

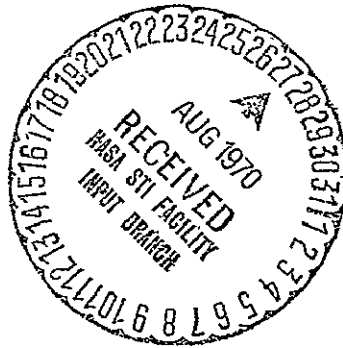
YES



NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

INTERNAL NOTE MSC-ED-R-69-82

ESTIMATING NUMERICAL INTEGRATION ERRORS



MANNED SPACECRAFT CENTER

HOUSTON, TEXAS

April 1969

FACILITY FORM 602

N70-35712
(ACCESSION NUMBER)

(THRU)

59

1

(PAGES)

(CODE)

TMX-64465

30

(NASA CR OR TMX OR AD NUMBER)

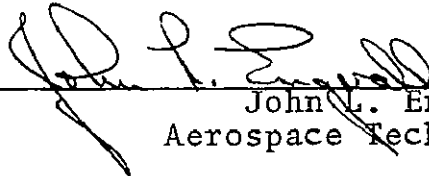
(CATEGORY)

Reproduced by
**NATIONAL TECHNICAL
INFORMATION SERVICE**
Springfield, Va. 22151


INTERNAL NOTE MSC-ED-R-69-82

ESTIMATING NUMERICAL INTEGRATION ERRORS

PREPARED BY




John L. Engvall
Aerospace Technologist




Melvin J. Arldt
Associate Scientist
Lockheed Electronics Company

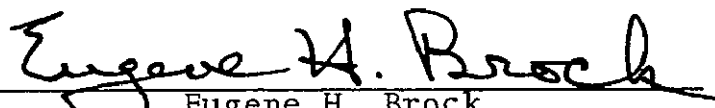
APPROVED BY



B. L. Carroll
Supervisor, Theory and Analysis Group
Lockheed Electronics Company



J. M. Lewallen
Chief, Theory and Analysis Office



Eugene H. Brock
Chief, Computation and Analysis Division

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
MANNED SPACECRAFT CENTER
HOUSTON, TEXAS

April 1969

TABLE OF CONTENTS

<u>Section</u>	<u>Page</u>
ABSTRACT	1
INTRODUCTION	1
PROBLEM DEFINITION	4
DERIVATION OF ALGORITHM I.	9
DERIVATION OF ALGORITHM II	12
NUMERICAL EXAMPLES	16
Brachistochrone Problem	19
Flat Earth Problem.	22
Earth-Jupiter Problem	25
COMPARISONS OF STANDARD INTEGRATION ALGORITHMS	29
NUMERICAL RESULTS.	33
Evaluation of Algorithm II.	35
Evaluation of Algoritihm I	37
CONCLUDING REMARKS	39
REFERENCES	41
TABLES	43

ACKNOWLEDGMENTS

The authors would like to express their appreciation to Dr. J. M. Lewallen for the many hours of discussion and explanation involving general optimization theory, specific numerical examples, and previous numerical results.

ABSTRACT

An algorithm is presented for use in estimating accumulated numerical integration errors for integrating systems of ordinary first-order differential equations. Another algorithm is presented for estimating errors in problems (such as trajectory optimization problems) having an additional set of necessary conditions. These conditions are often analogous to the momentum and energy constants of conventional physical problems.

Also, accuracies of the standard fourth-order Runge-Kutta and Adams-Moulton and fifth-order Runge-Kutta formulas are compared. The brachistochrone and "flat Earth" trajectory problems, both having closed form solutions, are used for numerical examples. Results for a low thrust, Earth-Jupiter transfer problem are presented as a third example.

INTRODUCTION*

The formulation of good error estimates or close-error bounds for numerically integrating first-order systems of ordinary differential equations is generally difficult. However, good estimates for some specific problems can be obtained. Ordinarily, trajectory optimization problems do not tend to be unstable, nor do they fall into the stiff spring class. Hence, two of the troublesome areas of numerical integration are avoided. The fact that linear estimates are exceptionally accurate for many nonlinear trajectory problems is also very encouraging. Still

*Portions of this paper have been reproduced, with permission of the authors, from MSC-ED-R-68-74.

another important factor is the existence of a constant of motion, a condition that exists for many optimization problems in the form of a generalized Hamiltonian. Not only can this function be monitored, but an analysis of the sensitivity of the function can be used to yield error estimates.

Most methods for solving trajectory optimization problems imply some iterative process involving many numerical integrations of an initial value problem. In theory, this numerical integration is assumed to be exact so that, at least in the terminal phases of the iteration process, accurate numerical integration is required. Computer time requirements must also be considered, depending on the number of iterations required and the complexity of the problem in question. Both of these criteria, computer expense and the need for accuracy, have stimulated several good reports concerning numerical integration errors for the two-body problem.

During the summer of 1966, Lewallen and Gerber⁽¹⁾ investigated error propagation for numerically integrating many different near Earth orbits. The effects of varying orbit shape, orbit size, coordinate systems, and integration step size were studied. All numerical tests were carried out using fixed step Adams-Moulton integration in either rectangular or spherical coordinates. Two significant results, the superiority of the spherical coordinate system and a similarity of the norms of the error in position and the error in velocity, were reported. Schwausch⁽²⁾ performed similar research by using additional coordinate systems and obtained best results with a form of elliptic coordinates.

Rainbolt⁽³⁾ presents an error analysis for the two-body problem, finding circular cylindrical and parabolic cylindrical to be more accurate than rectangular, spherical, parabaloidal, and elliptic cylindrical. Rainbolt also analyzed the effect of coordinate systems on convergence envelopes for indirect optimization methods of solving a two-dimensional, low-thrust, Earth-escape spiral. The investigation revealed that larger convergence envelopes are obtained using the circular cylindrical (polar in two dimensions) coordinate system; monitoring of the Hamiltonian also indicated that better accuracy is obtained when integrating the initial value problem in this coordinate system. Perhaps the system yielding the most accurate integration also yields the best convergence properties for solving the optimization problem.

All three references thus far base their conclusions on using the popular fourth-order Adams-Moulton integration. The authors propose in this study to fix the coordinate system and to vary the type of numerical integration. Tables are included to show the numerical errors propagated during the terminal integration of three different optimization problems. Both fixed-step and variable-step integration will be considered.

In addition to comparing different types of integration, this paper will compare methods for obtaining long-term accumulated error estimates. Although good error bounds are more desirable, there will be no attempt to obtain bounds; past experience indicates very little success in that direction. The authors prefer to hope for excellent estimates and to accept some uncertainty. Obtaining these

estimates will require a combination of mathematics, numerical results, additional labor and computer time, and common sense. Very logically, most methods to date depend heavily on the latter, and, fortunately, automated techniques can be devised to eliminate some of the additional costs. These techniques will be demonstrated with three numerical examples, starting with the simple classical brachistochrone problem, progressing to a modern day "flat Earth" trajectory problem for launching a satellite and ending with an Earth-Jupiter transfer orbit problem.

Much of the previous work in trajectory optimization has been focused on coordinate systems, transformation of variables such as regularization, and the effect of different numerical techniques on convergence envelopes. Emphasis has been placed on fixed-step size integration because of the natural application to the structure of many numerical algorithms. Trial and error procedures have played a dominant role in choosing this fixed-step size, and in many instances little effort is made to automate the process. A technique for estimating errors can be used to automate this process of choosing step size; hence, a possible saving of man-hours and computer time may result. However, the principle emphasis for obtaining error estimates is placed on having additional information about the numerical accuracy of the final solution.

PROBLEM DEFINITION

Although the numerical examples are restricted to Euler Lagrange equations, the basic theory can be stated in

general. The dominant theme is solving initial value problems numerically. That is, given

$$y' = f(y,t) , \quad y(t_0) = y_0 , \quad (1)$$

one must obtain an approximation \hat{y} to the solution y over a domain $[t_0, t_f]$ or at least for a discrete set of points in this domain.

If $z = y - \hat{y}$ denotes the error in \hat{y} , then an estimate to z can be obtained by integrating

$$\frac{d}{dt} (y - \hat{y}) = z' = Az + b , \quad z(t_0) = 0 \quad (2)$$

where $A = (a_{ij}) = \partial f_i / \partial y_j$ and b is a forcing function representing the error in \hat{y}' due to local truncation and rounding errors. The accuracy of such an estimate depends on the validity of the linearity assumption in (2), the type of forcing function b , the accuracy of the estimates for local errors, and the type(s) of integration used to integrate (1) and (2). An algorithm with a particular choice for these variables will be defined in the next section. This process, using the linear estimate for z' , will form one of the basic types of estimates considered.

The second basic type of error estimate can be applied only to those problems having an additional system of equations

$$g(y,t) = 0 \quad (3)$$

representing one or more necessary conditions that y satisfy the initial value problem. Although "t" can appear explicitly in (3) (and the algorithm can be applied to these problems), emphasis will be placed on those problems where g is explicitly a function of y alone. These problems have the unpleasant characteristic that error estimates are based on a system of equations having many solutions. For example, if $\hat{y}(t_f)$ is the numerical solution at $t = t_f$, then one would need to find a vector Δ such that

$$g(\hat{y}(t_f) + \Delta, t_f) = 0 \quad (4)$$

Assuming that t does not appear explicitly (and that (1) has a nontrivial solution), then there will be many vectors Δ satisfying (4), namely those such that

$$\Delta = y(t) - \hat{y}(t_f) \quad (5)$$

for some $t \in [t_0, t_f]$. For most problems, the range of y is a continuum so that there will be an uncountable number of solutions for Δ .

