Accelerated memory-less SR1 method with generalized secant equation for unconstrained optimization

Neculai Andrei
Center for Advanced Modeling and Optimization,
Academy of Romanian Scientists,
54, Splaiul Independenţei, Sector 5,
Bucharest, ROMANIA
E-mail: neculaiandrei70@gmail.com

Technical Report 9/2021
May 28, 2021

Abstract. The memory-less SR1 with generalized secant equation (MM-SR1gen) is presented and developed together with its numerical performances for solving a collection of 800 unconstrained optimization problems with the number of variables in the range [1000, 10000]. The convergence of the MM-SR1gen method is proved under the classical assumptions. Comparison between the MM-SR1gen versus the memory-less SR1 method, versus the memory-less BFGS method and versus the BFGS in implementation of Shanno and Phua from CONMIN show that MM-SR1gen is more efficient and more robust than these algorithms. By solving five applications from MINPACK-2 collection, each of them with 40,000 variables, we have the computational evidence that MM-SR1gen is more efficient than memory-less SR1 and than memory-less BFGS. The conclusion of this study is that the memory-less SR1 method with generalized secant equation is a rapid and reliable method for solving large-scale minimizing problems. Besides, it is shown that the accuracy of the Hessian approximations along the iterations in quasi-Newton methods is not as crucial in these methods as it is believed.

Keywords: Symmetric-rank one SR1; Quasi-Newton BFGS; Wolfe line search; Numerical experiments; Secant equation; Memory-less Sr1; Dolan and Moré performance profile

Mathematics Subject Classification: 90C30, 90C53, 90-08

1 Introduction

For solving the unconstrained optimization problem

$$\min f(x), \tag{1}$$

where $x \in \mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function, bounded from below, one of the most efficient methods is the quasi-Newton methods. Plenty of quasi-Newton methods are known. In these methods the search direction d_{k+1} is computed as solution of the following linear algebraic system

$$B_{k+1} d_{k+1} = -g_{k+1}, \tag{2}$$

where B_{k+1} is an approximation to the Hessian $\nabla^2 f(x_{k+1})$ and g_{k+1} is the gradient $\nabla f(x_{k+1})$ of the minimizing function, or directly as

$$d_{k+1} = -H_{k+1} g_{k+1}, \tag{3}$$

where H_{k+1} is an approximation to the inverse Hessian, $\nabla f(x_{k+1})^{-1}$, i.e. $H_{k+1} = B_{k+1}^{-1}$. Starting with an initial point x_0 the next approximation x_{k+1} to x^*, solution of (1), is computed as

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0,1,..., \tag{4}$$

where α_k is the stepsize often computed by the Wolfe line search [36, 37]:

$$f(x_k + \alpha_k d_k) < f(x_k) + \rho \alpha_k g_k^T d_k, \tag{5a}$$
$$\nabla f(x_k + \alpha_k d_k)^T d_k > \sigma \nabla f(x_k)^T d_k, \tag{5b}$$

where $0 < \rho < \sigma < 1$ are some parameters. Usually, $d_0 = -g_0$.

The quasi-Newton algorithms are efficient and robust for minimizing functions that satisfy certain assumptions and have a super-linear rate of local convergence. Currently, many variants of the updating formula for the approximation to the Hessian (or to the inverse Hessian) are known: symmetric rank-one (SR1) [9, 15, 16, 18] and the rank-two such as the Davidon-Fletcher-Powell (DFP) update [16, 19], Powell-symmetric-Broyden (PSB) [30] and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update [10, 20, 22, 31]. A unifying framework for many of these updates, including both rank-one and rank-two updates was given by Huang [24]. All these quasi-Newton methods are based on secant equation which is a formula linking the displacements $s_k = x_{k+1} - x_k$ and the change of gradients $y_k = g_{k+1} - g_k$ under the curvature condition. These updates have been very intensive studied and in general it is accepted that the BFGS method is the most efficient and robust for solving minimizing problems with differential functions. However, the main drawback of BFGS is that it is limited for solving minimizing problems with a small or a medium number of variables (let say 1000 variables). It requires a large amount of memory and therefore it involves a large amount of numerical operations.

The purpose of the paper is to introduce and study the memory-less symmetric-rank one SR1 quasi-Newton methods with generalized secant equation, as a technique for solving large-scale minimizing problems and to show their performances for solving large-scale unconstrained optimization problems. The structure of the paper is as follows. Section 2 describes the SR1 method. Section 3 is devoted to the memory-less SR1 method. In Section 4 the memory-less SR1 with generalized secant equation method is introduced. Section 5 is dedicated to the memory-less BFGS method. The memory-less SR1, the memory-less SR1 with generalized secant equation and the memory-less BFGS algorithms are presented in Section 6. The convergence of the memory-less SR1 method with generalized secant equation is described in Section 7. The global convergence is

proved under classical assumptions. The numerical results and comparisons among the algorithms are detailed in Section 8. For solving a collection of 800 unconstrained optimization problems up to 10000 variables we have the computational evidence that the memory-less SR1 method with generalized secant equation is more efficient and more robust than the memory-less SR1 method and the memory-less BFGS method. Comparing memory-less SR1 with generalized secant equation versus the BFGS from CONMIN [34, 35] shows that, for solving problems up to 1000 variables, memory-less SR1 with generalized secant equation is way more efficient and more robust. This section also presents the performances of these algorithms for solving five applications from MINPACK-2 collection, each of them with 40,000 variables. The conclusion of this study is that the memory-less SR1 method with generalized secant equation is a simple and efficient technique for solving large-scale problems. Besides, it seems that the accuracy of the Hessian approximations along the iterations in quasi-Newton methods is not as crucial in these methods as it is believed.

2 Symmetric rank-one SR1 method

In the quasi-Newton methods the basic requirement for the updating formula to the Hessian is the so called the *secant equation*, to be satisfied at each iteration, namely

$$B_{k+1}s_k = y_k \quad \text{or} \quad H_{k+1}y_k = s_k, \tag{6}$$

where $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$. The secant equation (6) is obtained from the requirement that the gradient of the quadratic model of the minimizing function should match the gradient of the minimizing function at the latest two iterations x_k and x_{k+1}.

The symmetric rank-one SR1 update formula, in which we are interested in this paper, can be derived as solution of the following simple problem. "*Given a symmetric matrix B_k and the vectors s_k and y_k, finds a new symmetric matrix B_{k+1} such that $B_{k+1} - B_k$ has rank one and such that the secant equation $B_{k+1}s_k = y_k$ is satisfied*." It is easy to see that if $(y_k - B_k s_k)^T s_k \neq 0,$ then the unique solution of the above problem is

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}. \tag{7}$$

If $y_k = B_k s_k$ then the solution is $B_{k+1} = B_k$. However, if $(y_k - B_k s_k)^T s_k = 0$ and $y_k \neq B_k s_k$, then there is no solution to the problem.

The main drawbacks of SR1 update are as follows.

