Applied Mathematics, 2024, 15, 267-278
https://www.scirp.org/journal/am
ISSN Online: 2152-7393
ISSN Print: 2152-7385

The Modified BAPG$_s$ Method for Support Vector Machine Classifier with Truncated Loss

Kexin Ren

School of Information Science and Technology, Jinan University, Guangzhou, China
Email: renkexin59@163.com

How to cite this paper: Ren, K.X. (2024) The Modified BAPG$_s$ Method for Support Vector Machine Classifier with Truncated Loss. *Applied Mathematics*, **15**, 267-278.
https://doi.org/10.4236/am.2024.154015

Received: March 20, 2024
Accepted: April 7, 2024
Published: April 10, 2024

Abstract

In this paper, we modify the Bregman APG$_s$ (BAPG$_s$) method proposed in (Wang, L, *et al.*) for solving the support vector machine problem with truncated loss (HTPSVM) given in (Zhu, W, *et al.*), we also add an adaptive parameter selection technique based on (Ren, K, *et al.*). In each iteration, we use the linear approximation method to get the explicit solution of the subproblem and set a function ϕ to apply the Bregman distance. Finally, numerical experiments are performed to verify the efficiency of BAPG$_s$.

Keywords

HTPSVM, Bregman Distance, BAPG$_s$ Algorithm

1. Introduction

SVM (Support Vector Machine) [1] is a supervised learning algorithm commonly used for classification tasks and has been successfully applied to many technological fields, such as text categorization [2], financial forecast [3], image classification [4] and so on. This paper focuses on a binary classification problem. Given training samples $\{(x_i, y_i), i = 1, \cdots, m\}$, where $x_i \in \mathbb{R}^n$, $y_i \in \{-1, 1\}$, the objective of SVM is to identify an optimal separating hyperplane to separate data points into two classes. Scholars have proposed some classic SVM models based on convex loss functions, such as the hinge loss (also called L_1 loss) in classic SVM [5], the least square loss in LSSVM [6] and the huberized pinball loss in HPSVM [7]. However, in practice, the real dataset often contain noise. Since convex loss functions are generally unbounded, convex losses are highly sensitive to outliers and potentially influenced by outliers. Therefore, some nonconvex loss functions are proposed to improve robustness compared with the convex loss functions [8]. For example, [9] proposed the ramp loss based on hinge loss, the truncated pinball loss was proposed by [10]. Recently, a noise in-sensitive and robust support vector machine classifier with huberied truncated pinball loss (HTPSVM) was proposed in [11], this loss is smooth and nonconvex loss function. The HTPSVM can be transformed into format of "Loss + Penalty", in which the penalty is a hybrid of l_1 norm and l_2 norm penalty.

Here, the HTPSVM model and algorithm of literature [11] are briefly introduced. Consider a classification problem with training samples $\{x_i, y_i\}_{i=1}^{m} \subset \mathbb{R}^d \times \{-1,1\}$. The HTPSVM seeks to solve the following regularization problem:

$$\min_{b\in\mathbb{R}, w\in\mathbb{R}^d} \frac{1}{m}\sum_{i=1}^{m} \ell_{htp}\left(y_i\left(b+w^{\mathrm{T}}x_i\right)\right)+\lambda\|w\|_1+\frac{\|w\|_2^2}{2}+\frac{b^2}{2}, \tag{1}$$

the huberied truncated pinball loss $\ell_{htp}(\cdot)$ function is defines as

$$\ell_{htp}(u)=\begin{cases} 1, & u\le -\frac{2}{5} \\ \frac{4}{5}-u-\frac{5}{4}u^2, & -\frac{2}{5}<u\le 0, \\ \frac{4}{5}-u, & 0<u\le\frac{3}{5}, \\ \frac{5}{4}(1-u)^2, & \frac{3}{5}<u\le 1, \\ \frac{5}{8}(1-u)^2, & 1\le u<\frac{7}{5}, \\ -\frac{1}{2}\left(\frac{6}{5}-u\right), & \frac{7}{5}\le u<\frac{8}{5}, \\ -\frac{1}{2}\left(\frac{6}{5}-u\right)-\frac{5}{8}\left(u-\frac{8}{5}\right)^2, & \frac{8}{5}\le u<2, \\ \frac{3}{10}, & u\ge 2, \end{cases} \tag{2}$$

which is a nonconvex and smooth function. The HTPSVM combine the benefits of both l_1 and l_2 norm regularizers and and it has been demonstrated in [11] that it can reduce the effects of noise in the training sample. Therefore, we consider that studying the HTPSVM model is meaningful. The APG algorithm was used to solve the model in [11]. [12] applied the APG$_s$ method (first proposed in [13]) to solve problem (1) and obtain better convergence behavior. However we find that the proximal operator for computing the l_1 norm causes the subproblem to be solved slowly in APG and APG$_s$ algorithms, we attempt to accelerate the solution process for this model. Recently, [14] propose the Bregman APG$_s$ (BAPG$_s$) method, which avoids the restrictive global Lipschitz gradient continuity assumption. In this paper, we improve BAPG$_s$ algorithm to solve the problem (1) and replace the Lipschitz constant by an appropriate positive definite matrix and obtain better results after we perform numerical experiments on 10 datasets to test our method.

