Direct Trajectory Optimization Using Nonlinear Programming and Collocation

C.R. Hargraves* and S.W. Paris*
Boeing Aerospace Company, Seattle, Washington

An algorithm for the direct numerical solution of an optimal control problem is given. The method employs cubic polynomials to represent state variables, linearly interpolates control variables, and uses collocation to satisfy the differential equations. This representation transforms the optimal control problem to a mathematical programming problem which is solved by sequential quadratic programming. The method is easy to program for a very general trajectory optimization problem and is shown to be very efficient for several sample problems. Results are compared with solutions obtained with other methods.

Introduction

A DIRECT trajectory optimization method that represents state and control variables by piecewise polynomials is described in this paper. An implicit integration scheme based on Hermite interpoiation is used to convert the optimal control problem to a nonlinear programming problem. The primary advantage of this method is that it is much easier to extend to general problems involving path constraints, discontinuous states, and control inequalities.

Numerous approaches are possible for direct trajectory optimization. The authors have described in earlier papers a method employing piecewise Chebyshev polynomials. Penalty functions were used to convert optimal control problems to unconstrained parameter optimization problems which were solved with a full second-order modified Newton algorithm.[1-3] The basic ideas used in this approach were described earlier by Johnson[4] and Hahn and Johnson.[5] A similar approach was described by Balakrishnan.[6] Additional direct approaches were described by Kelley,[7] Rader and Hull,[8] and Brauer et al.[9]

Birkoff and de Boor[10] hae shown that piecewise polynomials have many desirable properties for representing smooth curves. de Boor" and Russell and Shampine[12] described implicit methods for the solution of boundary value problems in ordinary differential equations using piecewise polynomials. Dickmanns and Wells[13] and Dickmanns[14,15] applied the maximum principal to reduce the optimal control problem to a boundary value problem which they solved using a collocation method; the method employed Hermite interpolation and collocation.[16]

The solution of optimal control problems using nonlinear programming has been demonstrated for a number of distinctive methods. Betts et al.[17] described a procedure in which a problem is divided into a series of subproblems which are solved by an indirect method. Boundary conditions on the subproblems are solved by nonlinear programming. Kraft[18] parameterized the control variables and integrated explicitly the equations. A direct and an indirect method were utilized and compared. Evtushenko[19] used nonlinear programming to define a variety of optimal control algorithms. Renes[20] proposed a procedure which is similar to that described in the next section. Renes did not give any numerical results and a number of details are different (e.g., different order splines are used, the collocation points are different, the independent variables were spline coefficients, etc.).

Received June 9, 1986, revision received Dec. 10, 1986.

*Senior Specialist Engineers. Members AIAA.

Mathematical Method

Statement of the Problem

Trajectory optimization problems that can be described by a sequence of vehicle/flight stages are considered. The control functions $u(t)$, the time points delimiting the various stages (E,, called events), and the vehicle design parameters (w) are sought which minimize a given performance index

$$J=\phi[x(E),u(E),\omega,E] \tag{1}$$

The ith stage is assumed to be a dynamical system subject to the differential constraint

$$x' = f^i(x,u,\omega,t) \tag{2}$$

on the interval $t\epsilon[E_i,E_{i+1}]$, where x is a vector of states governed by first-order differential equations, u a vector of controls (e.g., pitch angle, and ω a vector of design parameters (e.g., planform area, rocket nozzle diameter, etc.).
The prime denotes differentiation with respect to time. At each event E_i, nonlinear boundary conditions may be imposed of the form

$$\ell_{BE} \le a^i[x(E_i),(E_i),\omega] \le U_{BE} \tag{3}$$

where ℓ_{BE} is the vector lower-limit on the boundary conditions and U_{BE} is the upper-limit. Equality constraints are possible by setting the appropriate elements of ℓ_{BE} and U_{BE} equal to each other. Path constraints of the form

$$\ell_{PC} \le h_k(x,u,\omega,t) \le U_{PC} \tag{4}$$

may also be imposed on the system. Denote the collection of events by $E^T=(E_1,E_2,\ldots,E_{N+1})$, where N is the number of stages.

