SIAM J. SCI. COMPUT.
Vol. 47, No. 1, pp. C126–C150

A RELAXED VECTOR AUXILIARY VARIABLE ALGORITHM FOR UNCONSTRAINED OPTIMIZATION PROBLEMS*

SHIHENG ZHANG†, JIAHAO ZHANG†, JIE SHEN†, AND GUANG LIN†

Abstract. We present a novel optimization algorithm, a relaxed vector auxiliary variable (RVAV), that satisfies an unconditional energy dissipation law and exhibits improved alignment between the modified and the original energy. Our algorithm features rigorous proofs of linear convergence in the convex setting. Furthermore, we present a simple accelerated algorithm that improves the linear convergence rate to superlinear in the univariate case. We also propose an adaptive version of RVAV with Steffensen step size. We validate the robustness and fast convergence of our algorithm through ample numerical experiments.

Key words. optimization, gradient descent, machine learning, SAV, adaptive learning rate

MSC codes. 90C26, 68T99, 68W40

DOI. 10.1137/23M1611087

1. Introduction.

Optimization of neural network parameters is an area of active research with significant progress in recent years. However, it continues to pose formidable challenges, mainly due to vanishing gradients [9], overfitting [13], and the necessity for adaptive learning rate methods to avoid convergence to local minima [12, 28]. Several approaches, such as batch normalization [10] and adaptive gradient descent with energy (AEGD) [14], a relaxed scalar auxiliary variable (RSAV) [15, 29], have demonstrated promise in addressing some of these obstacles. The most commonly used approach for obtaining the update rule involves reducing a nonconvex loss function, for instance, the mean square error, $f(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i(\boldsymbol{x}))^2$ [16].

In the realm of mathematical optimization, it is customary to investigate the feasibility of unconstrained minimization problems that take the form

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}). \tag{1.1}$$

In this setting, we assume the function $f : \mathbb{R}^m \to \mathbb{R}$ is differentiable. Notably, this formulation represents a fundamental optimization problem that encompasses linear programming and least-square problems as particular cases. Furthermore, it has a complete analytical theory, as established in [4].

In the 1980s, a connection between the unconstrained minimization problem (1.1), in which the target function $f(\boldsymbol{x})$ is to be minimized over $\mathbb{R}^n$, and an ordinary differential equation (ODE) problem was established [5, 18, 33]. Specifically, a gradient

*Submitted to the journal's Machine Learning Methods for Scientific Computing section October 23, 2023; accepted for publication (in revised form) November 5, 2024; published electronically February 14, 2025.
https://doi.org/10.1137/23M1611087
Funding: This work is supported by the National Science Foundation (DMS-2053746, DMS-2134209, ECCS-2328241, CBET-2347401, and OAC-2311848), and U.S. Department of Energy (DOE) Office of Science Advanced Scientific Computing Research program DE-SC0023161, the Uncertainty Quantification for Multifidelity Operator Learning (MOLUcQ) project (project 81739), and DOE–Fusion Energy Science, under grant DE-SC0024583.

†Department of Mathematics, Purdue University, West Lafayette, IN 47907 USA. The first two authors (S. Zhang and J. Zhang) contributed equally to this paper. (zhan3722@purdue.edu, zhan2296@purdue.edu, shen7@purdue.edu, guanglin@purdue.edu).

descent (GD) method for problem (1.1), $\boldsymbol{x}^{n+1} = \boldsymbol{x}^n - \Delta t \nabla f(\boldsymbol{x}^n(t))$, can be considered as a numerical scheme of a gradient flow,

$$\frac{d\boldsymbol{x}}{dt} = -\nabla f(\boldsymbol{x}(t)), \tag{1.2}$$

where the initial point is $\boldsymbol{x}(0) = \boldsymbol{x}_0$. A minimizer $\boldsymbol{x}^*$ of $f(\boldsymbol{x})$ is then obtained as $\boldsymbol{x}^* = \lim_{t\to\infty} \boldsymbol{x}(t)$, where $\boldsymbol{x}(t)$ satisfies (1.2), and the optimal value of $f(\boldsymbol{x})$ is defined as $f^* = f(\boldsymbol{x}^*)$. Recently, there has been significant research examining the connection between minimization problems and ODE problems, including investigations into Nesterov's accelerated gradient (NAG) [26]. In the domain of machine learning, NAG has risen to prominence as a robust optimization tool, underscoring the need for effective numerical methods for solving such problems.

Additionally, (1.2) belongs to a notable class of ODEs known as gradient flows, which are ubiquitous in various fields such as fluid dynamics and material science problems [1, 2, 6]. It is desirable, sometimes necessary, for the numerical scheme to adhere to fundamental physical laws, including the energy dissipation law $\frac{df(\boldsymbol{x}(t))}{dt} \leq 0$. Certain contemporary literature has proposed several energy-dissipative numerical schemes, including the convex splitting schemes [7, 8, 19], stabilization methods [23, 32, 27, 30], scalar auxiliary variable (SAV) methods [20, 21, 22], and invariant energy quadratization (IEQ) approaches.

The treatment of the minimization problem as a gradient flow problem has gained popularity in optimization algorithms due to its robustness and generality. Recently, Liu and Tian [14] developed AEGD, which applied the IEQ to the optimization process, and Liu, Shen, and Zhang [15] applied the relaxed SAV technique to optimization. These methods ensure unconditional energy dissipation by introducing a kind of modified energy. However, the introduced modified energy may exhibit inconsistencies with the original energy, as the original energy may not necessarily monotonically decrease during iterations. Despite its potential, several challenges remain in the application of gradient flow methods to minimization problems. One of the challenges faced in designing optimization algorithms based on gradient flow is to maintain the physical law while designing the numerical scheme. Another challenge is to improve the convergence rate by selecting an appropriate step size. Avoiding oscillations in GD methods is also a challenge that needs to be addressed. Additionally, incorporating an adaptive algorithm can help save computation costs. To improve the performance of optimization algorithms based on gradient flow, further research is needed to address these challenges.

Within the context of the SAV approach, a new variable $r = \sqrt{f(\boldsymbol{x}(t))}$ is defined as the scalar auxiliary variable, and subsequently, an extended system needs to be solved. This approach, however, introduces two primary challenges in optimization. First, the numerical solution r^{n+1} may significantly deviate from $\sqrt{f(\boldsymbol{x}(t_{n+1}))}$. Second, r^{n+1} is prone to vanishing, especially as the dimensions of $\boldsymbol{x}(t)$ increase, leading to the vanishing of the learning rate $\Delta t \frac{r^{n+1}}{\sqrt{f(\boldsymbol{x}^n)}}$ when r^{n+1} does vanish. Drawing inspiration from the enhancement in IEQ and SAV presented in [11], the consistency between the modified and the original energy can be achieved by incorporating a relaxation step at the conclusion of each iteration. To prevent the vanishing of r^{n+1}, it is feasible to introduce a SAV for each element of $\boldsymbol{x}$, a method we will refer to as a vector auxiliary method (VAV). Here, we propose a new approach, the relaxed vector auxiliary variable (RVAV), in which the auxiliary variable is a vector instead of a scalar and a relaxation step is applied, allowing more flexibility in adjusting the learning

rate elementwise. Importantly, the introduced modified energy remains dissipative unconditionally, where the modified and original energy are inherently connected.

More precisely, the unconditionally modified energy dissipation can be obtained for each element of the vector $\boldsymbol{x}$, which facilitates the use of adaptive step size during iteration. To achieve this, we define an indicator, $\alpha = \text{mean}(\frac{\boldsymbol{r}^n}{\sqrt{f(\boldsymbol{x}^n)}})$, such that we can adjust the step size Δt according to the indicator's deviation from 1 and with the Steffensen step size [31], leading to an adaptive version of the RVAV, which, hereafter referred as ARVAV, may avoid oscillation and accelerate the convergence. We will show that by selecting the appropriate step size, the real energy will also be dissipative, which allows us to prove that it converges linearly. We also show that the convergence rate can be accelerated to superlinear in the univariate case.

In conclusion, our primary advancements include the following:

1. We propose a novel optimizer in Algorithm 2.1, RVAV, designed to significantly improve the performance of RSAV in high-dimensional problems, particularly in the context of machine learning. Compared to RSAV, RVAV mitigates the issue of vanishing modified energy by applying SAV in an elementwise manner before converging to the optimal value, which could otherwise halt the convergence.
2. We provide a rigorous proof of the convergence rate for the RSAV and RVAV algorithms in the convex setting in section 3.
3. We demonstrate that in the univariate case, the linear convergence rate can be elevated to a superlinear rate in section 4.
4. We introduce an indicator to monitor the performance of the optimization process and propose the ARVAV algorithm, which incorporates the indicator and provides guidelines on how to modify the step size when the indicator exceeds a certain threshold.
5. Through numerical experiments in section 5, we demonstrate that our algorithm achieves high accuracy and fast convergence.

The structure of this article is outlined as follows. Section 2 introduces the proposed RVAV algorithm. Section 3 presents the convergence analysis of RSAV and RVAV in the convex setting. We propose in section 4 an enhanced RVAV algorithm and show that it has a superlinear convergence rate. In section 5, several numerical experiments are presented to validate the effectiveness of the new algorithm, followed by some conclusions in section 6.

2. The RSAV and RVAV algorithms.

We start by recalling the SAV and RSAV schemes introduced in [15] for optimization problems, followed by the construction of VAV and RVAV schemes. We also show that these schemes are unconditionally stable with the modified energy.

2.1. SAV and relaxed SAV.

In general, we can split the cost function as follows:

$$f(\boldsymbol{x}(t)) = \frac{1}{2}(\mathcal{L}(t)\boldsymbol{x},\boldsymbol{x}) + \left[f(\boldsymbol{x}) - \frac{1}{2}(\mathcal{L}(t)\boldsymbol{x},\boldsymbol{x})\right], \tag{2.1}$$

where $\mathcal{L}(t)$ is a self-adjoint positive semidefinite linear operator. In this paper, we mostly consider the trivial splitting $\mathcal{L}(t) \equiv 0$ or $(\mathcal{L}(t)\boldsymbol{x})_i = \lambda_i(t)x_i$, $i = 1, 2, \ldots, m$, where $\boldsymbol{x} = (x_1, x_2, \ldots, x_i, \ldots, x_m)$.

Assuming, without loss of generality, $f(\boldsymbol{x}) \geq \delta > 0\ \forall \boldsymbol{x}$, we can define an auxiliary scalar variable as $r(t) = \sqrt{f(\boldsymbol{x}(t))}$, and subsequently extend the gradient flow (1.2) to

$$\frac{d\boldsymbol{x}}{dt} + \mathcal{L}(t)\boldsymbol{x} + \frac{r}{\sqrt{f(\boldsymbol{x}(t))}}(\nabla f(\boldsymbol{x}(t)) - \mathcal{L}(t)\boldsymbol{x}) = 0, \tag{2.2}$$

$$\frac{dr}{dt} = \frac{1}{2\sqrt{f(\boldsymbol{x}(t))}}\nabla f(\boldsymbol{x}(t))^T \frac{d\boldsymbol{x}}{dt}. \tag{2.3}$$

If we consider $r(0) = \sqrt{f(\boldsymbol{x}|_{t=0})}$, then the solution $\boldsymbol{x}$ of (1.2) along with $r(t) = \sqrt{f(\boldsymbol{x})}$ also represents a solution pair for the above-expanded system.

Then, we consider the following time discretization scheme for the expanded system:

$$\frac{\boldsymbol{x}^{n+1} - \boldsymbol{x}^n}{\Delta t} + \mathcal{L}^n(\boldsymbol{x}^{n+1} - \boldsymbol{x}^n) + \frac{r^{n+1}}{\sqrt{f(\boldsymbol{x}^n)}}\nabla f(\boldsymbol{x}^n) = 0, \tag{2.4}$$

$$\frac{r^{n+1} - r^n}{\Delta t} = \frac{1}{2\sqrt{f(\boldsymbol{x}^n)}}\nabla f(\boldsymbol{x}^n)^T \frac{\boldsymbol{x}^{n+1} - \boldsymbol{x}^n}{\Delta t}, \tag{2.5}$$

where we assume $\mathcal{L}^n \approx \mathcal{L}(t_n)$ to be self-adjoint and positive semidefinite. In the following context, we will refer to $f(\boldsymbol{x}^{n+1})$ as the "original energy" and $(r^{n+1})^2$ as the "modified energy."

The above SAV scheme is very efficient, since the coupled system (2.4)–(2.5) can be decoupled into two linear systems of the subsequent structure [21]:

$$\alpha\boldsymbol{x} + \mathcal{L}^n\boldsymbol{x} = \boldsymbol{h}. \tag{2.6}$$

The scheme (2.4)–(2.5) is unconditional energy dissipative for the modified energy $(r^{n+1})^2$. However, the equation used to compute r^{n+1} has little correlation with $\sqrt{f(\boldsymbol{x}^{n+1})}$, leading to inconsistencies between $\boldsymbol{r}^{n+1}$ and $\sqrt{f(\boldsymbol{x}^{n+1})}$ in numerical experiments. To address this issue, we adopt a relaxation step [11] that strengthens the relationship between r^{n+1} and $\sqrt{f(\boldsymbol{x}^{n+1})}$. More precisely, the RSAV scheme is as follows:

$$\frac{\boldsymbol{x}^{n+1} - \boldsymbol{x}^n}{\Delta t} + \mathcal{L}^n(\boldsymbol{x}^{n+1} - \boldsymbol{x}^n) + \frac{\tilde{r}^{n+1}}{\sqrt{f(\boldsymbol{x}^n)}}\nabla f(\boldsymbol{x}^n) = 0, \tag{2.7}$$

$$\frac{\tilde{r}^{n+1} - r^n}{\Delta t} = \frac{1}{2\sqrt{f(\boldsymbol{x}^n)}}\nabla f(\boldsymbol{x}^n)^T \frac{\boldsymbol{x}^{n+1} - \boldsymbol{x}^n}{\Delta t}, \tag{2.8}$$

$$r^{n+1} = \eta_0\tilde{r}^{n+1} + (1 - \eta_0)\sqrt{f(\boldsymbol{x}^{n+1})}, \tag{2.9}$$

where, for a given $\psi \in (0, 1)$, η_0 is the smallest number in $[0, 1]$ such that

$$(r^{n+1})^2 - (\tilde{r}^{n+1})^2 \le \frac{\psi}{\Delta t}\|\boldsymbol{x}^{n+1} - \boldsymbol{x}^n\|^2, \tag{2.10}$$

where ψ is a parameter of our choice and is usually set as $\psi = 0.95$ in practice. We refer to [11] (see (2.18) below) for an explicit formula to determine η_0.

