ELSEVIER

Journal of Computational and Applied Mathematics 120 (2000) 197–213

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

www.elsevier.nl/locate/cam

An SQP method for the optimal control of large-scale dynamical systems ☆

Philip E. Gill[a], Laurent O. Jay[b], Michael W. Leonard[c], Linda R. Petzold[d, *], Vivek Sharma[d]

[a] *Department of Mathematics, University of California, San Diego, La Jolla, CA 92093-0112, USA*
[b] *Department of Mathematics, The University of Iowa, Iowa City, IA 52242-1419, USA*
[c] *Department of Mathematics, University of California, Los Angeles, CA 90095-1555, USA*
[d] *Department of Computer Science, and Department of Mechanical and Environmental Engineering, University of California, Santa Barbara, CA 93106-5070, USA*

Received 6 July 1998; received in revised form 11 March 1999

Abstract

We propose a sequential quadratic programming (SQP) method for the optimal control of large-scale dynamical systems. The method uses modified multiple shooting to discretize the dynamical constraints. When these systems have relatively few parameters, the computational complexity of the modified method is much less than that of standard multiple shooting. Moreover, the proposed method is demonstrably more robust than single shooting. In the context of the SQP method, the use of modified multiple shooting involves a transformation of the constraint Jacobian. The affected rows are those associated with the continuity constraints and any path constraints applied within the shooting intervals. Path constraints enforced at the shooting points (and other constraints involving only discretized states) are not transformed. The transformation is cast almost entirely at the user level and requires minimal changes to the optimization software. We show that the modified quadratic subproblem yields a descent direction for the ℓ_1 penalty function. Numerical experiments verify the efficiency of the modified method.

Keywords: Multiple shooting; Optimal control; Sequential quadratic programming

☆ This research was partially supported by National Science Foundation grants CCR-98-96198 and DMI-9424639 and NSF Cooperative Agreement DMS-9615858, National Institute of Standards and Technology contract 60 NANB2D 1272, Department of Energy grant FG03-98ER25354, with computational resources from the NSF San Diego Supercomputer Center and DOE NERSC.

* Corresponding author.
E-mail address: petzold@engineering.ucsb.edu (L.R. Petzold).

0377-0427/00/$ - see front matter
PII: S0377-0427(00)00310-1

1. Introduction

We consider the ordinary differential equation (ODE) system

$$y' = F(t, y, p, u(t)), \quad y(t_0) = y_0,$$

where the control parameters p and the vector-valued control function $u(t)$ must be determined such that the objective function

$$\int_{t_0}^{t_{\max}} \Psi(t, y(t), p, u(t))\,\mathrm{d}t \quad \text{is minimized}$$

and some additional inequality constraints

$$G(t, y(t), p, u(t)) \geqslant 0$$

are satisfied. The optimal control function $u^*(t)$ is assumed here to be continuous. In many applications the ODE system is large scale. Thus, the dimension n_y of y is large. Often, for example, the ODE system arises from the spatial discretization of a time-dependent partial differential equation (PDE) system. In many such problems, the dimensions of the control parameters and of the representation of the control function $u(t)$ are much smaller. To represent $u(t)$ in a low-dimensional vector space, we use piecewise polynomials on $[t_0, t_{\max}]$, their coefficients being determined by the optimization. For ease of presentation we can therefore assume that the vector p contains both the parameters and these coefficients (we let n_p denote the combined number of these values) and discard the control function $u(t)$ in the remainder of the paper. Hence we consider

$$y' = F(t, y, p), \quad y(t_0) = y_0, \tag{1a}$$

$$\int_{t_0}^{t_{\max}} \psi(t, y(t), p)\,\mathrm{d}t \quad \text{is minimized}, \tag{1b}$$

$$g(t, y(t), p) \geqslant 0. \tag{1c}$$

There are a number of well-known methods for direct discretization of this optimal control problem (1). The *single shooting method* solves the ODEs (1a) over the interval $[t_0, t_{\max}]$, with the set of controls generated at each iteration by the optimization algorithm. However, it is well-known that single shooting can suffer from a lack of stability and robustness [1]. Moreover, for this method it is more difficult to maintain additional constraints and to ensure that the iterates are physical or computable. The *finite-difference method* or *collocation method* discretizes the ODEs over the interval $[t_0, t_{\max}]$ with the ODE solutions at each discrete time and the set of controls generated at each iteration by the optimization algorithm. Although this method is more robust and stable than the single-shooting method, it requires the solution of an optimization problem which for a large-scale ODE system is enormous, and it does not allow for the use of adaptive ODE or (in the case that the ODE system is the result of semi-discretization of PDEs) PDE software.

We thus consider the *multiple-shooting method* for the discretization of (1). In this method, the time interval $[t_0, t_{\max}]$ is divided into subintervals $[t_i, t_{i+1}]$ $(i = 0, \ldots, N-1)$, and the differential equations (1a) are solved over each subinterval, where additional intermediate variables y_i are introduced. On each subinterval we denote the solution at time t of (1a) with initial value y_i at t_i by $y(t, t_i, y_i, p)$. Continuity between subintervals is achieved via the continuity constraints

$$C_1^{i+1}(y_i, y_{i+1}, p) := y_{i+1} - y(t_{i+1}, t_i, y_i, p) = 0.$$

The additional constraints (1c) are required to be satisfied at the boundaries of the shooting intervals

$$C_3^i(y_i, p) := g(t_i, y_i, p) \geqslant 0, \quad C_3^N(y_N, p) := g(t_N, y_N, p) \geqslant 0,$$

and also at a finite number of intermediate times t_{ik} within each subinterval $[t_i, t_{i+1}]$

$$C_2^{ik}(y_i, p) := g(t_{ik}, y(t_{ik}, t_i, y_i, p), p) \geqslant 0.$$

Following common practice, we write

$$\Phi(t) = \int_{t_0}^{t} \psi(\tau, y(\tau), p)\, \mathrm{d}\tau,$$

which satisfies $\Phi'(t) = \psi(t, y(t), p)$, $\Phi(t_0) = 0$. This introduces another equation and variable into the differential system (1a). The discretized optimal control problem becomes

$$\underset{y_1,\ldots,y_N,p}{\text{minimize}}\ \Phi(t_{\max}) \tag{2}$$

subject to the constraints

$$C_1^{i+1}(y_i, y_{i+1}, p) = 0, \tag{3a}$$

$$C_2^{ik}(y_i, p) \geqslant 0, \tag{3b}$$

$$C_3^i(y_i, p) \geqslant 0 \quad \text{and} \quad C_3^N(y_N, p) \geqslant 0. \tag{3c}$$

This problem can be solved by an optimization code. We use the solver SNOPT [5], which incorporates a sequential quadratic programming (SQP) method. The SQP methods require a gradient and Jacobian matrix that are the derivatives of the objective function and constraints with respect to the optimization variables. We compute these derivatives via differential-algebraic equation (DAE) sensitivity software DASPKSO [8]. Our basic algorithm and software for the optimal control of dynamical systems are described in detail in [9].

