Electronic Transactions on Numerical Analysis.
Volume 49, pp. 103–125, 2018.
Copyright © 2018, Kent State University.
ISSN 1068–9613.
DOI: 10.1553/etna_vol49s103

ETNA
Kent State University and
Johann Radon Institute (RICAM)

A FULL-SPACE QUASI-LAGRANGE-NEWTON-KRYLOV ALGORITHM FOR TRAJECTORY OPTIMIZATION PROBLEMS[*]

HSUAN-HAO WANG[†], YI-SU LO[†], FENG-TAI HWANG[‡], AND FENG-NAN HWANG[†]

Abstract. The objectives of this work are to study and to apply the full-space quasi-Lagrange-Newton-Krylov (FQLNK) algorithm for solving trajectory optimization problems arising from aerospace industrial applications. As its name suggests, in this algorithm we first convert the constrained optimization problem into an unconstrained one by introducing the augmented Lagrangian parameters. The next step is to find the optimal candidate solution by solving the Karush-Kuhn-Tucker (KKT) system with a Newton-Krylov method. To reduce the computational cost of constructing the KKT system, we employ the Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula to build an approximation of the (1,1) subblock of the KKT matrix, which is the most expensive part of the overall computation. The BFGS-based FQLNK algorithm exhibits a superior speedup compared to some of the alternatives. We demonstrate our FQLNK algorithm to be a practical approach for designing an optimal trajectory of a launch vehicle in space missions.

Key words. launch vehicle mission, trajectory optimization, KKT system, BFGS, Lagrange-Newton-Krylov solver

AMS subject classifications. 65H10, 49M15

1. Introduction.

Optimal control is a commonly used technique with a broad range of applications in aerospace engineering such as spacecraft/launcher optimal design problems [4, 8, 29]. Trajectory optimization plays an important role in space missions [2, 3, 5, 10, 21, 24]. For instance, during the mission design stage, one of the main tasks is to find an optimal trajectory of the rocket to maximize the mass of a payload or to minimize the duration of the flight from launch to satellite insertion, subject to the physical constraints and the insertion conditions. Another example is an interplanetary probe traveling between two orbits. During the mission operation stage, aerospace engineers need to design an optimal trajectory to reduce the consumption of fuel to extend its mission life further. Both these practical examples of optimal trajectory missions can be modeled mathematically as some form of time-continuous optimal control problem.

Generally speaking, the solution algorithms for optimal control problems available in the literature can be divided into two categories: indirect methods and direct methods; see [2, 24, 31] and references therein for a comprehensive survey of these two classes of methods. Indirect methods are also referred to as *optimize-then-discretize* approaches and are mainly based on the calculus of variation. After recasting the optimal control problem, the resulting two-point boundary value problem is solved by some numerical ordinary differential equation (ODE) solver. In contrast to the indirect methods [8, 25], our proposed approach belongs to the class of direct methods [11, 13, 32] known as the *discretize-then-optimize* approach [4]. The standard procedure of a direct method is to reformulate a time-continuous optimal control problem as an algebraically constrained parameter optimization problem by using some numerical integrators [4, 11, 14, 32] such as the shooting, the multiple shooting, the collocation, and the pseudo-spectral method to transcribe the dynamical system into the algebraic constraints. Nonlinear programming techniques developed for constrained parameter

[*]Received July 6, 2017. Accepted March 29, 2018. Published online on May 30, 2018. Recommended by Xiao-Chuan Cai. This work was partially supported by the Ministry of Science and Technology of Taiwan, under Grant No. MOST-106-2115-M-008-010-MY2.

[†]National Central University, Jhongli District 32001, Taoyuan City, Taiwan (alexwang801018@gmail.com, yisu.luo@gmail.com, hwangf@math.ncu.edu.tw).

[‡]Division of Satellite Image, National Space Organization, Hsinchu, 30078, Taiwan (fthwang@narlabs.org.tw).

optimization problems can be implemented. These techniques can be classified as gradient-based methods [20] such as Newton-type methods and nonlinear conjugate gradient methods, etc., or gradient-free methods including genetic algorithms [27, 28, 33], particle swarm algorithms [22], or simulated annealing [18].

The main objective of this research work is to study the full-space quasi-Lagrange-Newton-Krylov (FQLNK) algorithm [7, 23] for the numerical solution of the optimal trajectory problem, which is formulated as an optimization problem constrained by a system of ODEs. To be more specific, we apply the FQLNK algorithm to the multistage satellite launch vehicle problem. As its name suggests, in this algorithm we first convert the constrained optimization problem into an unconstrained one by introducing the Lagrangian function and then find the candidate optimal solution from the first-order necessary condition, the Karush-Kuhn-Tucker (KKT) condition [20], and solve it with an inexact Newton method in conjunction with a backtracking technique. In each Newton iteration the resulting full KKT system for all variables is solved in one shot by a Krylov-subspace method combined with a preconditioner. A popular alternative approach is the so-called reduced-space method, where different field parameters are obtained sequentially. Both full-space method and reduced-space methods belong to the family of the well-known sequential quadratic programming (SQP) [20, 23]. An advantage of the reduced-space method is a reduced amount of memory usage. However, due to the dramatic increase in computer power, full-space methods recently gained popularity and have been successful especially for (partial) differential equation constrained optimization problems arising from different applications such as flow control problems [7, 23, 34]. Biro and Ghattas [6] reported several nontrivial boundary control problems of complex incompressible flows by using full-space type methods. Since for reduced-space methods many subiterations are needed for the outer iterations to converge, their numerical results showed that a full-space method was about ten times faster than a popular reduced-space method. Also, they asserted that other problems exhibit similar performance behaviors.

In practice, several computational issues need to be appropriately addressed to make the Lagrange-Newton-Krylov method more efficient and robust for large, sparse, constrained optimization problems. For example, the KKT system consists of a sub-block matrix corresponding to the second derivatives of the Lagrangian function, which is called the Hessian matrix. Deriving the Hessian matrix analytically is quite tedious, and its numerical approximation might be costly in terms of computational effort. Also, the KKT system is indefinite and often ill-conditioned so that the convergence rate of a Krylov-subspace method degrades. Hence, designing an efficient preconditioner is crucial in order to find a high-quality Newton direction. Furthermore, due to the highly local nonlinearity of the problem, the convergence of the Newton method becomes problematic. In this article, we focus on how to construct the Hessian matrix more efficiently. To reduce the computational cost of the KKT matrix construction, we propose, following the suggestion in [20], to employ the Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula [9, 17, 20] to construct the approximation of the Hessian of the KKT matrix. The BFGS method was initially developed for unconstrained optimization problems and is easily adapted for constrained cases. We carry out a comparative study of the proposed approach with some of the alternative methods including finite differences and automatic differentiation. We also discuss other computational issues regarding the efficiency of KKT system solvers and the robustness of the inexact Newton solver.

The remainder of this article is structured as follows. In the next section we build the mathematical optimal control model for a multistage launch vehicle system as a parameter optimization problem. In Section 3, we describe the FQLNK algorithm for solving the parameter optimization problems in detail. In Section 4 we present some numerical results and discussions. In addition to the multistage launch problem we also consider one additional

ETNA
Kent State University and
Johann Radon Institute (RICAM)

benchmark problem to verify the correctness of our code and to evaluate the performance of our proposed full-space algorithm. In Section 5 we summarize the main contributions of this article.

2. Multistage satellite launch vehicle problem.

2.1. Problem description. Consider our target application, the multistage satellite launch vehicle problem, as follows. The primary goal of the mission is to provide both sufficient speed and altitude for the satellite with an appropriate insertion angle so that it can be successfully delivered into a designated orbit. In practice, a shorter launch flight distance (or flight duration) can ensure that the telecommunication, the telemetry, tracking, and control (TT&C) system, is functional between the ground station and the launch vehicle. Generally speaking, for low-Earth orbit satellites, two common strategies based on different launch rocket procedures were proposed for inserting a satellite by riding a multistage rocket into an orbit. The first one is the so-called direct insertion approach (see the left of Figure 2.1): each stage of the rocket burns fuel continuously and accelerates until the speed of its last stage equals the insertion speed of the satellite at the burnout point. As shown on the right of Figure 2.1, the second strategy is similar to the first one but it allows some coasting-flight period during the launch procedure so that less fuel is consumed to insert the satellite into the orbit with higher altitude.

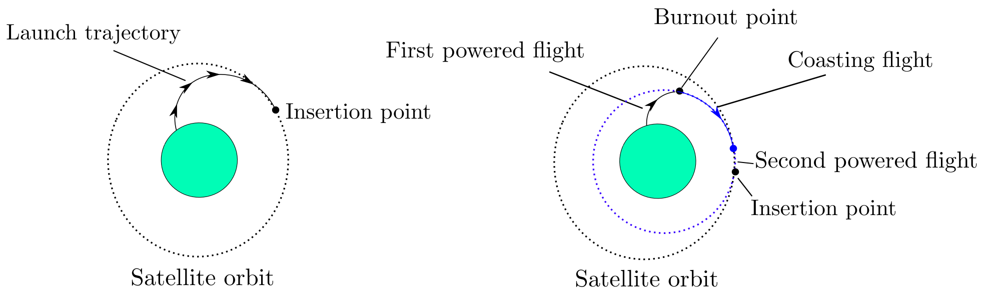

FIG. 2.1. *An illustration of two strategies for inserting a satellite into an orbit. Direct insertion (left) and an insertion with coasting-flight period (right).*

In the following sections we first build a mathematical model for the minimum-time trajectory design problem of a multistage launch vehicle with some coasting-flight period based on the second strategy as a free final-time optimal control problem. In this problem, we try to find an optimal trajectory that minimizes the flight duration from launch to an insertion point subject to the insertion condition and path constraints. After a change of variables by introducing some pseudo-time variable and then discretizing the differential constraints using the composite trapezoidal rule, we finally derive a large, sparse, algebraically constrained parameter optimization problem from the continuous free final-time control problem.

