Bull. Aust. Math. Soc. **98** (2018), 331–338
doi:10.1017/S0004972718000552

INITIAL IMPROVEMENT OF THE HYBRID ACCELERATED GRADIENT DESCENT PROCESS

STEFAN PANIĆ, MILENA J. PETROVIĆ✉ and MIROSLAVA MIHAJLOV CAREVIĆ

(Received 8 January 2018; accepted 21 May 2018; first published online 1 August 2018)

Abstract

We improve the convergence properties of the iterative scheme for solving unconstrained optimisation problems introduced in Petrovic *et al.* [‘Hybridization of accelerated gradient descent method’, *Numer. Algorithms* (2017), doi:10.1007/s11075-017-0460-4] by optimising the value of the initial step length parameter in the backtracking line search procedure. We prove the validity of the algorithm and illustrate its advantages by numerical experiments and comparisons.

2010 *Mathematics subject classification*: primary 65K05; secondary 90C30, 90C53.

Keywords and phrases: line search, gradient descent methods, Newton method, convergence rate.

1. Introduction

We analyse hybridisation of the accelerated gradient decent model, known as the HSM method. This algorithm is based on the Picard–Mann hybrid iteration presented in [3] combined with the accelerated characteristics of the gradient descent SM method from [9]. The HSM scheme is efficient and preserves the excellent convergence properties of the SM method as shown in [5]. It is described iteratively by

$$x_{k+1} = x_k + t_k d_k, \tag{1.1}$$

where x_k is the value of the objective function at the current iterative point and x_{k+1} is the value for the next iteration. The iterative step-size t_k is commonly calculated using one of the line search algorithms. The parameter d_k is a vector direction at the kth iteration. These two parameters dictate the efficiency of the iteration and control the number of iterations, the number of function evaluations and the CPU computation time.

We apply the HSM iterative algorithm to solve the unconstrained optimisation problem

$$\min f(x), \quad x \in \mathbb{R}^n, \tag{1.2}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is assumed to be uniformly convex and twice continuously differentiable. We also use the notation

$$g(x) = \nabla f(x), \quad G(x) = \nabla^2 f(x), \quad g_k = \nabla f(x_k), \quad G_k = \nabla^2 f(x_k),$$

where $\nabla f(x)$ denotes the gradient of f and $\nabla^2 f(x)$ denotes the Hessian. The step length parameter is determined by a backtracking algorithm and the vector direction has the form of accelerated gradient descent. More precisely,

$$d_k = -(\alpha_k + 1)\gamma_k^{-1} g_k, \tag{1.3}$$

where $\alpha_k \in (0, 1)$ is the parameter adopted in [3]. Using (1.1) and (1.3), we can write the HSM iteration step as

$$x_{k+1} = x_k - (\alpha_k + 1)t_k \gamma_k^{-1} g_k. \tag{1.4}$$

There are several conjugate and modified gradient descent algorithms [2, 7, 8, 11], as well as accelerated gradient models [4, 6, 10], comparable to the HSM method and its modifications. We can make some comparisons between the various approaches in terms of the vector direction d_k and the step-size value t_k.

Generally, we assume that the vector direction satisfies the descent condition

$$g_k^T d_k < 0.$$

In the Newton method equipped with the line search procedure, the vector direction is determined as a solution of the equation

$$G_k^T d = -g_k.$$

Some conjugate gradient methods calculate the vector direction by means of

$$d_k = \begin{cases} -g_k, & k = 0, \\ -g_k + \beta_k d_{k-1}, & k \geq 1. \end{cases} \tag{1.5}$$

