Notes 13: NPSOL and SNOPT: SQP Methods

1   Overview

Here we consider *sequential quadratic programming methods* (SQP methods) for the general optimization problem

$$
\begin{array}{ll}
\text{NCB} & \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \phi(x) \\
& \text{subject to} \quad c(x) = 0, \quad \ell \le x \le u.
\end{array}
$$

As with the BCL and LCL approaches, SQP methods use a sequence of subproblems (each representing a *major iteration*) to find estimates (x_k, y_k) that reduce certain augmented Lagrangians. An important function of the subproblems is to estimate which inequalities in $\ell \le x \le u$ are *active*. If an active-set method is used, it will perform some number of *minor iterations* on each subproblem.

In BCL and LCL methods, the subproblem objective is the current augmented Lagrangian, which must be evaluated each minor iteration. In MINOS, for example, the reduced-gradient method that solves the LCL subproblems requires an average of 2 or 3 evaluations of the functions and gradients per minor iteration. This is acceptable if the function and gradient evaluations are cheap (as they are in the GAMS and AMPL environments) but may be prohibitive in other applications. For example, shape optimization in aerospace design requires the solution of a partial differential equation (PDE) each time $\phi(x)$ and $c(x)$ are evaluated, and an adjoint PDE to obtain gradients.

SQP methods economize by using *quadratic programming* (QP) subproblems to estimate the set of active constraints. Many minor iterations may be needed to solve the subproblems, but the functions and gradients are not being evaluated during that process. Instead, each subproblem solution is used to generate a search direction $(\Delta x, \Delta y)$ for a linesearch on the augmented Lagrangian (regarded as a function of x and y). Typically only 1–5 function and gradients values are needed in each linesearch, regardless of problem size. However, SQP methods are likely to need more *major* and *minor* iterations than LCL methods (i.e., more linear algebra).

2   An NPSOL application

NPSOL [9] is a successful "dense" SQP implementation. It has found wide use within the NAG Library and has proved remarkably robust and efficient on highly nonlinear optimization problems, particularly within the aerospace industry.

In the 1990s, the largest application we encountered was a trajectory optimization at McDonnell Douglas Space Systems (Huntington Beach) involving about 2000 constraints and variables, computing controls for the last breath-taking seconds of flight of the DC-Y, a proposed SSTO (Single-Stage-To-Orbit) manned spacecraft. The aim was to minimize the fuel needed as the rocket, descending nose-first to Earth, started rotating at an altitude of 2800 feet and slowed down using rocket thrust just time to land on its tail. NPSOL's solution showed that the landing maneuver could be accomplished using half the fuel estimated by classical techniques.

A second optimization was used to determine *when* the rotation should begin. With an extra constraint limiting acceleration to 3g (since a human crew would be aboard!), the optimal rotation altitude proved to be only 1400 feet—the height of the Empire State Building. Such touchdowns would be far more abrupt and astonishing than the Apollo moon landings. (More like the Lunar Excursion Module's take-off from the moon played in reverse!)

3 Quadratic approximations in a subspace

Linearized constraints are a key component of SQP methods, just as they are for LCL methods. They define a subspace in which it is reasonable to form a quadratic approximation to the augmented Lagrangian. Let (x_k, y_k) be the current solution estimate and ρ_k be the current penalty parameter. Also define $J_k \equiv J(x_k)$. The following functions are needed:

Linear approximation to $c(x)$:
$$\bar{c}_k(x) \;=\; c(x_k) + J_k(x - x_k)$$

Departure from linearity:
$$d_k(x) \;=\; c(x) - \bar{c}_k(x)$$

Modified augmented Lagrangian:
$$M_k(x) \;=\; \phi(x) - y_k^T d_k(x) + \rho_k \|d_k(x)\|^2$$
$$\nabla M_k(x) \;=\; g_0(x) - \big(J(x) - J_k\big)^T \widetilde{y}_k(x)$$
$$\nabla^2 M_k(x) \;=\; H_0(x) - \textstyle\sum_i (\widetilde{y}_k(x))_i H_i(x) + \rho_k \big(J(x) - J_k\big)^T \big(J(x) - J_k\big)$$
$$\widetilde{y}_k(x) \;\equiv\; y_k - \rho_k d_k(x).$$