This basic multiple-shooting type of strategy can work very well for small-to-moderate size ODE systems, and has an additional advantage that it is inherently parallel. However, for large-scale ODE systems there is a problem because the computational complexity grows rapidly with the dimension of the ODE system. The difficulty lies in the computation of the derivatives of the continuity constraints with respect to the variables y_i. The solution of $\mathcal{O}(n_y)$ sensitivity systems is required to form the derivative matrix $\partial y(t)/\partial y_i$ for the multiple-shooting method. For the problems under consideration n_y can be very large (for example, for an ODE system obtained from the semi-discretization of a PDE system, n_y is the dimension of the semi-discretized PDE system). In contrast, the single-shooting method requires the solution of $\mathcal{O}(n_p)$ sensitivity systems, although the method is not as stable, robust or parallelizable.

The basic idea for reducing the computational complexity of the multiple shooting method for this type of problem is to make use of the structure of the continuity constraints to reduce the number of sensitivity solutions which are needed to compute the derivatives. To do this, we recast the continuity constraints in a form where only the matrix-vector products $(\partial y(t)/\partial y_i)w_j$ are needed, rather than the entire matrix $\partial y(t)/\partial y_i$. The matrix-vector products are directional derivatives; each can be computed via a single sensitivity analysis. The number of vectors w_j such that the directional sensitivities are needed is small, of order $\mathcal{O}(n_p)$. Thus, the computational work of the modified multiple shooting computation is reduced to $\mathcal{O}(n_p)$ sensitivity solves, roughly the same as that of single shooting.

Unfortunately, the reduction in computational complexity comes at a price: the stability of the modified multiple shooting algorithm suffers from similar limitations as single shooting. However, for many dissipative PDE systems this is not an issue, and the modified method is more robust for nonlinear problems.

There are other considerations in addition to complexity. There may be many inequality constraints in this type of optimal control problem. Any scheme for reducing the complexity of the derivative calculations for the continuity constraints should not cause additional complexity in forming or computing the derivatives of the inequality constraints. Whatever changes are made to the optimization problem or the optimization method must result in an algorithm where the merit function is decreasing. Finally, an optimization code such as SNOPT is highly complex. There is a strong motivation to be able to adapt such an optimization code to our optimal control algorithm with a minimum of changes to the optimizer. Our aim is not only to reduce the difficulties associated with writing and maintaining separate optimization software for the dynamical systems problems, but also to make it easier to adapt new optimization software and algorithms for these problems.

Although a number of papers have discussed algorithms related to the one proposed here, to our knowledge, none has all of the desirable properties mentioned above. The stabilized march method [1] for 2-point boundary value problems is closely related. The stabilized march method makes use of the structure of the continuity constraints by solving them for the internal variables (at the multiple shooting points) in terms of the unknown boundary conditions. This can be done efficiently through the use of directional derivatives. Schlöder [12] generalizes this method to multipoint boundary value problems from optimal control and parameter estimation. However, the method is not easily extended to general inequality constraints, mainly because it is a problem to write these inequality constraints in terms of the parameters in the optimization. Biegler et al. [2] present a reduced SQP method used with collocation, which reduces the size of the optimization problem by solving the constraints from the discretized ODE for the discretized state variables in terms of the optimization parameters. Schultz [13] introduces partially reduced SQP methods used with collocation and multiple shooting to overcome this limitation with respect to the inequality constraints. In the partially reduced SQP methods, the inequality constraints are reduced on the kernel of the continuity constraints. The method appears to require substantial modification to existing optimization solvers. Steinbach et al. [14] make use of the partially reduced SQP methods for mathematical optimization in robotics; the inequality constraints are treated via slack variables.

The remainder of the paper is organized as follows. In Section 2 we present an SQP formulation of the discretized optimal control problem (2)–(3). This leads to a discussion in Section 3 of the SQP Jacobian and our proposed modification of its structure. In Section 4, we discuss the resulting modified QP subproblem as well our choice of merit function for use with the altered QP. In Section 5 we conclude with numerical results that demonstrate the effectiveness of the proposed method.

2. Optimization problem for dynamical systems

The optimization problem (2)–(3) for dynamical systems can be rewritten in a more compact form. The variable N_y is used to denote the number of discretized states ($N_y = (N+1)(n_y+1)$). We let $x = (y, p)^{\mathrm{T}}$ denote the optimization vector in terms of the N_y discretized states and the parameters

(including discretized controls). We let $c_1(x) \in \mathbb{R}^{N_y}$ denote the vector of continuity constraints, i.e.,

$$c_1(x) = (C_1^1(y_0, y_1, p),\ C_1^2(y_1, y_2, p), \ldots, C_1^N(y_{N-1}, y_N, p))^{\mathrm{T}}$$

(here and throughout the paper we use the simpler notation $(a, b)^{\mathrm{T}}$ to denote the column vector $(a^{\mathrm{T}}\ b^{\mathrm{T}})^{\mathrm{T}}$). The vectors of inequality constraints c_2 and c_3 are defined in a similar manner in terms of their capitalized counterparts (c_3 can include *any* constraints that involve only discretized states). The objective function $\Phi(t_{\max})$ is denoted simply by $f(x)$. The problem now takes the form

$$\underset{x}{\text{minimize}}\ f(x), \quad c_1(x) = 0, \quad c_2(x) \geqslant 0, \quad c_3(x) \geqslant 0. \tag{4}$$

We use an SQP method to solve this optimization problem. In SQP, a sequence of iterates (x_k, π_k) is generated converging to a point (x^*, π^*) satisfying the first-order Karush–Kuhn–Tucker (KKT) conditions of optimality. Each iterate is the result of a "major" iteration that involves computing the solution $(\hat{x_k}, \hat{\pi}_k)$ of a QP subproblem, performing a *line search* (discussed in Section 4.2) and updating the QP Hessian H_k. The QP is derived from problem (4) and is written

$$\underset{x}{\text{minimize}} \quad f(x_k) + \nabla f(x_k)^{\mathrm{T}}(x - x_k) + \tfrac{1}{2}(x - x_k)^{\mathrm{T}} H_k (x - x_k), \tag{5a}$$

$$c_1(x_k) + J_1(x_k)(x - x_k) = 0, \tag{5b}$$

$$c_i(x_k) + J_i(x_k)(x - x_k) \geqslant 0, \quad i = 2, 3. \tag{5c}$$

The gradient $\nabla f(x)$ is a unit vector in our formulation because the objective function is the value of a state at $t_{\max}$ (see (2) and the preceding discussion). The matrices $J_i(x)$, $i = 1, 2, 3$, each form a block of the Jacobian matrix $J(x)$ defined by $J(x) = \partial c(x)/\partial x$, where $c(x) := (c_1(x), c_2(x), c_3(x))^{\mathrm{T}}$. (For example, $J_1(x) = \partial c_1(x)/\partial x$.) The matrix H_k is a positive-definite approximation to $\nabla^2 L^k(x_k, \pi_k)$, where $L^k(x, \pi_k)$ is the modified Lagrangian function

$$L^k(x, \pi_k) := f(x) - \pi_k^{\mathrm{T}}(c(x) - c^k(x)),$$

and $c^k(x)$ is the vector of *linearized constraints* $c^k(x) := c(x_k) + J(x_k)(x - x_k)$.