1) The denominator $(y_k - B_k s_k)^T s_k$ of the SR1 update term in (7) may vanish, i.e. $(y_k - B_k s_k)^T s_k \cong 0$, cases in which B_{k+1} is not well-defined.

2) The step directions computed by using the SR1 updating formula given by (7) may no longer be uniform linear independent, thus leading to slow down the convergence or even the stalling.

3) The SR1 Hessian approximation may not be positive definite along the iterations, thus resulting a direction that does not produce descent.

To prevent the method from failing due to the first drawback one simple remedy is to set $B_{k+1} = B_k$. However, possible this will slow down the convergence of the method. Conn, Gould and Toint [13] and Khalfan, Byrd and Schnabel [26] showed that the denominator of (7) vanishes rarely in practice and setting $B_{k+1} = B_k$ does not have a significant impact on the performances of the SR1 method subject to the number of the iterations or runtimes.

The second drawback is more delicate, being in close connection with the uniform linear independence of the search directions generated by the SR1 algorithm. A more precise definition of the uniform linear independence was given by Conn, Gould and Toint [13]. "A sequence $\{s_k\}$ is uniformly linearly independent if there exist $\xi > 0,$ k_0 and $m \geq n$ such that, for each $k \geq k_0,$ there is n distinct indices $k \leq k_1 \leq k_2 \leq \ldots \leq k_n \leq k + m$ for which the minimum singular value of the matrix $S = \left[\frac{s_{k_1}}{\|s_{k_1}\|}, \cdots, \frac{s_{k_n}}{\|s_{k_n}\|} \right]$ is at least $\xi.$" Conn, Gould and Toint [13] proved that the sequence of matrices generated by the SR1 formula converges to the exact Hessian, when the sequence of iterates converges to a limit point and the sequence of steps is uniformly linearly independent. Kelley and Sachs [25] provide similar convergence results removing the first of these assumptions. Fiacco and McCormick [18] showed that if the search directions are linearly independent and the denominator of (7) is always non-zero, then the SR1 method without line searches minimize a strongly convex quadratic function in at most $n+1$ steps. In this case B_{n+1} is exactly the Hessian of the quadratic function. Observe that this result is significant since it does not require exact line search, as is the case for the BFGS update. Generally, the above condition given by the definition of the uniform linear independency is not implemented in practice, it serves only as one of the main assumptions of a proof that the SR1 approximations to the Hessian converge to the true Hessian as the iterates converge to the solution of (1).

Subject to the uniform linear independency of the search directions Khalfan, Byrd and Schnabel [26] showed that many problems do not satisfy this requirement, but they proved the local convergence of the SR1 method using only the positive definiteness and boundedness assumptions for the approximate Hessian. More than this Conn, Gould and Toint [13] proved that if the minimizing function f is twice continuously differentiable and its Hessian is bounded and Lipschitz continuous and the iterates generated by the SR1 method converge to a point x^* and in addition for all $k,$

$$\left|(y_k - B_k s_k)^T s_k\right| \geq \xi \|y_k - B_k s_k\| \|s_k\|, \tag{8}$$

for some $\xi \in (0,1),$ and the steps s_k are uniformly linearly independent, then

$$\lim_{k \to \infty} \left\| B_k - \nabla^2 f(x^*) \right\| = 0. \tag{9}$$

Often condition (8) is used in implementations of the SR1 method in order to ensure that this update is well behaved. If this condition is not satisfied, then the update is skipped. Conn, Gould and Toint [13] and Khalfan, Byrd and Schnabel [26] provide theoretical and computational results, respectively, that if the uniform linear independence assumption is satisfied, then the approximations to the Hessian generated by the SR1 method are more accurate than those generated by BFGS, and SR1 converge faster to the true Hessian than BFGS. Therefore, if all these above drawbacks are addressed in a reliable and efficient manner, then SR1 can be used for solving (1) instead of the rank-two updates. More details on SR1 method concerning the undefined updates, choosing the initial approximate B_0, uniform linear independence of the steps, are found in [7, 11].

Now, let H_k be the inverse approximation to the Hessian at iteration k. By using the Sherman-Morrison-Woodbury formula in (7), the following update to the inverse Hessian for SR1 is

$$H_{k+1} = H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{(s_k - H_k y_k)^T y_k}. \tag{10}$$

This variant of the algorithm is only applicable in cases in which the inverse H_k exists. The search direction corresponding to the quasi-Newton SR1 method is $d_{k+1} = -H_{k+1} g_{k+1}$, i.e.

$$d_{k+1} = -H_k g_{k+1} - \frac{(s_k - H_k y_k)^T g_{k+1}}{(s_k - H_k y_k)^T y_k}(s_k - H_k y_k). \tag{11}$$

3 Memory-less SR1 method

The memory-less SR1 method is obtained by setting $B_k = I$ in (7), thus obtaining

$$B_{k+1} = I + \frac{(y_k - s_k)(y_k - s_k)^T}{(y_k - s_k)^T s_k}. \tag{12}$$

Now, applying the Sherman-Morrison-Woodbury formula in (12) the following update to the inverse Hessian for the memory-less SR1 method is obtained as

$$H_{k+1} = I + \frac{(s_k - y_k)(s_k - y_k)^T}{(s_k - y_k)^T y_k}. \tag{13}$$

With this, *the search direction corresponding to the memory-less SR1 method* is

$$d_{k+1} = -g_{k+1} - \frac{(s_k - y_k)^T g_{k+1}}{(s_k - y_k)^T y_k}(s_k - y_k). \tag{14}$$

Observe that if $(s_k - y_k)^T y_k \cong 0,$ then the search direction (14) is not defined. In this case the remedy is to skip this iteration and to set $d_{k+1} = -g_{k+1}$.

4 Memory-less SR1 method with generalized secant equation

This quasi-Newton method is based on the *generalized secant equation*

$$y_k = \gamma_k B_{k+1} s_k, \tag{15}$$

where γ_k is a positive parameter. The generalized secant equation is obtained from the requirement that the quadratic model of the minimizing function with scaled approximation to the Hessian should match the gradient of the minimizing function at the latest two iterations x_k and x_{k+1}.

Suppose that in the current point x_k we know the approximation B_k to the Hessian, which is a symmetric matrix. To derive the SR1 method with generalized secant equation we impose that B_{k+1}, the approximation to the Hessian in x_{k+1}, satisfies (15) and is obtained after a rank-one update of B_k, i.e. has the form

$$B_{k+1} = B_k + \delta_k u u^T, \tag{16}$$

where δ_k is a scalar and $u \in \mathbb{R}^n$. Substituting this form into the generalized secant equation, we get that

$$y_k = \gamma_k B_k s_k + \gamma_k \delta_k u u^T s_k,$$

or, alternatively

$$y_k - \gamma_k B_k s_k = \gamma_k \delta_k (u^T s_k) u.$$

Since $\gamma_k \delta_k (u^T s_k)$ is a scalar, in order to satisfy this equation we can simply set $\delta_k = \dfrac{1}{\gamma_k (u^T s_k)}$ and $u = y_k - \gamma_k B_k s_k$. Therefore, introducing these elements in (16) *the SR1 method with generalized secant equation* is obtained as

$$B_{k+1} = B_k + \frac{(y_k - \gamma_k B_k s_k)(y_k - \gamma_k B_k s_k)^T}{\gamma_k (y_k - \gamma_k B_k s_k)^T s_k}. \tag{17}$$

Note that the SR1 update formula (17) is unique, that is, there is exactly one rank-one update satisfying the generalized secant equation. Moreover, if $\gamma_k = 1$ in (17), then the SR1 update (7) is obtained.

