

Algorithm for Scaling Variables in Minimization Methods

Elena Tovbis ¹, Vladimir Krutikov ^{1,2} and Lev Kazakovtsev ^{1,*}

¹ Institute of Informatics and Telecommunications, Reshetnev Siberian State University of Science and Technology, 31 Krasnoyarskii Rabochii Prospekt, Krasnoyarsk 660037, Russia; sibstu2006@rambler.ru (E.T.); krutikovvn@rambler.ru (V.K.)

² Department of Applied Mathematics, Kemerovo State University, 6 Krasnaya Street, Kemerovo 650043, Russia

* Correspondence: levk@bk.ru

Abstract

Eliminating poor scaling of variables of minimized functions is a pressing issue in solving high-dimensional minimization problems where it is impossible to use methods that change the metric of the space with full-scale metric matrices. In this paper, we propose an iterative method for scaling variables using a diagonal metric matrix and apply it to the gradient minimization method and the conjugate gradient method. In conjugate gradient methods, for quadratic functions, the descent directions are orthogonal to the previous gradient differences. In the proposed method, the transformation of diagonal metric matrices is based on the noted property. For the gradient method with a diagonal metric matrix, an estimate for the convergence rate on strongly convex functions with a Lipschitz gradient was obtained. A computational experiment was conducted, and the presented methods were compared with the Hestenes–Stiefel conjugate gradient method. On the given set of test functions, the gradient method with scaling is comparable in convergence rate to the Hestenes–Stiefel conjugate gradient method, while the conjugate gradient method with scaling matrices significantly outperforms the Hestenes–Stiefel conjugate gradient method. The obtained results confirm the acceleration properties of scaling methods in the case of poor scaling of the variables of the function being minimized. This allows us to conclude that the studied methods can be used alongside conjugate gradient methods to solve smooth, high-dimensional optimization problems with a high degree of conditionality.

Keywords: minimization method; variables scaling; minimization methods; metric matrix

1. Introduction

Problems in different application areas can be represented as optimization problems of some objective function, for instance, load balancing and scheduling [1–3], energy management [4], machine learning and deep learning [5–7].

In this paper, we consider a problem of minimizing a convex differentiable function $f(x)$, $x \in R^n$, where R^n is a finite-dimensional Euclidean space and n is the problem dimension.

Many nature-inspired derivative-free optimizers are weak in optimizing functions with high conditioning [8,9]. To solve a problem with a high degree of conditionality, minimization methods with a change in the space metric are used: Newton’s method (NM) [10], quasi-Newton methods (QNM) [10,11], and relaxation subgradient methods (RSM) using metric matrices [12]. The efforts of many researchers have been directed towards modifying these methods. For instance, in Ref. [13], the symmetric rank-1 quasi-Newton method was accelerated with Nesterov’s gradient. Sampled versions of the quasi-Newton method were considered in [14,15]. Authors in [16] presented a stochastic quasi-



Received: 25 December 2025

Revised: 17 January 2026

Accepted: 19 January 2026

Published: 1 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

Newton method. A greedy version of QNM is considered in [17]. An online whitening approach operating in the black-box optimization paradigm was proposed in [8]. To reduce the conditionality of the problem, the authors transform the objective function with high conditioning into another function that is easier to model by surrogates.

A feature of quasi-Newton and subgradient methods is their acceleration properties, proven theoretically [11,12,18]. However, in high-dimensional minimization problems, these methods are inapplicable due to the requirement to store the metric matrices.

In many cases, ill-conditioning of a minimization problem may be caused by poor scaling of variables, which can be eliminated by using diagonal metric matrices. In this regard, the development of methods for scaling variables in the form of diagonal metric matrices seems relevant. Such methods for scaling variables can be used in both the gradient method and the conjugate gradient method. Note that the conjugate gradient method itself is an effective tool for solving ill-conditioned high-dimensional problems [10,19].

In the global optimization literature, there is also an approach that mitigates the negative effects of scaling and conditioning through algorithmic design. In particular, the paper [20] proposes a homogeneity-oriented approach and shows how algorithmic behavior under scaling is closely linked to robustness with respect to ill-conditioning, and formalizes strong homogeneity as a desirable property.

In Ref. [18], an iterative process of transforming the metric matrices, proposed for heuristic reasons in [21] to organize the relaxation subgradient method, is used in smooth optimization problems to accelerate the convergence rate of the gradient method and the conjugate gradient method. In the RSM, a descent direction is constructed using machine learning tools that forms an obtuse angle with all subgradients in the neighborhood of the current minimum. Minimization along such a direction allows one to go beyond this neighborhood. In Ref. [18], it is shown that the problem of finding the descent direction for the r-algorithm [21] is solved by an iterative method similar to the iterative least squares method (ILS). As a result, we obtain various RSMs with a change in the space metric, where one special case reproduces the well-known r-algorithm. In contrast to quasi-Newton methods, approximating the characteristics of second derivative matrices, the RSMs rely on the properties of the level surface topology structure and, similarly to conjugate gradient methods, generate conjugate descent directions in the quadratic case [18]. As shown in Ref. [18], the ILS metric matrix has the property of partial orthogonalization of the descent vectors to the directions of the difference of adjacent gradients. In the quadratic case, the use of the ILS metric allows one to increase the degree of conjugacy of the descent vectors and thereby increase the convergence rate of the minimization method. In [18], it is proved that the gradient method in the ILS metric on strongly convex functions with a Lipschitz gradient converges linearly and has acceleration properties qualitatively similar to those of the Newton method. In the case of smooth functions, the formulas for transforming the descent vectors of the subgradient method can be replaced by the Hestenes–Stiefel CGM formulas. The combination of the transformation by the ILS metric with the Hestenes–Stiefel CGM allows us to obtain a method with finite convergence on quadratic functions.

Unlike conjugate gradient methods, the metric matrices in [18] produce partial orthogonalization of the descent vectors to the gradient differences. The main contribution of this paper is the idea of preserving partial orthogonalization to gradient differences in the form of a metric matrix, which is implemented using diagonal metric matrices and applied in the gradient method and the Hestenes–Stiefel conjugate gradient method.

In this paper, we use methods from [18], which utilize a full-scale metric matrix transformation process. Based on this transformation, we derive a method for transforming the diagonal scaling matrix of variables. For the gradient method with diagonal matrix

transformation, we obtain an estimate for the convergence rate on strongly convex functions with a Lipschitz gradient.

A numerical study and comparison of the proposed methods with the Hestenes–Stiefel conjugate gradient method were conducted using a series of tests. It was found that the gradient method with scaling significantly outperforms the classical steepest descent method and is comparable in convergence rate to the Hestenes–Stiefel conjugate gradient method. The conjugate gradient method with scaling matrices significantly outperforms both gradient methods with scaling and the Hestenes–Stiefel conjugate gradient method in convergence rate.

These results indicate that the proposed methods can be useful in the presence of poorly scaled variables. In this case, the degree of conditionality of the problem can be reduced by using methods with scaling metric matrices.

The rest of the paper is organized as follows. In Section 2, the idea of a gradient minimization algorithm with variable scaling is described. In Section 3, the convergence rate of the gradient minimization method with a diagonal matrix for variable scaling is investigated. In Sections 4 and 5, a gradient-based algorithm with scaling metric matrices and a conjugate gradient method with scaling metric matrices were implemented. In Section 6, a numerical experiment was conducted, and its results were discussed. Section 7 concludes the work.

2. Gradient Minimization Algorithm with Variable Scaling

In Ref. [18], the gradient method and the conjugate gradient method use metric matrices to accelerate the convergence rate. Unlike the conjugate gradient methods, the metric matrices in this case produce a partial orthogonalization of the descent vectors to the gradient differences. These methods are theoretically justified and have proven effective in solving ill-conditioned test minimization problems.

