Learning-Rate-Free Learning by D-Adaptation

Aaron Defazio [1] **Konstantin Mishchenko** [2]

Abstract

D-Adaptation is an approach to automatically setting the learning rate which asymptotically achieves the optimal rate of convergence for minimizing convex Lipschitz functions, with no back-tracking or line searches, and no additional function value or gradient evaluations per step. Our approach is the first hyper-parameter free method for this class without additional multiplicative log factors in the convergence rate. We present extensive experiments for SGD and Adam variants of our method, where the method automatically matches hand-tuned learning rates across more than a dozen diverse machine learning problems, including large-scale vision and language problems. An open-source implementation is available[1].

1. Introduction

We consider the problem of unconstrained convex minimization,

$$\min_{x \in \mathbb{R}^p} f(x),$$

where f has Lipschitz constant G and a non-empty set of minimizers. The standard approach to solving it is the subgradient method that, starting at a point x_0, produces new iterates following the update rule:

$$x_{k+1} = x_k - \gamma_k g_k,$$

where $g_k \in \partial f(x_k)$ is a subgradient of f. After running for n steps, the average iterate $\hat{x}_n = \frac{1}{n+1}\sum_{k=0}^{n} x_k$ is returned. The *learning rate* γ_k, also known as the *step size*, is the main quantity controlling if and how fast the method converges.

[1] https://github.com/facebookresearch/dadaptation

[1] Meta AI, Fundamental AI Research (FAIR) team, New York [2] CNRS, ENS, INRIA SIERRA team, France. Correspondence to: Aaron Defazio <aaron.defazio@gmail.com>, Konstantin Mishchenko <konsta.mish@gmail.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

Algorithm 1 Dual Averaging with D-Adaptation

Input: x_0, $d_0 > 0$,
$s_0 = 0$, $g_0 \in \partial f(x_0)$, $\gamma_0 = 1/\left\|g_0\right\|$
If $g_0 = 0$, exit with $\hat{x}_n = x_0$
for $k = 0$ **to** n **do**
  $g_k \in \partial f(x_k)$
  $s_{k+1} = s_k + d_k g_k$
  $\gamma_{k+1} = \dfrac{1}{\sqrt{\sum_{i=0}^{k}\left\|g_i\right\|^2}}$
  $\hat{d}_{k+1} = \dfrac{\gamma_{k+1}\left\|s_{k+1}\right\|^2 - \sum_{i=0}^{k}\gamma_i d_i^2 \left\|g_i\right\|^2}{2\left\|s_{k+1}\right\|}$
  Option II: $\hat{d}_{k+1} = \dfrac{\sum_{i=0}^{k} d_i \gamma_i \left\langle g_i, s_i\right\rangle}{\left\|s_{k+1}\right\|}$
  $d_{k+1} = \max\left(d_k,\, \hat{d}_{k+1}\right)$
  $x_{k+1} = x_0 - \gamma_{k+1} s_{k+1}$
end for
Return $\hat{x}_n = \frac{1}{\sum_{k=0}^{n} d_k}\sum_{k=0}^{n} d_k x_k$

If the learning rate sequence is chosen too large, the method might oscillate around the solution, whereas small values lead to very slow progress. Setting γ_k optimally requires knowledge of the distance to a solution. In particular, denote x_* to be any minimizer of f, D to be the associated distance $D = \left\|x_0 - x_*\right\|$, and f_* to be the optimal value, $f_* = f(x_*)$. Then, using the fixed step size:

$$\gamma_k = \frac{D}{G\sqrt{n}},$$

the average iterate $\hat{x}_n$ converges in terms of function value at an inverse square-root rate:

$$f(\hat{x}_n) - f_* = \mathcal{O}(DG/\sqrt{n}).$$

This rate is worst-case optimal for this complexity class (Nesterov, 2018). Setting this step size requires knowledge of two problem constants, D and G. Adaptivity to G can be achieved using a number of approaches, the most practical of which is the use of AdaGrad-Norm step sizes (Streeter & McMahan, 2010; Duchi et al., 2011; Ward et al., 2019):

$$\gamma_k = \frac{D}{\sqrt{\sum_{i=0}^{k}\left\|g_i\right\|^2}},$$

together with projection onto the D-ball around the origin. AdaGrad-Norm step sizes still require knowledge of D, and they perform poorly when it is estimated wrong. In the (typical) case where we don't have knowledge of D, we can start with loose lower and upper bounds d_0 and $d_{\max}$, and perform a hyper-parameter grid search on a log-spaced scale. In most machine learning applications a grid search is the current standard practice.

In this work we take a different approach. We describe a method that achieves the optimal rate, for sufficiently large n, by maintaining and updating a lower bound on D (Algorithm 1). Using this lower bound is provably sufficient to achieve the optimal rate of convergence:

$$f(\hat{x}_n) - f(x_*) = \mathcal{O}\left(\frac{DG}{\sqrt{n+1}}\right),$$

with no additional log factors, avoiding the need for a hyper-parameter grid search.

Our method is highly effective across a broad range of practical problems, matching a carefully hand-tuned baseline learning rate across a broad range of machine learning problems within computer vision, Natural language processing and recommendation systems.

2. Algorithm

Our proposed approach is a simple modification of the AdaGrad step size applied to weighted dual averaging, together with our key innovation: D lower bounding. At each step, we construct a lower bound $\hat{d}_k$ on D using empirical quantities. If this bound is better (i.e. larger) than our current best bound d_k of D, we use $d_k = \hat{d}_k$ in subsequent steps. There are two options to estimate $\hat{d}_k$, but since they have exactly the same theoretical properties, we only discuss the first option below.

To construct the lower bound, we show that a weighted sum of the function values is bounded above as:

$$\sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) \\ \leq D \|s_{n+1}\| + \sum_{k=0}^{n} \frac{\gamma_k}{2} d_k^2 \|g_k\|^2 - \frac{\gamma_{n+1}}{2} \|s_{n+1}\|^2 .$$

There are two key differences from the classical bound (Orabona, 2019):

$$\sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) \leq \frac{1}{2}\gamma_{n+1}^{-1} D^2 + \sum_{k=0}^{n} \frac{\gamma_k}{2} d_k^2 \|g_k\|^2 .$$

Firstly, we are able to gain an additional negative term $-\frac{1}{2}\gamma_{n+1} \|s_{n+1}\|^2$. Secondly, we replace the typical D^2 error term with $D \|s_{n+1}\|$, following the idea of Carmon & Hinder (2022). This bound is tighter than the classical bound, and equivalent when $D = \|x_0 - x_{n+1}\|$, since:

$$D \|s_{n+1}\| - \frac{1}{2}\gamma_{n+1} \|s_{n+1}\|^2 \\ = \frac{1}{2}\gamma_{n+1}^{-1} \left(D^2 - \left(D - \|x_0 - x_{n+1}\|\right)^2\right) \leq \frac{1}{2}\gamma_{n+1}^{-1} D^2$$

From our bound, using the fact that

$$\sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) \geq 0,$$

we have:

$$0 \leq D \|s_{n+1}\| + \sum_{k=0}^{n} \frac{\gamma_k}{2} d_k^2 \|g_k\|^2 - \frac{\gamma_{n+1}}{2} \|s_{n+1}\|^2 ,$$

which can be rearranged to yield a lower bound on D, involving only known quantities:

$$D \geq \hat{d}_{n+1} = \frac{\gamma_{n+1} \|s_{n+1}\|^2 - \sum_{k=0}^{n} \gamma_k d_k^2 \|g_k\|^2}{2 \|s_{n+1}\|}.$$

This bound is potentially vacuous if $\|s_{n+1}\|^2$ is small in comparison to $\sum_{k=0}^{n} \gamma_k d_k^2 \|g_k\|^2$. This only occurs once the algorithm is making fast-enough progress that bound adjustment is not necessary at that time. The maximum over seen bounds can not be negative since our algorithm begins with a user-specified positive lower bound d_0, which sets the scale of the initial steps.

Theorem 2.1. *For a convex G-Lipschitz function f, Algorithm 1 returns a point $\hat{x}_n$ such that:*

$$f(\hat{x}_n) - f(x_*) = \mathcal{O}\left(\frac{DG}{\sqrt{n+1}}\right),$$

as $n \to \infty$, where $D = \|x_0 - x_*\|$ for any x_* in the set of minimizers of f, as long as $d_0 \leq D$.

The above result is asymptotic due to the existence of worst-case functions when n is fixed in advance. For any fixed choice of n, a function can be constructed such that Algorithm 1 run for n steps has a dependence on d_0. Despite this, we are able to show that the non-asymptotic convergence rate is only worse by a log factor.

Theorem 2.2. *Consider Algorithm 1 run for $n \geq 2\log_2(D/d_0)$ iterations with the step size modified to be*

$$\gamma_{k+1} = \frac{1}{\sqrt{G^2 + \sum_{i=0}^{k} \|g_i\|^2}}. \tag{1}$$

If we return the point $\hat{x}_t = \frac{1}{\sum_{k=0}^{t} d_k} \sum_{k=0}^{t} d_k x_k$ where t

is chosen to be $t = \arg\min_{k\le n} \frac{d_{k+1}}{\sum_{i=0}^{k} d_i}$*, then using the notation* $\log_{2+}(x) = \max(1, \log_2(x))$*:*

$$f(\hat{x}_t) - f_* \le 16\frac{\log_{2+}(d_{n+1}/d_0)}{n+1} D\sqrt{\sum_{k=0}^{t} \|g_k\|^2}$$

The modification to the step size can be avoided at the cost of having an extra term, namely we would have the following guarantee for the same iterate $\hat{x}_t$:

$$f(\hat{x}_t) - f_* \le \frac{16DG\log_{2+}(\frac{D}{d_0})}{\sqrt{n+1}} + \frac{8DG^2\log_{2+}(\frac{D}{d_0})}{(n+1)\|g_0\|}.$$

Notice that, unlike the bound in the theorem above, it also depends on the initial gradient norm $\|g_0\|$.

Our algorithm returns a weighted average iterate $\hat{x}_n$ rather than the last iterate x_{n+1}. This is standard practice when AdaGrad Norm schedules approaches are used, both for dual averaging and gradient descent. Techniques are known to obtain guarantees on the last-iterate either by the use of momentum (Defazio & Gower, 2021) or modified step-size sequences (Jain et al., 2019), although we have no explored if these approaches are compatible with D-Adaptation.

2.1. Why Dual Averaging?

The new bound we develop is actually general enough to apply to both gradient descent and dual averaging. Using the same proof techniques, D-Adaptation can also be applied on top of a gradient descent step. However, we do not use the gradient descent version above for a technical reason: the asymptotic convergence rate has an additional log factor. The practical performance of the two methods is very similar.

Theorem 2.3. *Gradient Descent with D-Adaptation (Algorithm 2), under the assumptions of Theorem 2.1, returns a point* $\hat{x}_n$ *such that:*

$$f(\hat{x}_n) - f = \mathcal{O}\left(\frac{DG}{\sqrt{n+2}}\log(n+2)\right).$$

This log factor arises whenever any-time step sizes are used on top of gradient descent when applied to unbounded domains, and is not specific to our method (Beck, 2014).

3. D-Adapted AdaGrad

The D-Adaptation technique can be applied on top of the coordinate-wise scaling variant of AdaGrad with appropriate modifications. Algorithm 3 presents this method. This variant estimates the distance to the solution in the ℓ_∞-norm instead of the Euclidean norm, $D_\infty = \|x_0 - x_*\|_\infty$. The theory for AdaGrad without D-Adaptation also uses the same norm to measure the distance to solution, so this modification is natural, and results in the same adaptive convergence rate as AdaGrad up to constant factors *without* requiring knowledge of D_∞.

Algorithm 2 Gradient Descent with D-Adaptation

Input: d_0, x_0
$s_0 = 0$
If $g_0 = 0$, exit with $\hat{x}_n = x_0$
for $k = 0$ **to** n **do**
  $g_k \in \partial f(x_k)$
  $$\lambda_k = \frac{d_k}{\sqrt{G^2 + \sum_{i=0}^{k}\|g_i\|^2}}$$
  $s_{k+1} = s_k + \lambda_k g_k$
  $$\hat{d}_{k+1} = \frac{\|s_{k+1}\|^2 - \sum_{i=0}^{k}\lambda_i^2\|g_i\|^2}{2\|s_{k+1}\|}$$
  $d_{k+1} = \max(d_k, \hat{d}_{k+1})$
  $x_{k+1} = x_k - \lambda_k g_k$
end for
Return $\hat{x}_n = \frac{1}{\sum_{k=0}^{n}\lambda_k}\sum_{k=0}^{n}\lambda_k x_k$

Algorithm 3 D-Adapted AdaGrad

Input: x_0, d_0 (default 10^{-6}), G_∞
$s_0 = 0$, $a_0 = [G_\infty, \dots, G_\infty]$
for $k = 0$ **to** n **do**
  $g_k \in \partial f(x_k, \xi_k)$
  $s_{k+1} = s_k + d_k g_k$
  $a_{k+1}^2 = a_k^2 + g_k^2$
  $A_{k+1} = \text{diag}(a_{k+1})$
  $$\hat{d}_{k+1} = \frac{\|s_{k+1}\|^2_{A_{k+1}^{-1}} - \sum_{i=0}^{k} d_i^2\|g_i\|^2_{A_i^{-1}}}{2\|s_{k+1}\|_1}$$
  $d_{k+1} = \max(d_k, \hat{d}_{k+1})$
  $x_{k+1} = x_0 - A_{k+1}^{-1}s_{k+1}$
end for
Return $\hat{x}_n = \frac{1}{\sum_{k=0}^{n} d_k}\sum_{k=0}^{n} d_k x_k$

Theorem 3.1. *For a convex p-dimensional function with* $G_\infty = \max_x\|\nabla f(x)\|_\infty$*, D-Adapted AdaGrad (Algorithm 3) returns a point* $\hat{x}_n$ *such that*

$$f(\hat{x}_n) - f_* = \mathcal{O}\left(\frac{\|a_{n+1}\|_1 D_\infty}{n+1}\right) = \mathcal{O}\left(\frac{pG_\infty D_\infty}{\sqrt{n+1}}\right),$$

as $n\to\infty$*, where* $D_\infty = \|x_0 - x_*\|_\infty$ *for any* x_* *in the set of minimizers of f, as long as* $d_0 \le D_\infty$*.*

Similarly to Theorem 2.2, we could achieve the same result up to higher order terms without using G_∞ in the initialization of a_0. Following the standard approach for AdaGrad, Algorithm 3 maintains a vector a to track the coordinate-wise denominator. We introduce a diagonal matrix A_{k+1} which allows us to avoid using coordinate-wise notation.

4. Discussion

Figure 1 depicts the behavior of D-Adaptation on a toy problem - minimizing an absolute value function starting at $x_0 = 1.0$. Here d_0 is started at 0.1, below the known D value of 1.0. This example illustrates the growth of d_k towards D. The value of d_k typically doesn't asymptotically approach D, as this is not guaranteed nor required by our theory. Instead, we show in Theorem F.1 that under a mild assumption, d_k is asymptotically greater than or equal to $D/(1+\sqrt{3})$. The lower bound $\hat{d}_k$ will often start to decrease, and even go negative, once d_k is large enough. Negative values of $\hat{d}_k$ were seen in most of the experiments in Section 7.