2.2. VAV and RVAV schemes. The scheme (2.4)–(2.5), in the case of $\mathcal{L}^n = 0$, can be interpreted as a GD scheme with a single learning rate $\Delta t\frac{r^{n+1}}{\sqrt{f(\boldsymbol{x}^n)}}$. However, it may converge slowly if the components of $\nabla f(\boldsymbol{x}^n)$ have large variations. In this case, it is preferable to have elementwise learning rates. To this end, we modify the SAV scheme (2.4)–(2.5) into the following VAV scheme:

$$\frac{x_i^{n+1}-x_i^n}{\Delta t}+(\mathcal{L}^n(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n))_i+\frac{r_i^{n+1}}{\sqrt{f(\boldsymbol{x}^n)}}\frac{\partial f(\boldsymbol{x}^n)}{\partial x_i}=0,\quad i=1,2,\dots,m, \tag{2.11}$$

$$\frac{r_i^{n+1}-r_i^n}{\Delta t}=\frac{1}{2\sqrt{f(\boldsymbol{x}^n)}}\frac{\partial f(\boldsymbol{x}^n)}{\partial x_i}\frac{x_i^{n+1}-x_i^n}{\Delta t},\quad i=1,2,\dots,m. \tag{2.12}$$

Note that with $\mathcal{L}^n=0$, the above scheme is essentially the same as the AEGD algorithm in [14].

Let $\boldsymbol{r}=(r_1,r_2,\dots,r_i,\dots,r_m)$ be denoted, with $(\cdot,\cdot)$ and $\|\cdot\|$ representing the inner product and norm, respectively, in $\mathbb{R}^m$.

THEOREM 2.1. *The VAV algorithm* (2.11)–(2.12) *is unconditionally energy dissipative, characterized by the fact that*

$$\|\boldsymbol{r}^{n+1}\|^2-\|\boldsymbol{r}^n\|^2=-\|\boldsymbol{r}^{n+1}-\boldsymbol{r}^n\|^2 - (\mathcal{L}^n(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n),\boldsymbol{x}^{n+1}-\boldsymbol{x}^n)-\frac{1}{\Delta t}\|\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\|^2\le 0.$$

In particular, if $(\mathcal{L}^n\boldsymbol{x})_i=\lambda_i^n x_i$ *with* $\lambda_i^n\ge 0\ \forall i,n$, *we have the elementwise inequality*

$$(r_i^{n+1})^2-(r_i^n)^2\le-\frac{1}{\Delta t}(x_i^{n+1}-x_i^n)^2\le 0 \text{ for } 1\le i\le m.$$

Proof. Multiplying (2.11) (resp., (2.12)) with $x_i^{n+1}-x_i^n$ (resp., $2r_i^{n+1}\Delta t$) and taking the sum of the results, we derive

$$\begin{aligned}&(r_i^{n+1})^2-(r_i^n)^2+(r_i^{n+1}-r_i^n)^2\\&\quad=-(\mathcal{L}^n(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n))_i\cdot(x_i^{n+1}-x_i^n)-\frac{1}{\Delta t}(x_i^{n+1}-x_i^n)^2.\end{aligned} \tag{2.13}$$

Summing up (2.13) for $i=1,2,\dots,m$, we derive

$$\begin{aligned}&\|\boldsymbol{r}^{n+1}\|^2-\|\boldsymbol{r}^n\|^2+\|\boldsymbol{r}^{n+1}-\boldsymbol{r}^n\|^2\\&\quad=-(\mathcal{L}^n(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n),\boldsymbol{x}^{n+1}-\boldsymbol{x}^n)-\frac{1}{\Delta t}\|\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\|^2.\end{aligned}$$

∎

Remark 2.2. For general $\mathcal{L}^n$, the components $(x_1^{n+1},\dots,x_m^{n+1})$ in (2.11)–(2.12) are coupled. However, if $(\mathcal{L}^n\boldsymbol{x})_i=\lambda_i^n x_i$, then $\{x_i^{n+1}\}$ in (2.11)–(2.12) are decoupled from each other and can be efficiently solved.

Accordingly, we can construct the RVAV scheme as follows: For $i=1,2,\dots,m$,

$$\frac{x_i^{n+1}-x_i^n}{\Delta t}+(\mathcal{L}^n(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n))_i+\frac{\tilde{r}_i^{n+1}}{\sqrt{f(\boldsymbol{x}^n)}}\frac{\partial f(\boldsymbol{x}^n)}{\partial x_i}=0, \tag{2.14}$$

$$\frac{\tilde{r}_i^{n+1}-r_i^n}{\Delta t}=\frac{1}{2\sqrt{f(\boldsymbol{x}^n)}}\frac{\partial f(\boldsymbol{x}^n)}{\partial x_i}\frac{x_i^{n+1}-x_i^n}{\Delta t}, \tag{2.15}$$

$$r_i^{n+1}=\eta_{i0}\tilde{r}_i^{n+1}+(1-\eta_{i0})\sqrt{f(\boldsymbol{x}^{n+1})}, \tag{2.16}$$

where, for a given $\psi\in(0,1)$, η_{i0} is the smallest number in $[0,1]$ such that

$$(r_i^{n+1})^2-(\tilde{r}_i^{n+1})^2\le\frac{\psi}{\Delta t}\left(x_i^{n+1}-x_i^n\right)^2. \tag{2.17}$$

Following [11], η_{i0} can be determined as follows:

$$\begin{aligned}&\eta_{i0}=\min_{\eta_i\in[0,1]}\eta_i \text{ such that}\\&(\eta_i\tilde{r}_i^{n+1}+(1-\eta_i)\sqrt{f(\boldsymbol{x}^{n+1})})^2-(\tilde{r}_i^{n+1})^2\le\frac{\psi}{\Delta t}\left(x_i^{n+1}-x_i^n\right)^2,\end{aligned}$$

which can be reduced to

$$\eta_{i0} = \min_{\eta_i \in [0,1]} \eta_i \text{ such that } a\eta_i^2 + b\eta_i + c \le 0, \tag{2.18}$$

where

$$\begin{aligned} a &= (\sqrt{f(\boldsymbol{x}^{n+1})} - \tilde{r}_i^{n+1})^2, \\ b &= 2\sqrt{f(\boldsymbol{x}^{n+1})}(\tilde{r}_i^{n+1} - \sqrt{f(\boldsymbol{x}^{n+1})}), \\ c &= f(\boldsymbol{x}^{n+1}) - (\tilde{r}_i^{n+1})^2 - \frac{\psi}{\Delta t}(x_i^{n+1} - x_i^n)^2. \end{aligned} \tag{2.19}$$

If $a = 0$, i.e., $\tilde{r}_i^{n+1} = \sqrt{f(\boldsymbol{x}^{n+1})}$, we set $\eta_{i0} = 0$. Otherwise, the solution to the problem (2.18) can be written as

$$\eta_{i0} = \max\left\{ \frac{-b - \sqrt{b^2 - 4ac}}{2a}, 0 \right\}. \tag{2.20}$$

It is easy to check that $b^2 - 4ac \ge 0$ for any Δt.

Assuming that $(\mathcal{L}^n \boldsymbol{x})_i = \lambda_i^n x_i$, the RVAV algorithm is given in Algorithm 2.1.

THEOREM 2.3. *The RVAV algorithm* (2.14)–(2.16) *is unconditionally energy dissipative, characterized by the fact that*

$$\|\boldsymbol{r}^{n+1}\|^2 - \|\boldsymbol{r}^n\|^2 \le -(\mathcal{L}^n(\boldsymbol{x}^{n+1} - \boldsymbol{x}^n), \boldsymbol{x}^{n+1} - \boldsymbol{x}^n) - \frac{1-\psi}{\Delta t}\|\boldsymbol{x}^{n+1} - \boldsymbol{x}^n\|^2 \le 0. \tag{2.21}$$

In particular, if $(\mathcal{L}^n \boldsymbol{x})_i = \lambda_i^n x_i$ *with* $\lambda_i^n \ge 0\ \forall i, n$, *we have the elementwise inequality*

$$(r_i^{n+1})^2 - (r_i^n)^2 \le -\frac{1-\psi}{\Delta t}(x_i^{n+1} - x_i^n)^2 \le 0 \text{ for } 1 \le i \le m. \tag{2.22}$$

Proof. The proof is exact as in Theorem 2.1, instead of (2.13), we can obtain

$$(\tilde{r}_i^{n+1})^2 - (r_i^n)^2 + (\tilde{r}_i^{n+1} - r_i^n)^2 = -(\mathcal{L}^n(\boldsymbol{x}^{n+1} - \boldsymbol{x}^n))_i \cdot (x_i^{n+1} - x_i^n) - \frac{1}{\Delta t}(x_i^{n+1} - x_i^n)^2.$$

Algorithm 2.1. RVAV.

Given a starting point $\boldsymbol{x}^0 \in \mathbf{dom} f$, a step size Δt, $\boldsymbol{r}^0 = \sqrt{f(\boldsymbol{x}^0)}(1, 1, \ldots, 1)$ and set $n = 0$, $\psi \in (0, 1)$.
while the termination condition is not met **do**
  Compute $\tilde{r}_i^{n+1} = (1 + \frac{\Delta t}{2(1+\Delta t \lambda_i^n) f(\boldsymbol{x}^n)}(\frac{\partial f(\boldsymbol{x}^n)}{\partial x_i})^2)^{-1} r_i^n$ for $i = 1, \ldots, m$
  Update $x_i^{n+1} = x_i^n - \Delta t(1 + \Delta t \lambda_i^n)^{-1} \frac{\tilde{r}_i^{n+1}}{\sqrt{f(\boldsymbol{x}^n)}} \frac{\partial f(\boldsymbol{x}^n)}{\partial x_i}$ for $i = 1, \ldots, m$
  Set $r_i^{n+1} = \eta_i \tilde{r}_i^{n+1} + (1 - \eta_i)\sqrt{f(\boldsymbol{x}^{n+1})}$ for $i = 1, \ldots, m$
  Compute $\eta_{i0} = \min_{\eta_i \in [0,1]} \eta_i$ such that $(r_i^{n+1})^2 - (\tilde{r}_i^{n+1})^2 \le \frac{\psi}{\Delta t}\left(x_i^{n+1} - x_i^n\right)^2$ for $i = 1, \ldots, m$
  Update $r_i^{n+1} = \eta_{i0} \tilde{r}_i^{n+1} + (1 - \eta_{i0})\sqrt{f(\boldsymbol{x}^{n+1})}$ for $i = 1, \ldots, m$
  Update $n = n + 1$
end while
return $\boldsymbol{x}^{n+1}$

Hence, summing up the above with (2.17), we find

$$(r_i^{n+1})^2 - (r_i^n)^2 \leq -(\mathcal{L}^n(\boldsymbol{x}^{n+1} - \boldsymbol{x}^n))_i \cdot (x_i^{n+1} - x_i^n) - \frac{1-\psi}{\Delta t}(x_i^{n+1} - x_i^n)^2, \tag{2.23}$$

which implies (2.22) if $(\mathcal{L}^n \boldsymbol{x})_i = \lambda_i^n x_i$ with $\lambda_i^n \geq 0$. Summing up (2.23) for $i = 1, 2, \ldots, m$, we obtain (2.21). ∎

2.3. VAV versus SAV: Addressing vanishing variables in high-dimensional problems. In both SAV and VAV, the modified energy variable, r^n, tends to decrease. Ensuring that r^n does not vanish before convergence is crucial, as its disappearance would halt algorithmic progress.

For the SAV method, according to (2.4) and (2.5), we have

$$r^{n+1} = \left(1 + \frac{\Delta t}{2f(\boldsymbol{x}^n)} \nabla f(\boldsymbol{x}^n)^T (I + \Delta t \mathcal{L}^n)^{-1} \nabla f(\boldsymbol{x}^n)\right)^{-1} r^n.$$

In high-dimensional problems, particularly in machine learning and deep learning, the weighted norm squared of the gradient, $\nabla f(\boldsymbol{x}^n)^T (I + \Delta t \mathcal{L}^n)^{-1} \nabla f(\boldsymbol{x}^n)$, escalates as the dimensions of $\boldsymbol{x}^n$. This escalation can drive r^{n+1} toward zero. Should r^{n+1} vanish before the optimal point is reached, the algorithm will cease to progress and fail to find the optimal point.