The functions f^i, a^i, and h_k are assumed to be smooth (class C_2 or better) within a given stage. However, no continuity restrictions are imposed at events. Specifically, discontinuities of the form

$$x(E_i^+) = x(E_i) + \delta_i \tag{5}$$

are allowed at the events. The δ_i's may be fixed quantities or may be included in the design parameter set w. Equations (5) account for physical events such as weight jettisons and impulsive velocity changes,

Method of Solution

The approach is to reduce the optimal control problem stated in the previous section to a nonlinear programming problem. To do this, the functions $x(t)$ and $u(t)$ are represented by piecewise polynomials and collocation is used to satisfy Eq. (2).

The length of each state is defined as $T_{si}=E_{i+1}-E_i$, as shown in Fig. 1. For each stage, the interval $[E_i,E_{i+1}]$ is subdivided into N_s segments. Let the ratio of the length of the jth segment to T_{si} be denoted as τ_{ji}. Thus, the length of the jth segment in the ith stage is $\tau_{ji}T_{si}$. Using Hermite interpolation, cubic polynomials are defined for each state on each segment using values of the states at the nodes (the boundaries at the segments) and the state time derivatives, as defined by the equations, at the nodes. The values of the states are then selected (by nonlinear programming) to force the interpolated derivatives to agree with the differential equations at the center of the segment. This procedure is illustrated in Fig. 2.

The basic procedure can be derived as follows. Let the states $\mathbf{x}$ be represented on each segment by cubics of the form

$$\mathbf{x}=C_0+C_1S+C_2S^2+C_3S^3 \tag{6}$$

where to simplify the argument, the segment length S is transformed such that $S\epsilon[0, 1]$. Let $x(0)=x_1$, $x(1)=x_2$, $dx/ds(0)=\dot{x}_1$, $dx/ds(1)=\dot{x}_2$. Differentiating Eq. (6) and evaluating at 0 and 1 gives

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix} \tag{7}$$

Inverting the 4×4 matrix gives

$$\begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -3 & -2 & 3 & -1 \\ 2 & 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix} \tag{8}$$

Now using Eq. (8), evaluating Eq. (6) at $S=1/2$, and transforming to segment length $T=\tau_{ji}$, T_{si}, we see that the interpolated value of $\mathbf{x}$ at the center of the segment is

$$x_c=(x_1+x_2)/2+T(f_1-f_2)/8 \tag{9}$$

where f_i is Eq. (2) evaluated at $\mathbf{x}$, (the superscript denoting the stage was dropped for simplicity). (Note $\dot{x}_1=Tf[x_i, u(t_i), t_i, \omega]$.) In the same way, the slope

$$x_c'=-3(x_1-x_2)/2T-(f_1+f_2)/4 \tag{10}$$

is obtained. Evaluating Eq. (2) at x_c gives f_c. Define the defect at the center of the segment as

$$\begin{aligned} \Delta &= f_c-x_c' \\ &= f_c+3(x_1-x_2)/2T+(f_1+f_2)/4 \end{aligned} \tag{11}$$

x_1 and x_2 are varied to drive $\Delta=0$.

If the cubic polynomial is capable of representing the solution on the given segment, then selecting x_1 and x_2 to drive Δ to zero will produce an accurate approximation to the solution of Eq. (2).

The defects for each state evaluated at the center of each segment constitute a set of nonlinear algebraic equations which are a function of the states and controls at each node, the events $\mathbf{E}$, and the design parameters ω. Linear interpolation is used to obtain the controls at the segment centers. Initial experiments with cubic polynomial controls did not work well. Also, quintic polynomials for the solution of second-order differential equations (without reduction to the first order) were tried. This approach did not work as well as the cubics. The boundary conditions [Eq. (3)] and the constraints [Eq. (4)] evaluated at both the nodes and centers of the segments provide additional equations (constraints) to be satisfied. Equations (5), connecting the segments constitute additional linear equations to be satisfied. Now, collect all of the independent variables into a single vector $\boldsymbol{P}$ defined by

