SQP METHODS AND THEIR APPLICATION TO NUMERICAL OPTIMAL CONTROL*

ALEX BARCLAY† PHILIP E. GILL† J. BEN ROSEN‡

Abstract. In recent years, general-purpose sequential quadratic programming (SQP) methods have been developed that can reliably solve constrained optimization problems with many hundreds of variables and constraints. These methods require remarkably few evaluations of the problem functions and can be shown to converge to a solution under very mild conditions on the problem.

Some practical and theoretical aspects of applying general-purpose SQP methods to optimal control problems are discussed, including the influence of the problem discretization and the zero/nonzero structure of the problem derivatives. We conclude with some recent approaches that tailor the SQP method to the control problem.

1. INTRODUCTION

Recently there has been considerable progress in the development of general-purpose sequential quadratic programming (SQP) methods for large-scale nonlinear optimization (see, e.g., Eldersveld [12], Tjoa and Biegler [34], Betts and Frank [4] and Gill, Murray and Saunders [16]). These methods are efficient and reliable, and can be applied to large sparse problems with a mixture of linear and nonlinear constraints. The methods require remarkably few evaluations of the problem functions and converge to a solution under very mild conditions on the problem. An important application for these methods is the class of problems derived by discretizing optimal control problems. These problems have several important characteristics: (i) many variables and constraints; (ii) sparse and structured constraint and objective derivatives; (iii) objective and constraint functions (and their first derivatives) that are expensive to evaluate; and (iv) many constraints active (i.e., exactly satisfied) at the solution.

In §2. we give a brief discussion of general-purpose SQP methods, with an emphasis on those aspects that most affect performance. In §3. we consider the application of these general-purpose methods to optimal control problems. In §4. we discuss the most common forms of discretization and consider their effect on the efficiency of general-purpose SQP methods. Finally, in §5. we briefly consider some SQP methods that are specially tailored to problems derived from optimal control problems.

*This research was partially supported by National Science Foundation grants CCR-95-27151 and DMI-9424639, Office of Naval Research grants N00014-90-J-1242 and N00014-96-1-0274.

†Department of Mathematics, University of California, San Diego, La Jolla, California 92093-0112.

‡Department of Computer Science and Engineering, University of California, San Diego, La Jolla, California 92093-0114.

2. SQP METHODS

The general nonlinearly constrained optimization problem can be written in the form

$$\begin{array}{ll} \underset{x\in \mathbb{R}^n}{\text{minimize}} & F(x) \\ \text{subject to} & b_l \le \left\{ \begin{array}{c} x \\ Ax \\ c(x) \end{array} \right\} \le b_u, \end{array} \tag{1}$$

where c is a vector of nonlinear functions, A is a constant matrix that defines the linear constraints, and b_l and b_u are constant upper and lower bounds. It is assumed that first derivatives of the problem are known explicitly, i.e., at any point x it is possible to compute the gradient $\nabla F(x)$ of the objective F and the Jacobian $J(x)$ of the nonlinear constraints c. Our principal concern is with large problems, although the precise definition of the size of a problem depends on several factors, including the zero/nonzero structure of the problem derivatives and the number of degrees of freedom at a solution (i.e., the number of variables less the number of active constraints at a solution)[1].

At a constrained minimizer x^*, the objective gradient ∇F can be written as a linear combination of the constraint gradients. The multipliers in this linear combination are known as the *Lagrange multipliers*. The Lagrange multipliers for an upper bound constraint are nonpositive, the multipliers for a lower bound constraint are nonnegative. The vector of Lagrange multipliers associated with the *nonlinear* constraints of (1) is denoted by π^*.

SQP methods are a class of optimization methods that solve a quadratic programming subproblem at each iteration. Each QP subproblem minimizes a quadratic model of a certain modified Lagrangian function subject to linearized constraints. A merit function is reduced along each search direction to ensure convergence from any starting point. The basic structure of an SQP method involves *major* and *minor* iterations. The major iterations generate a sequence of iterates (x_k, π_k) that converge to (x^*, π^*). At each iterate a QP subproblem is used to generate a search direction towards the next iterate (x_{k+1}, π_{k+1}). Solving such a subproblem is itself an iterative procedure, with the *minor* iterations of an SQP method being the iterations of the QP method. (For an overview of SQP methods, see, for example, Gill, Murray and Wright [18].)

In the SQP formulation considered here, the objective and constraint derivatives ∇F and J are required once each major iteration in order to define the objective and constraints of the QP subproblem. SQP methods have two important properties. First, they are most robust when the derivatives of the objective and constraint functions are computed exactly. Second, the SQP iterates do not usually satisfy the nonlinear equality constraints except as the solution is approached. (However, it is possible to ensure that the iterates always satisfy the *linear* constraints.)

Each QP subproblem minimizes a quadratic model of the *modified Lagrangian* $\mathcal{L}(x, x_k, \pi_k) = F(x) - \pi_k^T d_L(x, x_k)$, which is defined in terms of the *constraint linearization*, $c_L(x, x_k) = c(x_k) + J(x_k)(x - x_k)$, and the *departure from linearity*, $d_L(x, x_k) =$

[1] Some authors define the number of degrees of freedom as the number of variables less the number of equality constraints (i.e., constraints with $(b_l)_i = (b_u)_i$).

