Regularized Stochastic Learning with Dual Averaging Methods : An Application to Linear SVM

Maxime Jumelle

April 26, 2019

This article is a brief summary of regularized dual averaging (RDA) methods [1], with an application on linear SVM with hinge loss. We aim to demonstrate that on the studied dataset, RDA methods performs better than regular online algorithms such as Stochastic Gradient Descent (SGD).

1 Introduction

Regularization is a technique which finds its origin more than fifty years ago, especially when Ridge regression was proposed (A.E. Hoerl et al. [2]). This first model was originally designed for special settings of non-orthogonality situations. The overall intuition is to add information regarding learned parameters so that we can prevent over-fitted results. Later, several techniques of regularization were developed, but one especially performed above previous is the case of l_1-regularization introduced by R. Tibshirani [3] for the LASSO model. This regularization of parameters coupled with l_1 norm brought interesting situations. First, this regularization provides hard thresholding, that is to say, instead of l_2 norm as in Ridge regression, which only shrink components of parameter vector, l_1-regularization directly drops some components to 0 if there are no "useful" information on them. This act as a selection procedure, since we are able to find components that can explain output variables. Moreover, this directly prevent over-fitting as we force the model to only use necessary but critical components that can clearly explain a relationship between input (explanatory) and output (response) variables. Then, l_1-regularization is especially designed for *sparse* problems, that is to say where number of interesting (non-null) components of parameter vector are much smaller compared to dimension of feature space. This particularly arises in high-dimensional statistics, where number of dimension d and number of data n follows $n \ll d$.

Consider a loss function f which we seek to minimize and Ψ a regularization function : in *offline* convex optimization, we suppose that f is convex as well as Ψ, and for more convenience we also suppose that f and Ψ are differentiable almost everywhere.

$$\min_{w \in \mathcal{K}} f(w; \mathbf{x}) + \Psi(w) \tag{1}$$

Equation (1) is the general setting for offline convex optimization problems, where $\mathbf{x} = (\mathbf{x}_i, y_i)_{1 \leq i \leq n}$ are explanatory variables (i.e. training set in machine learning) and w parameter vector that we seek to optimize. For instance, if we suppose that f is the ordinary least square loss function $f(w; \mathbf{x}) = \sum_{(x,y) \in \mathbf{x}} \|y - x\|_2^2$, then considering different form of regularization function Ψ, we can recover from regular models. If $\Psi(w) = \lambda \|x\|_2^2, \lambda \geq 0$, referred as l_2^2-regularization, we obtain Ridge estimator. In case where $\Psi(w) = \lambda \|x\|_1$, the l_1-regularization, then problem is similar to LASSO procedure. With mixed l_1/l_2^2-regularization, as sum of both regularization terms, we encounter Elastic Net [4] model.

$$\min_{w \in \mathcal{K}} \mathbb{E}_{\mathbf{x}}[f(w; \mathbf{x})] + \Psi(w) \tag{2}$$

However, in *online* convex optimization setting, we do not have directly access to the loss function f. Indeed, it is supposed that we are not able to compute directly loss for all instances in the training set, whether because this is too much computationally expensive, or because we do not have access to a gradient oracle. Equation (2) materializes this aspect, as we cannot minimize a random quantity, but so as we can on the expected value of this random quantity. Such algorithms are then called *online* since we only use partial information of loss at each iteration. But with this different setting, there is still a sense at defining a regularization procedure, since learned parameter with online settings can still achieve over-fitting. For Stochastic Gradient Descent (SGD), we can for example choose a custom regularization function, which would be analogous to the *offline* regularized gradient descent with identical regularization term. In this *online* setting, as we are focusing on minimizing a certain "loss", we only have all previous computations available until iteration t. Then, a commonly used metric to estimate performance of online algorithms is the regret function :

$$\text{regret}_T = \sup_{\{f_1,\dots,f_T\}\subseteq\mathcal{F}} \left\{ \sum_{t=1}^{T} f_t(x_t) - \min_x \sum_{t=1}^{T} f_t(x) \right\}$$

As we can only compute regret until all T iterations are computed, we defined empirical regret as

$$\text{regret}_T = \sum_{t=1}^{T} f_t(x_t) - \min_x \sum_{t=1}^{T} f_t(x)$$

Observed losses f_t are acquired at each iteration, so that previous observations are needed to compute the associated regret. It can be shown that, for convex functions f, we are able to provide an upper bound of $O(\sqrt{t})$.