The rest of this paper is organized as follows. In the next section, we provide preliminary materials used in this work. In Section 3, we introduce the BAPG$_s$ algorithm proposed by [14] and present our algorithm based on the BAPG$_s$ method for solving the HTPSVM model (1). The convergence of our method is also discussed. Section 4 performs some experiments.

2. Preliminaries

In this paper, we let $\mathbb{R}$ denote the set of real numbers. We work in the Euclidean space $\mathbb{R}^n$, and the standard Euclidean inner product and the induced norm on $\mathbb{R}^n$ are denoted by $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$. The domain of the function $f:\mathbb{R}^n \to (-\infty,+\infty]$ is defined by $\operatorname{dom} f = \{x \in \mathbb{R}^n : f(x) < +\infty\}$. We say that *f* is proper if $\operatorname{dom} f \neq \varnothing$. A proper function *f* is said to be closed if it is lower semicontinuous at any $x \in \operatorname{dom} f$, *i.e.* $f(x) \le \liminf_{z\to x} f(z)$.

Definition 1. [[15], Definition 8.3] For a proper closed function *f*, the regular subdifferential of $f:\mathbb{R}^n \to \mathbb{R}\cup\{+\infty\}$ at $x \in \operatorname{dom} f$ is defined by

$$\hat{\partial} f(x) := \left\{ \hat{x} \in \mathbb{R}_n : \liminf_{z\to x, z\neq x} \frac{f(z) - f(x) - \langle \hat{x}, z - x\rangle}{\|z - x\|} \ge 0 \right\}. \tag{3}$$

The (general) subdifferential of *f* at $x \in \operatorname{dom} f$ is defined

$$\partial f(x) := \left\{ \hat{x} : \exists x^k \xrightarrow{f} x, \hat{x}^k \to \hat{x} \text{ with } \hat{x}^k \in \hat{\partial} f(x^k) \text{ for each } k \right\}, \tag{4}$$

where $x^k \xrightarrow{f} x$ means both $x^k \to x$ and $f(x^k) \to f(x)$. Note that if *f* is also convex, then the general subdifferential and regular subdifferential of *f* at $x \in \operatorname{dom} f$ reduce to the classical subdifferential [[15], Proposition 8.12], that is

$$\partial f(x) = \{ \hat{x} : f(y) \ge f(x) + \langle \hat{x}, y - x\rangle \text{ for all } y \}. \tag{5}$$

Definition 2. (Kernel Generating Distances and Bregman Distances [16] [17] [18]) Let *C* be a nonempty, convex and open subset of $\mathbb{R}^n$. Associated with *C*, a function $\phi:\mathbb{R}^n \to (-\infty,+\infty]$ is called a kernel generating distance if it satisfies the following:

1) ϕ is proper, lower semicontinuous and convex, with $\operatorname{dom}\phi \subset \overline{C}$ and $\operatorname{dom}\partial\phi = C$;

2) ϕ is continuously differentiable on $\operatorname{int}\operatorname{dom}\phi \equiv C$.

We denote the class of kernel generating distances by $\mathcal{G}(C)$. Given $\phi \in \mathcal{G}(C)$, the Bregman distance $D_\phi : \operatorname{dom}\phi \times \operatorname{int}\operatorname{dom}\phi \to (0,+\infty]$ is defined by

$$D_\phi(x,y) := \phi(x) - \phi(y) - \langle \nabla\phi(y), x - y\rangle.$$

For exmple, when $\phi(x) = \|x\|^2$, then $D_\phi(x,y) = \|x - y\|^2$. If $\phi(x) = x^{\mathrm{T}} A x$, then $D_\phi(x,y) = (x-y)^{\mathrm{T}} A (x-y)$. In this article, the gradient Lipschitz continuity condition of the function *f* is no longer required, instead it is replaced by the L-smooth adaptive function of pair (f,ϕ). The definition of L-smooth adaptable as follows.

Definition 3. A pair of functions (f,ϕ), $\phi \in \mathcal{G}(C)$, $f:\mathbb{R}^n \to (-\infty,+\infty]$ is a proper and lower semicontinuous function with $\operatorname{dom}\phi \subset \operatorname{dom} f$ and *f* is continuously differentiable on $C = \operatorname{int}\operatorname{dom}\phi$, is called L-smooth adaptable (L-smad) on *C* if there exists $L > 0$ such that $L\phi - f$ and $L\phi + f$ are convex on *C*.

