A model comparison of a supersonic aircraft minimum time-to-climb problem

by

Shaw Yang Ong

A Thesis Submitted to the
Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE

Major: Aerospace Engineering

Iowa State University
Ames, Iowa

1986

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

Aircraft performance optimization has been a subject of considerable attention over the past several years and, with the advent of high-speed aircraft, it is even more so now. One area of interest to many investigators is that of the minimum time-to-climb problem and, in particular, the minimum time-to-climb problem for supersonic aircraft. This area has received much attention because today's aircraft are better built, have larger flight envelopes, and thus have higher performance capability. This problem is especially important for the intercept mission.

To study this problem, mathematical models are needed to describe the aircraft and its motion. For subsonic aircraft, the quasi-steady approximation, where accelerations are neglected completely, is adequate for performance analysis. However, for supersonic aircraft the accelerations are so great that if accurate performance analysis is desired they cannot be neglected. Hence, a more complex model is required. However, as the complexity of the dynamic model increases, so does the amount of difficulty and computer expense required to determine the optimal flight path of the aircraft. This is clearly seen in Bryson's and Denham's investigation [1] in which they use a four-state variable model with velocity, flight path angle, height, and mass as the dependent variables.

To simplify the complexity of the dynamic model for high performance aircraft, Lush [2] proposes an energy approach. Rutowski [3] applies this approach to the minimum time-to-climb problem. Here, the aircraft performance problem is considered from the point of view of the balance that must exist between the potential energy and the kinetic energy change of the aircraft. Based upon this idea, Bryson et al. [4] present a very simple approximation model known now as the energy-state model. In this model, only one state variable is used, and that variable is the total energy per unit mass. This model saves much time and expense, but it leads to unrealistic slope discontinuities in velocity and altitude in the region of the dive and zoom-climb.

The minimum time-to-climb problem has also been treated by Garfinkel [5], Miele [6], Landgraf [7], Kelley [8], and Ardema [9]. These investigators contribute to the development of numerical techniques to enhance the solving of minimum time-to-climb problems. The results have been encouraging. Ardema [9] applies singular perturbation techniques to this problem and shows that the computational cost of the singular perturbation solution is considerably less than that of Bryson's steepest ascent solution [1]. However, the initial preparation time to set up the problem is high.

A better approach is to apply parameter optimization techniques to aircraft trajectories. This approach allows a lot of flexibility in changing models, performance indices, and constraints and yet requires only modest computer expense. Here, the optimal control problem is reformulated as a parameter optimization problem by choosing a form for the control time history which contains a finite number of parameters. Minimization then takes place over this set of parameters. Rader and Hull [10] demonstrate this technique on the minimum time-to-climb problem, and it has proven to be very useful. Based upon this idea, Pouliot, Pierson, and Brusch [11] have developed a highly efficient code, sequential quadratic programming, and the results obtained thus far have been impressive.

Although much work has been done on the minimum time-to-climb problem, it is interesting to note that most investigators [14-31], and the ones mentioned above, have been concerned mainly with the development of numerical techniques for solving this problem. The numerical techniques are applied to a specific dynamic model, and the corresponding results are compared. The advantages and disadvantages of one method over another can then be easily seen. These improved numerical techniques along with the simplified dynamic models save much time and money.

However, many questions arise. For example, is the choice of a particular dynamic model a "good" one or not? What happens if another dynamic model is used? Will a two-state model produce the same results as a three-state model? If so, which variable produces little or no change in the results and can therefore be excluded from the equations of motion to save computational expense? If not, what causes the change in the results? Is it because the equations of motion have been linearized? Is it because motion out of vertical plane is not included? Or is it because an important variable has been excluded? These are just a few of the many unanswered questions.

An earlier work by Ardema [12] answers some of these questions. He solves the minimum time-to-climb problem using the energy-state, two-state, and a modified two-state model. From his work, he concludes that thrust and weight influence the time-to-climb most strongly and that the modified two-state model is significantly better than the other two. However, no numerical results are presented for the three-state, four-state, and five-state models. Thus, it is likely that other factors may be present that influence significantly the minimum time-to-climb. Pierson [13] also compares several dynamics models with the help of a sequential quadratic programming method, but his study

involves a minimum-noise problem rather than the minimum time-to-climb problem treated here. However, his approach serves to motivate and guide the present study of a whole range of models so that a model comparison can be made.

In the following study, five dynamic models are examined. The models used range from the simple energy-state model to the complete five-state model. The descriptions of each model as well as the problem formulation are given in Chapter 2. To provide a basis for comparison, we use sequential quadratic programming [11] to solve the minimum time-to-climb problem. This method of solution is described in Chapter 3. In Chapter 4, numerical solutions and model comparison are presented, and in Chapter 5, we have the summary and conclusions.

CHAPTER 2. PROBLEM FORMULATION

It is desired that the flight path of a supersonic aircraft be found which gives a minimum-time climb from a given initial energy-level to a final energy-level. The aircraft is represented by five dynamic models, and the models used range from the simple energy-state approximation model to the complete five-state point-mass model. This Chapter presents the problem formulations for these five dynamic models.

Equations of Motion

Figure 1 shows the nomenclature commonly used for an aircraft flying in a vertical plane. The general system of equations of motion in a vertical plane is the fifth-order system which describes a variable weight point-mass moving over a flat non-rotating earth. By considering only motion in the vertical plane, we obtain the equations of motion [4]:

$$m \dot{V} = T \cos(\alpha + \varepsilon) - D - m g \sin\gamma \qquad (2\text{-}1)$$

$$m V \dot{\gamma} = T \sin(\alpha + \varepsilon) + L - m g \cos\gamma \qquad (2\text{-}2)$$

$$\dot{h} = V \sin\gamma \qquad (2\text{-}3)$$

$$\dot{x} = V \cos\gamma \qquad (2\text{-}4)$$

$$\dot{m} = -f \qquad (2\text{-}5)$$

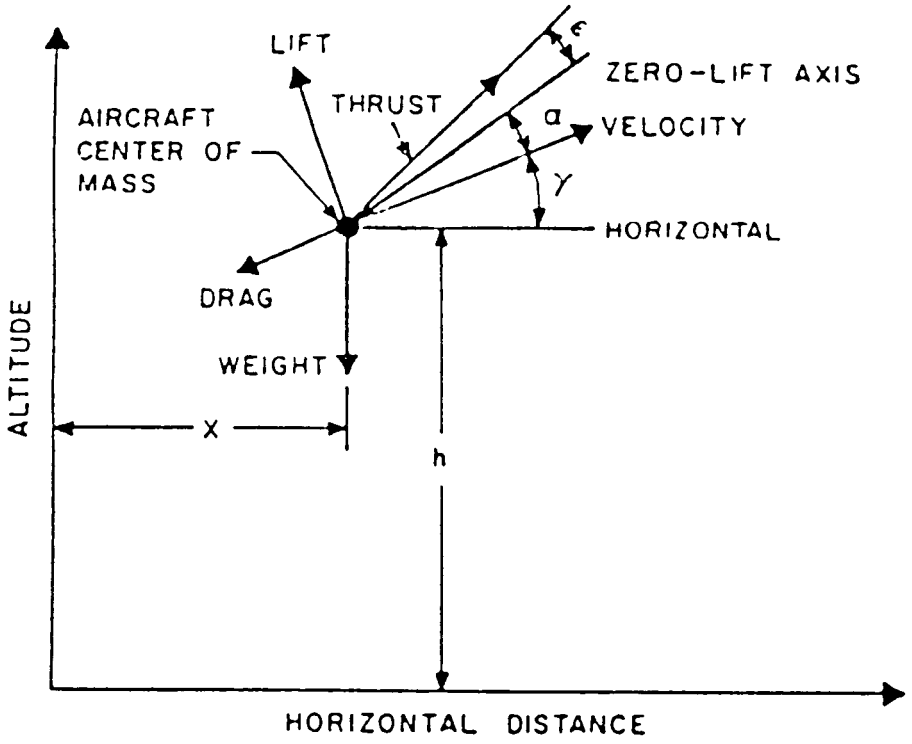

FIGURE 1. Aircraft nomenclature

Equations (2-1) and (2-2) represent the aircraft motion along and normal to the direction of the relative velocity respectively. These two equations results from Newton's second law ($\vec{F} = m \vec{a}$) applied to an aircraft in planar flight. Equation (2-3) shows the vertical height rate, while equation (2-4) is the horizontal rate. The last equation governs the mass loss due to fuel consumption.

The variables in the equations of motion are defined as follows:

V - relative velocity

γ - flight path angle

h - altitude

x - horizontal range

m - mass

α - angle of attack (measured from zero-lift axis)

ε - angle between thrust axis and zero-lift axis

g - gravity acceleration

T - thrust

D - drag

L - lift

f - fuel flow rate

The aerodynamic forces are approximated by assuming lift to be a linear and drag to be a quadratic function of α, i.e.,

$$L = q\ S\ C_{L_\alpha}\ \alpha \qquad (2\text{-}6)$$

$$D = q\ S\ (C_{D_o} + \eta\ C_{L_\alpha}\ \alpha^2) \qquad (2\text{-}7)$$

where $q = \rho\ V^2/2$ is the dynamic pressure, S is the aerodynamic reference area, C_{L_α} is the lift coefficient slope, C_{D_o} is the zero-lift drag coefficient, and η is the efficiency factor ($0 \leq \eta \leq 1$). In general, C_{L_α}, C_{D_o}, and η

depend on the Mach number. Tables and curve fittings for these aerodynamic characteristics are given in Appendix I.

The thrust is a function of both velocity and altitude. A fourth-order polynomial in velocity and altitude is fitted by a least-squares method to the thrust data and is given in Appendix II.

To make our minimum time-to-climb problem more realistic, the mass of the aircraft is not assumed to be constant. Instead, the higher order models (order 4 and 5) have the aircraft's mass dependent on the fuel consumption rate, f, which in turn is a function of thrust, i.e.,

$$f = T/cg \tag{2-8}$$

where c = 1600 seconds, and g = 32.174 ft/sec^2. The remaining lower order models have the aircraft's mass approximated by a linear function of range, i.e.,

$$m = k_1 x + k_2 \tag{2-9}$$

where k_1 and k_2 are constants determined from the aircraft's mass boundary conditions (see Table 1 for values).

All the models being examined utilize a variable atmospheric density. A standard exponential form for the atmospheric density is used, i.e.,

$$\rho(h) = \rho_o e^{-h/h_1} \tag{2-10}$$

where $\rho_o = 2.54 \times 10^{3}$ slug/ft^3 and $h_1 = 2.73 \times 10^{4}$ ft.

The above relationships enabled us to use explicit function representations to model the aircraft's thrust, lift, drag, and fuel consumption characteristics.

To illustrate a physical problem, Figure 2 shows a typical mission of a fighter aircraft intercepting a target in Earth's atmospheric space. Should the target be an enemy, it becomes important that the target be intercepted in the least possible time. Thus, a form of a minimum time-to-climb path is necessary.

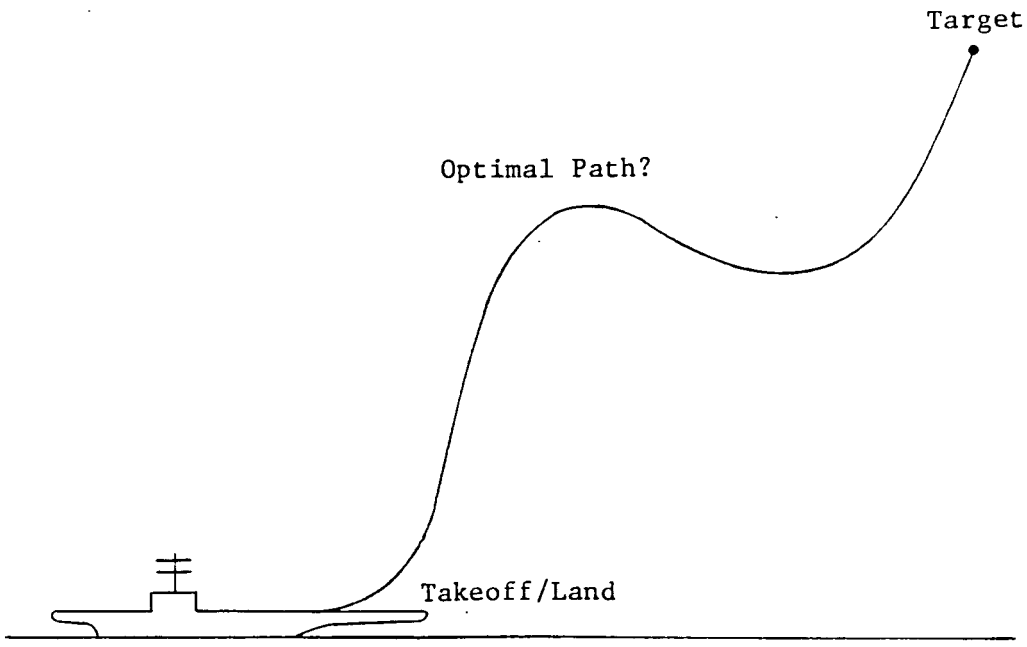

FIGURE 2. Aircraft intercept mission

Aircraft Models

The various aircraft models being examined will now be discussed. We will begin from the most complex model to the least complex one. In order of decreasing complexity, the models are:

1) five-state point-mass equations
2) four-state point-mass equations
3) three-state point-mass equations
4) two-state point-mass equations

and 5) energy-state approximation

The problem formulations for these five dynamic models are as follows:

Five-state point-mass equations

We want to minimize the time it takes for an aircraft to climb from one energy-level to another. Thus, an obvious possible function to be minimize is

$$\min J = t_f \tag{2-11}$$

where J is the performance index, and t_f is the final time.

Five state variables are involved. They are velocity, altitude, flight path angle, range, and mass with time as the independent variable. Only one control function is used, and that control is the angle of attack, α.

If we assumed that the angle between thrust axis and zero-lift axis (ε) is zero, as is often the case, the state equations are exactly the same as equations (2-1)-(2-5) except for $\varepsilon = 0$. Thus, the problem can then be stated as:

To find the angle of attack history, $\alpha(t)$, which minimizes

$$J = t_f \tag{2-12}$$

subject to:

$$m \dot{V} = T \cos\alpha - D - m g \sin\gamma \tag{2-13}$$

$$m v \dot{\gamma} = T \sin\alpha + L - m g \cos\gamma \tag{2-14}$$

$$\dot{h} = V \sin\gamma \tag{2-15}$$

$$\dot{x} = V \cos\gamma \tag{2-16}$$

$$\dot{m} = -f \tag{2-17}$$

and specified initial and final states and the constraint $h \geq 0$ (refer to Table 1 in Chapter 4 for specified states). The above formulation is an example of an optimal control Meyer problem and it features the usual point-mass equations of motion in a vertical plane. The dot notation ($\dot{}$) indicates that it is a time rate of change of the concern variable.

Four-state point-mass equations

In this and the remaining models, range is used to replace time as the independent variable. Since time is no longer a variable, the performance index (2-12) would have to change. We can think of J as the integral of time, i.e.,

$$J = \int_0^{t_f} dt \tag{2-18}$$

From equation (2-16), we can solve for dt in terms of dx to get

$$dt = \frac{1}{V \cos\gamma} dx \tag{2-19}$$

Substituting (2-19) into (2-18) for dt, the performance index becomes

$$J = \int_0^{R_f} \frac{1}{V \cos\gamma} dx \tag{2-20}$$

where R_f is the final range at t_f obtained from solving Model 1. This integral, which is now with respect to range, will still give us a minimum-time performance. As final range is specified, the problem is then of a fixed-range minimum time-to-climb problem.

The states are velocity, altitude, flight path angle, and mass with angle of attack as the control. The derivatives of these states are taken with respect to range and they can be easily obtained by dividing equations (2-13)-(2-15) and (2-17) by equation (2-16). In doing so, we obtain the following equations of motion:

$$V' = \frac{T \cos\alpha - D - m\, g \sin\gamma}{m\, V \cos\gamma} \tag{2-21}$$

$$\gamma' = \frac{T \sin\alpha + L - m\, g \cos\gamma}{m\, V^2 \cos\gamma} \tag{2-22}$$

$$h' = \tan\gamma \tag{2-23}$$

$$m' = -\frac{f}{V \cos\gamma} \tag{2-24}$$

The prime notation (') indicates that derivatives are taken with respect to range. We can now consider the following problem:

> To find the angle of attack history, $\alpha(x)$, which minimizes equation (2-20) subject to equations (2-21)-(2-24) and specified initial and final states and the constraint $h \geq 0$.

Model 1 and 2 are basically the same. The only difference is in the independent variable used. The first uses time while the latter uses range as the independent variable.

Three-state point-mass equations

This model is a simplification of the previous model. The four-state model is reduced to a three-state model by omitting the mass differential equation. Velocity, altitude, and flight path angle remain as the states with angle of attack still as the control. The mass, now, for simplicity, is approximated by a linear function of range, i.e.,

$$m = k_1 x + k_2 \qquad (2\text{-}25)$$

where k_1 and k_2 are constants determined from the aircraft's mass boundary conditions.

The problem statement is then: To find the angle of attack history, $\alpha(x)$, which minimizes

$$J = \int_0^{R_f} \frac{1}{V \cos\gamma} \, dx \tag{2-26}$$

subject to:

$$V' = \frac{T \cos\alpha - D - m g \sin\gamma}{m V \cos\gamma} \tag{2-27}$$

$$\gamma' = \frac{T \sin\alpha + L - m g \cos\gamma}{m V^2 \cos\gamma} \tag{2-28}$$

$$h' = \tan\gamma \tag{2-29}$$

and specified initial and final states and the constraint $h \geq 0$.

Two-state point-mass equations

This model results from further simplifications of Model 3. By inclusion of two additional assumptions, i.e.,

i) The angle of attack, α, is small so that by small angle approximation, $\cos\alpha \approx 1$

and ii) Flight path angle dynamics are neglected

the three-state model is reduced to a two-state model. Only velocity and altitude remain as the states with flight path angle now playing the role of the control.

The aerodynamic drag for this model, however, requires some modifications. The form that we have for our more complex models, i.e.,

$$D = q S (C_{D_o} + \eta C_{L_\alpha} \alpha^2) \tag{2-30}$$

could no longer be used here because of the small angle of attack assumption. A suitable form needs to be developed, and we can do so using the drag polar function

$$D = q\ S\ (C_{D_o} + \beta\ C_L^2) \tag{2-31}$$

where β is a constant.

By neglecting flight path angle dynamics and with the small angle of attack approximation, equation (2-2) reduces to

$$0 = L - m\ g\ \cos\gamma \tag{2-32}$$

which means that lift is equal to the component of the aircraft's weight normal to the flight path. The aerodynamic lift, L, is given by

$$L = q\ S\ C_L \tag{2-33}$$

Using (2-33) in (2-32), and solving for C_L, we get

$$C_L = \frac{m\ g\ \cos\gamma}{q\ S} \tag{2-34}$$

Substituting for C_L in the drag polar function (2-31), we obtain

$$D = q\ S\ [C_{D_o} + \beta(\frac{m\ g\ \cos\gamma}{q\ S})^2\ ] \tag{2-35}$$

This form of aerodynamic drag will be used for our model. The problem statement may now be given as:

To find the flight path angle history, $\gamma(x)$, which minimizes

$$J = \int_0^{R_f} \frac{1}{V \cos\gamma} \, dx \tag{2-36}$$

subject to:

$$V' = \frac{T - D - m g \sin\gamma}{m V \cos\gamma} \tag{2-37}$$

$$h' = \tan\gamma \tag{2-38}$$

and specified initial and final states and the constraint $h \geq 0$. As a reminder, mass is not constant, but is approximated by a linear function of range.