$c(x) - c_L(x, x_k)$. Given estimates (x_k, π_k) of (x^*, π^*), an improved estimate is found from $(\widehat{x}_k, \widehat{\pi}_k)$, the solution of the following QP subproblem:

$$\begin{array}{ll} \underset{x \in I\!R^n}{\text{minimize}} & F(x_k) + \nabla F(x_k)^T (x - x_k) + \tfrac{1}{2}(x - x_k)^T H_k (x - x_k) \\ \text{subject to} & b_l \le \left\{ \begin{array}{c} x \\ Ax \\ c(x_k) + J(x_k)(x - x_k) \end{array} \right\} \le b_u, \end{array}$$

where H_k is a positive-definite approximation to $\nabla^2_x \mathcal{L}(x_k, x_k, \pi_k)$. Once the QP solution $(\widehat{x}_k, \widehat{\pi}_k)$ has been determined, the major iteration proceeds by determining new variables (x_{k+1}, π_{k+1}) as $x_{k+1} = x_k + \alpha_k(\widehat{x}_k - x_k)$ and $\pi_{k+1} = \pi_k + \beta_k(\widehat{\pi}_k - \pi_k)$, where the scalars α_k and β_k are chosen to yield a sufficient decrease in a *merit function* (a suitable combination of the objective and constraint functions). Some methods determine α_k and β_k simultaneously using a linesearch, others use a linesearch for α_k but keep β_k fixed at 1. The choice of merit function will not be considered here, but choices that have proved effective in practice are the l_1 penalty function (see e.g., Fletcher [14]) and an augmented Lagrangian function (see Schittkowski [30] and Gill, Murray and Saunders [16]).

The definition of the QP Hessian H_k is crucial to the success of an SQP method. In the method of SNOPT (Gill, Murray and Saunders [16]), H_k is a limited-memory quasi-Newton approximation to $G = \nabla^2_x \mathcal{L}(x_k, x_k, \pi_k)$, the Hessian of the modified Lagrangian. Another possibility is to define H_k as a positive-definite matrix related to a finite-difference approximation to G. Exact second derivatives have also been proposed in this context (see, e.g., Fletcher [14], Eldersveld and Gill [13]).

The QP solver must repeatedly solve linear systems formed from rows and columns of the QP constraint matrices. The class of active-set QP methods solve a system of the form

$$\begin{pmatrix} H & W^T \\ W & 0 \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} g \\ h \end{pmatrix} \tag{2}$$

each minor iteration. The matrices H and W consists of a subset of the rows and columns of H_k, $J(x_k)$ and A specified by the *working-set*, which is an estimate of the constraints active at a solution. Eventually, the working sets of the QP subproblem usually “settle down” to the active set of the nonlinear problem. For this reason, the final working set from one QP subproblem is used to guide the initial working set in the next. Once the active set has been identified, the QP subproblems reach optimality in one iteration, and hence only a single system (2) need be solved in later subproblems.

The method used to solve (2) largely determines the characteristics of the QP method. When the system is large and sparse, we know of only two viable QP methods—the reduced-Hessian method and the Schur-complement method.

Reduced-Hessian QP methods. These methods solve (2) using a full-rank matrix Z that spans the null space of W (i.e., $WZ = 0$). The matrix Z is used to transform (2) into two smaller systems, one of which involves the *reduced Hessian* Z^THZ. The reduced Hessian is stored and updated in dense form, which is efficient as long as the number of degrees of freedom is small compared to the number of variables.

In particular, reduced Hessian methods are efficient if W is nearly square and products $H_k x$ can be formed efficiently.

If W is large and sparse, Z is usually maintained in "reduced-gradient" form, using sparse LU factors of a square matrix B that alters as the working set changes. The defining equations are

$$WP = \begin{pmatrix} B & S \end{pmatrix}, \qquad Z = P\begin{pmatrix} -B^{-1}S \\ I \end{pmatrix}, \tag{3}$$

where P is a permutation that ensures B is nonsingular. The number of degrees of freedom is the number of columns of S. Products of the form Zv and Z^Tg are obtained by solving with B or B^T. For more details, see Gill, Murray and Saunders [16].

Schur-complement QP methods. After each change to W, the system (2) is equivalent to a system of the form

$$\begin{pmatrix} H_0 & W_0^T & U \\ W_0 & 0 & 0 \\ U^T & 0 & V \end{pmatrix}\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix},$$

where the number of columns of U is equal to the number of QP iterations. Given the system

$$K_0 = \begin{pmatrix} H_0 & W_0^T \\ W_0 & 0 \end{pmatrix} \tag{4}$$

at the start of the QP, factorizations of K_0 and the Schur complement $C = V - U^T K_0^{-1} U$ can be used to solve (2). It is efficient to work with a sparse factorization of K_0 and dense factors of C. The factorization of K_0 may be computed using any suitable code. If the number of updates is small enough, C may be maintained as a dense factorization (based on either orthogonal or elementary transformations), and updated to reflect changes in W. As the dimension of C grows, it is eventually necessary to refactorize the system (2) from scratch. For more details, see Gill, Murray, Saunders and Wright [17]. The Schur-complement QP method is implemented in the code SOCS of Betts and Frank [4].

3. SQP METHODS FOR OPTIMAL CONTROL

In this section we give a brief review of the application of general-purpose SQP methods to optimal control problems in which the dynamics are determined by a system of ordinary differential equations (ODEs). The problem is assumed to be of the form

$$\begin{array}{lrcll}
\underset{u,y}{\text{minimize}} & y_1(t_f) & & & (5)\\
\text{subject to} & y(t_0) & = & y_0, & (6)\\
 & \dot{y}(t) & = & f(y,u,t), \quad t \in [0,t_f], & (7)\\
 & g(y,u,t) & \ge & 0, \qquad\qquad t \in [0,t_f], & (8)
\end{array}$$

where $y(t)$ is an n_y-vector of state variables and $u(t)$ is an n_u-vector of controls. The inequalities (8) often include upper and lower bounds on control variables $u(t)$ and state variables $y(t)$. It is assumed that given the initial condition y_0 and the control function $u = u(t)$, $t \in [0, t_f]$, the state vector function $y = y(t)$ is uniquely determined by the differential system (7). We also assume that the control $u(t)$ is continuous and satisfies some standard conditions needed for the existence of an optimal control (see, e.g., Leitman [25]).

4. DISCRETIZING THE CONTROL PROBLEM

Although the size of the ODE system (7) can be large, the dimension of the control vector $u(t)$ is typically much smaller. The usual approach is to represent $u(t)$ as a low-order spline or piecewise polynomial function on $[0, t_f]$. The coefficients of this spline or piecewise polynomial are then adjusted during the optimization.