Minimization method with space dilation in the direction of subgradients difference (r-algorithm) [21] for the case of smooth functions has the following form:

$$x_{k+1} = x_k - \gamma_k s_k, \quad s_k = H_k g_k, \quad \gamma_k = \underset{\gamma}{\operatorname{argmin}} f(x_k - \gamma s_k), \quad k = 0, 1, 2, \dots \quad (1)$$

$$H_{k+1} = H_k - \left(1 - \frac{1}{\alpha^2}\right) \frac{H_k y_k y_k^T H_k^T}{(y_k, H_k y_k)}, \quad \alpha > 1, \quad H_0 = I, \quad y_k = g_k - g_{k+1}. \quad (2)$$

Here, the initial point x_0 is given; s_k is the descent direction; α is the space dilation parameter; n is the problem dimension; and $x_k, s_k, g_k, y_k \in R^n, H_k \in R^{n \times n}, g_k = \nabla f(x_k)$.

In Ref. [18], the acceleration properties inherent in the algorithm, consisting of Equations (1) and (2), on strongly convex functions with a Lipschitz gradient are substantiated. A transformation of the space metric (2) is used to accelerate the conjugate gradient method. In the transformation of the matrices of metric (2), unlike quasi-Newton methods, the properties of the quadratic model of the function being minimized are not directly used. Nevertheless, representations based on the conjugacy of descent vectors are indirectly utilized.

For large dimensions, Method (1), (2) is inapplicable due to the need to store the matrix H . In this paper, we present algorithms similar to (1), (2) that use diagonal matrices, the transformation of which is carried out similarly to (2), but only for the diagonal elements of the matrices. Such algorithms require a vector of dimension n to store the matrix elements. To justify the proposed algorithm, we use a theoretical analysis of its convergence and an experimental analysis on tests, confirming its accelerating convergence properties.

We form an algorithm similar to the method (1), (2) but with a diagonal matrix H . We will assume that the non-diagonal elements of the matrices are zero, and only the diagonal elements are subject to transformations. We denote

$$\beta = 1 - \alpha^{-2}, \quad 1 > \beta \geq 0. \quad (3)$$

To denote the transformation of a diagonal matrix H , we will use the operators $H(H, y, \beta)$. Then, transformation (2) for diagonal matrices will take the form:

$$H_{k+1} = H(H_k, y_k, \beta), \quad 1 > \beta \geq 0, \quad H_0 = I, \quad y_k = g_k - g_{k+1}.$$

The transformation Formula (2) for the diagonal elements of the matrix H , taking into account (3), can be represented as follows:

$$H(H, y, \beta)_{i,i} = H_{i,i} - \beta \frac{(H_{i,i}y_i)^2}{(y, Hy)}, \quad i = 0, 1, 2, \dots, n. \quad (4)$$

Here and below, the scalar product with a diagonal matrix is $(y, Hy) = \sum_{i=1}^n H_{i,i}y_i^2$. In (4), only the diagonal elements of the matrix are transformed.

3. On the Convergence Rate of the Gradient Minimization Method with a Diagonal Matrix for Variables Scaling

We form a gradient minimization algorithm with diagonal scaling based on the matrix transformation (4):

$$x_{k+1} = x_k - \gamma_k s_k, \quad s_k = H_k g_k, \quad \gamma_k = \underset{\gamma}{\operatorname{argmin}} f(x_k - \gamma s_k), \quad k = 0, 1, 2, \dots, \\ H_{k+1} = H(H_k, y_k, \beta), \quad 1 > \beta \geq 0, \quad H_0 > 0, \quad y_k = g_k - g_{k+1}. \quad (5)$$

$$H(H, y, \beta)_{i,i} = H_{i,i} - \beta \frac{(H_{i,i}y_i)^2}{(y, Hy)}, \quad i = 0, 1, 2, \dots, n.$$

Let us study the convergence rate of the algorithm with diagonal scaling (1),(5),(4). Here and below, we assume that the matrices used are diagonal.

In the justification part, we omit the iteration index k , assuming that $y = g - g^+$, where g is the gradient vector at the current minimum point, and g^+ is the gradient vector at a point obtained as a result of minimization along the direction $s = Hg$.

Denote inverse matrix transformation operator for $H(H, y, \beta)$ as $A(A, y, \beta) = (H(H, y, \beta))^{-1}$ assuming $A = H^{-1}$. Inverting (4), we obtain:

$$A(A, y, \beta)_{i,i} = \left(H_{i,i} - \beta \frac{(H_{i,i}y_i)^2}{(y, Hy)} \right)^{-1}, \quad i = 0, 1, 2, \dots, n.$$

Taking into account the equality for diagonal matrices $A_{i,i} = 1/H_{i,i}$, transform the latter:

$$A(A, y, \beta)_{i,i} = \frac{1}{\frac{1}{A_{i,i}} - \beta \frac{(y_i/A_{i,i})^2}{(y, Hy)}} = \frac{A_{i,i}}{1 - \beta \frac{y_i^2/A_{i,i}}{\sum_{j=1}^n y_j^2/A_{j,j}}} = \frac{A_{i,i}}{1 - \beta z_i}. \quad (6)$$

According to (6), the following relations are valid for z_i :

$$0 \leq z_i = \frac{y_i^2}{A_{i,i}(y, Hy)} = \frac{\frac{y_i^2}{A_{i,i}}}{\sum_{j=1}^n \frac{y_j^2}{A_{j,j}}} \leq 1, \quad \sum_{i=1}^n z_i = 1. \quad (7)$$

Theorem 1. Let $A_k > 0$, matrix A_{k+1} is obtained as a result of the transformation (6) of the process (1),(5),(4). Then, $A_{k+1} > 0$ and

$$SpA_{k+1} \leq SpA_k \left(1 + \frac{\beta(y_k, y_k)}{(1 - \beta)(g_k, g_k)} \right), \quad (8)$$

$$\det A_{k+1} \geq (1 + \beta) \det A_k. \quad (9)$$

Proof of Theorem 1. Based on (7), (5), and (6), it follows that the matrix elements as a result of the transformation (6) can only increase, which proves the non-degeneracy of the matrix $A_{k+1} > 0$.

To transform expression (6), we need an inequality that is valid for $1 > \beta \geq 0$ and $1 \geq z_i \geq 0$.

$$(1 - \beta z_i)^{-1} \leq 1 + z_i \beta / (1 - \beta). \quad (10)$$

Inequality (10) holds for $z_i = 0$. Given the condition $1 \geq z_i > 0$, we transform (10):

$$1 \leq (1 + z_i \beta / (1 - \beta))(1 - \beta z_i) = 1 + z_i(\beta / (1 - \beta) - \beta) - z_i^2 \beta^2 / (1 - \beta)$$

From this follows the inequality:

$$0 \leq (1 / (1 - \beta) - 1) - z_i \beta / (1 - \beta) = (1 - z_i) \beta / (1 - \beta).$$

The latter, taking into account the restrictions $1 > \beta \geq 0$ and $1 \geq z_i > 0$, proves (10).

