APAP-ORMA

ORACLE-MAFIA
PROGRAM
VOLUME I

27 September 1967 05952-6178-T000

Prepared Under Contract NAS9-4810
Task MSC/TRW ASPO-19B

Prepared by

L. H. Smith, Programer
Subsystems Programing Department
Computation and Data Reduction Center

C. S. Powers, Head
Methods Development Section
Propulsion Analysis Department

Approved by

D. W. Routh, Manager
Subsystems Programing Department
Computation and Data Reduction Center

G. J. MacLeod, Manager
Propulsion Analysis Department
Power Systems Division

TRW SYSTEMS GROUP

TABLE OF CONTENTS

TABLE OF FIGURES

SECTION 1

1.0 PROBLEM DESCRIPTION

1.1 SUMMARY

This document describes the TRW computer program, ORACLE-MAFIA (also called MAFIA). This program has been written as a part of Task 19 of Contract NAS9-4810, for use in flight analysis of propulsion and propellant management systems on the Apollo spacecraft. The analysis technique employed in ORACLE-MAFIA is that of minimum variance estimation. Double precision computations are used where necessary to reduce the roundoff problem characteristic of minimum variance estimators. Both deterministic and state noise models are included.

The equations defining the engineering simulation are modularized. Hence, the program is not limited to analysis of propulsion systems, but may be readily applied to other types of systems. These systems may be described by either linear or nonlinear equations. An iterative scheme is incorporated to allow minimum variance estimation for nonlinear systems.

The computational algorithm does not provide for the case of measurements which have errors correlated in time (i.e., serial correlations). However, all other cross-correlations are properly treated. In addition, the state noise sigmas may be varied as piecewise linear functions of time (with a constant correlation coefficient matrix).

1.2 INTRODUCTION

The computer program, ORACLE-MAFIA, was written for use in flight analysis of propulsion and propellant management systems on the Apollo spacecraft.

ORACLE-MAFIA is the second of a pair of computer programs written for the Apollo flight analysis task, the first being ORACLE-B. The difference between the two programs lies in the computational algorithm employed. ORACLE-B is more oriented toward an in-flight analysis where accuracy may have to be sacrified for computational speed. ORACLE-MAFIA emphasizes computational stability by selected use of double precision computations. Lest the reader be concerned, the term MAFIA is derived from Mack Alford's FIltering Algorithm; Mack Alford having created the computational algorithm.

The accomplishment of the Apollo flight analysis task required a computer program that would determine - from flight test data - estimates of propulsion system performance that were better than those available prior to the flight test. Hence, the program of necessity must deal with test data. In addition, it must incorporate a simulation of the system to be analyzed. This simulation allows the program to compare an a-priori estimate of the system performance with the test data in order to refine the a-priori estimate. Since the test data and a-priori estimates both contain errors of a statistical type, a statistical technique must be employed.

The technique which has the desired characteristics is called minimum variance estimation. There is an extensive literature on minimum variance estimators (MVE). Appendix I-A of this section, as well as Sections 1.3 and 1.4, provides a discussion of the technique and a bibliography.

The simplest case of a MVE is the average of several measurements of a quantity to improve the estimate of that quantity. Least squares curve fits and weighted least squares curve fits are further examples of minimum variance estimators. These represent special cases in which the statistics describing the measurements are simple

and the system simulation is simple (for instance, described by a polynomial with unknown coefficients).

One feature of any MVE is that it must be possible to compute an estimate of each data point, $\hat{Y}$, as a function of estimated independent variables, $\hat{X}$. The estimate $\hat{Y}$ is compared with the measured value Y^* and $\hat{X}$ adjusted according to the minimum variance criteria. In the general case, $\hat{X}$ may be calculated from initial estimates, $\hat{X}_o$, by integrating differential equations. The equations allowing computation of $\hat{X}$ and $\hat{Y}$ are the system simulation. Also included in the simulation is the computation of the partial derivative matrices, $\partial \hat{Y}/\partial \hat{X}$ and $\partial \hat{X}/\partial \hat{X}_o$, which are required by the MVE algorithm. The system simulation capability included in ORACLE (both programs) is a general one with the user supplying the system description in his input, as described in Section 1.5. The simulation region of the program is called Nemesis.

Because of the importance of matrix inversion in a MVE, a special technique is used in ORACLE. Section 1.6 provides a description of this technique.

The flexibility available in ORACLE presents the user with an unusual problem. It is a very easy matter for the user to dimension a problem in violation of program restrictions. Over-dimensioning will cause errors, some of which may be difficult to detect or diagnose. Hence, Section 1.7 discusses this problem in some detail to inform the user on both the current restrictions and the relative ease or difficulty of modifying the restrictions.

1.3 MINIMUM VARIANCE ESTIMATORS

Appendix I-A to this section provides an extensive discussion of minimum variance estimators and their properties. The equations specifically implemented in MAFIA are given in Section 1.4. However, a few comments are in order about the type of minimum variance estimator implemented in MAFIA..

- MAFIA is a "total fit" estimator - as contrasted with a sequential estimator, such as a Kalman filter;
- MAFIA is a nonlinear MVE with an iterative scheme to correct for nonlinearities;
- MAFIA contains both a deterministic state model and a noise-in-the-state model;
- A-priori state estimates may be used, if desired;
- Non-diagonal covariance matrices may be used, if desired.