4.1. Different ways to estimate D

Algorithm 3 is presented with two options for estimating $\hat{d}_k$, where the numerator of the second option is provably larger or equal to that of the first option:

$$\sum_{k=0}^{n} \gamma_k d_k \langle g_k, s_k \rangle \geq \frac{\gamma_{n+1}}{2} \|s_{n+1}\|^2 - \sum_{k=0}^{n} \frac{\gamma_k}{2} d_k^2 \|g_k\|^2 .$$

We found the two options worked equally well in practice. The inner product between the step direction s_k and the gradient g_k, which shows up in the second option, is a quantity known as the (negative) hyper-gradient (Bengio, 2000; Domke, 2012; Pedregosa, 2016; Feurer & Hutter, 2019; Chandra et al., 2022; Wang et al., 2021). In classical applications of the hyper-gradient, the learning rate is increased when the gradient points in the same direction as the previous step, and it is decreased otherwise. In essence, the hyper-gradient indicates if the current learning rate is too large or to small. In works that use hyper-gradient to estimate learning rate, an additional hyper-learning rate parameter is needed to control the rate of change of the learning rate, whereas our approach requires no extra parameters.

In our approach, the hyper-gradient quantity is used to provide an actual estimate of the *magnitude* of the optimal learning rate (or more precisely a lower bound), which is far more information than just a directional signal of too-large or too-small. This is important for instance when a learning rate schedule is being used, as we can anneal the learning rate down over time, without the hyper-gradient responding by pushing the learning rate back up. This is also useful during learning rate warmup, as we are able to build an estimate of D during the warmup, which is not possible when using a classical hyper-gradient approach.

5. Related Work

We are not aware of any existing approach for convex Lipschitz (potentially non-smooth) optimization that avoids the need for knowledge of any hyper-parameters while still achieving the optimal rate asymptotically. Drori & Taylor (2020); Goujaud et al. (2022) give an algorithm for non-smooth convex problems which has no-tunable parameters, but requires an exact line-search. The Polyak step size (Polyak, 1987) is one approach that avoids requiring knowledge of D, instead, knowledge of f_* is required. Using estimates or approximations of f_* tend to result in unstable convergence, however a restarting scheme that maintains lower bounds on f_* can be shown to converge within a multiplicative log factor of the optimal rate (Hazan & Kakade, 2019).

Instead of running sub-gradient descent on every grid-point on a log spaced grid from d_0 to $d_{\max}$, a bisection algorithm can be run instead on the same grid, resulting in a double-logarithmic term (Carmon & Hinder, 2022).

Approaches from Online Learning can be applied here such as coin-betting (Orabona, 2019; Orabona & Tommasi, 2017; McMahan & Orabona, 2014; Zhang et al., 2022; Orabona & Pál, 2021). Asymptotic rates for these methods in the Offline Lipschitz setting are not currently known. In terms of non-asymptotic rates, their theoretical convergence rate is better by a factor $\sqrt{log(1+D/d_0)}$ than our non-asymptotic rate. Another approach in the Online Learning setting is Streeter & McMahan (2012)'s reward-doubling technique, which tracks similar norm quantities to our approach, although they estimate a different quantity than D.

Like our work, the DoG (Distance Over Gradients) approach of Ivgi et al. (2023) builds upon Carmon & Hinder (2022). They estimate D by

$$\bar{r}_k = \max_{i \leq k} \|x_i - x_0\| .$$

This estimator is not necessarily bounded; they show a convex counter-example where $\bar{r}_k$ goes to infinity. Nevertheless, by adding additional dampening in the denominator of the step-size, they are able to show learning-rate free convergence in the stochastic setting. Their result is more general than ours, as we only prove convergence in the non-stochastic setting, although their rate contains additional multiplicative log-factors compared to our rate. Their work is concurrent with ours, and appears in the same venue.

6. Machine Learning Applications

It is straightforward to adapt the D-Adaptation technique to stochastic optimization, although the theory no longer directly supports this case. Algorithm 4 and 5 are versions

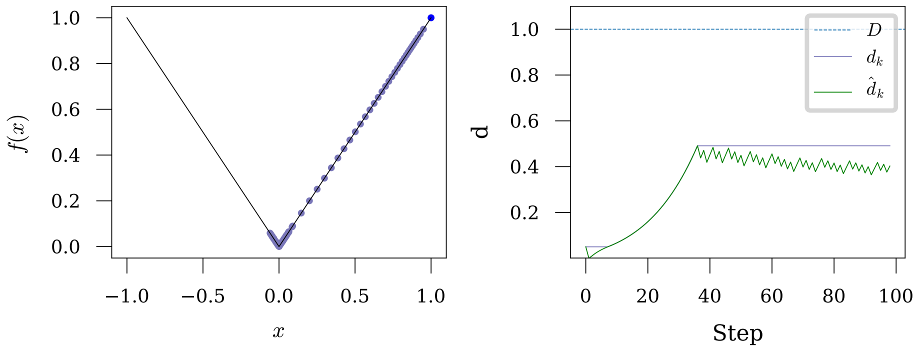

Figure 1. Toy problem illustrating the estimate of D over time, $f(x) = |x|$. $x_0 = 1.0$ is shown as a blue dot on the left plot, and the following iterates are shown in purple.

Algorithm 4 SGD with D-Adaptation

Input: x_0,
d_0 (default 10^{-6}),
γ_k (default 1),
$\beta = 0.9$,
G (default $\|g_0\|$)
$s_0 = 0, z_0 = x_0$
for $k = 0$ **to** n **do**
  $g_k \in \partial f(x_k, \xi_k)$
  $\lambda_k = \dfrac{d_k \gamma_k}{G}$
  $s_{k+1} = s_k + \lambda_k g_k$
  $z_{k+1} = z_k - \lambda_k g_k$
  $x_{k+1} = \beta x_k + (1-\beta) z_{k+1}$
  $\hat{d}_{k+1} = \dfrac{2\sum_{i=0}^{k} \lambda_i \langle g_i, s_i \rangle}{\|s_{k+1}\|}$
  $d_{k+1} = \max(d_k, \hat{d}_{k+1})$
end for

Algorithm 5 Adam with D-Adaptation

Input: x_0,
d_0 (default 10^{-6}),
γ_k (default 1),
$\beta_1, \beta_2, \epsilon$ (default 0.9, 0.999, 10^{-8}).
$s_0 = 0, m_0 = 0, v_0 = 0, r_0 = 0$
for $k = 0$ **to** n **do**
  $g_k \in \partial f(x_k, \xi_k)$
  $m_{k+1} = \beta_1 m_k + (1-\beta_1) d_k \gamma_k g_k$
  $v_{k+1} = \beta_2 v_k + (1-\beta_2) g_k^2$
  $A_{k+1} = \text{diag}(\sqrt{v_{k+1}} + \epsilon)$
  $x_{k+1} = x_k - A_{k+1}^{-1} m_{k+1}$
  Learning rate update
  $s_{k+1} = \sqrt{\beta_2} s_k + (1-\sqrt{\beta_2}) d_k \gamma_k g_k$
  $r_{k+1} = \sqrt{\beta_2} r_k + (1-\sqrt{\beta_2}) d_k \gamma_k \langle g_k, s_k \rangle_{A_{k+1}^{-1}}$
  $\hat{d}_{k+1} = \dfrac{r_{k+1}}{(1-\sqrt{\beta_2}) \|s_{k+1}\|_1}$
  $d_{k+1} = \max(d_k, \hat{d}_{k+1})$
end for

of D-Adaptation for SGD and Adam respectively. Both of the two methods solve the stochastic optimization problem,

$$\min_{x \in \mathbb{R}^p} \mathbb{E}[f(x, \xi)]$$

using stochastic subgradients $g_k \in \partial f(x_k, \xi_k)$.

For the SGD variant (Algorithm 1), we multiply the D bound by a factor of two compared to Algorithm 4. This improves the practical performance of the method. Our theoretical rate is still valid up to constant factors, for any constant multiplier applied to the step-size, so this change is still covered by our theory. For the denominator of the step size, we use $G = \|g_0\|$, which is a crude approximation to the true G but appears to work very well in practice.

We include momentum (β) implemented using the primal averaging technique, following the approach of Defazio (2020) and Defazio & Gower (2021). For Adam, we make the following modifications:

- The norms are now weighted instead of unweighted.
- Since s_k is now updated by an exponential moving average, a correction factor of $1 - \sqrt{\beta_2}$ in the D bound is needed to keep everything at the same scale.
- The Adam variant adapts quicker than the SGD variant and we found no constant multiplier was needed for $\hat{d}$.

A derivation of the weights of this Adam variant is included in Appendix G. We use $\hat{d}$ Option II for both methods, which only makes a practical difference for the Adam variant; for the SGD case it is exactly equivalent to Option I.

We include an optional γ_k constant sequence as input to the algorithms. This sequence should be set following a learning rate schedule if one is needed for the problem. This schedule should consider 1.0 as the base value, increase towards 1.0 during warm-up (if needed), and decrease from 1 during learning rate annealing. Typically the same schedule can be used as would normally be used without D-Adaptation.

7. Experimental Results

We compared our D-Adapted variants of Adam and SGD on a range of machine learning problems to demonstrate their effectiveness in practice. Unless otherwise mentioned, we used the standard learning rate schedule typically used for the problem, with the *base* learning rate set by D-Adaptation. Full hyper-parameter settings for each problem are included in the Appendix.

Convex Problems For our convex experiments, we considered logistic regression applied to 12 commonly used benchmark problems from the LIBSVM repository. In each case, we consider 100 epochs of training, with a stage-wise schedule with 10-fold decreases at 60, 80, and 95 epochs. The learning rate for Adam was chosen as the value that gave the highest final accuracy using a grid search. D-Adaptation matches or exceeds the performance of the grid-search based learning rate on all 12 problems, to within 0.5% accuracy (Figure 4, in the Appendix).

Convolutional Image Classification For a convolutional image classification benchmark, we used the three most common datasets used for optimization method testing: CIFAR10, CIFAR100 (Krizhevsky, 2009) and ImageNet 2012 (Russakovsky et al., 2015). We varied the architectures to show the flexibility of D-Adaptation, using a Wide ResNet (Zagoruyko & Komodakis, 2016), a DenseNet (Huang et al., 2017) and a vanilla ResNet model (He et al., 2016) respectively. D-Adaptation matches or exceeds the baseline learning rates on each problem.

LSTM Recurrent Neural Networks The IWSLT14 German-to-English dataset (Cettolo et al., 2014) is a standard choice for benchmarking machine translation models. We trained an LSTM model (Wiseman & Rush, 2016) commonly used for this problem. The standard training procedure includes an inverse-square-root learning rate schedule, which we used for both the baseline and for D-Adaptation. Our model achieves comparable performance to the baseline training regimen without any need to tune the learning rate.

Masked Language Modelling Bidirectional Encoder Representations from Transformers (BERT) is a popular approach to pretraining transformer models (Devlin et al., 2019). We use the 110M parameter RoBERTA variant (Liu et al., 2019) of BERT for our experiments. This model size provides a large and realistic test problem for D-Adaptation. We train on the Book-Wiki corpus (combining books from Zhu et al. (2015) and a snapshot of Wikipedia). D-Adaptation again matches the baseline in test-set perplexity.

Auto-regressive Language Modelling For our experiments on auto-regressive language modelling, we used the original GPT decoder-only transformer architecture (Radford et al., 2019). This model is small enough to train on a single machine, unlike the larger GPT-2/3 models. Its architecture is representative of other large language models. We trained on the large Book-Wiki corpus. D-Adaptation is comparable to the baseline with only a negligible perplexity difference.

Object Detection The COCO 2017 object detection task is a popular benchmark in computer vision. We trained as Faster-RCNN (Ren et al., 2015) model as implemented in Detectron2 (Wu et al., 2019). For the backbone model, we used a pretrained ResNeXt-101-32x8d (Xie et al., 2017), the largest model available in Detectron2 for this purpose. Our initial experiments showed D-Adaptation overfitting. We identified that the default decay of 0.0001 in the code-base was not optimized for this backbone model, and increasing it to 0.00015 improved the test set accuracy for both the baseline (42.67 to 42.99) and D-adapted versions (41.92 to 43.07), matching the published result of 43 for this problem.

Vision Transformers Vision transformers (Dosovitskiy et al., 2021) are a recently developed approach to image classification that differ significantly from the image classification approaches in Section 7. They are closer to the state-of-the-art than ResNet models, and require significantly more resources to train to high accuracy. We use the `vit_tiny_patch16_224` model in the *PyTorch Image Models* framework (Wightman, 2019) as it is small enough to train on 8 GPUs. The standard training pipeline uses a cosine learning rate schedule. This is an example of a situation where D-Adaptation under-performs the baseline learning rate. This problem appears to be highly sensitive to the initial learning rate, which may explain the observed differences.

fastMRI The fastMRI Knee Dataset (Zbontar et al., 2018) is a large-scale release of raw MRI data. The reconstruction task consists of producing a 2-dimensional, grey-scale image of the anatomy from the raw sensor data, under varying under-sampling regimes. We trained a VarNet 2.0 (Sriram et al., 2020) model, a strong baseline model on this dataset, using the code and training setup released by Meta (Knoll et al., 2020; Defazio, 2019). We again match the highly tuned baseline learning rate with D-Adaptation.