$$P^T=[Z^T,E^T,\omega^T] \tag{12}$$

where

$$Z^T=(x_1^T,u_1^T,x_2^T,u_2^T,\dots,x_{N+1}^T,u_{N+1}^T)$$

Collecting all of the nonlinear equations into a single vector equation gives

$$C^T=[\Delta^T,B_N^T,H_N^T] \tag{13}$$

where

$\Delta^T=(\Delta_{11},\Delta_{12},\dots,\Delta_{ij}\dots)$
Δ_{ij} =defect for ith state at jth node
B_N =collection of all nonlinear relationships from Eq. (3)
H_N =collection of all nonlinear relationships from Eq. (4)

It is now noted that our payoff function J is a function of P only. The trajectory optimization problem stated above can be expressed as

$$\text{minimize}\ \phi(P)$$

subject to

$$\ell\le\left\{\begin{matrix} P \\ AP \\ C(P) \end{matrix}\right\}\le u \tag{14}$$

where N_P is the dimension of $\boldsymbol{P}$, $\boldsymbol{AP}$ is composed of all the linear relationships from Eqs. (3–5), ℓ and u are the upper and lower bounds. For equality constraints $\ell_i=u_i$, and inequality constraints are handled by $\ell_i=-\infty$ or $u_i=\infty$. Equation (14) constitutes a nonlinear programming problem.

A large amount of literature in recent years has been devoted to the solution of nonlinear programming problems. Recent treatments of the subject are presented by Gill et al.,[21] Evtushenko,[22] and Reklaitis.[23] These books contain long lists of references on earlier work in this field.

Programming Considerations

The computer code implementing the aforementioned procedure is called Nonlinear Programming for Direct Optimization of Trajectories (NPDOT). NPDOT uses a nonlinear programming package called NPSOL which was developed by the Systems Optimization Laboratory at Stanford University.[24] NPSOL uses sequential quadratic programming (SQP). SQP methods were popularized by Biggs,[25] Han,[26] and Powell.[27]

NPSOL requires as inputs the matrix A and an initial guess for $\boldsymbol{P}$. Subroutines must be provided which evaluate $\phi(P)$, $C(P)$, and their first derivatives with respect to $\boldsymbol{P}$. In the method described here, these derivatives are computed by finite difference. This task is greatly simplified by the fact that the defects depend on states at adjacent nodes only and only on the time length of the stage in which it occurs. The Jacobian has the banded structure shown in Fig. 3.

To obtain accurate solutions to realistic problems, some refinements must be added to the procedure. The key refinements involve problem scaling, partial derivative computation, data smoothing, node selection, and node refinement. A discussion of equation scaling as applied to optimization problems may be found in Ref. 28. As a minimum, the independent variables should be scaled so that they have similar mangitudes. The following scaling procedure worked well. (No systematic comparison with other approaches was attempted.) States and controls were scaled by

$$x_s(t) = [x(t) - x_N(t)]/S_x(t)$$

where x_N is the nominal trajectory and S_x is an estimate of an upper bound for $x - x_r$. Times were scaled by

$$T_s = T/S_T$$

where S_T is the estimated upper bound for T.

For realistic trajectory problems, the computation of analytical derivatives can be a formidable task. It is usually preferable to compute derivatives directly by finite difference. Partial derivatives must be accurate to obtain convergence of the nonlinear programming algorithm. An algorithm for selecting the finite-difference perturbation size is given in Ref. 28 (pp. 341–345). The code described here uses this algorithm initially and retains the perturbation step sizes for subsequent iterations.

If tabular data are used, it is important that they be smooth. Smoothing data without changing the physical meaning can be a difficult problem. Good results were obtained with splines [see Ref. 29)] for tables with one independent variable. A satisfactory solution for multivariable tables has not yet been found.

