Marcello Bullo
Francesco Mandruzzato
MSc. students
University of Padova
ID: 1204533- 1204532
marcello.bullo@studenti.unipd.it
francesco.mandruzzato.1@studenti.unipd.it

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

FINAL PROJECT:
Accelegrad and UnixGrad: a Comparison Analysis

Optimization, Francesco Rinaldi
19th February 2021

Introduction

Accelegrad

Previous related work

The Accelerated Gradient Descent method devised by Nesterov is a popular optimization technique which provides faster convergence rate $\mathcal{O}(1/T^2)$ in the smooth setting, guaranteeing standard rate $\mathcal{O}(1/\sqrt{T})$ for a general convex objective function f. The structure of the algorithm is divided in two different steps:

1. **Extrapolation step**: We move along the direction of the difference between the last two iterates:

$$y_k = x_k + \beta_k(x_k - x_{k-1}), \tag{1}$$

were β_k is chosen depending on the properties of f. Ideally this parameter regulates the amount of information that we want to keep regarding past iterates.

2. **Gradient step** We performe a step likewise the GD algorithm in y_k to get the next point x_{k+1}:

$$x_{k+1} = y_k - \alpha_k \nabla f(y_k), \tag{2}$$

where $\alpha_k = 1/L$

Despite its efficiency, Accelerated Gradient Descent is inappropriate for handling noisy feedback because of the choice of the fixed step size α_k, moreover acceleration requires the knowledge of the objective's smoothness.

One of the main popular adaptive first order methods comes with AdaGrad [1]. Informally, AdaGrad exploits the geometry of the dataset in the sense that the adaptation of the step size α_k vary with the dimension considered, in other words, it associates a small step size to frequently occurring features and a large step size to infrequent features. This adaptation is encoded in its update rule:

$$x_{t+1} = \Pi_K^{G_t^{1/2}}(x_t - \eta G_t^{-1/2} g_t), \tag{3}$$

where $G_t = \sum_{\tau=1}^t g_t g_t^T$ is the matrix of the previous subgradients, and Π_K is the projection of the gradient into the set K. With this adaption AdaGrad may handle noisy feedback, however there is no guarantee that AdaGrad can ensure acceleration, moreover it was unknown whether AdaGrad is able to

exploit the smoothness in order to converge faster. Considering the advantage of handling noisy feedback to ensure a solid acceleration, Accelegrad takes inspiration from the update rule of AdaGrad.

Regarding acceleration, an elegant way of interpreting it comes from [2] and we provide the high level concept which is useful to understand the algorithm steps.

In *Allen-Zhu* and *Orecchia*, they combine standard Gradient Descent and Mirror Descent to yield a new and simple accelerated gradient method for smooth convex optimization problems. Given an L-smooth function f we are able to find a quadratic upper bound of the form:

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|y - x\|^2. \quad \forall y$$

This is particularly useful to find an upper bound to the maximal objective decrease given as:

$$f(x_{k+1}) - f(x_k) \leq -\frac{1}{2L}\|\nabla f(x_k)\|^2.$$

From this point it should be clear that the larger the magnitude of the gradient the better may be the convergence speed of Gradient Descent. Mirror Descent instead peeks a uniformly distributed sequences of points $(x_1, \ldots, x_n)$ and it combines them to construct a stronger lower bound to the function f which is not provided by the standard Gradient Descent, in particular this lower bound is obtained as:

$$f(y) \geqslant \frac{1}{n}\sum_{t=1}^{n} f(x_t) + \frac{1}{n}\sum_{t=1}^{n}\langle \nabla f(x_t), y - x_t \rangle \quad \forall y.$$

To construct a good upper bound instead we consider the mean of the queried points and by the convexity property we get $f(\hat{x}) \leq \frac{1}{n}\sum_{t=1}^{n} f(t_i)$. Combining this with the previous result we get:

$$f(\hat{x}) - f(y) \leq \frac{1}{n}\sum_{t=1}^{n}\langle \nabla f(x_t), y - x_t \rangle,$$

which is known as the regret bound $R_n(y)$. Intuitively since we want to keep as small as possible the regret of an algorithm, contrary to gradient descent we need to receive small gradients, in fact it is possible to prove that Mirror Descent converge faster with small gradient magnitudes as it's converge rate is $\mathcal{O}(\rho^2/\epsilon^2)$ where ρ^2 is equal to the average magnitude of the queried gradients.

In light of this consideration, they propose a linear combination of the two algorithms at each step, in particular following Nesterov, given the gradient step y_k and the mirror step z_k the next step will be given by:

$$x_{k+1} = \alpha z_k + (1 - \alpha)y_k, \tag{4}$$

where α_k is a tuneable parameter. It is possible to prove that this technique provides the same convergence guarantees of the accelerated gradient method of Nesterov in a nicer and simpler way, for this reason Accelegrad incorporates the notion of acceleration taking inspiration from this linear coupling framework.