Some of the above items deserve further amplification.

Firstly, a nonlinear system is assumed. Hence, an iterative scheme is provided. At each iteration, the initial state estimate vector, $\hat{X}_o$, is corrected by a ΔX_o vector. Convergence occurs when the weighted norm of ΔX_o, i.e., $||\Delta X_o / W X_o||$, is less than an input number.

If the system to be analyzed is described by deterministic equations, Linear Model 0 (also called Error Model 0) is used. If a noise-in-the-state model is desired, two are available. Error Model 1 assumes that the noise is small enough so that the relationship,

$$\hat{Y} = f(\hat{X}) + \left(\frac{\partial f}{\partial X}\right)\Big|_{\hat{X}} \hat{\varepsilon} \qquad (1.3\text{-}1)$$

is a sufficiently close approximation to

$$\hat{Y} = f(\hat{X} + \hat{\varepsilon}). \qquad (1.3\text{-}2)$$

Error Model 2 allows for a larger noise by using directly the relationship

$$\hat{Y} = f(\hat{X} + \hat{\epsilon}). \tag{1.3-3}$$

In both cases, the physical error model is the same. It characterizes the state as having a deterministic portion and a non-deterministic portion, i.e.,

$$X_i^* \triangleq X_i + \epsilon_i \tag{1.3-4}$$

where X is the solution at t_i of:

$$\frac{dX}{dt} = f(X, t) \tag{1.3-5}$$

subject to

$$X(t_o) = X_o \tag{1.3-6}$$

and where

$$E(\epsilon_i) = 0 \tag{1.3-7}$$

$$E(\epsilon_i \epsilon_i^T) = R_i \tag{1.3-8}$$

$$E(\epsilon_i \epsilon_j^T) = 0 \qquad i \neq j \tag{1.3-9}$$

Specifically, this model states that the displacement of the noisy state X_i^* at t_i from the deterministic state X_i is in no way related to the displacement at previous time points. In addition, the state noise has no effect on the state derivatives. This error model is less appealing for many physical problems than the one associated with a Kalman filter, for example (see Appendix 1-A). However, it permits a computational algorithm in which round-off errors can be readily controlled. In addition, since the state noise at every time point is computed from a consistent set of initial state estimates, the validity of the assumptions regarding the error model are easily checked - which is difficult with a Kalman filter.

In order to get to each higher error model, MAFIA always insists on convergence of the previous one. This is done, because Error Model 2 is computationally slower than Error Model 1 and Error Model 1 is computationally slower than Error Model 0. It is felt that convergence of the lower error models should reduce the number of iterations in the higher error models.

Other Statistical Assumptions

As stated in the Summary, the program does not consider serial correlations in the measurement errors. However, cross-correlations may be considered. There are three covariance matrices used: J_o, R and W. The program treats the diagonal and non-diagonal cases differently in order to maximize precision and minimize computing time.

A covariance matrix, A, may always have the form: $A = GCG$ where G is a diagonal matrix and C is a symmetric matrix with 1.0's on the diagonal. Let G be a diagonal matrix:

$$G = \begin{pmatrix} g_1 & & & \\ & g_2 & & \\ & & \ddots & \\ & & & g_n \end{pmatrix} \qquad (1.3\text{-}10)$$

Then

$$C_{ij} = A_{ij}/g_i g_j \qquad (1.3\text{-}11)$$

where

$$g_i = \sqrt{A_{ii}} \qquad (1.3\text{-}12)$$

$$g_j = \sqrt{A_{jj}} \qquad (1.3\text{-}13)$$

In MAFIA, in the case of J_o, the vector GJ is input and if the matrix is non-diagonal, the upper triangle of CJ (not including the diagonal elements which are 1.0) is input.

$$J_o^{-1} = (GJ)^{-1}(CJ)^{-1}(GJ)^{-1} \qquad \text{(1.3-14)}$$

If J_o non-diagonal

$$\begin{matrix} \frac{1}{g_1^2} & & \\ & \ddots & \\ & & \frac{1}{g_n^2} \end{matrix} \qquad \text{(1.3-15)}$$

if J_o diagonal

A similar computation is used for R except that the elements of GR are obtained by linear interpolation in a table, RAB.

The W matrix is handled somewhat differently to assure precise computations. A positive definite symmetric matrix, A, can always be expressed in the following form: $A = u\Omega u^T$, where

$$\Omega = \begin{matrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \ddots \\ & & & \lambda_n \end{matrix} \qquad \text{(1.3-16)}$$

and u is a matrix of orthonormal eigenvectors; i.e., $u_i^T u_j = \delta_{ij}$ where u_i and u_j are the $i^{\underline{th}}$ and $j^{\underline{th}}$ columns of u and $\delta_{ij} = 1$ if $i = j$, $\delta_{ij} = 0$ if $i \neq j$; and where $\lambda_i > 0$.

