Journal of Inequalities and Applications
a SpringerOpen Journal

RESEARCH Open Access

The global proof of the Polak–Ribière–Polak algorithm under the YWL inexact line search technique

Xiangrong Li[1], Tianshan Yang[2] and Xiaoliang Wang[3*]

*Correspondence:
xliangwang@126.com
[3]School of Mathematical Sciences, Dalian University of Technology, Dalian, P.R. China
Full list of author information is available at the end of the article

Abstract

This paper presents a new proof method about the paper (Yuan et al. in Appl. Math. Model. 47:811–825, 2017). In the proof, the global convergence of the Polak–Ribière–Polak algorithm is established without these two assumptions $d_k^T g_k < 0$ and $g_{k+1} d_k \leq -\sigma_1 g_k^T d_k$ which are needed in the above paper. This means that this paper has the same results under weaker conditions. More dimension functions for practical problems are tested to show the performance of the modified algorithm and the normal algorithm. An application of the fact engineering model is done to show the effectiveness of the given conjugate gradient algorithm.

MSC: 90C26

Keywords: PRP method; Global convergence; WWP line search; Nonconvex functions; Muskingum model

1 Introduction

Consider

$$\min\{f(x) \mid x \in \Re^n\}, \tag{1.1}$$

where $f : \Re^n \to \Re$ and $f \in C^2$. The Polak–Ribière–Polak (PRP) conjugate gradient (CG) method [16, 17] for (1.1) is designed by the following iterative formula:

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, 2, \ldots,$$

where x_k is the kth iterative point, α_k is a stepsize, and d_k is the search direction defined by

$$d_{k+1} = \begin{cases} -g_{k+1} + \beta_k^{\mathrm{PRP}} d_k, & \text{if } k \geq 1 \\ -g_{k+1}, & \text{if } k = 0, \end{cases} \tag{1.2}$$

where $g_{k+1} = \nabla f(x_{k+1})$ is the gradient of $f(x)$ at point x_{k+1}, $\beta_k^{\mathrm{PRP}} \in \Re$ is a scalar defined by

$$\beta_k^{\mathrm{PRP}} = \frac{g_{k+1}^T (g_{k+1} - g_k)}{\|g_k\|^2}, \tag{1.3}$$

where $g_k = \nabla f(x_k)$ and $\|\cdot\|$ denotes the Euclidean norm. The theory analysis and the numerical performance about the PRP method have been done by many scholars (see [2, 3, 17–19, 22] etc.), and many modified algorithms based on the normal PRP formula have been proposed to make a great progress ([6, 8–13, 20, 21, 23–25, 27, 29, 30] etc.). The well-known weak Wolfe–Powell (WWP) inexact line search for α_k satisfies

$$f(x_k + \alpha_k d_k) \leq f_k + \delta\alpha_k g_k^T d_k \tag{1.4}$$

and

$$g(x_k + \alpha_k d_k)^T d_k \geq \sigma g_k^T d_k, \tag{1.5}$$

where $\delta \in (0, 1/2)$ and $\sigma \in (\delta, 1)$. At present, the global convergence of the PRP CG algorithm for nonconvex functions under the WWP line search is a well-known open problem in optimization fields, and the counterexamples of [3, 18] tell us the reason. Motivated by the idea of [3], a modified WWP line search technique is given by Yuan et al. [28] and it is designed by

$$f(x_k + \alpha_k d_k) \leq f_k + \delta\alpha_k g_k^T d_k + \alpha_k \min\left[-\delta_1 g_k^T d_k, \delta\frac{\alpha_k}{2}\|d_k\|^2\right] \tag{1.6}$$

and

$$g(x_k + \alpha_k d_k)^T d_k \geq \sigma g_k^T d_k + \min\left[-\delta_1 g_k^T d_k, \delta\alpha_k\|d_k\|^2\right], \tag{1.7}$$

where $\delta \in (0, 1/2)$, $\delta_1 \in (0, \delta)$, and $\sigma \in (\delta, 1)$. Here we call it YWL line search. It is used for not only the PRP method but also the BFGS quasi-Newton method (see [26, 28] in detail). In the case $\min[-\delta_1 g(x_k)^T d_k, \delta\alpha_k\|d_k\|^2] = \delta\alpha_k\|d_k\|^2$, the global convergence of the PRP algorithm is established including the conditions $d_k^T g_k < 0$ and $g_{k+1} d_k \leq -\sigma_1 g_k^T d_k$. This paper will make a further study and obtain the global convergence similar to [28] without the conditions $d_k^T g_k < 0$ and $g_{k+1} d_k \leq -\sigma_1 g_k^T d_k$ by another proof way. This paper has the following features:

- The PRP algorithm for nonconvex functions with the YWL line search has the global convergence.
- The global convergence is established under weaker conditions than those of the paper [28].
- Larger scale dimension problems are tested to show the performance of the proposed algorithm.