The smoothness of data is closely related to node spacing since, in general, rapid changes in tabular functions will be acceptable only if there are sufficient nodes in the region of rapid change. This can be difficult to accomplish since the nodes are distributed in thie whereas tabular data may be functions of other variables such as Mach number of altitude. Regions where large changes occur as a function of Mach number for the nominal and optimal solution may be very different regions in time. An initial node selection can be made based on the nominal. Unless a nominal that is close to opitimal is known, an adaptive procedure is requried.

The node selection can then be updated after initial convergence or when optimizer behavior indicates possible problems due to the polynomial distribution. Dickmanns[14] has described several methods for refining node selection. The basic idea is to represent each segment by a higher-order method (polynomial, numerical integration, etc.) and add nodes when disagreement is found. After several refinement cycles, the adequacy of the total representation can be treated by explicit numerical integration of the trajectory using the polynomial control function and comparing the results with the polynomial representation of the trajectory.

Numerical Examples

Several test problems that have been solved with the new technique are described in this section. The problems range from simple analytical problems to contemporary aerospace performance-prediction problems. A stepped approach was used to validate NPDOT against known results, to gain experience before attempting new unknown solutions, and to demonstrate the method's inherent flexibility. The Chebyshev Trajectory Optimization Program (CTOP), described in Ref. 1, served as the performance reference and "truth" model for NPDOT. The academic problems were also compared to solutions using the maximum principle.[30]

The following problems have been solved using NPDOT: 1) Van der Pohl (Ref. 31 and 32), 2) Brachistochrone (Ref. 33), 3) minimum time interceptor (Ref. 34), 4) advanced booster. The results of NPDOT agreed with the previously published results for problems 1–3. A comparison of the solution times between NPDOT and CTOP is given in Fig. 4. The times shown in Fig. 4 are adjusted to be comparable for the Cray X-MP/24. (For approximate comparison with other computers, the X-MP has a clock cycle time of 9 ns.) Discussions of prob-

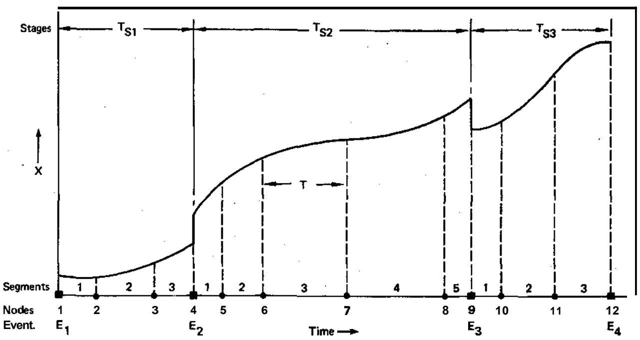

Fig. 1 Piecewise polynomial representation.

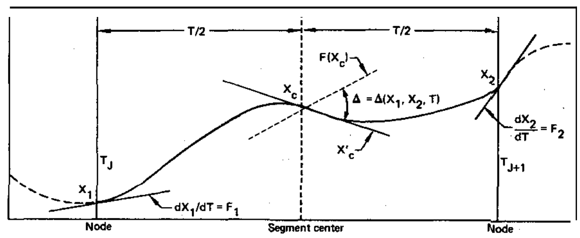

Fig. 2 Implicit integration illustration.

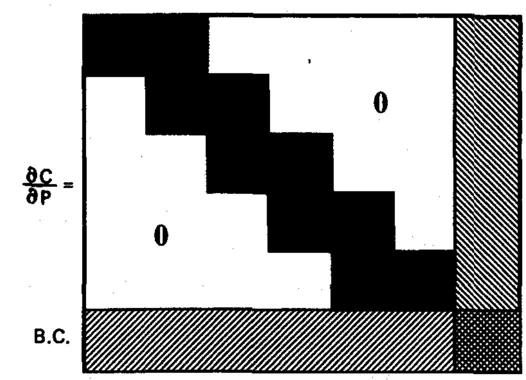

Fig. 3 Jacobian structure.

