Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 34(10): 7201–7221.
DOI: 10.3934/era.2026311
Received: 01 June 2026
Revised: 22 July 2026
Accepted: 24 July 2026
Published: 18 August 2026

Research article

Stochastic three-term conjugate gradient: a tensor correction algorithmic framework for nonconvex optimization in machine learning

Jiazhen Liu[1], **Gonglin Yuan**[2] **and Jiahuan Tang**[2,*]

[1] Postdoctoral Research Workstation, Guangxi Rural Commercial United Bank, Nanning 530022, China

[2] School of Mathematics & Center for Applied Mathematics of Guangxi (Guangxi University) & Post-doctoral Research Station of the First-level Discipline in Mathematics, Guangxi University, Nanning 530004, China

* **Correspondence:** Email: jhtang0704@163.com.

Abstract: In large-scale, nonconvex optimization for machine learning, such as the training of spine large models, the balance between efficient high-order geometric features of nonconvex landscapes identification and computational stability remains challenging. This paper proposes a tensor stochastic three-term conjugate gradient (TSTCG) algorithm to address two issues under a stochastic CG framework, the insufficient curvature awareness of the stochastic gradient descent and the heavy reliance on line searches. By introducing an implicit correction mechanism based on third-order tensor information, the algorithm fuses function values and gradient differences to reconstruct evolution vectors, effectively enhancing search precision in ill-conditioned curvature regions. Simultaneously, leveraging a specially designed three-term denominator structure allows the algorithm to process sufficient descent property, line search reliance reduction, and efficient fixed step-size updates. Finally, the nonconvexity proofs and numerical experiments demonstrate that TSTCG outperforms other mainstream stochastic optimization algorithms in terms of convergence efficiency and stability.

Keywords: machine learning; nonconvex optimization; spine large models; tensor; three-term conjugate gradient; sufficient descent

1. Introduction

In contemporary artificial intelligence, the emergence of massive datasets has become a key driver behind the breakthrough progress of modern machine learning models. Consequently, one of the central and intensely debated questions in the field is how to efficiently fit models to exceedingly large training datasets [1, 2]. To ensure strong generalization capabilities for making accurate predictions

on unseen data, the task is mathematically formulated as solving a nonconvex stochastic optimization problem (SOP) over an unknown data distribution:

$$\min_{x\in\mathbb{R}^d} f(x) := \mathbb{E}_{\xi}[f(x,\xi)], \tag{1.1}$$

where x represents the model parameters, ξ is a random variable following the unknown true data distribution, and $f(x,\xi)$ is the loss function associated with the random sample ξ.

Because the true distribution of the entire population is inaccessible in practical applications, researchers widely adopt the empirical risk minimization (ERM) principle as a proxy. The training process of most machine learning models today [3–5] is essentially equivalent to solving the following optimization problem in a finite-sum form:

$$\min_{x\in\mathbb{R}^d} f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \tag{1.2}$$

where n is the sample size, and $f_i(x)$ denotes the loss of the model on the ith data sample. This finite-sum optimization framework (1.2) is widely applied in numerous practical tasks, including image and signal processing, machine learning, and statistical inference [6–9]; typical examples include image registration [10] and principal component analysis (PCA) [11]. However, with the exponential growth in the scale of training datasets, the traditional full gradient descent method faces severe computational bottlenecks. This method requires a full pass over the entire dataset to compute the true deterministic gradient in each iteration, leading to prohibitively high computational costs. As a result, it has become inadequate for the training demands of modern large-scale machine learning.

To overcome the computational bottleneck of full gradient descent in the context of massive data, stochastic gradient descent (SGD) [12, 13] has naturally become the cornerstone algorithm for solving large-scale, nonconvex ERM problems. The standard iteration format of SGD is as follows:

$$x_{k+1} = x_k - \alpha_k g_k,$$

where α_k denotes the step size, and g_k is the stochastic gradient computed at the current iterate x_k. By computing the gradient of only a small subset of samples, SGD significantly reduces the per-iteration computational complexity. However, the primary limitation of SGD is the inherent variance (i.e., gradient noise) between the stochastic gradient g_k and the true deterministic gradient $\nabla f(x_k)$. This variance causes the algorithm to oscillate significantly during the convergence process, typically resulting in a theoretically sublinear convergence rate.

To alleviate this issue and accelerate convergence, a series of variance reduction techniques has been proposed, such as stochastic variance-reduced gradient (SVRG) [14], the stochastic average gradient accelerated method (SAGA) [15], and the stochastic recursive gradient algorithm (SARAH) [16]. These methods gradually eliminate gradient variance by introducing auxiliary variables or reusing historical gradient information, thereby achieving faster convergence rates. On the other hand, addressing the high sensitivity of SGD to step size selection, researchers have developed various adaptive methods, including AdaGrad [17], RMSProp [18], and Adam [19]. These algorithms achieve adaptive learning rate adjustments by accumulating historical gradient observations. Nevertheless, these improved algorithms still face their own limitations: Variance reduction methods often incur high storage overhead or require complex inner-outer loop structures.

Meanwhile, when dealing with complex nonconvex models, adaptive methods are prone to converging to suboptimal solutions [20], although some recent stochastic modified quasi-Newton methods show promising advancements [21].

In the field of deterministic optimization, the conjugate gradient (CG) method has long been regarded as a cornerstone for solving large-scale unconstrained optimization problems [22, 23]. Its prominence stems from its avoidance of computing and storing high-dimensional second-order Hessian matrices, as well as its superior convergence performance compared to the steepest descent method. The standard iterative update format of CG is as follows:

$$x_{k+1} = x_k + \alpha_k d_k,\ d_k = \begin{cases} -g_k, & k = 0, \\ -g_k + \beta_k d_{k-1}, & k \geq 1, \end{cases}$$

where d_k represents the search direction, and β_k is the conjugate parameter. Classical variants for computing β_k include the Fletcher-Reeves (FR) and Polak-Ribière-Polyak (PRP) formulas [24, 25]. Inspired by the low memory overhead of CG methods, early research attempted to introduce them into the machine learning community, proposing several initial stochastic CG (SCG) algorithms [26, 27]. However, the theoretical framework of traditional CG is built upon exact deterministic gradients. When the exact gradients are replaced by stochastic gradients g_k contaminated with random noise, the algorithm faces severe challenges. Stochastic noise significantly interferes with the calculation of the classical β_k formulas, which not only easily destroys the conjugacy of the search direction but may also lead to the loss of the descent property (i.e., failing to satisfy $d_k^\top g_k < 0$), eventually causing the algorithm to diverge. To overcome this predicament, existing SCG methods, such as conjugate gradient with variance reduction (CGVR) [28] and a mini-batch stochastic conjugate gradient algorithm with variance reduction (SCGA) [29], typically rely heavily on exact line search or strong Wolfe line search in both theoretical derivations and practical implementations to enforce the descent property at each iteration. However, for large-scale stochastic optimization tasks involving massive datasets, frequent evaluation of line search conditions incurs prohibitively high computational costs. This fatal computational bottleneck significantly restricts the practical potential of existing SCG methods for modern complex models such as deep learning.

Beyond the stability of the search direction, accurately capturing the high-order geometric features of complex loss surfaces is fundamental to the design of robust optimization algorithms. Traditional Newton and quasi-Newton methods primarily rely on the second-order Taylor expansion, which assumes that the local landscape is quadratic, a condition that is rarely met in deep neural networks [30]. In such highly nonconvex models, the loss function exhibits significant nonquadratic behavior where relying solely on the second-order model $\phi(x) = f_k + g_k^\top s + (1/2)s^\top \nabla^2 f_k s$ fails to characterize subtle curvature variations. Consequently, the classical secant equation $B_{k+1}s_k = y_k$ cannot accurately reflect the instantaneous rate of change of the Hessian matrix. To address this, high-order tensor methods have emerged as a powerful alternative, leveraging a third-order term Θ_k to provide a more precise local approximation [31, 32]:

$$\phi_\Theta(x_{k-1}) \approx f_k - g_k^\top s_k + \frac{1}{2}s_k^\top \nabla^2 f_k s_k - \frac{1}{6}\Theta_k s_k^3.$$

Although explicit tensor computation is prohibitive, recent breakthroughs suggest that these high-order features can be implicitly embedded into the descent direction by constructing a modified

secant relation using function value residuals and gradient differences $y_k = g_{k+1} - g_k$ [33]. This approach allows the algorithm to perceive the third-order geometry of the nonconvex surface at a negligible computational cost.