We transform equality (6), taking into account (10):

$$A(A, y, \beta)_{i,i} = \frac{A_{i,i}}{1 - \beta z_i} \leq A_{i,i} \left(1 + \frac{\beta}{1 - \beta} z_i \right) = A_{i,i} \left(1 + \frac{\beta y_i^2}{(1 - \beta)(y, Hy)} \right). \quad (11)$$

Due to the exact one-dimensional descent along the direction $s = Hg$, the gradient at the new point will be orthogonal to the descent direction $(s, g^+) = (Hg, g^+) = 0$. From this follows the inequality:

$$(Hy, y) = (Hg, g) + (Hg^+, g^+) - 2(Hg, g^+) \geq (Hg, g).$$

Denote $A^+ = A(A, y, \beta)$. We use the last inequality in (11). As a result, we obtain the estimate:

$$A_{i,i}^+ = A(A, y, \beta)_{i,i} \leq A_{i,i} \left(1 + \frac{\beta y_i^2}{(1 - \beta)(g, Hg)} \right).$$

Based on this inequality, we obtain inequalities for the matrix trace $Sp(A^+)$:

$$Sp(A^+) \leq Sp(A) + \frac{\beta(y, y)}{(1 - \beta)(g, Hg)} = Sp(A) \left(1 + \frac{\beta(y, y)}{(1 - \beta)(g, Hg)Sp(A)} \right). \quad (12)$$

Denote by M a maximal eigenvalue of matrix A . Transform the expression in the denominator of (12), taking into account the inequality $Sp(A) \geq M$:

$$(g, Hg)Sp(A) = (g, A^{-1}g)Sp(A) \geq (g, g)Sp(A)/M \geq (g, g).$$

Based on (12), taking into account the last inequality, we obtain the estimate:

$$Sp(A^+) \leq Sp(A) \left(1 + \frac{\beta(y, y)}{(1 - \beta)(g, Hg)Sp(A)} \right) \leq Sp(A) \left(1 + \frac{\beta}{(1 - \beta)} \frac{(y, y)}{(g, g)} \right).$$

The last inequality proves (8). According to (11), we write and transform the expression for the determinant of the matrix A^+ :

$$\det(A^+) = \prod_{i=1}^n \frac{A_{i,i}}{1 - \beta z_i} = \det(A) \prod_{i=1}^n \frac{1}{1 - \beta z_i} \geq \det(A) \prod_{i=1}^n (1 + \beta z_i).$$

From here and (7) we obtain:

$$\det(A^+) \geq \det(A) \prod_{i=1}^n (1 + \beta z_i) \geq \det(A)(1 + \beta),$$

which proves (9). $\square$

Let us determine the conditions under which the convergence rate of algorithm (1),(5),(4) is estimated.

Condition 1. We will assume that the function $f(x)$, $x \in R^n$, is minimized, differentiable, and strongly convex on R^n , i.e., there exists such ρ that $\forall x, y \in R^n$ and $\alpha \in [0, 1]$ the inequality holds

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y) - \alpha(1 - \alpha)\rho \|x - y\|^2 / 2,$$

And its gradient $g(x) = \nabla f(x)$ satisfies the Lipschitz condition:

$$\|g(x) - g(y)\| \leq L\|x - y\| \quad \forall x, y \in R^n, L > 0.$$

Denote by x^* a minimum point of a function $f(x)$, $f^* = f(x^*)$, $f_k = f(x_k)$ and $g_k = g(x_k) = \nabla f(x_k)$. The iteration of the gradient-consistent method for exact one-dimensional descent has the form:

$$x_{k+1} = x_k - \beta_k s_k, \quad \beta_k = \underset{\beta \geq 0}{\operatorname{argmin}} f(x_k - \beta s_k), \quad (s_k, g_k) > 0, k = 0, 1, 2, \dots, \quad (13)$$

where the initial point x_0 is set.

The following theorem shows that the presence of gradient changes as a result of iterations of the method (13) with exact one-dimensional descent leads to a decrease in the function value.

Theorem 2 ([11,12]). Let $f(x)$ satisfy Condition 1. Then, for the sequence $\{f_k\}$, $k = 0, 1, 2, \dots$ given by (13), the following estimate takes place:

$$f_{k+1} - f^* \leq (f_0 - f^*) \exp \left[-\frac{\rho^2}{L^2} \sum_{i=0}^k \frac{\|y_i\|^2}{\|g_i\|^2} \right], \quad (14)$$

where $y_i = g_{i+1} - g_i$.

Theorem 3. Let $f(x)$ satisfy Condition 1. Then, for the sequence $\{f_k\}$, $k = 0, 1, 2, \dots$, given by (1) and (4) with a bounded initial matrix $A_0 = H_0^{-1}$,

$$m_0 \leq (A_0 z, z) / (z, z) \leq M_0, \quad \forall z \in R^n, \|z\| > 0, \quad (15)$$

the following estimate takes place:

$$f_{k+1} - f^* \leq (f_0 - f^*) \exp \left\{ -\frac{\rho^2(k+1)}{L^2 n} \left[\frac{(1-\beta) \ln(1+\beta)}{\beta} + \frac{n(1-\beta) \ln(m_0/M_0)}{(k+1)\beta} \right] \right\}. \quad (16)$$

Proof of Theorem 3. Based on the relation between the arithmetic mean and the geometric mean of the matrix $A > 0$ eigenvalues, we obtain:

$$Sp(A)/n \geq [\det(A)]^{1/n}.$$

From here, using (8) and (9), we obtain:

$$\frac{Sp(A_0)}{n} \prod_{i=0}^k \left[1 + \frac{\beta}{(1-\beta)} \frac{\|y_i\|^2}{\|g(x_i)\|^2} \right] \geq \frac{Sp(A_{k+1})}{n} \geq (\det(A_{k+1}))^{1/n} = [(1+\beta)^{k+1} \det(A_0)]^{1/n}.$$

The last relation based on inequality $1 + p \leq \exp(p)$ we transform to:

$$\frac{Sp(A_0)}{n} \exp \left[\frac{\beta}{(1-\beta)} \sum_{i=0}^k \frac{\|y_i\|^2}{\|g(x_i)\|^2} \right] \geq (1+\beta)^{(k+1)/n} (\det(A_0))^{1/n}.$$

Taking into account the constraints (15), we transform the last inequality:

$$M_0 \exp \left[\frac{\beta}{(1-\beta)} \sum_{i=0}^k \frac{\|y_i\|^2}{\|g(x_i)\|^2} \right] \geq m_0 (1+\beta)^{(k+1)/n}.$$

Taking the logarithm of the latter, we find:

$$\left[\frac{\beta}{(1-\beta)} \sum_{i=0}^k \frac{\|y_i\|^2}{\|g(x_i)\|^2} \right] \geq \frac{(k+1) \ln(1+\beta)}{n} + \ln\left(\frac{m_0}{M_0}\right).$$

From this follows the inequality:

$$\sum_{i=0}^k \frac{\|y_i\|^2}{\|g(x_i)\|^2} \geq \frac{(k+1)(1-\beta) \ln(1+\beta)}{n\beta} + \frac{(1-\beta) \ln(m_0/M_0)}{\beta},$$

which, together with the estimate (14) of Theorem 1, proves (16). $\square$

4. Gradient-Based Minimization Algorithm with Scaling Metric Matrices

We use the presented variable scaling matrices to organize gradient methods. The matrix H_k is diagonal, with zero non-diagonal elements. Components will be added to the algorithms to eliminate infinite decreases in the matrix diagonal elements, as well as to eliminate variants of their significant differences.

The gradient algorithm (1),(5),(4) with the transformation of matrix H according to Formula (4) has the following form (Algorithm 1).

Algorithm 1 (HG, Gradient method with H-scaling)

1. Set the initial point x_0 , $g_0 = \nabla f(x_0)$, initial diagonal matrix H_0 and parameters $\varepsilon \in [10^{-2} - 10^{-9}]$ and $\beta \in [0.001 - 0.9]$.