In contrast, the VAV method, as delineated by (2.11) and (2.12), presents a different scenario:

$$r_i^{n+1} = \left(1 + \frac{\Delta t}{2f(\boldsymbol{x}^n)} (1 + \Delta t \lambda_i^n)^{-1} \left(\frac{\partial f(\boldsymbol{x}^n)}{\partial x_i}\right)^2\right)^{-1} r^n,\ i = 1, \ldots, m.$$

Here, the term $(1 + \Delta t \lambda_i^n)^{-1} (\frac{\partial f(\boldsymbol{x}^n)}{\partial x_i})^2$ is typically much smaller than $\nabla f(\boldsymbol{x}^n)^T (I + \Delta t \mathcal{L}^n)^{-1} \nabla f(\boldsymbol{x}^n)$. As a result, r_i^{n+1} is less likely to vanish. Consequently, VAV offers a more reliable method for high-dimensional optimization, reducing the risk of premature termination of the algorithm. The analysis concerning the risk of vanishing variables is similarly applicable to RVAV and RSAV.

3. Convergence analysis of the RSAV and RVAV schemes. In this section, we assume $f(\boldsymbol{x})$ to be L-smooth (see the definition below) and carry out a convergence analysis for the RSAV and RVAV schemes. We note that for the special case of $\mathcal{L} \equiv 0$, the rate at which both SAV and VAV schemes converge was established in [14]. We also note that in [15], the SAV scheme was formulated as a line search method and some convergence criteria were derived.

DEFINITION 3.1. *A function f is L-smooth if there is a nonnegative constant L with $\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|$ holding $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbf{R}^n$, i.e., ∇f is L-Lipschitz continuous.*

3.1. Positive lower bound of r^n for the RSAV scheme. The convergence theory of GD emphasizes the importance of maintaining the learning rate above a certain positive constant. As evidenced in [14], for the SAV scheme (2.4)–(2.5) when $\mathcal{L} \equiv 0$ and f is L-smooth, the term r^n remains bounded above a positive constant. We show below that this is also true for the RSAV scheme (2.7)–(2.9).

Let us denote $g(\boldsymbol{x}) = \sqrt{f(\boldsymbol{x})}$, and it can be readily demonstrated that $g(\boldsymbol{x})$ is also bounded below with a positive constant $\sqrt{\delta}$. We can rewrite (2.7)–(2.8) as

$$\frac{\boldsymbol{x}^{n+1}-\boldsymbol{x}^n}{\Delta t}+\mathcal{L}^n(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n)+2\tilde{r}^{n+1}\nabla g(\boldsymbol{x}^n)=0, \tag{3.1}$$

$$\frac{\tilde{r}^{n+1}-r^n}{\Delta t}=\nabla g(\boldsymbol{x}^n)^T\frac{\boldsymbol{x}^{n+1}-\boldsymbol{x}^n}{\Delta t}, \tag{3.2}$$

$$r^{n+1}=\eta_0\tilde{r}^{n+1}+(1-\eta_0)g(\boldsymbol{x}^{n+1}). \tag{3.3}$$

THEOREM 3.2. *Suppose that f is L-smooth with a positive lower bound δ, and let r^n be generated by the RSAV scheme (2.7)–(2.9) with $\mathcal{L}$ being positive semidefinite. Then $\lim_{n\to\infty}\boldsymbol{x}^n=\boldsymbol{x}^*$, and there exists a positive constant C_1 such that for $\Delta t\le C_1$, we then obtain*

$$\lim_{n\to\infty}r^n=r^*\ge\frac{\sqrt{\delta}}{2}>0,\quad \text{and } \nabla f(\boldsymbol{x}^*)=0.$$

Proof. First, it is easy to show that for an L-smooth function f, if f has a positive lower bound δ, then g is L_g-smooth with $L_g=\frac{L}{2\delta}$.

Taking the inner product with $(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n)$ of (3.1) and combining it with (3.2), we obtain

$$\left(\tilde{r}^{n+1}\right)^2-(r^n)^2+\left(\tilde{r}^{n+1}-r^n\right)^2=-\frac{1}{\Delta t}\|\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\|^2-\left(\mathcal{L}^n(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n),\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\right).$$

Summing up the above with (2.10), since $\mathcal{L}^n$ is positive semidefinite, we can obtain

$$\|\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\|^2\le\frac{\Delta t}{1-\psi}\left((r^n)^2-\left(r^{n+1}\right)^2\right). \tag{3.4}$$

Hence, $(r^{n+1})^2$ is a decreasing sequence and will converge to $(r^*)^2$ for some $r^*\ge 0$. It remains to show $r^*>0$.

Taking the sum of the aforementioned for $n=0,1,2,\ldots$, we find

$$\sum_{n=0}^{\infty}\|\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\|^2\le\frac{\Delta t}{1-\psi}\left(\left(r^0\right)^2-(r^*)^2\right). \tag{3.5}$$

Hence, $\lim_{n\to\infty}\boldsymbol{x}^n=\boldsymbol{x}^*$. On the other hand, we derive from (3.1)–(3.2) that

$$\tilde{r}^{n+1}=\frac{r^n}{1+2\Delta t\nabla g(\boldsymbol{x}^n)^T(I+\Delta t\mathcal{L}^n)^{-1}\nabla g(\boldsymbol{x}^n)}. \tag{3.6}$$

Hence we have $\tilde{r}^{n+1}\ge 0$ if $r^0\ge 0$. Furthermore, we observe from (2.9) that r^{n+1} is actually a convex combination of $\tilde{r}^{n+1}$ and $g(\boldsymbol{x}^{n+1})$. Hence $r^{n+1}\ge 0$ and it is also a decreasing sequence.

Without loss of generality, let's consider a positive integer N such that the inequality $r^n\le\sqrt{\delta}$ holds $\forall n\ge N$. If this were not the case, we could logically infer that $r^*\ge\sqrt{\delta}$. As a result, for every $n\ge N$, we can consequently derive

$$0\le\tilde{r}^n\le r^n\le g(\boldsymbol{x}^n). \tag{3.7}$$

For any $n\ge N$, we derive from (3.4), Taylor expansion, (3.2), and (3.7) that

$$\begin{aligned} g(\boldsymbol{x}^{n+1}) &\le g(\boldsymbol{x}^n)+\nabla g(\boldsymbol{x}^n)\left(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\right)+\frac{L_g}{2}\|\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\|^2 \\ &\le g(\boldsymbol{x}^n)+\tilde{r}^{n+1}-r^n+\frac{L_g}{2}\|\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\|^2 \\ &\le g(\boldsymbol{x}^n)+r^{n+1}-r^n+\frac{\Delta t L_g}{2(1-\psi)}\left((r^n)^2-\left(r^{n+1}\right)^2\right). \end{aligned} \tag{3.8}$$

Summing up the above from N to K, we obtain

$$g(\boldsymbol{x}^{K+1}) \le g(\boldsymbol{x}^N) + r^{K+1} - r^N + \frac{\Delta t L_g}{2(1-\psi)}\left(\left(r^N\right)^2 - \left(r^{K+1}\right)^2\right). \tag{3.9}$$

As $\boldsymbol{x}^*$ is the optimal point of $f(\boldsymbol{x})$ and $g(\boldsymbol{x}) = \sqrt{f(\boldsymbol{x})}$, $\boldsymbol{x}^*$ is also the optimal point of g. Letting K go to $+\infty$ in the above, since $g(\boldsymbol{x}^*) \le g(\boldsymbol{x}^{K+1})$, we obtain

$$g(\boldsymbol{x}^*) \le g(\boldsymbol{x}^N) + r^* - r^N + \frac{\Delta t L_g}{2(1-\psi)}\left(\left(r^N\right)^2 - \left(r^*\right)^2\right), \tag{3.10}$$

from which we can derive

$$r^* \ge g(\boldsymbol{x}^*) + r^N - g(\boldsymbol{x}^N) - \frac{\Delta t L_g}{2(1-\psi)}\left(r^N\right)^2. \tag{3.11}$$

Next we bound the difference between r^N and $g(\boldsymbol{x}^N)$. We derive from (3.3) that

$$r^N - g\left(\boldsymbol{x}^N\right) = \eta_0 \tilde{r}^N - \eta_0 g\left(\boldsymbol{x}^N\right) = \eta_0\left(\tilde{r}^N - g\left(\boldsymbol{x}^N\right)\right). \tag{3.12}$$

By (3.2) and Taylor expansion,

$$\begin{aligned}\tilde{r}^N - r^{N-1} &= \nabla g\left(\boldsymbol{x}^{N-1}\right)\cdot\left(\boldsymbol{x}^N - \boldsymbol{x}^{N-1}\right)\\ &= g\left(\boldsymbol{x}^N\right) - g\left(\boldsymbol{x}^{N-1}\right) - \frac{1}{2}\left(\boldsymbol{x}^N - \boldsymbol{x}^{N-1}\right)^T \nabla^2 g(\xi_N)\left(\boldsymbol{x}^N - \boldsymbol{x}^{N-1}\right).\end{aligned}$$

Hence, with the notation $\|a\|_H^2 = a^T H a$, we find from the above that

$$\begin{aligned}\tilde{r}^N - g\left(\boldsymbol{x}^N\right) &= r^{N-1} - g\left(\boldsymbol{x}^{N-1}\right) - \frac{1}{2}\|\boldsymbol{x}^N - \boldsymbol{x}^{N-1}\|^2_{\nabla^2 g(\xi_N)}\\ &= \eta_0\left(\tilde{r}^{N-1} - g\left(\boldsymbol{x}^{N-1}\right)\right) - \frac{1}{2}\|\boldsymbol{x}^N - \boldsymbol{x}^{N-1}\|^2_{\nabla^2 g(\xi_N)}\\ &= \eta_0\left(r^{N-2} - g\left(\boldsymbol{x}^{N-2}\right) - \frac{1}{2}\|\boldsymbol{x}^{N-1} - \boldsymbol{x}^{N-2}\|^2_{\nabla^2 g(\xi_{N-1})}\right)\\ &\quad - \frac{1}{2}\|\boldsymbol{x}^N - \boldsymbol{x}^{N-1}\|^2_{\nabla^2 g(\xi_N)}\\ &= \cdots\\ &= \eta_0\left(r^0 - g\left(\boldsymbol{x}^0\right)\right) - \frac{1}{2}\sum_{k=1}^{N}(\eta_0)^k\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\|^2_{\nabla^2 g(\xi_k)}.\end{aligned}$$

Since $r^0 = g\left(\boldsymbol{x}^0\right)$ and $a^T(\nabla^2 g)a \le L_g\|a\|^2$, we have

$$\begin{aligned}|\tilde{r}^N - g\left(\boldsymbol{x}^N\right)| &= \frac{1}{2}\sum_{k=1}^{N}\eta_0^k\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\|^2_{\nabla^2 g(\xi_k)}\\ &\le \frac{L_g}{2}\sum_{k=1}^{N}\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\|^2.\end{aligned} \tag{3.13}$$

Noting that $\|\boldsymbol{x}^{n+1} - \boldsymbol{x}^n\|^2 \le \frac{\Delta t}{1-\psi}((r^n)^2 - (r^{n+1})^2)\ \forall n$, we have

$$\begin{aligned}|\tilde{r}^N - g\left(\boldsymbol{x}^N\right)| &\le \frac{L_g\Delta t}{2(1-\psi)}\sum_{k=1}^{N}\left(\left(r^{k-1}\right)^2 - \left(r^k\right)^2\right)\\ &= \frac{L_g\Delta t}{2(1-\psi)}\left(\left(r^0\right)^2 - \left(r^N\right)^2\right).\end{aligned} \tag{3.14}$$

Hence, we derive from the above and (3.12) that

$$|r^N - g(\boldsymbol{x}^N)| \le \eta_0 \frac{L_g \Delta t}{2(1-\psi)} \left(\left(r^0\right)^2 - \left(r^N\right)^2 \right).$$

Letting $C_1 = \frac{1}{2} g(\boldsymbol{x}^*) / (\eta_0 \frac{L_g}{2(1-\psi)} (r^0)^2 + (1-\eta_0) \frac{L_g}{2(1-\psi)} (r^N)^2)$, we find from the above and (3.11) that for $\Delta t \le C_1$, we have

$$r^* \ge g(\boldsymbol{x}^*) + r^N - g(\boldsymbol{x}^N) - \frac{\Delta t L_g}{2\,(1-\psi)} \left(r^N\right)^2 \ge \frac{1}{2} g(\boldsymbol{x}^*) \ge \frac{\sqrt{\delta}}{2} > 0. \tag{3.15}$$

Finally, letting $n \to \infty$ in (2.7), we find $\nabla f(\boldsymbol{x}^*) = 0$. The proof is complete. ∎

3.2. Positive lower bound of r_i^n for the RVAV scheme. The study conducted in [14] demonstrates that when $\mathcal{L} \equiv 0$ and f is L-smooth, the values of r_i^n originating from the VAV scheme (2.11)–(2.12) exhibit a positive lower bound. Assuming $(\mathcal{L}^n \boldsymbol{x})_i = \lambda_i^n x_i$ with $\lambda_i^n \ge 0\ \forall i, n$, we show below that $\{r_i^n\}_{i=1,\dots,m}$ of the RVAV scheme (2.14)–(2.16) are bounded from below by a positive constant.

