AD NUMBER

AD867465

NEW LIMITATION CHANGE

TO

Approved for public release, distribution unlimited

FROM

Distribution authorized to U.S. Gov't. agencies and their contractors; Administrative/Operational Use; JAN 1970. Other requests shall be referred to Naval Electronics Laboratory Center, San Diego, CA 92152.

AUTHORITY

USNELC ltr, 22 Aug 1974

TECHNICAL DOCUMENT 82 Best Available Copy

AD 867465

DIRECT-SEARCH METHODS FOR THE SOLUTION OF THE TWO-POINT BOUNDARY VALUE PROBLEM (TPBVP)

Several direct-search algorithms are used to solve a TPBVP associated with trajectory optimization – prove simple in concept and computational formulation

W. J. Dejka Research and Development

30 January 1970

This do... controls and each transmitt... foreign nationals may be made on...

NAVAL ELECTRONICS LABORATORY CENTER
San Diego, California 92152

33

PROBLEM

Review the theory of optimization and mathematical programming. Specifically, investigate the use of direct-search algorithms for solving two-point boundary value problems (TPBVP). Compare direct-search methods with other accepted methods, such as quasilinearization and the method of perturbations, emphasizing the principal differences in the approaches.

RESULTS

1. Several direct-search algorithms are used to solve a TPBVP associated with Earth-Mars trajectory optimization.

2. Direct-search methods are shown to be simple in both concept and computational formulation, but to require greater computer run time.

RECOMMENDATION

Continue efforts to develop quadratically convergent search schemes. Try gradient methods on this problem, using approximations of the gradient.

ADMINISTRATIVE INFORMATION

Work was performed by the Advanced Control Systems Division under ZFXX-112-001 and ZFXX-212-001 (NELC Z223) from July 1968 to July 1969, and the document was approved for publication 30 January 1970.

CONTENTS

ILLUSTRATION

TABLE

INTRODUCTION

DIRECT-SEARCH TECHNIQUES DETERMINE THE INDEPENDENT VARIABLES THAT MINIMIZE OR MAXIMIZE A GIVEN OBJECTIVE FUNCTION

Mathematical programming concepts and the study of optimization have resulted in the development of direct-search algorithms that are applicable to numerous engineering design problems. These techniques determine the independent variables $x_1, x_2, \ldots, x_n$ that minimize or maximize a given objective function $f(x_1, x_2, \ldots, x_n)$. The objective function, or performance function, as it is frequently called, is repeatedly evaluated for different sets of independent variables, and each time the function values are compared and the 'best' or optimizing variables are retained. The iterative manner in which the independent variables $x_1, x_2, \ldots, x_n$ are computed affects the speed and efficiency of the direct-search methods. In general, the variables are changed (1) to improve the value of the objective function and (2) to give information about the objective function which will determine future changes in the variables.

Direct-search methods are analytically and computationally simple and have proved successful in many difficult design problems. They are also useful in the solution of the TPBVP if it can be formulated appropriately as an optimization problem. The TPBVP[1] is discussed in detail in the following section. It is difficult to solve because some of the boundary conditions are specified at the initial time while others are specified at the final time. Thus, the solution requires that a system of ordinary differential equations be numerically integrated until all the boundary conditions are satisfied. This problem therefore requires that an iterative procedure be employed. As with all iterative procedures, convergence is an important consideration.

TPBVP OCCURS IN MANY PHYSICAL SCIENCES – HAS BEEN EMPHASIZED IN OPTIMAL CONTROL THEORY

The TPBVP occurs commonly in many of the physical sciences. It occurs in flight mechanics for orbit determination as well as in intercept and rendezvous studies. It also occurs in science involving the transport and distribution of mass or energy. Other examples are the stress and strain distribution in a given material under a specified load and the flow of material through a heat exchanger. Perhaps the greatest emphasis on the TPBVP has occurred in the study of optimal control theory. In minimizing a time integral performance function subject to the differential equations describing a given system, a TPBVP must be solved. The major methods of TPBVP solution are discussed and evaluated in a recent article by Tapley and Lewallen[2] and their results are used in the comparison with the direct-search methods presented in this report.

After a cursory discussion of the TPBVP, the techniques that are employed are discussed. These include the method of Hooke and Jeeves[3] and the method of Flood and Leon[4]. A method by Zangwill[5], which is an improved version of Powell's method[6], is also presented. Though other search schemes are available, these algorithms are well known and demonstrate the principle of solving TPBVP by using direct-search methods.

These techniques are applied to an Earth-Mars trajectory problem[7] which results from a minimum-time optimal control problem. This control problem has been used as a standard TPBVP for trying the various methods of solution. The techniques are compared for

1. Simplicity of formulation
2. Convergence time
3. Computational requirements
4. Sensitivity to initial conditions

Numerical results of applying direct search to the trajectory problem are tabulated.

---

[1] See *REFERENCES*.

TWO-POINT BOUNDARY VALUE PROBLEM

The two-point boundary value problem is stated as follows: given the system of differential equations

$$\dot{\mathbf{y}}(t) = f(\mathbf{y}(t)) \tag{1}$$

where $\mathbf{y}(t)$ is an n-dimensional vector

$$\mathbf{y}^{\mathrm{T}}(t) = \left[y_1(t), y_2(t), \ldots, y_n(t)\right] \tag{2}$$

and where superscript T denotes matrix transpose and t is the independent variable. The initial conditions are given by the p-dimensional vector function

$$\begin{aligned} \mathbf{g}^{\mathrm{T}}(\mathbf{y}(t_0), t_0) &= \left[g_1(\mathbf{y}(t_0), t_0), g_2(\mathbf{y}(t_0), t_0), \ldots, g_p(\mathbf{y}(t_0), t_0)\right] \\ &= \mathbf{0}^{\mathrm{T}} \end{aligned} \tag{3}$$

$$p \leqslant n$$

and the final conditions by the q-dimensional vector function

$$\begin{aligned} \mathbf{h}^{\mathrm{T}}(\mathbf{y}(t_{\mathrm{f}}), t_{\mathrm{f}}) &= \left[\mathrm{h}_1(\mathbf{y}(t_{\mathrm{f}}), t_{\mathrm{f}}), \mathrm{h}_2(\mathbf{y}(t_{\mathrm{f}}), t_{\mathrm{f}}), \ldots, h_{\mathrm{q}}(\mathbf{y}(t_{\mathrm{f}}), t_{\mathrm{f}})\right] \\ &= \mathbf{0}^{\mathrm{T}} \end{aligned} \tag{4}$$

with t_0 specified, $q = n + 1 - p$. The assumption is that t_{f}, the final time, is not specified; hence, it must be determined along with the $n-p$ initial conditions. If all the initial conditions g and t_{f} are known, the problem is an initial-value problem and is solved directly by integrating the system represented by expression (1). However, in the TPBVP in which certain initial conditions have been replaced with final conditions, the solution of the problem is characterized by the use of successive approximations to match the given final conditions.

An example of a simple nonlinear two-point boundary value is

$$\dot{y}_1 = y_2(t) \tag{5}$$

$$\dot{y}_2 = -\left|y_1(t)\right| \tag{6}$$

with

$$y_1(t_0) = 0 \tag{7}$$

$$y_2(t_{\mathrm{f}}) = B \tag{8}$$

where t_0 and t_{f} are given. The initial slope $y_2(t_0)$ is unknown, and instead the final condition $y_2(t_{\mathrm{f}})$ is specified. The existence and uniqueness of the solution are dependent on the number B. The methods of solution reviewed are essentially computational realization of existence proofs. Other questions arise: Does the problem have a solution? One solution only? More than one solution? Reference 1 contains a lucid discussion of the existence and uniqueness of solutions to TPBVP.

Most of the methods of solving the TPBVP can be classed in one of two groups (1) those that solve a sequence of nonlinear initial-value problems and (2) those that solve a sequence of linear boundary value problems.

'SHOOTING' METHODS OF SOLVING THE TPBVP: SUCCESSIVE APPROXIMATIONS OF *INITIAL* CONDITIONS LEAD TO A MATCHING OF THE *GIVEN FINAL* CONDITIONS

1. Methods of the first class are called 'shooting' methods. These methods select trial initial conditions, integrate the system of differential equations, and then check to see whether the final boundary conditions are satisfied. The errors at the final boundary are used to correct the initial values and the process is repeated. In the context of this report the shooting method is summarized as follows:

If the unknown initial conditions are identified as scalars x_i, the dependence of the solution upon the initial conditions can be emphasized by writing $y(t,\mathbf{x})$.

Therefore, the final boundary conditions (4) will depend on $\mathbf{x}^T = (x_1, \ldots, x_p)$ as shown by

$$\mathbf{h}(\mathbf{x}, t_f) = \mathbf{0} \tag{9}$$

The shooting method therefore is a method of finding a $\mathbf{x}^0$ solution to (9); that is,

$$\mathbf{h}(\mathbf{x}^0, t_f) = \mathbf{0} \tag{10}$$

If the final time t_f is unknown, an additional scalar $x_{p+1} = t_f$ is defined and equation (9) becomes

$$\mathbf{h}(\mathbf{x}) = \mathbf{0} \tag{11}$$

with $\mathbf{x}^T = (x_1, x_2, \ldots, x_{p+1})$. Any standard procedure for finding roots of a vector function can be used. *Integration* of the system of differential equations is of course required to evaluate $\mathbf{h}(\mathbf{x})$.