Given the initial conditions y_0 and values of the spline coefficients, the controls can be evaluated at any point in $[0, t_f]$ and the state vector function $y = y(t)$ is uniquely determined by the differential system (7). However, it is neither necessary (nor advisable) to explicitly solve the differential system at each step. Instead, finite-dimensional nonlinear equations associated with certain well-known ODE methods are imposed as nonlinear constraints during the optimization. A feature of this approach is that, since SQP methods do not generally satisfy nonlinear equality constraints until the solution is approached, the differential system is only satisfied near the end of the computation.

4.1 Discretizing the control functions

The total time interval $[0, t_f]$ is divided into N equal subintervals of length Δt each. These subintervals define $N + 1$ *nodes*:

$$t_k = k\Delta t, \quad k = 0, 1, \ldots, N, \quad \text{with} \quad t_N = t_f. \tag{9}$$

A piecewise polynomial $u^k(t)$ is used to represent $u(t)$ for $t \in [t_k, t_{k+1}]$. For example, if $u^k(t)$ is a cubic polynomial, then $u^k(t)$ can be represented by $4n_u$ coefficients, which are then determined by the optimization. The continuity of the $u^k(t)$ and their derivatives at the nodes is enforced by means of appropriate linear equality constraints. Any bounds on the $u^k(t)$ at the nodes imply additional linear inequalities on the coefficients of the polynomial.

A convenient alternative representation can be defined using Hermite interpolation. In the case of a cubic approximation, optimization variables u_k and w_k are introduced to represent the values of $u(t)$ and $\dot{u}(t)$ at each node t_k. The unique piecewise cubic $u^k(t)$ is defined that has values (u_k, u_{k+1}) and derivatives (w_k, w_{k+1}) at the end points of $[t_k, t_{k+1}]$. This cubic allows the control at any point $t = t_k + \rho\Delta t$ in $[t_k, t_{k+1}]$ to be written as

$$u^k(t) = u_\rho + \rho(\rho - 1)\left[(u_{k+1} - u_k)(1 - 2\rho) + w_\rho \Delta t\right], \tag{10}$$

where $u_\rho = (1 - \rho)u_k + \rho u_{k+1}$ and $w_\rho = (\rho - 1)w_k + \rho w_{k+1}$ (see, e.g., Hairer, Nørsett and Wanner [20, p.190]). This scheme automatically gives continuity of $u^k(t)$ and its derivative at the nodes. Moreover, simple upper and lower bounds on the controls $u(t)$ lead to simple upper and lower bounds on the discretized variables $\{u_k\}$.

4.2 Discretizing the differential equations

Three methods for discretizing the ODEs (7) will be considered: single shooting, multiple shooting and collocation. Each of these methods gives a finite-dimensional problem with a different character.

Single shooting. Given the initial conditions y_0 and a piecewise polynomial control approximation $\{u^k(t)\}$, the state vector $y = y(t)$ is uniquely determined by the differential system (7). The method of single shooting minimizes with respect to the control variables (say, $\{u_k, w_k\}$) and solves (7) over $[0, t_f]$ at each new point generated by the optimization algorithm. It follows that the differential equations are being used to *eliminate* the state variables from the problem. It is common for an adaptive ODE solver to be used for this process.

The inequality constraints (8) can be imposed explicitly at each control subinterval boundary as requirements on the u_k. These become

$$g(y_k, u_k, t_k) \geq 0, \qquad k = 0, 1, \ldots, N, \tag{11}$$

where y_k denotes the state value at t_k. The objective value is the first component of y_N.

The discretized problem is a nonlinear optimization problem with variables $x = (u_0, w_0, u_1, w_1, \ldots, u_N, w_N)^T$. In order to apply an SQP method, it is necessary to be able to compute the derivatives of the inequality constraints (11). These calculations involve the partial derivatives $\partial y_i/\partial u_j$ and $\partial y_i/\partial w_j$, where y_i denotes the ODE solution evaluated at t_i. Each of these derivatives is itself the solution of a differential equation whose right-hand side involves the derivatives f'_y and f'_u. This system must be solved in conjunction with the original ODEs. Since the state value y_k is independent of (u_i, w_i) for $i > k$, the Jacobian of y with respect to u and w is block lower triangular. Packages are available that allow the efficient and reliable computation of these Jacobians (see, e.g., Buchauer, Hiltmann and Kiehl [7], Maly and Petzold [26]). These packages allow a constraint derivative to be computed with approximately the same effort and the same accuracy as the constraint value itself.

Multiple shooting. In this case, the interval $[0, t_f]$ is divided into subintervals and the differential equations are solved over each subinterval. Continuity of the solutions between the subintervals is achieved by enforcing the continuity conditions as nonlinear equations (see Ascher, Mattheij and Russell [1]). In the optimal control context, the multiple shooting equations are imposed as constraints in the optimization. When the constraints need to be evaluated, the differential equation (7) is regarded as an independent initial-value problem over each of the subintervals $[t_k, t_{k+1}]$. A continuous solution over $[0, t_f]$ is obtained by matching the initial conditions at t_k with the final values obtained from the previous subinterval $[t_{k-1}, t_k]$. The initial values of y for each subinterval are treated as variables of the optimization and they are adjusted to achieve continuity over $[0, t_f]$.

Let $y^k(t)$ denote the solution of the differential equations (7) on the interval $[t_k, t_{k+1}]$ with initial conditions $y^k(t_k) = y_k$. The value of y_0 is given, and the vectors y_k, $k = 1$, $2, \ldots, N-1$, are variables to be adjusted by the optimizer. For simplicity, we assume that $u(t)$ is represented by the Hermite piecewise cubic polynomial $u^k(t)$ defined above. Given y_k and the control values u_k and w_k at the nodes, the differential system (7) can be solved by a "black-box" solver to give $y^k(t_{k+1})$. In general, $y^k(t_{k+1})$ will not equal the initial value y_{k+1} for the next interval, and it is necessary to adjust y_k, u_k and w_k so that the continuity conditions

$$c_k = y^k(t_{k+1}) - y_{k+1} = 0, \qquad k = 0, 1, \ldots, N-1, \tag{12}$$

are satisfied at the subinterval boundaries. The last of these conditions involves y_N, which does not specify an initial value for any differential equation. This vector can be used to impose a condition on $y^{N-1}(t_N)$ arising from either an explicit condition on $y(t_f)$ or a condition on y from the inequality constraints $g \geq 0$, which are imposed pointwise at the nodes, as in single shooting. If there are no inequalities, y_N can be a free variable in the optimization. The initial-value problems for each subinterval are independent and can be solved in parallel (see, e.g., Kiehl [22], and Petzold *et al.* [29]).