If $y_k = \gamma_k B_k s_k$ then the solution is $B_{k+1} = B_k$. However, if $(y_k - \gamma_k B_k s_k)^T s_k = 0$ and $y_k \neq \gamma_k B_k s_k$, then there is no solution to the problem.

Let H_k be the inverse approximation to the Hessian at iteration k. By using the Sherman-Morrison-Woodbury formula in (17), the following update to *the inverse Hessian for SR1 with generalized secant equation* is obtained

$$H_{k+1} = H_k - \frac{(H_k y_k - \gamma_k s_k)(H_k y_k - \gamma_k s_k)^T}{(H_k y_k - \gamma_k s_k)^T y_k}. \tag{18}$$

From (18) after some simple algebraic manipulation *the search direction corresponding to the inverse Hessian SR1 updating with generalized secant equation* is $d_{k+1} = -H_{k+1}g_{k+1}$ i.e.

$$d_{k+1} = -H_k g_{k+1} + \frac{(H_k y_k - \gamma_k s_k)^T g_{k+1}}{(H_k y_k - \gamma_k s_k)^T y_k}(H_k y_k - \gamma_k s_k). \tag{19}$$

Now, the *memory-less SR1 method with generalized secant equation* is obtained by considering $B_k = I$ in (17), i.e.

$$B_{k+1} = I + \frac{(y_k - \gamma_k s_k)(y_k - \gamma_k s_k)^T}{\gamma_k (y_k - \gamma_k s_k)^T s_k}. \tag{20}$$

Observe that this is a very simple formula in which the information about the Hessian is not accumulated from iteration to iteration. Besides, we see that the memory-less SR1 method with generalized secant equation has the same drawbacks as the SR1 method, i.e. when the denominator $(y_k - \gamma_k s_k)^T s_k$, is zero, or is very close to zero, then the method is not defined.
Now, choosing in (18) $H_k = I$, i.e.

$$H_{k+1} = I - \frac{(y_k - \gamma_k s_k)(y_k - \gamma_k s_k)^T}{(y_k - \gamma_k s_k)^T y_k}. \tag{21}$$

the *memory-less inverse of SR1 method with generalized secant equation* is obtained. Therefore, from (21) the *memory-less SR1 search direction with generalized secant equation* is $d_{k+1} = -H_{k+1}g_{k+1}$, i.e.

$$d_{k+1} = -g_{k+1} + \frac{(y_k - \gamma_k s_k)^T g_{k+1}}{(y_k - \gamma_k s_k)^T y_k}(y_k - \gamma_k s_k). \tag{22}$$

The main advantage of the memory-less SR1 search direction with generalized secant equation (22) is that for its implementation in computer programs only two scalar products $(y_k - \gamma_k s_k)^T g_{k+1}$ and $(y_k - \gamma_k s_k)^T y_k$ have to be computed. This is very advantageous for solving large-scale problems. Observe that in this memory-less SR1 update the information on the Hessian approximation from the previous iteration is not accumulated to the current iteration.

Proposition 1 *If*

$$\gamma_k > \frac{y_k^T y_k}{s_k^T y_k}, \tag{23}$$

then the memory-less SR1 search direction with generalized secant equation (22) *is a descent direction.*

Proof. From (23) it follows that $(y_k - \gamma_k s_k)^T y_k < 0.$ Therefore, by direct computation, from (18) we get

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 + \frac{((y_k - \gamma_k s_k)^T g_{k+1})^2}{(y_k - \gamma_k s_k)^T y_k} < 0, \qquad \blacklozenge$$

Now, it can be noticed that the memory-less SR1 search direction with generalized secant equation (22) has three terms. The first one is the negative gradient $-g_{k+1}$, the last two terms involve s_k and y_k, both of them being multiplied by some scalars. It is very simple to see that the search direction (22) satisfies the conjugacy condition, i.e. $y_k^T d_{k+1} = -\gamma_k s_k^T g_{k+1},$ where γ_k is a positive parameter. Therefore, the memory-less SR1 method with generalized secant equation is a conjugate gradient method which satisfies the Dai and Liao [14] conjugacy condition.

5 Memory-less BFGS method

As we know the most effective quasi-Newton updating of the approximations to the Hessian is considered to be the BFGS formula [28, 29] where

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} \tag{24}$$

which is a *rank-two update* that satisfies the secant equation (6). If H_k is the inverse approximation to the Hessian at iteration $k,$ then from (24) by applying the Sherman-Morrison-Woodbury formula twice the following update to the inverse Hessian for BFGS is obtained

$$H_{k+1} = H_k - \frac{s_k y_k^T H_k + H_k y_k s_k^T}{y_k^T s_k} + \left(1 + \frac{y_k^T H_k y_k}{y_k^T s_k}\right)\frac{s_k s_k^T}{y_k^T s_k}. \tag{25}$$

The most important properties of BFGS are as follows. If H_k is positive definite, then also H_{k+1} given by (25) is positive definite for any $k,$ provided that $y_k^T s_k > 0$ (which always is satisfied when the Wolfe line search (5) are satisfied). Therefore, if H_0 is chosen to be positive definite, then the rest of all the approximations H_k will also be positive definite. Also BFGS has the self-correcting property, i.e. if H_k incorrectly approximates the curvature of the minimizing function and this estimate slows down the iteration, then the inverse Hessian approximation will tend to correct itself in the next few iterations. The self-correcting property depends on the quality of the implementation of the Wolfe line search. For the Wolfe line search, always the initial value $\alpha = 1$ is tried and this produce superlinear convergence of the method. All these properties of BFGS update make this quasi-Newton method one of the best in this class.