Lemma 1. (Full Extended Descent Lemma [19]) A pair of functions (f,ϕ) is L-smad on $C = \operatorname{int}\operatorname{dom}\phi$ if and only if:
$\left|f(x)-f(y)-\langle\nabla f(y),x-y\rangle\right| \le LD_\phi(x,y)$, $\forall x,y\in\operatorname{int}\operatorname{dom}\phi$.

Definition 4. $f:\mathbb{R}^n\to\mathbb{R}\cup\{+\infty\}$ is called μ-relative weakly convex to ϕ on C if there exists $\mu>0$ such that $f+\mu\phi$ is convex on C [14].

3. The Modified BAPG$_s$ Method for HTPSVM

In this section, we first describe the BAPG$_s$ method proposed in [14], then the modified BAPG$_s$ method with adaptive parameter is given for HTPSVM.

3.1. BAPG$_s$ Method

Consider the following optimization problem:

$$\min_{x\in\mathbb{R}^n} F(x) := f(x) + P_1(x) - P_2(x), \tag{6}$$

where f is a μ-relative weakly convex continuously differentiable function, P_1 is a proper, lower semicontinuous convex function and P_2 is continuous and convex. Besides, F is level-bounded *i.e.*, for every $\alpha\in\mathbb{R}$, the set $\{x\in\mathbb{R}^n \mid F(x)\le\alpha\}$ is bounded; F is bounded below *i.e.*, $\inf_{x\in\mathbb{R}^n} F(x) > -\infty$. The iterative scheme of BAPG$_s$ [14] for solving probelm (6) is shown in Algorithm 1, where D_ϕ is a Bregman distance defined in Section 2.

Algorithm 1 BAPG$_s$ algorithm

Initialization: $x^0, z^0 \in \operatorname{dom} P_1, \tau, L>0, \{\theta_k\}\subseteq(0,1]$.
Output: x^{k+1}, Iteration: k
for $k=0,1,2\cdots,$ **do**
  let any $\xi^k\in\partial P_2(x^k)$ and update

$$\begin{aligned}
y^k &= \theta_k z^k + (1-\theta_k)\,x^k,\\
z^{k+1} &= \operatorname{argmin}_{z\in\mathbb{R}^n}\left\{\langle\nabla f(y^k)-\xi^k, z-y^k\rangle + P_1(z) + \tau\theta_k L D_\phi(z,z^k)\right\},\\
x^{k+1} &= \theta_k z^{k+1} + (1-\theta_k)x^k.
\end{aligned} \tag{7}$$

end for

We see that when $D_\phi(x,y)=\frac{1}{2}\|x-y\|^2$, BAPG$_s$ reduces to APG$_s$ in [13]. [14] proved the global convergence of the iterates generated by BAPG$_s$ to a limiting critical point under some assumptions.

3.2. Adaptive BAPG$_s$ Method for HTPSVM

By writing the nonconvex loss ℓ_{htp} as the difference of three smooth convex functions, the problem (1) can be expressed as following from [12]

$$\min_{b\in\mathbb{R},w\in\mathbb{R}^d} F(b,w) = f_1(b,w) - f_2(b,w) - f_3(b,w) + P_1(b,w), \tag{8}$$

where $P_1(b,w)=\lambda\|w\|_1+\frac{\|w\|_2^2}{2}+\frac{b^2}{2}$, λ is the regularization parameter; for

$j=1,2$, $f_j(b,w)=\frac{1}{m}\sum_{i=1}^{m}\ell_j\left[y_i\left(b+w^{\mathrm{T}}x_i\right)\right]$, and the smooth convex functions ℓ_j are defined as

$$\ell_1(u)=\begin{cases}\frac{4}{5}-u, & \text{if} \quad u<\frac{3}{5},\\ \frac{5}{4}(1-u)^2, & \text{if} \quad \frac{3}{5}\le u<1,\\ \frac{5}{8}(1-u)^2, & \text{if} \quad 1\le u<\frac{7}{5},\\ -\frac{1}{2}\left(\frac{6}{5}-u\right), & \text{if} \quad u\ge\frac{7}{5},\end{cases} \tag{9}$$

$$\ell_2(u)=\begin{cases}-u-\frac{1}{5}, & \text{if} \quad u\le-\frac{2}{5},\\ \frac{5}{4}u^2, & \text{if} \quad -\frac{2}{5}<u\le 0,\\ 0, & \text{if} \quad u\ge 0,\end{cases} \tag{10}$$

$$\ell_3(u)=\begin{cases}0, & \text{if} \quad u\le\frac{8}{5},\\ \frac{5}{8}\left(u-\frac{8}{5}\right)^2, & \text{if} \quad \frac{8}{5}\le u<2,\\ -\frac{1}{2}\left(\frac{9}{5}-u\right), & \text{if} \quad u\ge 2.\end{cases} \tag{11}$$

Then we can apply the BAPG$_s$ to solve problem (8) in the form of (6)

- $\mathcal{P}_1$: $f=f_1-f_3$ (nonconvex), $P_2=f_2$ (convex);
- $\mathcal{P}_2$: $f=f_1-f_2$ (nonconvex), $P_2=f_3$ (convex).