In Section 2, we present the main algorithm an idea behind dual averaging methods. This will enable us to discuss about practical applications and reasons of why this method is developed in research articles. In Section 3, we discuss a special case that permits to accelerate convergence of algorithm. In Section 4, we present results based on previous algorithms and other popular ones among online convex optimization, with application on linear SVM with l_2-regularized hinge loss. In Section 5, we discuss about further improvements and related work.

2 Regularized Dual Averaging

In this section, we present the idea of dual averaging and propose a particular case that we refer as enhanced l_1-regularization. The key idea behind these methods is *structural convex optimization.* Most of online algorithms only act on black-box models, that is to say when we only have knowledge of gradient and function values. Adding more information related to model we are trying to optimize may bring better results as we would be able to perfectly choose a good regularization term. Starting from (2), as we denote g the subgradient of f, we decide to add a few improvements to this minimization problem :

- First, we suppose that we are not trying to minimize $f(w;\mathbf{x})$, but a *dual* version $\langle \bar{g}_t, w\rangle$ where

$$\bar{g}_t = \frac{1}{t}\sum_{i=1}^{t} g_i$$

  with $g_i \overset{\triangle}{=} g(x_i), x_i \in \mathbf{x}_i$ a randomly selected observation from the dataset at each iteration i. Here, the name *dual* refers to the fact that average subgradients $\bar{g}_t$ lives in a dual space vector of linear span of w.

- Moreover, an auxiliary function h, potentially strongly convex, is added as another minimizing term. This auxiliary function is multiplied by a $\frac{\beta_t}{t}$ term, with $(\beta_t)_{t\geq 0}$ and nonnegative and nondecreasing term, such that, ideally, $\frac{\beta_t}{t} \to 0$. The intuition to add such a decreasing value in front of h, which also acts as a regularization term, is to strongly penalize w during the first iterations, and reducing penalization as over iterations, w is already well fitted.

As a result, we are minimizing sum of three terms at each iteration : dual gradient $\langle \bar{g}_t, w \rangle$, regularization term Ψ and auxiliary function h. The RDA algorithm naturally arises from these steps :

```
Algorithm 1: Regularized Dual Algorithm (RDA) method
  Input: $h$ auxiliary function and $(\beta_t)_{t\geq 1}$ nonnegative and nondecreasing sequence.
  Set $w_1 = \arg\min_w h(w)$ and $g_0 = 0$. for $t = 1, \dots, T$ do
      Compute subgradient $\partial f_t(x_t)$ for a given $f_t$
      Updates dual subgradient
          $$\bar{g}_t = \frac{t-1}{t}\bar{g}_{t-1} + \frac{1}{t} g_t$$
      Compute next weight
          $$w_{t+1} = \arg\min_w \left\{ \langle \bar{g}_t, w \rangle + \Psi(w) + \frac{\beta_t}{t} h(w) \right\}$$
  end
  return $w_{T+1}$
```

Adding the last term allows to directly force sparse pattern for w : the more regularization strength we provide to h, the more sparse w will be from the first iterations. Then, after a few iterations, w will already be sparse enough, so that our new focus is to minimize the dual term alongside regularization term Ψ. Note that this is the general case, but if we choose special Ψ and h, we could have strong sparse solution, as for l_1-regularization. Let us consider l_1-regularization $\Psi(w) = \lambda\|w\|_1$ and mixed l_1/l_2^2-regularization for auxiliary function $h(w) = \frac{1}{2}\|w\|_2^2 + \rho\|w\|_1$. In this case, the minimization step can be expressed as follows

$$\begin{aligned} w_{t+1} &= \arg\min_w \left\{ \langle \bar{g}_t, w \rangle + \Psi(w) + \frac{\beta_t}{t} h(w) \right\} \\ &= \arg\min_w \left\{ \langle \bar{g}_t, w \rangle + \lambda\|w\|_1 + \frac{\gamma}{\sqrt{t}} \left(\frac{1}{2}\|w\|_2^2 + \rho\|w\|_1 \right) \right\} \\ &= \arg\min_w \left\{ \langle \bar{g}_t, w \rangle + \left(\lambda + \frac{\gamma\rho}{\sqrt{t}} \right) \|w\|_1 + \frac{\gamma}{2\sqrt{t}}\|w\|_2^2 \right\} \end{aligned}$$