On the surface, the second basic type of estimate appears to have no merit. For many problems the residuals of $g(y,t)$ are monitored, and, if several types of numerical integrations are used, the one yielding the smallest residuals is assumed to be the most accurate. (Many successful algorithms are based on assuming necessary conditions are also sufficient.) This presentation is neither a criticism of such techniques nor an attempt to discourage their use; it is an attempt to go further in

obtaining independent error estimates. However, in this approach, the method itself will often serve as a reminder that nonsufficient conditions have been used.

Since there are many solutions to (4), a logical criterion must be designed for accepting particular values of Δ as error estimates. One such method is to accept a solution to (4), say Δ_0 , if the Euclidean norm of Δ_0 is a minimum for all values $||\Delta||$ such that Δ satisfies (4). The continuity of g in y would be sufficient to guarantee the existence of such a Δ_0 . Delaying considerations of existence, implementation of numerical techniques, and convergence problems, consider for a moment the choice of Δ_0 as opposed to some other estimate. Both of the conditions—(a) $||\Delta_0||$ is a minimum and (b) equation (4) is only a necessary condition—combine to prove that

$$||E(\hat{y})|| \geq ||\Delta_0|| ,$$

where $E(\hat{y})$ denotes the error in \hat{y} ; i.e., Δ_0 is a lower bound (in norm) to the error and could at least be used to reject solutions. As a second criterion for estimates, all components of Δ can be required to be zero except for one component, and accept an estimate, say Δ_1 , if the Euclidean norm of $g(\hat{y} + \Delta)$ is a minimum. In most cases this will result in a unique solution Δ_1 since this forms a least squares problem rather than an overdetermined system. Physically, this is equivalent to assuming all error is in one component, such as the velocity along the x component, so that in a sense this seems to be an upper bound for the error. However, one is again reminded that (4) is only a

necessary condition; hence, this estimate can be viewed with mixed emotions. Since the residuals may not be zero for this estimate, indicating other variables having dominant errors, a more detailed study could be carried out allowing various combinations of nonzero elements in the Δ vector. As a third and last criterion, a solution, say Δ_3 , will be accepted if $\|w \Delta_3\| \leq \|w \Delta\|$ for any Δ satisfying (4), where w is a diagonal weighting matrix. In practice, this is often more realistic than minimizing $\|\Delta\|$. For example, suppose y is the state vector consisting of three position coordinates, $a_1 = (y_1, y_2, y_3)$, and three velocity coordinates, $a_2 = (\dot{y}_1, \dot{y}_2, \dot{y}_3) = (y_4, y_5, y_6)$. For this case, $\|y\|$ has very little physical significance; hence, $\|\Delta\|$ for the corresponding error estimate has little significance. Many choices of w can lend intuitive interpretation to Δ_3 . If the approximate norm ($\|a_1\|$) of the terminal position is w_1 , and the approximate norm ($\|a_2\|$) of the terminal velocity is w_2 , an appropriate choice for weighting would be to minimize

$$\left(\left(\frac{1}{w_1} \right)^2 (\Delta_1^2 + \Delta_2^2 + \Delta_3^2) + \left(\frac{1}{w_2} \right)^2 (\Delta_4^2 + \Delta_5^2 + \Delta_6^2) \right)^{1/2}.$$

Intuitively, this weighting forces errors affecting the n th significant place of the velocity norm to be considered as important as errors in the n th significant place of the position norm. Similarly, the vector y may contain mass, moments, Euler Lagrange multipliers, or some other variable where weights need be introduced for the minimization process to make sense.

DERIVATION OF ALGORITHM I

The first basic type of estimate is based on the approximation

$$z' = Az + b, \quad z(t_0) = 0. \quad (1)$$

In practice, the following logic is used for the integration algorithm. The standard fourth-order Runge-Kutta formula is used to integrate the equation

$$y' = f(y, t) \quad (2)$$

subject to $y(t_0) = y_0$. If an integration step size of h is specified, then $y(t_0 + h)$ is approximated by numerically integrating equation (2), using two steps of $h/2$ to obtain $\hat{y}(t_0 + h)$ and using one step of step size h to obtain $\hat{\hat{y}}(t_0 + h)$. The local truncation error is estimated by

$$\epsilon(t_0 + h) = \frac{\hat{y}(t_0 + h) - \hat{\hat{y}}(t_0 + h)}{15} \quad (3)$$

and the relative local errors by

$$\delta(t_0 + h) = \frac{\epsilon(t_0 + h)}{|\hat{y}(t_0 + h)|} \quad (3b)$$

(where coordinatewise division is meant; i.e.,

$$\delta_1 = \frac{\epsilon_1}{|\hat{y}_1|}, \quad \delta_2 = \frac{\epsilon_2}{|\hat{y}_2|}, \quad \text{etc.}$$

Absolute error is used when the divisor belongs to a user specified interval about zero.) When integrating in fixed step mode, equation (1) is immediately integrated using a constant forcing function

$$b(t) = \varepsilon(t_0 + h)/h \quad (4)$$

for $t \in [t_0, t_0 + h]$. If a variable step tolerance E is specified, then a doubling-halving technique is used to obtain a step size H such that the maximum magnitude of the components in $\delta(t_0 + H)$ is an element of $[E/100, E]$. After this step size has been chosen, equation (1) is numerically integrated with constant forcing function as described in (4). The process is then carried on iteratively so that the accumulated error estimates from integrating (1) are obtained stepwise just as the initial value problem itself is solved.

Recall that the matrix

$$A = \{a_{ij}\} = \partial f_i / \partial y_j \quad (5)$$

is a function of y , so that, depending on the choice of numerical integration for (1), the midpoint values $\hat{y}(t_0 + h/2)$ must sometimes be saved. The variations of this first algorithm will be obtained by choosing different types of numerical integration for solving equation (1) as follows: (I) Euler's Formula; (II) assume y constant in $[t_k, t_{k+1}]$ and use series approximation; and (III) standard fourth-order Runge-Kutta.

Method I does not require the midpoint values $\hat{y}(t_0 + h/2)$, and it is the least expensive in terms of computer time. Similarly, method II does not require the midpoint values, and it does provide a more accurate solution to $e^{A(t-t_0)+b}$, where in practice it has been quite satisfactory to evaluate $A(t_k + h)$ rather than $A(t_k)$ so that $y(t_k)$ can be discarded before integrating (1). Method II has the advantage that the accuracy can be varied easily by using series of different degrees. Method III requires storing the midpoint values $\hat{y}(t_0 + h/2)$ and it requires the most computer time, but this usually yields the most accurate estimate of the three methods.

The linear estimate for z' is perhaps the simplest of all error estimates. The method is not nearly so advanced as the work presented several years ago at the advent of the computer age by Sterne⁽⁴⁾ and Rademacher⁽⁵⁾. Their work can be used to compute an optimal integration step size (for fixed step mode only) using a minimum of computer time and memory locations. However, storage problems are becoming less and less critical, and computer costs (per computation) have rapidly decreased, while programmer costs have increased to the point where the number of programmer man-hours required for implementation is extremely important. Regardless of the choice of methods I, II, or III, the only additional man-hours required, above the usual cost of integrating equation (2), are those for deriving and coding the matrix A . Since the functions $f(y,t)$ are known analytically, and presumably are programmed for solving (2), a standard matrix differentiation subroutine using linear or quadratic approximations

can be used to eliminate the differentiation and coding errors in computing A. Other advantages to the algorithm are as follows: it can be applied easily to systems as well as single equations; solutions to the adjoint system are not required; it can be applied to variable step integration as easily as to fixed step integration; and the method can be applied easily to any integration formula with a good estimate for local truncation errors.

For the examples presented in this paper, rounding errors did not become a major problem when using the fourth-order Runge-Kutta. However, this algorithm can also be used to estimate accumulated rounding errors. One method of application is to leave all logic the same except for computation of the forcing function $b(t)$.

The local error estimates are now based on the assumption that all significant rounding errors are obtained in the addition step $\hat{y}(t_0 + h) = \hat{y}(t_0) + \Delta\hat{y}$. A logical choice is to assume that the errors are uniformly distributed from $-1/2$ to $+1/2$ in the last significant place carried in $\hat{y}(t)$. Zani⁽⁶⁾ has shown that this provides good estimates for several trajectory problems after several thousand integration steps and that a linear estimate provides good results for truncation errors. His examples also include passing near singular points.

DERIVATION OF ALGORITHM II

The second algorithm is similar to the first in two respects. First, a linearization is involved, and, second,

the analysis is quite simple. Algorithm II is different from the first in that it can be applied directly at the terminal time, but applications are restricted, as mentioned before, to those problems having an additional system of equations

$$g(y,t) = 0 , \quad (1)$$

representing a necessary condition that y satisfy the initial value problem. Given an approximation $\hat{y}(t_f)$ to $y(t_f)$, the object is to find a vector Δ such that

$$g(\hat{y} + \Delta, t_f) = 0 . \quad (2)$$

Although (2) is generally nonlinear, the authors propose a linear model and an iterative process to find Δ as follows:

$$g({}_n\hat{y}, t_f) + C({}_n\hat{y}, t_f) {}_{n-1}\Delta = 0 \quad (3)$$

where ${}_n\hat{y} = {}_{n-1}\hat{y} + {}_{n-1}\Delta$ and $C = (c_{ij}) = \partial g_i / \partial y_j$.