The optimality conditions for a solution $\hat{x}$ of the QP subproblem imply the existence of vectors $\hat{s} := (\hat{s}_1, \hat{s}_2, \hat{s}_3)^{\mathrm{T}}$ and $\hat{\pi} := (\hat{\pi}_1, \hat{\pi}_2, \hat{\pi}_3)^{\mathrm{T}}$ such that

$$\begin{aligned}
&\nabla f(x_k) + H_k(\hat{x} - x_k) = J(x_k)^{\mathrm{T}}\hat{\pi}, \quad \hat{\pi}_i \geqslant 0, \quad i = 2, 3,\\
&c(x_k) + J(x_k)(\hat{x} - x_k) = \hat{s}, \quad \hat{s}_1 = 0,\\
&\hat{\pi}^{\mathrm{T}}\hat{s} = 0, \quad \hat{s}_i \geqslant 0, \quad i = 2, 3.
\end{aligned} \tag{6}$$

The components of $\hat{\pi}$ are the Lagrange multipliers of the subproblem, and are referred to as the QP multipliers. Each component of $\hat{s}$ can be regarded as a slack variable for the associated constraint in c. In an active-set QP method, values $(\hat{x}, \hat{\pi}, \hat{s})$ satisfying (6) are determined using a sequence of "minor" iterations, each one of which solves an equality-constrained QP where certain of the linearized constraints are treated as "active" (equal to 0).

3. Structure and modification of the linearized constraints

Since the complexity problem for the basic multiple shooting method results mainly from computation of the derivative matrix J_1 of the continuity constraints, we first examine the structure of this matrix.

Linearizing the continuity constraints

$$C_1^{i+1}(y_i + \Delta y_i, y_{i+1} + \Delta y_{i+1}, p + \Delta p) = 0$$

at $(y_0, y_1, \ldots, y_N, p)$ (with $\Delta y_0 = 0$), we obtain

$$\Delta y_{i+1} - \frac{\partial y(t_{i+1})}{\partial p}\Delta p - \frac{\partial y(t_{i+1})}{\partial y_i}\Delta y_i + C_1^{i+1}(y_i, y_{i+1}, p) = 0,$$

where to simplify the notation $y(t_{i+1})$ stands for $y(t_{i+1}, t_i, y_i, p)$. In matrix notation we have

$$(J_{1y}\ \ J_{1p})\begin{pmatrix}\Delta y\\ \Delta p\end{pmatrix} + c_1 = 0, \tag{7}$$

where $\Delta y = (\Delta y_1, \Delta y_2, \ldots, \Delta y_N)^{\mathrm{T}}$. The matrices J_{1y} and J_{1p} are $\partial c_1(x)/\partial y$ and $\partial c_1(x)/\partial p$ (so $J_1 = (J_{1y}\ \ J_{1p})$) and satisfy explicitly

$$J_{1y} = \begin{pmatrix} I & O & O & \ldots & O\\ -\dfrac{\partial y(t_2)}{\partial y_1} & I & O & \ldots & O\\ O & -\dfrac{\partial y(t_3)}{\partial y_2} & I & \ldots & O\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ O & O & \ldots & -\dfrac{\partial y(t_N)}{\partial y_{N-1}} & I \end{pmatrix}, \quad J_{1p} = \begin{pmatrix} -\dfrac{\partial y(t_1)}{\partial p}\\ -\dfrac{\partial y(t_2)}{\partial p}\\ \vdots\\ -\dfrac{\partial y(t_N)}{\partial p}\end{pmatrix}$$

(we have temporarily dropped explicit reference to x_k).

The sensitivity matrices $\partial y(t)/\partial y_i$ are very costly to compute since the dimension of y is large. The jth column of this matrix is given by the solutions on the interval $[t_i, t_{i+1}]$ of the equations

$$s'_{ij} = \frac{\partial F}{\partial y}(t, y, p)s_{ij}, \quad s_{ij}(t_i) = e_j, \tag{8}$$

where e_j is the jth column of the identity. As an aside, we note that these equations can be evaluated using automatic differentiation tools (see [3]), and solved via ODE or DAE sensitivity analysis software (see [8]). They can also be evaluated using finite differences

$$s'_{ij} = \frac{1}{\delta_{ij}}(F(t, y + \delta_{ij}s_{ij}, p) - F(t, y, p)),$$

where δ_{ij} is a small scalar. In the numerical example that we will consider, the matrix $\partial F(t, y, p)/\partial y$ is sparse and the products $(\partial F(t, y, p)/\partial y)s_{ij}$ can be computed directly. In any case, each sensitivity matrix requires n_y "sensitivities", which implies that J_{1y} takes $(N-1)n_y$ sensitivities. Each column of $\partial y(t)/\partial p$ is defined similarly to the sensitivity (8) (see Maly and Petzold [8]) and takes approximately the same amount of work. It follows that J_1 requires approximately $N(n_y + n_p)$ sensitivities.

The matrix J_{1y}^{-1} can be used to transform Eq. (7) so that the number of sensitivities is reduced. Multiplying Eq. (7) by J_{1y}^{-1} gives the equivalent system

$$(I \quad J_{1y}^{-1}J_{1p})\begin{pmatrix}\Delta y\\ \Delta p\end{pmatrix} + J_{1y}^{-1}c_1 = 0, \tag{9}$$

which involves the "modified" Jacobian $(I \; J_{1y}^{-1}J_{1p})$. The important feature of this matrix is that the sensitivities associated with the identity block are available free of charge. The second block is the solution of the matrix system $J_{1y}X = J_{1p}$. A short inductive argument proves that the computation of X requires $n_p(2N-1)$ sensitivities. To make this argument, we partition X vertically into N blocks each denoted by X_i. Clearly, $X_1 = -\partial y(t_1)/\partial p$, which requires n_p sensitivities. Assume now that X_i has been computed. We find that

$$X_{i+1} = \frac{\partial y(t_{i+1})}{\partial y_i}X_i - \frac{\partial y(t_{i+1})}{\partial p},$$

which requires for $1 \leqslant i \leqslant N-1$ a total of $2n_p(N-1)$ sensitivities (noting that the matrix–matrix product $(\partial y(t_{i+1})/\partial y_i)X_i$ can be computed directly via n_p sensitivities). Adding n_p to this gives the desired result, which completes the argument. The modified system (9) also requires $N-1$ sensitivities for $J_{1y}^{-1}c_1$. Hence, the total number needed is approximately $N(2n_p+1)$, which can be substantially less than $N(n_y+n_p)$ when $n_p \ll n_y$. (We should note that the actual numbers of sensitivities for both systems (7) and (9) is less than we have written here because the controls have been included in p.)