Next, We will briefly illistrate that the problem (8) can be solved by the BAPG$_s$ [14].

Theorem 1. Let f as defined in $\mathcal{P}_1$ and $\mathcal{P}_2$. Set $\phi(x):=\frac{1}{2}x^{\mathrm{T}}Qx$, where $Q=\frac{1}{m}\sum_{i=1}^{m}\frac{5}{2}Q_i$, $Q_i=\left(y_i,y_ix_i^{\mathrm{T}}\right)^{\mathrm{T}}\left(y_i,y_ix_i^{\mathrm{T}}\right)$. Then, the pair (f,ϕ) is L-smooth adaptable on $\mathbb{R}^n$ with $L=1$.

Proof. Firstly, for $\mathcal{P}_1$, since

$$\ell'_{1-3}(u)+\frac{5}{2}u=\begin{cases}\frac{5}{2}u-1, & u\le\frac{3}{5},\\ 5u-\frac{5}{2}, & \frac{3}{5}<u\le 1,\\ \frac{15}{4}u-\frac{5}{4}, & 1\le u<\frac{7}{5},\\ \frac{5}{2}u+\frac{1}{2}, & \frac{7}{5}\le u<\frac{8}{5},\\ \frac{5}{4}u+\frac{5}{2}, & \frac{8}{5}\le u<2,\\ \frac{5}{2}u, & u\ge 2,\end{cases} \tag{12}$$

and

$$\frac{5}{2}u-\ell_{1-3}'(u)=\begin{cases}\frac{5}{2}u+1, & u\le\frac{3}{5},\\ \frac{5}{2}, & \frac{3}{5}<u\le 1,\\ \frac{5}{4}u+\frac{5}{4}, & 1<u\le\frac{7}{5},\\ \frac{5}{2}u-\frac{1}{2}, & \frac{7}{5}\le u<\frac{8}{5},\\ \frac{15}{4}u-\frac{5}{2}, & \frac{8}{5}\le u<2,\\ \frac{5}{2}u, & u\ge 2,\end{cases} \tag{13}$$

are monotonically increasing, it is easy to verify that $\ell_{1-3}(u)+\frac{5}{4}u^2$ and $\frac{5}{4}u^2-\ell_{1-3}(u)$ are convex. Then we can easily get the convexity of

$$\begin{aligned} f(b,w)+\frac{1}{2}x^{\mathrm{T}}Qx &= \frac{1}{m}\sum_{i=1}^{m}\ell_{1-3}\left[y_i\left(b+w^{\mathrm{T}}x_i\right)\right]+\frac{1}{2m}\sum_{i=1}^{m}\frac{5}{2}(b;w)^{\mathrm{T}}Q_i(b;w)\\ &= \frac{1}{m}\sum_{i=1}^{m}\left[\ell_{1-3}\left[y_i\left(b+w^{\mathrm{T}}x_i\right)\right]+\frac{5}{4}(b;w)^{\mathrm{T}}Q_i(b;w)\right]\\ &= \frac{1}{m}\sum_{i=1}^{m}\left[\ell_{1-3}\left[y_i\left(b+w^{\mathrm{T}}x_i\right)\right]+\frac{5}{4}\left[y_i\left(b+w^{\mathrm{T}}x_i\right)\right]^2\right], \end{aligned} \tag{14}$$

and

$$\frac{1}{2}x^{\mathrm{T}}Qx-f(b,w)=\frac{1}{m}\sum_{i=1}^{m}\left[\frac{5}{4}\left[y_i\left(b+w^{\mathrm{T}}x_i\right)\right]^2-\ell_{1-3}\left[y_i\left(b+w^{\mathrm{T}}x_i\right)\right]\right], \tag{15}$$

the proof is similar for $\mathcal{P}_2$. It is clear that (f,ϕ) is 1-smooth adaptable on $\mathbb{R}^n$, this further implies that there exists $0<\mu\le 1$ such that $f+\mu\phi$ is convex.