When $\gamma > 0, \rho > 0$ and $\lambda \geq 0$, we can express w_{t+1} as (Rockafellar, 1970, Section 27) :

$$w_{t+1}^{(j)} = \begin{cases} 0 & \text{if } |\bar{g}_t^{(j)}| \leq \lambda^{\text{RDA}} \\ -\frac{\sqrt{t}}{\gamma}(\bar{g}_t^{(j)} - \lambda^{\text{RDA}} \text{sgn}(\bar{g}_t^{(j)})) & \text{else} \end{cases}$$

with $\lambda^{\text{RDA}} = \lambda + \frac{\gamma\rho}{\sqrt{t}}$ and j denoting the j-th component of feature space. Then, Algortihm 3 becomes

Algorithm 2: Enhanced l_1-RDA method

```
Input: $\gamma > 0$ and $\rho \geq 0$.
Set $w_1 = 0$ and $g_0 = 0$. for $t = 1, \dots, T$ do
    Compute subgradient $\partial f_t(x_t)$ for a given $f_t$
    Updates dual subgradient
        $$\bar{g}_t = \frac{t-1}{t}\bar{g}_{t-1} + \frac{1}{t}g_t$$
    Let $\lambda_t^{\mathrm{RDA}} = \sigma + \gamma\rho\sqrt{t}$ and compute next weight
        $$w_{t+1}^{(i)} = \begin{cases} 0 & \text{if } |\bar{g}_t^{(i)}| \leq \lambda \\ -\frac{\sqrt{t}}{\gamma}\left(\bar{g}_t^{(i)} - \sigma \ \mathrm{sgn}(\bar{g}_t^{(i)})\right) & \text{otherwise} \end{cases}$$
end
return $w_{T+1}$
```

Which is name enhanced l_1-RDA methods. In the case where $\gamma = 0$, then $h(w)$ term vanishes and so minimization step is easier to compute, since we only aim at minimizing $w \mapsto \langle \bar{g}_t, w\rangle + \lambda\|w\|_1$.

$$w_{t+1}^{(j)} = \begin{cases} 0 & \text{if } |\bar{g}_t^{(j)}| \leq \lambda \\ -(\bar{g}_t^{(j)} - \lambda \mathrm{sgn}(\bar{g}_t^{(j)})) & \text{else} \end{cases}$$

As experiment will show, this previous variants may require higher computation time. Fortunately, the author developer an accelerated version of RDA, called ARDA, which improves computation time. This version requires gradients function g to be L-Lipschitz, as it provides better bounds and needed considerations to improve speed of convergence.

Algorithm 3: Regularized Dual Algorithm (RDA) method

```
Input: $h$ auxiliary function and $(\alpha_t)_{t\geq1}, (\beta_t)_{t\geq1}$ nonnegative and nondecreasing sequences.
Set $w_1 = \arg\min_w h(w)$ and $g_0 = 0$. for $t = 1, \dots, T$ do
    Compute
        $$A_t = A_{t-1} + \alpha_t \qquad \theta_t = \frac{\alpha_t}{A_t}$$
    Compute the query point
        $$u_t = (1-\theta_t)w_{t-1} + \theta_t v_{t-1}$$
    Compute subgradient $g_t$ for a given $x_t$ updating weighted average :
        $$\bar{g}_t = (1-\theta_t)\bar{g}_{t-1} + \theta_t g_t$$
    Compute next weight
        $$w_{t+1} = \arg\min_w \left\{ \langle \bar{g}_t, w\rangle + \Psi(w) + \frac{L+\beta_t}{A_t}h(w) \right\}$$
    Compute interpolation of $w_t$ :
        $$w_t = (1-\theta_t)w_{t-1} + \theta_t v_t$$
end
return $w_{T+1}$
```

For instance, if we set $\alpha_t = 1$ and $\beta_t = \gamma\sqrt{t+1}$ then the previous algorithm would be very similar to enhanced l_1-RDA if we choose the same regularization part.

3 Experiment

In order to study this algorithm, we propose an experiment on MNIST dataset [5] with linear SVM model. Our goal is to train a linear SVM with

$$f(w;\mathbf{x}) = \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n}\max(0, 1 - y_i(\mathbf{x}_i^\top w))$$

First, we compare different stochastic algorithms that do not inherently has regularization interface. This will provide us a common baseline for popular stochastic algorithms. Then, we will experiment on the same setting with RDA and its two variants, so that we would be able to compare with previous popular methods.

