K. R. Brown, Jr.
G. W. Johnson

Rapid Computation of Optimal Trajectories†

Abstract: A generalized "indirect" method of solving two-point boundary value problems is discussed in application to the problem of computing optimal trajectories in a vacuum. Improved numerical techniques make the method extremely fast when a good initial estimate of the solution is available, but it also converges, more slowly, from initial estimates that are far from the solution. Transversality conditions are combined with final-value constraints enabling the method to solve directly problems defined by constraints on arbitrary functions of final state.

Section 1 defines the differential equations and initial and terminal conditions for optimal rocket trajectories in a central gravitational field. The differential equations are given a particularly simple form and transversality conditions are formulated analytically for typical orbital injection missions. Section 2 defines efficient numerical procedures for solving the initial value problem of optimal trajectories and so reduces the boundary value problem to a multidimensional zero-finding problem. Section 3 describes the generalized version of Newton's method used to solve this multidimensional zero-finding problem. Section 4 summarizes the results of an IBM 7094 implementation, giving execution times and convergence properties.

Introduction

It is well known that the problem of optimal trajectory determination comes down ultimately to a two-point boundary-value problem in ordinary differential equations.[1,2] Normally there is no question of obtaining in closed form even a part of the solution to such a problem. Sometimes ingenious alterations to the differential equations can lead to an approximately equivalent problem that is partly tractable so that only a simple iterative process is required,[3–5] but with such an approach, accuracy and flexibility are definitely limited. The speed needed for real-time applications has been a primary motivation for efforts in that direction.

General approaches to two-point boundary-value problems, in principle capable of achieving arbitrary accuracy, can be thought of under two main headings: "direct" methods and "indirect" methods.[6] Roughly speaking, direct methods search over the space of functions satisfying the boundary value requirements for a function satisfying the differential equations; indirect methods search over the space of functions satisfying the differential equations for a function satisfying the boundary-value requirements. This partitioning of methods is not exhaustive; several hybrid methods have been proposed in which successive iterates satisfy neither the boundary conditions nor the differential equations. However, direct and indirect methods do represent the principal approaches to numerical solution of two-point boundary-value problems.*

Different factors may favor direct or indirect methods in different applications. There may, for example, be a trade-off between the ultimate speed of convergence of an indirect method and the reliability of convergence of a direct method. But such factors usually have more to do with the special features of each individual method than with whether its approach is basically direct or indirect.

There is one application, however—that of real-time guidance—that seems to offer a clear choice between direct and indirect methods as such. Until recently, general methods, both direct and indirect, were considered too time consuming for real-time use, so the question of their relative suitability did not arise. But once raised, the question is easily answered in favor of indirect methods, for successive control policies generated in real time can be viewed as successive iterates in an indirect method. Each tentative control policy generated by a real-time guidance algorithm needs to be replaced by its successor, not

† The work reported in this paper was performed under NASA contract NAS 8-14000.

* A different and less precise distinction between direct and indirect methods is sometimes based on whether the method "directly" attempts to reduce the cost functional or "indirectly" finds the optimum by incorporating derived necessary conditions.

because it fails to satisfy the differential equations, but because the initial (and perhaps also terminal) conditions upon which it was based have been revised.

Even in an adaptive system that revises in real time some of the parameters in the equation of motion, the revision can easily be translated into a change in terminal conditions so that an indirect method is still appropriate for obtaining a revised control policy. But there is no corresponding way of translating revised initial and terminal conditions into revised equations of motion to enable a direct method to handle the revision.

The indirect method reported here grew out of the development of a flexible real-time optimal guidance scheme.[7,8] But it has proved very valuable as a general-purpose tool in non-real-time optimal trajectory determination, particularly when no good initial estimate of the solution is available.

Indirect methods have been applied to the rocket steering problem with varying degrees of success and have received considerable attention in the published literature. A recent survey paper by Paiewonsky[9] contains a comprehensive summary of prior work. The algorithm described in this paper for the computation of non-atmospheric trajectories utilizes the same theoretical principles embodied in these previous efforts in combination with improved numerical techniques to yield a substantial improvement in both speed and flexibility. The principle sources of improvement include (1) an integration algorithm with adaptive step size that is tailored to the combination of optimal trajectory equations and associated initial value equations of the initial value problem, (2) the use of terminal constraint functions (including transversality conditions) which are relatively smooth as functions of the free initial variables, and (3) a generalization of Newton's method that extends the region of convergence of the boundary value search.

1. Optimal rocket steering as a boundary-value problem

In this section, typical optimal rocket steering problems in a central gravitational field are formulated as two-point boundary-value problems in a way that encourages insight and facilitates numerical solution. This formulation is in agreement with that used by others (e.g., Refs. 10 and 11), but is developed here for completeness of the presentation.

Assume Newtonian two-body, point-mass motion in a vacuum. Using a Cartesian coordinate system, the unforced equations of motion can be written in vector form as a second-order system

$$\ddot{\mathbf{r}} = \frac{-\mu \mathbf{r}}{r^3}, \tag{1}$$

where μ is a gravitational constant and $\mathbf{r}$ is the position vector relative to an origin at the attracting mass. Assume further that thrust furnishes an added acceleration $\mathbf{a}(t)$ whose magnitude $a(t)$ is prescribed and whose direction cosines $\mathbf{u}/u$ are determined by control $\mathbf{u}$. Then the equations of powered flight are

$$\ddot{\mathbf{r}} = \frac{-\mu \mathbf{r}}{r^3} + a(t)\,\frac{\mathbf{u}}{u}. \tag{2}$$

In a Lagrange formulation the goal of the variational problem is to choose the control history $\mathbf{u}(t)$ to minimize the performance functional

$$J[\mathbf{u}] = \int_{t_0}^{t_f} L(\mathbf{x}, \mathbf{u}, t)\, dt, \tag{3}$$

where L is the instantaneous rate of cost function and may depend on state $\mathbf{x}$ as well as control and time. In general there may be constraints on the initial and terminal state, and the equations of motion furnish the dynamic constraints

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t). \tag{4}$$

Pontriagin's maximum principle requires that at every instant the optimum control $\mathbf{u}^o$ be chosen so that

$$\mathbf{u}^o \text{ minimizes } L(\mathbf{x}, \mathbf{u}, t) + \mathbf{p}^T\mathbf{f}(\mathbf{x}, \mathbf{u}, t), \tag{5}$$

where $\mathbf{p}^T$ is costate, the gradient of the remaining cost of the mission with respect to the current state, and satisfies the Euler-Lagrange differential equations

$$\dot{\mathbf{p}}^T = -\mathbf{p}^T \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, t)}{\partial \mathbf{x}} - \frac{\partial L(\mathbf{x}, \mathbf{u}, t)}{\partial \mathbf{x}}. \tag{6}$$

