STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. , , pp 0–9.
Published online in International Academic Press (www.IAPress.org)

A New Hybrid Conjugate Gradient Method Combining BA and PRP Variants: Global Convergence Analysis and Numerical Performance

Nassima Rezgui*, Nabil Sellami, Romaissa Mellal

Department of Mathematics, 8th May 1945 University, Guelma, Algeria

Abstract This paper introduces a new hybrid nonlinear conjugate gradient algorithm for solving unconstrained optimization problems, based on a convex combination of the AlBayati–AlAssady and Polak-Ribiere-Polyak methods, in which the convex parameter independently satisfies the conjugacy condition. Through rigorous theoretical analysis, this new algorithm guarantees sufficient descent and global convergence properties under the strong Wolfe line search criteria. We tested our method on 21 benchmark unconstrained problems and used Dolan–Moré performance profiles to compare it with classical methods. The results show that the proposed NNR algorithm consistently outperforms the standard BA, PRP, and DY conjugate gradient methods.

Keywords Hybrid Nonlinear Conjugate Gradient, Strong Wolfe Line Search, Python.

AMS 2010 subject classifications 90C26, 90C30, 65K05.

DOI: 10.19139/soic-2310-5070-3898

1. Introduction

Nonlinear conjugate gradient methods (CGMs) represent a widely used class of optimization problems aimed at minimizing a differentiable function with respect to unconstrained variables. This problem is commonly written in the form:

$$\min_{x\in\mathbb{R}^n} f(x). \tag{1}$$

in which $f:\mathbb{R}^n \mapsto \mathbb{R}$ belongs to class $\mathcal{C}^1$.
The CGM generates an iterative sequence of the form:

$$x_{i+1} = x_i + \alpha_i d_i, \tag{2}$$

where x_i is the current iterate, α_i is the steplength determined through a line search procedure along the search direction defined as:

$$d_i = \begin{cases} -q_i, \text{ for } i = 1, \\ -q_i + \beta_i d_{i-1}, \text{ for } i \geq 2, \end{cases}.$$

where q_i is the gradient $\nabla f(x_i)$, and the scalar β_i is selected to ensure that d_i constitutes the i^{th} conjugate direction, provided that the objective function is quadratic and the line search is performed exactly [4].
Nonlinear conjugate gradient methods mainly differ in the specific choice of the parameter β_i.
The CGM was first introduced by Hestenes and Stiefel (HS) [12] for solving strictly convex quadratic linear systems. It was subsequently generalized by Fletcher and Reeves (FR) [10] to solve nonlinear unconstrained

*Correspondence to: Nassima Rezgui (Email: nassima.rezgui@yahoo.fr).

ISSN 2310-5070 (online) ISSN 2311-004X (print)

optimization problems. Since then, many nonlinear CG schemes have appeared in the literature, such as the Polak-Ribiere-Polyak (PRP) method [16, 17] and the Al-Bayati–Al-Assady (BA) method [1]. These algorithms are regarded as classical CGMs and reduce to the linear CG algorithm when f is a strictly convex quadratic function and the stepsize is chosen as the exact minimizer along the search direction.

Table 1. List of some classical conjugate gradient variants

No.	**Formula**	**Authors**
1	$\beta_i^{HS} = \dfrac{q_{i+1}^T y_i}{s_i^T y_i}$	Hestenes-Stiefel
2	$\beta_i^{FR} = \dfrac{\|q_{i+1}\|^2}{\|q_i\|^2}$	Fletcher–Reeves
3	$\beta_i^{PRP} = \dfrac{q_{i+1}^T y_i}{\|q_i\|^2}$	Polak-Ribière
4	$\beta_i^{BA} = \frac{\|y_i\|^2}{d_i^T y_i}$	AlBayati–AlAssady

where $y_i = q_{i+1} - q_i$.
In what follows, we focus on a major class of hybrid CG algorithms, namely those obtained by forming convex combinations of two term CGMs.
The first such formulation, proposed by Andrei [2], employs the convex structure $\beta_i^{\text{hybrid}} = (1-\eta_i)\,\beta_i^{\text{method1}} + \eta_i\,\beta_i^{\text{method2}}$, where $\eta_i \in [0,1]$ and in 2008, Andrei [2] introduced a specific combinations (DY + HS), defined by $\beta_i^{\text{DY-HS}} = \eta_i\beta_i^{\text{DY}} + (1-\eta_i)\beta_i^{\text{HS}}$. and in 2009, he proposed another hybrid conjugate gratient based on the above methods using conjugacy condition[3].
In addition, Djordjevic et al. proposed two additional hybrids: one combining (LS + CD) [7], and (HS + FR) [8]. Furthermore, Mellal et al. developed two hybrid conjugate gradient algorithms. The first one combines the WYL and BA methods [14] and the second one combines the RMIL and BA methods [15], where the convex parameter η_i is selected so as to align the resulting CG direction with the Newton direction. The expressions of these hybrid conjugate gradient variants are summarized in Table 2