Accelegrad algorithm setting

In the Accelegrad paper [3], there is no assumption about the smoothness parameter β (See Def.0.1 of smoothness), but they assumed to be given a bound between some initial point x_0 and a global minimum of f for a given compact convex set K which contains this global minimizer. This distance is bounded by the diameter of K i.e. $D = \max_{x,y \in K}\|x - y\|$. Finally they assume that the function f is G-Lipschitz for which we provide the definition in (Def.0.2).

First, we discuss the offline setting where we have always access to the exact gradients of f, the algorithm is presented in Algo.1. As we previously said, Accelegrad takes inspiration from [2] and linearly couples between two sequences $\{y_t\}_t$ and $\{z_t\}_t$ into $\{x_{t+1}\}_t$. These two sequences are updated in two different manners, the former (y_{t+1}), takes a step starting from x_{t+1}, the latter (z_{t+1}), takes a step starting from

z_t. Both sequences are updated with the same step size but for z_{t+1} the gradient is scaled by a factor of α_t and the final result is projected into K. The step size is defined taking inspiration from AdaGrad:

$$\eta_t = \frac{2D}{(G^2 + \sum_{\tau=0}^{t} \alpha_\tau^2 \|g_t\|^2)^2}, \tag{5}$$

where the only difference are the importances weights α_t defined as:

$$\alpha_t = \begin{cases} 1 & 0 \leq t \leq 2 \\ \frac{1}{4}(t+1) & t \geqslant 3 \end{cases}, \tag{6}$$

which increase with t, putting emphasis on recent queries of the gradient. As we are considering the smooth setting it is not required the function to be G-Lipschitz which usually becomes important in the non smooth setting.

Algorithm 1: Accelerated Adaptive Gradient Method (AcceleGrad)

Input : #Iterations T, $x_0 \in K$, diameter D, weights $\alpha_{t \in T}$, learning rate η_t

Set $y_0 = x_0 = z_0$

for $t = 1 \ldots T$ **do**

    Set $\tau_t = 1/\alpha_t$

    Update:

$$x_{t+1} = \tau_t z_t + (1 - \tau_t)y_t, \quad g_t = \nabla f(x_{t+1})$$

$$z_{t+1} = \Pi_K(z_t - \alpha_t \eta_t g_t)$$

$$y_{t+1} = x_{t+1} - \eta_t g_t$$

Output: $\hat{y}_T \propto \sum_{t=0}^{T-1} \alpha_t y_{t+1}$

For the smooth setting Accelegrad requires a number of iterations to get a solution within ϵ of $\mathcal{O}(\frac{c}{\epsilon})^{\frac{1}{2}}$ with $c = DG + \beta D^2 log(\beta D/G)$, which means that the algorithm is able to converge as fast as the Accelerated gradient method asymptotically. Actually, since we are dealing with smooth functions we don't need to consider G, which means that the constant c is actually equal to $c = \beta D log(\beta D/\|g_0\|)$.

Considering the non smooth setting the previous algorithm is able to ensure standard sublinear convergence rate of $\mathcal{O}(1/\sqrt{T})$. With this rate of convergence Accelegrad requires a number of iterations to get a solution within ϵ of..., this result is also true for the stochastic optimization setting. In the Appendix we provide a short version of the proofs regarding the convergence rate for these three cases.

UnixGrad

Previous related works

UnixGrad algorithm setting

Comparison Results

In this section we provide a comparison between AcceleGrad and UnixGrad. We test the two algorithm with some SVM problems taken from LIBSVM. In particular we worked with *breast-cancer* and *adult* datasets. We try to minimize the L2-regularized squared hinge loss using random mini-batches of size 5, with random weights initialization. For both UnixGrad and Accelegrad we measure the performance with respect to the average iterate which results in a more stable behaviour.

Online to Batch conversion

Online learning is the process of answering a sequence of questions given knowledge of the correct answers to the previous ones. Given a sequence of questions $S = (x_1, \ldots, x_t)$, for each one of them sequentially our algorithm makes a prediction and then after the prediction the correct answer is revealed incurring in a loss. The goal of the learner is to minimize the loss also exploiting past information. If there is no correlation between samples, learning is hopeless.

Intuitively, for each sample passed to the learner at stage t, the learner chose an hypothesis h from the hypothesis class H and predicts the label associated with the sample. For this reason we can define a regret function which measure "how sorry" the learner is not to have followed the best predictor h^*. Formally when running on a sequence of T examples:

$$R_T(h^*) = \sum_{t=1}^{T} l(p_t, y_t) - \sum_{t=1}^{T} l(h^*(x_t), y_t),$$

where p_t is the prediction obtained by h at stage t, and the regret w.r.t the hypothesis class H is defined as:

$$R_T(H) := \max_{h^* \in H} R_T(h^*).$$

Usually we look for an algorithm which converge sub-linearly to the lowest possible regret.