Recommendation Systems The Criteo Kaggle Display Advertising dataset is a large, sparse dataset of user click-through events. The DLRM (Naumov et al., 2019) model

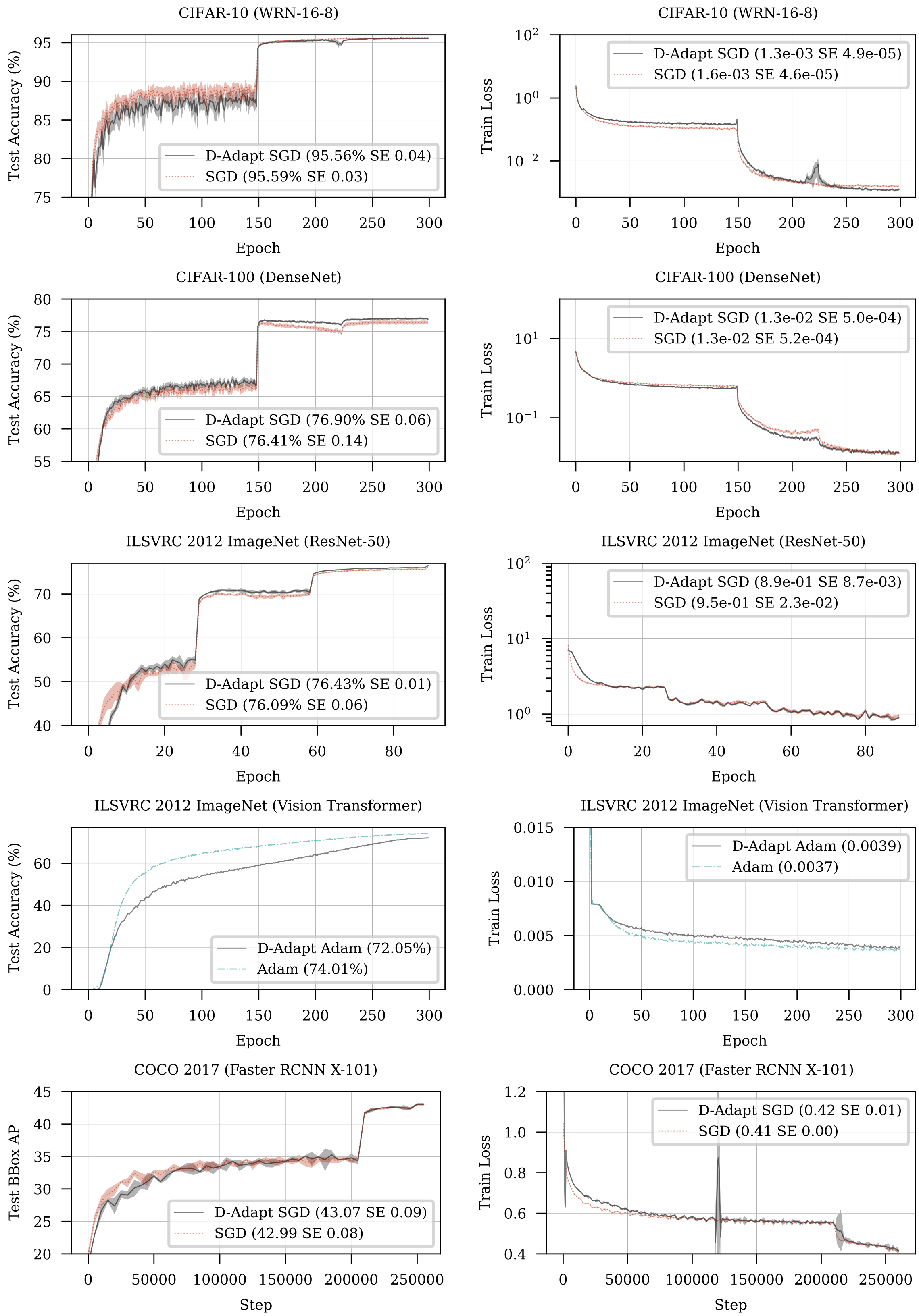

Figure 2. Vision experiments.

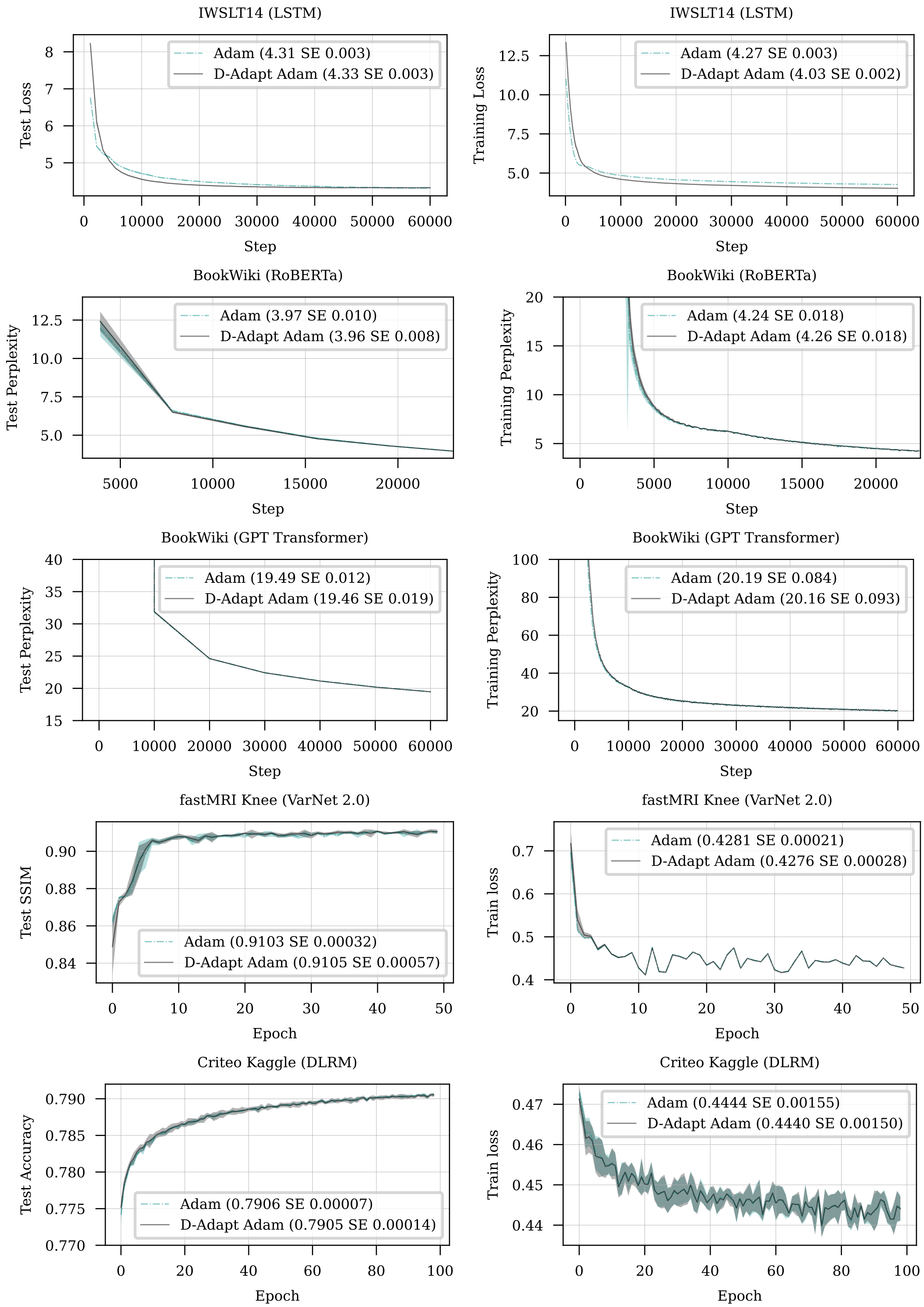

Figure 3. NLP, medical and recommendation systems.

Problem	Baseline LR	D-Adapted LR	Std. Dev.
CIFAR10	1.0	2.085	0.078
CIFAR100	0.5	0.4544	0.029
ImageNet	1.0	0.9227	0.084
IWSLT	0.01	0.003945	0.000086
GPT	0.001	0.0009218	0.000014
RoBERTa	0.001	0.0009331	0.000011
COCO	0.2	0.2004	0.0026
ViT	0.001	0.0073	0.00085
fastMRI	0.0003	0.0007596	0.00022
DLRM	0.0001	0.0001282	0.000056

Table 1. Comparison of baseline learning rates against final D-Adapted learning rates for the deep learning experiments, with average and standard deviation shown over multiple seeds.

is a common benchmark for this problem, representative of personalization and recommendation systems used in industry. Our method closely matches the performance of the tuned baseline learning rate.

7.1. Sensitivity to d_0

According to our theory, as long as each training run reaches the asymptotic regime the resulting final loss should be independent of the choice of d_0, as long as $d_0 \leq D$. We tested this hypothesis by running each of the 12 convex logistic regression problems using values of d_0 ranging from 10^{-16} to 10^{-2}. Figure 5 (Appendix E) shows that across every dataset, there is no dependence on the initial value of d_0. Given these results, we do not consider d_0 a hyper-parameter. There is no indication that d_0 should be tuned in practice.

7.2. Observed learning rates

Table 1 shows the learning rates obtained by D-Adaptation for each of our deep learning experiments. The adapted values show remarkable similarity to the hand-tuned values. The hand-tuned learning rates are given by either the paper or the public source code for each problem; It's unclear to what granularity they were tuned. In some cases D-Adaptation gives notably higher learning rates, such as for CIFAR-10. For SGD experiments, we used PyTorch's dampening parameter for implementation consistency with Adam. This requires the learning rate to be multiplied by $1/(1-\beta_1)$ compared to the undampened values, which is reflected in the baseline learning rates in this table.

We observed that in cases where there is a wide range of good learning rates that give equal final test results, D-Adaptation has a tendency to choose values at the higher end of the range. For instance, on CIFAR10, using learning rate 2.0 instead of the baseline 1.0 gives equal final test accuracy. The default of 1.0 is likely used in practice just for simplicity.

8. Conclusion

We have presented a simple approach to achieving parameter free learning of convex Lipshitz functions, by constructing successively better lower bounds on the key unknown quantity: the distance to solution $\|x_0 - x_*\|$. Our approach for constructing these lower bounds may be of independent interest. Our method is also highly practical, demonstrating excellent performance across a range of large and diverse machine learning problems.

Acknowledgements

We would like to thank Ashok Cutkosky for suggesting substantial simplifications to the proof of Lemma A.2.

References

Beck, A. *Introduction to Nonlinear Optimization*. Society for Industrial and Applied Mathematics, 2014. (Cited on page 3)

Bengio, Y. Gradient-based optimization of hyperparameters. In *Neural Computation*, 2000. (Cited on page 4)

Carmon, Y. and Hinder, O. Making SGD parameter-free. In *Conference on Learning Theory*. PMLR, 2022. (Cited on pages 2 and 4)

Cettolo, M., Niehues, J., Stüker, S., Bentivogli, L., and Federico, M. Report on the 11th IWSLT evaluation campaign. In *IWSLT*, 2014. (Cited on page 6)

Chandra, K., Xie, A., Ragan-Kelley, J., and Meijer, E. Gradient descent: The ultimate optimizer. In *36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022. (Cited on page 4)

Defazio, A. Offset sampling improves deep learning based accelerated mri reconstructions by exploiting symmetry. *arXiv preprint arXiv:1912.01101*, 2019. (Cited on page 6)

Defazio, A. Momentum via primal averaging: Theoretical insights and learning rate schedules for non-convex optimization, 2020. (Cited on page 5)

Defazio, A. and Gower, R. M. The power of factorial powers: New parameter settings for (stochastic) optimization. In *Proceedings of The 13th Asian Conference on Machine Learning*, volume 157 of *Proceedings of Machine Learning Research*. PMLR, 2021. (Cited on pages 3 and 5)

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the*

Association for Computational Lingustics, 2019. (Cited on page 6)

Domke, J. Generic methods for optimization-based modeling. In *Fifteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012. (Cited on page 4)

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. (Cited on page 6)

Drori, Y. and Taylor, A. B. Efficient first-order methods for convex minimization: a constructive approach. *Mathematical Programming*, 2020. (Cited on page 4)

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61), 2011. (Cited on pages 1 and 22)

Feurer, M. and Hutter, F. *Automated Machine Learning*, chapter Hyperparameter Optimization. Springer International Publishing, 2019. (Cited on page 4)

Goujaud, B., Taylor, A., and Dieuleveut, A. Optimal first-order methods for convex functions with a quadratic upper bound. Technical report, INRIA, 2022. (Cited on page 4)

Hazan, E. and Kakade, S. M. Revisiting the polyak step size. Technical report, Google AI Princeton, 2019. (Cited on page 4)

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016. (Cited on page 6)

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, 2017. doi: 10.1109/CVPR.2017.243. (Cited on page 6)

Ivgi, M., Hinder, O., and Carmon, Y. Dog is sgd's best friend: A parameter-free dynamic step size schedule, 2023. (Cited on page 4)

Jain, P., Nagaraj, D., and Netrapalli, P. Making the last iterate of sgd information theoretically optimal. *Conference On Learning Theory (COLT)*, 2019. (Cited on page 3)

Knoll, F., Zbontar, J., Sriram, A., Muckley, M. J., Bruno, M., Defazio, A., Parente, M., Geras, K. J., Katsnelson, J., Chandarana, H., Zhang, Z., Drozdzalv, M., Romero, A., Rabbat, M., Vincent, P., Pinkerton, J., Wang, D., Yakubova, N., Owens, E., Zitnick, C. L., Recht, M. P., Sodickson, D. K., and Lui, Y. W. fastMRI: A publicly available raw k-space and DICOM dataset of knee images for accelerated MR image reconstruction using machine learning. *Radiology: Artificial Intelligence*, 2(1), 2020. (Cited on page 6)

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. (Cited on page 6)

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. (Cited on page 6)

McMahan, H. B. and Orabona, F. Unconstrained online linear learning in hilbert spaces: Minimax algorithms and normal approximations. In *Proceedings of The 27th Conference on Learning Theory*, volume 35 of *Proceedings of Machine Learning Research*. PMLR, 2014. (Cited on page 4)

Naumov, M., Mudigere, D., Shi, H. M., Huang, J., Sundaraman, N., Park, J., Wang, X., Gupta, U., Wu, C., Azzolini, A. G., Dzhulgakov, D., Mallevich, A., Cherniavskii, I., Lu, Y., Krishnamoorthi, R., Yu, A., Kondratenko, V., Pereira, S., Chen, X., Chen, W., Rao, V., Jia, B., Xiong, L., and Smelyanskiy, M. Deep learning recommendation model for personalization and recommendation systems. *CoRR*, 2019. (Cited on page 6)

Nesterov, Y. *Lectures on Convex Optimization*. Springer Nature, 2018. (Cited on page 1)

Orabona, F. A modern introduction to online learning. Technical report, Boston University, 2019. (Cited on pages 2, 4, and 19)

Orabona, F. and Pál, D. Parameter-free stochastic optimization of variationally coherent functions, 2021. (Cited on page 4)

Orabona, F. and Tommasi, T. Training deep networks without learning rates through coin betting. In *Advances in Neural Information Processing Systems*, volume 30, 2017. (Cited on page 4)

Pedregosa, F. Hyperparameter optimization with approximate gradient. In *International conference on machine learning*, pp. 737–746. PMLR, 2016. (Cited on page 4)

Polyak, B. T. *Introduction to optimization*. Optimization Software, Inc., 1987. (Cited on page 4)

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pre-training. Technical report, OpenAI, 2019. (Cited on page 6)

Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. (Cited on page 6)

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3), 2015. (Cited on page 6)

Sriram, A., Zbontar, J., Murrell, T., Defazio, A., Zitnick, C. L., Yakubova, N., Knoll, F., and Johnson, P. End-to-end variational networks for accelerated MRI reconstruction. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2020. (Cited on page 6)

Streeter, M. and McMahan, H. B. Less regret via online conditioning, 2010. (Cited on pages 1 and 14)

Streeter, M. and McMahan, H. B. No-regret algorithms for unconstrained online convex optimization. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS)*, 2012. (Cited on page 4)

Wang, X., Yuan, S., Wu, C., and Ge, R. Guarantees for tuning the step size using a learning-to-learn approach. In *Proceedings of the 38th International Conference On Machine Learning*, 2021. (Cited on page 4)

Ward, R., Wu, X., and Bottou, L. Adagrad stepsizes: sharp convergence over nonconvex landscapes. In *International Conference on Machine Learning*, 2019. (Cited on page 1)

Wightman, R. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. (Cited on page 6)

Wiseman, S. and Rush, A. M. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2016. (Cited on page 6)

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., and Girshick, R. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019. (Cited on page 6)

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. (Cited on page 6)

Zagoruyko, S. and Komodakis, N. Wide residual networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016. (Cited on page 6)

Zbontar, J., Knoll, F., Sriram, A., Muckley, M. J., Bruno, M., Defazio, A., Parente, M., Geras, K. J., Katsnelson, J., Chandarana, H., et al. fastMRI: An open dataset and benchmarks for accelerated MRI. *arXiv preprint arXiv:1811.08839*, 2018. (Cited on page 6)

Zhang, Z., Cutkosky, A., and Paschalidis, I. C. Pde-based optimal strategy for unconstrained online learning. In *Proceedings of the 39th International Conference on Machine Learning (ICML 2022)*, 2022. (Cited on page 4)

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, 2015. (Cited on page 6)

A. Core Theory

Here, we are going to consider a more general form of Algorithm 1 with arbitrary positive weights λ_k that do not have to be equal to d_k. In particular, we will study the update rule

$$s_{n+1} = s_n + \lambda_n g_n \qquad \text{and} \qquad \hat{d}_{n+1} = \frac{\gamma_{n+1}\|s_{n+1}\|^2 - \sum_{k=0}^{n} \gamma_k \lambda_k^2 \|g_k\|^2}{2\|s_{n+1}\|}.$$

Later in the proofs, we will set $\lambda_k = d_k$, but most intermediate results are applicable with other choices of λ_k as well.