Table 2. List of some hybrid two-term conjugate gradient variants

No.	**Formula**	**Authors**
1	$\beta^{\text{DY-PRP}} = \eta\,\beta^{\text{DY}} + (1-\eta)\,\beta^{\text{PRP}}$	Andrei
2	$\beta^{\text{DY-HS}} = \eta\,\beta^{\text{DY}} + (1-\eta)\,\beta^{\text{HS}}$	Andrei
3	$\beta^{\text{LS-CD}} = \eta\,\beta^{\text{LS}} + (1-\eta)\,\beta^{\text{CD}}$	Djordjevic
4	$\beta^{\text{HS-FR}} = \eta\,\beta^{\text{FR}} + (1-\eta)\,\beta^{\text{HS}}$	Djordjevic
5	$\beta^{\text{RN}} = \eta\,\beta^{\text{BA}} + (1-\eta)\,\beta^{\text{WYL}}$	Mellal
6	$\beta^{\text{hnBARMIL}} = \eta\,\beta^{\text{BA}} + (1-\eta)\,\beta^{\text{RMIL}}$	Mellal

In this work, we introduce a new two term hybrid nonlinear CGM based on a convex combination of BA and PRP variants, denoted by NNR, in which the parameter β_i is selected by a combination of two classical coefficients (BA and PRP). The resulting method satisfies the sufficient descent condition and is proven to be globally convergent under the strong Wolfe line search conditions. The NNR algorithm is implemented in Python and using Dolan–Moré performance profiles we compare it to some classical and hybrid CGMs on a set of well khnown function test.

2. New hybrid Formula

In this part, we propose a new hybrid CG variant. This new convex combination chooses β_i such that

$$\beta_i^{NNR} = \eta_i \beta_i^{BA} + (1-\eta_i)\beta_i^{PRP}, \tag{3}$$

where $\eta_i \in [0,1]$ is a scalar weighting parameter. The recurrence relation for search direction is defined by :

$$d_{i+1}^{NNR} = \begin{cases} -q_{i+1}, & \text{for } i = 0, \\ -q_{i+1} + \beta_i^{NNR} d_i, & \text{for } i \geq 1, \end{cases} \tag{4}$$

where $q_i = \nabla f(x_i)$ denotes the gradient. The step size α_i in the iteration $x_{i+1} = x_i + \alpha_i d_i$ is determined using the strong Wolfe conditions:

$$\begin{aligned} f(x_i + \alpha_i d_i) &\leq f(x_i) + \rho \alpha_i q_i^T d_i, \\ |q_{i+1}^T d_i| &\leq \sigma |q_i^T d_i|, \end{aligned} \tag{5}$$

with constants $0 < \rho < \sigma < 1$ [19, 20].
If $\eta_i = 0$ then $\beta_i^{NNR} = \beta_i^{PRP}$ and if $\eta_i = 1$ then $\beta_i^{NNR} = \beta_i^{BA}$.
In this variant, η_i is selected to ensure that the conjugacy condition is satisfied at each iteration. From the formula (4), we have:

$$\begin{aligned} d_{i+1}^{NNR} = -q_{i+1} + \beta_i^{NNR} d_i &= -q_{i+1} + (1-\eta_i)\beta_i^{PRP} d_i + \eta_i \beta_i^{BA} d_i \\ &= -q_{i+1} + (1-\eta_i)\frac{q_{i+1}^T y_i}{\|q_i\|^2} + \eta_i \frac{||y_i||^2}{d_i^T y_i} d_i \end{aligned}$$

Multiplying both sides by y_i and using $d_{i+1}^T y_i = 0$:

$$0 = -q_{i+1}^T y_i + (1-\eta_i)\frac{q_{i+1}^T y_i}{\|q_i\|^2} d_i^T y_i + \eta_i \frac{||y_i||^2}{d_i^T y_i} d_i^T y_i$$

which simplifies to:

$$\eta_i = \frac{q_{i+1}^T y_i - \dfrac{q_{i+1}^T y_i}{\|q_i\|^2}\, d_i^T y_i}{\|y_i\|^2 - \dfrac{q_{i+1}^T y_i}{\|q_i\|^2}\, d_i^T y_i}.$$

Finally, we have

$$\eta_i^{nnr} = \eta_i = \frac{(q_{i+1}^{\mathsf{T}} y_i)\big(\|q_i\|^2 - d_i^{\mathsf{T}} y_i\big)}{\|y_i\|^2 \|q_i\|^2 - (q_{i+1}^{\mathsf{T}} y_i)(d_i^{\mathsf{T}} y_i)}. \tag{6}$$

To ensure numerical stability, we regularize η_i^{NNR} as:

$$\eta_i^{NNR} = \begin{cases} \eta_i^{nnr} & \text{if } 0 < \eta_i^{nnr} < 1, \\ 0, & \text{if } \eta_i^{nnr} \leq 0, \\ 1, & \text{if } \eta_i^{nnr} \geq 1. \end{cases} \tag{7}$$

The main steps of the proposed NNR conjugate gradient method are presented in the following algorithm.