What we described in short say that an online learning algorithm wants performance close to the single best hypothesis chosen in hindsight given all the data.

Different from this practice, an online to batch converter needs to construct a model based on the set of modules generated by the online learner, this can be done in different ways but one of the most classical way is to take the average of these models. This conversion is needed because the individual iterates of an online algorithm do not come with individual guarantees. The standard online to batch conversion algorithm format is shown in Algo.2. In particular since the algorithm outputs an average of the iterates, we can apply Jensen's inequality to show the following:

$$\mathbb{E}[L(\hat{w}) - L(x^*)] \leq \frac{\mathbb{E}[R_T(x^*)]}{T},$$

and as long as the algorithm obtains sublinear regret the left side part of the inequality will approach zero an average.

Algorithm 2: Online to batch conversion

Input : #Iterations T, Params S, Cost function l, Algorithm A

for $t = 1 \ldots T$ **do**
$\quad \lfloor$ let w_t be the prediction of A, provide the cost function $l(w_t, g_t)$ to A where $\mathbb{E}[g_t] = \nabla F(w_t)$

Output: $\hat{w} = \frac{1}{T} \sum_{t=1}^{T} w_t$

In [4] to avoid the guarantees problems of single iterates it is provided a black box "Anytime" Online to Batch conversion algorithm. The "Anytime property" comes from the fact that the last iterate is always a good estimate of x^*. The algorithm is very similar to the classical online to batch previously shown. The key difference is that they evaluate the stochastic gradient oracle at x_t rather than the iterates provided by the algorithm A, furthermore they incorporate the importance weights α_t which are useful to achieve faster convergence rates on smooth losses.

Algorithm 3: Anytime Online to batch conversion

Input : #Iterations T, Online Algorithm A with convex domain D, Non-negative weights α_t
$\qquad\qquad$ with $\alpha_t > 0$

for $t = 1 \ldots T$ **do**
$\quad$ Get $x_t = \frac{\sum_{i=1}^{t} \alpha_i w_i}{\sum_{i=1}^{t} \alpha_i}$
$\quad$ Play x_t and receive subgradient g_t
$\quad$ Send $l_t(x) = \langle \alpha_t g_t, x \rangle$ to a as the tth loss
$\quad$ Get w_{t+1} from A

Output: x_T

Different from the standard online to batch conversion, this algorithm is able to achieve the following guarantees for every $x^* \in D$:

$$\mathbb{E}[L(x_T) - L(x^*)] \leq \frac{\mathbb{E}[R_T(x^*)]}{\sum_{t=1}^{T} \alpha_t},$$

in particular this bound is valid also for loss functions with some known non-linearity. In the Appendix we provide a proof sketch of this bound.

General definitions

Definition 0.1. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is β-smooth if:*

$$f(y) \leq f(x) + \nabla f(x)^T (y - x) + \frac{\beta}{2} \|x - y\|^2, \forall x, y \in \mathbb{R}^n$$

Definition 0.2. *f is G-Lipschitz if $\forall w \in \Omega$ we have $\|\nabla f_i(w)\| \leq G$*

Definition 0.3. *Given a real-valued convex function f with elements $x_1, \ldots x_n$ in its domain and positive weights a_i, Jensen's inequality is stated as:*

$$f\left(\frac{\sum a_i x_i}{\sum a_i} \right) \leq \frac{\sum a_i f(x_i)}{\sum a_i}$$

References

[1] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 07 2011.

[2] Z. A. Zhu and L. Orecchia, "A novel, simple interpretation of nesterov's accelerated method as a combination of gradient and mirror descent," *CoRR*, vol. abs/1407.1537, 2014.

[3] K. Y. Levy, A. Yurtsever, and V. Cevher, "Online adaptive methods, universality and acceleration," 2018.

[4] A. Cutkosky, "Anytime online-to-batch conversions, optimism, and acceleration," 2019.

Appendices

Accelegrad

Theorem 1. *Assume that f is convex and β-smooth. Let K be a convex set with bounded diameter D, and assume there exists a global minimizer for f in K. Then Algorithm 1 with weights equal to Equation 6 and learning rate as in Equation 5 ensures:*

$$f(\hat{y}_t) - f(x^*) \leq \mathcal{O}\left(\frac{DG + \beta D^2 log(\beta D/G)}{T^2}\right)$$

Since the proof of this Theorem is very long, we provide the most meaningful parts which guides us into deriving the upper bound on the error.

