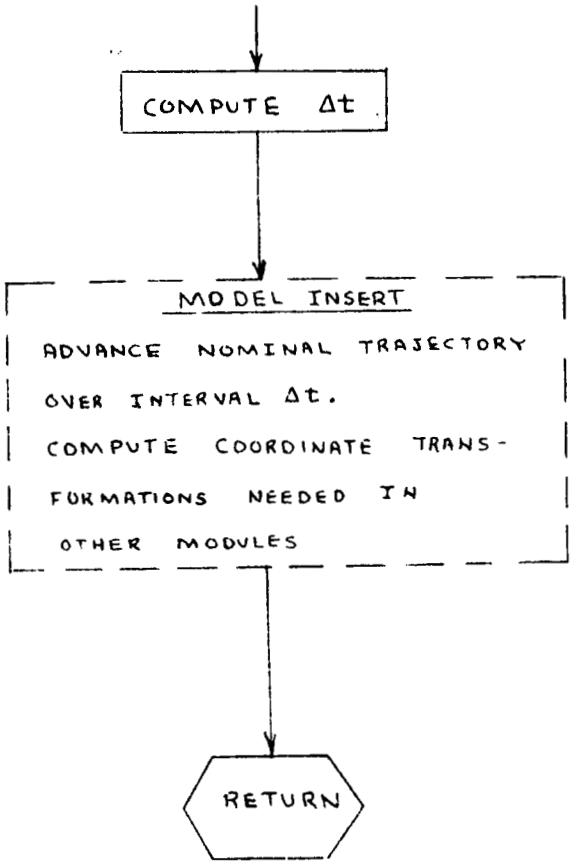

EXM2 INITIALIZATION
EXM3 INITIALIZATION
IS IIESN = IESNX
NO
YES
RETURN TO EXM1
IS INIF = 1
NO
YES
NEW MODEL SELECTION AND INITIALIZATION NEW DATA INPUT
UPDATE : JEV, IIESN TTEV
IS IGUIDF = 1
NO
YES
GO TO INFM FOR SUPPLEMENTAL PRINT OUT AT EVENTS
UPDATE: K, IIGEV SET IGUIDF = 0
IEVF = 0
GO TO EXM3

EXM3

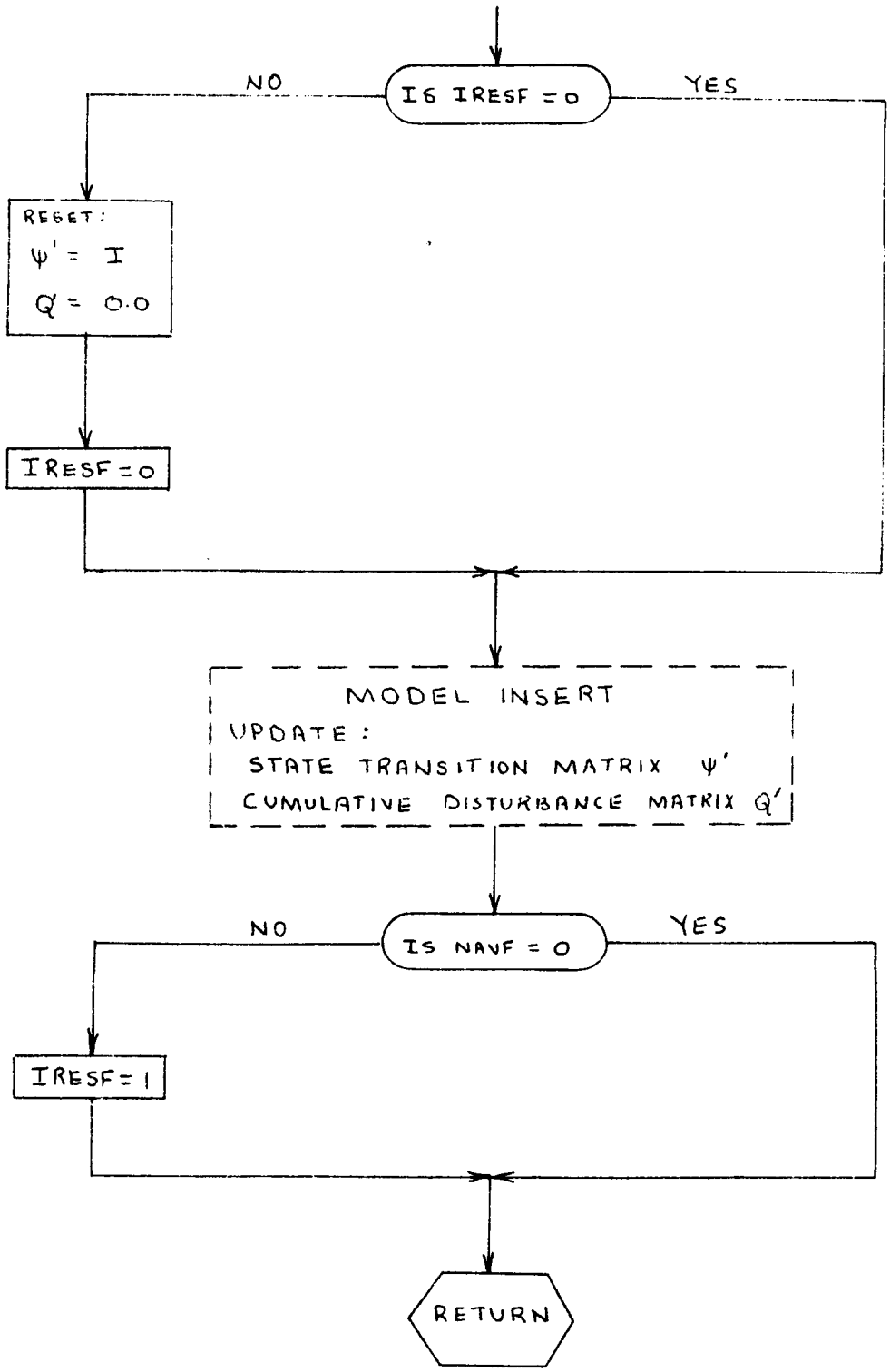

EXM2
REENTRY POINT
IS IEVF = 0
YES
NO
ADVANCE t_D BY Δt_D
IS NAVF = 0
YES
NO
ADVANCE t_N BY Δt_N
ADVANCE PRINT INDEX
SET NAVF = 0
IS TD ≥ TN
NO
YES
SET: $t_D = t_N$
NAVF = 1
IS PRINT INDEX ≥ PRINT INTERVAL
NO
YES
SET: IPRNTF = 1
PRINT INDEX = 0
IS $t_D \geq t_{event}$
NO
YES
SET: $t_D = t_{event}$
IEVF = 1, IPRNTF = 1
INIF = 1
IS THIS A GUIDANCE EVENT
NO
YES
SET: IGUIDF = 1
$t_N = t_{event}$
NAVF = 1
PRINT INDEX = 0
CONTINUED
A


A
CALL SUPPORTING GROUP OF MODULES
IS IGPRF = 1
YES
NO
IS IGUIDF = 1
YES
NO
CALL PØLM
CALL INFM
IS NAVF = 0
YES
NO
CALL HM
IS IEVF = 0
YES
NO
CALL DYNAMICS GROUP OF MODULES
MIDCOURSE CORRECTION UPDATE
CALL INFM
IS MCCF = 0
NO
YES
RETURN TO EXM3 RE-ENTRY

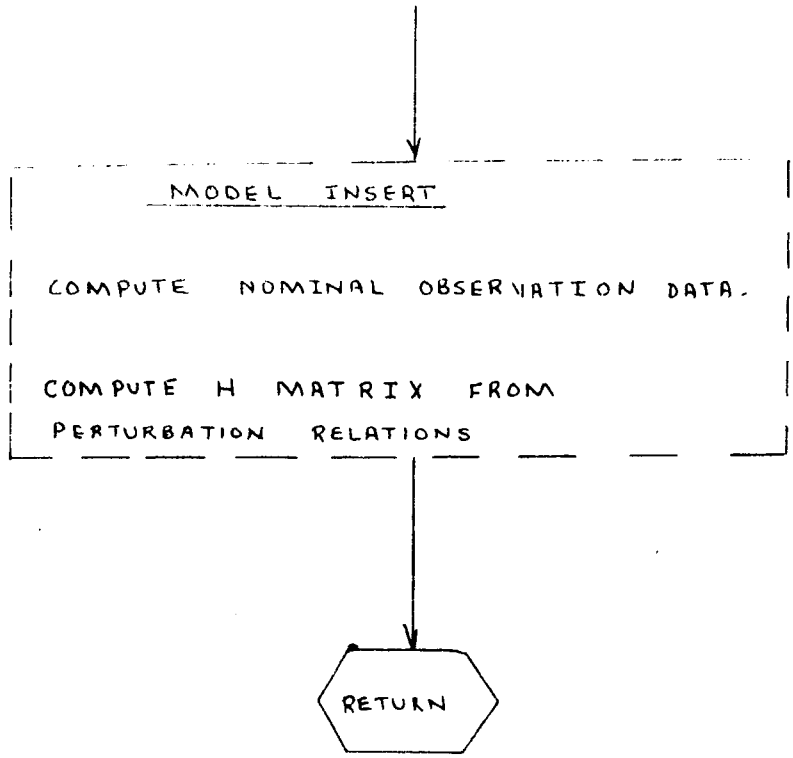
NTM


COMPUTE Δt
MODEL INSERT
ADVANCE NOMINAL TRAJECTORY OVER INTERVAL Δt.
COMPUTE COORDINATE TRANSFORMATIONS NEEDED IN OTHER MODULES
RETURN

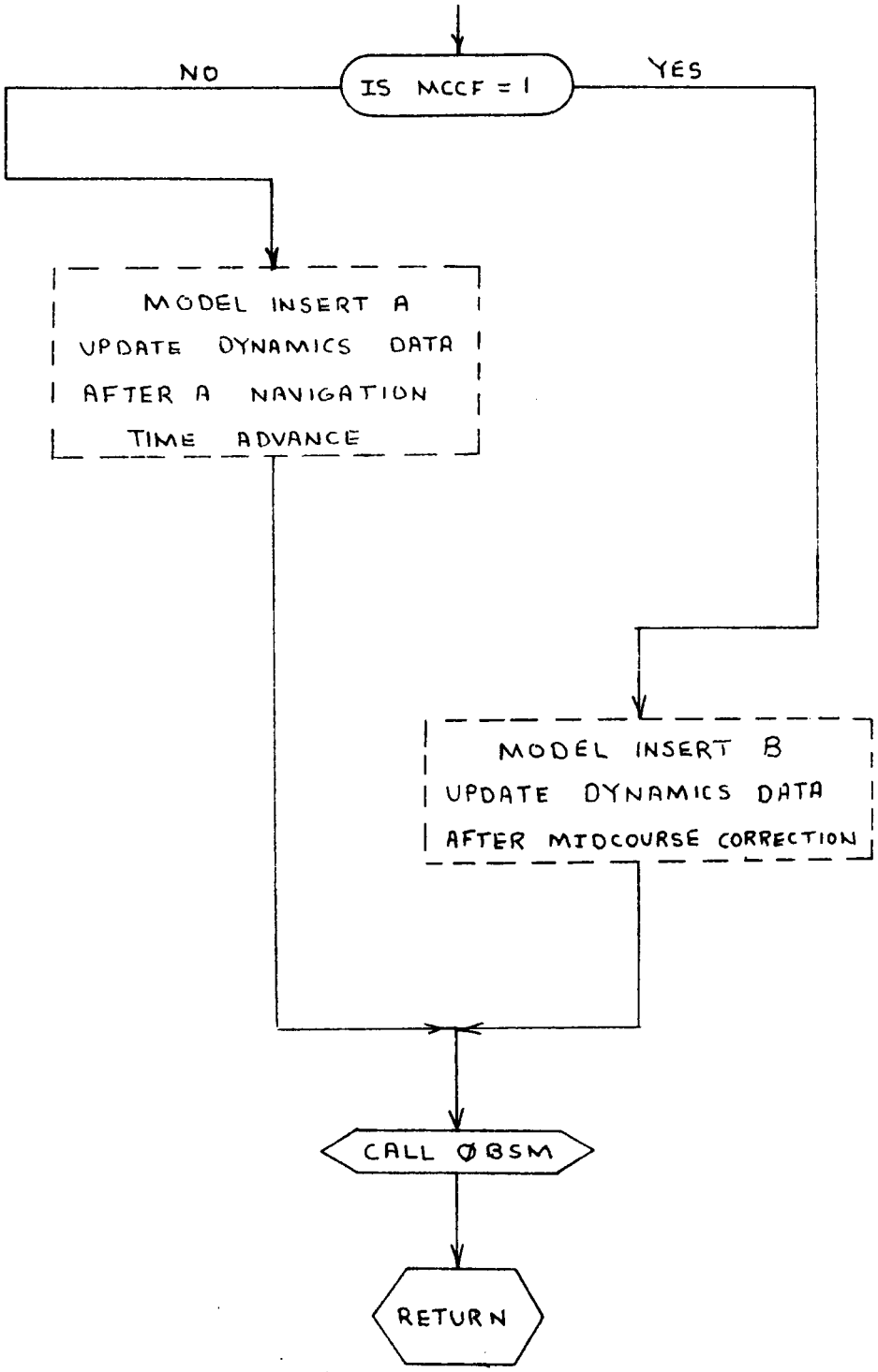
PSIM


IS IRESF = 0
NO
YES
RESET:
Ψ' = I
Q' = 0.0
IRESF = 0
MODEL INSERT
UPDATE:
STATE TRANSITION MATRIX Ψ'
CUMULATIVE DISTURBANCE MATRIX Q'
IS NAVF = 0
NO
YES
IRESF = 1
RETURN

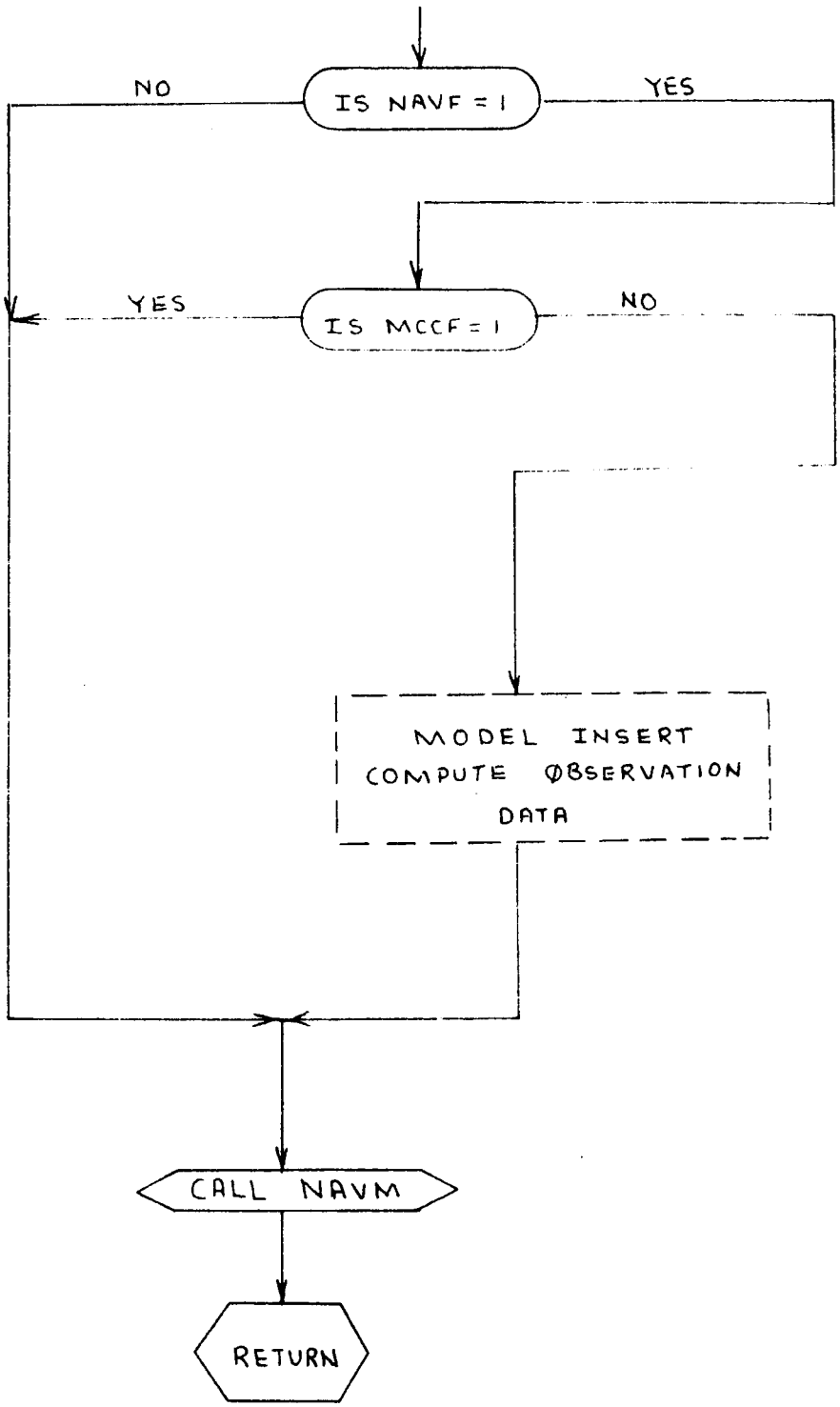
H M


MODEL INSERT
COMPUTE NOMINAL OBSERVATION DATA.
COMPUTE H MATRIX FROM PERTURBATION RELATIONS
RETURN

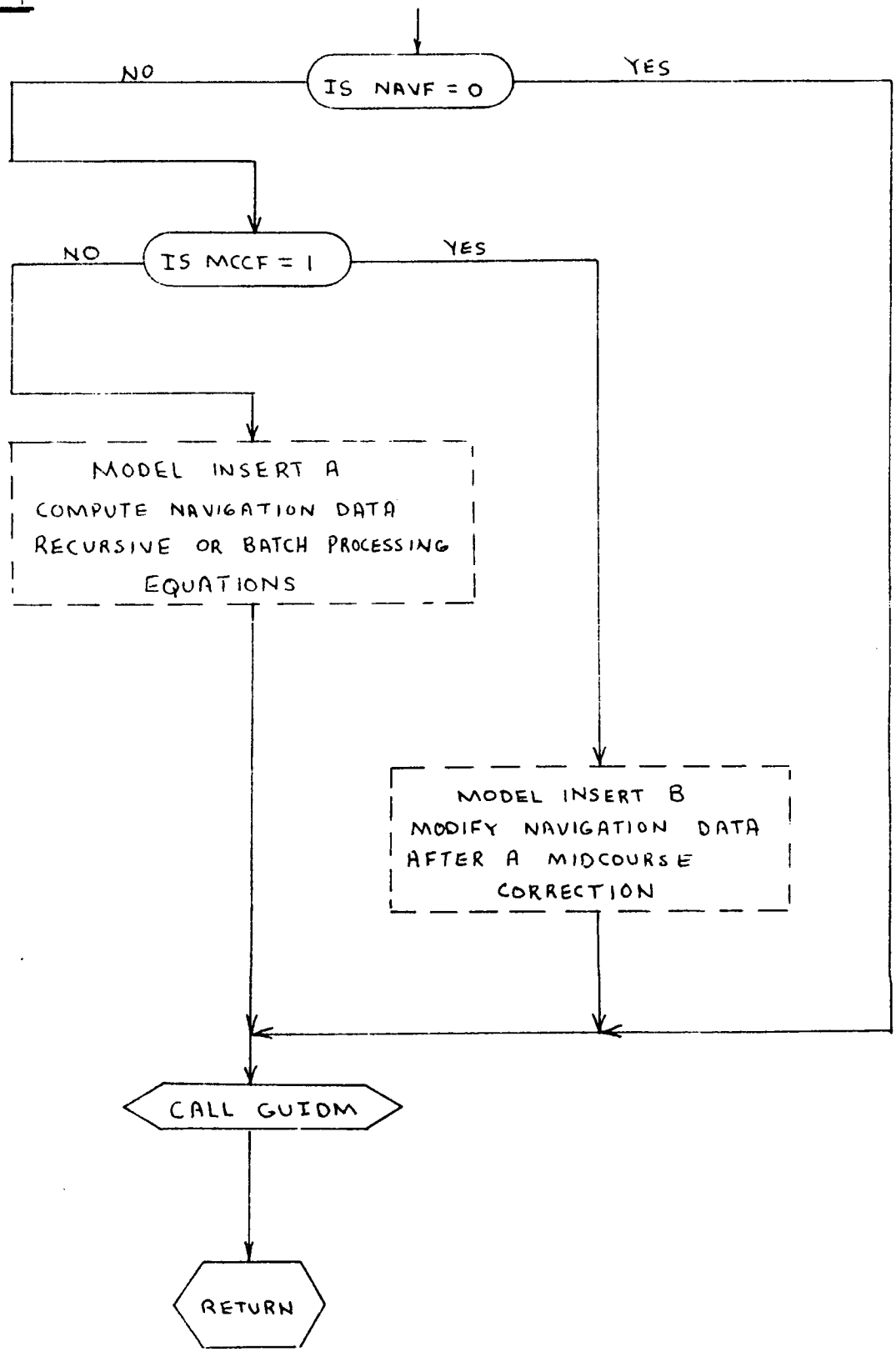
DYNM


IS MCCF = 1
NO
YES
MODEL INSERT A
UPDATE DYNAMICS DATA
AFTER A NAVIGATION
TIME ADVANCE
MODEL INSERT B
UPDATE DYNAMICS DATA
AFTER MIDCOURSE CORRECTION
CALL ØBSM
RETURN

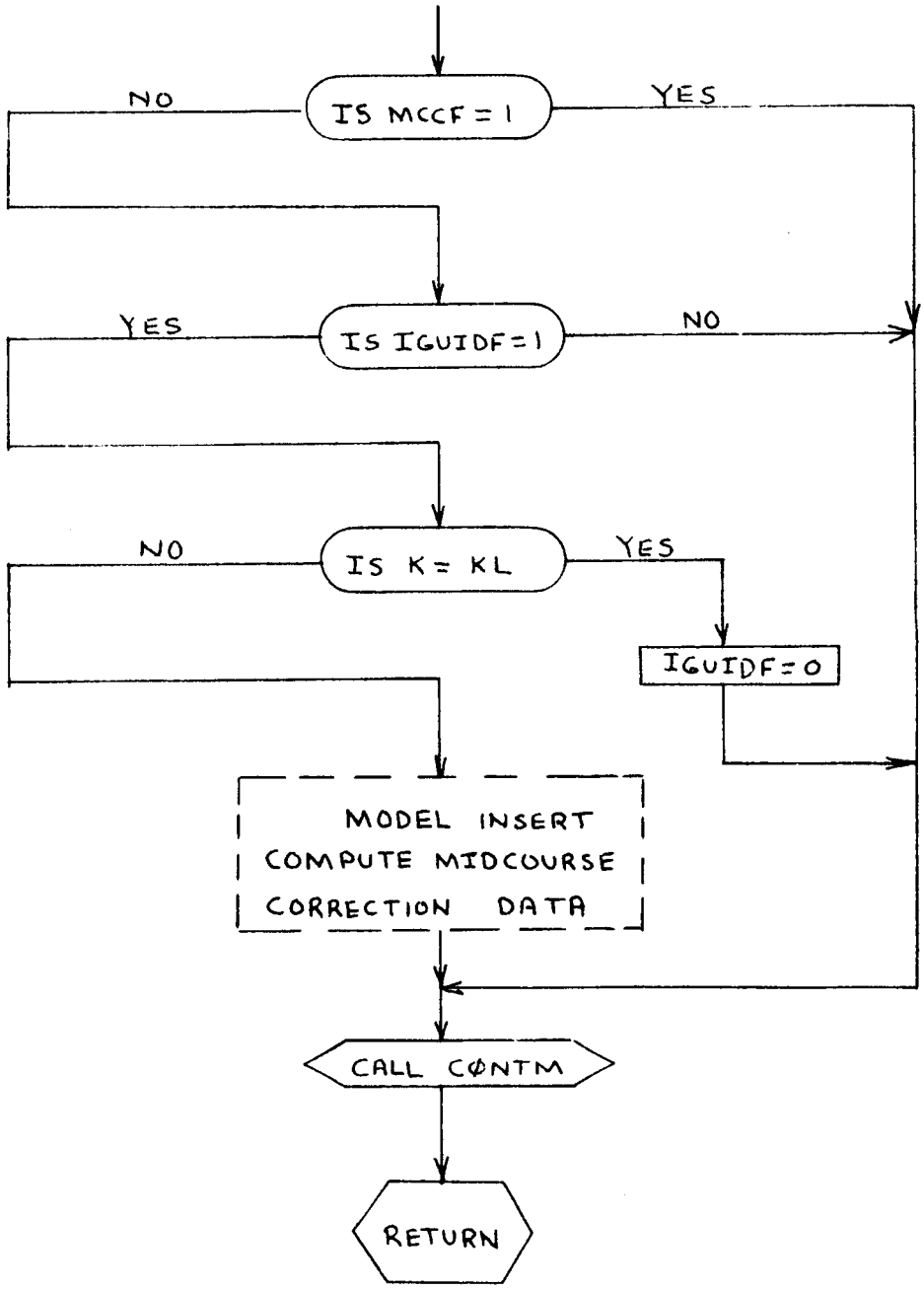
ØRSM


IS NAVF = 1
NO
YES
IS MCCF = 1
YES
NO
MODEL INSERT
COMPUTE ØBSERVATION
DATA
CALL NAVM
RETURN

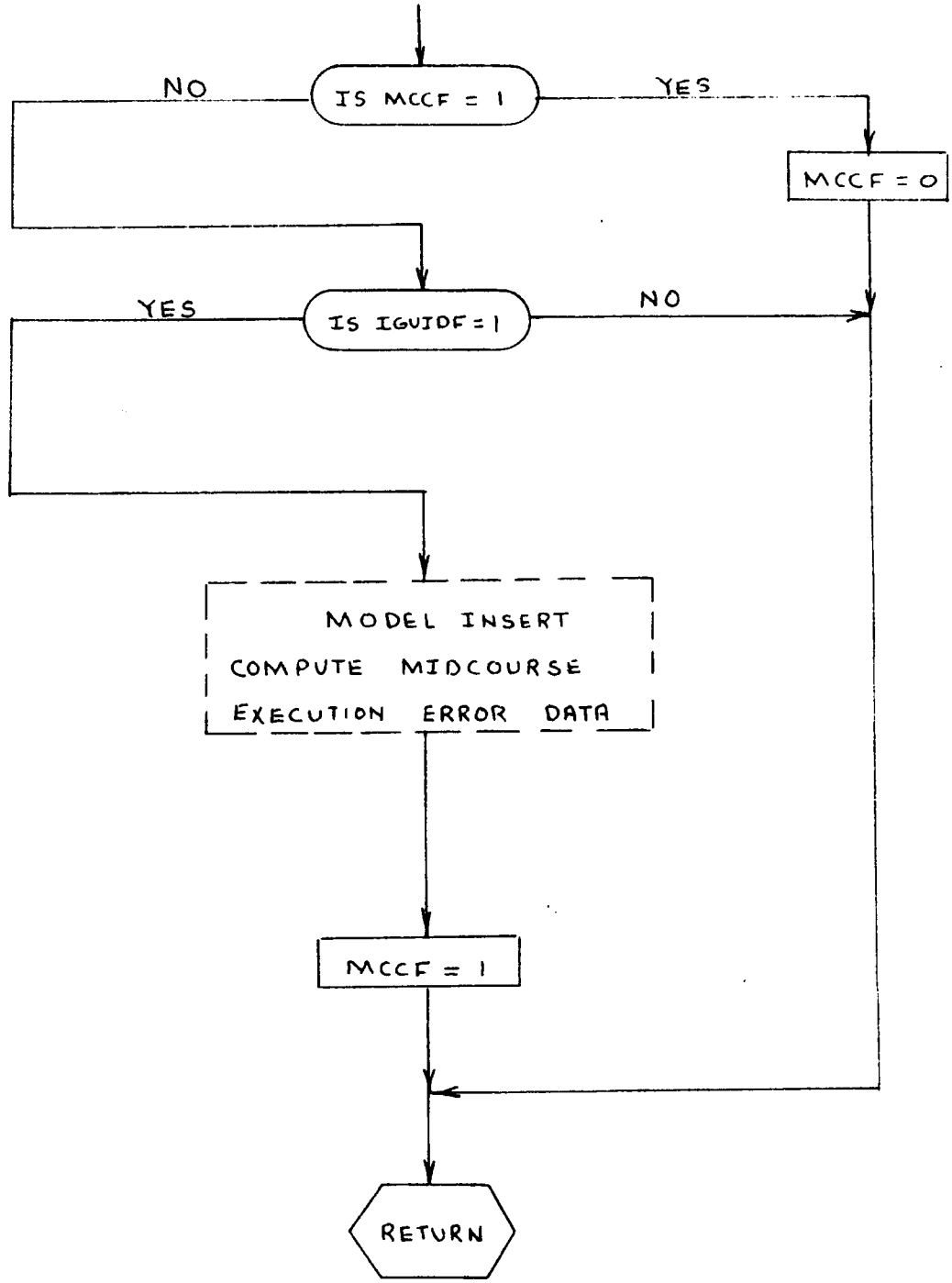
NAVM


IS NAVF = 0
NO
YES
IS MCCF = 1
NO
YES
MODEL INSERT A
COMPUTE NAVIGATION DATA
RECURSIVE OR BATCH PROCESSING
EQUATIONS
MODEL INSERT B
MODIFY NAVIGATION DATA
AFTER A MIDCOURSE
CORRECTION
CALL GUIDM
RETURN