Algorithm 1: NNR Conjugate Gradient Algorithm

Input: $x_0 \in \mathbb{R}^n$, parameters $\delta = 0.05$, $\sigma = 0.5$, maximum number of iterations $N_{\max}$.
Output: Approximate solution x^* of problem (1).
Compute $f(x_0)$ and $q_0 = \nabla f(x_0)$
Set $d_0 = -q_0$
Set $i \leftarrow 0$
while $\|q_i\|_\infty > 10^{-6}$ ***and*** $i < N_{\max}$ **do**
  `// Inexact line search (strong Wolfe conditions)`
  Compute step size α_i satisfying (5)
  Update $x_{i+1} = x_i + \alpha_i d_i$
  Compute $f(x_{i+1})$ and $q_{i+1} = \nabla f(x_{i+1})$
  Compute $\eta_i = \eta_i^{\text{NNR}}$ from (7)
  Compute $\beta_i = \beta_i^{\text{NNR}}$ from (3)
  if $|q_{i+1}^T q_i| \geq 0.2\|q_{i+1}\|^2$ **then**
    Set $d_{i+1} = -q_{i+1}$ and $\alpha_{i+1} = 1$
  else
    Set $d_{i+1} = -q_{i+1} + \beta_i d_i$
  $i \leftarrow i + 1$
Set $x^* = x_i$

3. Convergence analysis

In this section, we will present the theorems which guarantee the convergence of our method under the strong Wolfe line search.

Assumption 1

- The set $\Gamma = \{x \in \mathbb{R}^n : f(x) \leq f(x_1)\}$ is bounded, thus,

$$\exists\, B > 0 : \|x\| \leq B, \qquad \forall x \in \Gamma \tag{8}$$

- Within a neighborhood N of Γ, the function f belongs to class $\mathcal{C}^1$ with a Lipschitz continuous gradient.

$$\exists L > 0 : ||q(x) - q(y)|| \leq L||x - y||, \qquad \forall x, y \in \Gamma.$$

Under Assumptions 1 we have :

$$\exists\, M > 0 : ||q(x)|| \leq M, \qquad \forall x, y \in \Gamma \tag{9}$$

Theorem 1
Assuming Assumption 1 holds, the search direction generated by the NNR CGM satisfies :

$$q_i^T d_i^{NNR} \leq -c||q_i||^2, \;\; \forall i \;\geq 0. \tag{10}$$

Proof
If $i = 0$, then $q_0^T d_0 = -\|q_0\|^2$, so (10) holds.
In other hand, for $k > 0$, we have :

$$\begin{aligned} d_{i+1}^{NNR} &= -q_{i+1} + \beta_i^{NNR} d_i \\ &= -q_{i+1} + \left(\eta_i \beta_i^{BA} + (1-\eta_i)\beta_i^{PRP}\right) d_i \\ &= (1-\eta_i)(-q_{i+1} + \beta_i^{PRP} d_i) + \eta_i(-q_{i+1} + \beta_i^{BA} d_i) \end{aligned}$$

It follows that:

$$d_{i+1}^{NNR} = \eta_i d_i^{BA} + (1-\eta_i) d_i^{PRP}$$

Multiplying by q_{i+1}^T from the left side, we get

$$q_{i+1}^T d_{i+1}^{NNR} = \eta_i q_{i+1}^T d_{i+1}^{BA} + (1-\eta_i)\, q_{i+1}^T d_{i+1}^{PRP} \tag{11}$$

Firstly, if $\eta_i = 0$, d_i^{NNR}, we obtain :

$$d_{i+1}^{NNR} = d_{i+1}^{PRP} = -q_{i+1} + \beta_i^{PRP} d_i$$

They prouved in [16, 17]

$$q_{i+1}^T d_{i+1}^{NNR} = q_{i+1}^T d_{i+1}^{PRP} \leq -c_1 \left\| q_{i+1} \right\|^2, \; \forall\, i \tag{12}$$

where $c_1 = 1 - \sigma$.
Now, if $\eta_i = 1$, d_i^{NNR}, we obtain

$$d_{i+1}^{NNR} = d_{i+1}^{BA} = -q_{i+1} + \beta_i^{BA} d_i$$

Multiplying by q_{i+1}, we get

$$\begin{aligned} q_{i+1}^T d_{i+1}^{NNR} = q_{i+1}^T d_{i+1}^{BA} &= -q_{i+1}^T q_{i+1} + \beta_i^{BA} q_{i+1}^T d_i \\ &= -\left\| q_{i+1} \right\|^2 + \beta_i^{BA} q_{i+1}^T d_i, \; \forall\, i \end{aligned}$$

