Accelerating Adaptive Online Learning by Matrix Approximation

Yuanyu Wan and Lijun Zhang(✉)

National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
{wanyy,zhanglj}@lamda.nju.edu.cn

Abstract. Adaptive subgradient methods are able to leverage the second-order information of functions to improve the regret, and have become popular for online learning and optimization. According to the amount of information used, adaptive subgradient methods can be divided into diagonal-matrix version (ADA-DIAG) and full-matrix version (ADA-FULL). In practice, ADA-DIAG is the most commonly adopted rather than ADA-FULL, because ADA-FULL is computationally intractable in high dimensions though it has smaller regret when gradients are correlated. In this paper, we propose to employ techniques of matrix approximation to accelerate ADA-FULL, and develop two methods based on random projections. Compared with ADA-FULL, at each iteration, our methods reduce the space complexity from $O(d^2)$ to $O(\tau d)$ and the time complexity from $O(d^3)$ to $O(\tau^2 d)$ where d is the dimensionality of the data and $\tau \ll d$ is the number of random projections. Experimental results about online convex optimization show that both methods are comparable to ADA-FULL and outperform other state-of-the-art algorithms including ADA-DIAG. Furthermore, the experiments of training convolutional neural networks show again that our method outperforms other state-of-the-art algorithms including ADA-DIAG.

1 Introduction

Adaptive subgradient methods (ADAGRAD) dynamically integrate knowledge of the geometry of data observed in earlier iterations to guide the direction of updating [1]. Different from the conventional online methods [2], ADAGRAD employ adaptive proximal functions to control the learning rate for each dimension, and the proximal functions are iteratively modified by the algorithm instead of tuning manually. There are two versions of adaptive subgradient methods: ADA-DIAG which uses a diagonal matrix to define the proximal function, and ADA-FULL which uses a full matrix to define the proximal function. Because ADA-FULL is computationally intractable in high dimensions, ADA-DIAG is the most commonly studied and adopted version in practice.

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-319-93037-4_32) contains supplementary material, which is available to authorized users.

D. Phung et al. (Eds.): PAKDD 2018, LNAI 10938, pp. 405–417, 2018.
https://doi.org/10.1007/978-3-319-93037-4_32

However, compared with ADA-FULL, ADA-DIAG cannot capture the correlation in gradients. As a result, the regret bound of ADA-DIAG may be worse than that of ADA-FULL when the high-dimensional data is dense and has a low-rank structure. This dilemma prompts a question as to whether we can design algorithms that possess the merits of two versions: i.e., the light computation of ADA-DIAG and the small regret of ADA-FULL. In a recent work [3], Krummenacher et al. presented two approximation algorithms to accelerate ADA-FULL, namely ADA-LR and RADAGRAD. Although ADA-LR is equipped with a regret bound, its space and time complexities are quadratic in the dimensionality d, which is unacceptable when d is large. In contrast, the space and time complexities of RADAGRAD are linear in d, but it lacks theoretical guarantees.

Along this line of research, this paper aims to attain theoretical guarantees, and at the same time keeping the computations light. Note that ADA-FULL is computationally impractical mainly due to the fact it needs to maintain a matrix of gradient outer products, and compute its square root and inverse in each round. Actually, similar problems have been encountered in online Newton step (ONS) for exponentially concave functions [4]. Recently, Luo et al. proposed to accelerate ONS using matrix sketching methods including random projections [5]. Motivated by previous work, we first propose to employ random projections to construct a low-rank approximation of gradient outer products, and manipulate this low-rank matrix in subsequent calculations. In this way, the new algorithm, named ADA-GP, reduces the space complexity from $O(d^2)$ to $O(\tau d)$ and the time complexity from $O(d^3)$ to $O(\tau^2 d)$, implying both the space and time complexities have a linear dependence on the dimensionality d.

ADA-GP achieves excellent empirical performance in our experiments. However, due to subtle independence issues, it is difficult to analyze ADA-GP theoretically. To circumvent this problem, we propose to replace the outer product matrix of gradients in ADA-FULL with the outer product matrix of data, and then develop a similar method, named ADA-DP, that applies random projections to the outer product matrix of data. The space and time complexities of ADA-DP are similar to those of ADA-GP. Moreover, we present theoretical analysis for ADA-DP when the outer product matrix of data is low-rank, and further extend to the full-rank case. In the experiments, we first examine the performance of our methods on online convex optimization, and the results demonstrate that they are highly comparable to ADA-FULL and are much more efficient. Furthermore, we conduct experiments on training convolutional neural networks, and show that ADA-GP outperforms ADA-DIAG and RADAGRAD.

Finally, we would like to emphasize the difference between this work and the recent work [3]. First, although both studies make use of random projections, our ADA-GP and ADA-DP are much more simple than ADA-LR and RADAGRAD. Second, our ADA-GP and ADA-DP are very efficient in the sense that their computational complexities are linear in the dimensionality d, and ADA-DP is equipped with theoretical guarantees. In contrast, although RADAGRAD has a similar computational cost, it does not have theoretical justifications.