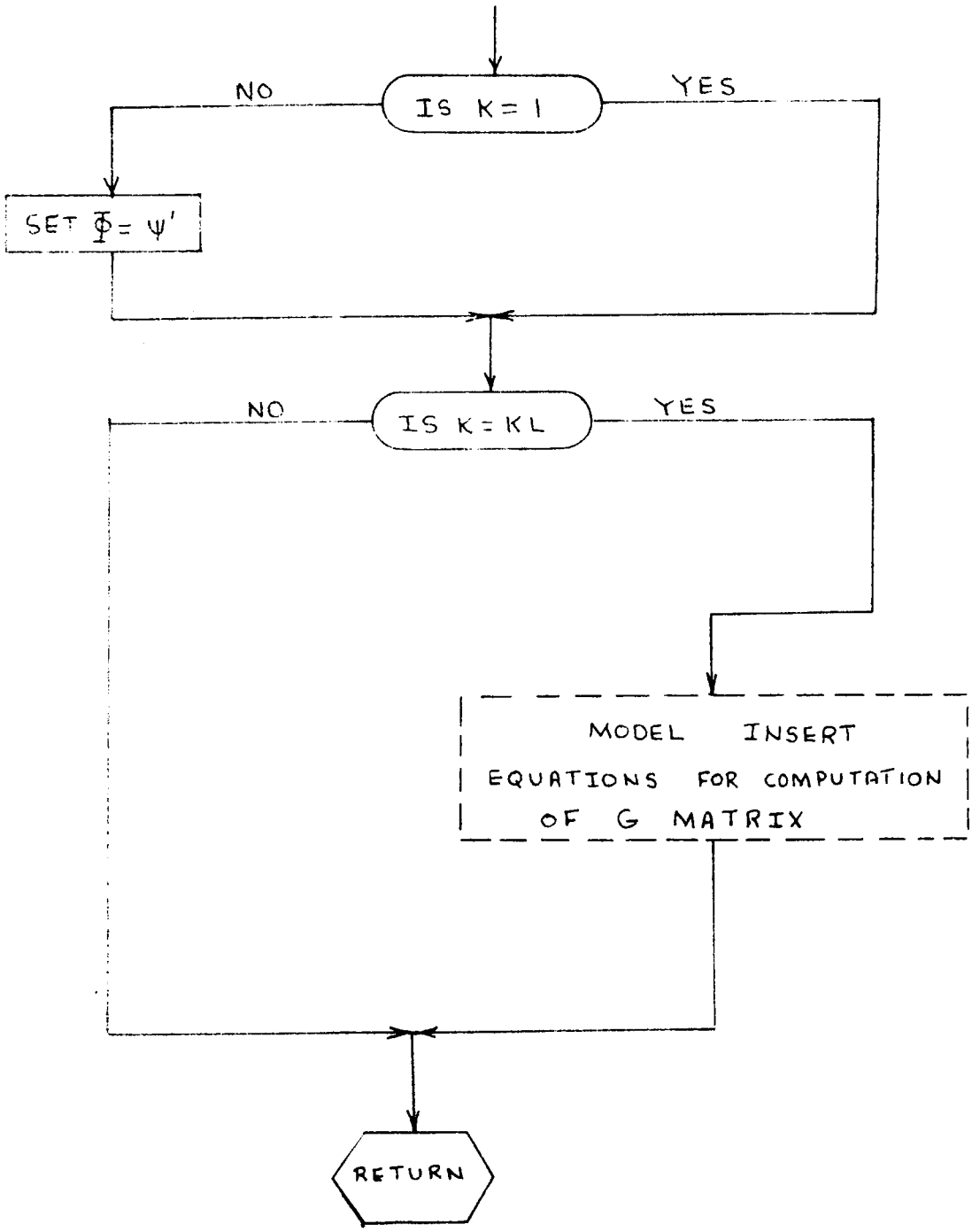
GUIDM


IS MCCF = 1
NO
YES
IS IGUIDF = 1
YES
NO
IS K = KL
NO
YES
IGUIDF = 0
MODEL INSERT
COMPUTE MIDCOURSE
CORRECTION DATA
CALL CØNTM
RETURN

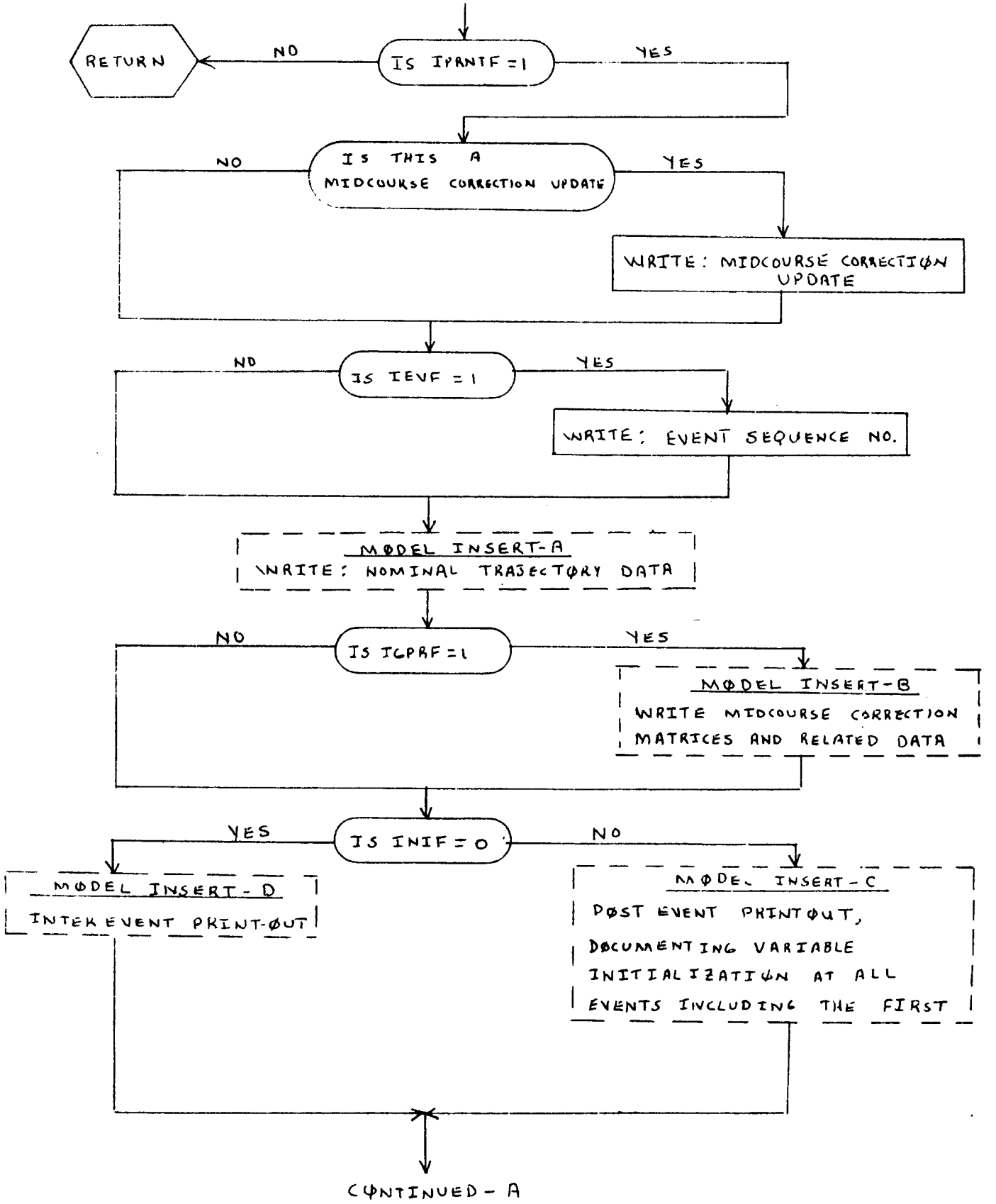
CØNTM

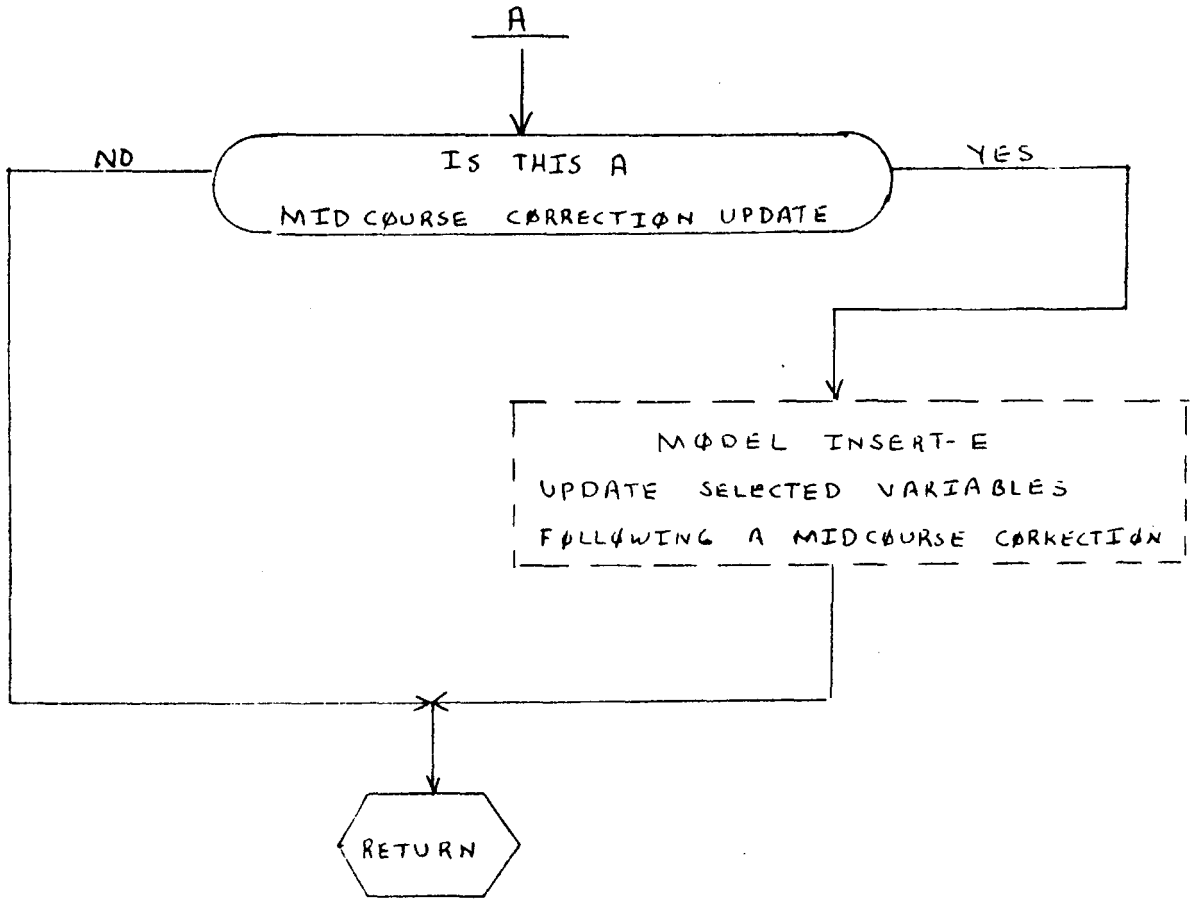

IS MCCF = 1
NO
YES
MCCF = 0
IS IGUIDF = 1
YES
NO
MODEL INSERT
COMPUTE MIDCOURSE
EXECUTION ERROR DATA
MCCF = 1
RETURN

PØLM


IS K = 1
NO
YES
SET Φ = ψ'
IS K = KL
NO
YES
MODEL INSERT
EQUATIONS FOR COMPUTATION
OF G MATRIX
RETURN

INFM

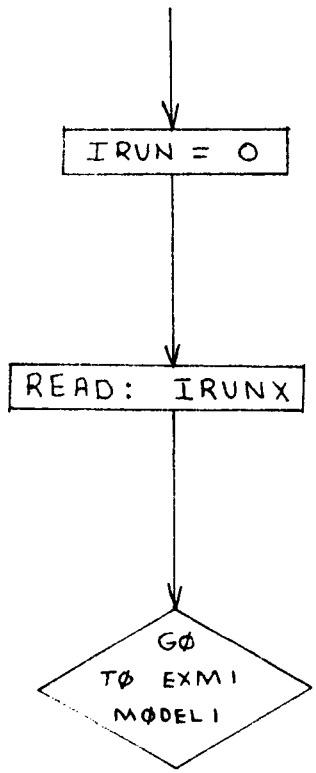

RETURN
NO
IS IPRNTF = 1
YES
NO
IS THIS A MIDCOURSE CORRECTION UPDATE
YES
WRITE: MIDCOURSE CORRECTION UPDATE
NO
IS IEVF = 1
YES
WRITE: EVENT SEQUENCE NO.
MODEL INSERT-A
WRITE: NOMINAL TRAJECTORY DATA
NO
IS IGPRF = 1
YES
MODEL INSERT-B
WRITE MIDCOURSE CORRECTION MATRICES AND RELATED DATA
YES
IS INIF = 0
NO
MODEL INSERT-D
INTER EVENT PRINT-OUT
MODEL INSERT-C
POST EVENT PRINTOUT, DOCUMENTING VARIABLE INITIALIZATION AT ALL EVENTS INCLUDING THE FIRST
CONTINUED-A


A
IS THIS A
MID COURSE CORRECTION UPDATE
NO
YES
MODEL INSERT-E
UPDATE SELECTED VARIABLES
FOLLOWING A MIDCOURSE CORRECTION
RETURN

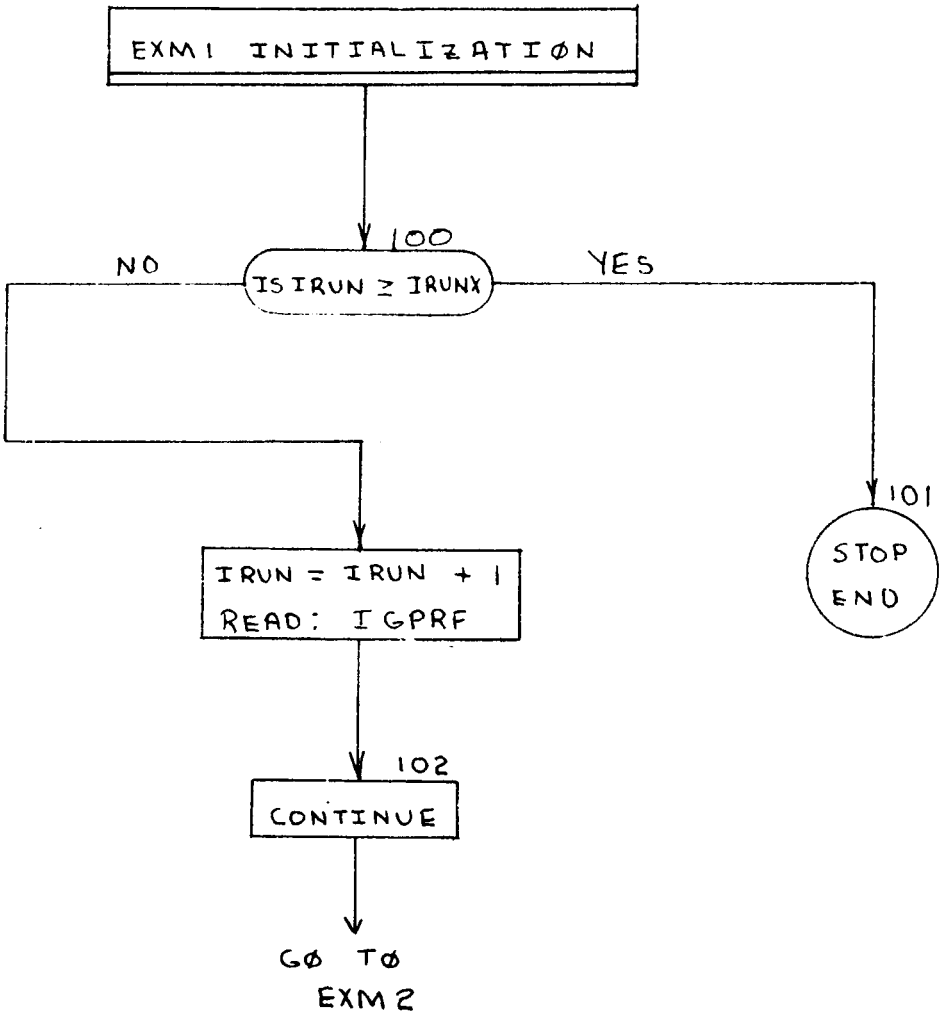
APPENDIX 2

Flow Charts and Listings for the Tutorial Program

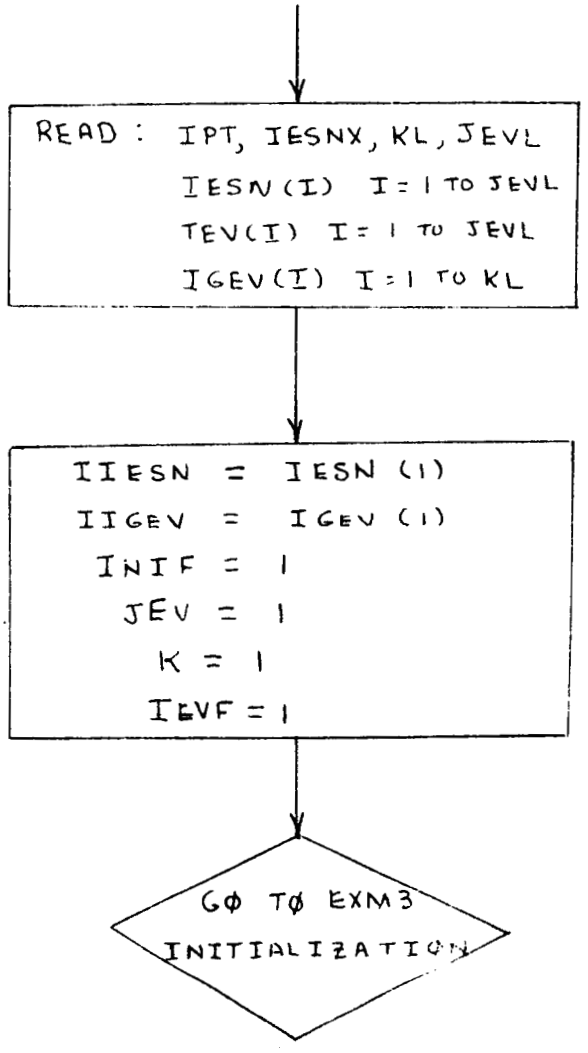
EXMI INITIALIZATIØN



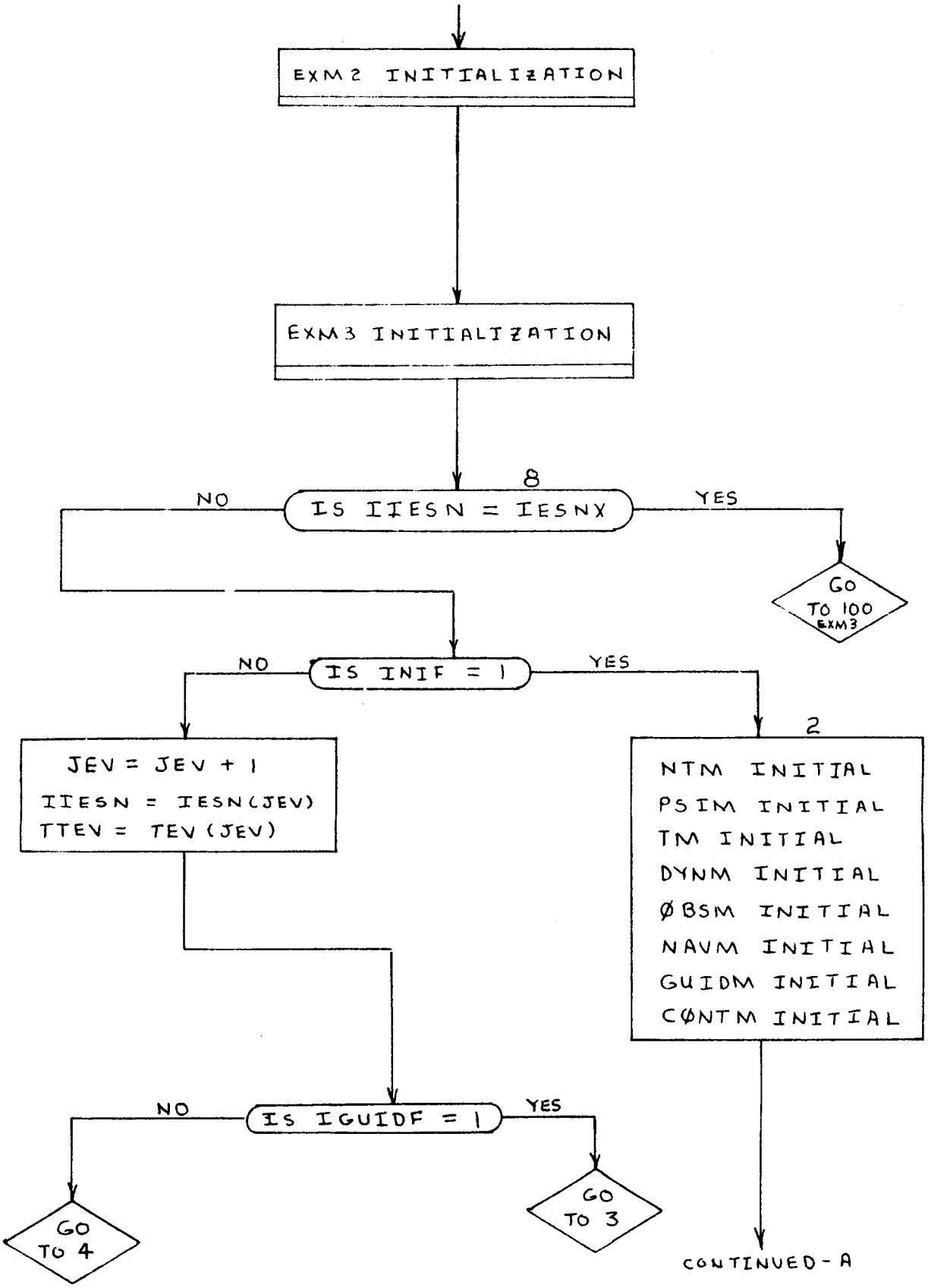
EXM1

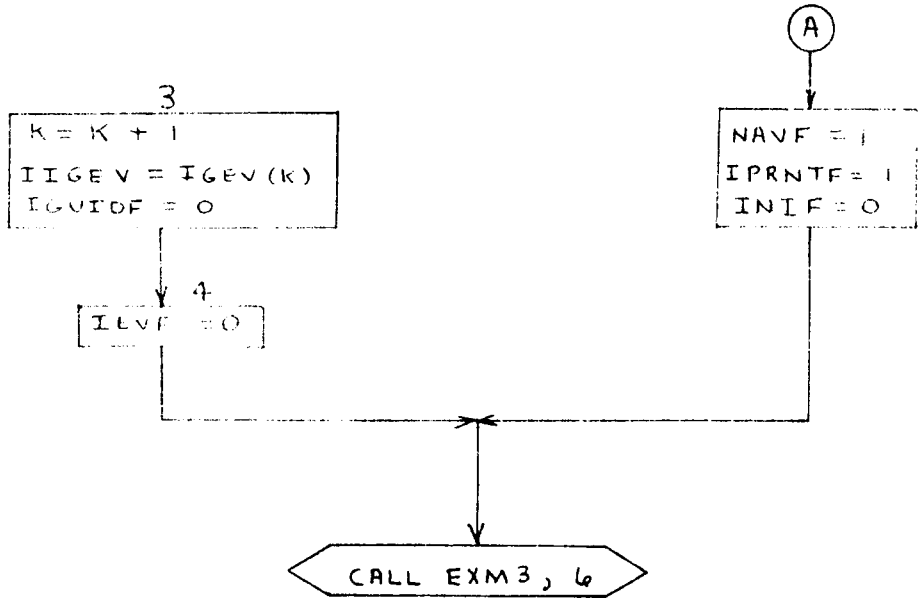

EXM1 INITIALIZATIØN
100
NO
IS IRUN ≥ IRUNX
YES
101
STOP
END
IRUN = IRUN + 1
READ: IGPRF
102
CONTINUE
GØ TØ
EXM2

EXM2 INITIALIZATIØN

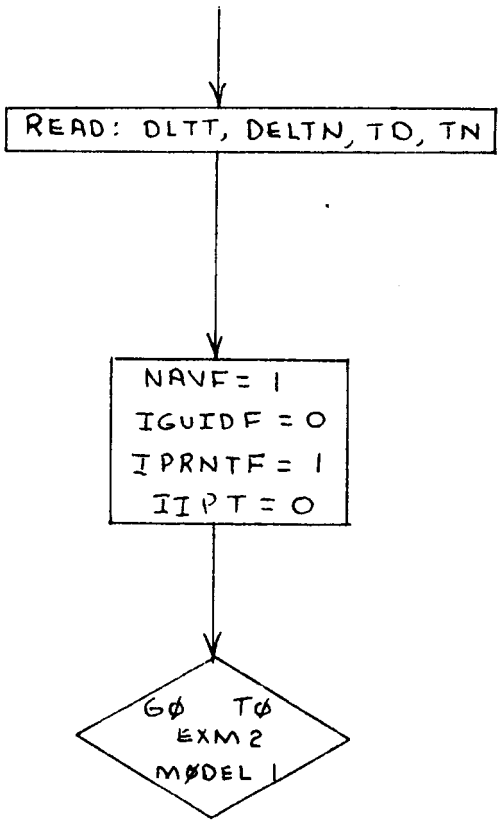


EXM2

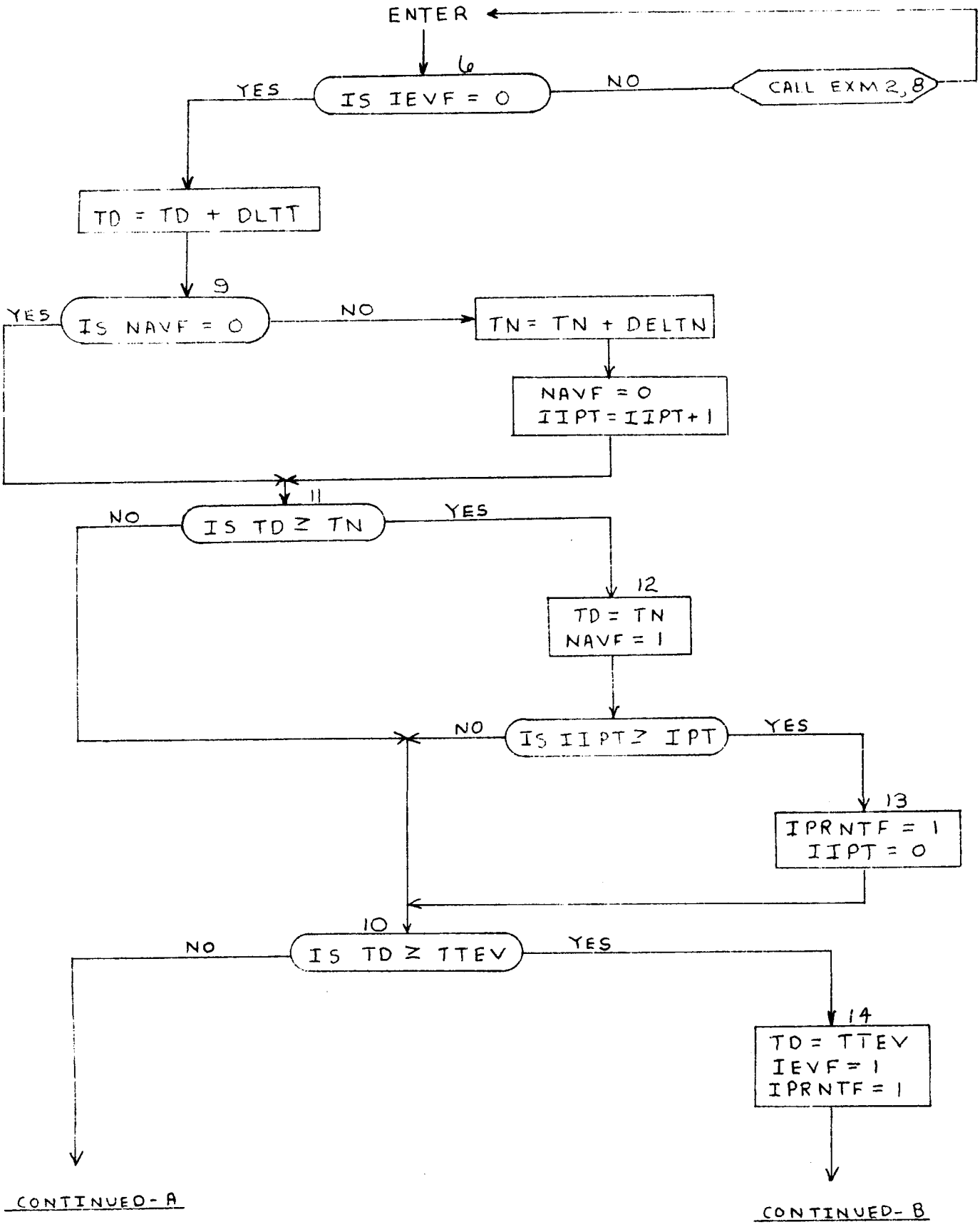

EXM2 INITIALIZATION
EXM3 INITIALIZATION
8
IS IIESN = IESNX
NO
YES
GO TO 100 EXM3
IS INIF = 1
NO
YES
JEV = JEV + 1
IIESN = IESN(JEV)
TTEV = TEV(JEV)
2
NTM INITIAL
PSIM INITIAL
TM INITIAL
DYNM INITIAL
ØBSM INITIAL
NAVM INITIAL
GUIDM INITIAL
CØNTM INITIAL
IS IGUIDF = 1
NO
YES
GO TO 4
GO TO 3
CONTINUED - A