Energy-state approximation

The last model to be examined is the well-known energy-state approximation model. In this approximation, only the specific energy, E, is treated as the state variable:

$$E = \frac{1}{2} V^2 + g h \tag{2-39}$$

The range rate of change of E is obtained by differentiating equation (2-39) with respect to range

$$E' = V V' + g h' \tag{2-40}$$

Using (2-37) and (2-38) to eliminate V' and h', we get

$$E' = \frac{T - D}{m \cos\gamma} \tag{2-41}$$

An additional assumption made in the energy-state approximation is that the flight path angle is small so that by the small angle approximation, $\cos\gamma \approx 1$. Thus, the range

rate of change of E reduces to

$$E' = \frac{T - D}{m} \tag{2-42}$$

In addition the altitude constraint has to be modified. If we solve for gh in equation (2-39), we get

$$g\,h = E - \frac{1}{2} V^2 \tag{2-43}$$

Thus, to ensure altitude remains positive, we require

$$E - \frac{1}{2} V^2 \geq 0 \tag{2-44}$$

The problem statement will now be stated:

To find the velocity history, V(x), which minimizes

$$J = \int_0^{R_f} \frac{1}{V}\, dx \tag{2-45}$$

subject to: $$E' = \frac{T - D}{m} \tag{2-46}$$

and specified initial and final states and the constraint (2-44). Note that velocity now plays the role of the control.

The problem formulations for the five dynamic models have been precisely stated above. Table 1 (see Chapter 4) summarizes these problem formulations. Also included in this table are flight conditions and control constraints. These problems, however, will not be solved as optimal

control problems. Rather, they will be solved as parameterized optimization problems. In the next Chapter, we will show how these optimal control problems can be reformulated as parameterized optimization problems. The choice of a numerical technique to solve the minimum time-to-climb problems to provide a basis for comparison will also be discussed.

CHAPTER 3. METHOD OF SOLUTION

The minimum time-to-climb problems can be solved in two ways, either as infinite-dimensional problems or as parameterized optimization problems. In this Chapter we discuss the latter approach. The numerical method chosen to solve these minimum time-to-climb problems is also given.

Optimal Control Problem Statement

A general statement of the optimal control problem treated here is:

Find the scalar control function, u(t), which minimizes the performance index

$$J = \phi[x(t_f)] + \int_{t_o}^{t_f} L(x,u,t)\, dt \qquad (3\text{-}1)$$

subject to the n state equations

$$\dot{x} = f(x,u,t) \qquad (3\text{-}2)$$

the initial and terminal state constraints

$$x(t_o) = x_o; \qquad x(t_f) = x_f \qquad (3\text{-}3)$$

and to the control constraints

$$u_l \leq u(t) \leq u_u \qquad (3\text{-}4)$$

Here, n refers to the number of states, and u_l and u_u are the lower and the upper control bounds, respectively.

The above optimal control problem can be treated as an infinite-dimensional problem, i.e., we can minimize the performance index while satisfying the specified constraints

for an entire control time history. Infinite-dimensional numerical techniques such as 'shooting' or quasilinearization methods can then be employed to solve this problem. This approach, however, has two main drawbacks. First, the initial preparation time to set up the problem is high. This is due to the analytic work involved in deriving the costate equations and the influence function equations. It is even more so and cumbersome as well when the particular problem involved is highly nonlinear. Second, the problem must be defined analytically in order that these infinite-dimensional methods can be implemented. We know often that such is not the case.

Thus, instead of posing the optimal control problem as an infinite-dimensional problem, we will reformulate it as a parameterized optimization problem.

Parameter Optimization

In this approach, the optimal control problem is transformed into a parameter optimization problem by choosing a form for the control function which contains a finite number of parameters. Rather than minimizing the performance index over the entire control history, we now minimize over this set of parameters. The resulting parameterized problem becomes simpler and easier to solve.

Control discretization

Figure 3 shows how the continuous control function, u(t), is discretized. We let the interval (t_o, t_f) be divided equally into q intervals. At each time point, we let u_i approximate the control value at time t_i, i.e.,

$$u_i \approx u(t_i) \tag{3-5}$$

where i = 0, 1, ..., q. We will have q+1 control nodes, the u_i's, and these control nodes serve as the control parameters. The control parameters are corrected at each iteration until some termination criterion is satisfied which results in an optimal solution. As the number of time interval increases, we will have more control nodes, and thus a closer approximation to the continuous control, u(t).

A piecewise linear interpolation scheme is used to calculate control values between control nodes. One major advantage of this scheme is that we can set an upper and a lower bound on these control nodes. In doing so, we ensure that the control constraints are never violated. Cubic interpolation scheme is another possibility. However, one has to be a little careful when using this scheme because the cubic nature of the interpolation can result in control constraint violations.

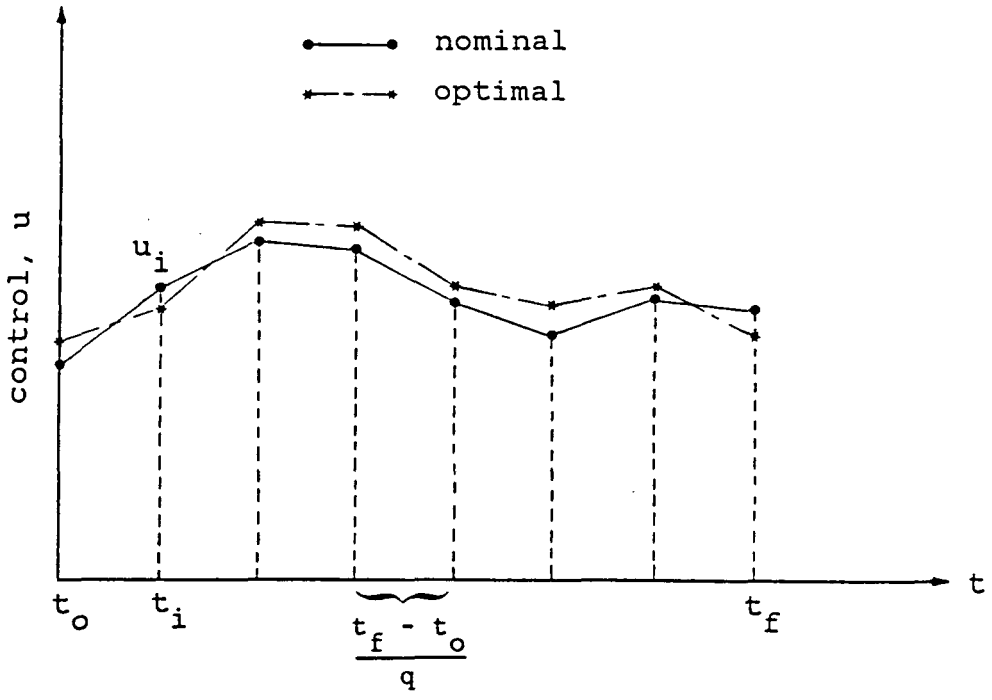

FIGURE 3. Control discretization

Choice of Numerical Method

The solution to our minimum time-to-climb problem involves integration of differential equations. The computational cost associated with doing this can be exceedingly high, especially if the differential equations are very complex and highly nonlinear. Since we want to make a model comparison between five dynamic models, the solutions to each model must also be accurate. Hence, we desire a method of solution that is both relatively accurate

and inexpensive. In addition, the method must be flexible to accommodate dynamic model, performance index, and constraint changes with relatively little reprogramming. It is for these reasons that the method of sequential quadratic programming [11] has been chosen.

The sequential quadratic programming algorithm is a constrained Quasi-Newton method which exhibits superlinear convergence. This method, which solves a series of quadratic programming problems, has proven to be very useful for problems with computationally expensive function and gradient evaluations. It consists of basically four steps [13]:

i) For an initial guess of the control parameters and an initial (positive definite) estimate of the Hessian matrix, compute the required first partial derivatives via numerical integration and finite-difference approximation and solve a quadratic programming problem for the corrections to the control parameter vector and the associated Lagrange multipliers.

ii) Perform a one-dimensional search along the direction of search vector obtained in step (i) by minimizing an auxiliary performance index. This step-size selection procedure is used to enhance convergence from poor initial control parameter estimates.

iii) Update the control parameter vector and test for convergence.

iv) If convergence is not achieved, update the Hessian matrix estimate by a variable-metric formula and repeat from step (i).

CHAPTER 4. NUMERICAL RESULTS AND MODEL COMPARISON

Numerical results of the minimum time-to-climb problem for our models are presented in this Chapter. All numerical computations were performed on the Iowa State University NAS/9160 computer using Fortran 77 with double precision arithmetic. A program listing for Model 1 is given in Appendix III.

Flight Conditions and Control Constraints

We want to compare the minimum time-to-climb results among five dynamic models. As the complexity of these models differ from one another, the initial and the final states for each model are modified accordingly so that the same initial and the same terminal flight conditions apply to each model. In our case, we always want the aircraft to fly from zero altitude at 400 ft/sec to an altitude of 65,600 feet at Mach number one.

All control variables are bounded. For Models 1, 2, and 3, the control angle of attack, α, is bounded between $\pm 10^{\circ}$. This is to prevent the aircraft from stalling. For Model 4, the control flight path angle, γ, is constrained to $\pm 80^{\circ}$. This is to avoid the singularity problem should γ approach 90°, a feature of using range as the independent variable. For Model 5, the control velocity, V, is

constrained to $0 \leq V \leq 1750$ ft/sec so that the physical capability of the aircraft is not violated. Table 1 summarizes the flight conditions and control constraints for the minimum time-to-climb problem.

Solutions to Models 1-5

Fifteen control points were used for each model. A piecewise linear interpolation scheme was used to interpolate between control points in each interval. Trapezoidal rule was used to evaluate the performance indices, and a fourth-order, fixed-step, Runge Kutta numerical integration scheme was used to integrate the differential equations of motion. One hundred integration steps were used.

Control histories

The optimal angle of attack histories for Models 1, 2, and 3 are shown in Figures 4-6, respectively. The flight path angle history for Model 4 is shown in Figure 7, and the velocity history for Model 5 is shown in Figure 8. It is apparent that the control points are not distributed evenly. A sensitivity analysis indicates that the distribution of the control points is highly important. In this case, more control points are needed in the earlier portion of the trajectory. All five models show this characteristic. When

TABLE 1. Flight conditions and control constraints

Model	Performance Index, J	State(s)	Control	Initial Condition(s)	Final Condition(s)
1	$J = t_f$	V - velocity h - height γ - flight path angle x - range m - mass	α - angle of attack $-10^\circ \leq \alpha \leq 10^\circ$	$V(t_o) = 400$ ft/sec $h(t_o) = 0$ ft $\gamma(t_o) = 0$ rad $x(t_o) = 0$ ft $m(t_o) = 1305$ lbs s^2/ft	$V(t_f) = 968.1$ ft/sec $h(t_f) = 65{,}600$ ft $\gamma(t_f)$ = free $x(t_f)$ = free $m(t_f)$ = free
2	$J = \int_{R_o}^{R_f} \frac{1}{V \cos\gamma}\, dx$	V, h, γ, m	α - angle of attack $-10^\circ \leq \alpha \leq 10^\circ$	$V(R_o) = 400$ ft/sec $h(R_o) = 0$ ft $\gamma(R_o) = 0$ rad $m(R_o) = 1305$ lbs s^2/ft	$V(R_f) = 968.1$ ft/sec $h(R_f) = 65{,}600$ ft $\gamma(R_f)$ = free $m(R_f)$ = free
3	$J = \int_{R_o}^{R_f} \frac{1}{V \cos\gamma}\, dx$	V, h, γ	α - angle of attack $-10^\circ \leq \alpha \leq 10^\circ$	$V(R_o) = 400$ ft/sec $h(R_o) = 0$ ft $\gamma(R_o) = 0$ rad	$V(R_f) = 968.1$ ft/sec $h(R_f) = 65{,}600$ ft $\gamma(R_f)$ = free
4	$J = \int_{R_o}^{R_f} \frac{1}{V \cos\gamma}\, dx$	V, h	γ - flight path angle $-80^\circ \leq \gamma \leq 80^\circ$	$V(R_o) = 400$ ft/sec $h(R_o) = 0$ ft	$V(R_f) = 968.1$ ft/sec $h(R_f) = 65{,}600$ ft
5	$J = \int_{R_o}^{R_f} \frac{1}{V}\, dx$	E - specific energy	V - velocity $0 \leq V \leq 1750$ ft/sec	$E(R_o) = 8 \times 10^4 \frac{ft^2}{sec^2}$	$E(R_f) = 2.56 \times 10^6 \frac{ft^2}{sec^2}$

For model 3, 4, and 5: $m(R) = -4.138889 \times 10^{-4} R + 1305.0 \frac{lbs\ s^2}{ft}$

these extra control points are not included, no solutions could be generated. The study reveals that sensitivity to control location varies throughout the climb trajectories. This in turn indicates that some portions of the climb trajectory are more sensitive than the rest, and that the portions which are more sensitive require more control points. Thus, to avoid difficulties in generating optimal trajectories a clear understanding of the sensitiveness of any problem to be solved is helpful. In our problem, the initial portion of the climb trajectories seems to be the most sensitive area. This is because of the altitude inequality constraint.

Figure 9 shows the comparison of flight path angle as a state (in Model 1), and as a control (in Model 4). We see clearly that the history of the flight path angle as a control is very similar to that when it is used as a state. However, when used as a control, we were able to employ a very simple 2-state model, rather than a much more complex one, thereby reducing the computational expense. The velocity, see Figure 10, also shows similar results, and could, therefore, be used as a state or as a control depending upon the complexity of the model desired.