In the present case, state $\mathbf{x}$ is a six-vector made up of two three-vectors, position $\mathbf{r}$ and velocity $\mathbf{v} = \dot{\mathbf{r}}$, and corresponding costate $\mathbf{p}$ can be thought of as a combination of two three-vectors $\mathbf{q}$ and $\mathbf{s}$. Moreover, the rate of cost L is the rate of propellant expenditure which is assumed to be independent of control and state, so the instantaneous minimization of (5) becomes

$$\mathbf{u}^o \text{ minimizes } L(t) + \mathbf{q}^T\mathbf{v} + \mathbf{s}^T\left[-\frac{\mu \mathbf{r}}{r^3} + a(t)\,\frac{\mathbf{u}}{u}\right] \tag{7}$$

and the costate equations (6) become

$$\dot{\mathbf{s}} = -\mathbf{q}; \quad \dot{\mathbf{q}} = \mathbf{r}(-3\mu r^{-5}\mathbf{r}^T\mathbf{s}) + \mathbf{s}(\mu r^{-3}). \tag{8}$$

Now it is easy to verify that condition (7) is satisfied if and only if the vector $\mathbf{u}$ points in the direction opposite to that of $\mathbf{s}$, that is, if and only if $\mathbf{u} = -k\mathbf{s}$ for some positive scalar k. Moreover, in view of (2), it makes no difference which scalar k we choose, so for convenience let us choose $k = 1$. Hence,

$$\mathbf{u} = -\mathbf{s}. \tag{9}$$

Then, expressing (8) as a second-order system in $\mathbf{s}$ and substituting (9), we obtain a simple differential equation for optimal control

$$\ddot{\mathbf{u}} = \mathbf{r}(3\mu r^{-5}\mathbf{r}^T\mathbf{u}) + \mathbf{u}(-\mu r^{-3}). \tag{10}$$

Together, Eqs. (2) and (10) constitute a complete set of differential equations for state and control.

Given an initial state $\mathbf{x}_0$ at time t_0 and a thrust magnitude history $a(t)$, a unique maneuver satisfying (2) and (10) is determined by selecting values for initial control $\mathbf{u}(t_0)$, control rate $\dot{\mathbf{u}}(t_0)$ and cut-off t_f. In fact, $\mathbf{u}(t_0)$ and $\dot{\mathbf{u}}(t_0)$ need only be defined to within a positive multiplicative constant k. For, given functions $\mathbf{u}(t)$ and $\mathbf{r}(t)$ satisfying (2) and (10), $\mathbf{u}(t)$ can obviously be replaced by $\mathbf{u}^*(t) = k\mathbf{u}(t)$ without disturbing either (2) or (10), provided only that $k > 0$.

Thus, given an initial state $\mathbf{x}_0$ at t_0, all time-optimal rocket maneuvers having a given thrust acceleration magnitude history $a(t)$ can be specified by choosing t_f and the direction of the vector $\mathbf{p} = (\dot{u}_1, \dot{u}_2, \dot{u}_3, -u_1, -u_2, -u_3)^T$ at t_0. Therefore, there are at most six degrees of freedom (one in the choice of t_f and five in the choice of the direction of $\mathbf{p}_0$) in the determination of an optimal maneuver. This corresponds exactly with the expected six degrees of freedom in final state $\mathbf{x}_f$. As a result, the necessary condition (10) of optimality turns out to be locally sufficient, barring singularities, for missions involving a complete set of six constraints on final state. Although multiple solutions of the boundary value problem are theoretically possible, no nonoptimal solutions have arisen in the orbital transfer missions we have considered.

Accordingly, we can state the problem of optimal steering to a prescribed final state as a boundary value problem as follows. Given an initial state $\mathbf{r}_0$, $\mathbf{v}_0$, a final state $\mathbf{r}_f$, $\mathbf{v}_f$, and a positive-valued function $a(t)$, find a scalar $t_f > t_0$ and find functions $\mathbf{u}(t)$ and $\mathbf{r}(t)$ satisfying (2) and (10) such that $\mathbf{r}(t_0) = \mathbf{r}_0$, $\dot{\mathbf{r}}(t_0) = \mathbf{v}_0$ and $\mathbf{r}(t_f) = \mathbf{r}_f$, $\dot{\mathbf{r}}(t_f) = \mathbf{v}_f$.

Similarly, the problem of optimal steering to orbital rendezvous involves a complete set of six constraints on final state, so we can express the problem of optimal steering to rendezvous with an orbiting body as a boundary value problem as follows. Given an initial state $\mathbf{r}_0$, $\mathbf{v}_0$, a positive-valued function $a(t)$, and a function $\mathbf{r}_b(t)$ obeying the unforced equation (1) find a scalar $t_f > t_0$ and find functions $\mathbf{u}(t)$ and $\mathbf{r}(t)$ satisfying (2) and (10) such that $\mathbf{r}(t_0) = \mathbf{r}_0$, $\dot{\mathbf{r}}(t_0) = \mathbf{v}_0$ and $\mathbf{r}(t_f) = \mathbf{r}_b(t_f)$, $\dot{\mathbf{r}}(t_f) = \dot{\mathbf{r}}_b(t_f)$.

Certain functions of state $g(\mathbf{x})$ are constant in time whenever the state obeys the unforced equation (1). These functions are orbital constants and include the geometrical parameters defining an orbit such as the size and shape of the conic, the orbital plane, and the orientation of the conic within its plane. Such a function, if it reaches a desired value g_d at thrust termination t_f, will maintain that desired value thereafter. Parking orbit injection missions normally fall in the category of missions in which all constraints are on such functions of final state. For purposes of this paper we will consider an orbital injection mission to be defined as one in which mission requirements consist of equality constraints on functions of final state which are orbital constants. Since the constraints defining orbital injection missions are constraints on functions of state which are constant in time, there can be at most five linearly independent such constraints. Accordingly, the mission definition cannot by itself provide the full complement of six constraints needed to select an optimal maneuver. If there are k constraints on final state ($k < 6$), then the best (in the sense of the performance index) final state $\mathbf{x}_f$ must be chosen from the $(6 - k)$ degrees of freedom available in final state.