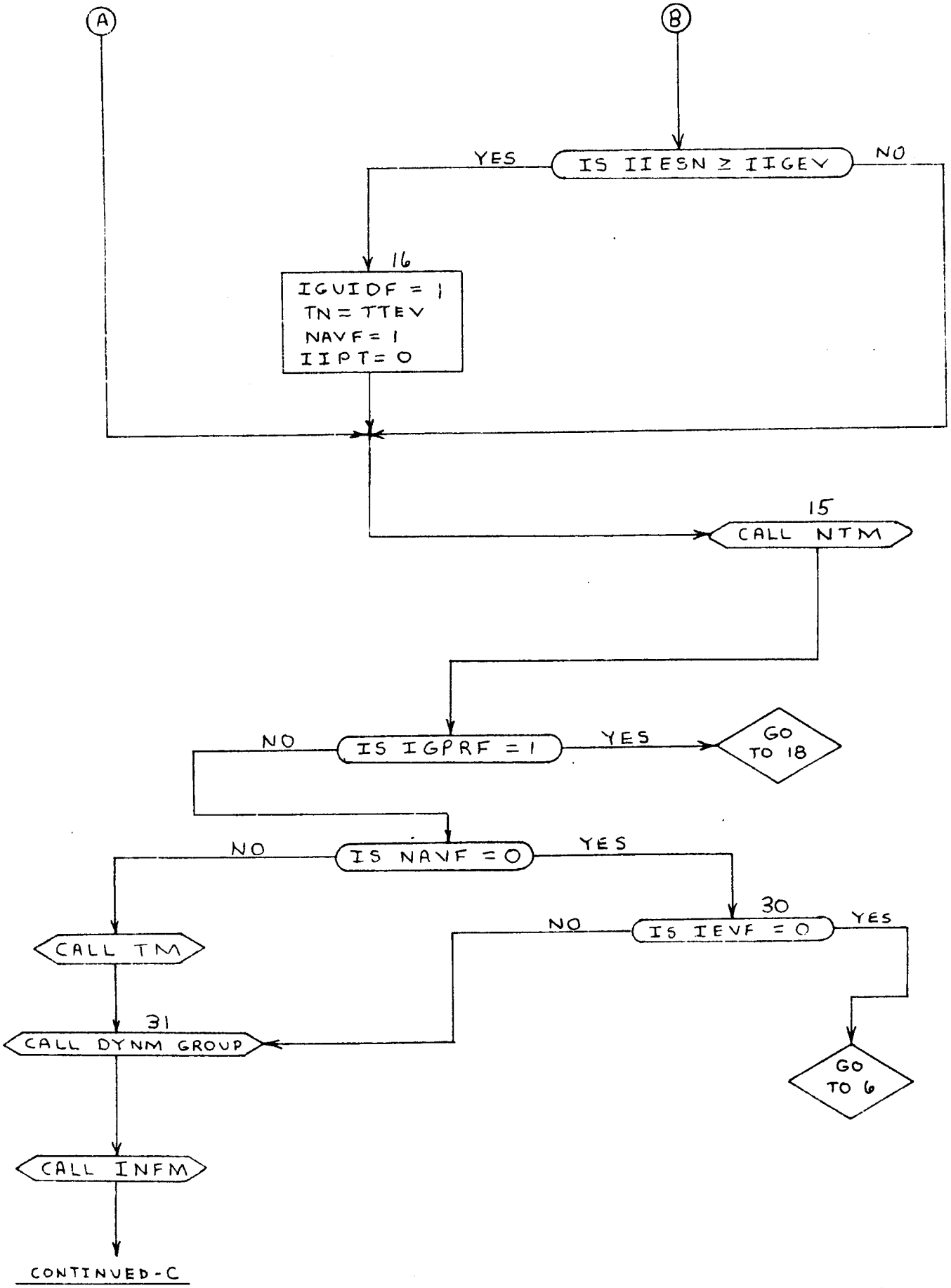

A
3
K = K + 1
IIGEV = IGEV(K)
IGUIDF = 0
NAVF = 1
IPRNTF = 1
INIF = 0
4
ILVF = 0
CALL EXM3, 6

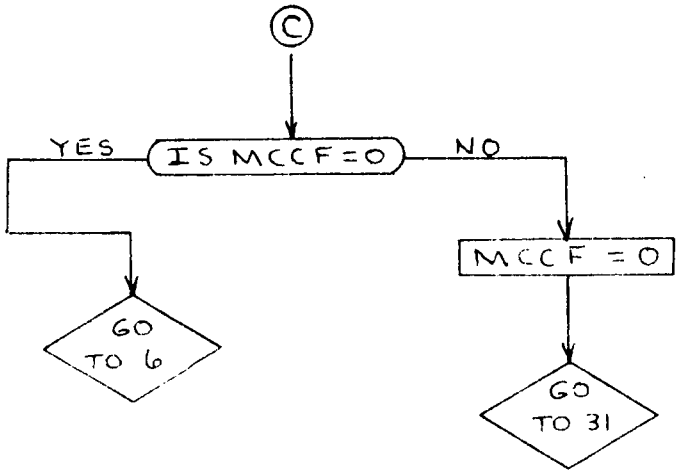
EXM3 INITIALIZATIØN

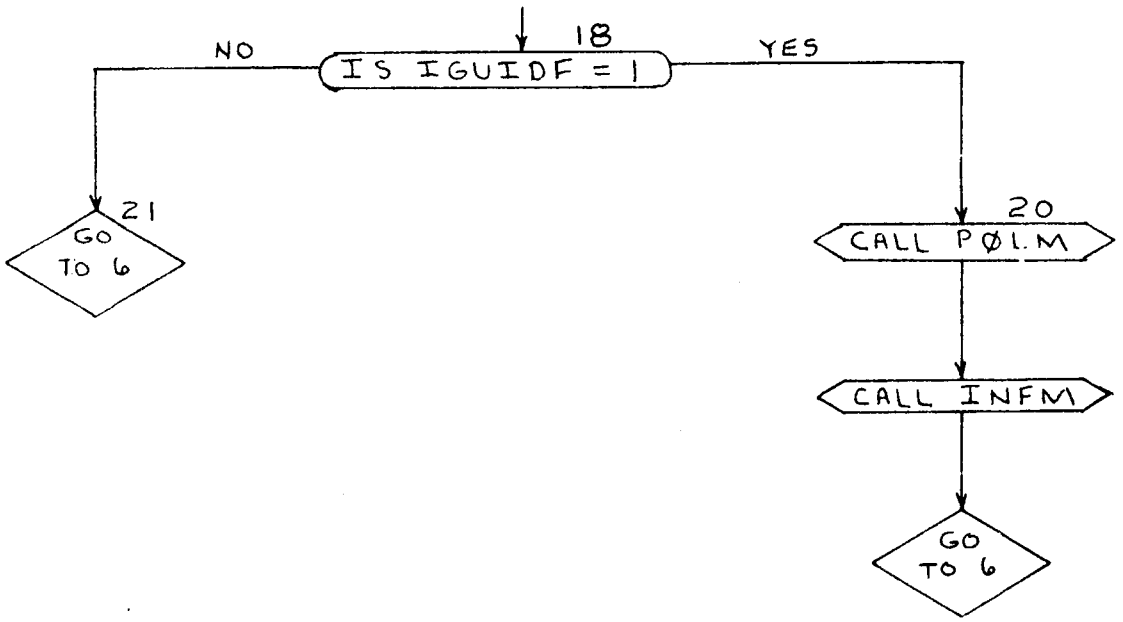


EXM3

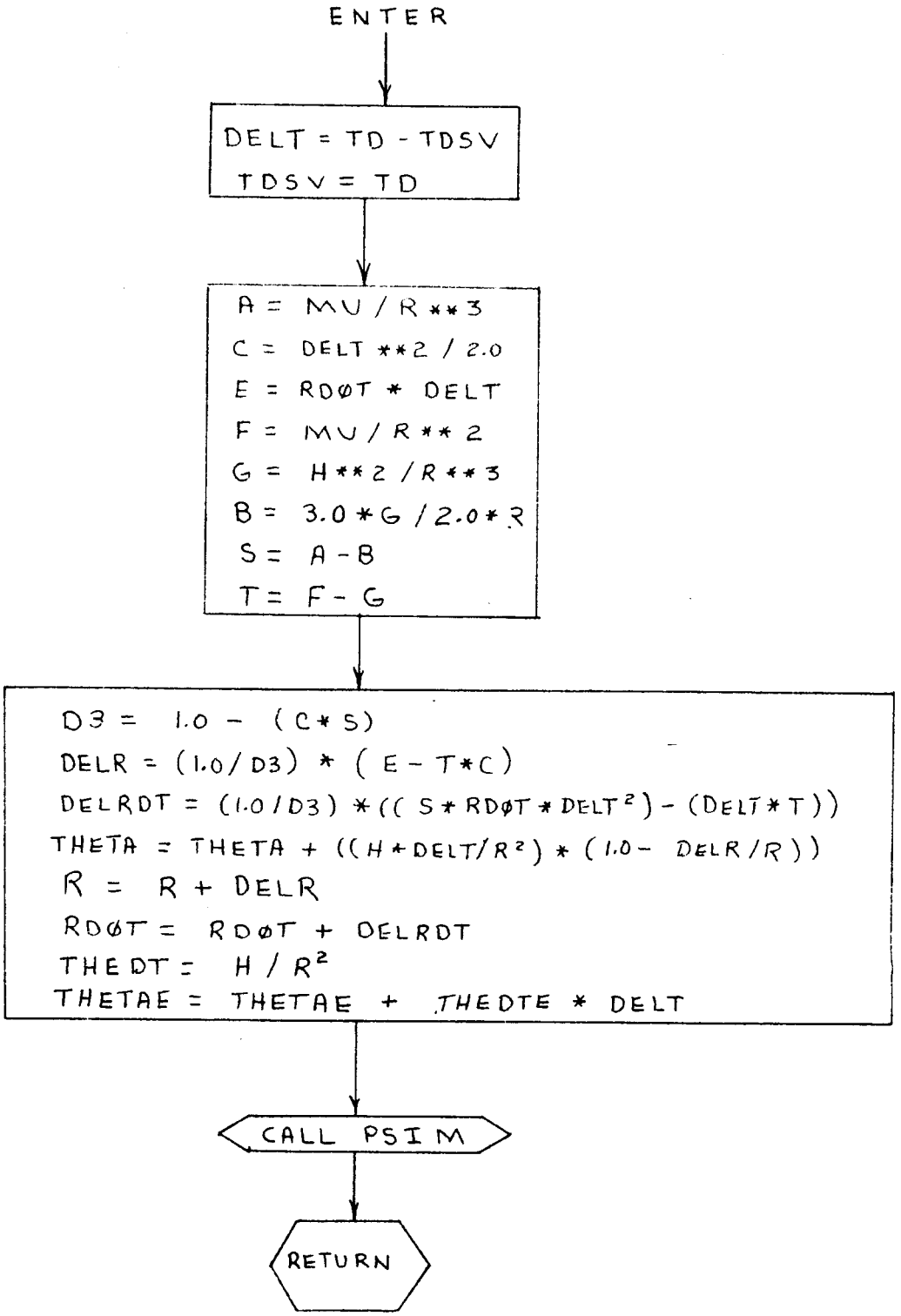

ENTER
6
IS IEVF = 0
YES
NO
CALL EXM2,8
TD = TD + DLTT
9
IS NAVF = 0
YES
NO
TN = TN + DELTN
NAVF = 0
IIPT = IIPT + 1
11
IS TD ≥ TN
NO
YES
12
TD = TN
NAVF = 1
IS IIPT ≥ IPT
NO
YES
13
IPRNTF = 1
IIPT = 0
10
IS TD ≥ TTEV
NO
YES
14
TD = TTEV
IEVF = 1
IPRNTF = 1
CONTINUED-A
CONTINUED-B


A
B
IS IIESN ≥ IIGEV
YES
NO
16
IGUIDF = 1
TN = TTEV
NAVF = 1
IIPT = 0
15
CALL NTM
IS IGPRF = 1
NO
YES
GO TO 18
IS NAVF = 0
NO
YES
30
IS IEVF = 0
NO
YES
CALL TM
31
CALL DYNM GROUP
GO TO 6
CALL INFM
CONTINUED-C

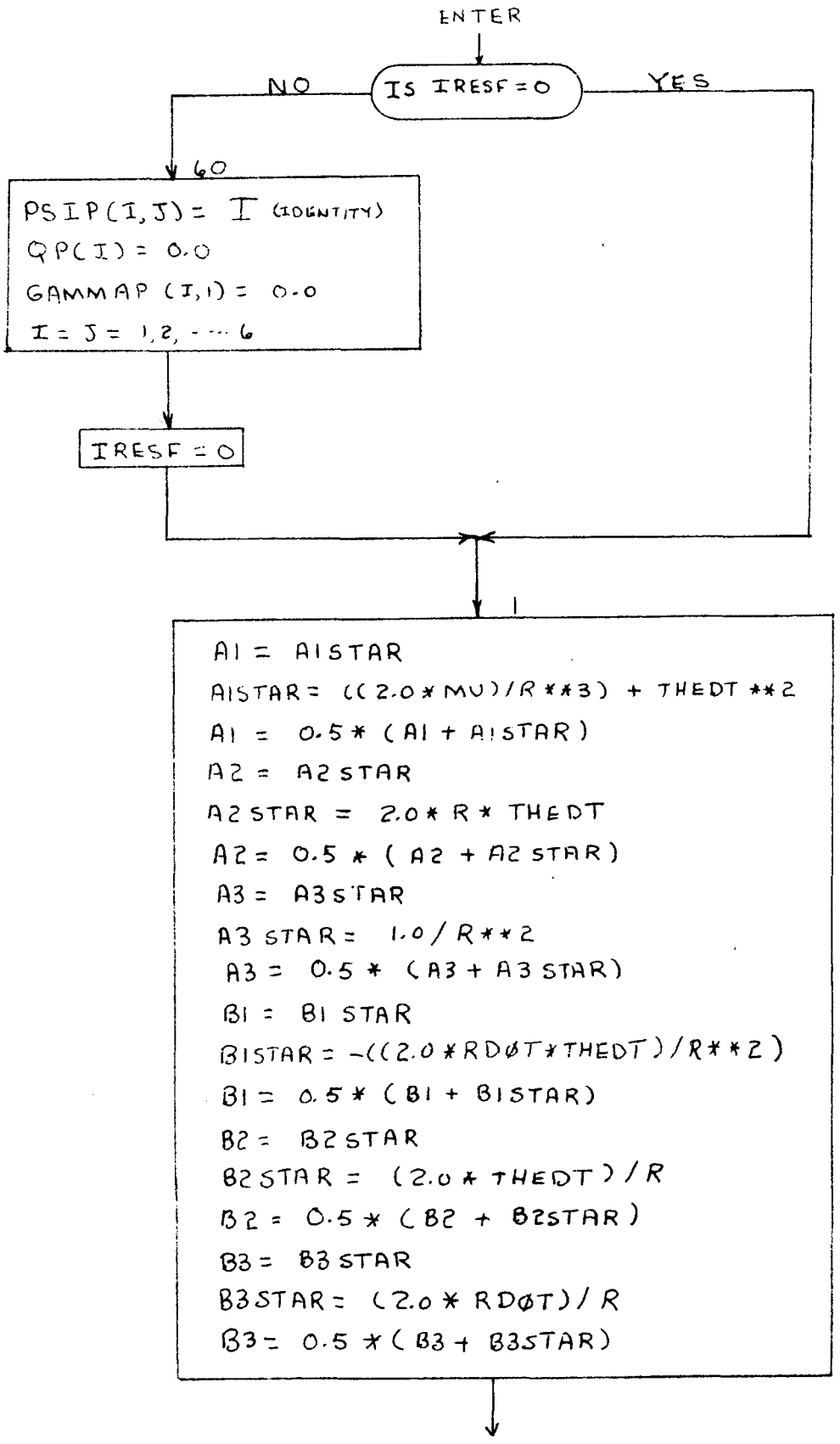

C
YES
IS MCCF=0
NO
MCCF = 0
GO TO 6
GO TO 31


18
NO
IS IGUIDF = 1
YES
21
GO TO 6
20
CALL PØI.M
CALL INFM
GO TO 6

NTM


ENTER
DELT = TD - TDSV
TDSV = TD
A = MU / R**3
C = DELT**2 / 2.0
E = RDØT * DELT
F = MU / R** 2
G = H**2 / R**3
B = 3.0*G / 2.0*R
S = A - B
T = F - G
D3 = 1.0 - (C*S)
DELR = (1.0/D3) * (E - T*C)
DELRDT = (1.0/D3) *((S*RDØT*DELT²) - (DELT*T))
THETA = THETA + ((H*DELT/R²) * (1.0- DELR/R))
R = R + DELR
RDØT = RDØT + DELRDT
THEDT = H / R²
THETAE = THETAE + THEDTE * DELT
CALL PSIM
RETURN

PSTM



CONTINUED - A

A

$$D_1 = -\left[1 + \frac{b_3}{2}\Delta t + (b_2 a_2 - a_1)\,\Delta t^2/4 + (b_1 a_2 - a_1 b_3)\,\Delta t^3/8\right]$$

$$A_1 = 1/D_1\left[-a_1 \Delta t + (a_2 b_1 - a_1 b_3)\,\Delta t^2/2\right]$$

$$A_2 = 1/D_1\left[(a_2 b_2 - a_1)\,\Delta t^2/2 + (a_2 b_1 - a_1 b_3)\,\Delta t^3/4\right]$$

$$A_3 = -\,a_2\,\Delta t / D_1$$

$$A_4 = 1/D_1\left[a_3 \Delta t + a_3 b_3\,\Delta t^2/2\right]$$

$$B_1 = 1/D_1\left[b_1 \Delta t + a_1 b_2\,\Delta t^2/2\right]$$

$$B_2 = 1/D_1\left[b_2 \Delta t + b_1\,\Delta t^2/2\right]$$

$$B_3 = 1/D_1\left[b_3 \Delta t + a_2 b_2\,\Delta t^2/2 + (a_2 b_1 - a_1 b_3)\,\Delta t^3/4\right]$$

$$B_4 = 1/D_1\left[-a_3 b_2\,\Delta t^2/2 - a_3 b_1\,\Delta t^3/4\right]$$

```
C = CSTAR
CSTAR = MU/R**3
C = 0.5 * (C + CSTAR)
```

$$D_2 = 1 + C\,\Delta t^2/4$$

$$C_1 = -\,C\,\Delta t^2/2\,D_2$$

$$C_2 = \Delta t / D_2$$

$$C_3 = -\,C\,\Delta t / D_2$$

PSI ARRAY (6×6)

```
PSI(1,1) = 1.0 + (CAPA1 * DELT)/2.0
PSI(1,4) = (CAPA2/2.0 + 1.0) * DELT
PSI(1,5) = (CAPA3 * DELT)/2.0 * .001
PSI(2,1) = ((CAPB1 * DELT)/2.0) * 1000.
PSI(2,2) = 1.0
PSI(2,4) = ((CAPB2 * DELT)/2.0) * 1000.
PSI(2,5) = (CAPB3/2.0 + 1.0) * DELT
PSI(3,3) = 1.0 + CAPC1
PSI(3,6) = CAPC2
PSI(4,1) = CAPA1
PSI(4,4) = CAPA2 + 1.0
PSI(4,5) = CAPA3 * 0.001
```

CONTINUED-B

B

```
PSI(5,1) =  CAPB1 * 1000.
PSI(5,4) =  CAPB2 * 1000.
PSI(5,5) =  1.0 + CAPB3
PSI(6,3) =  CAPC3
PSI(6,6) =  1.0 + CAPC1
  ALL OTHER  PSI(I,J) = 0.0
```

```
        GAMMA ARRAY (6x1)
GAMMA(1,1) = ((CAPA4 * DELT) / 2.0
GAMMA(2,1) = (((CAPB4 * DELT) / 2.0) * 1000.
GAMMA(3,1) = 0.0
GAMMA(4,1) = CAPA4
GAMMA(5,1) = CAPB4 * 1000.
GAMMA(6,1) = 0.0
```

50

```
SIPP(I,J) = SIDP(I,J) + PSI(I,MM) * PSIP(MM,J)
PSIP(I,J) = SIDP(I,J)
I = J = MM = 1, 2, 3 ------ 6
```

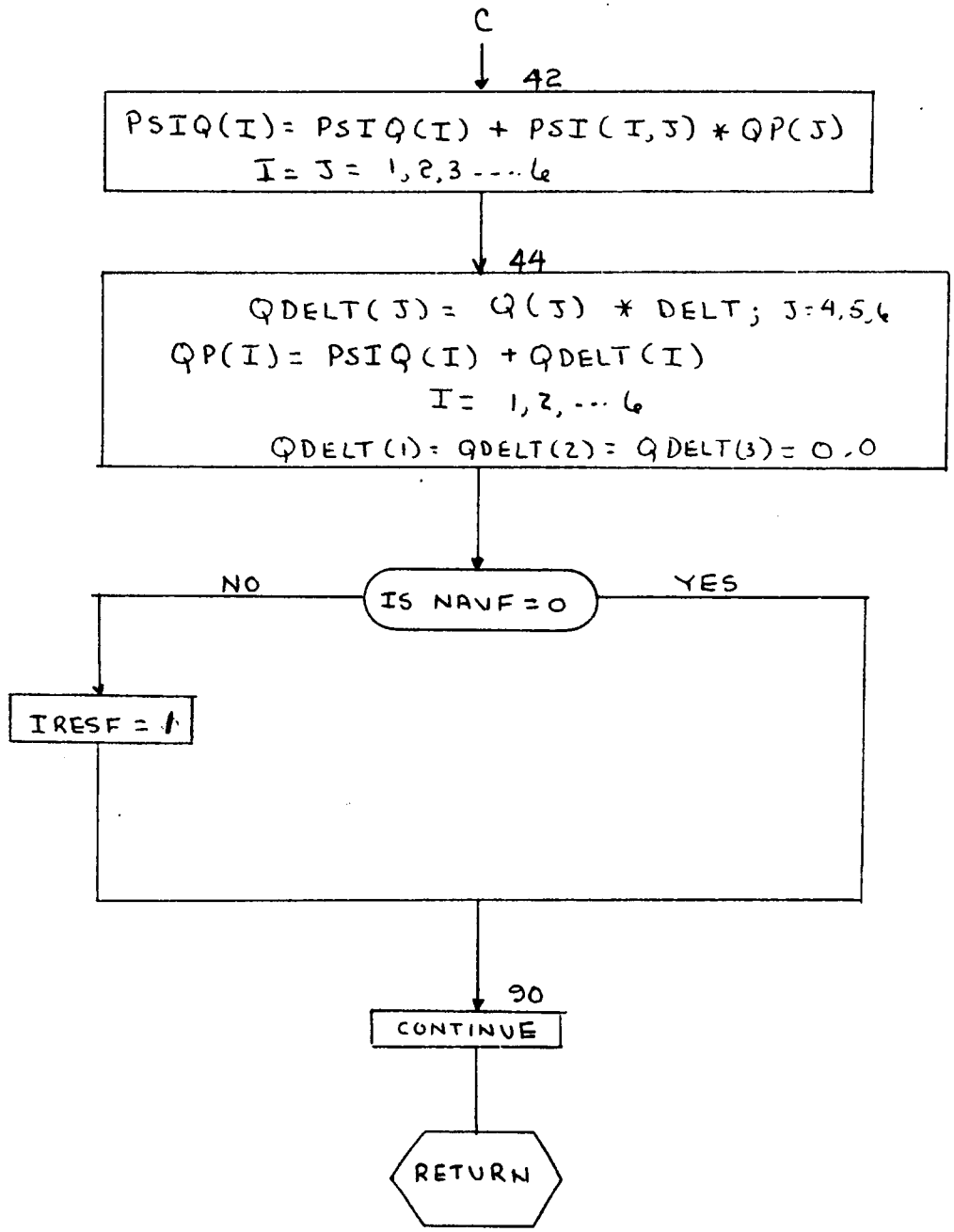
69

```
SIGAMP(I,1) = SIGAMP(I,1) + PSI(I,MM) * GAMMAP(MM,1)
GAMMAP(I,1) = SIGAMP(I,1) + GAMMA(I,1)
I = 1, 2, --- 6
```

40

```
Q(I) = SIGMQ(J) * ((SQRT(ALOG(1./RNUM(XX)**2))) * (SIN(TWOPI
       * RNUM(XX))))                                J = 1, --3   I = J + 3
```

CONTINUED - C


C
42
PSIQ(I) = PSIQ(I) + PSI(I,J) * QP(J)
I = J = 1,2,3 ---- 6
44
QDELT(J) = Q(J) * DELT ; J=4,5,6
QP(I) = PSIQ(I) + QDELT(I)
I = 1,2, --- 6
QDELT(1) = QDELT(2) = QDELT(3) = 0.0
NO
IS NAVF = 0
YES
IRESF = 1
90
CONTINUE
RETURN

TM

ENTER

```
A = THETAE - THETA
B = CØS (A)
C = SIN (A)
D = THEDTE - THEDT
```

```
RNG = [R² + RE² - 2.0 * R * RE * B ]^1/2
P   =   1.0 / RNG
RNGDT = P * [(R - RE * B) * RDØT + R * RE * C * D ]
```

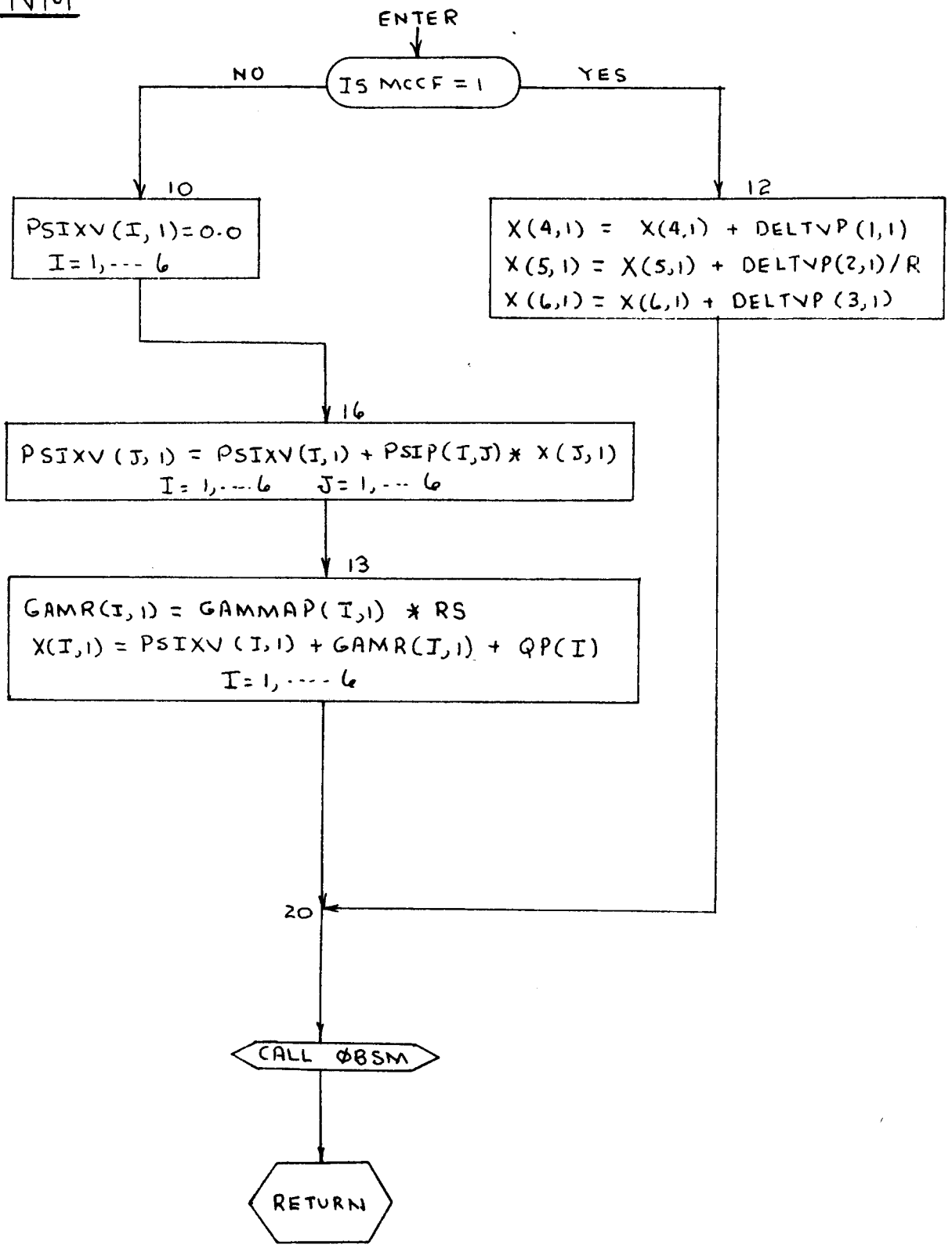
```
SALPHA =   R * C / RNG
CALPHA = (R² - RE² - RNG²) / (2.0 * RE * RNG)
```

```
E1 =  P * ( R - RE * B)
E2 =  (-P) * ( R * RE * C )
F1 = P * ((-RNGDT * E1) + RDØT + RE * C * D )
F2 = P * ((-RNGDT * E2) - (RDØT * RE * C) - R * RE * B * D)
G1 = ( C / (RNG * CALPHA)) * (1.0 - (E1 * R) / RNG)
G2 = (-R / (RNG * CALPHA)) * (B + (E2 * C) / RNG)
```