On the other hand, to address the over-reliance of SCG on line search techniques, the three-term CG method has been introduced [34, 35]. Its typical iterative update formula is defined as follows:

$$d_k = -g_k + \beta_k d_{k-1} - \eta_k y_k,$$

where $\beta_k = g_k^\top y_k / \|g_k\|^2$ and $\eta_k = g_k^\top d_{k-1} / \|g_k\|^2$ are conjugate parameters that determine the search direction. A key advantage of this method is that it theoretically guarantees the sufficient descent property of the search direction without requiring any line search techniques. This significantly reduces the computational burden and enhances search efficiency when dealing with large-scale, complex problems. If a new algorithm could be developed in a stochastic setting that integrates tensor curvature information with the favorable descent properties of three-term CG, thereby completely eliminating the need for line search and allowing for the use of fixed step sizes, it would significantly advance the efficiency of solving large-scale nonconvex optimization problems.

In summary, although stochastic CG methods possess significant potential, existing algorithms are often limited by complex line search procedures or expensive variance-reduction frameworks. To address these challenges, this paper proposes a lightweight and efficient stochastic three-term CG (TSTCG) algorithm specifically designed for large-scale nonconvex optimization in machine learning. The primary contributions of this work are as follows:

- Unlike traditional tensor algorithms, the proposed algorithm does not require the explicit computation of complex tensor matrices. Instead, by introducing a modified term H_k that incorporates function values and gradients, it skillfully transforms the traditional gradient difference y_k into Y_k, which carries high-order curvature information. This mechanism significantly enhances the accuracy of the search direction within complex nonconvex landscapes at minimal computational cost.
- A specialized denominator structure, regulated by a set of parameters $\{\gamma_1, \gamma_2, \gamma_3\}$, is introduced into the conjugate parameters β_k and η_k. This design theoretically guarantees the sufficient descent property of the search direction, which effectively mitigates the interference of stochastic gradient noise and allows the algorithm to remain robust in highly volatile stochastic environments.
- The proposed method completely eliminates the traditional reliance of conjugate gradient methods on Armijo or Wolfe line searches. In practice, the proposed algorithm achieves efficient updates using only mini-batch sampling and fixed step sizes, which drastically reduces the time complexity per iteration and simplifies the overall implementation of the algorithm.
- Without requiring strong convexity assumptions, we establish a rigorous convergence proof for TSTCG in nonconvex settings. Furthermore, extensive experimental results across multiple nonconvex machine learning models demonstrate that TSTCG outperforms the traditional SGD algorithm in terms of convergence speed, generalization ability, and robustness, while exhibiting strong competitiveness compared to current mainstream adaptive algorithms.

Beyond its immediate applications in training conventional machine learning models, the robustness and efficiency of the proposed TSTCG algorithm make it a promising candidate for tackling complex optimization problems in other high-stakes domains. For instance, the strategic

management of a commercial bank's business portfolio presents precisely the type of large-scale, nonconvex challenge that TSTCG is designed to address. Commercial banks maintain a diversified operational structure consisting of corporate banking, retail banking, bill financing, and fixed-income. Amid volatile macroeconomic conditions, classical mean-variance optimization is insufficient, as it ignores asymmetric market shocks. Modern bank asset allocation must incorporate higher-order risk moments (such as skewness), transforming the decision into a preliminary nonconvex stochastic optimization formulation:

$$\min_{x\in\mathcal{X}} F(x) = -\mathbb{E}[R^\top x] + \frac{\lambda_1}{2}x^\top \Sigma x - \frac{\lambda_2}{6}\mathcal{S}(x,x,x) + \Phi(x), \tag{1.3}$$

where $x \in \mathbb{R}^d$ is the asset allocation vector, R is the expected return, Σ is the covariance matrix, $\mathcal{S}$ represents the co-skewness tensor (capturing third-order asymmetric risks), and $\Phi(x)$ denotes nonconvex regulatory penalty functions. Traditional optimization approaches struggle to handle this formulation because modeling asymmetric risk inherently requires computing third-order tensor derivatives ($\nabla^3 F(x)$), leading to prohibitive computational costs and unstable convergence. In recognition of these limitations, the TSTCG algorithm is uniquely suited for such economic optimization. By implicitly encapsulating the third-order curvature information via the hybrid compensator H_k, TSTCG can efficiently navigate the flat and ill-conditioned regions generated by the co-skewness tensor without explicit tensor decomposition. Therefore, TSTCG could be exploited in the future to capture the complex nonlinear interactions within the diversified operational structure of commercial banking, providing a powerful mathematical tool to enhance the precision and reliability of bank asset allocation strategies.

The rest of the article is organized as follows: Section 2 thoroughly details the proposed algorithmic framework and its motivation. Section 3 provides a theoretical convergence analysis for nonconvex settings. Section 4 presents extensive numerical experiments on machine learning datasets. Finally, Section 5 concludes the paper.

2. The proposed algorithmic framework

The design motivation of Algorithm 1 originates from an in-depth exploration of the inherent tension between geometric precision and computational stability in nonconvex stochastic optimization. To establish a precise theoretical link between the modified curvature compensator and high-order tensor information, we assume that the objective function $f(x)$ is three-times continuously differentiable.

According to the third-order Taylor expansion and the error analysis of the trapezoidal rule, the exact mathematical relation connecting the function values, gradient differences, and the third-order tensor $\Theta_k = \nabla^3 f(x_k)$ along the search direction $s_k = x_k - x_{k-1}$ is strictly given by:

$$f(x_{k-1}) - f(x_k) + \frac{1}{2}(g_{k-1} + g_k)^\top s_k = \frac{1}{12}\Theta_k s_k^3 + o(\|s_k\|^3).$$

Although incorporating this exact pure tensor term ($(1/12)\Theta_k s_k^3$) is common in deterministic settings, in stochastic optimization, the term s_k^3 is typically exceedingly small and highly susceptible to being overwhelmed by stochastic gradient noise. To prevent the high-order curvature signal from vanishing,

we heuristically drop the coefficient $1/2$ in the algorithmic design, introducing a hybrid curvature compensator H_k:

$$H_k = f(x_{k-1}) - f(x_k) + (g_{k-1} + g_k)^\top s_k. \tag{2.1}$$

Mathematically, by decomposing H_k, its exact relationship with the tensor Θ_k becomes explicit:

$$H_k = \underbrace{\left[f(x_{k-1}) - f(x_k) + \frac{1}{2}(g_{k-1} + g_k)^\top s_k \right]}_{\approx \frac{1}{12}\Theta_k s_k^3} + \frac{1}{2}(g_{k-1} + g_k)^\top s_k.$$

$$\Rightarrow \quad H_k = \frac{1}{12}\Theta_k s_k^3 + \frac{1}{2}(g_{k-1} + g_k)^\top s_k + o(\|s_k\|^3).$$

This exact decomposition reveals that H_k acts as a powerful hybrid signal. It successfully captures the third-order nonconvex geometric features $(1/12)\Theta_k s_k^3$ while maintaining a robust baseline of aggregated secant momentum $(1/2)(g_{k-1} + g_k)^\top s_k$, thereby ensuring computational stability.

By incorporating this tensor-informed compensator into the traditional gradient difference $y_k = g_k - g_{k-1}$, we derive a modified evolution direction Y_k:

$$Y_k = y_k + \frac{H_k}{\|s_k\|^2} s_k. \tag{2.2}$$

The central motivation for this construction is to force the search direction d_k to perceive Hessian derivative information via Y_k without the explicit computation of third-order tensors. This enables the algorithm to generate more accurate descent paths in regions with ill-conditioned curvature or flat nonconvex landscapes, effectively addressing the limitations of traditional SCG algorithms, which often exhibit an insensitivity to curvature when navigating complex nonconvex terrains.

Simultaneously, to overcome the vulnerabilities of traditional stochastic CGVR methods, specifically their susceptibility to noise and high dependence on line search, the TSTCG algorithm adopts a more stable three-term CG structure:

$$d_k = \begin{cases} -g_k + \beta_k d_{k-1} - \eta_k Y_k, & \text{if } k \geq 1, \\ -\nabla f(x_k), & \text{if } k = 0. \end{cases} \tag{2.3}$$

The core design motivation lies in providing real-time compensation for the search direction through the additional third term $-\eta_k Y_k$. To ensure numerical robustness in noisy stochastic environments, we designed a specialized combined denominator structure for the conjugate parameters β_k and η_k:

$$\beta_k = \frac{Y_k^\top g_k}{\gamma_1 \|Y_k\| \|d_{k-1}\| + \gamma_2 \|Y_k\| \|g_k\| + \gamma_3 \|g_{k-1}\|^2}, \tag{2.4}$$

and

$$\eta_k = \frac{d_{k-1}^\top g_k}{\gamma_1 \|Y_k\| \|d_{k-1}\| + \gamma_2 \|Y_k\| \|g_k\| + \gamma_3 \|g_{k-1}\|^2}. \tag{2.5}$$

The fundamental motivation behind this specialized denominator structural design in (2.4) and (2.5) is to mathematically enforce robustness against stochastic gradient variance. By incorporating Cauchy-Schwarz compatible terms ($\|Y_k\| \|d_{k-1}\|$ and $\|Y_k\| \|g_k\|$) scaled by positive tuning parameters $\gamma_1, \gamma_2, \gamma_3$, we

strictly bound the magnitude of the conjugate parameters. This guarantees that the denominator always dominates the numerator, ensuring that the term $\beta_k d_{k-1} - \eta_k Y_k$ acts as a stable curvature corrector without overpowering the negative gradient direction $-g_k$. Consequently, this elegantly translates into the unconditional sufficient descent property ($g_k^\top d_k = -\|g_k\|^2 < 0$), completely circumventing the need for heuristic or computationally expensive line search steps. Through the synergy of implicit tensor approximation and this three-term structure without line search, TSTCG effectively addresses the challenges of gradient noise and complex curvature in large-scale, nonconvex training without adding a significant computational burden, thereby achieving an ideal balance between computational efficiency and optimization precision.