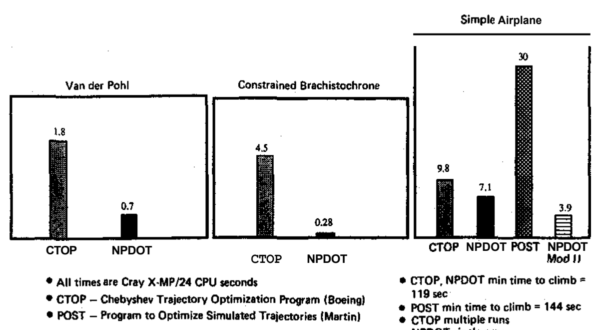

Fig. 4 CPU time comparison.

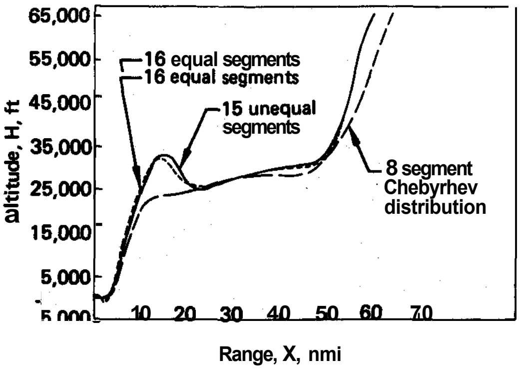

Fig. 5 Supersonic interceptor minimum-time climb trajectories.

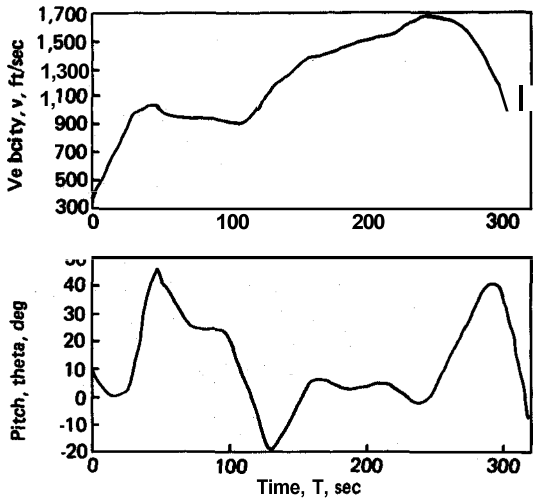

Fig. 6 Supersonic interceptor control function and velocity profile.

lems 3 and 4 follow to provide some insight into the method as applied to typical aerospace flight mechanics problems.

Supesonic Interceptor Minimum-Time Climb

This problem involves finding the pitch function $\theta(t)$ to take a supersonic interceptor from sea level, Mach = 0.38, to an altitude of 20 km, Mach= 1.0, in minimum time. The initial weight was 42,000 lbs. This problem was proposed and solved by Bryson.[34] The equations of motion were those for planar flight above a flat Earth. The aerodynamic coefficients were functions of Mach number. The thrust was a function of Mach number and altitude. The specific fuel consumption was constant. The values for the defining aerodynamic and propulsion data were taken from Ref. 35. The atmospheric density and speed of sound were taken from the NACA-1962 atmosphere model. The initial-state values were obtained by linear interpolation between the initial and final values; the control (pitch) was set to 0.18 rad. The initial estimate of the free variables was as follows: final range equal to 360,000 ft and the final mass set to 1204 slugs.

A sequence of solutions is shown on Fig. 5. A preliminary solution, shown by short dashes, was obtained with 16 equal segments. One can observe that this trajectory exploits the discretization to meet the constraints at the nodes and centers while violating such things as the altitude constraint in between. To counter this activity, an eight-segment solution with a Chebyshev node distribution was generated. This provided the trajectory, shown by long dashes, which had a time to climb of 325.2 s. This was not as good as the previous CTOP' results, so a solution was obtained with 15 segments. To obtain the node distribution, the absolute value of the difference between the time derivatives produced by the equations of motion and time derivatives of the polynomials were integrated to acquire an error estimate. This was done over the eight-segment trajectory. Five nodes were added to the center of the segments which were producing the majority of the error. A solution was generated for this 13-segment distribution and another error estimate was produced. Two additional nodes were added to yield a 15-unequal-segment distribution. This distribution yielded a time to climb of 317.3 s and produced a trajectory, shown as a solid line in Fig. 5, which matched our previously obtained CTOP result and published energy state results.[35] The control function and velocity profile are shown in Fig. 6. The airplane accelerates to a high subsonic speed, then climbs at a constant Mach number followed by a dive through Mach one and beyond, finishing up by a zoom climb to the final altitude. In the initial attempts for a solution, simple piecewise linear and piecewise cubit interpolation for the aero/propulsion data were used. This did not work well. The program could not reduce the defects or obtain a small gradient. This is consistent with previous trajectory optimization experience. These problems were resolved by using the taut spline and tensor spline packages as defined and coded by de Boor.[36]