2 Related Work

ADAGRAD. Adaptive subgradient methods use the second-order information to tune the step size of gradient descent adaptively [1]. For sparse data, the regret guarantee of ADAGRAD could be exponentially smaller in the dimension d than the non-adaptive regret bound. In the following, we provide a brief introduction of ADAGRAD. Note that the idea of ADAGRAD can be incorporated into either primal-dual subgradient method [6] or composite mirror descent [7]. For brevity, we take the composite mirror descent as an example.

In the t-th round, the learner needs to determine an action $\boldsymbol{\beta}_t \in \mathbb{R}^d$ and then observes a composite function $F_t(\boldsymbol{\beta}) = f_t(\boldsymbol{\beta}) + \varphi(\boldsymbol{\beta})$ where f_t and φ are convex. The learner suffers loss $F_t(\boldsymbol{\beta}_t)$, and the goal is to minimize the accumulated loss over T iterations. Let $\partial f_t(\boldsymbol{\beta})$ denote the subdifferential set of function f_t evaluated at $\boldsymbol{\beta}$ and $\mathbf{g}_t \in \partial f_t(\boldsymbol{\beta}_t)$ be a particular vector in the subdifferential set. Define the outer product matrix of gradients $\mathrm{G}_t = \sum_{i=1}^{t} \mathbf{g}_i \mathbf{g}_i^\top$. Then, we use the square root of G_t to construct a positive definite matrices H_t, and have the following two choices:

$$\mathrm{H}_t = \begin{cases} \sigma \mathrm{I}_d + \mathrm{diag}(\mathrm{G}_t)^{1/2} & \text{ADA-DIAG} \\ \sigma \mathrm{I}_d + \mathrm{G}_t^{1/2} & \text{ADA-FULL} \end{cases}$$

where $\sigma > 0$ is a parameter. The proximal term is given by $\Psi_t(\boldsymbol{\beta}) = \frac{1}{2}\langle \boldsymbol{\beta}, \mathrm{H}_t \boldsymbol{\beta} \rangle$ and the Bregman divergence associated with Ψ_t is

$$B_{\Psi_t}(\mathbf{x}, \mathbf{y}) = \Psi_t(\mathbf{x}) - \Psi_t(\mathbf{y}) - \frac{1}{2}\langle \nabla_{\Psi_t}(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle.$$

In each iteration, the composite mirror descent method updates by

$$\begin{aligned} \boldsymbol{\beta}_{t+1} &= \operatorname*{argmin}_{\boldsymbol{\beta}} \left\{ \eta \langle \mathbf{g}_t, \boldsymbol{\beta} \rangle + \eta \varphi(\boldsymbol{\beta}) + B_{\Psi_t}(\boldsymbol{\beta}, \boldsymbol{\beta}_t) \right\} \\ &= \boldsymbol{\beta}_t - \eta \mathrm{H}_t^{-1} \mathbf{g}_t, \quad \text{if } \varphi = 0 \end{aligned}$$

where $\eta > 0$ is a fixed step size. When the dimensionality d is large, ADA-FULL is impractical because the storage cost of G_t and the running time of finding its square root and inverse of H_t are unacceptable.

To make ADA-FULL scalable, Krummenacher et al. proposed two methods that approximate the proximal term $\Psi_t(\boldsymbol{\beta})$ [3]. Based on the fast randomized singular value decomposition (SVD) [8], they presented an algorithm ADA-LR that performs the following updates:

$$\begin{aligned} &\mathrm{G}_t = \mathrm{G}_{t-1} + \mathbf{g}_t \mathbf{g}_t^\top \\ &\widetilde{\mathrm{G}}_t = \mathrm{G}_t \Pi && \text{Random Projection} \\ &\mathrm{QR} = \widetilde{\mathrm{G}}_t && \text{QR-decomposition} \\ &\mathrm{B} = \mathrm{Q}^\top \mathrm{G}_t \\ &\mathrm{U} \Sigma \mathrm{V}^\top = \mathrm{B} && \text{SVD} \\ &\boldsymbol{\beta}_{t+1} = \boldsymbol{\beta}_t - \eta \mathrm{V} (\Sigma^{1/2} + \sigma \mathrm{I}_\tau)^{-1} \mathrm{V}^\top \mathbf{g}_t \end{aligned} \tag{1}$$

where $\Pi \in \mathbb{R}^{d\times\tau}$ is the random matrix of the subsampled randomized Fourier transform. We note that random projections are utilized in the 2nd step to generate a smaller matrix $\widetilde{\mathrm{G}}_t \in \mathbb{R}^{d\times\tau}$. It is easy to verify that the space and time complexities of ADA-LR are respectively $O(d^2)$ and $O(\tau d^2)$, which are still unacceptable when the d is large. To further improve the efficiency, they presented algorithm RADAGRAD by introducing more randomized approximations, the space and time complexities of which are respectively $O(\tau d)$ and $O(\tau^2 d)$. Unfortunately, RADAGRAD is a heuristic method and lacks theoretical guarantees.

