

## 1 Origins

The first version of MINOS (Murtagh and Saunders [12]) was designed to solve *linearly constrained* optimization problems of the form

LC1	$\begin{aligned} & \underset{x \in \mathbb{R}^n}{\text{minimize}} && \phi(x) \\ & \text{subject to} && \ell \leq \begin{pmatrix} x \\ Ax \end{pmatrix} \leq u, \end{aligned}$
-----	---

where  $\phi(x)$  is a smooth function (ideally involving only some of the variables), and  $A$  is a sparse  $m \times n$  matrix as in a typical linear program. The gradients of  $\phi(x)$  were assumed to be available, but no use could be made of second derivatives.

Algorithms for dense LC problems had been proposed, including the Gradient-Projection Method of Rosen [15], the Reduced-Gradient Method of Wolfe [17, 18], the Variable-Reduction Method of McCormick [9], and the active-set methods of Gill and Murray [7, 8]. The focus was on constraints  $Ax \geq b$ ,  $m > n$  with projections onto the set of active constraints requiring factorizations of matrices with changing *row dimensions*. The only large-scale implementation was that of Buckley [2].

The particular Reduced-Gradient/Variable-Reduction algorithm developed in MINOS was a natural way to combine the established *sparse-matrix technology of simplex implementations* with the fairly recent Quasi-Newton/Variable-Metric approach to *dense unconstrained optimization* [3]. In contrast to Buckley's  $Ax \geq b$  focus, MINOS follows Mathematical Programming Systems in working with the equality form, where  $A$  and  $x$  here include a full set of slack variables:

LC	$\begin{aligned} & \underset{x \in \mathbb{R}^n}{\text{minimize}} && \phi(x) \\ & \text{subject to} && Ax = b, \quad \ell \leq x \leq u. \end{aligned}$
----	---

We use the term Reduced-Gradient (RG) in the context of LC optimization, even though Variable-Reduction conveys the correct idea. Similarly, we use Quasi-Newton (QN) rather than Variable-Metric for unconstrained optimization. (At least this avoids using Variable as a noun in one context and an adjective in the other. Quasi-Newton also conveys the idea of approximate second derivatives.)

Although explicit *sparse QN methods* have been developed (Toint [16]), they are normally not useful for the Reduced-Gradient Method because the underlying *exact reduced Hessian* is usually not sparse. Instead, future research could focus on *limited-memory QN methods* (Nocedal [14]) and/or Conjugate-Gradient methods to extend the RG Method to problems with very large reduced Hessians.

## 2 Concave Objectives

If the objective function  $\phi(x)$  happens to be *concave*, an active-set algorithm could proceed exactly like the Simplex Method. Whenever a nonbasic variable is chosen to be moved from its current value, it would want to continue moving as far as possible until some variable reached a bound. Thus, all solutions of problem LC lie at a vertex of the feasible region, the same as for linear programs. However, there are likely to be many *local minima*, most of which will be of little interest.

The classic example is continually rediscovered by modelers trying to impose integer constraints. If the first  $N$  variables are supposed to be either 0 or 1, one might consider the function

$$\phi(x) = \sum_{j=1}^N x_j(1 - x_j)$$

with bounds  $0 \leq x_j \leq 1$ . Indeed, MINOS and other nonlinear solvers would satisfy the integer requirement and reduce  $\phi(x)$  to its *global* minimum value of zero. Thus a *feasible integer solution* is readily obtained. However, there could be  $2^N$  combinations of  $x_j = 0$  or 1 with the same optimum objective value. The constraints  $Ax = b$ ,  $\ell \leq x \leq u$  would need to exclude most of those combinations if the chosen one were to be useful.

In general,  $\phi(x)$  would include other terms as the “real objective function”, and the approach is likely to generate one of many feasible integer *local* optima.

On the other hand, significant success has been achieved at SOL by Murray and Ng [10, 13] with the *global smoothing* approach of solving a sequence of subproblems of the form

$\begin{aligned} \text{LC}(\gamma, \mu) \quad & \underset{x}{\text{minimize}} && \phi(x) + \gamma \sum x_j(1 - x_j) - \mu \sum \ln x_j(1 - x_j) \\ & \text{subject to} && Ax = b, \quad 0 \leq x \leq 1, \end{aligned}$
---

in which  $\gamma, \mu > 0$  and  $\mu$  is initially *large* in order to make the combined objective convex. As  $\mu$  is reduced, the concave  $\gamma$  term becomes increasingly effective. Thus, *concave objective functions* and *local LC solvers* have a promising new use. (Note that a specialized LC solver is probably needed. In order to deal effectively with saddle points it would ideally use second derivatives.)