As in single shooting, the calculation of the problem derivatives requires the solution of additional ODEs. However, in this case the Jacobian and Hessian are sparse and structured. If Hermite interpolation is used to define each $u^k(t)$, the vector of variables x for the finite-dimensional problem can be written in the form $x = (u_0, w_0, y_1, u_1, w_1, \ldots, y_N, u_N, w_N)^T$, with a grand total of $n = n_y N + 2n_u(N+1)$ variables. With this ordering, the Jacobian of the continuity conditions (12) has the form

$$J = \begin{pmatrix} U_0 & -I & & & & & & & \\ & Y_1 & U_1 & -I & & & & & \\ & & & Y_2 & U_2 & -I & & & \\ & & & & & \ddots & \ddots & \ddots & \\ & & & & & & Y_{N-1} & U_{N-1} & -I \end{pmatrix},$$

where $Y_k = \partial c_k / \partial y_k$ and U_k contains the partial derivatives $\partial c_k / \partial u_k$ and $\partial c_k / \partial w_k$ in appropriate order. The Jacobian for the inequality constraints $g \geq 0$ (11) will depend upon the particular application. The Lagrangian Hessian is also highly structured. If there are no nonlinear constraints other than the continuity conditions, then $\nabla^2_x \mathcal{L}(x, x_k, \pi_k)$ is a block-diagonal matrix with the same structure as $J^T J$.

Methods based upon multiple shooting have been proposed, amongst others, by Pantelides, Sargent and Vassiliadis [27], Gritsis, Pantelides and Sargent [19], and Petzold *et al.* [29]).

Collocation. Methods for discretizing the differential constraints using collocation have been proposed by Dickmanns and Well [11], Kraft [23], Hargraves and Paris [21], and Betts and Huffman [5]. The method of collocation is closely related to multiple shooting. Our discussion follows that of Brenan [6].

Discretization by collocation may be regarded as a form of multiple shooting in which an appropriate implicit Runge Kutta (IRK) formula is used to solve the initial-value problem over $[t_k, t_{k+1}]$. An s-stage IRK formula involves constants $\{\alpha_{ij}\}$, $\{\beta_j\}$

$$\begin{array}{c|ccc} \rho_1 & \alpha_{11} & \cdots & \alpha_{1s} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_s & \alpha_{s1} & \cdots & \alpha_{ss} \\ \hline & \beta_1 & \cdots & \beta_s \end{array} \qquad \begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{5}{24} & \frac{1}{3} & -\frac{1}{24} \\ 1 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array}$$

Figure 1: The Butcher tableau and its definition for Lobatto IIIA, $s = 3$.

and $\{\rho_i\}$ specified by a Butcher tableau (see Figure 1) and involves the evaluation of the right-hand side f at the points $t_{kj} = t_k + \rho_j \varDelta t$, for $j = 1, 2, \ldots, s$. The solution of the initial-value problem is the state vector $y^k(t_{k+1})$ such that

$$y^k(t_{k+1}) = y_k + \varDelta t \sum_{j=1}^{s} \beta_j f(y_{kj}, u_{kj}, t_{kj}), \tag{13}$$

where the quantities y_{kj} satisfy the s collocation conditions

$$y_{kj} - y_k - \varDelta t \sum_{l=1}^{s} \alpha_{jl} f(y_{kl}, u_{kl}, t_{kl}) = 0, \qquad j = 1, \ldots, s, \tag{14}$$

with the suffix ij denoting a quantity defined at t_{ij}. As in multiple shooting, the vector $y^k(t_{k+1})$ is matched with y_{k+1} by imposing the continuity condition $c_k = 0$ as an optimization constraint, where $c_k = y^k(t_{k+1}) - y_{k+1}$. The collocation conditions are also imposed as constraints, giving an additional s nonlinear equality constraints and s unknowns $\{y_{kl}\}$ at each node.

The number of variables and constraints is reduced if the IRK scheme includes $\rho_1 = 0$ or $\rho_s = 1$, since one or two redundant collocation equations can be eliminated. For example, with the Lobatto IIIA scheme with $s = 3$, the Butcher tableau of Figure 1 indicates that only three collocation points are required: $t_{k1} = t_k$, $t_{k2} = \frac{1}{2}(t_k + t_{k+1})$, and $t_{k3} = t_{k+1}$. The continuity conditions (12) are then defined with

$$y^k(t_{k+1}) = y_k + \frac{\varDelta t}{6}\big(f(y_k, u_k, t_k) + 4f(y_{k2}, u_{k2}, t_{k2}) + f(y_{k+1}, u_{k+1}, t_{k+1})\big), \tag{15}$$

where the state value y_{k2} satisfies the single collocation condition

$$y_k + \frac{\varDelta t}{24}\big(5f(y_k, u_k, t_k) + 8f(y_{k2}, u_{k2}, t_{k2}) - f(y_{k+1}, u_{k+1}, t_{k+1})\big) - y_{k2} = 0, \tag{16}$$

with $u_{k2} = \frac{1}{2}(u_k + u_{k+1}) + \frac{1}{8}\varDelta t(w_k - w_{k+1})$. A similar scheme using a 2-stage Lobatto IRK is proposed by Betts and Huffman [5]. See also, Pesch [28], Lamour [24], von Stryk and Bulirsch [36], Bulirsch *et al.* [8], von Stryk [35], Betts [3], and Schulz, Bock and Steinbach [32].