The next section states the algorithm and the global convergence of the presented algorithm. Section 3 does the experiments including the normal unconstrained optimization and an engineering problem. One conclusion is given in the last section.

2 PRP algorithm and global convergence

The PRP algorithm with the modified WWP line search for nonconvex functions is listed as follows.

Algorithm 1 (The PRP CG algorithm under the YWL line search rule)

Step 1: Choose an initial point $x_1 \in \Re^n$, $\varepsilon \in (0,1)\delta \in (0, \frac{1}{2})$, $\delta_1 \in (0, \delta)$, $\sigma \in (\delta, 1)$. Set $d_1 = -g_1 = -\nabla f(x_1)$, $k := 1$.

Step 2: If $\|g_k\| \leq \varepsilon$, stop.

Step 3: Compute the step size α_k using the YWL line search rule (1.6) and (1.7).

Step 4: Let $x_{k+1} = x_k + \alpha_k d_k$.

Step 5: If $\|g_{k+1}\| \leq \varepsilon$, stop.

Step 6: Calculate the search direction

$$d_{k+1} = -g_{k+1} + \beta_k^{\mathrm{PRP}} d_k. \tag{2.1}$$

Step 7: Set $k := k + 1$, and go to Step 3.

The normal assumptions for the nonconvex functions are needed as follows.

Assumption i

(A) The defined level set $L_0 = \{x \mid f(x) \leq f(x_0)\}$ is bounded.

(B) Let $f(x)$ be twice continuously differentiable and bounded below, and the gradient function $g(x)$ is Lipschitz continuous, namely there exists a constant $L > 0$ satisfying

$$\left\|g(x) - g(y)\right\| \leq L\|x - y\|, \quad x, y \in \Re^n. \tag{2.2}$$

Remark

(1) Define a case by *Case i*: $\min[-\delta_1 g(x_k)^T d_k, \delta\alpha_k \|d_k\|^2] = \delta\alpha_k \|d_k\|^2$. This case means that

$$-\delta_1 g(x_k)^T d_k \geq \delta\alpha_k \|d_k\|^2 \geq 0,$$

which can ensure that the modified WWP line search (1.6) and (1.7) is reasonable (see Theorem 2.1 in [28]). Then Algorithm 1 is well defined.

(2) In [28], the global convergence of Algorithm 1 is established for Case i, and it needs not only Assumption i conditions but also

$$d_k^T g_k < 0$$

and

$$g_{k+1} d_k \leq -\sigma_1 g_k^T d_k.$$

In this paper, we will give another proof way only needing Assumption i.

(3) Assumptions i(A) and i(B) imply that there exists a constant $G^* > 0$ such that

$$\left\|g(x)\right\| \leq G^*, \quad x \in L_0. \tag{2.3}$$

Lemma 2.1 *Let Assumption* i *hold. If there exists a positive constant* ϵ_* *such that*

$$\|g_k\| \geq \epsilon_*, \quad \forall k, \tag{2.4}$$

then we can deduce that there exists a constant D^* satisfying

$$\|d_k\| \le \omega^*, \quad \forall k. \tag{2.5}$$

Proof By (1.6), we get

$$\begin{aligned} f(x_k + \alpha_k d_k) &\le f_k + \delta\alpha_k g_k^T d_k + \alpha_k \min\left[-\delta_1 g_k^T d_k, \delta\frac{\alpha_k}{2}\|d_k\|^2\right] \\ &\le f_k + \delta\alpha_k g_k^T d_k - \alpha_k\delta_1 g_k^T d_k \\ &= f_k + (\delta - \delta_1)\alpha_k g_k^T d_k, \end{aligned}$$

then the following inequality

$$-(\delta - \delta_1)\alpha_k g_k^T d_k \le f(x_k) - f(x_{k+1})$$

holds. Using Assumption i(A) and summing these inequalities from $k = 0$ to ∞, we have

$$\delta\sum_{k=0}^{\infty}\left[-(\delta - \delta_1)\alpha_k g_k^T d_k\right] < \infty. \tag{2.6}$$