In summary, we can present the procedure of the proposed algorithmic framework as shown in Algorithm 1.

Algorithm 1 TSTCG

Input: Given $x_1 \in \mathbb{R}^d$, stepsize α, constants $[\gamma_1, \gamma_2, \gamma_3] \in (0, 1)^3$, and mini-batch size $|\Omega_k| = \sqrt{n}$.
Output: x_{k+1}

1: Initialization: Set $k = 0$, compute the full gradient descent direction $g_0 = \nabla f(x_0)$;
2: Compute the initial point $x_1 = x_0 - \alpha g_0$;
3: **for** $k = 1, 2, \ldots, N$ **do**
4: Randomly select a mini-batch Ω_k;
5: Compute the mini-batch gradient $g_k = \frac{1}{\Omega_k} \sum_{i=1}^{\Omega_k} \nabla f(x_k; \xi_{k,i})$;
6: Calculate the search direction d_k via (2.3);
7: Update the next iterate $x_{k+1} = x_k + \alpha d_k$.
8: **end for**

3. Convergence

In this section, we present the convergence analysis of the proposed TSTCG algorithm. Following the conventions of existing stochastic CG methods, we adopt the following standard assumptions:

Assumption 1. *The stochastic gradient function $\nabla f(x, \xi)$ of a sample is Lipschitz continuous. That is, there exists a constant L such that for all $x_1, x_2 \in \mathbb{R}^n$, the following inequality holds:*

$$\|\nabla f(x_1, \xi) - \nabla f(x_2, \xi)\| \leq L\|x_1 - x_2\|.$$

Remark 1. *If the stochastic gradient is Lipschitz continuous, then the full gradient is also Lipschitz continuous:*

$$\begin{aligned} \|\nabla f(x_1) - \nabla f(x_2)\| &= \|\mathbb{E}[\nabla f(x_1, \xi) - \nabla f(x_2, \xi)]\| \\ &\leq \mathbb{E}[\|\nabla f(x_1, \xi) - \nabla f(x_2, \xi)\|] \\ &\leq L\|x_1 - x_2\|. \end{aligned}$$

Assumption 2. *The objective function f satisfies the Polyak-Lojasiewicz (PL) condition. That is, there exists a constant η such that for all $x \in \mathbb{R}^n$,*

$$\|\nabla f(x)\|^2 \geq 2\eta(f(x) - f(x^*)),$$

where $x^* = \arg\min_x f(x)$.

Remark 2. *It is important to distinguish the PL condition from standard strong convexity. Although strong convexity strictly implies the PL condition, the reverse is not true. The PL condition does not require the function to be convex; it permits the existence of multiple global minima and flat landscapes, provided that every stationary point is a global minimum. This makes it a highly relevant and realistic assumption for analyzing the linear convergence of optimization trajectories in certain nonconvex machine learning landscapes (e.g., highly parameterized neural networks).*

Assumption 3. *The stochastic gradient g_k is an unbiased estimate of the full gradient $\nabla f(x_k)$, and its variance is bounded. That is, there exists a constant σ such that:*

$$\mathbb{E}[\|g_k - \nabla f(x_k)\|^2] \leq \sigma^2.$$

By the variance decomposition property, this is equivalent to:

$$\begin{aligned} \mathbb{E}[\|g_k\|^2] &= \|\nabla f(x_k)\|^2 + \mathbb{E}[\|g_k - \nabla f(x_k)\|^2] \\ &\leq \|\nabla f(x_k)\|^2 + \sigma^2. \end{aligned} \tag{3.1}$$

Lemma 1. *For the sequence $\{x_k\}$ generated by Algorithm 1, the search direction and the stochastic gradient satisfy the following inequality:*

$$g_k^\top d_k = -\|g_k\|^2, \tag{3.2}$$

and

$$\|d_k\| \leq (1 + \frac{2}{\gamma_1})\|g_k\|. \tag{3.3}$$

Proof. When $k = 0$, Eq (3.2) obviously holds. For $k \neq 0$, we multiply the search direction d_k by g_k on both sides:

$$\begin{aligned} g_k^\top d_k &= g_k^\top [-g_k + \frac{Y_k^\top g_k d_{k-1} - d_{k-1}^\top g_k Y_k}{\gamma_1\|Y_k\|\|d_{k-1}\| + \gamma_2\|Y_k\|\|g_k\| + \gamma_3\|g_{k-1}\|^2}] \\ &= -\|g_k\|^2 + \frac{Y_k^\top g_k d_{k-1}^\top g_k - d_{k-1}^\top g_k Y_k^\top g_k}{\gamma_1\|Y_k\|\|d_{k-1}\| + \gamma_2\|Y_k\|\|g_k\| + \gamma_3\|g_{k-1}\|^2} \\ &= -\|g_k\|^2. \end{aligned}$$

From the inequality $-\|g_k\|^2 = g_k^\top d_k \geq -\|g_k\|\|d_k\|$, it follows that $\|g_k\| \leq \|d_k\|$. Furthermore, taking the norm of both sides of the equation defining the search direction d_k yields:

$$\begin{aligned} \|d_k\| &= \| - g_k + \frac{Y_k^\top g_k d_{k-1} - d_{k-1}^\top g_k Y_k}{\gamma_1\|Y_k\|\|d_{k-1}\| + \gamma_2\|Y_k\|\|g_k\| + \gamma_3\|g_{k-1}\|^2}\| \\ &\leq \|g_k\| + \frac{2\|Y_k\|\|g_k\|\|d_{k-1}\|}{\gamma_1\|Y_k\|\|d_{k-1}\| + \gamma_2\|Y_k\|\|g_k\| + \gamma_3\|g_{k-1}\|^2} \\ &\leq (1 + \frac{2}{\gamma_1})\|g_k\|. \end{aligned}$$

Thus, we have proved that the search direction d_k satisfies the descent property and the trust-region property. □

Remark 3. *It is crucial to emphasize that the exact sufficient descent property ($g_k^\top d_k = -\|g_k\|^2$) established in Lemma 1 strictly relies on the algorithmic design choice outlined in Algorithm 1. Specifically, at the kth iteration, a single, identical mini-batch Ω_k is evaluated to compute the stochastic gradient g_k. This exact same vector g_k is then consistently reused to compute the numerator and denominator of the parameters β_k and η_k, as well as the final search direction d_k. If different mini-batches were to be used for the parameters and the descent direction (as seen in some decoupled stochastic schemes), this perfect algebraic cancellation would fail. Therefore, sharing the identical mini-batch within the same iteration is a fundamental requirement to theoretically guarantee the sufficient descent property in noisy environments.*

Theorem 1. *Suppose that Assumptions 1, 2, and 3 hold, and let $\{x_k\}$ be the sequence generated by Algorithm 1, where $x^* = \arg\min_x f(x)$. If $\alpha \le 1/(2LC_1^2)$, then we have:*

$$\mathbb{E}[f(x_K) - f(x^*)] \le \mu^K \mathbb{E}[f(x_0) - f(x^*)] + \frac{4L\sigma^2 C_1^4}{\eta}\alpha, \tag{3.4}$$

where $\mu = 1 - (\alpha\eta/2)$, and $C_1 = 1 + 2/\gamma_1$.