As previous mentioned, in [5], Luo et al. adopted matrix sketching methods to accelerate ONS that also encounters the similar problems as ADA-FULL. Specifically, their ONS updates by $\boldsymbol{\beta}_{t+1} = \boldsymbol{\beta}_t - \mathrm{A}_t^{-1}\mathbf{g}_t$ where $\mathrm{A}_t = \alpha\mathrm{I}_d + \sum_{i=1}^{t}\eta_i\mathbf{g}_i\mathbf{g}_i^\top$, $\alpha > 0$ and $\eta_i = O(1/\sqrt{i})$ for general convex functions. We can reformulate this update rule as

$$\boldsymbol{\beta}_{t+1} = \boldsymbol{\beta}_t - \eta\mathrm{H}_t^{-1}\mathbf{g}_t$$

where $\mathrm{H}_t = \delta\mathrm{I}_d + \sum_{i=1}^{t}\frac{1}{\sqrt{i}}\mathbf{g}_i\mathbf{g}_i^\top$. To accelerate ONS, they use matrix sketching methods to calculate a low-rank approximation of $\sum_{i=1}^{t}\frac{1}{\sqrt{i}}\mathbf{g}_i\mathbf{g}_i^\top$. Motivated by [5], our work employs random projections to calculate a low-rank approximation of full matrix. But there are obviously differences between our work and this related work. First, although both our methods and RP-SON in [5] use random projections to approximate the full matrix, we further propose to use the outer product matrix of data to replace the outer product matrix of gradients which leads to ADA-DP. Note that this simple change can avoid the dependence issue that the gradient $\mathbf{g}_t$ depends on the random vectors. Second, the theoretical analysis in our work is obviously different from [5]. The only common part is the property of the random projections for low-rank data. But we further exploit the property of the random projection for full-rank data. Third, our methods and this related work are designed for different tasks. Our paper aims to accelerate ADA-FULL, and this related work aims to accelerate ONS. Note that ADA-FULL is a data dependent algorithm for general convex function and ONS is proposed to derive a logarithmic regret for exponentially concave functions.

Random Projection. Random projection [9–11] is a simple yet powerful dimensionality reduction method. For a data point $\mathbf{x} \in \mathbb{R}^n$, random projection reduces its dimensionality to τ by $\mathrm{R}^\top\mathbf{x}$, where $\mathrm{R} \in \mathbb{R}^{n\times\tau}$ is a random matrix. It has been successfully applied to many machine learning tasks including classification [12,13], regression [14], clustering [15,16], manifold learning [17,18] and optimization [19,20]. Random projection can be implemented in various different ways [21,22], and the most classical one is the Gaussian random projection, where each entry of R is sampled from a Gaussian distribution. In this paper, we focus on Gaussian random projection due to its nice theoretical properties and easy implementations.

3 Main Results

In this section, we introduce our proposed methods and theoretical results. Due to the limitation of space, we defer the proof of theoretical results to the supplementary material.

3.1 Problem Setting

To facilitate presentations, we consider the case $\varphi = 0$, and our methods can be directly extended to the general case $\varphi \neq 0$. The goal of the learner is to minimize the regret, defined as $R(T) = \sum_{t=1}^{T} f_t(\boldsymbol{\beta}_t) - \sum_{t=1}^{T} f_t(\boldsymbol{\beta}^*)$ where $\boldsymbol{\beta}^*$ is a fixed optimal predictor.

3.2 The Proposed ADA-GP Method

From previous discussions, we know that if one can find a low-rank matrix to approximate G_t, then both space and time complexities of ADA-FULL can be reduced dramatically. Random projections provide an elegant way for low-rank matrix approximations, as explained below.

Define

$$\mathrm{A}_t^\top = [\mathbf{g}_1, ..., \mathbf{g}_t] \in \mathbb{R}^{d \times t}, \ \mathrm{R}_t = [\mathbf{r}_1, ..., \mathbf{r}_t] \in \mathbb{R}^{\tau \times t}$$

where the i-th column of $\mathrm{A}_t^\top$ is gradient $\mathbf{g}_i$, and each entry of R_t is a Gaussian random variable drawn from $\mathcal{N}(0, 1/\tau)$ independently. Then, we have

$$\mathrm{G}_t = \mathrm{A}_t^\top \mathrm{A}_t, \ \mathbb{E}[\mathrm{R}_t^\top \mathrm{R}_t] = \mathrm{I}_d.$$

To accelerate the computation, we define

$$\mathrm{S}_t = \mathrm{R}_t \mathrm{A}_t = \sum_{i=1}^{t} \mathbf{r}_i \mathbf{g}_i^\top \in \mathbb{R}^{\tau \times d}.$$

Note that S_t can be calculated on the fly as $\mathrm{S}_t = \mathrm{S}_{t-1} + \mathbf{r}_t \mathbf{g}_t^\top$. When τ is large enough, we expect $\mathrm{R}_t^\top \mathrm{R}_t \approx \mathrm{I}_d$, implying

$$\mathrm{S}_t^\top \mathrm{S}_t = \mathrm{A}_t^\top \mathrm{R}_t^\top \mathrm{R}_t \mathrm{A}_t \approx \mathrm{A}_t^\top \mathrm{A}_t = \mathrm{G}_t.$$