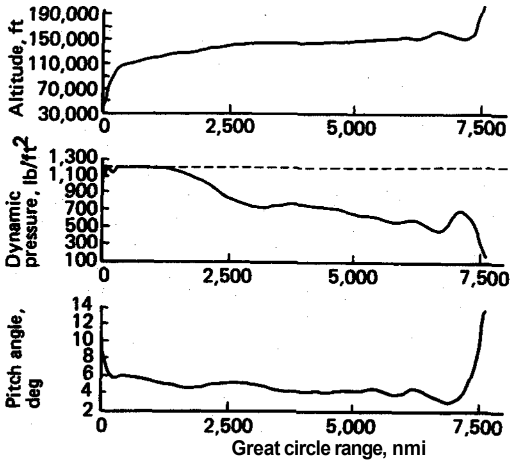

Fig. 7 Advanced booster trajectory.

Advanced Booster

This problem involves maximizing the final weight that can be injected into low Earth orbit by an advanced booster which utilizes airbreathing propulsion. The propulsion system has several discrete phases which depend on the vehicle's Mach number. The first phase spans from Mach 0–2. The thrust and specific impulse were modeled as tabular functions of Mach number. The second propulsion phase goes from Mach 2–20. The thrust was modeled as a thrust coefficient multiplied by dynamic pressure. The thrust coefficient was a function of Mach number and angle of attack. The specific impulse is a function of Mach number. The final propulsion phase, Mach numbers greater than 20, was a rocket in which both the thrust and specific impulse were constant.

The equations of motion were for three-dimensional flight above a spherical nonrotating Earth. Limits were imposed on dynamic pressure, altitude, and angle of attack. The second propulsion phase being proportional to dynamic pressure tended to drive the trejectory solutions to very high dynamic pressure values. The aerodynamics for this vehicle were done in body axis form, C_A and $C_N = C_{N_\alpha} \cdot \alpha$. C_A and C_{N_α} were functions of Mach number.

Initial attempts to generate trajectories using a conventional explicit integrating program for this class of vehicle required intensive "man-in-the-loop" interactions. The strategy employed was to accelerate the vehicle to a dynamic pressure q, limit and fly that limit until a predetermined velocity was reached. At that time, a rocket burn is performed which allows the vehicle to coast in a transfer orbit with the appropriate apogee. A single apogee burn is then used to circularize the orbit. All attempts to optimize this trajectory with an explicit integrating parameter-optimization program failed. The best results were achieved by using standard trade studies. Attempts to use CTOP on this problem resulted in very slowly converging trajectories. When NPDOT was applied to this problem, only the initial airbreathing portions and rocket burn were optimized. This results in the problem of maximizing the weight injected into a predefined transfer orbit. The transfer orbit is defined by a fixed velocity and flight path angle at a given altitude. This was done to benchmark NPDOT with the existing solution. A coverged NPDOT trajectory was obtained using 22 segments and resulted in a 10% gain in weight over the baseline trajectory. The trajectory, pitch profile, and dynamic pressure as a function of time are shown in Fig. **7**. It is interesting to note that the optimum trajectory does not stay on the dynamic pressure constraint but rather initially rides it and then gets off. To maximize acceleration, the vehicle wants to fly at maximum q; however, the vehicle's final conditions are orbital, and drag losses are largest at maximum q so there is a balance between the two.