Using Step 6 of Algorithm 1 and setting $s_k = x_{k+1} - x_k = \alpha_k d_k$, we have

$$\begin{aligned} \|d_{k+1}\| &\le \|g_{k+1}\| + \left|\beta_k^{\mathrm{PRP}}\right|\|d_k\| \\ &\le \|g_{k+1}\| + \frac{\|g_{k+1}\|\|g_{k+1} - g_k\|}{\|g_k\|}\|d_k\| \\ &\le G^* + \frac{G^*L^*}{\|g_k\|}\|s_k\|\|d_k\| \\ &\le G^* + \frac{G^*L^*}{\epsilon_*}\|s_k\|\|d_k\|, \end{aligned} \tag{2.7}$$

where the third inequality follows (2.2) and (2.3), and the last inequality follows (2.4). By the definition of Case i, we get

$$d_k^T g_k \le -\frac{\delta}{\delta_1}\alpha_k\|d_k\|^2.$$

Thus, by (2.6), we get

$$\sum_{k=0}^{\infty}\|s_k\|^2 = \sum_{k=0}^{\infty}\alpha_k\left(\alpha_k\|d_k\|^2\right) \le \frac{\delta_1}{\delta(\delta - \delta_1)}\left[(\delta - \delta_1)\sum_{k=0}^{\infty}\left(-\alpha_k g_k^T d_k\right)\right] < \infty.$$

Then we have

$$\|s_k\| \to 0, \quad k \to \infty.$$

This implies that there exist a constant $\varepsilon \in (0,1)$ and a positive integer $k_0 \geq 0$ satisfying

$$\frac{G^* L^* \|s_k\|}{\epsilon_*} \leq \varepsilon, \quad \forall k \geq k_0. \tag{2.8}$$

So, by (2.7), for all $k > k_0$, we obtain

$$\begin{aligned} \|d_{k+1}\| &\leq G^* + \varepsilon \|d_k\| \\ &\leq G^* \left(1 + \varepsilon + \varepsilon^2 + \cdots + \varepsilon^{k-k_0-1}\right) + \varepsilon^{k-k_0} \|d_{k_0}\| \\ &\leq \frac{G^*}{1-\varepsilon} + \|d_{k_0}\|. \end{aligned}$$

Let $\omega^* = \max\{\|d_1\|, \|d_2\|, \ldots, \|d_{k_0}\|, \frac{G_b}{1-\varepsilon} + \|d_{k_0}\|\}$. Therefore, we get

$$\|d_k\| \leq \omega^*, \quad \forall k \geq 0.$$

The proof is complete. □

Theorem 2.1 *Let the conditions of the above lemma hold. Then the following relation*

$$\lim_{k\to\infty} \inf \|g_k\| = 0 \tag{2.9}$$

holds.

Proof Suppose that (2.9) does not hold, we can deduce that there exists a constant $\epsilon_* > 0$ such that

$$\|g_k\| \geq \epsilon_*, \quad \forall k.$$

Using Lemma 2.1, we get (2.5). By a way similar to (2.6) and using the case $-\delta_1 g(x_k)^T d_k \geq \delta \alpha_k \|d_k\|^2$, we have

$$\begin{aligned} \frac{\delta - \delta_1}{\delta_1} \delta \|\alpha_k d_k\|^2 &\leq -\frac{\delta - \delta_1}{\delta_1} \delta_1 \alpha_k d_k^T g_k \\ &= -(\delta - \delta_1) \alpha_k g_k^T d_k \\ &\to 0, \quad k \to \infty, \end{aligned}$$

which generates

$$\|\alpha_k d_k\|^2 \to 0, \quad k \to \infty. \tag{2.10}$$

Then we discuss the above relation by the following cases.

Case 1: $\|d_k\| \to 0$, $k \to \infty$. By (3.1), (2.3), (2.2), and (2.10), we have

$$\begin{aligned} 0 &\leq \|g_{k+1}\| \\ &= \left\| -d_{k+1} + \beta_k^{\mathrm{PRP}} d_k \right\| \end{aligned}$$

$$\begin{aligned}
&\leq \|d_{k+1}\| + \frac{\|g_{k+1}\|\|g_{k+1} - g_k\|}{\|g_k\|}\|d_k\| \\
&\leq \|d_{k+1}\| + \frac{G^*L\|\alpha_k d_k\|}{\epsilon_*}\|d_k\| \\
&\to 0, \quad k \to \infty.
\end{aligned}$$

Then we get (2.9).