3.1 Comparison of stochastic algorithms

In this first experiment, we compared 4 online algorithms :

- Stochastic Randomized Diagonal ONS
- Stochastic Randomized Diagonal AdaGrad
- Stochastic Mirror Descent (SMD)
- Stochastic Exponentiated Gradients (SEG)

Sparsity patterns are plotted on Figure 1. For the first three algorithms, we obtain a last sparse parameter w_T. However, average parameters $\bar{w}_T$ shows that this sparsity was not achieved until the last iterations. Because sparsity pattern appear blurred for averages, we deduce that iterations at the beginning were either not sparse or not identical between successive iterations. Both ONS and SMD algorithms finds a sparse vector that refers to digit 1. On the contrary, we are not able to distinguish a precise pattern for digit 0, which is the case for AdaGrad. For this one, pattern for digit 0 is also sparse.

For SEG algorithm, average is more noisy than the three previous algorithms, as average pattern is more dense and more spread than other average patterns. Figure 2 shows regret function for each of these methods. We have a global linear upper bound $O(T)$ for each of the first three algorithms. However, SEG method is worse than others methods on a sparsity level, potentially due to a number of iterations that is not high enough. Nevertheless, we can show a constant profile of w for SEG algorithm. Average $\bar{w}_t$ is globally similar to last iteration's parameter w_T, which indicates that parameter's component sticks to the same values after a few first iterations. Figure 2 shows linear behaviour for regret functions, where SEG is slighly better than the three others algorithms.

3.2 Comparison of SGD and RDA

In this part, we now mainly focus on comparison between SGD and derivatives RDA, that is to say enhanced l_1-regularization (ERDA) with case where $h(w) = 0$ (RDA) and accelerated RDA (ARDA) with same regularization.

Figure 3 shows sparsity patterns for each algorithm. As expected, SGD algorithm is unable to detect a sparse behaviour in our explanatory variables, as we specifically omitted l_1-regularization. Although we can see a little more thresholds on RDA algorithm, it is clear that Ψ is not sufficient to select a sparse feature subspace. On ERDA, however, we are able to see well delimited bounds of a sparse feature subspace, and this from the early stages of convergence. Indeed, average plot (second row) shows that this sparse selection is performed earlier enough to vanish all low-valued components, likely during the first dozens iterations. As well as for ARDA, the auxiliary function enables to select the sparse feature subspace directly from the first iterations, as average is also much similar to sparsity pattern of last iteration.

In Figure 5, we displayed several sparsity patterns for our three RDA variants with different values on l_1-regularization parameter λ. As $\lambda = 10^{-3}$, regular RDA methods is not able to detect a sparse solution for w_T, as sparsity pattern shows noisy profile. This is explainable as a low value on λ would "kill" l_1-regularization.