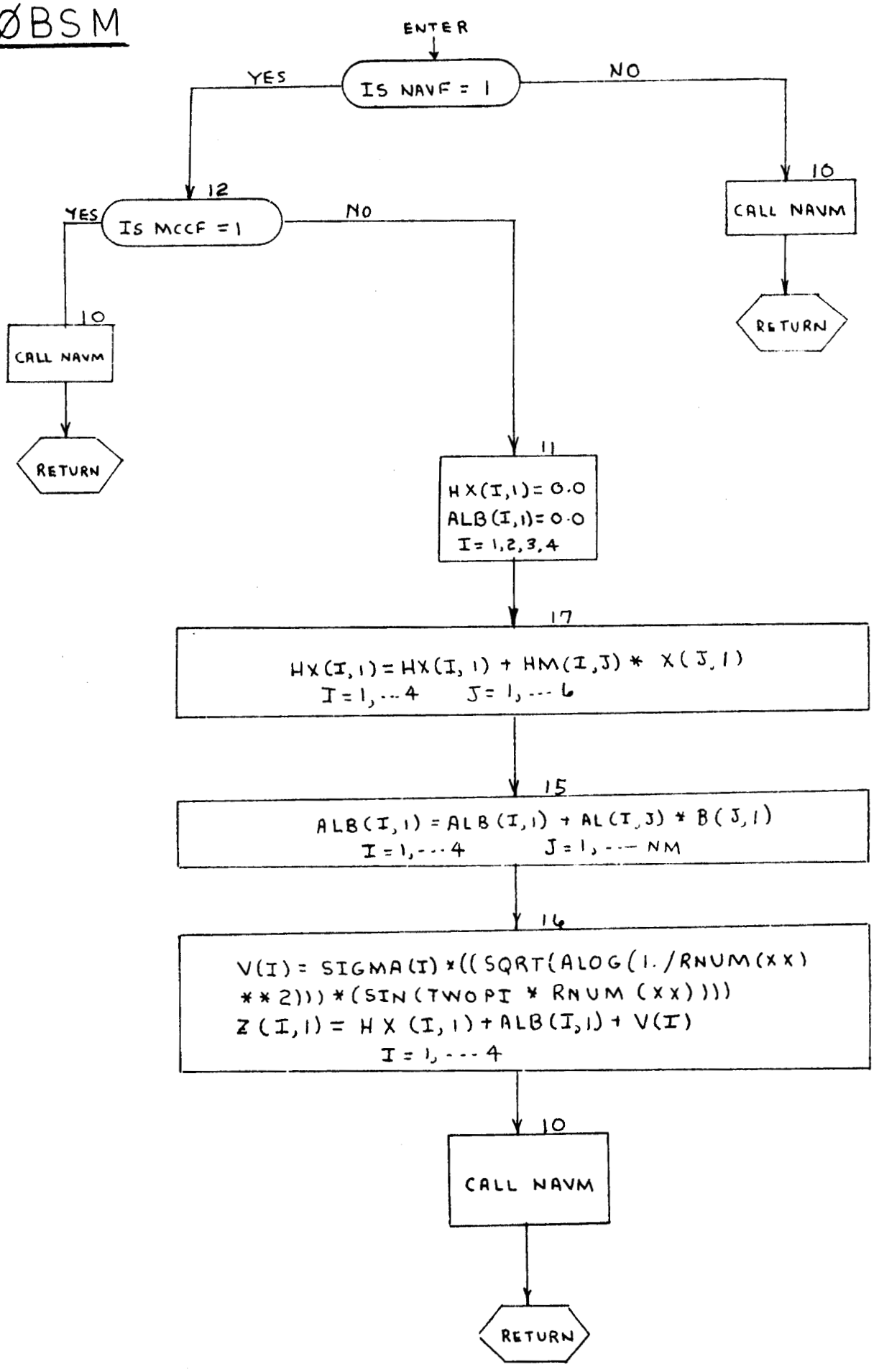
```
HM (1,1) = HM (2,4) =  E1  ;  HM (2,1) = F1
HM (1,2) = HM (2,5) = E2 * .001 ; HM (2,2) = F2 * .001
HM (3,1) = G1 * 1000 ; HM (3,2) = G2  ;  HM (4,3) = P * 1000
 ALL  OTHER  HM (I,J) = 0.0
     I = 1,2,3,4    J = 1,2, ---- 6
```

RETURN

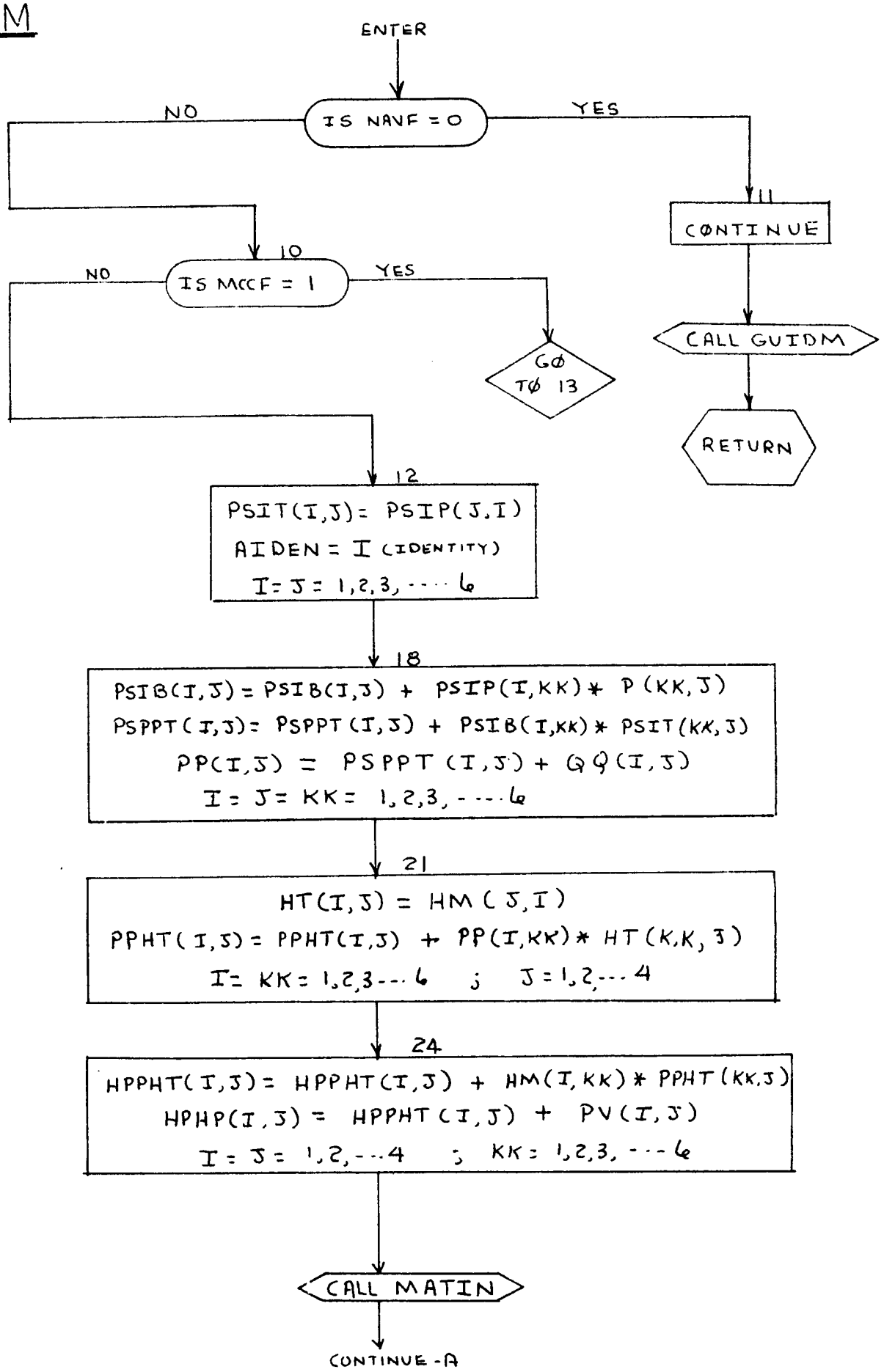
DYNM

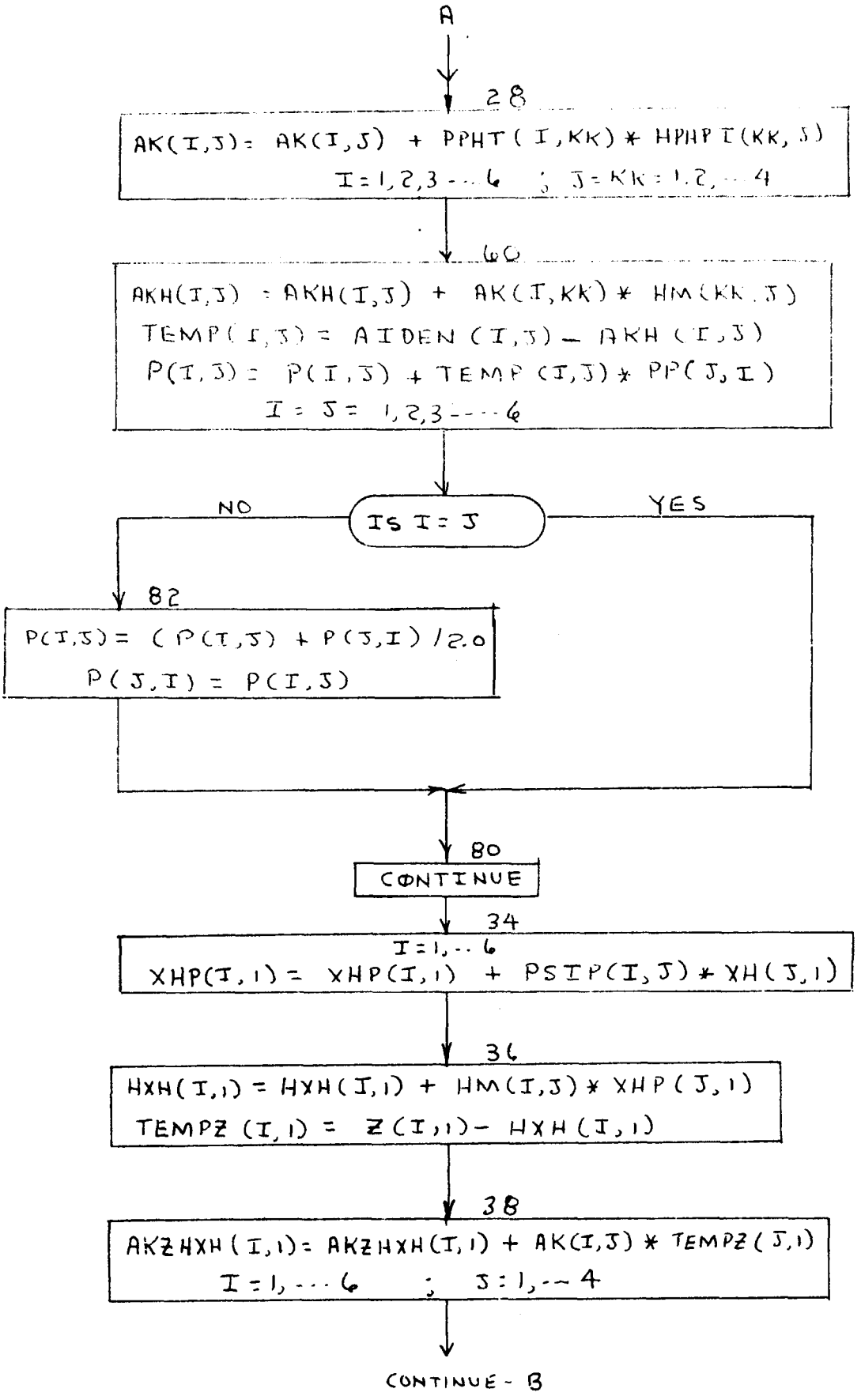

ENTER
IS MCCF = 1
NO
YES
10
PSIXV(I, 1)=0.0
I=1,--- 6
12
X(4,1) = X(4,1) + DELTVP(1,1)
X(5,1) = X(5,1) + DELTVP(2,1)/R
X(6,1) = X(6,1) + DELTVP(3,1)
16
PSIXV(J, 1) = PSIXV(I,1) + PSIP(I,J) * X(J,1)
I= 1,---6 J= 1,--- 6
13
GAMR(I,1) = GAMMAP(I,1) * RS
X(I,1) = PSIXV(I,1) + GAMR(I,1) + QP(I)
I= 1,----6
20
CALL ØBSM
RETURN

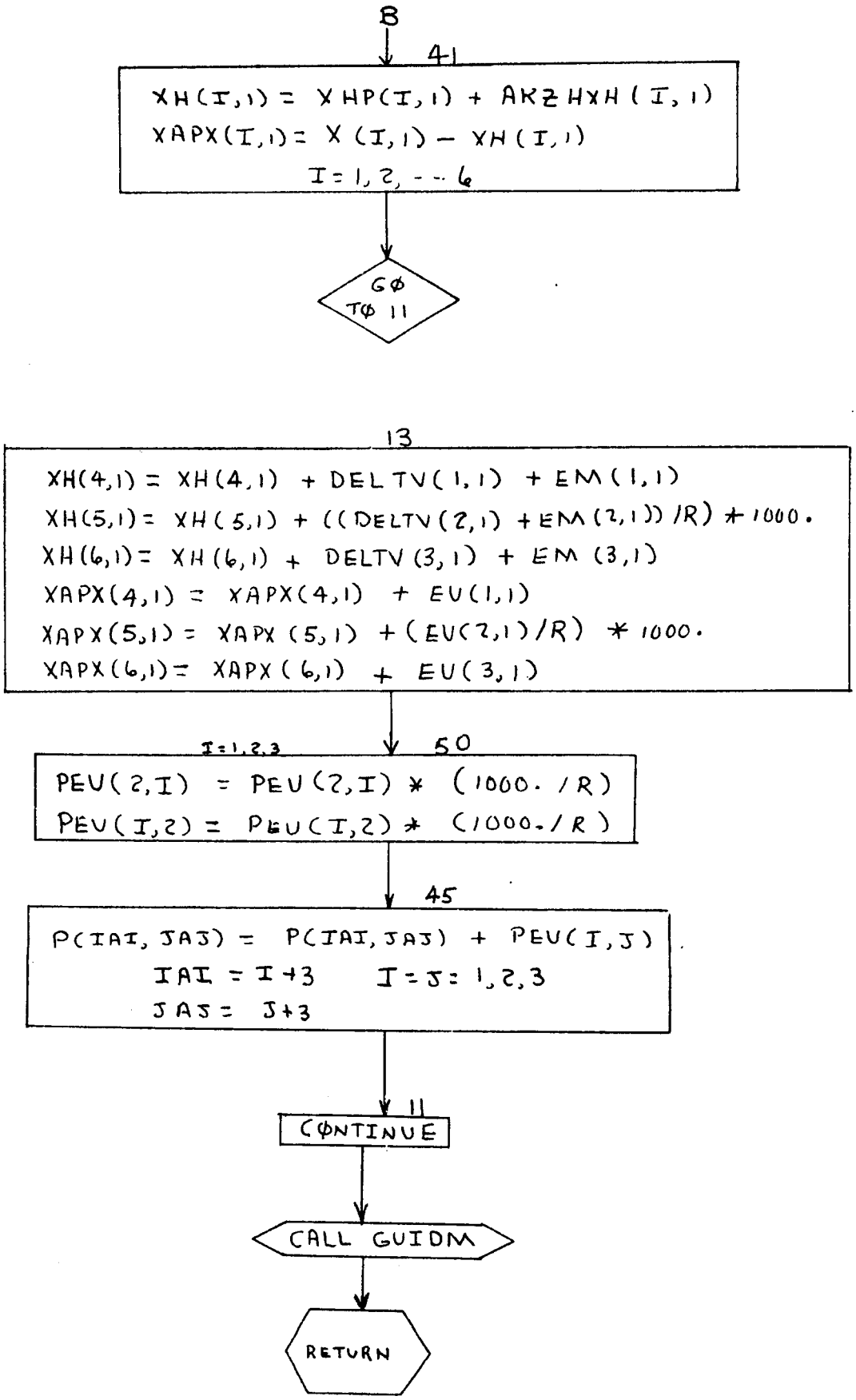
ØBSM


ENTER
IS NAVF = 1
YES
NO
10
CALL NAVM
RETURN
12
IS MCCF = 1
YES
NO
10
CALL NAVM
RETURN
11
HX(I,1) = 0.0
ALB(I,1) = 0.0
I = 1,2,3,4
17
HX(I,1) = HX(I,1) + HM(I,J) * X(J,1)
I = 1,---4 J = 1,---6
15
ALB(I,1) = ALB(I,1) + AL(I,J) * B(J,1)
I = 1,---4 J = 1,---NM
16
V(I) = SIGMA(I) *((SQRT(ALOG(1./RNUM(XX)
**2))) *(SIN(TWOPI * RNUM(XX))))
Z(I,1) = HX(I,1) + ALB(I,1) + V(I)
I = 1,---4
10
CALL NAVM
RETURN

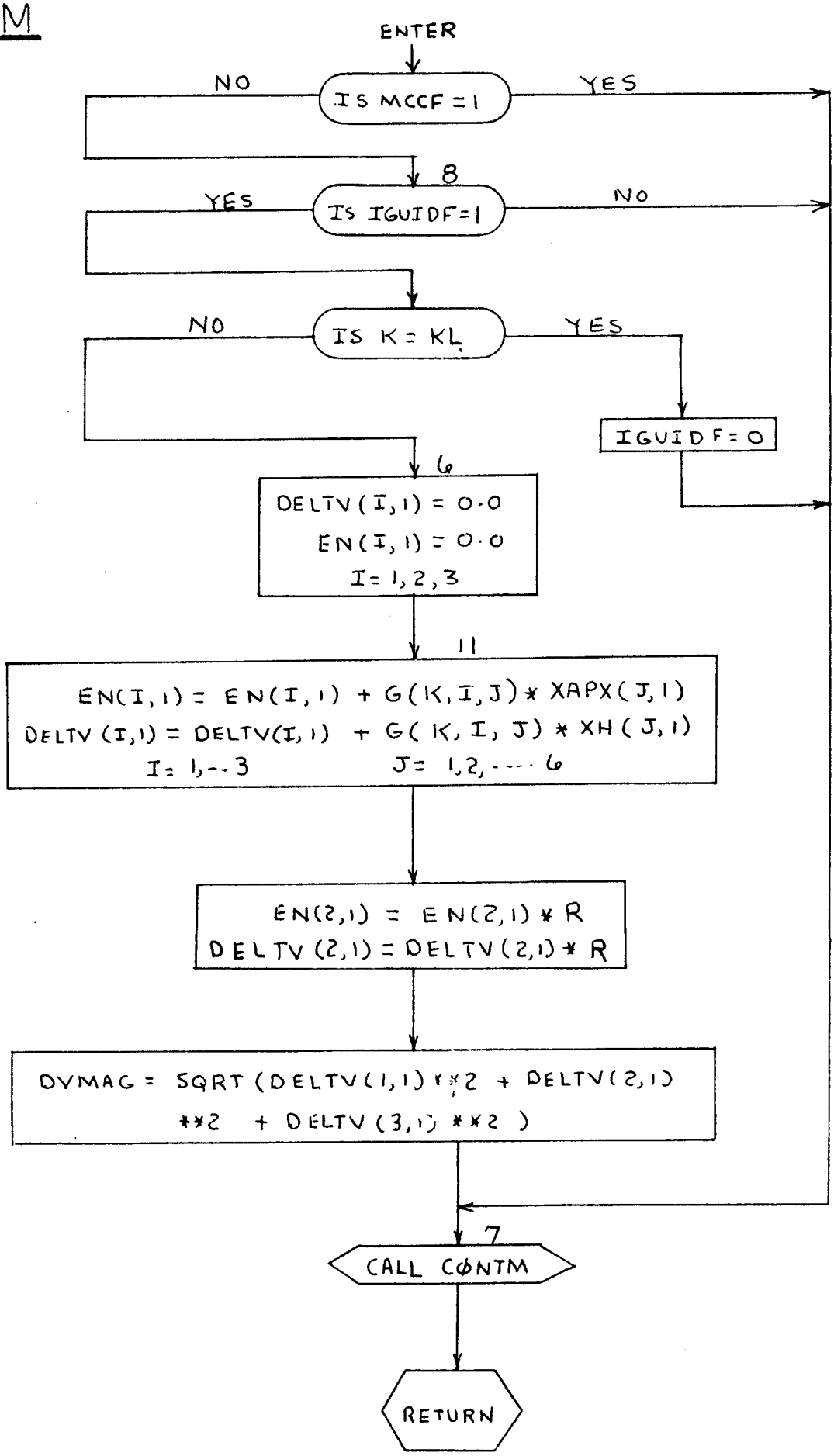
NAVM


ENTER
IS NAVF = 0
NO
YES
11
CONTINUE
CALL GUIDM
RETURN
10
IS MCCF = 1
NO
YES
GO TO 13
12
PSIT(I,J) = PSIP(J,I)
AIDEN = I (IDENTITY)
I = J = 1,2,3, ---- 6
18
PSIB(I,J) = PSIB(I,J) + PSIP(I,KK) * P(KK,J)
PSPPT(I,J) = PSPPT(I,J) + PSIB(I,KK) * PSIT(KK,J)
PP(I,J) = PSPPT(I,J) + QQ(I,J)
I = J = KK = 1,2,3, ---- 6
21
HT(I,J) = HM(J,I)
PPHT(I,J) = PPHT(I,J) + PP(I,KK) * HT(K,K,J)
I = KK = 1,2,3 --- 6 ; J = 1,2,--- 4
24
HPPHT(I,J) = HPPHT(I,J) + HM(I,KK) * PPHT(KK,J)
HPHP(I,J) = HPPHT(I,J) + PV(I,J)
I = J = 1,2,---4 ; KK = 1,2,3, --- 6
CALL MATIN
CONTINUE-A

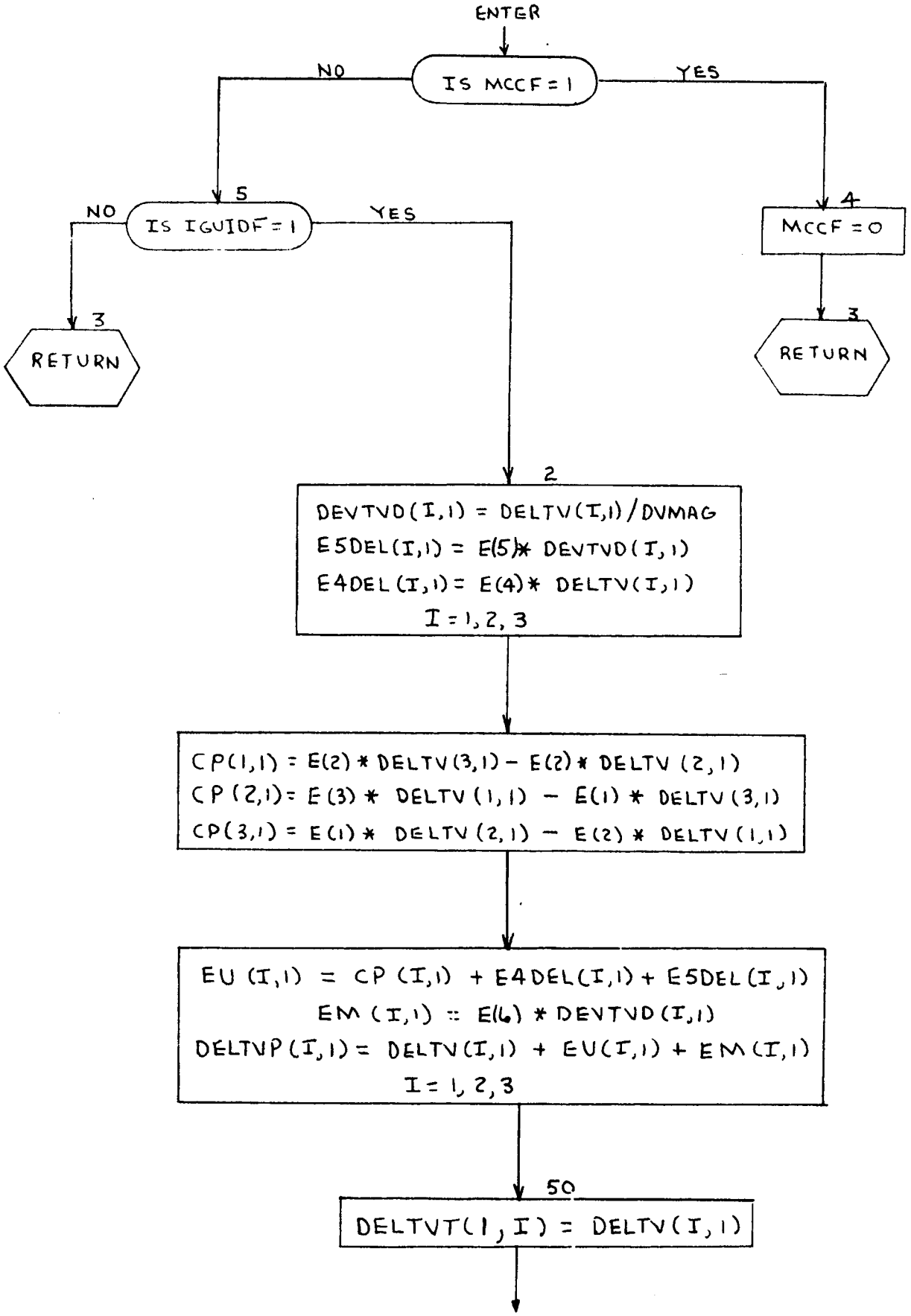

A
28
AK(I,J) = AK(I,J) + PPHT(I,KK) * HPHPT(KK,J)
I = 1,2,3 ... 6 ; J = KK = 1,2, ... 4
60
AKH(I,J) = AKH(I,J) + AK(I,KK) * HM(KK,J)
TEMP(I,J) = AIDEN(I,J) - AKH(I,J)
P(I,J) = P(I,J) + TEMP(I,J) * PP(J,I)
I = J = 1,2,3 ---- 6
NO
IS I = J
YES
82
P(I,J) = (P(I,J) + P(J,I)/2.0
P(J,I) = P(I,J)
80
CONTINUE
34
I = 1, .. 6
XHP(I,1) = XHP(I,1) + PSIP(I,J) * XH(J,1)
36
HXH(I,1) = HXH(I,1) + HM(I,J) * XHP(J,1)
TEMPZ(I,1) = Z(I,1) - HXH(I,1)
38
AKZHXH(I,1) = AKZHXH(I,1) + AK(I,J) * TEMPZ(J,1)
I = 1, ... 6 ; J = 1, -- 4
CONTINUE - B


8
41
XH(I,1) = XHP(I,1) + AKZHXH(I,1)
XAPX(I,1) = X(I,1) - XH(I,1)
I = 1, 2, ... 6
GO TO 11
13
XH(4,1) = XH(4,1) + DELTV(1,1) + EM(1,1)
XH(5,1) = XH(5,1) + ((DELTV(2,1) + EM(2,1))/R) * 1000.
XH(6,1) = XH(6,1) + DELTV(3,1) + EM(3,1)
XAPX(4,1) = XAPX(4,1) + EU(1,1)
XAPX(5,1) = XAPX(5,1) + (EU(2,1)/R) * 1000.
XAPX(6,1) = XAPX(6,1) + EU(3,1)
I=1,2,3
50
PEU(2,I) = PEU(2,I) * (1000./R)
PEU(I,2) = PEU(I,2) * (1000./R)
45
P(IAI, JAJ) = P(IAI, JAJ) + PEU(I,J)
IAI = I+3
I = J = 1,2,3
JAJ = J+3
11
CONTINUE
CALL GUIDM
RETURN