For example, for the Polak–Ribière–Polyak method [7, 8], the scalar β_k is determined by

$$\beta_k^{\mathrm{PRP}} = \frac{g_k^T (g_k - g_{k-1})}{g_{k-1}^T g_{k-1}}.$$

In [11], the sufficient descent condition

$$g_k^T d_k < -c\|g_k\|^2, \quad k \geq 0,\ c > 0.$$

is used and a modified version β_k^{MPRP} of the scalar β_k^{PRP} is introduced to describe several conjugate gradient methods. For another conjugate gradient method, the so-called the *Cg-descent iteration* [2], the scalar β_k is defined by

$$\overline{\beta_k}^{\mathrm{N}} = \max\left\{\frac{1}{d_k^T y_k}\left(y_k - 2\frac{d_k\|y_k\|^2}{d_k^T y_k}\right)^T g_{k+1},\ -\frac{1}{\|d_k\| \min\{\eta, \|g_k\|\}}\right\},$$

where $\eta > 0$ is a constant.

The methods presented in [4, 6, 9, 10] can be classified as methods of quasi-Newton type with accelerated approximation of the Hessian inverse, equipped with the line search technique. As in [9], we refer to these methods simply as *accelerated gradient*

https://doi.org/10.1017/S0004972718000552 Published online by Cambridge University Press

descent algorithms with line search. The scheme presented in this paper belongs to this class of algorithms. Accelerated gradient descent methods generally can be defined by

$$x_{k+1} = x_k - t_k\gamma_k^{-1}g_k,$$

where $\gamma_k = \gamma(x_k, x_{k-1}) \in \mathbb{R}$ and γ_k^{-1} is a scalar approximation to the inverse of the Hessian of the objective function, that is, $G_k = \nabla^2 f(x_k) = \gamma_k I$. This so-called *acceleration parameter* is calculated from the second-order Taylor series of the relevant iteration at two successive points. So, unlike the model (1.5) which defined the vector direction of conjugate gradient methods, in accelerated gradient descent methods [4, 6, 9, 10] the vector direction is the product of the negative gradient vector and a derived accelerated parameter.

Next, we consider the determination of the iterative step length, t_k. In the HSM model and in its modification presented in this paper, as well as in all the references mentioned above, a line search procedure is used to calculate t_k. In some cases, this is an exact line search algorithm given by

$$t_k = \arg\min_{t>0} f(x_k + td_k).$$

Theoretically, this algorithm outputs the optimal iterative step length value. But in practice the computation requires a lot of CPU time and generally reduces the efficiency. For this reason, inexact line search techniques which sufficiently decrease the value of the objective function f along the ray $x_k + td_k$, $t > 0$, are preferred. For example, in [2, 11] the Wolfe–Powell inexact procedures are used. In the scheme presented in this paper, as in [4, 6, 9, 10], we use Armijo's backtracking line search condition.

The paper is organised as follows. In Section 2 we describe the modified HSM method by deriving the correction for the initial step size parameter in the backtracking procedure. We apply this initial step size in Section 3 to prove the convergence of the modified HSM method. In Section 4 we give numerical results to compare the performance of the HSM method of [5] and the modified HSM method developed in this paper. The already good performance characteristics of the hybrid accelerated method, which are proved and numerically tested in [5], are notably improved using the initial correction of the starting value for the backtracking line search procedure.

2. Initial value correction in the backtracking procedure

We now return to the HSM method described by (1.1) for solving the unconstrained optimisation problem (1.2). The vector direction d_k is defined by (1.3) and the step length t_k is derived by applying Armijo's backtracking line search procedure described in Algorithm 2.1.

We derive the optimal value of the initial step length, which depends on the parameter α_k adopted in [3]. For that purpose we analyse the function

$$\Phi_{k+1}(t) = f(x_{k+1}) - (\alpha_{k+1}+1)tg_{k+1}^T\gamma_{k+1}^{-1}g_{k+1} + \tfrac{1}{2}(\alpha_{k+1}+1)^2t^2(\gamma_{k+1}^{-1}g_{k+1})^T\nabla^2 f(\xi)\gamma_{k+1}^{-1}g_{k+1},$$

https://doi.org/10.1017/S0004972718000552 Published online by Cambridge University Press