Now considering $H_k = I$ in (25), the *memory-less BFGS method for the inverse approximation to the Hessian* is obtained as

$$H_{k+1} = I - \frac{s_k y_k^T + y_k s_k^T}{y_k^T s_k} + \left(1 + \frac{y_k^T y_k}{y_k^T s_k}\right) \frac{s_k s_k^T}{y_k^T s_k}. \tag{26}$$

The corresponding search direction is $d_{k+1} = -H_{k+1} g_{k+1}$, where H_{k+1} is given by (26), i.e.

$$d_{k+1} = -g_{k+1} + \frac{(y_k^T g_{k+1}) s_k + (s_k^T g_{k+1}) y_k}{y_k^T s_k} - \left(1 + \frac{y_k^T y_k}{y_k^T s_k}\right) \frac{(s_k^T g_{k+1}) s_k}{y_k^T s_k}. \tag{27}$$

Observe that the numerical computation of d_{k+1} from (27) involves only four scalar products: $y_k^T s_k$, $y_k^T y_k$, $y_k^T g_{k+1}$ and $s_{k+1}^T g_{k+1}$. Therefore, the memory-less BFGS method is very suitable for solving large-scale problems. It is worth seeing that the search direction corresponding to the memory-less BFGS updating has three terms. Besides, it is easy to prove that this search direction satisfies the Dai-Liao conjugacy condition, i.e. $y_k^T d_{k+1} = -s_k^T g_{k+1}$. We showed that also the memory-less SR1 method with generalized secant equation has this property of satisfying the conjugacy condition. Therefore, there is close connection between the quasi-Newton and the conjugate gradient methods. It is worth mentioning that Shanno [32, 33] was the first who observed that *the conjugate gradient methods are precisely the quasi-Newton methods where the approximation to the inverse to the Hessian is restarted as the identity matrix at every iteration*.

There is a great difference between the memory-less SR1 method with generalized secant equation and the memory-less BFGS method. If in these methods the stepsize is computed by the Wolfe line search (5), then the memory-less BFGS method is well defined since at every iteration $y_k^T s_k \neq 0$. On the other hand, in case of SR1 method with generalized secant equation the Wolfe line search do not guarantee that $(y_k - \gamma_k s_k)^T y_k \neq 0$. However, both these methods are implemented in such a way that if the corresponding search directions are not defined, i.e. if in BFGS or in SR1 with generalized secant equation $y_k^T s_k$ or $(y_k - \gamma_k s_k)^T y_k$ are very close to zero, respectively, then the search direction considered at that iteration is the negative gradient, i.e. the steepest descent. In the following let us present the algorithms corresponding to the memory-less SR1, to the memory-less SR1 with generalized secant equation and to the memory-less BFGS methods.

6 Memory-less algorithms: SR1, SR1 with generalized secant equation and BFGS

The memory-less algorithms corresponding to SR1 (MM-SR1), to SR1 with generalized secant equation (MM-SR1gen) and to BFGS (MM-BFGS) are very simple. In all of them the stepsize is computed by the Wolfe line search (5). The search direction in memory-less SR1 method is computed as in (14). The search direction in memory-less SR1 method with generalized secant equation is computed as in (22), while the search direction in the memory-less BFGS method is computed as in (27). In our numerical

experiments with these algorithms an acceleration scheme developed by Andrei [1, 3] was implemented.

MM-SR1, MM-SR1gen and MM-BFGS Algorithms

	MM-SR1	MM-SR1gen	MM-BFGS
1.	Consider an initial point x_0. Set $k=0$. Select some values for the Wolfe line search conditions σ and ρ with $0<\rho<\sigma<1$. Compute $g_0=\nabla f(x_0)$ and set $d_0=-g_0$. Select the sufficiently small parameters: $\varepsilon>0$ used in the criterion for stopping the iterations, $\varepsilon_q>0$ used in search direction computation and $\varepsilon_A>0$ used in acceleration scheme		
2.	Test a criterion for stopping the iterations: if $\|g_k\|_\infty \le \varepsilon$ then stop the iterations, otherwise go to step 3		
3.	Compute the stepsize α_k using the standard Wolfe line search conditions		
4.	Update the variables $x_{k+1}=x_k+\alpha_k d_k$ and compute f_{k+1} and g_{k+1}. Compute $s_k=x_{k+1}-x_k$ and $y_k=g_{k+1}-g_k$		
5.	Acceleration scheme: a) Compute: $z=x_k+\alpha_k d_k$, $g_z=\nabla f(z)$ and $y_k=g_k-g_z$ b) Compute: $\bar{a}_k=\alpha_k g_k^T d_k$, and $\bar{b}_k=-\alpha_k y_k^T d_k$ c) If $\|\bar{b}_k\| \ge \varepsilon_A$, then compute $\xi_k=-\bar{a}_k/\bar{b}_k$ and update the variables as $x_{k+1}=x_k+\xi_k\alpha_k d_k$. Compute f_{k+1} and g_{k+1}. Compute $y_k=g_{k+1}-g_k$ and $s_k=x_{k+1}-x_k$		
6.	If $\|s_k^T y_k - y_k^T y_k\| \ge \varepsilon_q$, then compute the search direction d_{k+1} as in (14). Otherwise set $d_{k+1}=-g_{k+1}$	Select a value for parameter γ_k as in (23). If $\|y_k^T y_k - \gamma_k s_k^T y_k\| \ge \varepsilon_q$, then compute the search direction d_{k+1} as in (22). Otherwise set $d_{k+1}=-g_{k+1}$	If $\|y_k^T s_k\| \ge \varepsilon_q$, compute the search direction d_{k+1} as in (27). Otherwise, set $d_{k+1}=-g_{k+1}$
7.	Restart iterations. If $g_{k+1}^T d_{k+1} > -10^{-3}\|g_{k+1}\|\|d_{k+1}\|$, then set $d_{k+1}=-g_{k+1}$		
8.	Consider $k=k+1$ and go to step 2 ♦		

Observe that the algorithm is equipped with an acceleration scheme (see step 5) introduced by Andrei [1, 2]. This scheme modifies the stepsize determined by the Wolfe line search conditions (5) in such a way as to improve the reduction of the minimizing function values along the iterations. It is proved that this acceleration scheme is linear convergent, but the reduction in the function value is significantly improved.

If f is bounded along the direction d_k, then there exists a stepsize α_k satisfying the Wolfe line search conditions (5). The first trial of the stepsize crucially affects the practical behavior of the algorithm. At every iteration $k \ge 1$, the starting guess for the step α_k in the line search is computed as $\alpha_{k-1}\|d_{k-1}\|/\|d_k\|$. Observe that in step 6 of the

algorithms if the search direction for the memory-less SR1 is not defined, i.e. if $\left|(s_k - y_k)^T y_k\right|$ is close to zero, or if the search direction for the memory-less SR1 method with generalized secant equation is not defined, i.e. if $\left|(y_k - \gamma s_k)^T y_k\right|$ is close to zero, or if the search direction for the memory-less BFGS is not defined, i.e. if $y_k^T s_k$ is close to zero, then the search direction is commuted to be the steepest descent. In our algorithm we introduced a restarting condition. If this restarting condition is satisfied, then the algorithm is restarted with the negative gradient. Some other restarting procedures may be implemented, but we are interested in seeing the performances of these algorithms implementing this restart condition.

7 Convergence of the MM-SR1gen method

In this section the global convergence of the MM-SR1gen algorithm is established under the following assumptions:

(A1) The level set $\Omega = \{x \in \mathbb{R}^n : f(x) \le f(x_0)\}$ is bounded, i.e. there exists a constant $B > 0$ such that for any $x \in \Omega$, $\|x\| \le B$.