It is known that an optimal maneuver to the best final state $\mathbf{x}_f^0$ in a manifold of final states $\mathbf{x}_f$ satisfying k constraints $g_i(\mathbf{x}_f) = g_{id}$, $i = 1, \ldots, k$ satisfies the following transversality conditions:

$$\mathbf{p}_f^T\mathbf{a}_i = 0, \qquad i = k + 1, \cdots, 6, \tag{11}$$

where $\mathbf{p}_f$ is the limit, as t approaches t_f-, of the $\mathbf{p}$ vector and the $\mathbf{a}_i$ are a set of $(6 - k)$ independent 6-vectors lying in the hyperplane tangent to the terminal manifold. Thus, in order to express a specific orbital injection problem as a boundary value problem, it is necessary to choose specific functions $g_i(\mathbf{x}_f)$ to be constrained and determine the corresponding vectors $\mathbf{a}_i$ needed in the transversality condition.

The most fundamental orbital injection problem is the five-constraint problem. This is the problem in which the mission definition places constraints on five independent functions $g_i(\mathbf{x}_f)$ of final state. There is considerable freedom in the selection of the forms of the five functions individually. However, since there can be at most five independent functions of state which are constant in time, any selection of five such functions to be constrained has the effect of constraining all such functions. Therefore it is valid to speak of *the* five-constraint orbital injection problem.

For simplicity, we select as the five functions g_i to be constrained the components h_1, h_2, h_3, e_1 and e_2 of the following vectors:

$$\begin{aligned} \mathbf{h}(\mathbf{x}) &= \mathbf{r} \times \mathbf{v} \\ \mathbf{e}(\mathbf{x}) &= -\left(\frac{\mathbf{r}}{r} + \frac{\mathbf{h} \times \mathbf{v}}{\mu}\right). \end{aligned} \tag{12}$$

The vector $\mathbf{h}$ is the angular momentum of the orbiting body and $\mathbf{e}$ is a vector whose magnitude is the orbital eccentricity and whose direction is the direction of the pericenter of the orbit. Except in the singular case $h_3 = 0$

which can be accommodated by a redefinition of coordinates, the five components mentioned are linearly independent functions of state which are constant in time in the absence of thrust.

In order to state the five-constraint orbital injection problem as a boundary value problem, we need to express a vector $\mathbf{a}_6$ for use in the transversality condition (11). The sole requirement on $\mathbf{a}_6$ is that it lie in the tangent hyperplane to the terminal manifold, which in this case is a 1-dimensional manifold since $k = 5$. This requirement is equivalent to the following: $\mathbf{a}_6$ is orthogonal (in 6-space) to the gradient with respect to $\mathbf{x}$ of g_i, for $i = 1, \ldots, 5$. By definition, orbital injection constraint functions are functions of state whose time derivatives are zero in the absence of thrust, thus we have

$$\frac{d}{dt}[g_i(x)] = \left[\frac{\partial g_i}{\partial \mathbf{x}}\right]\dot{\mathbf{x}} = 0, \quad i = 1, \cdots 5, \tag{13}$$

where $\dot{\mathbf{x}}$, which we therefore adopt as $\mathbf{a}_6$, takes on the unforced value given by Eq. (1). Hence,

$$\mathbf{a}_6 = \left(v_1, v_2, v_3, -\frac{\mu r_1}{r^3}, -\frac{\mu r_2}{r^3}, -\frac{\mu r_3}{r^3}\right)^T. \tag{14}$$

We can now state the five-constraint orbital injection problem as a boundary value problem. Given an initial state $\mathbf{r}_0$, $\mathbf{v}_0$, a positive-valued function $a(t)$, and a set of five desired values g_{id}, $i = 1, \ldots, 5$, find a scalar $t_f > t_0$ and find functions $\mathbf{u}(t)$ and $\mathbf{r}(t)$ such that

(a) $\mathbf{r}(t_0) = \mathbf{r}_0$, $\dot{\mathbf{r}}(t_0) = \mathbf{v}_0$,
(b) $\mathbf{u}(t)$ and $\mathbf{r}(t)$ satisfy Eqs. (2) and (10),
(c) at $\mathbf{x} = \mathbf{x}(t_f)$, the first five components of the $\mathbf{h}$ and $\mathbf{e}$ vectors defined by (12) are equal to g_{id}, $i = 1, \ldots, 5$, and
(d) at $t = t_f$, Eq. (11) holds for $\mathbf{a}_6$ as defined by Eq. (14), i.e.,

$$\dot{\mathbf{u}}(t_f)^T\dot{\mathbf{r}}(t_f) + \frac{\mu}{r^3}\mathbf{u}(t_f)^T\mathbf{r}(t_f) = 0. \tag{15}$$

There is also a four-constraint orbital injection problem which is of particular interest. This problem is like the five-constraint problem except that the orientation of the conic within its orbital plane is allowed to be arbitrary. For this purpose, we have selected as the four constraint functions, the three components of the $\mathbf{h}$ vector defined by Eq. (12) and the orbital energy defined as follows

$$c = \frac{v^2}{2} - \frac{\mu}{r}. \tag{16}$$

Since the terminal manifold of the five-constraint problem is included in the terminal manifold of the present four-constraint problem, the vector $\mathbf{a}_6$ defined by Eq. (14) can be used here. It remains to define a vector $\mathbf{a}_5$ which is independent of $\mathbf{a}_6$ and which also lies in the terminal manifold of the present four-constraint problem. As before, the requirement that $\mathbf{a}_5$ lie in the terminal manifold is readily seen to be equivalent to the requirement that $\mathbf{a}_5$ be orthogonal to the gradient of c and to the gradients of the components of $\mathbf{h}$. It is not difficult to verify that the vector defined by

$$\mathbf{a}_5(\mathbf{x}) = \begin{pmatrix}\mathbf{r} \times \mathbf{h} \\ \mathbf{v} \times \mathbf{h}\end{pmatrix} \tag{17}$$

satisfies these requirements and is, moreover, independent of the vector $\mathbf{a}_6$ defined previously, provided $e \neq 0$.

We are now prepared to state the four-constraint orbital injection problem as a boundary value problem. Given an initial state $\mathbf{r}_0$, $\mathbf{v}_0$, a positive-valued function $a(t)$, and desired values g_{id}, $i = 1, \ldots, 4$, for $\mathbf{h}$ and c, find a scalar $t_f > t_0$ and find functions $\mathbf{u}(t)$ and $\mathbf{r}(t)$ such that

(a) $\mathbf{r}(t_0) = \mathbf{r}_0$, $\dot{\mathbf{r}}(t_0) = \mathbf{v}_0$,
(b) $\mathbf{u}(t)$ and $\mathbf{r}(t)$ satisfy Eqs. (2) and (10),
(c) at t_f, the components of $\mathbf{h}$ and c equal the g_{id}, $i = 1, \ldots, 4$, and
(d) at $t = t_f$, Eq. (11) is satisfied where $\mathbf{a}_5$ and $\mathbf{a}_6$ are defined by (14) and (17) i.e., Eqs. (15) and (18) hold:

$$\dot{\mathbf{u}}(t_f)^T[\mathbf{r}(t_f) \times \mathbf{h}(t_f)] - \mathbf{u}(t_f)^T[\dot{\mathbf{r}}(t_f) \times \mathbf{h}(t_f)] = 0. \tag{18}$$

Additional transversality vectors have been formulated analytically corresponding to 2- and 3-constraint orbital injection missions and are reported on in Ref. 8.