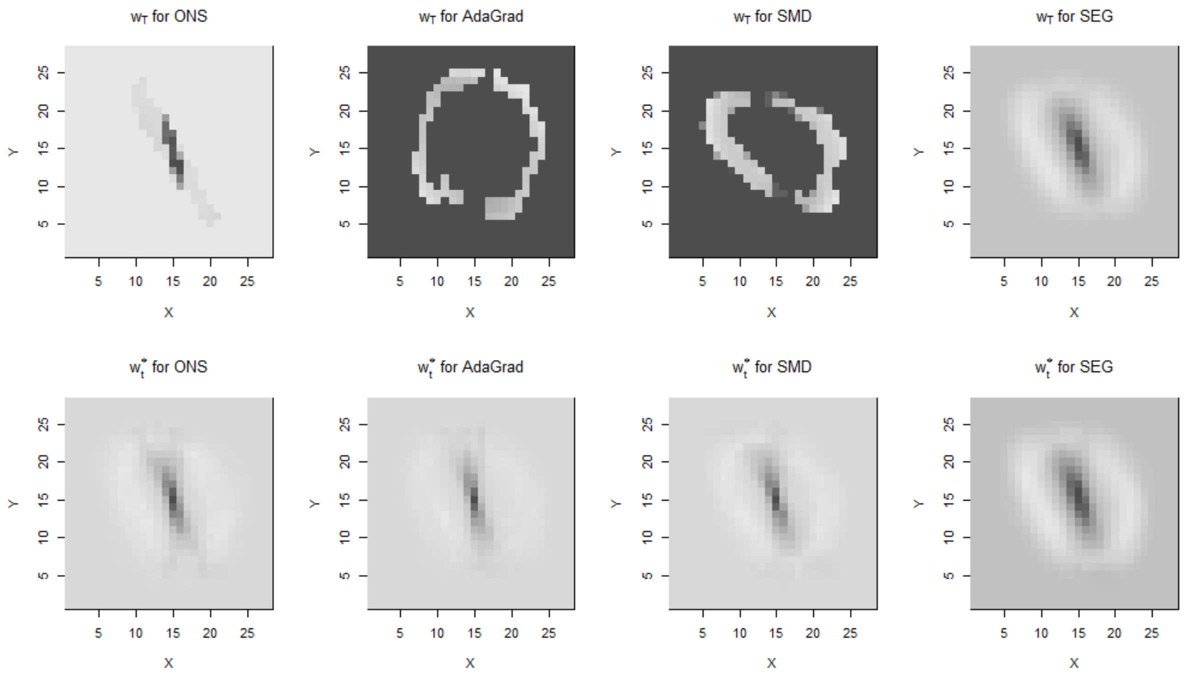

Figure 1: Sparsity patterns for $T = 200$ and $z = 0.5$. On first row are plotted parameter of last iteration w_T, expanded into a 28×28 size matrix from the 784-dimensional flattened vector. On the second row, we plot the average of parameters w_t over T iterations. Each column identifies an algorithm. White pixels refer to components that tend to 1, and black pixels those which tend to -1.