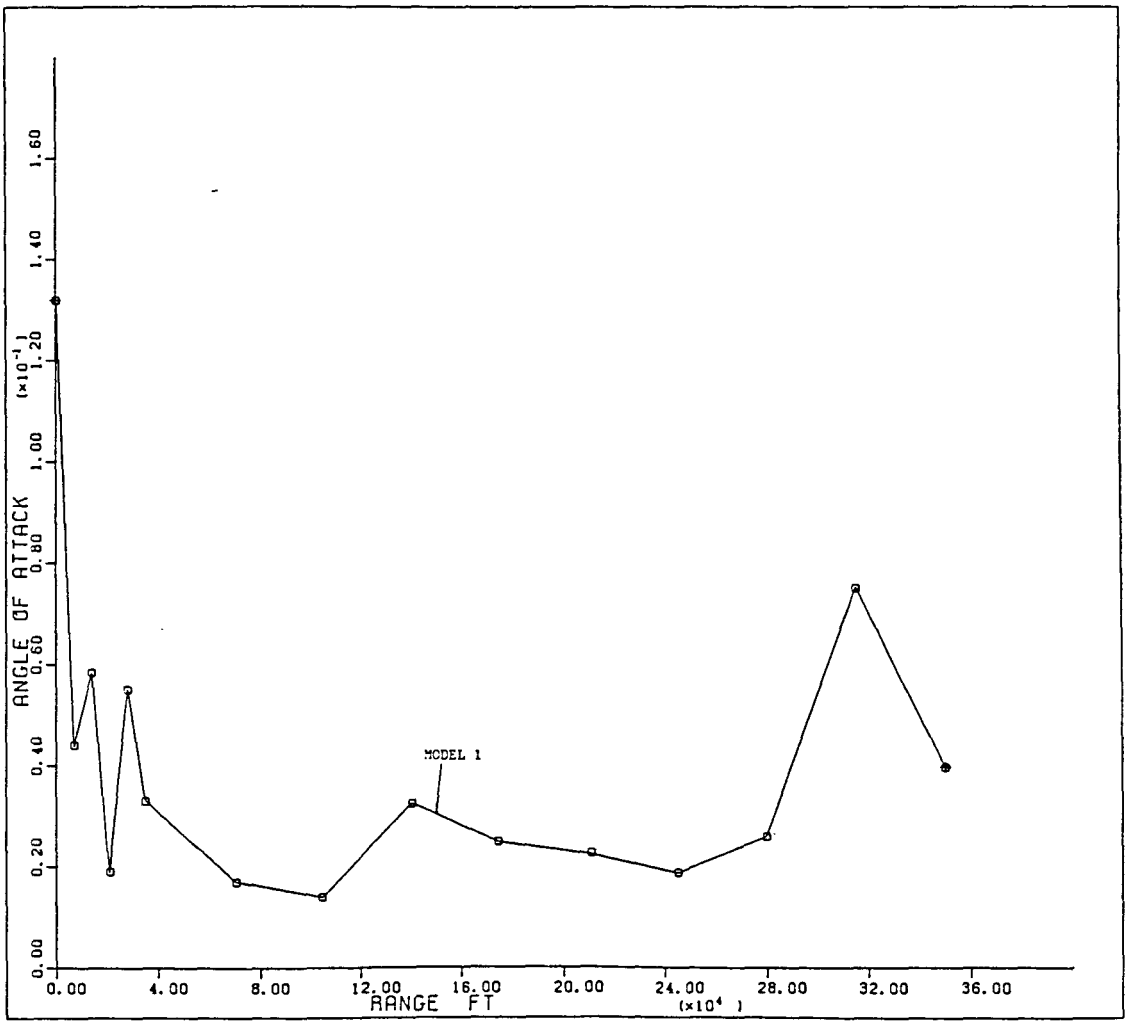

FIGURE 4. Angle of attack history for Model 1

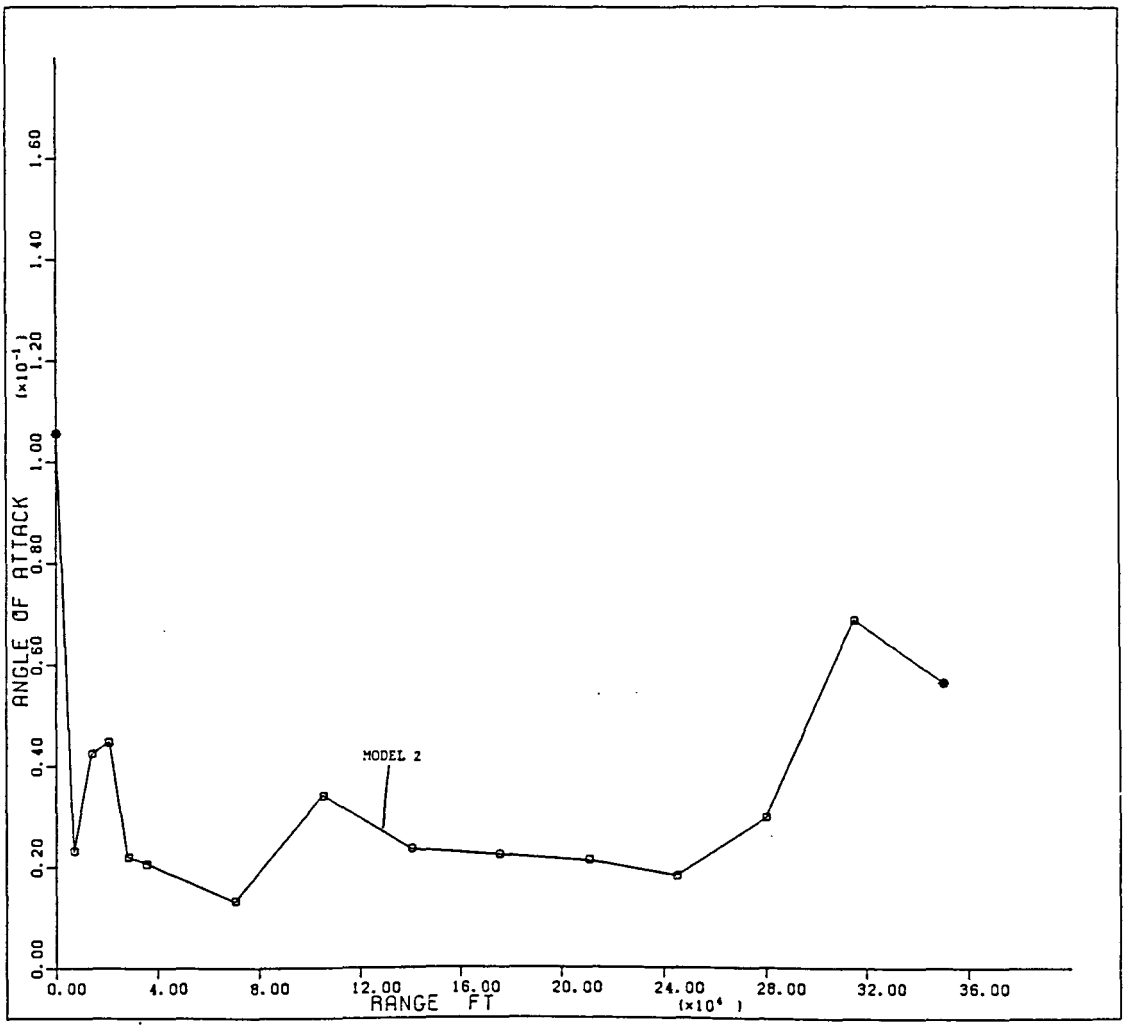

FIGURE 5. Angle of attack history for Model 2

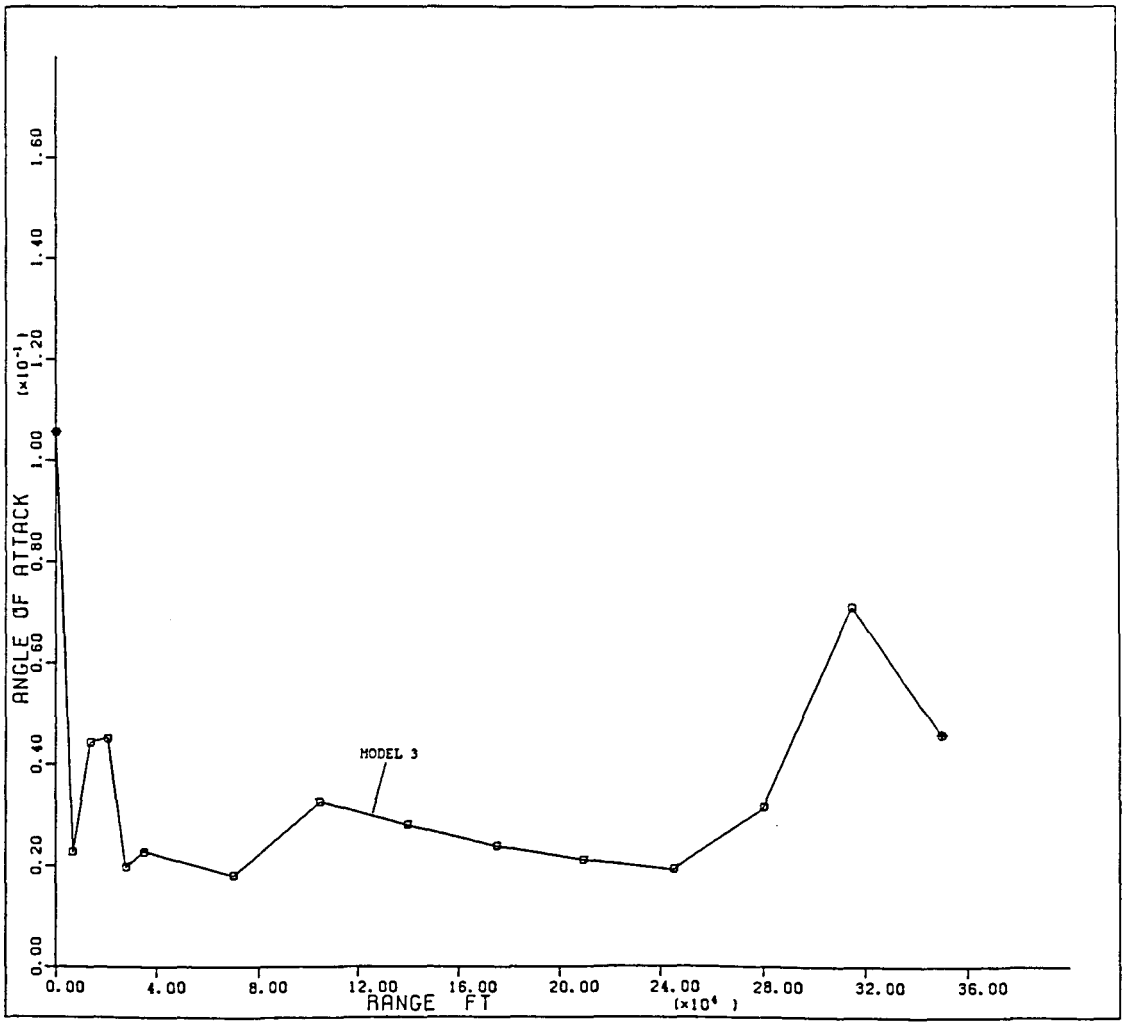

FIGURE 6. Angle of attack history for Model 3

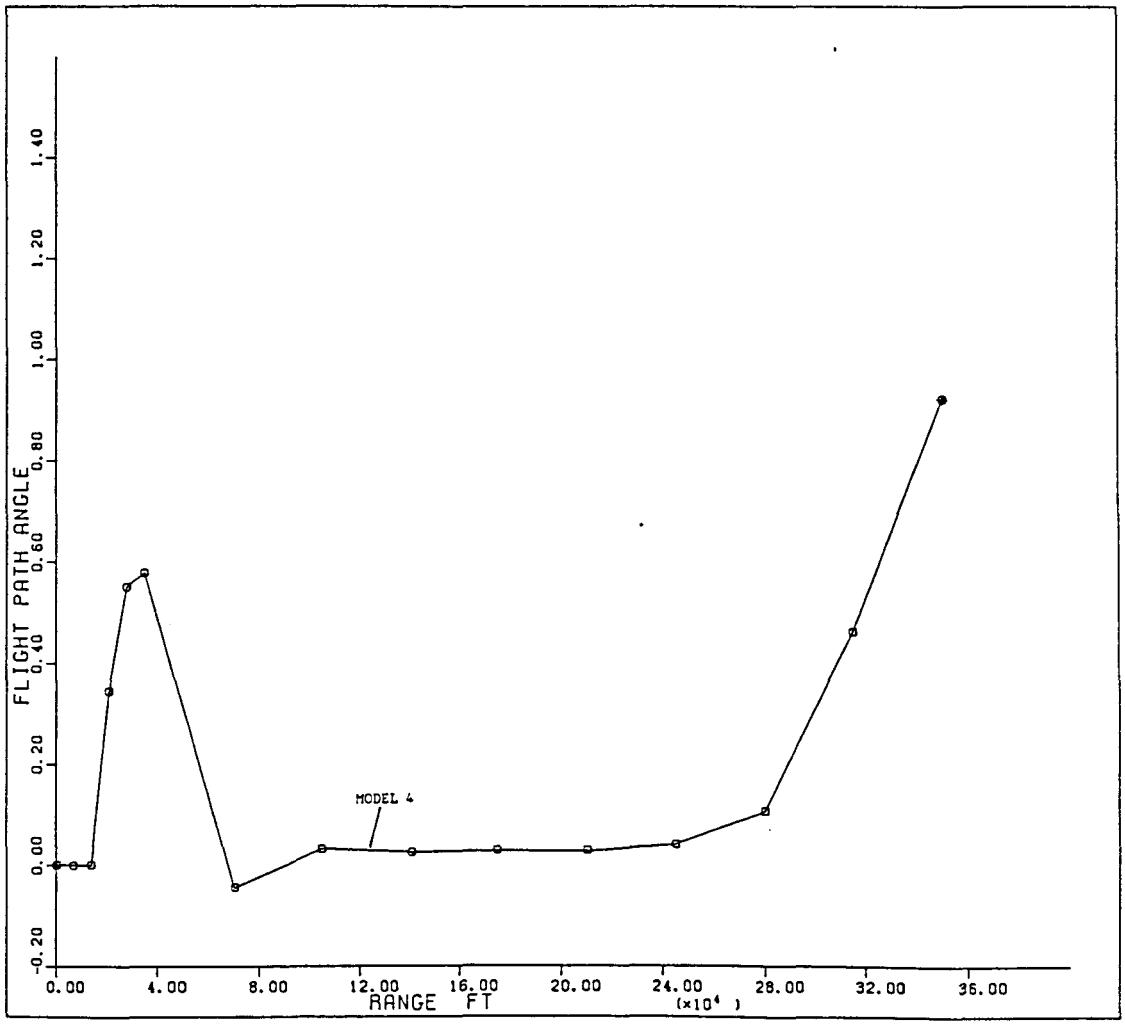

FIGURE 7. Flight path angle history for Model 4

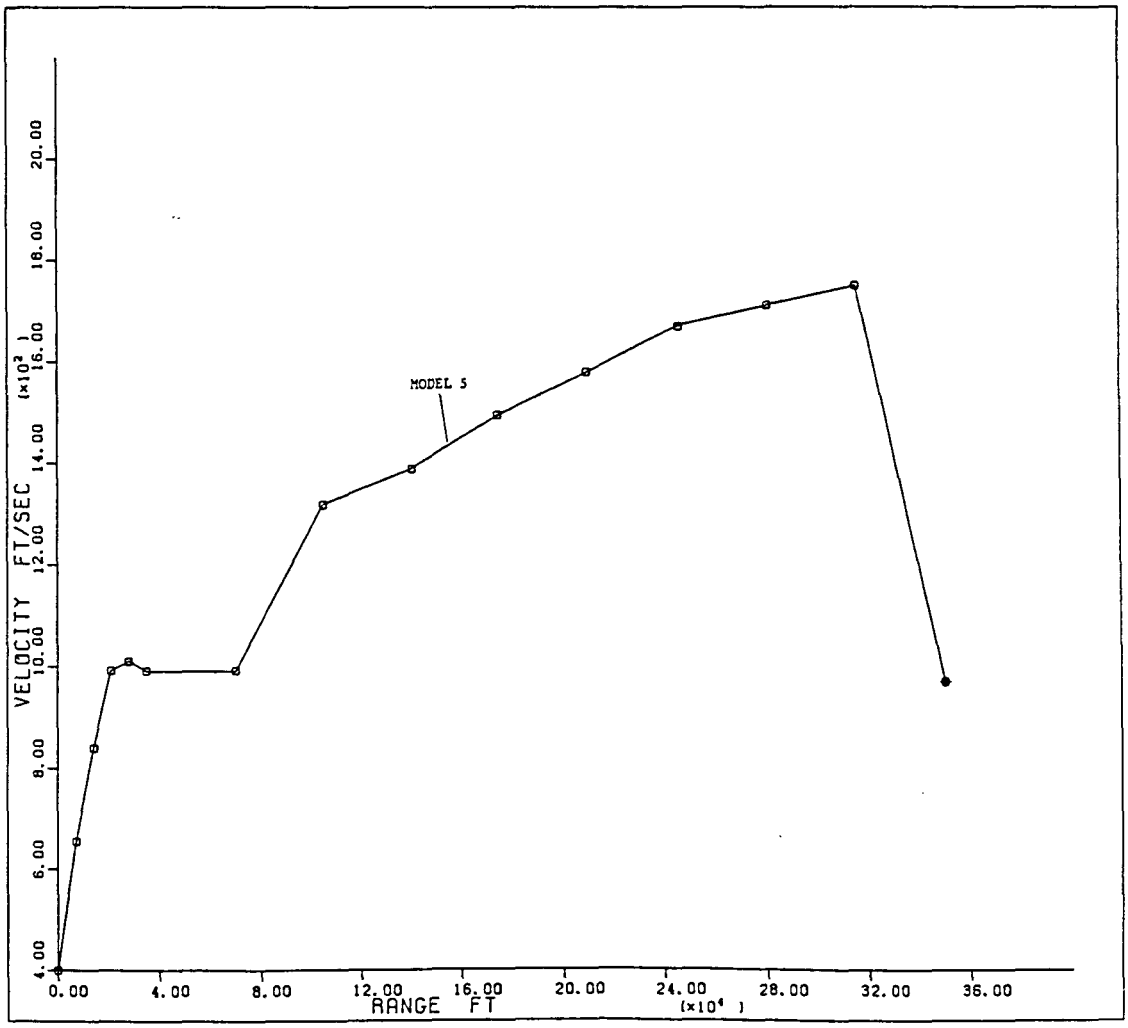

FIGURE 8. Velocity history for Model 5

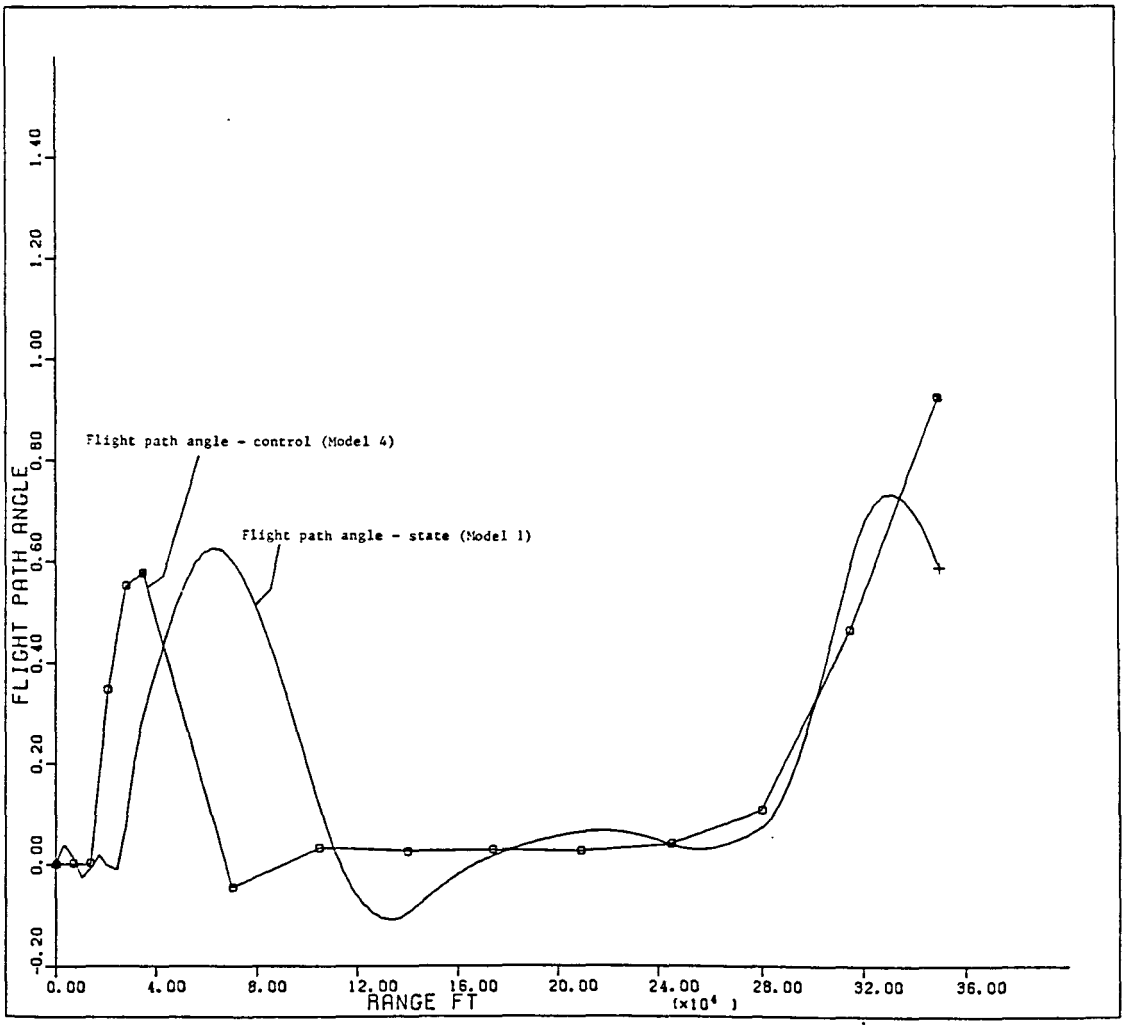

FIGURE 9. Flight path angle as a state or as a control comparison

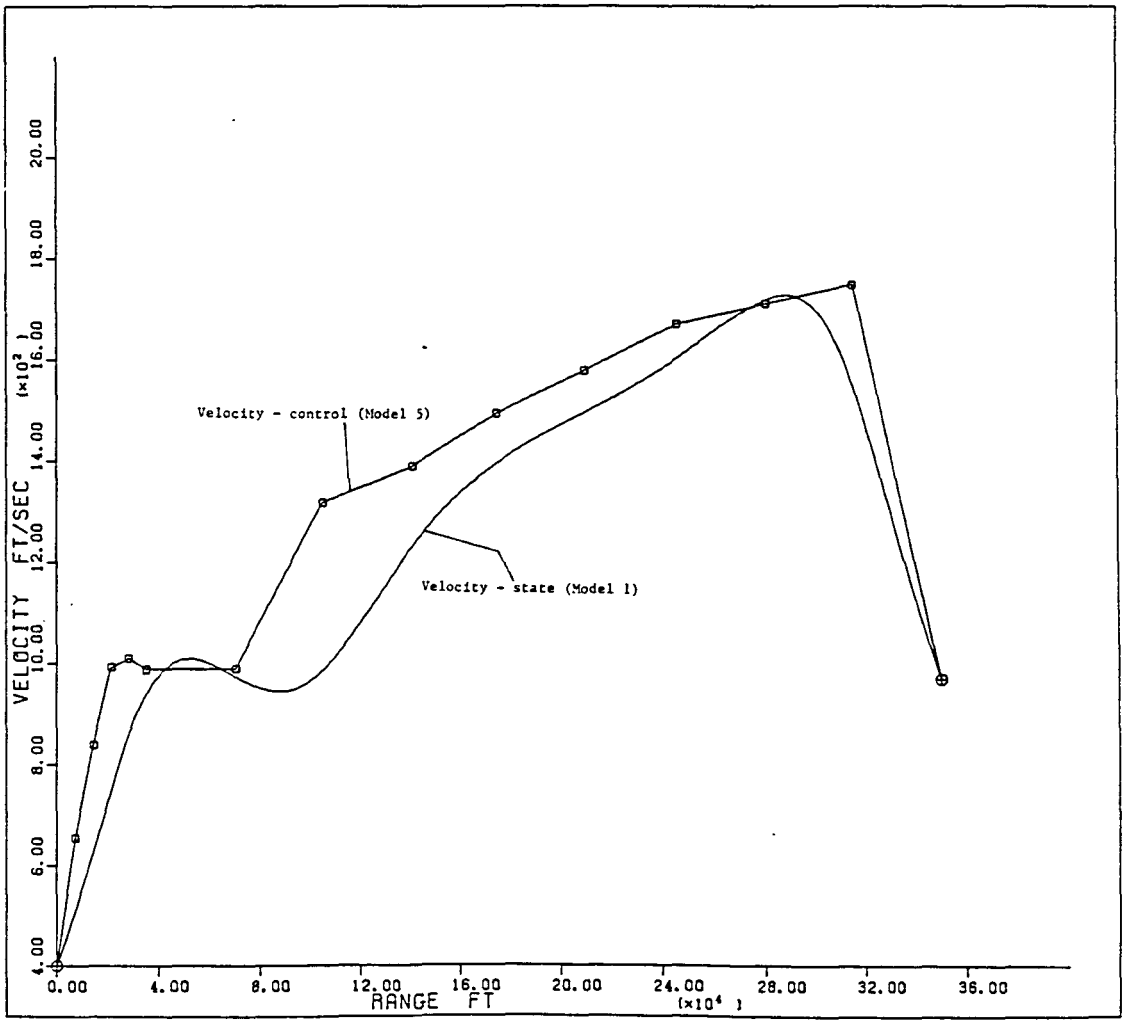

FIGURE 10. Velocity as a state or as a control comparison

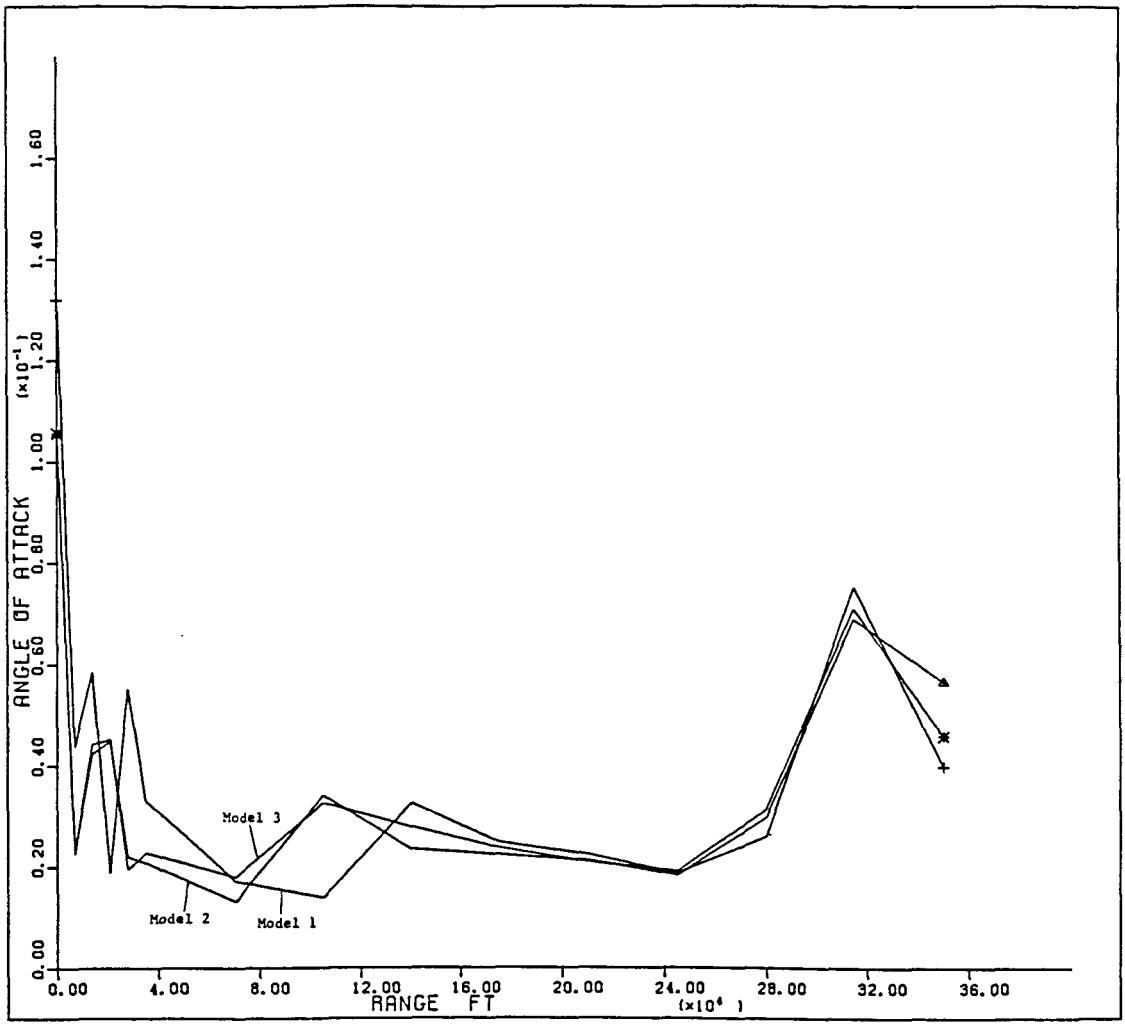

FIGURE 11. Angle of attack histories comparison

Figure 11 shows the comparison between the control histories for Model 1, 2, and 3. Qualitatively, these control histories are similar, as one would have expected. In addition, the angle of attack remained small for a large portion of the trajectory. This verifies the validity of our small angle of attack assumption.

Energy results

The energy results of all models, i.e., the exchange between kinetic energy and potential energy, are shown in Figure 12. An interesting feature of the results is that the trajectories categorize themselves into two distinct categories. Model 1, 2, and 3 fall into one group, while Model 4 and 5 fall into the other. In our study, the first three models include flight path angle dynamics while the latter two models do not. Although the results for all models are similar, the addition of flight path angle dynamics in the equations of motion can make a difference quantitatively. As a result, the flight path angle may significantly influence the minimum time-to-climb, in addition to the thrust and weight as indicated by Ardema [22]. This result is more apparent when the minimum-time climb for all models is compared later.

All models, except Model 4, exhibit dive and zoom climb trajectories from the initial energy level to the final

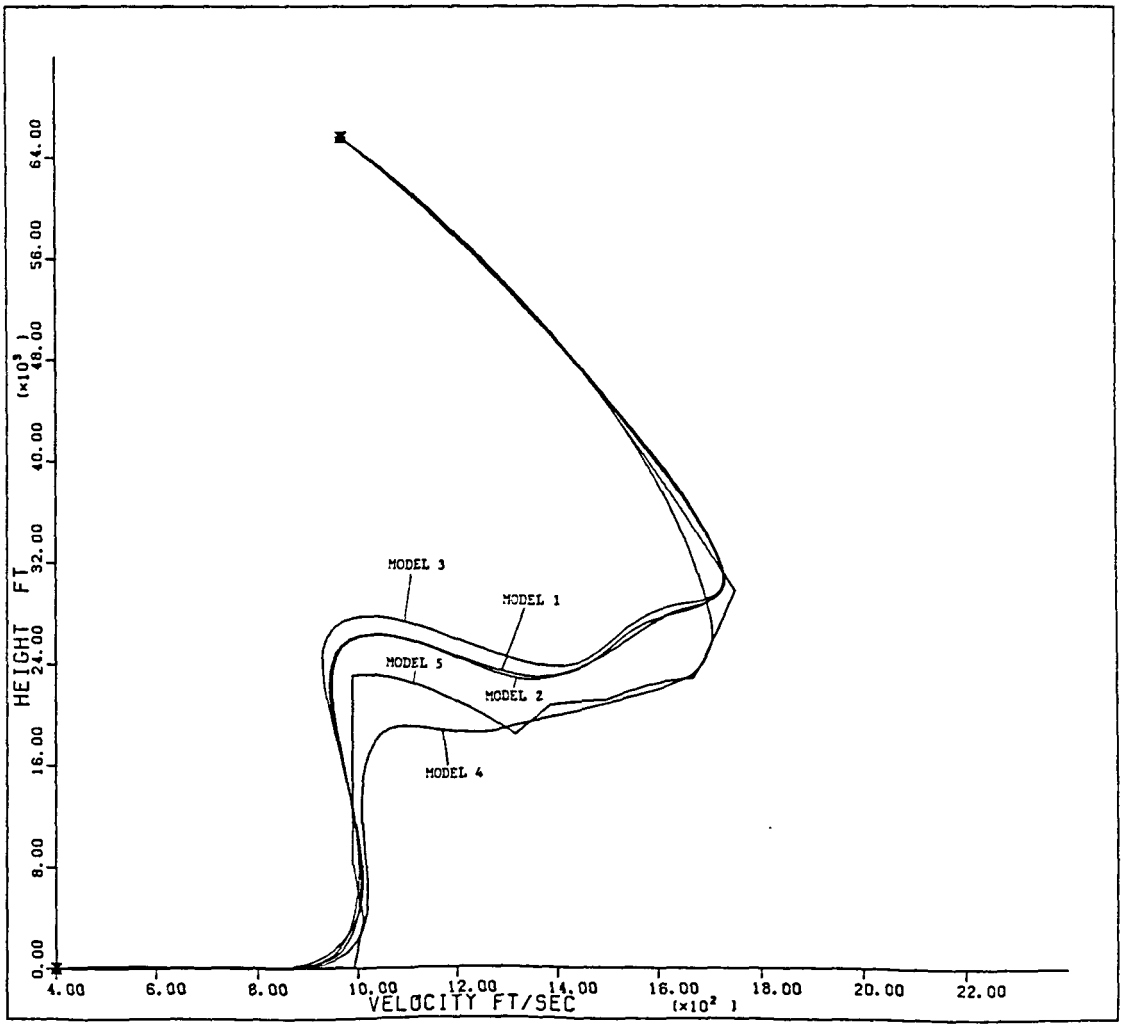

FIGURE 12. Energy trajectories

energy level. The dive portion of the trajectories occurs just before supersonic speed while the zoom-climb portion occurs near the upper bound of the velocity constraint, V_u = 1750 ft/sec. The altitude where these transitions occur, however, differ between each model. The transitions seem to occur at a higher altitude for those models that have flight path angle dynamics.

Observe also that all trajectories before the dive remain very close together. The zoom-climb region also exhibits the same characteristic. This can be explained by the fact that we have the same flight boundary conditions for all models. The trajectories at these two portions are flown mainly to satisfy these boundary conditions. It is obvious then that the primary role of the trajectories between these two portions is to minimize the time-to-climb.