Hence, one can define a matrix $A^{1/2}$ such that:

$$A^{1/2} \triangleq \Omega^{1/2} u^T \qquad \text{(1.3-17)}$$

where

$$\Omega^{1/2} = \begin{matrix} \sqrt{\lambda_1} & & \\ & \sqrt{\lambda_2} & \\ & & \ddots \\ & & & \sqrt{\lambda_n} \end{matrix} \qquad \text{(1.3-18)}$$

Clearly,

$$(A^{1/2})^T A^{1/2} = A \qquad (1.3\text{-}19)$$

In MAFIA, among the computations that need to be done precisely are sums of terms of matrix multiplications including:

$$(M_i'\phi_i)^T W_i^{-1}(M_i'\phi_i) \text{ and } (M_i'\phi_i)^T W_i^{-1}(Y_i^* - Y(\hat{X}_i))$$

where

M_i' is the measurement matrix $(\frac{\partial Y}{\partial X})_{\hat{X}}$ at t_i

ϕ_i is the state transition matrix $(\frac{\partial X_i}{\partial X_o})_{\hat{X}_o}$ at t_i

Y_i^* is the measured data vector at t_i

$\hat{Y}$ is the estimated data vector at t_i

$\hat{X}_i$ is the estimated state vector at t_i

Let

$$W_i^{-1} = (GW)^{-1}(CW1)^T(CW1)(GW)^{-1} \text{ if } W_i \text{ is non-diagonal.}$$

As asserted above, W_i^{-1} can always have this form. CW1 is found by finding the eigenvectors and eigenvalues of the correlation coefficient matrix of W. GW has the same relationship to W as GJ has to J_o.

$$(M'\phi)^T W^{-1}(M'\phi) = \phi^T M'^T (GW)^{-1}(CW1)^T(CW1)(GW)^{-1}M'\phi \qquad (1.3\text{-}20)$$

Let

$$M = (CW1)(GW)^{-1}M' \qquad (1.3\text{-}21)$$

The computation of M is performed in two parts. First, the matrix T is computed.

$$T = (GW)^{-1}M' \tag{1.3-22}$$

The matrix $(GW)^{-1}$ is diagonal; i.e.,

$$(GW)^{-1} = \begin{bmatrix} \frac{1}{\sigma_1} & & O \\ & \frac{1}{\sigma_2} & \\ & & \ddots \\ O & & \frac{1}{\sigma_n} \end{bmatrix} \tag{1.3-23}$$

Hence, the matrix T is readily computed from M'. The ij^{th} element of T, t_{ij}, is simply

$$t_{ij} = \frac{M'_{ij}}{\sigma_i}$$

If W is a non-diagonal matrix, then

$$M = (CW1)\,T \tag{1.3-24}$$

If W is a diagonal matrix, the matrix CW1 is an identity matrix. The program simply sets M = T.

By use of this definition of M, the MVE equations simplify:

$$(M'\emptyset)^T W^{-1}(M'\emptyset) = (M\emptyset)^T(M\emptyset) \tag{1.3-25}$$

Similarly,

$$(M'\emptyset)^T W^{-1}(Y^* - \hat{Y}) = (M\emptyset)^T(YR) \tag{1.3-26}$$

where

$$YR = (CW1)(GW)^{-1}(Y^* - \hat{Y}) \tag{1.3-27}$$

with the i^{th} element of

$$(GW)^{-1}(Y^* - \hat{Y}) = \frac{Y^*_i - \hat{Y}_i}{\sigma_i}.$$

Since the most common usage will have diagonal W matrices, this arrangement provides for computational efficiency and precision; in this case, at the sacrifice of the necessity of doing an external eigenvector-eigenvalue decomposition if one wants to use a non-diagonal W matrix.

It should be noted that all three correlation coefficient matrices are non-time varying.

Another special feature of MAFIA should be noted regarding the J_o matrix. It is often the case that the engineering user does not know how good his a-priori state estimates are. In order to be conservative, he may wish to assume that the sigmas corresponding to some or all of the a-priori state estimates are infinite. This corresponds to zeros on the diagonal of the $(GJ)^{-1}$ matrix. To accomplish this, the user inputs zeros in the locations in the GJ vector corresponding to the state estimate; e.g., if the fourth state variable is to have an infinite sigma, then GJ(4) = 0. This feature was provided so that the user would not have to decide what arbitrary large number best approximated the infinite sigma he really had. In the event that no a-priori estimates are known with any certainty, the user need not input the GJ vector at all, since it will then have all zero values.

Note: It is not possible to say an a-priori estimate is known perfectly, i.e., $\sigma = 0$. This is a consequence of the fact that, in MAFIA, the inverse of the a-priori state covariance matrix is used. The reciprocal of 0. yields either an unpleasantly large number or a zero depending on the computer used. Neither is really desired. Hence, if the user is really sure of an a-priori estimate, either the variable should be eliminated from the state vector or a small, but non-zero value, should be used in the GJ vector. This problem can arise because - as discussed in Section 1.5 - the only differential equations that the program will integrate are those for the state variables. Hence, any variable defined by a differential equation - rather than an algebraic equation - must be included in the state vector.

As has been indicated, MAFIA contains a general minimum variance estimation capability. In the general case, the system simulation may involve nonlinear differential and algebraic equations

in several variables, with perhaps some implicit equations. The equations defining the state of the system may not be completely deterministic - requiring a model for noise in the state. In addition, in the general case, all covariance matrices, i.e., the matrices describing the statistical properties of the measurements or the state noise, may have non-zero off-diagonal terms.