The last four expressions simplify greatly when $x = x_k$. In particular, $d_k(x_k) = 0$ and $\widetilde{y}_k(x_k) = y_k$, and the terms involving $J(x)$ and ρ_k vanish:

$$M_k(x_k) \;=\; \phi(x_k)$$
$$\nabla M_k(x_k) \;=\; g_0(x_k)$$
$$\nabla^2 M_k(x_k) \;=\; H_0(x_k) - \textstyle\sum_i (y_k)_i H_i(x_k).$$

With $\Delta x \equiv x - x_k$, the quadratic approximation to $M_k(x)$ at x_k on the subspace $\bar{c}_k(x) = 0$ is therefore

$$Q_k(x) = \phi(x_k) + g_0(x_k)^T \Delta x + \tfrac{1}{2} \Delta x^T \nabla^2 M_k(x_k) \Delta x,$$

where the quadratic term may or may not be available.

4 NPSOL

Recall that the MINOS subproblem takes the form

$$\boxed{\;\text{LC}_k \qquad\qquad \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad M_k(x) = \phi(x) - y_k^T d_k(x) + \tfrac{1}{2}\rho_k \|d_k(x)\|^2 \\ \text{subject to } \bar{c}_k(x) = 0, \quad \ell \le x \le u.\;}$$

The stabilized LCL method adapts early subproblems judiciously to ensure global convergence, but ultimately the subproblems become LC_k and their solutions (x_k^*, y_k^*) converge rapidly, as predicted by Robinson (1972).

  The SQP methods in NPSOL and SNOPT are based on the same subproblem, but with the objective $M_k(x)$ replaced by a quadratic approximation $Q_k(x)$:

$$\boxed{\;\text{QP}_k \qquad\qquad \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad Q_k(x) = \phi(x_k) + g_0(x_k)^T \Delta x + \tfrac{1}{2}\Delta x^T H_k \Delta x \\ \text{subject to } \bar{c}_k(x) = 0, \quad \ell \le x \le u,\;}$$

where H_k is a quasi-Newton approximation to $\nabla^2 M_k(x)$ (perhaps with $\rho_k = 0$). To achieve global convergence, QP_k is regarded as providing a *search direction* $(\Delta x, \Delta y)$ along which the augmented Lagrangian $M_k(x)$ (with $\rho_k \ge 0$) may be reduced as a function of both primal *and* dual variables.

  Note that the true Hessian of $M_k(x)$ is zero in rows and columns associated with linear variables. In NPSOL, slack variables are therefore excluded from H_k, and in SNOPT both

slacks and other linear variables in x are excluded. The quasi-Newton Hessian approximations are therefore of the form

$$H_k = \begin{pmatrix} \bar{H}_k & \\ & 0 \end{pmatrix},$$

where $\bar{H}_k$ is positive definite. The positive-definiteness provides an intrinsic bound on the relevant parts of Δx, improving the validity of the quadratic model.

4.1   NPSOL's merit function

Let (x_k^*, y_k^*) be the primal and dual solution of QP_k. NPSOL forms the search direction

$$\begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} x_k^* - x_k \\ y_k^* - y_k \end{pmatrix}$$

and seeks a steplength α such that moving to the point $(x, y) \equiv (x_k + \alpha \Delta x, y_k + \alpha \Delta y)$ (where $0 < \alpha \le 1$) produces a sufficient decrease in the augmented Lagrangian

$$L(x, y, \rho_k) = \phi(x) - y^T c(x) + \tfrac{1}{2} \rho_k \|c(x)\|^2$$

relative to $L(x_k, y_k, \rho_k)$. Initially $\rho_k = 0$. Later, ρ_k is increased if necessary to ensure descent for the merit function. It may also be decreased a limited number of times, because a finite (unknown) value is always adequate for well defined problems, and because $\rho_k = 0$ becomes acceptable within some neighborhood of a solution to NCB. The global convergence properties of NPSOL (and the strategy for adjusting ρ_k) are described in Gill, Murray, Saunders, and Wright [11]. The QP subproblems are assumed to be feasible, and to be solved accurately.