Proof. Our objective is to find a bound of the left-side in order to precisely state the convergence rate of Accelegrad in the smooth case. Given the fact that the algorithm outputs a weighted average of gradients we may apply the Jensen Inequality defined in 0.3 as:

$$f(\hat{y}_t) - f(x^*) \leq \frac{1}{\sum_{t=0}^{T-1} a_i} \sum_{t=0}^{T-1} a_i(f(y_{t+1}) - f(x^*))$$

Given the fact that $\sum_{t=0}^{T-1} a_i \geq \Omega(T^2)$ it is sufficient to bound the numerator by a constant in order to get the sublinear converge rate $\mathcal{O}(C/T^2)$ and the proof is concentrated precisely at this point.

Without re-writing the proof of Lemma described in the paper, we can bound our term of interest by:

$$\sum_{t=0}^{T-1} a_i(f(y_{t+1}) - f(x^*)) \leq (\alpha_t^2 - \alpha_t)(f(y_t) - f(y_{t+1})$$

$$+\frac{\alpha_t^2}{2}\left(\beta_t - \frac{1}{\eta_t}\right)\|y_{t+1} - x_{t+1}\|^2$$

$$+\frac{1}{2\eta_t}(\|z_t - z\|^2 - \|z_{t+1} - z\|^2).$$

Each term of the sum is in turn bounded, providing a more elegant overall bound:

$$\frac{1}{2}\sum_{t=0}^{T-1} a_i(f(y_{t+1}) - f(x^*)) \leq \underbrace{\frac{D^2}{2\eta_{T-1}} - \frac{1}{4}\sum_{t=\tau_*+1}^{T-1} \eta_t\alpha_t^2\|g_t\|^2}_{(*)} + \underbrace{\frac{\beta}{2}\sum_{t=0}^{\tau_*} \eta_t^2\alpha_t^2\|g_t\|^2}_{(**)}.$$

Now by fixing the learning rate equal to the value in Equation.5, we provide a final bound which is constant and we conclude the proof. These steps requires many computations and Lemmas that we skip in order to provide the higher point of view.

The final result is of the form:

$$\frac{1}{2}\sum_{t=0}^{T-1} a_i(f(y_{t+1}) - f(x^*)) \leq DG/4 + 2\beta D^2 + 2\beta D^2(1 + 2log(4\beta D/D)$$

Combining this result with our initial inequality we get:

$$f(\hat{y}_t) - f(x^*) \leq \frac{1}{\sum_{t=0}^{T-1} a_i} \sum_{t=0}^{T-1} a_i(f(y_{t+1}) - f(x^*))$$

$$\leq \frac{DG/2 + 8\beta D^2(1 + log(4\beta D/D)}{T^2/32}$$

$$\approx \mathcal{O}\left(\frac{DG + \beta D^2 log(4\beta D/D)}{T^2}\right),$$

where we used the fact that $\sum_{t=0}^{T-1} a_i \geq T^2/32$, which conclude the proof for the smooth case. $\qquad \square$

Theorem 2. *Assume that f is convex and G-Lipschitz. Let K be a convex set with bounded diameter D, and assume there exists a global minimizer for f in K. Then Algorithm 1 with weights equal to Equation 6 and learning rate as in Equation 5 ensures:*

$$f(\hat{y}_t) - f(x^*) \leq \mathcal{O}\left(GD\sqrt{logT}/\sqrt{T}\right)$$

Proof. The proof follows the same considerations of the previous one, this time we need to prove that $\sum_{t=0}^{T-1} a_i(f(y_{t+1}) - f(x^*))$ is bounded by $\mathcal{O}(T^{3/2})$. As in the previous case it is possible to prove that:

$$\alpha_t(f(y_{t+1}) - f(x^*)) \leq \eta_t\alpha_t^2\|g_t\|^2 + \eta_t\alpha_t^2\|g_t\| + \frac{1}{2\eta_t}(\|z_t^2 - z\| - \|z_{t+1}^2 - z\|)$$

$$+(\alpha_t^2 - \alpha_t)(f(y_t) - f(y_{t+1})).$$

As before, we can bound each term of this sum and the combine the results to conclude our proof. $\qquad\square$

Theorem 3. *Assume that f is convex and G-Lipschitz. Let K be a convex set with bounded diameter D, and assume there exists a global minimizer for f in K. Assume that we invole Algorithm 1 with weights equal to Equation 6 and learning rate as in Equation 5 ensures, but this time we provide it with noisy gradient estimates, then our algorithm ensures:*

$$\mathbb{E}[f(\hat{y}_t)] - f(x^*) \leq \mathcal{O}\left(GD\sqrt{logT}/\sqrt{T}\right)$$

Proof. In order to provide the proof of this theorem we assume that upon querying a first order oracle with a point x, we receive a bounded and unbiased gradient estimate $\tilde{g}$ such that $\mathbb{E}[\tilde{g}|x] = \nabla f(x)$ with $\|\tilde{g}\| \leq G$. As in the previous theorems we need to find an upper bound for $\mathbb{E}[f(\hat{y}_t)] - f(x^*)$ to obtain that convergence rate. Obviously this bound is a bit different from the previous ones but the concept is the same.