Hamdi et al. proved in [11] that

$$q_{i+1}^T d_{i+1}^{BA} \leq -c_2 \left\| q_{i+1} \right\|^2, \; \forall\, i \tag{13}$$

where $c_2 > 0$.
Finally, if $0 < \eta_i < 1$, $\exists\, \lambda_1, \lambda_{2 \in R}$:

$$0 < \lambda_1 \leq \eta_i \leq \lambda_2 < 1 \tag{14}$$

From the formula (11) and using (12) and (13), we conclude

$$q_{i+1}^T d_{i+1}^{NNR} \leq \lambda_1 q_{i+1}^T d_{i+1}^{BA} + (1-\lambda_2)\, q_{i+1}^T d_{i+1}^{PRP}$$

Let $c = \lambda_1 c_2 + (1-\lambda_2)\, c_1$, then from (11) and (14) we finally get

$$q_{i+1}^T d_{i+1}^{NNR} \leq -c \left\| q_{i+1} \right\|^2$$

□

Lemma 1
Suppose Assumption 1 is satisfied. Let any CGM be defined via (2) and (1), employing the step length α_i obtained by the SWL (5).
If $\sum_{i \geq 0} \frac{1}{\|d_i\|^2} < \infty$, then $\lim_{i\to\infty} \inf \|q_i\| = 0$.

Proof
[4] □

Lemma 2
Suppose that Assumption 1 is satisfied, if d_i is a descent direction and α_i satisfies

$$q_{i+1}^T d_i \geq \sigma q_i^T d_i, \text{: where : } 0 < \sigma < 1,$$

then

$$\alpha_i \geq \frac{(1-\sigma) q_i^T d_i}{L |d_i|^2}. \tag{15}$$

Proof
[13] □

Theorem 2
Consider the NNR CGM and suppose that assumption 1 holds. Then either $q_i = 0$, for some i, or

$$\lim_{i\to\infty} \inf \|q_i\| = 0. \tag{16}$$

Proof
Consider the NNR CGM and assume that Assumption 1 hold. Suppose that $q_i \neq 0$, for all i. We prove (16) by contradiction.
Assume that (16) does not hold, Then, $\exists t > 0$:

$$\|q_i\| \geq t,\ \forall\, i, \tag{17}$$

according to the relation (5) and (10), we obtain:

$$q_i^T d_i > c(1-\sigma)\|q_i\|^2 > c(1-\sigma)t^2. \tag{18}$$

By using the Lipschitz condition, we get:

$$\|y_i\| = \|q_{i+1} - q_i\| \leq LB, \tag{19}$$

where $B = \max\{\|x-y\|, x, y \in \Gamma\}$ is the diameter of Γ. We have

$$d_{i+1}^{NNR} = -q_{i+1} + (1-\eta_i)\beta_i^{PRP} d_i + \eta_i \beta_i^{BA} d_i,$$

since, $0 < \eta_i < 1$, we obtain:

$$\|d_{i+1}^{NNR}\| \leq \|q_{i+1}\| + (|\beta_i^{BA}| + |\beta_i^{PRP}|)\|d_i\|.$$

We have :

$$\left|\beta_i^{BA}\right| = \frac{||y_i||^2}{d_i^T y_i} \leq \frac{L^2\|s\|^2}{(1-\sigma)ct^2} \leq \frac{L^2B^2}{(1-\sigma)ct^2} \tag{20}$$

Otherwise

$$\left|\beta_i^{PRP}\right| = \frac{|q_{i+1}^T y_i|}{\|q_i\|^2} \leq \frac{MLB}{t^2} \tag{21}$$

Using the above relations (20) and (21), we obtain:

$$\begin{aligned}
\|d_{i+1}^{NNR}\| \leq \|q_{i+1}\| + (|\beta_i^{BA}| + |\beta_i^{PRP}|)\|d_i\| &\leq M + (\frac{L^2B^2}{(1-\sigma)ct^2} + \frac{MLB}{t^2})\frac{\|s_i\|}{\lambda} \\
&\leq M + (\frac{L^2B^2}{(1-\sigma)ct^2} + \frac{MLB}{t^2})\frac{B}{\lambda} \\
&\leq \kappa,\ \ \forall\, i
\end{aligned}$$

where

$$\kappa = \frac{M\lambda t^2(1-\sigma)c + L^2B^3 + MLB^2(1-\sigma)c}{(1-\sigma)c\,\lambda t^2}$$

