Yokohama Publishers
ISSN 1338-9151 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 2, No. 1, pp. 71–79, January 2006

COMPLEMENTARY PERSPECTIVES ON OPTIMIZATION ALGORITHMS

J.L. Nazareth

Abstract: Simplex, interior-point, and memoryless quasi-Newton (QN) optimization algorithms are each viewed from two contrasting perspectives: the first facilitates computer implementation but runs counter to intuition, the second is both insightful and efficiency-revealing. For the memoryless QN case, the discussion is illustrated by numerical experiments. Implications for limited-memory QN algorithms are briefly considered.

Key words: *linear programming, unconstrained optimization, simplex method, interior-point algorithm, central path, logarithmic barrier, conjugate gradients, limited memory, quasi-Newton*

Mathematics Subject Classification: *65K05, 65K10*

1 Introduction

An optimization technique that is counterintuitive at first sight may turn out to be surprisingly promising when viewed from a different perspective. We give three such examples from linear and nonlinear programming: vertex-descending vis-à-vis changing the simplex (Section 2); nonlinearizing a linear program using log-barrier transformations vis-à-vis defining and characterizing a central path (Section 3); and memoryless quasi-Newton updating vis-à-vis conjugate gradient-related search (Section 4). The first two examples are considered only briefly as an introduction to the underlying theme of this note. Our focus is on the third example, where we also provide a numerical illustration and discuss implications for limited-memory quasi-Newton extensions.

2 Vertex-Following vis-à-vis Simplex-Revision

Our first example concerns two contrasting perspectives on the simplex algorithm, which are described by Dantzig [7] as follows:

> It is my opinion that any well trained mathematician viewing the linear programming problem in the row geometry of the variables would have immediately come up with the idea of solving it by a vertex descending algorithm as did Fourier, de la Valée Poussin, and Hitchcock before me—each of us proposing it independently of the other. I believe, however, that if anyone had to consider it as a practical method, as I had to, he would quickly have rejected it on intuitive grounds as a very stupid idea without merit. My own contributions towards the

discovery of the simplex method were (1) independently proposing the algorithm, (2) initiating the software necessary for its practical use, and (3) observing by viewing the problem in the geometry of the columns rather than the rows that, contrary to geometric intuition, following a path on the outside of the convex polyhedron might be a very efficient procedure.

Nowadays, the remarkable efficiency of the simplex algorithm is accepted as empirical fact, and the method is usually defined, from the outset, as a vertex-descending, or 'active-set,' procedure, a point of view that facilitates implementation on a computer. The complementary, insightful view within the geometry of columns is often forgotten, along with the underlying rationale for using 'simplex' terminology. And then also in danger of being lost is a sense of wonder that the simplex algorithm can so effectively circumvent, literally, billions and billions of vertices of a feasible LP polytope, as it descends along a sequence of adjacent vertices to an optimal solution; see, in particular, Dantzig's historical account in his foreword to Dantzig and Thapa [8].

We will not extend this note unnecessarily by elaborating on the two complementary formulations of the simplex algorithm, within row and column geometries, respectively. Instead, we refer the reader to Dantzig's classic [6] or, alternatively, to Chapter 1 of Dantzig and Thapa [9], where their detailed description can be found.

3 Log-Barriers vis-à-vis Central Path

For our second example, we consider the interpretation of Karmarkar's interior-point LP algorithm as a specialized application of the log-barrier method of nonlinear programming (NLP). The latter technique was originally proposed by Frisch [12] and then studied in depth by Fiacco and McCormick [10], who promulgated it widely through their SUMT* implementation. Frisch [13] was also the first to propose its use for solving a linear program. However, this counterintutitive idea of 'nonlinearizing a linear program' never took hold, and, indeed, the log-barrier method for nonlinear programming itself fell into disuse. By the 1980s, it was considered, by many, to be an outmoded optimization technique. The fact that the log-barrier approach possessed a catalog of remarkable properties *in the setting of large-scale linear programming* was discovered during the decade of intense activity, by researchers worldwide, that followed Karmarkar's breakthrough (Karmarkar [19]). A conventional log-barrier viewpoint of 'pushing away from a constraint boundary' was supplanted by the notion of 'hewing to a path of centers,' tightly when the goal was polynomial-time convergence, or very loosely when the goal was to obtain good performance in practice.