Conclusions

A trajectory optimization method which uses an embedded collocation scheme in conjunction with mathematical programming has been described. The method has been used to solve a wide variety of test cases and found to be superior to other procedures in terms of cost and robustness.

Acknowledgment

This research was supported in part by the U.S. Air Force under Contract F33615-85-C-3009.

References

[1]Hargraves, C.R., Johnson, F.T., Paris, S.W., and Rettie, I., "Numerical Computation of Optimal Atmospheric Trajectories," ***AZAA Journal of Guidance and Control,*** Vol. 4, July–Aug. 1981, pp. 406–414.

[2]**Paris,** S.W., Joosten, B.K., and Fink, L.E., "Application of an Advanced Trajectory Optimization Method to Ramjet Propelled Missiles," ***Optimal Control Applications and Methods,*** Vol. 1, Oct.-Dec. 1980, pp. 319–334.

[3]**Hargraves,** C.R., "Numerical Computation of Optimal Atmospheric Trajectories Involving Staged Vehicles," AIAA Paper 82-0360, AIAA 20th Aerospace Sciences Meeting, Orlando, FL, Jan. 1982.

[4]Johnson, F.T., "Approximate Finite-Thrust Trajectory Optimization," ***AZAA Journal,*** Vol. 7, June 1969, pp. 993–997.

[5]**Hahn,** D.W., and Johnson, F.T., "Final Report for Chebyshev Trajectory Optimization Program (CHEBYTOP II)," The Boeing Co., Seattle, WS, Rept. D180-12916-1, prepared for NASA under Contract NAS2-5994, June 1971.

[6]**Balakrishnan,** A.V., "On a New Computing Method in Optimal Control," ***SZAM Journal on Control,*** Vol. 6, May 1968, pp. 149–173.

[7]**Kelley,** H.J., "Method of Gradients," ***Optimization Techniques,*** edited by ***G.*** Leitmann, Academic Press, New York, 1962, pp. 205–254.

[8]**Rader** J.E., and Hull, D.G., "Computation of Optimal Aircraft Trajectories Using Parameter Optimization Methods," ***Journal of Aircraft,*** Vol. 12, Nov. 1975, pp. 864–866.

[9]**Brauer,** G.L., Cornick, D.E., and Stevenson, R., "Capabilities and Applications of the Program to Optimize Simulated Trajectories (POST)," NASA CR-2770, Feb. 1977.

[10]Birkhoff, G. and de Boor, C.R., "Piecewise Polynomial Interpolation and Approximation," ***Approximation of Functions,*** edited by H.L. Garabedian, Elsevier, New York, 1965.

[11]de Boor, C.R., ***A Practical Guide to Splines,*** Springer-Verlag, New York, 1978.

[12]**Russell,** R.D. and Shampine, L.F., "A Collocation Method for Boundary Value Problems," ***Numerical Mathematics,*** Vol. 19, Jan. 1972, pp. 1–28.

[13]**Dickmanns,** F.D. and Wells, K.H., "Approximate Solution of Optimal Control Problems Using Third Order Hermite Polynomial Functions," ***Proceedings of the 6th Technical Conference on Optimization Techniques,*** Springer-Verlag, New York, IFIP-TC7, 1975.

[14]**Dickmanns,** E.D., "Efficient Convergence and Mesh Refinement Strategies for Solving General Ordinary Two-Point Boundary Value Problems by Collocated Hermite Approximation," 2nd IFAC Workshop on Optimization, Oberpfaff-Enhofen, FRG, Sept. 1980.

[15]**Dickmanns,** E.D., "Collocated Hermite Approximation of Third and Fifth Order with Automatic Mesh Refinement for the Solution of Optimal Control Boundary Value Problems," Hochschule der Bundeswehr Munchen, FRG, LRT/WE, Oct. 1979.

[16]**Dahlquist,** G., **Björck,** A., and Anderson, N., ***Numerical Methods,*** Prentice-Hall Englewood Cliffs, NJ, 1974, p. 285.