However, there are some restrictions. The most general minimum variance estimator which deals with multi-dimensional state and measurement vectors can consume an immense amount of computer core and calculation time. To circumvent this problem, assumptions are generally made regarding the statistical structure of the problem which it is hoped are valid. The most common of such assumptions has been made in ORACLE-MAFIA as well as in its predecessor programs; namely, that the measurement errors are not serially correlated in time. This assumption allows the processing of data at each time point without the necessity for storing data at other time points and drastically reduces the size of the problem and the computation time. It is also likely to be an erroneous assumption when data has been filtered prior to use in the program. However, studies have indicated that the usual effect on the solution of ignoring serial correlations is small so long as the filter span is short as compared with the data span used in the MVE. On the other hand, the statistics describing how well the solution is known will be optimistic. This is because the assumption of non-correlated errors implies the presence of more independent data points than really exist. When pre-filtered data is used, this assumption and its effects should be born in mind.

1.4 PROGRAM EQUATIONS

MAFIA processes multiple runs, cases, iterations and time points. Hence, the structure consists of loops within loops. Various operations are associated with each level of activity. This subsection and the next two describe some of the operations at the various levels with which the user is concerned.

A. New Run Loop

A new run in the MAFIA sense is required every time certain basic problem dimensions change. These are:

MD1	-	Total number of state variables
MD3	-	Total number of non-zero state variable derivatives (first partition of state vector)
MD4	-	Total number of measurement variables
N	-	Total number of state variables with noise

As described in Section 2, these variables are input in NAMELIST region SIZE. The data for every new run must start with input into SIZE.

For multiple case runs, the program will expect data for a new run if and only if LF(1) = 0 in the preceding case. If LF(1) = 1, the program expects new case data (see below), but not new run data.

As part of the new run loop, the following variables are cleared or initialized and any values different than the cleared or initialized ones must be loaded under NAMELIST region ATAD: MAXIT, EPS, EPS1, LAB1, LAB2, IBIAS, BIAS, ISCALE, SCALE, INDEX, LØCTRA, LF, WX0, WEPS, DELMAX, MXKNT1, MXKNT2, MXKNT3, MXKNT4, MXKNT.

B. New Case Loop

A new case in the MAFIA sense occurs every time the new case does not require re-dimensioning through input in NAMELIST region SIZE. Most multiple case runs are of this nature.

In any case, if a subsequent case (not run) is to be processed, the user must set the flag LF(1) = 1.

To reiterate, the first case in any MAFIA submission is a run - i.e., data must be provided in NAMELIST region SIZE. Subsequent user cases may be "runs" (new SIZE data) or "cases" (no new SIZE data). Any case to be followed by a new run must have LF(1) = 0. Any case to be followed by a new case must have LF(1) = 1. The last case may have either, but LF(1) = 0 is preferable.

A great variety of information may be input at the start of each new case. This data is input in NAMELIST region ATAD. In the first case, a large amount of information must be input, because the user supplies the entire engineering formulation as well as considerable control information.

In subsequent cases, only those input quantities which differ from the previous case must be input.

At the start of each case, MAFIA initializes a number of flags and arrays. Most of these operations don't concern the user. At this time, the inverse, $(J_o)^{-1}$, of the state a-priori covariance matrix, J_o, is calculated, since this is a constant for any case. In addition, the inverse, TC, of the state noise correlation coefficient matrix, CR, is also calculated.

In addition, the initial state estimate vector X0, is set equal to the input a-priori estimate vector, XA.

Also, the data editing parameter, RANGER, described in the Time Loop Section below, is set equal to the input quantity, RANGE1.

C. Iteration Loop

The initialization of each iteration involves the following calculations of interest to the user:

$$\hat{X} = X0 \qquad (1.4\text{-}1)$$

$$SUM1 = (J0)^{-1} \qquad (1.4\text{-}2)$$

$$SUM2 = 0 \qquad (1.4\text{-}3)$$

$$SUM3 = (J0)^{-1}(XA - X0) \qquad (1.4\text{-}4)$$

$$SUM4 = 0 \qquad (1.4\text{-}5)$$

$$SUM5 = (XA - X0)^{T}(J0)^{-1}(XA - X0) \qquad (1.4\text{-}6)$$

In addition, a number of flags are initialized and several arrays are cleared including the scaled measurement matrix, M, and the vectors $\hat{\varepsilon}$ and $\Delta\varepsilon$.

D. Time Loop

Most of MAFIA's calculations are performed in the Time Loop. Special calculations are performed at TIME = 0, and TIME = BURN, in addition to the regular calculations. These special computations will be indicated.

Time Base

MAFIA has two time bases, program or biased time, called TIME, and true or Range time, called TTIME. TTIME represents the time base on the input data tape, whereas TIME is the variable used for computed print times and End time.

There is a constant bias between TIME and TTIME. This bias is established by the first time point read on the data tape. The variable TIME always starts at 0, and ends at the value input for the parameter BURN. TTIME, on the other hand, is initialized at the first time value read off the data tape. At all subsequent time points, both TIME and TTIME are incremented identically.