Next we examine the structure of the path constraints C_2^{ik} $(1 \leqslant k \leqslant K_i)$ because these too can lead to a large number of sensitivity calculations. Linearizing these constraints leads to a matrix system

$$(J_{2y} \quad J_{2p})\begin{pmatrix}\Delta y\\ \Delta p\end{pmatrix} + c_2 \geqslant 0, \tag{10}$$

where J_{2y} and J_{2p} have the forms

$$J_{2y} = \begin{pmatrix} O & O & O & \dots & O\\ B_{1y} & O & O & \dots & O\\ O & B_{2y} & O & \dots & O\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ O & O & \dots & B_{(N-1)y} & O\end{pmatrix} \quad \text{and} \quad J_{2p} = \begin{pmatrix} B_{0p}\\ B_{1p}\\ \vdots\\ B_{(N-1)p}\end{pmatrix}. \tag{11}$$

The matrices B_{iy} and B_{ip} satisfy

$$B_{iy} = \begin{pmatrix} \dfrac{\partial g(t_{i1})}{\partial y(t_{i1})}\dfrac{\partial y(t_{i1})}{\partial y_i}\\ \dfrac{\partial g(t_{i2})}{\partial y(t_{i2})}\dfrac{\partial y(t_{i2})}{\partial y_i}\\ \vdots\\ \dfrac{\partial g(t_{iK_i})}{\partial y(t_{iK_i})}\dfrac{\partial y(t_{iK_i})}{\partial y_i}\end{pmatrix} \quad \text{and} \quad B_{ip} = \begin{pmatrix} \dfrac{\partial g(t_{i1})}{\partial y(t_{i1})}\dfrac{\partial y(t_{i1})}{\partial p} + \dfrac{\partial g(t_{i1})}{\partial p}\\ \dfrac{\partial g(t_{i2})}{\partial y(t_{i2})}\dfrac{\partial y(t_{i2})}{\partial p} + \dfrac{\partial g(t_{i2})}{\partial p}\\ \vdots\\ \dfrac{\partial g(t_{iK_i})}{\partial y(t_{iK_i})}\dfrac{\partial y(t_{iK_i})}{\partial p} + \dfrac{\partial g(t_{iK_i})}{\partial p}\end{pmatrix},$$

and require at most n_y and n_p sensitivities respectively (a *whole* sensitivity is associated with integration across the entire interval $[t_i, t_{i+1}]$). It follows that J_2 requires about $N(n_y + n_p)$ sensitivities, the same number needed for J_1.

The structure of J_2 can also be modified so that the number of sensitivities is reduced. However, we cannot use the same technique we used to modify J_1. Instead, we solve the continuity equations (7) for Δy and substitute the result into the path constraint system (10). This gives the matrix system

$$(O \;\; J_{2p} - J_{2y}J_{1y}^{-1}J_{1p}) \begin{pmatrix} \Delta y \\ \Delta p \end{pmatrix} - J_{2y}J_{1y}^{-1}c_1 + c_2 \geqslant 0. \tag{12}$$

Since $J_{1y}^{-1}J_{1p}$ and $J_{1y}^{-1}c_1$ are already computed, this system requires approximately $N(2n_p + 1)$ sensitivities (the same number as the modified continuity constraint system). Hence, we again obtain a savings when $n_p \ll n_y$.

Two comments are in order before we conclude this section. First, the substitution of Δy in terms of Δp in the path constraint equations (10) *does not* eliminate y as an optimization variable, since it still appears in the modified continuity constraints. Second, the linearized constraints

$$c_3(x_k) + J_3(x_k)(x - x_k) \geqslant 0, \tag{13}$$

involving only discretized states, are left unmodified since J_3 involves no sensitivities.

4. Modified QP subproblem

In Section 4.1 we will reformulate the QP subproblem in terms of the modified constraints (9) and (12). This leads to a complication in the line search, which we discuss in Section 4.2.

4.1. Reformulation of QP subproblem

The objective function for the modified QP is the same as before (see Eq. (5a)). In terms of a transformation matrix

$$M(x) := \begin{pmatrix} J_{1y}^{-1} & O & O \\ -J_{2y}J_{1y}^{-1} & I & O \\ O & O & I \end{pmatrix} \tag{14}$$

and transformed quantities $\bar{c}(x) = M(x)c(x)$ and $\bar{J}(x) = M(x)J(x)$, the constraints (9), (12) and (13) can be written more simply as

$$\begin{aligned} &\bar{c}_1(x_k) + \bar{J}_1(x_k)(x - x_k) = 0, \\ &\bar{c}_i(x_k) + \bar{J}_i(x_k)(x - x_k) \geqslant 0, \quad i = 2, 3. \end{aligned}$$

The optimality conditions for the modified QP subproblem are

$$\begin{aligned} &\nabla f(x_k) + H_k(\bar{x} - x_k) = \bar{J}(x_k)^{\mathrm{T}}\bar{\pi}, \quad \bar{\pi}_i \geqslant 0, \quad i = 2, 3, \\ &\bar{c}(x_k) + \bar{J}(x_k)(\bar{x} - x_k) = \bar{s}, \quad \bar{s}_1 = 0, \\ &\bar{\pi}^{\mathrm{T}}\bar{s} = 0, \quad \bar{s}_i \geqslant 0, \quad i = 2, 3. \end{aligned} \tag{15}$$

The next lemma shows that transformation by M does not fundamentally alter the solution of a given subproblem.

Lemma 1. *Consider a solution* $(\bar{x},\bar{s},\bar{\pi})$ *of the modified QP* (15). *Define vectors* $\hat{s}$ *and* $\hat{\pi}$ *such that* $\hat{\pi}=M(x_k)^{\mathrm{T}}\bar{\pi}$ *and* $\hat{s}=M(x_k)^{-1}\bar{s}$, *with* $M(x_k)$ *the transformation matrix* (14). *Then* $(\bar{x},\hat{s},\hat{\pi})$ *satisfies the conditions* (6) *and is therefore a solution of the unmodified QP* (5).

Proof. From the definition (14) of M we have

$$M(x)^{-1}=\begin{pmatrix} J_{1y} & O & O \\ J_{2y} & I & O \\ O & O & I \end{pmatrix}. \tag{16}$$

Forming the products $\hat{s}=M(x_k)^{-1}\bar{s}$ and $\hat{\pi}=M(x_k)^{\mathrm{T}}\bar{\pi}$, gives $\hat{s}=\bar{s}$ and $\hat{\pi}_i=\bar{\pi}_i$ for $i=2,3$.

It remains to show that $\bar{x}$ is feasible for (5). Multiplying the constraint equations in (15) by $M(x_k)^{-1}$ gives

$$c(x_k)+J(x_k)(\bar{x}-x_k)=M(x_k)^{-1}\bar{s}=\hat{s}=\bar{s}, \tag{17}$$

and the result follows from the optimality of $\bar{s}$ in (15). □

An SQP code capable of solving the problem (4) is necessarily complex. However, altering the code to solve the modified QP instead of the original QP is a simple matter of providing $\bar{c}$ and $\bar{J}$ in place of c and J. The only complication is that $\bar{J}$ is *not the Jacobian of* $\bar{c}$. In fact, the Jacobian of $\bar{c}$ is

$$M(x)J(x)+\frac{\partial M(x)}{\partial x}c(x)=\bar{J}(x)+\frac{\partial M(x)}{\partial x}c(x),$$

which means we are omitting the term $(\partial M(x)/\partial x)c(x)$. This omission has an effect on the choice of *merit function*, as we discuss in the next section.