2.2. Mathematical model for the launch vehicle system. For simplicity, we consider the launch vehicle as a point mass moving on a two-dimensional plane and the Earth as perfectly spherical. As shown in Figure 2.2, the launch point inertial frame is set to be the reference coordinate system [30]. For aerodynamics, we also take the air resistance effect into account, and the data for the mass distribution and the thrust of each stage of the launch vehicle is assumed available in advance.

Let the multistage rocket launch process, starting from time t_0 and ending at time t_f, consist of $(N+1)$ events $t_0 < t_1 < t_2, \ldots, < t_N = t_f$. The launch vehicle under con-

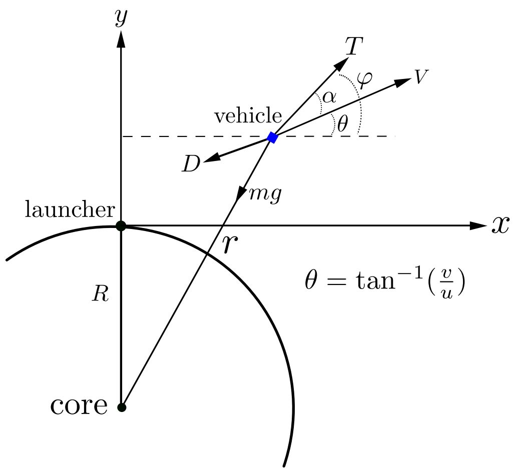

FIG. 2.2. *The geometric configuration for the multistage satellite launch vehicle problem.*

sideration involves more than one stage and possibly a complex mission sequence. In that case, some of the state variables or other variables may be discontinuous at particular time points which are referred to as events. The semi-closed time interval $[t_{i-1}, t_i)$ is called the ith phase, where $i = 1, \ldots, N$. The period for each phase is defined as $\triangle t_i = t_i - t_{i-1}$. Without loss of generality, we assume only one coasting period of duration $\triangle t_c$ during the kth phase $[t_k, t_{k+1})$. The generalization of the proposed method in the case of multiple coasting periods is straightforward. The launch vehicle trajectory design problem is formulated as a free final-time optimal control problem (OCP1) as follows. Find the piecewise continuous control history $\varphi(t)$ on the time interval $[t_0, t_f]$ that minimizes the *objective function*

$$J = t_f \tag{2.1}$$

subject to the *differential constraints*

$$\frac{d\mathbf{s}}{dt} = f^{(i)}(\mathbf{s}, \varphi, t), \qquad i = 1, \ldots, N, \tag{2.2}$$

and the *initial and final conditions* at time t_0 and t_f

$$\begin{aligned} \psi_0(\mathbf{s}(t_0), t_0) = 0 \qquad &\text{and} \\ \psi_f(\mathbf{s}(t_f), t_f) = 0, \end{aligned} \tag{2.3}$$

respectively. Here, the vector $\mathbf{s} = (u, v, x, y)^T$ is the set of the state variables, where (u, v) are the x- and y-components of the velocity of the rocket at position (x, y). The final time t_f is to be determined, and it is referred to as a design variable. Moreover, the differential constraint is explicitly defined as

$$f^{(i)}(\mathbf{s}, \varphi, t) = \begin{bmatrix} \frac{T^{(i)}}{M^{(i)}} \cos\varphi - \frac{D}{M^{(i)}} \cos\theta - g\frac{x}{\|r\|} \\ \frac{T^{(i)}}{M^{(i)}} \sin\varphi - \frac{D}{M^{(i)}} \sin\theta - g\frac{y+R}{\|r\|} \\ u \\ v \end{bmatrix}, \qquad i = 1, \ldots, N,$$

where $T^{(i)}$ and $M^{(i)}$ are the thrust and the total mass, including the structure mass and the fuel mass at the ith phase, respectively, and $\|r\|$ with $r = (x, y + R)$ is the distance between the rocket and the Earth's core. Here R is the Earth's radius. In addition, the control variable φ is the pitch angle relative to the positive y-axis and the flight path angle θ is defined as

$$\theta = \tan^{-1} \frac{v}{u}.$$

Note that during the coasting-flight period, the thrust is zero, i.e., $T^{(k)} = 0$, and the control variable φ is determined solely by the motion of the vehicle. The air resistance D is given via

$$D = \frac{1}{2}\rho V^2 C_D S_{ref},$$

where the two constants C_D and S_{ref} are the drag coefficient and the area of the cross-section of the vehicle, respectively. Furthermore, $V = \sqrt{u^2+v^2}$ is the total velocity, $\rho = \rho_0 \exp((R-r)/H)$ the density of the air, ρ_0 the density of air at the ground, H the thickness of the Earth's atmosphere, and the gravity g is defined as

$$g = g_0 \left(\frac{R}{r}\right)^2$$

with the gravity at the ground g_0. The initial condition is prescribed as

$$\psi_0(\mathbf{s}(t_0), t_0) = \begin{bmatrix} \|r(t_0)\| - r_0 \\ V(t_0) - V_0 \\ \theta(t_0) - \theta_0 \end{bmatrix} = 0, \tag{2.4}$$

and the final condition is prescribed as

$$\psi_f(\mathbf{s}(t_f), t_f) = \begin{bmatrix} \|r(t_f)\| - R - H(t_f) \\ V(t_f) - \sqrt{\mu/\|r(t_f)\|} \\ (x(t_f), y(t_f) + R) \cdot (u(t_f), v(t_f))/(\|r(t_f)\| V(t_f)) \end{bmatrix} = 0. \tag{2.5}$$

Above, (2.5) is an insertion condition to assure the launch vehicle reaches enough height $H(t_f)$ and sufficient speed $\sqrt{\mu/\|r(t_f)\|}$ with an appropriate insertion angle. Here, μ is the gravitational parameter of the Earth. In addition, all state parameters are assumed to be continuous at each t_i. Hence, a link condition between each stage is imposed, i.e.,

$$s(t_i^-) = s(t_i^+), \quad i = 1, 2, \ldots, (N-1),$$

where $s(t_i^-) \equiv \lim_{t \to t_i^-} s(t)$ and $s(t_i^+) \equiv \lim_{t \to t_i^+} s(t)$ are the left-hand and right-hand limits of s at $t = t_i$, respectively.

We remark, firstly, that the objective function considered here is known as in the Mayer form. Another possibility is the Lagrange form, which involves some integral terms such as the objective of minimizing the quantity of fuel consumed during the entire launch process, or the Bolza form, which is a mixed form combining both the Mayer and the Lagrange forms. Secondly, in many real-world applications trajectory optimization problems involve some inequality constraints. These inequality constraints can be in the form of control, state, and mixed state and control constraints [29]. For example, in a space mission, when the rocket lifts off, a large bending moment is generated due to a high density of the atmosphere. As a result, the large angle of attack of the vehicle not only causes the rocket to get out of control but also damages the rocket structure. To prevent this disastrous situation from happening, we may confine the angle of attack within a smaller range, say $5°$ in the first few seconds after takeoff. Here, the angle of attack α is defined in terms of the pitch angle φ and the flight path angle

θ as $\alpha \equiv \varphi - \theta$. To simplify the presentation, we first confine our discussion to the equality constraint case, which is transcribed from the optimal control problem arising in aerospace trajectory optimizations. In Section 4.7 we will discuss an extension of our proposed algorithm for problems with inequality constraints.

We next transform the free final-time optimal control problem (2.1)–(2.3) into a fixed final-time one by introducing a new pseudo-time variable τ,

$$\tau = \begin{cases} t & \text{in } [t_0, t_{k-1}), \\ t_{k-1} + (t - t_{k-1})/(\triangle t_c) & \text{in } [t_{k-1}, t_k), \\ (t - t_k) + t_{k-1} + 1 & \text{in } [t_k, t_f), \end{cases}$$

and perform a change of variable with respect to τ. The transformed temporal slots are

$$[\tau_0, \tau_1) \cup [\tau_1, \tau_2) \cup, \ldots, \cup[\tau_{k-1}, \tau_k) \cup, \ldots, \cup[\tau_{N-2}, \tau_{N-1}) \cup [\tau_{N-1}, \tau_f],$$

where $\Delta\tau_k \equiv (\tau_k - \tau_{k-1}) = 1$ and now τ_f is known. The differential constraint corresponding to the coasting-flight phase is rewritten as

$$\frac{d\hat{\mathbf{s}}(\tau)}{d\tau} = g^{(k)}(\hat{\mathbf{s}}, \hat{\varphi}, \tau)$$

with

$$g^{(k)}(\hat{\mathbf{s}}, \hat{\varphi}, \tau) = (\Delta t_c) \begin{bmatrix} \frac{T^{(k)}}{M^{(k)}} \cos\hat{\varphi}(\tau) - \frac{D}{M^{(k)}} \cos\hat{\theta}(\tau) - g\frac{\hat{x}(\tau)}{\|\hat{r}(\tau)\|} \\ \frac{T^{(k)}}{M^{(k)}} \sin\hat{\varphi}(\tau) - \frac{D}{M^{(k)}} \sin\hat{\theta}(\tau) - g\frac{\hat{y}(\tau))+R}{\|\hat{r}(\tau)\|} \\ \hat{u}(\tau) \\ \hat{v}(\tau) \end{bmatrix},$$

where $\hat{\varphi} = \varphi(h(\tau))$, $\hat{\mathbf{s}} = \mathbf{s}(h(\tau))$, $\hat{\theta} = \theta(h(\tau))$, and $h : [t_{k-1}, t_{k-1} + 1] \to \mathbb{R}$ is defined as $h(\tau) = \Delta t_c(\tau - t_{k-1}) + t_{k-1}$. Note that $g^{(i)} = f^{(i)}$ for $i = 1, \ldots, N$ and $i \neq k$.