This is the most commonly used technique for solving nonlinear equations. However, difficulty arises from the fact that (2) has an infinitude of solutions (probably a continuum) so that, assuming the linearization⁽³⁾ is reasonable, there can be an infinitude of solutions to (3). This is quite realistic; for example, there is often only one function g_1 and n variables; hence, C is a $1 \times n$ matrix. For linear problems, this does not form an impasse. The following results can be found in⁽⁷⁾.

Theorem: If C is a real $n \times m$ matrix, g an $n \times 1$ real vector, then there exists an $m \times 1$ vector Δ such that (Euclidean norm) (4) $||C \Delta - g||$ is a minimum. All such vectors are characterized by

$$\Delta = C^+g + (I - C^+C)q$$

q arbitrary, where C^+ is the generalized inverse of C . The vector Δ_0 such that $||\Delta||$ is also a minimum is given by

$$\Delta_0 = C^+g$$

Perhaps it should be pointed out that such a differential correction scheme need not be limited to this particular context nor to real valued functions. There are computational methods (see Decell and Kahng⁽⁸⁾) for computing the generalized inverse of an arbitrary $n \times m$ complex matrix.

For the second basic type of error estimate, three different criteria were listed as logical for choosing a solution Δ to equation (2). These criteria were:

(a.) choose Δ_1 such that it satisfies (2) and $||\Delta_1|| \leq ||\Delta||$ for any other Δ satisfying 2, (b.) choose Δ_2 such that all except one of the components of Δ_2 are zero, $||g(\hat{y} + \Delta_2, t_f)||$ is a minimum for all Δ of this form, and $||\Delta_2|| \leq ||\Delta||$ for any other Δ satisfying this property, (actually Δ_2 is probably unique with first property), (c.) choose Δ_3 such that Δ_3 satisfies (2) and $||w \Delta_3|| \leq ||w \Delta||$ for any vector Δ satisfying (2); (w a fixed diagonal weight matrix).

The basic assumption is now made (just as in most solutions to nonlinear problems) that performing an operation iteratively on the linearized system yields an estimate of this operation for the nonlinear system. The preceding theorem obviously makes it possible to obtain solutions to the linear system for criteria (a.) and (b.); another not so obvious application of the theorem will produce Δ_3 satisfying criteria (c.) (the authors tacitly assume a subroutine for computing C^+ is available). For a fixed matrix w , one must obtain Δ_3 satisfying $\|C \Delta_3 - g\|$ is a minimum and $\|w \Delta_3\| \leq \|w \Delta\|$ for any Δ such that $\|C \Delta - g\|$ is a minimum. Toward that end, recall that all vectors Δ such that $\|C \Delta - g\|$ is a minimum are characterized by

$$\Delta = C^+g + (I - C^+C)q ,$$

$q \in R^m$, q arbitrary. Then

$$w \Delta = \{wC^+g\} + \{w(I - C^+C)\}q$$

which is of the form

$$w \Delta = p + Rq$$

where p is a known vector and R is a known $m \times n$ matrix.

Thus, another application of the theorem yields a solution

$$q = R^+(-p) + (I - R^+R)\xi ;$$

ξ is arbitrary and $q_0 = R^+(-p)$ is chosen which yields

$$\Delta_3 = C^+g + (I - C^+C)R^+(-p) ,$$

and $\|C \Delta - g\|$ and $\|w \Delta\|$ have been simultaneously minimized.

It should be noted that, at the time of this writing, Speed and Decell⁽⁹⁾ have completed a paper for differential correction via generalized inverse for overdetermined systems. This method has converged rapidly when applied to several problems for which conventional methods either converge slowly or diverge.

NUMERICAL EXAMPLES

Since the numerical examples are based on trajectory optimization problems, and since this type of problem stimulated the applications for the second algorithm, a skeletal outline of the general trajectory optimization problem will be presented. A detailed description of this theory and numerical methods is made by Tapley and Lewallen⁽¹⁰⁾:

The trajectory optimization problem involves the determination of the m -vector of control variables $u(t)$ in the interval $t_0 \leq t \leq t_f$ such that a scalar performance index of the form

$$I = G(x_f, t_f) + \int_{t_0}^{t_f} Q(x, u, t) dt \quad (1)$$

is minimized, while the p-vector of initial conditions

$$L(x_0, t_0) = 0 \quad (2)$$

and the q-vector of terminal conditions

$$M(x_f, t_f) = 0 \quad (3)$$

are satisfied and the n first-order, nonlinear, differential equations

$$\dot{x} = f(x, u, t) \quad (4)$$

are satisfied. The vector x is an n-vector of state variables and t is the independent variable time.

The first necessary conditions corresponding to the above stated problem are as follows:

In the interval of interest

$$\dot{x} = \tilde{H}_\lambda^T \quad \dot{\lambda} = -\tilde{H}_x^T \quad 0 = \tilde{H}_u \quad (5)$$

at the known initial time

$$L(x_0, t_0) = 0 \quad \left[(P_x + \lambda^T) \right]_{t_0} dx_0 = 0 \quad (6)$$

and at the unknown final time

$$M(x_f, t_f) = 0$$

$$\left[(P_x - \lambda^T) \right]_{t_f} dx_f = 0 \quad (7)$$

$$\left[(P_t + \tilde{H}) \right]_{t_f} dt_f = 0$$

where the scalar functions P and \tilde{H} are defined as

$$P = G(x_f, t_f) + \mu^T L(x_0, t_0) + v^T M(x_f, t_f) \quad (8)$$

$$\tilde{H} = Q(x, u, t) + \lambda^T f(x, u, t) \quad (9)$$

The scalar \tilde{H} is referred to as the generalized Hamiltonian and μ and v are Lagrange multipliers.

In an indirect optimization method, the condition $\tilde{H}_u = 0$ yields m algebraic equations which can be used to eliminate the m control variables in Eqs. (5-a) and (5-b) to yield

$$\dot{x} = H_x^T \quad \dot{\lambda} = -H_\lambda^T \quad (10)$$

where $H = \tilde{H}[x, \lambda, u(x, \lambda, t), t]$. Eqs. (6), (7), and (10) lead to a conventional two-point boundary value problem. If the $2n$ -vectors z and $F(z, t)$ are defined as

$$z^T = [x^T \mid \lambda^T] \quad F^T = [H_\lambda \mid -H_x] \quad (11)$$

then Eq. (10) can be expressed as

$$\dot{z} = F(z,t) . \quad (12)$$

Furthermore, Eqs. (6) and (7) define the boundary conditions

$$g(z_0, t_0) = 0 \quad h(z_f, t_f) = 0 \quad (13)$$

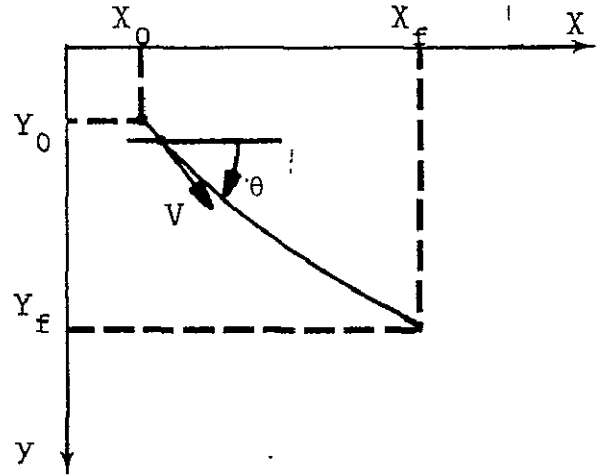
where g is an n -vector and h is an $n+1$ vector. Hence, the $2n$ equations in Eq. (12) are subjected to the $2n+1$ boundary conditions in Eq. (13).

In the present text, all theory is directed toward solving Eq. (12) as an initial value problem. Hence, the dependent variables will consist of an augmented vector of state variables and Lagrange multipliers. In solving optimization problems, a split boundary value problem is usually encountered where the state variables are known at t_0 and the multipliers at t_f . Many methods are used for solving such problems, but most of them involve integration of an initial value problem (if for no other reason) to verify the final solution. Methods for solving boundary value problems will not be discussed. Numerical results will consist of integrating Eq. (12) with the correct initial conditions to give an optimal solution.

Brachistochrone Problem

The Brachistochrone Problem studied in this investigation involves the determination of the closed-form solution for a particle moving under the influence of a constant gravitational acceleration from the point (x_0, y_0) at t_0

to the point (x_f, y_f) in such a manner that travel time is minimized.