An important property of collocation is that the partial derivatives of problem functions are relatively simple to compute. For the moment, suppose that the relation $u_{k2} = \frac{1}{2}(u_k + u_{k+1}) + \frac{1}{8}\varDelta t(w_k - w_{k+1})$ is imposed as the *linear constraint*

$$\frac{1}{2}(u_k + u_{k+1}) + \gamma(w_k - w_{k+1}) - u_{k2} = 0, \tag{17}$$

where for brevity, we have denoted $\Delta t/8$ by γ. It follows that at each subinterval, the differential constraints give three derivatives, c_k', c_{k2}' and A_k, defined by the continuity conditions, the collocation conditions and the linear constraints above. These derivatives may be listed schematically as follows:

	y_k	u_k	w_k	y_{k2}	u_{k2}	y_{k+1}	u_{k+1}	w_{k+1}
c_k'	$I + \bar{\beta}_1 Y_k$	$\bar{\beta}_1 U_k$		$\bar{\beta}_2 Y_{k2}$	$\bar{\beta}_2 U_{k2}$	$\bar{\beta}_3 Y_{k+1} - I$	$\bar{\beta}_3 U_{k+1}$	
c_{k2}'	$I + \bar{\alpha}_{21} Y_k$	$\bar{\alpha}_{21} U_k$		$\bar{\alpha}_{22} Y_{k2} - I$	$\bar{\alpha}_{22} U_{k2}$	$\bar{\alpha}_{23} Y_{k+1}$	$\bar{\alpha}_{23} U_{k+1}$	
A_k		$\frac{1}{2}I$	γI		$-I$		$\frac{1}{2}I$	$-\gamma I$

where Y_k and U_k denote the Jacobians f_y' and f_u' evaluated at t_k and $\bar{\alpha}$ and $\bar{\beta}$ denote the Lobatto coefficients scaled by Δt. It follows that the full Jacobian has a block diagonal structure, with N overlapping diagonal blocks. Except for the first block (which must account for the ODE initial conditions) each diagonal block has sparsity and structure identical to that defined above.

If the derivatives f_y' and f_u' are sparse, as they are in many applications, this constraint Jacobian will be sparse and structured. This structure will not be altered significantly if the linear constraints (17) are used to eliminate the variables u_{k2}. In this case the derivatives are

	y_k	u_k	w_k	y_{k2}	y_{k+1}	u_{k+1}	w_{k+1}
c_k'	—	$\bar{\beta}_1 U_k + \frac{1}{2}\bar{\beta}_2 U_{k2}$	$\gamma\bar{\beta}_2 U_{k2}$	—	—	$\bar{\beta}_3 U_{k+1} + \frac{1}{2}\bar{\beta}_2 U_{k2}$	$-\gamma\bar{\beta}_2 U_{k2}$
c_{k2}'	—	$\bar{\alpha}_{21} U_k + \frac{1}{2}\bar{\alpha}_{22} U_{k2}$	$\gamma\bar{\alpha}_{22} U_{k2}$	—	—	$\bar{\alpha}_{23} U_{k+1} + \frac{1}{2}\bar{\alpha}_{22} U_{k2}$	$-\gamma\bar{\alpha}_{22} U_{k2}$

where an entry "—" implies that the derivative is unchanged by the elimination.

The constraints can be simplified further if Hermite cubic interpolation is used to represent $y(t)$ on the interval $[t_k, t_{k+1}]$. It follows from the identity $\dot{y}_k = f(y_k, u_k, t_k)$, that the state variable at the collocation point can be represented in terms of the y_k's, u_k's and w_k's (as in (10)). Hence we are left with just one implicit nonlinear equation

$$y_{k+1} - y_k - \frac{\Delta t}{6}\left(f(y_k, u_k, t_k) + 4f(y_{k2}, u_{k2}, t_{k2}) + f(y_{k+1}, u_{k+1}, t_{k+1})\right) = 0,$$

where $y_{k2} = \frac{1}{2}(y_k + y_{k+1}) + \frac{\Delta t}{8}(f(y_k, u_k, t_k) - f(y_{k+1}, u_{k+1}, t_{k+1}))$. This is the basis of the Hermite-Simpson method.

Although the variable y_{k2} can be eliminated from the problem, there is a strong argument from the optimization viewpoint for not performing the elimination explicitly. If the explicit relation for y_{k2} is kept as an explicit constraint, the associated derivatives are given by

	y_k	u_k	w_k	y_{k2}	u_{k2}	y_{k+1}	u_{k+1}	w_{k+1}
c_k'	$I + \bar{\beta}_1 Y_k$	$\bar{\beta}_1 U_k$		$\bar{\beta}_2 Y_{k2}$	$\bar{\beta}_2 U_{k2}$	$\bar{\beta}_3 Y_{k+1} - I$	$\bar{\beta}_3 U_{k+1}$	
c_{k2}'	$\frac{1}{2}I + \gamma Y_k$	γU_k		$-I$		$\frac{1}{2}I - \gamma Y_{k+1}$	$-\gamma U_{k+1}$	
A_k		$\frac{1}{2}I$	γI		$-I$		$\frac{1}{2}I$	$-\gamma I$

As before, if f_y' and f_u' are sparse, this scheme will give a sparse Jacobian for the solver. Alternatively, if y_{k2} is not an explicit variable, the derivatives become significantly more complicated (and hence, less sparse). For example, the partial derivatives of c'_k with respect to y_k and u_k are $-(I + \bar{\beta}_1 Y_k + \bar{\beta}_2 Y_{k2}(\frac{1}{2}I + \gamma Y_k))$ and $-(\bar{\beta}_1 U_k + \bar{\beta}_2 \gamma Y_{k2} U_k)$ respectively.

When modeling large control problems, there is always the temptation to eliminate as many constraints and variables as possible. In some cases this is justified–even essential, but the elimination is often done at the expense of creating a constraint Jacobian that is less sparse than the original. In large-scale optimization, it is not the size of the problem that is important as much as the nonzero structure of the Jacobian and Hessian. To be more precise, it is the interaction of this structure with the matrix factorizations used to solve the linear systems (3) and (4) that is crucial. In general, there is little to be lost, and much to be gained by hesitating before eliminating variables and constraints by hand. This observation is based on the assumption that the more sparse the Jacobian, the better the sparse solver is able to exploit the sparsity during the factorizations.