Therefore, the fixed final-time optimal control problem can be stated as follows. Find the control history $\hat{\varphi}(\tau)$ and the design parameter Δt_c that minimizes the *objective function*

$$J = \Delta t_c \tag{2.6}$$

subject to the *differential constraints*

$$\frac{d\hat{\mathbf{s}}}{d\tau} = g^{(i)}(\hat{\mathbf{s}}, \hat{\varphi}, \tau), \qquad i = 1, \ldots, N, \tag{2.7}$$

and the *initial and final conditions* at time τ_0 and τ_f

$$\begin{aligned} \psi_0(\hat{\mathbf{s}}(\tau_0), \tau_0) &= 0, \\ \psi_f(\hat{\mathbf{s}}(\tau_f), \tau_f) &= 0. \end{aligned} \tag{2.8}$$

Here, ψ_0 and ψ_f are defined as in (2.4) and (2.5). To simplify the notation, we drop the hat symbol for all variables and replace the variable τ by t throughout the article.

2.3. A parameter-constrained optimization problem.

To convert the fixed final-time optimal control problem (2.6)–(2.8) into a finite-dimensional parameter optimization problem, we first discretize the dynamical system for the motion of the launch vehicle by partitioning each time interval (t_{i-1}, t_i) of phase i into M_i finite subintervals. For the sake of simplicity, we assume the time lengths for the subintervals in each phase to be equal, i.e.,

$$h^{(i)} = \frac{(t_i - t_{i-1})}{M_i}, \qquad i = 1, \dots, N.$$

Let the jth node of phase i be denoted by $t_j^{(i)}$. For $j = 0, \dots, M_i$ and $i = 1, \dots, N$, $\varphi_j^{(i)}$ and $s_j^{(i)} = (u_j^{(i)}, v_j^{(i)}, x_j^{(i)}, y_j^{(i)})^T$ represent the approximate solutions of the control and state variables at the time $t_j^{(i)}$, respectively. We take the integral on both sides of the differential constraints and then approximate the integral on the right-hand side by the trapezoidal rule,

$$\begin{aligned}&\frac{ds}{dt} = g^{(i)}(s, \varphi, t) \quad \Rightarrow \int_{t_{j-1}^{(i)}}^{t_j^{(i)}} ds = \int_{t_{j-1}^{(i)}}^{t_j^{(i)}} g^{(i)}(s, \varphi, t)dt\\ &\Rightarrow s_j^{(i)} - s_{j-1}^{(i)} \approx \frac{h^{(i)}}{2}(g_{j-1}^{(i)} + g_j^{(i)}).\end{aligned}$$

Next, we define the residual constraints at each $t_j^{(i)}$,

$$R_j^{(i)} \equiv s_j^{(i)} - s_{j-1}^{(i)} - \frac{h^{(i)}}{2}(g_{j-1}^{(i)} + g_j^{(i)}),$$

where $j = 1, \dots, M_i$ and $i = 1, \dots, N$. In addition, R_0 and R_f are the initial and final residual constraints, respectively. Higher-order integrators such as Simpson's rule or the 4th order implicit Runge-Kutta method could be employed as well [4]. For clarity, let $p_x \in \mathbb{R}^n$ be a vector containing all the discrete state variables $s_j^{(i)}$, the discrete control parameters $\varphi_j^{(i)}$, as well as the design parameter Δt_c. The design parameter is numbered first, followed by the state parameters and the control parameters in order at each time grid point. The objective function for this problem is defined as $\mathcal{J}(p_x) = \Delta t_c$. In addition, the constraint vector is defined as

$$c(p_x) = (R_0, R_1^{(1)}, R_2^{(1)}, \dots, R_{M_1}^{(1)}, R_1^{(2)}, \dots, R_{M_2}^{(2)}, \dots, R_{M_N}^{(N)}, R_f)^T \in \mathbb{R}^m.$$

As a result, the equality-constrained parameter optimization problem for the fixed final-time optimal control problem (2.6)–(2.8) reads as follows. Find the parameter vector p_x such that

$$\left\{\begin{array}{ll} \min\limits_{p_x} & \mathcal{J}(p_x) \equiv \Delta t_c \\ \text{subject to} & c(p_x) = 0. \end{array}\right. \tag{2.9}$$

Note that in the case that either $\mathcal{J}$ or the equality constraint condition c is nonlinear, the problem is referred to as a nonlinear programming problem; NLP. Such problems have a variety of applications in physics, chemistry, and engineering [4, 8, 14]. In the next section we describe the FQLNK algorithm for solving the equality-constrained parameter optimization problem (2.9).

3. Full-space quasi-Lagrange-Newton-Krylov algorithm.

3.1. A description of the algorithm. To solve the equality-constrained parameter optimization problem (2.9), we begin by defining the Lagrangian functional as

$$\mathcal{L}(p) \equiv \mathcal{J} - p_\lambda^T c$$

where $p = (p_x, p_\lambda)^T \in \mathbb{R}^{n+m}$ is the full-space unknown vector. Here, p_λ is a sub-vector corresponding to the Lagrangian multipliers. Then the quasi-Newton method with backtracking technique (QNB) is applied to solve the KKT condition given by

$$F(p) \equiv \nabla \mathcal{L}(p) = 0,$$

and the corresponding KKT matrix takes the form

$$K(p) = \begin{bmatrix} H(p) & G(p)^T \\ G(p) & \mathbf{0} \end{bmatrix},$$

where $H \equiv \nabla^2_{xx}\mathcal{L}$ is the Hessian matrix of the Lagrangian function and $G \equiv \nabla_x c$ is the Jacobian matrix of the constraints. With $g \equiv \nabla_x \mathcal{J}$ we denote the gradient of the objective function. As stated in Algorithm 1, the QNB algorithm consists of three key components: the numerical construction of the KKT matrix in step 3, the computation of the Newton step Δp in step 4, and the selection of an appropriate damping parameter $\alpha^{(k)}$ in step 5.

We next discuss these three components in order and in detail below.

Algorithm 1 A quasi-Newton algorithm with backtracking (QNB).

```
Input: Given initial guess vector $p^{(0)} = (p_x^{(0)}, p_\lambda^{(0)})^T$ and prescribed tolerances $atol$ and $rtol$
1: Set $k = 0$
2: while $\|F(p^{(k)}\| \geq atol$ and $\|F(p^{(k)})\| \geq rtol\|F(p^{(0)}\|$ do
3:     Form the KKT system approximately,
           $$K^{(k)} = \begin{bmatrix} H^{(k)} & {G^{(k)}}^T \\ G^{(k)} & \mathbf{0} \end{bmatrix} \text{ and } F^{(k)} = \begin{bmatrix} g^{(k)} - {G^{(k)}}^T p_\lambda^{(k)} \\ -c^{(k)} \end{bmatrix},$$
       where the superscript $k$ indicates that all terms of the KKT system are evaluated at $p^{(k)}$.
4:     Find the Newton search direction $\Delta p^{(k)}$ by solving $K^{(k)}\Delta p^{(k)} = -F^{(k)}$ inexactly.
5:     Choose a damping parameter $\alpha^{(k)} \in (0, 1]$ by using the backtracking technique.
6:     Update $p^{(k+1)} = p^{(k)} + \alpha^{(k)}\Delta p^{(k)}$ and set $k = k + 1$.
7: end while
Output: $p^{(k)}$
```

3.2. KKT matrix construction. The KKT matrix is in a 2×2 block form of saddle point type with a zero (2,2) block. Compared to the Hessian matrix, the computational cost of constructing the Jacobian matrix of the constraints is relatively low. Hence, we focus on the Hessian matrix in this work. Following the suggestion in [20], we propose the use of a BFGS method for SQP [9, 17, 20] to build a quasi-Newton approximation $B^{(k)}$ to $H^{(k)}$. Two alternative numerical approaches for computing the Hessian matrix $H^{(k)}$ are finite difference (FD) approximation and automatic differentiation, AD. The ideas of FD, AD, and BFGS are not new, and the three approaches have been discussed in many numerical optimization books; see, e.g., [20]. However, a comparison of these three approaches in the framework of direct full-space methods with applications to trajectory optimization problems is not available in the literature, and we will report a comparative study in terms of efficiency in Section 4.6. For the FD method we can, for example, use a second-order central scheme for $H = (H_{ij})$ at $p^{(k)}$, which is given as

$$H_{ij}^{(k)} \approx \frac{1}{2\eta\, 2\xi}\Big(\mathcal{L}(p^{(k)} + \eta e_i + \xi e_j) - \mathcal{L}(p^{(k)} - \eta e_i + \xi e_j) \\ - \mathcal{L}(p^{(k)} + \eta e_i - \xi e_j) + \mathcal{L}(p^{(k)} - \eta e_i - \xi e_j)\Big), \tag{3.1}$$

where $\eta, \xi > 0$, $0 \le i, j \le n$, and e_i is the ith unit vector. The accuracy of this approximation depends not only on the selection of η and ξ but also on the regularity of $\mathcal{L}$. Some potential shortcomings of FD are as follows: When the function value of $\mathcal{L}$ changes rapidly in some direction, the approximation (3.1) may result in a large error. Also, the element-wise calculation of the KKT matrix is quite costly if the gradient of $\mathcal{L}$ is not available. An improvement is to take advantage of the sparsity of the KKT system and compute it by skipping the known zero elements. On the other hand, AD is a class of computational techniques for constructing the derivatives of a given function in analytical form automatically by using a computer software package. The idea of AD is based on the fact that most functions can be expressed as a composition of some elementary functions and a series of arithmetic operators. By recursively applying the chain rule in calculus, the evaluation of a derivative turns into serial operations on the values of elementary functions and their derivatives. Therefore, it can be performed automatically with any programming language capable of operator overloading. For further details, interested readers may consult the reference [20].