4.2. The merit function

An important feature of SQP methods is that a merit function is used to force convergence from an arbitrary starting point. The properties of a merit function may be discussed with respect to a generic function $\mathscr{M}$ defined in terms of variables y. In general, y may include any of the variables appearing in the QP subproblem, including the slacks and dual variables. If $\hat{y}_k$ computed from the QP subproblem is an estimate of the solution y^*, a line search is used to find a scalar α_k $(0<\alpha_k\leqslant 1)$ that gives a *sufficient decrease* in $\mathscr{M}$, i.e.,

$$\mathscr{M}(y_k+\alpha_k\Delta y_k)<\mathscr{M}(y_k)-\alpha_k r(y_k), \tag{18}$$

where $\Delta y_k=\hat{y}_k-y_k$ and $r(y)$ is a positive function such that $r(y)\to 0$ only if $y\to y^*$. If the line search is to be successful, Δy_k must be a *direction of decrease* for $\mathscr{M}(y)$, i.e., there must exist a σ $(0<\sigma\leqslant 1)$ such that the sufficient decrease criterion (18) is satisfied for all $\alpha\in(0,\sigma)$.

In the method of SNOPT, y_k consists of the QP variables (x_k,s_k,π_k) and the merit function is the augmented Lagrangian function

$$\mathscr{M}(x,\pi,s,\rho)=f(x)-\pi^{\mathrm{T}}(c(x)-s)+\tfrac{1}{2}\rho\|c(x)-s\|_2^2, \tag{19}$$

where ρ is a nonnegative scalar penalty parameter. In this case, ρ is chosen at the start of the line search to ensure that $(\Delta x_k, \Delta\pi_k, \Delta s_k)$ is a direction of decrease for $\mathscr{M}$. (For more information see, e.g., [11,6].) The augmented Lagrangian is continuously differentiable, which allows α to be found using safeguarded polynomial interpolation. These methods use $\mathscr{M}$ to define a smooth function that has a minimizer satisfying (18). Safeguarded quadratic or cubic interpolation may then be used to generate a sequence (starting with $\alpha = 1$) that converges to this minimizer. The minimizing sequence is terminated at the first value α that satisfies the sufficient decrease criterion (18). This procedure is very efficient, with only one or two function evaluations being required to improve the merit function, even when far from the solution.

However, the multiplier vector π is not computed when solving the modified QP, and it follows that the augmented Lagrangian merit function (19) cannot be used in this situation. As an alternative, we use a merit function based on the "exact" or ℓ_1 penalty function (see, e.g., [4]). Let $v(x)$ denote the vector of constraint violations at any point x; i.e., $v_i(x) = \max[0, -c_i(x)]$ for an inequality constraint $c_i(x) \geqslant 0$, and $v_i(x) = |c_i(x)|$ for an equality constraint $c_i(x) = 0$. The ℓ_1 penalty function is given by

$$\mathscr{M}(x) = f(x) + \rho\|v(x)\|_1,$$

where ρ is a nonnegative penalty parameter. (For simplicity, our notation for $\mathscr{M}$ suppresses the dependence on ρ.) The main property of the ℓ_1 penalty function is that there exists a nonnegative ρ^* such that, for all $\rho > \rho^*$, a solution of the original problem (4) is also a local minimizer of $\mathscr{M}(x)$.

The function $\mathscr{M}(x)$ is not differentiable and therefore cannot be minimized efficiently using smooth polynomial interpolation. We use the popular alternative of a *backtracking line search* (see, e.g., [7, pp. 100–102]). This line search determines a step α_k for which the reduction in the merit function is no worse than a factor μ $(0 < \mu < \frac{1}{2})$ of the reduction predicted by a model function based on a linear approximation of f and c. The particular line-search model used is

$$\mathscr{M}^k(x) = f(x_k) + \nabla f(x_k)^{\mathrm{T}}(x - x_k) + \rho\|v^k(x)\|_1, \tag{20}$$

where $v^k(x)$ denotes the violations of the linearized constraints $c^k(x) := c(x_k) + J(x_k)(x - x_k)$.

Let ω be any constant in the range $0 < \omega < 1$ (often, $\omega = \frac{1}{2}$). The new SQP iterate is defined as $x_{k+1} = x_k + \alpha_k \Delta x_k$, where α_k is the first member of the sequence $\alpha^0 = 1$, $\alpha^j = \omega\alpha^{j-1}$ such that

$$\mathscr{M}(x_k + \alpha_k \Delta x_k) \leqslant \mathscr{M}(x_k) - \mu(\mathscr{M}^k(x_k) - \mathscr{M}^k(x_k + \alpha_k \Delta x_k)) \tag{21}$$

(cf. (18)). A standard result states that an interval of acceptable step lengths exists provided $\rho \geqslant \|\hat{\pi}\|_\infty$, where $\hat{\pi}$ are the QP multipliers of (6) (see, e.g., [10]). It remains to show that an appropriate bound on ρ can be calculated from quantities defined by the *modified* QP subproblem.

Theorem 2. *Let Δx_k be a nonzero search direction such that $\Delta x_k = \bar{x} - x_k$, where $\bar{x}$ satisfies the conditions* (15). *Let $\hat{\rho}$ denote the penalty value*

$$\hat{\rho} = \max[0, -\bar{c}(x_k)^{\mathrm{T}}\bar{\pi}/\|v(x_k)\|_1].$$

Then for all $\rho \geqslant \hat{\rho}$, there exists a positive σ such that the line search condition (21) *is satisfied for all $\alpha \in (0, \sigma)$.*

Proof. First, we show that for ρ sufficiently large and all $0 < \alpha \leqslant 1$, the predicted reduction in $\mathscr{M}$ is positive, i.e., $\mathscr{M}^k(x_k) - \mathscr{M}^k(x_k + \alpha\Delta x_k) > 0$. Substituting directly from the definition (20) of the model function yields

$$\mathscr{M}^k(x_k) - \mathscr{M}^k(x_k + \alpha\Delta x_k) = -\alpha\nabla f(x_k)^{\mathrm{T}}\Delta x_k + \rho(\|v^k(x_k)\|_1 - \|v^k(x_k + \alpha\Delta x_k)\|_1).$$

We derive a lower bound on $\|v^k(x_k)\|_1 - \|v^k(x_k+\alpha\Delta x_k)\|_1$ using the properties of the slack variables. For any nonnegative slack vector s we have $\|v(x_k)\|_1 \leqslant \|c(x_k) - s\|_1$, with equality for the vector s_0 with components $(s_0)_i = \max[0, c_i(x_k)]$ for a constraint $c_i(x) \geqslant 0$, and $(s_0)_i = 0$ for a constraint $c_i(x) = 0$. Consider the vector $\Delta s_k := \bar{s} - s_0$, where $\bar{s}$ is the slack vector (15) computed by the QP. The optimality conditions (15) for the modified QP imply that $\bar{s}$ is nonnegative, which allows us to assert that $s_0 + \alpha\Delta s_k \geqslant 0$ for all $0 \leqslant \alpha \leqslant 1$. It follows that

$$\begin{aligned}\|v^k(x_k + \alpha\Delta x_k)\|_1 &\leqslant \|c^k(x_k + \alpha\Delta x_k) - (s_0 + \alpha\Delta s_k)\|_1 \\ &= \|c^k(x_k) + \alpha J(x_k)\Delta x_k - (s_0 + \alpha\Delta s_k)\|_1.\end{aligned}$$

Using the identity (17) and the fact that $c^k(x_k) = c(x_k)$, we have

$$\|v^k(x_k + \alpha\Delta x_k)\|_1 \leqslant (1 - \alpha)\|c(x_k) - s_0\|_1$$

for all α such that $0 < \alpha \leqslant 1$. This inequality leads directly to the bound

$$\begin{aligned}\|v^k(x_k)\|_1 - \|v^k(x_k + \alpha\Delta x_k)\|_1 &\geqslant \|c(x_k) - s_0\|_1 - (1 - \alpha)\|c(x_k) - s_0\|_1 \\ &= \alpha\|c(x_k) - s_0\|_1 = \alpha\|v(x_k)\|_1. \end{aligned} \tag{22}$$