Figure 13 compares Bryson's, Rader and Hull's, and our Model 5 solution for the minimum time-to-climb problem. The results are basically similar in nature, except that the dive occurs at a higher altitude in both Bryson's and Rader and Hull's solutions. Two reasons account for this difference. First, different optimization techniques have been used to solve the minimum time-to-climb problem. Bryson uses steepest descent method, while Rader and Hull use the hard constraint approach. Second, the values for

our aerodynamic curve fit are not exactly the same as those of Bryson's and Rader and Hull's, although they are very close.

Climb trajectories

Figure 14 shows the plots of altitude with range. The distinction between models that include flight path dynamics and models that do not is perhaps seen more clearly. Models 1 and 2 have virtually the same optimal climb trajectories. This is to be expected because the equations of motion for both models are basically the same. The only difference is that Model 1 uses time whereas Model 2 uses range as the independent variable. However, the computation expense for these two models is not the same, as will be shown later.

Note that all models, except for Model 5, have smooth plots. This is because the states of these models are evaluated from differential equations. However, for Model 5, velocity is not a state. Rather, it is a control. It is for this reason we see slope discontinuities in the flight profile. The occurrence of each slope discontinuity in the plot is exactly at the locations of our control points. If more control points are used for Model 5, then one would expect a smoother plot, but at the expense of computation cost.

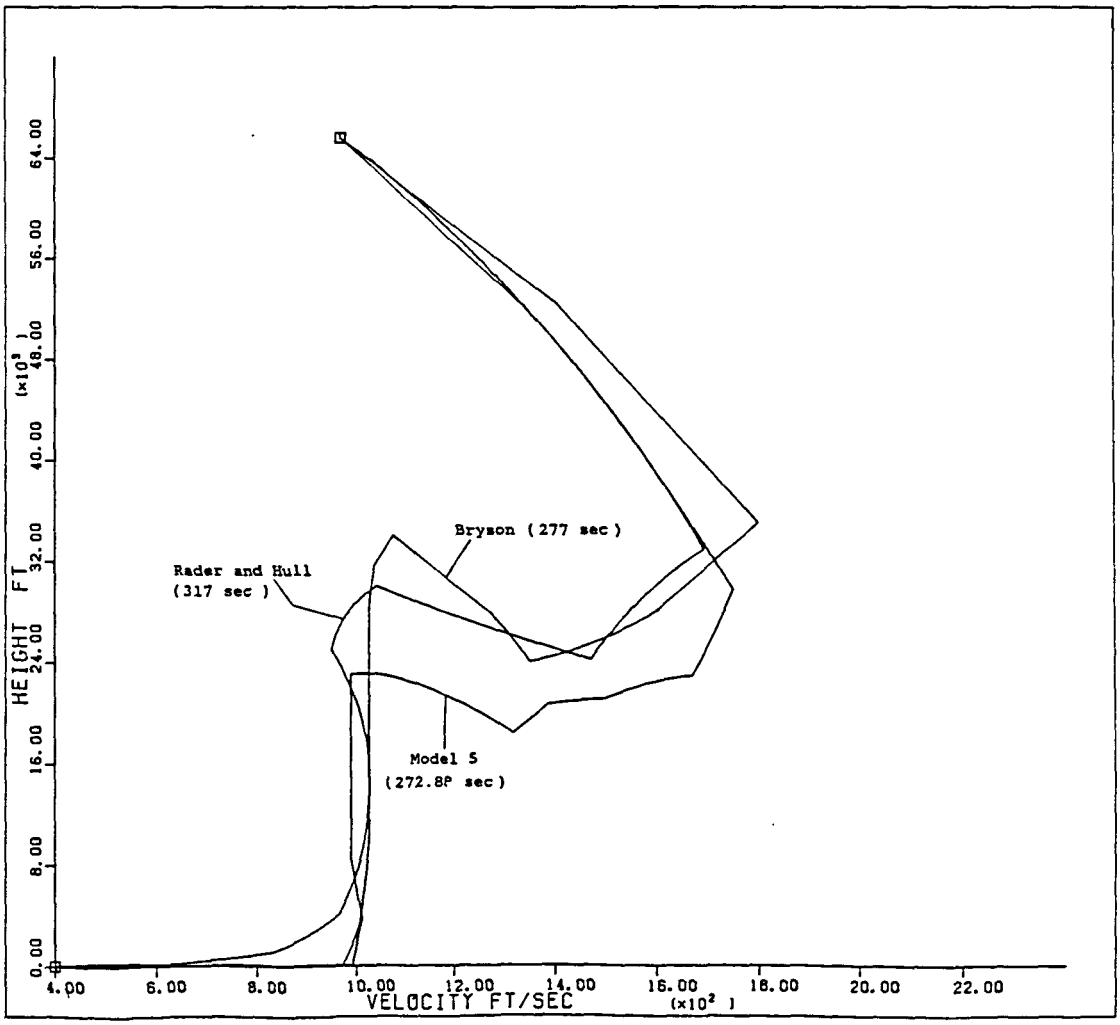

FIGURE 13. Energy comparison with published results

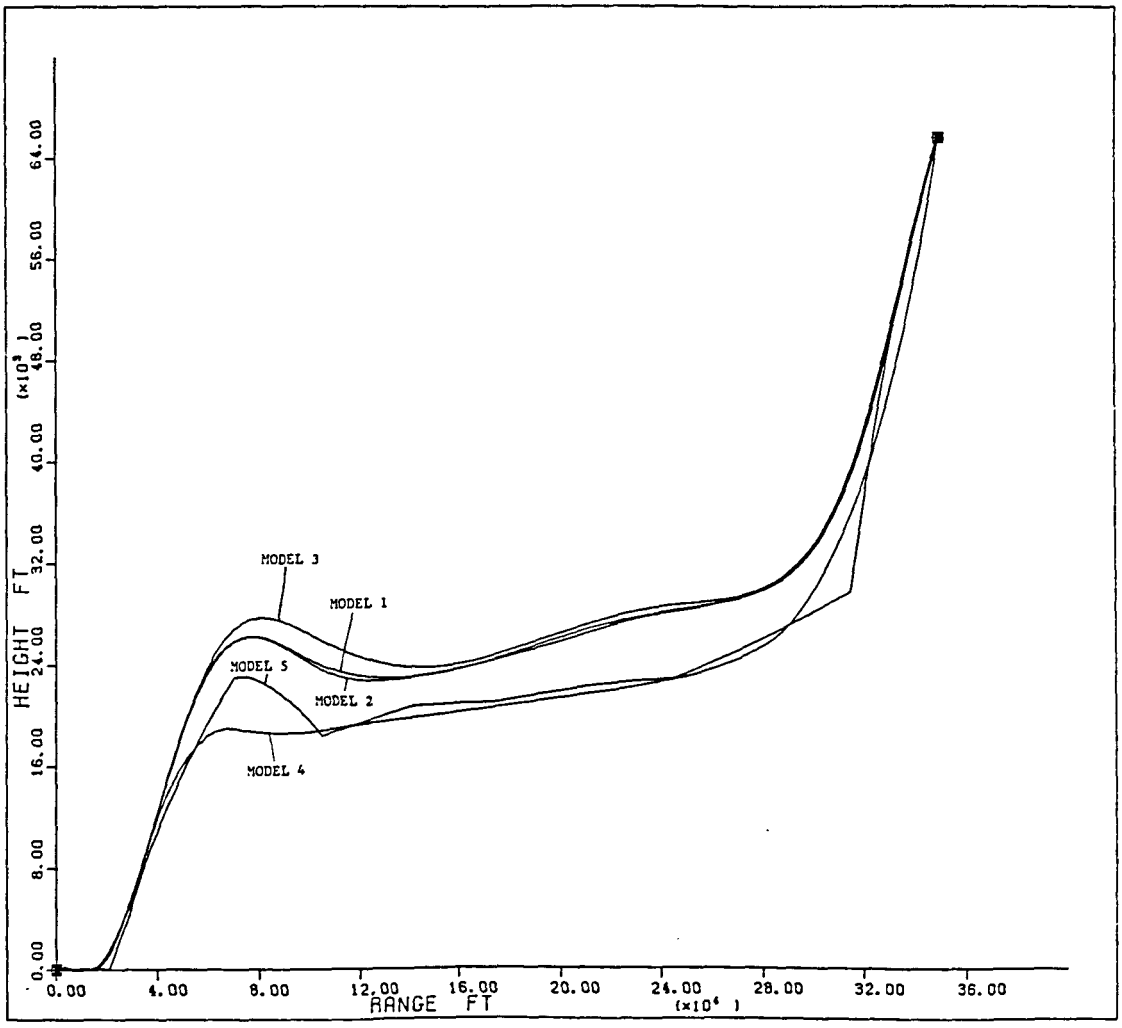

FIGURE 14. Optimal climb profiles

Again, we see that the two extreme portions of each trajectory serve to satisfy the initial and terminal constraints, while the middle portion serves to minimize the time-to-climb. Also, the monotonic increase in range as shown by the plots verifies the validity of using range as the independent variable.

Cost and time-of-flight comparison

TABLE 2. Cost and time-of-flight comparison

Model	Control points	CPU time per iter, sec	Time-of-flight sec
1	15	1.2604	290.09
2	15	1.1113	290.40
3	15	.9227	293.95
4	15	.7613	283.23
5	15	.3388	272.88
Bryson (Energy-state approximation)			277 sec
Rader and Hull (Hard Constraint)			317 sec

Table 2 shows the time-of-flight and computation time results for each model. Models 1 and 2 show essentially no difference in the time-of-flight as the equations of motion for these two models are basically the same, except in the independent variable used. However, their computation times differ. It takes only 1.1113 CPU sec per iteration for Model 2 compared to 1.2604 CPU sec per iteration for Model 1 - a saving of 12% in computation time.