(A2) The function $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and its gradient is Lipschitz continuous in a neighborhood N of Ω, i.e. there exists a constant $L > 0$ such that $\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|$, for any $x, y \in N$.

It is easy to see that under these assumptions, there exists a constant $\Gamma > 0$ such that $\|\nabla f(x)\| \le \Gamma$, for any $x \in \Omega$.

Although the search directions d_{k+1} generated by the above memory-less algorithms are always descent directions, in order to get the convergence of the algorithm we need to derive a lower bound for the stepsize α_k.

Proposition 2 *Suppose that d_k is a descent direction and ∇f satisfies the Lipschitz condition*

$$\|\nabla f(x) - \nabla f(x_k)\| \le L\|x - x_k\|$$

for all x on the line segment connecting x_k and x_{k+1}, where L is a constant. If the line search satisfies the standard Wolfe conditions (5)*, then*

$$\alpha_k \ge \frac{1-\sigma}{L}\frac{\left|g_k^T d_k\right|}{\|d_k\|^2}. \tag{28}$$

Proof Subtracting $g_k^T d_k$ from both sides of (5.b) and using the Lipschitz condition, it follows that

$$(\sigma - 1)g_k^T d_k \le (g_{k+1} - g_k)^T d_k \le \alpha_k L\|d_k\|^2.$$

But d_k is a descent direction and $\sigma < 1$, therefore (28) follows from the above inequality.♦

Proposition 3 *Suppose that* $(y_k - \gamma_k s_k)^T y_k < 0$ *and* $(y_k - \gamma_k s_k)(y_k - \gamma_k s_k)^T$ *is a matrix bounded above in norm. Then, the memory-less SR1 with generalized secant equation update matrix* H_{k+1} *given by* (21) *is bounded above in norm.*

Proof. From (21), since the denominator $(y_k - \gamma_k s_k)^T y_k$ is strictly negative, the numerator $(y_k - \gamma_k s_k)(y_k - \gamma_k s_k)^T$ is bounded above in norm and $\|s_k\| = \|x_{k+1} - x_k\| \leq \|x_{k+1}\| + \|x_k\| \leq 2B$, (see assumption (A1)), it follows that H_{k+1} given by (21) is bounded above in norm. ♦

Proposition 4 *Suppose that* $(y_k - \gamma_k s_k)^T y_k < 0$ *Then the memory-less SR1 with generalized secant equation search direction* (22), *is a descent direction, that is* $g_{k+1}^T d_{k+1} < 0$. *Besides,* $\|d_{k+1}\| \leq D$, *for some* $D > 0$.

Proof. Observe that the numerator of the update (21) is a rank-one positive semidefinite matrix and the denominator of (21) is strictly negative. Therefore the update matrix H_{k+1} given by (21) is positive semidefinite. Without loss of generality, H_{k+1} given by (21) is positive definite. Thus, $g_{k+1}^T d_{k+1} = -g_{k+1}^T H_{k+1} g_{k+1} < 0$.
The boundedness of d_{k+1} may be established as follows. If the restart condition is satisfied, then $d_{k+1} = -g_{k+1}$. Therefore, $\|d_{k+1}\| = \|g_{k+1}\| < D$, where $D = \Gamma$ by assumption (A2). On the other hand, for non-restart iterations the search direction is computed as $d_{k+1} = -H_{k+1} g_{k+1}$, where H_{k+1} is given by (21). In this case, by Proposition 3 the sequence of the Hessian matrices given by (21) remains bounded above in norm. Therefore, $\|d_{k+1}\| = \|-H_{k+1} g_{k+1}\| \leq \|H_{k+1}\| \|g_{k+1}\|$, which is bounded above by some constant $D > 0$. ♦

Theorem 1 *Suppose that the assumptions (A1) and (A2) are satisfied. If* f *is Lipschitz continuous, then* $\lim_{k \to \infty} \|g_k\| = 0$.

Proof The assumptions and the above Propositions 2, 3 and 4 show that the algorithm MM-SR1gen is globally convergent to a point in which the first-order optimality conditions are satisfied. ♦

8 Numerical results
In this section we report some numerical results obtained with MM-SR1, MM-SR1gen and with MM-BFGS methods for solving a collection of 800 unconstrained optimization problems. The codes are written in Fortran and compiled with f77 (default compiler settings) on a Workstation Intel Pentium 4 with 1.8 GHz. We selected a number of 80 large-scale unconstrained optimization test functions in generalized or extended form we presented in [3]. The vast majority of these problems are taken from CUTE collection [8]. For each test function we have taken ten numerical experiments with an increasing number of variables. Therefore, a number of 800 unconstrained optimization problems have been considered. The algorithms used in all numerical test implement the Wolfe line search conditions (5) with $\rho = 0.0001$, $\sigma = 0.8$ and the same stopping criterion

$\|g_k\|_\infty \le 10^{-6}$, where $\|.\|_\infty$ is the maximum absolute component of a vector. In all algorithms, we considered in our numerical studies, the maximum number of iterations is limited to 10000, while the maximum number of function and its gradient evaluations is limited to 10000. In our numerical experiments we selected: $\varepsilon = 10^{-15}$, $\varepsilon_A = 10^{-14}$ and $\varepsilon_q = 10^{-9}$. In MM-SR1gen the parameter γ_k is computed as $\gamma_k = 100(y_k^T y_k / s_k^T y_k)$.
The comparisons of algorithms are given in the following context. Let f_i^{ALG1} and f_i^{ALG2} be the optimal value found by ALG1 and ALG2, for problem $i = 1,\ldots,800$, respectively. We say that, in the particular problem i, the performance of ALG1 was better than the performance of ALG2 if:

$$\left| f_i^{ALG1} - f_i^{ALG2} \right| < 10^{-3} \tag{29}$$

and the number of iterations (#iter), or the number of function-gradient evaluations (#fg), or the CPU time of ALG1 was less than the number of iterations, or the number of function-gradient evaluations, or the CPU time corresponding to ALG2, respectively.

In the first set of numerical experiments MM-SR1gen is compared versus MM-SR1 for solving this set of 800 unconstrained minimization problems with the number of variables in the range [1000, 10000]. Figure 1 shows the Dolan and Moré [17] performance profiles of these algorithms.