We can see that the problem (8) satisfies the conditions required in [14] with $\phi(x)=\frac{1}{2}x^{\mathrm{T}}Qx$ for the pair (f,ϕ), where Q defined as Theorem 1. Therefore the BAPG$_s$ method (Algorithm 1), here we let $\tau=1$ and replace (7) with the following steps, can be used for solving (8)

$$\begin{aligned} &y^k=\theta_k z^k+(1-\theta_k)x^k,\\ &z^{k+1}=\arg\min_{z\in\mathbb{R}^n}\left\{\left\langle\nabla f\left(y^k\right)-\xi^k,z-y^k\right\rangle+P_1(z)+\frac{\theta_k}{2}\left[\left(z-z^k\right)^{\mathrm{T}}Q\left(z-z^k\right)\right]\right\},\\ &x^{k+1}=\theta_k z^{k+1}+(1-\theta_k)x^k. \end{aligned} \tag{16}$$

The selection of parameter $\{\theta_k\}$ in [14] as: for fixed positive integer N, let $\theta_0=1$,

$$\theta_{k+1}=\frac{\sqrt{\theta_k^4+4\theta_k^2}-\theta_k^2}{2},\ k=1,2,\cdots,N$$

and $\theta_k\equiv\theta_N$ for all $k>N$. It is to see that the value of the positive integer N is difficult to determine. Combining with the adaptive parameter selection criterion proposed in [12]: let $\theta_0 = 1$, $\theta_k = \frac{\sqrt{\theta_{k-1}^4 + 4\theta_{k-1}^2} - \theta_{k-1}^2}{2}$ for $k \geq 1$ and compute

$$d_k := \frac{H_{k-1} - H_k}{\left(x^k - x^{k-1}\right)^{\mathrm{T}} \left(x^k - x^{k-1}\right)}, \tag{17}$$

when $k \geq 2$, where $H_k := F\left(x^k\right) + \frac{\beta_k}{2}\left(x^k - x^{k-1}\right)^{\mathrm{T}} Q\left(x^k - x^{k-1}\right)$ and $\beta_k = \frac{\alpha_k}{\theta_{k-1}^2}$ (the assumption of sequence $\{\alpha_k\}$ given in [14, Assumption 2]). Let N be the first k satisfying $d_k \leq d_{k+1}$. The BAPG$_s$ algorithm with adaptive parameter for problem (8) (HTPSVM) is shown in Algorithm 2.

Algorithm 2 BAPG$_s$ with adaptive parameter for HTPSVM

Initialization: $x^0, z^0 \in \operatorname{dom} P_1, \theta_0 = 1$.
Output: x^{k+1}, Iteration: $k + k'$.
for $k' = 0, 1, 2, \cdots$, with $\theta_{k'+1} = \frac{\sqrt{\theta_{k'}^4 + 4\theta_{k'}^2} - \theta_{k'}^2}{2}$ **do**
  let any $\xi^{k'} \in \partial P_2(x^{k'})$ and update

$$\begin{aligned} y^{k'} &= \theta_{k'} z^{k'} + (1 - \theta_{k'}) x^{k'}, \\ z^{k'+1} &= \operatorname{argmin}_{z \in \mathbb{R}^n} \{\langle \nabla f(y^{k'}) - \xi^{k'}, z - y^{k'} \rangle + P_1(z) \\ &\qquad + \frac{\theta_{k'}}{2}[(z - z^{k'})^T Q (z - z^{k'})]\}, \\ x^{k'+1} &= \theta_{k'} z^{k'+1} + (1 - \theta_{k'}) x^{k'}, \end{aligned} \tag{18}$$

  compute $d_k'(k' \geq 2)$ as (17).
  if $k' \geq 2$ & $d_{k'+1} > d_{k'}$ **then**
    fix $N := k'$ and $\theta_N = \frac{\sqrt{\theta_N^4 + 4\theta_N^2} - \theta_N^2}{2}$;
    break
  end if
end for
$x^0 = x^N$, $z^0 = z^N$, $\theta \equiv \theta_N$.
for $k = 0, 1, 2 \cdots$, **do**
  let any $\xi^k \in \partial P_2(x^k)$ and update

$$\begin{aligned} y^k &= \theta z^k + (1 - \theta) x^k, \\ z^{k+1} &= \operatorname{argmin}_{z \in \mathbb{R}^n} \{\langle \nabla f(y^k) - \xi^k, z - y^k \rangle + P_1(z) \\ &\qquad + \frac{}{2}[(z - z^k)^T Q (z - z^k)]\}, \end{aligned} \tag{19}$$

$$x^{k+1} = \theta z^{k+1} + (1 - \theta) x^k.$$

end for

4. Numerical Results

In this section, we aim to show the performance of Algorithm 2 for solving problem (1) by using MATLAB R2020b on a 64-bit PC with an Intel(R) Core(TM) i7-10870H CPU (2.20GHz) and 16GB of RAM.