4.3 Properties of the discretizations

If there are few control variables, the number of degrees of freedom will be small compared to the number of variables, and methods based on the explicit estimation of the reduced Hessian $Z^T HZ$ will work efficiently.

Single shooting gives a discretized problem with the smallest number of variables and constraints. Moreover, the constraint Jacobian is essentially dense, and it is not necessary to use a sparse solver. Single shooting is usually used in conjunction with adaptive ODE software, and when it converges, it can be very efficient. In many cases, good results can be obtained from single shooting (see, e.g., Büskens and Maurer [9]). However, it is well-known that single shooting can suffer from a lack of robustness and stability (see, e.g., Ascher, Mattheij and Russell [1]). For some nonlinear problems it can generate intermediate iterates that are nonphysical and/or not computable. For some well-conditioned boundary-value problems, it can generate unstable initial-value problems.

If any one of the active inequality constraints (11) involves state variables, both single and multiple shooting require the calculation of the partial derivatives $\partial y_i/\partial u_j$ and $\partial y_i/\partial w_j$ at each node. There seems little justification for choosing single shooting in this case, given the superior stability properties of multiple shooting (see below). The decision is less straightforward if g does not involve the state variables, since no additional ODEs need to be solved in this case.

Multiple shooting with an appropriate ODE solver does not have the potential stability problems of single shooting. However, both single and multiple shooting have a weakness that is not present in collocation. In a “black-box” shooting scheme, it is common for the constraints and their derivatives to be evaluated using an adaptive ODE package in which the step sizes are chosen adaptively based on local error estimates. Current SQP methods require that the Jacobian elements be computed to high accuracy, which implies a relatively tight tolerance for the ODE solver. This can make the problem derivatives extremely expensive, with as much as 95% of the computation time being spent in the derivative calculation. By contrast, collocation derivatives can be obtained to (essentially) full precision at little cost if f'_y and f'_u are available. Moreover, if second derivatives of f can be computed (either by hand or with the use of a symbolic differentiation package), then second-derivative SQP methods can be applied.

Another advantage of collocation is that it is very convenient to implement a multigrid-type approach in which a solution on a coarse grid of nodes is used as an initial estimate for the larger problem on a finer grid. In this context, second derivative methods are likely to be significantly more efficient than first-derivative methods. Since quasi-Newton methods estimate second derivative information over a sequence of iterations, a poor initial Hessian may cause the algorithm to wander away from a good estimate of the solution. It may be seen that the reduced Hessian Z^THZ associated with different discretizations are not related in a simple way, even when one mesh is the refinement of another. This implies that the optimal reduced Hessian from one problem is not easily used to start an optimization on a finer grid. This difficulty does not occur with exact second derivatives since the reduced Hessian for the refined system is computed from scratch.

A disadvantage of collocation is that the number of variables and constraints is significantly increased compared to multiple shooting. However, as we have already indicated, this need not be a serious disadvantage if the problem derivatives are sufficiently sparse. A potentially more serious difficulty is that, for a given complexity of optimization problem, the state variables obtained by collocation may not be as accurate as those obtained by multiple shooting. In this case, a compromise is to use collocation to obtain a good starting estimate for a multiple shooting approach.

5. SQP METHODS THAT EXPLOIT STRUCTURE

A general-purpose SQP algorithm treats all variables and constraints equally, and does not distinguish between state and control variables. Many methods have been devised that exploit the structure of the problem with the aim of increasing efficiency. The formulation of such "modified" or "structured" SQP methods are considered next. For clarity, the notation will be changed so that y and u denote the components of the *discretized* versions of the state and control vectors, with y an n_1-vector and u an n_2-vector. The structured problem has n $(n = n_1 + n_2)$ variables and m $(m = m_1 + m_2)$ constraints, and is written as

$$\begin{array}{ll} \underset{y\in I\!R^{n_1},u\in I\!R^{n_2}}{\text{minimize}} & F(y,u) \\ \text{subject to} & c_1(y,u) = 0, \quad c_2(y,u) \geq 0. \end{array} \tag{18}$$

The m_1 constraints $c_1(y,u) = 0$ represent the continuity and collocation conditions associated with the discretized ODEs. The m_2 inequalities $c_2(y,u) \geq 0$ define all the inequality constraints, including simple upper and lower bounds. We assume that $n_1 = m_1$, so that the number of equality constraints is equal to the number of state variables. It follows that the constraint Jacobian for this problem has the form

$$J(x) = \begin{pmatrix} J_1 \\ J_2 \end{pmatrix} = \begin{pmatrix} J_1^y & J_1^u \\ J_2^y & J_2^u \end{pmatrix},$$

where J_1 and J_2 are $n_1 \times (n_1+n_2)$ and $m_2 \times (n_1+n_2)$, and the matrices J_1^y, J_1^u, J_2^y and J_2^u reflect the (y,u) partition of the variables. The QP subproblem associated with the structured nonliner problem (18) can be written as

$$\begin{array}{ll} \underset{\Delta y \in \mathbb{R}^{n_1}, \Delta u \in \mathbb{R}^{n_2}}{\text{minimize}} & g_y^T \Delta y + g_u^T \Delta u + \frac{1}{2}(\ \Delta y^T \quad \Delta u^T\ ) H \begin{pmatrix} \Delta y \\ \Delta u \end{pmatrix} \\ \text{subject to} & J_1^y \Delta y + J_1^u \Delta u = -c_1, \\ & J_2^y \Delta y + J_2^u \Delta u \geq -c_2, \end{array} \tag{19}$$

where Δy and Δu denote the predicted change in the state and control parameters respectively.