2. For $k = 0, 1, 2, \dots$ do

2.1 Find a new minimum approximation:

$$x_{k+1} = x_k - \gamma_k s_k, \quad s_k = H_k g_k, \quad \gamma_k = \underset{\gamma}{\operatorname{argmin}} f(x_k - \gamma s_k),$$

2.2 Perform a metric matrix transformation:

$$H_{i,i,k} = H_{i,i,k} / \max_{1 \leq j \leq n} H_{j,j,k}, \quad i = 0, 1, 2, \dots, n, \quad (17)$$

$$H_{i,i,k} = \max\{H_{i,i,k}, \varepsilon\}, \quad i = 0, 1, 2, \dots, n, \quad (18)$$

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

$$H_{k+1} = H(H_k, y_k, \beta). \quad (19)$$

2.3 If the stopping criterion is met, then stop.

Transformation (17) of the HG algorithm is designed to prevent excessive matrix decrease. According to the formulas of transformation (4), changing the overall scale of the matrix does not affect the proportions between the diagonal elements of the matrix obtained as a result of transformation (19). That is, multiplying the matrix H by the number does not affect these proportions. On the other hand, due to the exact one-dimensional descent, the scale of the matrix H also does not affect the iteration at Step 2.1. Thus, transformation (17) does not affect the results of the algorithm.

Transformation (18) eliminates the possibility of matrix degeneracy. After transformation (17), the maximum matrix element is equal to one. The value in (18) defines the minimum value of the matrix elements. In (19), the (4) transformation of the metric matrix is performed.

The presented algorithm can be viewed as a gradient method with metric scaling matrices. We will numerically study the convergence properties of the HG algorithm on smooth functions and show that the use of metric scaling matrices imparts acceleration properties to the gradient method. In this regard, we use these metric transformations in the conjugate gradient method.

5. Conjugate Gradient Method with Scaling Metric Matrices

Scaling transformations for the gradient method can be useful in other methods. In this paper, we will consider the use of metric transformations in the conjugate gradient method.

The Hestenes–Stiefel conjugate gradient method has the form [19]:

$$x_{k+1} = x_k - \gamma_k s_k, \quad \gamma_k = \underset{\gamma}{\operatorname{argmin}} f(x_k - \gamma s_k), \quad k = 0, 1, 2, \dots, \quad (20)$$

$$s_k = g_k - \frac{(g_k, y_k)}{(s_k, y_k)} s_k, \quad y_k = g_k - g_{k+1}. \quad (21)$$

In Ref. [18], the Hestenes–Stiefel method is proposed using the transformation of the space metric with full-scale matrices (2):

$$s_0 = g_0, \quad s_{k+1} = H_k g_{k+1} - \frac{(H_k g_{k+1}, y_k)}{(s_k, y_k)} s_k, \quad y_k = g_{k+1} - g_k, \quad (22)$$

$$H_0 = I, \quad H_{k+1} = H_k - (1 - \frac{1}{\alpha^2}) \frac{H_k y_k y_k^T H_k^T}{(y_k, H_k y_k)}, \quad \alpha > 1. \quad (23)$$

Algorithm (20),(22),(23) uses the full matrix H (2). As shown in [18], method (20),(22),(23) has the properties of the conjugate gradient method, and the ILS metric matrix has the property of partially orthogonalizing the descent vectors to the directions of the difference of adjacent gradients. In the conjugate gradient method, the descent vector is orthogonalized to the direction of the gradient difference at each iteration. In the quadratic case, using the ILS metric allows one to increase the degree of conjugacy of the descent vectors and thereby accelerate the convergence rate of the minimization method.

We use the scheme of method (20),(22),(23) to organize the algorithm with diagonal scaling (4):

$$H_{k+1} = H(H_k, y_k, \beta), \quad 1 > \beta \geq 0. \quad (24)$$

In the method (20),(22),(24), the diagonal matrix H is used. Detailed implementation of the method (20),(22),(24) is presented in Algorithm 2 below.

Algorithm 2 (HHS, Hestenes–Stiefel method with H-scaling)

1. Set the initial point x_0 , $g_0 = \nabla f(x_0)$, initial diagonal matrix $H_0 > 0$, $s_0 = H_0 g_0$ and parameters $\varepsilon \in [10^{-2} - 10^{-9}]$ and $\beta \in [0.001 - 0.9]$.

2. For $k = 0, 1, 2, \dots$ do

2.1 Find a new minimum approximation:

$$x_{k+1} = x_k - \gamma_k s_k, \quad \gamma_k = \underset{\gamma}{\operatorname{argmin}} f(x_k - \gamma s_k),$$

2.2 Compute a new descent direction

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

$$s_{k+1} = H_k g_{k+1} - \frac{(H_k g_{k+1}, y_k)}{(s_k, y_k)} s_k.$$

2.3 Perform a metric matrix transformation:

$$H_{i,i,k} = H_{i,i,k} / \max_{1 \leq j \leq n} H_{j,j,k}, \quad i = 0, 1, 2, \dots, n,$$

$$H_{i,i,k} = \max\{H_{i,i,k}, \varepsilon\}, \quad i = 0, 1, 2, \dots, n,$$

$$H_{k+1} = H(H_k, y_k, \beta).$$

2.4 If the stopping criterion is met, then stop.

Transformation (22) is performed at Step 2.2. Transformation (17) of the HG algorithm is intended to eliminate excessive decrease of the matrix. Transformation (18) eliminates the possible occurrence of matrix degeneracy by replacing excessively small values of the elements. In (19), transformation (4) of the metric matrix is performed.

6. Numerical Results and Discussion

To identify the capabilities of the presented algorithms for solving problems of minimizing smooth functions and to test the effectiveness of the presented methods, a numerical study was conducted. The methods are compared with the Hestenes–Stiefel conjugate gradient method.

All methods used in the study utilized the one-dimensional search, where the gradient and function value are used simultaneously as information for organizing the method. This is particularly advantageous in conditions where the costs of computing the gradient and function are comparable.

We will denote:

HG—Gradient method with H-scaling (1),(5),(4) (HG algorithm);

HHS—Hestenes–Stiefel method with H-scaling (20),(22),(24) (HHS algorithm);

HS—Hestenes–Stiefel method (20) and (21);

GR—Gradient method.

In the HG and HHS methods, full-scale metric matrices are absent, which allows one to use them in solving high-dimensional problems. One of the most successful methods for solving high-dimensional problems with a high degree of conditionality is the conjugate gradient method. Therefore, when testing the proposed methods, we also use the Hestenes–Stiefel conjugate gradient method (HS) as a benchmark.

The set of functions under testing includes a quadratic function with a known condition number. Since we know the function's condition number, we can estimate its complexity for minimization. Also, based on the results of minimizing the quadratic function, we obtain information about the method's behavior in a certain neighborhood of the current minimization point of the real function, where its quadratic representation is valid.

Another important point is the ability to specify the same degree of conditioning regardless of the problem dimension. Theoretical studies indicate that, with the same degree of conditioning, gradient methods should have approximately equal iteration costs. The parameters of non-quadratic functions also allow one to control the condition number.

The tests included functions with both straight and curved ravines. With curved ravines, the eigenvectors of the second derivative matrix change as we approach the minimum, leading to outdated metric matrices and a decrease in the convergence rate. Further successful progress requires retuning the method parameters. Using such functions, we will observe the method's response to the variability of the quadratic representation of the function as we approach the minimum.

We also included as a test a non-quadratic function whose isosurface topology matches that of a quadratic function. In this case, it is of interest to study the stability of the convergence rate for the pointed methods regarding the quadratic properties of the function.

One of the test problems reflects the variability of scaling by variables when approaching an extremum. In this case, there is an active change in the isosurface topology due to changes in scales along the coordinate axes. Finally, the last of the test functions includes ravines, where one side of the ravine has an exponential change, and the other side has a linear change.