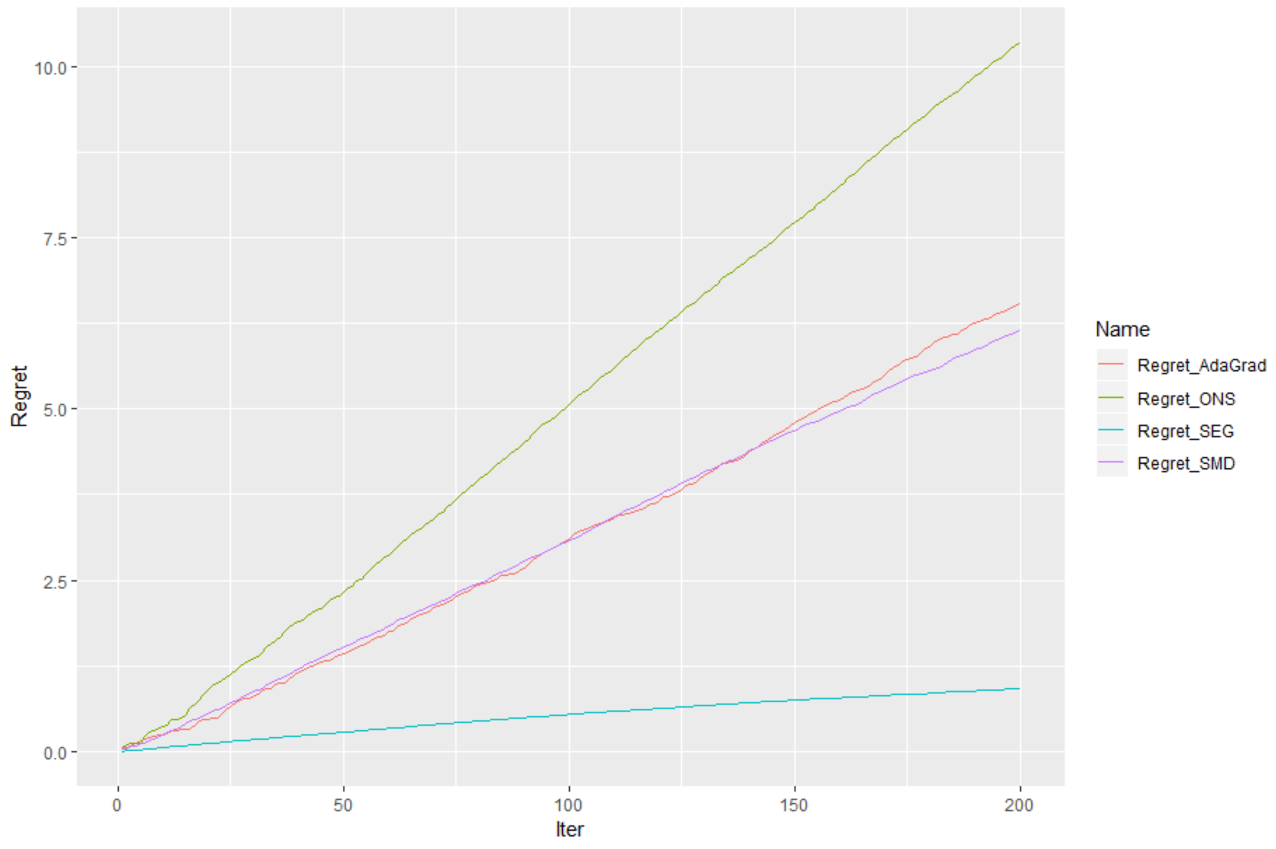

Figure 2: Regret curves for popular stochastic algorithms with $T = 200, C = 1$ and $z = 0.5$.

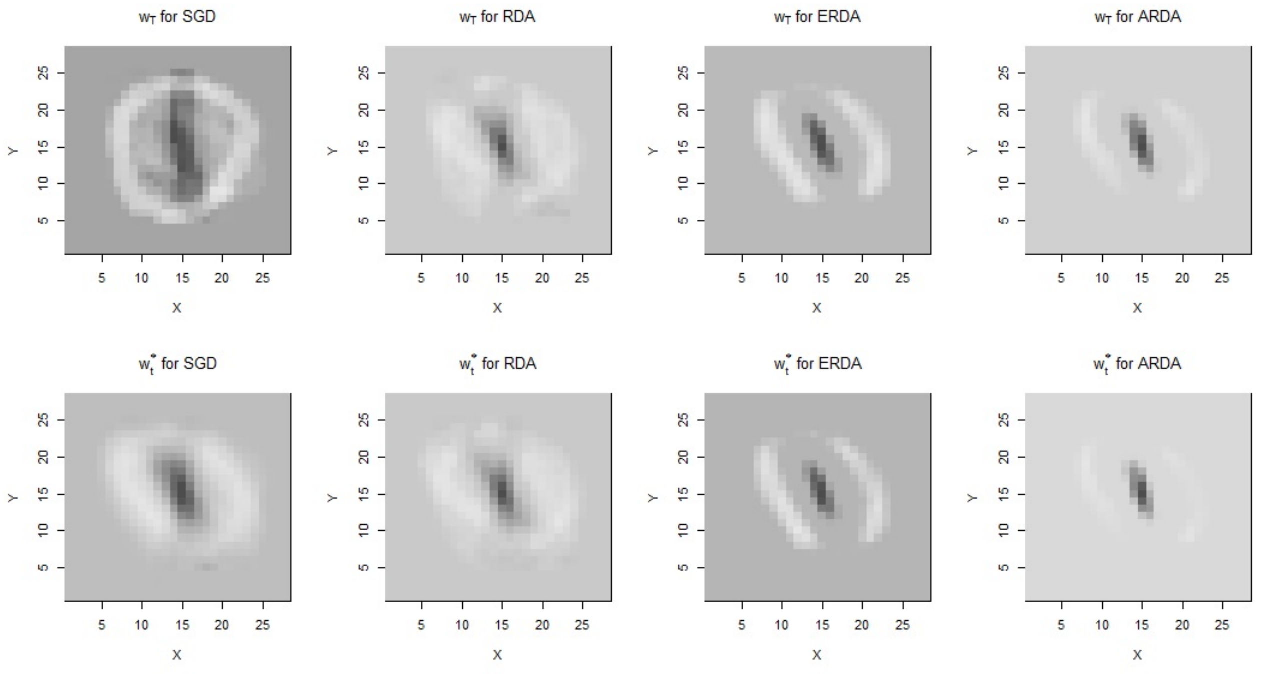

Figure 3: Sparsity patterns for $T = 200, \gamma = 500, \lambda = 0.01$ and $\rho = 0.005$. On first row are plotted parameter of last iteration w_T, expanded into a 28×28 size matrix from the 784-dimensional flattened vector. On the second row, we plot the average of parameters w_t over T iterations. Each column identifies an algorithm. White pixels refer to components that tend to 1, and black pixels those which tend to -1.