Thus, we conclude:

$$\sum_{i\geq 0} \frac{1}{\|d_{i+1}^{NNR}\|^2} = +\infty. \tag{22}$$

By applying Lemma 1, we conclude that:

$$\lim_{i\to\infty} \inf \|q_i\| = 0. \tag{23}$$

This contradicts (17) , and thus we have proved (16) . □

4. Analysis of Computational Performance

This section presents numerical experiments evaluating the proposed NNR CGM for problem (1).
These hybrid method, based on a convex combination of BA and PRP parameters with restart criteria, were tested on 21 standard benchmarks from [1, 5] (Table 3), (n ranging from n=2 to n=5000.) Under strong Wolfe line search conditions (5), our NNR algorithm outperforms nonlinear conjugate gradient methods (BA, DY and PRP) [1, 6, 17, 16] in iterations (NOI), CPU time, and function evaluations (NFE), as confirmed by Moré-Dolan profiles.

Table 3. Test Problems and Initial Points Used in Numerical Experiments

No.	Test Functions	Dim	Initial Points
1	Rosenbrock	2, 10, 100, 500, 1000, 1500, 5000	$(-1.2, -1.2)$ for $n = 2$, (1.2,...,1.2) for $n > 2$
2	Sphere	2	(0.5, 0.5)
3	Sum of Squares	2, 10, 100, 500, 1000, 1500	(1, 1,...,1)
4	Zakharov	2, 10, 100, 500, 1000, 1500, 5000	(1, 1,...,1)
5	Dixon-Price	2, 10	(2, 2,...,2)
6	Quadratic Function QF1	2, 10, 100, 500, 1000, 1500	(1, 1,...,1)
7	Raydan 1	2, 10, 100, 500, 1000, 1500	(0.5, 0.5,...,0.5)
8	Extended Rosenbrock	2, 10, 100, 500, 1000, 1500	(-1.2, -1.2,...,-1.2)
9	Extended DENSCHNF	2, 10, 100, 500, 1000, 1500	(1, 1,...,1)
10	Extended Tridiagonal	10	(0, 0,...,0)
11	Extended Himmelblau	2, 10, 100	(1, 1,...,1)
12	DBVF	2, 10	(0.1, 0.1,...,0.1)
13	BRYBND	2, 10, 100	(-1, -1,...,-1)
14	Perturbed Quadratic	2, 10, 100	(0.5, 0.5,...,0.5)
15	TRIDIA	2, 10, 100, 500	(1, 1,...,1)
16	Extended Penalty	100	(1, 1,...,1)
17	BALF	2, 10, 100	(0.5, 0.5,...,0.5)
18	Diagonal 1	2, 10, 100	(2, 2,...,2)
19	Diagonal 2	2, 10, 100	(2, 2,...,2)
20	Diagonal 4	2, 10, 100	(1, 1,...,1)
21	Extended Diagonal	100	(1, 1,...,1)

The codes were implemented in Python 3.13 and executed on a Lenovo ThinkPad PC featuring an AMD Ryzen 7 PRO 5850U Processor (1.90 GHz) with Radeon Graphics, 16.0 GB RAM, and Windows 10 Professional 64-bit operating system. The program terminates when $\|q_i\|_\infty < 10^{-6}$ is satisfied or after 5000 iterations are reached. Restart employs the steepest descent direction when the β denominator equals zero or Powell's restart criterion $| q_{i+1}^T q_i | \geq 0.2 \| q_{i+1} \|^2$ from [18] holds. Strong Wolfe conditions (SWC) parameters were set to $\rho = 0.05$ and $\sigma = 0.5$. Performance comparisons rely on iterations (NOI), CPU time, and function evaluations, with results presented in figure 1.

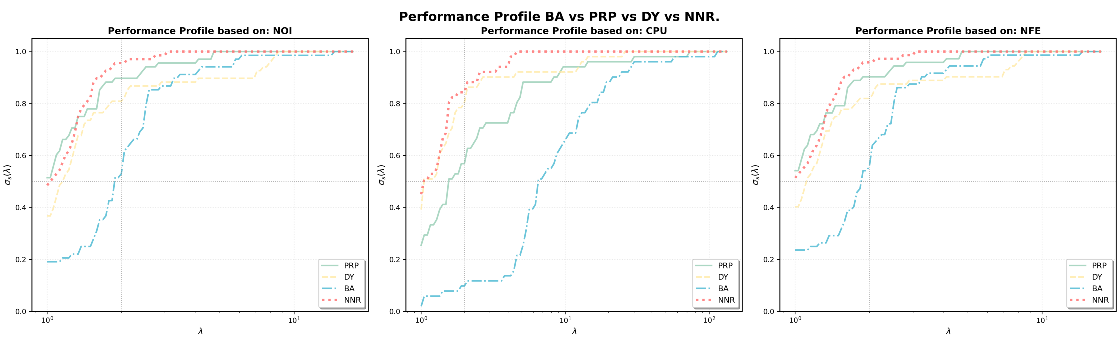

Figure 1. Performance Profile of NNR based on NOI, CPU and NFEV

To rigorously evaluate and compare the tested optimization algorithms, we employ the performance profiles methodology developed by Dolan and Moré [9] as our main assessment tool. This technique offers a thorough, statistically sound framework for analyzing the relative effectiveness and robustness of multiple solvers across a suite of benchmark test cases.