Lemma A.1. *The inner product $\gamma_k \lambda_k \langle g_k, s_k\rangle$ is a key quantity that occurs in our theory. We can bound the sum of these inner products over time by considering the following expansion:*

$$-\sum_{k=0}^{n} \gamma_k \lambda_k \langle g_k, s_k\rangle = -\frac{\gamma_{n+1}}{2}\|s_{n+1}\|^2 + \sum_{k=0}^{n} \frac{\gamma_k}{2}\lambda_k^2 \|g_k\|^2 + \frac{1}{2}\sum_{k=0}^{n}(\gamma_{k+1}-\gamma_k)\|s_{k+1}\|^2.$$

This simplifies when $\gamma_k = \gamma_{n+1}$ and the weighting sequence is flat, i.e., if $\lambda_k = 1$ for all k:

$$-\gamma_{n+1}\sum_{k=0}^{n} \langle g_k, s_k\rangle = -\frac{\gamma_{n+1}}{2}\|s_{n+1}\|^2 + \frac{\gamma_{n+1}}{2}\sum_{k=0}^{n}\|g_k\|^2,$$

with λ weights:

$$-\gamma_{n+1}\sum_{k=0}^{n} \lambda_k\langle g_k, s_k\rangle = -\frac{\gamma_{n+1}}{2}\|s_{n+1}\|^2 + \frac{\gamma_{n+1}}{2}\sum_{k=0}^{n}\lambda_k^2\|g_k\|^2.$$

Proof. This is straightforward to show by induction (it's a consequence of standard DA proof techniques, where $\|s_n\|^2$ is expanded).

$$\begin{aligned}\frac{\gamma_{n+1}}{2}\|s_{n+1}\|^2 &= \frac{\gamma_n}{2}\|s_{n+1}\|^2 + \frac{1}{2}(\gamma_{n+1}-\gamma_n)\|s_{n+1}\|^2 \\ &= \frac{\gamma_n}{2}\|s_n\|^2 + \gamma_n\lambda_n\langle g_n, s_n\rangle + \frac{\gamma_n}{2}\lambda_n^2\|g_n\|^2 + \frac{1}{2}(\gamma_{n+1}-\gamma_n)\|s_{n+1}\|^2.\end{aligned}$$

Therefore

$$-\gamma_n\lambda_n\langle g_n, s_n\rangle = \frac{\gamma_n}{2}\|s_n\|^2 - \frac{\gamma_{n+1}}{2}\|s_{n+1}\|^2 + \frac{\gamma_n}{2}\lambda_n^2\|g_n\|^2 + \frac{1}{2}(\gamma_{n+1}-\gamma_n)\|s_{n+1}\|^2.$$

Telescoping

$$-\sum_{k=0}^{n} \gamma_k \lambda_k \langle g_k, s_k\rangle = -\frac{\gamma_{n+1}}{2}\|s_{n+1}\|^2 + \sum_{k=0}^{n} \frac{\gamma_k}{2}\lambda_k^2 \|g_k\|^2 + \frac{1}{2}\sum_{k=0}^{n}(\gamma_{k+1}-\gamma_k)\|s_{k+1}\|^2.$$

□

Lemma A.2. *The iterates of Algorithm 1 satisfy*

$$\sum_{k=0}^{n}\lambda_k(f(x_k)-f_*) \leq \|x_0 - x_*\|\|s_{n+1}\| + \sum_{k=0}^{n}\frac{\gamma_k}{2}\lambda_k^2\|g_k\|^2 - \frac{\gamma_{n+1}}{2}\|s_{n+1}\|^2.$$

Proof. Starting from convexity:

$$\begin{aligned}\sum_{k=0}^{n}\lambda_k\left(f(x_k)-f_*\right) &\leq \sum_{k=0}^{n}\lambda_k\left\langle g_k, x_k-x_*\right\rangle \\ &= \sum_{k=0}^{n}\lambda_k\left\langle g_k, x_k-x_0+x_0-x_*\right\rangle \\ &= \left\langle s_{n+1}, x_0-x_*\right\rangle + \sum_{k=0}^{n}\lambda_k\left\langle g_k, x_k-x_0\right\rangle \\ &= \left\langle s_{n+1}, x_0-x_*\right\rangle - \sum_{k=0}^{n}\lambda_k\gamma_k\left\langle g_k, s_k\right\rangle \\ &\leq \left\|s_{n+1}\right\|\left\|x_0-x_*\right\| - \sum_{k=0}^{n}\lambda_k\gamma_k\left\langle g_k, s_k\right\rangle.\end{aligned}$$

We can further simplify with:

$$-\sum_{k=0}^{n}\gamma_k\lambda_k\left\langle g_k, s_k\right\rangle = -\frac{\gamma_{n+1}}{2}\left\|s_{n+1}\right\|^2 + \sum_{k=0}^{n}\frac{\gamma_k}{2}\lambda_k^2\left\|g_k\right\|^2 + \frac{1}{2}\sum_{k=0}^{n}\left(\gamma_{k+1}-\gamma_k\right)\left\|s_{k+1}\right\|^2.$$

Using the fact that $\gamma_{k+1}-\gamma_k \leq 0$ we have:

$$\begin{aligned}\sum_{k=0}^{n}\lambda_k\left(f(x_k)-f_*\right) &\leq \left\|x_0-x_*\right\|\left\|s_{n+1}\right\| - \sum_{k=0}^{n}\gamma_k\lambda_k\left\langle g_k, s_k\right\rangle \\ &\leq \left\|x_0-x_*\right\|\left\|s_{n+1}\right\| + \sum_{k=0}^{n}\frac{\gamma_k}{2}\lambda_k^2\left\|g_k\right\|^2 - \frac{\gamma_{n+1}}{2}\left\|s_{n+1}\right\|^2.\end{aligned}$$

□

Theorem A.3. *For Algorithm 1, the initial distance to solution, $D = \|x_0 - x_*\|$, can be lower bounded as follows*

$$D \geq \hat{d}_{n+1} = \frac{\gamma_{n+1}\left\|s_{n+1}\right\|^2 - \sum_{k=0}^{n}\gamma_k\lambda_k^2\left\|g_k\right\|^2}{2\left\|s_{n+1}\right\|}.$$

Proof. The key idea is that the bound in Lemma A.2,

$$\sum_{k=0}^{n}\lambda_k\left(f(x_k)-f_*\right) \leq D\left\|s_{n+1}\right\| + \sum_{k=0}^{n}\frac{\gamma_k}{2}\lambda_k^2\left\|g_k\right\|^2 - \frac{\gamma_{n+1}}{2}\left\|s_{n+1}\right\|^2,$$

gives some indication as to the magnitude of D in the case when the other terms on the right are negative. To proceed, we use $\sum_{k=0}^{n}\lambda_k\left(f(x_k)-f_*\right) \geq 0$, giving:

$$0 \leq D\left\|s_{n+1}\right\| + \sum_{k=0}^{n}\frac{\gamma_k}{2}\lambda_k^2\left\|g_k\right\|^2 - \frac{\gamma_{n+1}}{2}\left\|s_{n+1}\right\|^2,$$

which we can rearrange to:

$$D\left\|s_{n+1}\right\| \geq \frac{\gamma_{n+1}}{2}\left\|s_{n+1}\right\|^2 - \sum_{k=0}^{n}\frac{\gamma_k}{2}\lambda_k^2\left\|g_k\right\|^2.$$

Therefore:

$$D \geq \frac{\frac{\gamma_{n+1}}{2}\left\|s_{n+1}\right\|^2 - \sum_{k=0}^{n}\frac{\gamma_k}{2}\lambda_k^2\left\|g_k\right\|^2}{\left\|s_{n+1}\right\|}.$$

□

Lemma A.4. *In Algorithm 1, the norm of s_{n+1} is bounded by:*

$$\|s_{n+1}\| \leq \frac{2d_{n+1}}{\gamma_{n+1}} + \frac{\sum_{k=0}^{n} \gamma_k \lambda_k^2 \|g_k\|^2}{2d_{n+1}}. \tag{2}$$

Proof. Using the definition of $\hat{d}_{n+1}$ from Theorem A.3, and the property $\hat{d}_{n+1} \leq d_{n+1}$, we derive

$$\frac{\gamma_{n+1}}{2} \|s_{n+1}\|^2 - \sum_{k=0}^{n} \frac{\gamma_k}{2} \lambda_k^2 \|g_k\|^2 = \hat{d}_{n+1} \|s_{n+1}\| \leq d_{n+1} \|s_{n+1}\|.$$

Using inequality $2\alpha\beta \leq \alpha^2 + \beta^2$ with $\alpha^2 = \frac{2d_{n+1}^2}{\gamma_{n+1}}$ and $\beta^2 = \frac{\gamma_{n+1}}{2}\|s_{n+1}\|^2$ and then the bound above, we establish

$$2\alpha\beta = 2d_{n+1}\|s_{n+1}\| \leq \frac{2d_{n+1}^2}{\gamma_{n+1}} + \frac{\gamma_{n+1}}{2}\|s_{n+1}\|^2 \leq \frac{2d_{n+1}^2}{\gamma_{n+1}} + d_{n+1}\|s_{n+1}\| + \sum_{k=0}^{n} \frac{\gamma_k}{2}\lambda_k^2\|g_k\|^2.$$

Rearranging the terms, we obtain

$$d_{n+1}\|s_{n+1}\| \leq \frac{2d_{n+1}^2}{\gamma_{n+1}} + \sum_{k=0}^{n} \frac{\gamma_k}{2}\lambda_k^2\|g_k\|^2.$$

It remains to divide this inequality by d_{n+1} to get the desired claim. □

Proposition A.5. *(From Streeter & McMahan (2010)) The gradient error term can be bounded as:*

$$\sum_{k=0}^{n} \frac{\|g_k\|^2}{\sqrt{G^2 + \sum_{i=0}^{k-1} \|g_i\|^2}} \leq 2\sqrt{\sum_{k=0}^{n} \|g_k\|^2}. \tag{3}$$

Moreover, if $\gamma_k = \frac{1}{\sqrt{G^2 + \sum_{i=0}^{k-1} \|g_i\|^2}}$, then

$$\sum_{k=0}^{n} \frac{\gamma_k}{2} \|g_k\|^2 \leq \gamma_{n+1}\left(G^2 + \sum_{k=0}^{n} \|g_k\|^2\right). \tag{4}$$

Lemma A.6. *Setting $\lambda_k = d_k$, it holds for Algorithm 1:*

$$\sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) \leq 2Dd_{n+1}\sqrt{\sum_{k=0}^{n} \|g_k\|^2} + Dd_{n+1}\sum_{k=0}^{n} \gamma_k\|g_k\|^2.$$

Proof. First, recall the key bound from Lemma A.2:

$$\sum_{k=0}^{n} \lambda_k \left(f(x_k) - f_*\right) \leq D \|s_{n+1}\| - \frac{\gamma_{n+1}}{2} \|s_{n+1}\|^2 + \sum_{k=0}^{n} \frac{\gamma_k}{2} \lambda_k^2 \|g_k\|^2$$
$$\leq D \|s_{n+1}\| + \sum_{k=0}^{n} \frac{\gamma_k}{2} \lambda_k^2 \|g_k\|^2.$$

Now let us apply the bound from Lemma A.4:

$$\|s_{n+1}\| \leq \frac{2d_{n+1}}{\gamma_{n+1}} + \frac{\sum_{k=0}^{n} \gamma_k \lambda_k^2 \|g_k\|^2}{2d_{n+1}},$$

which gives

$$\sum_{k=0}^{n} \lambda_k \left(f(x_k) - f_*\right) \leq \frac{2Dd_{n+1}}{\gamma_{n+1}} + \frac{D\sum_{k=0}^{n} \gamma_k \lambda_k^2 \|g_k\|^2}{2d_{n+1}} + \sum_{k=0}^{n} \frac{\gamma_k}{2} \lambda_k^2 \|g_k\|^2.$$

Using $\lambda_k = d_k \leq d_{n+1} \leq D$ and plugging in the step size, we obtain

$$\begin{aligned}
\sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) &\leq \frac{2Dd_{n+1}}{\gamma_{n+1}} + \frac{D\sum_{k=0}^{n} \gamma_k d_{n+1}^2 \|g_k\|^2}{2d_{n+1}} + \sum_{k=0}^{n} \frac{\gamma_k}{2} d_{n+1}^2 \left\|g_k\right\|^2 \\
&\leq 2Dd_{n+1}\sqrt{\sum_{k=0}^{n} \left\|g_k\right\|^2} + \frac{1}{2}Dd_{n+1}\sum_{k=0}^{n}\gamma_k\|g_k\|^2 + \frac{1}{2}Dd_{n+1}\sum_{k=0}^{n}\gamma_k\left\|g_k\right\|^2 \\
&= 2Dd_{n+1}\sqrt{\sum_{k=0}^{n} \left\|g_k\right\|^2} + Dd_{n+1}\sum_{k=0}^{n}\gamma_k\|g_k\|^2.
\end{aligned}$$

This is exactly our result. □

A.1. Asymptotic analysis

Theorem. (Theorem 2.1) The average iterate $\hat{x}_n$ returned by Algorithm 1 satisfies:

$$f(\hat{x}_n) - f_* = \mathcal{O}\left(\frac{DG}{\sqrt{n+1}}\right).$$

Proof. In the case where $g_0 = 0$, $f(x_0) = f(x_*)$ and the theorem is trivially true, so we assume that $\left\|g_0\right\|^2 > 0$. We will show the result holds for some n, where we choose n sufficiently large so that a number of criteria are met:

Criterion 1: since d_k is a non-decreasing sequence upper bounded by D, there must exist some $\hat{n}$ such that after $\hat{n}$ steps, $d_k \geq \frac{1}{2}d_{n+1}$ for all $k, n \geq \hat{n}$. We take $n \geq 2\hat{n}$.