Thus, $\mathrm{S}_t^\top \mathrm{S}_t$ could be used as a low-rank approximation of G_t. The matrix H_t in the proximal term can be redefined as

$$\mathrm{H}_t = \sigma \mathrm{I}_d + (\mathrm{S}_t^\top \mathrm{S}_t)^{1/2}.$$

Let SVD of S_t be $\mathrm{S}_t = \mathrm{U} \Sigma \mathrm{V}^\top$, then we have $\mathrm{S}_t^\top \mathrm{S}_t = \mathrm{V} \Sigma^2 \mathrm{V}^\top$ and $\mathrm{H}_t = \sigma \mathrm{I}_d + \mathrm{V} \Sigma \mathrm{V}^\top$. According to Woodbury Formula [23], we have

$$\begin{aligned} \mathrm{H}_t^{-1} &= (\sigma \mathrm{I}_d + \mathrm{V} \Sigma \mathrm{V}^\top)^{-1} \\ &= \frac{1}{\sigma} \big(\mathrm{I}_\mathrm{d} - \mathrm{V} (\sigma \mathrm{I}_\tau + \Sigma)^{-1} \Sigma \mathrm{V}^T \big). \end{aligned}$$

Algorithm 1. ADA-GP

Input: $\eta > 0, \sigma > 0, \tau$, $\mathrm{S} = \mathbf{0}_{\tau \times d}$,$\boldsymbol{\beta}_1 = \mathbf{0}$;
for $t = 1, ..., T$ **do**
  Receive $\mathbf{g}_t = \nabla f_t(\beta_t)$
  $\mathrm{S}_t = \mathrm{S}_{t-1} + \mathbf{r}_t \mathbf{g}_t^\top$ {Random Projections}
  $\mathrm{U}\Sigma\mathrm{V}^\top = \mathrm{S}_t$ {SVD}
  $\boldsymbol{\beta}_{t+1} = \boldsymbol{\beta}_t - \frac{\eta}{\sigma}\big(\mathbf{g}_t - \mathrm{V}(\sigma \mathrm{I}_\tau + \Sigma)^{-1}\Sigma\mathrm{V}^\top \mathbf{g}_t\big)$
end for

As a result, in the t-th round, our algorithm performs the following updates

$$\begin{aligned} &\mathrm{S}_t = \mathrm{S}_{t-1} + \mathbf{r}_t \mathbf{g}_t^\top \quad \text{Random Projection} \\ &\mathrm{U}\Sigma\mathrm{V}^\top = \mathrm{S}_t \qquad\qquad \text{SVD} \\ &\boldsymbol{\beta}_{t+1} = \boldsymbol{\beta}_t - \frac{\eta}{\sigma}\big(\mathbf{g}_t - \mathrm{V}(\sigma \mathrm{I}_\tau + \Sigma)^{-1}\Sigma\mathrm{V}^\top \mathbf{g}_t\big). \end{aligned} \tag{2}$$

The detailed procedure is summarized in Algorithm 1, and named as adaptive online learning with gradient projection (ADA-GP).

Remark. First, it is easy to verify the time and space complexities of our ADA-GP is $O(\tau d)$ and $O(\tau^2 d)$, respectively. Thus, both of them are linear in the dimensionality d. Second, comparing (2) with (1), we observe that our updating rules are much more simple than those of ADA-LR. Note that the RADAGRAD algorithm of [3] is even more complicated than ADA-LR.

3.3 The Proposed ADA-DP Method

Although ADA-GP performs very well in our experiments, it is difficult to establish a regret bound due to dependence issues. To be specific, the gradient $\mathbf{g}_t$ depends on the random vectors $[\mathbf{r}_1, \cdots, \mathbf{r}_{t-1}]$, and as a result, standard concentration inequalities cannot be directly applied [24].

To avoid the aforementioned problem, we propose a strategy to get ride of the dependence issues and the new algorithm is equipped with theoretical guarantees. We consider the case $f_t(\boldsymbol{\beta}_t) = l(\boldsymbol{\beta}_t^\top \mathbf{x}_t)$ where $\mathbf{x}_t$ is a data vector. Then, we assume the data points $\mathbf{x}_1, \ldots, \mathbf{x}_t$ are independent from our algorithm. The key idea is to replace the outer product matrix of gradients G_t with the outer product matrix of data $\mathrm{X}_t = \sum_{i=1}^t \mathbf{x}_i \mathbf{x}_i^\top$. Accordingly, H_t will be defined as $\sigma \mathrm{I}_d + \mathrm{X}_t^{1/2}$. To accelerate computations, our problem becomes finding a low-rank approximation of X_t.

Let $\mathrm{C}_t^\top = [\mathbf{x}_1, ..., \mathbf{x}_t] \in \mathbb{R}^{d \times t}$, where each column is a data vector. Similar to ADA-GP, we define

$$\mathrm{S}_t = \mathrm{R}_t \mathrm{C}_t = \sum_{i=1}^t \mathbf{r}_i \mathbf{x}_i^\top \in \mathbb{R}^{\tau \times d}$$