The nonlinear differential equations that correspond to Eq. (12) are

$$\dot{z}_1 = \dot{x} = - \frac{\sqrt{2g(y - a)}}{\sqrt{\lambda_1^2 + \lambda_2^2}} \lambda_1 = F_1$$

$$\dot{z}_2 = \dot{y} = - \frac{\sqrt{2g(y - a)}}{\sqrt{\lambda_1^2 + \lambda_2^2}} \lambda_2 = F_2$$

$$\dot{z}_3 = \dot{\lambda}_1 = 0 = F_3$$

$$\dot{z}_4 = \dot{\lambda}_2 = \frac{g\sqrt{\lambda_1^2 + \lambda_2^2}}{\sqrt{2g(y - a)}} = F_4$$

where g is the acceleration of gravity and $a = Y_0 - \frac{V_0^2}{2g}$ and involves only the initial conditions. It should be noted that by Eq. (5-c) $\tan \theta = \lambda_2/\lambda_1$. The quantity a is introduced to avoid the singularity if it is desired to start

from $y(t_0) = Y_0 = 0$. In this case, an initial velocity must be given.

For numerical implementation, the following values were selected:

$$\begin{array}{lll} t_0 = 0 & X_f = 5.0 & g = 32.1741 \\ X_0 = 0 & Y_f = 8.0 & \\ Y_0 = 1.0 & a = 0.5 & \end{array}$$

These values lead to the closed-form determination of $c_1 = -5.711799$, $c_2 = -.068417163$, $\lambda_x = -.03573496$ and $t_f = .60766149$ which determine the solutions

$$x = \frac{1}{4g\lambda_x^2} \left[2\lambda_x(c_1 - gt) - \sin [2\lambda_x(c_1 - gt)] \right] + c_2$$

$$y = a + \frac{1}{2g\lambda_x^2} \sin^2 [\lambda_x(c_1 - gt)] .$$

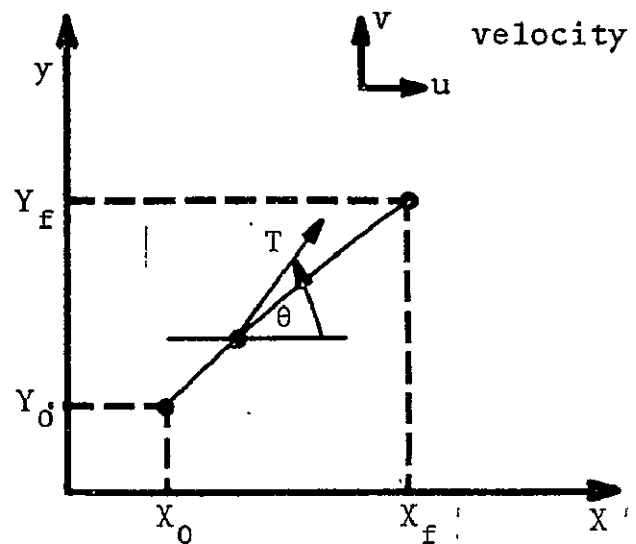
$$\lambda_y = \lambda_x \cot [\lambda_x(c_1 - gt)]$$

The generalized Hamiltonian used for algorithm II is given by

$$H = -\sqrt{\lambda_x^2 + \lambda_y^2} \sqrt{2g(y - a)}$$

Flat Earth Problem

The Flat Earth Problem studied in this investigation involves the determination of the closed-form solution for rocket flight over a flat Earth so that it will arrive at a fixed height Y_f at t_f with zero vertical velocity and maximum horizontal velocity. The thrust-to-mass ratio is constant and the horizontal displacement is not constrained.



The nonlinear differential equations that correspond to Eq. (12) are

$$\dot{z}_1 = \dot{x} = u = F_1$$

$$\dot{z}_2 = \dot{y} = v = F_2$$

$$\dot{z}_3 = \dot{u} = \frac{T\lambda_u}{m\sqrt{\lambda_u^2 + \lambda_v^2}} = F_3$$

$$\dot{z}_4 = \dot{v} = \frac{T\lambda_v}{m\sqrt{\lambda_u^2 + \lambda_v^2}} - g = F_4$$

$$\dot{z}_5 = \dot{\lambda}_x = 0 = F_5$$

$$\dot{z}_6 = \dot{\lambda}_y = 0 = F_6$$

$$\dot{z}_7 = \dot{\lambda}_u = -\lambda_x = F_7$$

$$\dot{z}_8 = \dot{\lambda}_v = -\lambda_y = F_8$$

where g is the acceleration of gravity and T/m is the thrust to mass ratio, which is constant. It should be noted that by Eq. (5-c) $\tan \theta = \lambda_v/\lambda_u$.

For numerical implementation, the following values were selected:

$$t_0 = 0 \qquad t_f = 274.28710 \text{ sec}$$

$$X_0 = 0 \qquad Y_f = 528,000.0 \text{ ft}$$

$$Y_0 = 0 \qquad T/m = 100.0 \text{ ft/sec}^2$$

$$U_0 = 0 \qquad g = 32.0 \text{ ft/sec}^2$$

$$V_0 = 0$$

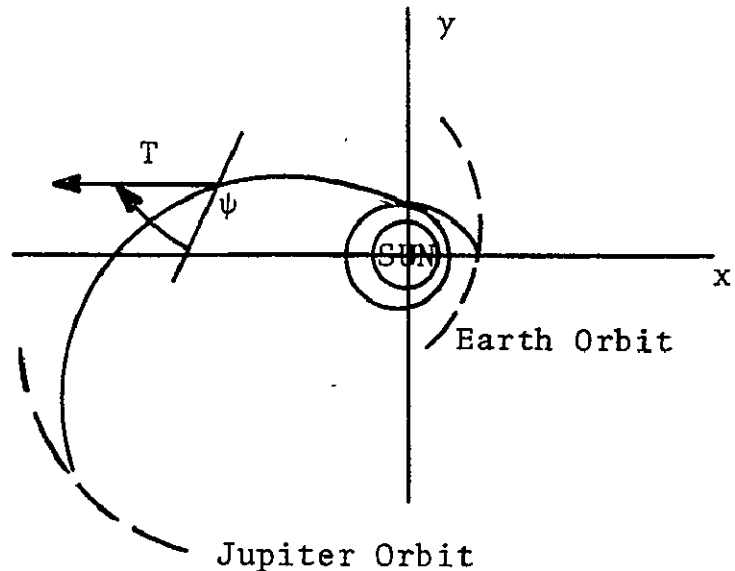
These values lead to the closed-form determination of
 $b = 0.90877929$ $c = .0038698512$ $u(t_f) = 25000.0$ ft/sec.
 which in turn determine the solutions

$$\begin{aligned}
 x &= \frac{a}{c^2} \left[\sqrt{1 + b^2} - \sqrt{1 + (b - ct)^2} \right. \\
 &\quad \left. - (b - ct) \log \left(\frac{b + \sqrt{1 + b^2}}{b - ct + \sqrt{1 + (b - ct)^2}} \right) \right] \\
 y &= \frac{a}{2c^2} \left[(b - ct) \sqrt{1 + (b - ct)^2} - b\sqrt{1 + b^2} \right] \\
 &\quad - \log \left(\frac{b + \sqrt{1 + b^2}}{b - ct + \sqrt{1 + (b - ct)^2}} \right) + 2ct \sqrt{1 + b^2} \\
 &\quad - \frac{1}{2} gt^2 \\
 u &= \frac{a}{c} \log \left[\frac{b + \sqrt{1 + b^2}}{b - ct + \sqrt{1 + (b - ct)^2}} \right] \\
 v &= \frac{a}{c} \left[\sqrt{1 + b^2} - \sqrt{1 + (b - ct)^2} \right] - gt
 \end{aligned}$$

$$\begin{aligned}\lambda_x &= 0 \\ \lambda_y &= c \\ \lambda_u &= 1 \\ \lambda_v &= \lambda_v(t_f) - \lambda_y(t_f - t)\end{aligned}$$

Earth-Jupiter Problem

The Earth-Jupiter Problem studied in this investigation does not have a closed-form solution. It involves minimizing the fuel expenditure for a three-dimensional, constant-magnitude thrust rocket that leaves Earth on December 1, 1983. The terminal conditions correspond to the state of Jupiter at the arrival time.



The nonlinear differential equations that correspond to Eq. (12) are

$$\dot{z}_1 = \dot{x} = u = F_1$$

$$\dot{z}_2 = \dot{y} = v = F_2$$

$$\dot{z}_3 = \dot{z} = w = F_3$$

$$\dot{z}_4 = \dot{u} = -\frac{\mu x}{r^3} + \frac{\beta c}{m} \cos \theta \cos \psi = F_4$$

$$\dot{z}_5 = \dot{v} = -\frac{\mu y}{r^3} + \frac{\beta c}{m} \cos \theta \sin \psi = F_5$$

$$\dot{z}_6 = \dot{w} = -\frac{\mu z}{r^3} + \frac{\beta c}{m} \sin \theta = F_6$$

$$\dot{z}_7 = \dot{m} = -\beta = F_7$$

$$\dot{z}_8 = \dot{\omega}_x = -\frac{\mu}{r^5} \left[\lambda_u (3x^2 - r^2) + 3\lambda_v xy + 3\lambda_w xz \right] = F_8$$

$$\dot{z}_9 = \dot{\omega}_y = -\frac{\mu}{r^5} \left[3\lambda_u xy + \lambda_v (3y^2 - r^2) + 3\lambda_w yz \right] = F_9$$

$$\dot{z}_{10} = \dot{\omega}_z = -\frac{\mu}{r^5} \left[3\lambda_u xy + 3\lambda_v yz + \lambda_w (3z^2 - r^2) \right] = F_{10}$$

$$\dot{z}_{11} = \dot{\lambda}_u = -\omega_x = F_{11}$$

$$\dot{z}_{12} = \dot{\lambda}_v = -\omega_y = F_{12}$$

$$\dot{z}_{13} = \dot{\lambda}_w = -\omega_z = F_{13}$$

$$\dot{z}_{14} = \dot{\lambda}_m = \frac{-T \sqrt{\lambda_u^2 + \lambda_v^2 + \lambda_w^2}}{m^2} = F_{14}$$

where $r = \sqrt{x^2 + y^2 + z^2}$, $\mu = GM$, G is the universal gravitational constant, M is the central body mass, $m = m_0 - \beta t$ is the mass of the spacecraft, β is the mass flow rate, and c is the relative exhaust velocity of the propellant. The angles ψ and θ are the in-plane and out-of-plane thrust angles, respectively. The terminal time t_f is unknown.