6.1. Quadratic Function

We used the following function as a quadratic one:

$$f_Q(x, [amax]) = \frac{1}{2} \sum_{i=1}^n a_i x_i^2, \quad a_i = a \max^{\frac{i-1}{n-1}}.$$

Starting point was $x_0 = 100(n - i + 1)$, $i = 1, 2, \dots, n$. Stopping criterion was $f(x_k) - f^* \leq 10^{-10}$, where f^* is the optimal function value. Eigenvalues a_i of this func-

tion have the boundaries $\lambda_{\min} = 1$ and $\lambda_{\max} = a_{\max}$. Sensitivity analysis of the algorithms depending on the learning rate parameter β is presented in Tables A1 and A2. According to the results, a higher convergence rate (smaller number of iterations) is ensured by a higher degree of learning rate, that is, by larger β values. The results indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms for any value of β compared to the HS method, which resulted in 702 iterations and 1405 gradient and function calculations for $a_{\max} = 10^4$ and 6438 iterations and 15,238 gradient and function calculations for $a_{\max} = 10^6$. In addition, increasing the degree of function degeneracy allows one to see the effects of accelerating convergence due to the use of scaling matrices in the conjugate gradient method.

Tables A3 and A4 show the results of minimization for HG, HHS, HS, and GR algorithms with $\beta = 0.9$. The results indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms. In Table A4, the GR method failed to solve the problem. Here, the results indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms, and the advantages of the presented methods over the Hestenes–Stiefel conjugate gradient method become obvious.

6.2. Function with Ellipsoidal Ravine

The following function has a multidimensional ellipsoidal ravine. Minimization occurs when moving along this ravine to the minimum point:

$$f_E(x, [a_{\max}, b_{\max}]) = (1 - x_1)^2 + a \max \left(1 - \sum_{i=1}^n x_i^2 / b_i \right)^2, b_i = b \max^{\frac{i-1}{n-1}}, \quad (25)$$

The stopping criterion was $f - f^* \leq 10^{-4}$. This function has two complexity levels: a curved ravine and a function with a degree greater than quadratic. At its minimum, the function is degenerate. For each type of experiment, we took a different starting point.

Table A5 shows the results of the sensitivity analysis of the algorithms depending on the learning rate parameter β for the f_E function ($n = 1000$). According to the results, a higher convergence rate is ensured by a higher degree of learning rate for the HG algorithm. For the HHS algorithm, the optimal β value is 0.5. The results indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms for almost any value of β compared to the HS method, which resulted in 1195 iterations and 3078 gradient and function calculations. Table A6 shows the results of minimization for HG, HHS, HS, and GR algorithms with a value of $\beta = 0.9$. The results indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms. The advantages of the presented methods over the Hestenes–Stiefel conjugate gradient method are revealed.

6.3. Function with Multidimensional Ellipsoidal Ravine and Non-Degenerate Minimum Point

The following function also has a multidimensional ellipsoidal ravine, and the minimization occurs when moving along a curvilinear ravine to a minimum point:

$$f_{EX}(x, [a_{\max}, b_{\max}]) = (1 - x_1)^2 + a \max \left(1 - \sum_{i=1}^n x_i^2 / b_i \right)^2 + \frac{1}{2} \sum_{i=2}^n x_i^2 / b_i, b_i = b \max^{\frac{i-1}{n-1}}.$$

The stopping criterion was $f - f^* \leq 10^{-10}$. Due to the additional quadratic term in f_{EX} , the minimum point ceases to be degenerate. This allows the gradient method to find the minimum of the function with higher accuracy compared to the function f_E . Table A7 shows the results of f_{EX} function minimization ($n = 1000$) for different values of learning rate parameter β . According to the Table A7 results, a convergence rate is approximately the same for $\beta \in [0.2-0.9]$. The results indicate an acceleration of the convergence rate

when using diagonal matrices in the algorithms for any value of β compared to HS method, which resulted in 32,294 iterations and 80,326 gradient and function calculations.

Table A8 shows the results of minimization for HG, HHS, HS and GR algorithms with value of $\beta = 0.9$, where the advantages of the presented methods over the Hestenes–Stiefel conjugate gradient method are revealed.

6.4. Non-Quadratic Function

The following function has topological similarity of level surfaces to a quadratic function.

$$f_{Q2}(x, [amax]) = \left(\sum_{i=1}^n a_i x_i^2 \right)^2, a_i = a \max^{\frac{i-1}{n-1}}.$$

The stopping criterion was $f - f^* \leq 10^{-10}$. The starting point was $x_0 = (1, 1, \dots, 1)$.

Table A9 shows the results of the f_{Q2} function minimization ($n = 1000$) for different values of the learning rate parameter β . According to the results, a higher convergence rate is ensured by $\beta = 0.4$ for the HG algorithm and by $\beta = 0.8$ for the HHS algorithm. The results indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms for any value of β compared to the HS method, which resulted in 490 iterations and 1000 gradient and function calculations. Here, the advantages of the presented methods compared to the Hestenes–Stiefel conjugate gradient method become obvious.

The results in Table A10 for $\beta = 0.9$ indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms. The advantages of the presented methods over the Hestenes–Stiefel conjugate gradient method are revealed.

The next experiment (Tables A11 and A12) was performed for a higher condition number.

According to the Table A11 results, a higher convergence rate is ensured by $\beta = 0.5$ for the HG algorithm and by $\beta = 0.8$ for the HHS algorithm. The results indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms for any value of β compared to the HS method, which resulted in 4746 iterations and 10,617 gradient and function calculations. Table A12 shows the results of minimization for HG, HHS, HS, and GR algorithms with a value of $\beta = 0.9$. The GR method failed to solve the problem. The advantages of the presented methods over the Hestenes–Stiefel conjugate gradient method are revealed.

6.5. Function with Scalable Variables

The next function used additional variables c_i to change the scales a_i for each variable.

$$f_{abc}(x, [a \max, b \max]) = \frac{1}{2} \sum_{i=1}^n a_i c_i x_i^2, a_i = a \max^{\frac{i-1}{n-1}},$$

$$c_i = \frac{b \max}{b_i} \left(\frac{x_i^2}{1 + x_i^2} \right) + b_i \left(1 - \frac{x_i^2}{1 + x_i^2} \right), b_i = b \max^{\frac{i-1}{n-1}}.$$

This function near the extremum will have the form:

$$f_{abc}(x, [amax, bmax]) \approx \frac{1}{2} \sum_{i=1}^n a_i b_i x_i^2.$$

Far from the extremum, we obtain a function in which the coefficients are used in reverse order.

$$f_{abc}(x, [amax, bmax]) \approx \frac{1}{2} \sum_{i=1}^n a_i \frac{b \max}{b_i} x_i^2.$$

Changes in the scales of the coefficients c_i according to the presented limiting variants of the function can be represented by the following transition:

$$a_i \frac{b_{\max}}{b_i} \rightarrow a_i b_i.$$

It follows that the first coefficient decreases by a factor of $b_{\max}$, and the last coefficient increases by a factor of $b_{\max}$. The starting point was $x_0 = (100, 100, \dots, 100)$. The stopping criterion was $f - f^* \leq 10^{-10}$.

Table A13 shows the results of the f_{Q^2} function minimization ($n = 1000$) for different values of the learning rate parameter β . A higher convergence rate here is ensured by $\beta = 0.9$ for the HG algorithm and by $\beta = 0.2$ for the HHS algorithm. The results indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms for any value of β compared to the HS method, which resulted in 30,602 iterations and 71,454 gradient and function calculations. Table A14 shows the results of minimization for HG, HHS, HS, and GR algorithms with a value of $\beta = 0.9$. Presented methods outperform the Hestenes–Stiefel conjugate gradient method.

6.6. Raydan1 Function

An offset has been introduced into the Raydan1 function to obtain a new function with a zero minimum value.

$$f_R(x, [amax]) = \sum_{i=1}^n \frac{a_i}{10} (\exp(x_i) - x_i - 1), a_i = a_{\max}^{\frac{i-1}{n-1}}.$$

The starting point was $x_i^0 = 3 * (n - i + 1) / n, i = 1, 2, \dots, n$, the stopping criterion was $f - f^* \leq 10^{-10}$.