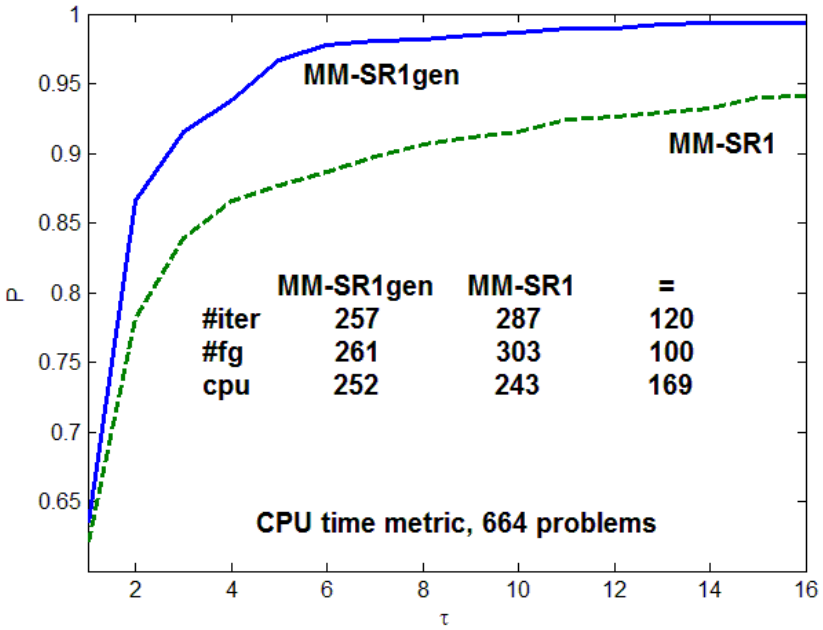

Fig. 1. MM-SR1gen versus MM-SR1, range [1000, 10000]

In a performance profile plot, the top curve corresponds to the method that solved the most problems in a time that was within a given factor of the best time. The percentage of

the test problems for which a method is the fastest is given on the left axis of the plot. The right side of the plot gives the percentage of the test problems that were successfully solved by these algorithms, respectively. Mainly the left side of the plot is a measure of the efficiency of an algorithm, while the right side is a measure of the robustness of an algorithm.
From Figure 1 we have the computational evidence that MM-SR1gen is slightly more efficient and obviously more robust. Comparing these two algorithms we see that subject to the number of iterations, MM-SR1gen was better in 257 problems, (i.e. it achieved the minimum number of iterations in solving 257 problems). MM-SR1 was better in 287 problems and they achieved the same number of iterations in solving 120 problems, etc. Out of 800 problems considered in this numerical experiment, only for 664 problems does the criterion (29) hold.

It is worth mentioning that for solving all 800 unconstrained optimization problems, MM-SR1 needed a number of 1087913 iterations out of which in 96657 iterations (i.e. a percentage of 8.88%) the negative gradient were used. In contrast, MM-SR1gen needed 589593 iterations, out of which in only 11227 iterations (i.e. a percentage of 1.90%) the negative gradient was used. There is a great difference between the search direction of MM-SR1 given by (14) and the search direction of MM-SR1gen given by (22). In MM-SR1 no information is accumulated along the iterations, the denominator $(s_k - y_k)^T y_k$ cannot be modified and consequently in a large number of iterations the steepest descent is used. On the other hand, in MM-SR1gen, at every iteration, the value of the scaling parameter γ_k from the generalized secant equation, computed as in (23), is used in the denominator $(y_k - \gamma_k s_k)^T y_k$ of the updating term of the search direction. Since $(y_k - \gamma_k s_k)^T y_k$ may be modified through an adequate value of γ_k in order to be negative, it follows that in a smaller number of iterations the steepest descent is used. This is motivation that MM-SR1gen is more efficient and more robust versus MM-SR1.

In the second set of numerical experiments MM-SR1gen versus MM-BFGS are compared for solving this set of 800 unconstrained minimization problems with the number of variables in the range [1000, 10000]. Figure 2 shows the Dolan and Moré [17] performance profiles of these algorithms.
From Figure 2 we see that MM-SR1gen algorithm is more efficient and more robust than the MM-BFGS method. Indeed, comparing MM-SR1gen versus MM-BFGS subject to the number of iterations, we see that MM-SR1gen was better in 308 problems (i.e. it achieved the minimum number of iterations in 308 problems). MM-BFGS was better in 315 problems and they achieved the same number of iterations in solving 101 problems, etc. Out of 800 problems considered in this numerical experiment, only for 724 problems does the criterion (29) hold. Subject to the CPU computing time, from Figure 2 we see that MM-SR1gen was faster in 305 problems, while MM-BFGS was faster only in 256 problems. Notice that both MM-SR1gen and MM-BFGS use the same implementation of the standard Wolfe line search (5) based on cubic interpolation [34], as well as the same optimization conditions.

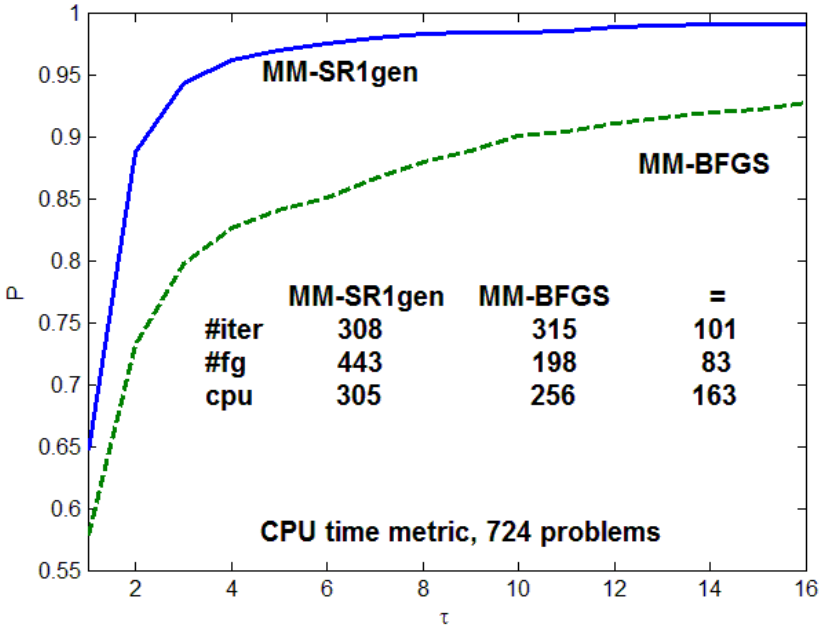

Fig. 2. MM-SR1gen versus MM-BFGS, range [1000, 10000]

In the third set of numerical experiments let us compare the performances of MM-SR1gen versus BFGS from CONMIN [35]. CONMIN package includes two optimization algorithms: a BFGS preconditioned conjugate gradient one and a variant of the BFGS quasi-Newton algorithm.