OTHER METHODS SOLVE A SEQUENCE OF LINEAR BOUNDARY VALUE PROBLEMS CONVERGING TO A SOLUTION

2. The other class of methods for solving TPBVP's involves solving a sequence of linear boundary value problems such that the sequence converges to the solution of the original problem. Among these methods are quasilinearization and the method of perturbations. The methods are useful, since solving linear TPBVP's is considerably simpler, though formulating and computing with these methods are more complex operations. The method of perturbations is presented to illustrate this class of methods.

The method of perturbations develops a Taylor's series expansion of the original problem and solves the linear problem that results. The n^{th} approximation is given by

$$\dot{\mathbf{y}}_{n+1} = \dot{\mathbf{y}}_n + \left[\frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right]_{\mathbf{y}_n} [\mathbf{y}_{n+1} - \mathbf{y}_n] + O^2 (\mathbf{y}_{n+1} - \mathbf{y}_n) \tag{12}$$

If the notation $\delta\mathbf{y} = \mathbf{y}_{n+1} - \mathbf{y}_n$ is used and only the linear terms in the expansion are retained, the linear perturbation is seen to be

$$\dot{\delta}\mathbf{y} = \left[\frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right]_{\mathbf{y}_n} \delta\mathbf{y} = A\delta\mathbf{y} \tag{13}$$

where the matrix of the partial derivatives

$$A = \left[\frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right] = \left[\left[\frac{\partial f_i}{\partial y_j}\right]\right]$$

is evaluated on the n^{th} trajectory (note that integrating the nonlinear equations is still required). The final boundary conditions are also expanded about the terminal conditions of the n^{th} trajectory. The result can be expressed as

$$d\mathbf{h} = \left[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}\right] d\mathbf{y} + \left[\frac{\partial \mathbf{h}}{\partial t}\right] dt_f \tag{14}$$

When the effect of a variation in the terminal time is considered

$$\mathbf{dy}_f = \delta \mathbf{y}_f + \dot{\mathbf{y}}_f \, dt_f \tag{15}$$

the final condition (14) becomes

$$\mathbf{dh} = \left[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}\right] \delta \mathbf{y}_f + \dot{\mathbf{h}} dt_f \tag{16}$$

Solving equation (13) puts the terminal conditions in terms of the initial condition variations ($\delta \mathbf{y}_f = \pi \delta \mathbf{y}_0$) where π is the characteristic solution, and the above becomes

$$\mathbf{dh} = \left[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}\right] \pi \delta \mathbf{y}_0 + \dot{\mathbf{h}} dt_f \tag{17}$$

Furthermore, since the variations in y_0 must satisfy $\mathbf{g}(\mathbf{y}(t_0), t_0) = \mathbf{0}$, it follows that

$$\mathbf{dg} = \left[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}\right]_{\mathbf{y}_0} \delta \mathbf{y}_0 = \mathbf{0} \tag{18}$$

Hence, once **dh** is determined, equations (17) and (18) are solved for unknown $\delta \mathbf{y}_0$ and dt_f. The procedure is then repeated until **dh** is within some specified convergence criterion about 0.

The methods in this category are many, with various differing analytical and computational properties. These methods are discussed briefly in order that comparisons can be made. The method of perturbation is recognized as a simple method to formulate. It is computationally simple, requiring only the integration of the original differential equations to evaluate the linearized equations.

The quasilinearization method is given by the equation

$$\dot{\mathbf{y}}_{n+1} = \dot{\mathbf{y}}_n + \left[\frac{\partial \mathbf{f}}{\partial \mathbf{y}}\right]_{\mathbf{y}_n} [\mathbf{y}_{n+1} - \mathbf{y}_n] \tag{19}$$

where n is the iteration number here. This equation is solved repeatedly. Past information is used, and both $\mathbf{y}_n$ and $\dot{\mathbf{y}}_n$ (for each n) from the previous trajectory are stored. The storage problem is severe. The results generally are good. (See Tapley and Lewallen[2] and Kenneth and McGill.[8])

In this study the above methods were not actually tried. Results given by Tapley and Lewallen are used in the comparison with the direct-search approach to TPBVP that follows.

DIRECT-SEARCH TECHNIQUES

The theory of direct search is employed to find a solution to the unconstrained optimization problem

$$\min_{\mathbf{x}} f(\mathbf{x}) \tag{20}$$

where $\mathbf{x} \in E^n$ is an n-dimensional Euclidean vector space with

$$\mathbf{x}^T = [x_1, x_2, \ldots, x_n] \tag{21}$$

but without using the usual necessary conditions for an optimum; for example, $\nabla f(\mathbf{x}^0) = 0$, or higher-order derivative information. The vector variable $\mathbf{x}$ is changed in a systematic manner and the scalar objective function evaluated after each change. The values of the objective function are compared and changes in the variables are continued until a value of $\mathbf{x}$ is found which minimizes $f(\mathbf{x})$. The theory in changing the variables affects the analytical and computational properties of these methods, and these techniques are discussed in the ensuing paragraphs. In general, these methods are easily understood and easily applied. Since they do not require the computation of the gradient, they lend themselves to optimization problems in which the gradient is extremely complex and difficult to compute or the objective function is discontinuous.

DIRECT-SEARCH METHODS LEND THEMSELVES TO OPTIMIZATION PROBLEMS

The difficulty in solving the optimization problem (20) is caused by the fact that the surface represented by

$$z = f(\mathbf{x}) \tag{22}$$

is often not of a convenient 'shape.' It would be convenient if $f(\mathbf{x})$ had a bowl shape, with its minimum clearly defined. Then a search through various $\mathbf{x}$ would yield the optimum. However, the surfaces often contain troughs, and direct-search methods often move about inefficiently looking for the optimum. The philosophy of search must include a technique to seek out a trough or ridge and follow it to an optimum. Each of the search techniques that follows has this property to some degree.

HOOKE AND JEEVES USED AN 'EXPLORATORY MOVE' OF SMALL, FIXED SIZE, FOLLOWED BY A 'PATTERN MOVE'

The method of Hooke and Jeeves was one of the earliest and most accurate of the search methods. It possesses the ability to follow a ridge, a property that accounts for its success. Further, Hooke and Jeeves introduced certain terms which have become standard terminology in search methods. The 'exploratory move' is a systematic search of the coordinate directions to determine an improved point. The method of Hooke and Jeeves (see illustration) performs an exploratory move with a small, fixed step size. Mathematically, the exploratory search evaluates

$$f(x_1, x_2, \ldots, x_i \pm \Delta, \ldots, x_n) \qquad i = 1, 2, \ldots, n \tag{23}$$

and compares it with the best optimum found on the previous coordinate search. This new point found as a result of an exploratory move is called a 'base point.'

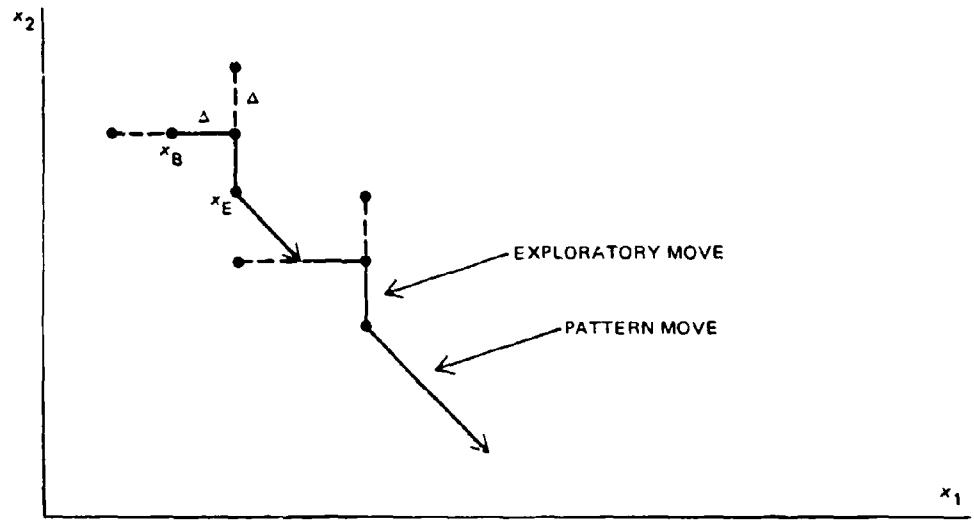
The 'pattern move' always follows a successful exploratory move. The direction for a pattern move is determined by the previous base point $\mathbf{x}_B$ and the best point of a successful exploratory move $\mathbf{x}_E$. The direction is given by