We now give a description of the BFGS algorithm. Let $B^{(0)}$ be a given positive definite matrix. Assume that $B^{(k-1)}$ is an approximation to $H^{(k-1)}$ and the current approximate solution $p^{(k)} = (p_x^{(k)}, p_\lambda^{(k)})^T$ is available. Since the update of the KKT matrix is limited to $B^{(k)}$, we can only consider

$$s^{(k)} \equiv p_x^{(k)} - p_x^{(k-1)} \quad \text{and} \quad y^{(k)} \equiv g(p_x^{(k)}, p_\lambda^{(k)}) - g(p_x^{(k-1)}, p_\lambda^{(k)})$$

rather than the full space vectors. Note that the curvature condition $[s^{(k)}]^T y^{(k)} > 0$, which ensures the positive definiteness of $B^{(k)}$ in the undamped BFGS method for unconstrained optimization, is not guaranteed in this case. Hence, we modify $y^{(k)}$, if necessary, and introduce a new variable $r^{(k)}$ as

$$r^{(k)} \equiv \theta^{(k)} y^{(k)} + (1 - \theta^{(k)}) B^{(k)} s^{(k)},$$

where the scalar $\theta^{(k)} \in [0, 1]$ is chosen as

$$\theta^{(k)} \equiv \begin{cases} 1 & \text{if } [s^{(k)}]^T y^{(k)} \ge 0.2\,[s^{(k)}]^T B^{(k)} s^{(k)}, \\ \dfrac{0.8[s^{(k)}]^T B^{(k-1)} s^{(k)}}{[s^{(k)}]^T B^{(k-1)} s^{(k)} - [s^{(k)}]^T y^{(k)}} & \text{otherwise.} \end{cases}$$

The selection of θ here is suggested in [20]. Note that such $r^{(k)}$ satisfies the curvature condition $[s^{(k)}]^T r^{(k)} > 0$, and it can be verified that the new $B^{(k)}$, updated by

$$B^{(k)} = B^{(k-1)} - \frac{B^{(k-1)} s^{(k)} [s^{(k)}]^T B^{(k-1)}}{[s^{(k)}]^T B^{(k-1)} s^{(k)}} + \frac{r^{(k)} [r^{(k)}]^T}{[s^{(k)}]^T r^{(k)}},$$

is symmetric and positive definite. In addition, such a choice of $\theta^{(k)}$ leads to $B^{(k+1)}$ being a positive definite matrix interpolating $B^{(k)}$ (for $\theta^{(k)} = 0$) and the matrix updated from the undamped BFGS method for $\theta^{(k)} = 1$.

3.3. Newton step computation. A tremendous amount of research is dedicated to the development of an efficient solution algorithm for solving saddle-point problems like the KKT system [1]. Some popular methods include full-space methods, e.g., direct block factorization methods or domain decomposition-based preconditioned Krylov-subspace iterative methods and reduced-space methods such as dual-space, range-space, or null-space methods [20], to name a few. Among them, both the Schwarz preconditioner [23] and the Schur preconditioner [7] belong to the family of domain decomposition methods. These two preconditioners are suitable to be implemented on distributed-memory machines and are designed for PDE-constrained optimization problems. However, solving the KKT system arising from the trajectory optimization problem only contributes a small portion of the total computational time, for example, only 11% for our targeted problem as shown in Table 4.6. The reasons for this phenomenon are twofold: Firstly, in this work we assume that the objective function and the constraints are provided by the user. The associated Hessian and gradient constructions are done numerically by one of the approaches mentioned in Section 3.2, which requires many evaluations of the constraints. Secondly, the number of state variables is relatively small compared to the one for PDE-constrained optimization problems. Hence, the benefit from parallel computing for our numerical solution of the KKT system is expected to be marginal. Here, we consider a variant of an incomplete lower/upper triangular decomposition to construct a preconditioner in conjunction with GMRES, namely incomplete LU decomposition with a threshold and pivoting, ILUTP [26].

3.4. Globalization strategies. Once the Newton search direction $\Delta p^{(k)}$ is determined, it needs to be appropriately scaled to ensure the global convergence of Newton's method, and a merit function is used to monitor its progress towards the desired solution. Here, we employ the augmented Lagrangian merit function $\mathcal{M}_\rho$: $\mathbb{R}^n \to \mathbb{R}$, which is defined as

$$\mathcal{M}_\rho(p) \equiv \mathcal{L}(p) + \frac{\rho}{2} c(p_x)^T c(p_x).$$

Note that this merit function tries to balance the conflict between forcing the objective function to decrease and satisfying the constraint.

The weight parameter $\rho^{(k)}$ as well as the scaling factor $\alpha^{(k)}$ are updated in sequence based on the following two rules.

1. Select $\rho^{(k)}$ so that $\Delta p^{(k)}$ is the descent direction of $\mathcal{M}_{\rho^{(k)}}$ at $p^{(k)}$. As suggested in [23], we choose $\rho^{(k)} = \max\{\bar{\rho}^{(k)}, \rho^{(k-1)}\}$ with $\rho^{(-1)} > 0$ and
$$\bar{\rho}^{(k)} = -2\,\frac{[\nabla \mathcal{L}^{(k)}]^T p^{(k)}}{[c^{(k)}]^T \nabla c^{(k)} p_x^{(k)}},$$
which is updated at each Newton iteration.
2. Choose $\alpha^{(k)} = \alpha \in [\alpha_{\min}, 1]$ such that the merit function $\mathcal{M}_{\rho^{(k)}}$ decreases sufficiently along $p^{(k)}$ to satisfy the *Armijo condition*
$$\mathcal{M}_{\rho^{(k)}}(p^{(k)} + \alpha\,\Delta p^{(k)}) \leq \mathcal{M}_{\rho^{(k)}}(p^{(k)}) + \alpha\,\beta\,[\nabla \mathcal{M}_{\rho^{(k)}}(p^{(k)})]^T p^{(k)}, \tag{3.2}$$
where the parameter $\beta \in (0, 0.5)$ is used to assure that the decrease of $\mathcal{M}_\rho$ is sufficient and the parameter $\alpha_{\min} > 0$ acts as a safeguard required for the strong global convergence. Here, a quadratic linesearch technique [9] is employed to determine the step length $\alpha^{(k)}$.

4. Numerical results and discussion. Besides the multistage satellite launch vehicle problem, we consider one additional benchmark problem called the Earth-to-Mars orbit transfer problem [8, 11, 32], which is commonly used for testing or evaluating the performance of new algorithms. This test problem can be viewed as a special case of the multistage satellite launch vehicle problem. The final time is given, i.e., t_f is known, and only a single stage is considered with constant thrust, mass, and gravity. Sections 4.1 and 4.2 provide detailed descriptions of these two numerical examples including the physical parameters used in the numerical experiments. Section 4.6 reports on the performance of the FQLNK algorithm for solving the parameter optimization problem (2.9). The FQLNK code was developed using Matlab, and all computations were carried out in double precision. We consider our solver as converged, when the following conditions are met: $\|F(p^{(k)})\|_2 \leq atol$ or $\|F(p^{(k)})\|_2 \leq rtol\|F(p^{(0)})\|_2$, where both the absolute tolerance $atol$ and the relative tolerance $rtol$ are set to 10^{-6}. ILUTP-preconditioned GMRES is used to solve the KKT system. The dropping tolerance τ of ILUTP is set to 10^{-6}. The accuracy of the solution to the KKT systems is controlled by the parameter η_k to force the condition

$$\|F(p^{(k)}) + K^{(k)}\Delta p\| \leq \eta_k \|F(p^{(k)})\|$$

to be satisfied. We set $\eta_k = 10^{-6}$.

4.1. Earth-to-Mars orbit transfer problem. The Earth-to-Mars orbit transfer problem, whose geometrical setting is shown in Figure 4.1, can be described mathematically in dimensionless form as follows. Find the control history $\varphi(t)$, $t \in [t_0, t_f]$, to minimize the performance index

$$\mathcal{J} = -r(t_f)$$

subject to the dynamic equations of motion

$$\dot{s} = \frac{d}{dt}\begin{bmatrix} r \\ u \\ v \end{bmatrix} = f(r,u,v,\varphi) \equiv \begin{bmatrix} u \\ \dfrac{v^2}{r} - \dfrac{1}{r^2} + \dfrac{T}{m}\sin\varphi \\ -\dfrac{uv}{r} + \dfrac{T}{m}\cos\varphi \end{bmatrix},$$

with the condition for the flight in the initial orbit

$$\psi(t_0) \equiv \begin{bmatrix} r(t_0) - 1.0 \\ u(t_0) \\ v(t_0) - 1.0 \end{bmatrix} = 0,$$

and the condition for the final orbit insertion

$$\psi(t_f) \equiv \begin{bmatrix} u(t_f) \\ v(t_f) - \sqrt{\dfrac{1}{r(t_f)}} \end{bmatrix} = 0, \tag{4.1}$$

where $s(t) = (r(t), u(t), v(t))^T$ are the state variables. Here, r is the radial distance between the spacecraft and the attracting center, and u and v are the radial and tangential components of the velocity, respectively. In addition, T is the thrust, and $m(t) = m_0 - |\dot{m}|t$ is the mass of the spacecraft with a given initial mass m_0 and a constant fuel consumption rate $|\dot{m}|$. In the numerical experiments, the values of these constants are specified as $t_0 = 0$, $t_f = 3.32$, $T = 0.1405$, $m_0 = 1.0$, and $|\dot{m}| = 0.0749$.

Note that this orbit transfer problem is relatively simple such that indirect methods can be easily applied [8]. The numerical solution to the system of two-point boundary ordinary differential equations obtained by the Matlab routine bvp4c serves as the reference solution used for comparison with the one obtained with our proposed method.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

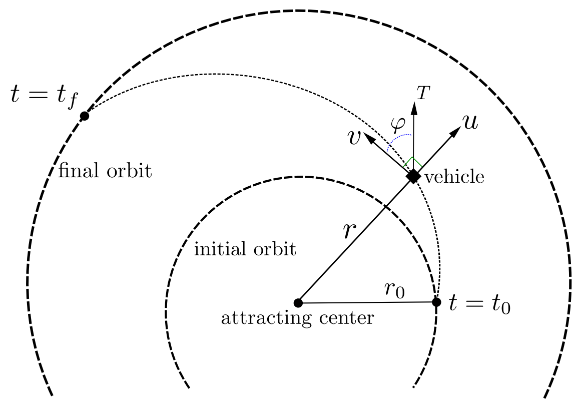

FIG. 4.1. *The geometrical configuration for the orbit transfer problem.*

4.2. Three-stage satellite launch vehicle problem. For the multistage launch vehicle problem, we consider a three-stage satellite launch vehicle as a numerical example. The main task of the mission is to deliver a micro-satellite of weight ranging between 40 and 120 kg into a low-Earth circular orbit with an altitude of 500 km. Table 4.1 lists the structure and propulsion data for each stage of the launch vehicle, and Table 4.2 summarizes its launch process. The physical parameters involved in the numerical experiment are specified as follows: the average radius of the Earth, $R = 6.378 \times 10^3$ km, the air density at ground, $\rho_0 = 1.225\,\mathrm{kg/m^3}$, the atmospheric scale height, $H = 7.6$ km, and the gravity at sea level, $g_0 = 9.80665 \times 10^{-3}$ km. Figure 4.2 displays a plot of the drag coefficient C_D as a function of the Mach number.