Stated another way, a key post-Karmarkar breakthrough was the identification of a particular geometric feature of the convex polytope of feasible points, which was named the *central path* of a linear program. This fundamental object can be defined in a variety of ways, amongst which the log-barrier formulation is the most compelling. For an elementary introduction, see Chapter 7 of Nazareth [29], where it is shown that the central path can be characterized, very conveniently, using certain geometric programming (GP) formulations.† Logarithmic transformations can then be used to unify several alternative GP characterizations of the central path and bring the subject of interior-point LP algorithms into the domain of log-barrier methods of nonlinear programming. Some technical clarifications of the discussion in [29] are as follows:

*Sequential Unconstrained Minimization Technique.

†Geometric programs are mathematical programs that are defined in terms of so-called posynomial and signomial functions. In [29], the term 'posynomial' is used *generically* to cover both cases within geometric programming. A good summary of GP can be found in Bazaraa, Sherali and Shetty [2], Section 11.5.

1. It is assumed that the LP convex polytope of feasible points is bounded, and, implicitly, that its maximizing ('worst') and minimizing ('best') points, for a given linear objective, are unique vertices. Then the central path, a *unique, smooth trajectory* within the interior of the feasible polytope, connects the worst vertex to the best. In the more general case, when the optimal points correspond to facets of the feasible polytope, the central path is a similar trajectory that connects the *analytic centers* of the facets where the linear objective function is, respectively, maximized and minimized over the feasible polytope. For convenience of discussion, we will continue to assume that these facets are unique vertices in items 2 and 3 below.

2. There is nonuniformity in the literature about the precise set of points that constitute the central path. In some formulations, the central path is defined to be the trajectory between the analytic center of the feasible polytope and the best vertex; see, in particular, the log-barrier transformation of a linear program (e.g., Megiddo [22], Fiacco and McCormick [10]), the direct parameterization of the LP primal-dual (KKT) conditions (e.g., Jarre and Stoer [18]), or the parameterized lower-bound characterization (e.g., Renegar [33], Huard [16]). Other formulations, notably, Bayer and Lagarias [1], characterize the central path *in its entirety* as in item 1 above, namely, a trajectory joining the worst and best vertices. Other relevant discussions can be found in Gonzaga [14], Iri and Imai [17], and Todd [35].

3. The central path can be generalized to an *infinite family of weighted central paths*,[‡] each member again being a smooth trajectory between the worst and best vertices. A weighted central path can be used in place of the (original) central path within an interior-point algorithm and the latter again shown to converge in polynomial time. Whether or not there exists a polynomial-time algorithm whose sequence of iterates, ideally vertices, lies on the boundary of the feasible polytope remains an open question.

A more detailed development of the foregoing three items is not needed here and will be pursued elsewhere. Our main objective is to highlight the two contrasting perspectives on the interior-point method: on the one hand, an *implementation-oriented* approach based, counterintuitively, on converting a given linear program into a nonlinear program via log-barrier transformations that penalize closeness to the boundary; and, on the other, an *efficiency-revealing* approach, where the central path—a fundamental object associated with a feasible polytope—is the conceptual mainstay, and log-barrier transformations provide an efficient mechanism for characterizing the central path and using it to guide progress within an algorithm.

4 Memoryless-QN vis-à-vis CG-Related

For our third example and the main focus of this note, we consider nonlinear unconstrained minimization using specialized versions of the quasi-Newton method. The latter, also called the variable metric method, was proposed by Davidon [4] and refined by Fletcher and Powell [11]. Following this algorithmic breakthrough, which was based on the so-called DFP quasi-Newton update, a large variety of other quasi-Newton updates were discovered. Among them, the BFGS update is today the recommended choice for practical applications. Other quasi-Newton updates that possess attractive mathematical properties include the well-known symmetric rank-one (SR1), Davidon's optimally conditioned (OC), Hoshino's, and

[‡]Each barrier term has a positive constant, or weight, associated with it.