Case 2: $\alpha_k \to 0$, $k \to \infty$. By (1.7), Remark (1), and the Taylor formula, we get

$$\begin{aligned}
g_k^T d_k + O\big(\|\alpha_k d_k\|^2\big) &= g(x_k + \alpha_k d_k)^T d_k \\
&\geq \sigma d_k^T g_k + \min\big[-\delta_1 g_k^T d_k, \delta\alpha_k\|d_k\|^2\big] \\
&\geq \sigma d_k^T g_k.
\end{aligned}$$

Combining with the case $-\delta_1 g(x_k)^T d_k \geq \delta\alpha_k\|d_k\|^2$ leads to

$$\begin{aligned}
O\big(\|\alpha_k d_k\|^2\big) &\geq -(1-\sigma)d_k^T g_k \\
&\geq \frac{\delta(1-\sigma)}{\delta_1}\alpha_k\|d_k\|^2.
\end{aligned}$$

So we have

$$O(\alpha_k) \geq \frac{\delta(1-\sigma)}{\delta_1}.$$

This contracts the case $\alpha_k \to 0$ ($k \to \infty$). Then we also obtain (2.9). All in all, we always have (2.9). The proof is complete. □

3 Numerical results

In this section, we do the numerical experiments of the given algorithm and the normal PRP algorithm for large scale unconstrained optimization problems and these problems are the same of the paper [28] which are from [1, 7] with the given initial points and are listed in Table 1, where the same results are not given anymore. Furthermore we also do an experiment about the fact engineering problem model by the given algorithm. Now we test them and give the results as follows.

3.1 Normal unconstrained optimization problems

To clearly show the normal PRP algorithm, its detailed steps are presented as follows.

Table 1 Tested problems

No.	Problem	No.	Problem	Character
1	Extended Penalty Function	9	EDENSCH Function (CUTE)	These sixteen tested functions come from practical problems such as the academic problems or engineer problems.
2	Extended Cliff Function	10	STAIRCASE S1 Function	
3	Extended Hiebert Function	11	LIARWHD Function (CUTE)	
4	A Quadratic Function QF2 Function	12	DIXON3DQ Function (CUTE)	
5	Extended EP1 Function	13	FLETCHCR Function (CUTE)	
6	Extended Tridiagonal-2 Function	14	COSINE Function (CUTE)	
7	ARWHEAD Function (CUTE)	15	BIGGSB1 Function (CUTE)	
8	EG2 Function (CUTE)	16	Scaled Quadratic SQ1 Function	

Algorithm 2 (The normal PRP CG algorithm)

Step 1: Choose an initial point $x_1 \in \Re^n$, $\varepsilon \in (0,1)\delta \in (0,\frac{1}{2})$, $\sigma \in (\delta,1)$. Set $d_1 = -g_1 = -\nabla f(x_1)$, $k := 1$.

Step 2: If $\|g_k\| \leq \varepsilon$, stop.

Step 3: Compute the step size α_k using the WWP line search rule (1.4) and (1.5).

Step 4: Let $x_{k+1} = x_k + \alpha_k d_k$.

Step 5: If $\|g_{k+1}\| \leq \varepsilon$, stop.

Step 6: Calculate the search direction

$$d_{k+1} = -g_{k+1} + \beta_k^{\mathrm{PRP}} d_k. \tag{3.1}$$

Step 7: Set $k := k + 1$, and go to Step 3.

The following *Himmeblau* stop rule and all parameters are the same to those of the paper [28].

Stop rules: If $|f(x_k)| > e_1$, let $stop1 = \frac{|f(x_k)-f(x_{k+1})|}{|f(x_k)|}$, or $stop1 = |f(x_k) - f(x_{k+1})|$. If the conditions $\|g(x)\| < \epsilon$ or $stop1 < e_2$ hold, the program stops, where $e_1 = e_2 = 10^{-5}$, $\epsilon = 10^{-6}$.

Parameters: $\delta = 0.1$, $\delta_1 = 0.05$, $\sigma = 0.9$.

Dimension: 30,000, 60,000, and 120,000 variables.

Experiments: All the programs were written in MATLAB 7.10 and run on a PC with a 1.80 GHz CPU and 4.00 GB of memory running the Windows 7 operating system.

Other cases: The program is also stopped if the number of iterations is greater than 1200. The step size α_k in the line search is accepted if the search number is greater than 10.

The columns of Table 2 have the following meanings:

No.: the number of tested problems. Dim: the dimension of tested problem.

Cputime: the CPU time in seconds. NI: the iteration number.

NFG: the total number both of the gradient value and the function value.

Numerical results of Table 2 show that both of these two algorithms have a good efficiency for these practical problems. The iteration number, the number of the function value and the gradient value, and the CPU time will increase with the dimension becoming large for most of the problems. However, the CPU time does not become bigger but smaller, such as problems 4, 13, and 14 for Algorithm 2 and problems 1 and 16 for Algorithm 1; the reason may lie in the system of computer. The numerical results indicate that Algorithm 1 is competitive to Algorithm 2 especially for the CPU time for most of the tested problems. To directly show the performance of these two algorithms, the tool of Dolan and Moré [4] is used, and Figs. 1–3 show the profiles of them relative to NI, NFG, and Cputime, respectively. These three figures have the similar trend, then we only analyze Fig. 3 about the CPU time. Figure 3 shows that Algorithm 1 is better than Algorithm 2, Algorithm 1 goes beyond Algorithm 2 about 11%, and Algorithm 1 has perfect robustness comparing with Algorithm 2. In a word, Algorithm 1 provides noticeable advantages.

