THE SYMMETRIC RANK-ONE QUASI-NEWTON METHOD IS A SPACE DILATION SUBGRADIENT ALGORITHM

By

Michael J. Todd

*Research supported by the National Science Foundation under Grant ECS 8215361.
ABSTRACT

It is well-known that a particular choice of the parameters in Shor's subgradient algorithm with space dilation in the direction of the gradient yields the ellipsoid method. We show that another choice of these parameters leads to the quasi-Newton method that uses the symmetric rank-one update and direct prediction. One curious feature is that the sequence of approximate inverse Hessian matrices lags by one in the former description; this necessitates a different starting matrix or an unusual update at the first step. While the similarity between update formulae in space-dilation and quasi-Newton methods has been observed by several researchers, our result seems to be the first showing a precise equivalence. We note that apparently this equivalence cannot be extended to other quasi-Newton methods (for smooth optimization) or space-dilation methods (for nonsmooth optimization). Nevertheless, we feel that our result gives insight into the relationship between these two classes of algorithms, and hope that it will suggest new efficient methods for nonsmooth problems.

Keywords: nonlinear programming, unconstrained optimization, quasi-Newton methods, space-dilation methods.
1. Introduction

In 1970, Shor [4] introduced the subgradient method with space dilation in the direction of the gradient, for convex nonsmooth optimization. When applied to minimize a smooth function \( f \), starting with an initial trial point \( x_1 \in \mathbb{R}^n \) and an initial symmetric positive definite matrix \( H_1 \), intended as an approximation to \( (\nabla^2 f(x_1))^{-1} \), the method proceeds as follows:

**Algorithm** SDG\((x_1, H_1)\):

**Iteration** \( k \). Compute \( g_k = \nabla f(x_k) \); STOP if \( g_k = 0 \).

Otherwise, set

\[
 s_k = -\alpha_k H_k g_k; \quad (1)
\]

\[
 x_{k+1} = x_k + s_k; \quad (2)
\]

\[
 H_{k+1} = H_k - \sigma_k H_k g_k^T g_k H_k. \quad (3)
\]

The method depends on the sequences \( \{\alpha_k\} \) and \( \{\sigma_k\} \) of parameters. In fact, Shor chose \( \alpha_k = h_k / (g_k^T H_k g_k)^{1/2} \) and \( \sigma_k = (1 - \beta^2) / g_k^T H_k g_k \) with \( 0 < \beta < 1 \) the dilation parameter. This ensures that all \( H_k \)'s are positive definite. We will need the flexibility of choosing \( \sigma_k > 1 / g_k^T H_k g_k \) later; in this case, the matrices \( H_k \) do not remain positive definite and we have therefore parametrized the step size by \( \alpha_k \) rather than \( h_k \).

The ellipsoid method of Yudin and Nemirovskii [7] corresponds to the parameters

\[
 \alpha_k = \left( \frac{n}{\sqrt{2} \nu n - 1} \right)^k \left( \frac{\nabla^2}{(n+1)(g_k^T H_k g_k)^{1/2}} \right) \left( \sqrt{2} \nu n - 1 \right)^{1/2},
\]

\[
 \sigma_k = 2 / (n+1) g_k^T H_k g_k.
\]
which is equivalent to $\beta = ((n-1)/(n+1))^{1/2}$ and thus guarantees that all $H_k$'s are positive definite.

In this note we wish to compare the space dilation method to the quasi-Newton algorithm that uses the symmetric rank-one update formula and direct prediction of step size (see e.g., Dennis and Schnabel [2]). Given an initial trial point $x_1 \in \mathbb{R}^n$ and symmetric positive definite matrix $\hat{H}_1$, with $g_1 = \nabla f(x_1)$, this algorithm proceeds as follows:

Algorithm $SR1(x_1, \hat{H}_1)$:

Iteration $k$. STOP if $g_k = 0$. Otherwise, set

$$s_k = -\hat{H}_k g_k;$$  \hfill (4)

$$x_{k+1} = x_k + s_k. \hfill (5)$$

Compute

$$g_{k+1} = \nabla f(x_{k+1}) \quad \text{and} \quad y_k = g_{k+1} - g_k.$$

Set

$$\hat{H}_{k+1} = \hat{H}_k + \frac{(s_k - \hat{H}_k y_k)(s_k - \hat{H}_k y_k)^T}{(s_k - \hat{H}_k y_k)^T y_k}. \hfill (6)$$

Because of the unit step size in (4)-(5), (6) can be simplified by observing that

$$s_k - \hat{H}_k y_k = -\hat{H}_k g_k+1;$$

hence

$$\hat{H}_{k+1} = \hat{H}_k - \hat{H}_k g_k+1 \hat{H}_k g_k+1^T \hat{H}_k / (g_k+1 \hat{H}_k y_k). \hfill (6')$$
The similarity to (3) is clear, although (6') involves the subsequent gradient \( g_{k+1} \) while (3) uses \( g_k \). We shall see that we can choose the parameters in algorithm SDG so that it generates the same iterates as algorithm SRI---however, the matrices \( H_k \) will lag one behind the matrices \( \hat{H}_k \).

While the similarity between (3) and quasi-Newton updates has been observed by many researchers, this result appears to be the first showing a precise equivalence between a space-dilation (for nonsmooth optimization) and a quasi-Newton (for smooth optimization) method. Unfortunately, it seems that the equivalence cannot be extended to other members of these classes. In particular, we cannot encompass subgradient algorithms with space dilation in the direction of the difference of successive gradients (Shor and Zhurbenko [6]), which are regarded as more efficient in practice than the SDG algorithms (see Shor [5]). Nor can we include a line search in the quasi-Newton method, nor use the preferable DFP or BFGS updates (see [2]).