Proof. From the L-smoothness property, we get

$$f(x_{k+1}) \le f(x_k) + \nabla f(x_k)^\top(\alpha d_k) + \frac{L}{2}\|\alpha d_k\|^2.$$

Taking the conditional expectation on both sides of the above inequality with respect to the random sample Ω_k at the current iteration,

$$\mathbb{E}_{\Omega_k}[f(x_{k+1})] \le f(x_k) + \alpha\mathbb{E}_{\Omega_k}[\nabla f(x_k)^\top d_k] + \frac{L\alpha^2}{2}\mathbb{E}_{\Omega_k}[\|d_k\|^2].$$

Let $e_k = g_k - \nabla f(x_k)$. From Lemma 1, we have

$$\begin{aligned}\nabla f(x_k)^\top d_k &= (g_k - e_k)^\top d_k \\ &= -\|g_k\|^2 - e_k^\top d_k.\end{aligned}$$

Applying the Cauchy-Schwarz inequality and Young's inequality along with the bound $\|d_k\| \le C_1\|g_k\|$, we get

$$-e_k^\top d_k \le \|e_k\|\|d_k\| \le C_1\|e_k\|\|g_k\| \le \frac{1}{2}\|g_k\|^2 + \frac{C_1^2}{2}\|e_k\|^2.$$

By $\mathbb{E}[\|e_k\|^2] \le \sigma^2$, we obtain

$$\begin{aligned}\mathbb{E}_{\Omega_k}[\nabla f(x_k)^\top d_k] &= \mathbb{E}_{\Omega_k}[-\|g_k\|^2 - e_k^\top d_k] \\ &\le \mathbb{E}_{\Omega_k}[-\|g_k\|^2 + \frac{1}{2}\|g_k\|^2 + \frac{C_1^2}{2}\|e_k\|^2] \\ &= -\frac{1}{2}\mathbb{E}_{\Omega_k}[\|g_k\|^2] + \frac{C_1^2\sigma^2}{2}.\end{aligned}$$

By (3.1), we have $\mathbb{E}[\|g_k\|^2] \ge \|\nabla f(x_k)\|^2$, and thus,

$$\mathbb{E}_{\Omega_k}[\nabla f(x_k)^\top d_k] \le -\frac{1}{2}\|\nabla f(x_k)\|^2 + \frac{C_1^2\sigma^2}{2}.$$

Furthermore, from Lemma 1 and (3.1), we have

$$\begin{aligned}\frac{L\alpha^2}{2}\mathbb{E}_{\Omega_k}[\|d_k\|^2] &\leq \frac{L\alpha^2 C_1^2}{2}\mathbb{E}_{\Omega_k}[\|g_k\|^2] \\ &= \frac{L\alpha^2 C_1^2}{2}(\|\nabla f(x_k)\|^2 + \sigma^2).\end{aligned}$$

Then, we obtain

$$\begin{aligned}\mathbb{E}_{\Omega_k}[f(x_{k+1})] &\leq f(x_k) + \alpha(-\frac{1}{2}\|\nabla f(x_k)\|^2 + \frac{C_1^2\sigma^2}{2}) + \frac{L\alpha^2 C_1^2}{2}(\|\nabla f(x_k)\|^2 + \sigma^2) \\ &= f(x_k) - \frac{\alpha(1 - L\alpha^2 C_1^2)}{2}\|\nabla f(x_k)\|^2 + (\frac{\alpha C_1^2}{2} + \frac{L\alpha^2 C_1^2}{2})\sigma^2.\end{aligned}$$

Because $\alpha \leq 1/(2LC_1^2)$, we have $1 - L\alpha C_1^2 \geq 1/2$. Applying the PL condition further yields

$$\mathbb{E}_{\Omega_k}[f(x_{k+1})] \leq f(x_k) - \frac{\alpha\eta}{2}(f(x_k) - f(x^*)) + \alpha C_1^2\sigma^2.$$

Subtracting $f(x^*)$ from both sides of the above inequality and taking the total expectation, while denoting $\Delta_k = \mathbb{E}[f(x_k) - f(x^*)]$, we obtain

$$\Delta_{k+1} \leq (1 - \frac{\alpha\eta}{2})\Delta_k + \alpha C_1^2\sigma^2.$$

By recursively applying the above inequality from $k = 0, 1, \ldots, K$, we obtain the following global bound for the Kth iteration:

$$\Delta_K \leq (1 - \frac{\alpha\eta}{2})^K\Delta_0 + \frac{4L\sigma^2 C_1^4}{\eta}\alpha.$$

□

Corollary 1. *(Complexity analysis) According to Theorem 1, to ensure that the expected error of Algorithm 1 achieves a given precision ϵ, i.e., $\mathbb{E}[f(x_k) - f(x^*)] \leq \epsilon$, the learning rate α should be proportional to ϵ such that $\alpha = \Theta(\epsilon)$. Under this condition, the total number of iterations K required satisfies $K = O((1/\epsilon)\log(1/\epsilon))$.*

Theorem 2. *(Standard nonconvex convergence) Suppose Assumptions 1 and 3 hold. Let $\{x_k\}$ be the sequence generated by Algorithm 1 with the step size $\alpha \leq 1/(2LC_1^2)$. Then, we have*

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(x_k)\|^2] \leq \frac{4(f(x_0) - f(x^*))}{\alpha K} + 2C_1^2\sigma^2(1 + L\alpha). \tag{3.5}$$

Proof. Following the derivation in the proof of Theorem 1, specifically the step prior to applying the PL condition, taking the total expectation on both sides yields:

$$\mathbb{E}[f(x_{k+1})] \leq \mathbb{E}[f(x_k)] - \frac{\alpha(1 - L\alpha C_1^2)}{2}\mathbb{E}[\|\nabla f(x_k)\|^2] + \left(\frac{\alpha C_1^2}{2} + \frac{L\alpha^2 C_1^2}{2}\right)\sigma^2.$$

Because $\alpha \leq 1/(2LC_1^2)$, we have $1 - L\alpha C_1^2 \geq 1/2$. This yields

$$\mathbb{E}[f(x_{k+1})] \leq \mathbb{E}[f(x_k)] - \frac{\alpha}{4}\mathbb{E}[\|\nabla f(x_k)\|^2] + \frac{\alpha C_1^2 \sigma^2}{2}(1 + L\alpha).$$

Rearranging the terms, we obtain:

$$\frac{\alpha}{4}\mathbb{E}[\|\nabla f(x_k)\|^2] \leq \mathbb{E}[f(x_k) - f(x_{k+1})] + \frac{\alpha C_1^2 \sigma^2}{2}(1 + L\alpha).$$

Summing this inequality from $k = 0$ to $K - 1$, and dividing both sides by $\alpha K/4$, we obtain:

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(x_k)\|^2] \leq \frac{4(f(x_0) - \mathbb{E}[f(x_K)])}{\alpha K} + 2C_1^2\sigma^2(1 + L\alpha).$$

Because $f(x_K) \geq f(x^*)$ for all K, we have $f(x_0) - \mathbb{E}[f(x_K)] \leq f(x_0) - f(x^*)$, which completes the proof. □

Remark 4. *Theorem 2 demonstrates that without the PL condition, TSTCG converges to a neighborhood of the stationary point. Specifically, if we choose a properly diminishing step size sequence (e.g., $\alpha = O(1/\sqrt{K})$) or a sufficiently small constant α, taking the limit as $K \to \infty$ ensures that $\liminf_{k\to\infty}\mathbb{E}[\|\nabla f(x_k)\|] = 0$. This confirms that the proposed algorithm theoretically applies to general standard nonconvex settings (such as deep neural networks) beyond PL-conditioned landscapes.*

4. Numerical experiments

In this section, we conduct numerical experiments to verify the practical performance of the proposed TSTCG algorithm. To provide a comprehensive evaluation, we apply the algorithm to four representative machine learning models covering both nonconvex and convex settings and then benchmark it against current mainstream stochastic optimization algorithms.

4.1. Experimental settings and datasets

All experiments were conducted on a personal computer equipped with a 12th Gen Intel(R) Core(TM) i5-12500H 2.50 GHz processor and 16 GB of RAM, running MATLAB 2023a. We selected ten real-world datasets from the UCI Machine Learning Repository [36] and the LIBSVM Repository [37], covering a diverse range of feature dimensions and sample sizes (see Table 1). The sample sizes vary from 252 to 581,012, and the feature dimensions range from 8 to 300. During the preprocessing phase, all datasets were mapped to the interval $[-1, 1]$ via max-min scaling.

4.2. Test models

To evaluate the algorithm comprehensively, we selected four standard machine learning models. Models 1 and 2 represent nonconvex optimization tasks, and Models 3 and 4 serve as fundamental convex benchmarks:

Table 1. The information of datasets.

Dataset	Training samples	Dimension	Source
Adult	32,562	123	UCI [36]
W8a	49,749	300	LIBSVM [37]
Covtype	581,012	54	UCI [36]
diabetes	768	8	UCI [36]
mushrooms	8124	112	UCI [36]
bodyfat	252	18	UCI [36]
a9a	32,561	123	LIBSVM [37]
inosphere	351	34	UCI [36]
ijcnn	49,990	22	LIBSVM [37]
german.number	1000	24	UCI [36]

- Nonconvex SVM:

$$f(x) = \frac{1}{n}\sum_{i=1}^{n}[1 - \tanh(b_i\langle x, a_i\rangle)] + \lambda\|x\|^2.$$

- Nonconvex regularized ERM (sigmoid):

$$f(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{1+\exp(b_i a_i^\top x)} + \frac{\lambda}{2}\|x\|^2.$$

- Convex logistic regression:

$$f(x) = \frac{1}{n}\sum_{i=1}^{n}\log(1+\exp(-b_i a_i^\top x)) + \frac{\lambda}{2}\|x\|^2.$$

- Convex ridge regression:

$$f(x) = \frac{1}{n}\sum_{i=1}^{n}(b_i - a_i^\top x)^2 + \lambda\|x\|^2.$$

In all formulations, $a_i \in \mathbb{R}^d$ denotes the feature vector, b_i is the label, and λ is the regularization coefficient.