$\qquad\square$

Theorem 4. *Suppose $g_1, \ldots, g_t$ satisfy $\mathbb{E}[g_t|x_t] \in \delta L(x_t)$ for some objective function L and g_t is independent of all other quantities given x_t. Let $R_T(x^*)$ be a bound on the linearized regret of the Anytime Online to Batch algorithm.*

$$R_T(x^*) \geqslant \sum_{t=1}^{T} \langle \alpha_t g_t, w_t - x^* \rangle$$

Then for all $x^* \in D$, the algorithm guarantees:

$$\mathbb{E}[L(x_t) - L(x^*)] \leq \mathbb{E}\left[\frac{R_T(x^*)}{\sum_{t=1}^{T}\alpha_t}\right]$$

Furthermore, suppose that D has a diameter B and $\|g_t\|_* \leq G$ with probability 1 for some G. Then with probability at least $1 - \delta$:

$$L(x_T) - L(x^*) \leq \frac{R_T(x^*) + 2BG\sqrt{\sum_{t=1}^{T}\alpha_t^2 log(2/\delta)}}{\alpha_{1:T}}$$

Proof. In order to prove this theorem we observe that $\alpha_t(x_t - w_t) = \alpha_{1:t-1}(x_{t-1} - x_t)$, where $\alpha_{1:t-1} = \sum_{i=1}^{t-1}\alpha_i$. First we use the property of convexity to bound $\mathbb{E}[\alpha_t(L(x_t) - L(x^*))]$ with $\mathbb{E}[\sum_{t=1}^{T}\alpha_t\langle g_t, x_t - x^*\rangle]$, then using the previous observation in conjunction with some simple math adjustments we obtain:

$$\mathbb{E}[\alpha_t(L(x_t) - L(x^*))] \leq \mathbb{E}\left[\sum_{t=1}^{T}\alpha_{1:t-1}(L(x_{t-1}) - L(x_t))\right],$$

8

To conclude the proof of this theorem is sufficient to subtract $\mathbb{E}[\sum_{t=1}^{T} \alpha_t L(x_t)]$ from both sides and then apply the telescope series to obtain:

$$\mathbb{E}[\alpha_{1:T} L(x_T) - \alpha_{1:T} L(x^*)] \leq \mathbb{E}[R_T(x^*)]$$

$$\mathbb{E}[L(x_T) - L(x^*)] \leq \mathbb{E}\left[\frac{R_T(x^*)}{\alpha_{1:T}}\right].$$

Furthermore, let's define H_{t-1} as the history $g_{t-1}, x_{t-1}, \ldots, g_1, x_1$ and define $G_t = \mathbb{E}[g_t | H_{t-1}, x_t, w_t]$ which is the expected value of the subgradient in relation to the past history. Let's define $\epsilon_t = \alpha_t \langle G_t, w_t - x^* \rangle - \langle g_t, w_t - x^* \rangle$ since g_t is independent of all other quantities given x_t, we have that $\mathbb{E}[\epsilon_t | H_{t-1}, x_t, w_t] = 0$. If we sum all over the t's we obtain:

$$\sum_{t=1}^{T} \epsilon_t = \sum_{t=1}^{T} \alpha_t \langle \underbrace{G_t}_{(\leq G)}, \underbrace{w_t - x^*}_{(\leq B)} \rangle - \sum_{t=1}^{T} \alpha_t \langle \underbrace{g_t}_{(\leq G)}, \underbrace{w_t - x^*}_{(\leq B)} \rangle,$$

where $|\epsilon_t| \leq 2\alpha_t BG$ with probability 1, and using the Azuma-Hoeffding bound with probability at least $1 - \delta$ we get $\sum_{t=1}^{T} \epsilon_t \leq 2BG\sqrt{\sum_{t=1}^{T} \alpha_t^2 log(2/\delta)}$. This bound is useful to conclude our proof, the steps are similar to the previous part of the proof and with some simple math adjustments we obtain the final result of the theorem. $\qquad\square$

Unixgrad

Regret-to-rate conversion

In order to provide the mathematical analysis of the convergence rate of UnixGrad in the different optimization scenarios, classical tools in the online learning literature are taken into account an the convergence analysis is performed in the sense of bounding "weighted regret". Hence, a conversion strategy is adopted to directly translate the weighted regret to convergence rate.