2. Numerical solution of the initial value problem

In Section 1, several optimal steering problems were translated into equivalent boundary value problems. In each case, the requirements on the solution functions $\mathbf{r}(t)$ and $\mathbf{u}(t)$, or equivalently on $\mathbf{x}(t)$ and $\mathbf{p}(t)$, took the form:

(a) $\mathbf{x}(t_0) = \mathbf{x}_0$,
(b) $\mathbf{x}(t)$ and $\mathbf{p}(t)$ satisfy (2) and (10),
(c) $\mathbf{x}(t_f)$ and $\mathbf{p}(t_f)$ satisfy 6 conditions consisting of k boundary value constraints and $6 - k$ transversality conditions.

In the present section, these boundary value problems are reduced to multi-dimensional zero-finding problems by a numerical integration scheme tailored to special features of the initial value problems.

More specifically, let the unknown initial vector $\mathbf{p}(t_0)$ and the unknown final time t_f be combined into a seven-vector $\mathbf{y}$ of independent variables, and let the six constrained functions of $\mathbf{x}_f$ and $\mathbf{p}_f$ together with the (arbitrarily

constrained) magnitude of $\mathbf{p}(t_0)$ be considered as a seven-vector $\mathbf{z}$ of dependent variables. Then the boundary value problem is this: find the value of the independent variable vector that causes the dependent variable vector to take on its desired value. Or, if the dependence of $\mathbf{z}$ on $\mathbf{y}$ is represented by

$$\mathbf{z} = \mathbf{f}(\mathbf{y}), \tag{19}$$

find a value of $\mathbf{y}$ which is a vector zero of

$$\mathbf{z}^* - \mathbf{f}(\mathbf{y}), \tag{20}$$

where $\mathbf{z}^*$ is the desired value of $\mathbf{z}$.

The algorithm of Section 3 will accomplish this if provided with a method of computing $\mathbf{f}(\mathbf{y})$ and the Jacobian matrix $[\partial \mathbf{f}(\mathbf{y})/\partial \mathbf{y}]$ of partial derivatives of $\mathbf{f}$ with respect to $\mathbf{y}$.

The problem of computing $\mathbf{f}(\mathbf{y})$ is plainly an initial value problem. For, given $\mathbf{x}_0$, t_0, and $\mathbf{y}$, that is, given $\mathbf{x}_0$, t_0, $\mathbf{p}_0$, and t_f, it is an initial value problem to solve (2) and (10) for the final values of $\mathbf{x}_f$ and $\mathbf{p}_f$ from which in turn the components of $\mathbf{z} = \mathbf{f}(\mathbf{y})$ can be computed.

To obtain also the matrix $[\partial f(\mathbf{y})/\partial \mathbf{y}]$, it will be sufficient, by the chain rule, to compute the matrix $[\partial(\mathbf{x}_f, \mathbf{p}_f)/\partial(\mathbf{p}_0, t_f)]$ since $\mathbf{z}$ consists of functions of $\mathbf{x}_f$ and $\mathbf{p}_f$ and $\mathbf{y}$ is made up of $\mathbf{p}_0$ and t_f. A part, $[\partial(\mathbf{x}_f, \mathbf{p}_f)/\partial t_f]$, of the latter matrix can be computed at once since its elements are simply those of $\dot{\mathbf{x}}_f$ and $\dot{\mathbf{p}}_f$ available from (2) and (10).

The rest, $[\partial(\mathbf{x}_f, \mathbf{p}_f)/\partial \mathbf{p}_0]$, can be obtained as the solution to an additional initial value problem, for differential equations and initial conditions can be derived for the time-varying matrix $[\partial \mathbf{x}(t), \mathbf{p}(t)/\partial \mathbf{p}_0]$ whose value at t_f is the desired matrix.

In general, given a first-order system of differential equations

$$\dot{\mathbf{w}} = \dot{\mathbf{w}}(\mathbf{w}), \tag{21}$$

we can define a related set of matrix differential equations

$$\dot{M} = \left[\frac{\partial \dot{\mathbf{w}}}{\partial \mathbf{w}}\right] M, \tag{22}$$

which are satisfied by the matrix function $M(t)$, where

$$M(t) = \left[\frac{\partial \mathbf{w}(t)}{\partial \mathbf{w}(t_0)}\right]. \tag{23}$$

The appropriate initial condition for the desired matrix M is

$$M(t_0) = I. \tag{24}$$

In the present case, let $\mathbf{w}$ be the 12-vector consisting of the six components of state $\mathbf{x}$ and the six components of the vector $\mathbf{p}$. Then the basic differential equation (21) corresponds to our equations (2) and (10). Since we are interested only in the partial derivatives of $\mathbf{x}_f$ and $\mathbf{p}_f$ with respect to $\mathbf{p}(t_0)$, we need only the right half of the M matrix appearing in Eqs. (22)-(24). More explicitly, we desire a 12×6 matrix Z defined by

$$Z(t) = \left[\frac{\partial \mathbf{w}(t)}{\partial \mathbf{p}(t_0)}\right]. \tag{25}$$

Such a matrix will satisfy

$$\dot{Z} = \left[\frac{\partial \dot{\mathbf{w}}}{\partial \mathbf{w}}\right] Z \tag{26}$$

and

$$Z(t_0) = \begin{pmatrix} O_6 \\ I_6 \end{pmatrix}, \tag{27}$$

where O_6 is the 6×6 zero submatrix and I_6 is the 6×6 identity submatrix.