4.2   Hessian updates

A dense triangular factorization $H_k = R_k^T R_k$ is maintained in NPSOL. After the linesearch, a BFGS update of the form

$$R_{k+1} = Q_k R_k (I + r_k s_k^T), \qquad Q_k \text{ orthogonal}$$

is made to improve the quality of the Hessian approximation within some space. Various choices are possible for the vectors r_k and s_k. They reflect the change in x and the change in gradient of some updated Lagrangian $L_k(x, y_{k+1}, \rho_{k+1})$ defined with the latest estimate of y (but with $\rho_{k+1} = 0$ where possible).

NPSOL uses the least-squares solver LSSOL [5] for the subproblems QP_k. LSSOL deals directly with the triangular factor R_k. It is unique in avoiding the squared condition number that other solvers fall prey to with Hessians of the form $R_k^T R_k$.

5   SNOPT

The reliability of NPSOL and LSSOL allowed them to be applied to increasingly large problems (particularly at McDonnell Douglas Space Systems, Huntington Beach, CA during the 1990s). However, LSSOL uses orthogonal transformations of $J(x_k)$ to solve the QP subproblems, and as problem size increases the total cost becomes dominated by the associated linear algebra.

To a large extent, SNOPT is a large-scale SQP implementation that combines the virtues of MINOS and NPSOL. Sparse Jacobians are handled as in the reduced-gradient method (using LUSOL to maintain a sparse basis factorization), and the same merit function as in NPSOL ensures global convergence.

The primary innovation is a method for treating the QP Hessian in sparse form. Also, infeasible problems NCB are dealt with more methodically by including an ℓ_1 penalty term within the problem definition (where e is a vector of ones):

$$
\begin{array}{ll}
\text{NCB}(\gamma) & \displaystyle\operatorname*{minimize}_{x,\,v,\,w} \quad \phi(x) + \gamma e^T(v + w) \\
& \text{subject to} \ \ c(x) + v - w = 0, \quad \ell \le x \le u, \quad v, w \ge 0.
\end{array}
$$

Initially $\gamma = 0$, but it is set to $\gamma \equiv 10^4(1 + \|g_0(x_k)\|)$ if one of the QP subproblems $\text{QP}_k(0)$ (see below) proves to be infeasible, or if the norm of the dual variables for $\text{QP}_k(0)$ becomes large: $\|y\|_\infty > 10^4$. SNOPT is then in *elastic mode* for subsequent major iterations.

Optimization continues with that value of γ until an optimal solution of $\text{NCB}(\gamma)$ has been obtained. If the nonlinear constraints are not satisfied ($\|c(x)\|_\infty$ not sufficiently small), γ is increased and optimization continues. If γ reaches an allowable maximum value (default 10^{10}) and the nonlinear constraints are still violated significantly, the problem is declared infeasible. At this stage, SNOPT has essentially minimized $\|c(x)\|_1$.

5.1   The QP subproblem

If SNOPT is in elastic mode, the SNOPT subproblems include the ℓ_1 penalty terms verbatim:

$$
\begin{array}{ll}
\text{QP}_k(\gamma) & \displaystyle\operatorname*{minimize}_{x,\,v,\,w} \quad \phi(x_k) + g_0(x_k)^T\Delta x + \tfrac{1}{2}\Delta x^T H_k \Delta x + \gamma e^T(v + w) \\
& \text{subject to} \ \ \bar c_k(x) + v - w = 0, \quad \ell \le x \le u, \quad v, w \ge 0,
\end{array}
$$

with $\gamma > 0$. But before elastic mode is activated, γ and v, w are zero.

5.2   SQOPT

SNOPT includes a sparse QP solver SQOPT, which employs a two-phase active set algorithm to achieve feasibility and optimality. SQOPT implements *elastic programming* implicitly. Any specified bounds on the variables or constraints may be elastic. The associated variables or slacks are given a piecewise linear objective function that penalizes values outside the "real" bounds.