We first rewrite (2.14)–(2.16) as follows:

$$\frac{x_i^{n+1} - x_i^n}{\Delta t} + (\mathcal{L}^n(\boldsymbol{x}^{n+1} - \boldsymbol{x}^n))_i = -2\tilde{r}_i^{n+1} \partial_i g(\boldsymbol{x}^n), \tag{3.16}$$

$$\tilde{r}_i^{n+1} - r_i^n = \partial_i g(\boldsymbol{x}^n) \left(x_i^{n+1} - x_i^n\right), \tag{3.17}$$

$$r_i^{n+1} = \eta_i \tilde{r}_i^{n+1} + (1-\eta_i) g(\boldsymbol{x}^{n+1}). \tag{3.18}$$

THEOREM 3.3. *Suppose f is L-smooth with a positive lower bound δ, and $(\mathcal{L}^n \boldsymbol{x})_i = \lambda_i^n x_i$ with $\lambda_i^n \ge 0\ \forall i, n$. Let r_i^n be generated by the scheme (2.14)–(2.16). Then we have $\lim_{n\to\infty} \boldsymbol{x}^n = \boldsymbol{x}^*$. And there exists $C_2 > 0$ such that for $\Delta t \le C_2$, we have*

$$\lim_{n\to\infty} r_i^n = r_i^* \ge \frac{\sqrt{\delta}}{2} \text{ for } 1 \le i \le m, \text{ and } \nabla f(\boldsymbol{x}^*) = 0.$$

Proof. With the assumption on $\mathcal{L}^n$, the scheme (2.14)–(2.16) is decoupled for each i, and similarly to (3.6), we can derive

$$\tilde{r}_i^{n+1} = \frac{r_i^n}{1 + 2\Delta t (1 + \Delta t \lambda_i^n)^{-1} (\partial_i g(\boldsymbol{x}^n))^2},\ i = 1, \dots, m. \tag{3.19}$$

Hence, along with (2.23), we derive that $r_i^n \ge 0$ is a decreasing sequence so that $\lim_{n\to\infty} r_i^n = r_i^*$. We only need to show that $r_i^* > 0$ for $1 \le i \le m$.

We first split $M := \{1, 2, \dots, m\}$ into I_1 and I_2, where

$$I_1 = \{i \in M : r_i^n \ge \sqrt{\delta}\ \forall n\}, \quad I_2 = M \setminus I_1. \tag{3.20}$$

Note that $\{r_i^n\}$ are decreasing sequences $\forall i$. Then for any $i \in I_1$, we can conclude that $r_i^* \ge \sqrt{\delta}$. So we only need to show that for any $i \in I_2$, $r_i^* > 0$. We can characterize I_2 as

$$I_2 = \{i \in M : \exists N_i, \text{ such that } r_i^n < \sqrt{\delta},\ \forall n \ge N_i\}. \tag{3.21}$$

For any $i \in I_2$, we have $r_i^n < \sqrt{\delta} \le g(x^n)$ for any $n \ge N$, where $N = \max_{i \in I_2} N_i$. We observe from (2.16) that r_i^n is a convex combination of $\tilde{r}_i^n$ and $g(x^n)$, so we have

$$\tilde{r}_i^n \le r_i^n \le g(\boldsymbol{x}^n). \tag{3.22}$$

From (2.23), we obtain

$$\left(x_i^{n+1}-x_i^n\right)^2 \le \frac{\Delta t}{(1-\psi)}\left((r_i^n)^2-(r_i^{n+1})^2\right),\ i=1,2,\ldots,m. \tag{3.23}$$

Taking the sum of the aforementioned for $n=0,1,2,\ldots$, we find

$$\sum_{n=0}^{\infty}\left(x_i^{n+1}-x_i^n\right)^2 \le \frac{\Delta t}{(1-\psi)}\left((r_i^0)^2-(r_i^*)^2\right),\ i=1,2,\ldots,m, \tag{3.24}$$

which implies that $\lim_{n\to\infty} x_i^n = x_i^*$, $i=1,2,\ldots,m$. Since $\tilde{r}_i^{n+1} \le r_i^n\ \forall n,i$ and $\tilde{r}_i^n \le r_i^n$ for $n \ge N$ and $i \in I_2$, we have that for $n \ge N$,

$$\begin{aligned} g(\boldsymbol{x}^{n+1}) &\le g(\boldsymbol{x}^n)+\nabla g(\boldsymbol{x}^n)\left(\boldsymbol{x}^{n+1}-\boldsymbol{x}^n\right)+\frac{L_g}{2}\|x^{n+1}-x^n\|^2 \\ &= g(\boldsymbol{x}^n)+\sum_{i=1}^m \partial_i g(\boldsymbol{x}^n)\left(x_i^{n+1}-x_i^n\right)+\frac{L_g}{2}\sum_{i=1}^m\left(x_i^{n+1}-x_i^n\right)^2 \\ &\le g(\boldsymbol{x}^n)+\sum_{i=1}^m \tilde{r}_i^{n+1}-r_i^n+\frac{\Delta t L_g}{2(1-\psi)}\sum_{i=1}^m\left((r_i^n)^2-\left(r_i^{n+1}\right)^2\right) \\ &\le g(\boldsymbol{x}^n)+\sum_{i\in I_2} \tilde{r}_i^{n+1}-r_i^n+\frac{\Delta t L_g}{2(1-\psi)}\sum_{i=1}^m\left((r_i^n)^2-\left(r_i^{n+1}\right)^2\right) \\ &\le g(\boldsymbol{x}^n)+\sum_{i\in I_2} r_i^{n+1}-r_i^n+\frac{\Delta t L_g}{2(1-\psi)}\sum_{i=1}^m\left((r_i^n)^2-\left(r_i^{n+1}\right)^2\right). \end{aligned} \tag{3.25}$$

Summing up the above from $n=N$ to K, we obtain

$$g(\boldsymbol{x}^{K+1}) \le g(\boldsymbol{x}^N)+\sum_{i\in I_2} r_i^{K+1}-r_i^N+\frac{\Delta t L_g}{2(1-\psi)}\sum_{i=1}^m\left(\left(r_i^N\right)^2-\left(r_i^{K+1}\right)^2\right). \tag{3.26}$$

Let K go to $+\infty$, and we know $g(\boldsymbol{x}^*) \le g(\boldsymbol{x}^{K+1})$, $r_i^* \le r_i^N$; then we have

$$\begin{aligned} g(\boldsymbol{x}^*) &\le g(\boldsymbol{x}^N)+\sum_{i\in I_2} r_i^*-r_i^N+\frac{\Delta t L_g}{2(1-\psi)}\sum_{i=1}^m\left(\left(r_i^N\right)^2-(r_i^*)^2\right) \\ &\le g(\boldsymbol{x}^N)+\left(\min_i r_i^*+\sum_{i\in I_2\setminus j} r_i^N-\sum_{i\in I_2} r_i^N\right)+\frac{\Delta t L_g}{2(1-\psi)}\sum_{i=1}^m\left(\left(r_i^N\right)^2-(r_i^*)^2\right) \\ &\le g(\boldsymbol{x}^N)+(\min_i r_i^*-r_j^N)+\frac{\Delta t L_g}{2(1-\psi)}\sum_{i=1}^m\left(r_i^N\right)^2, \end{aligned}$$

where $j=\operatorname{argmin}_{i\in I_2} r_i^*$. Hence,

$$r^* := r_j^* = \min_i r_i^* \ge g(\boldsymbol{x}^*)-\left(g(\boldsymbol{x}^N)-r_j^N\right)-\frac{\Delta t L_g}{2(1-\psi)}\sum_{i=1}^m\left(r_i^N\right)^2. \tag{3.27}$$

Next, we bound the distance between r_j^N and $g(\boldsymbol{x}^N)$. First, for any n and $1\le i\le m$, we have

$$\begin{aligned} r_i^n &= \eta_i \tilde{r}_i^n+(1-\eta_i)\cdot g\left(\boldsymbol{x}^n\right), \\ r_i^n-g\left(\boldsymbol{x}^n\right) &= \eta_i\tilde{r}_i^n-\eta_i g\left(\boldsymbol{x}^n\right)=\eta_i\left(\tilde{r}_i^n-g\left(\boldsymbol{x}^n\right)\right). \end{aligned}$$

Noticing that the third equation of (3.25) works $\forall n$, we have

$$g(\boldsymbol{x}^{n+1}) \le g(\boldsymbol{x}^n) + \sum_{i=1}^{m} \tilde{r}_i^{n+1} - r_i^n + \frac{\Delta t L_g}{2(1-\psi)} \sum_{i=1}^{m} \left((r_i^n)^2 - \left(r_i^{n+1}\right)^2 \right), \tag{3.28}$$

and

$$\begin{aligned}
& g\left(\boldsymbol{x}^N\right) - r_j^N \\
&= \eta_j \left(g\left(\boldsymbol{x}^N\right) - \tilde{r}_j^N \right) \\
&\le \eta_j \left(g\left(x^{N-1}\right) - r_j^{N-1} + \sum_{i \in M \setminus j} \tilde{r}_i^N - r_i^{N-1} + \frac{\Delta t L_g}{2(1-\psi)} \sum_{i=1}^{m} \left(r_i^{N-1}\right)^2 - \left(r_i^N\right)^2 \right) \\
&= \eta_j \left(\eta_j \left(g\left(\boldsymbol{x}^{N-1}\right) - \tilde{r}_j^{N-1} \right) + \sum_{i \in M \setminus j} \tilde{r}_i^N - r_i^{N-1} + \frac{\Delta t L_g}{2(1-\psi)} \sum_{i=1}^{m} \left(r_i^{N-1}\right)^2 - \left(r_i^N\right)^2 \right) \\
&\le \cdots \\
&\le \eta_j^N (g\left(\boldsymbol{x}^0\right) - r_j^0) + \sum_{k=1}^{N} \eta_j^k \left(\sum_{i \in M \setminus j} \tilde{r}_i^k - r_i^{k-1} + \frac{\Delta t L_g}{2(1-\psi)} \sum_{i=1}^{m} \left(r_i^{k-1}\right)^2 - \left(r_i^k\right)^2 \right) \\
&\le \sum_{k=1}^{N} \eta_j^k \left(\sum_{i \in M \setminus j} \Delta t \tilde{r}_i^k (\partial_i g(\boldsymbol{x}^{k-1}))^2 + \frac{\Delta t L_g}{2(1-\psi)} \sum_{i=1}^{m} \left(r_i^{k-1}\right)^2 - \left(r_i^k\right)^2 \right) \\
&= \Delta t \sum_{k=1}^{N} \eta_j^k \left(\sum_{i \in M \setminus j} \tilde{r}_i^k (\partial_i g(\boldsymbol{x}^{k-1}))^2 + \frac{L_g}{2(1-\psi)} \sum_{i=1}^{m} \left(r_i^{k-1}\right)^2 - \left(r_i^k\right)^2 \right) \\
&=: \Delta t C_N.
\end{aligned}$$

We can obtain $g\left(\boldsymbol{x}^N\right) - r_j^N \ge 0$ from (3.22) and thus $C_N \ge 0$. Hence,

$$r^* = \min_i r_i^* \ge g(\boldsymbol{x}^*) - \Delta t \left(C_N + \frac{L_g}{2(1-\psi)} \sum_{i=1}^{m} \left(r_i^N\right)^2 \right). \tag{3.29}$$

Let $C_2 = \frac{1}{2} g(\boldsymbol{x}^*) / (C_N + \frac{L_g}{2(1-\psi)} \sum_{i=1}^{m} (r_i^N)^2)$. Then, $\forall \Delta t \le C_2$, we have

$$r^* = \min_i r_i^* \ge \frac{1}{2} g(\boldsymbol{x}^*) \ge \frac{\sqrt{\delta}}{2} > 0. \tag{3.30}$$

Finally, letting $n \to \infty$ in (2.14), we derive $\partial_i f(\boldsymbol{x}^*) = 0$ for $i = 1, \ldots, m$, which implies $\nabla f(\boldsymbol{x}^*) = 0$. The proof is complete. ∎

3.3. Dissipation of the original energy. We showed in section 2 the modified energy of the RVAV approach remains dissipative. Next, we show that when $\mathcal{L} \equiv 0$, the original energy of RVAV is also dissipative when the step size Δt is sufficiently small.

THEOREM 3.4. *Assuming that f is L-smooth and has a positive lower bound δ, then the solution $\boldsymbol{x}^{n+1}$ of the RVAV scheme with $\mathcal{L} = 0$ satisfies the discrete dissipation law $f(\boldsymbol{x}^{n+1}) \le f(\boldsymbol{x}^n)$ with $\Delta t \le \min(C_2, \frac{\delta^{\frac{3}{2}}}{L f(\boldsymbol{x}^0)})$ and $\lim_{n\to\infty} f(\boldsymbol{x}^n) = f^*$.*

Proof. Denoting $\xi_i^n = \frac{r_i^{n+1}}{\sqrt{f(\boldsymbol{x}^n)}} = \frac{1}{\sqrt{f(x^n)}}(1+\Delta t\frac{(\partial_i f(\boldsymbol{x}^n))^2}{2f(\boldsymbol{x}^n)})^{-1} r_i^n$ and $\boldsymbol{\xi}^n = (\xi_1^n, \xi_2^n, \ldots, \xi_m^n)'$, we have

$$
\begin{aligned}
f(\boldsymbol{x}^{n+1}) &= f(\boldsymbol{x}^n - \Delta t \boldsymbol{\xi}^n \odot \nabla f(\boldsymbol{x}^n)) \\
&= f(\boldsymbol{x}^n) - \Delta t \left(\boldsymbol{\xi}^n \odot \nabla f(\boldsymbol{x}^n)\right)^T \nabla f(\boldsymbol{x}^n) \\
&\quad + \frac{1}{2}\left(\Delta t \boldsymbol{\xi}^n \odot \nabla f(\boldsymbol{x}^n)\right)^T \nabla^2 f(z) \left(\Delta t \boldsymbol{\xi}^n \odot \nabla f(\boldsymbol{x}^n)\right).
\end{aligned} \tag{3.31}
$$