Among Models 1, 2, and 3, those models that include flight path angle dynamics, Model 3 uses the least computation time - .9227 CPU sec per iteration, a saving of 27% in computation time compared to Model 1. No mass differential equation, however, is included in this model as well as in the remaining lower-order models. A linear function of range is used instead to approximate the aircraft's mass. It is this inexact mass approximation that results in a slight difference in the optimal climb trajectories. We have a higher altitude dive transition for Model 3 than for Models 1 and 2. This in turn leads to an additional 3.86 seconds in the flight time - 293.95 seconds compared to 290.09 seconds for Model 1. If a more accurate aircraft mass model is used, then we would expect negligible differences in the climb trajectories as well as for the flight time. The high saving in computation cost for Model

3 coupled with a very simple mass model, however, far exceed the desire to pursue an accurate mass modeling. Moreover, the accuracy of the result was not sacrificed to any large extent only a mere 1.3% difference in the flight time between Model 3 and Model 1.

Let's now consider the results of Models 4 and 5. For Model 4, it takes the aircraft 283.23 seconds of flight time to meet the specified terminal constraints; for Model 5, which is the energy-state model, it takes only 272.88 seconds. These are smaller times, but they should be considered only as rough approximations to the actual time-of-flight. The reason for this statement is that these models are so simplified that they become unrealistic, especially for the energy-state model where only one state variable is used. However, the results remain fairly accurate and can be obtained inexpensively. If we consider the computation time, it takes only .7613 CPU sec per iteration for Model 4, and .3388 CPU sec per iteration for Model 5. These are savings of 40% and 73%, respectively, in computation costs when compared to Model 1; a substantial savings. If one has no "feeling" for the solutions to the minimum time-to-climb problems for other supersonic aircraft, these two models, either Model 4 or Model 5, might be the models to be considered initially to generate a nominal solution for use in higher order models.

CHAPTER 5. SUMMARY AND CONCLUSIONS

The minimum time-to-climb problem is formulated as a parameterized optimal control problem and is solved using sequential quadratic programming. Five dynamic models are treated. The models used range from the simple energy-state model to the complete five-state point-mass model. The five-state model features the usual point-mass equations for flight in a vertical plane. Time is the independent variable, and speed, altitude, flight path angle, range, and mass are the dependent variables. Range is used to replace time as the independent variable for the remaining four models.

It is clear that sensitiveness plays an important role in optimal aircraft trajectory generation. A lack of this understanding can lead to difficulties in obtaining optimal trajectories. This difficulty can be avoided when more control points are used for portions of the trajectories that are highly sensitive.

The two-state and the energy-state approximation models provide easily solved but optimistic results for minimum time-to-climb. The results, however, remain fairly accurate and can be used as nominal solutions for higher-order models.

Although the addition of flight path angle dynamics complicates the solution process, its addition significantly influences the minimum time-to-climb. For accurate performance prediction of the flight-time, it is necessary that the flight path angle dynamics be included in the equations of motion.

The intermediate three-state model indicates that the aircraft's mass differential equation can be replaced by a simple linear function of range without significant loss in accuracy. This replacement eliminates the need to integrate the mass differential equation, thus simplifying the model by one order.

Consideration of our numerical example shows a fairly good agreement between values of the minimum time-to-climb as predicted by the five dynamic models. However, the computation time between models varies significantly; Model 1 is the most expensive, while Model 5 is the least expensive. It can be seen that the energy-state approximation, properly set up, is adequate for performance optimization of supersonic aircraft.

Many extensions to this study are possible. Further models might include: 1) rotational dynamics, 2) non-flat rotating earth, 3) aircraft structural dynamics, and 4) multiple control variables. The study can also be extended

to compare flight profiles for: 1) minimum fuel climb, 2) maximum range in given time, 3) maximum range for a given amount of fuel, or 4) maximum range glide between specified energy levels.

BIBLIOGRAPHY

1. Bryson, A. E. and Denham, W. F. "A Steepest-Ascent Method for Solving Optimum Programming Problems." J. Applied Mechanics, 29 (June 1962), 247-257.

2. Lush, K. J. "A Review of the Problem of Choosing a Climb Technique with Proposals for a New Climb Technique for High Performance Aircraft." Gt. Brit., Aeronautical Research Council, Reports and Memoranda, No. 2557 (1951), 1-15.

3. Rutowski, E. S. "Energy Approach to the General Aircraft Performance Problem." J. Aeronautical Science, 21, No. 3 (Mar. 1954), 187-195.

4. Bryson, A. E., Jr., Desai, M. N., and Hoffman, W. C. "Energy State Approximation in Performance Optimization of Supersonic Aircraft." J. Aircraft, 6 (Nov.-Dec. 1969), 481-488.

5. Garfinkel, B. "Minimal Problems in Airplane Performance." Quarterly of Applied Mathematics, 9, No. 2 (July 1951), 149-162.

6. Miele, A. "On the Non-Steady Climb of Turbojet Aircraft." J. Aeronautical Sciences, 21 (Nov. 1954), 781-783.

7. Landgraf, S. K. "Some Applications of Performance Optimization Techniques to Aircraft." J. Aircraft, 2 (Mar.-April 1965), 153-154.

8. Kelley, H. J. "An Investigation of Optimal Zoom-Climb Techniques." J. Aeronautical Sciences, 26 (1959), 794-802.

9. Ardema, M. D. "Solution of the Minimum Time-to-Climb Problem by Matched Asymtotic Expansions." AIAA Journal, 14 (July 1976), 843-850.

10. Rader, J. E., and Hull, D. G. "Compution of Optimal Aircraft Trajectories Using Parameter Optimization Methods." J. Aircraft, 12 (Nov. 1975), 864-866.

11. Pouliot, M. R., Pierson, B. L., and Brusch, R. G. "Recursive Quadratic Programming Solutions to Minimum-Time Aircraft Trajectory Problems." Proc. 2nd IFAC Workshop of Control Applications of Nonlinear Programming and Optimization, Oberpfaffenhofen, West Germany (Sept. 15-17, 1980), 253-261.

12. Ardema, M. D. "Approximation in the Minimum Time-To-Climb Problem." NASA TM X-62292 (Aug. 1973).

13. Pierson, B. L. "Optimal Aircraft Landing-Approach Trajectories: A Comparison of Two Dynamic Models." Proc. 5th IFAC Workshop of Control Applications of Nonlinear Programming and Optimization, Capri, Italy (June 11-14, 1985).

14. Heermann, H. and Kretsinger, P. "The Minimum Time Problem," J. Astronautical Sciences, 11, No. 4 (1964), 93-107.

15. Kelley, H. J., Cliff, E. M., and Weston, A. R. "Energy State Revisited." Optimal Control Applications and Methods, 7, No. 2 (April-June 1986), 195-200.

16. Merrit, S. R., Cliff, E. M., and Kelley, H. J. "Energy-modelled Climb and Climb-dash - The Kaiser Technique." Automatica, 21 (May 1985), 319-321.

17. Kelley, H. J. and Edelbaum, T. N. "Energy Climbs, Energy Turns, and Asymptotic Expansions." J. Aircraft, 7, No. 1 (Jan.-Feb. 1970), 93-95.

18. Weston, A., Cliff, E. M., and Kelley, H. J. "Onboard NearOptimal Climb-Dash Energy Management." J. Guidance, Control, and Dynamics, 8 (May-June 1985), 320-324.

19. Theodorsen, T. "Optimum Path of an Airplane - Minimum Time-To-Climb." J. Aerospace Sciences, 26, No. 10 (Oct. 1959)

20. Parsons, M. G., Bryson, A. E. Jr., and Hoffman, W. C. "Long-range Energy-State Maneuvers for Minimum Time to Specified Terminal Conditions." J. Optimization Theory and Application, 17 (Dec. 1975), 447-463.

21. Andrews, R. H. "Optimum Climb Trajectories at Constant Lift Coefficient." J. Aircraft, 6, No. 5 (Sept.-Oct. 1969), 475-477.

22. Carstoiu, J. "On a Minimum Time Flight Path of a Jet Aircraft." J. Aeronautical Sciences, 24 (Sept. 1957), 704-706.

23. Cicala, P. and Miele, A. "Branchistocronic Maneuvers of a Constant Mass Aircraft in a Vertical Plane." J. Aeronautical Sciences, 22 (April 1955), 286-288.

24. Cicala, P. and Miele, A. "Branchistocronic Maneuvers of a Variable Mass Aircraft in a Vertical Plane." J. Aeronautical Sciences, 22 (Aug. 1955), 577-578.

25. Breakwell, J. V. "Optimal Flight-Path-Angle Transitions in Minimum-Time Airplane Climb." J. Aircraft, 14 (Aug. 1977), 782-786.

26. Breakwell, J. V. "More about Flight-Path-Angle Transitions in Optimal Airplane Climbs." J. Guidance and Control, 1 (May-June 1978), 205-208.

27. Weston, A. R., Cliff, E. M., and Kelley, H. J. "Altitude Transitions in Energy Climbs." Automatica, 19, No. 2 (1983), 199-202.

28. Ashley, H. "On Making Things the Best - Aeronautical Uses of Optimization." J. Aircraft, 19 (Jan. 1982), 5-28.

29. Ivanov, V. V. and Sirazetdinov T. K. "Optimization of Airplane Time to Reach a Given Flight Altitude and Speed." Soviet Aeronautics, 23, No. 2 (1980), 40-44.

30. Schultz, R. L. and Zagalsky, N. R. "Aircraft Performance Optimization." J. Aircraft, 9 (Feb. 1972) 108-114.

31. Bryson, A. E. Jr. and Hoffman, W. C. "A Study of Techniques for Real-Time On-Line Flight Path Control-Minimum Time Turns to a Specified Track." Rept. ASI-TR-4 (Sept. 1971) Burlington, Mass.

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to my major professor, Dr. Bion L. Pierson, for his helpful suggestions, ideas, and contributions to this research. I would also like to thank him for his invaluable encouragement and guidance throughout my graduate studies. To Nick A. Thorp, I am grateful for his help in getting me started with Sequential Quadratic Programming optimization routine, and also for his much-appreciated discussions concerning this project.

APPENDIX I. AERODYNAMIC CHARACTERISTICS

The aerodynamic data used in our study are that of an early representation of the F-4 fighter aircraft [4]. Equations (2-6) and (2-7) show the aerodynamic lift and drag equations. The lift coefficient slope, C_{L_α}, the zero-lift drag coefficient, C_{D_o}, and the efficiency factor, η, are all Mach number dependent. These Mach number dependent aerodynamic parameters are restated here in Table 3.

It is desirable that the aerodynamic data be represented in terms of analytic functions. These functions should be continuous and should have continuous first derivatives as well. A third-order polynomial function of Mach number is chosen to represent each of these aerodynamic parameters within each interval.

To illustrate how this is done, let us consider the lift coefficient slope parameter, C_{L_α}. The particular form is

$$C_{L_\alpha}(M) = a_0 + a_1(M - M_i) + a_2(M - M_i)^2 + a_3(M - M_i)^3 \quad \text{(I-1)}$$

where M_i is the discrete Mach number value in Table 3 such that

$$M_i \leq M \leq M_{i+1} \quad \text{(I-2)}$$

TABLE 3. Lift and drag characteristics [4]

M	0	0.8	0.9	1.0	1.2	1.4	1.6	1.8
C_{L_α}	3.44	3.44	3.58	4.44	3.44	3.01	2.86	2.44
C_{D_o}	0.013	0.013	0.014	0.031	0.041	0.039	0.036	0.035
η	0.54	0.54	0.75	0.79	0.78	0.89	0.93	0.93

$C_L = C_{L_\alpha} \alpha$ $\qquad L = \frac{1}{2} \rho V^2 S C_L \qquad S = 530 \text{ ft}^2$

$C_D = C_{D_o} + \eta C_{L_\alpha} \alpha^2 \qquad D = \frac{1}{2} \rho V^2 S C_D$

By differentiating (I-1) with respect to Mach number, we obtain

$$C_{L_\alpha}' = a_1 + 2a_2(M - M_i) + 3a_3(M - M_i)^2 \qquad \text{(I-3)}$$

The slope C_{L_α}' can be estimated graphically from a plot of C_{L_α} vs. M at the selected Mach number, M_i. These slopes are tabulated in Table 4. Therefore, in a specified interval, (M_i, M_{i+1}), we have the following boundary values:

$$C_{L_\alpha}(M_i),\ C_{L_\alpha}(M_{i+1}),\ C_{L_\alpha}'(M_i),\ C_{L_\alpha}'(M_{i+1}) \qquad \text{(I-4)}$$

TABLE 4. Lift and drag parameter slopes with Mach number

M	0	0.8	0.9	1.0	1.2	1.4	1.6	1.8
C_{L_α}'	0	0	6.75	0	-4.0625	-1.25	-0.833	-0.2
C_{D_o}'	0	0	0.0231	0.155	0	-0.0157	-0.0106	0
η'	0	0	0.85	0.325	0.25	0.2	0.1313	0

$C_{L_\alpha}' = dC_{L_\alpha}/dM$ $\quad C_{D_o}' = dC_{D_o}/dM$ $\quad \eta' = d\eta/dM$

The polynomial coefficients a_0, a_1, a_2, and a_3 can then be obtained by solving equations (I-1) and (I-2) at each end of the interval, i.e.,

$$C_{L_\alpha}(M_i) = a_0 \tag{I-5}$$

$$C_{L_\alpha}(M_{i+1}) = a_0 + a_1 \Delta + a_2 \Delta^2 + a_3 \Delta^3 \tag{I-6}$$

$$C_{L_\alpha}'(M_i) = a_1 \tag{I-7}$$

$$C_{L_\alpha}'(M_{i+1}) = a_1 + 2 a_2 \Delta + 3 a_3 \Delta^2 \tag{I-8}$$

from which we get

$$a_0 = C_{L_\alpha}(M_i)$$

$$a_1 = C_{L_\alpha}'(M_i)$$

$$a_2 = \frac{3[C_{L_\alpha}(M_{i+1}) - C_{L_\alpha}(M_i)] - \Delta[C_{L_\alpha}'(M_{i+1}) + 2\,C_{L_\alpha}'(M_i)]}{\Delta^2}$$

$$a_3 = \frac{-2[C_{l_\alpha}(M_{i+1}) - C_{L_\alpha}(M_i)] + \Delta[C_{L_\alpha}'(M_{i+1}) + C_{L_\alpha}'(M_i)}{\Delta^3}$$

Here, $\Delta = M_{i+1} - M_i$. These coefficient are then used in equation (I-1) to calculate $C_{L_\alpha}(M)$ on the interval (M_i, M_{i+1}). The same procedure is repeated for the next interval and so on. The zero-lift drag coefficient, $C_{D_o}(M)$, and the efficiency factor, $\eta(M)$, are determined in the same manner. Table 5, 6, and 7 show the polynomial coefficients for C_{L_α}, C_{D_o}, and η, respectively, for each of the seven Mach number intervals.

Both the atmospheric density, ρ, and the speed of sound, a, vary with altitude. For the density, we have [31]:

$$\rho(h) = \rho_0\, e^{-h/h_1}$$

where $\rho_0 = 2.54 \times 10^{3}$ slug/ft^3 and $h_1 = 2.73 \times 10^{4}$ ft. For the speed of sound, we have [31]:

$$a(h) = \begin{cases} (k_1 - k_2 h)^{1/2}\,, & h \le 36{,}000 \text{ ft} \\ 968.1 \text{ ft/sec}\,, & h > 36{,}000 \text{ ft} \end{cases}$$

where $k_1 = 1.244 \times 10^{6}$ ft^2/sec^2, and $k_2 = 8.57$ ft/sec^2.

TABLE 5. Polynomial coefficients for C_{L_α}

Mach number interval	Polynomial coefficients for C_{L_α}			
	a_0	a_1	a_2	a_3
0 - 0.8	3.44	0	0	0
0.8 - 0.9	3.44	0	-25.50	395.00
0.9 - 1.0	3.58	6.75	123.00	-1045.00
1.0 - 1.2	4.44	0	-54.69	148.44
1.2 - 1.4	3.44	-4.06	14.63	-25.31
1.4 - 1.6	3.01	-1.25	5.42	-14.58
1.6 - 1.8	2.86	-0.83	-22.17	79.17

TABLE 6. Polynomial coefficients for C_{D_o}

Mach number interval	Polynomial coefficients for C_{D_o}			
	a_0	a_1	a_2	a_3
0 - 0.8	0.0130	0	0	0
0.8 - 0.9	0.0130	0	0.0688	0.3125
0.9 - 1.0	0.0140	0.0232	3.0875	-16.1875
1.0 - 1.2	0.0310	0.1550	-0.8000	1.3750
1.2 - 1.4	0.0410	0	-0.0714	0.1071
1.4 - 1.6	0.0390	-0.0157	-0.0147	0.0915
1.6 - 1.8	0.0360	-0.0106	0.0313	-0.0156

TABLE 7. Polynomial coefficients for η

Mach number interval	Polynomial coefficients for η			
	a_0	a_1	a_2	a_3
0 - 0.8	0.540	0	0	0
0.8 - 0.9	0.540	0	54.500	-335.000
0.9 - 1.0	0.750	0.850	-8.250	37.500
1.0 - 1.2	0.790	0.325	-0.375	0.625
1.2 - 1.4	0.845	0.250	-0.125	0
1.4 - 1.6	0.890	0.200	0.344	-1.719
1.6 - 1.8	0.930	0.131	-1.313	3.281

APPENDIX II. THRUST CHARACTERISTICS

In this Appendix, we discuss the curve fit for the thrust system of the F-4 fighter aircraft [31]. Thrust varies with both Mach number and altitude. These data [31] are given in Table 8. Like the aerodynamic characteristics curve fitting, we need a continuous function with continuous first derivative to approximate the thrust data. A fourth-order polynomial least-squares fit of Mach number and altitude has been chosen.

In this approximation, the form for the thrust fitting is

$$T(M, h) = [1, M, M^2, M^3, M^4] [A] \begin{bmatrix} 1 \\ h \\ h^2 \\ h^3 \\ h^4 \end{bmatrix}$$

where [A] is a 5x5 matrix of constant values to be determined to best represent the thrust data. By performing a least-squares analysis to these data, we obtain a set of twenty-five linear equations and twenty-five unknowns. These unknowns are the elements in the matrix [A]. The results are presented in Table 9. Table 10 shows the thrust values calculated from this fourth-order polynomial least squares fit. It can be seen by comparing tables 7 and 5

that these values approximate very closely the actual thrust data of the F-4 fighter aircraft.