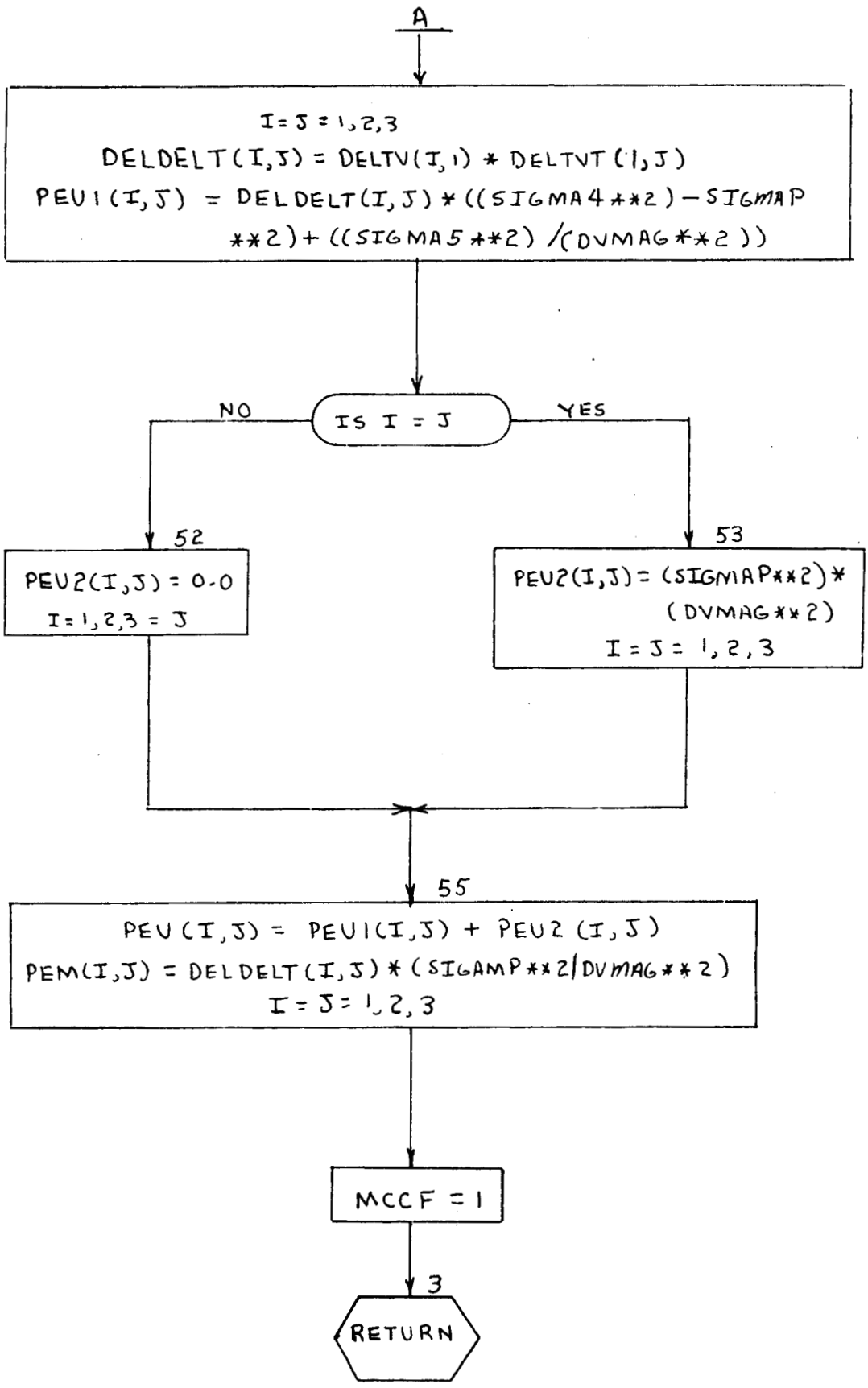
GUIDM


ENTER
IS MCCF = 1
NO
YES
8
IS IGUIDF = 1
YES
NO
IS K = KL
NO
YES
IGUIDF = 0
6
DELTV(I,1) = 0.0
EN(I,1) = 0.0
I = 1,2,3
11
EN(I,1) = EN(I,1) + G(K,I,J) * XAPX(J,1)
DELTV(I,1) = DELTV(I,1) + G(K,I,J) * XH(J,1)
I = 1,--3
J = 1,2,----6
EN(2,1) = EN(2,1) * R
DELTV(2,1) = DELTV(2,1) * R
DVMAG = SQRT(DELTV(1,1)**2 + DELTV(2,1)
2 + DELTV(3,1)2)
7
CALL CØNTM
RETURN

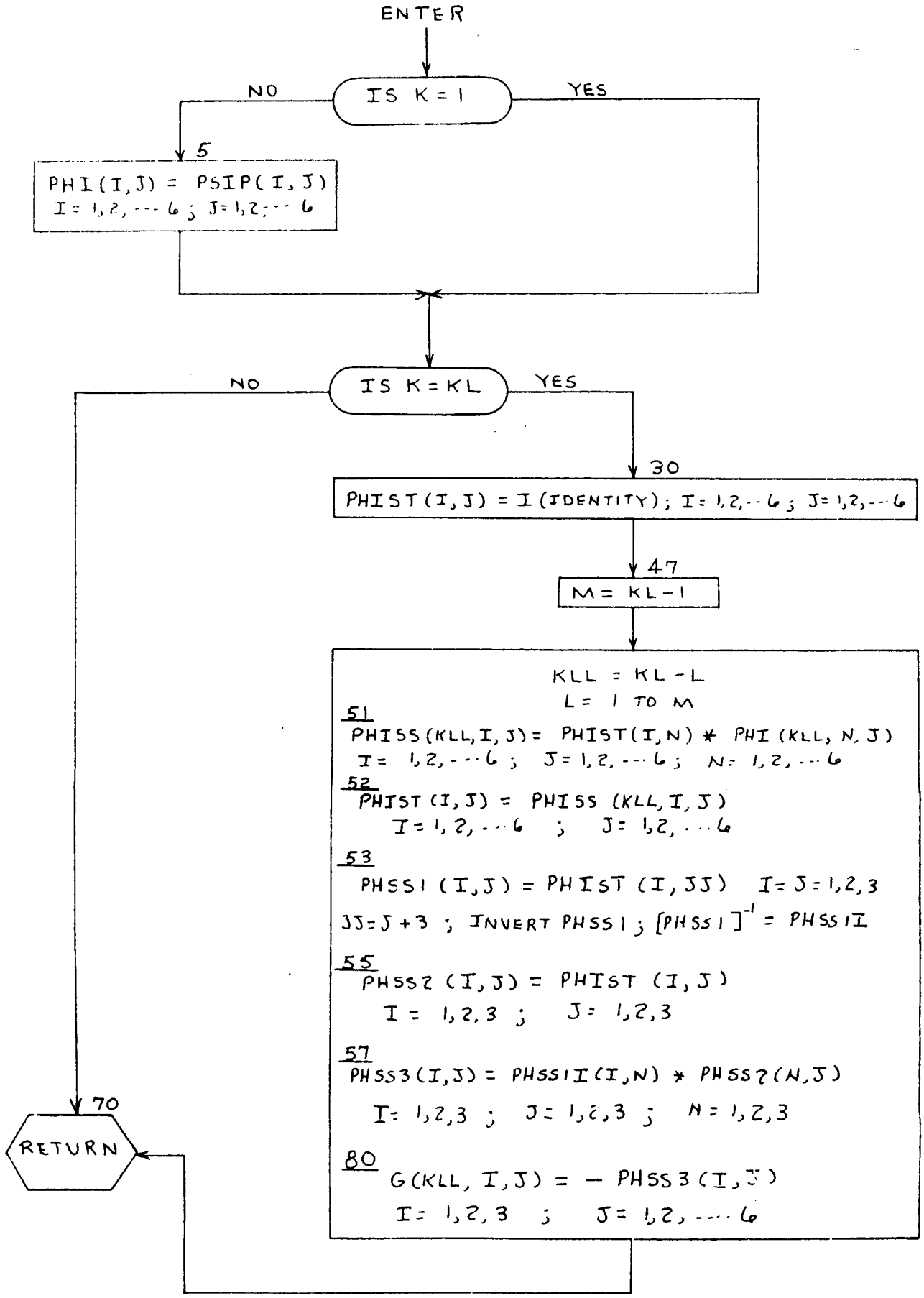
CØNTM



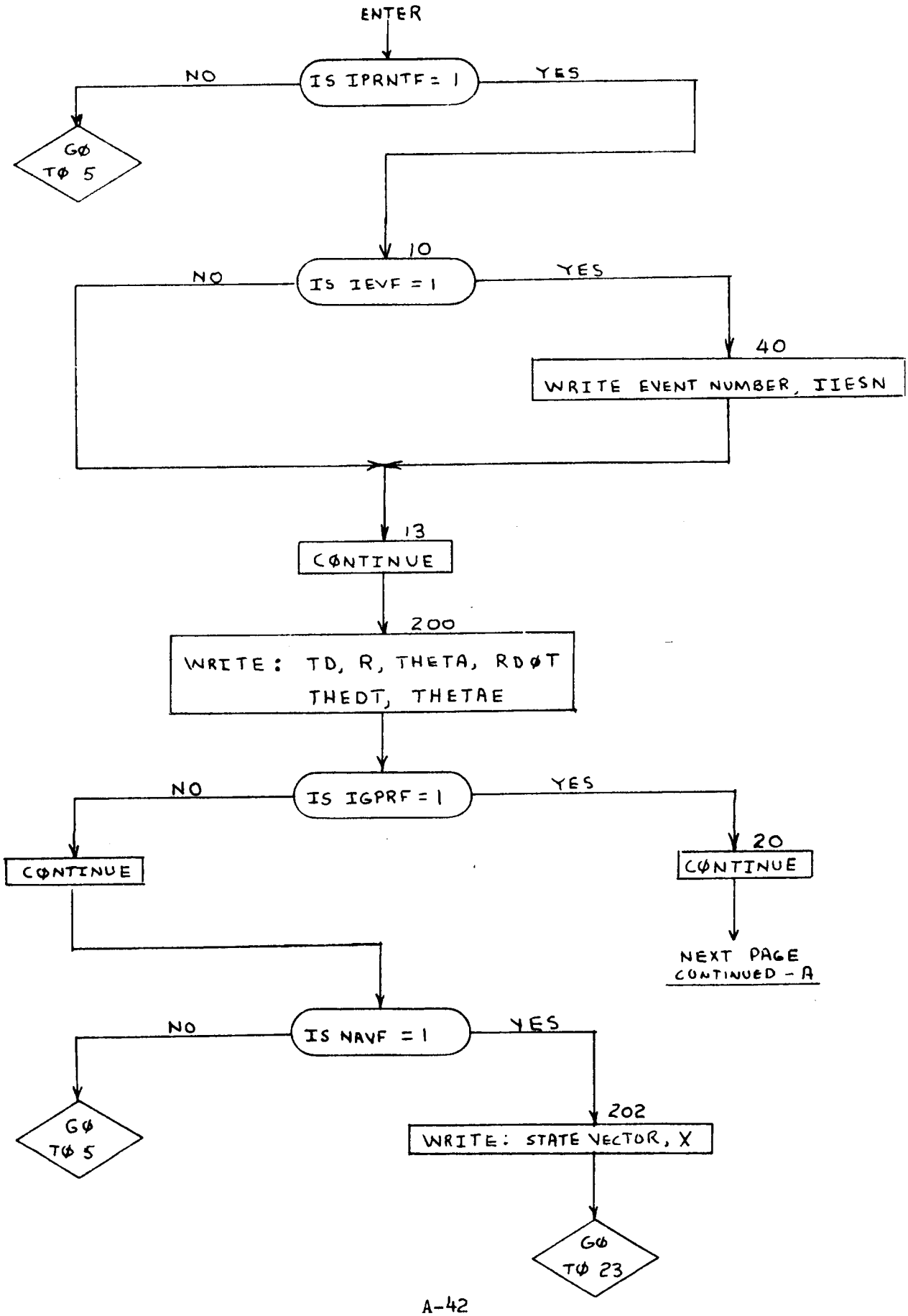
CONTINUED- A

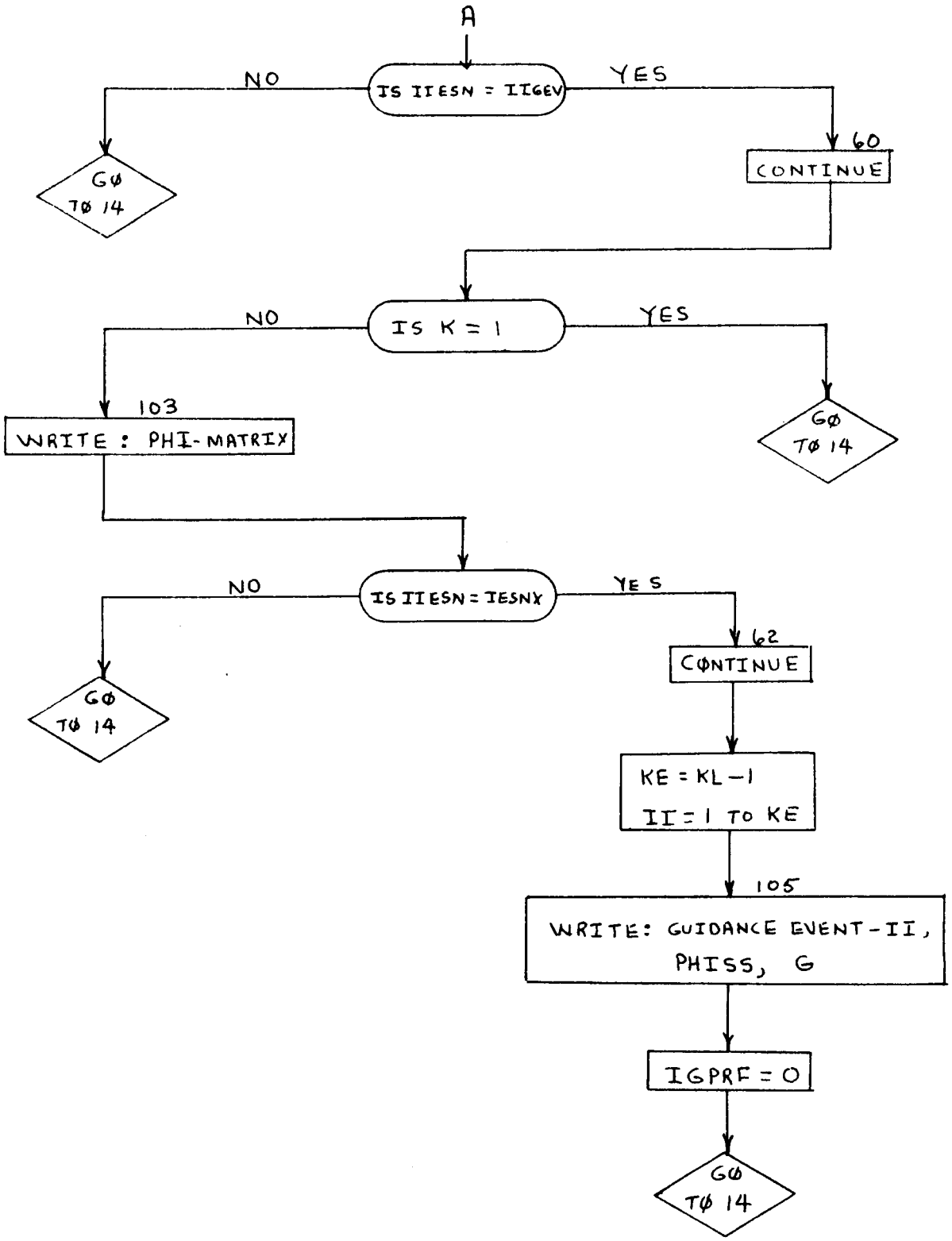

A
I = J = 1,2,3
DELDELT(I,J) = DELTV(I,1) * DELTVT(1,J)
PEU1(I,J) = DELDELT(I,J) * ((SIGMA4**2) - SIGMAP **2) + ((SIGMA5**2) /(DVMAG**2))
IS I = J
NO
YES
52
PEU2(I,J) = 0.0
I = 1,2,3 = J
53
PEU2(I,J) = (SIGMAP**2) * (DVMAG**2)
I = J = 1,2,3
55
PEU(I,J) = PEU1(I,J) + PEU2(I,J)
PEM(I,J) = DELDELT(I,J) * (SIGAMP**2/DVMAG**2)
I = J = 1,2,3
MCCF = 1
3
RETURN

PØLM

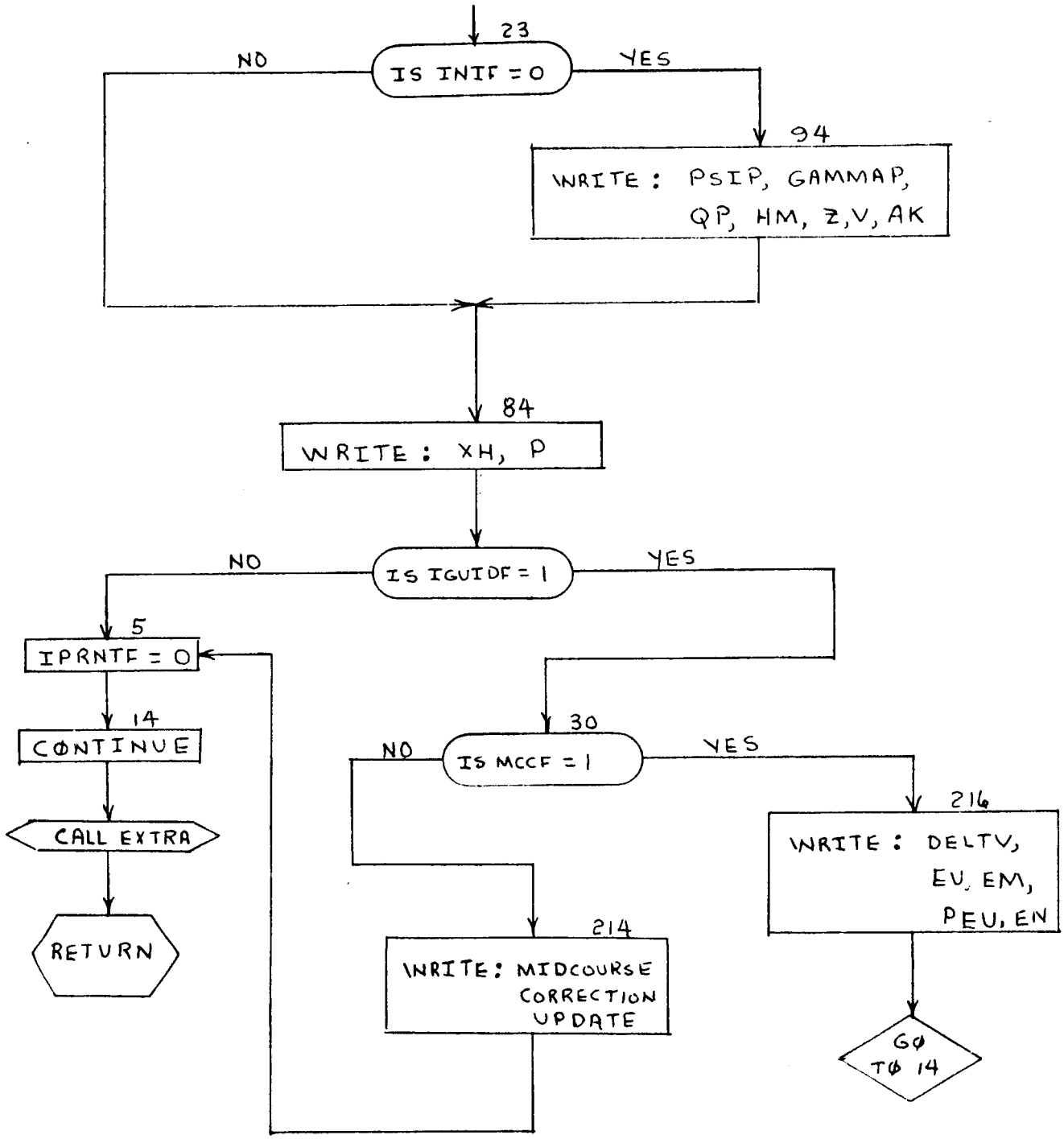

ENTER
IS K = 1
NO
YES
5
PHI(I,J) = PSIP(I,J)
I = 1,2,---6 ; J = 1,2,---6
IS K = KL
NO
YES
30
PHIST(I,J) = I (IDENTITY); I = 1,2,--6 ; J = 1,2,---6
47
M = KL-1
KLL = KL-L
L = 1 TO M
51
PHISS(KLL,I,J) = PHIST(I,N) * PHI(KLL,N,J)
I = 1,2,---6 ; J = 1,2,---6 ; N = 1,2,---6
52
PHIST(I,J) = PHISS(KLL,I,J)
I = 1,2,---6 ; J = 1,2,---6
53
PHSS1(I,J) = PHIST(I,JJ) I = J = 1,2,3
JJ = J+3 ; INVERT PHSS1 ; [PHSS1]^-1 = PHSS1I
55
PHSS2(I,J) = PHIST(I,J)
I = 1,2,3 ; J = 1,2,3
57
PHSS3(I,J) = PHSS1I(I,N) * PHSS2(N,J)
I = 1,2,3 ; J = 1,2,3 ; N = 1,2,3
80
G(KLL,I,J) = - PHSS3(I,J)
I = 1,2,3 ; J = 1,2,----6
70
RETURN

INFM

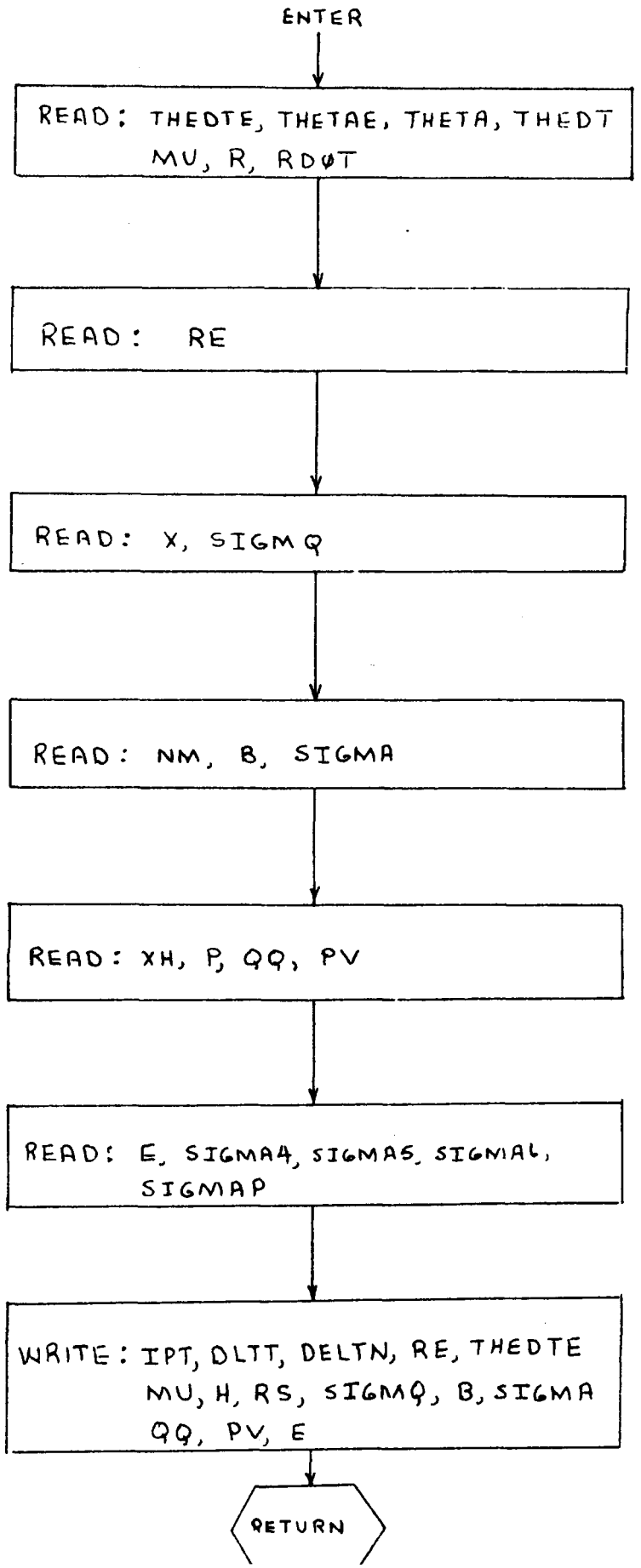

ENTER
IS IPRNTF = 1
NO
YES
GØ TØ 5
10
IS IEVF = 1
NO
YES
40
WRITE EVENT NUMBER, IIESN
13
CØNTINUE
200
WRITE: TD, R, THETA, RDØT
THEDT, THETAE
IS IGPRF = 1
NO
YES
CØNTINUE
20
CØNTINUE
NEXT PAGE
CONTINUED - A
IS NAVF = 1
NO
YES
GØ TØ 5
202
WRITE: STATE VECTOR, X
GØ TØ 23



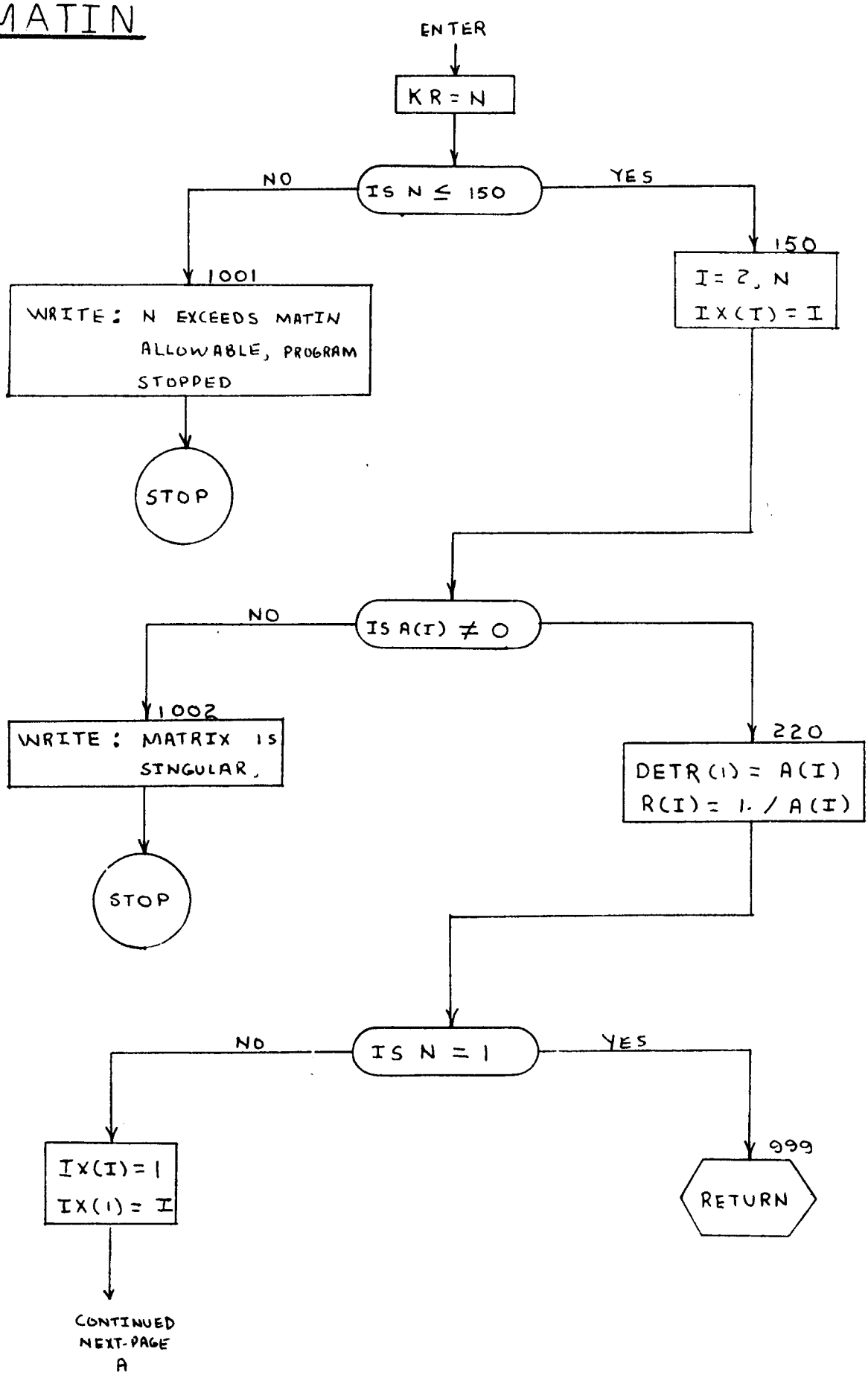
CØNTINUED - NEXT PAGE

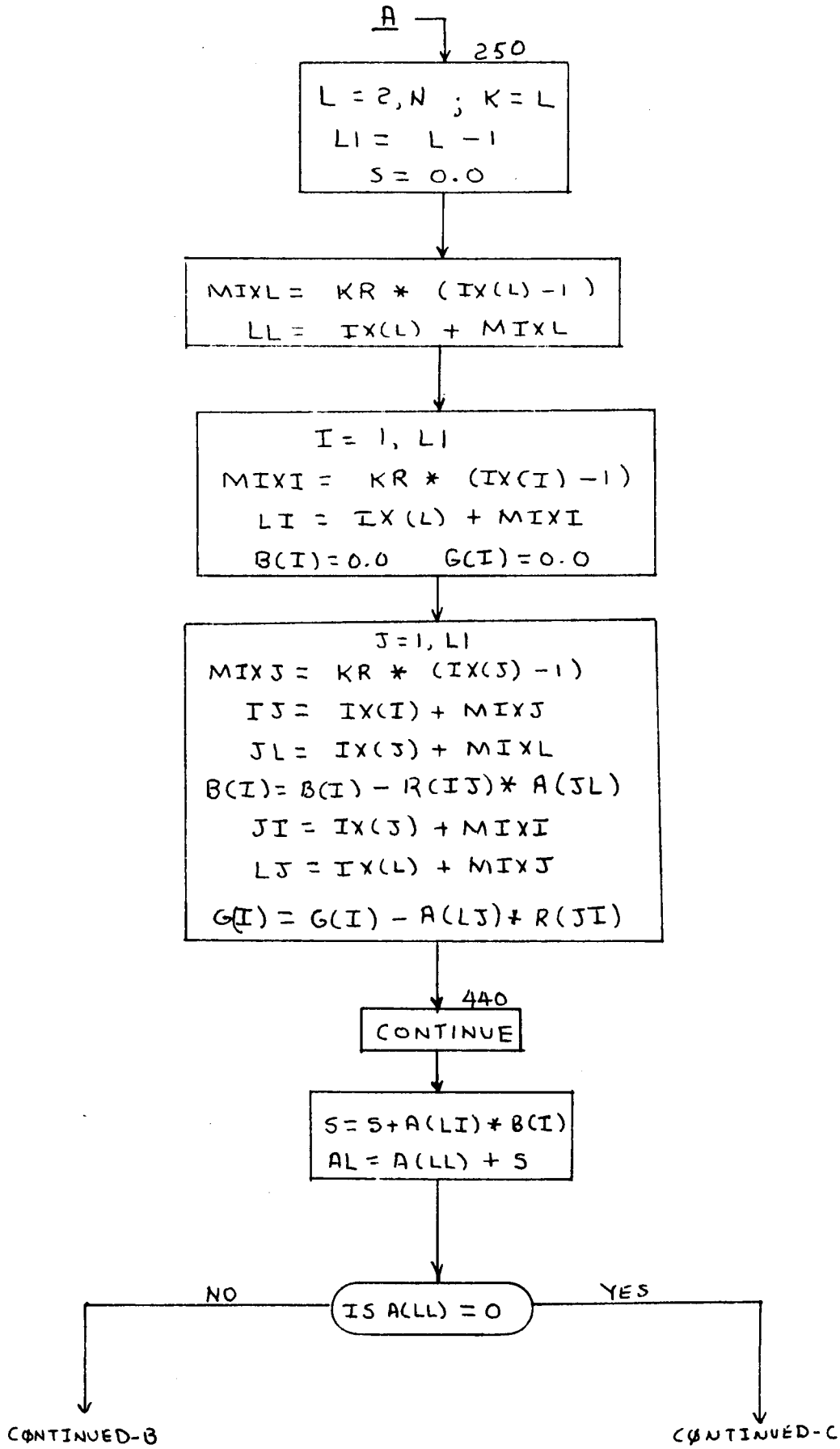

23
IS INIT = 0
NO
YES
94
WRITE : PSIP, GAMMAP, QP, HM, Z, V, AK
84
WRITE : XH, P
IS IGUIDF = 1
NO
YES
5
IPRNTF = 0
14
CONTINUE
CALL EXTRA
RETURN
30
IS MCCF = 1
NO
YES
214
WRITE: MIDCOURSE CORRECTION UPDATE
216
WRITE : DELTV, EU, EM, PEU, EN
GO TO 14