4.3. Algorithm comparison and parameter configuration

We benchmarked TSTCG against three mainstream stochastic optimization algorithms: SGD, SVRG, and Adam. To evaluate the intrinsic stability of each algorithm without the influence of complex dynamic step-size strategies, a fixed step-size policy was consistently employed across all methods. In the comparison experiments, the mini-batch size was specified as $|\Omega_k| = \lfloor\sqrt{n}\rfloor$. This specific choice is motivated by classical variance-reduction literature (e.g., SVRG), where $O(\sqrt{n})$ serves as an empirical optimum that heuristically balances the trade-off between suppressing the gradient variance σ^2 (which theoretically accommodates a larger stable step size) and maintaining a low per-iteration computational cost, without requiring exhaustive dataset-specific tuning. The regularization coefficient was set to $\lambda = 10^{-5}$ for all models.

Although theoretical analysis provides an upper bound for the step size $\alpha \leq 1/(2LC_1^2)$, the global Lipschitz constant L is generally unknown in practice. Therefore, following standard empirical practices in machine learning, we determined an appropriate fixed step size through a grid search strategy over the set $\{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$ for the proposed TSTCG. Specifically, for Adam, the exponential decay rates for the moment estimates were kept at their widely adopted default values ($\beta_1 = 0.9, \beta_2 = 0.999$). For SVRG, which is notoriously sensitive to its hyperparameter configurations, the step size was identically tuned via the grid search, and its inner epoch length was systematically set to $m = \lfloor n/|\Omega_k| \rfloor$. This configuration ensures that the inner loop performs exactly one effective full data pass per outer iteration, which is the standard optimal heuristic recommended in variance-reduction literature to perfectly balance variance elimination and computational overhead. Regarding the fixed learning rate, it is set to 10^{-1} for Models 1 and 2, but adjusted to 10^{-3} for Models 3 and 4. This adjustment is necessary because the convex loss functions in Models 3 and 4 (involving logarithmic exponential terms and quadratic terms) typically generate much larger gradient magnitudes during the initial iterations compared to the nonconvex Sigmoid-type losses. A smaller learning rate is therefore required to prevent overshooting and ensure numerical stability. For the baseline methods, Adam's hyperparameters were set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For the proposed TSTCG algorithm, the control parameters were configured as $\gamma_1 = 0.09$ and $\gamma_2 = \gamma_3 = 0.01$. All methods wereinitialized with a zero vector and updated following the same mini-batch samplingsequence to ensure a fair comparison.

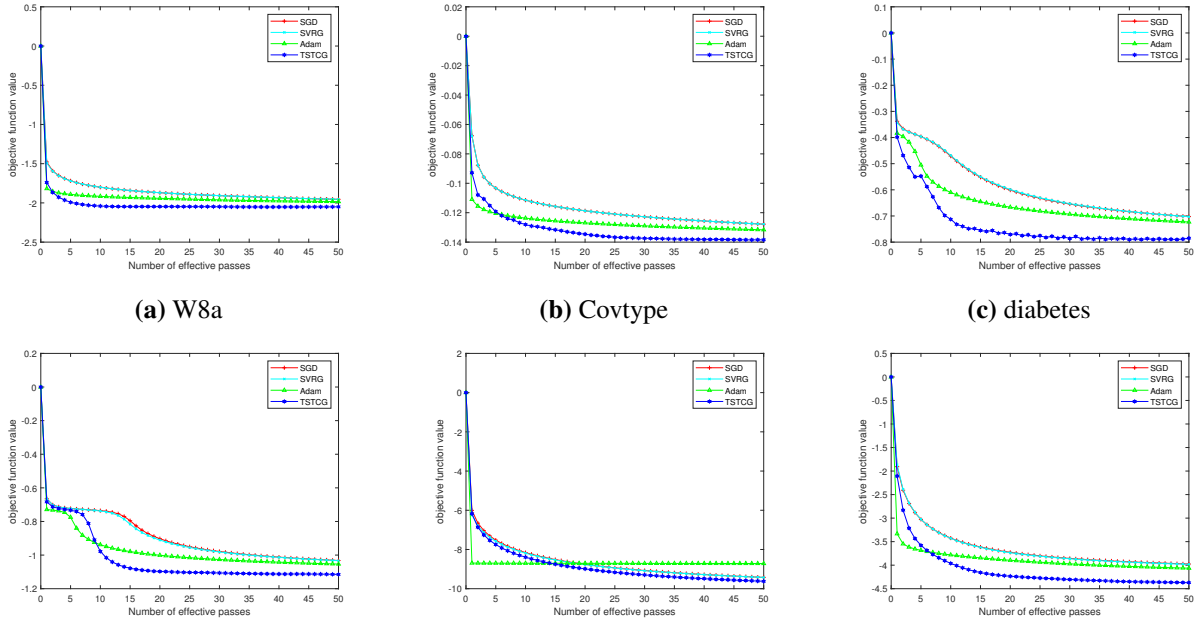
4.4. Analysis of experimental results

Performance on nonconvex models: Figures 1 and 2 illustrate the convergence for nonconvex Models 1 and 2. TSTCG maintains a competitive, often faster, initial descent rate compared to SGD and SVRG across varying dataset scales. Notably, on the *mushrooms* dataset (Figure 1), whereas Adam's convergence gradually slows, TSTCG sustains steady descent, benefiting from implicit tensor information navigating flat regions. Furthermore, on *Covtype* (Figure 2), although Adam drops rapidly initially, it fluctuates later. In contrast, guided by its theoretical sufficient descent property, TSTCG yields much smoother, more consistent trajectories without requiring dynamic step-size tuning.

Performance on convex models: Figures 3 and 4 demonstrate the performance on convex Models 3 and 4. Across datasets such as *a9a*, *diabetes*, and *ijcnn*, TSTCG reaches lower objective values faster than all baselines. This confirms that integrating high-order curvature information significantly accelerates convergence, extending TSTCG's versatility to standard convex landscapes.

Robustness to regularization coefficients: Figures 5 to 8 evaluate TSTCG under varying regularization parameters ($\lambda \in \{10^{-3}, 10^{-4}, 10^{-5}\}$). As theoretically expected, smaller λ allows deeper data fitting. Crucially, TSTCG consistently maintains highly stable and oscillation-free convergence trajectories across all λ scales, confirming its strong robustness to hyperparameter variations.

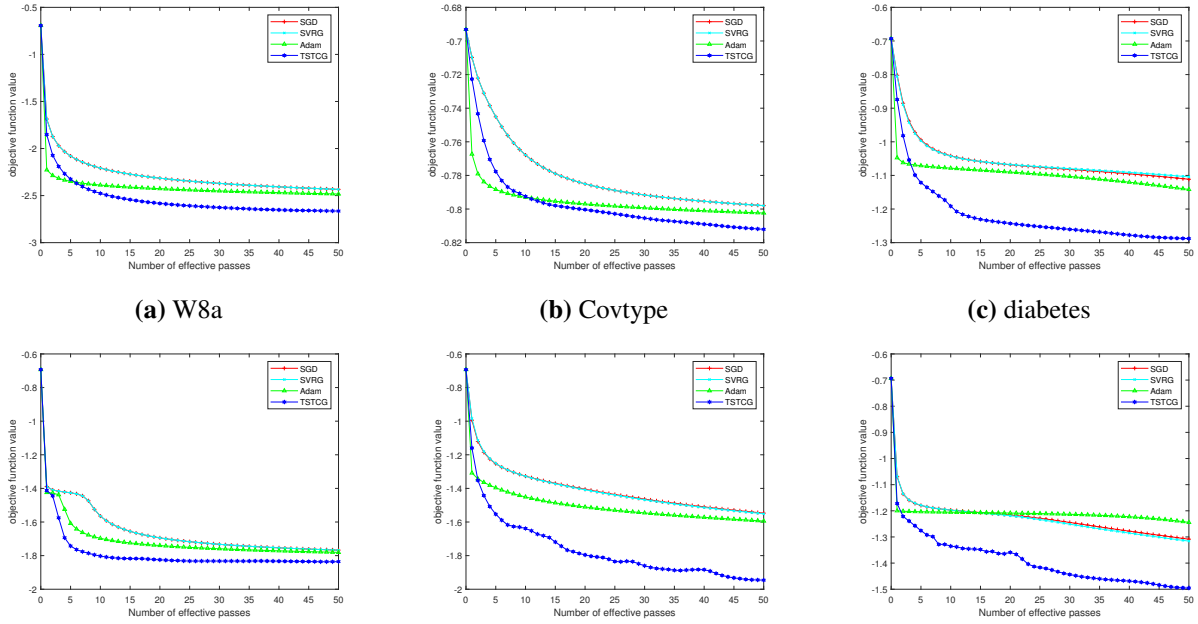
In summary, by synergizing tensor curvature with a robust three-term structure, TSTCG effectively mitigates gradient noise under strictly fixed step sizes, demonstrating superior speed, precision, and robustness across diverse machine learning tasks.