$$\mathbf{d} = \mathbf{x}_E - \mathbf{x}_B \tag{24}$$

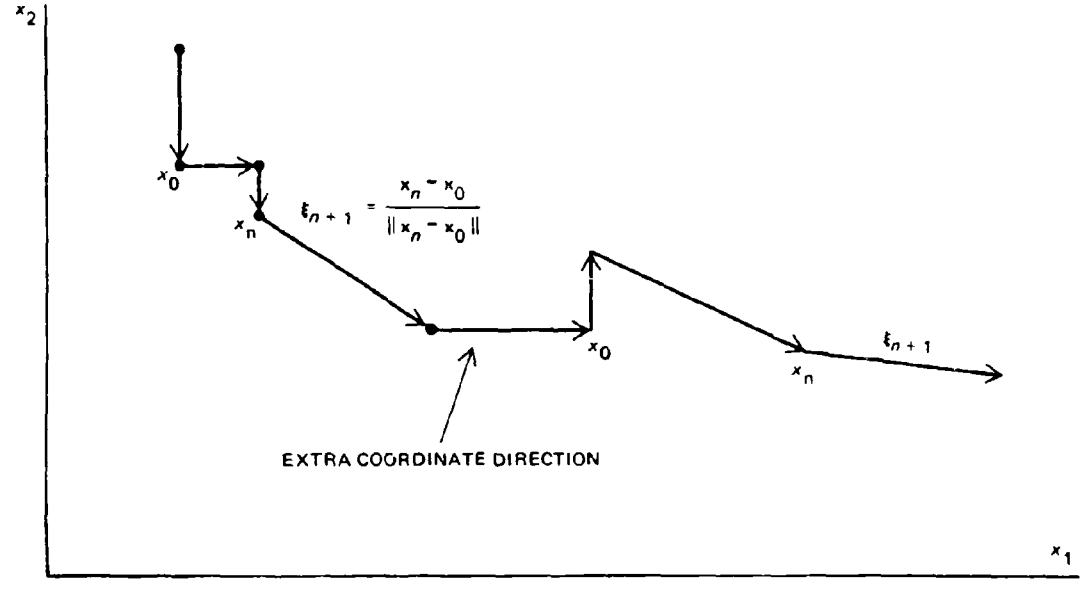
The point $\mathbf{x}_E$, the result of the last exploratory move, becomes the new base point and a new exploratory move is made from $\mathbf{x}_E + \mathbf{d}$. As long as the exploratory move yields an improved point, the distance $\mathbf{x}_E - \mathbf{x}_B$ increases and the method accelerates and simultaneously follows a ridge. When an exploratory move fails, the step size in the search scheme is reduced by a reduction factor (< 1.0) and the method continues.

This method does not determine the direction of steepest descent but settles for any improvement. However, its choice of direction yields an efficient and accurate search method. It is implemented in a computer routine called NELC DIRECT.

The method of Hooke and Jeeves (NELC DIRECT) has no formal convergence theorems to support its success, but it works and works well.



a. HOOKE AND JEEVES METHOD



b. ZANGWILL'S METHOD

Direct-search methods

FLOOD AND LEON USED EXPLORATORY MOVES TO FIND A MINIMUM ALONG COORDINATE DIRECTIONS, THEN A PATTERN MOVE

The method of Flood and Leon (UNIVAR) also has no formal theory but appears intuitively to be more plausible. It makes an exploratory move that searches along each coordinate direction until a *minimum* is found. Then it makes a pattern move using the original base point and the final improved point of the exploratory move to determine the new direction. The method then minimizes along that direction. In minimizing along a given direction, the method accelerates by taking increasingly larger steps. The method is briefly described by the following steps:

1. The exploratory move is made by incrementing an element of the vector $\mathbf{x}$, such that

$$f(x_1, x_2, \ldots, x_i + \rho_i^k \Delta, \ldots, x_n) < f(x_1, x_2, \ldots, x_i + \rho_i^{k-1} \Delta, \ldots, x_n) < \ldots < f(\mathbf{x}) \tag{25}$$

is a minimum; ρ_i is a number greater than 1, and by raising it to successive powers, $\rho_i^k \Delta$ is successively increased, resulting in an acceleration along that direction. Successive moves are continued until a failure occurs (functional value greater than the previous value); then the procedure sets $\mathbf{x} = x_1, \ldots, x_i + \rho_i^k \Delta, \ldots, x_n$ and reinitializes k to zero. Two successive iterations of this procedure are made to find a minimum along the coordinate direction. If a success is not found with $k = 0$, the direction of search along the coordinate directions is reversed, that is, $x_i - \rho_i^k \Delta$. If this fails, that point is assumed to be the minimum.

2. A pattern move is made if the functional value at the end of the exploratory move is better than the point at the beginning of the exploratory move. The direction of the move is determined by the difference of the last point $\mathbf{x}_E$ and the beginning point $\mathbf{x}_B$ of the exploratory move. It is given by

$$\mathbf{x} = \mathbf{x}_E + \rho^k (\mathbf{x}_E - \mathbf{x}_B) \tag{26}$$

and the accelerating factor ρ is raised to the k^{th} power ($k = 1, 2, \ldots$) so as to accelerate along that direction (k represents the number of successes in a given direction). After failure occurs, the procedure outlined in step 1 is repeated resetting k to zero. The negative direction is searched if failure occurs at $k = 0$.

The method of Zangwill is a search method with proved quadratic convergence. It is described as follows:

Let $\mathbf{c}_r, r = 1, 2, \ldots, n$ be the unit coordinate directions. The coordinate directions are used after one iteration of the following procedure: (1) Beginning with an initial point $\mathbf{x}_0$ and n directions ξ_i, the method minimizes $f(\mathbf{x}_{i-1} + \alpha_i \xi_i)$ setting $\mathbf{x}_i = \mathbf{x}_{i-1} + \alpha_i \xi_i$ for $i = 1, 2, \ldots, n$. (2) A direction ξ_{n+1} is computed:

$$\xi_{n+1} = \frac{\mathbf{x}_n - \mathbf{x}_0}{\|\mathbf{x}_n - \mathbf{x}_0\|} \tag{27}$$

where $\|\mathbf{x}_n - \mathbf{x}_0\|$ is the Euclidean norm (square root of the sum of the squares of the elements) and $\mathbf{x}_{n+1} = \mathbf{x}_n + \alpha_{n+1} \xi_{n+1}$, where α_{n+1} minimizes $f(\mathbf{x}_n + \alpha \xi_{n+1})$.

The directions are updated as follows:

$$\xi_i = \xi_{i+1} \qquad i = 1, 2, \ldots, n \tag{28}$$

ξ_1 is removed from the basis. (3) A direction $\mathbf{c}_t$ is introduced and the point $\mathbf{x}_{n+2} = \mathbf{x}_{n+1} + \alpha_{n+2} \mathbf{c}_t$ is determined so that $f(\mathbf{x}_{n+1} + \alpha_{n+2} \mathbf{c}_t)$ with respect to α_{n+2} is a minimum. If $\alpha_{n+2} \neq 0$ the index t ($t = 1, 2, \ldots, n$) is updated and the routine goes to part (1). If $\alpha_{n+2} = 0$ for $t = 1, \ldots, n$ the solution has been found. Note that t is cycled

through the coordinate directions; it is incremented through n and then reset to 1. This prevents the directions ξ_k, $k = 1, 2, \ldots, n$ from becoming linearly dependent. This routine is shown to yield convergence for a quadratic in a finite number of iterations.

FORMULATING A MATHEMATICAL PROGRAMMING PROBLEM

POSING THE TPBVP AS AN UNCONSTRAINED OPTIMIZATION PROBLEM NECESSITATES FORMULATING AN OBJECTIVE (OR WEIGHTING) FUNCTION

The two-point boundary value problem must necessarily be posed as an unconstrained optimization problem before the direct-search methods can be applied. The formulation of the unconstrained optimization problem can be obtained by an extension of the shooting method (see *TWO-POINT BOUNDARY VALUE PROBLEM*). The terminal or final conditions (4) for the system of differential equations are specified, and for any given set of initial conditions, the terminal conditions are computed and compared with the specified terminal conditions. The goal is to make the difference between the specified and computed final boundary conditions as small as possible; therefore, an objective (or 'weighting,' or 'penalty') function of the differences is minimized by choosing the unknown initial conditions. Because the specified final boundary conditions are zero in the TPBVP, the objective function must have the following properties:

$$F(\mathbf{h}) = 0 \quad \text{when } \mathbf{h}(\mathbf{y}(t_f), t_f) = \mathbf{0}$$

and

$$F(\mathbf{h}) > 0 \quad \text{when } |\mathbf{h}(\mathbf{y}(t_f), t_f)| \neq \mathbf{0} \tag{29}$$

One possible form of F, the objective function, is the sum of the absolute values of $h_i(\mathbf{y}(t_f), t_f)$; namely

$$F(\mathbf{h}) = \sum_{i=1}^{q} |h_i(\mathbf{x})| = F(\mathbf{x}) \tag{30}$$

where $\mathbf{x}$ replaces those initial conditions that are unknown and $x_n = t_f$. Hence, the weighting function F is a function of the initial conditions and the final time t_f, and the problem is now in the form of an unconstrained minimization problem

$$\min_{\mathbf{x}} F(\mathbf{h}(\mathbf{x})) = \min_{\mathbf{x}} F(\mathbf{x}) \tag{31}$$

The various direct-search algorithms are now applicable. It is understood, however, that to evaluate $F(\mathbf{x})$ we must *integrate* a system of nonlinear differential equations with the unknown initial conditions and final time t_f specified as $\mathbf{x}$ by the algorithm.