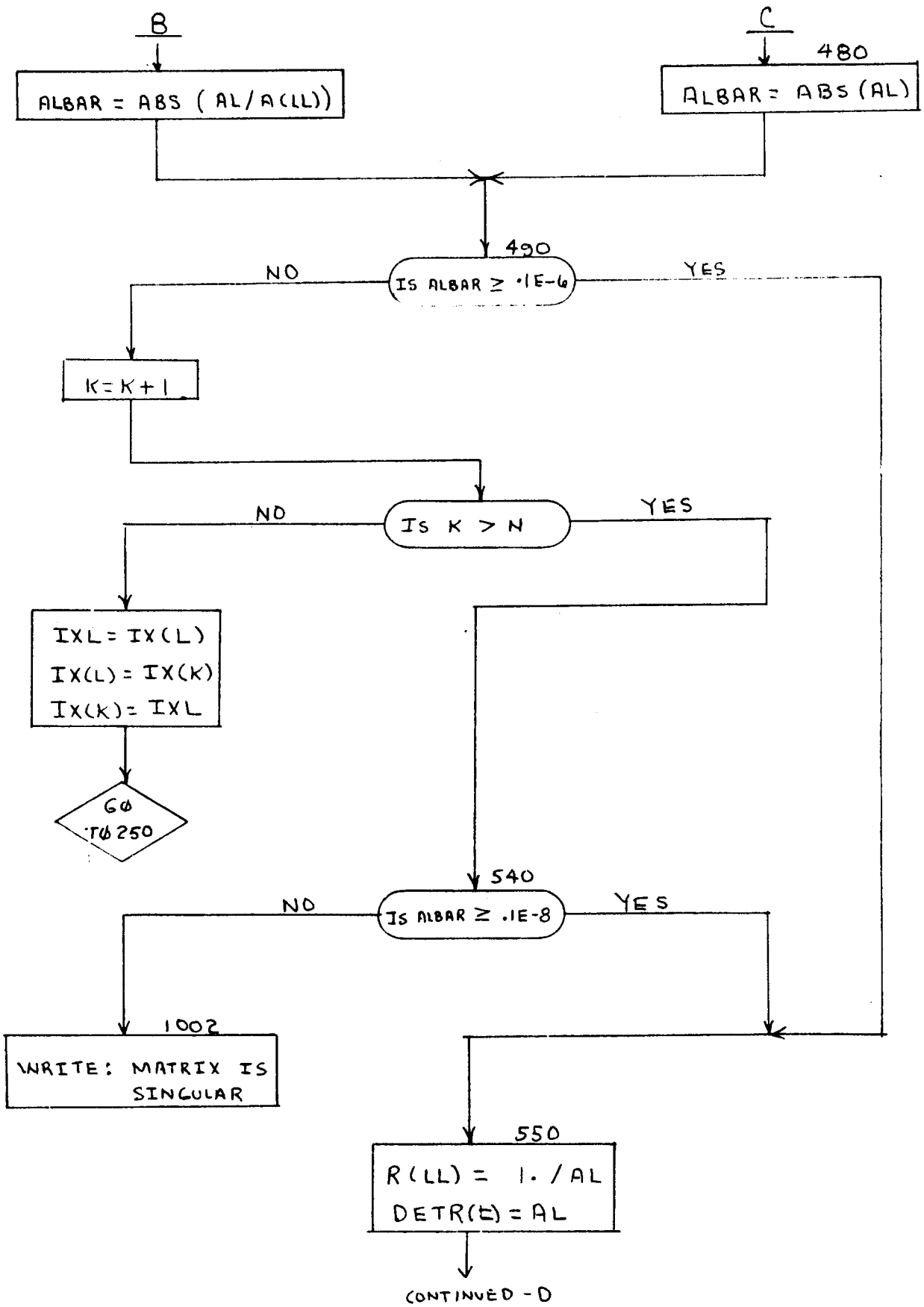
DATIN

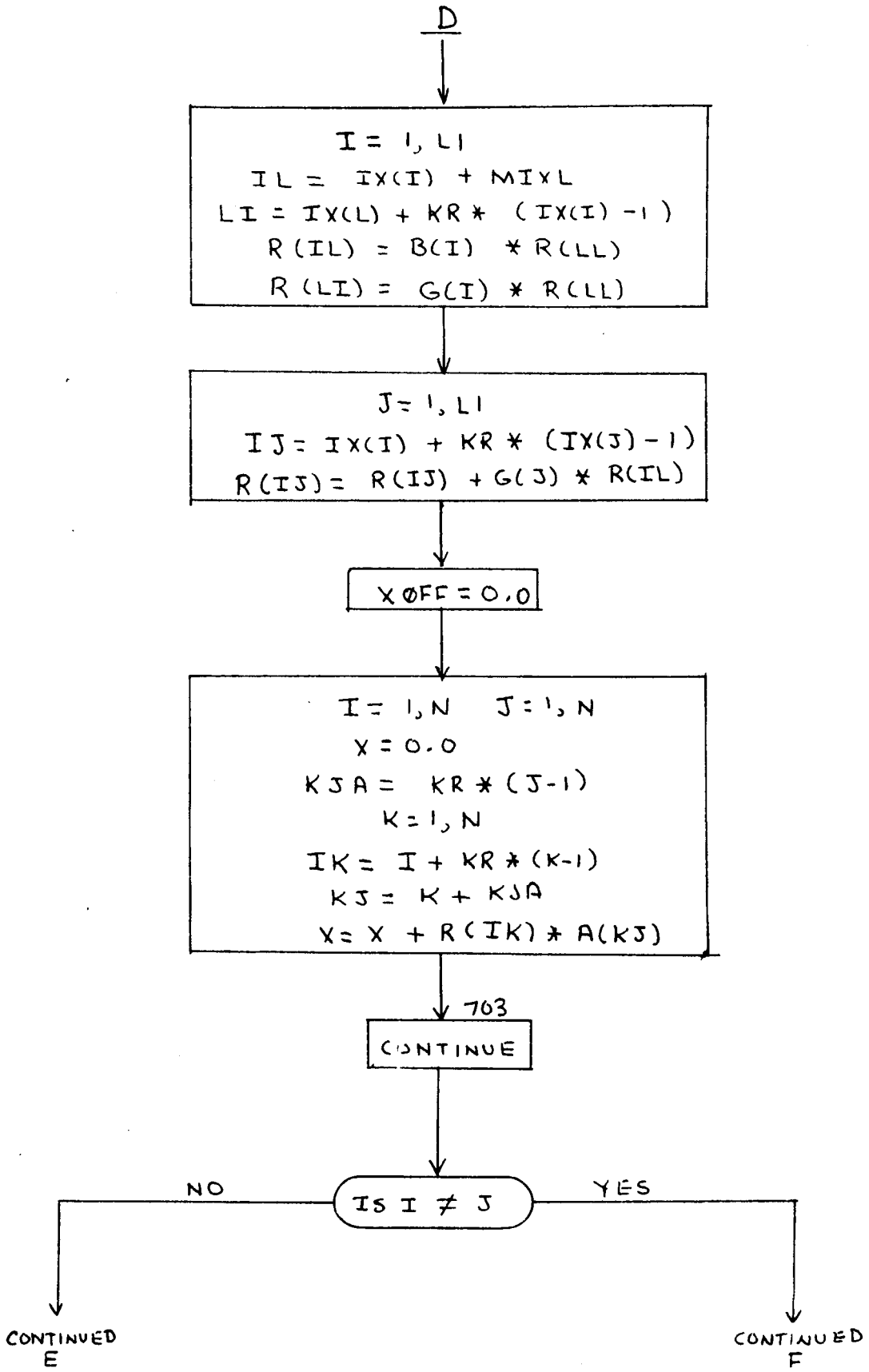

ENTER
READ: THEDTE, THETAE, THETA, THEDT
MU, R, RDØT
READ: RE
READ: X, SIGMQ
READ: NM, B, SIGMA
READ: XH, P, QQ, PV
READ: E, SIGMA4, SIGMA5, SIGMAL,
SIGMAP
WRITE: IPT, DLTT, DELTN, RE, THEDTE
MU, H, RS, SIGMQ, B, SIGMA
QQ, PV, E
RETURN

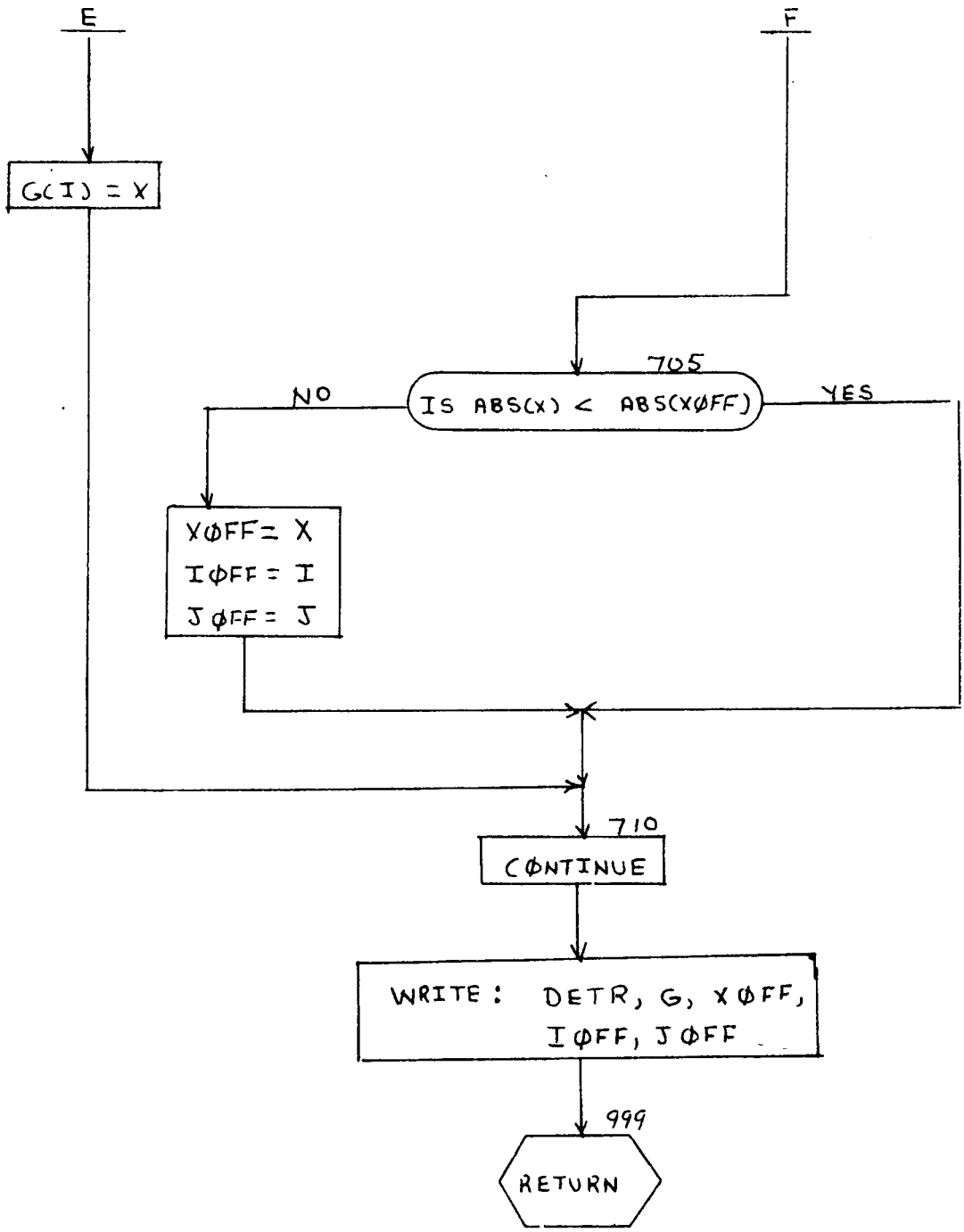
MATIN


ENTER
KR = N
IS N ≤ 150
NO
YES
1001
WRITE: N EXCEEDS MATIN ALLOWABLE, PROGRAM STOPPED
STOP
150
I = 2, N
IX(I) = I
IS A(I) ≠ 0
NO
1002
WRITE: MATRIX IS SINGULAR,
STOP
220
DETR(1) = A(I)
R(I) = 1. / A(I)
IS N = 1
NO
YES
IX(I) = 1
IX(1) = I
CONTINUED NEXT-PAGE A
999
RETURN

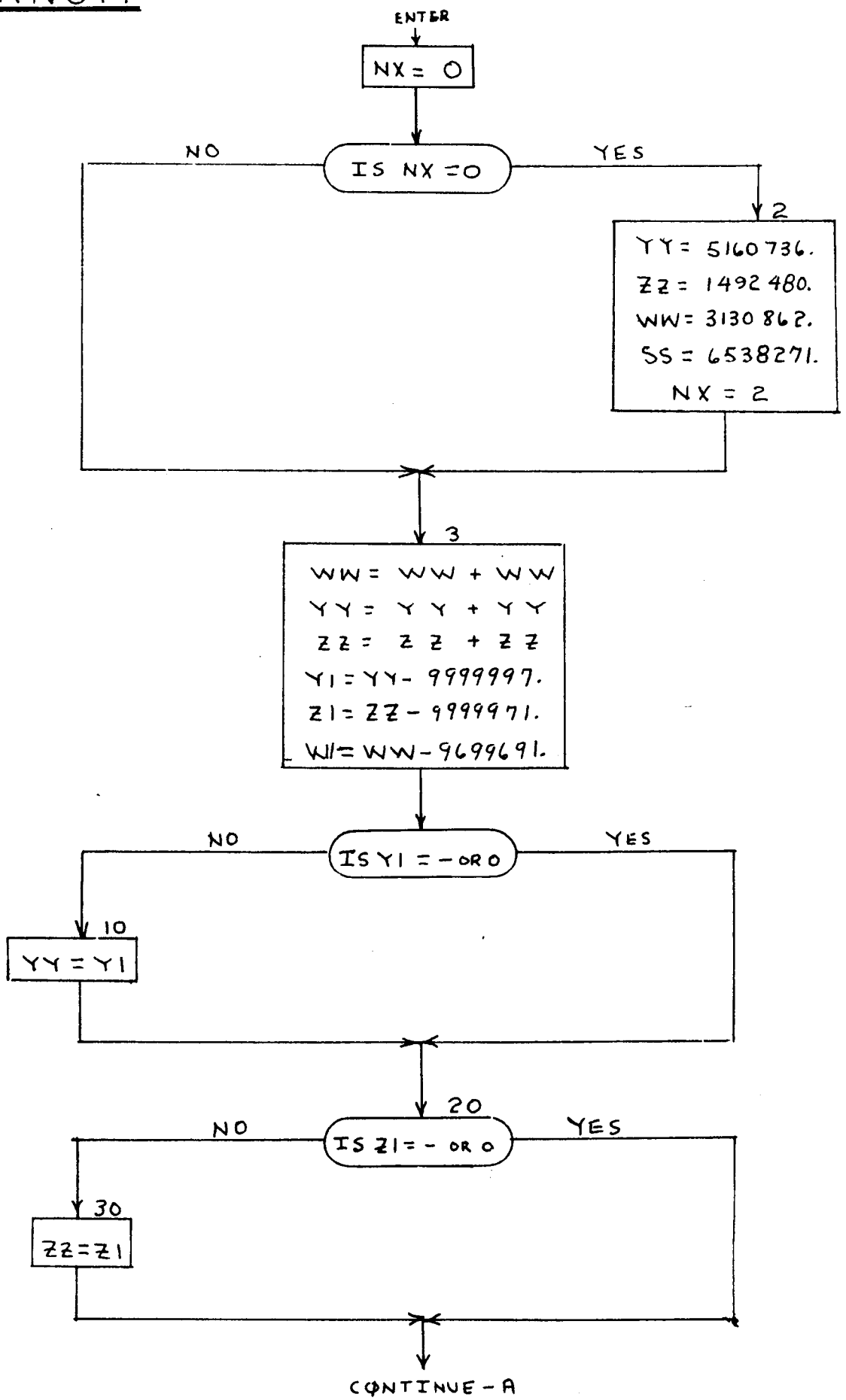

A
250
L = 2, N ; K = L
L1 = L - 1
S = 0.0
MIXL = KR * (IX(L) - 1)
LL = IX(L) + MIXL
I = 1, L1
MIXI = KR * (IX(I) - 1)
LI = IX(L) + MIXI
B(I) = 0.0 G(I) = 0.0
J = 1, L1
MIXJ = KR * (IX(J) - 1)
IJ = IX(I) + MIXJ
JL = IX(J) + MIXL
B(I) = B(I) - R(IJ) * A(JL)
JI = IX(J) + MIXI
LJ = IX(L) + MIXJ
G(I) = G(I) - A(LJ) * R(JI)
440
CONTINUE
S = S + A(LI) * B(I)
AL = A(LL) + S
IS A(LL) = 0
NO
YES
CØNTINUED-B
CØNTINUED-C

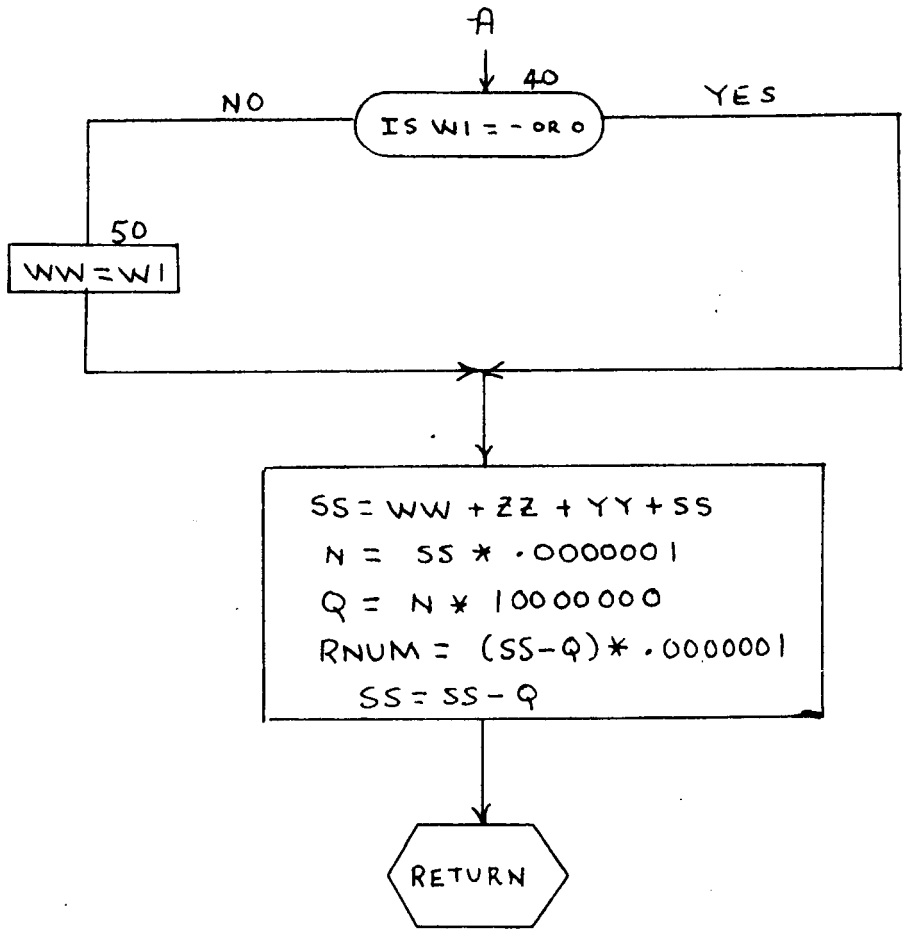

B
ALBAR = ABS (AL/A(LL))
C
480
ALBAR = ABS (AL)
490
IS ALBAR ≥ ·1E-6
NO
YES
K=K+1
IS K > N
NO
YES
IXL = IX(L)
IX(L) = IX(K)
IX(K) = IXL
60
TO 250
540
IS ALBAR ≥ .1E-8
NO
YES
1002
WRITE: MATRIX IS SINGULAR
550
R(LL) = 1./AL
DETR(E) = AL
CONTINUED - D


D
I = 1, L1
IL = IX(I) + MIXL
LI = IX(L) + KR * (IX(I) - 1)
R(IL) = B(I) * R(LL)
R(LI) = G(I) * R(LL)
J = 1, L1
IJ = IX(I) + KR * (IX(J) - 1)
R(IJ) = R(IJ) + G(J) * R(IL)
XØFF = 0.0
I = 1, N J = 1, N
X = 0.0
KJA = KR * (J-1)
K = 1, N
IK = I + KR * (K-1)
KJ = K + KJA
X = X + R(IK) * A(KJ)
703
CONTINUE
IS I ≠ J
NO
YES
CONTINUED
E
CONTINUED
F


E
F
G(I) = X
705
IS ABS(X) < ABS(XØFF)
NO
YES
XØFF = X
IØFF = I
JØFF = J
710
CØNTINUE
WRITE : DETR, G, XØFF,
IØFF, JØFF
999
RETURN

RNUM


ENTER
NX = 0
IS NX = 0
NO
YES
2
YY = 5160736.
ZZ = 1492480.
WW = 3130862.
SS = 6538271.
NX = 2
3
WW = WW + WW
YY = YY + YY
ZZ = ZZ + ZZ
Y1 = YY - 9999997.
Z1 = ZZ - 9999971.
W1 = WW - 9699691.
IS Y1 = - OR 0
NO
YES
10
YY = Y1
20
IS Z1 = - OR 0
NO
YES
30
ZZ = Z1
CONTINUE - A