Noticing that $\xi_{min}^n \|\nabla f(x^n)\|^2 \le (\boldsymbol{\xi}^n \odot \nabla f(\boldsymbol{x}^n))^T \nabla f(x^n) \xi_{max}^n \|\nabla f(x^n)\|^2$ where $\xi_{min}^n = \min(\boldsymbol{\xi}^n)$ and $\xi_{max}^n = \max(\boldsymbol{\xi}^n)$, we have

$$
f(\boldsymbol{x}^{n+1}) \le f(\boldsymbol{x}^n) - \Delta t \xi_{min}^n \|\nabla f(\boldsymbol{x}^n)\|^2 + \frac{L}{2}(\Delta t \xi_{max}^n)^2 \|\nabla f(\boldsymbol{x}^n)\|^2. \tag{3.32}
$$

If $\Delta t \le \frac{2\xi_{min}^n}{L(\xi_{max}^n)^2}$, then $f(\boldsymbol{x}^{n+1}) \le f(\boldsymbol{x}^n)$. Furthermore, utilizing the positive lower bound r^* in Theorem 3.3 and $r_i^0 = \sqrt{f(\boldsymbol{x}^0)}\ \forall i$, we can have

$$
\frac{2\xi_{min}^n}{L(\xi_{max}^n)^2} = \frac{2(r_i^{n+1})_{min}}{L(r_i^{n+1})_{max}^2}\sqrt{f(\boldsymbol{x}^n)} \ge \frac{2\delta}{L}\frac{(r_i^{n+1})_{min}}{(r_i^{n+1})_{max}^2} \ge \frac{2\delta}{L}\frac{(r_i^{n+1})_{min}}{(r_i^0)_{max}^2} \ge \frac{\delta^{\frac{3}{2}}}{Lf(\boldsymbol{x}^0)}.
$$

Therefore, if $\Delta t \le \min(C_2, \frac{\delta^{\frac{3}{2}}}{Lf(\boldsymbol{x}^0)})$, we can ensure $f(\boldsymbol{x}^{n+1}) \le f(\boldsymbol{x}^n)$. ∎

3.4. Convergence analysis of the RVAV. We first recall the following lemma.

LEMMA 3.5 (cf. [3]). *f satisfies the Polyak–Łojasiewicz inequality if there exists a $\mu > 0$ such that*

$$
\mu\left(f(\boldsymbol{x}) - f^*\right) \le \frac{1}{2}\|\nabla f(\boldsymbol{x})\|^2, \tag{3.33}
$$

where $f^* = f(\boldsymbol{x}^*)$ and $\boldsymbol{x}^*$ is the optimal point.

THEOREM 3.6. *Let a sequence $\{\boldsymbol{x}^n\}$ be generated by the RVAV with $\mathcal{L} \equiv 0$. Suppose f satisfying the Polyak–Łojasiewicz inequality and being bounded from below by positive constant δ; then for any $\gamma < \frac{\delta^2}{8L(f(\boldsymbol{x}^0))^2}$, there exists $\Delta t_1, \Delta t_2 > 0$ such that if $\Delta t_1 \le \Delta t \le \min(C_2, \frac{\delta^{\frac{3}{2}}}{Lf(\boldsymbol{x}^0)}, \Delta t_2)$, we have*

$$
f(\boldsymbol{x}^{n+1}) - f^* \le (1 - 2\mu\epsilon_n)\left(f(\boldsymbol{x}^n) - f^*\right), \tag{3.34}
$$

where $\nu > 0$ is the constant in (3.33) and $\gamma \le \epsilon_n \le \frac{1}{2L}$.

Proof. Subtracting f^* from both sides of inequality (3.32) in Theorem 3.4 and using Lemma 3.5, we have

$$
\begin{aligned}
f(\boldsymbol{x}^{n+1}) - f^* &\le f(\boldsymbol{x}^n) - f^* - \left(\Delta t \xi_{min}^n - \frac{L}{2}(\Delta t \xi_{max}^n)^2\right)\|\nabla f(\boldsymbol{x}^n)\|^2 \\
&\le f(\boldsymbol{x}^n) - f^* - 2\mu\left(\Delta t \xi_{min}^n - \frac{L}{2}(\Delta t \xi_{max}^n)^2\right)\left(f(\boldsymbol{x}^n) - f^*\right) \\
&=: (1 - 2\mu\epsilon_n)\left(f(\boldsymbol{x}^n) - f^*\right),
\end{aligned}
$$

where $\epsilon_n = \Delta t \xi_{min}^n - \frac{L}{2}(\Delta t \xi_{max}^n)^2$. To achieve the linear convergence rate, it is necessary that $\epsilon_n \in (0, 1)$.

An upper bound on ϵ_n can be obtained directly, as it is a quadratic function of Δt that achieves its maximum value at $\Delta t = \frac{\xi^n_{min}}{L(\xi^n_{max})^2}$.

$$\epsilon_n \leq \frac{\xi^n_{min}}{L(\xi^n_{max})^2}\xi^n_{min} - \frac{L}{2}\left(\frac{\xi^n_{min}}{L(\xi^n_{max})^2}\right)^2 (\xi^n_{max})^2 = \frac{(\xi^n_{min})^2}{2L(\xi^n_{max})^2} \leq \frac{1}{2L}.$$

To obtain a lower bound, we can rewrite ϵ_n as follows:

$$\epsilon_n = \Delta t \frac{(r_i^{n+1})_{min}}{\sqrt{f(\boldsymbol{x}^n)}} - \frac{L}{2}\left(\Delta t \frac{(r_i^{n+1})_{max}}{\sqrt{f(\boldsymbol{x}^n)}}\right)^2.$$

From Theorem 3.4, we can obtain $\epsilon_n \geq 0$ if $\Delta t \leq \min(C_2, \frac{\delta^{\frac{3}{2}}}{Lf(\boldsymbol{x}^0)})$. Clearly, if we need $\epsilon_n \geq \gamma > 0$, we need a tighter bound on Δt. Since $f(\boldsymbol{x}^{n+1}) \leq f(\boldsymbol{x}^n)$ and $r_i^0 = \sqrt{f(\boldsymbol{x}^0)}$ $\forall i$, we obtain

$$\epsilon_n \geq \Delta t \frac{r^*}{\sqrt{f(\boldsymbol{x}^0)}} - \frac{L}{2}\left(\Delta t \frac{(r_i^0)_{max}}{\sqrt{\delta}}\right)^2 \geq \Delta t \frac{\sqrt{\delta}}{2\sqrt{f(\boldsymbol{x}^0)}} - \frac{Lf(\boldsymbol{x}^0)}{2\delta}\Delta t^2 := h(\Delta t).$$

Denoting $\omega = \frac{\sqrt{\delta}}{\sqrt{f(\boldsymbol{x}^0)}}$, we can rewrite h as

$$h(\Delta t) = -\frac{L}{2\omega^2}\Delta t^2 + \frac{\omega}{2}\Delta t.$$

To obtain $h(\Delta t) \geq \gamma > 0$, we need

$$\frac{\omega^3 - \sqrt{\omega^6 - 8\gamma L\omega^2}}{2L} \leq \Delta t \leq \frac{\omega^3 + \sqrt{\omega^6 - 8\gamma L\omega^2}}{2L},$$

and $\omega^6 - 8\gamma L\omega^2 \geq 0$ which requires $\gamma \leq \frac{\omega^4}{8L} = \frac{\delta^2}{8L(f(\boldsymbol{x}^0))^2}$. Hence, setting $\Delta t_1 = \frac{\omega^3 - \sqrt{\omega^6 - 8\gamma L\omega^2}}{2L}$ and $\Delta t_2 = \frac{\omega^3 + \sqrt{\omega^6 - 8\gamma L\omega^2}}{2L}$, we can easily show that if $\Delta t_1 \leq \Delta t \leq \min(C_2, \frac{\delta^{\frac{3}{2}}}{Lf(\boldsymbol{x}^0)}, \Delta t_2)$, then $\gamma \leq \epsilon_n \leq \frac{1}{2L}$. ∎

Remark 3.7. An analogous result can be attained for the RSAV scheme (see also [15]).

4. Enhanced convergence rates. We showed in the last section that the RVAV algorithm exhibits a linear convergence rate. It is known that the selection of step size is a crucial factor in GD algorithms, as demonstrated by the superlinear convergence rate of the secant method with step size $\Delta t = \frac{x_n - x_{n-1}}{f'(x_n) - f'(x_{n-1})}$, and the quadratic convergence rate of the Newton method with step size $\frac{1}{f''(x_n)}$ for univariate optimization problems. In this section, we demonstrate that by selecting an appropriate step size, the linear convergence rate of VAV and RVAV can be enhanced to achieve a superlinear convergence rate in the univariate case and present an adaptive version of RVAV which accelerates the convergence rate of RVAV in the multivariate case. In the univariate case, the terms VAV and RVAV are equivalent to SAV and RSAV, respectively. To maintain consistency and clarity in the transition to the multivariate case, we shall continue to refer to them as VAV and RVAV in the following proof, despite their equivalence to SAV and RSAV.

4.1. Superlinear convergence rate of univariate VAV and RVAV. For the sake of simplifying the presentation, we consider the VAV scheme with $\mathcal{L} \equiv 0$ in the univariate case (note that similar results can be achieved for the RVAV scheme by substituting r^{n+1} with $\tilde{r}^{n+1}$):

$$\frac{x^{n+1}-x^n}{\Delta t} = -\frac{r^{n+1}}{\sqrt{f(x^n)}} f'(x^n), \tag{4.1}$$

$$\frac{r^{n+1}-r^n}{\Delta t} = \frac{1}{2\sqrt{f(x^n)}} f'(x^n) \frac{x^{n+1}-x^n}{\Delta t}. \tag{4.2}$$

We can derive from the above that $r^{n+1} = \frac{1}{1+\Delta t \frac{f'(x^n)^2}{2f(x^n)}} r^n = \frac{2f(x^n)}{2f(x^n)+\Delta t f'(x^n)^2} r^n$. Then, VAV can be rewritten as the following iterative method:

$$x^{n+1} = x^n - \Delta t \frac{2\sqrt{f(x^n)} r^n}{2f(x^n)+\Delta t f'(x^n)^2} f'(x^n) \tag{4.3}$$

with the learning rate $\eta_n = \Delta t \frac{2\sqrt{f(x^n)} r^n}{2f(x^n)+\Delta t f'(x^n)^2}$.

Assuming that x^* is the optimal point and denoting $\varepsilon_n = x^n - x^*$, we subtract x^* from both sides of (4.3) to derive

$$\begin{aligned} \varepsilon_{n+1} &= \varepsilon_n - \Delta t \frac{2\sqrt{f(x^n)} r^n}{2f(x^n)+\Delta t f'(x^n)^2} f'(x^n) \\ &= \frac{\varepsilon_n \left(2f(x^n)+\Delta t f'(x^n)^2\right) - 2\Delta t \sqrt{f(x^n)} r^n f'(x^n)}{2f(x^n)+\Delta t f'(x^n)^2}. \end{aligned} \tag{4.4}$$

Applying the Taylor expansion to f' around x^n, we derive

$$0 = f'(x^*) = f'(x^n) - f''(x^n)\varepsilon_n + \frac{f'''(\xi_n^*)}{2}\varepsilon_n^2, \tag{4.5}$$

where ξ_n^* lies between x^n and x^*. Substituting this expression into the numerator of the second equation in (4.4) yields

$$\begin{aligned} \varepsilon_{n+1} &= \frac{\varepsilon_n \left(2f(x^n)+\Delta t f'(x^n)^2\right) - 2\Delta t \sqrt{f(x^n)} r^n f'(x^n)}{2f(x^n)+\Delta t f'(x^n)^2} \\ &= \frac{\varepsilon_n \left(2f(x^n)+\Delta t \left(f''(x^n)\varepsilon_n - \frac{f'''(\xi_n^*)}{2}\varepsilon_n^2\right)^2\right)}{2f(x^n)+\Delta t f'(x^n)^2} \\ &\quad - \frac{2\Delta t \sqrt{f(x^n)} r^n \left(f''(x^n)\varepsilon_n - \frac{f'''(\xi_n^*)}{2}\varepsilon_n^2\right)}{2f(x^n)+\Delta t f'(x^n)^2} \\ &= \frac{\varepsilon_n \left(2f(x^n) - 2\Delta t \sqrt{f(x^n)} r^n f''(x^n)\right)}{2f(x^n)+\Delta t f'(x^n)^2} \\ &\quad + \frac{\Delta t \varepsilon_n^2 \left(\left(f''(x^n) - \frac{f'''(\xi_n^*)}{2}\varepsilon_n\right)^2 + 2\sqrt{f(x^n)} r^n \frac{f'''(\xi_n^*)}{2}\right)}{2f(x^n)+\Delta t f'(x^n)^2}. \end{aligned} \tag{4.6}$$

A straightforward approach to obtaining a quadratically convergent algorithm is to set $\Delta t = \frac{\sqrt{f(x^n)}}{r^n} \frac{1}{f''(x^n)}$. However, since computing second-order derivatives can be costly or may not be possible, we can instead set

$$\Delta t = \frac{\sqrt{f(x^n)}}{r^n} \frac{x^n - x^{n-1}}{f'(x^n) - f'(x^{n-1})}. \tag{4.7}$$

Then by taking Taylor expansion of $f'(x^{n-1})$ about x^n,

$$f'\left(x^{n-1}\right) = f'\left(x^n\right) + f''\left(x^n\right)\left(\varepsilon_{n-1} - \varepsilon_n\right) + \frac{f^{(3)}(\xi_k^{\ddagger})}{2}\left(\varepsilon_{n-1} - \varepsilon_n\right)^2, \tag{4.8}$$

where $\xi_n^{\ddagger}$ lies between x^{n-1} and x^n, we can rewrite Δt as follows:

$$\Delta t = \frac{\sqrt{f(x^n)}}{r^n} \frac{1}{f''\left(x^n\right) + \frac{1}{2} f^{(3)}(\xi_k^{\ddagger})\left(\varepsilon_{n-1} - \varepsilon_n\right)}. \tag{4.9}$$