Carrying out by means of (2) and (10), the partial differentiation needed to obtain explicit expressions for the components of the matrix $[\partial \dot{\mathbf{w}}/\partial \mathbf{w}]$, it turns out that (26) can be expressed in terms of 3×3 submatrices thus:

$$\begin{bmatrix} \dot{Z}_{11} & \dot{Z}_{12} \\ \dot{Z}_{21} & \dot{Z}_{22} \\ \dot{Z}_{31} & \dot{Z}_{32} \\ \dot{Z}_{41} & \dot{Z}_{42} \end{bmatrix} = \begin{bmatrix} O_3 & I_3 & O_3 & O_3 \\ \dfrac{\partial \ddot{\mathbf{r}}}{\partial \mathbf{r}} & O_3 & O_3 & -\dfrac{\partial \ddot{\mathbf{r}}}{\partial \mathbf{u}} \\ \dfrac{\partial \ddot{\mathbf{u}}}{\partial \mathbf{r}} & O_3 & O_3 & -\dfrac{\partial \ddot{\mathbf{u}}}{\partial \mathbf{u}} \\ O_3 & O_3 & -I_3 & O_3 \end{bmatrix} \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \\ Z_{31} & Z_{32} \\ Z_{41} & Z_{42} \end{bmatrix}. \tag{28}$$

Hence, if we make the following definitions

$$W = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{41} & Z_{42} \end{bmatrix},$$

$$B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \ddot{\mathbf{r}}}{\partial \mathbf{r}} & -\dfrac{\partial \ddot{\mathbf{r}}}{\partial \mathbf{u}} \\ -\dfrac{\partial \ddot{\mathbf{u}}}{\partial \mathbf{r}} & \dfrac{\partial \ddot{\mathbf{u}}}{\partial \mathbf{u}} \end{bmatrix}, \tag{29}$$

we obtain from (28) the second order 6×6 matrix equation,

$$\ddot{W} = BW,$$

where

$$\dot{W} = \begin{bmatrix} \dot{W}_{11} & \dot{W}_{12} \\ \dot{W}_{21} & \dot{W}_{22} \end{bmatrix} = \begin{bmatrix} Z_{12} & Z_{22} \\ -Z_{31} & -Z_{32} \end{bmatrix} \tag{30}$$

and from (27) the initial conditions:

$$W(t_0) = \begin{bmatrix} O_3 & O_3 \\ O_3 & I_3 \end{bmatrix}; \qquad \dot{W}(t_0) = \begin{bmatrix} O_3 & O_3 \\ -I_3 & O_3 \end{bmatrix}. \tag{31}$$

Thus, in order to obtain the desired partial derivatives of $\mathbf{x}_f$ and $\mathbf{p}_f$ with respect to $\mathbf{p}(t_0)$, it is sufficient to perform a numerical integration of the second order system (30)

with the initial condition (31). A basic requirement is that the nontrivial 3×3 submatrices $\partial\ddot{\mathbf{r}}/\partial\mathbf{r}$, $\partial\ddot{\mathbf{r}}/\partial\mathbf{u}$, $\partial\ddot{\mathbf{u}}/\partial\mathbf{r}$, and $\partial\ddot{\mathbf{u}}/\partial\mathbf{u}$ of the matrix B which appears as coefficient in (30), be available as needed. Since these submatrices are readily computable functions of $\mathbf{r}$, $\dot{\mathbf{r}}$, $\mathbf{u}$, and $\dot{\mathbf{u}}$, this requirement will be met if the integration of the variational equations (30) is performed concurrently with the integration of the basic equations (2) and (10).

We turn now to the problem of solving Eqs. (2), (10) and (30) forward in time from t_0 to t_f subject to the initial conditions $\mathbf{x} = \mathbf{x}_0$, $\mathbf{p} = \mathbf{p}_0$, and (31). This component of the optimal guidance algorithm will account for the bulk of the computations required by the algorithm as a whole, and efficiency is therefore a primary consideration in devising a numerical integration scheme for the solution of the initial value problem.

Many schemes are known for integrating a first-order scalar differential equation. Integration of a system of simultaneous differential equations is normally accomplished by applying a particular such numerical integration scheme simultaneously to all equations in the system. Thus all the differential equations are evaluated equally often, and the amount of computation required to solve the system of equations depends on the sum of the amounts of computation required to evaluate the individual equations.

In the case of the present system of differential equations, however, not all of the integrands need to be treated uniformly. In particular, Eqs. (2) and (10) do not depend on the matrix W involved in Eqs. (30). Thus it is possible to evaluate (30) less frequently than (2) and (10) without disrupting the integration process.

In fact, integration of (30) using a larger step size than that used in integrating (2) and (10) can greatly improve the efficiency of the integration as a whole. This is because the right-hand side of (30) requires considerably more computation than the right-hand sides of (2) and (10), and at the same time the matrix of partial derivatives resulting from the integration of (30) requires less accuracy than the final values of $\mathbf{x}_f$ and $\mathbf{p}_f$ resulting from the integration of (2) and (10).

More explicitly, it turns out that we can save considerable computation by adopting the following scheme of combination steps. Each combination step consists of a sequence of three steps of size h of integration of (2) and (10) followed by a single overlapping step of size $3h$ of integration of (30). Since the step size for (30) is three times the step size for (2) and (10), the expected truncation error is correspondingly 3^{n+1} times as great, where n is the order of the integration scheme used in carrying out all individual steps.

A rough count of the arithmetic operations involved shows that evaluation of (30) requires nearly an order of magnitude more computation than evaluation of (2) and (10). Hence we can add to the combination step, at negligible cost, a redundant step of size $3h$ of integration of (2) and (10) spanning the same interval as in the three steps of size h. Because of its redundancy, this added step makes possible an estimate of the error involved in the sequence of three steps in (2) and (10) by means of a technique called the "Richardson extrapolation."

The Richardson extrapolation assumes that the error resulting from a single integration step of size h is approximately proportional to h^{n+1} where n is the order of the integration method. Thus, if $y_{i+1}^{(h)}$ is the result of integrating from y_i in 3 steps of size h, and $y_{i+1}^{(3h)}$ is the result of integrating from y_i in one step of size $3h$, then approximately

$$y_{i+1}^{(h)} = y_{i+1} + 3q(h)^{n+1}$$

$$y_{i+1}^{(3h)} = y_{i+1} + q(3h)^{n+1}.$$

Therefore, eliminating q, we have approximately

$$y_{i+1} = y_{i+1}^{(h)} + \frac{y_{i+1}^{(h)} - y_{i+1}^{(3h)}}{3^n - 1},$$

of which the last term must be an estimate of the error in $y_{i+1}^{(h)}$.

Just as it would be wasteful to assign the same integration step size to all integrands regardless of their accuracy requirements, it would be wasteful to use the same integration step size throughout the integration regardless of variations in the physical situation. In the case of many high thrust rockets, the thrust acceleration magnitude function $a(t)$ varies by an order of magnitude in the space of a few minutes, and correspondingly serious variations in the integration truncation error are to be expected. For this reason it is desirable to use a numerical integration scheme whose step size can be altered conveniently from step to step.