TABLE 8. Thrust data, T (lbs. x 10^{-3}), with Mach number and altitude [31]

Mach Number,	Altitude, h (ft x 10^{-3})									
M	0	5	10	15	20	25	30	40	50	70
0	24.2									
0.2	28.0	24.6	21.1	18.1	15.2	12.8	10.7			
0.4	28.3	25.2	21.9	18.7	15.9	13.4	11.2	7.3	4.4	
0.6	30.8	27.2	23.8	20.5	17.3	14.7	12.3	8.1	4.9	
0.8	34.5	30.3	26.6	23.2	19.8	16.8	14.1	9.4	5.6	1.1
1.0	37.9	34.3	30.4	26.8	23.3	19.8	16.8	11.2	6.8	1.4
1.2	36.1	38.0	34.9	31.3	27.3	23.6	20.1	13.4	8.3	1.7
1.4		36.6	38.5	36.1	31.6	28.1	24.2	16.2	10.0	2.2
1.6				38.7	35.7	32.0	28.1	19.3	11.9	2.9
1.8						34.6	31.3	21.7	13.3	3.1

TABLE 9. Least-squares results for the matrix [A]

($lbs \times 10^{-3}$)	($lbs\text{-}ft^{-1} \times 10$)	($lbs\text{-}ft^{-2} \times 10^{5}$)	($lbs\text{-}ft^{-3} \times 10^{10}$)	($lbs\text{-}ft^{-4} \times 10^{15}$)
$A_{11} = 30.21$	$A_{12} = -0.6682$	$A_{13} = -6.877$	$A_{14} = 19.51$	$A_{15} = -15.12$
$A_{21} = -33.80$	$A_{22} = 3.347$	$A_{23} = 18.13$	$A_{24} = -58.65$	$A_{25} = 47.57$
$A_{31} = 100.8$	$A_{32} = -77.56$	$A_{33} = 5.441$	$A_{34} = 28.64$	$A_{35} = -33.55$
$A_{41} = -78.99$	$A_{42} = 101.4$	$A_{43} = -30.28$	$A_{44} = 32.36$	$A_{45} = -10.89$
$A_{51} = 18.74$	$A_{52} = -31.60$	$A_{53} = 12.04$	$A_{54} = -17.85$	$A_{55} = 9.417$

TABLE 10. Thrust data, T (lbs. x 10^{-3}), from fourth-order polynomial least-squares fit

Mach Number, M	Altitude, h (ft x 10^{-3})									
	0	5	10	15	20	25	30	40	50	70
0.4	28.24	25.14	22.01	18.95	16.08	13.46	11.12	7.34	4.45	-1.93
0.6	31.58	27.43	23.72	20.40	17.43	14.77	12.38	8.27	4.85	-0.52
0.8	34.92	30.72	26.81	23.19	19.85	16.81	14.04	9.37	5.69	0.61
1.0	36.96	34.08	30.71	27.08	23.41	19.86	16.55	10.98	6.95	1.40
1.2	37.17	36.80	34.75	31.60	27.82	23.82	19.92	13.22	8.56	1.89
1.4	35.70	38.37	38.18	35.96	32.45	28.28	23.92	16.04	10.34	2.14
1.6	33.45	38.55	40.14	39.14	36.35	32.45	28.00	19.17	12.03	2.28
1.8	32.02	37.26	39.68	39.82	38.16	35.18	31.27	22.15	13.25	2.48

APPENDIX III. OPTIMIZATION PROGRAM LISTING

A complete listing of the optimization program that solves the most complex model, i.e., Model 1, is given in this Appendix. A sample input to the optimization program, and a sample output from Sequential Quadratic Programming are also given.

Program listing for Model 1

```
//MODEL5 JOB I3546,'SHAW ONG'
/*JOBPARM LINES=8,DEST=RMT11
//S1 EXEC FORTVLG,FVPOPT=2,REGION.GO=512K,TIME.GO=4
//FORT.SYSIN DD *
C*********************************************************
C*                                                       *
C*    THIS IS THE DRIVER PROGRAM. IT SETS UP SQP TO      *
C*    SOLVE THESIS MODEL 1.                              *
C*                                                       *
C*********************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C-----DEFINE STORAGE REQUIREMENTS.
C
      PARAMETER (MAXX=3000)
      PARAMETER (MAXF=2   )
      PARAMETER (MAXG=100 )
      PARAMETER (MAXH=100 )
      PARAMETER (MAXIO=12)
      DIMENSION N(6),IO(MAXIO)
      DIMENSION X(MAXX),F(MAXF),H(MAXH),G(MAXG)
      DIMENSION MD(20)
C
C-----COMMON BLOCKS
C
      COMMON / STATE / Y1(102),Y2(102),Y3(102),
     >                 Y4(102),Y5(102)
      COMMON / CNTRL / U(102)
      COMMON / INTPLN / INTPLN
C
      NAMELIST / TYPE / INTPLN
C
      EXTERNAL EVAL
```

```
C
C-----SET INPUT/OUTPUT PARAMETERS.
C
      IPFLAG=1
      IPRINT=2
      ICARD=2
      IIN=5
      IOUT=6
      IPUNCH=7
      ISCALE=2
      IO(1)=IPFLAG
      IO(2)=IPRINT
      IO(6)=ICARD
      IO(7)=IIN
      IO(8)=IOUT
      IO(9)=IPUNCH
      IO(11)=ISCALE
C
C-----DEFINE VECTOR STORAGE SIZE
C
      N(4)=MAXX
      N(5)=MAXH
      N(6)=MAXG
C
C-----DEFINE PROBLEM SIZE
C
      NX=16
      NG=10
      NH=2
      N(1)=NX
      N(2)=NH
      N(3)=NG
C
      CALL ERRSET(208,256,-1,1,0,0)
C
C-----INPUT NAMELIST IF ANY
C
      READ(5,TYPE)
      CALL RQP(EVAL,N,X,F,G,H,IO,MD,0)
C
C-----PERFORM OPTIMIZATION
C
      CALL RQP(EVAL,N,X,F,G,H,IO,MD,-1)
C
C-----CONTROL AND STATE VARIABLES OUTPUT
C
      DO 3 I=1,NX
         WRITE(10,*) X(I)
3     CONTINUE
      DO 4 I=1,101
```

```
         WRITE(11,2000) Y1(I)
         WRITE(12,2000) Y2(I)
         WRITE(15,2000) Y3(I)
         WRITE(16,2000) Y4(I)
         WRITE(17,2000) Y5(I)
         WRITE(13,2000) U(I)
4     CONTINUE
C
2000  FORMAT(F15.6)
      STOP
      END
      SUBROUTINE EVAL(N,X,F,G,H,IO,IER)
C
C*************************************************************
C*                                                           *
C*    THIS IS THE USER PROVIDED SUBROUTINE. IT IS            *
C*    CALLED BY THE SQP OPTIMIZATION ROUTINE TO              *
C*    EVALUATE THE OBJECTIVE FUNCTION AND THE                *
C*    CONSTRAINTS.                                           *
C*                                                           *
C*    SUBROUTINES REQUIRED:                                  *
C*            RK4 -- RUNGE KUTTA 4TH ORDER                   *
C*            FGH -- EVALUATING CONSTRAINTS AND              *
C*                   PERFORMANCE INDEX                       *
C*         SPLINE -- CUBIC SPLINE INTERPOLATION              *
C*         LINEAR -- LINEAR INTERPOLATION                    *
C*                                                           *
C*************************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION N(*),X(*),F(*),G(*),H(*),IO(*)
      DIMENSION XIN(102),YIN(102),XOUT(102),YOUT(102)
C
C-----COMMON BLOCKS
C
      COMMON / STATE / Y1(102),Y2(102),Y3(102),
     >                 Y4(102),Y5(102)
      COMMON / CNTRL / U(102)
      COMMON / INTPLN / INTPLN
C
      IPRINT=IO(2)
      IOUT=IO(8)
      NX=N(1)
      NG=N(3)
      NH=N(2)
      IER=0
      IGO=1
      IF (IO(12) .EQ. 0) IGO=7
C
C-----OUTPUT CONTROL VECTOR U
```

```
C
      IF (IPRINT .GE. 5) THEN
          WRITE(IOUT,1000) (I,X(I),I=1,NX)
      ELSE
          IF (IGO .EQ. 7) THEN
              IF (IPRINT .GE. 3 .AND. IPRINT .LE. 4) THEN
                   WRITE(IOUT,1000) (I,X(I),I=1,NX)
              END IF
           END IF
      END IF
C
C-----SETTING UP DATA FOR CUBIC SPLINE/LINEAR INTERPOLATION
C
      DT=0.01D0*X(16)
      JOUT=101
      NSTEP=JOUT
      XIN(1)=0.D0
      XIN(2)=.02D0*X(16)
      XIN(3)=.04D0*X(16)
      XIN(4)=.06D0*X(16)
      XIN(5)=.08D0*X(16)
      XIN(6)=.10D0*X(16)
      XIN(7)=.20D0*X(16)
      XIN(8)=.30D0*X(16)
      XIN(9)=.40D0*X(16)
      XIN(10)=.50D0*X(16)
      XIN(11)=.60D0*X(16)
      XIN(12)=.70D0*X(16)
      XIN(13)=.80D0*X(16)
      XIN(14)=.90D0*X(16)
      XIN(15)=X(16)
      XOUT(1)=0.D0
      XOUT(JOUT)=X(16)
      DO 15 I=2,JOUT-1
         XOUT(I)=XOUT(I-1)+DT
15    CONTINUE
C
      U(102)=X(16)
C
C-----INTERPOLATION FLAG:     1 - CUBIC
C-----                        2 - LINEAR
C
      IF (INTPLN .EQ. 1) GO TO 17
      IF (INTPLN .EQ. 2) GO TO 18
17    CALL SPLINE(15,XIN,X,JOUT,XOUT,U,IERR)
      GO TO 19
18    CALL LINEAR(15,XIN,X,JOUT,XOUT,U,IERR)
      GO TO 19
19    CONTINUE
C
```

```
C-----INITIAL STATE VALUES
C
      NN=5
      Y1(1)=400.D0
      Y2(1)=0.D0
      Y3(1)=0.D0
      Y4(1)=0.D0
      Y5(1)=1305.D0
C
C-----EVALUATE STATE VARIABLES
C
      CALL RK4(NN,NSTEP,DT)
C
C-----EVALUATE CONSTRAINTS AND PERFORMANCE INDEX
C
      CALL FGH(F,G,H,DT)
C
C-----OUTPUT CONSTRAINTS AND OBJECTIVE FUNCTION
C
      IF (IPRINT .GE. 5) THEN
        WRITE(IOUT,1001) F(1)
        IF (NH .NE. 0) WRITE(IOUT,1002) (H(I),I=1,NH)
        IF (NG .NE. 0) WRITE(IOUT,1003) (G(I),I=1,NG)
      ELSE
        IF (IGO .EQ. 7) THEN
          IF (IPRINT .GE. 3 .AND. IPRINT .LE. 4) THEN
             WRITE(IOUT,1001) F(1)
             IF (NH .NE. 0) WRITE(IOUT,1002) (H(I),I=1,NH)
             IF (NG .NE. 0) WRITE(IOUT,1003) (G(I),I=1,NG)
          END IF
        END IF
      END IF
C
1000  FORMAT('0',6X,'U(CONT. VAR) =',5(1X,I3,1X,E15.8)/,1X,
     >     40(6(1X,I3,1X,E15.8)/,1X))
1001  FORMAT(1X,'OBJ. FUNCTION =',2X,E16.8)
1002  FORMAT(1X,'EQUALITIES =',5X,6(E15.8,2X)/,
     >     15(17X,6(E15.8,2X)/))
1003  FORMAT(1X,'INEQUALITIES =',3X,6(E15.8,2X)/,
     >     15(17X,6(E15.8,2X)/))
      RETURN
      END
      SUBROUTINE RK4(NN,NSTEP,DT)
C
C*************************************************************
C*                                                           *
C*    THIS SUBROUTINE USES RUNGE KUTTA 4TH ORDER FOR         *
C*    NUMERICAL INTEGRATION                                  *
C*                                                           *
C*    FUNCTION REQUIRED:                                     *
```

```
C*          RK4RHS -- EVALUATING RHS OF THE STATE          *
C*                    EQUATIONS                            *
C*                                                         *
C***********************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Y(10),YS(10),YSS(10),YSSS(10),T1(10),
     >          T2(10),T3(10),T4(10)
C
C-----COMMON BLOCKS
C
      COMMON / STATE / Y1(102),Y2(102),Y3(102),
     >                 Y4(102),Y5(102)
      COMMON / CNTRL / U(102)
C
C-----INITIAL STATE VALUES
C
      Y(1)=400.D0
      Y(2)=0.D0
      Y(3)=0.D0
      Y(4)=0.D0
      Y(5)=1305.D0
C
C-----THE MAIN LOOP
C
      DO 20 I=2,NSTEP
C
C-----TEMPORARY ARRAYS NEEDED FOR THE FUNCTIONS TO SAVE
C-----THEM FOR THE FINAL CORRECTOR STEP
C
C-----FIRST (HALF STEP) PREDICTOR
C
         DO 22 J=1,NN
            T1(J)=RK4RHS(J,Y,I)
            YS(J)=Y(J)+.5D0*DT*T1(J)
22       CONTINUE
C
C-----SECOND STEP (HALF STEP CORRECTOR)
C
         DO 24 J=1,NN
            T2(J)=RK4RHS(J,YS,I)
            YSS(J)=Y(J)+.5D0*DT*T2(J)
24       CONTINUE
C
C-----THIRD STEP (FULL STEP MID-POINT PREDICTOR)
C
         DO 26 J=1,NN
            T3(J)=RK4RHS(J,YSS,I)
            YSSS(J)=Y(J)+DT*T3(J)
26       CONTINUE
```

```
C
C-----FINAL STEP (SIMPSON'S RULE CORRECTOR)
C
         DO 28 J=1,NN
            T4(J)=RK4RHS(J,YSSS,I)
            Y(J)=Y(J)+DT/6.D0*(T1(J)+2.D0*(T2(J)+T3(J))+T4(J))
28       CONTINUE
C
C-----STORING STATE VARIABLES AT EACH INTEGRATING STEP
C
         Y1(I)=Y(1)
         Y2(I)=Y(2)
         Y3(I)=Y(3)
         Y4(I)=Y(4)
         Y5(I)=Y(5)
20     CONTINUE
       RETURN
       END
       SUBROUTINE FGH(F,G,H,DT)
C
C*********************************************************
C*                                                       *
C*     THIS SUBROUTINE EVALUATES THE CONSTRAINTS AND     *
C*     THE PERFORMANCE INDEX.                            *
C*                                                       *
C*********************************************************
C
       IMPLICIT DOUBLE PRECISION (A-H,O-Z)
       DIMENSION F(2),G(*),H(*)
C
C-----COMMON BLOCKS
C
       COMMON / STATE / Y1(102),Y2(102),Y3(102),
     >                  Y4(102),Y5(102)
       COMMON / CNTRL / U(102)
C
C-----EVALUATING PERF. INDEX
C
       F(1)=U(102)
C
C-----EVALUATING CONSTRAINTS
C
       H(1)=Y3(101)-65600.D0
       H(2)=Y1(101)-968.1D0
       DO 30 I=1,10
          G(I)=Y3(I+1)
30     CONTINUE
C
       RETURN
       END
```

```
      REAL FUNCTION RK4RHS(J,Y,II)
C
C*****************************************************
C*                                                   *
C*    THIS SUBROUTINE EVALUATES THE RIGHT HAND SIDE  *
C*    OF THE STATE EQUATIONS                         *
C*                                                   *
C*****************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Y(10)
      REAL LIFT
      COMMON / CNTRL / U(102)
C
C-----CONSTANTS
C
      DNSTY=.0023764D0
      S=530.D0
      AR=3.D0
      G=32.174D0
C
C-----ELEMENTS OF MATRIX [A]
C
      Q11=30.21D3
      Q12=-.6682D-1
      Q13=-6.877D-5
      Q14=19.51D-10
      Q15=-15.12D-15
C
      Q21=-33.8D3
      Q22=3.347D-1
      Q23=18.13D-5
      Q24=-58.65D-10
      Q25=47.57D-15
C
      Q31=100.8D3
      Q32=-77.56D-1
      Q33=5.441D-5
      Q34=28.64D-10
      Q35=-33.55D-15
C
      Q41=-78.99D3
      Q42=101.4D-1
      Q43=-30.28D-5
      Q44=32.36D-10
      Q45=-10.89D-15
C
      Q51=18.74D3
      Q52=-31.6D-1
      Q53=12.04D-5
```

```
      Q54=-17.85D-10
      Q55=9.417D-15
C
C-----EVALUATING SPEED OF SOUND
C
      IF (Y(3) .LT. 36000.D0) THEN
          SPSND=DSQRT(1.244D6-8.57D0*Y(3))
      ELSE
          SPSND=968.1D0
      END IF
C
C-----THRUST CALCULATIONS
C
      A=Y(1)/SPSND
      A2=A**2
      A3=A**3
      A4=A**4
      HT=Y(3)
      H2=HT**2
      H3=HT**3
      H4=HT**4
      THRST=Q11+A*Q21+A2*Q31+A3*Q41+A4*Q51+HT*(Q12+A*Q22+
     >      A2*Q32+A3*Q42+A4*Q52)+H2*(Q13+A*Q23+A2*Q33+
     >      A3*Q43+A4*Q53)+H3*(Q14+A*Q24+A2*Q34+A3*Q44+
     >      A4*Q54)+H4*(Q15+A*Q25+A2*Q35+
     >      A3*Q45+A4*Q55)
C
C-----FLAG TO DECIDE WHAT AERODYNAMIC CHARACTERISTIC
C-----VALUES TO USE
C
      IF (A .LE. 0.8D0) GO TO 4100
      IF (A .GT. 0.8D0 .AND. A .LE. 0.9D0) GO TO 4110
      IF (A .GT. 0.9D0 .AND. A .LE. 1.0D0) GO TO 4120
      IF (A .GT. 1.0D0 .AND. A .LE. 1.2D0) GO TO 4130
      IF (A .GT. 1.2D0 .AND. A .LE. 1.4D0) GO TO 4140
      IF (A .GT. 1.4D0 .AND. A .LE. 1.6D0) GO TO 4150
      IF (A .GT. 1.6D0) GO TO 4160
C
4100  CLA=3.44D0
      CDO=0.013D0
      ETA=0.54D0
      GO TO 4190
C
4110  AA=A-0.8D0
      CLA=3.44D0-25.5D0*AA**2+395.D0*AA**3
      CDO=0.013D0+0.06875D0*AA**2+0.3125D0*AA**3
      ETA=0.54D0+54.5D0*AA**2-335.D0*AA**3
      GO TO 4190
C
4120  AA=A-0.9D0
```

```
      CLA=3.58D0+6.75D0*AA+123.D0*AA**2-1045.D0*AA**3
      CDO=0.014D0+0.023125D0*AA+3.0875D0*AA**2-16.1875D0*AA**3
      ETA=0.75D0+0.85D0*AA-8.25D0*AA**2+37.5D0*AA**3
      GO TO 4190
C
4130  AA=A-1.0D0
      CLA=4.44D0-54.6875D0*AA**2+148.4375D0*AA**3
      CDO=0.031D0+0.155D0*AA-0.8D0*AA**2+1.375D0*AA**3
      ETA=0.79D0+0.325D0*AA-0.375D0*AA**2+0.625D0*AA**3
      GO TO 4190
C
4140  AA=A-1.2D0
      CLA=3.44D0-4.0625D0*AA+14.625D0*AA**2-25.3125D0*AA**3
      CDO=0.041D0-0.0714285D0*AA**2+0.10714125D0*AA**3
      ETA=0.845D0+0.25D0*AA-0.125D0*AA**2
      GO TO 4190
C
4150  AA=A-1.4D0
      CLA=3.01D0-1.25D0*AA+5.416665D0*AA**2-14.583325D0*AA**3
      CDO=0.039D0-0.0157143D0*AA-0.014732D0*AA**2+0.0915175D0*AA**3
      ETA=0.89D0+0.2D0*AA+0.34375D0*AA**2-1.71875D0*AA**3
      GO TO 4190
C
4160  AA=A-1.6D0
      CLA=2.86D0-0.8333D0*AA-22.1667D0*AA**2+79.16675D0*AA**3
      CDO=0.036D0-0.010625D0*AA+0.03125D0*AA**2-0.015625D0*AA**3
      ETA=0.93D0+0.13125D0*AA-1.3125D0*AA**2+3.28125D0*AA**3
      GO TO 4190
C
4190  CONTINUE
C
C-----LIFT AND DRAG CALCULATIONS
C
      UMID=(U(II-1)+U(II))/2.D0
      DRAG=.5D0*.00254D0*DEXP(-.00003663D0*Y(3))*Y(1)**2*
     >     S*(CDO+ETA*CLA*UMID**2)
      LIFT=.5D0*.00254D0*DEXP(-.00003663D0*Y(3))*Y(1)**2*
     >     S*CLA*UMID
C
C-----FLAG TO DIRECT THE EVALUATION OF
C-----THE APPROPRIATE RHS STATE EQUATION
C
      IF ( J .EQ. 1 ) GO TO 3000
      IF ( J .EQ. 2 ) GO TO 3010
      IF ( J .EQ. 3 ) GO TO 3020
      IF ( J .EQ. 4 ) GO TO 3030
      IF ( J .EQ. 5 ) GO TO 3040
C
3000      RK4RHS=(THRST*DCOS(UMID)-DRAG-Y(5)*G*DSIN(Y(2)))/
     >           Y(5)
```

```
          GO TO 3050
C
3010      RK4RHS=(THRST*DSIN(UMID)+LIFT-Y(5)*G*DCOS(Y(2)))/
     >           (Y(5)*Y(1))
          GO TO 3050
C
3020      RK4RHS=Y(1)*DSIN(Y(2))
          GO TO 3050
C
3030      RK4RHS=Y(1)*DCOS(Y(2))
          GO TO 3050
C
3040      RK4RHS=-THRST/(1600.D0*G)
          GO TO 3050
C
3050  CONTINUE
      RETURN
      END
      SUBROUTINE SPLINE(IIN,XIN,YIN,JOUT,XOUT,YOUT,IERR)
C
C*********************************************************
C*                                                       *
C*  THIS SUBROUTINE COMPUTES A CUBIC SPLINE FOR THE      *
C*                                                       *
C*  GIVEN DATA AND RETURNS INTERPOLATED VALUES OF THE    *
C*                                                       *
C*  FUNCTION AT SPECIFIED X LOCATIONS.                   *
C*                                                       *
C**                                                      *
C** NOTE:  INPUT DATA **MUST** BE IN ORDER OF            *
C**        INCREASING X !!!                              *
C**                                                      *
C*                                                       *
C*  VARIABLES:                                           *
C*                                                       *
C*  IIN = NUMBER OF INPUT DATA POINTS - IIN <= 25        *
C*  XIN(I) = LOCATION OF INPUT DATA POINTS               *
C*  YIN(I) = VALUE OF THE FUNCTION TO BE INTERPOLATED    *
C*           AT X=XIN(I)                                 *
C*  JOUT = NUMBER OF VALUES TO BE INTERPOLATED TO        *
C*  XOUT(J) = X LOCATIONS USED FOR INTERPOLATION         *
C*  YOUT(J) = INTERPOLATED VALUE OF Y AT XOUT(J)         *
C*                                                       *
C** IERR = ERROR FLAG (PLEASE CHECK THIS VARIABLE)       *
C*                                                       *
C*     IERR=0 - INTERPOLATION OK                         *
C*     IERR=1 - PROBLEMS WERE ENCOUNTERED DURING         *
C*              INTERPOLATION                            *
C*                                                       *
C*********************************************************
```

```
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION XIN(*),YIN(*),XOUT(*),YOUT(*)
      DIMENSION YPP(25),C1(25),C2(25),DX(25),DY(25)
      DIMENSION A(25),B(25),C(25),D(25)
C
C-----ZERO ERROR FLAG
C
      IERR=0
C
C-----SET VALUES OF SECOND DERIVATIVES AT ENDPOINTS
C-----ZERO VALUES INDICATE NATURAL CUBIC SPLINE
C
      YPP(1)=0.D0
      YPP(IIN)=0.D0
C
C-----SET UP TRIDIAGONAL MATRIX
C
      DO 40 I=1,IIN-1
      DX(I)=XIN(I+1)-XIN(I)
      DY(I)=YIN(I+1)-YIN(I)
40    CONTINUE
      IL=2
      IU=IIN-1
      DO 50 I=IL,IU
      A(I)=DX(I)
      D(I)=2.D0*(DX(I)+DX(I-1))
      B(I)=DX(I-1)
      C(I)=6.D0*(DY(I)/DX(I)-DY(I-1)/DX(I-1))
50    CONTINUE
      C(2)=C(2)-DX(1)*YPP(1)
      C(IU)=C(IU)-DX(IU)*YPP(IU+1)
      B(IL)=0.D0
      A(IU)=0.D0
C
C-----INVERT TRIDIAGONAL MATRIX TO OBTAIN YPP
C
      CALL SY(IL,IU,B,D,A,C)
      DO 60 I=2,IIN-1
      YPP(I)=C(I)
60    CONTINUE
      DO 70 I=1,IIN-1
      C1(I)=YIN(I+1)/DX(I)-YPP(I+1)*DX(I)/6.D0
      C2(I)=YIN(I)/DX(I)-YPP(I)*DX(I)/6.D0
70    CONTINUE
C
C-----DETERMINE SECTION OF CUBIC SPLINE FOR INTERPOLATION
C
      DO 120 J=1,JOUT
      IF (XOUT(J) .LT. XIN(1)) THEN
```

```
C
C-----OUTPUT LOCATION OUTSIDE INPUT DATA RANGE ( TOO
C-----SMALL )
C
      WRITE(5,80) J
80    FORMAT(//' ***** XOUT(',I2,') < XIN(1) -CHECK
     >YOUR DATA ***')
      IERR=1
      RETURN
      END IF
      IF (XOUT(J) .GT. XIN(IIN)) THEN
C
C-----OUTPUT LOCATION OUTSIDE INPUT DATA RANGE ( TOO
C-----LARGE )
C
      WRITE(5,90) J
90    FORMAT(//' ***** XOUT(',I2,') > XIN(IIN) - CHECK
     >YOUR DATA ***')
      IERR=1
      RETURN
      END IF
      DO 100 I=2,IIN
      IF (XOUT(J) .LE. XIN(I)) GO TO 110
100   CONTINUE
110   CONTINUE
      IS=I-1
C
C-----INTERPOLATED VALUE OF YOUT(J)
C
      YOUT(J)=YPP(IS)/(6.D0*DX(IS))*(XIN(IS+1)-XOUT(J))**3
     > +YPP(IS+1)/(6.D0*DX(IS))*(XOUT(J)-XIN(IS))**3
     > +C1(IS)*(XOUT(J)-XIN(IS))+C2(IS)*(XIN(IS+1)-XOUT(J))
120   CONTINUE
      RETURN
      END
C
      SUBROUTINE SY(IL,IU,BB,DD,AA,CC)
C
C*********************************************************
C*                                                       *
C*    SUBROUTINE SY SOLVES TRIDIAGONAL                   *
C*    SYSTEM BY ELIMINATION                              *
C*    IL = SUBSCRIPT OF FIRST EQUATION                   *
C*    IU = SUBSCRIPT OF LAST EQUATION                    *
C*    BB = COEFFICIENT BEHIND DIAGONAL                   *
C*    DD = COEFFICIENT ON DIAGONAL                       *
C*    AA = COEFFICIENT AHEAD OF DIAGONAL                 *
C*    CC = ELEMENT OF CONSTANT VECTOR                    *
C*                                                       *
C*********************************************************
```

```
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION AA(1),BB(1),CC(1),DD(1)
C
C-----ESTABLISH UPPER TRIANGULAR MATRIX
C
      LP=IL+1
      DO 130 I=LP,IU
      R=BB(I)/DD(I-1)
      DD(I)=DD(I)-R*AA(I-1)
      CC(I)=CC(I)-R*CC(I-1)
130   CONTINUE
C
C-----BACK SUBSTITUTION
C
      CC(IU)=CC(IU)/DD(IU)
      DO 140 I=LP,IU
      J=IU-I+IL
      CC(J)=(CC(J)-AA(J)*CC(J+1))/DD(J)
140   CONTINUE
C
C-----SOLUTION STORED IN CC
C
      RETURN
      END
      SUBROUTINE LINEAR(IIN,XIN,YIN,JOUT,XOUT,YOUT,IERR)
C
C*********************************************************
C*                                                       *
C*  THIS SUBROUTINE COMPUTES A LINEAR FIT FOR THE        *
C*                                                       *
C*  GIVEN DATA AND RETURNS INTERPOLATED VALUES OF THE    *
C*                                                       *
C*  FUNCTION AT SPECIFIED X LOCATIONS.                   *
C*                                                       *
C**                                                      *
C** NOTE:  INPUT DATA **MUST** BE IN ORDER OF            *
C**        INCREASING X !!!                              *
C**                                                      *
C*                                                       *
C*  VARIABLES:                                           *
C*                                                       *
C*  IIN = NUMBER OF INPUT DATA POINTS - IIN <= 25        *
C*  XIN(I) = LOCATION OF INPUT DATA POINTS               *
C*  YIN(I) = VALUE OF THE FUNCTION TO BE INTERPOLATED    *
C*           AT X=XIN(I)                                 *
C*  JOUT = NUMBER OF VALUES TO BE INTERPOLATED TO        *
C*  XOUT(J) = X LOCATIONS USED FOR INTERPOLATION         *
C*  YOUT(J) = INTERPOLATED VALUE OF Y AT XOUT(J)         *
C*                                                       *
```

```
C** IERR = ERROR FLAG (PLEASE CHECK THIS VARIABLE)      *
C*                                                      *
C*     IERR=0 - INTERPOLATION OK                        *
C*     IERR=1 - PROBLEMS WERE ENCOUNTERED DURING        *
C*              INTERPOLATION                           *
C*                                                      *
C********************************************************
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION XIN(*),YIN(*),XOUT(*),YOUT(*)
C
C-----ZERO ERROR FLAG
C
      IERR=0
C
C-----DETERMINE SECTION OF LINEAR FIT FOR INTERPOLATION
C
      DO 120 J=1,JOUT
      IF (XOUT(J) .LT. XIN(1)) THEN
C
C-----OUTPUT LOCATION OUTSIDE INPUT DATA RANGE ( TOO
C-----SMALL )
C
      WRITE(5,80) J
80    FORMAT(//' ***** XOUT(',I2,') < XIN(1) -CHECK
     >YOUR DATA ***')
      IERR=1
      RETURN
      END IF
      IF (XOUT(J) .GT. XIN(IIN)) THEN
C
C-----OUTPUT LOCATION OUTSIDE INPUT DATA RANGE ( TOO
C-----LARGE )
C
      WRITE(5,90) J
90    FORMAT(//' ***** XOUT(',I2,') > XIN(IIN) - CHECK
     >YOUR DATA ***')
      IERR=1
      RETURN
      END IF
      DO 100 I=2,IIN
      IF (XOUT(J) .LE. XIN(I)) GO TO 110
100   CONTINUE
110   CONTINUE
      IS=I-1
C
C-----INTERPOLATED VALUE OF YOUT(J)
C
      YOUT(J)=(YIN(I)-YIN(IS))/(XIN(I)-XIN(IS))*(XOUT(J)-XIN(IS))+
     >        YIN(IS)
```

```
120   CONTINUE
      RETURN
      END
//LKED.SYSLIN DD
//     DD DSN=N.I3137.OBJECT.LIB(SQP),DISP=SHR
//     DD DSN=N.I3137.OBJECT.LIB(SECOND),DISP=SHR
//GO.FT05F001 DD DSN=S.I3546.MODEL55.INPUT,DISP=SHR
//GO.FT06F001 DD SYSOUT=A
//GO.FT07F001 DD DSN=S.I3546.MODEL55.REST,
//   UNIT=DISK,
//   DISP=(NEW,CATLG),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160),
//   SPACE=(TRK,(10,10),RLSE)
//GO.FT10F001 DD DSN=S.I3546.CONTROL.DAT,
//   UNIT=DISK,
//   DISP=(NEW,CATLG),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160),
//   SPACE=(TRK,(10,10),RLSE)
//GO.FT11F001 DD DSN=S.I3546.YY1.DAT,
//   UNIT=DISK,
//   DISP=(NEW,CATLG),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160),
//   SPACE=(TRK,(10,10),RLSE)
//GO.FT12F001 DD DSN=S.I3546.YY2.DAT,
//   UNIT=DISK,
//   DISP=(NEW,CATLG),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160),
//   SPACE=(TRK,(10,10),RLSE)
//GO.FT15F001 DD DSN=S.I3546.YY3.DAT,
//   UNIT=DISK,
//   DISP=(NEW,CATLG),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160),
//   SPACE=(TRK,(10,10),RLSE)
//GO.FT16F001 DD DSN=S.I3546.YY4.DAT,
//   UNIT=DISK,
//   DISP=(NEW,CATLG),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160),
//   SPACE=(TRK,(10,10),RLSE)
//GO.FT17F001 DD DSN=S.I3546.YY5.DAT,
//   UNIT=DISK,
//   DISP=(NEW,CATLG),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160),
//   SPACE=(TRK,(10,10),RLSE)
//GO.FT13F001 DD DSN=S.I3546.UU.DAT,
//   UNIT=DISK,
//   DISP=(NEW,CATLG),
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160),
//   SPACE=(TRK,(10,10),RLSE)
```