By substituting the expression for Δt from (4.9) into the first term of the numerator in the last equation of (4.6), we obtain

$$\begin{aligned} 2f(x^n) - 2\Delta t\sqrt{f(x^n)} r^n f''(x^n) &= 2f(x^n)\left(1 - \frac{f''(x^n)}{f''\left(x^n\right) + \frac{1}{2} f^{(3)}(\xi_k^{\ddagger})\left(\varepsilon_{n-1} - \varepsilon_n\right)}\right) \\ &= f(x^n)\left(\frac{f^{(3)}(\xi_k^{\ddagger})\left(\varepsilon_{n-1} - \varepsilon_n\right)}{f''\left(x^n\right) + \frac{1}{2} f^{(3)}(\xi_k^{\ddagger})\left(\varepsilon_{n-1} - \varepsilon_n\right)}\right). \end{aligned}$$

Then the last equation of (4.6) can be expressed as

$$\begin{aligned} \varepsilon_{n+1} = &\frac{\varepsilon_n f(x^n)\left(\frac{f^{(3)}(\xi_k^{\ddagger})(\varepsilon_{n-1} - \varepsilon_n)}{f''(x^n) + \frac{1}{2} f^{(3)}(\xi_k^{\ddagger})(\varepsilon_{n-1} - \varepsilon_n)}\right)}{2f(x^n) + \Delta t f'(x^n)^2} \\ &+ \frac{\Delta t \varepsilon_n^2 \left(\left(f''(x^n) - \frac{f'''(\xi_n^*)}{2}\varepsilon_n\right)^2 + 2\sqrt{f(x^n)} r^n \frac{f'''(\xi_n^*)}{2}\right)}{2f(x^n) + \Delta t f'(x^n)^2}. \end{aligned}$$

We can then derive that

$$\lim_{n\to\infty} \frac{|\varepsilon_n|}{|\varepsilon_{n-1}|} = 0,$$

and then

$$\lim_{n\to\infty} \frac{|\varepsilon_{n+1}|}{|\varepsilon_n \varepsilon_{n-1}|} = \left(\frac{f^{(3)}\left(x^*\right)}{2f\left(x^*\right) f^{(2)}\left(x^*\right)}\right)^2. \tag{4.10}$$

To determine the order of convergence, we assume an asymptotic relationship $|\varepsilon_{n+1}| \approx A\,|\varepsilon_n|^p$, where A is a constant, and p is the order of convergence. Then we have

$$|\varepsilon_n| \approx A\,|\varepsilon_{n-1}|^p, \quad |\varepsilon_{n-1}| \approx A^{-1/p}\,|\varepsilon_n|^{1/p}.$$

Plugging the assumption and the above into (4.10), we will have

$$\lim_{n\to\infty} \frac{|\varepsilon_{n+1}|}{|\varepsilon_n \varepsilon_{n-1}|} = \lim_{n\to\infty} \frac{A\,|\varepsilon_n|^p}{A^{-1/p}\,|\varepsilon_n|^{1+1/p}} = \left(\frac{f^{(3)}\left(x^*\right)}{2f\left(x^*\right) f^{(2)}\left(x^*\right)}\right)^2. \tag{4.11}$$

This implies $p = 1 + 1/p$, from which we deduce that the convergence rate of the modified algorithm with Δt given by (4.7) is $\frac{1+\sqrt{5}}{2}$.

4.2. ARVAV with Steffensen step size. To fully take advantage of the unconditional energy dissipation of RVAV, we propose below an adaptive version of the algorithm, ARVAV. By carefully selecting the step size, we can accelerate the algorithm's performance. We introduce an indicator $\alpha = \text{mean}(\frac{\boldsymbol{r}^n}{\sqrt{E^n}})$, which demonstrates the ratio of the modified and the original energy. When the ratio is close to 1, we continue to use the current step size; when it deviates significantly from 1, it suggests that the step size needs to be reduced.

For further enhancement in efficacy, we should also consider using a suitable accelerated method. However, the accelerated method discussed above is only applicable in the univariate case, as extending (4.7) to the multivariate case is not straightforward. Nevertheless, this method provides us with insight that RVAV can be accelerated by choosing an appropriate step size. On the other hand, Steffensen's acceleration method [24, 25] is applicable to multivariate cases, so we will adopt the Steffensen step size [31] and introduce our adaptive step size:

$$\Delta t_n = \frac{\phi_n \left\|\nabla f\left(\boldsymbol{x}_n\right)\right\|^2}{\left[\nabla f\left(\boldsymbol{x}_n + \nabla f\left(\boldsymbol{x}_n\right)\right) - \nabla f\left(\boldsymbol{x}_n\right)\right]^\top \nabla f\left(\boldsymbol{x}_n\right)}, \tag{4.12}$$

where $\phi_n = \frac{\sqrt{f(\boldsymbol{x}^n)}}{r^n} \frac{\|\boldsymbol{x}^n - \boldsymbol{x}^{n-1}\|^2}{(\nabla f(\boldsymbol{x}^n) - \nabla f(\boldsymbol{x}^{n-1}))^T (\boldsymbol{x}^n - \boldsymbol{x}^{n-1})}$. We observe that (4.12) captures a significant amount of gradient information and is similar to (4.7) but applicable to multivariate cases.

Building on the aforementioned premise, we put forth the ARVAV algorithm, Algorithm 4.2: By incorporating Steffensen's step size in the ARVAV algorithm, we can improve its performance. However, we should also consider the computational cost of the Steffensen method and balance it against the potential performance gains. In practice, the decision of whether to use the Steffensen method may depend on the specific problem and available computational resources.

Algorithm 4.1. RVAV with enhanced convergence in univariate case.

Given a starting point $x^0 \in \mathbf{dom} f$, a step size Δt, $r^0 = \sqrt{f(x^0)}$ and set $n = 0$, $\psi \in (0, 1)$.
Compute x^1 and r^1 with Algorithm 2.1 and update $n = 1$.
while the termination condition is not met **do**
  Set $\Delta t = \frac{\sqrt{f(x^n)}}{r^n} \frac{x^n - x^{n-1}}{f'(x^n) - f'(x^{n-1})}$
  Compute $\tilde{r}^{n+1} = \frac{2f(x^n)}{2f(x^n) + \Delta t f'(x^n)^2} r^n$
  Update $x^{n+1} = x^n - \Delta t \frac{\tilde{r}^{n+1}}{\sqrt{f(x^n)}} f'(x^n)$
  Set $r^{n+1} = \eta \tilde{r}^{n+1} + (1 - \eta)\sqrt{f(x^{n+1})}$
  Compute $\eta_0 = \min_{\eta \in [0,1]} \eta$, such that $(r^{n+1})^2 - (\tilde{r}^{n+1})^2 \le \frac{\psi}{\Delta t} \left(x^{n+1} - x^n\right)^2$
  Update $r^{n+1} = \eta_0 \tilde{r}^{n+1} + (1 - \eta_0)\sqrt{f(x^{n+1})}$
  Update $n = n + 1$
end while
return x^{n+1}

Algorithm 4.2. ARVAV.

Given a starting point $\boldsymbol{x}^0 \in \mathbf{dom} f$, a step size Δt_0, $r^0 = \sqrt{f(\boldsymbol{x}^0)}$, the indicator threshold $\beta = 0.1$ and set $n = 0$, $\psi \in (0,1)$.
while the termination condition is not met **do**
Compute the indicator $\alpha = \text{mean}(\frac{\boldsymbol{r}^n}{\sqrt{E^n}})$
if $|1-\alpha| > \beta$ **then**
$\Delta t_n = \frac{\sqrt{f(\boldsymbol{x}^n)}}{r^n} \frac{\|\boldsymbol{x}^n - \boldsymbol{x}^{n-1}\|^2}{(\nabla f(\boldsymbol{x}^n) - \nabla f(\boldsymbol{x}^{n-1}))^T(\boldsymbol{x}^n - \boldsymbol{x}^{n-1})} \frac{\|\nabla f(\boldsymbol{x}_n)\|^2}{[\nabla f(\boldsymbol{x}_n + \nabla f(\boldsymbol{x}_n)) - \nabla f(\boldsymbol{x}_n)]^\top \nabla f(\boldsymbol{x}_n)}$
end if
Compute $\tilde{r}_i^{n+1} = (1 + \frac{\Delta t_n}{2(1+\Delta t_n \lambda_i^n) f(\boldsymbol{x}^n)} (\frac{\partial f(\boldsymbol{x}^n)}{\partial x_i})^2)^{-1} r_i^n$ for $i = 1, \ldots, m$
Update $x_i^{n+1} = x_i^n - \Delta t_n (1 + \Delta t_n \lambda_i^n)^{-1} \frac{\tilde{r}_i^{n+1}}{\sqrt{f(\boldsymbol{x}^n)}} \frac{\partial f(\boldsymbol{x}^n)}{\partial x_i}$ for $i = 1, \ldots, m$
Set $r_i^{n+1} = \eta_i \tilde{r}_i^{n+1} + (1 - \eta_i)\sqrt{f(\boldsymbol{x}^{n+1})}$ for $i = 1, \ldots, m$
Compute $\eta_{i0} = \min_{\eta_i \in [0,1]} \eta_i$, such that $(r_i^{n+1})^2 - (\tilde{r}_i^{n+1})^2 \le \frac{\psi}{\Delta t_n}(x_i^{n+1} - x_i^n)^2$ for $i = 1, \ldots, m$
Update $r_i^{n+1} = \eta_{i0} \tilde{r}_i^{n+1} + (1 - \eta_{i0})\sqrt{f(\boldsymbol{x}^{n+1})}$ for $i = 1, \ldots, m$
Update $n = n + 1$
end while
return $\boldsymbol{x}^{n+1}$

5. Experimental results.

5.1. Convex functions.

Consider the following minimization problem:

$$\min f(\boldsymbol{x}) = \sum_{i=1}^{N/2} x_{2i-1}^2 + \frac{1}{N} \sum_{i=1}^{N/2} x_{2i}^2, \tag{5.1}$$

where $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$. Consider the case where $N = 100$. In this scenario, $f(\boldsymbol{x}) = \sum_{k=1}^{50} x_{2k-1}^2 + \frac{1}{100} \sum_{k=1}^{50} x_{2k}^2$. The function $f(\boldsymbol{x})$ is obviously convex. However, the condition number of its Hessian matrix $\mathcal{H}$ is N. For large N, the Hessian matrix will have a poor condition number, which makes it difficult for the GD method to converge. This is because GD methods are sensitive to the step size, and a poorly conditioned Hessian matrix can cause the method to oscillate or converge slowly.

We consider two variants of our proposed method: RVAV and RVAVL. RVAV corresponds to $\mathcal{L} = 0$, while RVAVL corresponds to $\mathcal{L} =$ diagonal of $\mathcal{H}$. A constant C of 0.1 is added to $f(\boldsymbol{x})$ to prevent the function value from being negative or zero, as required in the SAV and VAV methods. To evaluate their performance under various step sizes, we compare them with three existing optimization methods, GD, RSAV, and VAV, using the initial condition $\boldsymbol{x}_0 = (1, 1, \ldots, 1)$. Table 1 presents the loss values obtained by each method after 1000 iterations.

Among the methods considered, VAV demonstrates superior performance with small step sizes ($\Delta t = 0.1$ and $\Delta t = 1$). However, as the step size increases to $\Delta t = 10$ and $\Delta t = 20$, RVAV and RVAVL outperform other methods. This indicates that the relaxed strategy, employed in RVAV and RVAVL, plays a crucial role in achieving faster and more accurate convergence when the step size is moderate.

We also examine the performance of each method at its respective best step size. The loss curves are presented in Figure 1. At each best step size, RVAV and RVAVL consistently outperform the other methods in terms of minimizing loss. Additionally, RVAVL can accelerate convergence by utilizing information from the Hessian matrix.

TABLE 1

The loss of the convex function $f(\boldsymbol{x})$. The table presents the loss of $f(\boldsymbol{x})$ after 1000 iterations, which is computed by $|f(\boldsymbol{x})-f(\boldsymbol{x}^*)|$. The global minimum $\boldsymbol{x}^*$ is $\mathbf{0}$, and $f(\boldsymbol{x}^*)=0$. "NAN" represents the method blowing up after some iterations.