In this function, the ravine banks vary significantly in steepness. The behavior of the gradient-based step-adaptive method under these conditions is of interest. Tables A15 and A16 present the results of minimizing the function for various degrees of the isosurfaces' elongation. The results indicate an acceleration of the convergence rate when using diagonal matrices in the algorithms. The advantages of the presented methods over the Hestenes–Stiefel conjugate gradient method are revealed.

For clarity, we present a histogram of the convergence rate of the HG, HHS, and HS methods for different n values and for all test functions (Figures 1–3).

To summarize, the following conclusions can be drawn.

1. On considered test functions, the gradient method with scaling (HG) significantly outperforms the standard steepest descent method and is as fast as the Hestens–Stiefel conjugate gradient method in convergence. These results confirm the acceleration properties of scaling methods.
2. The conjugate gradient method with scaling matrices (HHS) significantly outperforms the HG method on the problems with increasing conditionality, confirming the additional acceleration properties of the conjugate gradient method.
3. The conjugate gradient method with scaling matrices (HHS), despite losing the property of constructing conjugate vectors on quadratic functions, significantly outperforms the Hestens–Stiefel conjugate gradient method on some tests, confirming the additional acceleration properties of the conjugate gradient method.

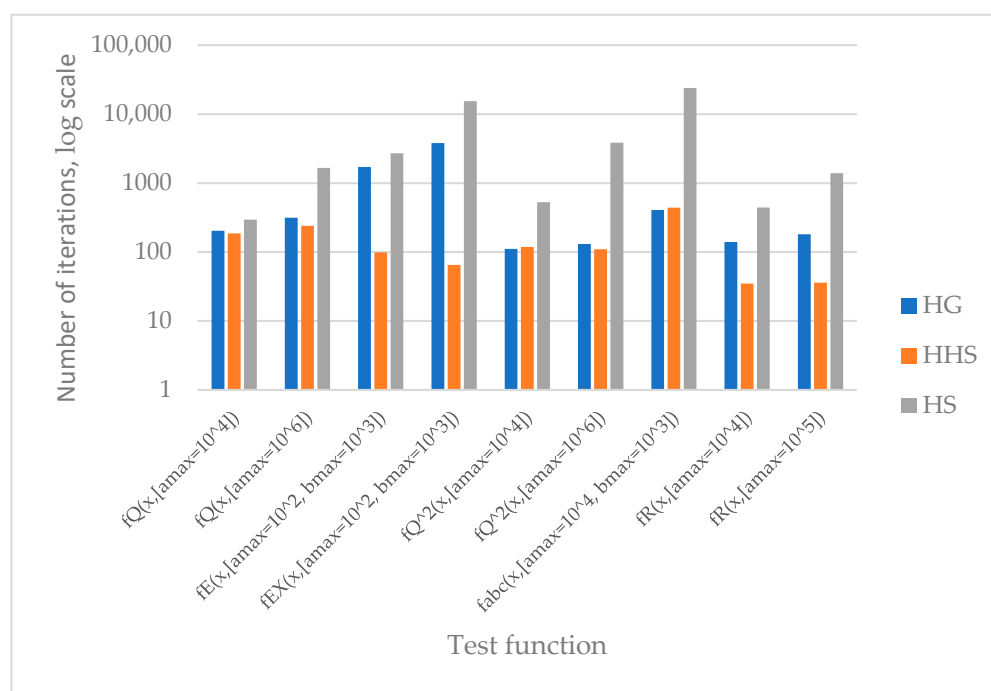


Figure 1. Convergence rate of the HG, HHS, and HS methods for $n = 100$ for all test functions on a logarithmic scale.

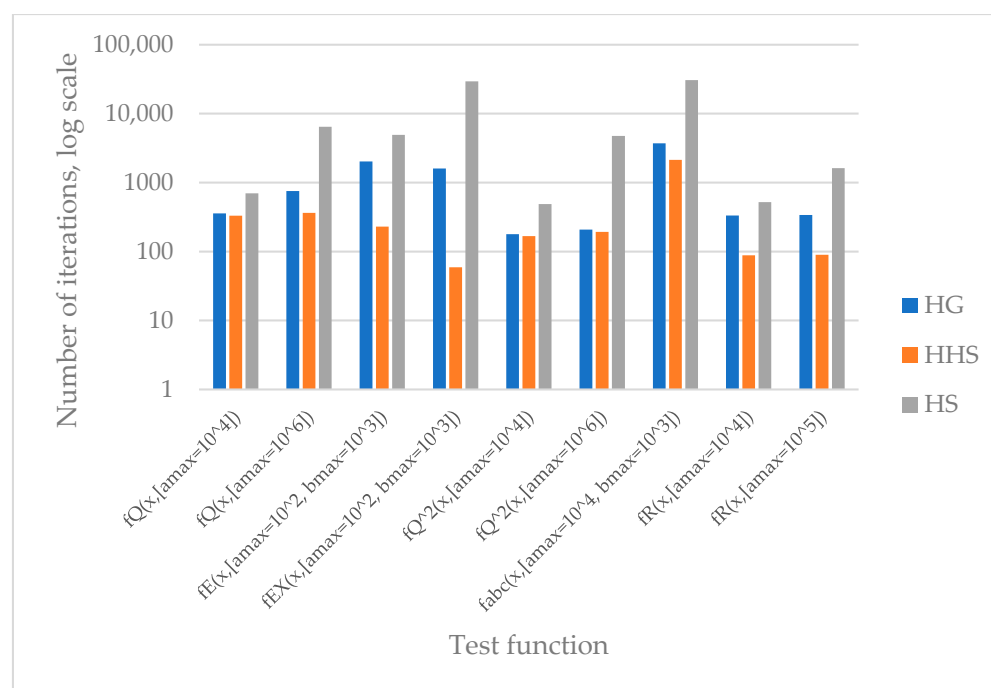


Figure 2. Convergence rate of the HG, HHS, and HS methods for $n = 1000$ for all test functions on a logarithmic scale.

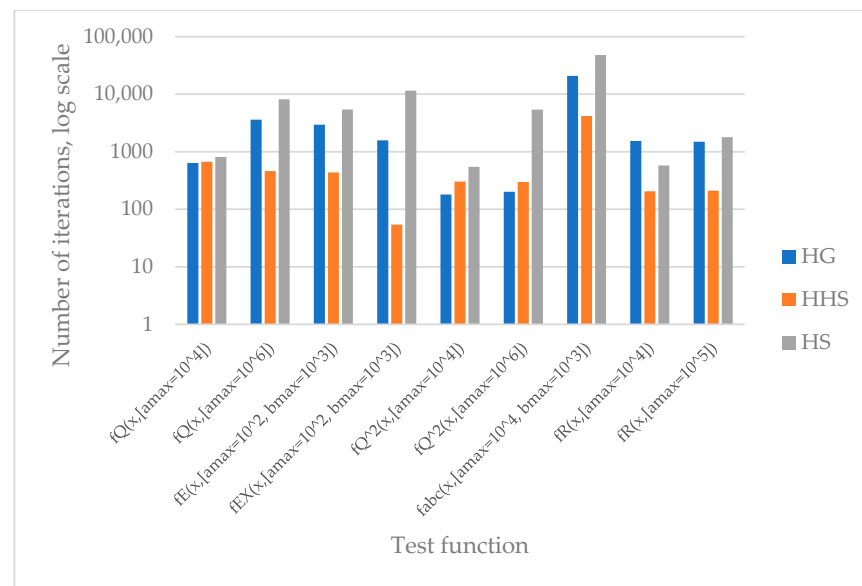


Figure 3. Convergence rate of the HG, HHS, and HS methods for $n = 10,000$ for all test functions on a logarithmic scale.