As h is not powerful enough to detect sparse pattern during the first iterations, optimized solution stays noisy. However, both for ERDA and ARDA, this sparse pattern is already set, as we can easily distinguish pattern for digits 0 and 1. Note that ARDA is able to directly find a more sparse solution than ERDA. This scheme is similar when $\lambda = 10^{-2}$, but different as soon as $\lambda = 10^{-1}$. RDA is able to find a sparse solution, similar to ARDA in the $\lambda = 10^{-3}$ case. But the result is unsatisfying for ERDA and ARDA : solution is too sparse, meaning that patterns are not clearly shown, which could lower accuracy of the model. With this parameter value, RDA methods is preferred over ERDA and ARDA to select a sparse solution. But elapsed time is another factor that we might control : in some cases, it is preferable to select a less sparse solution in order to gain computation time.

4 Conclusion

These dual averaging methods provide comforting results with interesting theoretical guarantees. Note that there are still improvements that we might add to the current setting of regularized dual averaging and to experiments.

- Extends current experiment to all 10 digits, as well as studying other models rather than linear SVM, but also other datasets that contains sparse data. This would help to empirically prove that RDA is efficient not only on specific dataset, but on any dataset where it would perform at least better than SGD for instance.
- Using other regularization functions and potentially discover closed-form solutions, as in this summary, we only considered l_1-regularization for Ψ and mixed l_1/l_2^2-regularization for h. This allowed us to find a closed-form solution for minimization step, but we could potentially provide a more general framework as we can still compute solutions for a wider range of regularization functions.

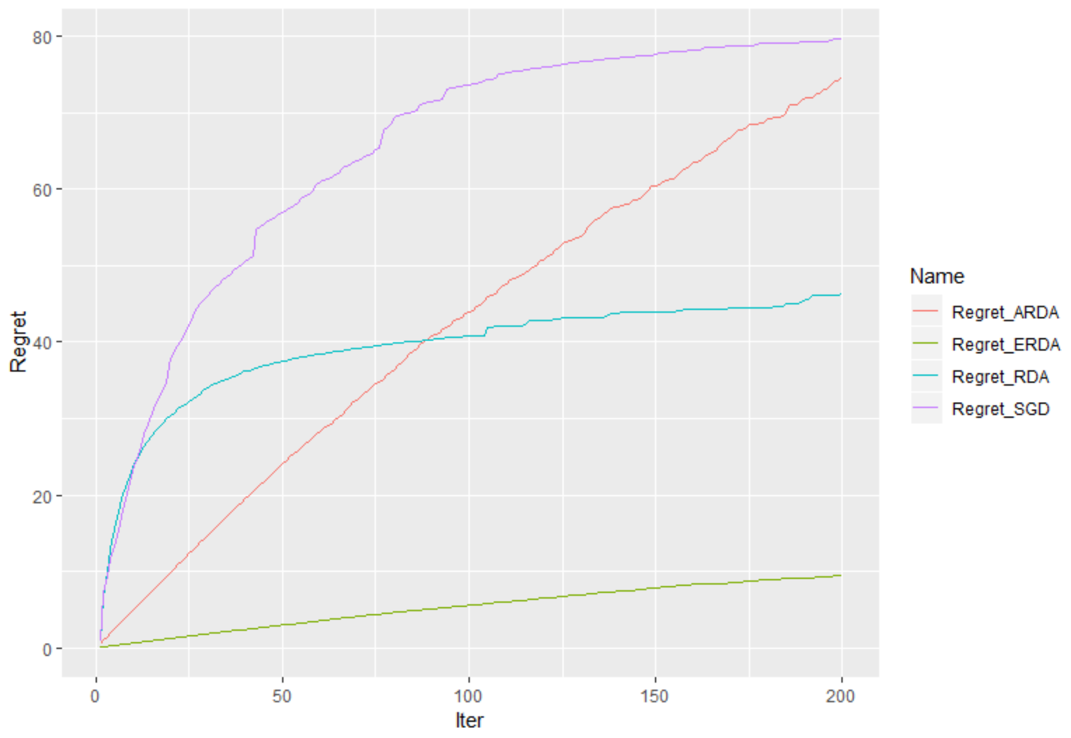

Figure 4: Regret curves for $T = 200, \gamma = 500, \lambda = 0.01$ and $\rho = 0.005$. As detailled, we can see for both SGD, RDA and ERDA a regret bound $O(\sqrt{t})$.