Criterion 2: since we assume the bound $\left\|g_k\right\|^2 \leq G^2$, there must exist some r such that $\left\|g_n\right\|^2 \leq \sum_{k=0}^{n-1}\left\|g_k\right\|^2$ for all $n \geq r$. Let us choose the smallest r that satisfies this condition, in which case $\|g_{r-1}\|^2 \geq \sum_{k=0}^{r-2}\|g_k\|^2$, otherwise we could have chosen $r-1$. Moreover, we have by definition $\gamma_k \leq \frac{1}{\|g_0\|}$ for all $k \leq r-1$. Combining this with the first bound from Proposition A.5, we derive

$$\begin{aligned}
\sum_{k=0}^{n}\gamma_k\left\|g_k\right\|^2 &= \sum_{k=r}^{n}\gamma_k\left\|g_k\right\|^2 + \sum_{k=0}^{r-1}\gamma_k\left\|g_k\right\|^2 \\
&\leq 2\sqrt{\sum_{k=r}^{n}\left\|g_k\right\|^2} + \frac{1}{\|g_0\|}\sum_{k=0}^{r-1}\left\|g_k\right\|^2 \\
&\leq 2\sqrt{\sum_{k=0}^{n}\left\|g_k\right\|^2} + \frac{2}{\|g_0\|}\left\|g_{r-1}\right\|^2 \\
&\leq 2\sqrt{\sum_{k=0}^{n}\left\|g_k\right\|^2} + 2\frac{G^2}{\|g_0\|}.
\end{aligned}$$

We continue with the bound from Lemma A.6:

$$\sum_{k=0}^{n} d_k\left(f(x_k) - f_*\right) \leq 2Dd_{n+1}\sqrt{\sum_{k=0}^{n}\left\|g_k\right\|^2} + Dd_{n+1}\sum_{k=0}^{n}\gamma_k\|g_k\|^2.$$

From Criterion 1, we have that:

$$\sum_{k=0}^{n} d_k \geq \sum_{k=\hat{n}}^{n} d_k \geq \sum_{k=\hat{n}}^{n}\frac{1}{2}d_{n+1} = \frac{1}{2}(n-\hat{n}+1)d_{n+1} \geq \frac{1}{4}(n+1)d_{n+1},$$

hence

$$\frac{1}{\sum_{k=0}^{n} d_k} \leq \frac{4}{(n+1)d_{n+1}}.$$

Plugging this back yields

$$\frac{1}{\sum_{k=0}^{n} d_k} \sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) \leq \frac{8D}{(n+1)} \sqrt{\sum_{k=0}^{n} \|g_k\|^2} + \frac{4D}{n+1} \sum_{k=0}^{n} \gamma_k \|g_k\|^2.$$

Using the bound obtained from Criterion 2, we further get

$$\frac{1}{\sum_{k=0}^{n} d_k} \sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) \leq \frac{8D}{(n+1)} \sqrt{\sum_{k=0}^{n} \|g_k\|^2} + \frac{4D}{n+1} \left(2\sqrt{\sum_{k=0}^{n} \|g_k\|^2} + 2\frac{G^2}{\|g_0\|}\right).$$

Using $\|g_k\|^2 \leq G^2$, we simplify this to

$$\frac{1}{\sum_{k=0}^{n} d_k} \sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) \leq \frac{16DG}{\sqrt{n+1}} + \frac{8DG^2}{(n+1)\|g_0\|}.$$

Using Jensen's inequality, we can convert this to a bound on the average iterate defined as

$$\hat{x}_n = \frac{1}{\sum_{k=0}^{n} d_k} \sum_{k=0}^{n} d_k x_k,$$

implying

$$f(\hat{x}_n) - f_* \leq \frac{12DG}{\sqrt{n+1}} + \frac{8DG^2}{(n+1)\|g_0\|}.$$

Note that the second term on the right decreases faster than the first term with respect to n, so

$$f(\hat{x}_n) - f_* = \mathcal{O}\left(\frac{DG}{\sqrt{n+1}}\right).$$

□

A.2. Non-asymptotic analysis

Lemma A.7. *Consider a sequence $d_0, \ldots d_{N+1}$, where for each k, $d_{k+1} \geq d_k$ and assume $N+1 \geq 2\log_2(d_{N+1}/d_0)$. Then*

$$\min_{n \leq N} \frac{d_{n+1}}{\sum_{k=0}^{n} d_k} \leq 4\frac{\log_{2+}(d_{N+1}/d_0)}{N+1}, \tag{5}$$

where $\log_{2+}(x) = \max(1, \log_2(x))$.

Proof. Let $r = \lceil \log_{2+}(d_{N+1}/d_0) \rceil$. We proceed by an inductive argument on r. In the base case, if $r \leq 2$, then $d_{n+1} \leq 4d_0$ and the result follows immediately:

$$\min_{n \leq N} \frac{d_{n+1}}{\sum_{k=0}^{n} d_k} \leq \frac{d_{N+1}}{\sum_{k=0}^{N} d_0} \leq \frac{4d_0}{(N+1)d_0}$$
$$= \frac{4}{N+1} \leq 4\frac{\log_{2+}(d_{N+1}/d_0)}{N+1}.$$

So assume that $r > 2$ and define $n' = \left\lceil N + 1 - \frac{N+1}{\log_{2+}(d_{N+1}/d_0)} \right\rceil$. First we show that no induction is needed, and we may take $n = N$, if $d_{n'} \geq \frac{1}{2} d_{N+1}$. In that case, since the sequence d_k is monotonic, it also holds

$$d_k \geq \frac{1}{2} d_{N+1}, \quad \text{for all } k \geq n'.$$

Then, it is easy to see that

$$\sum_{k=0}^{N} d_k \geq \sum_{k=n'}^{N} d_k \geq \frac{1}{2}\left(N+1-n'\right) d_{N+1} \geq \frac{1}{2}\left(N+1-\left(N+2-\frac{N+1}{\log_{2+}(d_{N+1}/d_0)}\right)\right) d_{N+1}$$
$$= \frac{1}{2}\left(\frac{N+1}{\log_{2+}(d_{N+1}/d_0)}-1\right) d_{N+1}.$$

Since we assume that $\frac{N+1}{\log_{2+}(d_{N+1}/d_0)} \geq 2$, we can reduce this bound to the following:

$$\sum_{k=0}^{N} d_k \geq \frac{1}{2}\left(\frac{(N+1)}{\log_{2+}(d_{N+1}/d_0)}-1\right) d_{N+1} \geq \frac{(N+1)\, d_{N+1}}{4\log_{2+}(d_{N+1}/d_0)}.$$

Rearranging this bound gives:

$$\frac{d_{N+1}}{\sum_{k=0}^{N} d_k} \leq 2\frac{\log_{2+}(d_{N+1}/d_0)}{N+1},$$

and therefore

$$\min_{n\leq N} \frac{d_{n+1}}{\sum_{k=0}^{n} d_k} \leq 4\frac{\log_{2+}(d_{N+1}/d_0)}{N+1}.$$

Thus, the claim holds if $d_{n'} \geq \frac{1}{2}d_{N+1}$.

Now, suppose that $d_{n'} \leq \frac{1}{2}d_{N+1}$. In that case, $\lceil \log_{2+}(d_{n'}/d_0)\rceil \leq \lceil \log_{2+}(\frac{1}{2}d_{N+1}/d_0)\rceil = r-1$ and by definition

$$n' \geq (N+1)\left(1-\frac{1}{\log_{2+}(d_{N+1}/d_0)}\right) \geq 2\log_2(d_{N+1}/d_0)\left(1-\frac{1}{\log_{2+}(d_{N+1}/d_0)}\right)$$
$$= 2\log_2(d_{N+1}/d_0)-1 = 2\log_2\left(\frac{1}{2}d_{N+1}/d_0\right) \geq 2\log_2(d_{n'}/d_0).$$

Therefore, we can apply the inductive hypothesis to the sequence $d_0,\ldots,d_{n'}$:

$$\min_{n\leq n'-1} \frac{d_{n+1}}{\sum_{k=0}^{n} d_k} \leq 4\frac{\log_{2+}(d_{n'}/d_0)}{n'}.$$

Under this inductive hypothesis, we note that:

$$\begin{aligned}
\frac{\log_{2+}(d_{n'}/d_0)}{n'} &= \frac{1}{\left\lceil N+1-\frac{N+1}{\log_{2+}(d_{N+1}/d_0)}\right\rceil}\log_{2+}(d_{n'}/d_0)\\
&\leq \frac{1}{N+1-\frac{N+1}{\log_{2+}(d_{N+1}/d_0)}}\log_{2+}(d_{n'}/d_0)\\
&= \frac{\log_{2+}(d_{N+1}/d_0)}{(N+1)\left(\log_{2+}(d_{N+1}/d_0)-1\right)}\log_{2+}(d_{n'}/d_0)\\
&= \frac{\log_{2+}(d_{N+1}/d_0)}{N+1}\cdot\frac{\log_{2+}(d_{n'}/d_0)}{\log_{2+}(d_{N+1}/d_0)-1}.
\end{aligned}$$

Let us now bound the last fraction. Since $r > 2$, we have $\log_2(d_{N+1}/d_0) \geq r-1 \geq 2$, so $\log_{2+}(\frac{1}{2}d_{N+1}/d_0) = \log_2(\frac{1}{2}d_{N+1}/d_0)$, and, therefore,

$$\log_{2+}(d_{n'}/d_0) \leq \log_{2+}\left(\frac{1}{2}d_{N+1}/d_0\right) = \log_{2+}(d_{N+1}/d_0)-1.$$

Plugging this back in yields:

$$\frac{\log_{2+}(d_{n'}/d_0)}{n'} \leq \frac{\log_{2+}(d_{N+1}/d_0)}{N+1}.$$

Putting it all together, we have that:

$$\min_{n\leq N}\frac{d_{n+1}}{\sum_{k=0}^{n}d_k}\leq\min_{n\leq n'}\frac{d_{n+1}}{\sum_{k=0}^{n}d_k}\leq 4\frac{\log_{2+}(d_{N+1}/d_0)}{N+1}.$$

□

Theorem. (Theorem 2.2) Consider Algorithm 1 run for n steps, where $n\geq 2\log_2(D/d_0)$, if we return the point $\hat{x}_t=\frac{1}{\sum_{k=0}^{t}d_k}\sum_{k=0}^{t}d_k x_k$ where t is chosen to be:

$$t=\arg\min_{k\leq n}\frac{d_{k+1}}{\sum_{i=0}^{k}d_i},$$

Then:

$$f(\hat{x}_t)-f_*\leq 16\frac{\log_{2+}(d_{n+1}/d_0)}{n+1}D\sqrt{\sum_{k=0}^{t}\|g_k\|^2}.$$

Proof. Consider the bound from Lemma A.6:

$$\begin{aligned}\frac{1}{\sum_{k=0}^{n}d_k}\sum_{k=0}^{n}d_k\left(f(x_k)-f_*\right)&\leq\frac{2Dd_{n+1}}{\sum_{k=0}^{n}d_k}\sqrt{\sum_{k=0}^{n}\|g_k\|^2}+\frac{Dd_{n+1}}{\sum_{k=0}^{n}d_k}\sum_{k=0}^{n}\gamma_k\|g_k\|^2\\&\overset{(3)}{\leq}\frac{2Dd_{n+1}}{\sum_{k=0}^{n}d_k}\sqrt{\sum_{k=0}^{n}\|g_k\|^2}+\frac{Dd_{n+1}}{\sum_{k=0}^{n}d_k}2\sqrt{\sum_{k=0}^{n}\|g_k\|^2}\\&=\frac{4Dd_{n+1}}{\sum_{k=0}^{n}d_k}\sqrt{\sum_{k=0}^{n}\|g_k\|^2}.\end{aligned}$$

Now using Lemma A.7, we can return the point $\hat{x}_t$ and at time $t=\arg\min_{k\leq n}\frac{d_{k+1}}{\sum_{i=0}^{k}d_i}$, ensuring that

$$\frac{d_{t+1}}{\sum_{k=0}^{t}d_k}=\min_{k\leq n}\frac{d_{k+1}}{\sum_{i=0}^{k}d_i}\overset{(5)}{\leq}4\frac{\log_{2+}(d_{n+1}/d_0)}{n+1},$$

giving us an upper bound:

$$f(\hat{x}_t)-f_*\leq 16\frac{\log_{2+}(d_{n+1}/d_0)}{n+1}D\sqrt{\sum_{k=0}^{t}\|g_k\|^2}.$$

□

We note that a similar proof can be used to remove the G^2 term from the numerator of γ_k. To this end, we could reuse the bound obtained in the proof of Theorem 2.1:

$$\sum_{k=0}^{n}\gamma_k\|g_k\|^2\leq 2\sqrt{\sum_{k=0}^{n}\|g_k\|^2}+2\frac{G^2}{\|g_0\|},$$

which holds for $\gamma_k=\frac{1}{\sqrt{\sum_{i=0}^{k-1}\|g_i\|^2}}$. In the proof of Theorem 2.1, this bound was stated for $n\geq r$, where r is the smallest number such that $\|g_k\|^2\leq\sum_{i=0}^{k-1}\|g_i\|^2$ for all $k\geq r$. However, the bound itself does not require $n\geq r$, since for $n<r$ it holds even without the first term in the right-hand side. The second term in that bound does not increase with n, and it would result in the following bound for the same iterate $\hat{x}_t$ as in Theorem 2.2:

$$f(\hat{x}_t)-f_*\leq\frac{16DG\log_{2+}(D/d_0)}{\sqrt{n+1}}+\frac{8DG^2\log_{2+}(D/d_0)}{(n+1)\|g_0\|}.$$

Since the leading term in the bound above is of order $\mathcal{O}\left(\frac{1}{\sqrt{n+1}}\right)$, the extra term for not using G is negligible.