These results indicate that the proposed methods can be useful under conditions of poor variable scaling. In this case, the problem conditioning can be reduced by using methods with scaling matrices of the HG and HHS metrics.

Nevertheless, in general, a situation is possible where the variables are well scaled, and the causality is high. In this case, the presented methods may not lead to faster convergence, and conjugate gradient methods with scaling will be weaker than the classical conjugate gradient method, since when scaling, they lose the property of constructing conjugate directions on quadratic functions.

7. Conclusions

The conditionality of the minimization problem determines the spread of the isosurfaces' elongation in different directions, which in turn determines the complexity of the problem solution. In minimization practice, it is often possible to reduce the elongation of the isosurfaces through some linear coordinate transformation, thereby increasing the convergence rate of the gradient method used in the new coordinate system. A method for obtaining such a coordinate transformation, which allows for the acceleration of the gradient method and the conjugate gradient method, is proposed in [18]. In this paper, the idea from [18] for organizing an accelerating coordinate transformation is implemented in the form of diagonal scaling matrices. Iterative methods for scaling variables with diagonal metric matrices are proposed and used in the gradient minimization method and the conjugate gradient method. The main idea behind organizing matrix transformations is to implement partial orthogonalization of the descent directions to the previous differences of adjacent gradients, which is literally implemented in conjugate gradient methods for quadratic functions. As a result, a gradient method with a metric change and a diagonal scaling matrix of variables was obtained. This transformation was used to develop the conjugate gradient method. For the gradient method with a diagonal metric matrix, an estimate of the convergence rate was obtained on strongly convex functions with a Lipschitz gradient.

A computational experiment was conducted, and the presented methods were compared with the Hestenes–Stiefel conjugate gradient method. On the given set of test functions, the gradient method with scaling is comparable in convergence rate to the

Hestenes–Stiefel conjugate gradient method, while the conjugate gradient method with a scaling matrix significantly outperforms the Hestenes–Stiefel conjugate gradient method. The obtained results confirm the acceleration properties of scaling methods in the case of poor scaling of the variables of the function being minimized. This allows us to conclude that it is possible to use the methods under study along with conjugate gradient methods to solve smooth, high-dimensional optimization problems with a high degree of conditionality, especially in non-quadratic settings.

This study relies heavily on an exact one-dimensional line search; the behavior of the proposed method under an inexact line search is a subject of our future studies, along with comparing the proposed methods against simple diagonal preconditioners.

Author Contributions: Conceptualization, V.K.; methodology, V.K. and E.T.; software, V.K.; validation, L.K. and E.T.; formal analysis, L.K.; investigation, V.K.; resources, V.K. and L.K.; data curation, E.T.; writing—original draft preparation, V.K.; writing—review and editing, E.T. and L.K.; visualization, V.K. and E.T.; supervision, L.K.; project administration, L.K.; funding acquisition, L.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Ministry of Science and Higher Education of the Russian Federation, project no. FEFE-2023-0004.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

Table A1. Number of iterations for the function $f_Q(x, [a_{max} = 10^4])$, $n = 1000$, for different values of matrix learning rate parameter β .

β	HG		HHS	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
0.1	561	1687	389	1004
0.2	465	1436	367	950
0.3	428	1431	356	917
0.4	408	1430	333	861
0.5	385	1470	327	846
0.6	382	1403	312	815
0.7	367	1347	286	753
0.8	361	1359	333	815
0.9	357	1345	332	810

Table A2. Number of iterations for the function $f_Q(x, [a_{max} = 10^6])$, $n = 1000$, for different values of matrix learning rate parameter β .

β	HG		HHS	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
0.1	3288	17,364	415	1112
0.2	2149	12,128	415	1048
0.3	1646	9727	411	1030
0.4	1350	8203	383	970
0.5	1152	7265	376	953
0.6	1006	6407	360	917
0.7	894	5728	378	947
0.8	816	5312	343	873
0.9	753	4955	363	909

Table A3. Number of iterations for the function $f_Q(x, [a_{max} = 10^4])$, for different values of problem dimension n , $\beta = 0.9$.

n	HG		HHS		HS		GR	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
100	204	421	187	424	296	594	59,683	149,086
1000	357	1345	332	810	702	1405	63,258	158,032
10,000	637	2647	666	1353	810	1624	67,428	179,954

Table A4. Number of iterations for the function $f_Q(x, [a_{max} = 10^6])$, for different values of problem dimension n , $\beta = 0.9$.

n	HG		HHS		HS		GR	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
100	315	1101	240	536	1660	3950	-	-
1000	753	4955	363	909	6438	15,238	-	-
10,000	3605	25,833	463	1574	8127	17,456	-	-

Table A5. Number of iterations for the function $f_E(x, [a_{max} = 10^2, b_{max} = 10^3])$, $n = 1000$, for different values of matrix learning rate parameter β . Starting point $x^0_1 = -1$, $x^0_i = i$, $i = 2, 3, \dots, n$.

β	HG		HHS	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
0.1	7942	19,370	241	766
0.2	220	701	228	727
0.3	230	733	245	782
0.4	220	707	258	820
0.5	215	697	209	679
0.6	221	703	256	815
0.7	216	695	242	778
0.8	217	704	257	820
0.9	212	687	276	880

Table A6. Number of iterations for the function $f_E(x, [a_{max} = 10^2, b_{max} = 10^3])$, for different values of problem dimension n , $\beta = 0.9$. Starting point $x^0_i = -1/(nb_i)^{1/2}$, $i = 1, 2, \dots, n$.

n	HG		HHS		HS		GR	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
100	1710	5387	99	282	2704	6607	510,137	1,276,967
1000	2024	6021	230	605	4895	11,804	352,274	882,316
10,000	2950	7708	436	1156	5418	13,174	307,850	771,223

Table A7. Number of iterations for the function $f_{EX}(x, [a_{max} = 10^2, b_{max} = 10^3])$, $n = 1000$, for different values of matrix learning rate parameter β . Starting point $x^0_1 = -1$, $x^0_i = i$, $i = 2, 3, \dots n$.

β	HG		HHS	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
0.1	401	1071	279	843
0.2	321	988	291	908
0.3	320	985	255	812
0.4	323	1001	306	954
0.5	319	994	285	906
0.6	322	980	284	905
0.7	315	968	279	884
0.8	316	974	277	882
0.9	319	972	288	918

Table A8. Number of iterations for the function $f_{EX}(x, [a_{max} = 10^2, b_{max} = 10^3])$, for different values of problem dimension n , $\beta = 0.9$. Starting point $x^0_i = -1/(nb_i)^{1/2}$, $i = 1, 2, \dots n$.

n	HG		HHS		HS		GR	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
100	3804	9412	65	171	15,490	38,053	1,093,797	2,734,641
1000	1599	4018	59	164	29,296	72,678	207,683	519,438
10,000	1578	4017	54	154	11,544	28,213	213,188	533,209

Table A9. Number of iterations for the function $f_{Q^2}(x, [a_{max} = 10^4])$, $n = 1000$, for different values of matrix learning rate parameter β .

β	HG		HHS	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
0.1	151	456	259	781
0.2	135	408	237	715
0.3	128	387	240	724
0.4	127	384	245	739
0.5	207	624	231	697
0.6	202	609	189	571
0.7	194	585	222	670
0.8	186	561	161	487
0.9	179	540	167	505

Table A10. Number of iterations for the function $f_{Q^2}(x, [a_{max} = 10^4])$, for different values of problem dimension n , $\beta = 0.9$.

n	HG		HHS		HS		GR	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
100	111	338	119	362	529	1079	29,151	63,299
1000	179	540	167	505	490	1000	33,581	71,995
10,000	181	546	302	911	547	1114	10,000	33,007

Table A11. Number of iterations for the function $f_{Q^2}(x, [a_{max} = 10^6])$, $n = 1000$, for different values of matrix learning rate parameter β .