Lemma 5 (Regret-to-rate conversion). *Consider weighted average*

$$\bar{x}_t = \frac{\alpha_t x_t + \sum_{i=1}^{t-1} \alpha_i x_i}{\sum_{i=1}^{t} \alpha_i}.$$

Let $R_T(x_*) = \sum_{t=1}^{T} \alpha_t \langle x_t - x_*, g_t \rangle$ denote the weighted regret after T iterations, $\alpha_t = t$ and $g_t = \nabla f(\bar{x}_t)$. Then,

$$f(\bar{x}_T) - f(x_*) \leq \frac{2R_T(x_*)}{T^2}$$

Proof. Proof. Let's define $A_t = \sum_{i=1}^{t} \alpha_i$. Then, by definition, we could express x_t as

$$x_t = \frac{A_t}{\alpha_t} \bar{x}_t - \frac{A_{t-1}}{\alpha_t} \bar{x}_{t-1} \tag{7}$$

Then, use Eq. (7) and replace g_t by $\nabla f(\bar{x}_t)$ in the weighted regret expression, i.e.

$$
\begin{aligned}
\sum_{t=1}^{T} \alpha_t \left\langle x_t - x_*, \nabla f(\bar{x}_t) \right\rangle &= \sum_{t=1}^{T} \alpha_t \left\langle \frac{A_t}{\alpha_t} \bar{x}_t - \frac{A_{t-1}}{\alpha_t} \bar{x}_{t-1} - x_*, \nabla f(\bar{x}_t) \right\rangle \\
&= \sum_{t=1}^{T} \alpha_t \left\langle \frac{A_t}{\alpha_t}(\bar{x}_t - x_*) - \frac{A_{t-1}}{\alpha_t}(\bar{x}_{t-1} - x_*), \nabla f(\bar{x}_t) \right\rangle \quad \left(-x_* = \frac{A_{t-1}}{\alpha_t} x_* - \frac{A_t}{\alpha_t} x_* \right) \\
&= \sum_{t=1}^{T} A_t \left\langle \bar{x}_t - x_*, \nabla f(\bar{x}_t) \right\rangle - A_{t-1} \left\langle \bar{x}_{t-1} - x_*, \nabla f(\bar{x}_t) \right\rangle \\
&= \sum_{t=1}^{T} (\alpha_t + A_{t-1}) \left\langle \bar{x}_t - x_*, \nabla f(\bar{x}_t) \right\rangle - A_{t-1} \left\langle \bar{x}_{t-1} - x_*, \nabla f(\bar{x}_t) \right\rangle \\
&= \sum_{t=1}^{T} \alpha_t \left\langle \bar{x}_t - x_*, \nabla f(\bar{x}_t) \right\rangle + A_{t-1} \left\langle \bar{x}_t - \bar{x}_{t-1}, \nabla f(\bar{x}_t) \right\rangle \\
&\geq \sum_{t=1}^{T} \alpha_t (f(\bar{x}_t) - f(x_*)) + \sum_{t=1}^{T} \sum_{i=1}^{t-1} \alpha_i (f(\bar{x}_t) - f(\bar{x}_{t-1})) \quad \left(f(x) - f(y) \leq \nabla f(x)^\top (x - y) \right)
\end{aligned}
$$

where in the last line we use $f(y) \geq f(x) + \nabla f(x)^\top (y - x)$ and multiply by (-1). We also take $\alpha_0 = 0$ and $A_0 = 0$. Then, we telescope the double summation and reorganize the terms

$$
\begin{aligned}
&= \sum_{t=1}^{T} \alpha_t (f(\bar{x}_t) - f(x_*)) + \sum_{t=1}^{T-1} \alpha_t (f(\bar{x}_T) - f(\bar{x}_t)) \\
&= \alpha_T (f(\bar{x}_T) - f(x_*)) + \sum_{t=1}^{T-1} \alpha_t (f(\bar{x}_t) - f(x_*) + f(\bar{x}_T) - f(\bar{x}_t)) \\
&= \sum_{t=1}^{T} \alpha_t (f(\bar{x}_T) - f(x_*))
\end{aligned}
$$

Having simplified the expression, we divide both sides by A_T and conclude the proof. Observe that $A_T \geq \frac{T^2}{2}$, hence,

$$
\begin{aligned}
\sum_{t=1}^{T} \alpha_t (f(\bar{x}_T) - f(x_*)) &\leq \sum_{t=1}^{T} \alpha_t \left\langle x_t - x_*, \nabla f(\bar{x}_t) \right\rangle \\
\frac{1}{A_T} \sum_{t=1}^{T} \alpha_t (f(\bar{x}_T) - f(x_*)) &\leq \frac{1}{A_T} \sum_{t=1}^{T} \alpha_t \left\langle x_t - x_*, \nabla f(\bar{x}_t) \right\rangle \\
f(\bar{x}_T) - f(x_*) &\leq \frac{2R_T(x_*)}{T^2}
\end{aligned}
$$

$\square$

Non-smooth Setting

Before starting with the mathematical analysis of the convergence rate of UnixGrad in the non-smooth setting, we recall some useful lemmas.