Algorithm 2.1 Armijo's backtracking line search starting from $t = 1$.

```
Require: Objective function $f(x)$, the direction $d_k$ of the search at the point $x_k$ and
    numbers $\sigma, \beta$ with $0 < \sigma < 0.5$ and $\beta \in (0, 1)$.
1: $t = 1$.
2: While $f(x_k + td_k) > f(x_k) + \sigma t g_k^T d_k$, take $t := t\beta$.
3: Return $t_k = t$.
```

where $\xi \in [x_{k+1}, x_{k+2}]$, $t \geq 0$ and $\gamma_{k+1} > 0$. If we use the substitution $\xi \approx x_{k+1}$,

$$\begin{aligned}\Phi_{k+1}(t) &= f(x_{k+1}) - (\alpha_{k+1} + 1)t\gamma_{k+1}^{-1}\|g_{k+1}\|^2 + \tfrac{1}{2}(\alpha_{k+1} + 1)^2 t^2 \gamma_{k+1}\gamma_{k+1}^{-2}\|g_{k+1}\|^2 \\ &= f(x_{k+1}) - (\alpha_{k+1} + 1)t\gamma_{k+1}^{-1}\|g_{k+1}\|^2 + \tfrac{1}{2}(\alpha_{k+1} + 1)^2 t^2 \gamma_{k+1}^{-1}\|g_{k+1}\|^2.\end{aligned}$$

Clearly, the function Φ is convex when $\gamma_{k+1} > 0$ and

(1) $\Phi_{k+1}(0) = f(x_{k+1})$,

(2) $\Phi'_{k+1}(t) = (\alpha_{k+1} + 1)(\alpha_{k+1}t + t - 1)\gamma_{k+1}^{-1}\|g_{k+1}\|^2$.

So the function Φ decreases when $\Phi'_{k+1}(t) < 0$ and that is true for $t \in (0, 1/(\alpha_{k+1} + 1))$. Moreover,

$$\Phi'_{k+1}(t) = 0 \quad \Leftrightarrow \quad t = \frac{1}{\alpha_{k+1} + 1}. \tag{2.1}$$

Since the minimum of Φ_{k+1} is achieved for $t = 1/(\alpha_{k+1} + 1)$, we use this step length value for starting the backtracking line search procedure. We adopt the same acceleration parameter γ_k introduced for the HSM method in [5]:

$$\gamma_{k+1} = 2\gamma_k \frac{\gamma_k[f(x_{k+1}) - f(x_k)] + (\alpha_k + 1)t_k\|g_k\|^2}{(\alpha_k + 1)^2 t_k^2 \|g_k\|^2}. \tag{2.2}$$

The modified HSM algorithm (MHSM) is described in Algorithm 2.2. We reset the third step in Algorithm 2.1 to derive the iterative step-size (2.1).

Algorithm 2.2 The MHSM algorithm defined by (1.4), (2.1) and (2.2).

```
Require: Function $f(x)$, $\alpha \in (1, 2)$, initial point $x_0 \in \mathrm{dom}(f)$.
1: Set $k = 0$ and calculate $f(x_0)$, $g_0 = \nabla f(x_0)$; set $\gamma_0 = 1$.
2: Check the test criteria: if stopping criteria are fulfilled then stop the algorithm;
    otherwise, go to the next step.
3: Apply Algorithm 2.1: compute the value of step size $t_k \in (0, 1/\alpha]$ by taking
    $d_k = -\gamma_k^{-1} g_k$.
4: Determine $x_{k+1} = x_k - \alpha t_k \gamma_k^{-1} g_k$, $f(x_{k+1})$ and $g_{k+1} = \nabla f(x_{k+1})$.
5: Compute $\gamma_{k+1}$, approximating the Hessian of $f$ at the point $x_{k+1}$ using (2.2).
6: If $\gamma_{k+1} < 0$, take $\gamma_{k+1} = 1$.
7: $k := k + 1$; go to the step 2.
8: Return $x_{k+1}$ and $f(x_{k+1})$.
```

https://doi.org/10.1017/S0004972718000552 Published online by Cambridge University Press