A
40
IS W1 = - OR 0
NO
YES
50
WW = W1
SS = WW + ZZ + YY + SS
N = SS * .0000001
Q = N * 10000000
RNUM = (SS-Q) * .0000001
SS = SS - Q
RETURN

```
C  THIS IS THE MAIN PROGRAM THAT INCLUDES EXM1,EXM2,AND EXM3
C
      DIMENSION IESN(10),  TEV(10),  IGEV(10)
      COMMON/POLR/K,KL,PSIP(6,6),PHI(6,6,6),KLL
      COMMON/PSIR/A1STAR,A2STAR,A3STAR,B1STAR,B2STAR,B3STAR,CSTAR
      COMMON/INI/MU,THEDTE,THETAE,DELT,R,RDOT,THETA,THEDT,H
      COMMON/MAIN/TD,TDSV
      COMMON/FIRS/P(6,6),QQ(6,6),PV(4,4)
      COMMON/SYNRR/X(6,1),Q(6),QP(6)
      COMMON/DYNR/RS,SIGMQ(3)
      COMMON/HMR/HM(4,6),RE
      COMMON/ITI/AL(4,6),B(6,1),NM,SIGMA(4)
      COMMON/FLAGS/MCCF,NAVF,IGUIDF,IGPRF,IRESF
      COMMON/INTA/E(6),SIGMA4,SIGMA5,SIGMA6,SIGMAP
      COMMON/NAVR/XH(6,1),XAPX(6,1)
      COMMON/FLAGG/IPRNTF,IIESN,IIGEV,IESNX,IEVF,PHISS(6,6,6)
      COMMON/INPT/IPT,DLTT,DELTN
      COMMON/FINIF/INIF
C
C  EXM1 INITIALIZATION - MODEL 1
C
      REAL MU
      IRUN = 0
      READ(5,200) IRUNX
  200 FORMAT(I5)
C
C EXM1 MODEL 1
C
  100 IF (IRUN .GE. IRUNX) GO TO 101
      IRUN = IRUN + 1
      READ(5,201) IGPRF
  201 FORMAT(I5)
C
  102 CONTINUE
C
C  EXM2 INITIALIZATION - MODEL 1
C
      READ(5,202) IPT,IESNX,KL,JEVL
  202 FORMAT(4I5)
      READ(5,203) (IESN(I), I=1,JEVL)
  203 FORMAT(10I5)
      READ(5,204) (TEV(I), I=1,JEVL)
  204 FORMAT(5E13.6)
      READ(5,205) (IGEV(I), I=1,KL)
  205 FORMAT(10I5)
C
   28 IIESN = IESN(1)
      IIGEV = IGEV(1)
      INIF = 1
      JEV = 1
      K = 1
      IEVF = 1
C
C  EXM3 INITIALIZATION MODEL 1
C
      READ(5,206) DLTT,DELTN,TD,TN
  206 FORMAT(4E13.6)
C
      NAVF = 1
```

```
      IGUIDF = 0
      IPRNTF = 1
      IIPT = 0
C  EXM2 MODEL-1
    8 IF(IIESN.EQ.IESNX) GO TO 100
      IF(INIF.EQ.1) GO TO 2
      JEV = JEV + 1
      IIESN = IESN(JEV)
      ITEV = TEV(JEV)
      IF(IGUIDF.EQ.1) GO TO 3
      GO TO 4
    2 CONTINUE
      CALL DATIN
C  THE FOLLOWING BLOCK OF DATA ARE INITILIZATION VALUES THAT ARE PERMENENT
C  DATA
C
C     NTM INITIAL
      TDSV = TD
      H = R*R*THEDT
C     PSIM INITIAL
      IRESF = 1
      A1STAR =  ((2.0*MU)/R**3)+THEDT**2
      A2STAR = 2.0*R*THEDT
      A3STAR = 1.0/R**2
      B1STAR= -((2.0*RDOT*THEDT)/R**2)
      B2STAR = (2.0*THEDT)/R
      B3STAR = (2.0*RDOT)/R
      CSTAR = MU/(R*R*R)
C     TM INITIAL
      DO 35 I = 1,4
      DO 35 J = 1,6
   35 HM(I,J) = 0.0
C      OBSM INITIAL
      DO 60 I=1,4
      DO 60 J=1,4
   60 AL(I,J) = 0.0
      AL(1,1)=1.
      AL(2,2)=1.
      AL(3,3)=1.
      AL(4,4)=1.
C     GUIDM INITIAL
      MCCF = 0
      CALL INFM
      INIF = 0
      GO TO 6
    3 K = K +1
      IIGEV = IGEV(K)
      IGUIDF = 0
    4 IEVF = 0
C  EXM3 MODEL-1
    6 IF(IEVF .EQ. 1) GO TO 8
      TD = TD + DLTT
    9 IF(NAVF.EQ.0) GO TO 11
      TN = TN + DELTN
      NAVF = 0.
      IIPT = IIPT + 1
   11 IF(TD.GE.TN) GO TO 12
      GO TO 10
   12 TD = TN
      NAVF = 1
```

```
      IF(IIPT .GE. IPT) GO TO 13
      GO TO 10
   13 IPRNTF = 1
      IIPT = 0
   10 IF(TD.GE.TTEV) GO TO 14
      GO TO 15
   14 TD =TTEV
      IEVF = 1
      IPRNTF = 1
      IF(IIESN.GE.IIGEV) GO TO 16
      GO TO 15
   16 IGUIDF = 1
      TN = TTEV
      NAVF = 1
      IIPT = 0
   15 CALL NTM
      IF(IGPRF.EQ.1) GO TO 18
      IF(NAVF.EQ.0) GO TO 30
      CALL TM
   31 CALL DYNM
   33 CALL INFM
      IF(MCCF.EQ.0) GO TO 6
      GO TO 31
   30 IF(IEVF.EQ.0) GO TO 6
      GO TO 33
   18 IF(IGUIDF.EQ.1) GO TO 20
   21 GO TO 6
   20 CALL POLM
      CALL INFM
      GO TO 6
  101 STOP
      END
```

```
      SUBROUTINE NTM
C  THIS SUBROUTINE GENERATES A NOMINAL TRAJECTORY USING MODEL-1. AN
C  APPROXIMATE METHOD IS USED TO SOLVE THE TWO-BODY PLANAR EQS. OF MOTION
C
      COMMON/INI/MU,THEDTE,THETAE,DELT,R,RDOT,THETA,THEDT,H
      COMMON/MAIN/TD,TDSV
      REAL MU
      DELT = TD-TDSV
      TDSV = TD
      A = MU/R**3
      C =  DELT**2 /2.0
      E = RDOT*DELT
      F = MU/R**2
      G = H**2 /R**3
      B=3.*G/(2.*R)
      S = A - B
      T = F - G
   20 D3 = 1.0 - (C*S)
   21 DELR = (1.0/D3) * (E -(T*C))
   22 DELRDT = (1.0/D3) * ((S*RDOT* DELT**2) - (DELT * T))
   23 THETA = THETA + ((H*DELT /R**2) *(1.0 -(DELR)/(R)))
   24 R = R + DELR
   25 RDOT = RDOT + DELRDT
   26 THEDT = H/R**2
   27 THETAE = THETAE + (THEDTE * DELT)
C  THE NEXT STATEMENT CALLS THE PSI MODULE TO CALCULATE THE ELEMENTS
C  OF THE TRANSITION MATRIX AND GAMMA MATRIX
      CALL PSIM
      RETURN
      END
```

```
      SUBROUTINE PSIM
C  THIS SUBROUTINE DETERMINES THE ELEMENTS OF THE PSI AND GAMMA MATRIX
C  THIS IS A SIMPLIFIED MODEL CALLED MODEL-1.
      DIMENSION PSI(6,6),GAMMA(6,1),SISIP(6,6), SIDP(6,6),SIGAMP(6,1),GA
     .MMPP(6,1),PSIQ(6),QDELT(6)
      COMMON/PSIR/A1STAR,A2STAR,A3STAR,B1STAR,B2STAR,B3STAR,CSTAR
      COMMON/FLAGS/MCCF,NAVF,IGUIDF,IGPRF,IRESF
      COMMON/INI/MU,THEDTE,THETAE,DELT,R,RDOT,THETA,THEDT,H
      COMMON/POLR/K,KL,PSIP(6,6),PHI(6,6,6),KLL
      COMMON/PSIMR/GAMMAP(6,1)
      COMMON/SYNRR/X(6,1),Q(6),QP(6)
      COMMON/DYNR/RS,SIGMQ(3)
      DATA TWOPI/6.28318/
      REAL MU
      IF (IRESF .EQ. 0) GO TO 1
      DO 80 I = 1,6
      DO 80 J = 1,6
      IF (I-J) 70,60,70
   60 PSIP(I,J) = 1.0
      GO TO 80
   70 PSIP(I,J)= 0.0
   80 CONTINUE
   81 DO 82 I = 1,6
      GP(I) = 0.0
   82 GAMMAP (I,1) = 0.0
      IRESF = 0
C  NOW THAT STARTING VALUES ARE OBTAINED WE CONTINUE THE PROGRAM
C  AND  DETERMINE  THE MATRIX ELEMENTS FOR PSI AND GAMMA
C
    1 A1 = A1STAR
    2 A1STAR=((2.0*MU)/R**3)+ THEDT**2
    3 A1 = 0.5*(A1 + A1STAR)
    4 A2 = A2STAR
    5 A2STAR = 2.0*R*THEDT
    6 A2 = 0.5 * (A2 + A2STAR)
    7 A3 = A3STAR
    8 A3STAR =  1.0 /R**2
    9 A3 = 0.5 * (A3 + A3STAR)
   10 B1 = B1STAR
   11 B1STAR =-((2.0*RDOT*THEDT)/R**2)
   12 B1 = 0.5 *(B1 + B1STAR)
   13 B2 = B2STAR
   14 B2STAR = (2.0 * THEDT)/R
   15 B2 = 0.5*(B2 + B2STAR)
   16 B3 = B3STAR
   17 B3STAR = (2.0 *RDOT)/R
      B3 = 0.5 * (B3 + B3STAR)
   18 CAPD1 = -(1.0 +(B3*DELT)/2.0 +((B2*A2-A1)*DELT**2)/4.0 + ((B1*A2
     1-A1*B3)*DELT**3)/8.0)
C
   19 CAPA1 = (1.0/CAPD1)*((-A1*DELT)+((A2*B1-A1*B3)*DELT**2)/2.0)
   20 CAPA2 = (1.0/CAPD1)*(((A2*B2-A1)*DELT**2)/2.0 + ((A2*B1-A1*B3)*
     1DELT**3)/4.0)
   21 CAPA3 = (-A2 * DELT) /CAPD1
   22 CAPA4 = (1.0 /CAPD1)*((A3*DELT) + (A3*B3*DELT**2)/2.0)
C
   23 CAPB1 = (1.0/CAPD1)*((B1*DELT)  + (A1*B2*DELT**2)/2.0)
   24 CAPB2 = (1.0/CAPD1)*((B2*DELT)  + (B1 *DELT**2)/2.0)
   25 CAPB3 = (1.0/CAPD1)*((B3*DELT)+ (A2*B2*DELT**2)/2.0 +((A2*B1-A1*
     1B3)*DELT**3)/4.0)
```

```
   26 CAPB4 = (1.0/CAPD1)*((-A3*DELT**2*B2)/2.0 - (A3*DELT**3*B1)/4.0)
C
   27 C = CSTAR
   28 CSTAR = MU/R**3
   29 C = 0.5 * (C + CSTAR)
   30 CAPD2 = 1.0 + (C)*(DELT**2/4.0)
   31 CAPC1 = -((C)*(DELT**2/(2.0*CAPD2)))
   32 CAPC2 = DELT/CAPD2
   33 CAPC3 = -((C)*(DELT/CAPD2))
C FROM CONSTANTS ABOVE WE WOW FORM THE PSI AND GAMMA MATRIX ELEMENTS
C                                        PSI ARRAY
      PSI(1,1) =1.0+(CAPA1*DELT)/2.0
      PSI(1,2) = 0.0
      PSI(1,3) = 0.0
      PSI(1,4) = (CAPA2/2.0 + 1.0)*DELT
      PSI(1,5) =(CAPA3 * DELT /2.0) * .001
      PSI(1,6) = 0.0
C
      PSI(2,1) =((CAPB1 * DELT)/2.0) * 1000.
      PSI(2,2) = 1.0
      PSI(2,3) = 0.0
      PSI(2,4) =((CAPB2*DELT)/2.0) * 1000.
      PSI(2,5) = (CAPB3/2.0 + 1.0) * DELT
      PSI(2,6) = 0.0
C
      PSI(3,1) = 0.0
      PSI(3,2) = 0.0
      PSI(3,3) = 1.0 + CAPC1
      PSI(3,4) = 0.0
      PSI(3,5) = 0.0
      PSI(3,6) = CAPC2
C
      PSI(4,1) = CAPA1
      PSI(4,2) = 0.0
      PSI(4,3) = 0.0
      PSI(4,4) = 1.0 + CAPA2
      PSI(4,5) = CAPA3 * .001
      PSI(4,6) = 0.0
C
      PSI(5,1) = CAPB1 * 1000.
      PSI(5,2) = 0.0
      PSI(5,3) = 0.0
      PSI(5,4) = CAPB2 * 1000.
      PSI(5,5) = 1.0 + CAPB3
      PSI(5,6) = 0.0
C
      PSI(6,1) = 0.0
      PSI(6,2) = 0.0
      PSI(6,3) = CAPC3
      PSI(6,4) = 0.0
      PSI(6,5) = 0.0
      PSI(6,6) = 1.0 + CAPC1
C                                        GAMMA ARRAY
      GAMMA(1,1) = (CAPA4 * DELT) /2.0
      GAMMA(2,1) =((CAPB4 * DELT) /2.0) * 1000.
      GAMMA(3,1) = 0.0
      GAMMA(4,1) = CAPA4
      GAMMA(5,1) = CAPB4 * 1000.
      GAMMA(6,1) = 0.0
C
```

```
C   CALCULATIONS FOR THE NEW PSI PRIME AND GAMMA PRIME MATRICES FOLLOW.
C
C   CALCULATIONS FOR THE PSI PRIME MATRIX
      DO 50 I = 1,6
      DO 50 J = 1,6
   50 SIDP  (I,J) = 0.0
      DO 55 I = 1,6
      DO 55 J = 1,6
      DO 55 MM = 1,6
      SIDP(I,J) = SIDP(I,J) + PSI(I,MM) * PSIP(MM,J)
   55 CONTINUE
      DO 67 I = 1,6
      DO 67 J = 1,6
   67 PSIP(I,J) = SIDP(I,J)
C
C   CALCULATIONS FOR THE GAMMA PRIME MATRIO
      DO 69 I = 1,6
   69 SIGAMP(I,1) = 0.0
      DO 65 I = 1,6
      DO 65 MM = 1,6
   65 SIGAMP(I,1) = SIGAMP(I,1) + PSI(I,MM) * GAMMAP(MM,1)
      DO 66 I = 1,6
   66 GAMMAP(I,1) = SIGAMP(I,1) + GAMMA(I,1)
C     THE RANDOM NOISE Q IS OBTAINED
      DO 40 I = 1,3
   40 Q(I) = 0.0
      DO 41 J = 1,3
      I = J + 3
   41 Q(I)=SIGMQ(J)*((SQRT(ALOG(1./RNUM(XX)**2)))*(SIN(TWOPI*RNUM(XX))))
      DO 42 I = 1,6
   42 PSIQ(I) = 0.0
      DO 43 I = 1,6
      DO 43 J = 1,6
   43 PSIQ(I) = PSIQ(I) + PSI(I,J) * QP(J)
      DO 44 I = 1,3
   44 QDELT(I) = 0.0
      DO 45 I = 1,3
      J = I + 3
   45 QDELT(J) = Q(J) * DELT
      DO 46 I = 1,6
   46 QP(I) = PSIQ(I) + QDELT(I)
      IF (NAVF .EQ. 0) GO TO 90
      IRESF = 1
   90 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TM
C  THIS SUBROUTINE IS A SIMPLE EARTH BASED TRACKING MODULE. RESULTS OF
C  THIS ROUTINE GIVE THE OBSERVATION MATRIX.
      COMMON/HMR/HM(4,6),RE
      COMMON/INI/MU,THEDTE,THETAE,DELT,R,RDOT,THETA,THEDT,H
      A = THETAE - THETA
      B = COS(A)
      C = SIN(A)
      D = THEDTE - THEDT
C  THE NEXT   STATEMENTS CALCULATE RANGE AND RANGE-RATE
      RNG = (R**2 + RE**2 - 2.0*R*RE*B)**0.5
      P = (1.0/RNG)
      RNGDT = P * ((R -RE*B)*RDOT + (R*RE*C*D))
C  THE NEXT STATEMENTS CALCULATE THE COSINE AND SIN OF RNG ANGLE-ALPHA
      SALPHA = (R*C)/RNG
      CALPHA = (R**2 - RE**2 - RNG**2) /(2.0*RE*RNG)
C  DETERMINE ELEMENTS OF THE OBSERVATION MATRIX. OBTAINED BY
C  PERTURBING RANGE AND RANGE ANGLE EQUATIONS.
      E1 = P * (R - RE*B)
      E2 = (-P)*(R*RE*C)
      F1 = P*((-RNGDT*E1)+RDOT+(RE*C*D))
      F2 = P*((-RNGDT*E2)- (RDOT*RE*C) - (R*RE*B*D))
      G1 = (C/(RNG*CALPHA))*(1.0 -(E1*R)/RNG)
      G2 = (-R/(RNG*CALPHA))* (B + (E2*C)/RNG)
C  SET UP THE ABOVE ELEMENTS IN AN ARRAY  FOR OBSERVATION MATRIX
      HM (1,1) = E1
      HM (1,2) = E2 * .001
      HM (1,3) = 0.0
      HM (1,4) = 0.0
      HM (1,5) = 0.0
      HM (1,6) = 0.0
      HM (2,1) = F1
      HM (2,2) = F2 * .001
      HM (2,3) = 0.0
      HM (2,4) = E1
      HM (2,5) = E2 *.001
      HM (2,6) = 0.0
      HM (3,1) = G1 * 1000.
      HM (3,2) = G2
      HM (3,3) = 0.0
      HM (3,4) = 0.0
      HM (3,5) = 0.0
      HM (3,6) = 0.0
      HM (4,1) = 0.0
      HM (4,2) = 0.0
      HM (4,3) = P * 1000.
      HM (4,4) = 0.0
      HM (4,5) = 0.0
      HM (4,6) = 0.0
      RETURN
      END
```

```
      SUBROUTINE DYNM
C   THIS DYNAMICS SUBROUTINE USES A SERIES-1 MODEL.  ITS PURPOSE IS TO
C   COMPUTE THE COVARIANCE MATRICES OF STATE, MIDCOURSE VELOCITY
C   CORRECTIONS, ERRORS IN THOSE CORRECTIONS, TARGET MISS ASSOCIATED
C   WITH THE CORRECTIONS, AND NAVIGATION ERRORS AT PRESCRIBED POINTS
C   ALOND THE NOMINAL TRAJECTORY, AND THE PROPAGATION OF THESE ERRORS
      DIMENSION  PSIXV(6,1), GAMR(6,1)
      COMMON/PSIMR/GAMMAP(6,1)
      COMMON/SYNRR/X(6,1),Q(6),QP(6)
      COMMON/DYNR/RS,SIGMQ(3)
      COMMON/POLR/K,KL,PSIP(6,6),PHI(6,6,6),KLL
      COMMON/INI/MU,THEDTE,THETAE,DELT,R,RDOT,THETA,THEDT,H
      COMMON/CONTR/EU(3,1),EM(3,1),DELTVP(3,1),PEU(3,3),PEM(3,3)
      COMMON/FLAGS/MCCF,NAVF,IGUIDF,IGPRF,IRESF
C
C     CHECK THE MIDCOURSE CORRECTION FLAG, IF IT IS EQUAL TO ONE UPDATE THE
C     DYNAMICS DATA AFTER MIDCOURSE CORRECTION
C     IF MCCF IS NOT EQUAL TO ONE UPDATE THE DYNAMICS DATA AFTER A NAVIGATION
C     TIME ADVANCE
C
      IF(MCCF-1) 10,12,10
C
C     CALCULATE X ON THE FIRST PASS WHERE X = PSI*X + GAMMA*R + Q
C
   10 DO 11 I = 1, 6
   11 PSIXV(I,1) = 0.0
C
C     MULTIPLE THE PSI MATRIX BY THE OLD X VALUE AND PLACE THE RESULTS IN
C     PSIXV
C
      DO 16  I =1,6
      DO 16  J =1,6
   16 PSIXV(I,1)= PSIXV(I,1) + PSIP(I,J)*X (J,1)
C
C     MULTIPLE THE GAMMA MATRIX BY RS, A VECTOR OF PARAMETER ERRORS, AND PLACE
C     THE RESULTS IN GAMR
C
      DO 13 I =1,6
      GAMR(I,1) =GAMMAP(I,1)*RS
C
C     THE NEW X IS CALCULATED BY ADDING PSIXV + GAMR + Q
   13 X(I,1) = PSIXV(I,1) + GAMR(I,1) + QP(I)
      GO TO 20
C
C     CALCULATE X FOLLOWING THE MIDCOURSE CORRECTION, ON A SECOND PASS USING
C     THE FORMAULA X = X + THE VECTOR WHERE THE ELEMENTS (1,1), (2,1), (3,1)
C     EQUAL TO ZERO AND THE ELEMENTS (4,1), (5,1), (6,1) EQUAL TO THE DELTA
C     PRIME VECTOR
C
   12 X(4,1)=X(4,1) + DELTVP(1,1)
      X(5,1) = X(5,1) + (DELTVP(2,1)/R) * 1000.0
      X(6,1) = X(6,1) + DELTVP(3,1)
   20 CONTINUE
C     CALL THE OBSERVATION MODULE
      CALL OBSM
      RETURN
      END
```

```
      SUBROUTINE OBSM
C   THIS OBSERVATION SUBROUTINE COMPUTES THE OBSERVATIONAL DATA FROM
C   THE STATE DATA.  THE MODEL USED IN THIS SUBROUTINE IS MODEL-1 AND
C   IS A DO NOTHING MODEL)  MODEL-2 IS A MORE SOPHISTICATED MODEL.
C
      DIMENSION HX(4,1),ALB(4,1)
      COMMON/OB/Z(4,1),V(4)
      COMMON/FLAGS/MCCF,NAVF,IGUIDF,IGPRF,IRESF
      COMMON/SYNRR/X(6,1),Q(6),QP(6)
      COMMON/HMR/HM(4,6),RE
      COMMON/ITI/AL(4,6),B(6,1),NM,SIGMA(4)
      DATA TWOPI/6.28318/
C
C     CHECK NAVIGATION FLAG, IF NOT ONE GO ON TO THE NAVIGATION MODULE
C     IF ONE CHECK THE MIDCOURSE CORRECTION FLAG
C
      IF(NAVF-1)10,12,10
C
C     CHECK MIDCOURSE CORRECTION FLAG, IF IT IS ONE CONTINUE ON TO THE
C     NAVIGATION MODULE, IF IS ISN,T ONE COMPUTE THE OBSERVATION DATA
C
   12 IF(MCCF-1)11,10,11
C
C     CALCULAGE Z WHERE Z = H*X + L*B + V
C
   11 DO 14 I = 1,4
      HX(I,1)= 0.0
   14 ALB(I,1)= 0.0
C
C     THE H MATRIX IS MULTIPLIED BY THE X MATRIX AND THE RESULT IS PLACED IN
C     HX
      DO 17 I = 1,4
      DO 17 J = 1,6
   17 HX(I,1) =HX(I,1) +HM(I,J)* X(J,1)
C
C     THE L MATRIX IS MULTIPLIED BY THE B MATRIX AND THE RESULT IS PLACED IN
C     ALB
      DO 15 I = 1,4
      DO 15 J = 1,NM
   15 ALB(I,1) = ALB(I,1) + AL(I,J)*B(J,1)
C
C     V WHICH IS A RANDOM NUMBER IS OBTAINED
C
      DO 16 I = 1,4
      V(I)=SIGMA(I)*((SQRT(ALOG(1./RNUM(XX)**2)))*(SIN(TWOPI*RNUM(XX))))
C
C     Z IS CALCULATED
C     HX + ALB + V IS EQUAL TO Z
C
   16 Z(I,1) = HX(I,1)+ALB(I,1) +V(I)
   10 CONTINUE
C     CALL THE NAVIGATION MODULE
      CALL NAVM
      RETURN
      END
```

```
      SUBROUTINE  NAVM
C    THIS NAVIGATION SUBROUTINE IS A COMPANION TO THE DYNAMICS MODULE
C    AND IT TOO IS A SERIES-1 MODEL.  ITS PURPOSE IS TO COMPUTE THE
C    STATE ESTIMATE AND RELATED DATA.
C
      DIMENSION PSIB(6,6), PSPPT(6,6),PSIT(6,6),AIDEN(6,6),HPHP(4,4),HPP
     1HT(4,4),PPHT(6,4),PP(6,6),HT(6,4),HPHPI(4,4),AKH(6,6),XHP(6,1),TEM
     2P(6,6),HXH(4,1),TEMPZ(4,1),AKZHXH(6,1),DVEM(3,1)
      COMMON/OB/Z(4,1),V(4)
      COMMON/FLAGS/MCCF,NAVF,IGUIDF,IGPRF,IRESF
      COMMON/NA/AK(6,4),PHIST(6,6)
      COMMON/CONTR/EU(3,1),EM(3,1),DELTVP(3,1),PEU(3,3),PEM(3,3)
      COMMON/POLR/K,KL,PSIP(6,6),PHI(6,6,6),KLL
      COMMON/SYNRR/X(6,1),Q(6),QP(6)
      COMMON/NAVR/XH(6,1),XAPX(6,1)
      COMMON/GUIDR/EN(3,1),DELTV(3,1),DVMAG
      COMMON/FIRS/P(6,6),QQ(6,6),PV(4,4)
      COMMON/INI/MU,THEDTE,THETAE,DELT,R,RDOT,THETA,THEDT,H
      COMMON/HMR/HM(4,6),RE
      COMMON/TTRC/TRP,TRV
C     CHECK NAVIGATION FLAG. IF ZERO,COMPUTE NAVIGATION DATA,IF NOT ZERO
C     CHECK MCCF FLAG. IF ONE, COMPUTE NAVIGATION DATA, IF NOT ONE, MODIFY
C    NAVIGATION DATA AFTER A MIDCOURSE CORRECTION
C
      IF(NAVF.EQ.0)  GO TO 11
   10 IF(MCCF.EQ.1) GO TO 13
   12 DO 14 I = 1,6
      DO 14 J = 1,6
C     FIND PSIP TRANSPOSE
      PSIT(I,J) = PSIP(J,I)
      PSIB(I,J) = 0.0
      PSPPT(I,J) = 0.0
      IF(I-J)15,16,15
   16 AIDEN(I,J) = 1.
      GO TO 14
   15 AIDEN(I,J) = 0.0
   14 CONTINUE
C     CREATE AIDEN AS A 6 BY 6 IDENITY MATRIX
C     CALCULATES P PRIME
C     P PRIME = PSIP*P*PSIP TRANSPOSE + Q
C     MULTIPLY MATRIX PSIP BY MATRIX P AND PUT RESULT IN PSIB
      DO 18 I = 1,6
      DO 18 J = 1,6
      DO 18 KK= 1,6
   18 PSIB(I,J)=PSIB(I,J) +PSIP(I,KK)*P(KK,J)
C     MULTIPLY THE ABOVE PSIB BY THE TRANSPOSE OF PSIP
      DO 19 I =1,6
      DO 19 J =1,6
      DO 19 KK=1,6
   19 PSPPT(I,J) = PSPPT(I,J) + PSIB(I,KK)*PSIT(KK,J)
      DO 20 I =1,6
      DO 20 J =1,6
C     CALCULATES P PRIME
   20 PP(I,J) = PSPPT(I,J) +QQ(I,J)
C
C     CALCULATES K
C     K = P PRIME * H TRANPOSE * THE INVERSE OF (H * P PRIME * H TRANSPOSE
C
      DO 21 I =1,6
      DO 21 J =1,4
```

```
C       FINDS THE TRANSPOSE OF H MATRIX
        HT(I,J) =HM(J,I)
   21 PPHT(I,J)=0.0
C       MULTIPLY P PRIME BY H TRANSPOSE AND PUT RESULTS IN PPHT
        DO 23 I =1,6
        DO 23 J =1,4
        DO 23 KK=1,6
   23 PPHT(I,J)= PPHT(I,J) + PP(I,KK)*HT(KK,J)
        DO 24 I =1,4
        DO 24 J =1,4
   24 HPPHT(I,J) =0.0
C       MULTIPLIES H BY THE ABOVE PPHT  AND PUTS RESULTS INTO HPPHT
        DO 25 I =1,4
        DO 25 J =1,4
        DO 25 KK=1,6
   25 HPPHT(I,J)=HPPHT(I,J) + HM(I,KK)*PPHT(KK,J)
C       ADDS HPPHT TO PV
        DO 26 I = 1,4
        DO 26 J = 1,4
   26 HPHP(I,J) = HPPHT(I,J)+PV(I,J)
C       SUBROUTINE MATIN TAKES INVERSE OF HPHP, AND PLACES THE RESULT IN HPHP
C       4 IS THE DIMENSION OF HPHP
        CALL MATIN (HPHP,HPHPI,4)
        DO 28 I =1,6
        DO 28 J =1,4
   28 AK(I,J) = 0.0
        DO 29 I =1,6
        DO 29 J =1,4
        DO 29 KK=1,4
   29 AK(I,J) = AK(I,J)+PPHT(I,KK)*HPHPI(KK,J)
C       CALCULATES P
C       P = (I-K*H)*P PRIME
C       CALCULATES K*H
        DO 60 I = 1,6
        DO 60 J = 1,6
   60 AKH(I,J) = 0.0
        DO 30 I =1,6
        DO 30 J =1,6
        DO 30 KK=1,4
   30 AKH(I,J) = AKH(I,J) +AK(I,KK)* HM(KK,J)
C       SUBTRACT K*H FROM THE IDENITY MATRIX
        DO 32 I = 1,6
        DO 32 J = 1,6
        TEMP(I,J)= AIDEN(I,J)-AKH(I,J)
   32 P(I,J) = 0.0
        DO 33 I = 1,6
        DO 33 J = 1,6
        DO 33 KK= 1,6
   33 P(I,J) = P(I,J) + TEMP(I,KK)*PP(KK,J)
        DO 80 I = 1,6
        DO 80 J = 1,6
        IF (I - J) 82,80,82
   82 P(I,J) = (P(I,J) + P(J,I))/2.0
        P(J,I) = P(I,J)
   80 CONTINUE
        TRP = P(1,1) + P(2,2)*R*R*1.0E-06  + P(3,3)
        TRV = P(4,4) + P(5,5)*R*R*1.0E-06  + P(6,6)
C       CALCULATES X HAT
C       X HAT = X HAT PRIME + K(Z-H*X HAT)
        DO 34 I =1,6
```

```
   34 XHP(I,1) =0.0
      DO 35 I =1,6
      DO 35 J =1,6
C     CALCULATES X HAT PRIME
C     X HAT PRIME = PSIP * X HAT
   35 XHP(I,1) = XHP(I,1) + PSIP(I,J)*XH(J,1)
      DO 36 I = 1,4
   36 HXH(I,1) =0.0
      DO 37 I =1,4
      DO 37 J =1,6
   37 HXH(I,1)= HXH(I,1) + HM(I,J)*XHP(J,1)
      DO 38 I = 1,4
C     CALCULATES Z - THE ABOVE H * X HAT PRIME
      TEMPZ(I,1) =  Z(I,1) - HXH(I,1)
      AKZHXH(1,1)= 0.0
   38 AKZHXH(I+2,1)=0.0
      DO 40 I =1,6
      DO 40 J =1,4