References

[1] Lin Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *J. Mach. Learn. Res.*, 11:2543–2596, December 2010.

[2] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

[3] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.

[4] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net (vol b 67, pg 301, 2005). *Journal of the Royal Statistical Society Series B*, 67:768–768, 02 2005.

[5] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

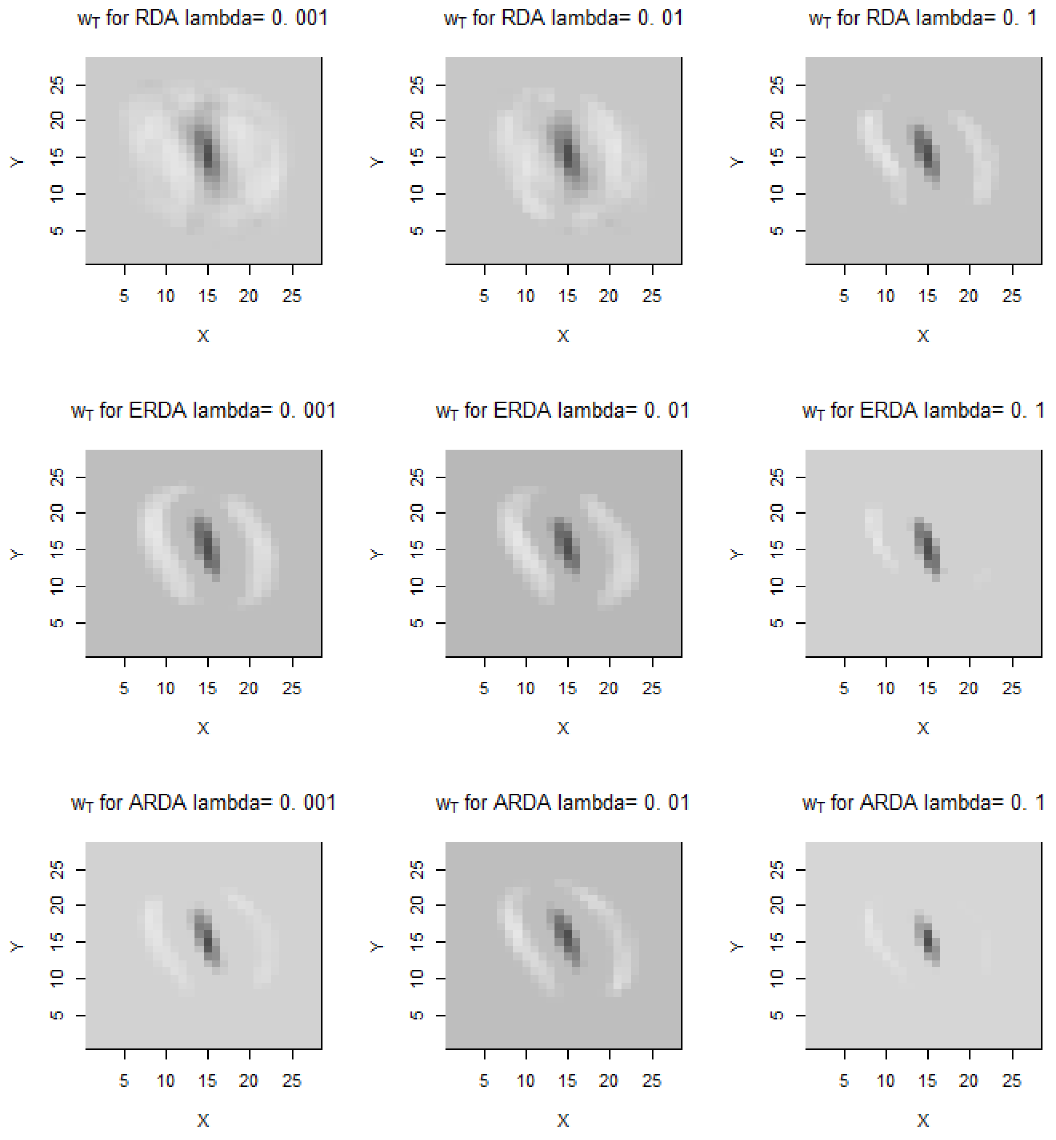

Figure 5: Sparsity patterns for $T = 200, \gamma = 500, \rho = 0.005$ and $\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}\}$.