Another form of weighting function which is used often in practice and which is employed in the trial problem is

$$F(\mathbf{x}) = \mathbf{h}(\mathbf{x})^T \mathbf{h}(\mathbf{x}) \tag{32}$$

This is the sum of the squares of the errors in the final conditions, and the mathematical programming problem takes the form of least squares; that is, $\mathbf{x}$ is chosen so as to minimize the sum of the squares of the error.

Other independent constraints may be imposed in the formulation of the TPBVP, but these have not been considered here. There are certain constraints, such as on

the variable t_f (for example, $t_f \geq 0$), which can be included as a penalty. Any time t_f violates the constraint, F is equated to a very large number (10^6). In general, the addition of other constraints will result in a far more complex problem.

The formulation of the TPBVP as a mathematical programming problem is easily effected. It remains to discuss how successful direct search is in its solution.

RESULTS AND CONCLUSIONS

The technique of using direct search was applied to a two-point boundary value problem that resulted from the calculation of a minimum-time Earth-Mars trajectory. This particular problem has been extensively used by many investigators[2,7,8] in studying methods of solving the TPBVP, and some valid comparisons can be made between direct-search methods and other recognized methods. The trajectory problem and a list of the constants are given in appendix A. The results of computer runs on the CDC 1604 are tabulated on the following page.

The general characteristics of TPBVP solution methods considered are: (1) simplicity of formulation and implementation, (2) computer storage requirements, (3) convergence sensitivity, and (4) convergence time. Of these characteristics, the last is the most difficult to compare realistically. Different numerical integration schemes as well as different computers may significantly affect convergence time, and any comparisons are dependent on these variables. In contrast, the other characteristics are independent of the numerical integration scheme and computer.

1. Simplicity of Formulation

The direct-search method applied to the TPBVP is by far the simplest to formulate and implement. A penalty, or weighting, function of the terminal boundary conditions is chosen. The calculation of the penalty function requires only the integration of the differential equations and is therefore simpler than any method involving the calculation and integration of additional perturbation equations. The initial conditions are specified directly by the search method that is employed.

2. Computer Storage Requirements

Computer storage requirements in applying the direct-search methods to TPBVP's are minimal. Since only forward integrations of the differential equations are necessary, no intermediate storage is required. Other methods for TPBVP solutions are more complex, since the intermediate values of differential equation variables must be stored for future integration. Such storage poses a severe problem in solving large TPBVP's.

3. Convergence Sensitivity

Methods of direct search are highly insensitive to initial conditions. Convergence sensitivity of the direct methods was tested with ±20-percent change in the initial guess of final time t_f. All methods converged. The method of UNIVAR demonstrated the capability of searching a large parameter space in a short time. NELC DIRECT converged for large changes in initial conditions – however, with a larger computer run time. The insensitivity to initial conditions of these methods is a decided advantage of this approach to solving TPBVP's.

TPBVP RESULTS WITH VARIOUS SEARCH ALGORITHMS (CDC 1604)

Search Technique	Integration Method	Integration Error	Initial Point			Search Data			Solution			f_{min}	Run Time	Functional Evaluations
			λ_2	λ_3	t_f	Δ	Δ_{min}	ρ	λ_2	λ_3	t_f			
Tapley-Lewallen's result (perturb)			0.52	0.3	3.03	0.0005			0.4948	1.078	3.3194	6.17×10^{-6}		91
UNIVAR	RUNGE-KUTTA	10	0.52	0.3	3.03	0.0005		1.618	0.4894	1.094	3.032	3.34×10^{-3}	6 min 12 sec	147
UNIVAR	RUNGE-KUTTA	10^{-6}	0.52	0.3	3.03	0.0005		1.618	0.4826	1.1043	3.405	9.9×10^{-4}	4 min 42 sec	106
UNIVAR	RUNGE-KUTTA	10^{-6}	0.52	0.3	3.03	0.0005		1.682	0.4892	1.094	3.0346	3.3×10^{-3}	6 min 51 sec	144
UNIVAR	RUNGE-KUTTA	10^{-6}	0.52	0.3	3.03	0.0005		1.						
UNIVAR	RUNGE-KUTTA	10^{-6}	0.52	0.3	3.03	0.0001		1.682	0.4826	1.106	3.038	5×10^{-3}	10 min	
DIRECT		10^{-6}	0.52	0.3	3.03	0.01		0.625	0.4952	1.077	3.284	1×10^{-6}	16 min[2]	432
UNIVAR	DIFESYS[1]	10^{-6}	0.52	0.3	3.03	0.04		1.682	0.492	1.104	3.256	2.8×10^{-3}	7 min 41 sec	75
UNIVAR	DIFESYS	10^{-6}	0.52	0.3	3.03	0.005		1.682	0.466	1.127	3.668	5.6×10^{-3}	10 min	90
UNIVAR	DIFESYS	10^{-6}	0.52	0.3	3.03	0.005		1.682	0.474	1.120	3.636	3.138	10 min	120
DIRECT	DIFESYS	10^{-6}	0.52	0.3	3.03	0.1	0.0001	0.625	0.52	1.024	3.050	4.4×10^{-3}	10 min	108
DIRECT	DIFESYS	10^{-6}	0.52	1.04	2.7	0.01	0.0001	0.625	0.4899	1.094	3.356	4.3×10^{-4}	20 min[2]	
DIRECT	DIFESYS	10^{-6}	0.52	1.04	2.7	0.01	0.0001	0.625	0.4899	1.09	3.350	1×10	10 min	
ZANGWILL	RUNGE-KUTTA	10^{-6}	0.52	0.3	3.03	0.1			0.4945	1.080	3.3154	4.5×10^{-6}	10 min	214

Note 1: DIFESYS is an extrapolation method by Bulisch & Stoer.

Note 2: Computer runs were 10 minutes, maximum, with two exceptions above.

4. Convergence Time

Convergence times of the direct methods could not be realistically compared. For different step sizes and different search methods, different convergence times result. The tendency in the computational work was to choose the step size δ too small in UNIVAR and NELC DIRECT. Often the larger step size results in quicker convergence but less accuracy. However, there are results which stand out for their accuracy. There are other factors in the UNIVAR search scheme, such as the search acceleration factor ρ, which further compound the difficulty of convergence evaluation. The acceleration factor was chosen at 1.618 and never changed in the computation. Zangwill's method gave the greatest accuracy with the shortest run time (10 minutes' maximum run time).

In general, the search schemes performed remarkably well. Simplicity stands out as their strongest computation characteristic. The results were satisfactory with Zangwill's search method, which has quadratic convergence properties. However, the methods of UNIVAR and DIRECT exhibited dependence on the choice of search step size (delta or deltai) and the rho accelerating factor. As the theory of search methods evolves, additional algorithms will be developed which will add to the attractiveness of the direct-search approach for TPBVP's. New and faster computers will further contribute to the utilization of these methods for solution of the TPBVP. However, the direct-search approach always required longer computer time to reach a solution than the indirect methods given in references 2, 8, and 7.

REFERENCES

1. Bailey, P. B. and others, *Nonlinear Two Point Boundary Value Problems*, Academic Press, 1968

2. Tapley, B. D. and Lewallen, J. M., "Comparison of Several Numerical Optimization Methods," *Journal of Optimization Theory and Application*, v. 1, p. 1-32, July 1967

3. Hooke, R. and Jeeves, T. A., " 'Direct Search' Solution of Numerical and Statistical Problems," *Association for Computing Machinery. Journal*, v. 8, p. 212-229, April 1961

4. California. University. Space Sciences Laboratory Internal Working Paper 20, *Comparison Among Eight Known Optimization Procedures*, by A. Leon, August 1964

5. Zangwill, W. I., "Minimizing a Function Without Calculating Derivatives," *Computer Journal*, v. 10, p. 293-296, 1967-1968

6. Powell, M. J. D., "An Efficient Method For Finding the Minimum of a Function of Several Variables Without Calculating Derivatives," *Computer Journal*, v. 7, p. 155-162, 1964-1965

7. Kelley, H. J., "Method of Gradients," Chapter 6 in *Optimization Techniques*, by G. Leitmann, Academic Press, 1962

8. Kenneth, P. and McGill, R., "Two-Point Boundary-Value-Problem Techniques," Chapter 2 in *Advances in Control Systems*, v. 3, Academic Press, 1966

APPENDIX A: THE OPTIMAL CONTROL PROBLEM

The problem is to determine the minimum-time trajectory from Earth to Mars. The optimal control problem is to minimize

$$F = \int_{t_0}^{t_f} dt \tag{A-1}$$

subject to

$$\dot{x}_1 = \frac{(x_2)^2}{x_3} - \frac{K}{(x_3)^2} + \frac{T}{m}\sin\beta \tag{A-2}$$

$$\dot{x}_2 = \frac{x_1 x_2}{x_3} + \frac{T}{m}\cos\beta \tag{A-3}$$

$$\dot{x}_3 = x_1 \tag{A-4}$$

with respect to x, λ, t_f, and β (these equations are derived in appendix B), with

$$m = m_0 + ct \tag{A-5}$$

$x_1(t_0), x_2(t_0), x_3(t_0)$, and $x_1(t_f), x_2(t_f), x_3(t_f)$ given (K, T, m_0, and c are known constants).