If $\gamma > 0$ in $\text{NCB}(\gamma)$ (perhaps because previous QP subproblems were infeasible), SNOPT defines elastic bounds on the slacks associated with linearized nonlinear constraints. SQOPT then solves each $\text{QP}_k(\gamma)$ with v and w implemented implicitly.

Separately, SNOPT can use SQOPT to find a point close to a user-provided starting point x_0. One option is to set $\ell_j = u_j = (x_0)_j$ for the nonlinear variables and specify that those bounds are elastic. (Linear variables retain their true bounds.) SQOPT will then solve the "proximal point problem"

$$
\begin{array}{ll}
\text{PP1} & \displaystyle\operatorname*{minimize}_{x \in \mathbb{R}^n} \quad \|x - x_0\|_1 \\
& \text{subject to} \ \ \text{the linear constraints and (modified) bounds.}
\end{array}
$$

Alternatively, SNOPT's PP2 option asks SQOPT to minimize $\frac{1}{2}\|x - x_0\|_2^2$ subject to the linear constraints and true bounds. For further details of SNOPT and SQOPT, see [6, 8, 7].

5.3   Limited-memory Hessian updates

For moderate-sized problems, SNOPT maintains a dense Hessian approximation similar to that in NPSOL. For large problems with thousands of nonlinear variables, a *limited-memory* procedure is used to represent H_k. At certain iterations $k = r$, say, a positive-definite

diagonal estimate $H_r = R_r^2$ is used. In particular, R_0 is a multiple of the identity. Up to ℓ quasi-Newton updates (default $\ell = 10$) are accumulated to represent the Hessian as

$$H_k = R_k^T R_k, \qquad R_k = R_r \prod_{j=r}^{k-1} (I + r_j s_j^T).$$

SQOPT accesses H_k by requesting products of the form $H_k v$. The work and storage required grows linearly with $k - r$. When $k = r + \ell$, the diagonals of R_k are computed and saved to form the next R_r. Storage is then "reset" by discarding the vectors $\{r_j, s_j\}$.

   More elaborate limited-memory methods are known; notably Nocedal et al. [16, 2]. The simple form above has an advantage for problems with purely linear constraints: the reduced Hessian required by the reduced-gradient implementation in SQOPT can be *updated* between major iterations. This is important on large problems with many superbasic variables. As the SQP iterations converge, very few minor iterations are needed to solve QP$_k$ and the cost becomes dominated by the formation of $Z^T H_k Z$ and its factors, but when the constraints are linear, this expense is avoided. (The LU factors of the basis can also be retained.)

   The thesis research of Andrew Bradley [1] led to a new limited-memory quasi-Newton implementation for SNOPT7. A circular buffer is used to hold the latest pairs of update vectors $\{s_j, y_j\}$ (recording the recent changes in x and the gradient of the Lagrangian), and the initial diagonal Hessian approximation H_0 is continually updated. The resulting LMCBD procedure (limited-memory circular buffer with diagonal update) produces a robust Hessian approximation that on average outperforms SNOPT's previous limited-memory implementation.

5.4   Quasi-Newton and CG

To avoid forming and factorizing the reduced Hessian at the start of each major iteration, SNOPT can use a quasi-Newton option in SQOPT. It maintains dense triangular factors $R^T R \approx Z^T H_k Z$ analogous to the RG method in MINOS. A further option is to use the Conjugate-Gradient method to solve the reduced-Hessian system $Z^T H_k Z \Delta x_S = -Z^T g$. This often converges in remarkably few iterations even when there are many thousands of superbasic variables. The "diagonal plus low-rank" structure of H_k plus the large identity matrix in Z must be largely responsible, though it would be easier to explain if the diagonal part $H_r = R_r^2$ were always the identity.