Runge-Kutta methods have the advantage of continuously variable step size. Moreover, they are self-starting and have been widely used in trajectory calculations with very good results. Normally, the most serious disadvantage of Runge-Kutta methods is that there is no simple means of estimating the truncation error, and therefore there is no simple criterion on which to base changes in the integration step size. However, we have already seen that in the present problem a redundant step can be taken as part of each combination step without significant extra computation, providing a good empirical estimate of the local truncation error in the most critical integrands.

By comparison with various multi-step methods, Runge-Kutta methods are not usually considered to be efficient. For example, a typical predictor-corrector method requires two evaluations of the right-hand sides of the differential equations for each integration step, while the most widely used Runge-Kutta method (fourth-order) requires

four evaluations. However, since the differential equations under discussion can be written as a second-order system whose right sides do not involve first derivatives, one of the four evaluations needed in the Runge-Kutta scheme becomes unnecessary. Thus the Runge-Kutta scheme requires only 50% more computation than a typical predictor-corrector method, and this disadvantage is outweighed by the advantage of the step size flexibility. Accordingly, we have selected a fourth-order Runge-Kutta numerical scheme for performing the individual steps comprising the combination step described above.

The total integration from t_0 to t_f consists of a sequence of combination steps whose sizes H_i are adjusted in accordance with estimates of the local truncation errors of the Runge-Kutta integration steps. The policy for the adjustments of step sizes is based upon the assumption that over a short period of time numerical integration truncation error remains approximately proportional to H^{n+1} where n is the order of the integration scheme (in this case $n = 4$). In so far as this assumption is valid, one can estimate the error E_{i+1}, given the error E_i and size H_i of step i. Thus,

$$E_{i+1} \approx E_i\left(\frac{H_{i+1}}{H_i}\right)^5.$$

Hence, insofar as the assumption is valid, one can compute the step size H_{i+1} which will cause the error E_{i+1} to take on a desired value E_d:

$$H_{i+1} = H_i\left(\frac{E_d}{E_i}\right)^{1/5}.$$

Since this procedure for controlling local truncation error is only approximate, one cannot expect the local truncation error as estimated by the Richardson extrapolation technique to maintain precisely the desired level E_d. However, minor fluctuations in this error level do no harm, and, in the rare event of a sudden and substantial increase in the error level, the current numerical integration step may be sacrificed and the step size reduced for an additional attempt.

3. A generalized Newton's method

In the previous section, a method was developed for computing a vector function $\mathbf{z} = \mathbf{f}(\mathbf{y})$ and also its Jacobian matrix $[\partial \mathbf{f}(\mathbf{y})/\partial \mathbf{y}]$ of partial derivatives. These are the requisites for applying Newton's method to compute the value of $\mathbf{y}$ that gives rise to a desired value $\mathbf{z}^*$ of the function. But there is also a large family of other techniques making use of the same capability of computing $\mathbf{f}$ and its partials, and Newton's method may be viewed as a special—even degenerate—case of this more general family of methods. In this section the general family of methods is introduced and discussed. Similar but less adaptive methods have been suggested before, e.g., by N. B. Hemesath,[12] although without discussion of stability, error estimation, and step-size control.

First recall that Newton's method is an iterative method for finding a vector zero of

$$\mathbf{z}^* - \mathbf{f}(\mathbf{y}). \tag{32}$$

Starting from an initial estimate $\mathbf{y}_0$ of the solution, each successive estimate $\mathbf{y}_{i+1}$ is computed from its predecessor $\mathbf{y}_i$ by solving the linear system

$$\mathbf{z}^* - \mathbf{f}(\mathbf{y}_i) = \left[\frac{\partial \mathbf{f}(\mathbf{y}_i)}{\partial \mathbf{y}_i}\right](\mathbf{y}_{i+1} - \mathbf{y}_i). \tag{33}$$

Theoretically, the method will converge quadratically provided $\mathbf{y}_0$ is sufficiently close to the solution.

The chief drawback of Newton's method is that in many applications sufficiently close initial estimates are not available. In such cases some other method must be used in a preliminary search to obtain an approximate solution. This limitation is due to the limited "range of linearity" within which the linear approximation,

$$\Delta \mathbf{z} = \left[\frac{\partial \mathbf{f}(\mathbf{y})}{\partial \mathbf{y}}\right]\Delta \mathbf{y}, \tag{34}$$

upon which Eq. (33) is based, remains valid. If the increment from $\mathbf{y}_i$ to $\mathbf{y}_{i+1}$ far exceeds the range of linearity, there is no reason for the resulting change to $\mathbf{z}$ to correspond to it according to Eq. (34).

A widely used remedy for this difficulty is to scale the y-increment and z-increment by a scalar factor R that is less than unity. Thus instead of solving Eq. (33) to obtain $\mathbf{y}_{i+1}$, one solves

$$R(\mathbf{z}^* - \mathbf{z}_i) = \left[\frac{\partial \mathbf{f}(\mathbf{y}_i)}{\partial \mathbf{y}}\right](\mathbf{y}_{i+1} - \mathbf{y}_i). \tag{35}$$

By choosing R sufficiently small, the resulting increments can be made to agree arbitrarily well with (34) provided the Jacobian matrix is continuous and non-singular.

Let us call this version of Newton's method, in which a scale factor less than one is used, the "modified Newton's method." The modified Newton's method has been applied with varying degrees of success to a number of practical problems. When it fails, the trouble is usually that R needs to be so small for good agreement with (34) that the number of iterations required is prohibitive.

This suggests the possibility of a limiting process as R goes to zero that always reaches the solution. The limiting relationship between successive values of $\mathbf{z}$ would be simply, by (33) and (35),

$$\mathbf{z}_{i+1} - \mathbf{z}_i = R(\mathbf{z}^* - \mathbf{z}_i),$$

a difference equation whose solution is a sequence of points on a straight line from $\mathbf{z}_0$ to $\mathbf{z}^*$. It is natural then to pass from this difference equation to a differential equation

for $\mathbf{z}$ as a function of a parameter t whose solution is a straight line segment to $\mathbf{z}^*$. The simplest such is

$$\dot{\mathbf{z}} = \mathbf{z}^* - \mathbf{z},$$

whose solution, given the initial value $\mathbf{z}(0) = \mathbf{z}_0$, is

$$\mathbf{z}(t) = \mathbf{z}_0 e^{-t} + \mathbf{z}^*(1 - e^{-t}).$$

This solution approaches $\mathbf{z}^*$ only asymptotically as time becomes infinite because the magnitude of $\dot{\mathbf{z}}$ decreases with the distance to go. In order to reach $\mathbf{z}^*$ in finite time, the magnitude of $\dot{\mathbf{z}}$ can be made uniform by normalizing. Thus,

$$\dot{\mathbf{z}} = \frac{\mathbf{z}^* - \mathbf{z}}{||\mathbf{z}^* - \mathbf{z}||}, \tag{36}$$

whose solution, given the initial condition $\mathbf{z}(0) \equiv \mathbf{z}_0$, is

$$\mathbf{z}(t) = \mathbf{z}_0 + \frac{\mathbf{z}^* - \mathbf{z}_0}{||\mathbf{z}^* - \mathbf{z}_0||} t, \tag{37}$$

reaching $\mathbf{z}^*$ at $t = ||\mathbf{z}^* - \mathbf{z}_0||$.