Further success has been achieved at SOL with a local LC solver on problems with many discrete and continuous variables, using local improvement to solutions from a series of LC subproblems (Murray and Shanbhag [11]).

## 3 Superbasic Variables

With nonlinear objective functions, we generally do not expect an optimal point to be a basic solution. For example, the problem

$$\min \phi(x) = x_1^2 + x_2^2 \quad \text{subject to} \quad x_1 + x_2 = 2, \quad 0 \leq x \leq 3,$$

has a unique optimum at  $(1, 1)$  with  $\phi(x) = 2$ , whereas both possible basic solutions give  $\phi(x) = 4$ . If we started a simplex-type method at the basic solution  $(2, 0)$  and began to increase the (only) nonbasic variable  $x_2$ , the objective function would start to decrease as desired. However, it would reach a minimum *before*  $x_1$  reached 0 or  $x_2$  reached its upper bound of 3.

One of the innovations in MINOS was to extend the concept of basic and nonbasic variables by introducing a set of *superbasic variables* to include variables like  $x_2$  that move away from their bound but end up in “mid-air” without causing some other variable to reach a bound. The constraints  $Ax = b$  are partitioned as follows, with  $B$  nonsingular as for Simplex:

$$Ax = \begin{array}{|c|c|c|} \hline & B & S & N \\ \hline & m & s & n - m - s \\ \hline \end{array} \begin{pmatrix} x_B \\ x_S \\ x_N \end{pmatrix} = b.$$

### 3.1 The Size of $S$

The number of superbasic variables is a measure of the *nonlinearity* of the problem in several ways:

- If the objective function is linear (so that problem LC is a linear program), the Reduced-Gradient Method in MINOS unknowingly mimics the Simplex Method. The value of  $s$  oscillates between 0 and 1.
- On nonlinear programs with only  $n_{NL}$  variables appearing nonlinearly in the objective,  $s$  need not be larger than  $n_{NL} + 1$ .
- Even if all variables appear nonlinearly, the objective may be “nearly linear” and  $s$  could remain small. For example,  $\phi(x) = c^T x + \psi(x)$  might have  $c$  significantly larger than  $\nabla\psi(x)$  at all feasible points  $x$ , or  $\nabla\psi(x)$  might be nearly constant.

There is no vital difference between basic and superbasic variables. All we need is a nonsingular  $B$ , ideally not too ill-conditioned. MINOS performs two kinds of “basis repair” at times in order to preserve this situation:

*BR factorization* checks the rank of  $B$  (using LUSOL’s TRP option) and may replace some of its columns by unit vectors belonging to slacks in  $S$  or  $N$ .

*BS factorization* may shuffle the columns of  $(B \ S)$ , using LU factors of  $(B \ S)^T$  to choose a better-conditioned  $B$ .

### 3.2 Optimality Conditions

A triple  $(x, y, z)$  satisfies the *first-order KKT conditions* for problem LC if

$$Ax = b, \tag{1}$$

$$z = g(x) - A^T y, \tag{2}$$

$$\min(x - \ell, z) = 0, \tag{3}$$

$$\min(u - x, -z) = 0, \tag{4}$$

where  $g(x) = \nabla\phi(x)$  is the gradient of the objective, and  $z$  is a vector of *reduced gradients*. Note that (3)–(4) enforce the bounds  $\ell \leq x \leq u$  as well as the more familiar complementarity conditions  $(x - \ell)^T z = (u - x)^T z = 0$ .