Algorithm 2. ADA-DP

Input: $\eta > 0, \sigma > 0, \tau$, $\mathrm{S} = \mathbf{0}_{\tau \times d}, \boldsymbol{\beta}_1 = \mathbf{0}$;
for $t = 1, ..., T$ **do**
  Receive $\mathbf{x}_t$ and $\mathbf{g}_t = \nabla f_t(\beta_t) = l'(\boldsymbol{\beta}_t^\top \mathbf{x}_t)\mathbf{x}_t$
  $\mathrm{S}_t = \mathrm{S}_{t-1} + \mathbf{r}_t \mathbf{x}_t^\top$ {Random Projections}
  $\mathrm{U}\Sigma\mathrm{V}^\top = \mathrm{S}_t$ {SVD}
  $\boldsymbol{\beta}_{t+1} = \boldsymbol{\beta}_t - \frac{\eta}{\sigma}\left(\mathbf{g}_t - \mathrm{V}(\sigma \mathrm{I}_\tau + \Sigma)^{-1}\Sigma \mathrm{V}^\top \mathbf{g}_t\right)$
end for

where $\mathrm{R}_t \in \mathbb{R}^{\tau \times t}$ is the Gaussian random matrix. In this case, since R_t is independent of C_t, we have

$$\mathbb{E}[\mathrm{S}_t^\top \mathrm{S}_t] = \mathrm{C}_t^\top \mathbb{E}[\mathrm{R}_t^\top \mathrm{R}_t]\mathrm{C}_t = \mathrm{C}_t^\top \mathrm{C}_t = \mathrm{X}_t$$

which means $\mathrm{S}_t^\top \mathrm{S}_t$ is an unbiased estimation of X_t.

The rest steps are similar to that of ADA-GP. The detailed procedure is summarized in Algorithm 2, named as adaptive online learning with data projection (ADA-DP). It is obvious that the computation cost of ADA-DP is almost the same as that of ADA-GP. Thus, both the space and time complexities of ADA-DP are linear in d.

The main advantage of ADA-DP is that it has formal theoretical guarantees. We first consider the case that the data matrix C_T is low-rank, and have the following theorem regarding the regret of Algorithm 2.

Theorem 1. *Let $r \ll d$ be the rank of C_T, and $0 < \delta < 1$ be the confidence parameter. Assume each entry of $\mathbf{r}_t \in \mathbb{R}^\tau$ is a Gaussian random variable drawn from $\mathcal{N}(0, 1/\tau)$ independently, $\tau = \Omega(\frac{r+\log(T/\delta)}{\epsilon^2})$ and $\sigma \geq 0$, then ADA-DP ensures*

$$R(T) \leq \frac{\sigma}{2\eta}\|\boldsymbol{\beta}_*\|_2^2 + \frac{1}{2\eta}\max_{t \leq T}\|\boldsymbol{\beta}^* - \boldsymbol{\beta}_t\|_2^2 tr(\mathrm{X}_T^{1/2}) + \frac{2\eta}{\sqrt{1-\epsilon}}\max_{t \leq T} l'(\boldsymbol{\beta}_t^\top \mathbf{x}_t)^2 \mathrm{tr}(\mathrm{X}_T^{1/2}) + \frac{\epsilon}{2\eta}\max_{t \leq T}\|\boldsymbol{\beta}^* - \boldsymbol{\beta}_t\|_2^2 \sum_{t=1}^{T}\|\mathrm{X}_t^{1/2}\|$$

with probability at least $1 - \delta$.

Remark. Theorem 1 means that we can set the number of random projections as $\tau = \widehat{\Omega}(r)$ when the data matrix is low-rank.

When the data matrix is full-rank, Theorem 1 is inappropriate because it implies the number of random projections is on the order of the dimensionality. Let $\lambda_i(\cdot)$ be the i-th largest eigenvalue of a matrix. For the full-rank case, we further establish the following theorem to bound the regret of Algorithm 2.

Theorem 2. *Let $c \geq 1/32$, $\sigma \geq \sqrt{\alpha} > 0$, $\sigma_{ti}^2 = \lambda_i(\mathrm{C}_t^\top \mathrm{C}_t)$, $\tilde{r}_t = \sum_i \frac{\sigma_{ti}^2}{\alpha + \sigma_{ti}^2}$, $\tilde{r}_* = \max_{1 \leq t \leq T} \tilde{r}_t$, $\sigma_{*1}^2 = \max_{1 \leq t \leq T} \sigma_{t1}^2$, and $0 < \delta < 1$. Assume each entry of $\mathbf{r}_t \in \mathbb{R}^\tau$*

is an independent random Gaussian variable drawn from $\mathcal{N}(0, 1/\tau)$, $\tau \geq \frac{\tilde{r}_* \sigma_{*1}^2}{c\epsilon^2(\alpha+\sigma_{*1}^2)} \log \frac{2dT}{\delta}$ *and then ADA-RP ensures*

$$R(T) \leq \frac{\sigma}{2\eta}\|\boldsymbol{\beta}_*\|_2^2 + \frac{1}{2\eta}\max_{t\leq T}\|\boldsymbol{\beta}^* - \boldsymbol{\beta}_t\|_2^2 tr(\mathrm{X}_T^{1/2}) + \frac{2\eta}{\sqrt{1-\epsilon}}\max_{t\leq T} l'(\boldsymbol{\beta}_t^\top \mathbf{x}_t)^2 \mathrm{tr}(\mathrm{X}_T^{1/2}) + \frac{\epsilon}{2\eta}\max_{t\leq T}\|\boldsymbol{\beta}^* - \boldsymbol{\beta}_t\|_2^2 \sum_{t=1}^{T}\|\mathrm{X}_t^{1/2}\| + \frac{\sqrt{\epsilon\alpha}T}{\eta}\max_{t\leq T}\|\boldsymbol{\beta}^* - \boldsymbol{\beta}_t\|_2^2.$$

with probability at least $1 - \delta$.