3.2 Fact engineering problem of the Muskingum model

The subsection studies an application of the presented algorithm for a fact engineering problem, namely the well-known hydrologic engineering application problem often called

Table 2 The numerical results of Algorithm 1 and Algorithm 2

Nr.	Dim	Algorithm 1			Algorithm 2		
		NI	NFG	Cputime	NI	NFG	Cputime
1	30,000	2	5	0.0624	168	527	3.681624
	60,000	2	5	0.124801	173	544	7.86245
	120,000	2	5	0.109201	180	567	16.083703
2	30,000	21	161	1.981213	97	310	4.851631
	60,000	21	161	3.962425	100	319	9.703262
	120,000	21	161	7.456848	104	331	18.579719
3	30,000	2	5	0.0001	4	24	0.078
	60,000	2	5	0.0624	4	24	0.249602
	120,000	2	5	0.124801	4	24	0.374402
4	30,000	2	14	0.0624	2	6	0.0624
	60,000	2	14	0.124801	2	6	0.0001
	120,000	2	14	0.187201	2	6	0.0624
5	30,000	2	15	0.124801	12	35	0.436803
	60,000	2	15	0.280802	15	44	1.060807
	120,000	2	15	0.390002	20	59	2.730017
6	30,000	3	27	0.187201	9	28	0.249602
	60,000	43	128	2.745618	4	16	0.249602
	120,000	56	167	6.910844	4	16	0.390003
7	30,000	3	17	0.124801	34	101	0.826805
	60,000	3	17	0.156001	34	101	1.59121
	120,000	3	17	0.436803	34	101	3.229221
8	30,000	4	20	0.234002	6	50	0.624004
	60,000	4	20	0.624004	6	51	1.185608
	120,000	4	20	1.029607	6	50	2.012413
9	30,000	1097	3290	184.237181	28	85	4.555229
	60,000	1439	4316	387.475284	28	85	7.316447
	120,000	1500	4499	559.403986	28	85	10.218066
10	30,000	2	5	0.0624	1500	4504	27.690178
	60,000	2	5	0.124801	1500	4504	55.567556
	120,000	2	5	0.124801	1500	4504	112.757523
11	30,000	96	287	3.260421	3	19	0.124801
	60,000	86	257	5.896838	3	19	0.234002
	120,000	86	257	11.668875	6	25	0.811205
12	30,000	11	122	0.530403	1500	4505	25.272162
	60,000	11	122	0.873606	1500	4505	49.748719
	120,000	11	122	1.747211	1500	4505	101.587851
13	30,000	2	14	0.327602	4	21	0.561604
	60,000	2	14	0.592804	3	8	0.390003
	120,000	2	14	1.029607	3	8	0.608404
14	30,000	108	323	6.879644	19	164	6.146439
	60,000	149	446	18.454918	10	75	1.965613
	120,000	206	617	41.153064	12	93	4.258827
15	30,000	11	122	0.452403	1500	4503	24.460957
	60,000	11	122	0.936006	1500	4503	51.667531
	120,000	11	122	1.778411	1500	4503	99.232236
16	30,000	2	5	0.0624	1500	4517	63.88241
	60,000	2	5	0.0156	1500	4517	123.443591
	120,000	2	5	0.0624	1500	4517	212.16136

Figure 1 Performance profiles of Algorithm 1 and Algorithm 2 (NI)

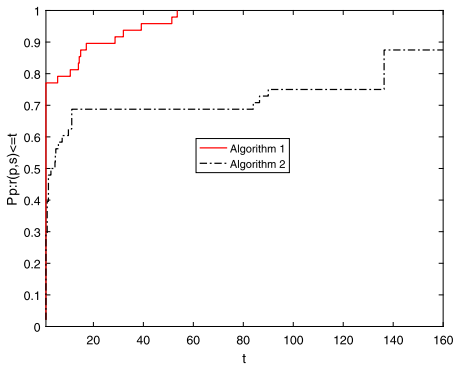


Figure 2 Performance profiles of Algorithm 1 and Algorithm 2 (NFG)