TABLE 4.1
Data of the three-stage launch vehicle.

Stage	I	II	III
Reference area (m^2)	0.7854	0.7854	0.1564
Motor mass (kg)	10091	1906	344
Propellant mass (kg)	8880	1677	296
Thrust (Nt)	243824	57555.4	6085.8
Burn time (sec)	100	80	133

TABLE 4.2
A list of the key events during the launch process.

Time (sec)	Events
$t_0 = 0$	Stage 1 ignition and liftoff
$t_1 = 5$	Beginning of kick-turn
$t_2 = 100$	Stage 1 burnout, stages 1 & 2 separation, and stage 2 ignition
$t_3 = 180$	Stage 2 burnout, stages 2 & 3 separation, and beginning of free flight
$t_4 = 180 + \Delta t_c$	End of free flight period and stage 3 ignition
$t_5 = 313 + \Delta t_c$	Stage 3 burnout and orbit insertion

Some additional detailed information is given below. First of all, the launch vehicle takes

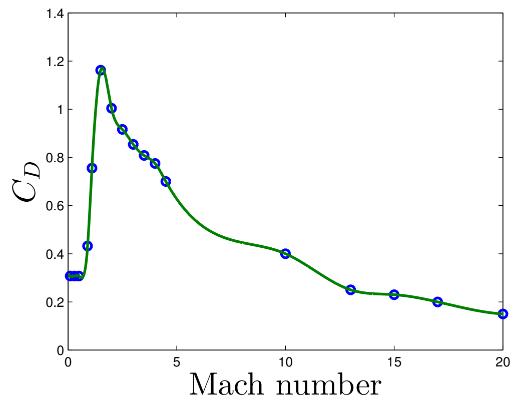

FIG. 4.2. *The drag coefficient C_D as function of the value of Mach number.*

off and climbs vertically before a gravity turn begins. Therefore, an additional path constraint

$$\varphi(t) = \frac{\pi}{2}, \qquad t \in [t_0, t_1],$$

is imposed. Second, the payload fairing, weighing 50 kg, that encapsulates and provides protection for the payload, separates when the launch vehicle reaches the altitude of 100 km. Finally, the total mass of the launch vehicle steadily decreases in the powered flight due to the burning of fuel and drops suddenly when each stage, as well as the payload fairing, separates from the launch vehicle at the end of its burn time. By taking these facts into account, this test problem has five phases.

We notice that all state parameters and corresponding dynamic equations with units differ by several orders of magnitude. Poor scaling may lead to slow convergence of an iterative method or inaccuracy of the numerical solutions. To achieve better scaling, we transformed the problem into a dimensionless one by introducing canonical units including the time unit $TU = 806.8$ sec, the distance unit $DU = R = 6378.165$ km, and the initial total mass of the launch vehicle M_{ref} as the characteristic of time, the characteristic of length, and the characteristic of mass, respectively. As a consequence, the dimensionless variables and parameters are defined as

$$\bar{u} = \frac{u}{DU/TU}, \quad \bar{v} = \frac{v}{DU/TU}, \quad \bar{x} = \frac{v}{DU}, \quad \bar{y} = \frac{v}{DU}.$$
$$\bar{t} = \frac{t}{TU}, \quad \bar{M} = \frac{M}{M_{ref}}, \quad \bar{g} = \frac{g}{DU/TU^2}, \quad \bar{T} = \frac{T}{(M_{ref}DU)/TU^2}.$$
$$\bar{S}_{ref} = \frac{S_{ref}}{DU^2}, \quad \bar{\rho}_0 = \frac{\rho_0}{M_{ref}/DU^3}, \quad \bar{\mu} = \frac{\mu}{DU^3/TU^2} = 1.$$

Both differential constraints and boundary conditions take the same form in dimensional and dimensionless formulation. All of our calculations are done in dimensionless form.

4.3. Selection of the initial guess. The selection of an initial guess vector $p^{(0)}$ as input for Algorithm 1 is crucial since the convergence of most nonlinear iterative methods strongly depends on the initial guess, and there is no exception for the Newton-type method. Failure of the algorithm may happen due to a bad initial guess. In general, we desire a guess that is simple (even trivial) and easy to obtain such as the zero vector. However, our numerical experience suggests that such a naive initial guess sometimes causes a highly ill-conditioned KKT system. In this case, the choice of a "good" initial guess could be problematic. For the trajectory optimization problem, we borrow the idea from the reduced-space approach to generate a more reasonable (and maybe much better) initial guess vector. To start with, we give a guess for the design parameter, the coasting-flight period Δt_c, if needed, and set some "reasonable" control variables based on some a priori physical knowledge. Then the discrete state variables on each grid point can be calculated by performing some numerical integration for the right-hand side of the differential constraints. Finally, to initialize the discrete Lagrangian multiplier vector, we employ the *least-squares multipliers estimate*, which equivalently solves the normal equation

$$\lambda^{(0)} = \left[G^{(0)} {G^{(0)}}^T\right]^{-1} G^{(0)} g^{(0)},$$

which is motivated from minimizing the first term on the left-hand side of the KKT system in step 4 in Algorithm 1. Figure 4.3 displays the two initial guesses for the control variable used in the numerical experiments.

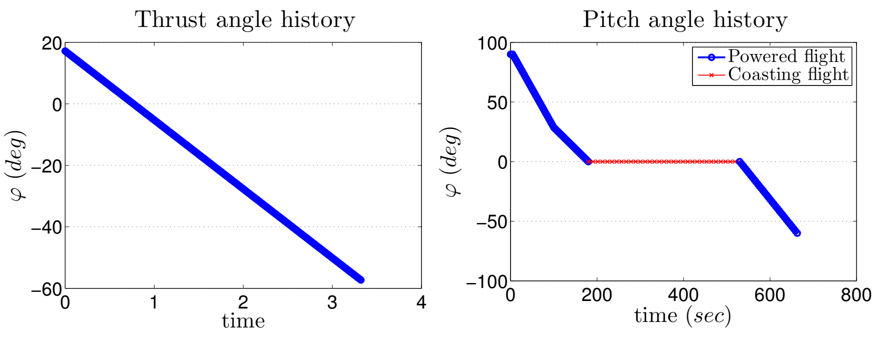

FIG. 4.3. *The initial guesses for the control variable for two test cases. Left: orbit transfer problem, right: three-stage lunch vehicle problem.*

In Figure 4.4, the convergence sensitivity analysis of our proposed method for the three-stage launch vehicle problem is presented. We tested different values of the coasting-flight period Δt_c ranging from 70 to 250 seconds. Beyond this range, we will produce either overshooting or undershooting trajectories. From this figure, we observe that except for the extreme values, the number of Newton iterations depends mildly on the choice of the coasting-flight period.

4.4. Grid test and its comparisons with indirect solution. To validate the implementation of the FQLNK algorithm, we perform a grid independent study for both test problems. For the Earth-to-Mars orbit transfer problem, we use a set of grids with different sizes from $3.32/8$ to $3.32/256$. For the three-stage launch vehicle problem, we select a fixed number of grids for each phase: $M = 4, 8, 12, 16$, and 32. Tables 4.3 and 4.4 provide a convergence analysis of the control and state parameters for the two test cases. All errors are computed in the infinity norm, and the finest grid solution is used as the exact solution. From the tables, we observe that all numerical state parameters converge to the desired solution at a quadratic convergence rate. The convergence behavior of our numerical solution is consistent with the theoretical prediction when the composite trapezoidal rule is employed for the dynamical constraints.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

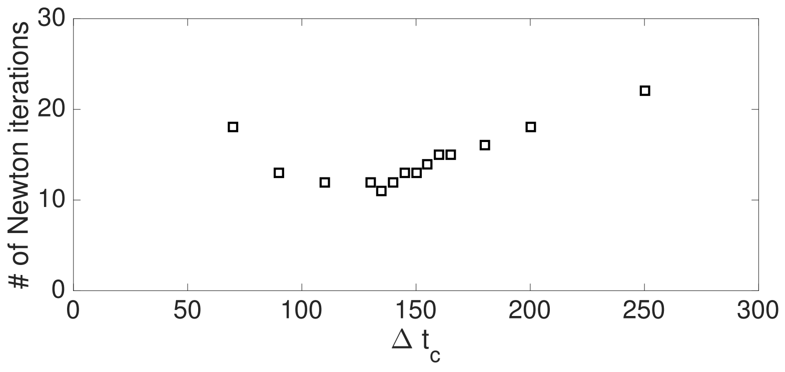

FIG. 4.4. *Sensitivity analysis of the FLNKS algorithm for different coasting-flight periods.*

TABLE 4.3
Grid test for the Earth-to Mars orbit transfer problem, "p.i." is the optimal value of the performance index. The numerical solution with the finest grid $h = 3.32/256$ is used as the exact solution for the error calculation.