B. Gradient Descent Variant

The gradient descent variant (Algorithm 2) results in the following specializations of earlier theorems resulting from plugging in $\gamma_k = 1$:

Theorem B.1. *It holds for the iterates of Algorithm 2,*

$$\sum_{k=0}^{n} \lambda_k \left[f(x_k) - f_*\right] \leq \|s_{n+1}\| \|x_0 - x_*\| - \sum_{k=0}^{n} \lambda_k \langle g_k, s_k \rangle .$$

Lemma B.2. *Gradient descent iterates satisfy*

$$\begin{aligned} -\sum_{k=0}^{n} \lambda_k \langle g_k, s_k \rangle &= \frac{1}{2}\sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2 - \frac{1}{2}\|s_{n+1}\|^2 \\ &\leq \frac{1}{2}\sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2 . \end{aligned}$$

Lemma B.3. *Algorithm 2 satisfies*

$$\|s_{n+1}\| \leq 2d_{n+1} + \frac{\sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2}{2d_{n+1}}.$$

The logarithmic terms in the convergence rate of gradient descent arise from the use of the following standard lemma:

Lemma B.4. *(Lemma 4.13 from Orabona (2019)) Let a_t be a sequence with $a_0 \geq 0$ and ϕ be non-increasing for non-negative values, then:*

$$\sum_{k=1}^{n} a_k \phi\left(\sum_{i=0}^{k} a_i\right) \leq \int_{a_0}^{\sum_{k=0}^{n} a_k} \phi(x) dx.$$

Corollary B.5. *For any vectors $g_0, \ldots, g_n$ such that $\|g_k\| \leq G$ for all k, it holds*

$$\sum_{k=0}^{n} \frac{\|g_k\|^2}{G^2 + \sum_{i=0}^{k} \|g_i\|^2} \leq \log(n+2).$$

Proof. Applying Lemma B.4 with $a_0 = G^2$ and $a_k = \|g_{k-1}\|^2$ up to $a_{n+1} = \|g_n\|^2$ to the function $\phi(x) = 1/x$ gives:

$$\begin{aligned} \sum_{k=1}^{n+1} a_k \phi\left(\sum_{i=0}^{k} a_i\right) &\leq \int_{a_0}^{\sum_{k=0}^{n+1} a_k} \phi(x) dx \\ &= \log\left(\sum_{k=0}^{n+1} a_k\right) - \log(a_0) \\ &= \log\left(\frac{1}{G^2}\sum_{k=0}^{n+1} a_k\right) \\ &= \log\left(\frac{1}{G^2}\left(G^2 + \sum_{k=0}^{n} \|g_k\|^2\right)\right) \\ &\leq \log(n+2). \end{aligned}$$

□

Lemma B.6. *For Algorithm 2, we have*

$$\sum_{k=0}^{n} \lambda_k \left[f(x_k) - f_*\right] \leq 4d_{n+1} D \log(n+2).$$

Proof. Consider the result of Theorem B.1:

$$\sum_{k=0}^{n}\lambda_k\left[f(x_k)-f_*\right]\leq\|s_{n+1}\|\,D-\sum_{k=0}^{n}\lambda_k\left\langle g_k,s_k\right\rangle.$$

We may simplify this by substituting Lemmas B.2 and B.3:

$$\begin{aligned}\sum_{k=0}^{n}\lambda_k\left[f(x_k)-f_*\right]&\leq\left(2d_{n+1}+\frac{\sum_{k=0}^{n}\lambda_k^2\|g_k\|^2}{2d_{n+1}}\right)D+\frac{1}{2}\sum_{k=0}^{n}\lambda_k^2\|g_k\|^2\\&=2d_{n+1}D+\frac{1}{2}\left(\frac{D}{d_{n+1}}+1\right)\sum_{k=0}^{n}\lambda_k^2\|g_k\|^2.\end{aligned}$$

Now apply Corollary B.5:

$$\begin{aligned}\sum_{k=0}^{n}\lambda_k\left[f(x_k)-f_*\right]&\leq 2d_{n+1}D+\frac{1}{2}\left(\frac{D}{d_{n+1}}+1\right)d_{n+1}^2\log\left(n+2\right)\\&=2d_{n+1}\left[D+\frac{1}{2}\left(D\frac{d_{n+1}}{d_{n+1}}+d_{n+1}\right)\log\left(n+2\right)\right]\\&\leq 2d_{n+1}D\left[1+\log\left(n+2\right)\right]\\&\leq 4d_{n+1}D\log\left(n+2\right).\end{aligned}$$

□

B.1. Asymptotic case

Theorem. (Theorem 2.3) It holds for Algorithm 2:

$$f(\hat{x}_n)-f=\mathcal{O}\left(\frac{DG}{\sqrt{n+2}}\log\left(n+2\right)\right).$$

Proof. Following the same logic as for the proof of Theorem 2.1, we may we take n large enough such that

$$\sum_{k=0}^{n}d_k\geq\frac{1}{4}(n+2)d_{n+1}. \tag{6}$$

Then from Jensen's inequality:

$$\frac{1}{\sum_{k=0}^{n}\lambda_k}\sum_{k=0}^{n}\lambda_k\left[f(x_k)-f_*\right]\geq f(\hat{x}_n)-f.$$

Applying Lemma B.6, we get

$$f(\hat{x}_n)-f\leq\frac{4d_{n+1}D\log\left(n+2\right)}{\sum_{k=0}^{n}\lambda_k}.$$

Consider the denominator:

$$\begin{aligned}\sum_{k=0}^{n}\lambda_k=\sum_{k=0}^{n}\frac{d_k}{\sqrt{G^2+\sum_{i=0}^{k}\|g_i\|^2}}&\geq\frac{1}{G}\sum_{k=0}^{n}\frac{d_k}{\sqrt{1+(k+1)}}\\&\geq\frac{1}{G\sqrt{n+2}}\sum_{k=0}^{n}d_k\\&\overset{(6)}{\geq}\frac{\sqrt{n+2}}{4G}d_{n+1}.\end{aligned}$$

So:

$$f(\hat{x}_n)-f\leq\frac{16DG}{\sqrt{n+2}}\log\left(n+2\right).$$

□

B.2. Non-asymptotic case

Theorem B.7. *For Algorithm 2 run for $n \geq 2\log_2(D/d_0)$ iterations, with t chosen as:*

$$t = \arg\min_{k \leq n} \frac{d_{k+1}}{\sum_{i=0}^{k} d_i},$$

we have:

$$f(\hat{x}_t) - f \leq \frac{12DG}{\sqrt{n+1}} \log{(n+2)} \log_{2+}(d_{n+1}/d_0).$$

Proof. Firstly, since f is convex, we can apply Jensen's inequality:

$$f(\hat{x}_t) - f \leq \frac{1}{\sum_{k=0}^{t} \lambda_k} \sum_{k=0}^{t} \lambda_k \left[f(x_k) - f_*\right].$$

Applying Lemma B.6 to the right-hand side, we get

$$f(\hat{x}_t) - f \leq \frac{4d_{n+1}D\log{(n+2)}}{\sum_{k=0}^{t} \lambda_k}.$$

Plugging-in the definition of λ_k, we obtain

$$\begin{aligned}
\sum_{k=0}^{t} \lambda_k = \sum_{k=0}^{t} \frac{d_k}{\sqrt{G^2 + \sum_{i=0}^{k} \|g_i\|^2}} &\geq \frac{1}{G} \sum_{k=0}^{t} \frac{d_k}{\sqrt{1 + (k+1)}} \\
&\geq \frac{1}{G\sqrt{t+2}} \sum_{k=0}^{t} d_k \\
&\geq \frac{(n+1)\, d_{n+1}}{2G\sqrt{t+2}\log_{2+}(d_{n+1}/d_0)}.
\end{aligned}$$

So:

$$\begin{aligned}
f(\hat{x}_t) - f &\leq \frac{8DG\sqrt{t+2}}{n+1} \log{(n+2)} \log_{2+}(d_{n+1}/d_0) \\
&\leq \frac{12DG}{\sqrt{n+1}} \log{(n+2)} \log_{2+}(d_{n+1}/d_0).
\end{aligned}$$

□

C. Coordinate-wise setting

In the coordinate-wise setting we define the matrices A_{n+1} as diagonal matrices with diagonal elements a_i at step n defined as

$$a_{(n+1)i} = \sqrt{G_\infty^2 + \sum_{k=0}^{n} g_{ki}^2}.$$

Let p be the number of dimensions. Define:

$$D_\infty = \|x_0 - x_*\|_\infty$$

and:

$$\hat{d}_{n+1} = \frac{\|s_{n+1}\|_{A_{n+1}^{-1}}^2 - \sum_{k=0}^{n} \lambda_k^2 \|g_k\|_{A_k^{-1}}^2}{2\|s_{n+1}\|_1}.$$

The following lemma applies to Algorithm 3 with general weights λ_k.

Lemma C.1. *The inner product $\lambda_k \left\langle g_k, A_k^{-1} s_k \right\rangle$ is a key quantity that occurs in our theory. Suppose that $A_{n+1} \succeq A_n$ for all n, then we can bound the sum of these inner products as follows:*

$$-\sum_{k=0}^{n} \lambda_k \left\langle g_k, A_k^{-1} s_k \right\rangle \leq -\frac{1}{2} \left\| s_{n+1} \right\|_{A_{n+1}^{-1}}^2 + \frac{1}{2} \sum_{k=0}^{n} \lambda_k^2 \left\| g_k \right\|_{A_k^{-1}}^2 .$$

Proof. We start by expanding $\frac{1}{2} \left\| s_{n+1} \right\|_{A_{n+1}^{-1}}^2$

$$\begin{aligned} \frac{1}{2} \left\| s_{n+1} \right\|_{A_{n+1}^{-1}}^2 &\leq \frac{1}{2} \left\| s_{n+1} \right\|_{A_n^{-1}}^2 \\ &= \frac{1}{2} \left\| s_n \right\|_{A_n^{-1}}^2 + \lambda_n \left\langle g_n, A_n^{-1} s_n \right\rangle + \frac{1}{2} \lambda_n^2 \left\| g_n \right\|_{A_n^{-1}}^2 . \end{aligned}$$

Therefore

$$-\lambda_n \left\langle g_n, A_n^{-1} s_n \right\rangle \leq \frac{1}{2} \left\| s_n \right\|_{A_n^{-1}}^2 - \frac{1}{2} \left\| s_{n+1} \right\|_{A_{n+1}^{-1}}^2 + \frac{1}{2} \lambda_n^2 \left\| g_n \right\|_{A_n^{-1}}^2 .$$

Telescoping over time gives:

$$-\sum_{k=0}^{n} \lambda_k \left\langle g_k, A_k^{-1} s_k \right\rangle \leq -\frac{1}{2} \left\| s_{n+1} \right\|_{A_{n+1}^{-1}}^2 + \frac{1}{2} \sum_{k=0}^{n} \lambda_k^2 \left\| g_k \right\|_{A_k^{-1}}^2 .$$

□

Below, we provide the analogue of Proposition A.5 for the coordinate-wise setting.

Proposition C.2. *(From Duchi et al. (2011)) The gradient error term can be bounded as:*

$$\sum_{j=1}^{p} \sum_{k=0}^{n} \frac{g_{kj}^2}{\sqrt{G^2 + \sum_{i=0}^{k-1} g_{ij}^2}} \leq 2 \sum_{j=1}^{p} \sqrt{G^2 + \sum_{k=0}^{n-1} g_{kj}^2}, \tag{7}$$

as long as $G \geq g_{ij}$ for all i, j.

Lemma C.3. *It holds for the iterates of Algorithm 3*

$$\sum_{k=0}^{n} \lambda_k \left(f(x_k) - f_* \right) \leq \left\| s_{n+1} \right\|_1 D_\infty - \frac{1}{2} \left\| s_{n+1} \right\|_{A_{n+1}^{-1}}^2 + \frac{1}{2} \sum_{k=0}^{n} \lambda_k^2 \left\| g_k \right\|_{A_k^{-1}}^2 .$$

Proof. We start by applying convexity:

$$\begin{aligned} \sum_{k=0}^{n} \lambda_k \left(f(x_k) - f_* \right) &\leq \sum_{k=1}^{n} \lambda_k \left\langle g_k, x_k - x_* \right\rangle \\ &= \sum_{k=1}^{n} \lambda_k \left\langle g_k, x_k - x_0 + x_0 - x_* \right\rangle \\ &= \left\langle s_{n+1}, x_0 - x_* \right\rangle + \sum_{k=1}^{n} \lambda_k \left\langle g_k, x_k - x_0 \right\rangle \\ &= \left\langle s_{n+1}, x_0 - x_* \right\rangle - \sum_{k=1}^{n} \lambda_k \left\langle g_k, A_k^{-1} s_k \right\rangle \\ &\leq \left\| s_{n+1} \right\|_1 \left\| x_0 - x_* \right\|_\infty - \sum_{k=1}^{n} \lambda_k \left\langle g_k, A_k^{-1} s_k \right\rangle . \end{aligned}$$

Applying Lemma C.1 we have:

$$\sum_{k=0}^{n} \lambda_k \left(f(x_k) - f_* \right) \leq \left\| s_{n+1} \right\|_1 \left\| x_0 - x_* \right\|_\infty - \frac{1}{2} \left\| s_{n+1} \right\|_{A_{n+1}^{-1}}^2 + \frac{1}{2} \sum_{k=0}^{n} \lambda_k^2 \left\| g_k \right\|_{A_k^{-1}}^2 .$$

□

Theorem C.4. *Consider the iterates of Algorithm 3. The initial ℓ_∞-distance $D_\infty = \|x_0 - x_*\|_\infty$ satisfies*

$$D_\infty \ge \hat{d}_{n+1} = \frac{\|s_{n+1}\|^2_{A^{-1}_{n+1}} - \sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2_{A_k^{-1}}}{2\|s_{n+1}\|_1}.$$

Proof. Applying $f(x_k) - f_* \ge 0$ to the bound from Lemma C.3 gives:

$$0 \le \|s_{n+1}\|_1 D_\infty - \frac{1}{2}\|s_{n+1}\|^2_{A^{-1}_{n+1}} + \frac{1}{2}\sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2_{A_k^{-1}}.$$

Rearranging this inequality, we obtain

$$\|s_{n+1}\|_1 D_\infty \ge \frac{1}{2}\|s_{n+1}\|^2_{A^{-1}_{n+1}} - \frac{1}{2}\sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2_{A_k^{-1}}.$$

and, therefore,

$$D_\infty \ge \frac{\|s_{n+1}\|^2_{A^{-1}_{n+1}} - \sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2_{A_k^{-1}}}{2\|s_{n+1}\|_1}.$$

□

Lemma C.5. *The ℓ_1-norm of s_{n+1} is bounded by:*

$$\|s_{n+1}\|_1 \le 3d_{n+1}\|a_{n+1}\|_1.$$

Proof. By the definition of $\hat{d}_{n+1}$ we have:

$$\frac{1}{2}\|s_{n+1}\|^2_{A^{-1}_{n+1}} = \hat{d}_{n+1}\|s_{n+1}\|_1 + \frac{1}{2}\sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2_{A_k^{-1}}.$$

and since $\hat{d}_{n+1} \le d_{n+1}$,

$$\frac{1}{2}\|s_{n+1}\|^2_{A^{-1}_{n+1}} \le d_{n+1}\|s_{n+1}\|_1 + \frac{1}{2}\sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2_{A_k^{-1}}.$$

Furthermore, using $\lambda_k = d_k \le d_{n+1}$ and Proposition C.2, we obtain

$$\begin{aligned}
\frac{1}{2}\sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2_{A_k^{-1}} &\le \frac{1}{2}d_{n+1}^2 \sum_{k=0}^{n} \|g_k\|^2_{A_k^{-1}} \\
&\overset{(7)}{\le} d_{n+1}^2 \sum_{i=1}^{p} \sqrt{G_\infty^2 + \sum_{k=0}^{n-1} g_{ki}^2} \\
&= d_{n+1}^2 \|a_{n+1}\|_1.
\end{aligned}$$

Therefore, using inequality $2\alpha\beta \le \alpha^2 + \beta^2$ with $\alpha^2 = 2d_{n+1}^2 a_{(n+1)i}$ and $\beta^2 = \frac{s^2_{(n+1)i}}{2a_{(n+1)i}}$, we get

$$\begin{aligned}
2d_{n+1}\|s_{n+1}\|_1 = \sum_{i=1}^{p} 2d_{n+1}|s_{(n+1)i}| &\le \sum_{i=1}^{p}\left(2d_{n+1}^2 a_{(n+1)i} + \frac{s^2_{(n+1)i}}{2a_{(n+1)i}}\right) \\
&= 2d_{n+1}^2\|a_{n+1}\|_1 + \frac{1}{2}\|s_{n+1}\|^2_{A^{-1}_{n+1}} \\
&\le 2d_{n+1}^2\|a_{n+1}\|_1 + d_{n+1}\|s_{n+1}\|_1 + \frac{1}{2}\sum_{k=0}^{n} \lambda_k^2 \|g_k\|^2_{A_k^{-1}} \\
&\le 2d_{n+1}^2\|a_{n+1}\|_1 + d_{n+1}\|s_{n+1}\|_1 + d_{n+1}^2\|a_{n+1}\|_1.
\end{aligned}$$

Rearranging, we get

$$d_{n+1}\|s_{n+1}\|_1 \leq 3d_{n+1}^2\|a_{n+1}\|_1.$$

□

Theorem. (Theorem 3.1) For a convex function with $G_\infty = \max_x \|\nabla f(x)\|_\infty$, D-Adapted AdaGrad returns a point $\hat{x}_n$ such that

$$f(\hat{x}_n) - f_* = \mathcal{O}\left(\frac{\|a_{n+1}\|_1 D_\infty}{n+1}\right) = \mathcal{O}\left(\frac{pG_\infty D_\infty}{\sqrt{n+1}}\right)$$

as $n \to \infty$, where $D = \|x_0 - x_*\|_\infty$ for any x_* in the set of minimizers of f, as long as $d_0 \leq D_\infty$.