5.5   QP solvers for many degrees of freedom

Hanh Huynh's thesis research [15] included the development of a convex QP solver QPBLU that periodically factorizes the current KKT matrix

$$K_0 = \begin{pmatrix} H_0 & W^T \\ W & \end{pmatrix}, \qquad H_0 \text{ diagonal,}$$

and uses block-LU updates to accommodate up to about 100 active-set changes to W as well as a few BFGS updates to H_0. The aim was to handle many degrees of freedom with a direct method rather than the CG option above (allowing W to be large with many more columns than rows). QPBLU was implemented in Fortran 95 and designed to use any available sparse solver to factorize the current K_0. It included interfaces to LUSOL [10], MA57 [4], PARDISO [18], SuperLU [3], and UMFPACK [19]. Such solvers are assumed to give a factorization $K_0 = LU$ and to permit separate solves with L, L^T, U, and U^T. For example, MA57's symmetric factorization $K_0 = LDL^T$ was regarded as $K_0 = L(DL^T)$. At the time, PARDISO did not separate its factors, so QPBLU used "L" $= I$ and "U" $= K_0$.

   This work was continued by Chris Maes for his thesis research [17]. A new QP solver QPBLUR was developed with modules from QPBLU serving as starting point. QPBLUR used a BCL method to solve a sequence of *regularized* convex QP subproblems (in order to solve the given convex QP). It was incorporated into SNOPT7 as an alternative to SQOPT. The aim was to allow use of the most advanced (possibly parallel) linear system solvers within

an active-set QP solver and hence within an SQP algorithm. In May 2010, Chris installed a more recent version of PARDISO that allowed separate solves with L and U and optionally used multiple CPUs during its Factor phase. The intention was for SNOPT to use SQOPT while the number of superbasic variables remains below 2000 (say), then switch to QPBLUR if necessary. With its warm-start capability, QPBLUR bridged the gap between null-space active-set methods and full-space interior methods. Comprehensive numerical comparisons are given in [17].

Meanwhile, Elizabeth Wong for her thesis research [20] developed a convex and non-convex QP solver icQP, based again on block-LU factors of KKT systems (but without regularization). More recently, Gill and Wong developed SQIC [14, 13, 12], a further block-LU/KKT-based solver for convex and nonconvex QP intended for use within SNOPT9 (a Fortran 2003 implementation of SNOPT). SQIC manages the transition from a reduced-Hessian method to a block-LU method for updating the active set when the number of superbasic variables becomes large.

References

[1] A. M. Bradley. *Algorithms for the Equilibration of Matrices and Their Application to Limited-Memory Quasi-Newton Methods*. PhD thesis, ICME, Stanford University, May 2010.

[2] R. H. Byrd, J. Nocedal, and R. B. Schnabel. Representations of quasi-Newton matrices and their use in limited-memory methods. *Math. Program.*, 63:129–156, 1994.

[3] J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li, and J. W. H. Liu. A supernodal approach to sparse partial pivoting. *SIAM J. Matrix Anal. Appl.*, 20(3):720–755, 1999.

[4] I. S. Duff. MA57: a Fortran code for the solution of sparse symmetric definite and indefinite systems. *ACM Trans. Math. Software*, 30(2):118–144, 2004. See `ldl` in MATLAB.

[5] P. E. Gill, S. J. Hammarling, W. Murray, M. A. Saunders, and M. H. Wright. User's guide for LSSOL (Version 1.0): a Fortran package for constrained linear least-squares and convex quadratic programming. Report SOL 86-1, Department of Operations Research, Stanford University, CA, 1986.

[6] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review*, 47(1):99–131, 2005. SIGEST article.

[7] P. E. Gill, W. Murray, and M. A. Saunders. User's guide for SQOPT 7: Software for large-scale linear and quadratic programming. `http://www.scicomp.ucsd.edu/~peg` (see Software), 2006.

[8] P. E. Gill, W. Murray, M. A. Saunders, and E. Wong. User's Guide for SNOPT 7.7: Software for Large-Scale Nonlinear Programming. Center for Computational Mathematics Report CCoM 18-1, Department of Mathematics, University of California, San Diego, La Jolla, CA, 2018.

[9] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. User's guide for NPSOL (Version 4.0): a Fortran package for nonlinear programming. Report SOL 86-2, Department of Operations Research, Stanford University, Stanford, CA, 1986.

[10] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Maintaining LU factors of a general sparse matrix. *Linear Algebra and its Applications*, 88/89:239–270, 1987.

[11] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Some theoretical properties of an augmented Lagrangian merit function. In P. M. Pardalos, editor, *Advances in Optimization and Parallel Computing*, pages 101–128. North Holland, North Holland, 1992.