# subint.	h	p.i.	φ	r	u	v
8	4.1500e-01	3.8179e-02	3.4605e+00	4.1236e-02	5.5937e-02	7.3486e-02
16	2.0750e-01	5.0925e-03	1.4181e-01	5.3232e-03	1.1195e-02	3.5663e-03
32	1.0375e-01	1.2153e-03	3.0599e-02	1.2953e-03	2.9655e-03	1.3774e-03
64	5.1875e-02	2.9106e-04	8.3307e-03	3.1329e-04	7.6265e-04	4.3971e-04
128	2.5938e-02	5.8325e-05	2.7179e-03	6.2582e-05	1.7809e-04	9.1695e-05
conv. rate		2.2838	2.4718	2.2815	2.0466	2.2313

Adaptive mesh refinement (AMR) with an a-posteriori error estimation techniques is one of the practical approaches for finding an optimal solution with the minimal number of grid points employed. But for our target application, the optimal control problem (an optimization problem constrained by a system of ODEs), its problem size is linearly dependent on the number of time steps. Hence, the dimension issue is not as severe as the one for the PDE-constrained problem even when just using a uniform time step refinement. On the other hand, the accuracy of the optimal solution for our approach depends on the choice of the time integrator for the dynamic constraint. For example, for the composite trapezoidal rule as used in the article, all numerical state parameters converge with a quadratic convergence rate, while the control parameters converge superlinearly. Such an a-priori error estimate is useful for engineers or practitioners to determine an appropriate time step size based on their needs.

TABLE 4.4
The grid test for the three-stage launch vehicle problem, "p.i." is the optimal value of the performance index. The numerical solution with the finest grid (2, 32, 32, 32, 32) is used as the exact solution for the error calculation.

# subint.	p.i.	φ	u	v	x	y
(2,4,4,4,4)	1.9827e-02	1.2540e-01	1.0406e-02	3.0956e-02	1.2449e-02	2.1882e-03
(2,8,8,8,8)	4.8262e-03	4.5499e-02	2.2711e-03	7.3778e-03	3.0313e-03	6.0661e-04
(2,12,12,12,12)	2.0843e-03	2.4556e-02	1.0070e-03	3.1631e-03	1.3059e-03	2.7793e-04
(2,16,16,16,16)	1.1222e-03	1.4625e-02	5.7386e-04	1.6858e-03	7.0082e-04	1.4367e-04
conv. rate	2.1654	1.6114	2.1903	2.1948	2.1694	2.0363

Figures 4.5 and 4.6 provide a comparison of two solution plots for the Earth-to-Mars

orbit transfer problem obtained by the indirect method and the proposed method, respectively. These series of plots include the history of the thrust angle, the radius variation, and the velocity in both the radial and tangential directions. At first glance, except for the control parameter profiles, the two sets of solution plots are almost indistinguishable. The relative difference of the radii of the final orbit between the two solutions is smaller than 0.01. As shown in Figure 4.6, the only noticeable difference for the control parameter profile is that the discontinuity of the direct solution near the middle of the computational domain is stronger than that of the indirect solution. To evaluate the performance of our method and of the indirect method, we perform a numerical simulation by using two computed control profiles, displayed in Figure 4.6, as the guidance law. We compare the simulation errors at the final time (4.1). Here, an explicit fourth-order Runge-Kutta time integrator is used for the simulation. We find that our method produces more accurate numerical results than the indirect method does. The numerical errors for our methods are 3.9e-3 and 1.5e-3, respectively, which is one order of magnitude smaller than the ones for the direct method which yields the errors 2.4e-2 and 6.2e-2. We should mention that such discontinuity is mainly due to the domain of the control variable θ which is chosen to be $[-\pi, \pi]$; see also [11, 32]. On the other hand, the right-hand side of Figure 4.6 displays a mathematically equivalent plot but now defined on $[0, 2\pi]$, which is continuous everywhere. This figure suggests that we control the thrust angle rapidly around $t = 1.6$ to steer the spacecraft towards a feasible point of orbit insertion.

4.5. Typical numerical solutions for the three-stage launch problem. In this section we present a typical numerical solution of the three-stage launch vehicle problem for the case that the weight of the payload is set to 100 kg. The numerical solution is obtained on a grid with 32 subintervals for each phase. In this case, the launch vehicle reaches the orbit insertion point after 410.80 sec, including a coasting-flight period of 97.76 sec. Figure 4.7 displays the histories of the controlled pitch angle, the velocity, the down range, and the optimal trajectory.

We next study how the payload weight affects the optimal trajectory of the launch vehicle. As shown in Figure 4.8, all optimal trajectories are similar for different payload weights but the duration of the coasting-flight period is not. The coasting flight time becomes longer as the weight of the payload increases. When the weight of the payload is less than 43 kg, the launch vehicle does not require a coasting-flight period in the space mission.

4.6. Performance evaluation of the LNK algorithm. We further numerically investigate the robustness and the efficiency of our proposed algorithm, where the BFGS formula is used for the update of the Hessian matrix $B^{(k)}$. To begin with, we build an initial Hessian matrix $B^{(0)}$ by using the AD approach. For comparison, we also report the numerical results obtained by using the FD and AD approach for all Hessian matrices $H^{(k)}$. We refer to these three solution algorithms as BFGS, FD, and AD, respectively. For FD, the second-order central difference formula (3.1) with $\eta = 10^{-3}$ and $\xi = 10^{-3}$ is used. For AD, an open Matlab source code [12] is employed. Besides, for all cases, the Jacobian matrices of the constraints $G^{(k)}$ are computed approximately by using a forward finite difference scheme. Tables 4.5 and 4.6 give the total number of Newton iterations, the average number of GMRES iterations per Newton iteration, and the computing time in seconds spent by the proposed algorithm with different grid sizes. The timing percentages spent by some selected main components are also included. Figure 4.10 presents the history of the nonlinear residual norm $\|F^{(k)}\|$ of AD and BFGS with or without using backtracking techniques. In the following we summarize some key findings observed from the tables.

First, for the Earth-to-Mars problem, we find that the number of Newton iterations increases as the grid is refined no matter which approach is used for the Hessian matrix construction. This is a typical example of a problem with local nonlinearity for which a Newton-type method has difficulties to converge. One possible solution is to employ some

ETNA
Kent State University and
Johann Radon Institute (RICAM)

nonlinear preconditioner to enhance the efficiency and robustness of the inexact Newton method. Interested readers are referred to [15, 16, 34] for details. On the other hand, we notice that in some cases FD and AD might converge to some nonphysical local minimum or saddle point, see Figure 4.9, while BFGS converges nicely to the desired solution. The performance index for the nonphysical case is 1.504 which is slightly worse than the one for the physical case with the value 1.525.

Second, Figure 4.10 suggests that AD exhibits local quadratic convergence behavior, while BFGS yields, as expected, only super-linear convergence which reflects the different natures of the Newton-type and the quasi-Newton methods. Besides, the importance of the merit function and backtracking process is evident. AD without backtracking technique results in either convergence failure or a slower rate of convergence.

Third, GMRES in conjunction with the ILUPT preconditioner is quite effective, the average numbers of GMRES iterations are all less than five and are almost independent of the problem size.

The data listed in the fourth and fifth columns indicates that the most computationally demanding part of the entire algorithm is the construction of the Hessian matrix. This component contributes more than 72% of the total computing time for all cases. On the other hand, for PDE-constrained optimization problems, we have a different situation: the most expensive component is typically the numerical solution of the KKT system.

Finally, we note that BFGS demonstrates an excellent speedup compared to AD and FD. BFGS is eight times faster than AD and FD.

TABLE 4.5
The Earth-to-Mars orbit transfer problem. A comparison of the FQLNK algorithm with three approaches for constructing the KKT system. The symbol "" indicates that the FQLNK algorithm converges to an nonphysical local minimum.*

# subint.	Newton Ites	Avg. GMRES Ites	Avg. Hessian Form (sec)	Avg. KKT Solve (sec)	Total Time (sec)	Speedup
			FD			
8	6	1.2	0.29 (97.2%)	0.001 (<1%)	1.79	–
16	8	1.4	0.98 (99.1%)	0.002 (<1%)	7.91	–
32	8	1.8	3.62 (99.5%)	0.003 (<1%)	29.11	–
64	14	2.0	13.81 (99.6%)	0.010 (<1%)	194.07	–
128^*	21	2.1	52.60 (99.8%)	0.051 (<1%)	1107.38	–
			AD			
8	6	1.2	0.28 (94.4%)	0.007 (<1%)	1.78	1.0x
16	8	1.4	0.50 (95.7%)	0.004 (<1%)	4.13	1.9x
32	8	1.8	1.20 (98.5%)	0.005 (<1%)	9.75	3.0x
64	14	2.0	3.18 (98.7%)	0.010 (<1%)	45.12	4.3x
128^*	21	2.1	18.92 (99.4%)	0.045 (<1%)	399.71	2.8x
			BFGS			
8	14	1.3	0.03 (76.3%)	0.003 (7.6%)	0.55	3.3x
16	17	1.8	0.04 (72.3%)	0.003 (5.4%)	0.94	8.4x
32	16	2.0	0.10 (78.4%)	0.009 (7.1%)	2.04	14.3x
64	14	2.0	0.27 (77.9%)	0.043 (12.4%)	4.85	40.2x
128	15	2.1	1.45 (83.0%)	0.231 (13.1%)	26.22	42.2x
256	25	3.0	7.21 (83.2%)	1.320 (15.2%)	216.62	–

ETNA
Kent State University and
Johann Radon Institute (RICAM)

TABLE 4.6
The three-stage launch vehicle problem. A comparison of the FQLNK algorithm with three approaches for constructing the KKT system.