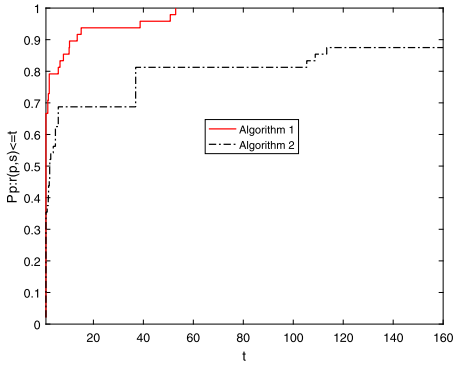
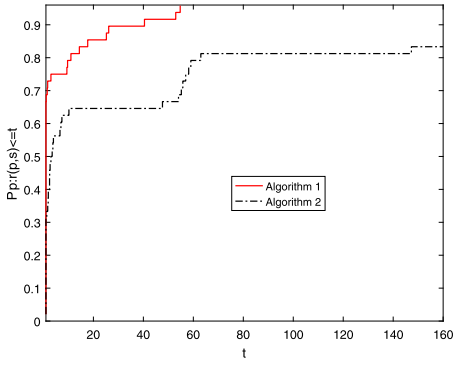


Figure 3 Performance profiles of Algorithm 1 and Algorithm 2 (Cputimes)



the parameter estimation problem of the nonlinear Muskingum model. The Muskingum model has the following definition.

Muskingum model [14]: The parameter estimation of the model is designed by

$$\begin{aligned}\min f(x_1,x_2,x_3) = \sum_{i=1}^{n-1}\Bigg(&\left(1-\frac{\Delta t}{6}\right)x_1\left(x_2I_{i+1}+(1-x_2)Q_{i+1}\right)^{x_3}\\&-\left(1-\frac{\Delta t}{6}\right)x_1\left(x_2I_i+(1-x_2)Q_i\right)^{x_3}-\frac{\Delta t}{2}(I_i-Q_i)\\&+\frac{\Delta t}{2}\left(1-\frac{\Delta t}{3}\right)(I_{i+1}-Q_{i+1})\Bigg)^2,\end{aligned}$$

Table 3 Results of these algorithms

Algorithms	x_1	x_2	x_3
BFGS [5]	10.8156	0.9826	1.0219
HIWO [14]	13.2813	0.8001	0.9933
Algorithm 1	11.1884	1.0034	0.9993

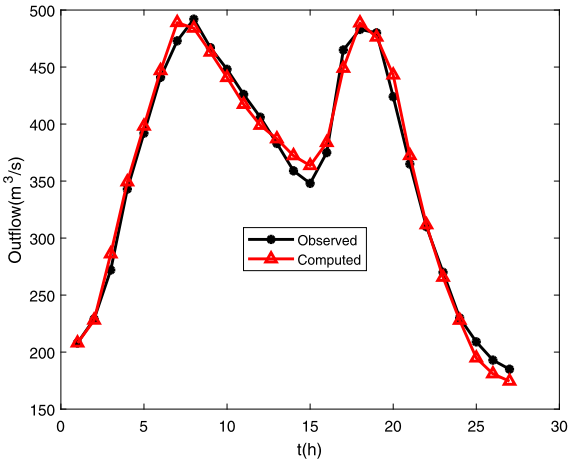


Figure 4 Performance of data in 1960

where x_1 is the storage time constant, x_2 is the weighting factor, and x_3 is an additional parameter; at time t_i ($i = 1, 2, \ldots, n$), n denotes the total time number, Δt is the time step, I_i and Q_i are the observed inflow discharge and observed outflow discharge, respectively. The Muskingum model, as a hydrologic routing method, is a popular model for flood routing, whose storage depends on the water inflow and outflow. This subsection uses actual observed data of the flood run-off process between Chenggouwan and Linqing of Nanyunhe in the 8 Haihe Basin, Tianjin, China, where $\Delta t = 12(h)$. The detailed I_i and Q_i of the data of 1960, 1961, and 1964 can be found in [15]. In the numerical experiments, we set the initial point $x = [0, 1, 1]^T$. The tested results are listed in Table 3.

Figures 4–6 are the data curves of 1960, 1961, and 1964 about the observed flows and computed flows by Algorithm 1 for estimating the parameters of the nonlinear Muskingum model, which shows that the given algorithm has good approximation for these data and Algorithm 1 is effective for the nonlinear Muskingum model. The results of Table 3 and Figs. 4–6 tell us at least two conclusions: (1) Algorithm 1 can be successfully used for solving the nonlinear Muskingum model because of its good approximation; (2) the points x_1, x_2, and x_3 obtained by Algorithm 1 are different from the BFGS method and the HIWO method, which shows that the Muskingum model may be have several optimum approximated points.