We assume that  $Ax = b$ ,  $B^T y = g_B$  and  $z = g - A^T y$  can be satisfied with sufficient accuracy throughout (so that  $z_B \approx 0$  by construction). A point  $(x, y, z)$  is regarded as feasible and optimal to within tolerances  $\delta_P, \delta_D$  ( $= 10^{-6}$ , typically) if

$$\min(x - \ell, u - x) \geq -\delta_P, \tag{5}$$

$$\min(x - \ell, z) \geq -\delta_D, \tag{6}$$

$$\min(u - x, -z) \geq -\delta_D. \tag{7}$$

### 3.3 Suboptimization

As in the Simplex Method, the nonbasic variables are temporarily frozen at their current value. The variables in  $B$  and  $S$  are then optimized (to some extent) before another nonbasic variable is chosen to be part of  $S$ . We may think of this as suboptimization on the problem

<div style="display: flex; justify-content: space-between;"> <div style="width: 15%;">BSprob</div> <div style="width: 85%;"> <math display="block">\begin{aligned} &amp; \underset{x_B, x_S}{\text{minimize}} &amp;&amp; \phi(x) \\ &amp; \text{subject to} &amp;&amp; (B \ S) \begin{pmatrix} x_B \\ x_S \end{pmatrix} = b - Nx_N, \\ &amp; &amp;&amp; \ell \leq x \leq u, \quad x_N = x_N. \end{aligned}</math> </div> </div>
---

Recall that Phase 1 Simplex performs just *one* iteration on the current modified problem before defining the next modified problem. Similarly here—we do just one iteration before checking if  $(B, S, N)$  should be redefined:

- If a superbasic variable reaches a bound, it is moved from  $S$  to  $N$ .
- If a basic variable reaches a bound, a judicious column of  $S$  is chosen to enter  $B$ , and the basic variable is moved into  $N$ .
- If BSprob is sufficiently optimized, a nonbasic variable  $x_s$  is chosen to move from  $N$  to  $S$ , as in a Simplex pricing operation.

The *active-set strategy* in MINOS is designed to avoid zig-zagging (bouncing on and off the same set of constraints) while limiting the effort devoted to any particular BSprob. Note that BSprob is optimal if the largest superbasic reduced gradient satisfies  $\|z_S\|_\infty \leq \delta_D$ . A dynamic tolerance  $\delta_S$  is therefore defined and used as follows:

- Whenever a nonbasic variable  $x_s$  is moved from  $N$  to  $S$  to define a new BSprob, the dynamic tolerance is defined to be  $\delta_S = 0.2|z_s|$ , say (where 0.2 is the default **Subspace tolerance**—it may be specified at run-time).
- If  $|z_s|$  is not sufficiently large ( $|z_s| \leq 1.1\|z_S\|_\infty$ , say) the chosen nonbasic variable is *not* moved to  $S$ . The dynamic tolerance is reduced ( $\delta_S \leftarrow 0.9\|z_S\|$ ) and BSprob remains the same as before.
- The current BSprob is optimized until  $\|z_S\|_\infty \leq \max\{\delta_S, \delta_D\}$ . Only then is  $z_N$  evaluated to suggest a new variable for  $S$ .

### 3.4 Unconstrained Optimization

A single iteration on problem BSprob requires a search direction  $(\Delta x_B, \Delta x_S, \Delta x_N)$  satisfying

$$B\Delta x_B + S\Delta x_S = 0, \quad \Delta x_N = 0.$$

This may be thought of as finding a search direction for an *unconstrained* problem involving the superbasic variables  $x_S$ . It is equivalent to defining

$$\Delta x = Z\Delta x_S, \quad \text{where} \quad Z = \begin{pmatrix} -B^{-1}S \\ I \\ 0 \end{pmatrix}. \quad (8)$$

For this “reduced-gradient operator”  $Z$ , we have

$$\phi(x + Z\Delta x_S) = \phi(x) + g^T Z\Delta x_S + \frac{1}{2}\Delta x_S^T Z^T H Z\Delta x_S + \dots,$$

where  $g$  and  $H$  are the current values of  $\nabla\phi(x)$  and  $\nabla^2\phi(x)$ . Thus, the objective function behaves like an unconstrained function of the superbasic variables with gradient  $Z^T g(x)$  and Hessian  $Z^T H(x)Z$ . In MINOS we use a nonsingular upper-triangular matrix  $R$  to maintain a quasi-Newton approximation

$$R^T R \approx Z^T H(x)Z. \quad (9)$$

For each subproblem BSprob, the primary steps to generate a search direction are as follows:

- Solve  $R^T R \Delta x_S = -Z^T g$  (where  $B^T y = g_B$  and  $Z^T g = g_S - S^T y$ ).
- Define  $\Delta x = Z\Delta x_S$  (i.e., solve  $B\Delta x_B = -S\Delta x_S$ ).
- Find the maximum feasible steplength  $\alpha_{\max}$  such that  $\ell \leq x + \alpha_{\max}\Delta x \leq u$ .
- Perform a linesearch to find  $\alpha$ , an approximation to the step  $\alpha^*$  that minimizes  $\phi(x + \alpha^*\Delta x)$  within the interval  $0 < \alpha^* \leq \alpha_{\max}$ .
- Perform a quasi-Newton update on  $R$  to account for the “unconstrained” optimization step, using  $\alpha$ ,  $\Delta x_S$ , and the change in reduced gradient  $Z^T g$ .
- If the linesearch encountered a bound ( $\alpha = \alpha_{\max}$ ), redefine BSprob and perform one or two “static” updates on  $R$ .

Note that operations with  $Z$  and  $Z^T$  involve solves with  $B$  and  $B^T$  as in the Simplex Method.

## 4 Utility

MINOS has been an effective large-scale solver (alongside CONOPT [4]) since the earliest days of GAMS [1] and AMPL [5, 6], especially for problems with linear constraints and cheap functions and gradients, and for cases where the number of superbasics variables remains below 2000 (say). Warm starts are straightforward.

Many other solvers are now accessible from GAMS and AMPL (including non-linear interior methods), and a few of them can use second derivatives.

## References

- [1] A. Brooke, D. Kendrick, and A. Meeraus. *GAMS: A User's Guide*. The Scientific Press, Redwood City, California, 1988.
- [2] A. Buckley. An alternative implementation of Goldfarb's minimization algorithm. *Math. Program.*, 8:207–231, 1975.
- [3] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ, 1983. Reprinted as Classics in Applied Mathematics 16, SIAM, Philadelphia, 1996.
- [4] A. Drud. CONOPT: A GRG code for large sparse dynamic nonlinear optimization problems. *Math. Program.*, 31:153–191, 1985.
- [5] Robert Fourer, David M. Gay, and Brian W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. The Scientific Press, South San Francisco, 1993.
- [6] Robert Fourer, David M. Gay, and Brian W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press / Brooks/Cole Publishing Company, second edition, 2002.
- [7] P. E. Gill and W. Murray. Newton-type methods for linearly constrained optimization. In P. E. Gill and W. Murray, editors, *Numerical Methods for Constrained Optimization*, pages 29–66. Academic Press, London, 1974.
- [8] P. E. Gill and W. Murray. Quasi-Newton methods for linearly constrained optimization. In P. E. Gill and W. Murray, editors, *Numerical Methods for Constrained Optimization*, pages 67–92. Academic Press, London, 1974.
- [9] G. P. McCormick. The variable-reduction method for nonlinear programming. *Management Science*, 17(3):146–160, 1970.
- [10] W. Murray and K.-M. Ng. Algorithms for global and discrete problems based on methods for local optimization. In P. M. Pardalos and H. E. Romeijn, editors, *Handbook of Global Optimization, Volume 2*, pages 87–113. Kluwer, Dordrecht, 2002.
- [11] W. Murray and V. V. Shanbhag. A local relaxation approach for the siting of electrical substations. *J. of Computational Optimization and Applications*, 33:7–49, 2006. COAP 2006 Best Paper Award.
- [12] B. A. Murtagh and M. A. Saunders. Large-scale linearly constrained optimization. *Math. Program.*, 14:41–72, 1978.
- [13] K.-M. Ng. *A Homotopy Algorithm for Nonlinear Discrete Problems*. PhD thesis, Dept of Management Science and Engineering, Stanford University, 2002.
- [14] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Math. Comp.*, 35:773–782, 1980.
- [15] J. B. Rosen. The gradient projection method for nonlinear programming. Part I: linear constraints. *SIAM J. Appl. Math.*, 8:181–217, 1960.
- [16] Ph. L. Toint. Sparsity exploiting quasi-Newton methods for unconstrained optimization. In L. C. W. Dixon, E. Spedicato, and G. P. Szego, editors, *Nonlinear Optimization: Theory and Algorithms*, pages 65–90. Birkhäuser, Boston, 1980.
- [17] P. Wolfe. The reduced gradient method. Unpublished manuscript, RAND Corporation, 1962.
- [18] P. Wolfe. Methods of nonlinear programming. In J. Abadie, editor, *Nonlinear Programming*, pages 97–131. North-Holland, Amsterdam, 1967.