# subint.	Newton Ites	Avg. GMRES Ites	Avg. Hessian Form (sec)	Avg. KKT Solve (sec)	Total Time (sec)	Speedup
			FD			
8	7	3.0	16.14 (99.7%)	0.004 (<1%)	113.32	–
12	8	3.0	32.19 (99,8%)	0.009 (<1%)	258.13	–
16	9	3.6	53.76 (99.8%)	0.015 (<1%)	484.74	–
32	8	2.9	190.21(99.9%)	0.072 (<1%)	1523.54	–
			AD			
8	10	2.8	2.64 (98.1%)	0.004 (<1%)	26.92	4.2x
12	11	2.7	5.64 (98.6%)	0.008 (<1%)	62.76	4.1x
16	10	3.2	9.68 (99.1%)	0.014 (<1%)	97.72	5.0x
32	10	4.1	101.10 (99.8%)	0.088 (<1%)	1013.40	1.5x
			BFGS			
8	23	3.5	0.16 (72.7%)	0.018 (8.2%)	5.06	22.4x
12	24	3.0	0.30 (74.0%)	0.040 (9.9%)	9.73	26.5x
16	20	4.5	0.59 (77.4%)	0.085 (11.1%)	15.24	31.8x
32	20	4.7	5.37 (90.0%)	0.420 (7.0%)	119.29	12.7x

4.7. An extension to a problem with inequality constraints. We now consider the OCP1 problem along with an additional mixed state and control constraint, i.e.,

$$-\frac{\pi}{36} \leq \varphi(t) - \theta(t) \leq \frac{\pi}{36}, \qquad t \in [0, 40]. \tag{4.2}$$

Many numerical techniques are available for handling the inequality constraint: active set methods [20], barrier function methods, semismooth Newton methods [35], and the introduction of a slack variable, to name a few. Our proposed FQLNK algorithm is general and can be readily employed in conjunction with one of these techniques with some modification. For illustration purposes, we use the slack variable approach [14, p. 64], which is relatively simple, along with FQLNK. Our FQLNK code, developed only for equality constraints, can be generalized to the trajectory optimization with inequality constraints in a straightforward manner, and the overhead of our FQLNK is expected to be rather marginal. As shown on the left of Figure 4.11, the right inequality condition $\alpha < \pi/36$ has already been satisfied for the original solution. Hence, we introduce a new slack variable ϵ to reformulate the one-sided inequality constraint only on the left of the equality constraint condition,

$$-\frac{\pi}{36} \leq \varphi(t) - \theta(t) \quad \Rightarrow \quad \varphi - \theta + \frac{\pi}{36} - \epsilon^2 = 0. \tag{4.3}$$

Then the mixed state and control equality constraint (4.3) is discretized on each grid point as we did before for the dynamic constraints, and FQLNK can be directly applied. Note that the discrete slack variables are treated as some new auxiliary components of the full space unknown vector, and their values are determined when the optimal solution has been found by FQLNK. Figure 4.11 displays a comparison of histories of two angles of attack before (right) and after (left) the inequality condition (4.2) is imposed.

5. Concluding remarks. In this work we have proposed and studied the full-space quasi-Lagrange-Newton-Krylov method for solving the parameter optimization problem resulting from the optimal trajectory design in aerospace applications. One of the main contributions is to show numerically that using the BFGS method is an efficient way to construct the Hessian matrix in the KKT system. The BFGS-based FQLNK algorithm exhibits an impressive speedup compared to the FD- and AD-based ones. Other key ingredients of the FQLNK algorithm include the efficient ILU type preconditioner for GMRES in the numerical solution of the KKT system that provide a high quality of the Newton search direction, along with an appropriate merit function and backtracking technique that assure the progress of the inexact Newton method toward the desired solution. Taking both the three-stage launch vehicle problem and the Earth-to-Mars orbit transfer problem as numerical examples, we have demonstrated the robustness and efficiency of the FQLNK algorithm for solving trajectory optimization problems. Our numerical results show that the BFGS-based FQLNK algorithm is a promising approach for solving trajectory optimization problems with aerospace engineering applications. Some further possible works along this research direction include the extension of 3D dynamic constraints for the motion of launch vehicles and the generalization of multi-objective optimization problems [19], which are currently under development.

Acknowledgements. The authors thank the anonymous reviewers for constructive comments for improving the presentation of the manuscript.

REFERENCES

[1] M. BENZI, G. H. GOLUB, AND J. LIESEN, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.
[2] J. T. BETTS, *Survey of numerical methods for trajectory optimization*, J. Guid. Contr. Dynam., 21 (1998), pp. 193–207.
[3] ———, *Very low-thrust trajectory optimization using a direct SQP method*, J. Comput. Appl. Math., 120 (2000), pp. 27–40.
[4] ———, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed., SIAM, Philadelphia, 2010.
[5] N. BIEHN, S. L. CAMPBELL, L. JAY, AND T. WESTBROOK, *Some comments on DAE theory for IRK methods and trajectory optimization*, J. Comput. Appl. Math., 120 (2000), pp. 109–131.
[6] G. BIROS AND O. GHATTAS, *Parallel Lagrange–Newton–Krylov–Schur methods for PDE-constrained optimization. II. The Lagrange–Newton solver and its application to optimal control of steady viscous flows*, SIAM J. Sci. Comput., 27 (2005), pp. 714–739.
[7] ———, *Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization. I. The Krylov–Schur solver*, SIAM J. Sci. Comput., 27 (2005), pp. 687–713.
[8] A. BRYSON AND Y.-C. HO, *Applied Optimal Control*, Hemisphere Publishing, Washington, 1975.
[9] J. DENNIS JR. AND R. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM, Philadelphia, 1996.
[10] D. J. ESTEP, D. H. HODGES, AND M. WARNER, *The solution of a launch vehicle trajectory problem by an adaptive finite-element method*, Comput. Methods Appl. Mech. Engrg., 190 (2001), pp. 4677–4690.
[11] F. FAHROO AND I. ROSS, *Costate estimation by a Legendre pseudospectral method*, J. Guid. Contr. Dynam., 24 (2001), pp. 270–277.
[12] M. FINK, *Automatic Differentiation for Matlab*, MATLAB Central File Exchange. Retrieved May 18, 2015, 2007.
[13] P. E. GILL, L. O. JAY, M. W. LEONARD, L. R. PETZOLD, AND V. SHARMA, *An SQP method for the optimal control of large-scale dynamical systems*, J. Comput. Appl. Math., 120 (2000), pp. 197–213.
[14] D. G. HULL, *Optimal Control Theory for Applications*, Springer, New York, 2003.
[15] F.-N. HWANG, H.-L. LIN, AND X.-C. CAI, *Two-level nonlinear elimination based preconditioners for inexact Newton methods with application in shocked duct flow calculation*, Electron. Trans. Numer. Anal., 37 (2010), pp. 239–251.
http://etna.ricam.oeaw.ac.at/vol.37.2010/pp239-251.dir/pp239-251.pdf
[16] F.-N. HWANG, Y.-C. SU, AND X.-C. CAI, *A parallel adaptive nonlinear elimination preconditioned inexact Newton method for transonic full potential equation*, Comput. & Fluids, 110 (2015), pp. 96–107.
[17] C. T. KELLEY, *Iterative Methods for Linear and Nonlinear Equations*, SIAM, Philadelphia, 1995.
[18] P. LU AND M. KHAN, *Nonsmooth trajectory optimization-an approach using continuous simulated annealing*, J. Guid. Contr, Dynam., 17 (1994), pp. 685–691.
[19] R. T. MARLER AND J. S. ARORA, *Survey of multi-objective optimization methods for engineering*, Struct. Multidiscip. Optim., 26 (2004), pp. 369–395.

[20] J. NOCEDAL AND S. J. WRIGHT, *Numerical Optimization*, 2nd ed., Springer, New York, 2006.
[21] M. A. PATTERSON AND A. V. RAO, $\mathbb{GPOPS}-\mathbb{II}$*: a MATLAB software for solving multiple-phase optimal control problems using hp-adaptive Gaussian quadrature collocation methods and sparse nonlinear programming*, ACM Trans. Math. Software, 41 (2014), Art. 1 (37 pages).
[22] M. PONTANI, *Particle swarm optimization of ascent trajectories of multistage launch vehicles*, Acta Astronaut., 94 (2014), pp. 852–864.
[23] E. E. PRUDENCIO, R. BYRD, AND X.-C. CAI, *Parallel full space SQP Lagrange-Newton-Krylov-Schwarz algorithms for PDE-constrained optimization problems*, SIAM J. Sci. Comput., 27 (2006), pp. 1305–1328.
[24] A. V. RAO, *Trajectory optimization: a survey*, in Optimization and Optimal Control in Automotive Systems, H. Waschl, I. Kolmanovsky, M. Steinbuch, and L. del Re, eds., vol. 455 of Lect. Notes Control Inf. Sci., Springer, Cham, 2014, pp. 3–21.
[25] W. ROH AND Y. KIM, *Trajectory optimization for a multi-stage launch vehicle using time finite element and direct collocation methods*, Eng. Optim., 34 (2002), pp. 15–32.
[26] Y. SAAD, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.
[27] M. R. SENTINELLA AND L. CASALINO, *Cooperative evolutionary algorithm for space trajectory optimization*, Celestial Mech. Dynam. Astronom., 105 (2009), pp. 211–227.
[28] K. SUBBARAO AND B. SHIPPEY, *Hybrid genetic algorithm collocation method for trajectory optimization*, J. Guid. Contr, Dynam., 32 (2009), pp. 1396–1403.
[29] S. SUBCHAN AND R. ŻBIKOWSKI, *Computational Optimal Control: Tools and Practice*, Wiley, New York, 2009.
[30] B. SURESH AND K. SIVAN, *Integrated Design for Space Transportation System*, Springer, New Dehli, 2015.
[31] O. VON STRYK AND R. BULIRSCH, *Direct and indirect methods for trajectory optimization*, Ann. Oper. Res., 37 (1992), pp. 357–373.
[32] P. WILLIAMS, *Jacobi pseudospectral method for solving optimal control problems*, J. Guid. Contr. Dynam., 27 (2004), pp. 293–297.
[33] A. WUERL, T. CRAIN, AND E. BRADEN, *Genetic algorithm and calculus of variations-based trajectory optimization technique*, J. Spacecraft Rockets, 40 (2003), pp. 882–888.
[34] H. YANG, F.-N. HWANG, AND X.-C. CAI, *Nonlinear preconditioning techniques for full-space Lagrange-Newton solution of PDE-constrained optimization problems*, SIAM J. Sci. Comput., 38 (2016), pp. A2756–A2778.
[35] H. YANG, S. SUN, AND C. YANG, *Nonlinearly preconditioned semismooth Newton methods for variational inequality solution of two-phase flow in porous media*, J. Comput. Phys., 332 (2017), pp. 1–20.