C     MULTIPLIES K TIMES THE ABOVE Z - H*X HAT PRIME
   40 AKZHXH(I,1) = AKZHXH(I,1) + AK(I,J)*TEMPZ(J,1)
      DO 41 I =1,6
C     ADD X HAT PRIME TO K*(Z-H*X HAT PRIME)
      XH(I,1)=XHP(I,1)+ AKZHXH(I,1)
C     CALCULATES X APPROXIMATE
   41 XAPX(I,1) = X(I,1)- XH(I,1)
C     X APPROXIMATE = X - X HAT
      GO TO 11
C
C     SECOND PASS
C     UPDATE X HAT, X APPROXIMATE, AND THE P MATRIX FOLLOWING A MIDCOURSE
C     CORRECTION
C
   13 XH(4,1) = XH(4,1) + DELTV(1,1) + EM(1,1)
      XH(5,1) = XH(5,1) + ((DELTV(2,1) + EM(2,1))/R) * 1000.0
      XH(6,1) = XH(6,1) + DELTV(3,1) + EM(3,1)
      XAPX(4,1) = XAPX(4,1) + EU(1,1)
      XAPX(5,1) = XAPX(5,1) + (EU(2,1)/R) * 1000.
      XAPX(6,1) = XAPX(6,1) + EU(3,1)
  200 CONTINUE
      DO 50 I =1,3
      PEU(2,1) = PEU(2,I) * (1000.0/R)
      PEU(1,2) = PEU(I,2) * (1000.0/R)
   50 CONTINUE
      DO 45 I =1,3
      DO 45 J =1,3
      IAI = I + 3
      JAJ = J + 3
   45 P(IAI,JAJ) = P(IAI,JAJ) + PEU(I,J)
   11 CONTINUE
C     CALL GUIDANCE  MODULE
      CALL GUIDM
      RETURN
      END
```

```
      SUBROUTINE  GUIDM
C    THIS GUIDANCE MODULE USES A SERIES-1 MODEL AND IS A COMPANION TO THE
C    DYMAMICS MODULE.  ITS MAIN FUNCTION IS TO COMPUTE THE REQUIRED
C    MIDCOURSE GUIDANCE CORRECTIONS AND OTHER RELATED DATA SUCH AS THE
C    COVARIANCE MATRIX OF THE ERROR IN THE CORRECTION DUE TO THE
C    NAVIGATION UNCERTAINTITY.
C
      COMMON/INI/MU,THEDTE,THETAE,DELT,R,RDOT,THETA,THEDT,H
      COMMON/GUIDR/EN(3,1),DELTV(3,1),DVMAG
      COMMON/POL/G(6,6,6)
      COMMON/POLR/K,KL,PSIP(6,6),PHI(6,6,6),KLL
      COMMON/NAVR/XH(6,1),XAPX(6,1)
      COMMON/FLAGS/MCCF,NAVF,IGUIDF,IGPRF,IRESF
C
C     CHECK THE MISCOURSE CORRECTION FLAG, IF EQUAL TO ONE GO ON TO THE
C     CONTROL MODULE,  IF NOT EQUAL TO ONE
C     CHECK THE GUIDANCE FLAG, IF IT IS EQUAL TO ONE COMPUTE MIDCOURSE
C     CORRECTION DATA, IF IT IS NOT ONE CONTINUE ON TO THE CONTROL MODULE
C
      IF(MCCF.EQ.1) GO TO 7
    8 IF(IGUIDF - 1) 7,15,7
   15 CONTINUE
      IF(K - KL) 6,20,6
   20 CONTINUE
      IGUIDF = 0
      GO TO 7
C
C     CALCULATE THE EN MATRIX, THE DELTA V MATRIX, AND THE MAGNITUDE OF THE
C     DELTA V MATRIX OR VECTOR
C
    6 DO 10 I = 1,3
      DELTV (1,1)=0.0
   10 EN(I,1) = 0.0
C
C     MULTIPLE G MATRIX BY THE APPROXIMATE X MATRIX AND PUT THE RESULTS IN
C     THE MATRIX CALLED EN
C     MULTIPLE THE G MATRIX BY THE X HAT MATRIX AND PUT THE RESULTS
C     IN THE MATRIX CALLED DELTV
C
      DO 11 I = 1,3
      DO 11 J = 1,6
      EN(I,1) = EN(I,1) + G(K,I,J)*XAPX(J,1)
   11 DELTV (I,1)= DELTV (I,1) + G(K,I,J)*XH(J,1)
      EN(2,1) = EN(2,1) * R* 0.001
      DELTV(2,1) = DELTV(2,1) * R*0.001
C
C     FIND THE MAGNITUDE OF THE  DELTA V VECTOR
C
      DVMAG = SQRT(DELTV (1,1)**2+DELTV (2,1)**2+DELTV (3,1)**2)
    7 CONTINUE
C     CALL THE CONTROL MODULE
      CALL CONTM
      RETURN
      END
```

```
      SUBROUTINE CONTM
C     THIS CONTROL MODULE IS A SERIES-1 MODEL AND IS A COMPANION TO THE
C     DYNAMICS MODEL.  THE PURPOSE OF THIS ROUTINE IS TO COMPUTE
C     COVARIANCE MATRICES OF MONITORED AND UNMONITORED ERRORS MADE IN
C     EXECUTING THE MIDCOURSE GUIDANCE COMMAND DELTA-V
C
      DIMENSION DEVTVD(3,1),E5DEL(3,1),E4DEL(3,1),CP(3,1)
      DIMENSION DELTVT(1,3),DELDELT(3,3),PEU1(3,3),PEU2(3,3)
      COMMON/CONTR/EU(3,1),EM(3,1),DELTVP(3,1),PEU(3,3),PEM(3,3)
      COMMON/GUIDR/EN(3,1),DELTV(3,1),DVMAG
      COMMON/INTA/E(6),SIGMA4,SIGMA5,SIGMA6,SIGMAP
      COMMON/FLAGS/MCCF,NAVF,IGUIDF,IGPRF,IRESF
C
C     CHECK THE MIDCOURSE CORRECTION FLAG, IF ONE RESET THE MIDCOURSE [redacted]
C     FLAG AND THE GUIDANCE FLAG TO ZERO, THEN CONTINUE ON TO THE [redacted]
C     MODULE
C
      IF(MCCF-1)5,4,5
    4 MCCF = 0
      GO TO 3
C
C     IF NOT ONE, CHECK THE GUIDANCE FLAG, IF IT ISN,T ONE CONTINUE ON TO [redacted]
C     NEXT MODULE
C     IF IT IS ONE COMPUTE THE MIDCOURSE CORRECTION EXECUTION ERROR DATA
C
    5 IF(IGUIDF -1) 3,2,3
C
C     CALCULATE THE EU, THE EM, THE PEU, THE PEM, AND THE DELTA V PRIME [redacted]
C
    2 DO 6 I = 1,3
      DEVTVD(I,1) = DELTV(I,1)/DVMAG
      E5DEL(I,1)  = E(5)*DEVTVD(I,1)
    6 E4DEL(I,1)  = E(4)*DELTV(I,1)
      CP(1,1) = E(2)*DELTV(3,1)- E(3)*DELTV(2,1)
      CP(2,1) = E(3)*DELTV(1,1)- E(1)*DELTV(3,1)
      CP(3,1) = E(1)*DELTV(2,1)- E(2)*DELTV(1,1)
      DO 7 I = 1,3
C
C          E3                                          THE DELTA V
C     EU = E2 X DELTA V + E4 * DELTA V + E5 * ----------------------------
C          E1                                          THE MAGNITUDE OF DELTA V
C
      EU(I,1)=  CP(I,1) + E4DEL(I,1) + E5DEL(I,1)
C
C                          DELTA V
C     EM = E6 * ------------------------------
C                     THE MAGNITUDE OF DELTA V
C
      EM(I,1)=  E(6)*DEVTVD(I,1)
C
C     CALCULATE DELTA V PRIME
C
    7 DELTVP(I,1) = DELTV(I,1) + EU(I,1) + EM(I,1)
C
C     CALCULATE PEU
C     CALCULATE PEM
C
      DO 50 I = 1,3
   50 DELTVT(1,I) = DELTV(I,1)
      DO 51 I = 1,3
```

```
      DO 51 J = 1,3
      DELDELT(I,J) = DELTV(I,1)*DELTVT(1,J)
      PEU1(I,J) = DELDELT(I,J)*((SIGMA4**2)-(SIGMAP**2)+((SIGMA5**2)/(DV
     .MAG**2)))
      IF(I-J) 52,53,52
   53 PEU2(I,J) = (SIGMAP**2)*(DVMAG**2)
      GO TO 55
   52 PEU2(I,J) = 0.0
   55 PEU(I,J) = PEU1(I,J) + PEU2(I,J)
   51 PEM(I,J) = DELDELT(I,J) *(SIGMA6**2/DVMAG**2)
      MCCF=1
    3 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE POLM
C THIS SUBROUTINE IS THE POLICY MODULE, ITS PURPOSE IS TO GENERATE
C MIDCOURSE CORRECTION MATRICES IN ACCORDANCE WITH THE GUIDANCE POLICY
      DIMENSION SIDP(6,6),PHSS1(3,3),PHSS1I(3,3),PHSS2(3,3),PHSS3(3,6)
      COMMON/POLR/K,KL,PSIP(6,6),PHI(6,6,6),KLL
      COMMON/POL/G(6,6,6)
      COMMON/NA/AK(6,4),PHIST(6,6)
      COMMON/FLAGG/IPRNTF,IIESN,IIGEV,IESNX,IEVF,PHISS(6,6,6)
C MAKE FLAG CHECK FOR K FROM INILIZATION PASS
      IF (K-1) 5,20,5
    5 CONTINUE
C TO COMPUTE TRANSITION MATRIX, ELEMENTS ARE OBTAINED FROM PSIM MODULE
      DO 10 I = 1,6
      DO 10 J = 1,6
   10 PHI(K-1,I,J) = PSIP(I,J)
   20 CONTINUE
C NOW MAKE A CHECK FOR K=KL, IF K=KL THEN WE COMPUTE G MATRIX
      IF(K-KL) 70,30,70
C COMPUTE IDENTITY MATRIX FOR PHI STAR
   30 DO 47 I = 1,6
      DO 47 J = 1,6
      IF (I-J) 45,40,45
   40 PHIST (I,J) = 1.0
      GO TO 47
   45 PHIST (I,J) = 0.0
   47 CONTINUE
C THE FOLLOWING QUANTITY OF STATEMENTS WILL INVOLVE DO LOOPS IN ORDER TO
C OBTAIN THE G MATRIX
      M = KL - 1
      DO 62 L = 1,M
      KLL = KL - L
C  A DO LOOP TO OBTAIN PHI STAR-STAR
      DO 50 I = 1,6
      DO 50 J = 1,6
   50 PHISS(KLL ,I,J) = 0.0
      DO 51 I = 1,6
      DO 51 J = 1,6
      DO 51 N = 1,6
   51 PHISS(KLL ,I,J) = PHISS(KLL ,I,J) + PHIST(I,N)*PHI(KLL ,N,J)
C  SET PHI-STAR EQUAL TO PHI STAR-STAR
      DO 52 I =1,6
      DO 52 J =1,6
   52 PHIST (I,J)  = PHISS(KLL ,I,J)
C  NOW CALCULATE PARTS OF THE G MATRIX
C  THE FIRST PART WILL BE CALLED PHSS1, A 3X3 MATRIX
      DO 53 I = 1,3
      DO 53 J = 1,3
      JJ = J + 3
   53 PHSS1(I,J) = PHIST(I,JJ)
C  NOW TAKE THE INVERSE OF PHSS1
      CALL MATIN (PHSS1, PHSS1I,3)
C  UPON RETURN FROM MATIN PHSS1I IS THE INVERSE AND IS 3X3
C  THE NEXT CALCULATION WILL DO THE 2ND 3X3 MATRIX AND IS PHSS2
      DO 55 I = 1,3
      DO 55 J = 1,3
   55 PHSS2(I,J) = PHIST(I,J)
C  NOW MULTIPLY  THE 2 MATRICES,  PHSS1I * PHSS2
      DO 56 I =1,3
      DO 56 J =1,3
```

```
   56 PHSS3 (I,J) = 0.0
      DO 57  I = 1,3
      DO 57  J = 1,3
      DO 57  N = 1,3
   57 PHSS3 (I,J) =PHSS3(I,J)  +  PHSS1I (I,N) * PHSS2(N,J)
      PHSS3(1,4) = 1.0
      PHSS3(1,5) = 0.0
      PHSS3(1,6) = 0.0
      PHSS3(2,4) = 0.0
      PHSS3(2,5) = 1.0
      PHSS3(2,6) = 0.0
      PHSS3(3,4) = 0.0
      PHSS3(3,5) = 0.0
      PHSS3(3,6) = 1.0
C ALL THE CALCULATIONS ARE COMPLETED FOR THE G MATRIX,WE NOW PUT ALL
C THE ELEMENTS TOGETHER
      DO 80 I = 1,3
      DO 80 J = 1,6
   80 G(KLL,I,J) = -PHSS3(I,J)
   62 CONTINUE
   70 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INFM
C  THIS SUBROUTINE IS THE INFORMATION MODULE, ALL PRINT-OUT IS
C  CONTROLLED BY THIS ROUTINE
C
      COMMON/MAIN/TD,TDSV
      COMMON/POL/G(6,6,6)
      COMMON/INI/MU,THEDTE,THETAE,DELT,R,RDOT,THETA,THEDT,H
      COMMON/FLAGS/MCCF,NAVF,IGUIDF,IGPRF,IRESF
      COMMON/SYNRR/X(6,1),Q(6),QP(6)
      COMMON/PSIMR/GAMMAP(6,1)
      COMMON/POLR/K,KL,PSIP(6,6),PHI(6,6,6),KLL
      COMMON/HMR/HM(4,6),RE
      COMMON/OB/Z(4,1),V(4)
      COMMON/NAVR/XH(6,1),XAPX(6,1)
      COMMON/FIRS/P(6,6),QQ(6,6),PV(4,4)
      COMMON/NA/AK(6,4),PHIST(6,6)
      COMMON/GUIDR/EN(3,1),DELTV(3,1),DVMAG
      COMMON/CONTR/EU(3,1),EM(3,1),DELTVP(3,1),PEU(3,3),PEM(3,3)
      COMMON/FLAGG/IPRNTF,IIESN,IIGEV,IESNX,IEVF,PHISS(6,6,6)
      COMMON/FINIF/INIF
      COMMON/TTRC/TRP,TRV
C
C  CHECK FOR PRINT FLAG, IF 1 CHECK EVENT FLAG, IF 0 GO TO 5 AND EXIT
C
      IF (IPRNTF - 1) 5,10,5
C
C  STATEMENT 10 CHECKS EVENT FLAG, IF 1 WRITE OUT EVENT NUMBER
C  IF 0 WRITE OUT TRAJECTORY DATA FROM NTM
C
   10 CONTINUE
      WRITE(6,300)
  300 FORMAT(1H1)
      IF(IEVF - 1) 13,40,13
C
   40 WRITE(6,215) IIESN
  215 FORMAT(10X,15HEVENT NUMBER = ,I5)
   13 CONTINUE
      WRITE(6,200)
  200 FORMAT( ///17X,2HTD,18X,1HR,16X,5HTHETA,14X,4HRDOT,15X,5HTHEDT,13X
     .,6HTHETAE/)
      WRITE(6,201) TD,R,THETA,RDOT,THEDT,THETAE
  201 FORMAT(9X,6(E18.9,1X))
C
C  CHECK GUIDANCE PRESET FLAG, IF 1 WRITE OUT PHI-MATRIX CALCULATED IN
C  POLM, IF 0 WRITE OUT STATE-X
C
      IF(IGPRF - 1) 6,20,6
    6 CONTINUE
      IF(NAVF - 1) 5,83,5
   83 CONTINUE
      WRITE(6,202)
  202 FORMAT(/////50X,14HSTATE VECTOR X/)
      WRITE(6,203) (X(I,1),I =1,6)
  203 FORMAT(45X,E18.9/)
      GO TO 23
   20 CONTINUE
      IF(IIESN - IIGEV) 14,60,14
   60 CONTINUE
      IF(K - 1) 61,14,61
   61 WRITE (6,103)
```

```
  103 FORMAT (/////40X,10HPHI MATRIX//)
      WRITE (6,104) ((PHI(K-1,I,J),J=1,6),I=1,6)
  104 FORMAT (6E18.9/)
C
C  CHECK- IS IIESN = IESNX, IF EQUAL WRITE OUT CUMULATIVE PHI-STAR
C  AND G MATRIX, IF NOT GO TO 14 AND CONTINUE
      IF(IIESN - IESNX) 14,62,14
   62 CONTINUE
      KE = KL - 1
      DO 15 II = 1,KE
      WRITE (6,105) II
  105 FORMAT(///10X,17HGUIDANCE EVENT = ,I5)
      WRITE (6,106)
  106 FORMAT(/////40X,20HPHI STAR STAR MATRIX//)
      WRITE (6,107) ((PHISS(II,I,J),J=1,6),I=1,6)
  107 FORMAT(6E18.9/)
      WRITE (6,108)
  108 FORMAT(/////40X,17HGUIDANCE MATRIX-G//)
      WRITE (6,109) ((G(II,I,J),J=1,6),I=1,3)
  109 FORMAT(6E18.9/)
   15 CONTINUE
      IGPRF = 0
C
      GO TO 14
C
C  CHECK INIF FLAG, IF 0 WRITE OUT PSIP MATRIX, GAMMAP VECTOR, AND
C  QP VECTOR ALSO OBSERVATION MATRIX-HM, OBSERVATION VECTOR-Z AND
C  RANDOM NOISE VECTOR V, OPTIMAL GAIN MATRIX-K, ESTIMATE OF THE
C  STATE-XH, AND COVARIANCE MATRIX-P,  IF 1 WRITE OUT XH AND P.
C
   23 IF(INIF .EQ. 0) GO TO 81
  400 CONTINUE
      GO TO 84
   81 WRITE (6,94)
   94 FORMAT(/////40X,23HPSI PRIME MATRIX (PSIP)//)
      WRITE (6,95) ((PSIP(I,J),J=1,6),I=1,6)
   95 FORMAT (6E18.9/)
      WRITE (6,98)
   98 FORMAT(/////40X,27HGAMMA PRIME MATRIX (GAMMAP)//)
      WRITE(6,99) (GAMMAP(I,1), I=1,6)
   99 FORMAT(45X,E18.9/)
      WRITE(6,90)
   90 FORMAT(/////40X,23HCUMULATIVE NOISE VECTOR/)
      WRITE(6,91) (QP(I), I =1,6)
   91 FORMAT(45X,E18.9/)
   25 WRITE (6,100)
  100 FORMAT (/////40X,21HOBSERVATION MATRIX-HM//)
      WRITE (6,101) ((HM(I,J),J=1,6),I=1,4)
  101 FORMAT (6E18.9/)
      WRITE(6,206)
  206 FORMAT(/////10X,20HOBSERVATION VECTOR Z,20X,21HRANDOM NOISE VECTOR
     . V/)
      WRITE(6,207)(Z(I,1),V(I),I = 1,4)
  207 FORMAT(12X,E18.9,20X,E18.9)
      WRITE(6,212)
  212 FORMAT(/////40X,21HOPTIMAL GAIN MATRIX K/)
      WRITE(6,213) ((AK(I,J),J =1,4),I=1,6)
  213 FORMAT(20X,4E18.9/)
   84 WRITE(6,208)
  208 FORMAT(/////10X,27HESTIMATE OF STATE VECTOR XH/)
```

```
      WRITE(6,209)(XH(I,1),I = 1,6)
  209 FORMAT(10X,E18.9/)
      WRITE(6,210)
  210 FORMAT(/////42X,19HCOVARIANCE MATRIX P/)
      WRITE(6,211)((P(I,J),J = 1,6),I = 1,6)
  211 FORMAT(6E18.9/)
      WRITE(6,500) TRP
  500 FORMAT(/////5X,23HPOSITION TRACE (TRP) = ,E18.9)
      WRITE(6,501) TRV
  501 FORMAT(///5X,23HVELOCITY TRACE (TRV) = ,E18.9)
C
C  CHECK GUIDANCE FLAG, IF 1 CHECK MIDCOURSE CORRECTION FLAG, IF 0
C  GO TO 5 AND CONTINUE
C
      IF (IGUIDF - 1) 5,30,5
C
C  CHECK MIDCOURSE CORRECTION FLAG
C  IF THE MIDCOURSE CORRECTION FLAG IS 1 WRITE OUT DELTV FROM GUIDM,
C  AND EV,EM,PEU FROM CONTM MODULE, IF 0 WRUTE MIDCOURES CORRECTION
C  TIME OF UPDATE
C
   30 IF(MCCF - 1) 12,35,12
   12 WRITE(6,214)
  214 FORMAT(/////50X,27HMIDCOURSE CORRECTION UPDATE/////)
      GO TO 5
C
   35 WRITE(6,216)
  216 FORMAT(/////10X,35HMIDCOURSE CORRECTION VECTOR DELTA V/)
      WRITE(6,209)(DELTV(I,1), I=1,3)
      WRITE(6,217)
  217 FORMAT(/////20X,9HEU VECTOR,20X,9HEM VECTOR,20X,9HEN VECTOR/)
      WRITE(6,218) (EU(I,1),EM(I,1),EN(I,1), I=1,3)
  218 FORMAT(15X,E18.9,15X,E18.9,15X,E18.9/)
      WRITE(6,219)
  219 FORMAT(/////50X,10HPEU MATRIX/)
      WRITE(6,220)((PEU(I,J),J=1,3),I = 1,3)
  220 FORMAT(27X,3E18.9/)
      GO TO 14
C
C  STATEMENT 5 SETS PRINT FLAG TO 0
C
    5 IPRNTF = 0
   14 CONTINUE
      CALL EXTRA
      RETURN
      END
```

```
      SUBROUTINE DATIN
C  THIS ROUTINE READS IN ALL INPUT AND INITIALIZATION DATA
      COMMON/PCLR/K,KL,PSIP(6,6),PHI(6,6,6),KLL
      COMMON/PSIR/A1STAR,A2STAR,A3STAR,B1STAR,B2STAR,B3STAR,CSTAR
      COMMON/INI/MU,THEDTE,THETAE,DELT,R,RDOT,THETA,THEDT,H
      COMMON/MAIN/TD,TDSV
      COMMON/FIRS/P(6,6),QQ(6,6),PV(4,4)
      COMMON/SYNRR/X(6,1),Q(6),QP(6)
      COMMON/DYNR/RS,SIGMQ(3)
      COMMON/HMR/HM(4,6),RE
      COMMON/ITI/AL(4,6),B(6,1),NM,SIGMA(4)
      COMMON/FLAGS/MCCF,NAVF,IGUIDF,IGPRF,IRESF
      COMMON/INTA/E(6),SIGMA4,SIGMA5,SIGMA6,SIGMAP
      COMMON/NAVR/XH(6,1),XAPX(6,1)
      COMMON/FLAGG/IPRNTF,IIESN,IIGEV,IESNX,IEVF,PHISS(6,6,6)
      COMMON/INPT/IPT,DLTT,DELTN
      REAL MU
C  NTM INITIAL DATA
      READ(5,100) THEDTE,THETAE,THETA,THEDT
  100 FORMAT(4E13.6)
      READ(5,101) MU,R,RDOT
  101 FORMAT(3E13.6)
C
C  TM INITIAL DATA
      READ(5,102) RE
  102 FORMAT( E13.6)
C
C  DYNM INITIAL DATA
      READ(5,103) (X(I,1), I =1,6)
  103 FORMAT(6E13.6)
      READ (5,104) RS, (SIGMQ(I), I =1,3)
  104 FORMAT(4E13.6)
C
C  OBSM INITIAL DATA
      READ(5,105) NM, (B(I,1), I =1,4)
  105 FORMAT(I5,8X,4E13.6)
      READ (5,106) (SIGMA(I), I =1,4)
  106 FORMAT(4E13.6)
C
C  NAVM INITIAL DATA
      READ(5,107) (XH(I,1), I =1,6)
  107 FORMAT(6E13.6)
      READ(5,108) ((P(I,J), J=1,6), I=1,6)
  108 FORMAT(6E13.6)
      READ(5,109) ((QQ(I,J), J=1,6), I=1,6)
  109 FORMAT(6E13.6)
      READ(5,110) ((PV(I,J), J=1,4), I=1,4)
  110 FORMAT(4E13.6)
C
C  CONTM INITIAL DATA
      READ(5,111) (E(I), I =1,6)
  111 FORMAT(6E13.6)
      READ(5,125) SIGMA4,SIGMA5,SIGMA6,SIGMAP
  125 FORMAT(4E13.6)
C
C  NOW WRITE OUT INITIAL INPUT DATA
      WRITE(6,112) IPT,DLTT,DELTN,RE,THEDTE,MU,H,RS
  112 FORMAT(/////5X,I5/,7(E18.9/))
      WRITE(6,113) (SIGMQ(I),I=1,3)
  113 FORMAT(E18.9/)
```

```
      WRITE(6,114) (B(I,1),I =1,4)
  114 FORMAT(E18.9/)
      WRITE(6,115) (SIGMA(I),I =1,4)
  115 FORMAT(E18.9/)
      WRITE(6,116) ((QQ(I,J), J =1,6), I =1,6)
  116 FORMAT(6E18.9/)
      WRITE(6,117) ((PV(I,J),J =1,4),I=1,4)
  117 FORMAT(4E18.9/)
      WRITE(6,118) (E(I),I=1,6)
  118 FORMAT(E18.9/)
      RETURN
      END
```

```
      SUBROUTINE MATIN(A,R,N)
      DIMENSION  A(1), R(1), IX(150), B(150), G(150), DETR(150)
C
C  MATRIX INVERSION (A**-1 = R) -- BORDERING METHOD.
C  THE DETERMINANT RATIO DET(I+1) / DET(I) IS PRINTED. DET(I) IS THE
C  DETERMINANT OF THE FIRST I BY I SUB-MATRIX OF A.
C  THE INVERSION CHECK R*A IS CALCULATED AND PRINTED.
C  MATRICES A AND R MAY SHARE SAME CORE LOCATIONS.(R*A CHECK IS INVALID)
C  THE MAXIMUM SIZES ARE
C     N = 150
C
C     SUBROUTINE ARGUMENTS
C  A  = INPUT  MATRIX TO BE INVERTED. SIZE(N,N).
C  R  = OUTPUT RESULT MATRIX. SIZE(N,N).
C  N  = INPUT  SIZE OF MATRIX A AND R. (MAX = 150)
C
 1001 FORMAT (26H1N EXCEEDS INV1 ALLOWABLE./ 17H0PROGRAM STOPPED.)
 1002 FORMAT (1H1,10X,18HMATRIX IS SINGULAR/11X,17HPROGRAM HAS ENDED)
C
      KR = N
      IF (N .LE. 150) GO TO 150
      WRITE (6,1001)
      STOP
C
  150 DO 160 I=2,N
      IX(I) = I
  160 CONTINUE
C     INVERT FIRST NON-ZERO ELEMENT IN FIRST COLUMN.
      DO 190 I=1,N
      IF (A(I) .NE. 0.) GO TO 220
  190 CONTINUE
      WRITE (6,1002)
      STOP
C
C     START INVERSION WITH ROW I.
  220 DETR(1) = A(I)
      R(I) = 1. / A(I)
      IF (N .EQ. 1) GO TO 999
C
      IX(I) = 1
      IX(1) = 1
C     BORDERING LOOP.
      DO 630 L=2,N
          K = L
          L1 = L - 1
  250 S = 0.
                    MIXL = KR * (IX(L) - 1)
                    LL = IX(L) + MIXL
      DO 450 I=1,L1
                    MIXI = KR * (IX(I) - 1)
                    LI = IX(L) + MIXI
      B(I) = 0.
      G(I) = 0.
      DO 440 J=1,L1
                    MIXJ = KR * (IX(J) - 1)
                    IJ = IX(I) + MIXJ
                    JL = IX(J) + MIXL
      B(I) = B(I) - R(IJ)* A(JL)
                    JI = IX(J) + MIXI
                    LJ = IX(L) + MIXJ
```

```
      G(I) = G(I) - A(LJ)* R(JI)
  440 CONTINUE
      S = S + A(LI)* B(I)
  450 CONTINUE
      AL = A(LL)+ S
      IF (A(LL) .EQ. 0.) GO TO 480
      ALBAR = ABS (AL / A(LL))
      GO TO 490
  480 ALBAR = ABS (AL)
  490 IF (ALBAR .GE. .1E-6) GO TO 550
C
C     INTERCHANGE ROWS AND COLUMNS.
      K = K + 1
      IF (K .GT. N) GO TO 540
      IX L = IX(L)
      IX(L) = IX(K)
      IX(K) = IX L
      GO TO 250
  540 IF (ALBAR .GE. .1E-8) GO TO 550
  545 WRITE (6,1002)
      STOP
C
  550 R(LL)= 1. / AL
      DETR(L) = AL
      DO 570 I=1,L1
                        IL = IX(I) + MIXL
      LI = IX(L) + KR * (IX(I) - 1)
      R(IL)= B(I) * R(LL)
      R(LI)= G(I) * R(LL)
      DO 570 J=1,L1
      IJ = IX(I) + KR * (IX(J) - 1)
      R(IJ)= R(IJ)+ G(J) * R(IL)
  570 CONTINUE
  630 CONTINUE
C
C     COMPUTE INVERSION CHECK R*A.
      XOFF = 0.0
      DO 720 I=1,N
      DO 710 J=1,N
      X = 0.0
      KJA = KR * (J-1)
      DO 703 K=1,N
      IK = I + KR*(K-1)
      KJ = K + KJA
      X = X + R(IK) * A(KJ)
  703 CONTINUE
      IF (I .NE. J) GO TO 705
      G(I) = X
      GO TO 710
  705 IF (ABS(X) .LT. ABS(XOFF)) GO TO 710
      XOFF = X
      IOFF = I
      JOFF = J
  710 CONTINUE
  720 CONTINUE
C
C
  999 RETURN
      END
```

```
      FUNCTION RNUM(DUMY)
      DATA NX/0/
      IF(NX)3,2,3
    2 YY=5160736.
      ZZ=1492480.
      WW=3130862.
      SS=6538271.
      NX = 2
    3 WW= WW+WW
      YY= YY+YY
      ZZ= ZZ+ZZ
      Y1 =YY-9999997.
      Z1 =ZZ-9999971.
      W1 =WW-9699691.
      IF(Y1)20,20,10
   10 YY = Y1
   20 IF (Z1) 40,40,30
   30 ZZ=Z1
   40 IF(W1) 60,60,50
   50 WW =W1
   60 SS=WW+ZZ+YY+SS
      N=SS*.0000001
      Q=N*10000000
      RNUM=(SS-Q)*.0000001
      SS =SS-Q
      RETURN
      END
```

```
      SUBROUTINE EXTRA
C
C  THIS IS A DUMMY SUBROUTINE THAT IS USED FOR DEBUGGING PURPOSES, IT
C  MAY ALSO BE USED FOR ANY ADDITIONAL WRITE STATEMENTS
C
      RAY = 3
      FRED = 4
C
      RETURN
      END
```