Finally, from (15), we have

$$-\nabla f(x_k)^{\mathrm{T}}\Delta x_k = \Delta x_k^{\mathrm{T}} H_k \Delta x_k - \Delta x_k^{\mathrm{T}} \bar{J}(x_k)^{\mathrm{T}}\bar{\pi} = \Delta x_k^{\mathrm{T}} H_k \Delta x_k + (\bar{c}(x_k) - \bar{s})^{\mathrm{T}}\bar{\pi},$$

which may be simplified to become

$$-\nabla f(x_k)^{\mathrm{T}}\Delta x_k = \Delta x_k^{\mathrm{T}} H_k \Delta x_k + \bar{c}(x_k)^{\mathrm{T}}\bar{\pi} \tag{23}$$

using the optimality condition $\bar{s}^{\mathrm{T}}\bar{\pi} = 0$. Eqs. (22) and (23) allow us to write the reduction in $\mathscr{M}^k$ as

$$\mathscr{M}^k(x_k) - \mathscr{M}^k(x_k + \alpha\Delta x_k) \geqslant \alpha(\Delta x_k^{\mathrm{T}} H_k \Delta x_k + \bar{c}(x_k)^{\mathrm{T}}\bar{\pi} + \rho\|v(x_k)\|_1). \tag{24}$$

As H_k is positive definite by assumption, we need only consider the term $\bar{c}(x_k)^{\mathrm{T}}\bar{\pi} + \rho\|v(x_k)\|_1$. If $\bar{c}(x_k)^{\mathrm{T}}\bar{\pi} \geqslant 0$, we may choose $\hat{\rho} = 0$. Otherwise, if $v(x_k) \neq 0$, it is sufficient to choose $\hat{\rho} = |\bar{c}(x_k)^{\mathrm{T}}\bar{\pi}|/\|v(x_k)\|_1$. If $v(x_k) = 0$, it follows that $c_1(x_k) = 0$, $c_2(x_k) \geqslant 0$ and $c_3(x_k) \geqslant 0$. These combined with the definition of $\bar{c}(x_k)$ and the nonnegativity of $\bar{\pi}_2$ and $\bar{\pi}_3$ imply that $\bar{c}(x_k)^{\mathrm{T}}\bar{\pi} \geqslant 0$, so we again can choose $\hat{\rho} = 0$.

Next we show that

$$\lim_{\alpha \to 0^+} \frac{\mathscr{M}(x_k) - \mathscr{M}(x_k + \alpha\Delta x_k)}{\mathscr{M}^k(x_k) - \mathscr{M}^k(x_k + \alpha\Delta x_k)} = 1. \tag{25}$$

Since $\mu < 1$, this implies that there must exist a positive σ such that (21) is satisfied for all $\alpha \in (0, \sigma)$.

The expression

$$\frac{1}{\alpha}(\mathscr{M}(x_k) - \mathscr{M}(x_k + \alpha\Delta x_k)) \tag{26}$$

can be written as

$$\frac{1}{\alpha}(f(x_k) - f(x_k + \alpha\Delta x_k)) + \rho\frac{1}{\alpha}(\|v(x_k)\|_1 - \|v(x_k + \alpha\Delta x_k)\|_1).$$

If we assume the existence of second derivatives for f, standard arguments give

$$\lim_{\alpha\to 0^+} \frac{1}{\alpha}(f(x_k) - f(x_k + \alpha\Delta x_k)) = -\nabla f(x_k)^{\mathrm{T}}\Delta x_k.$$

Making the same assumption for c and using the relation $v(x_k) = v^k(x_k) + \mathrm{o}(\alpha)$ with (22) gives

$$\lim_{\alpha\to 0^+} \frac{1}{\alpha}(\|v(x_k)\|_1 - \|v(x_k + \alpha\Delta x_k)\|_1) = \|v(x_k)\|_1.$$

These limits combined with (24) and (26) imply the desired result (25), which completes the proof. □

Backtracking generally requires more evaluations of f and c than polynomial interpolation. However, this disadvantage can be offset by the savings gained by the use of the modified QP, as we will see in Section 5.

At each iteration, a penalty parameter ρ_k is used to estimate the quantity ρ^* that ensures that x^* a local minimizer of $\mathscr{M}$. The value of ρ_k is determined by retaining a "current" value, which is increased if necessary to satisfy the lower bound of Theorem 2. For example, at iteration k, the penalty parameter ρ_k can be defined by $\rho_k = \max\{\hat{\rho}_k, 2\rho_{k-1}\}$, where $\rho_0 = 0$ and $\hat{\rho}_k$ is defined by Theorem 2.

5. Numerical results

This section presents numerical solutions to an optimal control test problem using the proposed algorithm. The results are compared with those obtained using the standard single shooting and multiple shooting technique on the Cray C90 supercomputer.

5.1. Optimal control problem formulation

Consider the following optimal control problem of following a specified temperature trajectory over a given two-dimensional domain.

A rectangular domain in space is heated by controlling the temperature on its boundaries. It is desired that the transient temperature in a specified interior sub-domain follow a prescribed temperature-time trajectory as closely as possible. The domain Ω is given by

$$\Omega = \{(x, y) \,|\, 0 \leqslant x \leqslant x_{\max},\ 0 \leqslant y \leqslant y_{\max}\},$$

and the control boundaries are given by

$$\partial\Omega_1 = \{(x, y) \,|\, y = 0\} \quad \text{and} \quad \partial\Omega_2 = \{(x, y) \,|\, x = 0\}.$$

The temperature distribution in Ω, as a function of time, is controlled by the heat sources across the boundaries, represented by control functions $u_1(x,t)$ on $\partial\Omega_1$, and $u_2(y,t)$ on $\partial\Omega_2$. The other two boundaries ($x = x_{\max}$ and $y = y_{\max}$) are assumed to be insulated, so that no energy flows into or

out of Ω along the normals to these boundaries. The objective is to control the temperature in the sub-domain

$$\Omega_c = \{(x, y) \,|\, x_c \leqslant x \leqslant x_{\max},\ y_c \leqslant y \leqslant y_{\max}\}$$

so as to follow a specified trajectory $\tau(t),\ t \in [0, t_{\max}]$.