Lemma 6. *Let $\{a_i\}_{i=1,\dots,n}$ be a sequence of non negative numbers. Then, it holds that*

$$
\sqrt{\sum_{i=1}^{n} a_i} \leq \sum_{i=1}^{n} \frac{a_i}{\sum_{j=1}^{i} a_j} \leq 2 \sqrt{\sum_{i=1}^{n} a_i}
$$

Proof. Please refer to [3] for the proof. $\square$

Lemma 7.
$$\alpha_t \left\| g_t - M_t \right\|_* \left\| x_t - y_t \right\| = \inf_{\rho > 0} \left\{ \frac{\rho}{2} \left\| g_t - M_t \right\|_*^2 + \frac{\alpha_t^2}{2\rho} \left\| x_t - y_t \right\|^2 \right\} \tag{8}$$

Proof. Given two nonnegative real numbers $a \geq 0, b \geq 0$ and two real number $p > 0$ and $q > 0$ such that $1/p + 1/q = 1$, then

$$ab \leq \frac{a^p}{p} + \frac{b^q}{q}. \qquad \text{(Young's inequality)}$$

By using it with $p, q = 2$ we have that

$$\alpha_t \left\| g_t - M_t \right\|_* \left\| x_t - y_t \right\| \leq \frac{1}{2} \left\| g_t - M_t \right\|_*^2 + \frac{\alpha_t^2}{2} \left\| x_t - y_t \right\|^2.$$

Now, since the equality holds when $a^p = b^q$ we can use mathematical trick and formulate the statement as

$$\alpha_t \left\| g_t - M_t \right\|_* \left\| x_t - y_t \right\| = \inf_{\rho > 0} \left\{ \frac{\rho}{2} \left\| g_t - M_t \right\|_*^2 + \frac{\alpha_t^2}{2\rho} \left\| x_t - y_t \right\|^2 \right\}.$$

$\square$

Deterministic setting

Theorem 8. *Consider the constrained optimization setting, where $f : \mathcal{K} \to \mathbb{R}$ is a proper, convex and G-Lipschitz function defined over compact, convex set $\mathcal{K}$. Let $x^* \in \min_{x \in \mathcal{K}} f(x)$. Then, UnixGrad algorithm guarantees*

$$f\left(\bar{x}_T \right) - \min_{x \in \mathcal{K}} f(x) \leq \frac{7D\sqrt{1 + \sum_{t=1}^T \alpha_t^2 \left\| g_t - M_t \right\|_*^2} - D}{T^2} \leq \frac{6D}{T^2} + \frac{14GD}{\sqrt{T}}$$

Proof.

$$\sum_{t=1}^T \alpha_t \left\langle x_t - x_*, g_t \right\rangle = \sum_{t=1}^T \underbrace{\alpha_t \left\langle x_t - y_t, g_t - M_t \right\rangle}_{(A)} + \underbrace{\alpha_t \left\langle x_t - y_t, M_t \right\rangle}_{(B)} + \underbrace{\alpha_t \left\langle y_t - x^*, g_t \right\rangle}_{(C)}$$

Bounding (A)

$$\sum_{t=1}^T \alpha_t \left\langle x_t - y_t, g_t - M_t \right\rangle \leq \sum_{t=1}^T \alpha_t \left\| g_t - M_t \right\|_* \left\| x_t - y_t \right\| \qquad \text{(Hölder's Inequality)}$$

$$\leq \sum_{t=1}^T \frac{\rho}{2} \left\| g_t - M_t \right\|_*^2 + \frac{\alpha_t^2}{2\rho} \left\| x_t - y_t \right\|^2 \qquad \text{(Equation (8))}$$

By setting $\rho = \alpha_t^2 \eta_{t+1}$ $(\alpha_t, \eta_{t+1} > 0)$, we get the following upper bound for term (A),

$$\sum_{t=1}^T \alpha_t \left\langle x_t - y_t, g_t - M_t \right\rangle \leq \sum_{t=1}^T \frac{\alpha_t^2 \eta_{t+1}}{2} \left\| g_t - M_t \right\|_*^2 + \frac{1}{2\eta_{t+1}} \left\| x_t - y_t \right\|^2$$

Bounding (B)
In order to find an upper-bound for this term, we have to exploit the first order optimality condition of x_t, that is $\nabla f(x_t)^\top (y - x_t) \geq 0$, $\forall y$ feasible (all feasible directions from x are aligned with increasing gradient). Since $\nabla f(x_t) = \alpha_t M_t + \frac{1}{\eta_t} \nabla_x \mathcal{D}_\mathcal{R}(x_t, y_{t-1})$, then

$$\alpha_t \left\langle y_t - x_t, M_t \right\rangle + \frac{1}{\eta_t} \nabla_x \mathcal{D}_\mathcal{R}(x_t, y_{t-1})^\top (y_t - x_t) \geq 0.$$

By simply rearranging the above inequality we get

$$\frac{1}{\eta_t} \nabla_x \mathcal{D}_\mathcal{R}(x_t, y_{t-1})^\top (y_t - x_t) \geq \alpha_t \left\langle x_t - y_t, M_t \right\rangle.$$