The constants of motion to be used to evaluate the numerical error estimates for algorithm II are

$$E_1 = \frac{\mu}{r^3} (\lambda_u x + \lambda_v y + \lambda_w z) + \frac{T \sqrt{\lambda_u^2 + \lambda_v^2 + \lambda_w^2}}{m} - [\omega_x u + \omega_y v + \omega_z w] - \lambda_m \beta$$

$$E_2 = \lambda_v w - \lambda_w v - \omega_y z + \omega_z y$$

$$E_3 = -\lambda_u w + \lambda_w u + \omega_x z - \omega_z x$$

$$E_4 = \lambda_u v - \lambda_v u - \omega_x y + \omega_y x$$

Since a closed-form solution to the differential equations was not known, the problem had to be solved numerically. This solution was obtained in double precision by the Method of Perturbation Functions as outlined in Ref. 10. Hence, the initial conditions used for this study were

$$\frac{\beta c}{m} = .24497092E - 4$$

$$\begin{aligned}
\mu &= .296007536E - 3 \\
x(t_0) &= .8321727E - 0 \\
y(t_0) &= -.52919964E - 0 \\
z(t_0) &= .0 \\
u(t_0) &= .92217756E - 2 \\
v(t_0) &= .14807388E - 1 \\
w(t_0) &= .0 \\
m(t_0) &= .1E + 1 \\
\omega_x(t_0) &= .82198202E + 1 \\
\omega_y(t_0) &= .46426936E + 1 \\
\omega_z(t_0) &= -.78153625E + 1 \\
\lambda_u(t_0) &= .16289938 \\
\lambda_v(t_0) &= -.22067387E - 1 \\
\lambda_w(t_0) &= -.59754685E - 3 \\
\lambda_m(t_0) &= .18510952E + 4 \\
t_f &= .90507346E + 3
\end{aligned}$$

COMPARISONS OF STANDARD INTEGRATION ALGORITHMS

The two error estimate algorithms are easily applied to trajectory optimization problems, and implementation does not require a prohibitive number of programmer man-hours. Another prerequisite for a numerical algorithm to be usable is that it must require a reasonable amount of computer time compared to other methods of similar accuracy and dependability. In this section, data will be presented to show the accuracy and speed of several well known types of numerical integration algorithms. Any measurement of speed or accuracy of an algorithm is highly dependent on the type of computer used, programming techniques, and many other factors. All data presented in tables 1 through 12 were obtained using the UNIVAC 1108 computer. In compiling this data, all execution times are recorded for those executions having no output, derivative and calling programs were kept identical wherever possible, and executions of programs were all made on the same machine with the same system and near the same time when possible. Every effort has been taken to provide a good relative test of the speed and accuracy of these algorithms. In addition, these data will provide a yardstick for measuring the percentage of additional time required to compute the estimates in algorithms I and II.

Three basic types of numerical integration are compared in tables 1 through 6. The standard fourth-order Runge-Kutta (RK) and Adams-Moulton (AM) integration methods were chosen because of their widespread usage and reputation for dependability. As a third choice, a fifth-order Runge-Kutta (RK5) formula was chosen. Since this formula is not well known, the authors state it here.

Denote $y(t_0)$ by y_0 and y' by $f(y,t)$ then

$$y(t_0 + h) = y_0 + 1/12[K_1 + 5K_3 + 5K_5 + K_6] ,$$

where

$$K_1 = hf(y_0, t_0)$$

$$K_2 = hf(y_0 + K_1/2, t_0 + h/2)$$

$$K_3 = hf\left(y_0 + 1/10(2K_1 - (3 - \sqrt{5})K_2), t_0 + \left(\frac{5 - \sqrt{5}}{10}\right)h\right)$$

$$K_4 = hf(y_0 + 1/4\{K_1 + K_2\}, t_0 + h/2)$$

$$K_5 = hf(y_0 + 1/20[\{1 - \sqrt{5}\}K_1 - 4K_2 + \{5 + 3\sqrt{5}\}K_3 + 8K_4], t_0 + 1/10\{5 + \sqrt{5}\}h)$$

$$K_6 = hf(y_0 + 1/4[\{\sqrt{5} - 1\}K_1 + \{2\sqrt{5} - 2\}K_2 + \{5 - \sqrt{5}\}K_3 - 8K_4] + \{10 - 2\sqrt{5}\}K_5, t_0 + h) .$$

A derivation for this formula can be found in Ref. 11. This formula has not been used nearly so much as the first two, but the algorithm has been programmed and successfully used at MSC for several problems.

The data for integrating examples 1 through 3, using fixed step integration, are found in tables 1 through 3, respectively. All numerical solutions are acceptable, and the execution times are extremely good. Each of the three numerical examples is often used to "test" numerical algorithms for solving optimization problems. Hence, step sizes were known such that at least four to five significant (decimal) place accuracy would be maintained using fourth-order fixed step Adams-Moulton integration. Using this step size brings attention to some interesting facts. The "flat Earth" trajectory and the Earth-Jupiter transfer integrations require 275 and 906 steps, respectively. For each of these problems, the fourth-order Runge-Kutta routine yields more accurate results than the fifth-order Runge-Kutta. The indication is that the fifth-order RK is much more susceptible to accumulated roundoff error. The fact that the variable step fifth-order RK obtained excellent results, using a larger step, also tends to uphold this assumption. It is suggested that total double precision be used for integrating a large number of steps using RK5. The fourth-order RK is also more accurate than the Adams-Moulton solution for the Earth-Jupiter problem, indicating slight numerical instabilities may have been encountered.

Data for variable step integrations are found in tables 4, 5, and 6. In each of these tables, the first column represents data from an integration using a test on the estimate for absolute local truncation errors to determine step size control, whereas all other data are taken from routines using relative error tests.

Two significant results, the increased relative speed and the increased relative accuracy of the fifth-order RK5, are at once evident. The small step sizes used for fixed step integrations were a bit unfavorable to each of the partial double precision routines (especially to RK5). Even though the same error tolerance, 10^{-5} , was provided for each routine, the fifth-order Runge-Kutta routine maintained much better accuracy than any of the other routines using relative error tests. This data indicates that if all routines were forced to maintain the same accuracy, then RK5 would probably execute as fast as any of the other routines.

Although there is an attempt to keep the number of tables and the amount of data down to a minimum, one detail should be mentioned. Of the three numerical examples in this paper, the largest system of derivatives requires less than 1 millisecond per evaluation (using the UNIVAC 1108). For systems requiring much more time to evaluate, the Adams-Moulton integration is often faster than Runge-Kutta, provided the same error tolerance is used for both types of integration. Thus, for problems in this category, one might consider an application of the linear estimate, used in algorithm I, to Adams-Moulton integration. The error equations $z' = Az + b$ could be integrated by Euler's method or a higher order multistep method, requiring no more backward values than those retained for the Adams-Moulton integration.

The results of this section, in the sense of comparing one algorithm to another, are exactly what one would expect. No single algorithm appears to be "best." Even the large

errors (E4 in table 5) in the vertical velocity—for the flat Earth trajectory—are a natural result of allowing 10^{-5} relative errors when this velocity had built up to more than 1,000 ft/sec during the trajectory. This is a good example where an absolute error test is sometimes preferred over a relative one. However, the large relative errors as this velocity approached zero at the terminal time were expected, and the maximum absolute errors of .137 ft/sec are perhaps acceptable. Generally, accuracy is maintained and execution time decreased by using variable step integration. Execution times and accuracy for the variable fourth-order Runge-Kutta will be presented along with the data for algorithm I. Here again, the data are comparable to the results in tables 4, 5, and 6.

NUMERICAL RESULTS |

In general, the numerical results using algorithms I and II are very encouraging. The accuracy is much better, and the computer execution time is much shorter than had been anticipated. The results can be found in tables 7 through 11. Only data obtained from variable step size integration will be presented. In many instances algorithm I predicts the error to two significant figures, and the estimate is seldom off by more than an order of magnitude. For example, using a fourth-degree series approximation and an error tolerance of 10^{-6} , the estimate is off by more than an order of magnitude only twice, and this discrepancy is due to rounding errors rather than to poor truncation estimates. Moreover, in each of these two cases, an application of algorithm II accurately indicated the true error (see table 11).