Data Read

The input data tape having been rewound, the first N1 records are skipped (N1 being the number loaded in LF(5)). Then, two data records are read at TIME = 0. The first record provides the first value of TTIME, the true time, and the data, YS, associated with TTIME. The second record provides the next value of true time - called NUTIME - and the data, YT, associated with NUTIME.

The size of the time step to the next time point, DELTA, is calculated:

$$DEL = NUTIME - TTIME \qquad (1.4\text{-}7)$$

$$DELTA = Min(DEL, DELMAX) \qquad (1.4\text{-}8)$$

i.e., DELTA is either DEL or DELMAX, whichever is smaller. Thus, it may be seen that the MAFIA time base is advanced to the time of each set of data unless the interval is greater than the input quantity, DELMAX.

MAFIA always has two sets of data available. This is necessary because not only does it need data at the current time point, but it must also know at what time the next set of data is available in order to step to that point. It does this by using the quantity DEL which always contains the time increment between the current time and the next data point.

At time points other than the first, a data record may or may not be read. If there is data available at the time point (data read at a preceding time point into the vector YT), the program moves the data from YT into YS. It then reads a new set of data into YT and the corresponding time value into NUTIME. It then executes Equations 1.4-7 and 1.4-8 to establish the increment to the next time point.

If data is not available - a condition which the program recognizes because the value of DEL calculated at the previous time point is greater than DELMAX - the program reduces DEL by DELMAX, so that it correctly reflects the time to the next data point. But a new data point is not read, since the next one to be used is already available. Equation 1.4-8 is then executed to compute a new value of DELTA.

If a set of data is moved from YT to YS, an additional action is taken. In order to initialize the computations in the nonlinear equation solver, data values may be useful. (The nonlinear equation solver is part of the engineering region, Nemesis, described in the next section.) In addition, it is sometimes desirable to be able to use measured data for other Nemesis calculations. Therefore, if a set of data YS is available, it is loaded into the vector, $\hat{Y}$. Prior to doing this, an editing check is made of each element of YS; i.e., if $|YS_i - \hat{Y}_i| \leq$ RANGER $* \sigma_i$, then $\hat{Y}_i = YS_i$, where RANGER is the editing parameter (also used in the MVE equations). On the first iteration, RANGER equals the input quantity, RANGE1. On subsequent iterations, RANGER equals the input quantity, RANGE2. The values $\hat{Y}$ used for this check are the values calculated in NEMESIS at the preceding time point. At the first time point, the values of $\hat{Y}$ are those input into the YY vector.

Engineering Calculations

After the tape read, MAFIA executes a call to Nemesis. As is described in Section 1.5, Nemesis calculates

$\hat{Y}$	-	The estimates of the measurements
M'	-	The measurement matrix, $\partial Y / \partial X$
dX/dt	-	The state variable derivatives
$\dot{U}$	-	The derivatives of the state transition matrix, $\frac{\partial}{\partial X}\left(\frac{dX}{dt}\right)$

If linear Model II is being executed, the program computes

$$\hat{X}^* = \hat{X} + \hat{\epsilon} \tag{1.4-9}$$

and then calls Nemesis a second time to compute $\hat{Y}$ as a function of $\hat{X}^*$. This second call to Nemesis is identical to the first call except that the dX/dt and $\dot{U}$ values computed in the first call are not replaced by the values computed in the second call.

State Transition and Updating Matrices

Next, the state transition matrix, $\emptyset$, is computed:

If TIME = 0:

$$\emptyset_{o/o} = I \tag{1.4-10}$$

where I is a MD1 x MD1 identity matrix.

If TIME $\neq$ 0:

$$U_{i/i-1} = I + \sum_{j=1}^{3} \frac{1}{j!} (\Delta t_{i/i-1})^j \left(\frac{\dot{U}_i + \dot{U}_{i-1}}{2}\right)^j \tag{1.4-11}$$

and

$$\emptyset_{i/o} = U_{i/i-1} * \emptyset_{i-1/o} \tag{1.4-12}$$

where

$$\Delta t_{i/i-1} = t_i - t_{i-1}.$$

The matrix $U_{i/i-1}$ is considered by the program to be partitioned as follows:

$$U_{i/i-1} = \begin{pmatrix} \text{I} & \text{II} \\ \text{III} & \text{IV} \end{pmatrix}$$

Partition I is a MD3 x MD3 general matrix.

Partition II is a MD3 x (MD1 - MD3) general matrix.
(Set LF(6) = 1.)

Partition III is a (MD1 - MD3) x MD3 zero matrix.
(Set LF(7) = 0.)

Partition IV is a (MD1 - MD3) x (MD1 - MD3) identity matrix.
(Set LF(8) = 0.)

The reason for this is because of the meaning of the updating matrix, $U_{i/i-1}$, and the state transition matrix, $\phi_{i/o}$.

$$U_{i/i-1} \triangleq \frac{\partial X_i}{\partial X_{i-1}} \qquad (1.4\text{-}13)$$

and

$$\phi_{i/o} \triangleq \frac{\partial X_i}{\partial X_o} \qquad (1.4\text{-}14)$$

where the subscripts refer to the time points t_i, t_{i-1}, and t_o.