The quasi-Newton method using the symmetric rank-one update and a unit step size at each iteration, as above, is rarely used. We conclude this section by noting some of its properties. Broyden, Dennis, and More [1] have shown that it can fail to be locally convergent, since the denominator in (6) or (6') can vanish for \( k = 1 \) even with \( x_1 \) and \( \hat{H}_1 \) arbitrarily close to a minimizer \( x_* \) and \( (\nabla^2 f(x_*)^{-1}) \). If the algorithm does not break down in this way, then it yields the minimizer of a convex quadratic function within \( n \) steps. If instead exact line searches are used (and again assuming (6) remains well-defined), then Dixon [3] has shown that it generates the same sequence of points as the DFP and BFGS methods with exact line searches, on arbitrary smooth functions \( f \).
2. The Result

The observation below (6') indicates that the first update of SDG might have to be special so that $H_2 = \hat{H}_1$. Instead, we will use a dummy 0th step and a different initial matrix $H_1$ so that all iterations are identical.

Theorem. Given $x_1$, $g_1$ and $\hat{H}_1$, choose $s_0$ so that $0 \neq g_1^T s_0 \neq g_1^T H_1 g_1$. Set

$$H_1 = \hat{H}_1 + \hat{H}_1 g_1 g_1^T H_1 / g_1^T (s_0 - \hat{H}_1 g_1).$$

Then, if we choose the parameters by

$$\alpha_k = g_k^T s_{k-1} / g_k^T (s_{k-1} + H_k g_k),$$

$$\sigma_k = 1 / g_k^T (s_{k-1} + H_k g_k),$$

the algorithms SDG($x_1, H_1$) and SR1($x_1, \hat{H}_1$) generate identical iterates (if they do not simultaneously break down) and $H_{k+1} = \hat{H}_k$ for $k \geq 1$.

Proof. From (7) we deduce

$$H_1 g_1 = \lambda \hat{H}_1 g_1,$$

where

$$\lambda = g_1^T s_0 / g_1^T (s_0 - \hat{H}_1 g_1).$$

Hence $g_1^T H_1 g_1 = \lambda g_1^T \hat{H}_1 g_1$; substituting for $g_1^T \hat{H}_1 g_1$ in (11) and solving for $\lambda^{-1}$ yields
\[
\lambda^{-1} = g_1^T s_0 / g_1^T (s_0 + H_1 g_1). \tag{12}
\]

Thus
\[
H_1 g_1 = \lambda^{-1} H_1 g_1 = -\alpha_1 H_1 g_1
\]

and the first steps agree—the two algorithms generate the same \( x_2 \). Also,
\[
H_2 = H_1 - (g_1^T (s_0 + H_1 g_1))^{-1} H_1 g_1 g_1^T H_1 = \hat{H}_1 + \hat{H}_1 g_1 g_1^T H_1 ((g_1^T (s_0 - \hat{H}_1 g_1))^{-1} - \lambda^2 (g_1^T (s_0 + H_1 g_1))^{-1})
\]

by (7) and (10); using (11) and (12), the second term vanishes, whence
\[
H_2 = \hat{H}_1. \tag{13}
\]

Now assume that both algorithms generate the same iterates \( x_j, j \leq k \), and \( H_k = \hat{H}_{k-1} \). Then from (4) we have
\[
g_k^{\hat{H}_k} y_{k-1} = g_k^T H_k y_{k-1} = g_k^T (H_k g_k - \hat{H}_{k-1} g_{k-1})
\]
\[
= g_k^T (s_{k-1} + H_k g_k) \tag{14}
\]

and similarly
\[
1 - g_k^T H_k g_k / g_k^T H_k y_{k-1} = -g_k^T H_k g_{k-1} / g_k^T H_k y_{k-1}
\]
\[
= g_k^T s_{k-1} / g_k^T (s_{k-1} + H_k g_k)
\]
\[
= \alpha_k.
\]
Hence
\[
-\hat{H}_k g_k = -(H_k - H_k g_k g_k^T H_k g_k g_k^T H_k y_{k-1}) g_k \\
= -H_k g_k (1 - g_k^T H_k g_k / g_k^T H_k y_{k-1}) \\
= -\alpha_k H_k g_k
\]
so that both algorithms generate the same point $x_{k+1}$. Moreover, the equation $\hat{H}_{k-1} = H_k$, the update formulae (3) and (6'), and (9) and (14) together imply $\hat{H}_k = H_{k+1}$. Thus by induction both algorithms generate the same iterates $\{x_k\}$ and $\hat{H}_k = H_{k+1}$ for all $k$.

To complete the proof, note that both algorithms are well-defined as long as the quantity in (14) remains nonzero. If this is zero for some $k$, then algorithm SDG fails in the $k$th iteration since $\alpha_k$ is not defined, and algorithm SRI fails in the $(k-1)$st iteration after generating $x_k$ since $\hat{H}_k$ is not defined. (In particular, if $f$ is not smooth but piecewise-linear and $g_k$ is a subgradient of $f$ at $x_k$, it is very possible that $g_k = g_{k-1}$ so that $y_{k-1}$ is zero and (14) vanishes. Thus this version of algorithm SDG is disastrous for such problems, while other variants remain applicable.)

This concludes the proof of the theorem. We hope that it will suggest new efficient methods for nonsmooth optimization where the parameters are chosen to be different from those that lead to the typically slow ellipsoid method. Note that, if some form of line search yields $g_k^T s_{k-1} > 0$, then $\sigma_k$ in (9) will maintain positive definiteness in $H_k$. 
REFERENCES