Prob	Dim	Range	Prob	Dim	Range
1	3	[b-36,44-s]	10	2	[so-15,28-b]
2	6	[s-35,47-x]	11	4	[x-38,46-bh*]
3	3	[all-4,4-all]	12	3	[h-23,37-s]
4	2	[hx-184,243-b]	13	4	[o-15,18-s]
5	3	[s-31,57-x]	14	4	[s-50,67-x]
6	20	[all-23,23-all]	15	4	[s-38,58-x]
7	12	[s-48,70-x]	16	2	[soh-16,22-x]
8	10	[b-175,220-o]	17	4	[h-72,79-b]
9	10	[b-692,945-x]	18	10	[o-34,40-x]

Table 4.1: Summary of Performance of QN-Updates

Xie-Yuan's; see [5], [15], [36]. In particular, all four are self-complementary, or self-dual. For a detailed discussion of their properties, see Zhu [37].

The foregoing updates do not significantly outperform one another. This is clearly demonstrated by numerical results obtained by Zhu [37] for the Minpack test problems and their standard starting points; see, in particular, Table 2.2, page 42 of [37]. For the reader who may not have access to this dissertation, we provide a summary in Table 4.1 for 18 Minpack problems. These are succesively numbered (in the first and fourth columns of the table) as in Table 2.1 of Zhu [37], which then identifies the corresponding problems by their Minpack names; details of these named problems can be found in Moré et al. [23]. Corresponding problem dimensions are given in the second and fifth columns of Table 4.1. Our summary of results only gives the minimum and maximum number of calls to the function/gradient, or f/g. evaluation routine—each call returns a function value and gradient vector at a specified point—taken over the f/g counts for the BFGS, SR1, Davidon's Optimally Conditioned, Hoshino, and Xie-Yuan updates reported in Table 2.2 of [37] (the DFP update was not tested). This pair of numbers is given within each bracketed entry in the third and sixth columns of Table 4.1, under the heading 'Range.' The updates that produced these numbers are also identified by letter: 'b' for BFGS, 's' for SR1, 'o' for Optimally Conditioned, 'h' for Hoshino, and 'x' for Xie-Yuan. These are placed before the minimum and after the maximum within each bracketed entry. For example, for problem 1 of dimension 3, the minimum number of f/g evaluations was 36, which was obtained with the BFGS update, and the maximum number of evaluations was 44, with the SR1. The other three updates required an intermediate number between 36 and 44. Additional letters within a bracketed entry indicate that more than one update produced the associated number, for example, for problem 4 the Hoshino and the Xie-Yuan updates required the minimum number of evaluations of 184. When the minimum and maximum numbers are the same, then 'all' is used to name the five updates simultaneously, for example, problem 3. If an algorithm failed, this is indicated by a '*' superscript attached to a letter (this occurred only once, for the Hoshino update on problem 11). For full implementation details of the algorithms in the study, see Zhu [37].

From the tabulated results, it can be seen that there is *relatively little variablity* in performance between different updates; the SR1 is the most frequent winner (corresponding to the left-hand letter 's' within bracketed entries of the table); and each update is best on at least two test problems.

When computer storage is at a premium, let us now consider the counterintuitive idea of discarding all, or almost all, the memory in a quasi-Newton algorithm. In its most

basic form, this yields the so-called *memoryless QN algorithm*, where a simple initial matrix with positive diagonal entries, typically the identity matrix, is updated over *only* the most recent step and its associated gradient-change. The updated matrix, which approximates the Hessian, or its inverse, and can be represented implicitly by storing vectors, is then used to define a QN search direction at the current iterate, a line search is performed, and the process repeated.