(a) W8a **(b)** Covtype **(c)** diabetes

(d) Adult **(e)** mushrooms **(f)** bodyfat

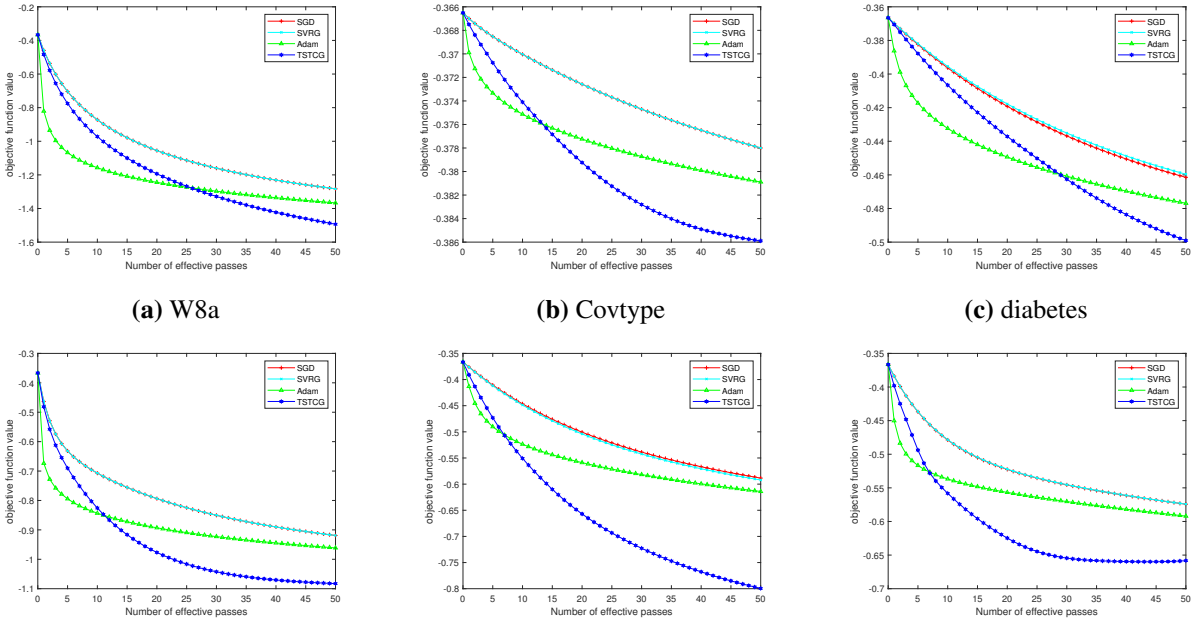
Figure 1. Numerical performance of SGD, SVRG, Adam, and TSTCG for solving Model 1 on six datasets.



(a) W8a **(b)** Covtype **(c)** diabetes

(d) a9a **(e)** inosphere **(f)** german.number

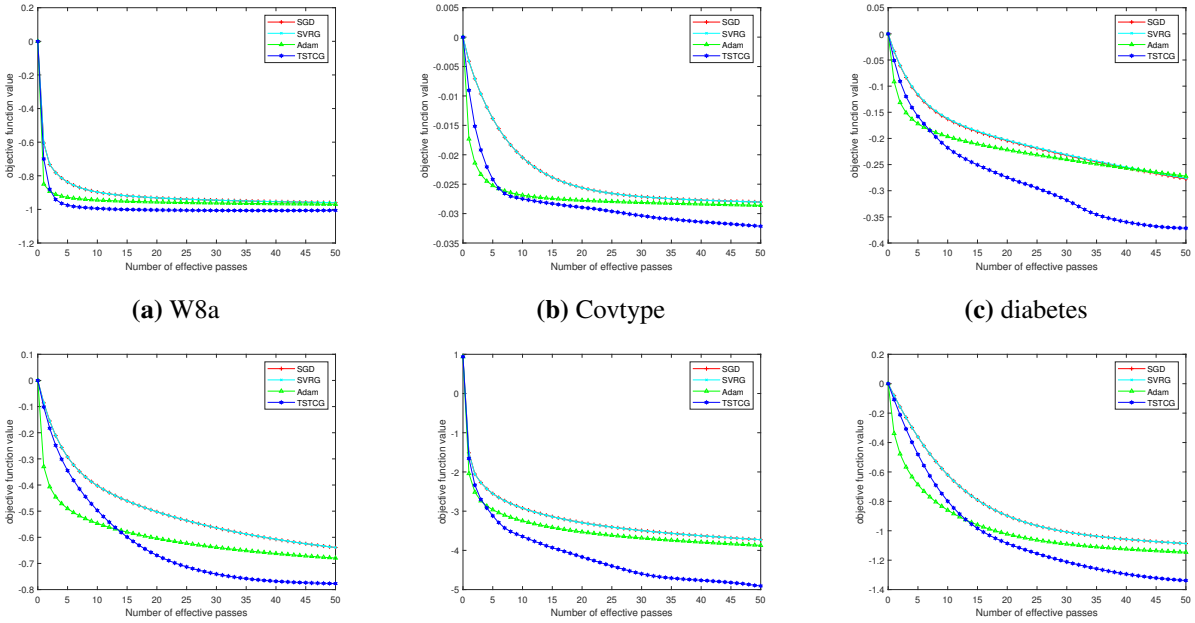
Figure 2. Numerical performance of SGD, SVRG, Adam, and TSTCG for solving Model 2 on six datasets.



(a) W8a **(b)** Covtype **(c)** diabetes

(d) a9a **(e)** inosphere **(f)** german.number

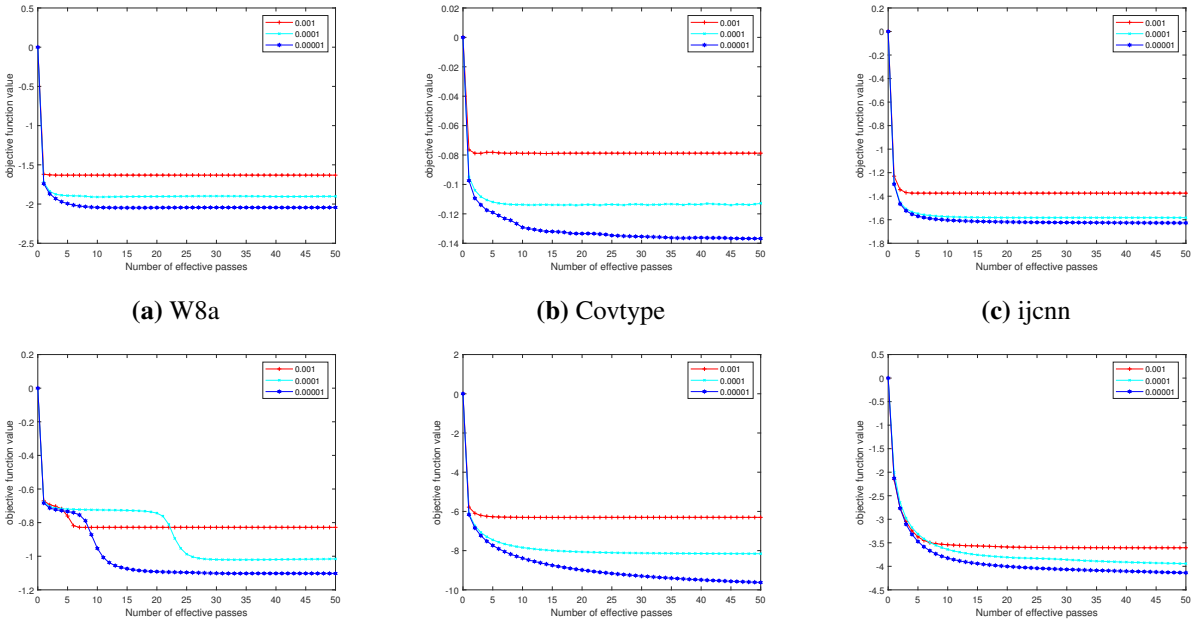
Figure 3. Numerical performance of SGD, SVRG, Adam, and TSTCG for solving Model 3 on six datasets.