5.1 Structured Reduced-Hessian methods

Under the assumption that J_1 is invertible, the equality constraints $J_1^y \Delta y + J_1^u \Delta u = -c_1$ can be used to express Δy as a function of Δu, with

$$\Delta y = -(J_1^y)^{-1} c_1 - (J_1^y)^{-1} J_1^u \Delta u. \tag{20}$$

This expression may be used to establish the identity

$$\begin{pmatrix} \Delta y \\ \Delta u \end{pmatrix} = Y_1 c_1 + Z_1 \Delta u,$$

where Z_1 and Y_1 are $n \times n_2$ and $n \times n_1$ matrices such that

$$Y_1 = \begin{pmatrix} -(J_1^y)^{-1} \\ 0 \end{pmatrix}, \quad \text{and} \quad Z_1 = \begin{pmatrix} -(J_1^y)^{-1} J_1^u \\ I \end{pmatrix}.$$

Substituting for Δy in the definition of the QP subproblem gives

$$\begin{array}{ll} \underset{\Delta u \in \mathbb{R}^{n_2}}{\text{minimize}} & (g + HY_1 c_1)^T Z_1 \Delta u + \frac{1}{2} \Delta u^T Z_1^T H Z_1 \Delta u \\ \text{subject to} & J_2 Z_1 \Delta u \geq -J_2 Y_1 c_1 - c_2, \end{array} \tag{21}$$

which is a problem involving only the control variables Δu. Once Δu has been calculated, the state increment is determined from (20). The dimensions of the Hessian and constraints for this problem are $n_2 \times n_2$ and $m_2 \times n_2$, respectively. In the common situation where the number of control variables is significantly smaller than the number of state variables, then $n_2 \ll n$, and the dimension of this QP is considerably smaller than that of (19). Tanartkit and Biegler [33] propose forming the matrices $Z_1^T H Z_1$ and $J_2 Z_1$ explicitly and solving the subproblem (21) using a dense QP algorithm. A feature of this method is that any additional structure in the derivatives matrix J_2 is lost (the extreme case of this is when the inequalities $c_2(y,u)$ consist of only simple bounds. In this case the bounds are transformed into dense general inequalities.

Other approaches involve solving (21) with different choices for $g + HY_1 c_1$ (see, e.g., Beltracchi and Gabriele [2] and Schultz [31]). A structured problem of the form (21) lies at the heart of many methods for optimal control. Dennis, Heinkenschloss and Vicente [10] propose a trust-region SQP method for the case where the only inequalities are simple bounds.

5.2 Using approximate Jacobians

A disadvantage of the straightforward implementation of multiple shooting is that it is necessary to compute the n_y^2 Jacobian elements at each major iteration. Gill, Jay and Petzold [15] propose an algorithm that has the numerical complexity of single shooting and the stability and robustness of multiple shooting. The first step is to transform the nonlinear constraints of (18) and define the problem

$$\begin{array}{ll} \underset{y\in I\!R^{n_1}, u\in I\!R^{n_2}}{\text{minimize}} & F(y,u) \\ \text{subject to} & c_1^M(y,u) = 0, \quad c_2^M(y,u) \geq 0, \end{array} \tag{22}$$

where the m_1-vector $c_1^M(y,u)$ and m_2-vector $c_2^M(y,u)$ form elements of the vector c such that

$$c^M(x) = M(x)c(x), \quad \text{with} \quad M(x) = \begin{pmatrix} (J_1^y)^{-1} & 0 \\ -J_2^y(J_1^y)^{-1} & I \end{pmatrix}.$$

The idea is now to use an SQP algorithm to solve the problem (22). The only modification to the standard SQP strategy is that an *approximate* Jacobian $\widetilde{J}$ is used in place of $J(x)$. The approximation is given by

$$\widetilde{J}(x) = M(x)J(x) = \begin{pmatrix} I & (J_1^y)^{-1}J_1^u \\ 0 & J_2^u - J_2^y(J_1^y)^{-1}J_1^u \end{pmatrix}.$$

The entries for the unit columns of this Jacobian are available with no work. Moreover, the products $(J_1^y)^{-1}J_1^u$ used to define the other columns can be found with single-shooting complexity (but multiple shooting stability) using directional sensitivity techniques (see Maly and Petzold [26]).

The original variables of (18) are not subject to the transformation, which implies that only minor modifications need to be made to general-purpose SQP methods in order to accommodate the approximate Jacobian (one implementation is based on the SQP code SNOPT of Gill, Murray and Saunders [16]). In addition, linear constraints and selected nonlinear inequalities can be left untransformed, allowing their structure to be exploited by the solver in the usual way. For more details, see Gill, Jay and Petzold [15].

Acknowledgment

We extend sincere thanks to Laurent Jay and Linda Petzold for many discussions on ordinary differential equations.

REFERENCES

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

[2] J. T. Beltracchi and G. A. Gabriele. An investigation of using a RQP based method to calculate parameter sensitivity derivatives. In *Recent advances in multidisciplinary analysis and optimization.* NASA conference publication 3031, Part 2, proceedings of a symposium held in Hampton, Virginia, September 28–30, 1988, 1988.

[3] J. T. Betts. Issues in the direct transcription of optimal control problems to sparse nonlinear programs. In R. Bulirsch and D. Kraft, editors, *Control Applications of Optimization*, volume 115 of *Internat. Ser. Numer. Math.*, pages 3–17, Basel, 1994. Birkhäuser.

[4] J. T. Betts and P. D. Frank. A sparse nonlinear optimization algorithm. *J. Optim. Theory and Applics.*, 82:519–541, 1994.

[5] J. T. Betts and W. P. Huffman. The application of sparse nonlinear programming to trajectory optimization. *J. of Guidance, Control, and Dynamics*, 14:338–348, 1991.

[6] K. E. Brenan. Differential-algebraic equations issues in the direct transcription of path constrained optimal control problems. *Ann. Numer. Math.*, 1(1–4):247–263, 1994.

[7] O. Buchauer, P. Hiltmann, and M. Kiehl. Sensitivity analysis of initial value problems with application to shooting techniques. *Numer. Math.*, 67:151–159, 1994.