Loss	GD	RSAV	VAV	RVAV	RVAVL
$\Delta t=0.1$	0.0091	0.0092	0.0053	0.0091	0.0092
$\Delta t=1$	50.0	4.33×10^{-18}	9.78×10^{-22}	5.61×10^{-15}	3.15×10^{-18}
$\Delta t=10$	NAN	0.9059	0.0069	8.94×10^{-110}	2.18×10^{-159}
$\Delta t=20$	NAN	1.6401	1.65×10^{4}	0	0
$\Delta t=30$	NAN	0.4078	4.05×10^{4}	0	0

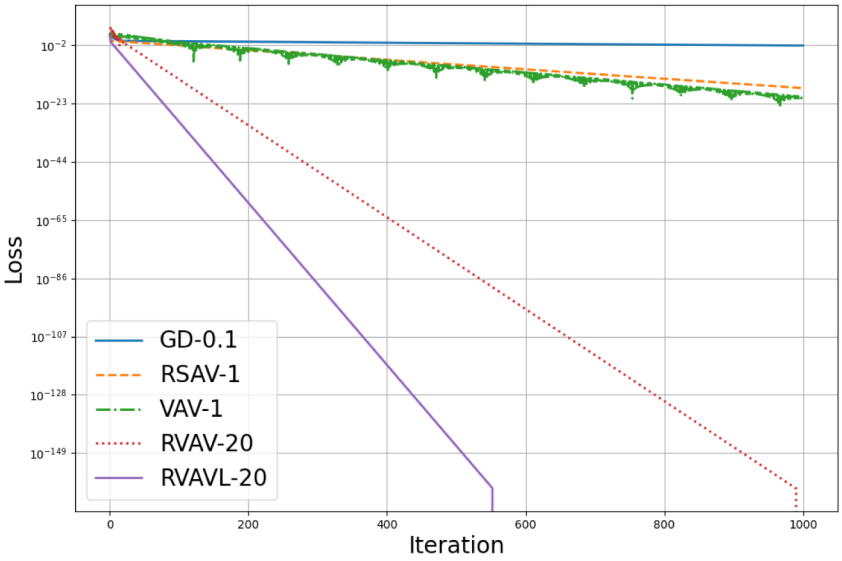


FIG. 1. *The loss of the convex function $f(\boldsymbol{x})$ while changing the number of iterations at different step sizes. It shows the comparison of loss curves for different optimization methods at their respective best step sizes, for example, the best step size for GD is $\Delta t=0.1$. It also illustrates the loss curves for GD, RSAV, VAV, RVAV, and RVAVL on the quadratic function. RVAV and RVAVL consistently outperform the other methods, achieving lower loss values at each best step size.*

5.2. Nonconvex functions. We demonstrated the superiority of RVAV and RVAVL over GD, RSAV, and VAV for convex functions. To further test the performance of RVAV and RVAVL, we consider a nonconvex Rosenbrock function and compare them with GD, RSAV, and VAV. The objective function is given by

$$f(x_1,x_2)=(1-x_1)^2+100\left(x_2-x_1^2\right)^2 \tag{5.2}$$

with the global minimum at $x^*=(1,1)$ and the minimal value of $\mathbf{0}$. The initial point for the numerical experiment was set to $(-2,-4)$. In this example, we consider RVAVL where $\mathcal{L}$ is set as a diagonal matrix λI, with $\lambda=100$. We conduct a performance comparison using a small step size of $\Delta t=0.0015$ and a larger step size of $\Delta t=0.01$. Figures 2 and 3 illustrate the error curves over 20000 iterations. With both step sizes, RVAV and RVAVL outperform the other methods. With a small step size, RVAVL shows little difference from RVAV. However, with a larger step size, only RVAVL attains the optimal point. Choosing an appropriate $\mathcal{L}$ as a diagonal matrix, λI, indeed helps RVAV approach the optimal point with a larger step size.

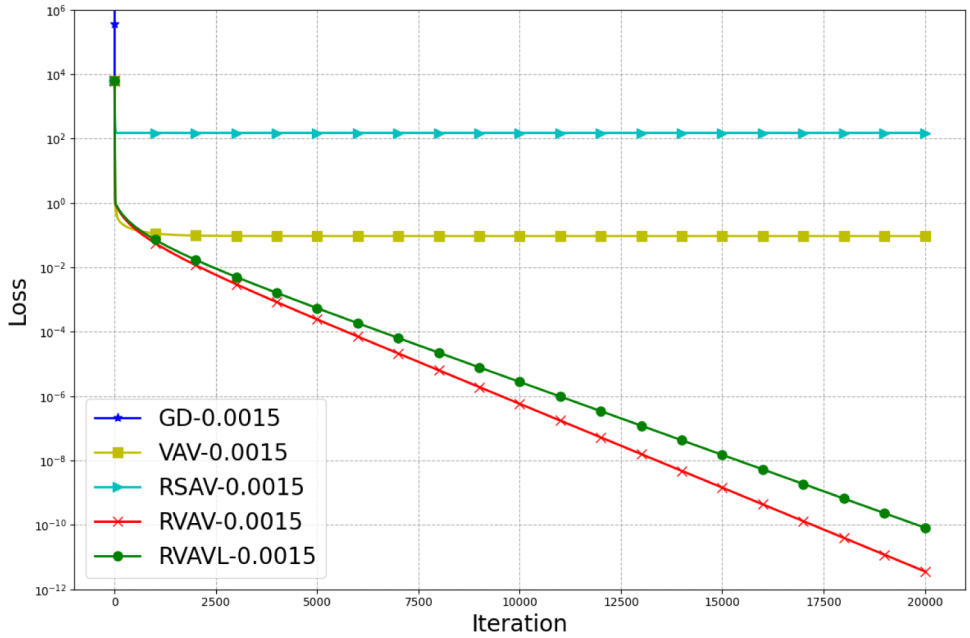


FIG. 2. *Nonconvex function: The figure shows the loss,* $|f(\boldsymbol{x}) - f(\boldsymbol{x}^*)|$*, of GD, VAV, RSAV, RVAV, and RVAVL while changing the number of iterations under the step size* 0.0015.

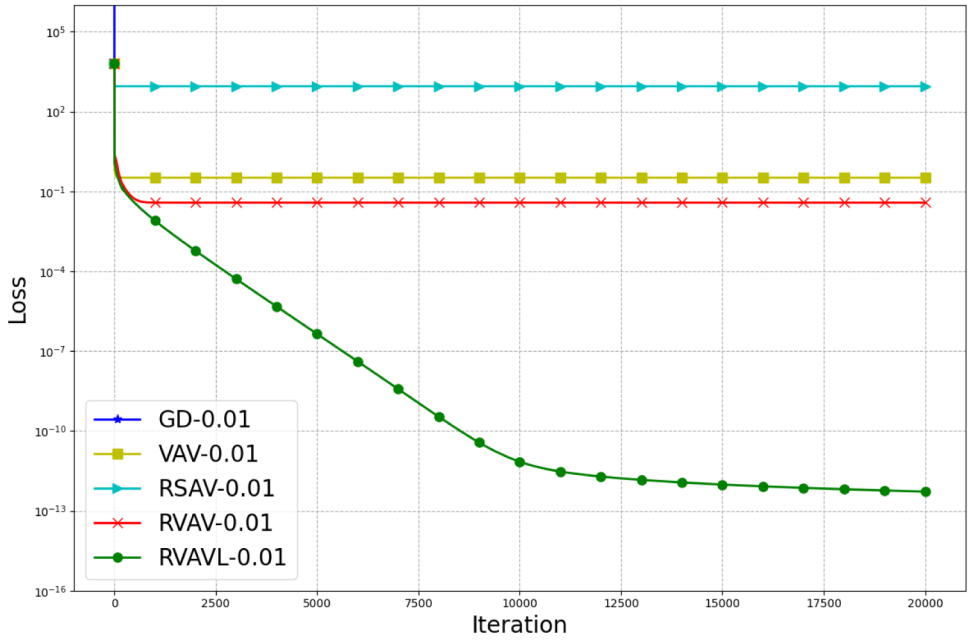
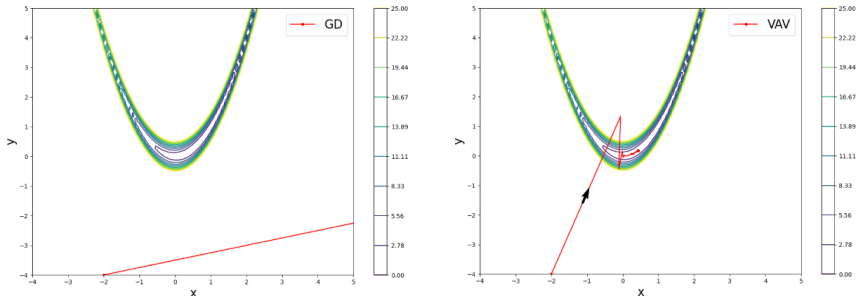
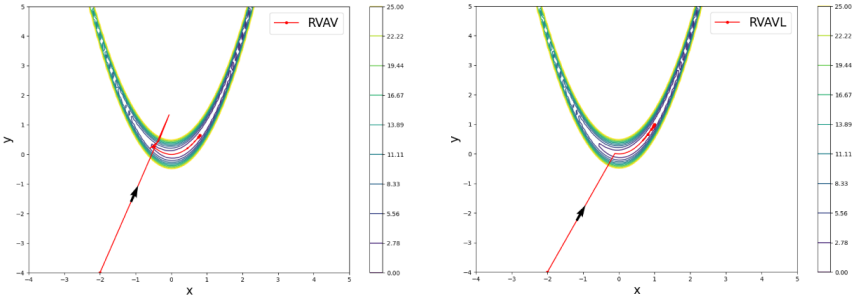


FIG. 3. *Nonconvex function: The figure shows the loss,* $|f(\boldsymbol{x}) - f(\boldsymbol{x}^*)|$*, of GD, VAV, RSAV, RVAV, and RVAVL while changing the number of iterations under the step size* $\Delta t = 0.01$.

Furthermore, we plot the trajectories of GD, VAV, RVAV, and RVAVL using $\Delta t = 0.01$ in Figure 4, with markers placed every 500 steps. The trajectories reveal that GD's iteration deviates in the wrong direction, while VAV and RVAV exhibit



(a) Trajectory of GD. (b) Trajectory of VAV.



(c) Trajectory of RVAV. (d) Trajectory of RVAVL.

FIG. 4. *Trajectories of iterative points of the nonconvex function: The trajectories of iterative points of a nonconvex function are shown, with the optimal point being* (1,1). *The red line depicts the trajectory of GD, VAV, RVAV, and RVAVL, while the black arrow indicates the direction of iterative points. (Color figures are online.)*

oscillations during iterations and get stuck near the optimal points. In contrast, RVAVL accurately approaches the optimal point rapidly.

5.3. Burgers' equations. Next, we explore using physics-informed neural networks (PINN) [17] to solve Burgers' equation with ARVAV. Applying ARVAV to PINN can lead to highly accurate solutions of Burgers' equation. We consider the Burgers' equation

$$
\begin{aligned}
&u_t + (uu_x) - \left(\frac{0.01}{\pi} u_{xx}\right) = 0, \\
&u(x,0) = \sin(\pi x), \quad x \in [-1,1].
\end{aligned} \tag{5.3}
$$

Our implementation of PINN consists of a simple dense network with 8 hidden layers, 20 neurons in each layer, and a total of 3441 trainable parameters. The activation function used is tanh.

To demonstrate the effectiveness of ARVAV, we compare its performance to that of GD, SAV, RSAV, and the adaptive version of RSAV using the default step size $\Delta t = 0.01$. With ARVAV, we can use a larger step size, and in this case, we use $\Delta t = 0.05$. Our goal is to show that ARVAV works better even with a larger step size.

In Figure 5, we compare the training loss during the iteration. We observe that ARVAV is much more stable compared to GD, SAV, RSAV, and RSAV-Adaptive as GD displays large oscillations and others appear to be stuck at a certain point. Additionally, the final training loss of ARVAV is substantially smaller than that of other methods. Figure 6 shows the comparison between the reference solution obtained by

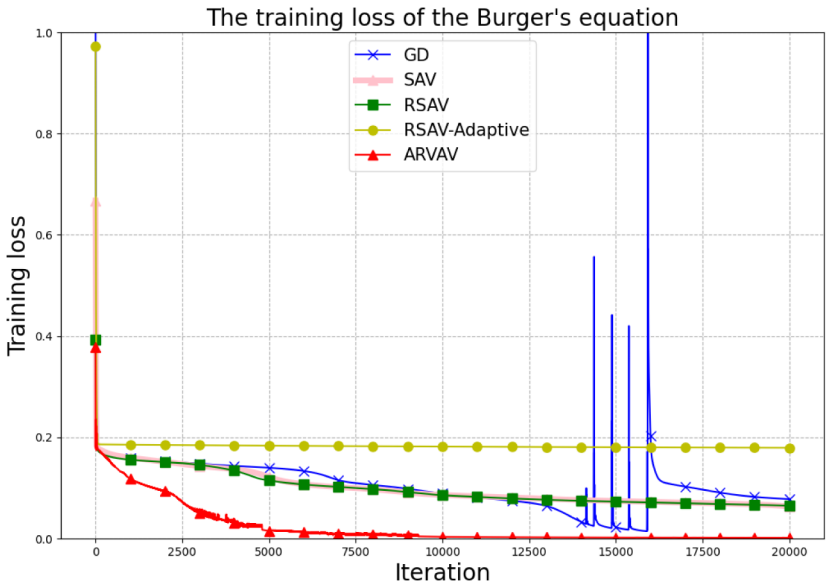
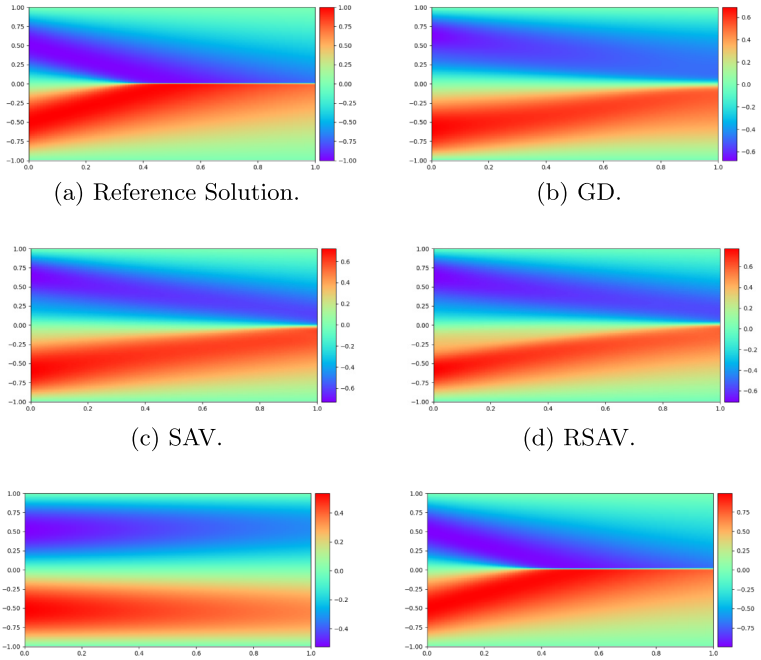


FIG. 5. *Training loss of the Burgers' equation, which indicates the training loss of ARVAV is more stable compared to the one of GD, which has large oscillations. Moreover, the final training loss of ARVAV is significantly smaller than that of other methods.*



(a) Reference Solution. (b) GD.