The application of algorithm I always yields error estimates for each dependent variable; such is not the case for applying algorithm II. In fact, three different types of situations can be seen in the three different numerical examples. For the brachistochrone problem, there is one equation $g(y,t)$ in two variables including a control variable. In one sense, this is logical (note x does not appear in $g(y,t)$); in obtaining actual numerical solutions, one might integrate until $x = x_f$ and then predict the errors in y and λ_y . For the flat Earth problem only one variable appears in $g(y,t)$. Note that this is because the proper initial conditions were known in closed form; for any other iteration, the variables would have appeared. Moreover, the remaining variable—the terminal vertical velocity—is critical. In each of these two problems, only a minimum norm (1) and a least squares (2) solution are obtained using algorithm II.

For the Earth-Jupiter transfer problem, there are four equations, and all 14 variables appear explicitly. The first equation is analogous to the conservation of energy principle, and the last three equations are analogous to the conservation of angular momentum principle. All three variations of algorithm II are applied to the transfer problem. For the weighted variation (3), the diagonal weight matrix $w = \{w_{ii}\} = 1/|\hat{y}_i(t_f)|$, $w_{ij} = 0$ is used.

One would not expect algorithm II to be as accurate as the first, and, indeed, this is the case. However, good estimates are obtained as can quickly be seen in tables 7 and 8. A detailed investigation of the accuracy of each algorithm is perhaps warranted.

Evaluation of Algorithm II

All numerical results for applying algorithm II are based on the terminal solution $\hat{y}(t_f)$ obtained from a variable step Runge-Kutta integration with an error tolerance of 10^{-6} . For comparison purposes, the corresponding algorithm I series estimates are included in each table. These data are found in tables 7, 8, and 11. Although the algorithm I estimates are more accurate for the brachistochrone problem, note that the least squares algorithm II estimates are off only by about one multiplicative factor of 2. Moreover, the algorithm II estimate for the flat Earth trajectory problem is very accurate and much better than the algorithm I estimates. Data for applying algorithm II to the transfer problem are found in table 11. Of all the data presented in this paper, the most careful attention must be given to results of the Earth-Jupiter transfer problem, for there is no closed form solution available for this problem. It should be pointed out that a great deal of work has already been done concerning this problem, and the Adams-Moulton fixed step total double precision integration (column II) has been considered the best available solution to date. Notice also that the Runge-Kutta fixed step integration (table 3) yields smaller maximum errors E1 through E4. The error estimates using three variations of algorithm I, involving a variable step integration with two different tolerances for each variation, also agree very closely with the estimates obtained using the "best available solutions" (see tables 9 and 10). For each variable in the Earth-Jupiter transfer problem, the small interval formed by the numbers in columns II and III of table 11 will be referred to as the estimate interval.

The estimates in columns IV, V, and VI of table 11 are, respectively, the (c) weighted minimum norm, (b) least squares, and (a) minimum norm solutions derived for algorithm II. Recall that (c) and (a) are lower bound types of estimates (in norm), so that choosing the larger of the two estimates would be logical. If this is done, the resulting estimate misses the estimate interval by an order of magnitude for five variables (indicated by asterisks), but never by as much as two orders of magnitude. It is of interest to note that on each of these five occasions, algorithm I accurately predicts the error.

Although algorithm II seems less accurate than the first, it should be pointed out that this is almost a free estimate. Deriving, coding, and checking the matrix of partials for the system $g(y,t)$ is usually much simpler than for the system $y' = f(y,t)$. The algorithm can be applied directly at any time during the trajectory, including the terminal time, without storing any previous values of y or performing any previous computations. Also, the computer time required is certainly not prohibitive. Moreover, one computation has replaced a set of "small" residuals for the system $g(y,t)$ with individual error estimates for the components of y : these estimates are derived in a manner entirely independent of other methods such as changing types of numerical integration, changing variable step error tolerances, or halving-doubling step size techniques for fixed step integrations.

Evaluation of Algorithm I

In this section, the authors would like to show that algorithm I always provides the "best" error estimate available; however, natural laws seem to preclude proving such a general statement. It has already been pointed out that this algorithm has many features making it comparable to other methods (i.e., the method can be applied to systems of equations; implementation is straightforward and requires few programmer man-hours; variable step integration can be used; and no adjoint system need be solved. Final consideration involves the estimates and the speed of the algorithm as compared to others of comparable accuracy. Evaluations of algorithm I, simultaneously considering accuracy and speed, vary between the following two extremes. Using an error tolerance of 10^{-5} for integrating the flat Earth problem, the solution and excellent error estimates were obtained in less time than the other methods which obtained only the solution. In fact, in every numerical example using a 10^{-5} error tolerance and Euler's method for integrating the error equations, algorithm I proved to be faster than the fifth-order Runge-Kutta. At the other extreme, using an error tolerance of 10^{-6} and RK4 for integrating the error equations, algorithm I requires about twice as much time as RK5 and four times as much as the single precision Adams-Moulton. Here again, excellent estimates are obtained, even with Euler's method (which requires about the same amount of time as RK5). Even this most unfavorable comparison indicates that algorithm I would be competitive with the simplest error estimating devices such as comparing solutions using different types of integration, successively smaller step size, or smaller error tolerances.

At the outset of this investigation, error estimates missing the true error by two orders of magnitude were expected. Contrary to these expectations, using an error tolerance of 10^{-6} , the estimates seldom missed the true error by as much as a multiplicative factor of 2. For example, of the 18 estimates in table 7, for the brachistochrone problem, one significant figure of accuracy occurs in 14 of the estimates, and a multiplicative factor of 2 would sufficiently modify the four remaining.

Similarly, of the 24 estimates in table 8, for the flat Earth problem, no estimate misses the true error by as much as a multiplicative factor of 2. For the Earth-Jupiter transfer problem, inspection of the differential equations shows that Runge-Kutta integration would be exact for the seventh variable so that the estimates for this variable are essentially correct and the error is from propagated rounding errors. The same is probably true of the fourteenth variable, since extremely small local errors occurred and each of the three estimates agree. Notice that these two variables could have been eliminated, since one can easily solve for m , and λ_m is not needed. Applications of algorithm II correctly estimated these errors. It was pointed out earlier that algorithm I correctly predicted an error when the second algorithm estimate missed the true error by two orders of magnitude. Assuming estimates from both algorithms are available, a test for consistency between the two would have indicated all bad estimates in these data. Of the remaining variables in table 10, no other estimate is in error by as much as a factor of 2 except for the thirteenth variable which is in error by a factor of 5. Another

interesting characteristic appears in this thirteenth row of table 10. There seems to be a monotonic growth toward the correct estimate in going from columns 1 to 2 to 3. This behavior was predicted, although Euler's method sometimes provides better results than the series. If all three methods of algorithm I are computed, then variations in the estimates can also be used to detect faulty estimates. Notice that the Runge-Kutta integration of the error equations to obtain the estimates in table 9 are much better than those obtained using Euler's method or a series. The large variations indicate the possibility of faulty estimates, and, in fact, several estimates are poor compared to those in the remaining tables. This added dependability check requires the computation of all three estimates; essentially, however, the first two are required for the RK4 computations and, as shown in table 12, the time required to integrate the error equations is only a fraction of the total execution time. All three estimates can be obtained in slightly more time than that required by the RK4 variation.

CONCLUDING REMARKS

No new mathematical theory is presented in this paper. However, there is a definite lack of literature concerning numerical error estimates for integrating differential equations. Zani's paper is one of the few good recent publications on the subject. This lack of literature directly affects computation center libraries. For example, to the authors' knowledge, there is no linear estimate program similar to algorithm I in the CAD subroutine library

or in use by any customers; yet, more than 50 percent of the computer time (in the scientific area) is often used to integrate differential equations. Moreover, there has been a great deal of research concerning problems where constants of motion are monitored to provide an error check for numerical integration. Although generalized matrix inversion had been established as a method for solving these overdetermined systems, there seem to be no applications to estimating errors for this type of initial value problem. Since the accuracy of estimates cannot be established by proof, one must resort to numerical examples. The authors are indebted to those who work in the field of trajectory optimization for the nontrivial examples making the extrapolation to some unsolved problems seem realistic.

REFERENCES

1. Gerber, W. J., Lewallen, J. M.: Private Communication, 1966.
2. Schwausch, O. A.: "Numerical Error Comparisons for Integration of Near Earth Orbits in Various Coordinate Systems," Engineering Mechanics Research Laboratory Report 1094, January 1968.
3. Rainbolt, M. B.: "Coordinate Systems Influence on the Numerical Solution of the Trajectory Optimization Problem," Masters Thesis, University of Houston, 1968.
4. Sterne, T. E.: "The Accuracy of Numerical Solutions of Ordinary Differential Equations," Mathematical Tables and Other Aids to Computation, VII, 1953.
5. Rademacher, H. A.: On the Accumulation of Errors in Processes of Integration on High-Speed Calculating Machines, Proceedings of a Symposium on Large-Scale Digital Calculating Machinery, Cambridge, Massachusetts, Harvard University Press, 1948.
6. Zani, R. H. and Hannen, R.: "A Computer Study of the Estimated Propagation in the Numerical Integration of Trajectory Equations," Journal of the Astronautical Sciences, Vol. XIV, No. 1, January 1968.
7. Decell, H. P.: "The Concept of Generalized Inversion of Arbitrary Complex $n \times m$ Matrix," NASA-MS-64-ED6, 1964.