[8] R. Bulirsch, E. Nerz, H. J. Pesch, and O. von Stryk. Combining direct and indirect methods in optimal control: Range maximization of a hang glider. In R. Bulirsch, A. Miele, J. Stoer, and K. H. Well, editors, *Calculus of Variations, Optimal Control Theory and Numerical Methods.*, volume 111 of *Internat. Ser. Numer. Math.*, pages 273–288, Basel, 1993. Birkhäuser.

[9] C. Büskens and H. Maurer. Sensitivity analysis and real-time control of nonlinear optimal control systems via nonlinear programming. In R. Bulirsch, L. Bittner, W. Schmidt, and K. Heier, editors, *12th Conference on Variational Calculus, Optimal Control and Applications*, International Series of Numerical Mathematics, Basel, 1997. Birkhäuser.

[10] J. E. Dennis, Jr., M. Heinkenschloss, and L. N. Vicente. Trust-region interior SQP methods for a class of nonlinear programming problems. Technical Report 94-45 (revised 1996), Dept of Mathematical Sciences, Rice University, Houston, TX, 1994.

[11] E. D. Dickmanns and K. H. Well. Approximate solution of optimal control problems using third-order Hermite polynomial functions. In *Proc. 6th Technical Conference on Optimization Techniques*, Berlin, Heidelberg, New York, London, Paris and Tokyo, 1975. Springer Verlag.

[12] S. K. Eldersveld. *Large-scale sequential quadratic programming algorithms.* PhD thesis, Department of Operations Research, Stanford University, Stanford, CA, 1991.

[13] S. K. Eldersveld and P. E. Gill. Preconditioned limited-storage quasi-Newton methods for large-scale constrained optimization. Presented at the Fifth SIAM Conference on Optimization, Victoria, British Columbia, May 20–22, 1996.

[14] R. Fletcher. An ℓ_1 penalty method for nonlinear constraints. In P. T. Boggs, R. H. Byrd, and R. B. Schnabel, editors, *Numerical Optimization 1984*, pages 26–40, Philadelphia, 1985. SIAM.

[15] P. E. Gill, L. O. Jay, and L. Petzold. An SQP method for the optimization of dynamical systems. To appear.

[16] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. Numerical Analysis Report 97-1, Department of Mathematics, University of California, San Diego, La Jolla, CA, 1997.

[17] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Sparse matrix methods in optimization. *SIAM J. on Scientific and Statistical Computing*, 5:562–589, 1984.

[18] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press, London and New York, 1981. ISBN 0-12-283952-8.

[19] D. M. Gritsis, C. C. Pantelides, and R. W. H. Sargent. Optimal control of systems described by index two differential-algebraic equations. *SIAM J. Sci. Comput.*, 16:1349–1366, 1995.

[20] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I.* Springer Verlag, New York, Berlin and Heidelberg, second revised edition, 1993. ISBN 0387-56670-8.

[21] C. R. Hargraves and S. W. Paris. Direct trajectory optimization using nonlinear programming and collocation. *J. of Guidance, Control, and Dynamics*, 10:338–348, 1987.

[22] M. Kiehl. Parallel multiple shooting for the solution of initial value problems. *Parallel Comput.*, 20:275–295, 1994.

[23] D. Kraft. On converting optimal control problems into nonlinear programming problems. In K. Schittkowski, editor, *Computational Mathematical Programming*, NATO ASI Series F: Computer and Systems Sciences 15, pages 261–280. Springer Verlag, Berlin and Heidelberg, 1985.

[24] R. Lamour. A well posed shooting method for transferable DAEs. *Numer. Math.*, 59:815–829, 1991.

[25] G. Leitmann. *The Calculus of Variations and Optimal Control.* Mathematical Concepts and Methods in Science and Engineering, 24. Plenum Press, New York and London, 1981. ISBN 0-306-40707-8.

[26] T. Maly and L. R. Petzold. Numerical methods and software for sensitivity analysis of differential-algebraic systems. to appear in Applied Numerical Mathematics, 1996.

[27] C. C. Pantelides, R. W. H. Sargent, and V. S. Vassiliadis. Optimal control of multistage systems described by high-index differential-algebraic equations. *Internat. Ser. Numer. Math.*, 115:177–191, 1994.

[28] H. J. Pesch. Real-time computation of feedback controls for constrained optimal control problems. *Optimal Control Appl. Methods*, 10:129–171, 1989.

[29] L. Petzold, J. B. Rosen, P. E. Gill, L. O. Jay, and K. Park. Numerical optimal control of parabolic PDEs using DASOPT. In L. T. Biegler, T. F. Coleman, A. R. Conn, and F. N. Santosa, editors, *Large Scale Optimization with Applications,*

Part II: Optimal Design and Control, volume 93 of *IMA Volumes in Mathematics and its Applications*. Springer Verlag, Berlin, Heidelberg and New York, 1997.

[30] K. Schittkowski. NLPQL: A Fortran subroutine for solving constrained nonlinear programming problems. *Ann. Oper. Res.*, 11:485–500, 1985/1986.

[31] V. H. Schultz. *Reduced SQP methods for large-scale optimal control problems in DAE with application to path planning problems for satellite mounted robots.* PhD thesis, University of Heidelberg, 1996.

[32] V. H. Schulz, H.-G. Bock, and M. C. Steinbach. Exploiting invariants in the numerical solution of multipoint boundary value problems for DAE. submitted to SIAM J. Sci. Comput., 1996.

[33] P. Tanartkit and L. T. Biegler. Stable decomposition for dynamic optimization. *Ind. Eng. Chem. Res.*, pages 1253–1266, 1995.

[34] I.-B. Tjoa and L. T. Biegler. A reduced SQP strategy for errors-in-variables estimation. *Comput. Chem. Eng.*, 16:523–533, 1992.

[35] O. von Stryk. Numerical solution of optimal control problems by direct collocation. In R. Bulirsch, A. Miele, J. Stoer, and K. H. Well, editors, *Control Applications of Optimization*, volume 111 of *Internat. Ser. Numer. Math.*, pages 129–143, Basel, 1993. Birkhäuser.

[36] O. von Stryk and R. Bulirsch. Direct and indirect methods for trajectory optimization. *Ann. Oper. Res.*, 37(1–4):357–373, 1992.