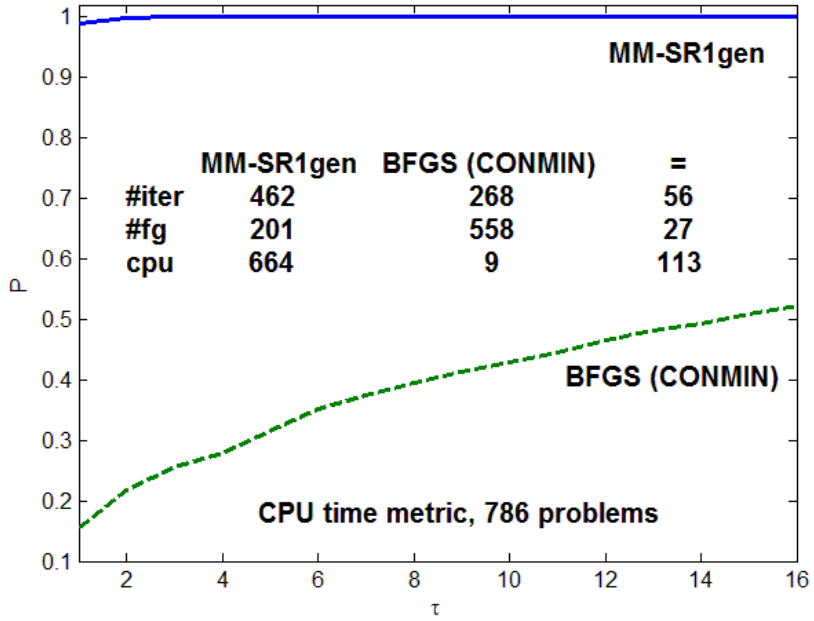

Fig. 3. MM-SR1gen versus BFGS from CONMIN, range [100, 1000]

Figure 3 shows the performance profiles of these algorithms for solving 800 unconstrained optimization problems from our collection with the number of variables in the range [100, 1000].
Obviously, MM-SR1gen is way more efficient and more robust than BFGS from CONMIN, one of the best implementation of this quasi-Newton method. Observe that subject to the CPU computing time MM-SR1 was faster in 664 problems, while BFGS from CONMIN was faster only in 9 problems. The BFGS from CONMIN is a variable metric method with initial scaling which approximately needs $n^2/2+11n/2$ double precision words of working storage. In comparison MM-SR1gen requires approximately $6n$ double precision words of working storage. BFGS requires more memory and involves a greater computational effort. We emphasize that at every iteration BFGS from CONMIN update an approximation to the Hessian by accumulating the information from the previous iterations. On the other hand memory-less MM-SR1gen algorithm at every iteration update the identity matrix (see (20) or (21) for the inverse Hessian updating). Obviously, MM-SR1gen doesn't accumulate the information from iteration to iteration when updating the approximation to the Hessian. However, MM-SR1gen having a very simple updating formula is way more efficient and more robust than BFGS from CONMIN.

In the last set of numerical experiments let us present comparisons between MM-SR1gen and MM-BFGS algorithms for solving five applications from the MINPACK-2 test problem collection [5]. MINPACK-2 contains applications from different fields, such as: elasticity, fluid dynamics, combustion, lubrication, molecular conformation, nondestructive testing, chemical kinetics, etc. Table 1, presents the applications considered in our numerical experiments, as well as the values of their parameters. The infinite-dimensional version of these problems is transformed into a finite element approximation by triangulation. Thus a finite-dimensional minimization problem is obtained whose variables are the values of the piecewise linear function at the vertices of the triangulation. The discretization steps are $nx=200$ and $ny=200$, thus obtaining minimization problems with $nx\times ny=40{,}000$ variables.

Table 1
Applications from the MINPACK-2 collection

A1	Elastic–plastic torsion [21, pp. 41–55], c = 5
A2	Pressure distribution in a journal bearing [12], b = 10, ε = 0.1
A3	Optimal design with composite materials [23], λ = 0.008
A4	Steady-state combustion [4, pp. 292–299], [6], λ = 5
A5	Minimal surfaces with Enneper conditions [27, pp. 80–85]

The performances of the MM-SR1gen method versus MM-SR1 and versus the MM-BFGS method are given in Table 2, where #iter is the number of iterations, #fg is the number of function and its gradient evaluations, #ig is the number of iterations in which the search direction is the negative gradient and cpu is the CPU time computing for solving these applications.

Table 2
Performances of MM-SR1gen versus MM-SR1 and versus MM-BFGS
(40,000 variables, CPU seconds)

	MM-SR1gen				**MM-SR1**				**MM-BFGS**			
	#iter	**#fg**	**cpu**	**#ig**	**#iter**	**#fg**	**cpu**	**#ig**	**#iter**	**#fg**	**cpu**	**#ig**
A1	372	772	7.79	0	13138	26297	326.04	1	6711	20133	180.56	6611
A2	1257	2547	31.97	0	66930	133892	1343.10	0	9312	27937	298.50	8883
A3	4093	10001	202.28	0	88142	176330	4915.73	822	861	2584	53.28	15
A4	609	1260	65.10	0	49631	99287	3708.33	1	666	1997	140.00	416
A5	308	697	9.47	0	11370	22775	320.38	1	2177	6520	148.46	199
Total	**6639**	**15277**	**316.61**	**0**	**229211**	**458481**	**10613.58**	**825**	**19727**	**59171**	**820.80**	**16124**

From Table 2 we see that MM-SR1gen is more efficient for solving these applications from MINPACK-2 collection, each of them with 40,000 variables. For solving all these applications MM-SR1gen needs 316.61 seconds, while MM-SR1 needs 10613.58 seconds, and MM-BFGS needs 820.80 seconds. In other words, MM-SR1gen is 33.52 times faster than MM-SR1 and MM-SR1gen is 2.60 times faster than MM-BFGS. It is worth showing that for solving all these applications MM-SR1gen does not use the negative gradient in any iteration. This is in sharp contrast with MM-SR1 and MM-BFGS. In case of MM-SR1 out of 229211 iterations in exactly 825 iterations (i.e. 0.35%) the negative gradient was used. For MM-BFGS out of 19727 iterations for solving all five applications, in exactly 16124 iterations (i.e. 81.73%) the negative gradient was used.

Both these algorithms, MM-SR1gen, MM-SR1 and MM-BFGS, are memory-less, i.e. they do not accumulate the information about Hessian from iteration to iteration. They use the same initialization and exactly the same implementation of the Wolfe line search conditions (5). The differences are in the formula for the search direction computation. Observe that the search direction corresponding to MM-SR1gen (22), based on the generalized secant equation (15), is obtained from the memory-less inverse SR1 method (21) which involves the scalar parameter γ_k, updated at every iteration as in (23). On the other hand, the search direction corresponding to MM-SR1 (14), based on the secant equation (6), is obtained from the memory-less SR1 method (13). Similarly, the search direction corresponding to MM-BFGS (27), based on the secant equation (6), is obtained from the memory-less BFGS method (26). Therefore, in contrast to MM-SR1 and MM-BFGS, the proposed memory-less inverse SR1 method (21) accumulates information from an unlimited number of past iterations without storing any history. This is the reason why the memory-less SR1 method with generalized secant equation is more efficient and more robust versus the memory-less SR1 or the memory-less BFGS methods.