The nomenclature for the matrix of partials of the p-dimensional vector W with respect to the r-dimensional vector Z defines the matrix, A, as follows:

$$A = \frac{\partial Z}{\partial W} \triangleq \begin{matrix} \frac{\partial Z_1}{\partial W_1} & \frac{\partial Z_1}{\partial W_2} & \frac{\partial Z_1}{\partial W_3} & \cdots & \frac{\partial Z_1}{\partial W_p} \\ \frac{\partial Z_2}{\partial W_1} & \frac{\partial Z_2}{\partial W_2} & \frac{\partial Z_2}{\partial W_3} & \cdots & \frac{\partial Z_2}{\partial W_p} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \frac{\partial Z_r}{\partial W_1} & \frac{\partial Z_r}{\partial W_2} & \frac{\partial Z_r}{\partial W_3} & \cdots & \frac{\partial Z_r}{\partial W_p} \end{matrix} \qquad (1.4\text{-}15)$$

where A is an r x p matrix.

MAFIA expects the state vector to contain variables with zero time derivatives as well as variables with non-zero time derivatives. The first MD3 variables in the state vector are the only ones the program expects to have non-zero time derivatives. The remaining MD1-MD3 variables in the state vector are assumed by the program to be constants. The partial derivatives of a constant state variable at t_i with respect to the same state variable at t_j is equal to 1. The partial derivative of a constant state variable at t_i with respect to any other state variable at t_j is zero. Hence, the third and fourth partitions of U and $\emptyset$ are a zero matrix and an identity matrix, respectively. These two partitions contain the partials of the constant part of the state vector t_i with respect to the state vector at t_{i-1} in the case of $U_{i/i-1}$ or t_o in the case of $\emptyset_{i/o}$.

The first and second partitions contain the partials of the non-constant portion of the state vector at t_i with respect to the state variables at t_{i-1}, or t_o. The U matrix is computed from its derivative, $\dot{U}$, by the series expansion given in Equation 1.4-11.

If the state variables, X, are defined by the vector differential equation:

$$\frac{dX}{dt} = f(X, t) \qquad (1.4\text{-}16)$$

then the state transition matrix, $\phi_{i/o}$, and the updating matrix, $U_{i/i-1}$, satisfy the matrix differential equations:

$$\frac{d}{dt}\phi = \frac{\partial f}{\partial X}\phi \qquad (1.4\text{-}17)$$

$$\phi(t_o) = I \qquad (1.4\text{-}18)$$

$$\frac{d}{dt}U = \frac{\partial f}{\partial X}U \qquad (1.4\text{-}19)$$

$$U(t_{i-1}) = I \qquad (1.4\text{-}20)$$

The matrix $\frac{\partial f}{\partial X}$ is called $\dot{U}$ in this writeup.

A matrix equation of the form

$$\frac{\partial W}{\partial t} = AW \qquad (1.4\text{-}21)$$

$$W(t_o) = W_o \qquad (1.4\text{-}22)$$

where A is a matrix of constant coefficients has the solution:

$$W = W_o e^{A(t-t_o)} \qquad (1.4\text{-}23)$$

The matrix $e^{A(t-t_o)}$ may be expressed in an infinite series:

$$e^{A(t-t_o)} = I + \sum_{j=1}^{\infty} \frac{1}{j!} A^j (t-t_o)^j \qquad (1.4\text{-}24)$$

The assumption is made in MAFIA that the partial matrix $\frac{\partial f}{\partial X}$ is sufficiently constant over the interval t_{i-1} to t_i so that the matrix A may be approximated by the expression:

$$A \simeq 1/2 \left\{ \left(\frac{\partial f}{\partial X}\right)_{t_{i-1}} + \left(\frac{\partial f}{\partial X}\right)_{t_i} \right\} \qquad (1.4\text{-}25)$$

and that the matrix $A(t_i - t_{i-1})$ has elements sufficiently small compared to unity that the series expansion will converge using only the first

four terms. Equation 1.4-11 results from these assumptions. Equation 1.4-12 is a simple application of the chain rule of partial differentiation in conjuction with the definitions of U and $\emptyset$ given in Equations 1.4-13 and 1.4-14.

Data Editing

For each data value, Y*, the following check is made:

$$|YRP_i| > RANGER * \sigma_i \qquad (1.4\text{-}26)$$

$$i = 1, 2, \ldots, MD4$$

where

$$YRP_i = Y_i^* - \hat{Y}_i \qquad (1.4\text{-}27)$$

$\hat{Y}_i$ is the estimated value of Y_i calculated in Nemesis and σ_i is the input sigma for the i^{th} measurement.

RANGER is the editing parameter and is iteration dependent as mentioned above. It is equal to the input quantity, RANGE1, on the first iteration, and to the input quantity, RANGE2, on subsequent iterations. This allows a fairly loose editing criteria on the first iteration when the initial state estimates may be poorly known, and, hence, the computed estimates, $\hat{Y}$, may be relatively poor. On subsequent iterations, when the estimates, $\hat{Y}$, are improved, a tighter editing criteria may be used.