Remark 2.1. Clearly, $\alpha_k \in (0, 1)$ implies $\alpha_k + 1 \in (1, 2)$. In our further investigation, we make the simplest choice by taking $\alpha_k + 1$ as a constant $\alpha \in (1, 2)$. This choice is applied in Algorithm 2.2 and it transforms the main iteration (1.4) into

$$x_{k+1} = x_k - \alpha t_k \gamma_k^{-1} g_k.$$

3. Convergence properties

With one exception dealt with below, the convergence analysis in [5] applies to the MHSM scheme and we will not repeat these arguments. This analysis guarantees that the MHSM algorithm is convergent and well defined. However, [5, Lemma 3.2], which shows that the objective function decreases at each iteration, no longer applies to MHSM and must be replaced by Lemma 3.1 which we prove next.

Lemma 3.1. *For the twice continuously differentiable and uniformly convex function f defined on $\mathbb{R}^n$ and the sequence $\{x_k\}$ defined by Algorithm 2.2,*

$$f(x_k) - f(x_{k+1}) \geq \nu \|g_k\|^2,$$

where

$$\nu = \min\left\{\frac{\sigma\alpha}{2M}, \frac{\sigma(1-\sigma)}{L}\beta\right\},$$

$L > 0$ is the Lipschitz constant, $M \geq 1$ is a constant such that $\gamma_k < M$ and $\alpha \in (1, 2)$ is the parameter defined in Remark 2.1.

Remark 3.2. The constant $M \geq 1$ is defined in [5, Lemma 3.1]. The estimate $\gamma_k < M$ comes from the fact that γ_k approximates the kth Hessian.

Proof. From (2.1), the starting point for the backtracking line search procedure is $t = 1/\alpha$ with $\alpha \in (1, 2)$, so $t \in (\frac{1}{2}, 1)$. We analyse the left and the right boundary values, $t_k > \frac{1}{2}$ and $t_k < 1$. The analysis of the case $t_k < 1$ is the same as described in [5], giving the final inequalities

$$t_k > \frac{\beta(1-\sigma)\gamma_k}{L\alpha} \tag{3.1}$$

and

$$f(x_k) - f(x_{k+1}) > \frac{\sigma(1-\sigma)\beta}{L}\|g_k\|^2. \tag{3.2}$$

For the left boundary value with $t_k > \frac{1}{2}$, we apply the exit condition for the backtracking algorithm, the iterative vector direction $d_k = -\alpha\gamma_k^{-1} g_k$ in the MHSM algorithm and relation (3.1), giving

$$f(x_k) - f(x_{k+1}) \geq -\sigma t_k g_k^T d_k = \sigma t_k g_k^T \alpha \gamma_k^{-1} g_k > \frac{1}{2}\frac{\sigma\alpha g_k^T g_k}{\gamma_k} = \frac{1}{2}\frac{\sigma\alpha}{\|g_k\|^2}. \tag{3.3}$$

The desired conclusion follows from (3.2) and (3.3). □

The linear convergence of the MHSM algorithm is equivalent to the similar result for the HSM algorithm in [5].

https://doi.org/10.1017/S0004972718000552 Published online by Cambridge University Press

THEOREM 3.3. *For the twice continuously differentiable and uniformly convex function f defined on $\mathbb{R}^n$ and the sequence $\{x_k\}$ generated by Algorithm 2.2,*

$$\lim_{k\to\infty} \|g_k\| = 0,$$

and the sequence $\{x_k\}$ converges to x^* at least linearly.

As in [5], we investigate the convergence of MHSM on the set of strictly convex quadratic functions

$$f(\mathbf{x}) = \tfrac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x}, \tag{3.4}$$

where A is a real $n \times n$ symmetric positive definite matrix and $\mathbf{b} \in \mathbb{R}^n$. The proof is concluded by means of the following statements for MHSM which are proved in the same way as Lemma 4.1 and Theorem 4.1 in [5]. We omit the proofs.