First, consider the optimality condition (19) of Algorithm 2

$$\begin{aligned}0 &\in \nabla f\left(y^{k'}\right)-\xi^{k}+\partial P_1\left(z^{k+1}\right)+\theta Q\left(z^{k+1}-z^{k}\right)\\ &=\nabla f\left(y^{k}\right)-\xi^{k}+\lambda\partial\left\|z^{k+1}\right\|_1+z^{k+1}+\theta Q\left(z^{k+1}-z^{k}\right).\end{aligned}$$

Due to there is no explicit solution for this subproblem, we try to instead the l_1 norm by linear approximation, that is, $\|z\|_1 \approx \left\|x^k\right\|_1 + {v^k}^{\mathrm{T}}\left(z-x^k\right)$, where $v^k \in \partial\left\|x^k\right\|_1$ (here we take $v^k := \operatorname{sign}\left(x^k\right)$), then we construct a new iteration step to replace the subproblem in Algorithm 2 as

$$\begin{aligned}z^{k+1} = \arg\min_{z\in\mathbb{R}^n} &\left\{\left\langle \nabla f\left(y^k\right)-\xi^k, z-y^k\right\rangle + \left\|x^k\right\|_1 + \lambda {v^k}^{\mathrm{T}}\left(z-x^k\right)\right.\\ &\left.+\frac{1}{2}z^{\mathrm{T}}z+\frac{\theta}{2}\left[\left(z-z^k\right)^{\mathrm{T}}Q\left(z-z^k\right)\right]\right\},\end{aligned} \tag{20}$$

it is easy to calculate its solution:

$$0=\nabla f\left(y^k\right)-\xi^k+\lambda {v^k}^{\mathrm{T}}+z^{k+1}+\theta Q\left(z^{k+1}-z^k\right),$$

which means

$$(I+\theta Q)z^{k+1}=\theta Q z^k-\nabla f\left(y^k\right)+\xi^k-\lambda v^k.$$

Then the update (18) and (19) are replaced by

$$\begin{cases} z^{k'+1}=\left(I+\theta_{k'}Q\right)^{-1}\left(\theta_{k'}Qz^{k'}-\nabla f\left(y^{k'}\right)+\xi^{k'}-\lambda v^{k'}\right),\\ z^{k+1}=\left(I+\theta Q\right)^{-1}\left(\theta Qz^{k}-\nabla f\left(y^{k}\right)+\xi^{k}-\lambda v^{k}\right),\end{cases} \tag{21}$$

in experiments, where $v^{k'}=\operatorname{sign}\left(x^{k'}\right)$ and $v^k=\operatorname{sign}\left(x^k\right)$. The experiments are conducted on several real world datasets. We select 10 datasets from UCI [20], to compare the Algorithm 2 with APG (method in [11]), APG$_s$ [12] and GIST [21], where in GIST, we set $F=f+P_1$ with $f=f_1-f_2-f_3$. The corresponding parameters of these methods are set the same as in [12]. For each dataset, The 21 initial points are used commonly for all methods: one zero vector, and 5 vectors selected independently from $N(0, \sigma^2 I)$ for each $\sigma\in\{1,2,4,8\}$. All algorithms stop if $\dfrac{\left\|\left(b^{k+1};w^{k+1}\right)-\left(b^k;w^k\right)\right\|}{\max\left\{1,\left\|\left(b^k;w^k\right)\right\|\right\}}<10^{-6}$ or the number of iterations hits 3000. The average results are given in **Table 1** and **Table 2**, including the number of iterations (iter), objective function value (fval) and CPU time in seconds (CPU) at termination with $\lambda=1\times10^{-3}$ and 5×10^{-4}, where BAPG$_s$- $\mathcal{P}_1$ and APG$_s$- $\mathcal{P}_1$ represent using BAPG$_s$ (algorithm 2) and APG$_s$ [12] for $\mathcal{P}_1$ respectively ($\mathcal{P}_1$ described in section 3.2).

Table 1. Comparison on 10 datasets with $\lambda=1\times10^{-3}$.

Dataset	Iter					
	BAPG$_s$- $\mathcal{P}_1$	BAPG$_s$- $\mathcal{P}_2$	APG$_s$- $\mathcal{P}_1$	APG$_s$- $\mathcal{P}_1$	APG	GIST
Magic	89.81	**87.05**	91.81	89.33	3000.00	325.57
Rice	78.95	**78.81**	80.10	80.24	171.43	1478.81
Hepatitis	226.90	**217.14**	235.24	224.38	2580.14	1197.33
Tic-Tac-Toe	**150.76**	151.71	158.33	159.14	161.62	177.19