4 Conclusion

This paper studies the proof method and proposes a simple proof technique to get the global convergence of the known algorithm in the paper [28]. The following conclusions are obtained by this paper:

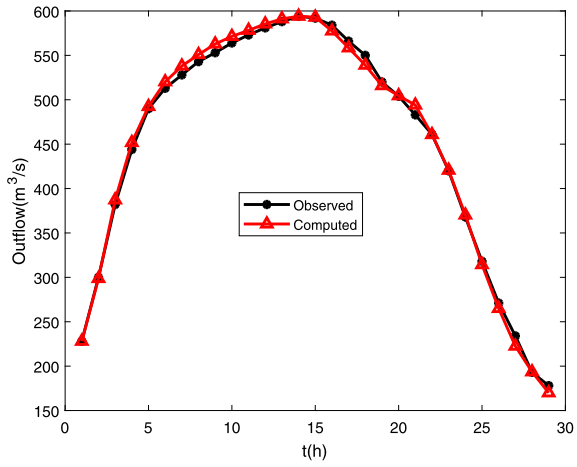


Figure 5 Performance of data in 1961

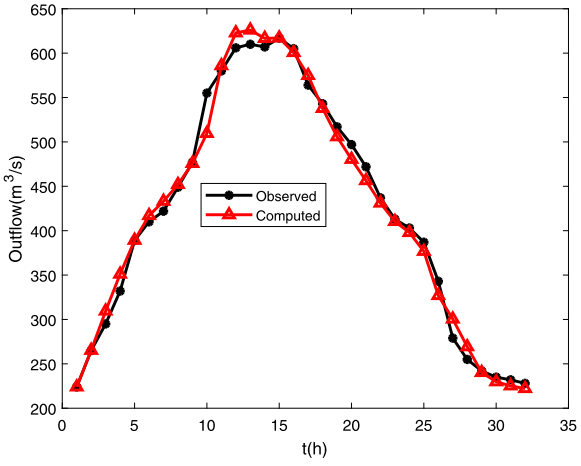


Figure 6 Performance of data in 1964

(1) This paper gives a new proof method for the paper [28] and gets the same result under weaker conditions. This new proof technique is more simple than those of the paper [28].
(2) More larger-scale dimension problems are done comparing with [28] to show that the given algorithm is competitive to the normal algorithm. The nonlinear Muskingum model coming from the fact engineering problem is done by the given algorithm to estimate its parameters, which demonstrates that Algorithm 1 is very successful.
(3) One interesting question and work is whether there exist some other proof methods to get the global convergence of Algorithm 1, which is one of the works of ours in the future.

Acknowledgements
The authors would like to express the sincerest appreciation to editors and the anonymous referees for their valuable comments that helped improve the manuscript.

Funding
This work is supported by the National Natural Science Foundation of China (Grant No. 11661009), the Guangxi Science Fund for Distinguished Young Scholars (No. 2015GXNSFGA139001), and the Guangxi Natural Science Key Fund (No. 2017GXNSFDA198046).

Competing interests
There is no potential conflicts of interest.

Authors' contributions
XL organized this whole paper, TY wrote some parts of the proof, and XW did the experiments and wrote some parts of the proof. All authors read and approved the final manuscript.

Author details
[1]College of Mathematics and Information Science, Guangxi University, Nanning, P.R. China. [2]School of Finance and Economics, Nanning College For Vocational Technology, Nanning, P.R. China. [3]School of Mathematical Sciences, Dalian University of Technology, Dalian, P.R. China.