LEMMA 3.4. *Suppose $A \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix and λ_1 and λ_n are respectively the smallest and the largest eigenvalues of A. When the MHSM Algorithm 2.2 is applied on the strictly convex quadratic function f given by (3.4),*

$$\lambda_1 \le \frac{\gamma_{k+1}}{t_{k+1}} \le \frac{4\cdot\lambda_n}{\sigma}.$$

THEOREM 3.5. *Suppose $A \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix, λ_1 and λ_n are respectively the smallest and the largest eigenvalues of A, and $\lambda_n < 2\lambda_1/(\alpha_k + 1)$. Let $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\}$ be the orthonormal set of eigenvectors of the matrix A. Let $\{\mathbf{x}_k\}$ be the sequence of values constructed by Algorithm 2.2 applied to the function (3.4), and write the gradients $\mathbf{g}_k = A\mathbf{x}_k - \mathbf{b}$ of (3.4) as*

$$\mathbf{g}_k = \sum_{i=1}^{n} d_i^k \mathbf{v}_i,$$

for some real constants $d_1^k, d_2^k, \dots, d_n^k$. Then, for $i = 1, 2, \dots$,

$$(d_i^{k+1})^2 \le \delta^2 (d_i^k)^2, \quad \delta = \max\left\{1 - \frac{\sigma\lambda_1(\alpha_k+1)}{4\lambda_n}, \frac{\lambda_n(\alpha_k+1)}{\lambda_1} - 1\right\}$$

and

$$\lim_{k\to\infty} \|\mathbf{g}_k\| = 0.$$

4. Numerical comparisons

In this section we show that the MHSM iterative scheme generally has better numerical characteristics than the HSM scheme in terms of the number of iterations, CPU time and number of function evaluations.

We tested the same set of 25 test functions selected in [5] from the test function collection [1]. As in [5], 11 numerical tests were taken on each chosen test function for the parameter values 1000, 2000, 3000, 5000, 7000, 8000, 10 000, 15 000, 20 000, 30 000 and 50 000 and the number of iterations, CPU time and number of function evaluations

https://doi.org/10.1017/S0004972718000552 Published online by Cambridge University Press

TABLE 1. Results of numerical experiments for MHSM and HSM algorithms.

	Iterations		CPU time		Function evaluations	
Test function	MHSM	HSM	MHSM	HSM	MHSM	HSM
Extended Penalty	550	504	20	23	7 806	7 343
Perturbed Quadratic	21 446	49 149	453	1402	196 877	443 917
Raydan-1	11 282	13 694	206	288	74 994	91 106
Diagonal 1	7 887	11 016	189	414	92 929	115 076
Diagonal 3	6 452	7 801	263	241	54 020	62 530
Generalized Tridiagonal – 1	443	468	0	0	1 562	1 720
Extended Himmelblau	292	308	0	0	1 068	1 023
Quadr. Diag. Perturbed	34 356	57 448	715	1461	343 321	649 740
Quadratic QF1	14 283	45 885	262	1555	12 1867	394 504
Extended Quad. Penalty QP1	339	413	7	8	2 899	3 064
Extended Quad. Penalty QP2	2 640	1 876	73	78	18 473	14 166
Quadratic QF2	27 778	49 186	1079	1955	275 850	465 172
Extended EP1	286	186	6	4	2 664	1 847
Almost Perturbed Quadratic	21 133	40 190	524	1135	194 973	359 019
Engval1	599	540	11	8	3 180	2 983
Quartc	11	11	0	0	33	33
Diagonal 6	127	140	0	0	274	290
Tridia	79 796	256 476	2382	7567	858 752	2597 726
Indef	11	11	0	0	33	33
Nonscomp	11	11	0	0	33	33
Dixon 3dq	11	11	0	0	33	33
Biggsb 1	11	11	0	0	33	33
Hager	1 023	1 031	13	18	6 004	5 879
Raydan 2	88	84	0	0	181	189
Arwhead	651	276	16	15	4 789	3 717

were examined. The stopping criteria were the same as in [5], namely

$$\|g_k\| \leq 10^{-6} \quad \text{and} \quad \frac{|f(x_{k+1}) - f(x_k)|}{1 + |f(x_k)|} \leq 10^{-16},$$

as were the backtracking parameters $\sigma = 0.0001$ and $\beta = 0.8$. The code was written in the visual C++ language and the testing carried out on a Workstation Intel Celeron 1.6 GHz.