9 Conclusions

The memory-less SR1 method with generalized secant equation prove to be one of the best quasi-Newton method for solving large-scale unconstrained optimization problems. This method involved two ingredients: the memory-less technique and the generalized secant equation. By the memory-less technique in the formula for updating the Hessian, at every iteration, instead of using the approximation to the Hessian from the previous

iteration the identity matrix is used. The generalized secant equation, used in this method, is derived from the requirement that the quadratic model of the minimizing function with scaled approximation to the Hessian should match the gradient of the minimizing function at the latest two iterations. The convergence of the algorithm is proved under the classical assumptions. Comparisons for solving a large class of unconstrained optimization problems with different structures and complexities, and large-scale applications from MINPACK-2 collection showed that the memory-less SR1 method with generalized secant equation is more efficient and more robust than the memory-less SR1, than the memory-less BFGS method and than the BFGS from CONMIN. The advantage of the memory-less SR1 with generalized secant equation over the memory-less SR1 and over the memory-less BFGS method is that the formula for updating the inverse Hessian of this method, accumulate information from an unlimited number of past iterations without storing any history of iterations.

References

1. Andrei, N.: An acceleration of gradient descent algorithm with backtracking for unconstrained optimization. Numerical Algorithms, **42**(1), 63-73 (2006)
2. Andrei, N.: Acceleration of conjugate gradient algorithms for unconstrained optimization. Applied Mathematics and Computation, **213**(2), 361-369 (2009)
3. Andrei, N.: Nonlinear conjugate gradient methods for unconstrained optimization. Springer Optimization and Its Applications 158 (2020)
4. Aris, R.: The Mathematical Theory of Diffusion and Reaction in Permeable Catalysts, Oxford (1975)
5. Averick, B.M., Carter, R.G., Moré, J.J., Xue, G.L.: The MINPACK-2 test problem collection, Mathematics and Computer Science Division, Argonne National Laboratory, Preprint MCS-P153-0692 (1992)
6. Bebernes, J., Eberly, D.: Mahematical Problems from Combustion Theory, in: Applied Mathematical Sciences, vol. 83, Springer-Verlag (1989)
7. Benson, H.Y., Shanno, D.F.: Cubic regularization in symmetric rank-1 quasi-Newton methods. Math. Progr. Comput. **10**, 457-486 (2018)
8. Bongartz, I., Conn, A.R., Gould, N.I.M., Toint, Ph.L.: CUTE: constrained and unconstrained testing environments. ACM Transactions on Mathematical Software, **21**, 123-160 (1995)
9. Broyden, C.G.: Quasi-newton methods and their applications to function minimization. Mathematics of Computation **21**(99), 368-381 (1967)
10. Broyden, C.G.: The convergence of a class of double-rank minimization algorithms. I. General considerations. Journal of the Institute of Mathematics and Its Applications, **6**, 76-90 (1970)
11. Chen, H., Lam, W.H., Chan, S.C.: On the convergence analysis of cubic regularized symmetric rank-1 quasi-Newton method and the incremental version in the application of large-scale problems. IEEE Access, volume 7, 114042-114059 (2019)
12. Cimatti, G.: On a problem of the theory of lubrication governed by a variational inequality, Applied Mathematics and Optimization **3**, 227–242 (1977)

13. Conn, A.R., Gould, N.I.M., Toint, Ph.L.: Convergence of quasi-newton matrices generated by the symmetric rank one update. Mathematical Programming **50**(1-3), 177-195 (1991)
14. Dai, Y.H., Liao, L.Z.: New conjugate conditions and related nonlinear conjugate gradient methods, Applied Mathematics & Optimization, **43**, 87-101 (2001)
15. Davidon, W.C.: Variable metric method for minimization. (Research and Development Report ANL-5990. Argonne National Laboratories.) (1959)
16. Davidon, W.C.: Variable metric method for minimization. SIAM J. Optim. **1**(1), 1-17 (1991)
17. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Mathematical Programming, **91**, 201-213 (2002)
18. Fiacco, A.V., McCormick, G.P.: Nonlinear programming: sequential unconstrained minimization techniques. Research Analysis Corporation, McLean Virginia, (1968). Republished in 1990 by SIAM, Philadelphia.
19. Fletcher, R., Powell, M.J.D.: A rapidly convergent descent method for minimization, Computer Journal, 163–168 (1963)
20. Fletcher, R.: A new approach to variable metric algorithms. The Computer Journal, **13**, 317-322 (1970)
21. Glowinski, R.: Numerical Methods for Nonlinear Variational Problems, Springer-Verlag, Berlin (1984)
22. Goldfarb, D.: A family of variable metric method derived by variation mean. Mathematics of Computation, **23**, 23-26 (1970)
23. Goodman, J., Kohn, R., Reyna, L.: Numerical study of a relaxed variational problem from optimal design, Computer Methods in Applied Mechanics and Engineering **57**, 107–127 (1986)
24. Huang, H.Y.: Unified approach to quadratically convergent algorithms for functions minimization. Journal of Optimization Theory and Applications **5**(6), 405-423 (1970)
25. Kelley, C.T., Sachs, E.W.: Local convergence of the symmetric rank one iteration. Computational Optimization and Applications, **9**, 43–63 (1998)
26. Khalfan, H.F., Byrd, R.H., Schnabel, R.B.: A theoretical and experimental study of the symmetric rank-one update. SIAM J. Optimization **3**(1), 1-24 (1993)
27. Nitsche, J.C.C.: Lectures on Minimal Surfaces, Vol. 1, Cambridge University Press (1989)
28. Nocedal, J.: Theory of algorithms for unconstrained optimization. Acta Numerica, **1,** 199-242 (1992)
29. Nocedal, J., Wright, S.J.: Numerical optimization. Springer Series in Operations Research. Springer Science+Business Media, New York, Second edition, (2006)
30. Powell, M.J.D.: A new algorithm for unconstrained optimization. In: J.B. Rosen, O.L. Mangasarian, K. Ritter (Eds.) Nonlinear Programming. Academic Press, New York, 31-66 (1970)
31. Shanno, D.F.: Conditioning of quasi-Newton methods for function minimization. Mathematics of Computation, **24**, 647-656 (1970)
32. Shanno, D.F.: Conjugate gradient methods with inexact searches, Mathematics of Operations Research, **3**, 244-256 (1978)

33. Shanno, D.F.: On the convergence of a new conjugate gradient algorithm, SIAM Journal on Numerical Analysis, **15**, 1247-1257 (1978)
34. Shanno, D.F.: CONMIN – A Fortran subroutine for minimizing an unconstrained nonlinear scalar valued function of a vector variable x either by the BFGS variable metric algorithm or by a Beale restarted conjugate gradient algorithm. Private communication, October 17 (1983)
35. Shanno, D.F., Phua, K.H.: Algorithm 500. Minimization of unconstrained multivariable functions. ACM Transactions on Mathematical Software, **2**, 87-94 (1976)
36. Wolfe, P.: Convergence conditions for ascent methods. SIAM Review, **11**, 226-235 (1969)
37. Wolfe, P.: Convergence conditions for ascent methods. II: Some corrections. SIAM Review, **13**, 185-188 (1971)

May 28, 2021

-----oooooOoooooo-----