If there is a function $\mathbf{y}(t)$ corresponding to the solution $\mathbf{z}(t)$ of (36) in the sense that $\mathbf{f}[\mathbf{y}(t)] = \mathbf{z}(t)$, then, by the chain rule,

$$\dot{\mathbf{z}} = \left[\frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right]\dot{\mathbf{y}}, \tag{38}$$

so by (36), $\mathbf{y}(t)$ satisfies

$$\dot{\mathbf{y}} = \left[\frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right]^{-1} \frac{\mathbf{z}^* - \mathbf{z}}{||\mathbf{z}^* - \mathbf{z}||}; \tag{39}$$

that is, since $\mathbf{z} = \mathbf{f}(\mathbf{y})$, $\mathbf{y}(t)$ satisfies

$$\dot{\mathbf{y}} = \left[\frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right]^{-1}\left[\frac{\mathbf{z}^* - \mathbf{f}(\mathbf{y})}{||\mathbf{z}^* - \mathbf{f}(\mathbf{y})||}\right]. \tag{40}$$

It follows that given any initial value $\mathbf{y}_0$, the solution of (40) (if it exists and is unique) is an arc in y-space that corresponds in z-space to the straight line segment from $\mathbf{f}(\mathbf{y}_0)$ to $\mathbf{z}^*$.

So in order to obtain a solution of $\mathbf{z}^* = \mathbf{f}(\mathbf{y})$, we need only solve (40) as an initial value problem starting at $t = 0$ from any initial condition $\mathbf{y}(0) = \mathbf{y}_0$ and stopping at $t = ||\mathbf{z}^* - \mathbf{f}(\mathbf{y}_0)||$. The word "only" may seem uncalled for since usually initial value problems are reckoned more difficult than zero-finding problems. But the relative difficulty in practice varies widely for different zero-finding problems and different initial value problems.

To propose numerical solution of (40) in order to obtain a zero of $\mathbf{z}^* - \mathbf{f}(\mathbf{y})$ is not to offer a specific zero-finding algorithm but simply to reformulate the zero-finding problem in such a way that we are bringing to bear a great deal of methodology on solving initial value problems. There remain the questions, what difference equations shall we use to approximate the solution to (40) and what step sizes shall we use?

There are many answers to these questions. The very simplest difference equations, those of Euler's integration method, used with a policy of choosing the step size to be the total distance to go $||\mathbf{z}^* - \mathbf{z}||$ give what amounts to Newton's method. The same difference equations used with a more flexible step-size policy give the equivalent of the modified Newton's method. More sophisticated difference equations such as Runge-Kutta equations, can be used to achieve larger step sizes for a given error level.

Newton's method and the modified Newton's method thus emerge as only the simplest members of a large family of methods aimed at solving (40). Different choices of difference equations or step-size adjustment policy give rise to different methods. In fact, the family of methods can be enlarged still further by multiplying the right-hand side of (40) by an arbitrary positive-valued scalar function provided it does not grow or approach zero too rapidly. Such a multiplication has the effect of stretching or compressing the time scale of (40) without changing the locus of the solution.

The problem of solving (40) is distinguished from most initial value problems by the fact that if time to go is recomputed after each step, then even very large errors in integration do not damage the final solution Equation (36) and therefore also (40) steer $\mathbf{z}$ to $\mathbf{z}^*$ from *any* initial conditions (provided that (40) satisfies a Lipschitz condition); the only effect of the initial condition $\mathbf{z}_0$ is on the time to go $||\mathbf{z}^* - \mathbf{z}_0||$. Thus, an error committed in the i^{th} step may delay the final solution or hasten it depending on its effect on $||\mathbf{z}^* - \mathbf{z}_{i+1}||$, but it can not affect the final value of $\mathbf{y}$ unless there are multiple solutions to the zero-finding problem or Eq. (40) becomes ill-conditioned.

A second peculiarity of the problem of integrating (40) is that there is a built-in error indication. Since the solution of the corresponding equation, (36), in z-space is known, one can compare at the end of each step the value of $\mathbf{z}$ actually reached and the value

$$\mathbf{z}_i + [(\mathbf{z}^* - \mathbf{z}_i)/||\mathbf{z}^* - \mathbf{z}_i||]h$$

that would have resulted from following the solution of (40) exactly. To be sure, this measured error in $\mathbf{z}$ cannot readily be translated into an equivalent error in $\mathbf{y}$, but, after all, it is distance in z-space that determines time to go, so z-error is entirely adequate for evaluating progress toward a solution.

An important feature of (36) that carries over to (40) is its increasing sensitivity to $\mathbf{z}$ as $\mathbf{z}$ approaches $\mathbf{z}^*$. In fact, the right-hand sides of both equations are undefined at $\mathbf{z} = \mathbf{z}^*$. One can guarantee the existence and uniqueness of solutions to (36) by stipulating that the right-hand side is 0 at $\mathbf{z} = \mathbf{z}^*$, but this does not remove the discontinuity of $\mathbf{z}$ as a function of $\mathbf{z}$ or affect its sensitivity in the vicinity of $\mathbf{z}^*$.

Each of these special features of the differential equations is important in the choice of a method for their numerical solution. Consider first the fact that integration error at each step does not affect the final solution but only the distance (= time) to go, $||z^* - z_i||$. If we define progress as the reduction of this distance and if we make the pessimistic assumption that z-error is always in the worst direction, $-(z^* - z_i)/||z^* - z_i||$, then the progress of one step can be represented by $||z^* - z_i|| - ||z^* - z_{i+1}||$; that is, by

$$||z^* - z_i|| - \left|\left| z^* - \left[z_i + h_i \frac{z^* - z_i}{||z^* - z_i||} - e_i \frac{z^* - z_i}{||z^* - z_i||} \right] \right|\right|$$

if e_i is the magnitude of the z-error. This reduces to simply $h_i - e_i$.