If $|YRP_i| > RANGER * \sigma_i$, then

$$YRP_i = 0 \qquad (1.4\text{-}28)$$

and

$$M'_{ij} = 0 \quad \text{for } j = 1, 2, \ldots, MD1 \qquad (1.4\text{-}29)$$

where

$$YRP_i = Y_i^* - \hat{Y}_i \qquad (1.4\text{-}30)$$

and

$$M'_{ij} = \frac{\partial Y_i}{\partial X_j} \qquad (1.4\text{-}31)$$

In other words, the data residual and the row in the measurement matrix corresponding to the i^{th} measurement are set to zero. Both steps are necessary if a non-diagonal measurement covariance matrix is used. In addition, a message is printed giving the time and the number of the edited measurement.

If all data is edited out, a message to that effect is given and all the MVE equations are skipped.

MVE Equations - Calculated only if data is available.

If W is diagonal (LF(12) = 0):

$$YR_i = YRP_i/\sigma_i \qquad (1.4\text{-}32)$$

$$i = 1, 2, \ldots, MD4$$

$$M_{ij} = M'_{ij}/\sigma_i \qquad (1.4\text{-}33)$$

$$i = 1, 2, \ldots, MD4$$
$$j = 1, 2, \ldots, MD1$$

If W is non-diagonal (LF(12 $\neq$ 0):

$$T_i = YRP_i/\sigma_i \qquad (1.4\text{-}34)$$

$$i = 1, 2, \ldots, MD4$$

$$YR = (CW1) * T \qquad (1.4\text{-}35)$$

$$T1_{ij} = M'_{ij}/\sigma_i \qquad (1.4\text{-}36)$$

$$i = 1, 2, \ldots, MD4$$
$$j = 1, 2, \ldots, MD1$$

$$M = (CW1) * T1 \qquad (1.4\text{-}37)$$

This difference in computation of the YR vector and the M matrix between the diagonal and non-diagonal cases is discussed in Section 1.3.

$$MPHI = M*\emptyset \qquad (1.4\text{-}38)$$

Noise in the State Computations - Only for Error Models 1 and 2.

As discussed in Section 1.3, the state noise covariance matrix, R, has the form:

$$R = (GR)(CR)(GR) \qquad (1.4\text{-}39)$$

where

$$GR = \begin{pmatrix} \sigma_{r1} & & & \\ & \sigma_{r2} & & \\ & & \ddots & \\ & & & \sigma_{rN} \end{pmatrix} \qquad (1.4\text{-}40)$$

and where CR is either an N*N identity matrix or is an input correlation coefficient matrix with values of 1.0 on the diagonal.

The values of σ_{rj} are computed from the input vector RAB whose format is described in Section 2. For each state variable for which there is noise, a set of values are entered into the RAB vector. These may yield either of two computations of σ_{rj}:

1. $$\sigma_{rj} = A_o + B_o * TIME \qquad (1.4\text{-}41A)$$

   where A_o and B_o are input in RAB

2. $$\sigma_{rj} = A_i + \frac{(A_{i+1} - A_i)}{(B_{i+1} - B_i)} * (TIME - B_i) \qquad (1.4\text{-}41B)$$

   where $B_i < TIME \leq B_{i+1}$

   and where A_i, A_{i+1}, B_i, B_{i+1} are input in RAB.

If σ_{rj} is to be constant, then B_o is input as zero and Equation 1.4-41A is used.

As mentioned, if it is not an identity matrix, the matrix TC is calculated as part of the New Case Loop:

$$TC = (CR)^{-1} \tag{1.4-42}$$

If TC is not an Identity Matrix (LF(11) ≠ 0):

$$RI_{ij} = TC_{ij} / \sigma_{ri} * \sigma_{rj} \tag{1.4-43}$$

$i = 1, 2, \ldots, N$
$j = 1, 2, \ldots, N$

If TC is an Identiy Matrix:

$$RI_{ii} = 1/\sigma^2_{ri} \tag{1.4-44}$$

$i = 1, 2, \ldots, N$

$$RI_{ij} = 0 \tag{1.4-45}$$

$i \neq j$

where RI is the inverse of the state noise covariance matrix, R. In this discussion, RI is also called R^{-1}.

Next, the matrix MPSI is calculated. The matrix MPSI is the scaled measurement matrix for the state variables which have noise. It has MD4 rows, but only N columns. In general, not all state variables will have noise, i.e., N < MD1, and, hence, the MPSI matrix will be smaller than the M matrix. Each column in the M matrix corresponds to a state variable. An input vector, PSI, contains the locations of each state variable having noise. For instance, if X_2, X_4, and X_5 have noise, then PSI(1) = 2, PSI(2) = 4 and PSI(3) = 5. The matrix MPSI is made up of the columns of M whose indices are stored in the PSI vector. In the example given:

$$\text{MPSI} = M\psi = \begin{matrix} M_{12} & M_{14} & M_{15} \\ M_{22} & M_{24} & M_{25} \\ \vdots & \vdots & \vdots \\ M_{MD4,2} & M_{MD4,4} & M_{MD4,5} \end{matrix} \qquad (1.4\text{-}46)$$

Next, d^{-1}, K and T are calculated.

$$d^{-1} = \left\{ R^{-1} + (M\psi)^T (M\psi) \right\}^{-1} \qquad (1.4\text{-}47)$$

$$K = (M\emptyset)^T (M\psi) \qquad (1.4\text{-}48)$$

$$T = d^{-1} \left\{ (M\psi)^T * (YR) - R^{-1} \hat{\varepsilon} \right\} \qquad (1.4\text{-}49)$$

and d^{-1}, K and T are written on tape.