Remark. Following [20], we introduce the quantity $\tilde{r}_t$ to measure the effective rank of the data matrix C_t. When the eigenvalues of $\mathrm{C}_t^\top \mathrm{C}_t$ decrease rapidly, $\tilde{r}_t$ could be significantly smaller than d, even when C_t is full-rank. Compared with Theorem 1, the upper bound in this theorem contains an additional term caused by the approximation error of full-rank matrices. Note that Theorem 2 means that we can set the number of random projections as $\tau = \widehat{\Omega}(\max_t \tilde{r}_t)$ when the data matrix has low effective rank.

Note that our methods and theories can be extended to the general case $\varphi \neq 0$. We just need to replace the updating rule as

$$\boldsymbol{\beta}_{t+1} = \operatorname*{argmin}_{\boldsymbol{\beta}} \left\{\eta\langle \mathbf{g}_t, \boldsymbol{\beta}\rangle + \eta\varphi(\boldsymbol{\beta}) + B_{\Psi_t}(\boldsymbol{\beta}, \boldsymbol{\beta}_t)\right\}.$$

Although the updating of $\boldsymbol{\beta}_{t+1}$ may not have closed-form solution, the computational cost of H_t^{-1} can still be reduced dramatically. The regret bound remains on the same order.

4 Experiments

In this section, we conduct numerical experiments to demonstrate the efficiency and effectiveness of our methods.

4.1 Online Convex Optimization

First, we compare our two methods against ADA-FULL, ADA-DIAG, RADAGRAD [3] and RP-SON [5] on a synthetic data, which is approximately low-rank. Let $\boldsymbol{\beta}_* = \hat{\boldsymbol{\beta}}_*/\|\hat{\boldsymbol{\beta}}_*\|_2$ where each entry of $\hat{\boldsymbol{\beta}}_*$ is drawn independently from $\mathcal{N}(0, 1)$. We consider the problem of online regression where $f_t(\boldsymbol{\beta}) = |\boldsymbol{\beta}^\top \mathbf{x}_t - y_t|$ and $y_t = \boldsymbol{\beta}_*^\top \mathbf{x}_t$. We generate a regression dataset with $T = 10000$ and $d = 500$. In order to meet the requirement of low-rankness, each data point $\mathbf{x}_t$ is sampled independently from a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ where $\boldsymbol{\mu} = \mathbf{1}$ and Σ has rapidly decaying eigenvalues $\lambda_j(\Sigma) = \lambda_0 j^{-\alpha}$ with $\alpha = 2$ and $\lambda_0 = 100$.

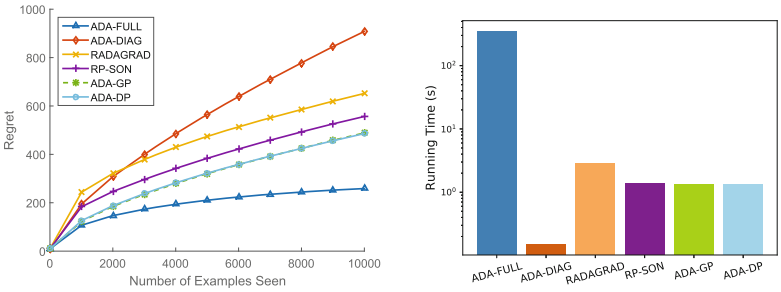


Fig. 1. The left is the comparison of regret among different algorithms on the synthetic data, and the right is the comparison of running time

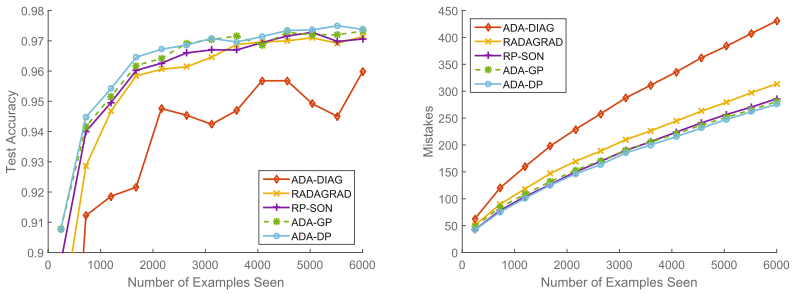


Fig. 2. The left is the comparison of test accuracy among different algorithms on Gisette dataset, and the right is the comparison of mistakes during training

The parameters η and σ are searched in $\{1e{-}4, 1e{-}3, \cdots, 100\}$, and we choose the best values for each algorithm. For fairness, all the algorithms are running with the same permutations of the function sequence. For ADA-GP, ADA-DP, RADAGRAD and RP-SON, their results are averaged over 5 runs. Figure 1 shows the regret and running time of different algorithms where we set $\tau = 10$ for methods using random projections. The regret of our two methods are very close and better than ADA-DIAG, RADAGRAD and RP-SON, which indicates our methods approximate ADA-FULL very well. Moreover, our two methods are obviously faster than ADA-FULL according to the comparison of running time.