A crude model for the error magnitude e as a function of the step size h is $e = ch^{n+1}$ where c is a constant and n is the order of accuracy of the integration formula. In terms of this model, we want to choose h_i to maximize the progress, $h_i - e_i$, or

$$h_i = ch_i^{n+1}. \tag{41}$$

Setting to zero the derivative of (41) with respect to h_i yields

$$\frac{1}{n+1} = ch_i^n. \tag{42}$$

But ch_i^n is e_i/h_i, the relative error of the step, so the best strategy according to this crude and pessimistic model is to maintain a *relative* error of $1/(n+1)$ at each step.

A simple way of achieving approximately a desired relative error level e_r, again using the model $e = ch^n$, is to use the rule:

$$h_{i+1} = h_i \left[\frac{e_r}{e_i/h_i} \right]^{1/n} \tag{43}$$

for updating step sizes. Numerical experiments using Euler's method ($n = 1$) and a Runge-Kutta method ($n = 4$) have confirmed that relative error levels of $1/2$ and $1/5$ are nearly optimum although somewhat conservative.

A policy of maintaining such high relative error levels is not without difficulties. The high-order difference equations usually used in predictor-corrector integration schemes are subject to "parasitic" solutions that can easily dominate the intended solution unless error levels are kept very low. Therefore, such methods are not promising choices for integration of (40). Instead, methods using first-order difference equations, the "single-step" methods, are suggested. Runge-Kutta schemes can be used successfully at relatively high error levels with large step sizes. Like all single-step methods they are self-starting and permit readjustment of step size after each step, for example by the rule (43). So the usual Runge-Kutta scheme with fourth-order accuracy is a natural choice for integration of (40) when large step size is a primary concern.

But it should be noticed that near the end of the integration the step size will be determined by time to go, not by error considerations. The capability of a fourth-order method to accept large step sizes will be wasted when time to go is short compared to the allowable step size. Euler's first-order method is then more appropriate since its computational burden per step is much smaller. Moreover, usual Runge-Kutta schemes involve one evaluation of the differential equation at the end of each step, and because of the increasing sensitivity of (40) already noted, this last evaluation contributes almost pure noise to the step when time to go is short. Thus the added labor of a Runge-Kutta step as compared to an Euler step would be a definite hindrance to convergence near the end of the integration.

4. Implementation results

The initial value algorithm of Section 2 was implemented in a computer program for purposes of solving 3-, 4-, 5-, and 6-constraint orbital injection problems for the Saturn program. The IBM 7094 compilation resulted in a program taking up about four thousand words of memory and requiring less than one-half second per solution of the initial value problem (including variational equations) to compute typical trajectories lasting four to five hundred seconds. Integration accuracy was maintained at a level approximately two orders of magnitude better than state-of-the-art navigation equipment can achieve, and single precision proved sufficient in all cases to avoid ill-conditioning of the matrix of partial derivatives.

The boundary value iteration has been solved using two different methods of integrating Eq. (40). One method uses the Euler equations so that the boundary value iteration is essentially the modified Newton's method. The other is a combination method employing a branching logic such that Euler equations are used near the solution $z = z^*$ and the standard fourth-order Runge-Kutta equations are used when $||z^* - z||$ is large in comparison with current step size. Since each evaluation of the right-hand side of (40) requires one complete solution of the initial value problem—hence, say, one-half second of 7094 time—a good measure of cost of the boundary value search is the number of right-hand side evaluations. As might be expected, the combination method proved superior on the whole to the simple Euler method despite the fact that a Runge-Kutta step requires four right-hand side evaluations instead of one, for the extra computation is usually more than compensated by increased step sizes where $||z^* - z||$ is large.

The number of right-hand side evaluations varied from 2 to 5 in the case of trajectories differing slightly from known trajectories (for example, worst case perturbations of Saturn rocket performance) to 10, 20, or even 50 in cases where the solution differed radically from the initial estimate (for example, initial thrust direction misaligned 90 or 180 degrees). Nearly all cases of failure of convergence that have arisen in practice proved attributable to gross mistakes in the input data. A tendency was observed for regions of *practical* convergence, i.e., convergence in less than, say, 50 right-hand side evaluations, to become smaller for extended missions.

Acknowledgment

The authors are grateful for the continued encouragement and technical support received from N. R. Ruest and E. W. Smythe and members of their development groups at the IBM Space Systems Center, Huntsville, Alabama.

References

1. L. S. Pontraiagin, et al., *The Mathematical Theory of Optimal Processes*, Interscience Publishers, John Wiley & Sons, Inc., New York & London, 1962.
2. S. C. Dreyfus, *Dynamic Programming and the Calculus of Variations*, Academic Press, New York and London, 1965.
3. D. MacPherson, "An Explicit Solution to the Powered Flight Dynamics of a Rocket Vehicle," Aerospace Corp., Report No. TDR-169(3126) TN-2, October 31, 1962.
4. I. E. Smith, "A Three Dimensional Ascending Iterative Guidance Mode," NASA-MSFC, Report No. MTP-AERO-63-49, June 24, 1963.
5. G. W. Cherry, "A General Explicit, Optimizing Guidance Law for Rocket-Propelled Spaceflight," AIAA Paper No. 64–638, August 1964.
6. G. Leitmann, *Optimization Techniques*, Academic Press, New York & London, 1962.
7. K. R. Brown and G. W. Johnson, "Optimal Guidance for Orbital Transfer," IBM Report No. 65-221-0003H, 30 August 1965.
8. K. R. Brown and G. W. Johnson, "Real-Time Optimal Guidance," *IEEE Trans. Autom. Control*, to be published.
9. B. Paiewonsky, "Optimal Control: A Review of Theory and Practice," *AIAA Journal* **3**, 1985–2006 (1965).
10. W. G. Melbourne, C. G. Saur and D. E. Richardson, "Interplanetary Trajectory Optimization with Power Limited Propulsion Systems," *Proceedings IAS Symposium on Vehicle Systems Optimization*, Garden City, N. Y., pp. 138–150, November, 1961.
11. R. H. Hillsley and H. M. Robbins, "A Steepest-Ascent Trajectory Optimization Method which Reduces Memory Requirements," *Computing Methods in Optimization Problems*, ed. A. V. Balakrishnan and L. W. Neustadt, Academic Press, 1964.
12. N. B. Hemesath, "A novel method for solving the vector equation $\mathbf{f}(\mathbf{x}) = \mathbf{0}$," *IEEE Trans. Autom. Control* **AC-10**, 483 (1965).

Received October 4, 1966