To illustrate and compare the behaviour of memoryless QN algorithms, we now describe the results of a simple numerical experiment that is patterned on (and identical in its implementation details to) an earlier case study of nonlinear conjugate gradient algorithms; see Nazareth [27], [28]. This used a testbed of four problems from the Minpack collection [23]—extended Rosenbrock (EX-R), Brown-Dennis (B-D), Watson (W), and Biggs6 (B6)—each run from its standard starting point. Problem dimensions are $n = 10, 4, 9, 6$, respectively. The line search employed is described in [26], Chapter 5, Section 5.3, and relatively high accuracy was prescribed by setting the exit tolerance ACC to 0.1. Each tested CG algorithm was terminated when the gradient norm at the current iterate fell below the convergence tolerance 10^{-3}. The numbers of f/g calls required by several nonlinear CG algorithms, in particular, the PPR and PPR$^+$ choices—see [27] for their definition—are tabulated in [28]. For purposes of comparison with memoryless QN algorithms below, these results are quoted in the first two lines of Table 4.2.

Algorithms based on six memoryless QN updates—henceforth, M-DFP, M-BFGS, M-SR1, M-OC, M-Hoshino, and M-Xie-Yuan—were implemented using the same line search and termination criteria as the CG study. Efficiency in terms of both computer storage and overhead cost were not of concern in our experiment, which simulates a situation where the cost of information, namely, function/gradient evaluation, is dominant, and only the numbers of requests for information were counted. For example, it would make no difference to our experiment whether the update was stored implicitly using vectors and the direction obtained by vector operations. Thus, each memoryless update of the identity matrix[§] over the current step was stored as a full matrix approximating the inverse Hessian, and the associated quasi-Newton search direction obtained by a matrix-vector multiplication. The M-SR1 update is the only one among the above that can lose positive definiteness. If this occured then the inverse Hessian approximation was reset to the identity matrix, i.e., the algorithm was restarted along the negative gradient vector.

The test results obtained are reported in the lower part of Table 4.2. Again, each entry is the number of calls to the f/g evaluation routine. An entry '*' in the table indicates a failure to find a solution in the maximum number of f/g called permitted, namely, 2000.

The results in Table 4.2 support the intuition that discarding all memory in a QN algorithm is an idea without merit, *except for the case of the BFGS update*. For this counterintutitive exception, note the overall similarity of the M-BFGS results to those of the PPR-based algorithms in the upper part of Table 4.2. This is no coincidence. It arises from the well-known BFGS-CG relationship discussed in Nazareth [24], [25], Buckley [3], and Kolda et al. [20]. The poor performance of other memoryless QN updates highlights the fact that it is indeed this complementary perspective that furnishes the necesary insight, lacking in the 'discarding of memory' perspective, into the effectiveness of the M-BFGS quasi-Newton algorithm.

Variable-storage conjugate gradient, or VSCG, algorithms based on the BFGS-CG relationship were proposed in Nazareth [24], [25] and also explored in Nazareth and Nocedal

[§]A refinement would employ an Oren-Luenberger scaling factor, which enhances efficiency in practice. This is not needed in our numerical comparison of different updates relative to one another, assuming all updates benefit equally.

Algorithm	EX-R	B-D	W	B6	Total
PPR	103	309	668	43	1123
PPR^+	69	154	403	63	689
M-DFP	*	207	*	648	*
M-BFGS	94	294	240	71	699
M-SR1	*	207	*	*	*
M-OC	*	207	*	*	*
M-Hoshino	*	207	*	*	*
M-Xie-Yuan	72	188	*	1175	*