Table 4. Performance of BA, DY, PRP and NNR on Standard Test Functions

Test Function	n	BA			DY			PRP			NNR		
		NOI	CPU	NFEV	NOI	CPU	NFEV	NOI	CPU	NFEV	NOI	CPU	NFEV
Rosenbrock	2	120	0.0988	12002	77	0.0170	7702	37	0.0160	3702	57	0.0130	5702
	10	NAN	0.2967	50001	115	0.0400	11502	159	0.0520	15902	134	0.0500	13402
	100	NAN	0.3461	50001	206	0.0960	20602	222	0.0950	22202	197	0.0896	19702
	500	NAN	0.6518	50001	231	0.2070	23102	113	0.1007	11302	141	0.1210	14102
	1000	NAN	1.5856	50001	204	0.4833	20402	164	0.3899	16402	209	0.4285	20902
	1500	NAN	2.2532	50001	253	0.7837	25302	186	0.5721	18602	191	0.5319	19102
	5000	1129	10.1072	112902	100	0.5496	10002	178	1.0448	17802	140	0.7933	14002
SumSquares	2	17	0.0040	1702	21	0.0010	2102	14	0.0001	1402	3	0.0010	302
	10	61	0.0130	6102	37	0.0030	3702	30	0.0030	3002	34	0.0030	3402
	100	227	0.2783	22702	122	0.0744	12202	92	0.0510	9202	120	0.0770	12002
	500	NAN	3.6836	50001	206	0.8268	20602	276	1.1859	27602	295	1.0676	29502
	1000	NAN	6.9353	50001	299	3.2921	29902	379	4.1681	37902	375	3.0562	37502
	1500	NAN	9.9471	50001	332	4.1791	33202	NAN	7.7346	50001	480	6.5047	48002
Sphere	2	NAN	NAN	NAN	NAN	NAN	NAN	NAN	NAN	NAN	NAN	NAN	NAN
Zakharov	2	19	0.0320	1902	14	0.0020	1402	13	0.0060	1302	8	0.0020	802
	10	NAN	1.7447	50001	41	0.0110	4102	36	0.0310	3602	49	0.0150	4902
	100	NAN	1.7697	50001	106	0.0670	10602	NAN	1.7440	50001	52	0.0320	5202
	500	NAN	2.0273	50001	149	0.1441	14902	NAN	2.0845	50001	79	0.0820	7902
	1000	NAN	2.3725	50001	NAN	0.6427	50001	NAN	2.3379	50001	78	0.0998	7802
	1500	NAN	2.5123	50001	371	0.5522	37102	NAN	2.5750	50001	71	0.0997	7102
	5000	NAN	49.5092	500001	5000	16.0894	500001	5000	48.9777	500001	85	0.2911	8502
DixonPrice	2	37	0.0250	3702	31	0.0060	3102	23	0.0120	2302	22	0.0040	2202
	10	116	0.0820	11602	NAN	0.1687	50001	94	0.0270	9402	70	0.0160	7002
QuadraticQF1	2	15	0.0200	1502	7	0.0010	702	17	0.0240	1702	9	0.0010	902
	10	24	0.0330	2402	16	0.0020	1602	14	0.0030	1402	13	0.0030	1302
	100	63	0.3631	6302	40	0.0840	4002	38	0.0790	3802	34	0.0700	3402
	500	177	6.3460	17702	88	1.6275	8802	84	1.2601	8402	84	1.0815	8402
	1000	215	19.0412	21502	140	7.9143	14002	124	5.0191	12402	129	6.7167	12902
	1500	283	72.2084	28302	155	14.6660	15502	179	22.0357	17902	155	18.9837	15502
Raydan1	2	17	0.0070	1702	12	0.0001	1202	14	0.0010	1402	6	0.0001	602
	10	47	0.0190	4702	30	0.0030	3002	31	0.0030	3102	29	0.0030	2902
	100	257	1.4462	25702	115	0.3946	11502	141	0.8053	14102	210	1.7113	21002
	500	488	12.7067	48802	255	7.9934	25502	386	13.0336	38602	NAN	20.7478	50001
	1000	NAN	23.9089	50001	343	21.8889	34302	NAN	32.9111	50001	NAN	33.7209	50001
	1500	NAN	17.5559	50001	438	47.8704	43802	NAN	41.9734	50001	NAN	45.0240	50001
ExtRosenbrock	2	120	0.0840	12002	77	0.0120	7702	37	0.0100	3702	57	0.0090	5702
	10	239	0.1990	23902	85	0.0180	8502	40	0.0180	4002	57	0.0120	5702
	100	119	0.5621	11902	86	0.1150	8602	47	0.1137	4702	58	0.0700	5802
	500	153	3.6065	15302	93	0.6348	9302	48	0.6551	4802	58	0.3411	5802
	1000	188	10.0164	18802	93	1.2976	9302	49	1.3590	4902	58	0.7367	5802
	1500	147	11.0499	14702	93	1.8936	9302	49	2.0947	4902	60	1.1203	6002

NOI: number of iterations; CPU: time in seconds; NFEV: number of function evaluations.
NAN : failed to converge.