Table 1 shows the total number of iterative steps, CPU time (in seconds) and the number of function evaluations for the HSM method in [5] and the MHSM method of this paper for the tests on each of the 25 test functions. MHSM outperforms HSM on around half the tests with respect to the number of iterations and the number of function evaluations, and also with respect to the CPU time, but on a slightly different set of tests.

A more convincing comparison of the performance of the two algorithms is revealed in Table 2. This table shows the average values of the number of iterations, the CPU time and the number of function evaluations over all the tests. MHSM outperforms HSM by more than a factor of 2.

TABLE 2. Average results for 25 test functions on 11 numerical experiments.

Average performance	MHSM	HSM
Number of iterations	9 260.24	21 469.04
CPU time (sec)	248.76	646.88
Number of function evaluations	90 505.92	208 847.09

References

[1] N. Andrei, ‘An unconstrained optimization test functions collection’, *Adv. Model. Optim.* **10**(1) (2008), 147–161.

[2] W. Hager and H. Zhang, ‘Algorithm 851: a conjugate gradient method with guaranteed descent’, *ACM Trans. Math. Software* **32** (2006), 113–137.

[3] S. H. Khan, ‘A Picard–Mann hybrid iterative process’, *Fixed Point Theory Appl.* **2013** (2013), 69.

[4] M. J. Petrović, ‘An accelerated double step size method in unconstrained optimization’, *Appl. Math. Comput.* **250** (2015), 309–319.

[5] M. Petrović, V. Rakocević, N. Kontrec, S. Panić and D. Ilić, ‘Hybridization of accelerated gradient descent method’, *Numer. Algorithms* (2017), doi:10.1007/s11075-017-0460-4.

[6] M. J. Petrović and P. S. Stanimirović, ‘Accelerated double direction method for solving unconstrained optimization problems’, *Math. Probl. Eng.* **2014** (2014), 965104.

[7] E. Polak and G. Ribière, ‘Note sur la convergence de directions conjuguées’, *Rev. Fran. Inf. Rech. Opérat.* **3** (1969), 35–43.

[8] B. T. Polyak, ‘The conjugate gradient method in extreme problems’, *Comput. Math. Math. Phys.* **9** (1969), 94–112.

[9] P. S. Stanimirović and M. B. Miladinović, ‘Accelerated gradient descent methods with line search’, *Numer. Algorithms* **54** (2010), 503–520.

[10] P. S. Stanimirović, G. V. Milovanović and M. J. Petrović, ‘A transformation of accelerated double step size method for unconstrained optimization’, *Math. Probl. Eng.* **2015** (2015), 283679.

[11] G. Yuan, ‘Modified nonlinear conjugate gradient methods with sufficient descent property for large-scale optimization problems’, *Optim. Lett.* **3** (2009), 11–21.

STEFAN PANIĆ, Faculty of Sciences and Mathematics,
University of Priština, LoleRibara 29,
29000 Kosovska Mitrovica, Serbia
e-mail: stefanpnc@yahoo.com

MILENA J. PETROVIĆ, Faculty of Sciences and Mathematics,
University of Priština, LoleRibara 29,
29000 Kosovska Mitrovica, Serbia
e-mail: milena.petrovic@pr.ac.rs

MIROSLAVA MIHAJLOV CAREVIĆ,
Faculty of Mathematics and Computer Science,
Alfa BK University, Palmira Toljatija 3,
11070 Belgrade, Serbia
e-mail: miroslavamihajlovcarevic@yahoo.com

https://doi.org/10.1017/S0004972718000552 Published online by Cambridge University Press