Now the upper-bound becomes straightforward.

$$\sum_{t=1}^{T} \alpha_t \left\langle x_t - y_t, M_t \right\rangle \leq \sum_{t=1}^{T} \frac{1}{\eta_t} \nabla_x D_{\mathcal{R}} \left(x_t, y_{t-1} \right)^T \left(y_t - x_t \right) \quad \text{(Optimality for } x_t)$$

$$= \sum_{t=1}^{T} \frac{1}{\eta_t} \left(D_{\mathcal{R}} \left(y_t, y_{t-1} \right) - D_{\mathcal{R}} \left(x_t, y_{t-1} \right) - D_{\mathcal{R}} \left(y_t, x_t \right) \right)$$

Bounding (C)

Following the same procedure used in the previous bound we find the upper-bound also for the third term.

$$\sum_{t=1}^{T} \alpha_t \left\langle y_t - x^*, g_t \right\rangle \leq \sum_{t=1}^{T} \frac{1}{\eta_t} \nabla_x D_{\mathcal{R}} \left(y_t, y_{t-1} \right)^T \left(x^* - y_t \right) \quad \text{(Optimality for } y_t)$$

$$= \sum_{t=1}^{T} \frac{1}{\eta_t} \left(D_{\mathcal{R}} \left(x^*, y_{t-1} \right) - D_{\mathcal{R}} \left(y_t, y_{t-1} \right) - D_{\mathcal{R}} \left(x^*, y_t \right) \right)$$

Final Bound

$$\sum_{t=1}^{T} \alpha_t \left\langle x_t - x_*, g_t \right\rangle \leq \sum_{t=1}^{T} \frac{\alpha_t^2 \eta_{t+1}}{2} \left\| g_t - M_t \right\|_*^2 + \frac{1}{2\eta_{t+1}} \left\| x_t - y_t \right\|^2$$

$$+ \frac{1}{\eta_t} \left(D_{\mathcal{R}} \left(x^*, y_{t-1} \right) - D_{\mathcal{R}} \left(x^*, y_t \right) - D_{\mathcal{R}} \left(x_t, y_{t-1} \right) - D_{\mathcal{R}} \left(y_t, x_t \right) \right)$$

$$\leq \sum_{t=1}^{T} \frac{\alpha_t^2 \eta_{t+1}}{2} \left\| g_t - M_t \right\|_*^2 + \frac{1}{2\eta_{t+1}} \left\| x_t - y_t \right\|^2$$

$$+ \frac{1}{\eta_t} \left(D_{\mathcal{R}} \left(x^*, y_{t-1} \right) - D_{\mathcal{R}} \left(x^*, y_t \right) - \frac{1}{2} \left(\left\| x_t - y_t \right\|^2 + \left\| x_t - y_{t-1} \right\|^2 \right) \right)$$

$$\leq \sum_{t=1}^{T} \frac{\alpha_t^2 \eta_{t+1}}{2} \left\| g_t - M_t \right\|_*^2 + \sum_{t=1}^{T-1} \left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t} \right) D_{\mathcal{R}} \left(x^*, y_t \right)$$

$$+ \sum_{t=1}^{T} \left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t} \right) \left\| x_t - y_t \right\|^2 + \frac{1}{\eta_1} D^2$$

$$\leq \sum_{t=1}^{T} \frac{\alpha_t^2 \eta_{t+1}}{2} \left\| g_t - M_t \right\|_*^2 + \sum_{t=1}^{T} \left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t} \right) \left\| x_t - y_t \right\|^2 + \frac{D^2}{\eta_T} + \frac{D}{2}$$

$$\leq \sum_{t=1}^{T} \frac{\alpha_t^2 \eta_{t+1}}{2} \left\| g_t - M_t \right\|_*^2 + D^2 \left(\frac{2}{\eta_{T+1}} + \frac{1}{\eta_T} \right) + \frac{D}{2}$$

$$\leq D \sum_{t=1}^{T} \frac{\alpha_t^2 \left\| g_t - M_t \right\|_*^2}{\sqrt{1 + \sum_{i=1}^{t} \alpha_t^2 \left\| g_t - M_t \right\|_*^2}} + \frac{3}{2} D \sqrt{1 + \sum_{t=1}^{T} \alpha_t^2 \left\| g_t - M_t \right\|_*^2} + \frac{D}{2}$$

$$\leq \frac{7}{2} D \sqrt{1 + \sum_{t=1}^{T} \alpha_t^2 \left\| g_t - M_t \right\|_*^2} - \frac{D}{2}$$

$$\leq 3D + 7GD \sqrt{\sum_{t=1}^{T} \alpha_t^2}$$

$$\leq 3D + 7GD T^{3/2}$$

We obtain the rate by applying Lemma 1 to the weighted regret bound above $\square$