(a) W8a **(b)** Covtype **(c)** diabetes

(d) Adam **(e)** mushrooms **(f)** ijcnn

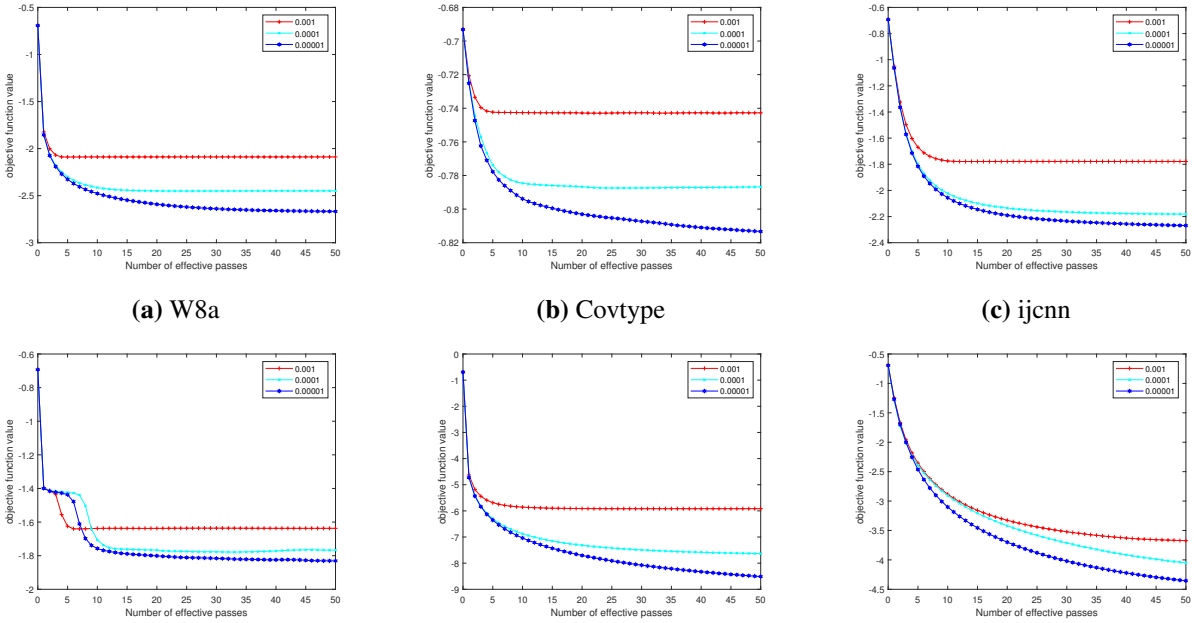
Figure 4. Numerical performance of SGD, SVRG, Adam, and TSTCG for solving Model 4 on six datasets.



(a) W8a **(b)** Covtype **(c)** ijcnn

(d) Adult **(e)** mushrooms **(f)** bodyfat

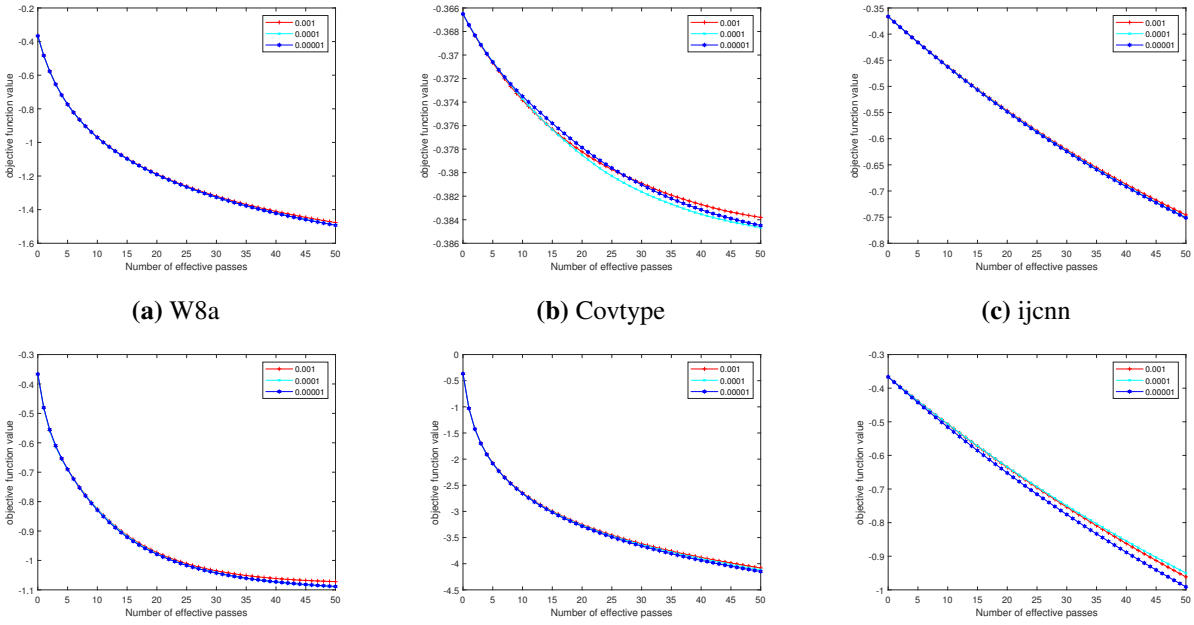
Figure 5. Performance of TSTCG on Model 1 under varying regularization coefficients.



(a) W8a **(b)** Covtype **(c)** ijcnn

(d) a9a **(e)** mushrooms **(f)** bodyfat

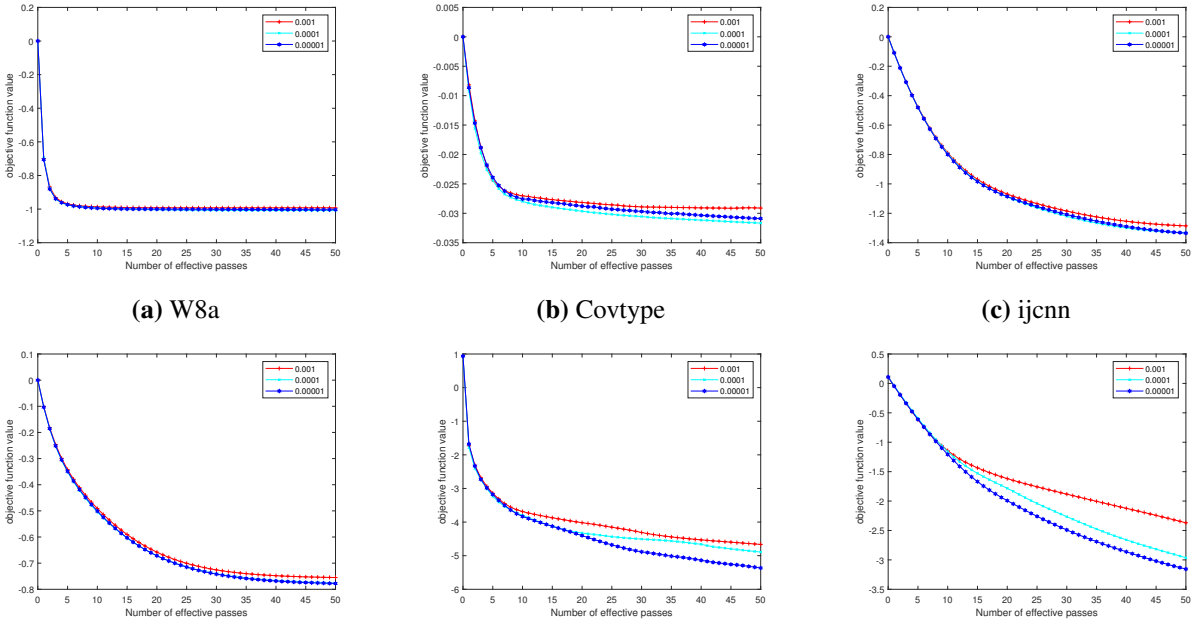
Figure 6. Performance of TSTCG on Model 2 under varying regularization coefficients.



(a) W8a **(b)** Covtype **(c)** ijcnn

(d) a9a **(e)** mushrooms **(f)** bodyfat

Figure 7. Performance of TSTCG on Model 3 under varying regularization coefficients.



(a) W8a **(b)** Covtype **(c)** ijcnn

(d) Adult **(e)** mushrooms **(f)** bodyfat

Figure 8. Performance of TSTCG on Model 4 under varying regularization coefficients.

5. Conclusions

This paper proposes and validates the TSTCG algorithm, aimed at efficiently addressing nonconvex optimization problems in large-scale machine learning. Research indicates that by implicitly integrating third-order tensor information, the algorithm significantly enhances the capture precision of high-order geometric features within complex nonconvex terrains, providing more accurate navigation for the descent path.

Furthermore, the designed three-term conjugate structure theoretically satisfies the sufficient descent property through adaptive parameter regulation. This characteristic allows the algorithm to completely eliminate its reliance on computationally expensive line searches while demonstrating exceptional numerical stability and resilience against gradient noise under fixed step sizes. Extensive numerical experiments demonstrate that TSTCG outperforms traditional SGD and mainstream adaptive algorithms in terms of convergence efficiency, precision, and robustness, representing significant practical value for engineering applications.