8. Decell, H. P. and Kahng, S. W.: "An Iterative Method for Computing the Generalized Inverse of a Matrix," NASA Technical Note D3464, June 1966.
9. Decell, H. P. and Speed, F. M.: "Differential Correction Schemes in Nonlinear Regression," Technometrics (to appear).
10. Tapley, B. D. and Lewallen, J. M.: "Comparison of Several Numerical Optimization Methods," Journal of Trajectory Optimization Theory and Applications, Vol. 1, No. 1, July 1967.
11. Luther, H. A. and Konen, H. P.: "Some Fifth-Order Classical Runge-Kutta Formulas," SIAM Review, Vol. 7, No. 4, October 1965.

TABLES

PRECEDING PAGE BLANK NOT FILMED.

TABLE 1

BRACHISTOCHRONE PROBLEM

FIXED STEP MODE

STEP SIZE = 0.025

INTEGRATION ROUTINE	AM		RK5		AM		RK4	
TYPE OF INTEGRATION	ADAMS-MOULTON		RUNGE-KUTTA		ADAMS-MOULTON		RUNGE-KUTTA	
USE OF DOUBLE PRECISION ARITHMETIC	TOTAL		ACCUMULATION		ACCUMULATION		ACCUMULATION	
INTEGRATION TIMES IN MILLISECONDS	38		53		32		42	
NUMBER OF INTEGRATION STEPS	25		25		25		25	
NUMBER OF DERIVATIVE EVALUATIONS	61		152		61		103	
STORAGE USED	1777 ₈		1510 ₈		2234 ₈		1153 ₈	
MAXIMUM ERRORS	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E
E1	-3.3E-4	-7.4E-5	-1.1E-6	-2.6E-7	-3.3E-4	-7.4E-5	-1.2E-5	-1.1E-5
E2	4.2E-4	5.2E-5	1.3E-6	1.7E-7	4.2E-4	5.2E-5	3.5E-5	4.4E-6
E3	-7.8E-6	-1.9E-4	-2.3E-8	6.8E-7	-7.8E-6	-1.9E-4	-3.7E-7	9.5E-6

TABLE 2
 FLAT EARTH TRAJECTORY PROBLEM
 FIXED STEP MODE
 STEP SIZE = 1.0

INTEGRATION ROUTINE	' AM '		RK5		AM		RK4	
TYPE OF INTEGRATION	ADAMS-MOULTON		RUNGE-KUTTA		ADAMS-MOULTON		RUNGE-KUTTA	
USE OF DOUBLE PRECISION ARITHMETIC	TOTAL		ACCUMULATION		ACCUMULATION		ACCUMULATION	
INTEGRATION TIMES IN MILLISECONDS	376		597		352		499	
NUMBER OF INTEGRATION STEPS	275		275		275		275	
NUMBER OF DERIVATIVE EVALUATIONS	561		1652		561		1103	
STORAGE USED	1777 _B		1510 _B		2234 _B		1153 _B	
MAXIMUM ERRORS	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E
E1	-7.4E-5	-1.6E-10	-3.2E-0	-9.8E-7	-3.1E-2	-9.0E-9	-1.4E-1	-4.9E-8
E2	1.2E-4	-5.8E-10	-7.7E-1	-1.5E-6	-3.1E-3	-8.2E-9	-1.5E-2	-4.2E-8
E3	2.6E-7	-2.2E-11	-1.8E-2	-7.1E-7	3.5E-5	-1.2E-9	-7.3E-4	-2.9E-8
E4	5.9E-7	6.0E-4	-3.0E-3	1.4E-0	4.7E-5	5.2E-2	5.0E-5	5.4E-2

TABLE 3

EARTH JUPITER TRANSFER PROBLEM

FIXED STEP MODE

STEP SIZE = 1.0

INTEGRATION ROUTINE	AM		RK5		AM		RK4	
TYPE OF INTEGRATION	ADAMS- MOULTON		RUNGE- KUTTA		ADAMS- MOULTON		RUNGE- KUTTA	
USE OF DOUBLE PRECISION ARITHMETIC	TOTAL		ACCUMULATION		ACCUMULATION		ACCUMULATION	
INTEGRATION TIMES IN MILLISECONDS	3630		6103		2973		4424	
NUMBER OF INTEGRATION STEPS	906		906		906		906	
NUMBER OF DERIVATIVE EVALUATIONS	1824		5439		1824		3628	
STORAGE USED	1777 ₈		1510 ₈		2234 ₈		1153 ₈	
MAXIMUM ERRORS	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E
E1	-6.1E-9	7.2E-6	-7.6E-9	-9.0E-6	6.1E-9	7.2E-6	7.3E-11	8.6E-8
E2	-2.7E-9	2.2E-6	9.9E-9	-8.1E-6	-2.7E-9	2.2E-6	-1.0E-10	8.4E-8
E3	-7.8E-7	-7.1E-5	-7.9E-8	-7.1E-6	-7.8E-7	-7.1E-5	5.0E-9	4.5E-7
E4	-3.7E-8	-3.7E-8	-3.7E-6	-3.7E-6	-3.9E-8	-3.9E-8	1.5E-8	1.5E-8

TABLE 4
 BRACHISTOCHRONE PROBLEM
 VARIABLE STEP MODE
 INITIAL STEP SIZE 0.025
 $\epsilon = 10^{-5}$

INTEGRATION ROUTINE	ABS ERROR TEST AM		RK5		AM		(RELATIVE TEST) AM	
TYPE OF INTEGRATION	ADAMS-MOULTON		RUNGE-KUTTA		ADAMS-MOULTON		ADAMS-MOULTON	
USE OF DOUBLE PRECISION ARITHMETIC	TOTAL		ACCUMULATION		ACCUMULATION		TOTAL	
INTEGRATION TIMES IN MILLISECONDS	34		58		25		32	
NUMBER OF INTEGRATION STEPS	20		10		17		17	
NUMBER OF DERIVATIVE EVALUATIONS	58		161		52		52	
STORAGE USED	1777 ₈		1510 ₈		2234 ₈		1777 ₈	
MAXIMUM ERRORS	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E
E1	-3.4E-4	-7.4E-5	-5.4E-7	-1.3E-7	-4.0E-4	-8.0E-5	-4.0E-4	-8.0E-5
E2	4.3E-4	5.4E-5	1.1E-6	1.3E-7	4.9E-4	6.0E-5	4.9E-4	6.0E-5
E3	-7.8E-6	2.1E-4	-1.5E-8	5.2E-7	-7.8E-6	2.6E-4	-7.8E-6	2.6E-4

TABLE 5

FLAT EARTH TRAJECTORY PROBLEM

VARIABLE STEP MODE

INITIAL STEP SIZE = 1.0

$\epsilon = 10^{-5}$

INTEGRATION ROUTINE	ABS ERROR TEST AM		'RKS		AM		(RELATIVE TEST) AM	
TYPE OF INTEGRATION	ADAMS-MOULTON		RUNGE-KUTTA		ADAMS-MOULTON		ADAMS-MOULTON	
USE OF DOUBLE PRECISION ARITHMETIC	TOTAL		ACCUMULATION		ACCUMULATION		TOTAL	
INTEGRATION TIMES IN MILLISECONDS	415		54		45		50	
NUMBER OF INTEGRATION STEPS	275		9		30		30	
NUMBER OF DERIVATIVE EVALUATIONS	561		144		99		99	
STORAGE USED	1777 ₈		1510 ₈		2234 ₈		1777 ₈	
MAXIMUM ERRORS	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E	A B S O L U T E	R E L A T I V E
E1	-7.4E-5	-1.6E-10	-1.7E-1	-5.3E-8	2.0E0	1.16E-6	-2.0E-0	-1.1E-6
E2	1.2E-4	-5.9E-10	4.0E-1	7.5E-7	6.2E0	1.2E-5	6.2E-0	1.2E-5
E3	2.6E-7	-2.2E-11	-1.7E-3	-6.8E-8	2.3E-2	9.5E-7	2.3E-2	9.5E-7
E4	5.9E-7	6.0E-4	-7.0E-5	-8.7E-2	3.2E-2	9.7E-1	3.2E-2	9.7E-1

TABLE 6
 EARTH-JUPITER TRANSFER PROBLEM
 VARIABLE STEP MODE
 INITIAL STEP SIZE = 1.0
 $\epsilon = 10^{-5}$

INTEGRATION ROUTINE	ABS ERROR TEST AM		RK5		AM		(RELATIVE TEST) AM	
TYPE OF INTEGRATION	ADAMS-MOULTON		RUNGE-KUTTA		ADAMS-MOULTON		ADAMS-MOULTON	
USE OF DOUBLE PRECISION ARITHMETIC	TOTAL		ACCUMULATION		ACCUMULATION		TOTAL	
INTEGRATION TIMES IN MILLISECONDS	1207		2054		1220		1482	
NUMBER OF INTEGRATION STEPS	265		103		303		303	
NUMBER OF DERIVATIVE EVALUATIONS	626		1809		772		772	
STORAGE USED	1777 ₈		1510 ₈		2234 ₈		1777 ₈	
MAXIMUM ERRORS	A B S O L U T I O N	R E L A T I V E	A B S O L U T I O N	R E L A T I V E	A B S O L U T I O N	R E L A T I V E	A B S O L U T I O N	R E L A T I V E
E1	-8.7E-8	-1.0E-4	1.8E-9	-2.2E-6	-2.1E-7	-2.4E-4	-2.1E-7	-2.4E-4
E2	1.5E-7	-1.3E-4	2.2E-9	-2.0E-6	-6.3E-7	5.2E-4	-6.7E-7	5.5E-4
E3	5.9E-6	5.4E-4	2.0E-9	-1.1E-6	2.7E-5	2.5E-3	2.7E-5	2.5E-3
E4	-1.2E-7	-1.2E-7	-7.9E-7	-7.9E-7	1.2E-6	1.2E-6	1.2E-6	1.2E-6