Continued

Spect heart	94.10	**90.71**	96.38	92.81	886.76	723.76
Fourclass	44.81	**42.86**	46.05	43.48	3000.00	48.24
German	258.81	**250.62**	268.67	258.71	1873.33	1523.71
Ionosphere	219.95	**216.52**	228.00	223.19	660.90	1485.95
Jain	41.57	**41.19**	41.81	41.81	2028.10	2857.24
Haberman	394.43	363.43	404.29	3000.00	152.00	**86.57**
			CPU			
Magic	0.353	**0.258**	0.355	0.263	17.368	5.424
Rice	0.421	**0.376**	0.849	0.767	3.141	34.529
Tic-Tac-Toe	0.215	**0.143**	0.419	0.268	0.454	0.162
Spect heart	0.058	0.046	0.050	**0.037**	0.569	1.087
Fourclass	0.075	**0.049**	0.076	0.050	3.454	0.056
German	0.389	**0.283**	0.410	0.312	4.222	3.766
Ionosphere	0.156	**0.104**	0.150	0.107	0.402	1.059
Jain	0.037	**0.035**	0.039	0.036	1.549	3.213
Haberman	0.194	0.128	0.189	0.120	1.585	**0.051**
			Fval			
Magic	0.474381	**0.473398**	**0.473398**	0.516992	0.474964	**0.473398**
Rice	0.346266	**0.346256**	**0.346256**	0.346626	0.721512	**0.346256**
Tic-Tac-Toe	0.438651	**0.368594**	0.426611	**0.368594**	0.397603	18.161908
Spect heart	0.388207	**0.388016**	**0.388016**	**0.388016**	0.392196	**0.388016**
Fourclass	0.660204	**0.657473**	**0.657473**	**0.657473**	0.663052	**0.657473**
German	0.579826	**0.579553**	**0.579553**	**0.579553**	0.584878	7.715915
Ionosphere	0.506393	**0.473385**	0.502754	**0.473385**	0.506175	3.154305
Jain	0.436192	**0.435656**	**0.435656**	**0.435656**	0.464155	2.600706
Haberman	0.584050	**0.583851**	**0.583851**	**0.583851**	0.606839	**0.583851**

Table 2. Comparison on 10 datasets with $\lambda = 5\times10^{-4}$.

Dataset	Iter					
	BAPG$_s$- $\mathcal{P}_1$	BAPG$_s$- $\mathcal{P}_2$	APG$_s$- $\mathcal{P}_1$	APG$_s$- $\mathcal{P}_1$	APG	GIST
Magic	90.05	**87.19**	92.05	89.48	3000.00	326.38
Rice	78.95	**78.76**	80.10	80.19	171.62	699.48
Hepatitis	223.48	**214.14**	231.57	221.19	2580.05	1028.95
Tic-Tac-Toe	**150.71**	151.76	158.24	159.14	159.62	168.43
Spect heart	94.10	**90.71**	96.38	92.81	745.90	688.52
Fourclass	44.81	**42.90**	46.05	43.52	3000.00	48.67
German	255.48	**246.57**	265.19	254.52	1873.24	3000.00
Ionosphere	219.81	**217.00**	228.00	223.76	659.33	740.19
Jain	41.52	**41.19**	41.76	41.81	2167.62	2848.19
Haberman	456.10	421.81	477.29	431.00	3000.00	**95.24**

Continued

			CPU			
Magic	0.347	**0.251**	0.354	0.261	16.689	5.462
Pima	0.308	0.216	0.550	0.372	**0.144**	0.795
Rice	0.420	**0.369**	0.800	0.726	1.643	15.001
Tic-Tac-Toe	0.215	**0.145**	0.318	0.199	0.240	0.171
Spect heart	0.058	0.047	0.049	**0.037**	0.463	1.039
Fourclass	0.076	0.050	0.071	**0.046**	3.489	0.058
German	0.376	**0.284**	0.383	0.279	4.208	7.883
Ionosphere	0.152	**0.105**	0.152	0.108	0.408	0.617
Jain	0.038	0.043	0.027	**0.025**	1.662	3.356
Haberman	0.220	0.136	0.210	0.131	1.592	**0.055**
			Fval			
Magic	0.474348	**0.473364**	**0.473364**	**0.473364**	0.518649	**0.473364**
Rice	0.345804	**0.345794**	**0.345794**	**0.345794**	0.346151	**0.345794**
Tic-Tac-Toe	0.438395	**0.368334**	0.426351	**0.368334**	0.397342	18.144564
Spect heart	0.387614	**0.387422**	**0.387422**	**0.387422**	0.390809	**0.387422**
Fourclass	0.660128	**0.657403**	**0.657403**	**0.657403**	0.663006	**0.657403**
German	0.579145	**0.578871**	**0.578871**	**0.578871**	0.584183	7.718112
Ionosphere	0.505612	**0.472627**	0.501982	**0.472627**	0.505449	3.153792
Jain	0.436192	**0.435316**	0.435854	**0.435316**	**0.435316**	2.600706
Haberman	0.583966	**0.583769**	**0.583769**	**0.583769**	0.606836	**0.583769**

From the above tables, we see that Algorithm 2 for $\mathcal{P}_2$ always obtain the smaller function values and converge faster than others, this means that Algorithm 2 for solving HTPSVM model (1) performs well.