We measure the difference between $T(x, y, t)$ and $\tau(t)$ on Ω_c by the function

$$\phi(u) = \int_0^{t_{\max}} \int_{y_c}^{y_{\max}} \int_{x_c}^{x_{\max}} w(x, y, t)[T(x, y, t) - \tau(t)]^2 \,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}t,$$

where $w(x, y, t) \geqslant 0$ is a specified weighting function. The control functions u_1 and u_2 are determined so as to

$$\underset{u}{\text{minimize}}\ \phi(u),$$

subject to $T(x, y, t)$ satisfying the following PDE, boundary conditions, and bounds

$$\begin{aligned}
&T_t = \alpha(T)[T_{xx} + T_{yy}] + S(T), \quad (x, y, t) \in \Omega \times [0, t_{\max}]\\
&T(x, 0, t) - \lambda T_y = u_1(x, t), \quad x \in \partial\Omega_1\\
&T(0, y, t) - \lambda T_x = u_2(y, t), \quad y \in \partial\Omega_2\\
&T_x(x_{\max}, y, t) = 0,\\
&T_y(x, y_{\max}, t) = 0,\\
&0 \leqslant T(x, y, t) \leqslant T_{\max}.
\end{aligned}$$

The controls u_1 and u_2 are also required to satisfy the bounds

$$0 \leqslant u_1,\ u_2 \leqslant u_{\max}.$$

The initial temperature distribution $T(x, y, 0)$ is a specified function. The coefficient $\alpha(T) = \lambda/c(T)$, where λ is the heat conduction coefficient and $c(T)$ is the heat capacity. The source term $S(T)$ represents internal heat generation, and is given by

$$S(T) = S_{\max} \mathrm{e}^{-\beta_1/(\beta_2 + T)}$$

where $S_{\max}$, β_1, $\beta_2 \geqslant 0$ are specified nonnegative constants.

The PDE is semi-discretized in space via finite differences. A uniform rectangular grid is constructed on the domain Ω

$$\begin{aligned}
&x_i = i\Delta x, \quad i = 0, 1, \ldots, m, \quad \Delta x = x_{\max}/m,\\
&y_j = j\Delta y, \quad j = 0, 1, \ldots, n, \quad \Delta y = y_{\max}/n.
\end{aligned}$$

Then let

$$\begin{aligned}
&T_{ij}(t) = T(x_i, y_j, t), \quad u_{1i}(t) = u_1(x_i, t), \quad \alpha_{ij}(t) = \alpha(T_{ij}(t)),\\
&S_{ij}(t) = S(T_{ij}(t)), \quad u_{2j}(t) = u_2(y_j, t).
\end{aligned}$$

The PDE is then approximated in the interior of Ω by the following system of $(m-1)(n-1)$ ODEs

$$\frac{\mathrm{d}T_{ij}}{\mathrm{d}t} = \frac{\alpha_{ij}}{\Delta x^2}[T_{i-1,j} - 2T_{ij} + T_{i+1,j}] + \frac{\alpha_{ij}}{\Delta y^2}[T_{i,j-1} - 2T_{ij} + T_{i,j+1}] + S_{ij}, \tag{27}$$

for $i = 1,2,\ldots,m-1$, $j = 1,2,\ldots,n-1$. Each of the $2(m+n)$ boundary points also satisfies a differential equation similar to (27). These will include values outside Ω, which are eliminated by using the boundary conditions. Specifically, we use

$$T_{i,n+1} = T_{i,n-1}, \quad i = 0,1,\ldots,m,$$

$$T_{m+1,j} = T_{m-1,j} \quad j = 0,1,\ldots,n,$$

to approximate the conditions $T_y = 0$ and $T_x = 0$.

The finite-difference approximations to the boundary conditions on $\partial\Omega_1$ and $\partial\Omega_2$ are given by

$$T_{i0} - \frac{\lambda}{2\Delta y}(T_{i1} - T_{i,-1}) = u_{1i}, \quad i = 0,1,\ldots,m, \tag{28a}$$

$$T_{0j} - \frac{\lambda}{2\Delta x}(T_{1j} - T_{-1,j}) = u_{2j}, \quad j = 0,1,\ldots,n. \tag{28b}$$

These relations are used to eliminate the values $T_{i,-1}$ and $T_{-1,j}$ from the differential equations (as in (27)), for the functions T_{ij} on $\partial\Omega_1$ and $\partial\Omega_2$. As a result, the control functions u_{1i} and u_{2j} are explicitly included in these differential equations, giving $2(m+n)$ additional differential equations. Together with the $(m-1)(n-1)$ ODEs given by (27), this gives a total of $(m+1)(n+1)$ ODEs for the same number of unknown functions $T_{ij}(t)$.

5.2. Solution

The numerical solution was obtained by solving the semi-discretized PDE in time via DASPKSO (using $\texttt{RTOL} = 10^{-6}$ and $\texttt{ATOL} = 10^{-6}$) using (1) single shooting, (2) multiple shooting, and (3) modified multiple shooting. For all cases, the PDE parameters were assumed to be constant, with the values $\alpha = 1.0$, $\beta_1 = 0.2$, $\beta_2 = 0.05$, $\lambda = c = 0.5$, $S_{\max} = 0.5$, $T_{\max} = 0.7$. The solutions correspond to $t_{\max} = 2.0$, $x_{\max} = 0.8$, $y_{\max} = 1.6$, $u_{\max} = 1.0$, $x_c = 0.6$, and $y_c = 0.6$. Controls on the boundaries are given by

$$u_1(x,t) = \begin{cases} u(t), & 0 \leqslant x \leqslant 0.2, \\ \left(1 - \dfrac{x-0.2}{1.2}\right) u(t), & 0.2 \leqslant x \leqslant 0.8, \end{cases}$$

$$u_2(x,t) = \begin{cases} u(t), & 0 \leqslant y \leqslant 0.4, \\ \left(1 - \dfrac{y-0.4}{2.4}\right) u(t), & 0.4 \leqslant y \leqslant 1.6. \end{cases}$$

The initial conditions for states and controls are $T(0) = 0$, $u(0) = 0$. For the control parameterization, the time integration interval was divided into 20 equally spaced subintervals where the control function $u(t)$ is represented by a quadratic polynomial

$$u_j(t) = \overline{u}_{j0} + \overline{u}_{j1}(t - t_j) + \overline{u}_{j2}(t - t_j)^2. \tag{29}$$

Continuity in time was enforced at the extremities of each control subinterval among all $u_j(t)$ and their derivative $u_j'(t)$.

Table 1
Number of iterations and CPU time (in s), with $n_p = 60$, for single shooting (SS), multiple shooting (MS), and modified multiple shooting (MMS)

Problem size		Major itns			Computation time		
Mesh	n_y	SS	MS	MMS	SS	MS	MMS
4×4	25	12	8	12	214	141	188
4×8	45	14	12	13	446	577	376
8×8	81	7	6	15	566	1264	970
4×16	85	14	6	15	1132	1382	1023
8×16	153	15	7	20	2818	5306	3072
16×16	289	20	6	11	1 3551	24789	6283

The target trajectory $\tau(t)$ was given by

$$\tau(t) = \begin{cases} 1.25(t - 0.2) & \text{if } 0.2 < t \leqslant 0.6, \\ 0.5 & \text{if } 0.6 < t \leqslant 1.0, \\ 0.5 - 0.75(t - 1.0) & \text{if } 1.0 < t \leqslant 1.4, \\ 0.2 & \text{if } 1.4 < t \leqslant 2.0, \\ 0 & \text{otherwise.} \end{cases}$$

The weight function $w(x, y, t)$ was taken to be 1 in the interior of Ω_c, 0.5 in the interior of the boundary lines of Ω_c, 0.25 on the corners of the boundary of Ω_c, and 0 elsewhere.

The optimizers were started with initial guess for states and controls as constant over the entire time interval, equal to their values at $t = 0$. The feasibility and optimality tolerance for convergence were taken as 10^{-3} and 10^{-4}, respectively.