The problem can be solved by using the maximum principle. A generalized Hamiltonian is formed

$$H = -1 + \lambda_1\left[\frac{(x_2)^2}{x_3} - \frac{K}{(x_3)^2} + \frac{T}{m}\sin\beta\right] + \lambda_2\left[-\frac{x_1 x_2}{x_3} + \frac{T}{m}\cos\beta\right] + \lambda_3 x_1 \tag{A-6}$$

The necessary conditions are

$$\dot{\lambda}_1 = -\frac{\partial H}{\partial x_1} = \frac{\lambda_2 x_2}{x_3} - \lambda_3 \tag{A-7}$$

$$\dot{\lambda}_2 = -\frac{\partial H}{\partial x_2} = -\frac{2x_2\lambda_1}{x_3} + \frac{x_1\lambda_2}{x_3} \tag{A-8}$$

$$\dot{\lambda}_3 = -\frac{\partial H}{\partial x_3} = +\frac{\lambda_1(x_2)^2}{(x_3)^2} - 2\frac{K}{(x_3)^3} - \frac{\lambda_2 x_2 x_1}{(x_3)^2} \tag{A-9}$$

$$\dot{x}_1 = \frac{(x_2)^2}{x_3} - \frac{K}{(x_3)^2} + \frac{T}{m}\sin\beta \tag{A-10}$$

$$\dot{x}_2 = -\frac{x_1 x_2}{x_3} + \frac{T}{m}\cos\beta \tag{A-11}$$

$$\dot{x}_3 = x_1 \tag{A-12}$$

$$\frac{\partial H}{\partial\beta} = \lambda_1\frac{T}{m}\cos\beta - \lambda_2\frac{T}{m}\sin\beta = 0 \tag{A-13}$$

$x_1(t_0), x_2(t_0), x_3(t_0), x_1(t_f), x_2(t_f), x_3(t_f)$ given with transversality condition

$$\left[-1 + \lambda_1 \left(\frac{(x_2)^2}{x_3} - \frac{K}{(x_3)^2} + \frac{T}{m} \sin\beta \right) + \lambda_2 \left(-\frac{x_1 x_2}{x_3} + \frac{T}{m} \cos\beta \right) + \lambda_3 x_1 \right]_{t_f} = 0 \tag{A-14}$$

(Further discussion of this problem is found in reference 1.)

It is possible to solve explicitly for β

$$\tan\beta = \frac{\lambda_1}{\lambda_2} \tag{A-15}$$

or

$$\sin\beta = \frac{\lambda_1}{\sqrt{(\lambda_1)^2 + (\lambda_2)^2}} \tag{A-16}$$

$$\cos\beta = \frac{\lambda_2}{\sqrt{(\lambda_1)^2 + (\lambda_2)^2}} \tag{A-17}$$

The transversality conditions (A-14) can be modified to

$$-1 + \frac{T}{m} \frac{(\lambda_1(t_f))^2 + (\lambda_2(t_f))^2}{\sqrt{(\lambda_1(t_f))^2 + (\lambda_2(t_f))^2}} = 0$$

$$-1 + \frac{T}{m} \sqrt{(\lambda_1)^2 + (\lambda_2)^2} = 0$$

$$\sqrt{(\lambda_1)^2 + (\lambda_2)^2} = \frac{m}{T}$$

$$(\lambda_1)^2 + (\lambda_2)^2 = \left(\frac{m}{T}\right)^2 \tag{A-18}$$

where

$$x_1(t_f) = 0, \left[\frac{(x_2(t_f))^2}{x_3(t_f)} = \frac{K}{(x_3(t_f))^2} \right]$$

In minimum time problems, one multiplier can be specified and hence the number of unknowns reduced. This occurs when the state equation depends only on the ratio λ_1/λ_2, and it is possible to replace the final condition (A-18) by a normalized variable $\lambda_1(t_0) = 1.0$. The transversality condition is eliminated and the number of unknown initial conditions is reduced by one.

The TPBVP is now given by:

$$\dot{x}_1 = \frac{(x_2)^2}{x_3} - \frac{K}{(x_3)^2} + \frac{T}{m} \frac{\lambda_1}{\sqrt{(\lambda_1)^2 + (\lambda_2)^2}} \tag{A-19}$$

$$\dot{x}_2 = -\frac{x_1 x_2}{x_3} + \frac{T}{m} \frac{\lambda_2}{\sqrt{(\lambda_1)^2 + (\lambda_2)^2}} \tag{A-20}$$

$$\dot{x}_3 = x_1 \tag{A-21}$$

$$\dot{\lambda}_1 = \frac{\lambda_2 x_2}{x_3} - \lambda_3 \tag{A-22}$$

$$\dot{\lambda}_2 = -\frac{2x_2\lambda_1}{x_3} + \frac{x_1\lambda_2}{x_3} \tag{A-23}$$

$$\dot{\lambda}_3 = \frac{\lambda_1 (x_2)^2}{(x_3)^2} - \frac{2K}{(x_3)^3} - \frac{\lambda_2 x_2 x_1}{(x_3)^2} \tag{A-24}$$

with

$$x_1 (t_0) = 0.0 \tag{A-25}$$

$$x_2 (t_0) = 1.0 \tag{A-26}$$

$$x_3 (t_0) = 1.0 \tag{A-27}$$

$$\lambda_1 (t_0) = 1.0 \tag{A-28}$$

$$x_1 (t_f) = 0.0 \tag{A-29}$$

$$x_2 (t_f) = 0.8098 \tag{A-30}$$

$$x_3 (t_f) = 1.525 \tag{A-31}$$

This formulation is programmed in the FORTRAN routine with

$$y^T = [x_1, x_2, x_3, \lambda_1, \lambda_2, \lambda_3] \tag{A-32}$$

The computer routine is self-contained; comment cards indicate the necessary data to be specified (appendix C).

APPENDIX B: DYNAMICS FOR A TRAJECTORY OPTIMIZATION PROBLEM

The problem to be solved involves the control of a constant-thrust rocket en route from Earth to Mars. The angle of thrust with respect to a local horizon (see page 20) is the control variable. The problem is assumed coplanar (that is, Earth and Mars are assumed to be points in a plane) with Earth and Mars exhibiting negligible gravitational force. The derivation of the dynamics involves Newton's law ($F = ma$). In a polar coordinate system, Newton's law takes the following form:

$$m\left(\frac{d^2r}{dt^2} - r\left(\frac{d\theta}{dt}\right)^2\right) = \Sigma f_r \tag{B-1}$$

$$m\left(r\frac{d^2\theta}{dt^2} + \theta\frac{dr}{dt}\frac{d\theta}{dt}\right) = \Sigma f_\theta \tag{B-2}$$

where

$$\frac{dr}{dt} = u = \text{radial velocity}$$

$$r\frac{d\theta}{dt} = v = \text{tangential velocity}$$

$$\dot{v} = \dot{r}\frac{d\theta}{dt} + r\frac{d^2\theta}{dt^2} = u\frac{d\theta}{dt} + r\frac{d^2\theta}{dt^2} \tag{B-3}$$

f are the forces

r is the radial position of the vehicle

θ is its angular position

The first equation (B-1) of motion is rewritten

$$m\left(\dot{u} - r\frac{d\theta}{dt}\frac{d\theta}{dt}\right) = \Sigma f_r \tag{B-4}$$

or

$$m\left(\dot{u} - v\frac{d\theta}{dt}\right) = -W + T\sin\beta$$

where

W = weight of the rocket

and

T = thrust of the rocket

since

$$\frac{d\theta}{dt} = \frac{v}{r} \tag{B-5}$$

Equation (B-4) becomes

$$m\left(\dot{u} - \frac{v^2}{r}\right) = -W + T\sin\beta \tag{B-6}$$

$$\dot{u} = \frac{v^2}{r} - \frac{W}{m} + \frac{T}{m}\sin\beta$$

The second equation of motion becomes

$$m\left(\dot{v} + \frac{dr}{dt}\frac{d\theta}{dt}\right) = \Sigma f_\theta$$

or

$$m\,\dot{v} + m\frac{uv}{r} = T\cos\beta \tag{B-7}$$

which is written

$$\dot{v} = \frac{uv}{r} + \frac{T}{m}\cos\beta$$

One further equation is required – the equation describing change of mass with time. This equation is given by

$$m = m_0 + ct \tag{B-8}$$

which assumes a constant change of mass.

These equations are a simplification of the trajectory problem but represent an approximate model of the system.