Inputs to Model 1

```
C*****************************************************
C*                                                   *
C*     INPUTS TO MODEL 1 BEFORE CALLING              *
C*     SQP                                           *
C*                                                   *
C*****************************************************
C
      seventh
         INTPLN = 2,
      &END
      &SQP
         ISCALE = 2,
         MAXNPI = 200,
         MAXFUN = 200,
         ILOMAX = 200,
         IUPMAX = 200,
         IPRINT = 2,
         ICARD  = 2,
         IRSTRT = 2,
         MGRAD  = 1,
         FDPCT  = 0.D0,
         FDP(1) = 14*1.D-6,1.D-5,1.D-2,
         TOLHNP = 5.D-3,
         TOLGNP = 1.D-3,
         TOLFNP = 1.D-8,
         TOLSNP = 1.D-5,
         ISFMOD = 1,
         XSCALE(1)=16*1.D0,
         FSCALE = 1.D0,
         GSCALE = 1.D0,
         HSCALE = 1.D0,
         BOXL(1) = 15*-.02D0,0.D0,
         BOXU(1) = 15*.17D0,1000.D0,
         DELTAX(1)=15*.00001D0,1.D0,
         XR(1) = .105443114500D0,
         XR(2) = .211899586749D-1,
         XR(3) = .447023441329D-1,
         XR(4) = .382973212640D-1,
         XR(5) = .292484688648D-1,
         XR(6) = .128786412930D-1,
         XR(7) = .149126419715D-1,
         XR(8) = .308402949874D-1,
         XR(9) = .207402984632D-1,
         XR(10)= .213691408635D-1,
         XR(11)= .202589380879D-1,
         XR(12)= .182242251625D-1,
         XR(13)= .265208946206D-1,
         XR(14)= .636955316069D-1,
```

```
   XR(15)= .768118631641D-1,
   XR(16)= 480.D0,
&END
```