Proof. As in the proof of Theorem 2.1, we will show the result holds for some sufficiently n. Since d_k is a non-decreasing sequence upper bounded by D, there must exist some $\hat{n}$ such that after $\hat{n}$ steps, $d_k \geq \frac{1}{2}d_{n+1}$ for all $k, n \geq \hat{n}$. We take $n \geq 2\hat{n}$.

Then:

$$\sum_{k=0}^{n} d_k \geq \sum_{k=\hat{n}}^{n} d_k \geq \sum_{k=\hat{n}}^{n} \frac{1}{2}d_{n+1} = \frac{1}{2}(n - \hat{n} + 1)d_{n+1} \geq \frac{1}{4}(n+1)d_{n+1},$$

and, therefore,

$$\frac{1}{\sum_{k=0}^{n} d_k} \leq \frac{4}{(n+1)d_{n+1}}.$$

Combining this with Lemma C.3 yields

$$\frac{1}{\sum_{k=0}^{n} d_k} \sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) \leq \frac{4}{(n+1)d_{n+1}} \left(\|s_{n+1}\|_1 D_\infty + \frac{1}{2}\sum_{k=0}^{n} d_k^2 \|g_k\|_{A_k^{-1}}^2 \right).$$

From Proposition C.2 we have:

$$\begin{aligned} \frac{1}{2}\sum_{k=0}^{n} d_k^2 \|g_k\|_{A_k^{-1}}^2 &\leq \frac{1}{2}d_{n+1}^2 \sum_{k=0}^{n} \|g_k\|_{A_k^{-1}}^2 \\ &\leq d_{n+1}^2 \|a_{n+1}\|_1 . \end{aligned}$$

Plugging this in together with Lemma C.5 gives:

$$\begin{aligned} \frac{1}{\sum_{k=0}^{n} d_k} \sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) &\leq \frac{4}{(n+1)d_{n+1}} \left(3d_{n+1} \|a_{n+1}\|_1 D_\infty + d_{n+1}^2 \|a_{n+1}\|_1\right) \\ &= \frac{4}{n+1} \left(3 \|a_{n+1}\|_1 D_\infty + d_{n+1} \|a_{n+1}\|_1\right). \end{aligned}$$

So using $d_{n+1} \leq D_\infty$ we have:

$$\frac{1}{\sum_{k=0}^{n} d_k} \sum_{k=0}^{n} d_k \left(f(x_k) - f_*\right) \leq \frac{16}{n+1} \|a_{n+1}\|_1 D_\infty.$$

Using Jensen's inequality on the left:

$$f(\hat{x}_n) - f_* \leq \frac{16}{n+1} \|a_{n+1}\|_1 D_\infty.$$

We can further simplify using $\|a_{n+1}\|_1 = \sum_{j=1}^{p} \sqrt{G_\infty^2 + \sum_{k=0}^{n} g_{kj}^2} \leq p\sqrt{n+1}G_\infty$:

$$f(\hat{x}_n) - f_* \leq \frac{16pG_\infty D_\infty}{\sqrt{n+1}},$$

which yields the result.

□

D. Parameter settings

In this section, we list the parameters, architectures and hardware that we used for the experiments. The information is collected in Tables 2–12.

Table 2. Logistic regression experiment. The problems are part of the LIBSVM repository. Since there are no standard train/test splits, and due to the small sizes of the datasets, we present training accuracy curves only.

Hyper-parameter	Value
Epochs	100
GPUs	1×V100
Batch size	16
Epochs	100
LR schedule	60,80,95 tenthing
Seeds	10
Decay	0.0
Momentum	0.0
Baseline LR	grid search

Table 3. CIFAR10 experiment. Our data augmentation pipeline followed standard practice: random horizontal flipping, then random cropping to 32×32 (padding 4), then normalization by centering around (0.5, 0.5, 0.5).

Hyper-parameter	Value
Architecture	Wide ResNet 16-8
Epochs	300
GPUs	1×V100
Batch size per GPU	128
LR schedule	150-225 tenthing
Seeds	10
decay	0.0001
Momentum	0.9
SGD LR	0.1

Table 4. CIFAR100 experiment. Following standard practice, we normalized the channels by subtracting ((0.5074,0.4867,0.4411) and dividing by (0.2011,0.1987,0.2025)). Augmentations used at training time were: random horizontal flips, random crop (32, padding=4, reflect).

Hyper-parameter	Value
Architecture	DenseNet [6,12,24,16], growth rate 12
Epochs	300
GPUs	1×V100
Batch size per GPU	64
LR schedule	150-225 tenthing
Seeds	10
Decay	0.0002
Momentum	0.9
SGD LR	0.05

Table 5. ImageNet experiment. Normalization of the color channels involved subtracting (0.485, 0.456, 0.406), and dividing by (0.229, 0.224, 0.225). For data augmentation at training we used PyTorch's RandomResizedCrop to 224, then random horizontal flips. At test time images were resized to 256 then center cropped to 224.

Hyper-parameter	Value
Architecture	ResNet50
Epochs	100
GPUs	8×V100
Batch size per GPU	32
LR schedule	30-60-90 tenthing
Seeds	5
Decay	0.0001
Momentum	0.9
SGD LR	0.1

Table 6. fastMRI experiment. We used the implementation from `https://github.com/facebookresearch/fastMRI`.

Hyper-parameter	Value
Architecture	12 layer VarNet 2.0
Epochs	50
GPUs	8×V100
Batch size per GPU	1
Acceleration factor	4
Low frequency lines	16
Mask type	Offset-1
LR schedule	flat
Seeds	5
Decay	0.0
Adam LR	0.0003
β_1, β_2	0.9, 0.999

Table 7. IWSLT14 experiment. Our implementation used FairSeq `https://github.com/facebookresearch/fairseq` defaults except for the parameters listed below. Note that the default Adam optimizer uses decoupled weight decay.

Hyper-parameter	Value
Architecture	lstm_wiseman_iwslt_de_en
Max Epoch	55
GPUs	1×V100
Max tokens per batch	4096
Warmup steps	4000
Dropout	0.3
Label smoothing	0.1
Share decoder, input, output embed	True
Float16	True
Update Frequency	1
LR schedule	Inverse square-root
Seeds	10
Decay	0.05
Adam LR	0.01
β_1, β_2	0.9, 0.98

Table 8. RoBERTa BookWiki experiment. Our implementation used FairSeq defaults except for the parameters listed below.

Hyper-parameter	Value
Architecture	roberta_base
Task	masked_lm
Max updates	23,000
GPUs	8×V100
Max tokens per sample	512
Dropout	0.1
Attention Dropout	0.1
Max sentences	16
Warmup	10,000
Sample Break Mode	Complete
Float16	True
Update Frequency	16
LR schedule	Polynomial decay
Seeds	5
Decay	0.0
Adam LR	0.001
β_1, β_2	0.9, 0.98

Table 9. GPT BookWiki experiment. Our implementation used FairSeq defaults except for the parameters listed below.

Hyper-parameter	Value
Architecture	transformer_lm_gpt
Task	language_modeling
Max updates	65,000
GPUs	8×V100
Max tokens per sample	512
Dropout	0.1
Attention Dropout	0.1
Max sentences	1
Warmup	10,000
Sample Break Mode	Complete
Share decoder, input, output embed	True
Float16	True
Update Frequency	16
LR schedule	Polynomial decay
Seeds	5
Decay	0.005
Adam LR	0.001
β_1, β_2	0.9, 0.98

Table 10. COCO Object Detection experiment. We used the Detectron2 codebase https://github.com/facebookresearch/detectron2, with the `faster_rcnn_X_101_32x8d_FPN_3x` configuration. We list its key parameters below.

Hyper-parameter	Value
Architecture	X-101-32x8d
Solver Steps (Schedule)	210000, 250000
Max Iter	270000
IMS Per Batch	16
Momentum	0.9
Decay	0.0001
SGD LR	0.02

Table 11. Vision Transformer experiment. We used the Pytorch Image Models codebase https://github.com/rwightman/pytorch-image-models.

Hyper-parameter	Value
Model	vit_tiny_patch16_224
Epochs	300
Batch Size	512
Sched	Cosine
Warmup Epochs	5
Hflip	0.5
aa	rand-m6-mstd0.5
mixup	0.1
cutmix	1.0
Crop Pct	0.9
BCE Loss	True
Seeds	5
Decay	0.1
Adam LR	0.001
β_1, β_2	0.9, 0.999

Table 12. Criteo Kaggle experiment. We used our own implementation of DLRM, based on the codebase provided at https://github.com/facebookresearch/dlrm.

Hyper-parameter	Value
Iterations	300 000
Batch Size	128
Schedule	Flat
Emb Dimension	16
Seeds	5
Decay	0.0
Adam LR	0.0001
β_1, β_2	0.9, 0.999

E. Additional figures

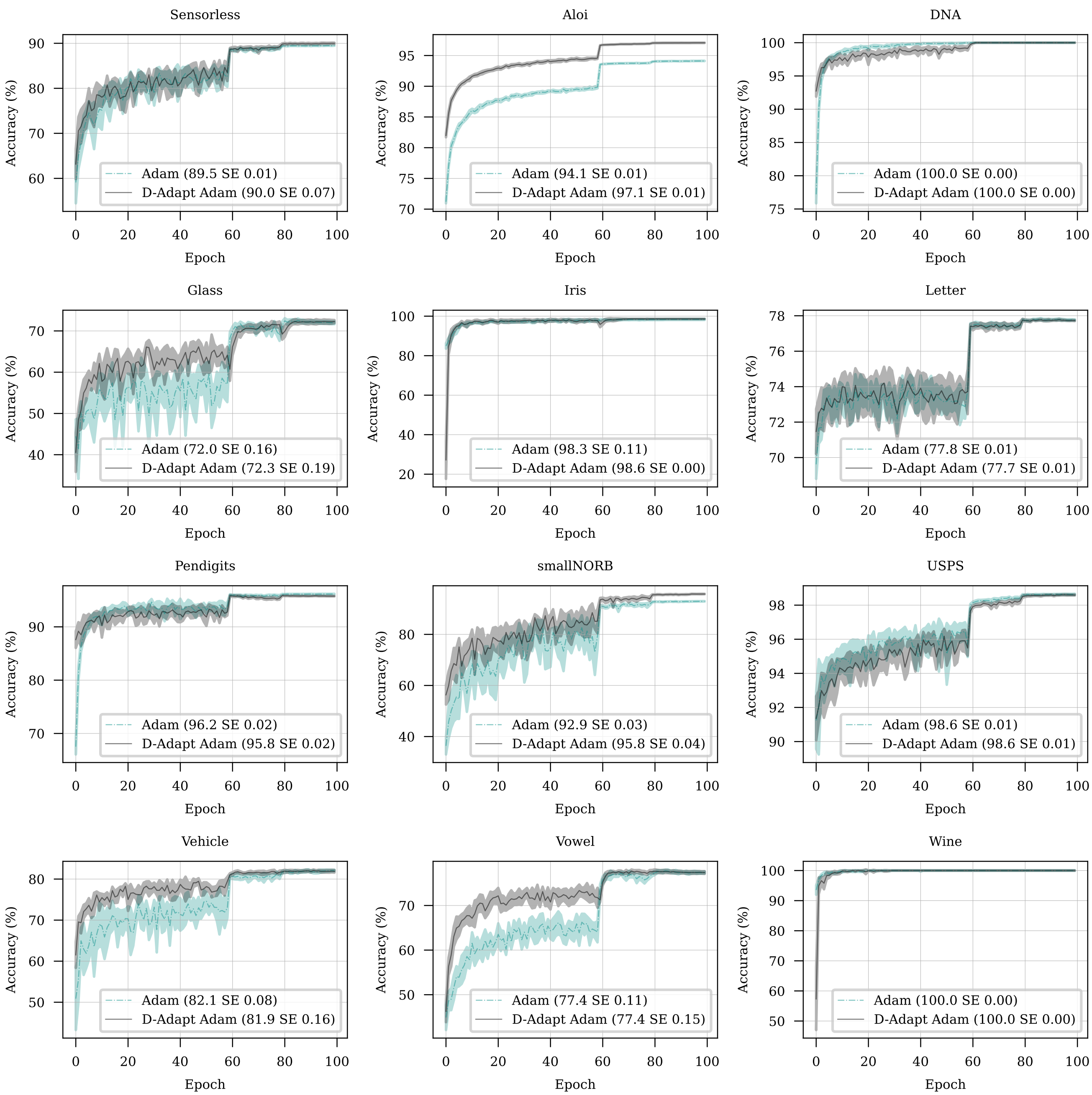

Figure 4. Logistic Regression experiments.