[12] P. E. Gill and E. Wong. Numerical results for SQIC: Software for large-scale quadratic programming. Report CCoM 14-4, Center for Computational Mathematics, University of California, San Diego, La Jolla, CA, 2014.

[13] P. E. Gill and E. Wong. User's guide for SQIC: Software for large-scale quadratic programming. Report CCoM 14-2, Center for Computational Mathematics, University of California, San Diego, La Jolla, CA, 2014.

[14] P. E. Gill and E. Wong. Methods for convex and general quadratic programming. *Math. Prog. Comp.*, 7:71–112, 2015.

[15] H. M. Huynh. *A Large-scale Quadratic Programming Solver Based On Block-LU Updates of the KKT System*. PhD thesis, SCCM, Stanford University, Sep 2008.

[16] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Program.*, 45:503–528, 1989.

[17] C. M. Maes. *A Regularized Active-Set Method for Sparse Convex Quadratic Programming*. PhD thesis, ICME, Stanford University, Nov 2010.

[18] PARDISO parallel sparse solver. `http://www.pardiso-project.org`.

[19] UMFPACK solver for sparse $Ax = b$. `http://faculty.cse.tamu.edu/davis/suitesparse.html`.

[20] E. Wong. *Active-Set Methods for Quadratic Programming*. PhD thesis, Department of Mathematics, University of California, San Diego, June 2011.

Summary

The problem

$$
\begin{array}{ll}
\text{NCB} & \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \phi(x) \\
& \text{subject to} \quad c(x) = 0, \quad \ell \le x \le u.
\end{array}
$$

Definitions

Major iterations	Solving a sequence of subproblems
Minor iterations	Solving each subproblem
Augmented Lagrangian	$L_k(x) = \phi(x) - y_k^T c(x) + \frac{1}{2}\rho_k \|c(x)\|^2$
Linear approximation at x_k	$\bar{c}_k(x) = c(x_k) + J_k(x - x_k)$
Departure from linearity	$d_k(x) = c(x) - \bar{c}_k(x)$
Modified augmented Lagrangian	$M_k(x) = \phi(x) - y_k^T d_k(x) + \frac{1}{2}\rho_k \|d_k(x)\|^2$
Quadratic approx to $M_k(x)$	$Q_k(x) = g_0(x_k)^T \Delta x + \frac{1}{2}\Delta x^T H_k \Delta x$
Elastic departure from linearity	$d_k(x,v,w) = c(x) - \bar{c}_k(x) - v + w$
Modified augmented Lagrangian	$M_k(x,v,w) = \phi(x) - y_k^T d_k(x,v,w) + \frac{1}{2}\rho_k \|d_k(x,v,w)\|^2$
Augmented Lagrangian	$L_k(x,r) = \phi(x) + y_k^T r + \frac{1}{2}\rho_k \|r\|^2, \ \ c(x) + r = 0$

The big picture

	Code	Method	Subproblem objective	Subproblem constraints
LANCELOT B	f90	BCL	$L(x, y_k, \rho_k)$	bounds
Robinson (1972)		LCL	$M(x, y_k, 0)$	$\bar{c}_k(x) = 0$, bounds
MINOS	f77	LCL	$M(x, y_k, \rho_k)$	$\bar{c}_k(x) = 0$, bounds
NPSOL	f77	SQP	$Q_k(x)$	$\bar{c}_k(x) = 0$, bounds
DNOPT, SNOPT	f77	SQP	$Q_k(x) + \sigma_k \|\bar{c}_k(x)\|_1$	bounds
			$\equiv Q_k(x) + \sigma_k e^T(v+w)$	$\bar{c}_k(x) + v - w = 0$, bounds
Knossos	f90	sLCL	$M_k(x,v,w) + \sigma_k \|\bar{c}_k(x)\|_1$	bounds
			$\equiv M_k(x,v,w) + \sigma_k e^T(v+w)$	$\bar{c}_k(x) + v - w = 0$, bounds
	AMPL	NCL	$L_k(x,r)$	$c(x) + r = 0$, bounds