The state variable model is obtained by defining

$$x_1 = u = \frac{dr}{dt} = \text{radial velocity} \tag{B-9}$$

$$x_2 = v = \text{tangential velocity} \tag{B-10}$$

$$x_3 = r = \text{radial position of vehicle} \tag{B-11}$$

The equations are

$$\dot{x}_1 = \frac{(x_2)^2}{x_3} - \frac{K}{(x_3)^2} + \frac{T}{m}\sin\beta \tag{B-12}$$

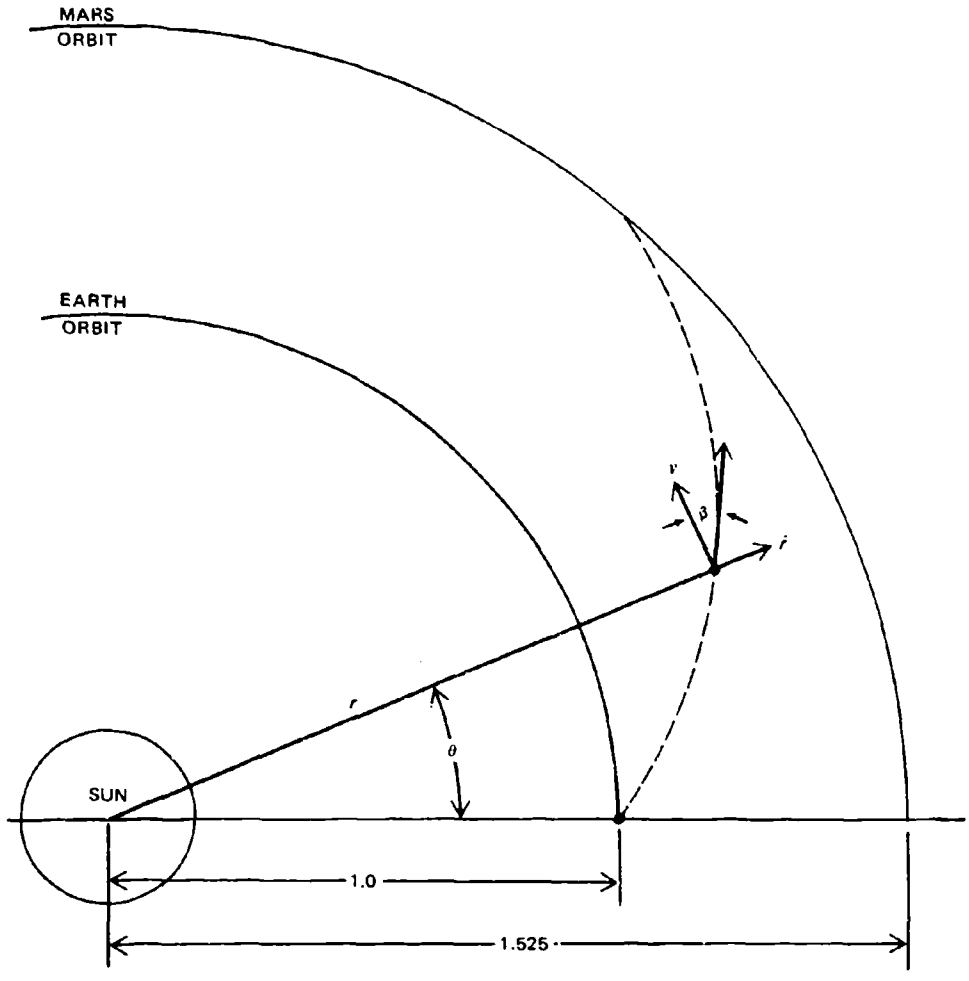
$$\dot{x}_2 = \frac{x_1 x_2}{x_3} + \frac{T}{m}\cos\beta \tag{B-13}$$

$$\dot{x}_3 = x_1 \tag{B-14}$$

where

$$\frac{w}{m} = \frac{K}{(x_3)^2}$$

and K is the gravitational constant of the sun.



Geometry of Earth and Mars trajectories

APPENDIX C: COMPUTER FLOW CHARTS AND INSTRUCTIONS

1969 045

```
      PROGRAM TPBVPSCH
C     THIS PROGRAM WILL SOLVE A TWO-POINT BOUNDARY VALUE PROBLEM
C     USING A DIRECT SEARCH SCHEME  (UNIVAR)  AND
C     THE INTEGRATION SCHEME  IS THE FOURTH ORDER RUNGE-KUTTA
C     AS MODIFIED BY GILL
C
C     THE XI ARE THE INITIAL CONDITIONS THAT ARE UNKNOWN AND WHICH
C            ARE SPECIFIED INITIALLY IN THE PROGRAM DRIVER
C
C     K REPRESENTS THE NUMBER OF          UNKNOWN VARIABLES.
C     N = THE NUMBER OF DIFFERENTIAL EQUATIONS TO BE INTEGRATED
C
C     THE DELTAI REPRESENTS A STEP CHANGE IN THE INITIAL CONDITIONS
C     AND WHICH IS USED BY THE SEARCH ROUTINE.
C     THE DELTAI SHOULD BE LARGE INITIALLY. THEN IF RESULTS ARE NOT
C     SATISFACTORY,DELTAI CAN BE REDUCED.
C
C     THE SUBROUTINE FN IS USED FOR INTEGRATION AND ALSO TO COMPUTE
C     A WEIGHING FUNCTION OF THE FINAL BOUNDARY CONDITIONS
C
C     A WEIGHING FUNCTION OF FINAL CONDITIONS MUST BE FORMULATED
C     THE WEIGHING FUNCTION IS USED BY THE DIRECT SEARCH SCHEME
C     TO DETERMINE THAT THE FINAL CONDITIONS ARE MET
C     THE FINAL CONDITIONS ARE WRITTEN AS EQUALITIES   P(Y)=0.0
C     THE WEIGHING FUNCTION IS THE SUM OF THE SQUARES
C      G=P(Y(1))*P(Y(1))+P(Y(2))*P(Y(2)) + ETC.
C
C     IF THE FINAL TIME IS UNKNOWN , AN ADDITION   VARIABLE  XI(K  ) IS
C     USED TO REPRESENT THE FINAL TIME AND
C     INSERTED IN THE INTEGRATION STOPPING CRITERION. OTHERWISE
C     THE TRUE FINAL VALUE OF THE INDEPENDENT VARIABLE IS USED
C
C     PROBLEMS CAN  BE ENCOUNTERED WITH STEP-SIZE DELTAI AND THE
C     INTEGRATION STEP-SIZE H. THE LOCAL VS GLOBAL PROBLEM IS PRESENT
C            OCCURRED AND IT IS NECESSARY TO RESTART THE      ROUTINE
C     WITH NEW INITIAL CONDITIONS TO GUARANTEE A GLOBAL OPTIMUM SOLUTION.
C
C     ADVANTAGE OF THIS ROUTINE IS THAT IT ONLY REQUIRES INTEGRATION
C     OF THE DIFFERENTIAL EQUATION .    NO PERTURBATION EQUATIONS
C     ARE REQUIRED.
C
      DIMENSION XI(4),X(4),DP(4),DEL(4),XR(4)
      EXTERNAL F
C
                        XI(1)=0.7
                        XI(2)=0.3
                        XI(3)=3.03
```

1969 045

```
C
C   K= NO. OF UNKNOWNS
                          K=3
C
C          INITIAL DELTA
                          DELTAI=0.04
      RHO=1.618
      CALL UNIVAR(K,XI,RHO,DELTAI,X,DP,DEL,XR,FR)
      END
```