5. Commentaires

Figure 1 displays the Dolan–Moré performance profiles for the proposed NNR method and the PRP, DY, and BA methods over 21 test functions from table 3 and sourced from [1, 5]. When compared to BA, PRP and DY classical CGMs (tables 4 and 5), NNR achieves a success rate of 92.5%, clearly exceeding DY (76.7%), BA (68.4%), and PRP (58.3%). Moreover, NNR is able to solve more than 90% of the test problems, whereas PRP, DY, and BA reach

Table 5. Performance of BA, DY, PRP and NNR on Standard Test Functions

Test Function	n	BA			DY			PRP			NNR		
		NOI	CPU	NFEV	NOI	CPU	NFEV	NOI	CPU	NFEV	NOI	CPU	NFEV
ExtDENSCHNF	2	35	0.0200	3502	141	0.0150	14102	25	0.0150	2502	19	0.0030	1902
	10	39	0.0360	3902	155	0.0300	15502	27	0.0220	2702	19	0.0030	1902
	100	41	0.3178	4102	171	0.2576	17102	29	0.1710	2902	20	0.0180	2002
	500	45	1.6675	4502	186	1.4003	18602	31	0.9789	3102	21	0.1090	2102
	1000	45	3.3574	4502	193	3.0373	19302	31	2.1345	3102	21	0.2004	2102
	1500	47	5.6029	4702	194	4.9225	19402	31	3.1776	3102	22	0.3297	2202
ExtTridiag	10	248	0.2527	24802	117	0.0480	11702	106	0.0460	10602	160	0.0620	16002
ExtHimmelblau	2	32	0.0190	3202	13	0.0010	1302	21	0.0060	2102	16	0.0020	1602
	10	32	0.0320	3202	13	0.0030	1302	21	0.0090	2102	17	0.0030	1702
	100	36	0.2551	3602	14	0.0190	1402	23	0.0810	2302	19	0.0290	1902
DBVF	2	231	0.1857	23102	24	0.0060	2402	21	0.0120	2102	16	0.0050	1602
	10	500	2.0459	50001	270	0.1966	27002	500	2.0757	50001	136	0.0697	13602
BRYBND	2	12	0.0110	1202	8	0.0020	802	13	0.0090	1302	15	0.0060	1502
	10	43	0.0720	4302	500	0.8365	50001	25	0.0430	2502	18	0.0160	1802
	100	42	9.1581	4202	500	42.1593	50001	29	2.8643	2902	20	1.0636	2002
TRIDIA	2	35	0.0200	3502	58	0.0110	5802	21	0.0160	2102	20	0.0030	2002
	10	152	0.1980	15202	500	0.5768	50001	69	0.0400	6902	71	0.0420	7102
	100	73	10.4066	7302	500	32.9441	50001	41	2.9236	4102	46	1.6706	4602
	500	194	81.6116	19402	500	135.6580	50001	43	11.1990	4302	37	5.2662	3702
Diagonal1	2	17	0.0040	1702	21	0.0020	2102	14	0.0010	1402	3	0.0001	302
	10	61	0.0160	6102	37	0.0030	3702	30	0.0030	3002	34	0.0030	3402
	100	227	0.3357	22702	122	0.1060	12202	92	0.0650	9202	126	0.0870	12602
Diagonal2	2	18	0.0060	1802	7	0.0010	702	13	0.0010	1302	9	0.0010	902
	10	49	0.0270	4902	42	0.0070	4202	29	0.0050	2902	28	0.0050	2802
	100	231	1.4270	23102	133	0.8045	13302	162	1.6209	16202	187	2.0310	18702
Diagonal4	2	17	0.0040	1702	16	0.0040	1602	16	0.0050	1602	15	0.0050	1502
	10	26	0.0060	2602	17	0.0048	1702	18	0.0060	1802	16	0.0050	1602
	100	35	0.0710	3502	22	0.0251	2202	22	0.0339	2202	23	0.0287	2302
BALF	2	11	0.0070	1102	7	0.0010	702	6	0.0001	602	10	0.0010	1002
	10	11	0.0110	1102	8	0.0020	802	7	0.0010	702	11	0.0020	1102
	100	13	0.1140	1302	9	0.0090	902	7	0.0060	702	12	0.0120	1202
ExtendedPenalty	100	5	0.0290	502	2	0.0010	202	5	0.0297	502	3	0.0010	302
PerturbedQuad	2	17	0.0090	1702	20	0.0030	2002	14	0.0010	1402	5	0.0001	502
	10	59	0.0240	5902	37	0.0050	3702	29	0.0050	2902	31	0.0040	3102
	100	185	0.6211	18502	114	0.1360	11402	95	0.1010	9502	116	0.1230	11602
ExtendedDiag	100	11	0.0840	1102	19	0.0620	1902	19	0.1210	1902	16	0.0700	1602