Future research can integrate the TSTCG algorithm framework with DSGE or CGE models to establish a macro-micro linkage for banking analysis. Embedding the TSTCG algorithm into the DSGE model in bank asset allocation decisions would allow systematic investigation of how adjustments in corporate lending, retail credit, bill business, and fixed-income investment propagate to banking variables such as revenues, write-offs, and credit spreads. Furthermore, the combination of the optimized banking structure from the TSTCG algorithm and multi-sector CGE model can quantify the real-sector impacts of bank resource reallocation, including effects on corporate investment, household consumption, and fiscal policy efficiency. Such integration would bridge micro-level nonconvex tensor optimization and macro general equilibrium analysis, offering deeper insights into the design of coordinated financial regulation and macroeconomic policies.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work is supported by the Guangxi Key Technologies R & D Program (Grant No. FN2504240023, AB25069188, FN2504240016), the Guangxi Science and Technology Base and Talent Project (Grant No. AD22080047), and the special foundation for Guangxi Ba Gui Scholars (Grant No. GXR-6BG2424008).

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. K. J. Bergen, P. A. Johnson, M. V. de Hoop, G. C. Beroza, Machine learning for data-driven discovery in solid Earth geoscience, *Science*, **363** (2019), eaau0323. https://doi.org/10.1126/science.aau0323
2. S. Malley, C. Reina, S. Nacy, J. Gilles, B. Koohbor, G. Youssef, Predictability of mechanical behavior of additively manufactured particulate composites using machine learning and data-driven approaches, *Comput. Ind.*, **142** (2022), 103739. https://doi.org/10.1016/j.compind.2022.103739
3. D. V. Akman, M. Malekipirbazari, Z. D. Yenice, A. Yeo, N. Adhikari, Y. K. Wong, et al., K-best feature selection and ranking via stochastic approximation, *Expert Syst. Appl.*, **213** (2023), 118864. https://doi.org/10.1016/j.eswa.2022.118864
4. N. Khan, M. R. Saleem, D. Lee, M. W. Park, C. Park, Utilizing safety rule correlation for mobile scaffolds monitoring leveraging deep convolution neural networks, *Comput. Ind.*, **129** (2021), 103448. https://doi.org/10.1016/j.compind.2021.103448
5. Y. Ma, F. Rusu, K. Wu, A. Sim, Adaptive stochastic gradient descent for deep learning on heterogeneous CPU+ GPU architectures, in *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, (2021), 6–15. https://doi.org/10.1109/IPDPSW50202.2021.00012
6. Y. LeCun, Y. Bengio, G. Hinton, Deep learning, *Nature*, **521** (2015), 436–444. https://doi.org/10.1038/nature14539
7. S. P. Awate, R. T. Whitaker, Unsupervised, information-theoretic, adaptive image filtering for image restoration, *IEEE Trans. Pattern Anal. Mach. Intell.*, **28** (2006), 364–376. https://doi.org/10.1109/TPAMI.2006.52
8. Y. Zhang, M. J. Wainwright, J. C. Duchi, Communication-efficient algorithms for statistical optimization, *Adv. Neural Inf. Process. Syst.*, **25** (2012), 1502–1510.
9. I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, *Deep Learning*, MIT Press, Cambridge, 2016.
10. S. Klein, M. Staring, J. P. Pluim, Evaluation of optimization methods for nonrigid medical image registration using mutual information and B-splines, *IEEE Trans. Image Process.*, **16** (2007), 2879–2890. https://doi.org/10.1109/TIP.2007.909315
11. Z. Allen-Zhu, Y. Yuan, Improved SVRG for non-strongly-convex or sum-of-nonconvex objectives, in *International Conference on Machine Learning*, (2016), 1080–1089.
12. H. Robbins, S. Monro, A stochastic approximation method, *Ann. Math. Statist.*, **22** (1951), 400–407. https://doi.org/10.1214/aoms/1177729586
13. L. Bottou, Large-scale machine learning with stochastic gradient descent, in *Proceedings of COMPSTAT'2010*, (2010), 177–186. https://doi.org/10.1007/978-3-7908-2604-3_16
14. R. Johnson, T. Zhang, Accelerating stochastic gradient descent using predictive variance reduction, *Adv. Neural Inf. Process. Syst.*, **26** (2013), 315–323.
15. A. Defazio, F. Bach, S. Lacoste-Julien, SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives, *Adv. Neural Inf. Process. Syst.*, **27** (2014), 1646–1654.

16. L. M. Nguyen, J. Liu, K. Scheinberg, M. Takáč, SARAH: A novel method for machine learning problems using stochastic recursive gradient, in *International Conference on Machine Learning*, (2017), 2613–2621.
17. J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, *J. Mach. Learn. Res.*, **12** (2011), 2121–2159.
18. G. Hinton, N. Srivastava, K. Swersky, *Neural Networks for Machine Learning*, 2012. Available from: https://www.cs.toronto.edu/ tijmen/csc321/slides/lecture_slides_lec6.pdf.
19. D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, preprint, arXiv:1412.6980. https://doi.org/10.48550/arXiv.1412.6980
20. S. J. Reddi, S. Kale, S. Kumar, On the convergence of Adam and beyond, preprint, arXiv:1904.09237. https://doi.org/10.48550/arXiv.1904.09237
21. M. Yousefi, Á. Martínez Calomardo, A stochastic modified limited memory BFGS for training deep neural networks, in *Science and Information Conference*, (2022), 9–28. https://doi.org/10.1007/978-3-031-10464-0_2
22. M. R. Hestenes, E. Stiefel, Methods of conjugate gradients for solving linear systems, *J. Res. Natl. Bur. Stand.*, **49** (1952), 409–436. https://doi.org/10.6028/jres.049.044
23. R. Fletcher, C. M. Reeves, Function minimization by conjugate gradients, *Comput. J.*, **7** (1964), 149–154. https://doi.org/10.1093/comjnl/7.2.149
24. E. Polak, G. Ribiere, Note sur la convergence de méthodes de directions conjuguées, *Rev. Fr. Inf. Rech. Oper.*, **3** (1969), 35–43. https://doi.org/10.1051/m2an/196903R100351
25. B. T. Polyak, The conjugate gradient method in extremal problems, *USSR Comput. Math. Math. Phys.*, **9** (1969), 94–112. https://doi.org/10.1016/0041-5553(69)90035-4
26. N. N. Schraudolph, T. Graepel, Combining conjugate direction methods with stochastic approximation of gradients, in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, (2003), 248–253.
27. H. Jiang, P. Wilford, A stochastic conjugate gradient method for the approximation of functions, *J. Comput. Appl. Math.*, **236** (2012), 2529–2544. https://doi.org/10.1016/j.cam.2011.12.012
28. X. B. Jin, X. Y. Zhang, K. Huang, G. G. Geng, Stochastic conjugate gradient algorithm with variance reduction, *IEEE Trans. Neural Netw. Learn. Syst.*, **30** (2018), 1360–1369. https://doi.org/10.1109/TNNLS.2018.2868841
29. C. Kou, H. Yang, A mini-batch stochastic conjugate gradient algorithm with variance reduction, *J. Global Optim.*, **87** (2023), 1009–1025. https://doi.org/10.1007/s10898-023-01306-0
30. J. Z. Zhang, N. Y. Deng, L. Chen, New quasi-Newton equation and related methods for unconstrained optimization, *J. Optim. Theory Appl.*, **102** (1999), 147–167. https://doi.org/10.1023/A:1021798319692
31. A. Bouaricha, Tensor methods for large, sparse unconstrained optimization, *SIAM J. Optim.*, **7** (1997), 732–756. https://doi.org/10.1137/S105262349427218X
32. J. Zhang, C. Xu, Properties and numerical performance of quasi-Newton methods with modified quasi-Newton equations, *J. Comput. Appl. Math.*, **137** (2001), 269–278. https://doi.org/10.1016/S0377-0427(00)00713-1

33. F. Biglari, M. A. Hassan, W. J. Leong, New quasi-Newton methods via higher order tensor models, *J. Comput. Appl. Math.*, **235** (2011), 2412–2422. https://doi.org/10.1016/j.cam.2010.10.040

34. L. Zhang, W. Zhou, D. Li, Some descent three-term conjugate gradient methods and their global convergence, *Optim. Methods Softw.*, **22** (2007), 697–711. https://doi.org/10.1080/10556780600812232

35. K. Sugiki, Y. Narushima, H. Yabe, Globally convergent three-term conjugate gradient methods that use secant conditions and generate descent search directions for unconstrained optimization, *J. Optim. Theory Appl.*, **153** (2012), 733–757. https://doi.org/10.1007/s10957-011-9964-1

36. M. Kelly, R. Landreville, K. Nottingham, *The UCI Machine Learning Repository*, 2023. Available from: https://archive.ics.uci.edu.

37. C. C. Chang, C. J. Lin, LIBSVM: A library for support vector machines, *ACM Trans. Intell. Syst. Technol.*, **2** (2011), 1–27. https://doi.org/10.1145/1961189.1961199

AIMS Press