Second, following [1], we perform online classification with the squared hinge loss (i.e., $f_t(\boldsymbol{\beta}) = \frac{1}{2}\left(\max\left(0, 1 - y_t \boldsymbol{\beta}^\top \mathbf{x}_t\right)\right)^2$) to evaluate the performance of our methods. In each round, the learning algorithm receives a single example and ends with a single pass through the training data. There are two metrics to measure the performance: the online mistakes and the offline accuracy on the testing data.

We conduct numerical experiments on a real world dataset from LIBSVM repository [25]: Gisette which is high-dimensional (i.e. $d = 5000$) and dense. Similar as before, parameters η and σ are searched in $\{1e{-}4, 1e{-}3, \cdots, 10\}$ and

$\{2e{-}4, 2e{-}3, \cdots, 20\}$, and we choose the best values for each algorithm. To reduce the computational cost, we set the number of projections $\tau = 10$ for methods using random projections. We omit the result of ADA-FULL, because it is too slow.

For training data, we generate 5 random permutations, and report the average result. Figure 2 shows the comparison of test accuracy and mistakes among different algorithms. From Fig. 2, we have some conclusions as following. First, the performance of ADA-DIAG is much worse than all the other methods, which means only keeping a diagonal matrix is insufficient to capture the second-order information. Second, our two methods, ADA-GP and ADA-DP, are better than RADAGRAD and RP-SON. Third, ADA-GP and ADA-DP are close to ADA-FULL, which indicates that random projections cause little adverse affect on the performance.

4.2 Non-convex Optimization in Convolutional Neural Networks

Recently, ADA-DIAG becomes popular for non-convex optimization such as training neural networks, and Krummenacher et al. also show that RADAGRAD performs well for training neural networks [3]. Therefore, we also examine the performance of our method on training convolutional neural networks (CNN). Because the convolutional layer does not meet the case $f_t(\boldsymbol{\beta}_t) = l(\boldsymbol{\beta}_t^\top \mathbf{x}_t)$, we only perform ADA-GP on training CNN. We use the simple and standard architecture shown in Fig. 3 to perform classification on the MNIST [26], CIFAR10 [27] and SVHN [28] datasets.

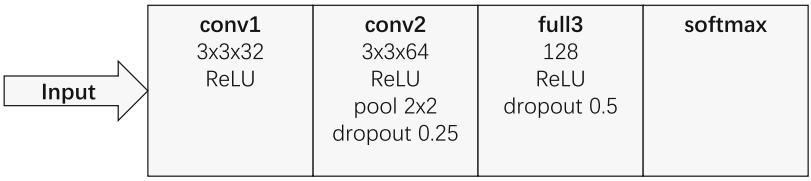
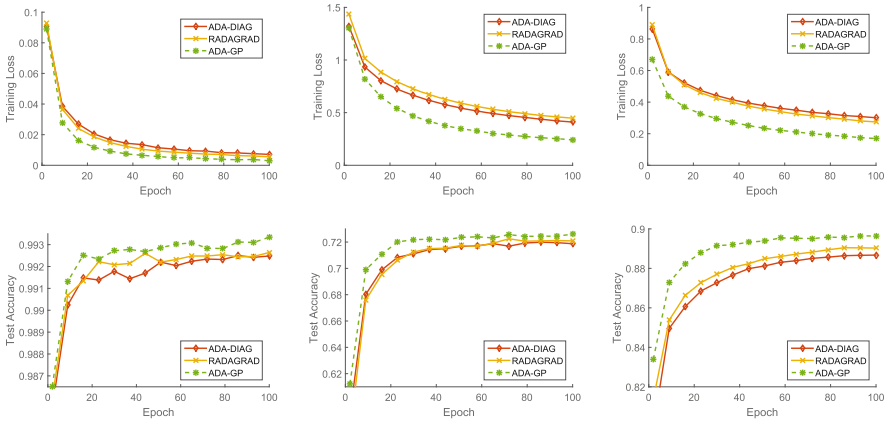


Fig. 3. The 4-layer CNN architecture used in our experiment

Parameters η of all algorithms and δ of ADA-GP and RADAGRAD are searched in $\{1e{-}4, 1e{-}3, \cdots, 1\}$. For ADA-DIAG, δ is set to $1e{-}8$ as it is typically recommended. We choose the best values for each algorithm. Following as [3], we only consider applying ADA-GP and RADAGRAD to the convolutional layer, and other layers are still trained with ADA-DIAG for all datasets. For all algorithms, we run 5 times with batch size 128 and report the average results. Figure 4 shows the comparison of training loss and test accuracy during training among different algorithms where we set $\tau = 20$. We find that ADA-GP obviously improves the performance of ADA-DIAG on all datasets, and note that RADAGRAD is outperformed by ADA-DIAG in term of training loss on CIFAR10. This results shows that ADA-GP is a better approximation of ADA-FULL than RADAGRAD.