β	HG		HHS	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
0.1	174	525	322	971
0.2	150	453	297	896
0.3	137	414	280	845
0.4	130	393	209	632
0.5	127	384	192	576
0.6	240	723	186	558
0.7	227	684	180	545
0.8	216	651	170	514
0.9	207	624	193	583

Table A12. Number of iterations for the function $f_{Q^2}(x, [a_{max} = 10^6])$, for different values of problem dimension n , $\beta = 0.9$.

n	HG		HHS		HS		GR	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
100	131	398	110	329	3856	9185	-	-
1000	207	624	193	583	4746	10,617	-	-
10,000	202	609	297	895	5402	10,822	-	-

Table A13. Number of iterations for the function $f_{abc}(x, [a_{max} = 10^4, b_{max} = 10^3])$, $n = 1000$, for different values of matrix learning rate parameter β .

β	HG		HHS	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
0.1	27,629	68,511	1224	2804
0.2	15,118	37,430	404	890
0.3	10,118	25,159	1300	2992
0.4	7919	19,613	2403	5637
0.5	6457	15,868	2714	6331
0.6	5313	13,040	1771	4102
0.7	4791	11,745	1479	3393
0.8	4129	10,124	2072	4720
0.9	3700	9079	2123	4744

Table A14. Number of iterations for the function $f_{abc}(x, [a_{max} = 10^4, b_{max} = 10^3])$, for different values of problem dimension n , $\beta = 0.9$.

n	HG		HHS		HS		GR	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
100	409	995	440	969	24,005	58,476	-	-
1000	3700	9079	2123	4744	30,602	71,454	-	-
10,000	20,822	50,115	4173	9968	48,022	105,163	-	-

Table A15. Number of iterations for the function $f_R(x, [a_{max} = 10^4])$, for different values of problem dimension n , $\beta = 0.9$.

n	HG		HHS		HS		GR	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
100	140	286	35	78	443	960		
1000	333	699	88	193	520	1051		
10,000	1535	3088	205	443	577	1184		

Table A16. Number of iterations for the function $f_R(x, [a_{max} = 10^5])$, for different values of problem dimension n , $\beta = 0.9$.

n	HG		HHS		HS		GR	
	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations	Number of Iterations	Number of Gradient and Function Calculations
100	181	375	36	81	1395	2995		
1000	339	756	90	197	1615	3236		
10,000	1498	3352	210	457	1793	3610		

References

- Choi, B.; Park, M. Minimizing total completion time in single-machine scheduling with convex resource consumption and job rejection. *Theor. Comput. Sci.* **2025**, *1064*, 115708. [\[CrossRef\]](#)
- Kis, T.; Szögi, E. Scheduling jobs to minimize a convex function of resource usage. *Comput. Oper. Res.* **2024**, *169*, 106748. [\[CrossRef\]](#)
- Mohan, P.; Rajasekaran, V.A.; Santhanam, P.; Raja, K.; Jayagopal, P.; Kumar, S.; Mallik, S.; Qin, H. TPEMLB: A novel two-phase energy minimized load balancing scheme for WSN data collection with successive convex approximation using mobile sink. *Ain Shams Eng. J.* **2024**, *15*, 102849. [\[CrossRef\]](#)
- Gbadega, P.; Sun, Y. Primal–dual interior-point algorithm for electricity cost minimization in a prosumer-based smart grid environment: A convex optimization approach. *Energy Rep.* **2022**, *8*, 681–695. [\[CrossRef\]](#)
- Prasad, S.; Balasundaram, S. On Lagrangian L2-norm pinball twin bounded support vector machine via unconstrained convex minimization. *Inf. Sci.* **2021**, *571*, 279–302. [\[CrossRef\]](#)
- Zhao, W.; Huang, H. Adaptive stepsize estimation based accelerated gradient descent algorithm for fully complex-valued neural networks. *Expert Syst. Appl.* **2024**, *236*, 121166. [\[CrossRef\]](#)
- Chen, H.; Natsuaki, R.; Hirose, A. Polarization-aware prediction of mobile radio wave propagation based on complex-valued and quaternion neural networks. *IEEE Access* **2022**, *10*, 66589–66600. [\[CrossRef\]](#)
- Bagheri, S.; Konen, W.; Back, T. Solving Optimization Problems with High Conditioning by Means of Online Whitening. In Proceedings of the Genetic and Evolutionary Computation Conference Companion, Prague, Czech Republic, 13–17 July 2019; pp. 243–244. [\[CrossRef\]](#)
- Vaziri, A.; Fang, H. Optimal inferential control of convolutional neural networks. In Proceedings of the 2025 American Control Conference (ACC), Denver, CO, USA, 8–10 July 2025; pp. 2603–2610. [\[CrossRef\]](#)
- Polyak, B.T. *Introduction to Optimization*; Optimization Software: New York, NY, USA, 1987.
- Krutikov, V.; Tovbis, E.; Stanimirović, P.; Kazakovtsev, L. On the Convergence Rate of Quasi-Newton Methods on Strongly Convex Functions with Lipschitz Gradient. *Mathematics* **2023**, *11*, 471. [\[CrossRef\]](#)
- Tovbis, E.; Krutikov, V.; Kazakovtsev, L. Newtonian Property of Subgradient Method with Optimization of Metric Matrix Parameter Correction. *Mathematics* **2024**, *12*, 1618. [\[CrossRef\]](#)
- Indrapriyadarsini, S.; Mahboubi, S.; Ninomiya, H.; Kamio, T.; Asai, H. Accelerating Symmetric Rank-1 Quasi-Newton Method with Nesterov’s Gradient for Training Neural Networks. *Algorithms* **2022**, *15*, 6. [\[CrossRef\]](#)
- Liu, D.C.; Nocedal, J. On the limited memory BFGS method for large scale optimization. *Math. Program.* **1989**, *45*, 503–528. [\[CrossRef\]](#)
- Berahas, A.S.; Jahani, M.; Richtárik, P.; Takác, M. Quasi-Newton Methods for Machine Learning: Forget the Past, Just Sample. *Optim. Methods Softw.* **2022**, *37*, 1668–1704. [\[CrossRef\]](#)

16. Byrd, R.H.; Hansen, S.L.; Nocedal, J.; Singer, Y. A stochastic quasi-Newton method for large-scale optimization. *SIAM J. Optim.* **2016**, *26*, 1008–1031. [[CrossRef](#)]
17. Rodomanov, A.; Nesterov, Y. Greedy quasi-Newton methods with explicit superlinear convergence. *SIAM J. Optim.* **2021**, *31*, 785–811. [[CrossRef](#)]
18. Krutikov, V.; Tovbis, E.; Gutova, S.; Rozhnov, I.; Kazakovtsev, L. Properties and Application of Incomplete Orthogonalization in the Directions of Gradient Difference in Optimization Methods. *Mathematics* **2025**, *13*, 4036. [[CrossRef](#)]
19. Hestenes, M.R.; Stiefel, E. Method of Conjugate Gradients for Solving Linear Systems. *J. Res. Natl. Bur. Stand.* **1952**, *49*, 409–436. [[CrossRef](#)]
20. Sergeyev, Y.D.; Kvasov, D.E.; Mukhametzhanov, M.S. On strong homogeneity of a class of global optimization algorithms working with infinite and infinitesimal scales. *Commun. Nonlinear Sci. Numer. Simul.* **2018**, *59*, 319–330. [[CrossRef](#)]
21. Shor, N. *Minimization Methods for Nondifferentiable Functions*; Springer: Berlin/Heidelberg, Germany, 1985.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.