1969 043

```
      SUBROUTINE FN(G,K,XI)
      DIMENSION XI(K)
C
C  THESE QUANTITIES ARE NEEDED FOR INTEGRATION. THE ARRAYS HAVE THE
C  SAME ORDER AS THE DIFFERENTIAL EQUATIONS
      DIMENSION Y(6),F(6)
      TYPE REAL MAGSQ,MAG,MOM
C
      EXTERNAL RKLDEQ
           N=6
C
C     THE KNOWN INITIAL CONDITIONS ARE SPECIFIED HERE
      Y(1)=0.0
      Y(2)=1.0
      Y(3)=1.0
      Y(4)=XI(1)
      Y(5)=XI(2)
      Y(6)=1.0
C
C  THE INITIAL VALUE OF X, THE INDEPENDENT VARIABLE AND THE    INTEGRATION
C  STEP SIZE H ARE SPECIFIED . THE RUNGE-KUTTA INTEGRATION  SCHEME
C  IS SENSITIVE TO STEP SIZE AND H MUST BE CHOSEN CAREFULLY
      H=0.030$ X=0.0
C
    2 NT=0
C
C     THE RIGHT-HAND SIDE OF THE DIFFERENTIAL EQUATIONS
C                  DY/DX = F(Y,X)
C     IS EVALUATED NEXT
C
    6 MAGSQ=Y(4)*Y(4)+Y(5)*Y(5)
      QM=1.0
      MAG=SORTF(MAGSQ)
      MOM=1.0-0.0749*X
      SB= Y(4)/MAG
      CB= Y(5)/MAG
      F(1)=Y(2)*Y(2)/Y(3)-GM/(Y(3)*Y(3))+0.140500 *SB/MOM
      F(2)=-Y(1)*Y(2)/Y(3)+0.140500 *CB/MOM $ F(3)=Y(1)
```

```
      F(4)=Y(2)*Y(5)/Y(3)-Y(6)
      F(5)=-2.0*Y(2)*Y(4)/Y(3)+(Y(1)*Y(5))/Y(3)
      F(6)=((Y(2)*Y(2)-2.0*GM/Y(3))*Y(4)-Y(1)*Y(2)*Y(5)          )/
     *(Y(3)*Y(3))
C
    3 S=RKLDEQ(N,Y,F,X,H,NT)
      IF(S-1.0)79,6,4
C
C  IF THE FINAL TIME IS UNKNOWN IT IS USED TO STOP THE INTEGRATION.
C  IF THE FINAL TIME IS KNOWN IT IS USED HERE
4     IF(X.QE.XI(3))5.2
    5 CONTINUE
C
C  THE WEIGHING FUNCTION IS THE SUM OF THE SQUARES OF THE
C  ERROR IN THE FINAL CONDITIONS. HOWEVER THE SUM OF THE ABSOLUTE VALUE
C  OF THE ERRORS CAN ALSO BE USED. ALSO THE ERRORS CAN BE WEIGHTED IF
C  THEY DIFFER IN IMPORTANCE.
C
C     A WEIGHING FUNCTION OF FINAL CONDITIONS MUST BE FORMULATED  HERE
      G= Y(1)*Y(1)+(Y(2)-0.8098)*(Y(2)-0.8098)+(Y(3)-1.525)*
     *(Y(3)-1.525)
      IF(Y(1).LT.-20.0) G=100.0
      IF(Y(1).QT.20.0)  G=100.0
   79 CONTINUE
      PRINT 10.G
10    FORMAT(X17HOBJECTVNE FUNC =       ,   E20.10)
      DO 20   J=1,N
20     PRINT   25,J,Y(J)
25    FORMAT(X3HY( ,I2,3H) =,  E20.10)
      DO 30 J=1.K
   30 PRINT 35 ,J,XI(J)
35    FORMAT(X4HXI( ,I2,3H) =,  E20.10)
      RETURN
      END
```

```
      FUNCTION RKLDEQ(N,Y,F,X,H,NT)                                  RKLDQ  1
C     D2 UCSD RKLDEQ                                                   F 63
C     MODIFIED MAY 1963 (Q REMOVED FROM CALLING SEQUENCE)
C     TEST OF ALGOL ALGORITHM
      DIMENSION Y(10),F(10),Q(10)                                    RKLDQ  2
C     REAL X,H--INTEGER N,NT--COMMENT--BEGIN INTEGER I,J,L-REAL A
      NT=NT+1                                                        RKLDQ  3
      GO TO (1.2,3,4),NT                                             RKLDQ  4
C     GO TO S(NT)
 1    DO 11 J=1,N                                                    RKLDQ  5
 11   Q(J)=0.                                                        RKLDQ  6
      A=.5                                                           RKLDQ  7
      X=X+H/2.                                                       RKLDQ  8
      GO TO 5                                                        RKLDQ  9
```

```
2     A=.29289321881                                                RKLDQ 10
      GO TO 5                                                       RKLDQ 11
3     A=1.7071067812                                                RKLDQ 12
      X=X+H/2.                                                      RKLDQ 13
      GO TO 5                                                       RKLDQ 14
4     DO 41 I=1,N                                                   RKLDQ 15
41    Y(I)=Y(I)+H*F(I)/6.-Q(I)/3.                                   RKLDQ 16
      NT=0                                                          RKLDQ 17
      RKLDEQ=2.                                                     RKLDQ 18
      GO TO 6                                                       RKLDQ 19
5     DO 51 L=1,N                                                   RKLDQ 20
      Y(L)=Y(L)+A*(H*F(L)-Q(L))                                     RKLDQ 21
51    Q(L)=2.*A*H*F(L)*(1.-3.*A)*Q(L)                               RKLDQ 22
      RKLDEQ=1.                                                     RKLDQ 23
6     CONTINUE                                                      RKLDQ 24
      RETURN                                                        RKLDQ 25
      END                                                           RKLDQ 26
```

```
      PROGRAM PENKT
C     THIS PROGRAM WILL SOLVE A TWO-POINT BOUNDARY VALUE PROBLEM
C     USING A DIRECT SEARCH SCHEME  (NELC DIRECT) AND
C     THE INTEGRATION SCHEME  IS THE FOURTH ORDER RUNGE-KUTTA
C     AS MODIFIED BY GILL
C
C     THE XI ARE THE INITIAL CONDITIONS THAT ARE UNKNOWN AND WHICH
C           ARE SPECIFIED INITIALLY IN THE PROGRAM DRIVER
C
C     K REPRESENTS THE NUMBER OF          UNKNOWN VARIABLES.
C     N = THE NUMBER OF DIFFERENTIAL EQUATIONS TO BE INTEGRATED
C
C     THE DELTA  REPRESENTS A STEP CHANGE IN THE INITIAL CONDITIONS
C     AND WHICH IS USED BY THE SEARCH ROUTINE.
C     THE DELTA SHOULD BE LARGE INITIALLY. THEN IF RESULTS ARE NOT
C     SATISFACTORY,DELTA  CAN BE REDUCED.
C
C     THE SUBROUTINE FN IS USED FOR INTEGRATION AND ALSO TO COMPUTE
C     A WEIGHING FUNCTION OF THE FINAL BOUNDARY CONDITIONS
C
C     A WEIGHING FUNCTION OF FINAL CONDITIONS MUST BE FORMULATED
C     THE WEIGHING FUNCTION IS USED BY THE DIRECT SEARCH SCHEME
C     TO DETERMINE THAT THE FINAL CONDITIONS ARE MET
C     THE FINAL CONDITIONS ARE WRITTEN AS EQUALITIES   P(Y)=0.0
C     THE WEIGHING FUNCTION IS THE SUM OF THE SQUARES
C      G=P(Y(1))*P(Y(1))+P(Y(2))*P(Y(2)) + ETC.
C
C     IF THE FINAL TIME IS UNKNOWN, AN ADDITIONAL VARIABLE XI(K  ) IS
C     USED TO REPRESENT THE FINAL TIME AND
C     INSERTED IN THE INTEGRATION STOPPING CRITERION. OTHERWISE
C     THE TRUE FINAL VALUE OF THE INDEPENDENT VARIABLE IS USED
```

```
C
C      PROBLEMS CAN BE ENCOUNTERED WITH STEP-SIZE DELTAI AND THE
C      INTEGRATION STEP-SIZE H. THE LOCAL VS GLOBAL PROBLEM IS PRESENT
C            OCCURRED AND IT IS NECESSARY TO RESTART THE      ROUTINE
C      WITH NEW INITIAL CONDITIONS TO GUARANTEE A GLOBAL OPTIMUM SOLUTION.
C
C      ADVANTAGE OF THIS ROUTINE IS THAT IT ONLY REQUIRES INTEGRATION
C      OF THE DIFFERENTIAL EQUATION .    NO PERTURBATION EQUATIONS
C      ARE REQUIRED.
C
       DIMENSION EX(12), XI(12),OX(12),RX(12),PX(12)
       EXTERNAL RKLDEQ
                          XI(1)=0.52
                          XI(2)=0.3
                          XI(3)=3.03
C    K= NO. OF UNKNOWNS
                          K=3
C           INITIAL DELTA
       DELTA=0.05
       DEL=0.001
        DELF=1.0E-06