5. Conclusions and Suggestions

In this paper, based on the BAPG$_s$ method proposed by [14], we construct the modified BAPG$_s$ with the adaptive parameter selection technique introduced in [12] for solving the HTPSVM model. The linear approximation method is used to improve the subproblem in algorithm and a function ϕ with a suitable matrix Q is set to obtain the L-smad property. Finally, numerical experiments show that our algorithm convergence faster.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Vapnik, V. and Cortes, C. (1995) Support-Vector Networks. *Machine Learning*, **20**,

273-297. https://doi.org/10.1007/BF00994018

[2] Joachims, T. (1998) Text Categorization with Support Vector Machines: Learning with Many Relevant Features. 10*th European Conference on Machine Learning*, Chemnitz, 21-23 April 1998, 137-142. https://doi.org/10.1007/BFb0026683

[3] Zhang, X., Li, A. and Pan, R. (2016) Stock Trend Prediction Based on a New Status Box Method and AdaBoost Probabilistic Support Vector Machine. *Applied Soft Computing*, **49**, 385-398. https://doi.org/10.1016/j.asoc.2016.08.026

[4] Chandra, M. and Bedi, S. (2021) Survey on SVM and Their Application in Image Classification. *International Journal of Information Technology*, **13**, 1-11. https://doi.org/10.1007/s41870-017-0080-1

[5] Rong-En, F., Kai-Wei, C., *et al.* (2008) LIBLINEAR: A Library for Large Linear Classification. *The Journal of Machine Learning Research*, **9**, 1871-1874.

[6] Suykens, J. and Vandewalle, J. (1999) Least Squares Support Vector Machine Classifiers. *Neural Processing Letters*, **9**, 293-300. https://doi.org/10.1023/A:1018628609742

[7] Zhu, W., Song, Y. and Xiao, Y. (2021) Support Vector Machine Classifier with Huberized Pinball Loss. *Engineering Applications of Artificial Intelligence*, **91**, Article 103635. https://doi.org/10.1016/j.engappai.2020.103635

[8] Zhao, L., Mammadov, M., *et al.* (2010) From Convex to Nonconvex: A Loss Function Analysis for Binary Classification. 2010 *IEEE International Conference on Data Mining Workshops*, Sydney, 13 December 2010, 1281-1288. https://doi.org/10.1109/ICDMW.2010.57

[9] Collobert, R., Sinz, F, *et al.* (2006) Large Scale Transductive SVMS. *Journal of Machine Learning Research*, **7**, 1687-1712.

[10] Shen, X., Niu, L., Qi, Z. and Tian, Y. (2017) Support Vector Machine Classifier with Truncated Pinball Loss. *Pattern Recognition*, **68**, 199-210.

[11] Zhu, W., Song, Y. and Xiao, Y. (2022) Robust Support Vector Machine Classifier with Truncated Loss Function by Gradient Algorithm. *Computers & Industrial Engineering*, **172**, Article 108630.

[12] Ren, K., Liu, C. and Wang, L. (2024) The Modified Second APG Method for a Class of Nonconvex Nonsmooth Problems. (In Press)

[13] Lin, D. and Liu, C. (2019) The Modified Second APG Method for DC Optimization Problems. *Optimization Letters*, **13**, 805-824. https://doi.org/10.1007/s11590-018-1280-8

[14] Wang, L., Liu, C. and Ren, K. (2024) The Bregman Modified Second APG Method for DC Optimization Problems. (In Press)

[15] Rockafellar, R. and Wets, R. (2009) Variational Analysis. Springer Science & Business Media, Berlin.

[16] Bolte, J., Sabach, S., Teboulle, M., *et al.* (2017) First Order Methods beyond Convexity and Lipschitz Gradient Continuity with Applications to Quadratic Inverse Problems. *SIAM Journal on Optimization*, **28**, 2131-2151. https://doi.org/10.1137/17M1138558

[17] Lev, M.B. (1967) The Relaxation Method of Finding the Common Point of Convex Sets and Its Application to the Solution of Problems in Convex Programming. *USSR Computational Mathematics and Mathematical Physics*, **7**, 200-217. https://api.semanticscholar.org/corpusid:121309410

[18] Wu, Z., Li, C., *et al.* (2021) Inertial Proximal Gradient Methods with Bregman Re-

gularization for a Class of Nonconvex Optimization Problems. *Journal of Global Optimization*, **79**, 617-644. https://doi.org/10.1007/s10898-020-00943-7

[19] Bauschke, H., Bolte, J. and Teboulle, M. (2017) A Descent Lemma beyond Lipschitz Gradient Continuity: First-Order Methods Revisited and Applications. *Mathematics of Operations Research*, **42**, 330-348. https://doi.org/10.1287/moor.2016.0817

[20] Asuncion, A. and Newman, D. (2007) UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[21] Chen, X., Lu, Z., *et al*. (2016) Penalty Methods for a Class of Non-Lipschitz Optimization Problems. *SIAM Journal on Optimization*, **26**, 1465-1492. https://doi.org/10.1137/15M1028054