NOI: number of iterations; CPU: time in seconds; NFEV: number of function evaluations.
NAN : failed to converge.

only 80%, 72%, and 60%, respectively. Analysis using the Dolan-Moré performance profiles figure 1, confirms that the NNR curve is systematically the highest in all panels (iterations, CPU time, and function evaluations) provides strong evidence of its superior efficiency and robustness.

6. Conclusion

In this paper, we have introduced and analyzed new hybrid CGM denoted NNR, based on two convex combinations of BA and PRP CGMs. We provided a thorough convergence analysis under sufficient descent conditions and demonstrated their global convergence properties. Extensive numerical experiments using performance profile of Dolan and Moré revealed that the NNR methods significantly outperform established algorithms including classical methods (PRP, BA and DY).

REFERENCES

1. A. Y. Al-Bayati and N. H. Al-Assady, *Conjugate gradient method.*, Tech. Research School of Comp. Studies, Leeds Univ., UK, 1986.
2. N. Andrei, *Another hybrid conjugate gradient algorithm for unconstrained optimization*, Numerical Algorithms **47(1)** 143–156, 2008.
3. N. Andrei, *Another nonlinear conjugate gradient algorithm for unconstrained optimization.*, Optimization Methods and Software **24(1)**, 2, 89-104, 2009.
4. N. Andrei, *Nonlinear conjugate gradient methods for unconstrained optimization*, Springer Optimization and its Applications, Romania, 2020.
5. I. Bongartz, A. R. Conn, N. Gould, P. L. Toint, *CUTE: constrained and unconstrained testing environments*, ACM Trans. Math. Soft. **21**, 123–160, 1995.
6. Y. H. Dai, Y. Yuan, *A nonlinear conjugate gradient method with a strong global convergence property*, SIAM J. Optim. , 10, 177-182, 1999.
7. S. Djordjevic, New hybrid conjugate gradient method as a convex combination of LS and CD methods, Filomat,6,1813–1825, 2017.
8. S. Djordjevic, *New hybrid conjugate gradient method as a convex combination of HS and FR*, Journal of Applied Mathematics and Computation, **9**, 366-378, 2018.
9. E. D. Dolan, J. J. Moré, *Benchmarking optimization software with performance profiles*, Math. Program. **91**, 201-213, 2002.
10. R. Fletcher, C. M. Reeves, *Function minimization by conjugate gradients*, Comput. J. **7**, 149-154, 1964.
11. A. Hamdi, B. Sellami and M. Belloufi *A New hybrid conjugate gradient method as a convex combination of BA and DY methods* Commun. Optim. Theory, 2020.
12. M. R. Hestenes, E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Natl. Bur. Stand. **49**, 409–436, 1952.
13. J. K. Liu and S. J. Li, *New hybrid conjugate gradient method for unconstrained optimization.* Applied Mathematics and Computation, 245, 36-43, 2014.
14. R. Mellal, N. Sellami, (2025). “A Novel Hybrid Conjugate Gradient Algorithm for Solving Unconstrained Optimization Problems”. *Statistics, Optimization and Information Computing*, doi: 10.19139/soic-2310-5070-2807.
15. R. Mellal, N. Sellami, B. A. Hassan (2025). “Novel Hybrid Conjugate Gradient Technique Based on the Newton Direction Applied to Image Restoration Problem”. *Statistics, Optimization and Information Computing*, doi: 10.19139/soic-2310-5070-2897.
16. E. Polak, G. Ribiere, *Note sur la convergence des méthodes de directions conjuguées*, Rev. Fran. Infor. Rech. Oper. **16**, 35-43, 1969.
17. B. T. Polyak, *The conjugate gradient method in extreme problems*, U.S.S.R. Comput. Math. Phys. **9**, 94-112, 1969.
18. M. J. D. Powell, *Restart procedures of the conjugate gradient method*, Math. Program. **2**, 241–254, 1977.
19. P. Wolfe, *Convergence conditions for ascent methods*, SIAM Rev. **11**, 226-235, 1969.
20. P. Wolfe, *Convergence conditions for ascent methods. II: Some corrections*, SIAM Rev. **13**, 185-188, 1971.