TABLE 7*

BRACHISTOCHRONE PROBLEM

		ALGORITHM I	ALGORITHM II ESTIMATES	
	ABSOLUTE ERRORS	SERIES	MIN NORM	LEAST SQ
E1	.107 E-05	.137 E-05	N/A	N/A
E2	.477 E-05	.544 E-05	.3 E-9	.118 E-04
E3	.284 E-07	.317 E-07	.58 E-7	.58 E-07
		ALGORITHM I ESTIMATES		
$\epsilon = 10^{-5}$		SERIES	EULER'S	RK4
E1	.66 E-05	.676 E-05	.9 E-05	.62 E-05
E2	.28 E-04	.324 E-04	.3 E-04	.32 E-04
E3	.15 E-06	.149 E-06	.16 E-06	.15 E-06
$\epsilon = 10^{-6}$		SERIES	EULER'S	RK4
E1	.83 E-06	.15 E-05	.17 E-05	.13 E-05
E2	.52 E-05	.57 E-5	.53 E-05	.53 E-05
E3	.27 E-07	.35 E-07	.33 E-07	.32 E-07

*ABSOLUTE VALUES OF ERRORS AND ESTIMATES ARE USED IN TABLES 7 THROUGH 11.

TABLE 8

"FLAT EARTH" TRAJECTORY PROBLEM

$\epsilon = 10^{-5}$	ABSOLUTE ERRORS	ALGORITHM I ESTIMATES			ALGORITHM II
		SERIES	EULER'S	RK4	
E1	.17 E+01	.198 E+01	.151 E+01	.198 E+01	ESTIMATES USING GENERALIZED HAMILTONIAN
E2	.12 E+01	.10 E+01	.13 E+01	.10 E+01	
E3	.34 E-02	.39 E-02	.27 E-02	.39 E-02	
E4	.65 E-02	.66 E-02	.65 E-02	.66 E-02	
$\epsilon = 10^{-6}$					
E1	.23 E-00	.143 E 00	.14 E 00	.14 E 00	N/A
E2	.11 E 00	.73 E-01	.81 E-01	.73 E-01	N/A
E3	.35 E-03	.28 E-03	.31 E-03	.31 E-03	N/A
E4	.32 E-03	.46 E-03	.46 E-03	.46 E-03	.303 E-03

TABLE 9*

EARTH-JUPITER TRANSFER PROBLEM

 $E = 10^{-5}$

	ALGORITHM I ESTIMATES			FIXED AM
	EULER'S	SERIES	RK4	
1	.27 E-01	.36 E-01	.69 E-02	.91 E-03
2	.19 E-01	.23 E-01	.12 E-02	.51 E-02
3	.14 E-02	.16 E-02	.27 E-04	.65 E-05
4	.1 E-03	.13 E-03	.25 E-05	.36 E-05
5	.79 E-04	.92 E-04	.54 E-05	.21 E-05
6	.48 E-05	.52 E-05	.48 E-07	.7 E-08
7	.6 E-08	.6 E-08	.6 E-08	.31 E-06
8	.65 E-04	.11 E-03	.11 E-04	.11 E-04
9	.69 E-04	.1 E-03	.86 E-05	.77 E-05
10	.13 E-05	.37 E-06	.32 E-06	.26 E-07
11	.52 E-01	.73 E-01	.34 E-02	.42 E-02
12	.13 E-02	.23 E-02	.27 E-02	.2 E-02
13	.28 E-03	.56 E-03	.7 E-04	.6 E-04
14	.62 E-03	.82 E-03	.1 E-04	.14 E-02

*FOURTH COLUMN OBTAINED USING TOTAL DOUBLE PRECISION

TABLE 10*

EARTH-JUPITER TRANSFER PROBLEM

E = 10^{-6}

	ALGORITHM I ESTIMATES			FIXED	FIXED
	EULERS	SERIES	RK4	AM	RK
1	.27 E-04.	.5 E-04	.31 E-04	.74 E-04	.35 E-05
2	.9 E-04	.15 E-03	.13 E-03	.15 E-03	.61 E-04
3	.26 E-05	.68 E-05	.57 E-05	.85 E-05	.54 E-05
4	.69 E-07	.19 E-06	.10 E-06	.18 E-06	.32 E-06
5	.46 E-06	.69 E-06	.58 E-06	.52 E-06	.19 E-06
6	.57 E-08	.2 E-07	.17 E-07	.33 E-07	.98 E-08
7	.4 E-08	.4 E-08	.4 E-08	.23 E-06	.21 E-06
8	.58 E-06	.41 E-06	.48 E-06	.75 E-06	.5 E-06
9	.75 E-06	.49 E-06	.32 E-06	.77 E-07	.16 E-06
10	.24 E-07	.26 E-07	.22 E-07	.26 E-07	.14 E-07
11	.16 E-03	.16 E-04	.70 E-04	.1 E-03	.12 E-03
12	.25 E-03	.23 E-03	.17 E-03	.11 E-03	.64 E-04
13	.53 E-05	.34 E-05	.29 E-05	.46 E-06	.12 E-05
14	.71 E-04	.78 E-04	.73 E-04	.2 E-02	.21 E-02

*FOURTH COLUMN OBTAINED USING TOTAL DOUBLE PRECISION

TABLE 11

EARTH-JUPITER TRANSFER PROBLEM

	I	II	III	ALGORITHM II ESTIMATES			
				IV	V	VI	
1	.50 E-04	.9 E-04	.15 E-04	.96 E-05	.12 E-05	.5 E-07	
2	.15 E-03	.15 E-03	.58 E-04	.72 E-06	.11 E-05	.11 E-06	
3	.68 E-05	.85 E-05	.32 E-05	.87 E-06	.23 E-07	.23 E-05	
4	.19 E-06	.22 E-06	.21 E-07	.88 E-08	.47 E-08	.16 E-04	
5	.69 E-06	.5 E-06	.17 E-06	.97 E-06	.43 E-08	.2 E-04	
6	.20 E-07	.33 E-07	.1 E-07	.11 E-07	.84 E-10	.5 E-06	
7	.40 E-08	.23 E-06	.22 E-06	.13 E-06	.1 E-02	.19 E-06	
8	.41 E-06	.6 E-06	.39 E-06	.19 E-05	.88 E-08	.15 E-04	
9	.49 E-06	.12 E-07	.1 E-07	.31 E-06	.14 E-07	.42 E-05	
10	.26 E-07	.2 E-07	.11 E-07	.31 E-08	.2 E-09	.58 E-07	
11	.16 E-04	.7 E-04	.3 E-04	.12 E-06	.12 E-04	.5 E-08	
12	.23 E-03	.11 E-03	.6 E-04	.89 E-06	.63 E-05	.59 E-07	
13	.34 E-05	.22 E-06	.5 E-06	.12 E-06	.96 E-07	.23 E-05	
14	.78 E-04	.2 E-02	.22 E-02	.20 E-02	.19 E-02	.7 E-07	
	.29 E-03	.20 E-02	.22 E-02	.20 E-02	.21 E-02	.30 E-04	Δ

I. LINEAR ESTIMATOR USING SERIES SOLUTION (ALG I).

II. DIFFERENCED TO TOTAL DOUBLE PRECISION AM FIXED H=1.

III. DIFFERENCED TO FOURTH-ORDER RK FIXED H=1.

IV. WEIGHTED MINIMUM NORM ($w_i = 1/|y_i|$)

V. MINIMUM NORM LEAST SQ (4 EQ, 1 VAR) (5 ITERATIONS)

VI. MIN NORM (14 VAR, 4 EQ)

TABLE 12

TIME COMPARISON DATA*

ALGORITHM I

	SOLUTIONS & ESTIMATES			ESTIMATES ONLY			SOLUTION ONLY
	EULER'S	SERIES	RK4	EULER'S	SERIES	RK4	
$\epsilon = 10^{-5}$							
BRACHISTOCHRONE	45	53	62	5	13	22	40
FLAT EARTH	39	48	53	3	12	17	36
E-JT	1254	1798	2156	300	844	1202	954
$\epsilon = 10^{-6}$							
BRACHISTOCHRONE	70	81	95	9	20	24	61
FLAT EARTH	56	69	73	4	17	21	52
E-JT	1893	2745	3252	454	1306	1813	1439

ALGORITHM II

	LEAST SQ	MIN NORM	WEIGHTED MIN NORM
E-JT	126	84	1188
FLAT EARTH	5		
BRACHISTOCHRONE	12		

*ALL TIMES IN MILLISECONDS