Next, SUM2 and SUM4 are updated.

$$SUM2_i = SUM2_{i-1} + K_i d_i^{-1} K_i^T \qquad (1.4\text{-}50)$$

and

$$SUM4_i = SUM4_{i-1} + K_i d_i^{-1} \left\{ (M\psi)_i^T YR_i - R_i^{-1} \hat{\varepsilon}_i \right\} \qquad (1.4\text{-}51)$$

where the subscripts i and i-1 refer to the current and previous time points, respectively.

In Linear Model 1, $\hat{\varepsilon}$ is zero. Hence, the $R^{-1}\hat{\varepsilon}$ term does not appear in the program for Equations 1.4-49 and 1.4-51 unless Linear Model 2 is being executed.

The remainder of the time loop calculations are performed for all three error models.

$$SUM5_i = SUM5_{i-1} + (YR)_i^T * (YR)_i \qquad (1.4\text{-}52)$$

$$SUM1_i = SUM1_{i-1} + (M\emptyset)_i^T * (M\emptyset)_i \qquad (1.4\text{-}53)$$

$$SUM3_i = SUM3_{i-1} + (M\emptyset)_i^T * (YR)_i \qquad (1.4\text{-}54)$$

where the subscripts i and i-1 refer to the current and previous time points, respectively.

Computation of Next TIME and Check for Print Flag

MAFIA checks if the final time has been reached: If the check:

$$| TIME - BURN | \leq 1. \times 10^{-4} \qquad (1.4\text{-}55)$$

is satisfied, the program skips to the end of Time Loop calculations.

If this check fails, then both time bases are incremented to the next time value:

$$TIME = TIME + DELTA \qquad (1.4\text{-}56)$$

$$TTIME = TTIME + DELTA \qquad (1.4\text{-}57)$$

Also, checks are made to compute new print times: If

$$| TIME - PRTLIM | \leq 1. \times 10^{-4}$$

then the print flag is set (to govern the print on the next execution of the Time Loop) and the next print point computed:

$$PRTLIM = PRTLIM + DELPRT \qquad (1.4\text{-}58)$$

$$PRTLIM = Min (BURN, PRTLIM) \qquad (1.4\text{-}59)$$

Equation 1.4-59 provides for print occurring at TIME = BURN, even if it is not a normal print time.

Integration of State Equations

The program integrates the state vector X forward from t_i to t_{i+1}.

$$\hat{X}_{i+1} = \hat{X}_i + \Delta t_{i+1/i} * \dot{X}_i + \frac{(\Delta t_{i+1/i})^2}{2} \frac{(\dot{X}_i - \dot{X}_{i-1})}{\Delta t_{i/i-1}} \qquad (1.4\text{-}60)$$

where

$$\Delta t_{i+1/i} = t_{i+1} - t_i = \text{DELTA}$$

$$\Delta t_{i/i-1} = t_i - t_{i-1} = \text{Previous value of DELTA}$$

This constitutes the end of the Time Loop.

E. End of Iteration Calculations

The computation of the matrix, Bll, is performed with considerable care in MAFIA to preserve the necessary precision. The following calculations are performed:

$$F = \text{SUM1} - \text{SUM2} \qquad (1.4\text{-}61)$$

$$g_i = (F_{ii})^{1/2} \qquad (1.4\text{-}62)$$

$$CF_{ij} = F_{ij}/g_i * g_j \qquad (1.4\text{-}63)$$

$$A = (CF)^{-1} \qquad (1.4\text{-}64)$$

$$Bll_{ij} = A_{ij}/g_i * g_j \qquad (1.4\text{-}65)$$

The computation of Bll as described above is made by decomposing the matrix F into the form:

$$F = GF * CF * GF \qquad (1.4\text{-}66)$$

where

$$GF = \begin{pmatrix} g_1 & & \\ & g_2 & O \\ & O & \ddots \\ & & & g_{MD1} \end{pmatrix} \qquad (1.4\text{-}67)$$

and CF is the correlation coefficient matrix of F.

Hence,

$$B11 = F^{-1} = (GF)^{-1} * (CF)^{-1} * (GF)^{-1} \qquad (1.4\text{-}68)$$

It should be noted that B11 is the covariance matrix of ΔX_o and, hence, of X_o

The matrices SUM1 and SUM2 are both double precision matrices. Their values are computed in the Time Loop (Equations 1.4-53 and 1.4-50) with the initialization given in Equations 1.4-2 and 1.4-3. The equations are:

$$SUM1 = \sum_{i=0}^{BURN} (M\emptyset)_i^T * (M\emptyset)_i + (J_o)^{-1} \qquad (1.4\text{-}69)$$

$$SUM2 = \sum_{i=0}^{BURN} K_i \, d_i^{-1} \, K_i^T \qquad (1.4\text{-}70)$$

If Linear Model 0 is being executed, SUM2 is a zero matrix.

A check on the matrix inversion is also made. Let

$$B = A*(CF) - I \qquad (1.4\text{-}71)$$

where $A = (CF)^{-1}$ and I is an identity matrix. Then, the column norm,

$$\left\{ \sum_{i=1}^{MD1} b_{ij}^2 \right\}^{1/2},$$