(c) SAV. (d) RSAV.

(e) RSAV-Adaptive. (f) ARVAV.

FIG. 6. *The predicted solutions of Burgers' equations : Figure* (a) *shows the reference solution obtained by the pseudospectral method. The x-axis represents the time variable t, the y-axis represents the space variable x, whereas the color intensity corresponds to the value of* $u(x,t)$.

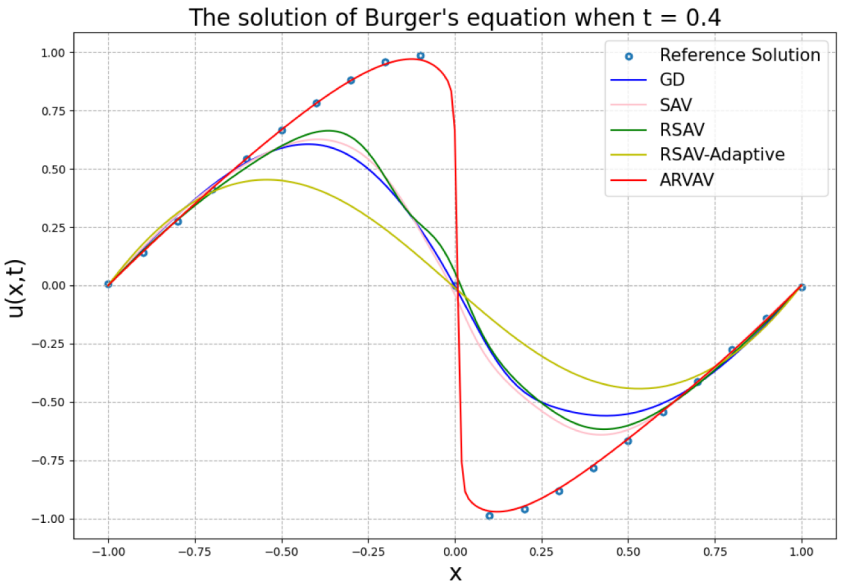


FIG. 7. *The predicted solutions of Burgers' equations: Comparison of GD, SAV, RSAV, RSAV-Adaptive, and ARVAV with the reference solution at* $t = 0.4$. *ARVAV predicts more accurately even at the sharp location.*

the spectral method and the solution obtained by GD, SAV, RSAV, RSAV-Adaptive, and ARVAV. To provide a clear comparison, we focus on the curve when $t = 0.4$ and compare the reference solution with GD in Figure 7. As Burgers' equation is challenging due to its nonlinearity, GD, SAV, RSAV, and RSAV-Adaptive struggle to produce accurate solutions. However, the ARVAV algorithm significantly outperforms others, demonstrating its effectiveness in solving challenging nonlinear problems like Burgers' equations.

5.4. Superlinear convergence in univariate case. Finally, we can focus on demonstrating the superlinear convergence in the context of univariate optimization of Algorithm 4.1, employing two distinct yet simple function forms for illustration purposes. We remain cognizant that exploration of the multivariate scenario constitutes a rich area for further investigation.

The two functions selected for this study are $f(x) = \frac{1}{3}x^3 - 100x + 1000$, defined on the interval $[0, 20]$, and $g(x) = (\sin(x) - \frac{1}{2})^2 + 5$, defined on the interval $[-1, 2]$. The function $f(x)$ exhibits a minimum at $x = 10$, whereas the function $g(x)$ is minimized at $x = \frac{\pi}{6}$.

Our previously derived results inform us that the convergence rate, here denoted by q, is given by the $\frac{1+\sqrt{5}}{2}$. This relationship can be expressed in the context of error at each iteration as $\varepsilon_{n+1} = C\varepsilon_n^q$. It is equivalent to stating that $q = \frac{\ln(\varepsilon_{n+1}/\varepsilon_n)}{\ln(\varepsilon_n/\varepsilon_{n-1})}$. To simplify notation, we introduce $q_n = \frac{\ln(\varepsilon_{n+1}/\varepsilon_n)}{\ln(\varepsilon_n/\varepsilon_{n-1})}$ to represent the rate of convergence at each step.

To elucidate these concepts, we tabulate the aforementioned variables in Table 2. As the optimization process gravitates toward the minimizer, the empirical convergence rate approximates 1.6, corroborating the theoretical superlinear convergence predicted in our analysis.

TABLE 2

This table enumerates the values of ε and the derived convergence rate, q, at each iteration, n, for the optimization of functions $f(x)$ and $g(x)$. The functions under study are $f(x) = \frac{1}{3}x^3 - 100x + 1000$ and $g(x) = (\sin(x) - \frac{1}{2})^2 + 5$. The calculated rates of convergence, denoted by q_n, provide an empirical validation of the superlinear convergence observed in these scenarios.

f(x)	ε_n	q_n	g(x)	ε_n	q_n
$n=1$	0.4931	–	$n=1$	0.1096	–
$n=2$	0.1604	2.4921	$n=2$	0.0201	1.4949
$n=3$	0.0098	1.7382	$n=3$	0.0016	1.6101
$n=4$	7.54×10^{-5}	1.5673	$n=4$	2.70×10^{-5}	1.6140
$n=5$	3.69×10^{-8}	1.6385	$n=5$	3.72×10^{-8}	1.6192
$n=6$	1.39×10^{-13}	–	$n=6$	8.68×10^{-13}	–

6. Conclusions. We proposed a new optimization algorithm, vector auxiliary variable with relaxation (RVAV), that satisfies an unconditional energy dissipation law and possesses excellent convergence properties. We provided rigorous proofs for its linear convergence rate in the convex setting and proposed an improved algorithm which is shown to have a superlinear convergence rate in the univariate case. We also proposed an adaptive version of the RVAV (ARVAV) which combines the advantages of RVAV with adaptive step size based on Steffensen's method. The unconditional energy dissipation property of our algorithm is particularly useful in ensuring the stability of the optimization process. Our numerical results for convex/nonconvex optimizations and for using PINN to solve Burgers' equation demonstrate that the ARVAV algorithm outperforms some existing optimization methods, providing a new and powerful tool for solving optimization problems. It is hoped that our contributions will serve as a catalyst for continued exploration in this field and make a significant impact on the development of optimization algorithms for solving complex problems in machine learning, material science, and fluid dynamics.

REFERENCES

[1] S. M. Allen and J. W. Cahn, *A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening*, Acta Metall. Mater., 27 (1979), pp. 1085–1095, https://doi.org/10.1016/0001-6160(79)90196-2.

[2] D. M. Anderson, G. B. McFadden, and A. A. Wheeler, *Diffuse-interface methods in fluid mechanics*, Annu. Rev. Fluid Mech., 30 (1998), pp. 139–165, https://doi.org/10.1146/annurev.fluid.30.1.139.

[3] H. Attouch, J. Bolte, and B. F. Svaiter, *Convergence of descent methods for semi-algebraic and tame problems: Proximal algorithms, forward–backward splitting, and regularized Gauss–Seidel methods*, Math. Program., 137 (2013), pp. 91–129, https://doi.org/10.1007/s10107-011-0484-9.

[4] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.

[5] A. A. Brown and M. C. Bartholomew-Biggs, *Some effective methods for unconstrained optimization based on the solution of systems of ordinary differential equations*, J. Optim. Theory Appl., 62 (1989), pp. 211–224, https://doi.org/10.1007/BF00941054.

[6] K. Elder, M. Katakowski, M. Haataja, and M. Grant, *Modeling elasticity in crystal growth*, Phys. Rev. Lett., 88 (2002), 245701, https://doi.org/10.1103/PhysRevLett.88.245701.

[7] C. M. Elliott and A. M. Stuart, *The global dynamics of discrete semilinear parabolic equations*, SIAM J. Numer. Anal., 30 (1993), pp. 1622–1663, https://doi.org/10.1137/0730084.

[8] D. J. Eyre, *Unconditionally gradient stable time marching the Cahn-Hilliard equation*, MRS Online Proceedings Library, 529 (1998), 39, https://doi.org/10.1557/PROC-529-39.

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, UK, 2016.

[10] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, in Proceedings of the International Conference on Machine Learning, PMLR, 2015, pp. 448–456.
[11] M. Jiang, Z. Zhang, and J. Zhao, *Improving the accuracy and consistency of the scalar auxiliary variable (SAV) method with relaxation*, J. Comput. Phys., 456 (2022), 110954, https://doi.org/10.1016/j.jcp.2022.110954.
[12] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, in Proceedings of the International Conference for Learning Representations, 2015.
[13] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature, 521 (2015), pp. 436–444, https://doi.org/10.1038/nature14539.
[14] H. Liu and X. Tian, *AEGD: Adaptive gradient descent with energy,* Numer. Algebra Control Optim., 15 (2025), pp. 315–340, https://doi.org/10.3934/naco.2023015.
[15] X. Liu, J. Shen, and X. Zhang, *An efficient and robust scalar auxialiary variable based algorithm for discrete gradient systems arising from optimizations*, SIAM J. Sci. Comput., 45 (2023), pp. A2304–A2324, https://doi.org/10.1137/23M1545744.
[16] M. A. Nielsen, *Neural Networks and Deep Learning*, Determination Press, San Francisco, 2015, http://neuralnetworksanddeeplearning.com/.
[17] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, J. Comput. Phys., 378 (2019), pp. 686–707, https://doi.org/10.1016/j.jcp.2018.10.045.
[18] D. Saupe, *Discrete versus continuous Newton's method: A case study*, Acta. Appl. Math., 13 (1988), pp. 59–80, https://doi.org/10.1007/BF00047502.
[19] J. Shen, C. Wang, X. Wang, and S. M. Wise, *Second-order convex splitting schemes for gradient flows with Ehrlich–Schwoebel type energy: Application to thin film epitaxy*, SIAM J. Numer. Anal., 50 (2012), pp. 105–125, https://doi.org/10.1137/110822839.
[20] J. Shen and J. Xu, *Convergence and error analysis for the scalar auxiliary variable (SAV) schemes to gradient flows*, SIAM J. Numer. Anal., 56 (2018), pp. 2895–2912, https://doi.org/10.1137/17M1159968.
[21] J. Shen, J. Xu, and J. Yang, *The scalar auxiliary variable (SAV) approach for gradient flows*, J. Comput. Phys., 353 (2018), pp. 407–416, https://doi.org/10.1016/j.jcp.2017.10.021.
[22] J. Shen, J. Xu, and J. Yang, *A new class of efficient and robust energy stable schemes for gradient flows*, SIAM Rev., 61 (2019), pp. 474–506, https://doi.org/10.1137/17M1150153.
[23] J. Shen and X. Yang, *Numerical approximations of Allen-Cahn and Cahn-Hilliard equations,* Discrete Contin. Dyn. Syst, 28 (2010), pp. 1669–1691, https://doi.org/10.3934/dcds.2010.28.1669.
[24] J. Steffensen, *Remarks on iteration*, Scand. Actuar. J., 1933 (1933), pp. 64–72, https://doi.org/10.1080/03461238.1933.10419209.
[25] J. Steffensen, *Further remarks on iteration*, Scand. Actuar. J., 1945 (1945), pp. 44–55, https://doi.org/10.1080/03461238.1945.10404918.
[26] W. Su, S. Boyd, and E. Candes, *A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights*, in Advances in Neural Information Processing Systems, 2014, 27.
[27] X. Yang, *Linear, first and second-order, unconditionally energy stable numerical schemes for the phase field model of homopolymer blends*, J. Comput. Phys., 327 (2016), pp. 294–316, https://doi.org/10.1016/j.jcp.2016.09.029.
[28] M. D. Zeiler, *ADADELTA: An Adaptive Learning Rate Method*, preprint, arXiv:1212.5701, 2012.
[29] J. Zhang, *Numerical Method Based Neural Network and Its Application in Scientific Computing, Operator Learning and Optimization Problem*, Ph.D. thesis, Purdue University, 2022, https://doi.org/10.25394/PGS.20359674.v1.
[30] J. Zhao, Q. Wang, and X. Yang, *Numerical approximations for a phase field dendritic crystal growth model based on the invariant energy quadratization approach*, Internat. J. Numer. Methods Engrg., 110 (2017), pp. 279–300, https://doi.org/10.1002/nme.5372.
[31] M. Zhao, Z. Lai, and L.-H. Lim, *Stochastic Steffensen method*, Comput. Optim. Appl., 89 (2024), pp. 1–32, https://doi.org/10.1007/s10589-024-00583-7.
[32] J. Zhu, L.-Q. Chen, J. Shen, and V. Tikare, *Coarsening kinetics from a variable-mobility Cahn-Hilliard equation: Application of a semi-implicit Fourier spectral method*, Phys. Rev. E, 60 (1999), pp. 3564–3572, https://doi.org/10.1103/PhysRevE.60.3564.
[33] P. J. Zufiria and R. S. Guttalu, *On an application of dynamical systems theory to determine all the zeros of a vector function*, J. Math. Anal. Appl., 152 (1990), pp. 269–295, https://doi.org/10.1016/0022-247X(90)90103-M.