       RHO= 0.625
    13 CALL DIRECT(F  ,K,RHO,DEL,DELF, XI,DELTA,EX,OX,RX,PX)
       END



       SUBROUTINE FN(G,K,XI)
       DIMENSION XI(K)
C
C   THESE QUANTITIES ARE NEEDED FOR INTEGRATION. THE ARRAYS HAVE THE
C   SAME ORDER AS THE DIFFERENTIAL EQUATIONS
       DIMENSION Y(6),F(6)
       TYPE REAL MAGSQ,MAG,MOM
C
       EXTERNAL RKLDEQ
            N=6
C
C      THE KNOWN INITIAL CONDITIONS ARE SPECIFIED HERE
       Y(1)=0.0
       Y(2)=1.0
       Y(3)=1.0
       Y(4)=XI(1)
       Y(5)=XI(2)
       Y(6)=1.0
C
C   THE INITIAL VALUE OF X, THE INDEPENDENT VARIABLE AND THE     INTEGRATION
C   STEP SIZE H ARE SPECIFIED . THE RUNGE-KUTTA INTEGRATION  SCHEME
C   IS SENSITIVE TO STEP SIZE AND H MUST BE CHOSEN CAREFULLY
       H=0.030$ X=0.0
```

```
C
    2 NT=0
C
C     THE RIGHT-HAND SIDE OF THE DIFFERENTIAL EQUATIONS
C                    DY/DX = F(Y,X)
C     IS EVALUATED NEXT
C
    6 MAGSQ=Y(4)*Y(4)+Y(5)*Y(5)
      GM=1.0
      MAG=SQRTF(MAGSQ)
      MOM=1.0-0.0749*X
      SB= Y(4)/MAG
      CB= Y(5)/MAG
      F(1)=Y(2)*Y(2)/Y(3)-GM/(Y(3)*Y(3))+0.140500 *SB/MOM
      F(2)=-Y(1)*Y(2)/Y(3)+0.140500 *CB/MOM $ F(3)=Y(1)
      F(4)=Y(2)*Y(5)/Y(3)-Y(6)
      F(5)=-2.0*Y(2)*Y(4)/Y(3)+(Y(1)*Y(5))/Y(3)
      F(6)=((Y(2)*Y(2)-2.0*GM/Y(3))*Y(4)-Y(1)*Y(2)*Y(5)        )/
     *(Y(3)*Y(3))
C
    3 S=RKLDEQ(N,Y,F,X,H,NT)
      IF(S-1.0)79,6,4
C
C  IF THE FINAL TIME IS UNKNOWN IT IS USED TO STOP THE INTEGRATION.
C  IF THE FINAL TIME IS KNOWN IT IS USED HERE
4     IF(X.GE.XI(3))5,2
    5 CONTINUE
C
C  THE WEIGHING FUNCTION IS THE SUM OF THE SQUARES OF THE
C  ERROR IN THE FINAL CONDITIONS. HOWEVER THE SUM OF THE ABSOLUTE VALUE
C  OF THE ERRORS CAN ALSO BE USED. ALSO THE ERRORS CAN BE WEIGHTED IF
C  THEY DIFFER IN IMPORTANCE.
C
C     A WEIGHING FUNCTION OF FINAL CONDITIONS MUST BE FORMULATED  HERE
      G= Y(1)*Y(1)+(Y(2)-0.8098)*(Y(2)-0.8098)+(Y(3)-1.525)*
     *(Y(3)-1.525)
      IF(Y(1).LT.-20.0) G=100.0
      IF(Y(1).QT.20.0)  G=100.0
   79 CONTINUE
      PRINT 10,G
10    FORMAT(X17HOBJECTIVE FUNC =        ,   E20.10)
      DO 20   J=1,N
20     PRINT   25,J,Y(J)
25    FORMAT(X3HY( ,I2,3H) =,  E20.10)
      DO 30 J=1,K
   30 PRINT 35 ,J,XI(J)
35    FORMAT(X4HXI( ,I2,3H) =,  E20.10)
      RETURN
      END
```

```
      PROGRAM TPBVPSCH
C     THIS PROGRAM WILL SOLVE A TWO-POINT BOUNDARY VALUE PROBLEM
C     USING A DIRECT SEARCH SCHEME  (UNIVAR)  AND
C     THE INTEGRATION SCHEME  IS THE EXTRAPOLATION METHOD BY
C     BOLESCH AND STOER
C
C     THE XI ARE THE INITIAL CONDITIONS THAT ARE UNKNOWN AND WHICH
C            ARE SPECIFIED INITIALLY IN THE PROGRAM DRIVER
C
C     K REPRESENTS THE NUMBER OF          UNKNOWN VARIABLES.
C     N = THE NUMBER OF DIFFERENTIAL EQUATIONS TO BE INTEGRATED
C
C     THE DELTAI REPRESENTS A STEP CHANGE IN THE INITIAL CONDITIONS
C     AND WHICH IS USED BY THE SEARCH ROUTINE.
C     THE DELTAI SHOULD BE LARGE INITIALLY. THEN IF RESULTS ARE NOT
C     SATISFACTORY,DELTAI CAN BE REDUCED.
C
C     THE SUBROUTINE FN IS USED FOR INTEGRATION AND ALSO TO COMPUTE
C     A WEIGHING FUNCTION OF THE FINAL BOUNDARY CONDITIONS
C
C     A WEIGHING FUNCTION OF FINAL CONDITIONS MUST BE FORMULATED
C     THE WEIGHING FUNCTION IS USED BY THE DIRECT SEARCH SCHEME
C     TO DETERMINE THAT THE FINAL CONDITIONS ARE MET
C     THE FINAL CONDITIONS ARE WRITTEN AS EQUALITIES   P(Y)=0.0
C     THE WEIGHING FUNCTION IS THE SUM OF THE SQUARES
C      G=P(Y(1))*P(Y(1))+P(Y(2))*P(Y(2)) + ETC.
C
C     IF THE FINAL TIME IS UNKNOWN , AN ADDITION   VARIABLE  XI(K  ) IS
C     USED TO REPRESENT THE FINAL TIME AND
C     INSERTED IN THE INTEGRATION STOPPING CRITERION. OTHERWISE
C     THE TRUE FINAL VALUE OF THE INDEPENDENT VARIABLE IS USED
C
C     PROBLEMS CAN  BE ENCOUNTERED WITH STEP-SIZE DELTAI AND THE
C     INTEGRATION STEP-SIZE H. THE LOCAL VS GLOBAL PROBLEM IS PRESENT
C            OCCURRED AND IT IS NECESSARY TO RESTART THE      ROUTINE
C     WITH NEW INITIAL CONDITIONS TO GUARANTEE A GLOBAL OPTIMUM SOLUTION.
C
C     ADVANTAGE OF THIS ROUTINE IS THAT IT ONLY REQUIRES INTEGRATION
C     OF THE DIFFERENTIAL EQUATION .    NO PERTURBATION EQUATIONS
C     ARE REQUIRED.
C
      DIMENSION XI(4),X(4),DP(4),DEL(4),XR(4)
      EXTERNAL F
C
                        XI(1)=0.7
                        XI(2)=0.3
                        XI(3)=3.03
C
C   K= NO. OF UNKNOWNS
                        K=3
```

```
C
C          INITIAL DELTA
                          DELTAI=0.04
      RHO=1.618
      CALL UNIVAR(K,XI,RHO,DELTAI,X,DP,DEL,XR,FR)
      END



      SUBROUTINE FN(G,K,XI)
      DIMENSION XI(K)
      EXTERNAL F
C
C  THESE QUANTITIES ARE NEEDED FOR INTEGRATION, THE ARRAYS HAVE THE
C  SAME ORDER AS THE DIFFERENTIAL EQUATIONS
      DIMENSION Y(6)
C
      TYPE REAL MAGSQ,MAG,MOM
C
C     THE KNOWN INITIAL CONDITIONS ARE SPECIFIED HERE
           N=6
      Y(1)=0.0
      Y(2)=1.0
      Y(3)=1.0
      Y(4)=XI(1)
      Y(5)=XI(2)
      Y(6)=1.0
C
C  THE INITIAL VALUE OF X, THE INDEPENDENT VARIABLE AND THE     INTEGRATION
C  STEP SIZE H ARE SPECIFIED . THE RUNGE-KUTTA INTEGRATION  SCHEME
C  IS SENSITIVE TO STEP SIZE AND H MUST BE CHOSEN CAREFULLY
C  H IS CHANGED BY THE DIRECT SEARCH ROUTINE HERE
      X=0.0$ H=XI(3)$
C
C
C     THE RIGHT-HAND SIDE OF THE DIFFERENTIAL EQUATIONS
C                 DY/DX = F(Y,X)
C     IS EVALUATED NEXT
C
1     CALL DIFF SYS(6,F,1.E-5,H,X,Y)
C
C  IF THE FINAL TIME IS UNKNOWN IT IS USED TO STOP THE INTEGRATION.
C  IF THE FINAL TIME IS KNOWN IT IS USED HERE
    5 CONTINUE
C
C  THE WEIGHING FUNCTION IS THE SUM OF THE SQUARES OF THE
C  ERROR IN THE FINAL CONDITIONS. HOWEVER THE SUM OF THE ABSOLUTE VALUE
C  OF THE ERRORS CAN ALSO BE USED. ALSO THE ERRORS CAN BE WEIGHTED IF
C  THEY DIFFER IN IMPORTANCE.
```

```
C
C     A WEIGHING FUNCTION OF FINAL CONDITIONS MUST BE FORMULATED  HERE
      G= Y(1)*Y(1)+(Y(2)-0.8098)*(Y(2)-0.8098)+(Y(3)-1.525)*
     *(Y(3)-1.525)
   79 CONTINUE
      PRINT 10,G
10    FORMAT(X17HOBJECTIVE FUNC =        ,   E20.10)
      DO 20   J=1,N
20     PRINT   25,J,Y(J)
25    FORMAT(X3HY( ,I2,3H) =,  E20+10)
      DO 30 J=1,K
   30 PRINT 35 ,J,XI(J)
35    FORMAT(X4HXI( ,I2,3H) =,  E20.10)
      RETURN



      SUBROUTINE F(X,Y,Z)
      TYPE REAL MAGSQ, MAG,MOM
      DIMENSION Y(6),Z(6)
    6 MAGSQ=Y(4)*Y(4)+Y(5)*Y(5)
      GM=1.0
      MAG=SQRTF(MAGSQ)
      MOM=1.0-0.0749*X
      SB= Y(4)/MAG
      CB= Y(5)/MAG
      Z(1)=Y(2)*Y(2)/Y(3)-GM/(Y(3)*Y(3))+0.140500 *SB/MOM
      Z(2)=-Y(1)*Y(2)/Y(3)+0.140500 *CB/MOM $ Z(3)=Y(1)
      Z(4)=Y(2)*Y(5)/Y(3)-Y(6)
      Z(5)=-2.0*Y(2)*Y(4)/Y(3)+(Y(1)*Y(5))/Y(3)
      Z(6)=((Y(2)*Y(2)-2.0*GM/Y(3))*Y(4)-Y(1)*Y(2)*Y(5)          )/
     *(Y(3)*Y(3))
      RETURN
      END
```