Sample output from Sequential Quadratic Programming

```
                SEQUENTIAL QUADRATIC PROGRAMMING OPTIMIZATION DRIVER
                        VERSION A.1 2/18/83

 ..................................................

UNSCALED X-VECTOR=      0.40000000E+03    0.65477110E+03    0.84563675E+03    0.10034264E+04    0.10153470E+04
        0.99704449E+03    0.98876609E+03    0.13343115E+04    0.13955517E+04    0.15013656E+04    0.15836165E+04
        0.16724459E+04    0.17093397E+04    0.17201235E+04    0.97508712E+03
SQP STORAGE REQUIREMENTS
------------------------

X, H, G - VECTOR STORAGE AVAILABLE 3000   100   100
X, H, G - VECTOR STORAGE ACQUIRED  1898     6    33
ENTERED VMCON -- SQP DRIVER
FDPCT= 0.000E+00  IPRINT=   2
NPROBV=  15  NEQLTY=   2  NI=  11
ILIMAX= 200  IUPMAX= 200  PARA, PARB=     0.20000E+03     0.20000E+00
MAXNPI= 200  MAXFUN= 200  MGRAD=   1
TOLGNP=     0.10000E-02  TOLHNP=     0.10000E-02  TOLFNP=     0.10300E-07  TOLSNP=     0.10000E-02
BIGVAL=     0.10300E+07  SMALL=     0.10000E-07
TLPZRO=     0.50000E-10  VZERO =     0.10000E-19  TLMXDE=     0.10300E+03
ISFMOD=   1  IZRMOD=   1
MX1DOP=   4  MODPRT=   1   ICHKOP=   0   MODEOP=   1   MAXBID=   5   GLAMI=  0.1000E+01
UNSCLD UPR BOX=  0.17500000E+04   0.17500000E+04   0.17500000E+04   0.17500003E+04   0.17500030E+04   0.17500000E+04
                 0.17500000E+04   0.17500000E+04   0.17500000E+04   0.17500003E+04   0.17500030E+04   0.17500000E+04
                 0.17500000E+04   0.17500000E+04   0.17500000E+04
UNSCLD LWR BOX=  0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000003E+00   0.00000030E+00   0.00000000E+00
                 0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000003E+00   0.00000030E+00   0.00000000E+00
                 0.00000000E+00   0.00000000E+00   0.00000000E+00
UNSCALED PERT.=  0.10000000E-04   0.10000000E-04   0.10000000E-04   0.10000003E-04   0.10000030E-04   0.10000000E-04
                 0.10000000E-04   0.10000000E-04   0.10000000E-04   0.10000003E-04   0.10000030E-04   0.10000000E-04
                 0.10300000E-04   0.10000000E-04   0.10000000E-04
SCALE FACTORS=   0.10000000E+01   0.10000000E+01   0.10000000E+01   0.10000000E+01   0.10000030E+01   0.10300000E+01
                 0.10300000E+01   0.10000000E+01   0.10000000E+01   0.10000000E+01   0.10000030E+01   0.10300000E+01
                 0.10000000E+01   0.10000000E+01   0.10000000E+01
MAX X-CHANGES=   0.13000000E-01   0.10000000E-01   0.10000000E-01   0.10000003E-01   0.10000030E-01   0.13000000E-01
                 0.13000000E-01   0.10000000E-01   0.10000000E-01   0.10000000E-01   0.10000030E-01   0.13000000E-01
                 0.13000000E-01   0.10000000E-01   0.10000000E-01
 *** CPU TIME FOR PARTIAL WITH RESPECT TO   1 TH CONTROL VARIABLE =       0.012 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO   2 TH CONTROL VARIABLE =       0.020 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO   3 TH CONTROL VARIABLE =       0.020 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO   4 TH CONTROL VARIABLE =       0.020 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO   5 TH CONTROL VARIABLE =       0.008 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO   6 TH CONTROL VARIABLE =       0.043 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO   7 TH CONTROL VARIABLE =       0.020 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO   8 TH CONTROL VARIABLE =       0.031 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO   9 TH CONTROL VARIABLE =       0.008 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO  10 TH CONTROL VARIABLE =       0.023 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO  11 TH CONTROL VARIABLE =       0.008 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO  12 TH CONTROL VARIABLE =       0.012 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO  13 TH CONTROL VARIABLE =       0.028 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO  14 TH CONTROL VARIABLE =       0.028 SECONDS ***
 *** CPU TIME FOR PARTIAL WITH RESPECT TO  15 TH CONTROL VARIABLE =       0.028 SECONDS ***
MATRIX OF CONSTRAINT NORMALS
-------------------------------------------------------------------------------------------------------------------
  0.16062665E+03    0.29815244E+03    0.26391237E+03   -0.28384881E+03   -0.13532744E+04   -0.32696178E+04   -0.52208973E
 -0.37929773E+04   -0.33191548E+04   -0.30830760E+04   -0.25492218E+04   -0.22592525E+04   -0.13588097E+04   -0.25656329E
  0.12500095E+03
-------------------------------------------------------------------------------------------------------------------
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  3.10000000E+01
-------------------------------------------------------------------------------------------------------------------
 -0.40000000E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
 -0.22708514E+03   -0.26486478E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
  0.61032086E+02   -0.63872540E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
  0.57635350E+02   -0.30273857E+03   -0.38172104E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
  0.54484761E+02    0.10133366E+03   -0.83510340E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
  0.51767242E+02    0.96089451E+02   -0.40042286E+03   -0.47154428E+03    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
  0.49575549E+02    0.92011987E+02    0.81444983E+02   -0.10108657E+04    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
  0.47773350E+02    0.88676110E+02    0.78492214E+02   -0.54175306E+03   -0.51659524E+03    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
  0.45623611E+02    0.84685745E+02    0.74960120E+02   -0.80623176E+02   -0.10953543E+04    0.00000000E+00    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
  0.43317430E+02    0.80405051E+02    0.71171037E+02   -0.75547837E+02   -0.73008143E+03   -0.51501849E+03    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
  0.41086393E+02    0.76263834E+02    0.67505412E+02   -0.72695261E+02   -0.26941579E+03   -0.10838927E+04    0.00000000E
  0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
  0.00000000E+00
-------------------------------------------------------------------------------------------------------------------
```

```
*** FUNCTION EVALUATION CPU TIME=            0.020 SECONDS ***
*** TOTAL GRADIENT EVALUATION CPU TIME=      0.309 SECONDS ***

**** DERIVATIVE SCALING CHECK AND MOD **

GRAD, CNORM, AND X WILL BE RESCALED **
CONTROL VARIABLE #  MIN DERIVATIVE      MAX DERIVATIVE      RECOMMENDED XSCALE CHANGE FACTOR
        1          0.00000000E+00      0.40000000E+03          0.10000000E+01
        2          0.00300000E+00      0.63872540E+03          0.10000000E+01
        3          0.00000000E+00      0.83518340E+03          0.10000000E+01
        4          0.00000000E+00      0.10108657E+03          0.10000000E+00
        5          0.00000000E+00      0.10953543E+03          0.10000000E+00
        6          0.00000000E+00      0.32696178E+03          0.10000000E+00
        7          0.00000000E+00      0.52208970E+03          0.10000000E+00
        8          0.00000000E+00      0.37929770E+03          0.10000000E+00
        9          0.00000000E+00      0.33191548E+03          0.10000000E+00
       10          0.00000000E+00      0.30030760E+03          0.10000000E+00
       11          0.00000000E+00      0.24492218E+03          0.10000000E+00
       12          0.00000000E+00      0.22592525E+03          0.10000000E+00
       13          0.00000000E+00      0.13388097E+03          0.10000000E+00
       14          0.00000000E+00      0.25656328E+03          0.10000000E+01
       15          0.00000000E+00      0.12580095E+03          0.10000000E+01


***  QP BASIS MATRIX DETERMINANT =      0.00000000E+00

******** ITERATION SUMMARY *********
ITER=   1   # FUNCTION EVALS=   1   TOTAL FUNC EVALS=  16   # GRADIENT EVALS=   1   F(X)= 0.27201159E+03
GLNORM= 0.000E+00   CON NORM= 0.102E+06   SEARCH NORM= 0.000E+00   DELTA(F)= 0.000E+00   ALPHA= 0.000E+00
X-VECTOR=          0.43000000E+03    0.65477110E+03    0.84560675E+03    0.10034264E+05    0.10153470E+05    0.99704649E+04
                   0.99870609E+04    0.13343115E+05    0.13955517E+05    0.15013656E+05    0.15836165E+05    0.16724458E+05
                   0.17093397E+05    0.17201235E+04    0.97508712E+03
GRADIENT=         -0.17229394E-01   -0.17565179E-01   -0.10051730E-01   -0.72411694E-03   -0.68411111E-03   -0.21190533E-02
                  -0.32424094E-02   -0.21463165E-02   -0.17814739E-02   -0.15638966E-02   -0.13957219E-02   -0.12651907E-02
                  -0.12040647E-02   -0.14390247E-01   -0.12454376E-01
EQUALITIES=       -0.10087301E+06    0.69871235E+01
INEQUALITIES=      0.00000000E+00    0.53024333E+04   -0.35583915E+04   -0.12244357E+04   -0.69534869E+04   -0.52760966E+04
                  -0.10225734E+05    0.54021725E+05    0.11497240E+06    0.18758292E+06    0.25827950E+06
*** QUADRATIC PROGRAM CPU TIME=            0.890 SECONDS ***

**********QUADRATIC PROGRAM CONSTRAINT PARTIAL CORRECTION FACTOR=     0.68120749Z6E-03
ONED1 INFORMATION:   ITERATION=   1
VMULT=             0.21276309E-03    0.63815802E-01    0.11351116E-03    0.10000000E+01    0.13735396E-03    0.10000000E+01
                   0.88111514E-04    0.10000000E+01    0.20345025E-02    0.10000000E+01    0.10000000E+01    0.10000000E+01
                   0.10000000E+01
*** SEARCH STEP = 0.0  F(X)= 0.27201159E+03  C NORM= 0.10183457E+06  LAG= 0.27201159E+03  PHI= 0.68163494E+04  DPSI=-0.57
*** SEARCH STEP = 1.0  F(X)= 0.27201256E+03  C NORM= 0.10174467E+06  LAG= 0.27201257E+03  PHI= 0.68043786E+04  DPHI=-0.11
EXACT PENALTY FUNCTION LINEARIZATION FACTOR: RHAT =   0.20761202E+01
MAX X-CHANGES=     0.10000000E+01    0.10000000E+01    0.10000000E+01    0.40000000E-01    0.10000000E+01    0.10000000E+01
                   0.13000000E+01    0.10000000E+01    0.10000000E+01    0.10000000E+01    0.10000000E+01    0.10000000E+01
                   0.10000000E+01    0.40000000E+01    0.10000000E+01


******** ITERATION SUMMARY *********
ITER=   2   # FUNCTION EVALS=   2   TOTAL FUNC EVALS=  32   # GRADIENT EVALS=   2   F(X)= 0.27201256E+03
GLNORM= 0.139E+00   CON NORM= 0.102E+06   SEARCH NORM= 0.138E+00   DELTA(F)= 0.359E-05   ALPHA= 0.100E+01
X-VECTOR=          0.43000000E+03    0.65476619E+03    0.84559882E+03    0.10034164E+05    0.10153459E+05    0.99704323E+04
                   0.99873086E+04    0.13343077E+05    0.13955484E+05    0.15013626E+05    0.15836139E+05    0.16724435E+05
                   0.17093384E+05    0.17201135E+04    0.97508097E+03
GRADIENT=         -0.17229456E-01   -0.17565412E-01   -0.10051912E-01   -0.72412943E-03   -0.68411396E-03   -0.21190687E-02
                  -0.32424390E-02   -0.21463302E-02   -0.17814824E-02   -0.15639029E-02   -0.13957265E-02   -0.12651935E-02
                  -0.12040668E-02   -0.14390400E-01   -0.12454535E-01
EQUALITIES=       -0.10078415E+06    0.69809566E+01
INEQUALITIES=      0.00000000E+00    0.53037336E+04   -0.35552558E+04   -0.12199220E+04   -0.69473594E+04   -0.52686711E+04
                  -0.10216723E+05    0.54026628E+05    0.11497335E+06    0.18758518E+06    0.25828314E+06
**********QUADRATIC PROGRAM CONSTRAINT PARTIAL CORRECTION FACTOR=     0.11120408Z0E-02
ONED1 INFORMATION:   ITERATION=   2
VMULT=             0.12298583E-03    0.40969565E-01    0.73985803E-04    0.50000000E+00    0.94469816E-04    0.50000000E+00
                   0.82863555E-03    0.50000000E+00    0.11378669E-02    0.50000000E+00    0.50000000E+00    0.50000000E+00
                   0.50000000E+00
*** SEARCH STEP = 0.0  F(X)= 0.27201256E+03  C NORM= 0.10174467E+06  LAG= 0.27201256E+03  PHI= 0.35467081E+04  DPSI=-0.36
*** SEARCH STEP = 1.0  F(X)= 0.27201369E+03  C NORM= 0.10163134E+06  LAG= 0.27201369E+03  PHI= 0.35391526E+04  DPHI=-0.75
EXACT PENALTY FUNCTION LINEARIZATION FACTOR: RHAT =   0.20754308E+01
MAX X-CHANGES=     0.13000000E+01    0.10000000E+01    0.40000000E+01    0.40000000E-01    0.10000000E+01    0.10000000E+01
                   0.10000000E+01    0.10000000E+01    0.10000000E+01    0.10000000E+01    0.10000000E+01    0.10000000E+01
                   0.10000000E+01    0.40000000E+01    0.10000000E+01


******** ITERATION SUMMARY *********
ITER=   3   # FUNCTION EVALS=   1   TOTAL FUNC EVALS=  48   # GRADIENT EVALS=   3   F(X)= 0.27201369E+03
GLNORM= 0.731E+03   CON NORM= 0.102E+06   SEARCH NORM= 0.175E+00   DELTA(F)= 0.413E-05   ALPHA= 0.100E+01
X-VECTOR=          0.43000000E+03    0.65476000E+03    0.84558882E+03    0.10034038E+05    0.10153446E+05    0.99703904E+04
                   0.99869415E+04    0.13343028E+05    0.13955440E+05    0.15013587E+05    0.15836104E+05    0.16724406E+05
                   0.17093368E+05    0.17201092E+04    0.97507320E+03
GRADIENT=         -0.17229526E-01   -0.17565708E-01   -0.10052143E-01   -0.72414537E-03   -0.68411850E-03   -0.21190891E-02
                  -0.32424771E-02   -0.21463678E-02   -0.17814938E-02   -0.15639108E-02   -0.13957327E-02   -0.12651975E-02
                  -0.12040721E-02   -0.14390486E-01   -0.12454677E-01
EQUALITIES=       -0.10067212E+06    0.69732032E+01
INEQUALITIES=      0.00000000E+00    0.53053731E+04   -0.35513022E+04   -0.12142310E+04   -0.69396337E+04   -0.52593164E+04
                  -0.10205362E+05    0.54032820E+05    0.11497457E+06    0.18758807E+06    0.25828781E+06
**********QUADRATIC PROGRAM CONSTRAINT PARTIAL CORRECTION FACTOR=     0.14340611Z9E-02
ONED1 INFORMATION:   ITERATION=   3
VMULT=             0.69883999E-04    0.33914715E+00    0.69965576E-04    0.25000000E+00    0.70377773E-04    0.25000000E+00
                   0.42559402E-03    0.25000000E+00    0.59661329E-03    0.25000000E+00    0.25000000E+00    0.25000000E+00
                   0.25000000E+00
*** SEARCH STEP = 0.0  F(X)= 0.27201369E+03  C NORM= 0.10163134E+06  LAG= 0.27201369E+03  PHI= 0.19089977E+04  DPSI=-0.23
*** SEARCH STEP = 1.0  F(X)= 0.27201502E+03  C NORM= 0.10148537E+06  LAG= 0.27201502E+03  PHI= 0.19041171E+04  DPHI=-0.48
EXACT PENALTY FUNCTION LINEARIZATION FACTOR: RHAT =   0.20759603E+01
MAX X-CHANGES=     0.13000000E+01    0.10000000E+01    0.40000000E+01    0.40000000E-01    0.10000000E+01    0.10000000E+01
                   0.10000000E+01    0.10000000E+01    0.10000000E+01    0.10000000E+01    0.10000000E+01    0.10000000E+01
                   0.10000000E+01    0.40000000E+01    0.40000000E+01
```

.
.
.
.

```
******* ITERATION SUMMARY ********
ITER= 33   # FUNCTION EVALS= 35   TOTAL FUNC EVALS= 928   # GRADIENT EVALS= 33   F(X)= 0.27287709E+03
GLNORM= 0.962E-03   CON NORM= 0.175E-01   SEARCH NORM= 0.356E+00   DELTA(F)= 3.231E-05   ALPHA= 0.100E+01
X-VECTOR=          3.48000000E+03   0.64917507E+03   0.83655196E+03   0.99197138E+04   0.10108508E+05   0.98932697E+04
                   0.99915420E+04   0.13174293E+05   0.13860210E+05   0.14934840E+05   0.15788286E+05   0.16695350E+05
                   0.17167139E+05   0.17500000E+04   0.96810000E+03
GRADIENT=         -0.17296691E-01  -0.17835458E-01  -0.10266081E-01  -0.73955675E-03  -0.69200212E-03  -0.21419390E-02
                  -0.32625070E-02  -0.21877225E-02  -0.18087576E-02  -0.15801737E-02  -0.14046844E-02  -0.12691106E-02
                  -0.11967154E-02  -0.14084912E-01  -0.12416206E-01
EQUALITIES=       -3.17333371E-01   0.00000000E+00
INEQUALITIES=      0.00000000E+00   0.67805634E+04   0.14551915E-10   0.38990478E+04   0.29103830E-10   0.31521947E+04
                  -0.34966389E-07   0.61041562E+05   0.11919702E+06   0.19411347E+06   0.26706540E+06
ONEDI INFORMATION: ITERATION= 33
VMULT=             0.13952340E-04   0.14222989E-01   0.54068612E-04   0.23283064E-09   3.34314021E-04   0.23283064E-09
                   0.16446570E-04   0.23283064E-09   0.49100329E-05   0.23283064E-09   0.23283064E-09   0.23283064E-09
                   0.23283064E-09
*** SEARCH STEP = 0.0  F(X)= 0.27287709E+03  C NORM= 0.17333371E-01  LAG= 0.27287709E+03  PHI= 0.27287709E+03  DPSI=-0.34
*** SEARCH STEP = 1.0  F(X)= 0.27287709E+03  C NORM= 0.25013625E-03  LAG= 0.27287709E+03  PHI= 0.27287709E+03  DPHI=-0.34
EXACT PENALTY FUNCTION LINEARIZATION FACTOR: RHAT =  0.99882028E+00
SOLUTION FOUND.  CRITERION 1 IS SATISFIED
MINIMIZATION SOLUTION SUMMARY
ITER= 33   # FUNCTION EVALS= 34   TOTAL FUNC EVALS= 544   # GRADIENT EVALS= 34   F(X)= 0.27287709E+03
GLNORM= 0.725E-03   CON NORM= 0.250E-03   SEARCH NORM= 0.699E-01   DELTA(F)= 0.114E-07   ALPHA= 0.100E+01
X-VECTOR=          3.48000000E+03   0.64917507E+03   0.83655196E+03   0.99197138E+04   0.10108522E+05   0.98932567E+04
                   3.98914991E+04   0.13174341E+05   0.13863220E+05   0.14934846E+05   0.15788259E+05   0.16695349E+05
                   3.17137123E+05   0.17500000E+04   0.96813000E+03
GRADIENT=         -0.17296696E-01  -0.17835458E-01  -0.10266081E-01  -0.73955675E-03  -0.69200098E-03  -0.21419424E-02
                  -0.32625218E-02  -0.21877645E-02  -0.18087536E-02  -0.15801726E-02  -0.14046839E-02  -0.12691112E-02
                  -0.11967171E-02  -0.14084916E-01  -0.12416211E-01
EQUALITIES=       -0.25013625E-03   0.00000000E+00
INEQUALITIES=      3.00000000E+00   0.67805634E+04   0.14551915E-10   0.38990478E+04   0.29103930E-10   0.31521947E+04
                   3.00000000E+00   0.61040858E+05   0.11919556E+06   0.19411218E+06   0.26706437E+06
LAG. W. - EQ=      0.13232573E-04  -0.14222989E-01
LAG. W. - INE=     3.53732754E-04   0.00000000E+00   3.34255527E-04   0.00000000E+00   0.15039721E-04   0.00000000E+00
                   3.34194138E-05   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
LAG. W. -BOXU=     0.00000000E+00   3.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
                   3.00000000E+00   3.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
                   3.00000000E+00   0.12054597E-01   0.00000000E+00
LAG. W. -BOXL=     0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
                   3.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
                   0.00000000E+00   0.00000000E+00   0.00000000E+00
LAG. GRAD=         3.14377308E-04  -0.25911623E-04  -0.41591127E-04  -0.33877465E-03  -0.13445919E-03  -0.26194694E-03
                   3.19058931E-03   0.26945081E-04   0.10680967E-03   0.22729633E-03   0.30119703E-03   0.32335418E-03
                  -0.15686603E-03  -0.68222732E-04   0.84005953E-04
FIRST-ORDER PARTIALS OF PERFORMANCE INDEX WRT CONSTRAINTS
( F(X) )/D( W(X) ) =               0.13232573E-04  -0.14222989E-01
( F(X) )/D( G(X) ) =               3.53732754E-04   0.12000000E+00   3.34255527E-04   0.00000000E+00   0.15039721E-04
                   0.00000000E+00  3.34194128E-05   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
D( F(X) )/D( BOXU ) =              0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
                   0.00000000E+00  0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
                   0.00000000E+00  0.00000000E+00   0.12054597E-01   0.00000000E+00
D( F(X) )/D( BOXL ) =              0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
                   0.00000000E+00  0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00   0.00000000E+00
                   0.00000000E+00  0.00000000E+00   0.00000000E+00
MATRIX OF CONSTRAINT NORMALS
-----------------------------------------------------------------------------------------------------------------------
 0.83266804E+02    0.15433922E+03    0.14364419E+03   -0.45431475E+01   -0.42136821E+02   -0.14207290E+03   -0.25868477E
-3.16734446E+03   -0.14476498E+03   -0.13659402E+03   -0.12891607E+03   -0.12034403E+03   -0.78581381E+02   -0.14827496E
 0.13019214E+03
-----------------------------------------------------------------------------------------------------------------------
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.10000000E+01
-----------------------------------------------------------------------------------------------------------------------
-3.48000000E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
-3.22550945E+03   -3.26341411E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
 0.61349456E+02   -3.63303200E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 3.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
 0.57655443E+02   -3.29922210E+03   -3.37792114E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00    3.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
 0.54499946E+02    0.10101796E+03   -0.82563004E+03    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
 0.51749911E+02    3.95920468E+02   -0.39356938E+03   -0.46623027E+02    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
 0.49500242E+02    0.91755798E+02    0.85392949E+02   -0.99628333E+02    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
 0.47723179E+02    0.88456935E+02    0.82327332E+02   -0.50583468E+02   -0.51214173E+02    0.00000000E+00    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
 0.45740319E+02    0.84781622E+02    0.78906698E+02   -0.24956315E+01   -0.13640797E+03    0.00000000E+00    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
 0.43470283E+02    0.80574614E+02    0.74990654E+02   -0.23717762E+01   -0.47811590E+02   -0.51145015E+02    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
 0.41292431E+02    0.76537216E+02    0.71235587E+02   -0.22529995E+01   -0.23895764E+02   -0.10601484E+03    0.00000000E
 0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E+00    0.00000000E
 0.00000000E+00
-----------------------------------------------------------------------------------------------------------------------
```