Table 4.2: Performance of Memoryless QN Algorithms

[30]. The M-BFGS algorithm was proposed by Shanno [34], who built on earlier work of Perry [32]. M-BFGS was generalized to the limited-memory BFGS algorithm, or L-BFGS, by Nocedal [31], Liu and Nocedal [21]. The latter is known to be an effective algorithm[¶] even when almost all memory is discarded, i.e., updates are performed over a very small number of prior steps, typically, between 2 and 5. If one develops analogous limited-memory algorithms based on other quasi-Newton updates, again performed over very few prior steps, it is reasonable to conjecture that numerical results analogous to those of Table 4.2 would be obtained, suggesting in turn that limited-memory algorithms based on updates in the Broyden family *other than the BFGS* are unlikely to be effective when all, or almost all, memory is discarded. However, confirmation of this conjecture requires further numerical study

In conclusion, we have seen a third instance of the overall theme of this article. The counterintuitive idea of discarding all, or almost all, memory in a quasi-Newton algorithm is useful for purposes of computer implementation and yields particular algorithms, M-BFGS and L-BFGS, that have been discovered to be surprisingly effective; and the complementary point of view, derived from the BFGS-CG relationship, provides the prerequisite insight into their potential efficiency.

References

[1] D.A. Bayer and J.C. Lagarias, The nonlinear geometry of linear programming. I. Affine and projective scaling trajectories, *Trans. Amer. Math. Soc.* 314 (1989) 499–526.

[2] M.S. Bazaraa, H.D. Sherali and C.M. Shetty, *Nonlinear Programming: Theory and Algorithms*, (Second Edition), Wiley, New York, 1993.

[3] A. Buckley, Extending the relationship between the conjugate gradient and BFGS algorithms, *Math. Program.* 15 (1978) 343–348.

[¶]It also employs refinements to reduce storage and computational overhead, i.e., matrices are defined implicitly by storing vectors, and recurrence relations are used to obtain search directions, but these are not relevant to our study. Again full matrices, which are computed at each iteration by updating an identity (or scaled identity) matrix over a set of preserved steps and associated gradient changes, in conjunction with matrix-vector multiplications, could be used within L-BFGS without altering its f/g counts. See also footnote 4.

[4] W.C. Davidon, Variable metric method for minimization, Argonne National Laboratory, Report ANL-5990 (Rev.), Argonne, Illinois, 1959 (reprinted, with a new preface, in *SIAM J. Optim.* 1 (1991) 1–17.

[5] W.C. Davidon, Optimally conditioned optimization algorithms without line searches, *Math. Program.* 9 (1975) 1–30.

[6] G.B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, Princeton, New Jersey, 1963.

[7] G.B. Dantzig, Reminiscences about the origins of linear programming, in *Mathematical Programming: The State of the Art, Bonn, 1982*, A. Bachem, M. Grotschel, and B. Korte (eds.), Springer-Verlag, Berlin, 1983, pp. 78–86.

[8] G.B. Dantzig and M.N. Thapa, *Linear Programming. 1: Introduction*, Springer Series in Operations Research, Springer-Verlag, New York, 1997.

[9] G.B. Dantzig and M.N. Thapa, *Linear Programming. 2: Theory and Extensions*, Springer Series in Operations Research, Springer-Verlag, New York, 2003.

[10] A.V. Fiacco and G.P. McCormick, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, Wiley, New York, 1968.

[11] R. Fletcher and M.J.D. Powell, A rapidly convergent descent method for minimization, *Comput. J.* 6 (1963) 163–168.

[12] K.R. Frisch, The logarithmic potential method for convex programming, manuscript, Institute of Economics, University of Oslo, Oslo, Norway, 1955.

[13] K.R. Frisch, La résolution des problèmes de programme lineaire par la méthode du potential logarithmique, *Cahiers du Séminaire D'Econométrie* 4 (1956) 7–20.

[14] C.C. Gonzaga, Path-following methods for linear programming, *SIAM Review* 34 (1992) 167–227.

[15] S. Hoshino, A formulation of variable metric methods, *J. Inst. Math. Appl.* 10 (1972) 394–403.

[16] P. Huard, Resolution of mathematical programming with nonlinear constraints by the method of centers, in *Nonlinear Programming*, J. Abadie (ed.), North Holland, Amsterdam, 1967, pp. 207–219.