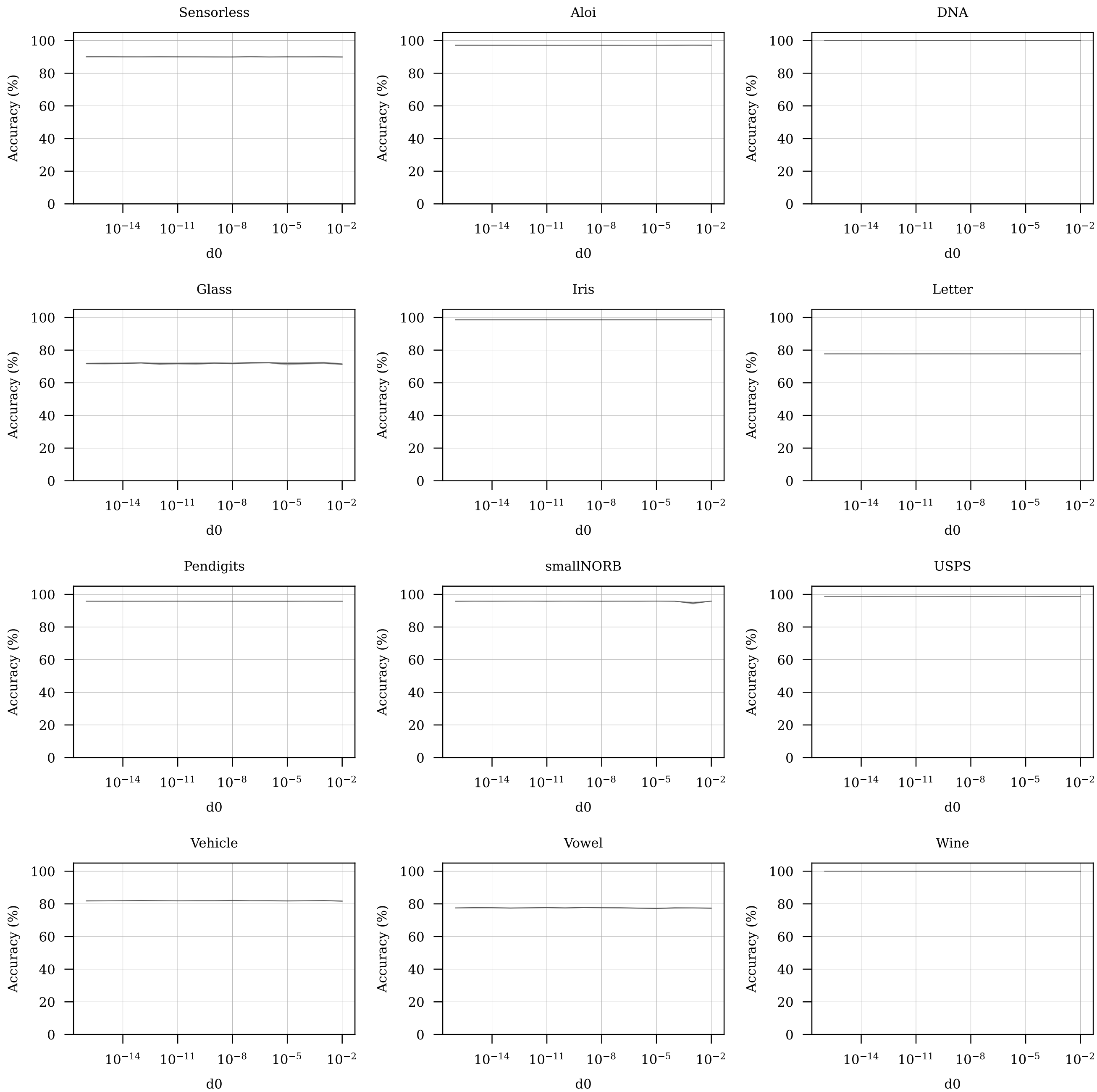

Figure 5. Final accuracy as a function of d_0. Setup described in Section 7. Error bars show a range of 2 standard errors above and below the mean of the 10 seeds. For most problems error bars are too narrow to be visible.

F. Additional notes

Theorem F.1. *If* $\|x_n - x_*\| \to 0$*, and the learning rate (1) is used, then:*

$$\lim_{n \to \infty} d_n \geq \frac{D}{1 + \sqrt{3}}.$$

Proof. By triangle inequality, we can bound the distance to x_* as

$$D = \|x_0 - x_*\| \leq \|x_n - x_*\| + \|x_n - x_0\| = \|x_n - x_*\| + \gamma_n \|s_n\|.$$

We need to upper bound the last term $\gamma_n\|s_n\|$. To this end, we use the same argument as in the proof of Lemma A.4, starting with the definition of $\hat{d}_{n+1}$ and plugging-in $\lambda_k = d_k$:

$$\frac{\gamma_{n+1}}{2}\left\|s_{n+1}\right\|^2 - \sum_{k=0}^{n}\frac{\gamma_k}{2}d_k^2\left\|g_k\right\|^2 = \hat{d}_{n+1}\left\|s_{n+1}\right\| \le d_{n+1}\left\|s_{n+1}\right\|.$$

The main change from the proof of Lemma A.4 is that now we will use inequality $2\alpha\beta \le \alpha^2 + \beta^2$ with $\alpha^2 = \theta\frac{d_{n+1}^2}{\gamma_{n+1}}$ and $\beta^2 = \frac{\gamma_{n+1}}{\theta}\|s_{n+1}\|^2$ with θ to be chosen later to make the bound optimal. Plugging this inequality into the previous bound, we derive

$$2\alpha\beta = 2d_{n+1}\|s_{n+1}\| \le \frac{\theta d_{n+1}^2}{\gamma_{n+1}} + \frac{\gamma_{n+1}}{\theta}\|s_{n+1}\|^2 \le \frac{\theta d_{n+1}^2}{\gamma_{n+1}} + \frac{2}{\theta}d_{n+1}\|s_{n+1}\| + \frac{1}{\theta}\sum_{k=0}^{n}\gamma_k d_k^2\|g_k\|^2.$$

Since the sequence d_k is non-decreasing, we have $d_k \le d_{n+1}$, further giving us

$$\frac{1}{\theta}\sum_{k=0}^{n}\gamma_k d_k^2\|g_k\|^2 \le \frac{d_{n+1}^2}{\theta}\sum_{k=0}^{n}\gamma_k\|g_k\|^2 \overset{(4)}{\le} \frac{2}{\theta}\gamma_{n+1}d_{n+1}^2\left(G^2 + \sum_{k=0}^{n-1}\|g_k\|^2\right) = \frac{2d_{n+1}^2}{\theta\gamma_{n+1}}.$$

Plugging this back and rearranging, we get

$$2\left(1 - \frac{1}{\theta}\right)d_{n+1}\|s_{n+1}\| \le \frac{\theta d_{n+1}^2}{\gamma_{n+1}} + \frac{2d_{n+1}^2}{\theta\gamma_{n+1}} = (\theta + 2/\theta)\frac{d_{n+1}^2}{\gamma_{n+1}}.$$

Now it is time for us to choose θ. Clearly, the optimal value of θ is the one that minimizes the ratio $\frac{\theta+2/\theta}{2(1-1/\theta)} = \frac{\theta^2+2}{2(\theta-1)}$. It can be shown that the value of $\theta_* = 1+\sqrt{3}$ is optimal and gives $\frac{\theta_*^2+2}{2(\theta_*-1)} = 1+\sqrt{3}$. Thus, we have

$$\gamma_{n+1}\|s_{n+1}\| \le (1+\sqrt{3})d_{n+1}.$$

Now, assume that $x_n \to x_*$ in norm, so $\|x_n - x_*\| \to 0$. In that case, the bounds combined yield

$$D \le \lim_n(\|x_n - x_*\| + \gamma_n\|s_n\|) = \lim_{n\to\infty}\gamma_n\|s_n\| \le (1+\sqrt{3})\lim_{n\to\infty}d_n.$$

Thus, the value of d_n is asymptotically lower bounded by $\frac{D}{1+\sqrt{3}}$. □

F.1. A tighter lower bound on D

Using Lemma A.1, we can obtain a slightly tighter bound than in Theorem A.3. In particular, we have previously used the following bound:

$$\begin{aligned}
\sum_{k=0}^{n}\lambda_k\left(f(x_k) - f_*\right) &\le \sum_{k=0}^{n}\lambda_k\left\langle g_k, x_k - x_*\right\rangle \\
&= \sum_{k=0}^{n}\lambda_k\left\langle g_k, x_k - x_0 + x_0 - x_*\right\rangle \\
&= \left\langle s_{n+1}, x_0 - x_*\right\rangle + \sum_{k=0}^{n}\lambda_k\left\langle g_k, x_k - x_0\right\rangle \\
&= \left\langle s_{n+1}, x_0 - x_*\right\rangle - \sum_{k=0}^{n}\lambda_k\gamma_k\left\langle g_k, s_k\right\rangle \\
&\le \left\|s_{n+1}\right\|\left\|x_0 - x_*\right\| - \sum_{k=0}^{n}\lambda_k\gamma_k\left\langle g_k, s_k\right\rangle.
\end{aligned}$$

From here, we can immediately conclude that

$$D = \|x_0 - x_*\| \geq \widetilde{d}_{n+1} = \frac{\sum_{k=0}^{n} \lambda_k \gamma_k \langle g_k, s_k \rangle}{\|s_{n+1}\|}.$$

Notice that it always holds $\widetilde{d}_n \geq \hat{d}_n$. The only complication that we can face is with Lemma A.4, where we used the definition of $\hat{d}_n$ to obtain the upper bound. Nevertheless, one can prove the same bound with $\hat{d}_n$ replaced by $\widetilde{d}_n$ by repeating the same argument:

$$\frac{\gamma_{n+1}}{2} \|s_{n+1}\|^2 - \sum_{k=0}^{n} \frac{\gamma_k}{2} \lambda_k^2 \|g_k\|^2 = \hat{d}_{n+1} \|s_{n+1}\| \leq \widetilde{d}_{n+1} \|s_{n+1}\| \leq d_{n+1} \|s_{n+1}\|.$$

From that place, the rest of the proof of Lemma A.4 follows in exactly the same way. The other proofs only use the monotonicity of the sequence and its boundedness by D, $d_k \leq d_{n+1} \leq D$, which would remain valid if replace $\hat{d}_n$ with $\widetilde{d}_n$.

G. Adam Derivation

Lemma G.1. *Consider a positive constant c. Define the two sequences:*

$$u_{k+1} = u_k + \frac{1}{c^k} g_k,$$

$$\hat{u}_{k+1} = c\hat{u}_k + (1 - c)\, g_k.$$

Then the following relationship holds between the two sequences:

$$\hat{u}_{k+1} = c^k (1 - c)\, u_{k+1},$$

assuming that $\hat{u}_0 = (1 - c)\, u_0$.

In this section, we use hat notation to denote the exponential moving averages of each quantity (other than $\hat{d}$). We drop the hat notation for simplicity when we present the method (Algorithm 5). We also treat each quantity as 1-dimensional, with the understanding that the final result holds also when applied element-wise.

Our goal is to derive the EMA updates, given the following weighted updates:

$$\lambda_k = \sqrt{\beta_2^{-k}},$$

$$s_{k+1} = s_k + \lambda_k g_k,$$

$$v_{k+1} = v_k + \lambda_k^2 g_k^2,$$

$$\gamma_{k+1} = \frac{1}{\sqrt{(1 - \beta_2)\, v_{k+1}}},$$

$$r_{k+1} = r_k + \gamma_{k+1} \lambda_k \langle g_k, s_k \rangle,$$

$$\hat{d}_{n+1} = \frac{\sum_{k=0}^{n} \gamma_k \lambda_k \langle g_k, s_k \rangle}{\|s_{n+1}\|_1} = \frac{r_{k+1}}{\|s_{n+1}\|_1}.$$

Note that we normalized by γ_{k+1} rather than γ_k for this implemented variant. We also introduce the Adam denominator through gamma, in the style of DA method, rather than the step size as implemented in Algorithm 5. This is the only way currently supported by our theory. However, we will still use the non-DA step:

$$x_{k+1} = x_k - \lambda_k g_k.$$

The denominator of γ is chosen to ensure that the step is properly normalized. To see that, note that:

$$\hat{v}_{k+1} = \beta_2^k (1 - \beta_2)\, v_{k+1},$$

and so:

$$\gamma_{k+1} = \frac{1}{\sqrt{(1-\beta_2)\, v_{k+1}}} = \frac{\sqrt{\beta_2^k\,(1-\beta_2)}}{\sqrt{(1-\beta_2)\,\hat{v}_{k+1}}}$$
$$= \frac{\sqrt{\beta_2^k}}{\sqrt{\hat{v}_{k+1}}},$$

therefore:

$$x_{k+1} = x_k - \frac{\sqrt{\beta_2^k}}{\sqrt{\hat{v}_{k+1}}}\frac{1}{\sqrt{\beta_2^k}} g_k = x_k - \frac{1}{\sqrt{\hat{v}_{k+1}}} g_k.$$

Note that:

$$\hat{s}_{k+1} = \beta_2^{k/2}\left(1-\sqrt{\beta_2}\right) s_{k+1},$$

and so:

$$\hat{s}_{k+1} = \sqrt{\beta_2}\hat{s}_k + \left(1-\sqrt{\beta_2}\right) g_k.$$

So we have:

$$\begin{aligned}
r_{k+1} &= r_k + \gamma_{k+1}\lambda_k \left\langle g_k, s_k \right\rangle \\
&= r_k + \frac{1}{\sqrt{\beta_2^k}\left(1-\sqrt{\beta_2}\right)}\gamma_{k+1}\frac{1}{\sqrt{\beta_2^k}}\left\langle g_k, \hat{s}_k \right\rangle \\
&= r_k + \frac{1}{\sqrt{\beta_2^k}\left(1-\sqrt{\beta_2}\right)}\frac{\sqrt{\beta_2^k}}{\sqrt{\hat{v}_{k+1}}}\frac{1}{\sqrt{\beta_2^k}}\left\langle g_k, \hat{s}_k \right\rangle \\
&= r_k + \frac{1}{\left(1-\sqrt{\beta_2}\right)}\frac{1}{\sqrt{\beta_2^k}}\frac{1}{\sqrt{\hat{v}_{k+1}}}\left\langle g_k, \hat{s}_k \right\rangle
\end{aligned}$$

. Now define

$$r'_{k+1} = r'_k + \frac{1}{\sqrt{\beta_2^k}}\frac{1}{\sqrt{\hat{v}_{k+1}}}\left\langle g_k, \hat{s}_k \right\rangle,$$

then $r'_{k+1} = \left(1-\sqrt{\beta_2}\right) r_{k+1}$. Now using

$$\hat{r}_{k+1} = \sqrt{\beta_2}\hat{r}_k + \left(1-\sqrt{\beta_2}\right)\frac{1}{\sqrt{\hat{v}_{k+1}}}\left\langle g_k, \hat{s}_k \right\rangle,$$

we get

$$\hat{r}_{k+1} = \beta_2^{k/2}\left(1-\sqrt{\beta_2}\right) r'_{k+1} = \beta_2^{k/2}\left(1-\sqrt{\beta_2}\right)^2 r_{k+1}.$$

Plugging this in gives:

$$\begin{aligned}
\hat{d}_{n+1} = \frac{r_{k+1}}{\left\|s_n\right\|_1} &= \frac{\hat{r}_{k+1}}{\beta_2^{k/2}\left(1-\sqrt{\beta_2}\right)^2\left\|s_n\right\|_1} \\
&= \frac{\beta_2^{k/2}\left(1-\sqrt{\beta_2}\right)\hat{r}_{k+1}}{\beta_2^{k/2}\left(1-\sqrt{\beta_2}\right)^2\left\|\hat{s}_n\right\|_1} \\
&= \frac{\hat{r}_{k+1}}{\left(1-\sqrt{\beta_2}\right)\left\|\hat{s}_n\right\|_1}.
\end{aligned}$$