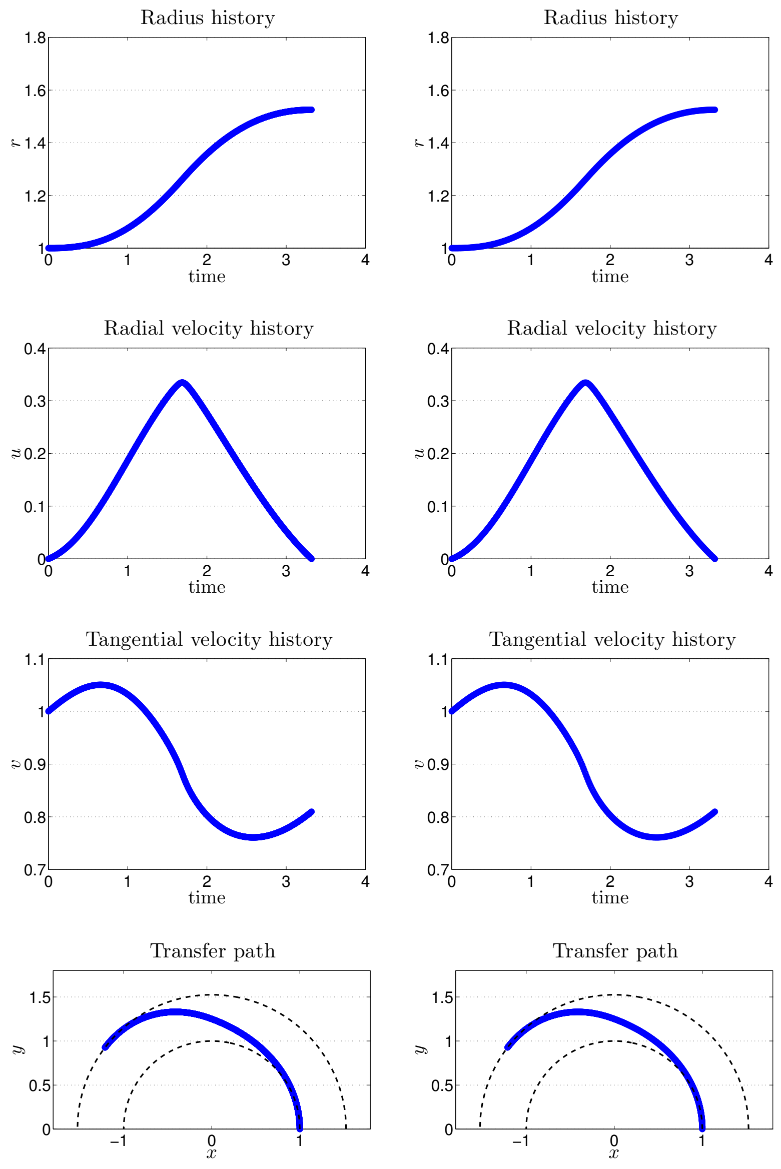

FIG. 4.5. *A comparison of two numerical state variables of the Earth-to-Mars orbit transfer problem obtained by using the indirect method (left column) and our method (right column).*

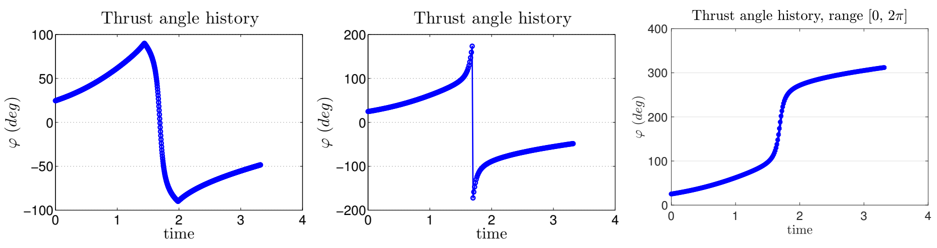

FIG. 4.6. *A comparison of two numerical control variables of the Earth-to-Mars orbit transfer problem obtained by using the indirect method (left), our method (middle), and the mathematically equivalent plot for our method (right).*

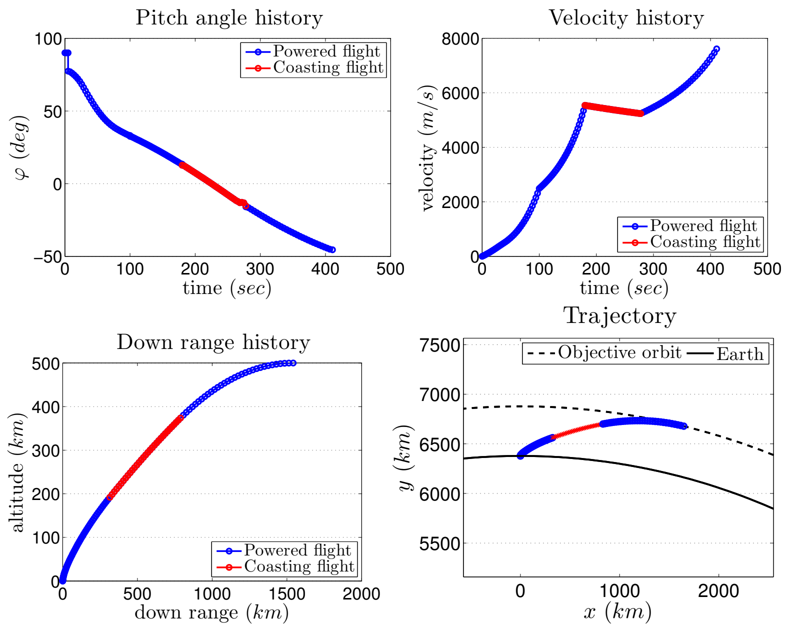

FIG. 4.7. *The histories of the control pitch angle (top-left), the velocity (top-right), the downrange (bottom-left), and optimal trajectory (bottom-right) for the multistage launch vehicle problem.*

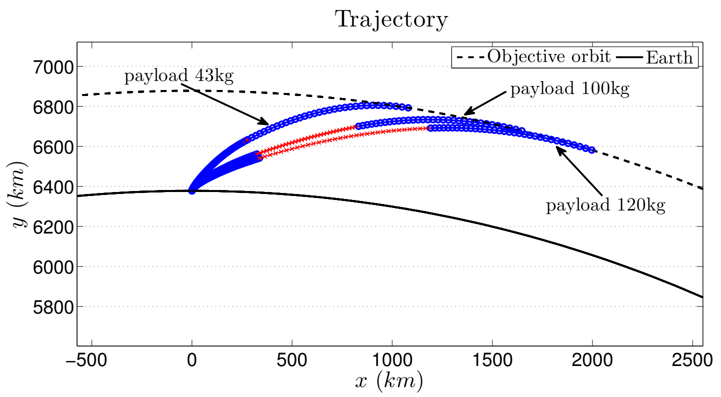

FIG. 4.8. *A comparison of optimal trajectories for different payload weights.*

ETNA
Kent State University and
Johann Radon Institute (RICAM)

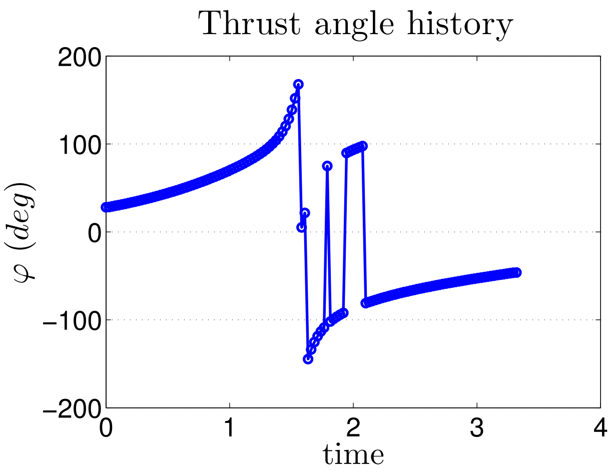

FIG. 4.9. *Earth-to-Mars orbit transfer problem. An nonphysical local minimum is obtained by FD. The performance index for this case is 1.516, which is worse than the value 1.525 obtained with BFGS.*

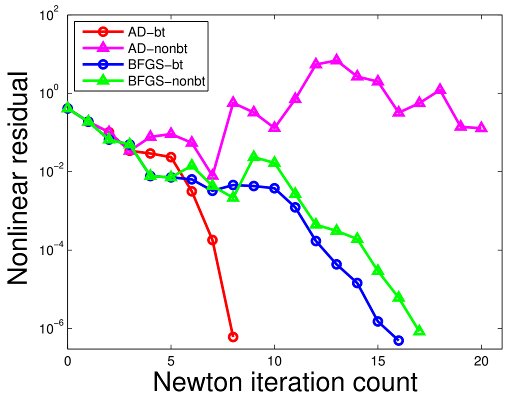

FIG. 4.10. *Earth-to-Mars orbit transfer problem. History of the nonlinear residual norms for AD and BFGS with and without the backtracking technique.*

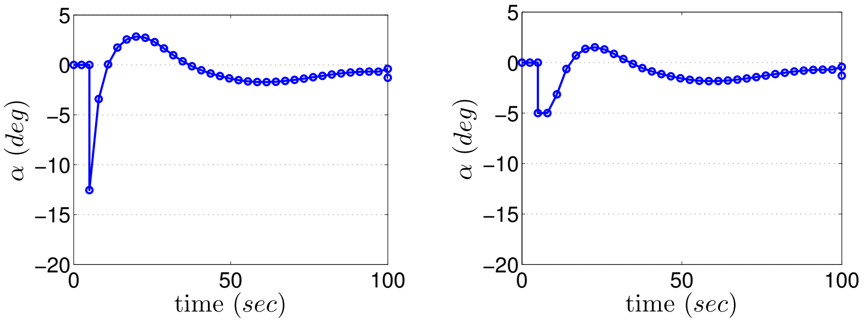

FIG. 4.11. *A comparison of histories of the angle of attack before (left) and after (right) inequality constraints imposed for the three-stage launch vehicle problem.*