[17] M. Iri and H. Imai, Multiplicative barrier function method of linear programming, *Algorithmica* 1 (1986) 455–482.

[18] F. Jarre and J. Stoer, *Optimierung*, Springer-Verlag, Berlin, 2004.

[19] N. Karmarkar, A new polynomial-time algorithm for linear programming, *Combinatorica* 4 (1984) 373–395.

[20] T.G. Kolda, D.P. O'Leary and J.L. Nazareth, BFGS with update skipping and varying memory, *SIAM J. Optim.* 8 (1998) 1060–1083.

[21] D.C. Liu and J. Nocedal, On the limited memory method for large scale optimization, *Math. Program.* (Series B) 45 (1989) 503–528.

[22] N. Megiddo, Pathways to the optimal set in linear programming, in *Progress in Mathematical Programming: Interior-Point and Related Methods*, N. Megiddo (ed.), Springer-Verlag, New York, 1989, pp. 131–158.

[23] J.J. Moré, B.S. Garbow and K.E. Hillstrom, Testing unconstrained optimization software, *ACM Trans. Math. Softw.* 7 (1981) 17–41.

[24] J.L. Nazareth, A relationship between the BFGS and conjugate gradient algorithms, Tech. Memo. ANL-AMD 282, Applied Mathematics Division, Argonne National Laboratory, Argonne, Illinois, 1976. (Presented at the SIAM-SIGNUM Fall 1975 Meeting, San Francisco, California.)

[25] J.L. Nazareth, A relationship between the BFGS and conjugate gradient algorithms and its implications for new algorithms, *SIAM J. Numer. Anal.* 16 (1979) 794–800.

[26] J.L. Nazareth, *Differentiable Optimization and Equation Solving: A Treatise on Algorithmic Science and the Karmarkar Revolution*, Springer-Verlag, New York, 2003.

[27] J.L. Nazareth, On conjugate gradient algorithms as objects of scientific study, *Optim. Meth. Softw.* 18 (2003) 555–565.

[28] J.L. Nazareth, On CG algorithms as objects of scientific study: an appendix, *Optim. Meth. Softw.* 19 (2004) 437–438.

[29] J.L. Nazareth, *An Optimization Primer: On Models, Algorithms and Duality*, Springer-Verlag, New York, 2004.

[30] J.L. Nazareth and J. Nocedal, Conjugate direction methods with variable storage, *Math. Program.* 23 (1982) 326–340.

[31] J. Nocedal, Updating quasi-Newton matrices with limited storage, *Math. Comp.* 35 (1980) 773–782.

[32] A. Perry, A modified CG algorithm, *Oper. Res.* 26 (1978) 1073–1078.

[33] J. Renegar, *A Mathematical View of Interior-Point Methods in Convex Programming*, SIAM, Philadelphia, 2001.

[34] D.F. Shanno, Conjugate gradient methods with inexact searches, *Math. Oper. Res.* 3 (1978) 244–256.

[35] M.J. Todd, Recent developments and new directions in linear programming, in *Mathematical Programming: Recent Developments and Applications*, M. Iri and K. Tanabe (eds.), KTK Scientific Publishers, Tokyo, 1989, pp. 109–157.

[36] Y. Xie, The study of optimal variable metric formulae under the variational principles: the derivation of a new variable metric update, *J. Math.*, China, 32 (1989) 721–726.

[37] M. Zhu, *Techniques for Nonlinear Optimization: Principles and Practice*, Ph.D. Dissertation, Department of Pure and Applied Mathematics, Washington State University, Pullman, Washington, 1997.

Manuscript received 24 May 2004
revised 24 July 2005
accepted for publication 7 November 2005

J.L. Nazareth
Professor Emeritus, Washington State University and Affiliate Professor, University of Washington
Postal address: CDSS, P.O. Box 10509, Bainbridge Island, WA 98110, USA
E-mail address: nazareth@amath.washington.edu