Performance results for the three methods on the test problem are given in Table 1. This test problem has the property that the size of the optimization problem can be increased by simply using a finer spatial grid. This readily permits the dependence of solution time on problem size to be observed. Fig. 1 shows optimal solutions computed using single shooting for increasing mesh size. The other methods yielded virtually indistinguishable results for this problem.

In general, the single-shooting technique requires more iterations as compared to the other techniques. During the process of obtaining optimal trajectories, we confirmed the lack of robustness of single shooting with respect to the initial guess and bounds on optimizing variables. For instance, the method failed to converge unless the control was constrained to be nonnegative.

For multiple shooting, the total time interval was divided into 10 equal shooting intervals. The computation times were significant and increased rapidly as the mesh became finer.

The modified multiple shooting has two important advantages over the other two techniques. (1) It is more robust than single shooting with respect to initial guess, and (2) the increase in computation time for finer mesh size ($n_p \ll n_y$) is less than that in the case of multiple shooting.

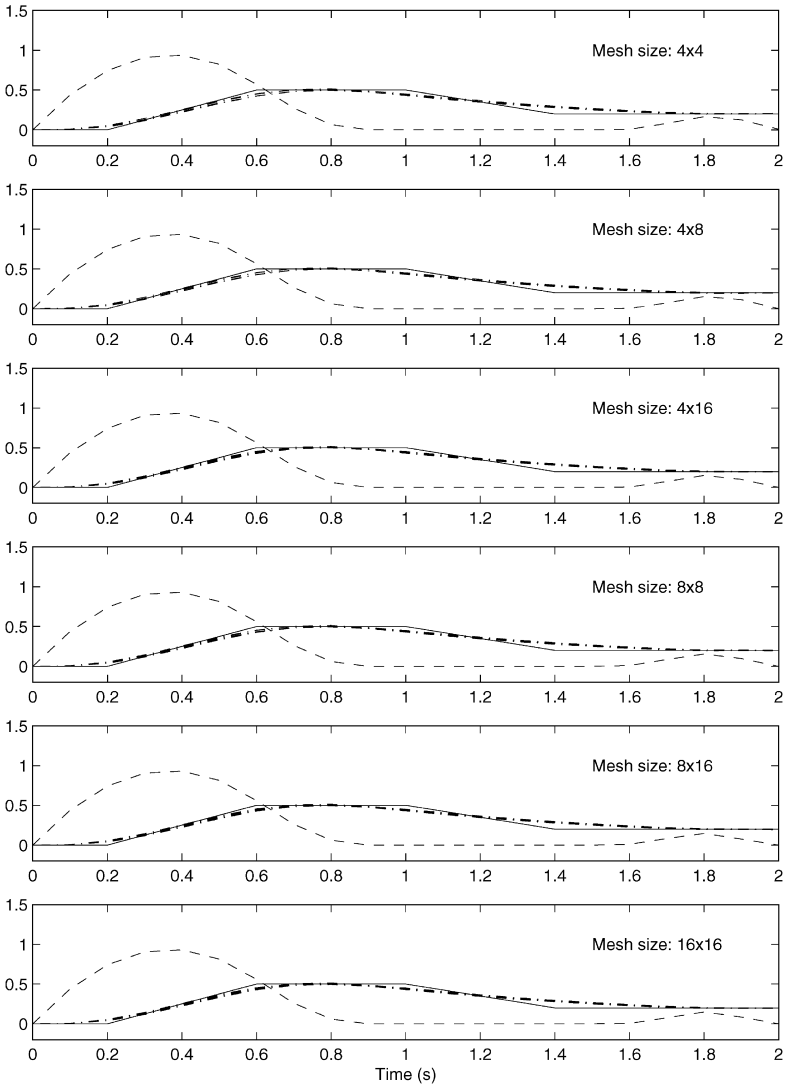

Fig. 1. Optimal solutions for increasing mesh size. Solid line: $\tau(t)$. Dashed line: $u(t)$. Dash-dot lines: $T_{ij}(t)$ in Ω_c.

References

[1] U.M. Ascher, R.M.M. Mattheij, R.D. Russell, Numerical Solution of Boundary Value Problems for Ordinary Differential Equations, Classics in Applied Mathematics, Vol. 13, Society for Industrial and Applied Mathematics (SIAM) Publications, Philadelphia, PA, 1995, ISBN 0-89871-354-4.

[2] L.T. Biegler, J. Nocedal, C. Schmid, A reduced Hessian method for large-scale constrained optimization, SIAM J. Optim. 5 (1995) 314–347.

[3] C. Bischof, A. Carle, G. Corliss, A. Griewank, P. Hovland, ADIFOR — generating derivative codes from Fortran programs, Sci. Programming 1 (1992) 11–29.

[4] R. Fletcher, ℓ_1 penalty method for nonlinear constraints, in: P.T. Boggs, R.H. Byrd, R.B. Schnabel (Eds.), Numerical Optimization 1984, SIAM, Philadelphia, PA, 1984, pp. 26–40.

[5] P.E. Gill, W. Murray, M.A. Saunders, SNOPT: an SQP algorithm for large-scale constrained optimization, Numerical Analysis Report 97-2, Department of Mathematics, University of California, San Diego, La Jolla, CA, 1997.
[6] P.E. Gill, W. Murray, M.A. Saunders, M.H. Wright, Some theoretical properties of an augmented Lagrangian merit function, in: P.M. Pardalos (Ed.), Advances in Optimization and Parallel Computing, North-Holland, Amsterdam, 1992, pp. 101–128.
[7] P.E. Gill, W. Murray, M.H. Wright, Practical Optimization, Academic Press, London, 1981, ISBN 0-12-283952-8.
[8] T. Maly, L.R. Petzold, Numerical methods and software for sensitivity analysis of differential-algebraic systems, Appl. Numer. Math. 20 (1996) 57–79.
[9] L. Petzold, J.B. Rosen, P.E. Gill, L.O. Jay, K. Park, Numerical optimal control of parabolic PDEs using DASOPT, in: L. Biegler, T. Coleman, A. Conn, F. Santosa (Eds.), Large Scale Optimization with Applications, Part II: Optimal Design and Control, IMA Volumes in Mathematics and its Applications, Vol. 93, 1997, pp. 271–300.
[10] M.J.D. Powell, Variable metric methods for constrained optimization, in: A. Bachem, M. Grötschel (Eds.), Mathematical Programming: The State of the Art, Springer, London, 1983, pp. 288–311.
[11] K. Schittkowski, NLPQL: a Fortran subroutine for solving constrained nonlinear programming problems, Ann. Oper. Res. 11 (1985/1986) 484–500.
[12] J.P. Schlöder, Numerische methoden zur behandlung hochdimensionaler aufgaben der parameteridentifizierung, Ph.D. Dissertation, University of Bonn, 1988.
[13] V.H. Schulz, Reduced SQP methods for large-scale optimal control problems in DAE with application to path planning problems for satellite mounted robots, Ph.D. Dissertation, University of Heidelberg, 1996.
[14] M.C. Steinbach, H.G. Bock, G.V. Kostin, R.W. Longman, Mathematical optimization in robotics: towards automated high speed motion planning, Preprint SC 97-03, Konrad-Zuse Zentrum für Informationstechnik Berlin, 1997.