[17]**Betts,** J., Bauer, T., Hoffman, W., and Zondervan, K., "Solving the Optimal Control Problem Using a Nonlinear Programming Technique, Part 1: General Formulation," AIAA Paper 84-2037, Aug. 1984.

[18]Kraft, D., "On the Choice of Minimization Algorithms in Parametric Optimal Control, Springer-Verlag, New York, 1981.

[19]**Evtushenko,** Y.C., ***Numerical Optimization Techniques,*** Optimization Software, Inc., Pub. Div., New York, 1985, pp. 388–398.

[20]Renes, J.J., "On the Use of Splines and Collocation in a Trajectory Optimization Algorithm Based on Mathematical Programming," National Aerospace Lab., Amsterdam, the Netherlands, NLR-TR-78016 U, Sept. 1978.

[21]**Gill,** P.E., Murray, W., and Wright, M.H., ***Practical Optimization,*** Academic Press, New York, 1981.

[22]**Evtushenko,** Y.G., ***Numerical optimization Techniques,*** Optimization Software, Inc., Pub. Div., New York, 1985, pp. 196–364.

[23]**Reklaitis,** G.V., Ravindran, A., and Ragsdell, K.M., ***Engineering Optimization,*** Wiley, New York, 1983.

[24]**Gill,** P.E., Murray, W., Saunders, M.A., and Wright, M.H., "User's Guide for NPSOL (Version 2.1): A FORTRAN Package for Nonlinear Programming," Systems Optimization Lab., Dept. of Operations Research, Stanford Univ., Stanford, CA, Technical Rept. SOL 84-7, Sept. 1984.

[25]**Biggs,** M.C., "Constrained Minimization Using Recursive Equalty Quadratic Programming," ***Numerical Methods for Nonlinear Optimization,*** edited by F.A. Lootsma, Academic Press, London, 1972, pp. 411–428.

[26]**Han,** S.P., "Superlinearly Convergent Variable Metric Algorithms for General Nonlinear Programming Problems," ***Mathematical Programming,*** Vol. 11, 1976, pp. 263–282.

[27]**Powell,** M.J.D., "A Fast Algorithm for Nonlinearly Constrained Optimization Calculations," Univ. of Cambridge, England, Rept. DAMTP 77/NA 2, 1977.

[28]**Gill,** P.E., Murray, W., and Wright, M.H., ***Practical Optimization,*** Academic Press, New York, 1981, pp. 341–345; pp. 346–356.

[29]**de** Boor, C.R., ***A Practical Guide to Splines,*** Springer-Verlag, New York, 1978, pp. 299–314.

[30]**Pontryagin,** L.S., Boltyanskii, V.G., Gamkrelidze, R.V., and Mishchenko, E.F., ***The Mathematical Theory of Optimal Processes,*** Interscience, New York, 1962.

[31]**Bullock,** T.E., and Franklin, G.F., "A Second-Order Feedback Method for Optimal Control Computations," ***ZEEE Transactions on Automatic Control,*** Vol. AC-12, No. 6, Dec. 1967.

[32]**Willoughby,** J.K., "Adaptations of the Conjugate Gradient Method to Optimal Problems with Terminal State Constraints," Engineering Research Institute, Iowa State Univ. Technical Rept. ERI-62500, Feb. 1970.

[33]**Bryson,** A.E. and Ho, Y., ***Applied Optimal Control,*** Blaisdell Publishing, Waltham, MA, 1969, p. 81; 119–120.

[34]**Bryson,** A.E., and Denham, W.F., "A Steepest-Ascent Method for Solving Optimum Programming Problems," ***Journal of Applied Mechanics,*** Vol. 29, June 1962.

[35]**Bryson,** A.E., Desai, M.N., and Hoffman, W.C., "Energy-State Approximation in Performance Optimization of Supersonic Aircraft," ***Journal of Aircraft,*** Vol. 6, Nov.-Dec. 1969, pp. 481–488.

[36]**de** Boor, C.R., ***A Practical Guide to Splines,*** Springer-Verlag, New York, 1981.