Publisher's Note

Received: 11 March 2019 Accepted: 1 July 2019 Published online: 15 July 2019

References

1. Bongartz, I., Conn, A.R., Gould, N.I., Toint, P.L.: CUTE: constrained and unconstrained testing environment. ACM Trans. Math. Softw. **21**, 123–160 (1995)
2. Dai, Y.: Analysis of conjugate gradient methods. Ph.D. thesis, Institute of Computational Mathematics and Scientific/Engineering Computing, Chese Academy of Sciences (1997)
3. Dai, Y.: Convergence properties of the BFGS algorithm. SIAM J. Optim. **13**, 693–701 (2003)
4. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)
5. Geem, Z.W.: Parameter estimation for the nonlinear Muskingum model using the BFGS technique. J. Hydrol. Eng. **132**, 474–478 (2006)
6. Gilbert, J.C., Nocedal, J.: Global convergence properties of conjugate gradient methods for optimization. SIAM J. Optim. **2**, 21–42 (1992)
7. Gould, N.I., Orban, D., Toint, P.L.: CUTEr and SifDec: a constrained and unconstrained testing environment, revised. ACM Trans. Math. Softw. **29**, 373–394 (2003)
8. Hager, W., Zhang, H.: A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM J. Optim. **16**, 170–192 (2005)
9. Hager, W., Zhang, H.: Algorithm 851: *CG_DESCENT*, a conjugate gradient method with guaranteed descent. ACM Trans. Math. Softw. **32**, 113–137 (2006)
10. Mingqi, X., Rădulescu, V.D., Zhang, B.: Combined effects for fractional Schrödinger–Kirchhoff systems with critical nonlinearities. ESAIM Control Optim. Calc. Var. **24**, 1249–1273 (2018)
11. Mingqi, X., Rădulescu, V.D., Zhang, B.: Nonlocal Kirchhoff diffusion problems: local existence and blow-up of solutions. Nonlinearity **31**, 3228–3250 (2018)
12. Mingqi, X., Rădulescu, V.D., Zhang, B.: Fractional Kirchhoff problems with critical Trudinger Moser nonlinearity. Calc. Var. Partial Differ. Equ. **58**, 57 (2019). https://doi.org/10.1007/s00526-019-1499-y
13. Mingqi, X., Rădulescu, V.D., Zhang, B.: A critical fractional Choquard–Kirchhoff problem with magnetic field. Commun. Contemp. Math. **21**(4), 185004 (2019)
14. Ouyang, A., Liu, L., Sheng, Z., Wu, F.: A class of parameter estimation methods for nonlinear Muskingum model using hybrid invasive weed optimization algorithm. Math. Probl. Eng. **2015**, Article ID 573894 (2015)
15. Ouyang, A., Tang, Z., Li, K., Sallam, A., Sha, E.: Estimating parameters of Muskingum model using an adaptive hybrid PSO algorithm. Int. J. Pattern Recognit. Artif. Intell. **28**, Article ID 1459003 (2014)
16. Polak, E.: The conjugate gradient method in extreme problems. Comput. Math. Math. Phys. **9**, 94–112 (1969)
17. Polak, E., Ribière, G.: Note sur la convergence de directions conjugees. Rev. Fr. Inform. Rech. Oper. **3**, 35–43 (1969)
18. Powell, M.J.D.: Nonconvex minimization calculations and the conjugate gradient method. In: Numerical Analysis. Lecture Notes in Mathematics, vol. 1066, pp. 122–141. Spinger, Berlin (1984)
19. Powell, M.J.D.: Convergence properties of algorithm for nonlinear optimization. SIAM Rev. **28**, 487–500 (1986)
20. Sun, Y.: Indirect boundary integral equation method for the Cauchy problem of the Laplace equation. J. Sci. Comput. **71**, 469–498 (2017)
21. Wei, Z., Yao, S., Liu, L.: The convergence properties of some new conjugate gradient methods. Appl. Math. Comput. **183**, 1341–1350 (2006)
22. Yuan, G.: Modified nonlinear conjugate gradient methods with sufficient descent property for large-scale optimization problems. Optim. Lett. **3**, 11–21 (2009)
23. Yuan, G., Lu, X.: A modified PRP conjugate gradient method. Ann. Oper. Res. **166**, 73–90 (2009)
24. Yuan, G., Lu, X., Wei, Z.: A conjugate gradient method with descent direction for unconstrained optimization. J. Comput. Appl. Math. **233**, 519–530 (2009)
25. Yuan, G., Meng, Z., Li, Y.: A modified Hestenes and Stiefel conjugate gradient algorithm for large-scale nonsmooth minimizations and nonlinear equations. J. Optim. Theory Appl. **168**, 129–152 (2016)
26. Yuan, G., Sheng, Z., Wang, B., Hu, W., Li, C.: The global convergence of a modified BFGS method for nonconvex functions. J. Comput. Appl. Math. **327**, 274–294 (2018)

27. Yuan, G., Wei, Z., Li, G.: A modified Polak–Ribière–Polyak conjugate gradient algorithm for nonsmooth convex programs. J. Comput. Appl. Math. **255**, 86–96 (2014)
28. Yuan, G., Wei, Z., Lu, X.: Global convergence of the BFGS method and the PRP method for general functions under a modified weak Wolfe–Powell line search. Appl. Math. Model. **47**, 811–825 (2017)
29. Yuan, G., Wei, Z., Yang, Y.: The global convergence of the Polak–Ribière–Polyak conjugate gradient algorithm under inexact line search for nonconvex functions. J. Comput. Appl. Math. **362** 262–275 (2019). https://doi.org/10.1016/j.cam.2018.10.057
30. Yuan, G., Zhang, M.: A three-terms Polak–Ribière–Polyak conjugate gradient algorithm for large-scale nonlinear equations. J. Comput. Appl. Math. **286**, 186–195 (2015)