(a) MNIST (b) CIFAR10 (c) SVHN

Fig. 4. The comparison of training loss (top row) and test accuracy (bottom row) among different algorithms

5 Conclusions and Future Work

In this paper, we present ADA-GP and ADA-DP to approximate ADA-FULL using random projections. The time and space complexities of both algorithms are linear in the dimensionality d, and thus they are able to accelerate the computation significantly. Furthermore, according to our theoretical analysis, the number of random projections in ADA-DP is on the order of the low rank or low effective rank. Numerical experiments on online convex optimization show that our methods outperform ADA-DIAG, RADAGRAD and RP-SON and are close to ADA-FULL. And experiments on training convolutional neural networks show that ADA-GP outperforms ADA-DIAG and RADAGRAD.

Besides random projection, there exist other ways for low-rank matrix approximations, such as matrix sketching [11]. In the future, we will investigate different techniques to approximate ADA-FULL.

Acknowledgements. This work was partially supported by the NSFC (61603177), JiangsuSF (BK20160658), YESS (2017QNRC001), and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

1. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**, 2121–2159 (2011)
2. Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: Proceedings of the 20th International Conference on Machine Learning, pp. 928–936 (2003)

3. Krummenacher, G., McWilliams, B., Kilcher, Y., Buhmann, J.M., Meinshausen, N.: Scalable adaptive stochastic optimization using random projections. Adv. Neural Inf. Process. Syst. **29**, 1750–1758 (2016)
4. Hazan, E., Agarwal, A., Kale, S.: Logarithmic regret algorithms for online convex optimization. Mach. Learn. **69**(2), 169–192 (2007)
5. Luo, H., Agarwal, A., Cesa-Bianchi, N., Langford, J.: Efficient second order online learning by sketching. Adv. Neural Inf. Process. Syst. **29**, 902–910 (2016)
6. Xiao, L.: Dual averaging method for regularized stochastic learning and online optimization. Adv. Neural Inf. Process. Syst. **22**, 2116–2124 (2009)
7. Duchi, J., Shalev-Shwartz, S., Singer, Y., Tewari, A.: Composite objective mirror descent. In: Proceedings of the 23rd Annual Conference on Learning Theory, pp. 14–26 (2010)
8. Nalko, N., Martinsson, P.G., Tropp, J.A.: Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev. **53**(2), 217–288 (2011)
9. Kaski, S.: Dimensionality reduction by random mapping: Fast similarity computation for clustering. In: Proceedings of the 1998 IEEE International Joint Conference on Neural Networks, vol. 1, pp. 413–418 (1998)
10. Magen, A., Zouzias, A.: Low rank matrix-valued Chernoff bounds and approximate matrix multiplication. In: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1422–1436 (2011)
11. Woodruff, D.P.: Sketching as a tool for numerical linear algebra. Found. Trends Mach. Learn. **10**(1–2), 1–157 (2014)
12. Fradkin, D., Madigan, D.: Experiments with random projections for machine learning. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 517–522 (2003)
13. Rahimi, A., Recht, B.: Random features for large-scale kernel machines. Adv. Neural Inf. Process. Syst. **21**, 1177–1184 (2008)
14. Maillard, O.A., Munos, R.: Linear regression with random projections. J. Mach. Learn. Res. **13**, 2735–2772 (2012)
15. Fern, X.Z., Brodley, C.E.: Random projection for high dimensional data clustering: a cluster ensemble approach. In: Proceedings of the 20th International Conference on Machine Learning, pp. 186–193 (2003)
16. Boutsidis, C., Zouzias, A., Drineas, P.: Random projections for k-means clustering. Adv. Neural Inf. Process. Syst. **23**, 298–306 (2010)
17. Dasgupta, S., Freund, Y.: Random projection trees and low dimensional manifolds. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, pp. 537–546 (2008)
18. Freund, Y., Dasgupta, S., Kabra, M., Verma, N.: Learning the structure of manifolds using random projections. Adv. Neural Inf. Process. Syst. **21**, 473–480 (2008)
19. Gao, W., Jin, R., Zhu, S., Zhou, Z.H.: One-pass AUC optimization. In: Proceedings of the 30th International Conference on Machine Learning, pp. 906–914 (2013)
20. Zhang, L., Mahdavi, M., Jin, R., Yang, T., Zhu, S.: Recovering the optimal solution by dual random projection. In: Proceedings of the 26th Annual Conference on Learning Theory, pp. 135–157 (2013)
21. Achlioptas, D.: Database-friendly random projections: Johnson-Lindenstrauss with binary coins. J. Comput. Syst. Sci. **66**(4), 671–687 (2003)
22. Liberty, E., Ailon, N., Singer, A.: Dense fast random projections and lean walsh transforms. Discrete Computat. Geom. **45**(1), 34–44 (2011)
23. Hager, W.W.: Updating the inverse of a matrix. SIAM Rev. **31**(2), 221–239 (1989)

24. Tropp, J.A.: An introduction to matrix concentration inequalities. Found. Trends Mach. Learn. **8**(1–2), 1–230 (2015)
25. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 1–27 (2011)
26. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**, 2278–2324 (1998)
27. Krizhevsky, A.: Learning multiple layers of features from tiny images. Technical report, University of Toronto (2009)
28. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011 (2011)